В.В. Новый

Операционные системы. Безопасность и механизмы защиты ОС

Лекционные материалы по дисциплине «Операционные системы» для студентов 2 курса специальности Прикладная математика (1-31 03 03)

2010

Лекция. Безопасность и механизмы защиты ОС

Содержание: Понятие безопасности. Угрозы для безопасности. Аутентификация и авторизация пользователей. Проблемы безопасности. Механизмы защиты. Списки контроля доступа. Криптографическая защита данных.

С ростом объема информации, хранящейся в компьютерных системах возрастает важность ***обеспечения ее защиты***.

Безопасность – общая проблема, связанная с гарантированием того, что файлы не читаются и не модифицируются неавторизованными лицами.
Механизмы защиты – специфические механизмы ОС, используемые для обеспечения информационной безопасности в компьютерных системах.

Безопасность обеспечивается путем ***контроля доступа к ресурсам*** – разрешением доступа к ресурсу только ***авторизованным*** пользователям этого ресурса.

Авторизация – назначение ***прав*** и ***привилегии*** в системе пользователю, согласно которым определяется что он может делать в системе;
Аутентификация – установление подлинности пользователя (определение, не выдает ли пользователь системы себя за кого-то другого).

Право – возможность выполнять некоторые операции над объектами системы.
Привилегия – более общее понятие, означающее возможность выполнять действия в отношении других объектов и субъектов системы безопасности. Привилегия – атрибут субъекта, а не объекта.

Формально, все ресурсы могут быть разбиты на 2 категории:
– **Объекты** – ресурсы, которые требуется защитить (файлы, память и т.д.);
– **Субъекты** – активные ресурсы, которые выполняют операции над объектами (процессы, потоки) – ***осуществляют доступ субъектов к объектам***.

Для каждой пары (субъект, объект) определяется множество операции, которые субъект может выполнить над объектом.

Существует субъект, который контролирует доступ других субъектов к объектам руководствуясь некоторыми правилами – **политикой безопасности**.

Субъект реализующий политику безопасности называется **менеджером** или **монитором безопасности**. Не следует путать менеджер безопасности - программный компонент и ***администратора компьютерной системы*** – человека, который выполняет регистрацию пользователей, назначение им прав и привилегии, надзор за результатами выполнения политики безопасности.

На основании политики безопасности строится ***модель безопасности*** которая включает:

- **объекты;**
- наборы **операции** над объектами;
- **субъекты;**
- **атрибуты защиты** объектов (описывают права доступа субъектов к объектам).

Это является основой для построения ***матрицы доступа***.

Дискреционная политика безопасности:

- Для каждого объекта определяется набор операций, которые можно над ним выполнять;
- Субъект может выполнить операцию, если он имеет право на выполнение этой операции;
- Субъект, который имеет права на выполнение некоторых операции над объектом, может передать эти права другому субъекту.
  - Без контроля менеджера безопасности – либеральная дискреционная политика;
  - Только при наличии специальных полномочии – строгая дискреционная политика.

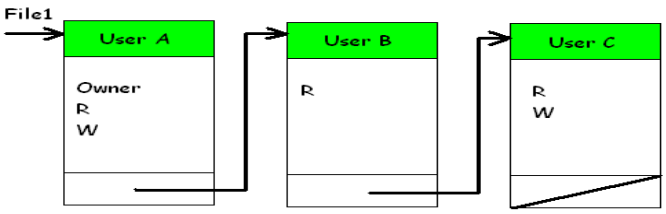
Матрица доступа

	Файл 1	Файл 2	Файл 3	Файл 4	Файл 5
User A	Owner RW			Owner RW	RW
User B	R	Owner RW	RX	W	
Admin	RW CP	CP	Owner RWX C	C	

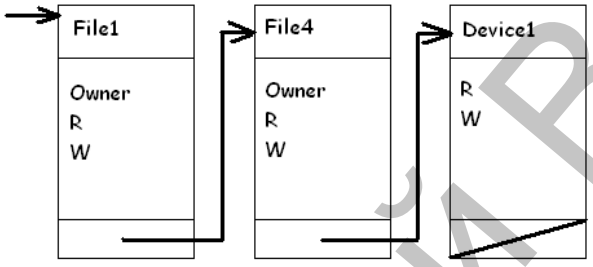
Одно измерение матрицы – идентификаторы субъектов, которым требуется получить доступ (пользователи, группы пользователей, узлы и т.д.). Второе измерение – список объектов, к которым предоставляется доступ. В каждом элементе матрицы указаны права доступа данного субъекта к данному объекту (R – READ, W – WRITE, X – EXECUTE, C – CONTROL, CP – CONTROL_WITH_PASSING_ABILITY).

В виду разреженности матрицы она применяется на практике редко. Обычно используется один из следующих способов.

- разложение матрицы по столбцам – **списки контроля доступа** (**access control list или ACL**);



– разложение матрицы по строкам – **мандаты возможностей** (**capability tickets**).



Так как субъекты могут передавать права управления объектами другим субъектам, то должны быть определены правила, которым подчиняются субъекты при передаче прав. Набор таких правил определяет ***модель управления*** в дискреционной модели безопасности. Существуют 4 модели управления, которые могут использоваться в дискреционной модели безопасности:

– иерархическое управление (hierarchical control);
– управление правами доступа владельцем объекта (concept of ownership);
– либеральное управление (laissez-fair);
– централизованное управление (centralized control).

Помимо дискреционной модели безопасности используются многоуровневые модели безопасности. Наиболее широкое распространение получила **модель Белла – Ла Падулы**. Она была разработана для обеспечения военной системы безопасности, но также применима и в мирных целях ☺

Модель Белла-Ла Падулы (гарантия защищенности):

– ***Простое свойство секретности*** (Процесс, работающий на любом уровне секретности, может читать только объекты своего или более низкого уровня)
– ***Свойство **.*** (Процесс, работающий на любом уровне секретности, может писать только в объекты своего или более высокого уровня секретности).

Недостаток модели Белла-Ла Падулы состоит в том, что она была разработана для хранения секретов, а не для обеспечения целостности данных. Для обеспечения целостности данных понадобиться модель с

противоположными свойствами – модель Биба:

Модель целостности (модель Биба)

- ***Простой принцип целостности.*** (Процесс, работающий на любом уровне секретности, может писать только в объекты своего или более низкого уровня)
- ***Свойство целостности **.*** (Процесс, работающий на любом уровне секретности, может читать только объекты своего или более высокого уровня секретности).

Принципы проектирования систем безопасности

1. устройство системы не должно быть секретом.
2. по умолчанию доступ не должен предоставляться
3. необходимо проверять текущее состояние прав доступа
4. предоставляйте каждому процессу как можно меньше привилегий
5. механизм защиты должен быть простым, одинаковым для всех и встроенным в самые нижние уровни системы
6. выбранная схема должна быть психологически приемлемой.

Угрозы безопасности

Можно выделить **3** основных **аспекта** обеспечения **безопасности** компьютерной системы:

- природа угроз;
- природа злоумышленников;
- случайная потеря данных.

Рассмотрим природу угроз компьютерным системам. Выделяют 3 основных требования к безопасности компьютерной системы, известные как триада CIA (**C**onfidentiality, **I**ntegrity, **A**vailability):

- **конфиденциальность**;
- **целостность** данных и системы;
- **доступность** системы.

Соответсвенно этим требованиям, выделяют 3 группы угроз безопасности. Согласно терминологии, принятой в RFC 2828:

- для ***конфиденциальности*** данных – несанкционированная демонстрация данных (Unauthorized disclosure), включающая разглашение (exposure), перехват (interception), извлечение содержимого (inference) – типичный пример – анализ трафика, вторжение (intrusion);
- для **целостности** данных и системы – порча и подделка данных (deception), включающая в себя выдачу себя за другое лицо (masquerade), подделку данных (falsification), препятствование передаче, приему или хранению валидных данных (repudiation), добавление нежелательных операции в систему путем модификации

ее функции и данных (corruption);
- для **доступности** системы – разрушение (disruption) и незаконное присвоение (usurpation), включая прерывание или предотвращение работы системных операции отключением компонентов системы (incapacitation), неавторизованный доступ и логический или физический контроль ресурсов системы (misappropriation), выполнение функции или действий в обход системы безопасности (misuse).

Атаки принято делить на:
- пассивные – направленные на перехват информации без ее модификации, например, извлечение данных или анализ трафика;
- активные – направленные на модификацию данных, например, на подмену передаваемых по сети данных, или на вывод системы из строя.

Человека, выполняющего атаку на компьютерную систему в литературе по безопасности принято называть **злоумышленником** или **неприятелем.**

Также разделяются на:
- пассивные злоумышленники. Просто пытаются прочитать файлы, которые им не разрешено читать;
- активные злоумышленники. Пытаются незаконно изменить данные.

Категории злоумышленников:
- случайные любопытные пользователи, не применяющие специальных технических средств;
- члены самой организации, занимающиеся шпионажем («инсайдеры»);
- лица преследующие цели личного обогащения;
- занимающиеся коммерческим и военным шпионажем.

Случайная потеря данных:
1. Форс-мажор: пожары, наводнения, землетрясения, войны, восстания, крысы, изгрызшие провода или диски.
2. Аппаратные и программные ошибки: сбои центрального процессора, нечитаемые диски, ошибки при передаче данных, ошибки в программах.
3. Человеческий фактор: неправильный ввод данных, неверный установленный диск, запуск не той программы, потерянный диск и т.д.

Меры по контролю доступа можно разделить на две категории:
- методы контроля доступа, ориентированные на пользователя;
- методы контроля доступа, ориентированные на данные.

Наиболее распространенный метод контроля доступа, ориентированный на пользователя в совместно используемой среде – процедура регистрации, цель

которой определить идентичность пользователя.

Методы аутентификации

– **распознавание чего-то, известного пользователю** (пароль, персональный идентификационный номер (Personal Indentification Number – PIN), ответы на предложенные вопросы и т.д.);
– **распознавание чего-то, чем владеет пользователь** (электронные карты, смарт-карты, физические ключи и т.д.). Обычно этот предмет называется ключом или токеном (token);
– **распознавание чего-то, чем является пользователь** (статическая биометрическая идентификация – отпечатки пальцев, сканирование сетчатки, распознование лиц и т.д.);
– **распознование чего-то, что умеет пользователь** (динамическая биометрическая идентификация – распознавание голоса, рукописного ввода, ритма печати и т.д.).

Наиболее распространенным методом аутентификации является **Аутентификация с использованием паролей**. Пользователю назначается не только имя или идентификатор (ID), но и пароль. Процедура аутентификации заключается в проверке пары (имя, пароль) с хранящимися в системе.

Обладает низкой защищенностью против взлома методом «грубой силы» (brute force).

Методы защиты паролей

– использование шифрования паролей (**hashed passwords**);
– использование «соли» (**salt value**);
– Разрешение на доступ к файлу паролей только через специальную процедуру.

Требования к паролям

– Пароль должен содержать как минимум семь символов.
– Пароль должен содержать как строчные, так и прописные символы
– Пароль должен содержать как минимум одну цифру или специальный символ
– Пароль не должен представлять собой слово, содержащееся в словаре, имя собственное и т. д.

Улучшенные варианты парольной аутентификации

– Одноразовые пароли (схема Лесли Лампорта)
– Система аутентификации «пароль-отзыв»

Аутентификация с использованием физического объекта

– электронные карты (сохраняют информацию, но не могут ее

обрабатывать); чаще всего используются совместно с какой-либо разновидностью пароля или персонального идентификационного номера;
- smart-карты (кроме хранения информации умеют ее обрабатывать – содержат микропроцессор).

Аутентификация с использованием биометрических данных

Этот метод аутентификации основан на измерении физических характеристик пользователя, которые трудно подделать. Такие характеристики называются ***биометрическими параметрами***.

Биометрические параметры

- Отпечатки пальцев
- Рисунок сетчатки
- Анализ подписи
- Измерение характеристик голоса и т.д.

Контрмеры против злоумышенников

- Ограничение времени регистрации
- Задержка регистрации
- Число повторных попыток
- Ловушки для взломщика

Рассмотрим возможные атаки на компьютерные системы.

Атаки изнутри

- **троянские кони** (**Trojan horse**) – невинные с виду программы, содержащие кроме полезных функции процедуры, выполняющие нежелательные действия);
- **фальшивая программа регистрации**;
- **логические бомбы** (**logic bomb**) – код, добавленный в программу злоумышленником и неактивный до наступления какого-либо события;
- **потайные двери** (**backdoor**, **trapdoor**) – секретный вход в программу, который позволяет тому, кто о нем осведомлен миновать процедуры безопасности;
- **переполнение буфера** (**buffer overflow**);
- **кейлоггеры** (**keyloggers**) – программы, захватывающие ввод пользователя на скомпрометированных системах;
- **руткиты** (**rootkit**) – набор программных инструментов, которые используются злоумышленником после проникновения в компьютерную систему и получения административных полномочии для сокрытия факта вторжения и присутствия;
- **spyware** – программы, которые собирают информацию на компьютере и передают ее другой системе;

– **adware** – реклама, встроенная в программное обеспечение, которая отображает всплывающие окна или перенаправляет пользователя на другой сайт.

Атаки системы снаружи

– компьютерные **вирусы** (**computer virus**);
– **сетевые черви** (**worm**) – саморазмножающиеся программы, основанные на уязвимостях ПО, которые могут запускаться независимо и могут создавать свою полностью рабочую копию на других хостах в сети;
– **зомби** (**zombie, bot**) – программа активирующаяся на зараженной машине, которая используется для атаки на другие машины;
– **мобильный код** (**mobile code**) – программное обеспечение (аплеты, агенты, скрипты и другое активное содержимое), которое может быть в неизменном виде запущено на различных платформах с одинаковой семантикой;
– **эксплойты** (**exploits**) – код, специфический для определенной уязвимости или набора уязвимостей;
– **downloaders** – программы, которые инсталлируют другие элементы на атакуемую машину. Чаще всего распространяются по e-mail или web.

Один из наиболее распространенных классов вредоносного ПО – компьютерные вирусы.

Компьютерный вирус имеет 3 части:
1. механизм заражения;
2. триггер – событие или условие при котором активируется нагрузка;
3. нагрузка – то, что вирус делает – причинение ущерба или какое-либо другое действие.

Фазы развития вируса
– фаза покоя – вирус бездействует;
– фаза размножения – вирус заражает программы на инфицированном компьютере;
– фаза запуска – срабатывает заложенное условие или событие и вирус выполняет нагрузку;
– фаза выполнения – вирус выполняется и причиняет ущерб компьютерной системе.

Разновидности вирусов
1. загрузочные вирусы – заражают MBR или BR и выполняются, когда производится загрузка с зараженного носителя;
2. вирусы-компаньоны;
3. вирусы заражающие исполняемые файлы:
   – перезаписывающие;
   – паразитические;
   – полостные;

4. резидентные вирусы;
5. вирусы драйверов устройств;
6. макровирусы;
7. вирусы, заражающие исходные тексты программ;

Стандарты безопасности ОС

1985 г. **«Оранжевая книга»** Министерства обороны США (стандарт DoD 5200.28 Trusted Computer System Evaluation Criteria, TCSEC) – в настоящий момент устаревший – определял 7 категорий ОС по безопасности. Хотя стандарт и был заменен более сложным, он до сих пор представляет хорошее руководство в области безопасности систем.

A1 Verified Desing
B3 Security Domains
B2 Structured Protection
B1 Labeled Security Protection
C2 Controlled Access Protection
C1 Discretionary Access Protection (устарел)
D Minimal Protection

Windows NT 4 была сертифицирована на уровень безопасности C2.

В настоящее время международным стандартом является стандарт **Common Criteria**

Common Criteria

В 1999 проект Common Criteria был ратифицирован как стандарт. Первоначально Канадой, Францией, Германией и США. Позднее к ним присоединились Австралия, Финляндия, Греция, Израиль, Италия, Новая Зеландия, Норвегия, Испания и Нидерланды.

Common Criteria for Information Technology Security Evaluation (CCITSE), также известна как Common Criteria (ISO 15408 версия 2.1)

В 2008 году Windows XP SP2 и x64 SP2, Windows 2003 Server SP2 R2 были сертифицированы на уровень EAL4+ сертификат 4 уровня Evaluation Assurance Level4 (+ - систематическое устранение недостатков).

Криптографическая защита данных

Во многих случаях система безопасности для надежности дополняется шифрованием данных.

Проблемой защиты информации путем ее преобразования занимается **криптология** (kryptos - тайный, logos - наука). Криптология разделяется на два направления:

– **криптографию**
– **криптоанализ.**

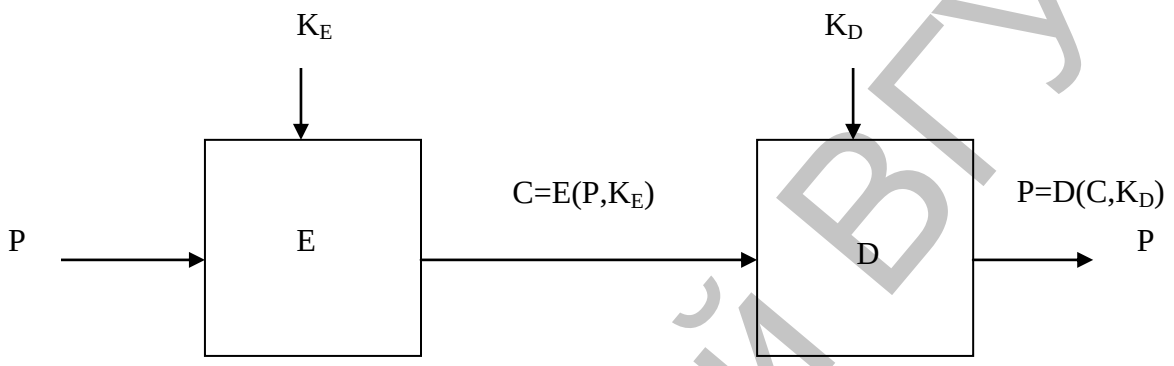
Цели этих направлений прямо противоположны.

Криптография занимается поиском и исследованием математических методов преобразования информации.

Сфера интересов **криптоанализа** – исследование возможности

расшифровывания информации без знания ключей.

Задача криптографии: взять сообщение или файл, называемый **открытым текстом**, и преобразовать его в **зашифрованный текст** таким образом, чтобы только посвященные могли преобразовать его обратно в открытый текст.
Общая схема выглядит следующим образом:



Здесь:

P – открытый (незашифрованный текст);
K_E – ключ шифрования;
K_D – ключ дешифрования;
C – зашифрованный текст.

Зашифрованный текст получается как результат процесса шифрования открытого текста P с помощью ключа K_E алгоритмом E.

Для расшифровки зашифрованного текста применяется алгоритм дешифрования D к зашифрованному тексту C и ключу дешфрования K_D: P = D(C, K_D).

Современная криптография включает 4 крупных раздела:

- **симметричные** криптосистемы;
- криптосистемы **с открытым ключом**;
- системы **электронной подписи**;
- **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.

Криптографическая система представляет собой семейство T 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:

- **бесключевые** (не используют каких-либо ключей в процессе шифрования);
- **симметричные** (шифрование с секретным (закрытым) ключом);
- **с открытым ключом** (шифрование с публичным ключом).

В **симметричных криптосистемах** и для шифрования, и для дешифрования используется один и тот же ключ (или ключ расшифровывания легко вычисляется из ключа зашифровывания).

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.

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.

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.

Симметричные криптосистемы

Симметричное шифрование бывает двух видов:

- блочное (информация разбивается на блоки фиксированной длины, после чего эти блоки поочередно шифруются);
- потоковое (шифруют данные побитно или посимвольно).

Все многообразие существующих криптографических методов можно свести к следующим классам преобразований:

– **моно- и многоалфавитные подстановки**

Наиболее простой вид преобразований, заключающийся в замене символов исходного текста на другие (того же алфавита) по более или менее сложному правилу. Для обеспечения высокой криптостойкости требуется использование больших ключей.

– **перестановки**

Несложный метод криптографического преобразования. Используется как правило в сочетании с другими методами.

– **гаммирование**

Этот метод заключается в наложении на исходный текст некоторой псевдослучайной последовательности, генерируемой на основе ключа.

– **блочные шифры**

Представляют собой последовательность (с возможным повторением и чередованием) основных методов преобразования, применяемую к блоку (части) шифруемого текста. Блочные шифры на практике встречаются чаще, чем "чистые" преобразования того или иного класса в силу их более высокой криптостойкости. Наиболее известные представители – американский DES (1980г.), AES (он же Rijndael, 1997г), ГОСТ 28147-89 (РФ), IDEA.

Слабое место симметричных криптосистем – распределение ключей. Для того, чтобы был возможен обмен конфиденциальной информацией между двумя субъектами, ключ должен быть сгенерирован одним из них, а затем каким-то образом опять же в конфиденциальном порядке передан другому. Т.е. в общем случае для передачи ключа опять же требуется использование какой-то криптосистемы.

Для решения этой проблемы на основе результатов, полученных классической и современной алгеброй, были предложены системы с **открытым ключом.**

Суть их состоит в том, что каждым адресатом генерируются два ключа, связанные между собой по определенному правилу. Один ключ объявляется **открытым**, а другой **закрытым**. Открытый ключ публикуется и доступен любому, кто желает послать сообщение адресату. Секретный ключ сохраняется в тайне.

Исходный текст шифруется открытым ключом адресата и передается ему. Зашифрованный текст в принципе не может быть расшифрован тем же открытым ключом. Дешифрование сообщение возможно только с использованием закрытого ключа, который известен только самому адресату.

Криптографические системы с открытым ключом используют так называемые ***необратимые*** или ***односторонние функции***, которые обладают следующим свойством: при заданном значении x относительно просто вычислить значение f(x), однако если y=f(x), то нет простого пути для вычисления значения x.

Множество классов необратимых функций и порождает все разнообразие систем с открытым ключом. Однако не всякая необратимая функция годится для использования в реальных условиях.

В самом определении необратимости присутствует неопределенность. Под необратимостью понимается не теоретическая необратимость, а практическая невозможность вычислить обратное значение используя современные вычислительные средства за обозримый интервал времени.

Поэтому чтобы гарантировать надежную защиту информации, к системам с открытым ключом (СОК) предъявляются два важных и очевидных требования:

- преобразование исходного текста должно быть необратимым и исключать его восстановление на основе открытого ключа.

– определение закрытого ключа на основе открытого также должно быть невозможным на современном технологическом уровне. При этом желательна точная нижняя оценка сложности (количества операций) раскрытия шифра.

Алгоритмы шифрования с открытым ключом получили широкое распространение в современных информационных системах. Так, алгоритм RSA стал мировым стандартом де-факто для открытых систем и рекомендован МККТТ.

Вообще же все предлагаемые сегодня криптосистемы с открытым ключом опираются на один из следующих типов необратимых преобразований:

– разложение больших чисел на простые множители;
– вычисление логарифма в конечном поле;
– вычисление корней алгебраических уравнений.

Следует отметить, что алгоритмы криптосистемы с открытым ключом (СОК) можно использовать в трех назначениях.

– как самостоятельные средства защиты передаваемых и хранимых данных;
– как средства для распределения ключей;
– как средства аутентификации пользователей.

Одна из наиболее известных СОК – криптосистема RSA, разработанная в 1977 году и названная по именам создателей: Рона Ривеста, Ади Шамира и Леонарда Эйдельмана.

В настоящее время алгоритм RSA используется во многих стандартах, среди которых SSL, TLS, SHTTP, SSH, PGP, S/MIME и др.

Среди других СОК – Elgamal, DSA, ГОСТ Р 34.10-2001, McEliece.

Модель безопасности в Windows

В ОС Windows реализована дискреционная модель безопасности. В качестве субъектов этой модели рассматриваются процессы и потоки, каждый из которых работает от имени некоторого пользователя. Когда пользователь входит в систему, то для него создается ***маркер доступа*** (access token), который идентифицирует пользователя и содержит его привилегии. Каждый процесс пользователя имеет маркер доступа этого пользователя. Маркер доступа используется для контроля доступа к объектам, которые называются в Windows ***охраняемыми объектами***. К ним относятся все объекты Windows, которые могут иметь имя, а также потоки и процессы. Каждый охраняемый объект имеет дескриптор безопасности (security descriptor, SD), который создается вместе с объектом и содержит информацию, необходимую для защиты объекта. При доступе к охраняемому объекту система сверяет информацию о пользователе, заданную в маркере доступа с информацией в дескрипторе безопасности. Если в SD указано, что пользователю разрешен доступ к объекту, то процесс получает требуемый

доступ.

Для хранения информации о пользователях, которым разрешен или запрещен доступ к охраняемым объектам каждый SD содержит ***список управления дискреционным доступом*** (**Discretionary Access-Control List, DACL**). Кроме DACL для управлением аудитом к объекту в SD хранится ***список управления системным доступом*** (**System Access-Control List**, **SACL**). Общее название этих списков – ***списки управления доступом*** (**Access-Control Lists**, **ACL**).

Для каждой учетной записи ОС создает ***идентификатор безопасности*** (**Security Identifier**, **SID**), который хранится в реестре в базе данных менеджера учетных записей SAM.

SID является бинарным представлением учетной записи и используется системой безопасности для идентификации учетных записей.

Символически SID может быть описан следующим образом:

S – R – I – SA0 – SA1 – SA2 – SA3 – SA4 - …

Здесь:

S – символ S, обозначающий, что дальнейшее числовое значение является идентификатором безопасности;

R – версия (Revision Level) формата SID. С NT 3.1 R=1;

I – 48-битное число, обозначающее уровень авторизации учетной записи (Top-level Authority или Identifier Authority);

SA – 32-битное число, которое уточняет уровень авторизации учетной записи (Subauthority), связанной с данным SID. Это число также называется ***относительным идентификатором учетной записи*** (Relative Identifier, **RID**).

В общем случае количество SA может быть произвольным, поэтому необходимо особое внимание уделить выделению памяти для хранения SID.

В случае пользователей и групп поле SA0 уточняет авторизацию учетной записи, SA1-SA2-SA3 представляют собой уникальный 96-битный идентификатор компьютера или домена, которому принадлежит пользователь, SA4 – нумерует идентификаторы безопасности, создаваемые внутри системы. Номера от 0 до 999 зарезервированы для использования ОС, а начиная с 1000 присваиваются новым SID.

SD охраняемого объекта содержит заголовок с управляющими флагами и указателями на следующую информацию:

- SID владельца объекта;
- SID первичной группы владельца объекта;
- указатель на DACL;
- указатель на SACL.

Каждый ACL содержит элементы, которые называются входами управления доступом (Access-Control Entries, ACE), которые содержат следующую информацию:

- SID субъекта, которому разрешен или запрещен

доступ;
- маску доступа, которая специфицирует права доступа субъекта к охраняемому объекту (специфические, стандартные, родовые (GENERIC_READ) и SACL access rights);
- флаг, который определяет тип ACE;
- флаги, определяющие наследование данного элемента ACE;
- флаги, управляющие аудитом доступа к охраняемому объекту.

Каждый ACE может быть одного из двух типов: ACCESS_ALLOWED_ACE или ACCESS_DENIED_ACE – определять либо разрешения на доступ, либо запрет на доступ.

При определении разрешен или нет доступ к объекту ОС извлекает из маркера доступа SID пользователя и сравнивает с хранящимися в DACL объекта ACE. Если такой ACE не найден субъект получает отказ. Иначе ОС проверяет тип элемента ACE и затребованный доступ.

При определении доступа к охраняемому объекту система различает отсутствие DACL (доступ разрешен для всех) и пустой DACL (доступ запрещен для всех).

Часть WinAPI, отвечающую за работу с подсистемой безопасности Windows можно разделить на следующие категории:

- управление пользователями (создание учетной записи, перечисление пользователей, получение информации о пользователе и т.д.);
- управление группами (создание группы, перечисление групп, добавление пользователей в группу и т.д.);
- работа с ACL на высоком уровне;
- работа с SID (создание, определение имени по SID и наоборот, получение характеристик SID и т.д.);
- работа с SD (создание нового, получение SD, получение данных из SD и т.д.);
- работа с маркерами доступа (открытие, получение информации из маркера, настройка и дублирование и т.д.);
- работа с привилегиями;
- работа с ACL и ACE на низком уровне.

Эти группы функции достаточно подробно описаны в MSDN, поэтому остановимся подробнее только на 2 разделах: работе с SID и работе с SD.

Работа с SID

Как было сказано ранее, SID – бинарное представление учетной записи. В Windows SID представлен структурой следующего типа:

```
typedef struct _SID {
 BYTE Revision;
```

```
  BYTE SubAuthorityCount;
  SID_IDENTIFIER_AUTHORITY IdentifierAuthority;
  DWORD SubAuthority[ANYSIZE_ARRAY];
} SID;
```

Все действия со структурой проводятся не напрямую, а посредством системных вызовов.

Из описания видно, что это структура переменно длины. Поэтому, прежде чем создавать SID следует определиться с его длиной. Для определения длины SID используется функция `GetSidLengthRequired`. После этого резервируется память для SID.

Сама структура SID инициализируется при помощи функции `InitializeSid`, причем эта функция инициализирует только первых 3 члена структуры, остальные должны инициализироваться отдельно.

После инициализации SID следует проверить его структуру. Для этого используется вызов `IsValidSid`.

Все выше перечисленные действия можно выполнить одной функцией – `AllocateAndInitializeSid`, однако в этом случае зарезервированную память нужно освобождать функцией `FreeSid`.

Для определения учетной записи по SID используется функция `LookupAccountSid`, имеющая следующий прототип:

```
BOOL LookupAccountSid(
  LPCTSTR lpSystemName,
  PSID lpSid,
  LPTSTR lpName,
  LPDWORD cchName,
  LPTSTR lpReferencedDomainName,
  LPDWORD cchReferencedDomainName,
  PSID_NAME_USE peUse
);
```

Здесь,
`lpSystemName` – имя компьютера,
`lpSid` – указатель на SID,
`lpName` – имя учетной записи,
`cbName` – длина имени учетной записи,
`lpReferencedDomainName` – имя домена,
`cbReferencedDomainName` – длина имени домена,
`peUse` – тип SID.

В случае успешного завершения функция возвращает ненулевое значение, иначе – `FALSE`.

Замечание: при работе с подсистемой безопасности желательно использовать кодировку UNICODE. Т.е. при написании программы следует добавить:

```
#ifndef UNICODE
```

```
#define UNICODE
#endif
```

Обратное определение – SID по имени учетной записи выполняется функцией `LookupAccountName`.

Для определения длины SID используют `GetLengthSid`.

Для преобразования SID в строку используют `ConvertSidToStringSid`, для обратного преобразования – `ConvertStringSidToSid`.

Работа с SD

Дескриптор безопасности (далее **SD)** представлен в системе структурой **SECURITY_DESCRIPTOR**. Также как и SID, SD не модифицируется напрямую, а только посредством соответствующего API.

Рассмотрим некоторые из функции этого API. Получить SD охраняемого объекта можно с помощью двух функции `GetSecurityInfo` и `GetNamedSecurityInfo`. Первая из них используется для получения SD для объекта, заданного дескриптором, вторая – для именованного объекта.

```
DWORD GetSecurityInfo(
  HANDLE handle,
  SE_OBJECT_TYPE ObjectType,
  SECURITY_INFORMATION SecurityInfo,
  PSID* ppsidOwner,
  PSID* ppsidGroup,
  PACL* ppDacl,
  PACL* ppSacl,
  PSECURITY_DESCRIPTOR* ppSecurityDescriptor
);
```

Здесь,

`handle` – дескриптор объекта, SD которого следует получить;

`ObjectType` – тип объекта с которым ведется работа;

`SecurityInfo` – задает тип получаемой информации (DACL, SACL и т.д.);

`ppsidOwner, ppsidGroup, ppDacl, ppSacl` – указатель на переменную в которой возвращается запрошенный в `SecurityInfo` параметр;

`ppSecurityDescriptor` – указатель на SD объекта, возвращаемый функцией. Когда данный указатель больше не нужен он может быть освобожден функцией `LocalFree`.

Выполнить проверку SD на валидность можно с помощью `IsValidSecurityDescriptor`.

Полученный SD можно проанализировать используя функции

GetSecurityDescriptorDacl, GetSecurityDescriptorGroup, GetSecurityDescriptorLength, GetSecurityDescriptorOwner, GetSecurityDescriptorSacl.

При создании нового SD его необходимо инициализировать, используя InitializeSecurityDescriptor. Установка SD выполняется с помощью функции SetSecurityInfo и SetNamedSecurityInfo.