Федеральное агентство по образованию

Томский государственный университет систем управления радиоэлектроники (ТУСУР)

Кафедра комплексной информационной безопасности электронно-вычислительных систем (КИБЭВС)

Р.В. Мещеряков, Г.А. Праскурин, А.А. Шелупанов

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ

Лабораторный практикум
для студентов специальности 090105 «Комплексное обеспечение информационной безопасности автоматизированных систем»

2006

УДК 681.3

В данном пособии приводятся лабораторные работы по дисциплине «Теоретические основы компьютерной безопасности» для студентов высших учебных заведений, обучающихся по специальности 090105 – Комплексное обеспечение информационной безопасности автоматизированных систем.

Содержание

Лабораторная работа 1
Парольные системы защиты

Цель работы

Целью лабораторной работы является изучение структуры, характеристик, сильных и слабых сторон парольных систем защиты операционных систем, офисных пакетов, архиваторов, закрепление на практике навыков по определению стойкости парольных систем, а так же получение практических навыков по работе с парольными системами и реализация некоторых видов атак на них.

Характеристики парольных системы защиты

Парольная система защиты является «первым рубежом» на пути злоумышленника к защищаемой информации. Поэтому во многом от стойкости парольной системы защиты зависит успешность реализации злоумышленником своих замыслов. Существует множество реализаций парольных систем, в структуре которых можно выделить несколько наиболее важных компонентов:

- интерфейс пользователя;
- интерфейс администратора;
- база учетных записей пользователей;
- модуль сопряжения с другими подсистемами безопасности.

Несмотря на то, что существуют характерные угрозы, направленные на каждый из элементов парольной системы защиты, основные злоумышленные действия направлены на базу учетных записей, т.к. завладев и раскрыв эту базу, злоумышленник может зарегистрироваться в системе от имени любого санкционированного пользователя и скрытно выполнить любые действия. Поэтому информация, хранимая в базе учетных записей должна быть надежно защищена.

В большинстве систем пользователи имеют возможность самостоятельно выбирать пароли или получают их от системных администраторов. При этом для уменьшения деструктивного влияния описанного выше человеческого

фактора необходимо реализовать ряд требований к выбору и использованию паролей:

1. Установление минимальной длины пароля.
2. Использование в пароле различных групп символов.
3. Проверка и отбраковка пароля по словарю.
4. Установление максимального срока действия пароля.
5. Ведение журнала истории паролей. Применение эвристического алгоритма, бракующего пароли на основании данных журнала истории.
6. Ограничение числа попыток ввода пароля.
7. Поддержка режима принудительной смены пароля пользователя.
8. Использование задержки при вводе неправильного пароля.
9. Запрет на выбор пароля самими пользователями и автоматическая генерация паролей.
10. Принудительная смена пароля при первой регистрации пользователя в системе.

Злоумышленник может реализовывать угрозы парольной системы защиты в двух режимах – интерактивном, т.е. с применением штатных средств парольной системы (интерфейса пользователя), и неинтерактивном (например, он может завладеть базой учетных записей и применить программу для определения паролей в этой базе). Наиболее опасным является неинтерактивный вариант, т.к. злоумышленник может скрытно и с большой скоростью «подбирать» пароли.

Для численной оценки параметров парольной системы защиты используются следующие показатели:

A — мощность алфавита паролей, т. е. то множество знаков, которое может применяться при вводе пароля.

L — длина пароля (в знаках). Может изменяться для обеспечения заданной стойкости парольной системы.

S — мощность пространства паролей, т. е. множество всех возможных паролей в системе.

Мощность пространства паролей связана с мощностью алфавита паролей и длиной паролей следующим выражением:

$$S=A^L.$$

V — скорость подбора пароля (соответственно различают скорость подбора пароля для интерактивного (1-5 паролей /секунду) и неинтерактивного (10 и > паролей / секунду) подбора паролей).

T — срок действия (жизни) пароля (обычно задается в днях).

P — вероятность подбора пароля в течение срока его действия.

Вероятность подбора пароля можно определить следующим образом:

$$P=V*T/S.$$

В конкретной ситуации задают некоторые желательные значения для одних параметров (например, очень маленькое значение вероятности подбора пароля) и высчитывают остальные параметры.

Очевидно, что с увеличением длины пароля и/или мощности алфавита паролей вероятность подбора пароля уменьшается. А при увеличении срока жизни пароля, вероятность его подбора увеличивается.

Примеры

Пример 1.

Задание: определить время перебора всех паролей, состоящих из 6 цифр.

Решение: алфавит составляют цифры (A=10). Длина пароля 6 символов (L=6). Таким образом, получаем количество вариантов: $S=A^L=10^6$ (паролей).

Примем скорость перебора паролей V=10 паролей/секунду. Получаем время перебора всех паролей

$T=S/V=10^5$ секунд ≈ 1667минут ≈ 28часов ≈ 1,2 дня.

Примем, что после каждого из m=3 неправильно введенных паролей идет пауза в v=5 секунд. Получаем продолжительность всех пауз при переборе всех паролей

$T_{\text{пауза.}}= T*v/m = T*5/3 =$ 16667секунд ≈ 2778минут ≈ 46часов ≈ 1,9 дня.

$T_{\text{итог}} = T+T_{\text{пауза}} = 1,2 + 1,9 = 3,1$ дня

Пример 2.

Определить минимальную длину пароля, алфавит которого состоит из 10 символов, время перебора которого было не меньше 10 лет.

Алфавит составляют символы $A = 10$.

Длина пароля рассчитывается: $L = \log_A S = \lg S$.

Определим количество вариантов $S = T*V =$ 10лет*10паролей/сек. = 10*10*365*24*60*60 ≈ 3,15* 10^9 вариантов

Таким образом, получаем длину пароля: L=lg (3,15*10^9) = 9,5

Очевидно, что длина пароля должна быть не менее 10 символов.

Задание

1. Определить время перебора всех словарных паролей (объем словаря равен 1 000 000 слов), если скорость перебора составляет 100 паролей/сек.

2. Определить время перебора всех паролей с параметрами.

Алфавит состоит из A символов.

Длина пароля символов L.

Скорость перебора V паролей в секунду.

После каждого из m неправильно введенных паролей идет пауза в v секунд

Вариант	*A*	*L*	*V*	*m*	*v*
1	33	10	100	0	0
2	26	12	13	3	2
3	52	6	30	5	10
4	66	7	20	10	3
5	59	5	200	0	0
6	118	9	50	7	12
7	128	10	500	0	0
8	150	3	200	5	3
9	250	8	600	7	3
10	500	5	1000	10	10

3. Определить минимальную длину пароля, алфавит которого состоит из *A* символов, время перебора которого было не меньше *T* лет.

Скорость перебора *V* паролей в секунду.

Вариант	*A*	*T*	*V*
1	33	100	100
2	26	120	13
3	52	60	30
4	66	70	20
5	59	50	200
6	118	90	50
7	128	100	500
8	150	30	200
9	250	80	600
10	500	50	1000

4. Определить количество символов алфавита, пароль состоит из *L* символов, время перебора которого было не меньше *T* лет.

Скорость перебора *V* паролей в секунду.

Вариант	*L*	*T*	*V*

1	5	100	100
2	6	120	13
3	10	60	30
4	7	70	20
5	9	50	200
6	11	90	50
7	12	100	500
8	6	30	200
9	8	80	600
10	50	50	1000

Задание на самостоятельную работу

1. Подготовить доклады на тему:
    1) парольная система защиты ОС Windows;
    2) парольная система защиты ОС семейства Unix;
    3) парольные системы защиты различных служб Интернета (Web-сервера, электронная почта, FTP и т.д.).

    В докладах предлагается отразить следующие вопросы:
    1) Организация (структура) парольной системы
    2) Место, способ хранения паролей и штатные средства защиты базы учетных записей
    3) Средства, предоставляемые администраторам для управления парольной системой
    4) Известные уязвимости парольной системы и методы преодоления парольной системы злоумышленником.
2. Исследовать парольные системы защиты архиваторов и офисных программ (Word, Excel, PowerPoint и т.д.). Определить стойкость парольных систем защиты архиваторов и офисных программ с помощью приложений для взлома пароля (Advanced Office Password Recovery, Archive Password Recovery и подобных).
3. (Дополнительное) Написать программу, которая должна эмулировать работу парольной системы защиты.

Программа должна реализовывать 5 из 10 требований, предъявляемых к парольным системам защиты.

Лабораторная работа 2
Модель Кларка-Вилсона

Цель работы

Целью работы является закрепление полученного теоретического материала по моделям целостности и применение на практике положений модели целостности Кларка-Вилсона к вычислительным системам.

Основные положения модели целостности Кларка-Вилсона

Модель Кларка-Вилсона появилась в результате проведенного авторами анализа реально применяемых методов обеспечения целостности документооборота в коммерческих компаниях. В отличие от моделей Биба и Белла-ЛаПадулы, она изначально ориентирована на нужды коммерческих заказчиков, и, по мнению авторов, более адекватна их требованиям, чем предложенная ранее коммерческая интерпретация модели целостности на основе решеток. Основные понятия рассматриваемой модели — это корректность транзакций и разграничение функциональных обязанностей. Модель задает правила функционирования компьютерной системы и определяет две категории объектов данных и два класса операций над ними.

Все содержащиеся в системе данные подразделяются на контролируемые и неконтролируемые элементы данных (constrained data items — CDI и unconstrained data items — UDI соответственно). Целостность первых обеспечивается моделью Кларка-Вилсона. Последние содержат информацию, целостность которой в рамках данной модели не контролируется (этим и объясняется выбор терминологии).

Далее, модель вводит два класса операций над элементами данных: процедуры контроля целостности (integrity verification procedures — IVP) и процедуры преобразования (transformation procedures — TP). Первые из них обеспечивают проверку целостности контролируемых элементов данных (CDI), вторые изменяют состав множества всех CDI (например, преобразуя элементы UDI в CDI).

Так же модель содержит девять правил, определяющих взаимоотношения элементов данных и процедур в процессе функционирования системы.

Правило C1. Множество всех процедур контроля целостности (IVP) должно содержать процедуры контроля целостности любого элемента данных из множества всех CDI.

Правило C2. Все процедуры преобразования (TP) должны быть реализованы корректно в том смысле, что не должны нарушать целостность обрабатываемых ими CDI. Кроме того, с каждой процедурой преобразования должен быть связан список элементов CDI, которые допустимо обрабатывать данной процедурой. Такая связь устанавливается администратором безопасности.

Правило E1. Система должна контролировать допустимость применения TP к элементам CDI в соответствии со списками, указанными в правиле C2.

Правило E2. Система должна поддерживать список разрешенных конкретным пользователям процедур преобразования с указанием допустимого для каждой TP и данного пользователя набора обрабатываемых элементов CDI.

Правило C3. Список, определенный правилом C2, должен отвечать требованию разграничения функциональных обязанностей.

Правило E3. Система должна аутентифицировать всех пользователей, пытающихся выполнить какую-либо процедуру преобразования.

Правило C4. Каждая TP должна записывать в журнал регистрации информацию, достаточную для восстановления полной картины каждого применения этой TP. Журнал регистрации — это специальный элемент CDI, предназначенный только для добавления в него информации.

Правило C5. Любая TP, которая обрабатывает элемент UDI, должна выполнять только корректные преобразования этого элемента, в результате которых UDI превращается в CDI.

Правило E4. Только специально уполномоченное лицо может изменять списки, определенные в правилах C2 и E2. Это лицо не имеет права выполнять какие-либо действия, если оно уполномочено изменять регламентирующие эти действия списки.

Публикация описания модели Кларка-Вилсона вызвала широкий отклик среди исследователей, занимающихся проблемой контроля целостности. В ряде научных статей рассматриваются практические аспекты применения модели, предложены некоторые ее расширения и способы интеграции с другими моделями безопасности.

Роль каждого из девяти правил модели Кларка-Вилсона в обеспечении целостности информации можно пояснить показав, каким из теоретических принципов политики контроля целостности отвечает данное правило:

1. корректность транзакций;
2. аутентификация пользователей;
3. минимизация привилегий;
4. разграничение функциональных обязанностей;
5. аудит произошедших событий;
6. объективный контроль.

Соответствие правил модели Кларка-Вилсона перечисленным принципам показано в таблице. Как видно из таблицы, принципы 1 (корректность транзакций) и 4 (разграничение функциональных обязанностей) реализуются большинством правил, что соответствует основной идее модели.

Таблица

Правило модели Кларка-Вилсона	Принципы политики контроля целостности, реализуемые правилом
C1	1,6
C2	1
E1	3,4
E2	1,2,3,4
C3	4
E3	2

C4	5
C5	1
E4	4

Модель Кларка-Вилсона выражается в терминах набора правил функционирования и обслуживания данного компьютерного окружения или приложения. Эти правила вырабатываются для обеспечения уровня защиты целостности для некоторого заданного подмножества данных в этом окружении или приложении. Критическим понятием модели Кларка-Вилсона является то, что эти правила выражаются с использованием так называемых правильно сформированных транзакций, в которых субъект инициирует последовательность действий, которая выполняется управляемым и предсказуемым образом. В этом параграфе представим несколько исходных концепций модели Кларка-Вилсона, включая правильно сформированные транзакции, и опишем основные правила модели Кларка-Вилсона. Определение каждого правила включает обсуждение проблем практической реализации.

Пример применения модели

В качестве примера рассмотрим систему форумов FORUM.TOMSK.RU

- CDI – контролируемые элементы данных: логин и пароль
- UDI – неконтролируемые элементы данных: вводимые данные через интерфейс пользователя.
- IVP – процедуры контроля целостности проверяют соответствие введенных пользователем логина и пароля с зарегистрированным логином и паролем (которые хранятся в базе данных системы). Процедуры проверки корректности данных ввода и хранимой информации.

- TP – процедуры преобразования: процедуры преобразования изменяют состав множества всех контролируемых элементов данных путем редактирования, создания, ввода, удаления и т.п. В частности – редактирование писем и их распределение по папкам.

C1: все процедуры преобразования данных соответствуют определенному пользователю, что в свою очередь обеспечивает конфиденциальность хранимой пользователем информации.

C2: все процедуры преобразования реализованы таким образом, чтобы не изменять системные файлы и папки. Для процедуры преобразования – удаление – установлен список тех данных, на которых эта процедура не сможет воздействовать и сможет влиять только в рамках тех прав, которые были установлены для пользователя с конкретными логином и паролем (пользователь не может удалить информацию о другом сеансе).

E1: должна обеспечиваться на уровне программного средства, должен быть установлен контроль доступа в соответствии со списками, которые были установлены в правиле C2, о применении процедур преобразования к соответствующим CDI.

E2: соответствие логина и пароля с доступом к определенной информации, отождествление пользователей с предназначенным для них списком прав. Четкое разграничение функциональных возможностей и обязанностей для каждого пользователя системы.

C3: администратор имеет право изменять логин и пароль, а также права доступа к информации, но не может менять данные в базе.

E3: при попытке пользователя совершить какую-либо операцию, каждый раз производиться аутентификация пользователя (более того, аутентификация пользователя происходит каждые 10 мин).

C4: Каждая операция, совершаемая пользователем, записывается в журнал историй.

C5: Сначала производиться поиск логина , далее соответствующего пароля в базе, а затем определяются права доступа к информации.

E4: должно быть определено уполномоченное лицо – администратор системы. Он определяет права и контролирует работу системы.

Задание

Необходимо взять какую-либо информационную систему и определить:

- CDI – контролируемые элементы данных.
- UDI – неконтролируемые элементы данных.
- IVP – процедуры контроля целостности.
- TP – процедуры преобразования.

Выделить объект(ы), в которых хранятся списки, определенные в правилах C2 и E2, а также хранится журнал регистрации событий. Указать кто может изменять списки, определенные с правилах C2 и E2.

Создать правила, которые соответствовали ли бы девяти правилам, определяющим взаимоотношения элементов данных и процедур в процессе функционирования системы.

Провести проверку целостности системы с использованием теоретических принципов политики контроля целостности.

Варианты объектов исследования:

1. www-сервер
2. ftp-сервер
3. Почтовая база данных
4. База данных Microsoft Access
5. Операционная система
6. Жесткий диск
7. Сотовый телефон

Лабораторная работа 3
Стеганография

Цель работы

Целью работы является закрепление теоретического материала по стеганографии

Основные положения стеганографии

Стеганография – это метод организации связи, который собственно скрывает само наличие связи. В отличие от криптографии, где неприятель точно может определить является ли передаваемое сообщение зашифрованным текстом, методы стеганографии позволяют встраивать секретные сообщения в безобидные послания так, чтобы невозможно было заподозрить существование встроенного тайного послания.

Слово "стеганография" в переводе с греческого буквально означает "тайнопись" (steganos – секрет, тайна; graphy – запись). К ней относится огромное множество секретных средств связи, таких как невидимые чернила, микрофотоснимки, условное расположение знаков, тайные каналы и средства связи на плавающих частотах и т. д.

Стеганография занимает свою нишу в обеспечении безопасности: она не заменяет, а дополняет криптографию. Сокрытие сообщения методами стеганографии значительно снижает вероятность обнаружения самого факта передачи сообщения. А если это сообщение к тому же зашифровано, то оно имеет еще один, дополнительный, уровень защиты.

В настоящее время в связи с бурным развитием вычислительной техники и новых каналов передачи информации появились новые стеганографические методы, в основе которых лежат особенности представления информации в компьютерных файлах, вычислительных сетях и т. п. Это дает нам возможность говорить о становлении нового направления – компьютерной стеганографии.

Термины и определения

Несмотря на то что стеганография как способ сокрытия секретных данных известна уже на протяжении тысячелетий, компьютерная стеганография – молодое и развивающееся направление.

Как и любое новое направление, компьютерная стеганография, несмотря на большое количество открытых публикаций и ежегодные конференции, долгое время не имела единой терминологии.

До недавнего времени для описания модели стеганографической системы использовалась предложенная 1983 году Симмонсом [3] так называемая "проблема заключенных". Она состоит в том, что два индивидуума (Алиса и Боб) хотят обмениваться секретными сообщениями без вмешательства охранника (Вилли), контролирующего коммуникационный канал. При этом имеется ряд допущений, которые делают эту проблему более или менее решаемой. Первое допущение облегчает решение проблемы и состоит в том, что участники информационного обмена могут разделять секретное сообщение (например, используя кодовую клавишу) перед заключением. Другое допущение, наоборот, затрудняет решение проблемы, так как охранник имеет право не только читать сообщения, но и модифицировать (изменять) их.

Позднее, на конференции Information Hiding: First Information Workshop в 1996 году было предложено использовать единую терминологию и обговорены основные термины [4].

Стеганографическая система или **стегосистема** – совокупность средств и методов, которые используются для формирования скрытого канала передачи информации.

При построении стегосистемы должны учитываться следующие положения:

- противник имеет полное представление о стеганографической системе и деталях ее реализации. Единственной информацией, которая остается неизвестной потенциальному противнику, является

ключ, с помощью которого только его держатель может установить факт присутствия и содержание скрытого сообщения;

- если противник каким-то образом узнает о факте существования скрытого сообщения, это не должно позволить ему извлечь подобные сообщения в других данных до тех пор, пока ключ хранится в тайне;
- потенциальный противник должен быть лишен каких-либо технических и иных преимуществ в распознавании или раскрытии содержания тайных сообщений.

Обобщенная модель стегосистемы представлена на рисунке:

В качестве данных может использоваться любая информация: текст, сообщение, изображение и т. п.

В общем же случае целесообразно использовать слово "сообщение", так как сообщением может быть как текст или изображение, так и, например, аудиоданные. Далее для обозначения скрываемой информации, будем использовать именно термин сообщение.

Контейнер - любая информация, предназначенная для сокрытия тайных сообщений.

Пустой контейнер – контейнер без встроенного сообщения; заполненный контейнер или стего - контейнер, содержащий встроенную информацию.

Встроенное (скрытое) сообщение – сообщение, встраиваемое в контейнер.

Стеганографический канал или просто стегоканал – канал передачи стего.

Стегоключ или просто ключ – секретный ключ, необходимый для сокрытия информации. В зависимости от количества уровней защиты (например, встраивание предварительно зашифрованного сообщения) в стегосистеме может быть один или несколько стегоключей.

По аналогии с криптографией, по типу стегоключа стегосистемы можно подразделить на два типа:

- с секретным ключом;
- с открытым ключом.

В стегосистеме с секретным ключом используется один ключ, который должен быть определен либо до начала обмена секретными сообщениями, либо передан по защищенному каналу.

В стегосистеме с открытым ключом для встраивания и извлечения сообщения используются разные ключи, которые различаются таким образом, что с помощью вычислений невозможно вывести один ключ из другого. Поэтому один ключ (открытый) может передаваться свободно по незащищенному каналу связи. Кроме того, данная схема хорошо работает и при взаимном недоверии отправителя и получателя.

Требования

Любая стегосистема должна отвечать следующим требованиям:

- Свойства контейнера должны быть модифицированы, чтобы изменение невозможно было выявить при визуальном контроле. Это требование определяет качество сокрытия внедряемого сообщения: для обеспечения беспрепятственного прохождения стегосообщения по каналу связи оно никоим образом не должно привлечь внимание атакующего.
- Стегосообщение должно быть устойчиво к искажениям, в том числе и злонамеренным. В процессе передачи изображение (звук или другой контейнер) может претерпевать различные трансформации: уменьшаться или увеличиваться,

преобразовываться в другой формат и т. д. Кроме того, оно может быть сжато, в том числе и с использованием алгоритмов сжатия с потерей данных.

- Для сохранения целостности встраиваемого сообщения необходимо использование кода с исправлением ошибки.
- Для повышения надежности встраиваемое сообщение должно быть продублировано.

Приложения

В настоящее время можно выделить три тесно связанных между собой и имеющих одни корни направления приложения стеганографии: сокрытие данных (сообщений), цифровые водяные знаки и заголовки.

Сокрытие внедряемых данных, которые в большинстве случаев имеют большой объем, предъявляет серьезные требования к контейнеру: размер контейнера в несколько раз должен превышать размер встраиваемых данных.

Цифровые водяные знаки используются для защиты авторских или имущественных прав на цифровые изображения, фотографии или другие оцифрованные произведения искусства. Основными требованиями, которые предъявляются к таким встроенным данным, являются надежность и устойчивость к искажениям.

Цифровые водяные знаки имеют небольшой объем, однако, с учетом указанных выше требований, для их встраивания используются более сложные методы, чем для встраивания просто сообщений или заголовков.

Третье приложение, заголовки, используется в основном для маркирования изображений в больших электронных хранилищах (библиотеках) цифровых изображений, аудио- и видеофайлов.

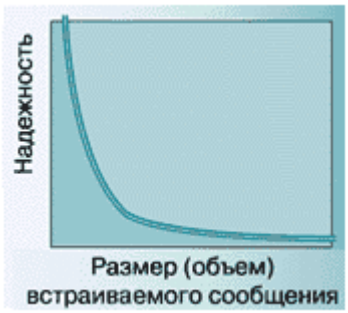
В данном случае стеганографические методы используются не только для внедрения идентифицирующего заголовка, но и иных индивидуальных признаков файла.

Внедряемые заголовки имеют небольшой объем, а предъявляемые к ним требования минимальны: заголовки должны вносить незначительные искажения и быть устойчивы к основным геометрическим преобразованиям.

Ограничения

Каждое из перечисленных выше приложений требует определенного соотношения между устойчивостью встроенного сообщения к внешним воздействиям (в том числе и стегоанализу) и размером самого встраиваемого сообщения.

Для большинства современных методов, используемых для сокрытия сообщения в цифровых контейнерах, имеет место следующая зависимость надежности системы от объема встраиваемых данных:



Данная зависимость показывает, что при увеличении объема встраиваемых данных снижается надежность системы (при неизменности размера контейнера). Таким образом, используемый в стегосистеме контейнер накладывает ограничения на размер встраиваемых данных.

Контейнеры

Существенное влияние на надежность стегосистемы и возможность обнаружения факта передачи скрытого сообщения оказывает выбор контейнера.

Например, опытный глаз цензора с художественным образованием легко обнаружит изменение цветовой гаммы при внедрении сообщения в репродукцию "Мадонны" Рафаэля или "Черного квадрата" Малевича.

По протяженности контейнеры можно подразделить на два типа: непрерывные (потоковые) и ограниченной (фиксированной) длины. Особенностью потокового контейнера является то, что невозможно определить его начало или конец. Более того, нет возможности узнать заранее, какими будут последующие шумовые биты, что приводит к необходимости включать скрывающие сообщение биты в поток в реальном масштабе времени, а сами скрывающие биты выбираются с помощью специального генератора, задающего расстояние между последовательными битами в потоке.

В непрерывном потоке данных самая большая трудность для получателя - определить, когда начинается скрытое сообщение. При наличии в потоковом контейнере сигналов синхронизации или границ пакета, скрытое сообщение начинается сразу после одного из них. В свою очередь, для отправителя возможны проблемы, если он не уверен в том, что поток контейнера будет достаточно долгим для размещения целого тайного сообщения.

При использовании контейнеров фиксированной длины отправитель заранее знает размер файла и может выбрать скрывающие биты в подходящей псевдослучайной последовательности. С другой стороны, контейнеры фиксированной длины, как это уже отмечалось выше, имеют ограниченный объем и иногда встраиваемое сообщение может не поместиться в файл-контейнер.

Другой недостаток заключается в том, что расстояния между скрывающими битами равномерно распределены между наиболее коротким и наиболее длинным заданными

расстояниями, в то время как истинный случайный шум будет иметь экспоненциальное распределение длин интервала. Конечно, можно породить псевдослучайные экспоненциально распределенные числа, но этот путь обычно слишком трудоемок. Однако на практике чаще всего используются именно контейнеры фиксированной длины, как наиболее распространенные и доступные.

Возможны следующие варианты контейнеров:

- Контейнер генерируется самой стегосистемой. Примером может служить программа MandelSteg, в которой в качестве контейнера для встраивания сообщения генерируется фрактал Мандельброта. Такой подход можно назвать конструирующей стеганографией.
- Контейнер выбирается из некоторого множества контейнеров. В этом случае генерируется большое число альтернативных контейнеров, чтобы затем выбрать наиболее подходящий для сокрытия сообщения. Такой подход можно назвать селектирующей стеганографией. В данном случае при выборе оптимального контейнера из множества сгенерированных важнейшим требованием является естественность контейнера. Единственной же проблемой остается то, что даже оптимально организованный контейнер позволяет спрятать незначительное количество данных при очень большом объеме самого контейнера.
- Контейнер поступает извне. В данном случае отсутствует возможность выбора контейнера и для сокрытия сообщения берется первый попавшийся контейнер, не всегда подходящий к встраиваемому сообщению. Назовем это безальтернативной стеганографией.

Методы

В настоящее время существует достаточно много различных методов (и их вариантов) встраивания сообщений

(имеется в виду и встраивание цифровых водяных знаков). Из-за ограниченности объема публикации невозможно описать все используемые методы, однако некоторые из них достаточно хорошо описаны в специальной литературе [8, 9, 10, 11].

Методы сокрытия информации

В настоящее время наиболее распространенным, но наименее стойким является метод замены наименьших значащих битов или LSB-метод. Он заключается в использовании погрешности дискретизации, которая всегда существует в оцифрованных изображениях или аудио- и видеофайлах. Данная погрешность равна наименьшему значащему разряду числа, определяющему величину цветовой составляющей элемента изображения (пикселя). Поэтому модификация младших битов в большинстве случаев не вызывает значительной трансформации изображения и не обнаруживается визуально. Более подробно LSB-метод описан в статье В. Н. Кустова и А. А. Федчука "Методы встраивания скрытых сообщений" ("Защита информации. Конфидент", №3, 2000, стр. 34).

Другим популярным методом встраивания сообщений является использование особенностей форматов данных, использующих сжатие с потерей данных (например JPEG). Этот метод (в отличии от LSB) более стоек к геометрическим преобразованиям и обнаружению канала передачи, так как имеется возможность в широком диапазоне варьировать качество сжатого изображения, что делает невозможным определение происхождения искажения. Более подробно этот метод описан в статье С. Ф. Быкова "Алгоритм сжатия JPEG с позиции компьютерной стеганографии" ("Защита информации. Конфидент", №3, 2000, стр. 26).

Для встраивания цифровых водяных знаков используются более сложные методы.

Цифровые водяные знаки

В современных системах формирования цифровых водяных знаков используется принцип встраивания метки,

являющейся узкополосным сигналом, в широком диапазоне частот маркируемого изображения. Указанный метод реализуется при помощи двух различных алгоритмов и их возможных модификаций. В первом случае информация скрывается путем фазовой модуляции информационного сигнала (несущей) с псевдослучайной последовательностью чисел. Во втором - имеющийся диапазон частот делится на несколько каналов и передача производится между этими каналами. Относительно исходного изображения метка является некоторым дополнительным шумом, но так как шум в сигнале присутствует всегда, его незначительное возрастание за счет внедрения метки не дает заметных на глаз искажений. Кроме того, метка рассеивается по всему исходному изображению, в результате чего становится более устойчивой к вырезанию.

В настоящее время компьютерная стеганография продолжает развиваться: формируется теоретическая база, ведется разработка новых, более стойких методов встраивания сообщений. Среди основных причин наблюдающегося всплеска интереса к стеганографии можно выделить принятые в ряде стран ограничения на использование сильной криптографии, а также проблему защиты авторских прав на художественные произведения в цифровых глобальных сетях. Поэтому в ближайшее время можно ожидать новых публикаций и разработок в этой области.

ИСТОРИЧЕСКИЕ ЗАМЕТКИ

История стеганографии – это история развития человечества. Местом зарождения стеганографии многие называют Египет, хотя первыми "стеганографическими сообщениями" можно назвать и наскальные рисунки древних людей.

Первое упоминание о стеганографических методах в литературе приписывается Геродоту, который описал случай передачи сообщения Демартом, который соскабливал воск с

дощечек, писал письмо прямо на дереве, а потом заново покрывал дощечки воском.

Другой эпизод, который относят к тем же временам - передача послания с использованием головы раба. Для передачи тайного сообщения голову раба обривали, наносили на кожу татуировку, и когда волосы отрастали, отправляли с посланием.

В Китае письма писали на полосках щелка. Поэтому для сокрытия сообщений, полоски с текстом письма, сворачивались в шарики, покрывались воском и затем глотались посыльными.

Темное средневековье породило не только инквизицию: усиление слежки привело к развитию как криптографии, так и стеганографии. Именно в средние века впервые было применено совместное использование шифров и стеганографических методов.

В XV веке монах Тритемиус (1462-1516), занимавшийся криптографией и стеганографией, описал много различных методов скрытой передачи сообщений. Позднее, в 1499 году, эти записи были объединены в книгу "Steganographia", которую в настоящее время знающие латынь могут прочитать в Интернет.

XVII - XVIII века известны как эра "черных кабинетов" - специальных государственных органов по перехвату, перлюстрации и дешифрованию переписки. В штат "черных кабинетов", помимо криптографов и дешифровальщиков, входили и другие специалисты, в том числе и химики. Наличие специалистов-химиков было необходимо из-за активного использования так называемых невидимых чернил. Примером может служить любопытный исторический эпизод: восставшими дворянами в Бордо был арестован францисканский монах Берто, являвшийся агентом кардинала Мазарини. Восставшие разрешили Берто написать письмо знакомому священнику в город Блэй. Однако в конце этого письма религиозного содержания, монах сделал приписку, на которую никто не обратил внимание: "Посылаю Вам глазную

мазь; натрите ею глаза и Вы будете лучше видеть". Так он сумел переслать не только скрытое сообщение, но и указал способ его обнаружения. В результате монах Берто был спасен.

Стеганографические методы активно использовались и в годы гражданской войны между южанами и северянами. Так, в 1779 году два агента северян Сэмюэль Вудхулл и Роберт Тоунсенд передавали информацию Джорджу Вашингтону, используя специальные чернила.

Различные симпатические чернила использовали и русские революционеры в начале XX века, что нашло отражение в советской литературе: Куканов в своей повести "У истоков грядущего" описывает применение молока в качестве чернил для написания тайных сообщений. Впрочем, царская охранка тоже знала об этом методе (в архиве хранится документ, в котором описан способ использования симпатических чернил и приведен текст перехваченного тайного сообщения революционеров).

Особое место в истории стеганографии занимают фотографические микроточки. Да, те самые микроточки, которые сводили с ума спецслужбы США во время второй мировой войны. Однако микроточки появились намного раньше, сразу же после изобретения Дагером фотографического процесса, и впервые в военном деле были использованы во времена франко-прусской войны (в 1870 году).

Конечно, можно еще упомянуть акростихи и другие языковые игры. Однако описать все изобретенные человечеством стеганографические методы нам не позволит объем данной публикации.

Литература:

1. Kahn D. The Codebreakers. N-Y, 1967.
2. Жельников В. Криптография от папируса до компьютера. М., 1996.

3. Simmons G.J. The prisoner`s problem and the subliminal channel, Proc. Workshop on Communications Security (Crypto`83), 1984, 51-67.
4. Pfitzmann B. Information Hiding Terminology, in Information Hiding, Springer Lecture Notes in Computer Science, v.1174, 1996, 347-350.
5. Aura T. Invisible communication. In Proc. of the HUT Seminar on Network Security '95, Espoo, Finland, November 1995. Telecommunications Software and Multimedia Laboratory, Helsinki University of Technology.
6. Ross J. Anderson. Stretching the limits of steganography. In IH96 [3], pages 39-48.
7. Zollner J., Federrath H.,Klimant H., Pfitzmann A., Piotraschke R., Westfeld A., Wicke G., Wolf G. Modeling the security of steganographic system, Proc. 2nd International Workshop on Information Hiding, 1998, LNCS, v.1525, 344-354.
8. E. Franz, A. Jerichow, S. Moller, A. Pfitzmann, I. Stierand. Computer Based Steganography: How it works and why therefore any restrictions on cryptography are nonsense, at best, In Information hiding: first international workshop, Cambridge, UK. Lecture Notes in Computer Science, vol. 1174, Berlin Heidelberg New York: Springer-Verlag, 1996.
9. N.F. Johnson, S. Jajodia. Exploring Steganography: Seeing the Unseen, IEEE Computer, February 1998, vol. 31, no. 2, pp.26-34.
10. Walter Bender, Daniel Gruhl, Norishige Morimoto, and Anthony Lu. Techniques for data hiding. IBM Systems Journal, 35(3 & 4):313{336, 1996.
11. Raymond B. Wolfgang and Edward J. Delp. A watermark for digital images. In International Conference on Images Processing, pages 219-222, Lausanne, Switzer-land, September 1996. IEEE.

Задание

Необходимо выполнить:

1. Рассмотреть работу двух программ, позволяющих проводить стеганографические преобразования.
2. Выбрать контейнер и выполнить внедрение в него некоторой информации.
3. Попробовать извлечь информацию из стегоконтейнера, созданного другой программой.
4. Представить результаты в отчете

Лабораторная работа 4
Криптография. Шифрование

Цель работы

Целью работы является закрепление теоретического материала по применению криптографических алгоритмов шифрования и получение практических навыков по работе прикладными пакетами, реализующими шифрование.

Теория

Проблема защиты информации путем ее преобразования, исключающего ее прочтение посторонним лицом волновала человеческий ум с давних времен. История криптографии — ровесница истории человеческого языка. Более того, первоначально письменность сама по себе была криптографической системой, так как в древних обществах ею владели только избранные. Священные книги Древнего Египта, Древней Индии тому примеры.

Разные люди понимают под шифрованием разные вещи. Дети играют в игрушечные шифры и секретные языки. Это, однако, не имеет ничего общего с настоящей криптографией. Настоящая криптография (**strong cryptography**) должна обеспечивать такой уровень секретности, чтобы можно было надежно защитить критическую информацию от расшифровки крупными организациями — такими как мафия, транснациональные корпорации и крупные государства. Настоящая криптография в прошлом использовалась лишь в военных целях. Однако сейчас, с становлением информационного общества, она становится центральным инструментом для обеспечения конфиденциальности.

По мере образования информационного общества, крупным государствам становятся доступны технологические средства тотального надзора за миллионами людей. Поэтому криптография становится одним из основных инструментов обеспечивающих конфиденциальность, доверие, авторизацию, электронные платежи, корпоративную

безопасность и бесчисленное множество других важных вещей.

Криптография не является более придумкой военных, с которой не стоит связываться. Настала пора снять с криптографии покровы таинственности и использовать все ее возможности на пользу современному обществу. Широкое распространение криптографии является одним из немногих способов защитить человека от ситуации, когда он вдруг обнаруживает, что живет в тоталитарном государстве, которое может контролировать каждый его шаг.

Бурное развитие криптографические системы получили в годы первой и второй мировых войн. Начиная с послевоенного времени и по нынешний день появление вычислительных средств ускорило разработку и совершенствование криптографических методов.

Почему проблема использования криптографических методов в информационных системах (ИС) стала в настоящий момент особо актуальна?

С одной стороны, расширилось использование компьютерных сетей, в частности глобальной сети Интернет, по которым передаются большие объемы информации государственного, военного, коммерческого и частного характера, не допускающего возможность доступа к ней посторонних лиц.

С другой стороны, появление новых мощных компьютеров, технологий сетевых и нейронных вычислений сделало возможным дискредитацию криптографических систем еще недавно считавшихся практически не раскрываемыми.

Проблемой защиты информации путем ее преобразования занимается *криптология* (*kryptos* — тайный, *logos* — наука). Криптология разделяется на два направления — *криптографию* и *криптоанализ*. Цели этих направлений прямо противоположны.

Криптография занимается поиском и исследованием математических методов преобразования информации.

Сфера интересов *криптоанализа* — исследование возможности расшифровывания информации без знания ключей.

Современная криптография включает в себя четыре крупных раздела:

1. Симметричные криптосистемы.
2. Криптосистемы с открытым ключом.
3. Системы электронной подписи.
4. Управление ключами.

Основные направления использования криптографических методов — передача конфиденциальной информации по каналам связи (например, электронная почта), установление подлинности передаваемых сообщений, хранение информации (документов, баз данных) на носителях в зашифрованном виде.

Терминология

Итак, криптография дает возможность преобразовать информацию таким образом, что ее прочтение (восстановление) возможно только при знании ключа.

В качестве информации, подлежащей шифрованию и дешифрованию, будут рассматриваться *тексты*, построенные на некотором *алфавите*. Под этими терминами понимается следующее.

Алфавит — конечное множество используемых для кодирования информации знаков.

Текст — упорядоченный набор из элементов алфавита.

В качестве примеров алфавитов, используемых в современных ИС можно привести следующие:

- алфавит Z_{33} - 32 буквы русского алфавита и пробел;
- алфавит Z_{256} - символы, входящие в стандартные коды ASCII и КОИ-8;
- бинарный алфавит - $Z_2 = \{0,1\}$;
- восьмеричный алфавит или шестнадцатеричный алфавит;

Шифрование — преобразовательный процесс: *исходный*

текст, который носит также название *открытого текста*, заменяется *шифрованным текстом*.

Дешифрование — обратный шифрованию процесс. На основе ключа шифрованный текст преобразуется в исходный.

Ключ - информация, необходимая для беспрепятственного шифрования и дешифрования текстов.

Криптографическая система представляет собой семейство $\mathbf{T\ [T_1, T_2, \ldots, T_k]}$ преобразований открытого текста. Члены этого семейства индексируются, или обозначаются символом *k*; параметр *k* является *ключом*. Пространство ключей *K* — это набор возможных значений ключа. Обычно ключ представляет собой последовательный ряд букв алфавита.

Криптосистемы разделяются на *симметричные* и *с открытым ключом*.

В *симметричных криптосистемах* и для шифрования, и для дешифрования используется *один и тот же ключ*.

В *системах с открытым ключом* используются два ключа — *открытый* и *закрытый*, которые математически связаны друг с другом. Информация шифруется с помощью открытого ключа, который доступен всем желающим, а расшифровывается с помощью закрытого ключа, известного только получателю сообщения [29].

Термины *распределение ключей* и *управление ключами* относятся к процессам системы обработки информации, содержанием которых является составление и распределение ключей между пользователями.

Электронной (цифровой) подписью называется присоединяемое к тексту его криптографическое преобразование, которое позволяет при получении текста другим пользователем проверить авторство и подлинность сообщения.

Криптостойкостью называется характеристика шифра, определяющая его стойкость к дешифрованию без знания ключа (т.е. криптоанализу). Имеется несколько показателей криптостойкости, среди которых:

- количество всех возможных ключей;
- среднее время, необходимое для криптоанализа.

Преобразование T_k определяется соответствующим алгоритмом и значением параметра k. Эффективность шифрования с целью защиты информации зависит от сохранения тайны ключа и криптостойкости шифра.

Требования к криптосистемам

Процесс криптографического закрытия данных может осуществляться как программно, так и аппаратно. Аппаратная реализация отличается существенно большей стоимостью, однако ей присущи и преимущества: высокая производительность, простота, защищенность и т.д. Программная реализация более практична, допускает известную гибкость в использовании.

Для современных криптографических систем защиты информации сформулированы следующие общепринятые требования:

- зашифрованное сообщение должно поддаваться чтению только при наличии ключа;
- число операций, необходимых для определения использованного ключа шифрования по фрагменту шифрованного сообщения и соответствующего ему открытого текста, должно быть не меньше общего числа возможных ключей;
- число операций, необходимых для расшифровывания информации путем перебора всевозможных ключей должно иметь строгую нижнюю оценку и выходить за пределы возможностей современных компьютеров (с учетом возможности использования сетевых вычислений);
- знание алгоритма шифрования не должно влиять на надежность защиты;
- незначительное изменение ключа должно приводить к существенному изменению вида зашифрованного сообщения даже при использовании одного и того же

ключа;

- структурные элементы алгоритма шифрования должны быть неизменными;
- дополнительные биты, вводимые в сообщение в процессе шифрования, должен быть полностью и надежно скрыты в шифрованном тексте;
- длина шифрованного текста должна быть равной длине исходного текста;
- не должно быть простых и легко устанавливаемых зависимостью между ключами, последовательно используемыми в процессе шифрования;
- любой ключ из множества возможных должен обеспечивать надежную защиту информации;
- алгоритм должен допускать как программную, так и аппаратную реализацию, при этом изменение длины ключа не должно вести к качественному ухудшению алгоритма шифрования.

Основные алгоритмы шифрования

Метод шифровки/дешифровки называют **шифром** (**cipher**). Некоторые алгоритмы шифрования основаны на том, что сам метод шифрования (алгоритм) является секретным. Ныне такие методы представляют лишь исторический интерес и не имеют практического значения. Все современные алгоритмы используют **ключ** для управления шифровкой и дешифровкой; сообщение может быть успешно дешифровано только если известен ключ. Ключ, используемый для дешифровки может не совпадать с ключом, используемым для шифрования, однако в большинстве алгоритмов ключи совпадают.

Алгоритмы с использованием ключа делятся на два класса: симметричные (или алгоритмы секретным ключом) и асимметричные (или алгоритмы с открытым ключом). Разница в том, что симметричные алгоритмы используют один и тот же ключ для шифрования и для дешифрования (или же ключ для дешифровки просто вычисляется по ключу

шифровки). В то время как асимметричные алгоритмы используют разные ключи, и ключ для дешифровки не может быть вычислен по ключу шифровки.

Симметричные алгоритмы подразделяют на **потоковые шифры** и **блочные шифры**. Потоковые позволяют шифровать информацию побитово, в то время как блочные работают с некоторым набором бит данных (обычно размер блока составляет 64 бита) и шифруют этот набор как единое целое.

Асимметричные шифры (также именуемые алгоритмами с открытым ключом, или — в более общем плане — криптографией с открытым ключом) допускают, чтобы открытый ключ был доступн всем (скажем, опубликован в газете). Это позволяет любому зашифровать сообщение. Однако расшифровать это сообщение сможет только нужный человек (тот, кто владеет ключом дешифровки). Ключ для шифрования называют **открытым ключом**, а ключ для дешифрования — **закрытым ключом** или **секретным ключом**.

Современные алгоритмы шифровки/дешифровки достаточно сложны и их невозможно проводить вручную. Настоящие криптографические алгоритмы разработаны для использования компьютерами или специальными аппаратными устройствами. В большинстве приложений криптография производится программным обеспечением и имеется множество доступных криптографических пакетов.

Вообще говоря, симметричные алгоритмы работают быстрее, чем асимметричные. На практике оба типа алгоритмов часто используются вместе: алгоритм с открытым ключом используется для того, чтобы передать случайным образом сгенерированный секретный ключ, который затем используется для дешифровки сообщения.

Многие качественные криптографические алгоритмы доступны широко — в книжном магазине, библиотеке, патентном бюро или в Интернет. К широко известным симметричным алгоритмам относятся DES и IDEA, Наверное

самым лучшим асимметричным алгоритмом является RSA. В России за стандарт шифрования принят ГОСТ 28147-89.

В таблице приведена классификация криптографического закрытия информации.

Таблица - Криптографическое закрытие информации

Вид преобразований	Способ преобразования	Разновидность способа	Способ реализации
Шифрование	Замена (подстановка)	Простая (одноалфавитная)	П
		Многоалфавитная одноконтурная обыкновенна	П
		Многоалфавитная одноконтурная монофоническая	П
		Многоалфавитная многоконтурная	П
	Перестановка	Простая	П
		Усложненная по таблице	П
		Усложненная по маршрутам	П
	Аналитическое преобразование	По правилам алгебры матриц	П
		По особым зависимостям	П
	Гаммирование	С конечной короткой гаммой	АП
		С конечной динной гаммой	АП
		С бесконечной гаммой	АП
	Комбинированные	Замена+перестановка	АП
		Замена+гаммирование	АП
		Перестановка+ гаммирование	АП
		Гаммрование+гаммирование	АП
Кодирование	Смысловое	По специальным таблицам (словарям)	П
	Символьное	По кодовому алфавиту	П
Другие виды	Рассечение-разнесение	Смысловое	АП
		Механическое	П
	Сжатие-расширение		

Примечание. Способ реализации: А — аппаратный , П — программный.

Криптоанализ и атаки на криптосистемы

Криптоанализ — это наука о дешифровке закодированных сообщений не зная ключей. Имеется много криптоаналитических подходов. Некоторые из наиболее важных для разработчиков приведены ниже.

Атака со знанием лишь шифрованного текста (**ciphertext-only attack**): Это ситуация, когда атакующий не знает ничего о содержании сообщения, и ему приходится работать лишь с самим шифрованным текстом. На практике, часто можно сделать правдоподобные предположения о структуре текста, поскольку многие сообщения имеют стандартные заголовки. Даже обычные письма и документы начинаются с легко предсказуемой информации. Также часто можно предположить, что некоторый блок информации содержит заданное слово.

Атака со знанием содержимого шифровки (**known-plaintext attack**): Атакующий знает или может угадать содержимое всего или части зашифрованного текста. Задача заключается в расшифровке остального сообщения. Это можно сделать либо путем вычисления ключа шифровки, либо минуя это.

Атака с заданным текстом (**chosen-plaintext attack**): Атакующий имеет возможнот получить шифрованный документ для любого нужного ему текста, но не знает ключа. Задачей является нахождение ключа. Некоторые методы шифрования и, в частности, RSA, весьма уязвимы для атак этого типа. При использовании таких алгоритмов надо тщательно следить, чтобы атакующий не мог зашифровать заданный им текст.

Атака с подставкой (**Man-in-the-middle attack**): Атака направлена на обмен шифрованными сообщениями и, в особенности, на протокол обмена ключами. Идея заключается в том, что когда две стороны обмениваются ключами для секретной коммуникации (например, используя шифр Диффи-Хелмана, Diffie-Hellman), противник внедряется между ними на линии обмена сообщениями. Далее

противник выдает каждой стороне свои ключи. В результате, каждая из сторон будет иметь разные ключи, каждый из которых известен противнику. Теперь противник будет расшифровывать каждое сообщение своим ключом и затем зашифровывать его с помощью другого ключа перед отправкой адресату. Стороны будут иметь иллюзию секретной переписки, в то время как на самом деле противник читает все сообщения.

Одним из способов предотвратить такой тип атак заключается в том, что стороны при обмене ключами вычисляют криптографическую хэш-функцию значения протокола обмена (или по меньшей мере значения ключей), подписывают ее алгоритмом цифровой подписи и посылают подпись другой стороне. Получатель проверит подпись и то, что значение хэш-функции совпадает с вычисленным значением. Такой метод используется, в частности, в системе Фотурис (Photuris).

Атака с помощью таймера (**timing attack**): Этот новый тип атак основан на последовательном измерении времен, затрачиваемых на выполнение операции возведения в стенень по модулю целого числа. Ей подвержены по крайней мере следующие шифры: RSA, Диффи-Хеллман и метод эллиптических кривых.

Имеется множество других криптографических атак и криптоаналитических подходов. Однако приведенные выше являются, по-видимому, наиболее важными для практической разработки систем. Если кто-либо собирается создавать свой алгоритм шифрования, ему необходимо понимать данные вопросы значительно глубже.

Выбор для конкретных ИС должен быть основан на глубоком анализе слабых и сильных сторон тех или иных методов защиты. Обоснованный выбор той или иной системы защиты в общем-то должен опираться на какие-то *критерии эффективности*. К сожалению, до сих пор не разработаны подходящие методики оценки эффективности криптографических систем.

Наиболее простой критерий такой эффективности — *вероятность раскрытия ключа* или *мощность множества ключей (M)*. По сути это то же самое, что и *криптостойкость*. Для ее численной оценки можно использовать также и сложность раскрытия шифра путем перебора всех ключей.

Однако, этот критерий не учитывает других важных *требований к криптосистемам*:

- невозможность раскрытия или осмысленной модификации информации на основе анализа ее структуры,
- совершенство используемых протоколов защиты,
- минимальный объем используемой ключевой информации,
- минимальная сложность реализации (в количестве машинных операций), ее стоимость,
- высокая оперативность.

Желательно конечно использование некоторых интегральных показателей, учитывающих указанные факторы.

Для учета стоимости, трудоемкости и объема ключевой информации можно использовать удельные показатели — отношение указанных параметров к мощности множества ключей шифра.

Часто более эффективным при выборе и оценке криптографической системы является использование экспертных оценок и имитационное моделирование.

В любом случае выбранный комплекс криптографических методов должен сочетать как удобство, гибкость и оперативность использования, так и надежную защиту от злоумышленников циркулирующей в ИС информации.

Задание

1. Разобрать алгоритм работы с программой и ее возможности

2. Научиться шифровать, получать хеш-значения, цифровые подписи
3. Попробовать внести изменения в зашифрованный файл, цифровую подпись или хэш-значение и проверить реакцию программы на эти изменения
4. Сравнить различные алгоритмы шифрования/хэширования/цифровой подписи по размеру ключа/размеру файла/скорости работы/…
5. Представить результаты в отчете

Лабораторная работа 5
Криптография. Электронно-цифровая подпись и хеширование

Цель работы

Целью работы является закрепление теоретического материала по применению алгоритмов электронной цифровой подписи и хеширования и получение практических навыков по работе прикладными пакетами, реализующими ЭЦП и хеширование.

Теоретическая часть работы

Цифровые подписи

Некоторые из асимметричных алгоритмов могут использоваться для генерирования **цифровой подписи**. Цифровой подписью называют блок данных, сгенерированный с использованием некоторого секретного ключа. При этом с помощью открытого ключа можно проверить, что данные были действительно сгенерированы с помощью этого секретного ключа. Алгоритм генерации цифровой подписи должен обеспечивать, чтобы было невозможно без секретного ключа создать подпись, которая при проверке окажется правильной.

Цифровые подписи используются для того, чтобы подтвердить, что сообщение пришло действительно от данного отправителя (в предположении, что лишь отправитель обладает секретным ключом, соответствующим его открытому ключу). Также подписи используются для проставления **штампа времени** (**timestamp**) на документах: сторона, которой мы доверяем, подписывает документ со штампом времени с помощью своего секретного ключа и, таким образом, подтверждает, что документ уже существовал в момент, объявленный в штампе времени.

Цифровые подписи также можно использовать для удостоверения (**сертификации — to certify**) того, что документ принадлежит определенному лицу. Это делается так: открытый ключ и информация о том, кому он принадлежит подписываются стороной, которой доверяем.

При этом доверять подписывающей стороне мы можем на основании того, что ее ключ был подписан третьей стороной. Таким образом возникает иерархия доверия. Очевидно, что некоторый ключ должен быть корнем иерархии (то есть ему мы доверяем не потому, что он кем-то подписан, а потому, что мы верим a-priori, что ему можно доверять). В **централизованной инфраструктуре ключей** имеется очень небольшое количество корневых ключей сети (например, облеченные полномочиями государственные агентства; их также называют **сертификационными агентствами — certification authorities**). В **распределенной инфраструктуре** нет необходимости иметь универсальные для всех корневые ключи, и каждая из сторон может доверять своему набору корневых ключей (скажем своему собственному ключу и ключам, ею подписанным). Эта концепция носит название **сети доверия** (**web of trust**) и реализована, например, в PGP.

Цифровая подпись документа обычно создается так: из документа генерируется так называемый **дайджест** (**message digest**) и к нему добавляется информация о том, кто подписывает документ, штамп времени и прочее. Получившаяся строка далее зашифровывается секретным ключом подписывающего с использованием того или иного алгоритма. Получившийся зашифрованный набор бит и представляет собой подпись. К подписи обычно прикладывается открытый ключ подписывающего. Получатель сначала решает для себя доверяет ли он тому, что открытый ключ принадлежит именно тому, кому должен принадлежать (с помощью сети доверия или априорного знания), и затем дешифрует подпись с помощью открытого ключа. Если подпись нормально дешифровалась, и ее содержимое соответствует документу (дайджест и др.), то сообщение считается подтвержденным.

Свободно доступны несколько методов создания и проверки цифровых подписей. Наиболее известным является алгоритм RSA, ГОСТ 34.10-94.

Криптографические хэш-функции

Криптографические хэш-функции используются обычно для генерации дайджеста сообщения при создании цифровой подписи. Хэш-функции отображают сообщение в имеющее фиксированный размер **хэш-значение** (**hash value**) таким образом, что все множество возможных сообщений распределяется равномерно по множеству хэш-значений. При этом криптографическая хэш-функция делает это таким образом, что практически невозможно подогнать документ к заданному хэш-значению.

Криптографические хэш-функции обычно производят значения длиной в 128 и более бит. Это число значительно больше, чем количество сообщений, которые когда-либо будут существовать в мире.

Много хороших криптографических хэш-функций доступно бесплатно. Широко известные включают MD5 и SHA.

Криптографические генераторы случайных чисел

Криптографические генераторы случайных чисел производят случайные числа, которые используются в криптографических приложениях, например — для генерации ключей. Обычные генераторы случайных чисел, имеющиеся во многих языках программирования и программных средах, не подходят для нужд криптографии (они создавались с целью получить статистически случайное распределение, криптоаналитики могут предсказать поведение таких случайных генераторов).

В идеале случайные числа должны основываться на настоящем физическом источнике случайной информации, которую невозможно предсказать. Примеры таких источников включают шумящие полупроводниковые приборы, младшие биты оцифрованного звука, интервалы между прерываниями устройств или нажатиями клавиш. Полученный от физического источника шум затем "дистиллируется"

криптографической хэш-функцией так, чтобы каждый бит зависел от каждого бита. Достаточно часто для хранения случайной информации используется довольно большой пул (несколько тысяч бит) и каждый бит пула делается зависимым от каждого бита шумовой информации и каждого другого бита пула криптографически надежным (**strong**) способом.

Когда нет настоящего физического источника шума, приходится пользоваться псевдослучайными числами. Такая ситуация нежелательна, но часто возникает на компьютерах общего назначения. Всегда желательно получить некий шум окружения — скажем от величины задержек в устройствах, цифры статистики использования ресурсов, сетевой статистики, прерываний от клавиатуры или чего-то иного. Задачей является получить данные, непредсказуемые для внешнего наблюдателя. Для достижения этого случайный пул должен содержать как минимум 128 бит настоящей энтропии.

Криптографические генераторы псевдослучайных чисел обычно используют большой пул (seed-значение), содержащий случайную информацию. Биты генерируется путем выборки из пула с возможным прогоном через криптографическую хэш-функцию, чтобы спрятать содержимое пула от внешнего наблюдателя. Когда требуется новая порция бит, пул перемешивается путем шифровки со случайным ключом (его можно взять из неиспользованной пока части пула) так, чтобы каждый бит пула зависел от каждого другого бита. Новый шум окружения должен добавляться к пулу перед перемешиваниям, дабы сделать предсказание новых значений пула еще более сложным.

Несмотря на то, что при аккуратном проектировании криптографически надежный генератор случайных чисел реализовать не так уж и трудно, этот вопрос часто упускают из вида. Таким образом, следует подчеркнуть важность криптографического генератора случайных чисел — если он сделан плохо, он может легко стать самым уязвимым элементом системы.

Задание

1. Разобрать алгоритм работы с программой и ее возможности
2. Научиться шифровать, получать хеш-значения, цифровые подписи
3. Попробовать внести изменения в зашифрованный файл, цифровую подпись или хэш-значение и проверить реакцию программы на эти изменения
4. Сравнить различные алгоритмы шифрования/хэширования/цифровой подписи по размеру ключа/размеру файла/скорости работы/…
5. Представить результаты в отчете

Лабораторная работа 6
Субъект-объектная модель. Изолированная программная среда

Цель работы

Закрепление теоретического материала по субъект-объектной модели политики безопасности, исследование заданной системы на соответствие требованиям заданной политики безопасности

Теоретическая часть

Понятия доступа и монитора безопасности

В теории информационной безопасности практически всегда рассматривается модель произвольной АС а виде конечного множества элементов. Указанное множество можно разделить на два подмножества: множество объектов и множество субъектов. Данное разделение основано на свойстве элемента "быть активным" или "получать управление" (применяется также термин "использовать ресурсы" или "пользоваться вычислительной мощностью"). Оно исторически сложилось на основе модели вычислительной системы, принадлежащей фон Нейману, согласно которой последовательность исполняемых инструкций (программа, соответствующая понятию "субъект") находится в единой среде с данными (соответствующими понятию "объект").

Модели, связанные с реализацией политики безопасности, как правило, не учитывают возможности субъектов по изменению состояния АС. Этот факт не является недостатком политик безопасности. Достоверность работы механизмов реализации политики безопасности считается априорно заданной, поскольку в противном случае невозможна формализация и анализ моделей. Однако вопрос гарантий политики безопасности является ключевым как в теории, так и в практике.

Рассматривая активную роль субъектов в АС необходимо упомянуть о ряде важнейших их свойств, на которых базируется излагаемая ниже модель. Во-первых,

человек-пользователь воспринимает объекты и получает информацию о состоянии АС через те субъекты, которыми он управляет и которые отображают информацию в воспринимаемом человеком виде. Во-вторых, угрозы компонентам АС исходят от субъекта как активного компонента, порождающего потоки и изменяющего состояние объектов в АС. В-третьих, субъекты могут влиять друг на друга через изменяемые ими объекты, связанные с другими субъектами, порождая в конечном итоге в системе субъекты (или состояния системы), которые представляют угрозу для безопасности информации или для работоспособности системы.

Будем считать разделение АС на субъекты и объекты априорным. Будем считать также, что существует априорный безошибочный критерий различения субъектов и объектов в АС (по свойству активности). Кроме того, предполагаем, что декомпозиция АС на субъекты и объекты фиксирована.

Подчеркнем отличие понятия субъекта АС от человека-пользователя следующим определением. Пользователь — лицо (физическое лицо), аутентифицируемое некоторой информацией и управляющее субъектам АС через органы управления компьютера. Пользователь АС является, таким образом, внешним фактором, управляющим состоянием субъектов. В связи с этим далее будем считать пользовательское управляющее воздействие таким, что свойства субъектов, сформулированные в ниже приводимых определениях, не зависят от него (т. е. свойства субъектов не изменяются внешним управлением). Смысл данного условия состоит в предположении того факта, что пользователь, управляющий программой, не может через органы управления изменить ее свойства (условие неверно для систем типа компиляторов, средств разработки, отладчиков и др.).

Будем также полагать, что в любой дискретный момент времени множество субъектов АС не пусто (в противной случае рассматриваются не соответствующие моменты

времени, а отрезки с ненулевой мощностью множества субъектов).

Аксиома 4. Субъекты в АС могут быть порождены из объектов только активным компонентом (субъектами).

Специфицируем механизм порождения новых субъектов следующим определением.

Определение 1. Объект O_i называется источником для субъекта S_m, если существует субъект S_j, в результате воздействия которого на объект O_i в АС возникает субъект S_m.

Субъект S_j, порождающий новый субъект из объекта O_i, называется активизирующим субъектом для субъекта S_m. S_m назовем порожденным объектом.

Введем обозначение: $Create(S_j, O_i) \rightarrow S_m$ — из объекта O_i порожден субъект S_m при активизирующем воздействии субъекта S_j. *Create* назовем операцией порождения субъектов:

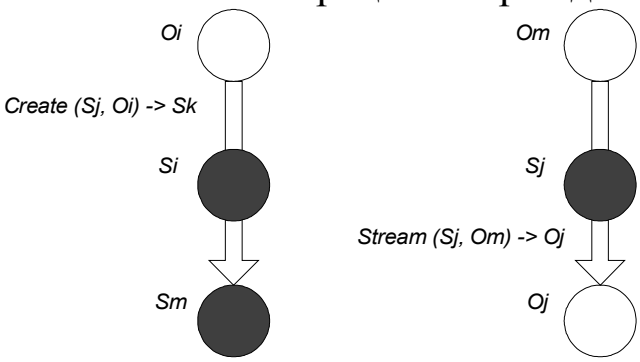


Рисунок Порождение субъекта и поток

Операция *Create* задает отображение декартова произведения множеств субъектов и объектов на объединение множества субъектов с пустым множеством. Заметим также, что в АС действует дискретное время и фактически новый субъект S_m порождается в момент времени t+1 относительно момента t, в который произошло воздействие порождающего субъекта на объект-источник. Очевидно, что операция порождения субъектов зависит как от свойств активизирующего субъекта, так и от содержания объекта-источника.

Считаем, что если $Create(S_j, O_i) \rightarrow Ø$, то порождение нового субъекта из объекта O_i, при активизирующем

воздействии S_j невозможно. Так, практически во всех операционных средах существует понятие исполняемого файла - объекта, способного быть источником порождения субъекта. Например, для ОС MS-DOS файл *winword.com* является объектом-источником порождения субъекта-программы текстового редактора, а порождающим субъектом является, как травило, командный интерпретатор (объект-источник — *command.com*).

Из архитектуры фон Неймана следует также, что с любым субъектом связан (или ассоциирован) некоторый объект (объекты), отображающий его состояние (например, для активной программы (субъекта) ассоциированным объектом будет содержание участка оперативной памяти с исполняемым кодом данной программы).

Определение 2. Объект O_i в момент времени t ассоциирован с субъектом S_m, если состояние объекта O_i повлияло на состояние субъекта в следующий момент времени (т. е. субъект S_m использует информацию, содержащуюся в объекте O_i).

Введем обозначение "множество объектов $\{O_m\}_t$ ассоциировано с субъектом S_i в момент времени t": $S_i(\{O_m\}_t)$. Данное определение не является в полной мере формально строгим, поскольку состояние субъекта описывается упорядоченной совокупностью ассоциированных с ним объектов, а ассоциированный объект выделяется по принципу влияния на состояние субъекта, т. е. в определении прослеживается некая рекурсия. С другой стороны, известны рекурсивные определения различных объектов (например, определение дерева). Зависимость от времени позволяет однозначно выделять ассоциированные объекты в том случае, если в начальный момент ассоциированный объект можно определить однозначно (как правило, это вектор исполняемого кода и начальные состояния некоторых переменных программы).

Субъект в общем случае реализует некоторое отображение множества ассоциированных объектов в момент

времени t на множество ассоциированных объектов в момент времени $t + 1$. В связи с этим можно выделить ассоциированные объекты, изменение которых изменяет вид отображения ассоциированных объектов (объекты, содержащие, как правило, код программы — функционально ассоциированные) и ассоциированные объекты-данные (являющиеся аргументом операции, но не изменяющие вида отображения). Далее под ассоциированными объектами понимаются функционально ассоциированные объекты, в иных случаях делаются уточнения.

Следствие (к определению 2). В момент порождения субъекта S_m из объекта O_i он является ассоциированным объектом для субъекта S_m.

Необходимо заметить, что объект-источник может быть ассоциированным для активизирующего субъекта, тогда порождение является автономным (т. е. не зависящим от свойств остальных субъектов и объектов). Если же объект-источник является неассоциированным (внешним) для активизирующего субъекта, то порождение не является автономным и зависит от свойств объекта-источника.

Свойство субъекта "быть активным" реализуется и в возможности выполнения действия над объектами. При этом отметим, что пассивный статус объекта необходимо требует существования потока информации от объекта к объекту (в противном случае невозможно говорить об изменении объектов), причем данный поток инициируется субъектом.

Определение 3. Потоком информации между объектом O_m и объектом O_j называется произвольная операция над объектом O_j, реализуемая в субъекте S_i и зависящая от O_m.

Заметим, что как O_j, так и O_m могут быть ассоциированными или неассоциированными объектами, а также "пустыми" объектами (Ø).

Обозначим: $Stream(S_i, O_m) \rightarrow O_j$ — поток информации от объекта O_m к объекту O_j (см. рисунок 5). При этом будем выделять источник (O_m) и получатель (приемник) потока (O_j). В определении подчеркнуто, что поток информации

рассматривается не между субъектом и объектом, а между объектами, например объектом и ассоциированными объектами субъекта (либо между двумя объектами). Активная роль субъекта выражается в реализации данного потока (это означает, что операция порождения потока локализована в субъекте и отображается состоянием его функционально ассоциированных объектов). Отметим, что операция *Stream* может создавать новый объект или уничтожать его.

На рисунке схематически изображены различные виды потоков. Далее для краткости будем говорить о потоке, подразумевая введенное понятие потока информации.

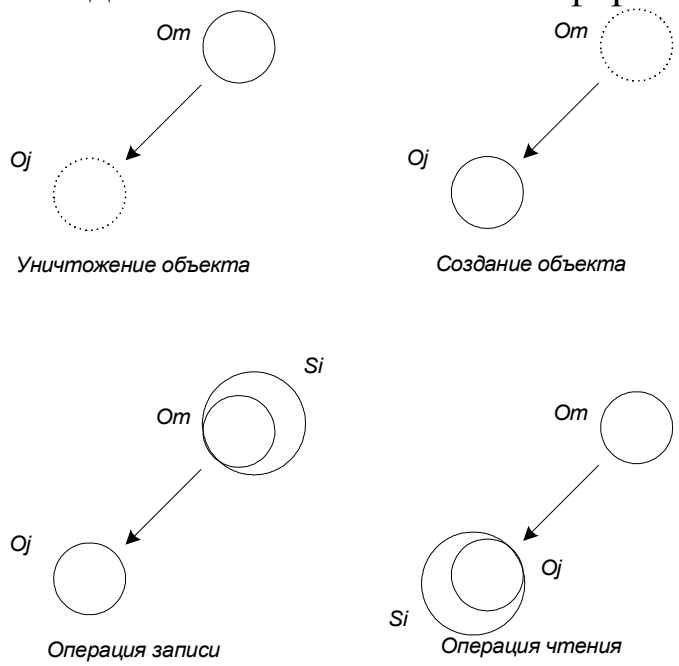


Рисунок Виды информационных потоков

Понятие ассоциированных с субъектом объектов, как легко видеть из изложенного выше, не является искусственной конструкцией. Корректно говорить о потоках информации можно лишь между одинаковыми сущностями, т. е. объектами. Кроме того, в ассоциированных объектах отображается текущее состояние субъекта. Отображениями *Stream* и *Create* описываются с точки зрения разделения на субъекты и объекты все события (изменения субъектов и объектов), происходящие в АС.

Из определения 3 следует также, что поток всегда инициируется (порождается) субъектом.

Определение 4. Доступом субъекта S_i к объекту O_j будем называть порождение потока информации между некоторым объектом (например, ассоциированным с субъектом объектами S_i ($\{O_m\}$)) и объектом O_j.

Выделим все множество потоков P для фиксированной декомпозиции АС на субъекты и объекты во все моменты времени (все множество потоков является объединением потоков по всем моментам дискретного времени) и произвольным образом разобьем его на два непересекающихся подмножества: N и L. Имеем

$P = N \cup L, N \cup L = 0.$

Обозначим: N — подмножество потоков, характеризующее несанкционированный доступ; L — подмножество потоков, характеризующих легальный доступ. Дадим некоторые пояснения к разделению на множества L и N. Понятие "безопасности" подразумевает наличие и некоторого состояния опасности — нежелательных состояний какой-либо системы (в данном случае АС). Будем считать парные категории типа "опасный — безопасный" априорно заданным для АС и описываемыми политикой безопасности, а результатом применения политики безопасности к АС — разделение всего множество потоков на множества "опасных" потоков N и множество "безопасных" L. Деление на L и N может описывать как свойство целостности (потоки из N нарушают целостность АС) или свойство конфиденциальности (потоки из N нарушают конфиденциальность АС), так и любое "другое произвольное свойство.

Определение 5. Правила разграничения доступа субъектов к объектам есть формально описанные потоки, принадлежащие подмножеству L.

В предлагаемой субъектно-ориентированной модели не уточняются известные модели политик безопасности (политика безопасности описывает только критерии

разбиения на множества L и N), но формулируются условия корректного существования элементов АС, обеспечивающих реализацию той или иной политики безопасности. Поскольку критерий разбиения на множества L и N не связан со следующими далее утверждениями (постулируется лишь наличие субъекта, реализующего фильтрацию потоков), то можно говорить об инвариантности субъектно-ориентированной модели относительно любой принятой в АС политики безопасности (не противоречащей условиям утверждений).

Определение 6. Объекты O_i и O_j тождественны в момент времени t, если они совпадают как слова, записанные в одном языке.

Например, при представлении в виде байтовых последовательностей объекты $O_1 = (o_{11}, o_{12}, \ldots, o_{1m})$ и $O_2 = (o_{21}, o_{22}, \ldots, o_{2k})$ одинаковы, если $m = k$ и $o_{1i} = o_{2i}$ для всех $i = 1, 2, \ldots, k$ (o_{ij} – байты).

Для введения понятия тождественности субъектов условимся о наличии процедуры сортировки ассоциированных объектов, которая позволяет говорить о возможности попарного сравнения. На практике всегда существует алгоритм, обеспечивающий возможность попарного сравнения и зависящий от конкретной архитектуры АС. Например, достаточно легко выделить и попарно сравнивать, например, участки оперативной памяти, отвечающие коду программ (отличающиеся абсолютным адресом загрузки в оперативную память) или содержанию переменных и массивов.

Определение 7. Субъекты S_i и S_j тождественны в момент времени t, если попарно тождественны все ассоциированные с ними объекты.

Следствие (из определений 6 и 7). Порожденные субъекты тождественны, если тождественны порождающие субъекты и объекты-источники.

Верность данного следствия вытекает из тождества функционально ассоциированных объектов в порождающих

субъектах, которые отвечают за порождение нового субъекта, а также из тождества аргументов (ассоциированных объектов-данных), которые отвечают объектам-источникам.

Для разделения всего множества потоков в АС на подмножества L и N необходимо существование активного компонента (субъекта), который:

- активизировался бы при возникновении любого потока;
- производил бы фильтрацию потоков в соответствии с принадлежностью множествам L или N.

Заметим, что если существуют $Stream(S_i, O_j) \rightarrow O_m$ и $Stream(S_k, O_m) \rightarrow O_l$, то существует и $Stream((S_i, S_k), O_j) \rightarrow O_l$, т. е. отношение "между объектами существует поток" является транзитивным (относительно пары субъектов). Именно в этом смысле будем говорить об участии субъекта (S_k) в потоке (если O_m является ассоциированным объектом субъекта, не тождественного S_i). Введем несколько определений.

Определение 8. Монитор обращений (МО) — субъект, активизирующийся при возникновении потока от любого субъекта к любому объекту.

Можно выделить два вида МО: индикаторный МО — устанавливающий только факт обращения субъекта к объекту; содержательный МО — субъект, функционирующий таким образом, что при возникновении потока от ассоциированного объекта O_m любого субъекта S_i к объекту O_j и обратно существует ассоциированный с МО объект O_{m0} (в данном случае речь идет об ассоциированных объектах-данных), тождественный объекту O_m или одному из $S_i(\{O_m\})$. Содержательный МО полностью участвует в потоке от субъекта к объекту (в том смысле, что информация проходит через его ассоциированные объекты-данные и существует тождественное отображение объекта, на какой-либо ассоциированный объект МО).

Теперь сформулируем понятие монитора безопасности (в литературе также применяется понятие монитора ссылок).

Это понятие связано с упоминаемой выше задачей фильтрации потоков. Поскольку целью является обеспечение безопасности АС, то и целевая функция монитора — фильтрация для обеспечения безопасности (отметим еще раз, что разделение на N и L задано априорно).

Определение 9. Монитор безопасности объектов (МБО) — монитор обращений, который разрешает поток, принадлежащий только множеству легального доступа L. Разрешение потока в данном случае понимается как выполнение операции над объектом-получателем потока, а запрещение — как невыполнение (т. е. неизменность объекта-получателя потока).

Монитор безопасности объектов фактически является механизмом реализации политики безопасности а АС.

Постулируя наличие в АС субъекта, реализующего политику безопасности рассмотрим основные политики безопасности.

Задание

1. Для заданной вычислительной системы произвести деление на субъекты и объекты по признаку активности (субъектов)
2. Для каждого субъекта выделить из множества объектов ассоциированные объекты
3. Выделить в системе специальные субъекты - Монитор безопасности объектов (МБО), Монитор безопасности субъектов (МБС) и субъект, контролирующий неизменность субъектов
4. Определить корректность субъектов относительно друг-друга, МБО и МБС
5. Выделить в множестве субъектов те, которые могут образовать Изолированную программную среду (ИПС)

Лабораторная работа 7
Работа с матрицей доступов. Домены безопасности.

Цель работы

Закрепление на практике материала по дискреционным политикам безопасности, создание матрицы доступов

Основные типы политики безопасности

Существуют два типа политики безопасности: дискреционная и мандатная.

Основой дискреционной (дискретной) политики безопасности является дискреционное управление доступом (Discretionary Access Control — DAC), которое определяется двумя свойствами:

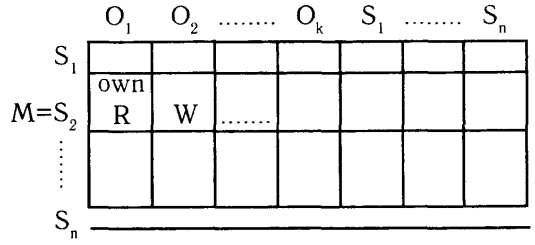
- все субъекты и объекты должны быть идентифицированы;
- права доступа субъекта к объекту системы определяются на основании некоторого внешнего по отношению к системе правила.

Термин «дискреционная политика» является дословным переводом Discretionary policy, еще одним вариантом перевода является следующий — разграничительная политика. Рассматриваемая политика — одна из самых распространенных в мире, в системах по умолчанию имеется ввиду именно эта политика.

Пусть *O* — множество объектов, *S* - множество субъектов, $S \subseteq O$. Пусть $U=\{U_1,...,U_m\}$ - множество пользователей. Определим отображение: own: 0◊ U.

В соответствии с этим отображением каждый объект объявляется собственностью соответствующего пользователя. Пользователь, являющийся собственником объекта, имеет все права доступа к нему, а иногда и право передавать часть или все права другим пользователям. Кроме того, собственник объекта определяет права доступа других субъектов к этому объекту, то есть политику безопасности в отношении этого объекта. Указанные права доступа записываются в виде матрицы доступа, элементы которой -

суть подмножества множества R, определяющие доступы субъекта S, к объекту 0_i(i = 1, 2,...,; j = 1, 2,...).



Существует несколько вариантов задания матрицы доступа.

1. Листы возможностей: Для каждого субъекта S_i создается лист (файл) всех объектов, к которому имеет доступ данный объект.

2. Листы контроля доступа: для каждого объекта создается список всех субъектов, имеющих право доступа к этому объекту.

Дискреционная политика связана с исходной моделью таким образом, что траектории процессов в вычислительной системе ограничиваются в каждом доступе. Причем вершины каждого графа разбиваются на классы и доступ в каждом классе определяется своими правилами каждым собственником. Множество неблагоприятных траекторий *N* для рассматриваемого класса политик определяется наличием неблагоприятных состояний, которые в свою очередь определяются запретами на некоторые дуги. Дискреционная политика, как самая распространенная, больше всего подвергалась исследованиям. Существует множество разновидностей этой политики. Однако многих проблем защиты эта политика решить не может. Одна из самых существенных слабостей этого j класса политик - то, что они не выдерживают атак при помощи "Троянского коня". Это означает, в частности, что система защиты, реализующая дискреционную политику, плохо защищает от проникновения вирусов в систему и других средств скрытого разрушающего воздействия. Покажем на примере принцип атаки "Троянским конем" в случае дискреционной политики.

Пример 1. Пусть U_1 - некоторый пользователь, а U_2 - пользователь-злоумышленник, O_1 - объект, содержащий ценную информацию, O_2 - программа с "Троянским конем" T, и M - матрица доступа, которая имеет вид:

	O_1	O_2
U_1	own r w	w
U_2		own r w

Проникновение программы происходит следующим образом. Злоумышленник U_2 создает программу O_2 и, являясь ее собственником, дает U_1 запускать ее и писать в объект O_2 информацию. После этого он инициирует каким-то образом, чтобы U_1 запустил эту программу (например, O_2 - представляет интересную компьютерную игру, которую он предлагает U_1 для развлечения). U_1 запускает O_2 и тем самым запускает скрытую программу T, которая обладая правами U_1 (т.к. была запущена пользователем U_1), списывает в себя информацию, содержащуюся в O_1. После этого хозяин U_2 объекта O_2, пользуясь всеми правами, имеет возможность считать из O_2 ценную информацию объекта O_1.

Следующая проблема дискреционной политики - это автоматическое определение прав. Так как объектов много, то задать заранее вручную перечень прав каждого субъекта на доступ к объекту невозможно. Поэтому матрица доступа различными способами агрегируется, например, оставляются в качестве субъектов только пользователи, а в соответствующую ячейку матрицы вставляются формулы функций, вычисление которых определяет права доступа субъекта, порожденного пользователем, к объекту O. Разумеется, эти функции могут изменяться во времени. В частности, возможно изъятие прав после выполнения

некоторого события. Возможны модификации, зависящие от других параметров.

Одна из важнейших проблем при использовании дискреционной политики - это проблема контроля распространения прав доступа. Чаще всего бывает, что владелец файла передает содержание файла другому пользователю и тот, тем самым, приобретает права собственника на информацию. Таким образом, права могут распространяться, и даже, если исходный владелец не хотел передавать доступ некоторому субъекту S к своей информации в O, то после нескольких шагов передача прав может состояться независимо от его воли. Возникает задача об условиях, при которых в такой системе некоторый субъект рано или поздно получит требуемый ему доступ. Эта задача исследовалась в модели "Take-Grant", когда форма передачи или взятия прав определяются в виде специального права доступа (вместо own). Некоторые результаты этих исследований будут приведены в главе "Математические методы анализа политики безопасности".

К достоинствам дискреционной политики безопасности можно отнести относительно простую реализацию соответствующих механизмов защиты. Этим обусловлен тот факт, что большинство распространенных в настоящее время АС обеспечивают выполнение положений именно данной политики безопасности.

В качестве примера реализаций дискреционной политики безопасности в АС можно привести матрицу доступов, строки которой соответствуют субъектам системы, а столбцы — объектам; элементы матрицы характеризуют права доступа. К недостаткам относится статичность модели. Это означает, что данная политика безопасности не учитывает динамику изменений состояния АС, не накладывает ограничений на состояния системы.

Кроме этого, при использовании дискреционной политики безопасности возникает вопрос определения правил распространения прав доступа и анализа их влияния

на безопасность АС. В общем случае при использовании данной политики безопасности перед МБО, который при санкционировании доступа субъекта к объекту руководствуется некоторым набором правил, стоит алгоритмически неразрешимая задача: проверить приведут ли его действия к нарушению безопасности или нет.

В то же время имеются модели АС, реализующих дискреционную политику безопасности (например, модель Take-Grant), которые предоставляют алгоритмы проверки безопасности.

Так или иначе, матрица доступов не является тем механизмом, который бы позволил реализовать ясную и четкую систему защиты информации в АС. Этим обуславливается поиск других более совершенных политик безопасности.

Основу мандатной (полномочной) политики безопасности составляет мандатное управление доступом (Mandatory Access Control — MAC), которое подразумевает, что:

- все субъекты и объекты системы должны быть однозначно идентифицированы;
- задан линейно упорядоченный набор меток секретности;
- каждому объекту системы присвоена метка секретности, определяющая ценность содержащейся в нем информации — его уровень секретности в АС;
- каждому субъекту системы присвоена метка секретности, определяющая уровень доверия к нему в АС — максимальное значение метки секретности объектов, к которым субъект имеет доступ; метка секретности субъекта называется его уровнем доступа.

Основная цель мандатной политики безопасности — предотвращение утечки информации от объектов с высоким уровнем доступа к объектам с низким уровнен доступа, т. е. противодействие возникновению в АС информационных каналов сверху вниз.

Чаще всего мандатную политику безопасности описывают в терминах, понятиях и определениях свойств модели Белла-ЛаПадула. В рамках данной модели доказывается важное утверждение, указывающее на принципиальное отличие систем, реализующих мандатную защиту, от систем с дискреционной защитой: если начальное состояние системы безопасно, и все переходы системы из состояния в состояние не нарушают ограничений, сформулированных политикой безопасности, то любое состояние системы безопасно.

Кроме того, по сравнению с АС, построенными на основе дискреционной политики безопасности, для систем, реализующих мандатную политику, характерна более высокая степень надежности. Это связано с тем, что МБО такой системы должен отслеживать не только правила доступа субъектов системы к объектам, но и состояния самой АС. Таким образом, каналы утечки в системах данного типа не заложены в нее непосредственно (что мы наблюдаем в положениях предыдущей политики безопасности), а могут появиться только при практической реализации системы вследствие ошибок разработчика. В дополнении к этому правила мандатной политики безопасности более ясны и просты для понимания разработчиками и пользователями АС, что также является фактором, положительно влияющим на уровень безопасности системы. С другой стороны, реализация систем с политикой безопасности данного типа довольно сложна и требует значительных ресурсов вычислительной системы.

Задание

Исследуемая система состоит из множества субъектов и объектов

Исходные данные:

СУБЪЕКТЫ

1. Пользователь1 (Администратор)
2. Пользователь 2

3. Пользователь 3
4. Текстовый редактор Word
5. Редактор формул
6. Модуль проверки правописания

ОБЪЕКТЫ

1. Документ пользователя 1
2. Документ пользователя 2
3. Документ пользователя 3
4. файл текстового редактора Word WINWORD.EXE
5. файл редактора формул EQUATION.DLL
6. файл модуля проверки правописания SPELL.DLL
7. файл-словарь DICTIONARY.DOC

Политика безопасности системы устанавливает такой порядок работы, при котором:

Пользователь 1 имеет возможность работы со своим документом с помощью программы WORD, может только просматривать документы пользователя 2 и 3, может проверять правописание в своем документе и вставлять в него формулы

Так же пользователь1 может добавлять новые слова в словарь.

Пользователь 2 имеет возможность работать только со своим документом, может проверять правописание, но не может добавлять новые слова в словарь и не может вставлять в документ формулы

Пользователь 3 имеет возможность работать со своим документов и документом пользователя 2, может проверять правописание в обоих документах, может добавлять в документы формулы, но не может добавлять новые слова в словарь.

Программа Word может быть запущена только пользователями системы, и может вызывать редактор формул и модуль проверки правописания.

Только модуль проверки правописания может изменять файл-словарь DICTIONARY.DOC

Задание

Составить множество возможных прав доступа в системе. Для заданного множества субъектов и объектов построить матрицу доступов и заполнить ее в соответствии заданной политикой безопасности и с принципом минимизации привилегий

Дополнить матрицу доступов временными доменами (например, добавить строку "Программа Word, запущенная от имени первого пользователя или "Редактор формул, запущенный третьим пользователем из программы Word")

Лабораторная работа 8
Модель Take-Grant

Цель работы

Закрепление теоретического материала по модели распространения прав доступа Take-Grant, применяемой для систем защиты, использующих дискреционное разграничение доступа.

Теория

В модели Take-Grant строится граф доступов субъектов к объектам, используются права доступа R={r,w,x,…,**t**,**g**}.

t - право брать права доступа, g - право передавать права доступа.

Так же в модели Take-Grant расматриваются правила преобразования графа доступов.

В базовой модели используется 4 правила преобразования графа доступов:

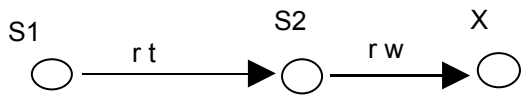
1. S take a for Y from X
2. S grant a for y to X
3. S create a for new object X
4. S remove a for X

Основная задача модели - показать что существует такая последовательность команд предобазования графа доступов, которая добавляет право a субъекту S, который раньше этого права не имел. В модели Take-Grant доказано несколько теорем, которые утверждают, что если сцбъекты S1 и S2 связаны tg-путем, и субъект S1 имеет право a на объект X, то и субъект S2 может получить это право.

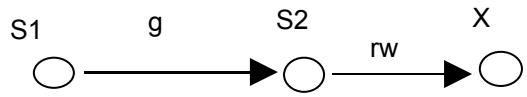
Задание

1. Для заданной задачи показать последовательность команд, которая дает право r субъекту S1 на объект X.

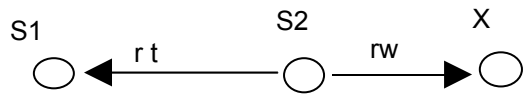
Задача 1



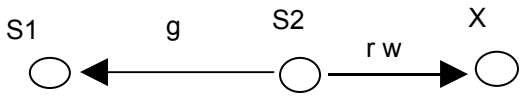
Задача 2



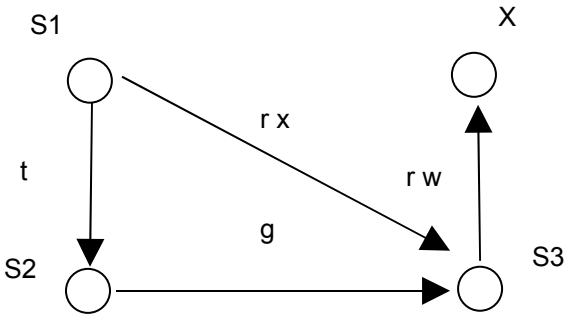
Задача 3



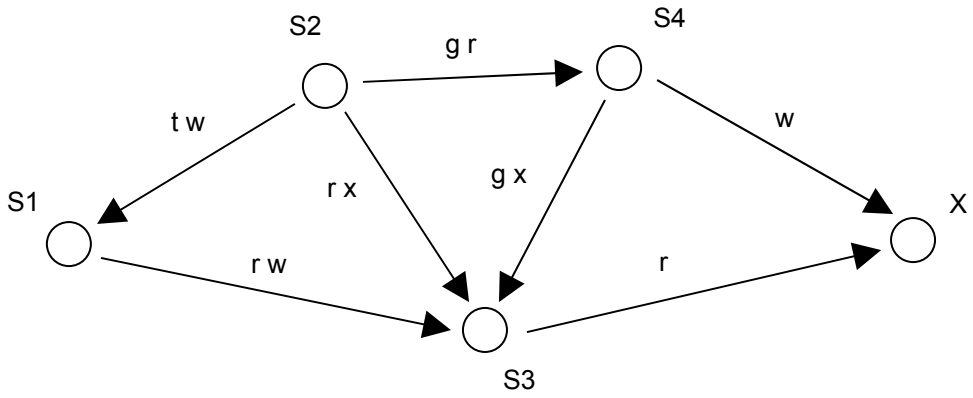
Задача 4



Задача 5



Задача 6



2. Написать программу, которая для заданной матрицы доступов (матрица смежности графа доступов) определяет возможность получения некоторого права субъектом на объект, если у этого субъекта такого права нет.

Лабораторная работа 9
Нарушение дискреционной политики безопасности Троянским конем

Цель работы

Закрепление теоретического материала по дискреционной политике безопасности.

Теория

В системах с дискреционным управлением доступом субъекты наделяются некоторым набором прав на объекты. Причем программа или процесс, запущенные от имени какого-то субъекта, получают все права этого субъекта на все объекты, к которым он имеет доступ. Эту особенность дискреционных политик безопасности могут использовать программы типа "Троянский конь". Такая программа составляется злоумышленником. Помимо каких-то явных, полезных действий, эта программа выполняет некоторые скрытые действия, нарушающие политику безопасности системы. Например, копирует данные пользователя, запустившего программу, в папку злоумышленника.

Задание

Выполняется группой студентов из двух человек. Один из них будет играть роль легального пользователя, а второй - роль злоумышленника.

1. Создать две папки и назначить к этим папкам доступ штатными средствами операционной системы в соответствии со следующей схемой:

Папка пользователя. Легальный пользователь имеет полный доступ к своей папке, включающий права чтения файлов. Злоумышленник не имеет доступа к папке пользователя, т.е. Не может открывать папку, просматривать ее содержимое, открывать файлы или запускать программы.

Папка злоумышленника. Легальный пользователь имеет набор прав доступа, необходимый для просмотра содержимого папки, создания файлов и папок, запуска программ. Удаление файлов легальным пользователем в

папке злоумышленника должно быть запрещено. Злоумышленник имеет полный доступ к своей папке.

2. Убедиться, что с помощью штатный средств операционной системы доступ злоумышленника в папку легального пользователя запрещен.

3. Написать программу (или пакетный файл), реализующую функции "Троянского коня", которая копирует файлы из папки легального пользователя в папку злоумышленника. Записать программу в папку злоумышленника.

4. Запустить программу-"Троянский конь" от имени злоумышленника и убедиться, что копирование файлов не происходит. Запустить программу-"Троянский конь" от имени легального пользователя и убедиться что произошло копирование файлов из папки легального пользователя в папку злоумышленника. Убедиться, что легальный пользователь не может удалить или изменить скопированные файлы. Убедиться, что злоумышленник имеет полный доступ к скопированным файлам.

5. Объяснить поэтапно механизм нарушения политики безопасности программой "Троянский конь" с подробным рассмотрением прав доступа пользователей и программы на каждом этапе.

6. Предложить пути усиления системы защиты, которые позволили бы системе с дискреционным распределением доступа противостоять программам типа "Троянский конь".

Лабораторная работа 10
Мандатные политики безопасности

Цель работы

Закрепление теоретического материала по мандатной политике безопасности.

Основные типы политики безопасности

Существуют два типа политики безопасности: дискреционная и мандатная.

Основой дискреционной (дискретной) политики безопасности является дискреционное управление доступом (Discretionary Access Control — DAC), которое определяется двумя свойствами:

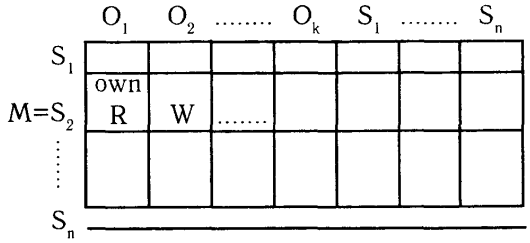
- все субъекты и объекты должны быть идентифицированы;
- права доступа субъекта к объекту системы определяются на основании некоторого внешнего по отношению к системе правила.

Термин «дискреционная политика» является дословным переводом Discretionary policy, еще одним вариантом перевода является следующий — разграничительная политика. Рассматриваемая политика — одна из самых распространенных в мире, в системах по умолчанию имеется ввиду именно эта политика.

Пусть O — множество объектов, S - множество субъектов, $S \subseteq O$. Пусть $U=\{U_1,...,U_m\}$ - множество пользователей. Определим отображение: own: 0◊ U.

В соответствии с этим отображением каждый объект объявляется собственностью соответствующего пользователя. Пользователь, являющийся собственником объекта, имеет все права доступа к нему, а иногда и право передавать часть или все права другим пользователям. Кроме того, собственник объекта определяет права доступа других субъектов к этому объекту, то есть политику безопасности в отношении этого объекта. Указанные права доступа

записываются в виде матрицы доступа, элементы которой - суть подмножества множества R, определяющие доступы субъекта S, к объекту 0_i(i = 1, 2,...,; j = 1, 2,...).



Существует несколько вариантов задания матрицы доступа.

1. Листы возможностей: Для каждого субъекта S_i создается лист (файл) всех объектов, к которому имеет доступ данный объект.

2. Листы контроля доступа: для каждого объекта создается список всех субъектов, имеющих право доступа к этому объекту.

Дискреционная политика связана с исходной моделью таким образом, что траектории процессов в вычислительной системе ограничиваются в каждом доступе. Причем вершины каждого графа разбиваются на классы и доступ в каждом классе определяется своими правилами каждым собственником. Множество неблагоприятных траекторий *N* для рассматриваемого класса политик определяется наличием неблагоприятных состояний, которые в свою очередь определяются запретами на некоторые дуги. Дискреционная политика, как самая распространенная, больше всего подвергалась исследованиям. Существует множество разновидностей этой политики. Однако многих проблем защиты эта политика решить не может. Одна из самых существенных слабостей этого j класса политик - то, что они не выдерживают атак при помощи "Троянского коня". Это означает, в частности, что система защиты, реализующая дискреционную политику, плохо защищает от проникновения вирусов в систему и других средств скрытого разрушающего воздействия. Покажем на примере принцип атаки "Троянским конем" в случае дискреционной политики.

Пример 1. Пусть U_1 - некоторый пользователь, а U_2 - пользователь-злоумышленник, O_1 - объект, содержащий ценную информацию, O_2 - программа с "Троянским конем" T, и M - матрица доступа, которая имеет вид:

	O_1	O_2
U_1	own r w	w
U_2		own r w

Проникновение программы происходит следующим образом. Злоумышленник U_2 создает программу O_2 и, являясь ее собственником, дает U_1 запускать ее и писать в объект O_2 информацию. После этого он инициирует каким-то образом, чтобы U_1 запустил эту программу (например, O_2 - представляет интересную компьютерную игру, которую он предлагает U_1 для развлечения). U_1 запускает O_2 и тем самым запускает скрытую программу T, которая обладая правами U_1 (т.к. была запущена пользователем U_1), списывает в себя информацию, содержащуюся в O_1. После этого хозяин U_2 объекта O_2, пользуясь всеми правами, имеет возможность считать из O_2 ценную информацию объекта O_1.

Следующая проблема дискреционной политики - это автоматическое определение прав. Так как объектов много, то задать заранее вручную перечень прав каждого субъекта на доступ к объекту невозможно. Поэтому матрица доступа различными способами агрегируется, например, оставляются в качестве субъектов только пользователи, а в соответствующую ячейку матрицы вставляются формулы функций, вычисление которых определяет права доступа субъекта, порожденного пользователем, к объекту O. Разумеется, эти функции могут изменяться во времени. В частности, возможно изъятие прав после выполнения

некоторого события. Возможны модификации, зависящие от других параметров.

Одна из важнейших проблем при использовании дискреционной политики - это проблема контроля распространения прав доступа. Чаще всего бывает, что владелец файла передает содержание файла другому пользователю и тот, тем самым, приобретает права собственника на информацию. Таким образом, права могут распространяться, и даже, если исходный владелец не хотел передавать доступ некоторому субъекту S к своей информации в O, то после нескольких шагов передача прав может состояться независимо от его воли. Возникает задача об условиях, при которых в такой системе некоторый субъект рано или поздно получит требуемый ему доступ. Эта задача исследовалась в модели "Take-Grant", когда форма передачи или взятия прав определяются в виде специального права доступа (вместо own). Некоторые результаты этих исследований будут приведены в главе "Математические методы анализа политики безопасности".

К достоинствам дискреционной политики безопасности можно отнести относительно простую реализацию соответствующих механизмов защиты. Этим обусловлен тот факт, что большинство распространенных в настоящее время АС обеспечивают выполнение положений именно данной политики безопасности.

В качестве примера реализаций дискреционной политики безопасности в АС можно привести матрицу доступов, строки которой соответствуют субъектам системы, а столбцы — объектам; элементы матрицы характеризуют права доступа. К недостаткам относится статичность модели. Это означает, что данная политика безопасности не учитывает динамику изменений состояния АС, не накладывает ограничений на состояния системы.

Кроме этого, при использовании дискреционной политики безопасности возникает вопрос определения правил распространения прав доступа и анализа их влияния

на безопасность АС. В общем случае при использовании данной политики безопасности перед МБО, который при санкционировании доступа субъекта к объекту руководствуется некоторым набором правил, стоит алгоритмически неразрешимая задача: проверить приведут ли его действия к нарушению безопасности или нет.

В то же время имеются модели АС, реализующих дискреционную политику безопасности (например, модель Take-Grant), которые предоставляют алгоритмы проверки безопасности.

Так или иначе, матрица доступов не является тем механизмом, который бы позволил реализовать ясную и четкую систему защиты информации в АС. Этим обуславливается поиск других более совершенных политик безопасности.

Основу мандатной (полномочной) политики безопасности составляет мандатное управление доступом (Mandatory Access Control — MAC), которое подразумевает, что:

- все субъекты и объекты системы должны быть однозначно идентифицированы;
- задан линейно упорядоченный набор меток секретности;
- каждому объекту системы присвоена метка секретности, определяющая ценность содержащейся в нем информации — его уровень секретности в АС;
- каждому субъекту системы присвоена метка секретности, определяющая уровень доверия к нему в АС — максимальное значение метки секретности объектов, к которым субъект имеет доступ; метка секретности субъекта называется его уровнем доступа.

Основная цель мандатной политики безопасности — предотвращение утечки информации от объектов с высоким уровнем доступа к объектам с низким уровнен доступа, т. е. противодействие возникновению в АС информационных каналов сверху вниз.

Чаще всего мандатную политику безопасности описывают в терминах, понятиях и определениях свойств модели Белла-ЛаПадула. В рамках данной модели доказывается важное утверждение, указывающее на принципиальное отличие систем, реализующих мандатную защиту, от систем с дискреционной защитой: если начальное состояние системы безопасно, и все переходы системы из состояния в состояние не нарушают ограничений, сформулированных политикой безопасности, то любое состояние системы безопасно.

Кроме того, по сравнению с АС, построенными на основе дискреционной политики безопасности, для систем, реализующих мандатную политику, характерна более высокая степень надежности. Это связано с тем, что МБО такой системы должен отслеживать не только правила доступа субъектов системы к объектам, но и состояния самой АС. Таким образом, каналы утечки в системах данного типа не заложены в нее непосредственно (что мы наблюдаем в положениях предыдущей политики безопасности), а могут появиться только при практической реализации системы вследствие ошибок разработчика. В дополнении к этому правила мандатной политики безопасности более ясны и просты для понимания разработчиками и пользователями АС, что также является фактором, положительно влияющим на уровень безопасности системы. С другой стороны, реализация систем с политикой безопасности данного типа довольно сложна и требует значительных ресурсов вычислительной системы.

Задание

Задача1: Заданы документы с различным уровнем секретности, заданы пользователи с различным уровнем доступа (список документов и пользователей и их уровни доступа/секретности составить самостоятельно. Не менее 5 пользователей и 5 документов).

1. Для каждого пользователя составить список документов, доступных ему для работы при условии, что пользователь не понижает своего уровня допуска.

2. Для одного из пользователей составить список документов, доступных ему для работы при условии, что пользователь может понизить свой уровень доступа на один уровень.

3. Один из пользователей имеет возможность работать с несколькими документами. На основе этих документов он создает новый документ. Какой гриф секретности нужно присвоить этому документу.

4. Показать на примере одного из пользователей, что мандатная политика безопасности не может быть нарушено программой типа "Троянский конь".

Задача 2: Заданы субъекты и объекты с различными уровнями допуска/секретности. Задана матрица доступов для первоначального состояния системы z0 (см. ниже).

Показать, что первоначальное состояние z0 является безопасным состоянием - т.е. удовлетворяет трем свойствам безопасности - ss-свойству, *-свойству и ds-свойству (при учловии, что fc=fs и осуществляются только доступы, описанные в матрице доступов).

Если условия безопасности не выполняются, то показать пути изменения исходного задания (понижение уровня доступа пользователей или изменение матрицы доступов), которые позволят выполнить условия безопасности)

S={s1,s2,s3,s4,s5}

O={o1,o2,o3,o4,o5}

L={1-низкий, 2-средний, 3-высокий, 4-максимальный}

S	s1	s2	s3	s4	s5
fs	1	4	3	2	3

O	o1	o2	o3	o4	o5
fo	1	3	2	4	4

M	o1	o2	o3	o4	o5
s1	re				
s2		ra	ea		
s3		rw	wa	we	
s4				rwe	
s5					rwea

Лабораторная работа 11
Стандарты в области защиты информации в компьютерных системах

Задание

Подготовить доклады на следующие темы:

1. История (хронология) разработки и создания стандартов в области защиты информации в компьютерных системах.
2. Сравнение стандартов: Руководящие документы ГТК и TCSEC.
3. Сравнение стандартов: Руководящие документы ГТК и Единые критерии безопасности информационных технологий.
4. Пример профиля защиты некоторой системы. (Посмотреть на сайте www.fstec.ru в разделе "Материалы, предназначенные для предприятий и организаций, получивших лицензии ФСТЭК России")