Содержание:

Введение

Информационная безопасность – сравнительно молодая, быстро развивающаяся область информационных технологий. Под информационной безопасностью будем понимать защищенность информации и поддерживающей инфраструктуры от случайных или преднамеренных воздействий естественного или искусственного характера, которые могут нанести неприемлемый ущерб субъектам информационных отношений, в том числе владельцам и пользователям информации и поддерживающей инфраструктуры.

Защита информации – это комплекс мероприятий, направленных на обеспечение информационной безопасности.

С методологической точки зрения правильный подход к проблемам информационной безопасности начинается с выявления субъектов информационных отношений и интересов этих субъектов, связанных с использованием информационных систем (ИС). Угрозы информационной безопасности – это оборотная сторона использования информационных технологий.

Информационная безопасность – многогранная область деятельности, в которой успех может принести только систематический, комплексный подход. Для решения данной проблемы рассматриваются меры законодательного, административного, процедурного и программно-технического уровня.

Спектр интересов субъектов, связанных с использованием информационных систем, можно разделить на следующие категории: обеспечение доступности, целостности и конфиденциальности информационных ресурсов и поддерживающей инфраструктуры.

Успех практически любой деятельности в немалой степени зависит от умения распоряжаться такой ценностью, как информация. В законе РФ "Об информации, информатизации и защите информации" определено:

- "информационные ресурсы являются объектами собственности граждан, организаций, общественных объединений, государства";

- "информация – сведения о лицах, предметах, событиях, явлениях и процессах (независимо от формы их представления), отраженные на материальных носителях, используемые в целях получения знаний и практических решений".

Информация имеет ряд особенностей:

- не материальна;
- хранится и передается с помощью материальных носителей;
- любой материальный объект содержит информацию о самом себе либо о другом объекте.

Информации присущи следующие свойства:

- Ценность информации определяется степенью ее полезности для владельца. Законом РФ "Об информации, информатизации и защите информации" гарантируется право собственника информации на ее использование и защиту от доступа к ней других лиц (организаций). Если доступ к информации ограничен, то такая информация называется конфиденциальной. Конфиденциальная информация может содержать государственную или коммерческую тайну.
- Достоверность информации определяется достаточной для владельца точностью отражать объекты и процессы окружающего мира в определенных временных и пространственных рамках. Информация, искаженно представляющая действительность, может нанести владельцу значительный материальный и моральный ущерб. Если информация искажена умышленно, то ее называют дезинформацией.
- Своевременность информации, т.е. соответствие ценности и достоверности определенному временному периоду, может быть выражена формулой

$$C(t) = C_0 e^{-2,3t/\tau},$$

где C_0 – ценность информации в момент ее возникновения; t – время от момента возникновения информации до момента определения ее стоимости; τ – время от момента возникновения информации до момента ее устаревания.

Предметом защиты является информация, хранящаяся, обрабатываемая и передаваемая в компьютерных (информационных) системах. Особенностями данного вида информации являются:

- двоичное представление информации внутри системы, независимо от физической сущности носителей исходной информации;
- высокая степень автоматизации обработки и передачи информации;
- концентрация большого количества информации в КС.

Основные проблемы информационной безопасности

Классификация угроз

Знание возможных угроз, а также уязвимых мест защиты, которые эти угрозы обычно эксплуатируют, необходимо для того, чтобы выбирать наиболее экономичные средства обеспечения безопасности.

Общей классификации угроз не существует. Имеет смысл различать неумышленные и умышленные угрозы.

Неумышленные угрозы связаны с:

1. Ошибками оборудования или матобеспечения: сбои процессора, питания, нечитаемые дискеты, ошибки в коммуникациях, ошибки в программах;

2. Ошибками человека: некорректный ввод, неправильная монтировка дисков или лент, запуск неправильных программ, потеря дисков или лент;

3. Форс-мажорными обстоятельствами.

Умышленные угрозы, в отличие от случайных, преследуют цель нанесения ущерба пользователям ОС и, в свою очередь, подразделяются на активные и пассивные. Пассивная угроза - несанкционированный доступ к информации без изменения состояния системы, активная - несанкционированное изменение системы. Можно выделить следующие типы угроз [3]:

1. Незаконное проникновение под видом легального пользователя;

2. Нарушение функционирования системы с помощью программ-вирусов или программ-червей;

3. Нелегальные действия легального пользователя;

Типизация угроз не слишком строгая.

В.И. Грекул [11] приводит свой список наиболее успешных атак на ОС:

1. Попытки чтения страниц памяти, дисков и лент, которые сохранили информацию предыдущего пользователя;

2. Попытки выполнения нелегальных системных вызовов, или системных вызовов с нелегальными параметрами;

3. Внедрение программы, которая выводит на экран слово login . Многие легальные пользователи, видя такое, начинают пытаться входить в систему, и их попытки могут протоколироваться (вариант Троянского коня);

4. Попытки торпедировать программу проверки входа в систему путем многократного нажатия клавиш del, break, rubout, cancel и.т.д. В некоторых системах проверочная программа погибает, и вход в систему становится возможным;

5. Подкуп персонала. Например, малооплачиваемого секретаря;

6. Использование закладных элементов (дыр), специально оставленных дизайнерами системы.

Много говорят и пишут и о программных вирусах, червях, троянских конях. В этой связи обратим внимание на следующий факт, несмотря на экспоненциальный рост числа известных вирусов, аналогичного роста количества инцидентов, вызванных вирусами, не зарегистрировано.

Соблюдение несложных правил компьютерной гигиены сводит риск заражения практически к нулю. Многопользовательские компьютеры меньше страдают от вирусов по сравнению с персональными компьютерами, поскольку там имеются системные средства защиты. Мы не будем останавливаться на уточнении понятий "зловредный код", "вирус", "червь", "Троянский конь", бомба.

Таковы основные угрозы, на долю которых приходится львиная доля урона, наносимого информационным системам.

Формализация подхода к обеспечению информационной безопасности

Классы безопасности

Существует ряд основополагающих документов, в которых регламентированы основные подходы к проблеме информационной безопасности:

- оранжевая (по цвету обложки) книга МО США;
- гармонизированные критерии европейских стран;
- руководящие документы Гостехкомиссии при Президенте РФ;
- рекомендации X.800 по защите распределенных систем;
- федеральный закон «Об информации, информатизации и защите информации».

Основополагающие документы открыли путь к ранжированию информационных систем по степени надежности. Так, в Оранжевой книге определяется четыре уровня безопасности - D , C , B и A. По мере перехода от уровня D до A к надежности систем предъявляются все более жесткие требования. Уровни C и B подразделяются на классы (C1, C2, B1, B2, B3). Чтобы система в результате процедуры сертификации могла быть отнесена к некоторому классу, ее политика безопасности и гарантированность должны удовлетворять оговоренным требованиям.

В качестве примера рассмотрим требования класса C2, которому удовлетворяют некоторые популярные ОС (Windows NT, отдельные реализации Unix и др.):

Каждый пользователь должен быть идентифицирован уникальным входным именем и паролем для входа в систему. Система должна быть в состоянии использовать эти уникальные идентификаторы, чтобы следить за действиями пользователя.

Операционная система должна защищать объекты от повторного использования.

Владелец ресурса (например, такого как файл) должен иметь возможность контролировать доступ к этому ресурсу.

Системный администратор должен иметь возможность учета всех событий, относящихся к безопасности.

Система должна защищать себя от внешнего влияния или навязывания, такого как модификация загруженной системы или системных файлов, хранимых на диске.

Политика безопасности

Основополагающие документы содержат определения многих ключевых понятий связанных с информационной безопасностью. Так, например, важность и сложность проблемы обеспечения безопасности требует выработки политики информационной безопасности, которая подразумевает ответы на следующие вопросы:

1. Какую информацию защищать?

2. Какого рода атаки на безопасность системы могут быть предприняты?

3. Какие средства использовать для защиты каждого вида информации?

В дальнейшем мы будем оперировать понятиями субъект и объект безопасности. Субъект безопасности - активная, а объект - пассивная системные составляющие, к которым применяется политика безопасности. Примерами субъектов могут служить пользователи и группы пользователей, а объектов - файлы, системные таблицы, принтер и т.п. Политика безопасности состоит в присвоении субъектам и объектам идентификаторов и фиксации набора правил, используемых для определения, имеет ли данный субъект авторизацию, достаточную для получения к данному объекту данного типа доступа.

Формируя политику безопасности необходимо учитывать несколько базовых принципов. Так, Saltzer и Schroeder (1975) на основе своего опыта работы с MULTICS сформулировали следующие принципы разработки ОС:

1. Проектирование системы должно быть открытым. Нарушитель и так все знает (криптографические алгоритмы открыты);

2. Не должно быть доступа по умолчанию. Ошибки с отклонением легитимного доступа могут быть выявлены скорее, чем ошибки, там, где разрешен неавторизованный доступ;

3. Тщательно проверять текущее авторство. Так, многие системе проверяют привилегии доступа при открытии файла и не делают этого после. В результате

пользователь может открыть файл и держать его открытым в течение недели и иметь к нему доступ, хотя владелец уже сменил защиту;

4. Давать каждому процессу минимум возможных привилегий;

5. Защитные механизмы должны быть просты, постоянны и встроены в нижний слой системы, это не аддитивные добавки. (Известно много неудачных попыток улучшения защиты слабо приспособленной для этого ОС MS-DOS);

6. Физиологическая приемлемость. Если пользователь видит, что защита требует слишком много усилий, он от нее откажется.

Можно добавить еще ряд, например:

1. Принцип комплексного подхода, баланс надежности защиты всех уровней;

2. Принцип единого контрольно-пропускного пункта;

3. Принцип баланса возможного ущерба от реализации угрозы и затрат на ее предотвращение и ряд других.

Приведенные соображения показывают необходимость продумывания и встраивания защитных механизмов на ранних стадиях проектирования системы.

Защитные механизмы операционных систем

Операционная система есть специально организованная совокупность программ, которая управляет ресурсами системы (ЭВМ, вычислительной системы, других компонентов ИВС) с целью наиболее эффективного их использования и обеспечивает интерфейс пользователя с ресурсами.

Операционные системы, подобно аппаратуре ЭВМ, на пути своего развития прошли несколько поколений.

ОС первого поколения были направлены на ускорение и упрощение перехода с одной задачи пользователя на другую задачу (другого пользователя), что поставило проблему обеспечения безопасности данных, принадлежащих разным задачам.

Второе поколение ОС характеризовалось наращиванием программных средств обеспечения операций ввода-вывода и стандартизацией обработки прерываний.

Надежное обеспечение безопасности данных в целом осталось нерешенной проблемой.

К концу 60-х гг. XX в. начал осуществляться переход к мультипроцессорной организации средств ВТ, поэтому проблемы распределения ресурсов и их защиты стали более острыми и трудноразрешимыми. Решение этих проблем привело к соответствующей организации ОС и широкому применению аппаратных средств защиты (защита памяти, аппаратный контроль, диагностика и т.п.).

Основной тенденцией развития вычислительной техники была и остается идея максимальной доступности ее для пользователей, что входит в противоречие с требованием обеспечения безопасности данных.

Под механизмами защиты ОС будем понимать все средства и механизмы защиты данных, функционирующие в составе ОС. Операционные системы, в составе которых функционируют средства и механизмы защиты данных, часто называют защищенными системами.

Под безопасностью ОС будем понимать такое состояние ОС, при котором невозможно случайное или преднамеренное нарушение функционирования ОС, а также нарушение безопасности находящихся под управлением ОС ресурсов системы.

Укажем следующие особенности ОС, которые позволяют выделить вопросы обеспечения безопасности ОС в особую категорию:

− управление всеми ресурсами системы;

− наличие встроенных механизмов, которые прямо или косвенно влияют на безопасность программ и данных, работающих в среде ОС;

− обеспечение интерфейса пользователя с ресурсами системы;

− размеры и сложность ОС.

Большинство ОС обладают дефектами с точки зрения обеспечения безопасности данных в системе, что обусловлено выполнением задачи обеспечения максимальной доступности системы для пользователя.

Рассмотрим типовые функциональные дефекты ОС, которые могут привести к созданию каналов утечки данных.

1. Идентификация. Каждому ресурсу в системе должно быть присвоено уникальное имя – идентификатор. Во многих системах пользователи не имеют возможности удостовериться в том, что используемые ими ресурсы действительно принадлежат системе.

2. Пароли. Большинство пользователей выбирают простейшие пароли, которые легко подобрать или угадать.

3. Список паролей. Хранение списка паролей в незашифрованном виде дает возможность его компрометации с последующим НСД к данным.

4. Пороговые значения. Для предотвращения попыток несанкционированного входа в систему с помощью подбора пароля необходимо ограничить число таких попыток, что в некоторых ОС не предусмотрено.

5. Подразумеваемое доверие. Во многих случаях программы ОС считают, что другие программы работают правильно.

6. Общая память. При использовании общей памяти не всегда после выполнения программ очищаются участки оперативной памяти (ОП).

7. Разрыв связи. В случае разрыва связи ОС должна немедленно закончить сеанс работы с пользователем или повторно установить подлинность субъекта.

8. Передача параметров по ссылке, а не по значению (при передаче параметров по ссылке возможно сохранение параметров в ОП после проверки их корректности, нарушитель может изменить эти данные до их использования).

9. Система может содержать много элементов (например, программ), имеющих различные привилегии.

Основной проблемой обеспечения безопасности ОС является проблема создания механизмов контроля доступа к ресурсам системы. Процедура контроля доступа заключается в проверке соответствия запроса субъекта предоставленным ему правам доступа к ресурсам. Кроме того, ОС содержит вспомогательные средства защиты, такие как средства мониторинга, профилактического контроля и аудита. В совокупности механизмы контроля доступа и вспомогательные средства защиты образуют механизмы управления доступом.

Средства профилактического контроля необходимы для отстранения пользователя от непосредственного выполнения критичных с точки зрения безопасности данных

операций и передачи этих операций под контроль ОС. Для обеспечения безопасности данных работа с ресурсами системы осуществляется с помощью специальных программ ОС, доступ к которым ограничен.

Средства мониторинга осуществляют постоянное ведение регистрационного журнала, в который заносятся записи о всех событиях в системе. В ОС могут использоваться средства сигнализации о НСД, которые используются при обнаружении нарушения безопасности данных или попыток нарушения.

Контроль доступа к данным. При создании механизмов контроля доступа необходимо, прежде всего, определить множества субъектов и объектов доступа.

Субъектами могут быть, например, пользователи, задания, процессы и процедуры.

Объектами – файлы, программы, семафоры, директории, терминалы, каналы связи, устройства, блоки ОП и т.д. Субъекты могут одновременно рассматриваться и как объекты, поэтому у субъекта могут быть права на доступ к другому субъекту. В конкретном процессе в данный момент времени субъекты являются активными элементами, а объекты – пассивными.

Для осуществления доступа к объекту субъект должен обладать соответствующими полномочиями. Полномочие есть некий символ, обладание которым дает субъекту определенные права доступа по отношению к объекту, область защиты определяет права доступа некоторого субъекта ко множеству защищаемых объектов и представляет собой совокупность всех полномочий данного субъекта.

При функционировании системы необходимо иметь возможность создавать новые субъекты и объекты. При создании объекта одновременно создается и полномочие субъектов по использованию этого объекта. Субъект, создавший такое полномочие, может воспользоваться им для осуществления доступа к объекту или же может создать несколько копий полномочия для передачи их другим субъектам.

С традиционной точки зрения средства управления доступом позволяют специфицировать и контролировать действия, которые субъекты (пользователи и процессы) могут выполнять над объектами (информацией и другими компьютерными ресурсами). В данном разделе речь пойдет о логическом управлении доступом, которое, в отличие от физического, реализуется программными средствами. Логическое управление доступом – это основной механизм многопользовательских систем, призванный обеспечить

конфиденциальность и целостность объектов и, до некоторой степени, их доступность (путем запрещения обслуживания неавторизованных пользователей).

Рассмотрим формальную постановку задачи в традиционной трактовке. Имеется совокупность субъектов и набор объектов. Задача логического управления доступом состоит в том, чтобы для каждой пары "субъект-объект" определить множество допустимых операций и контролировать выполнение установленного порядка.

Отношение "субъекты-объекты" можно представить в виде матрицы доступа, в строках которой перечислены субъекты, в столбцах – объекты, а в клетках, расположенных на пересечении строк и столбцов, записаны дополнительные условия (например, время и место действия) и разрешенные виды доступа.

Матрица доступа

Модель безопасности, специфицированная в предыдущем разделе, имеет вид матрицы, которая называется матрицей доступа. Какова может быть эффективная реализация матрицы доступа? В общем случае она будет разреженной, то есть большинство ее клеток будут пустыми. Хотя существуют структуры данных для представления разреженной матрицы, они не слишком полезны для приложений, использующих возможности защиты. Поэтому на практике матрица доступа применяется редко. Эту матрицу можно разложить по столбцам, в результате чего получаются списки прав доступа (access control list - ACL). В результате разложения по строкам получаются мандаты возможностей (capability list илиcapability tickets). Отношение "субъекты-объекты" можно представить в виде матрицы доступа, в строках которой перечислены субъекты, в столбцах – объекты, а в клетках, расположенных на пересечении строк и столбцов, записаны дополнительные условия (например, время и место действия) и разрешенные виды доступа.

Фрагмент матрицы может выглядеть, например, как показано в табл. 1.

Таблица 1. Фрагмент матрицы доступа

	Файл	Программа	Линия связи	Реляционная таблица

	o r w		rw
Пользователь 1	с системной консоли	e	с 8:00 до 18:00
Пользователь 2	a		

О б о з н а ч е н и е : "o" – разрешение на передачу прав доступа другим пользователям, "r" – чтение, "w" – запись, "e" – выполнение, "a" – добавление информации.

Тема логического управления доступом – одна из сложнейших в области информационной безопасности. Дело в том, что само понятие объекта (а тем более видов доступа) меняется от сервиса к сервису. Для операционной системы к объектам относятся файлы, устройства и процессы. Применительно к файлам и устройствам обычно рассматриваются права на чтение, запись, выполнение (для программных файлов), иногда на удаление и добавление. Отдельным правом может быть возможность передачи полномочий доступа другим субъектам (так называемое право владения). Процессы можно создавать и уничтожать. Современные операционные системы могут поддерживать и другие объекты.

Для систем управления реляционными базами данных объект – это база данных, таблица, представление, хранимая процедура. К таблицам применимы операции поиска, добавления, модификации и удаления данных, у других объектов. В результате при задании матрицы доступа нужно принимать во внимание не только принцип распределения привилегий для каждого сервиса, но и существующие связи между сервисами (приходится заботиться о согласованности разных частей матрицы). Аналогичная трудность возникает при экспорте/импорте данных, когда информация о правах доступа, как правило, теряется (поскольку на новом сервисе она не имеет смысла).

Следовательно, обмен данными между различными сервисами представляет особую опасность с точки зрения управления доступом, а при проектировании и реализации разнородной конфигурации необходимо позаботиться о согласованном распределении прав доступа субъектов к объектам и о минимизации числа способов экспорта/импорта данных.

Матрицу доступа, ввиду ее разреженности (большинство клеток – пустые), неразумно хранить в виде двухмерного массива. Обычно ее хранят по столбцам, т.е. для каждого объекта поддерживается список "допущенных" субъектов вместе с их правами. Элементами списков могут быть имена групп и шаблоны субъектов, что служит большим подспорьем администратору. Некоторые проблемы возникают только при удалении субъекта, когда приходится удалять его имя из всех списков доступа; впрочем, эта операция производится нечасто.

Списки доступа – исключительно гибкое средство. С их помощью легко выполнить требование о гранулярности прав с точностью до пользователя. Посредством списков несложно добавить права или явным образом запретить доступ (например, чтобы наказать нескольких членов группы пользователей). Безусловно, списки являются лучшим средством произвольного управления доступом.

Подавляющее большинство операционных систем и систем управления базами данных реализуют именно произвольное управление доступом. Основное достоинство произвольного управления – гибкость. К сожалению, у "произвольного" подхода есть ряд недостатков. Рассредоточенность управления доступом ведет к тому, что доверенными должны быть многие пользователи, а не только системные операторы или администраторы. Из-за рассеянности или некомпетентности сотрудника, владеющего секретной информацией, эту информацию могут узнать и все остальные пользователи. Следовательно, произвольность управления должна быть дополнена жестким контролем за реализацией избранной политики безопасности.

Второй недостаток, который представляется основным, состоит в том, что права доступа существуют отдельно от данных. Ничто не мешает пользователю, имеющему доступ к секретной информации, записать ее в доступный всем файл или заменить полезную утилиту ее "троянским" аналогом. Подобная "разделенность" прав и данных существенно осложняет проведение несколькими системами согласованной политики безопасности и, главное, делает практически невозможным эффективный контроль согласованности.

Возвращаясь к вопросу представления матрицы доступа, укажем, что для этого можно использовать также функциональный способ, когда матрицу не хранят в явном виде, а каждый раз вычисляют содержимое соответствующих клеток. Например, при принудительном управлении доступом применяется сравнение меток безопасности субъекта и объекта.

Удобной надстройкой над средствами логического управления доступом является ограничивающий интерфейс, когда пользователя лишают самой возможности попытаться совершить несанкционированные действия, включив в число видимых ему объектов только те, к которым он имеет доступ. Подобный подход обычно реализуют в рамках системы меню (пользователю показывают лишь допустимые варианты выбора) или посредством ограничивающих оболочек, таких как restricted shell в ОС Unix.

При принятии решения о предоставлении доступа обычно анализируется следующая информация:

1. Идентификатор субъекта (идентификатор пользователя, сетевой адрес компьютера и т.п.). Подобные идентификаторы являются основой произвольного (или дискреционного) управления доступом;
2. Атрибуты субъекта (метка безопасности, группа пользователя и т.п.). Метки безопасности – основа мандатного управления доступом.

Непосредственное управление правами доступа осуществляется на основе одной из моделей доступа:

- матричной модели доступа (модель Харрисона-Руззо-Ульмана);
- многоуровневой модели доступа (модель Белла-Лападулы).

Разработка и практическая реализация различных защищенных ОС привела Харрисона, Руззо и Ульмана к построению формальной модели защищенных систем. Схема модели Харрисона, Руззо и Ульмана (HRU-модели) приведена на рис. 1.

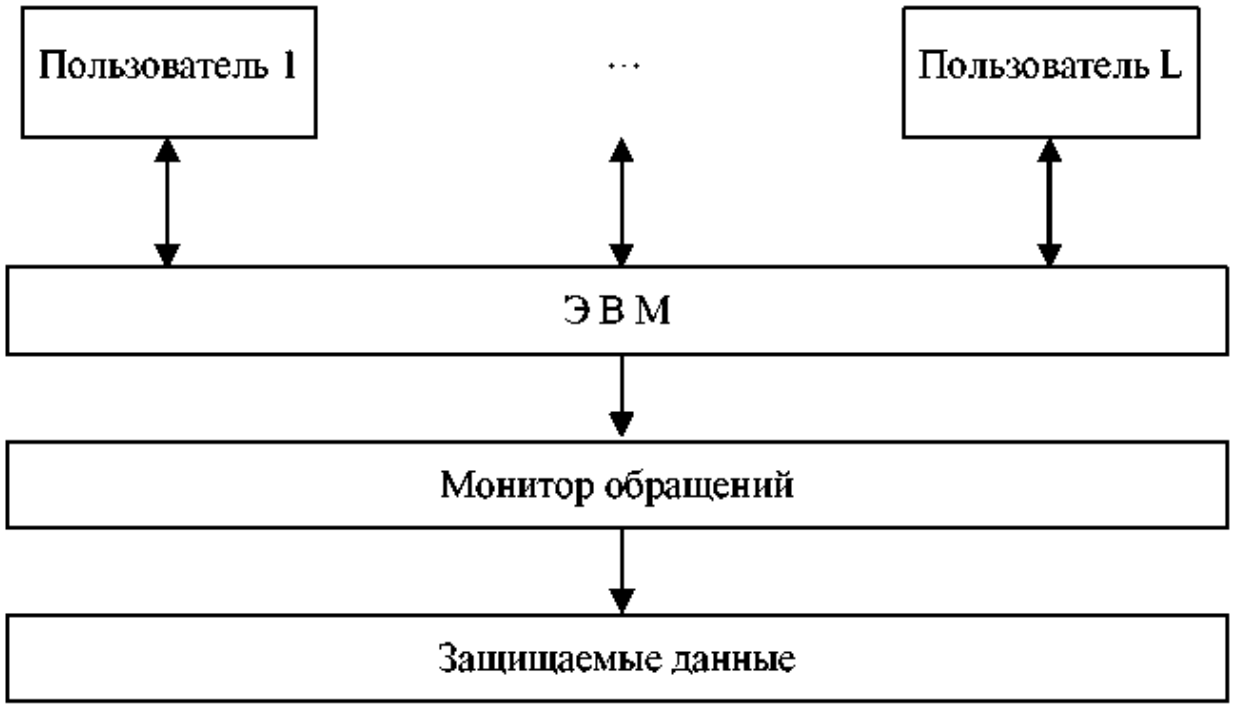

Рисунок 1. Схема модели Харрисона, Руззо и Ульмана

Список прав доступа. Access control list

Каждая колонка в матрице может быть реализована как список доступа для одного объекта. Очевидно, что пустые клетки могут не учитываться. В результате для каждого объекта имеем список упорядоченных пар <domain, rights-set>, который определяет все домены с непустыми наборами прав для данного объекта.

Элементами списка могут быть процессы, пользователи или группы пользователей. При реализации широко применяется предоставление доступа по умолчанию для пользователей, права которых не указаны. Например, в Unix все субъекты-пользователи разделены на три группы (владелец, группа и остальные), и для членов каждой группы контролируются операции чтения, записи и исполнения (rwx). В итоге имеем ACL - 9-битный код, который является атрибутом разнообразных объектов Unix.

Мандаты возможностей. Capability list

Как отмечалось выше, если матрицу доступа хранить по строкам, то есть если каждый субъект хранит список объектов и для каждого объекта - список допустимых операций, то такой способ хранения называется "мандаты" или "перечни возможностей" (capability list) . Каждый пользователь обладает несколькими мандатами и может иметь право передавать их другим. Мандаты

могут быть рассеяны по системе и вследствие этого представлять большую угрозу для безопасности, чем списки контроля доступа. Их хранение должно быть тщательно продумано.

Примерами систем, использующих перечни возможностей, являются Hydra, Cambridge CAP System.

Другие способы контроля доступа

Иногда применяется комбинированный способ. Например, в том же Unix на этапе открытия файла происходит анализ ACL (операция open). В случае благоприятного исхода файл заносится в список открытых процессом файлов, и при последующих операциях чтения и записи проверки прав доступа не происходит. Список открытых файлов можно рассматривать как перечень возможностей.

Существует также схема lock-key, которая является компромиссом между списками прав доступа и перечнями возможностей. В этой схеме каждый объект имеет список уникальных битовых шаблонов (patterns), называемых locks. Аналогично каждый домен имеет список уникальных битовых шаблонов, называемых ключами (keys). Процесс, выполняющийся в домене, может получить доступ к объекту, только если домен имеет ключ, который соответствует одному из шаблонов объекта.

Как и в случае мандатов, список ключей для домена должен управляться ОС. Пользователям не разрешается проверять или модифицировать списки ключей (или шаблонов) непосредственно.

Смена домена

В большинстве ОС для определения домена применяются идентификаторы пользователей. Обычно переключение между доменами происходит, когда меняется пользователь. Но почти все системы нуждаются в дополнительных механизмах смены домена, которые используются, когда некая привилегированная возможность необходима большому количеству пользователей. Например, может понадобиться разрешить пользователям иметь доступ к сети, не заставляя их писать собственные сетевые программы.

В таких случаях для процессов ОС Unix предусмотрена установка бита set-uid. В результате установки этого бита в сетевой программе она получает привилегии ее создателя (а не пользователя), заставляя домен меняться на время ее выполнения. Таким образом, рядовой пользователь может получить нужные привилегии для доступа к сети.

Недопустимость повторного использования объектов

Контроль повторного использования объекта предназначен для предотвращения попыток незаконного получения конфиденциальной информации, остатки которой могли сохраниться в некоторых объектах, ранее использовавшихся и освобожденных другим пользователем.

Безопасность повторного применения должна гарантироваться для областей оперативной памяти (в частности, для буферов с образами экрана, расшифрованными паролями и т.п.), для дисковых блоков и магнитных носителей в целом. Очистка должна производиться путем записи маскирующей информации в объект при его освобождении (перераспределении). Например, для дисков на практике применяется способ двойной перезаписи освободившихся после удаления файлов блоков случайной битовой последовательностью.

Выявление вторжений. Аудит системы защиты

Даже самая лучшая система защиты рано или поздно будет взломана. Обнаружение попыток вторжения является важнейшей задачей системы защиты, поскольку ее решение позволяет минимизировать ущерб от взлома и собирать информацию о методах вторжения. Как правило, поведение взломщика отличается от поведения легального пользователя. Иногда эти различия можно выразить количественно, например, подсчитывая число некорректных вводов пароля во время регистрации.

Основным инструментом выявления вторжений является запись данных аудита. Отдельные действия пользователей протоколируются, а полученный протокол используется для выявления вторжений.

Аудит, таким образом, заключается в регистрации специальных данных о различных типах событий, происходящих в системе и так или иначе влияющих на

состояние безопасности компьютерной системы.

К числу таких событий обычно причисляют следующие:

- вход или выход из системы;
- операции с файлами (открыть, закрыть, переименовать, удалить);
- обращение к удаленной системе;
- смена привилегий или иных атрибутов безопасности (режима доступа, уровня благонадежности пользователя и т. п.).

Если фиксировать все события, объем регистрационной информации, скорее всего, будет расти слишком быстро, а ее эффективный анализ станет невозможным. Следует предусматривать наличие средств выборочного протоколирования как в отношении пользователей, когда слежение осуществляется только за подозрительными личностями, так и в отношении событий. Слежка важна в первую очередь как профилактическое средство. Можно надеяться, что многие воздержатся от нарушений безопасности, зная, что их действия фиксируются.

Помимо протоколирования, можно периодически сканировать систему на наличие слабых мест в системе безопасности. Такое сканирование может проверить разнообразные аспекты системы:

- короткие или легкие пароли;
- неавторизованные set-uid программы, если система поддерживает этот механизм;
- неавторизованные программы в системных директориях;
- долго выполняющиеся программы;
- нелогичная защита как пользовательских, так и системных директорий и файлов. Примером нелогичной защиты может быть файл, который запрещено читать его автору, но в который разрешено записывать информацию постороннему пользователю;
- потенциально опасные списки поиска файлов, которые могут привести к запуску "троянского коня";
- изменения в системных программах, обнаруженные при помощи контрольных сумм.

Любая проблема, обнаруженная сканером безопасности, может быть как ликвидирована автоматически, так и передана для решения менеджеру системы.

Идентификация и аутентификация

Для начала рассмотрим проблему контроля доступа в систему. Наиболее распространенным способом контроля доступа является процедура регистрации. Обычно каждый пользователь в системе имеет уникальный идентификатор. Идентификаторы пользователей применяются с той же целью, что и идентификаторы любых других объектов, файлов, процессов.

Идентификация заключается в сообщении пользователем своего идентификатора. Для того чтобы установить, что пользователь именно тот, за кого себя выдает, то есть что именно ему принадлежит введенный идентификатор, в информационных системах предусмотрена процедура аутентификации (authentication, опознавание, в переводе с латинского означает "установление подлинности"), задача которой - предотвращение доступа к системе нежелательных лиц.

Обычно аутентификация базируется на одном или более из трех пунктов:

- то, чем пользователь владеет (ключ или магнитная карта);
- то, что пользователь знает (пароль);
- атрибуты пользователя (отпечатки пальцев, подпись, голос);

Пароли, уязвимость паролей.

Наиболее простой подход к аутентификации - применение пользовательского пароля.

Когда пользователь идентифицирует себя при помощи уникального идентификатора или имени, у него запрашивается пароль. Если пароль, сообщенный пользователем, совпадает с паролем, хранящимся в системе, система предполагает, что пользователь легитимен. Пароли часто используются для защиты объектов в компьютерной системе в отсутствие более сложных схем защиты.

Недостатки паролей связаны с тем, что трудно сохранить баланс между удобством пароля для пользователя и его надежностью. Пароли могут быть угаданы, случайно показаны или нелегально переданы авторизованным пользователем неавторизованному.

Есть два общих способа угадать пароль. Один связан со сбором информации о пользователе. Люди обычно используют в качестве паролей очевидную

информацию (скажем, имена животных или номерные знаки автомобилей). Для иллюстрации важности разумной политики назначения идентификаторов и паролей можно привести данные исследований, проведенных в AT&T, показывающие, что из 500 попыток несанкционированного доступа около 300 составляют попытки угадывания паролей или беспарольного входа по пользовательским именам guest, demo и т. д.

Другой способ - попытаться перебрать все наиболее вероятные комбинации букв, чисел и знаков пунктуации (атака по словарю). Например, четыре десятичные цифры дают только 10 000 вариантов, более длинные пароли, введенные с учетом регистра символов и пунктуации, не столь уязвимы, но, тем не менее, таким способом удается разгадать до 25% паролей. Чтобы заставить пользователя выбрать трудноугадываемый пароль, во многих системах внедрена реактивная проверка паролей, которая при помощи собственной программы-взломщика паролей может оценить качество пароля, введенного пользователем.

Несмотря на все это, пароли распространены, поскольку они удобны и легко реализуемы.

Шифрование пароля

Для хранения секретного списка паролей на диске во многих ОС используется криптография. Система задействует одностороннюю функцию, которую просто вычислить, но для которой чрезвычайно трудно (разработчики надеются, что невозможно) подобрать обратную функцию.

Например, в ряде версий Unix в качестве односторонней функции используется модифицированный вариант алгоритма DES. Введенный пароль длиной до 8 знаков преобразуется в 56-битовое значение, которое служит входным параметром для процедуры crypt(), основанной на этом алгоритме. Результат шифрования зависит не только от введенного пароля, но и от случайной последовательности битов, называемой привязкой (переменная salt). Это сделано для того, чтобы решить проблему совпадающих паролей. Очевидно, что саму привязку после шифрования необходимо сохранять, иначе процесс не удастся повторить.

Модифицированный алгоритм DES выполняется, имея входное значение в виде 64-битового блока нулей, с использованием пароля в качестве ключа, а на каждой

следующей итерации входным параметром служит результат предыдущей итерации. Всего процедура повторяется 25 раз. Полученное 64-битовое значение преобразуется в 11 символов и хранится рядом с открытой переменной salt.

В ОС Windows NT преобразование исходного пароля также осуществляется многократным применением алгоритма DES и алгоритма MD4. Хранятся только кодированные пароли. В процессе аутентификации представленный пользователем пароль кодируется и сравнивается с хранящимися на диске. Таким образом, файл паролей нет необходимости держать в секрете.

При удаленном доступе к ОС нежелательна передача пароля по сети в открытом виде. Одним из типовых решений является использование криптографических протоколов. В качестве примера можно рассмотреть протокол опознавания с подтверждением установления связи путем вызова - CHAP (Challenge Handshake Authentication Protocol).

Опознавание достигается за счет проверки того, что у пользователя, осуществляющего доступ к серверу, имеется секретный пароль, который уже известен серверу.

Пользователь инициирует диалог, передавая серверу свой идентификатор. В ответ сервер посылает пользователю запрос (вызов), состоящий из идентифицирующего кода, случайного числа и имени узла сервера или имени пользователя. При этом пользовательское оборудование в результате запроса пароля пользователя отвечает следующим ответом, зашифрованным с помощью алгоритма одностороннего хеширования, наиболее распространенным видом которого является MD5. После получения ответа сервер при помощи той же функции с теми же аргументами шифрует собственную версию пароля пользователя. В случае совпадения результатов вход в систему разрешается. Существенно, что незашифрованный пароль при этом по каналу связи не посылается.

В микротелефонных трубках используется аналогичный метод.

В системах, работающих с большим количеством пользователей, когда хранение всех паролей затруднительно, применяются для опознавания сертификаты, выданные доверенной стороной.

Авторизация. Разграничение доступа к объектам ОС

После успешной регистрации система должна осуществлять авторизацию (authorization) - предоставление субъекту прав на доступ к объекту. Средства авторизации контролируют доступ легальных пользователей к ресурсам системы, предоставляя каждому из них именно те права, которые были определены администратором, а также осуществляют контроль возможности выполнения пользователем различных системных функций. Система контроля базируется на общей модели, называемой матрицей доступа. Рассмотрим ее более подробно.

Как уже говорилось в предыдущей лекции, компьютерная система может быть смоделирована как набор субъектов (процессы, пользователи) и объектов. Под объектами мы понимаем как ресурсы оборудования (процессор, сегменты памяти, принтер, диски и ленты), так и программные ресурсы (файлы, программы, семафоры), то есть все то, доступ к чему контролируется. Каждый объект имеет уникальное имя, отличающее его от других объектов в системе, и каждый из них может быть доступен через хорошо определенные и значимые операции.

Операции зависят от объектов. Например, процессор может только выполнять команды, сегменты памяти могут быть записаны и прочитаны, считыватель магнитных карт может только читать, а файлы данных могут быть записаны, прочитаны, переименованы и т. д.

Желательно добиться того, чтобы процесс осуществлял авторизованный доступ только к тем ресурсам, которые ему нужны для выполнения его задачи. Это требование минимума привилегий, уже упомянутое в предыдущей лекции, полезно с точки зрения ограничения количества повреждений, которые процесс может нанести системе. Например, когда процесс P вызывает процедуру A, ей должен быть разрешен доступ только к переменным и формальным параметрам, переданным ей, она не должна иметь возможность влиять на другие переменные процесса. Аналогично компилятор не должен оказывать влияния на произвольные файлы, а только на их хорошо определенное подмножество (исходные файлы, листинги и др.), имеющее отношение к компиляции. С другой стороны, компилятор может иметь личные файлы, используемые для оптимизационных целей, к которым процесс P не имеет доступа.

Различают дискреционный (избирательный) способ управления доступом и полномочный (мандатный).

При дискреционном доступе, подробно рассмотренном ниже, определенные операции над конкретным ресурсом запрещаются или разрешаются субъектам или группам субъектов. С концептуальной точки зрения текущее состояние прав доступа при дискреционном управлении описывается матрицей, в строках которой перечислены субъекты, в столбцах - объекты, а в ячейках - операции, которые субъект может выполнить над объектом.

Полномочный подход заключается в том, что все объекты могут иметь уровни секретности, а все субъекты делятся на группы, образующие иерархию в соответствии с уровнем допуска к информации. Иногда это называют моделью многоуровневой безопасности, которая должна обеспечивать выполнение следующих правил.

Простое свойство секретности. Субъект может читать информацию только из объекта, уровень секретности которого не выше уровня секретности субъекта. Генерал читает документы лейтенанта, но не наоборот.

*-свойство. Субъект может записывать информацию в объекты только своего уровня или более высоких уровней секретности. Генерал не может случайно разгласить нижним чинам секретную информацию.

Некоторые авторы утверждают [Таненбаум, 2002], что последнее требование называют *-свойством, потому что в оригинальном докладе не смогли придумать для него подходящего названия. В итоге во все последующие документы и монографии оно вошло как *-свойство.

Отметим, что данная модель разработана для хранения секретов, но не гарантирует целостности данных. Например, здесь лейтенант имеет право писать в файлы генерала. Более подробно о реализации подобных формальных моделей рассказано в [Столлингс, 2002], [Таненбаум, 2002].

Большинство операционных систем реализуют именно дискреционное управление доступом. Главное его достоинство - гибкость, основные недостатки - рассредоточенность управления и сложность централизованного контроля.

Чтобы рассмотреть схему дискреционного доступа более детально, введем концепцию домена безопасности (protection domain). Каждый домен определяет

набор объектов и типов операций, которые могут производиться над каждым объектом. Возможность выполнять операции над объектом есть права доступа, каждое из которых есть упорядоченная пара <object-name, rights-set>. Домен, таким образом, есть набор прав доступа. Например, если домен D имеет права доступа <file F, {read, write}>, это означает, что процесс, выполняемый в домене D, может читать или писать в файл F, но не может выполнять других операций над этим объектом. Связь конкретных субъектов, функционирующих в операционных системах, может быть организована следующим образом.

Каждый пользователь может быть доменом. В этом случае набор объектов, к которым может быть организован доступ, зависит от идентификации пользователя.

Каждый процесс может быть доменом. В этом случае набор доступных объектов определяется идентификацией процесса.

Каждая процедура может быть доменом. В этом случае набор доступных объектов соответствует локальным переменным, определенным внутри процедуры. Заметим, что когда процедура выполнена, происходит смена домена.

Рассмотрим стандартную двухрежимную модель выполнения ОС. Когда процесс выполняется в режиме системы (kernel mode), он может выполнять привилегированные инструкции и иметь полный контроль над компьютерной системой. С другой стороны, если процесс выполняется в пользовательском режиме, он может вызывать только непривилегированные инструкции.

Следовательно, он может выполняться только внутри предопределенного пространства памяти. Наличие этих двух режимов позволяет защитить ОС (kernel domain) от пользовательских процессов (выполняющихся в user domain). В мультипрограммных системах двух доменов недостаточно, так как появляется необходимость защиты пользователей друг от друга. Поэтому требуется более тщательно разработанная схема.

В ОС Unix домен связан с пользователем. Каждый пользователь обычно работает со своим набором объектов.

Анализ некоторых популярных ОС с точки зрения их защищенности

Анализировать выполнение современными универсальными ОС требований, задаваемых для класса защищенности АС 1В, не имеет смысла в принципе. Для

большинства ОС либо полностью не реализуется основной для данных приложений мандатный механизм управления доступом к ресурсам, либо не выполняется его важнейшее требование "Должно осуществляться управление потоками информации с помощью меток конфиденциальности. При этом уровень конфиденциальности накопителя должен быть не ниже уровня конфиденциальности записываемой на него информации". В связи с этим далее будем говорить лишь о возможном соответствии средств защиты современных ОС классу АС 1Г (защита конфиденциальной информации).

В качестве альтернативных реализаций ОС рассмотрим семейства Unix и Windows (естественно, Windows NT/2000, так как о встроенных механизмах защиты ОС Windows 9x/Me говорить вообще не приходится).

Сначала остановимся на принципиальном, даже, можно сказать, концептуальном противоречии между реализованными в ОС механизмами защиты и принятыми формализованными требованиями. Концептуальном в том смысле, что это противоречие характеризует не какой-либо один механизм защиты, а общий подход к построению системы защиты.

Противоречие состоит в принципиальном различии подходов (соответственно требований) к построению схемы администрирования механизмов защиты и, как следствие, это коренным образом сказывается на формировании общих принципов задания и реализации политики безопасности в организации, распределения ответственности за защиту информации, а также на определении того, кого относить к потенциальным злоумышленникам (от кого защищать информацию).

Для иллюстрации из совокупности формализованных требований к системе защиты конфиденциальной информации рассмотрим следующие два требования:

1) право изменять правила разграничения доступа (ПРД) должно предоставляться выделенным субъектам (администрации, службе безопасности и т.д.);

2) должны быть предусмотрены средства управления, ограничивающие распространения прав на доступ.

Данные требования жестко регламентируют схему (или модель) администрирования механизмов защиты. Это должна быть централизованная схема, единственным элементом которой выступает выделенный субъект, в частности, администратор (администратор безопасности). При этом конечный пользователь исключен в принципе из схемы администрирования

механизмов защиты.

При реализации концепции построения системы защиты, регламентируемой рассматриваемыми требованиями, пользователь не наделяется элементом доверия, так как он может считаться потенциальным злоумышленником, что и имеет место на практике.

Теперь в общих чертах рассмотрим концепцию, реализуемую в современных универсальных ОС. Здесь "владельцем" файлового объекта, т.е. лицом, получающим право на задание атрибутов (или ПРД) доступа к файловому объекту, является лицо, создающее файловый объект. Так как файловые объекты создают конечные пользователи, то именно они и назначают ПРД к создаваемым им файловым объектам. Другими словами, в ОС реализуется распределенная схема назначения ПРД, где элементами схемы администрирования являются собственно конечные пользователи.

В данной схеме пользователь должен наделяться практически таким же доверием, как и администратор безопасности, при этом нести наряду с ним ответственность за обеспечение компьютерной безопасности. Отметим, что данная концепция реализуется и большинством современных приложений, в частности СУБД, где пользователь может распространять свои права на доступ к защищаемым ресурсам. Кроме того, не имея в полном объеме механизмов защиты компьютерной информации от конечного пользователя, в рамках данной концепции невозможно рассматривать пользователя в качестве потенциального злоумышленника. А как мы увидим далее, именно с несанкционированными действиями пользователя на защищаемом компьютере (причем как сознательными, так и нет) связана большая часть угроз компьютерной безопасности.

Отметим, что централизованная и распределенная схемы администрирования – это две диаметрально противоположные точки зрения на защиту, требующие совершенно различных подходов к построению моделей и механизмов защиты. При этом сколько-нибудь гарантированную защиту информации можно реализовать только при принятии концепции полностью централизованной схемы администрирования, что подтверждается известными угрозами ОС.

Возможности моделей, методов и средств защиты будем рассматривать применительно к реализации именно концепции централизованного администрирования. Одним из элементов данной концепции является рассмотрение пользователя в качестве потенциального злоумышленника,

способного осуществить НСД к защищаемой информации.

MS-DOS

ОС MS-DOS функционирует в реальном режиме (real-mode) процессора i80x86. В ней невозможно выполнение требования, касающегося изоляции программных модулей (отсутствует аппаратная защита памяти). Уязвимым местом для защиты является также файловая система FAT, не предполагающая у файлов наличия атрибутов, связанных с разграничением доступа к ним. Таким образом, MS-DOS находится на самом нижнем уровне в иерархии защищенных ОС.

NetWare, IntranetWare

Замечание об отсутствии изоляции модулей друг от друга справедливо и в отношении рабочей станции NetWare. Однако NetWare - сетевая ОС, поэтому к ней возможно применение и иных критериев. Это на данный момент единственная сетевая ОС, сертифицированная по классу С2 (следующей, по-видимому, будет Windows 2000). При этом важно изолировать наиболее уязвимый участок системы безопасности NetWare - консоль сервера, и тогда следование определенной практике поможет увеличить степень защищенности данной сетевой операционной системы. Возможность создания безопасных систем обусловлена тем, что число работающих приложений фиксировано и пользователь не имеет возможности запуска своих приложений.

OS/2

OS/2 работает в защищенном режиме (protected-mode) процессора i80x86. Изоляция программных модулей реализуется при помощи встроенных в этот процессор механизмов защиты памяти. Поэтому она свободна от указанного выше коренного недостатка систем типа MS-DOS. Но OS/2 была спроектирована и разработана без учета требований по защите от несанкционированного доступа. Это сказывается, прежде всего, на файловой системе. В файловых системах OS/2 HPFS (high performance file system) и FAT нет места ACL. Кроме того, пользовательские программы имеют возможность запрета прерываний. Следовательно, сертификация OS/2 на соответствие какому-то классу защиты не представляется возможной.

Считается, что такие операционные системы, как MS-DOS, Mac OS, Windows, OS/2, имеют уровень защищенности D (по оранжевой книге). Но, если быть точным,

нельзя считать эти ОС даже системами уровня безопасности D, ведь они никогда не представлялись на тестирование.

Unix

Рост популярности Unix и все большая осведомленность о проблемах безопасности привели к осознанию необходимости достичь приемлемого уровня безопасности ОС, сохранив при этом мобильность, гибкость и открытость программных продуктов. В Unix есть несколько уязвимых с точки зрения безопасности мест, хорошо известных опытным пользователям, вытекающих из самой природы Unix (см., например, раздел "Типичные объекты атаки хакеров" в книге [Дунаев, 1996]). Однако хорошее системное администрирование может ограничить эту уязвимость.

Относительно защищенности Unix сведения противоречивы. В Unix изначально были заложены идентификация пользователей и разграничение доступа. Как оказалось, средства защиты данных в Unix могут быть доработаны, и сегодня можно утверждать, что многие клоны Unix по всем параметрам соответствуют классу безопасности C2.

Обычно, говоря о защищенности Unix, рассматривают защищенность автоматизированных систем, одним из компонентов которых является Unix-сервер. Безопасность такой системы увязывается с защитой глобальных и локальных сетей, безопасностью удаленных сервисов типа telnet и rlogin/rsh и аутентификацией в сетевой конфигурации, безопасностью X Window-приложений. На системном уровне важно наличие средств идентификации и аудита.

В Unix существует список именованных пользователей, в соответствии с которым может быть построена система разграничения доступа.

В ОС Unix считается, что информация, нуждающаяся в защите, находится главным образом в файлах.

По отношению к конкретному файлу все пользователи делятся на три категории:

1. владелец файла;

2. члены группы владельца;

3. прочие пользователи.

Для каждой из этих категорий режим доступа определяет права на операции с файлом, а именно:

- право на чтение;
- право на запись;
- право на выполнение (для каталогов - право на поиск).

В итоге девяти (3x3) битов защиты оказывается достаточно, чтобы специфицировать ACL каждого файла.

Аналогичным образом защищены и другие объекты ОС Unix, например семафоры, сегменты разделяемой памяти и т. п. Указанных видов прав достаточно, чтобы определить допустимость любой операции с файлами.

Например, для удаления файла необходимо иметь право на запись в соответствующий каталог. Как уже говорилось, права доступа к файлу проверяются только на этапе открытия. При последующих операциях чтения и записи проверка не выполняется. В результате, если режим доступа к файлу меняется после того, как файл был открыт, это не сказывается на процессах, уже открывших этот файл. Данное обстоятельство является уязвимым с точки зрения безопасности местом.

Наличие всего трех видов субъектов доступа: владелец, группа, все остальные - затрудняет задание прав "с точностью до пользователя", особенно в случае больших конфигураций. В популярной разновидности Unix - Solaris имеется возможность использовать списки управления доступом (ACL), позволяющие индивидуально устанавливать права доступа отдельных пользователей или групп.

Среди всех пользователей особое положение занимает пользователь root, обладающий максимальными привилегиями. Обычные правила разграничения доступа к нему не применяются - ему доступна вся информация на компьютере.

В Unix имеются инструменты системного аудита - хронологическая запись событий, имеющих отношение к безопасности. К таким событиям обычно относят: обращения программ к отдельным серверам; события, связанные с входом/выходом в систему и другие. Обычно регистрационные действия выполняются специализированным syslog-демоном, который проводит запись событий в регистрационный журнал в соответствии с текущей конфигурацией. Syslog-демон стартует в процессе загрузки системы.

Таким образом, безопасность ОС Unix может быть доведена до соответствия классу С2. Однако разработка на ее основе автоматизированных систем более высокого класса защищенности может быть сопряжена с большими трудозатратами. Операционная система Unix относится к категории многопользовательских многопрограммных ОС, работающих в режиме разделения времени. Богатые возможности, заложенные в ОС Unix, сделали ее наиболее популярной в мире. ОС Unix поддерживается практически на всех типах ЭВМ.

Организация работ в ОС Unix основана на понятии последовательного процесса как единицы работы, управления и потребления ресурсов. Взаимодействие процессов внутри ядра (процесс вызывает ядро как подпрограмму) происходит по принципу сопрограмм. Последовательность вычислений внутри процесса строго выдерживается: процесс, в частности, не может активизировать ввод-вывод и продолжать вычисление параллельно с ним. В этом случае требуется создать параллельный процесс.

Ядро ОС Unix состоит из двух основных частей: управления процессами и управления устройствами. Управление процессами резервирует ресурсы, определяет последовательность выполнения процессов и принимает запросы на обслуживание. Управление устройствами контролирует передачу данных между ОП и периферийными устройствами.

В любой момент времени выполняется либо программа пользователя (процесс), либо команда ОС. В каждый момент времени лишь один пользовательский процесс активен, а все остальные приостановлены. Ядро ОС Unix служит для удовлетворения потребностей процессов.

Процесс – это программа на этапе выполнения. В некоторый момент времени программе могут соответствовать один или несколько процессов, или не соответствовать ни одного. Считается, что процесс является объектом, учтенным в специальной таблице ядра системы. Наиболее важная информация о процессе хранится в двух местах: в таблице процессов и в таблице пользователя, называемой также контекстом процесса. Таблица процессов всегда находится в памяти и содержит на каждый процесс по одному элементу, в котором отражается состояние процесса: адрес в памяти или адрес свопинга, размер, идентификаторы процесса и запустившего его пользователя. Таблица пользователя существует для каждого активного процесса и к ней могут непосредственно адресоваться только программы ядра (ядро резервирует по одному контексту на каждый активный процесс).

В этой таблице содержится информация, требуемая во время выполнения процесса: идентификационные номера пользователя и группы, предназначенные для определения привилегий доступа к файлам, ссылки на системную таблицу файлов для всех открытых процессом файлов, указатель на индексный дескриптор текущего каталога в таблице индексных дескрипторов и список реакций на различные ситуации. Если процесс приостанавливается, контекст становится недоступным и немодифицируемым.

Каталоги файловой системы ОС Unix "спрятаны" от пользователей и защищены механизмами ОС. Скрытой частью файловой организации в ОС Unix является индексный дескриптор файла, который описывает расположение файла, его длину, метод доступа к файлу, даты, связанные с историей создания файла, идентификатор владельца и т.д.

Работа с таблицами является привилегией ядра, что обеспечивает сохранность и безопасность системы.

При взаимодействии с ОС Unix пользователь может обращаться к большому числу информационных объектов или файлов, объединенных в каталоги. Файловая система ОС Unix имеет иерархическую структуру.

В ОС Unix используется четыре типа файлов: обычные, специальные, каталоги, а в некоторых версиях ОС и FIFO-файлы (First In – First Out).

Обычные файлы содержат данные пользователей. Специальные файлы предназначены для организации взаимодействия с устройствами ввода-вывода. Доступ к любому устройству реализуется как обслуживание запроса к специальному (дисковому) файлу. Каталоги используются системой для поддержания файловой структуры. Особенность каталогов состоит в том, что пользователь может читать их содержимое, но выполнять записи в каталоги (изменять структуру каталогов) может только ОС. В ОС Unix, организуются именованные программные каналы, являющиеся соединительным средством между стандартным выводом одной программы и стандартным вводом другой.

Схема типичной файловой системы ОС Unix приведена на рис. 2. Рассмотрим основные механизмы защиты данных, реализованные в ОС Unix.

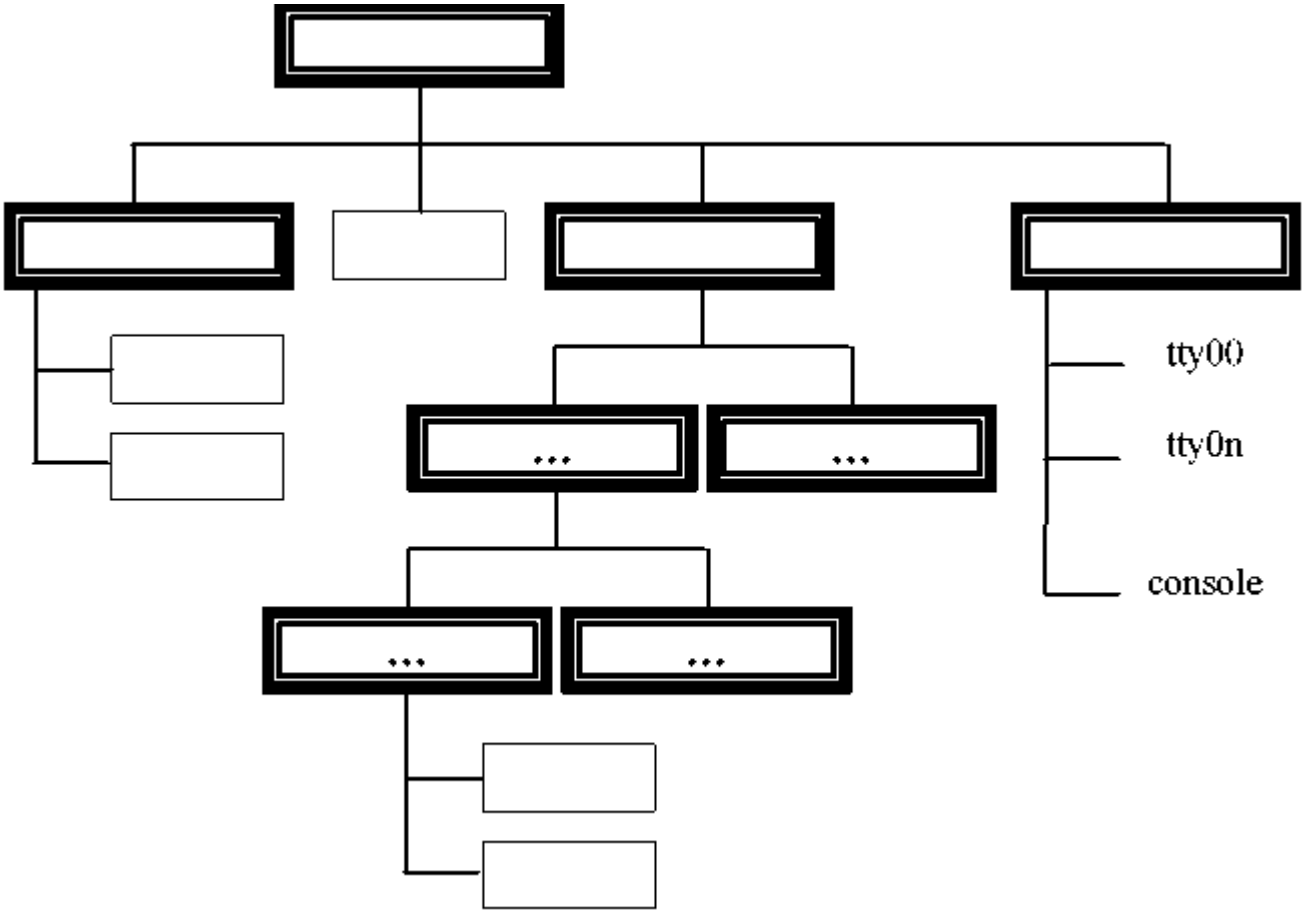

Рисунок 2. Схема файловой системы ОС Unix

Управление доступом к системе. При включении пользователя в число абонентов ему выдается регистрационное имя (идентификатор) для входа в систему и пароль, который служит для подтверждения идентификатора пользователя. В отдельных версиях ОС Unix, помимо идентификатора и пароля, требуется ввод номера телефона, с которого выполняется подключение к системе. Администратор системы и пользователь могут изменить пароль командой passwd. При вводе этой команды ОС запрашивает ввод текущего пароля, а затем требует ввести новый пароль. Если предложенный пароль не удовлетворяет требования системы, то запрос на ввод пароля может быть повторен. Если предложенный пароль удовлетворителен, ОС просит ввести его снова, чтобы убедиться в корректности ввода пароля.

Пользователи, которым разрешен вход в систему, перечислены в учетном файле пользователей /etc/passwd. Этот текстовый файл содержит следующие данные: имя пользователя, зашифрованный пароль, идентификатор пользователя, идентификатор группы, начальный текущий каталог и имя исполняемого файла, используемого в качестве интерпретатора команд. Пароль шифруется, как

правило, с использованием DES-алгоритма.

Управление доступом к данным. Операционная система Unix поддерживает для любого файла комплекс характеристик, определяющих санкционированность доступа, тип файла, его размер и точное местоположение на диске. При каждом обращении к файлу система проверяет право пользоваться им. Операционная система Unix допускает выполнение трех типов операций над файлами: чтение, запись и выполнение.

Чтение файла означает, что доступно его содержимое, а запись – что возможны изменения содержимого файла. Выполнение приводит либо к загрузке файла в ОП, либо к выполнению содержащихся в файле команд системного монитора Shell. Разрешение на выполнение каталога означает, что в нем допустим поиск с целью формирования полного имени на пути к файлу. Любой из файлов в ОС Unix имеет определенного владельца и привязан к некоторой группе. Файл наследует их от процесса, создавшего файл. Пользователь и группа, идентификаторы которых связаны с файлом, считаются его владельцами.

Идентификаторы пользователя и группы, связанные с процессом, определяют его права при доступе к файлам. По отношению к конкретному файлу все процессы делятся на три категории:

1. владелец файла (процессы, имевшие идентификатор пользователя, совпадающий с идентификатором владельца файла);
2. члены группы владельца файла (процессы, имеющие идентификатор группы, совпадающий с идентификатором группы, которой принадлежит файл);
3. прочие (процессы, не попавшие в первые две категории).

Владелец файла обладает одними привилегиями на доступ к нему, члены группы, в которую входит файл – другими, все остальные пользователи – третьими. Каждый файл содержит код защиты, который присваивается файлу при его создании. Код защиты располагается в индексном дескрипторе файла и содержит десять символов, причем первый символ определяет тип файла, а последующие девять – право на доступ к нему. Три вида операций (чтение, запись и выполнение) и три категории (уровни привилегий на доступ: владельцев, групп и прочих пользователей) дают в совокупности девять возможных вариантов разрешений или запретов на доступ к файлу. Первые три символа определяют возможности чтения (r), записи (w) и выполнения (e) на уровне владельца, следующие три – на уровне группы, в которую входит владелец, и последние три – на уровне остальных

пользователей. Наличие символов r, w и e указывает на соответствующее разрешение.

Если процесс требует доступа к файлу, то сначала определяется категория, в которую по отношению к этому файлу он попадает. Затем из кода защиты выбираются те три символа, которые соответствуют данной категории, и выполняется проверка: разрешен ли процессу требуемый доступ. Если доступ не разрешен, системный вызов, посредством которого процесс сделал запрос на доступ, отвергается ядром ОС.

По соглашению, принятому в ОС Unix, привилегированный пользователь имеет идентификатор, равный нулю. Процесс, с которым связан нулевой идентификатор пользователя, считается привилегированным. Независимо от кода защиты файла привилегированный процесс имеет право доступа к файлу для чтения и записи. Если в коде защиты хотя бы одной категории пользователей (процессов) есть разрешение на выполнение файла, привилегированный процесс тоже имеет право выполнять этот файл.

С помощью специальных команд владелец файла (привилегированный пользователь) может изменять распределение привилегий. Команда Change mode позволяет изменить код защиты, команда Change owner меняет право на владение файлом, а команда Change group – принадлежность к той или иной группе. Пользователь может изменять режимы доступа только для файлов, которыми он владеет.

Защита хранимых данных. Для защиты хранимых данных в составе ОС Unix имеется утилита crypt, которая читает данные со стандартного ввода, шифрует их и направляет на стандартный вывод. Шифрование применяется при необходимости предоставления абсолютного права владения файлом.

Восстановление файловой системы. Операционная система Unix поддерживает три основных набора утилит копирования: программы volcopy/labelit, dump/restor и cpio. Программа volcopy целиком переписывает файловую систему, проверяя с помощью программы labelit соответствие меток требуемых томов. Программа dump обеспечивает копирование лишь тех файлов, которые были записаны позднее определенной даты (защита накоплением). Программа restor может анализировать данные, созданные программой dump, и восстанавливать отдельные файлы или всю файловую систему полностью. Программа cpio предназначена для создания одного большого файла, содержащего образ всей файловой системы или какой-

либо ее части.

Для восстановления поврежденной, например, в результате сбоев в работе аппаратуры файловой системы используются программы fsck и fsdb.

За сохранность файловой системы, адаптацию программного обеспечения к конкретным условиям эксплуатации, периодическое копирование пользовательских файлов, восстановление потерянных данных и другие операции ответственность возложена на администратора системы.

Усложненное управление доступом. В составе утилит ОС Unix находится утилита cron, которая предоставляет возможность запускать пользовательские программы в определенные моменты (промежутки) времени и, соответственно, ввести временные параметры для ограничения доступа пользователей. Для управления доступом в ОС Unix также применяется разрешение установки идентификатора владельца. Такое разрешение дает возможность получить привилегии владельца файла на время выполнения соответствующей программы. Владелец файлов может установить режим, в котором другие пользователи имеют возможность назначать собственные идентификаторы режима.

Доступ, основанный на полномочиях, использует соответствие меток. Для этого вводятся метки объектов (файлов) и субъектов (процессов), а также понятия доминанты и равенства меток (для выражения отношения между метками). Создаваемый файл наследует метку от создавшего его процесса. Вводятся соотношения, определяющие права процессов по отношению к файлам. Интерфейс дискретного доступа существенно детализирует имеющиеся механизмы защиты ОС Unix.

Вводимые средства можно разделить на следующие группы:

1. работа со списками доступа при дискретной защите;
2. проверка права доступа;
3. управление доступом на основе полномочий;
4. работа привилегированных пользователей.

В рамках проекта Posix создан интерфейс системного администратора. Указанный интерфейс определяет объекты и множества действий, которые можно выполнить

над объектами. В качестве классов субъектов и объектов предложены: пользователь, группа пользователей, устройство, файловая система, процесс, очередь, вход в очередь, машина, система, администратор, программное обеспечение и др. Определены атрибуты таких классов, операции над классами и события, которые могут с ними происходить.

Windows NT/2000/XP

С момента выхода версии 3.1 осенью 1993 года в Windows NT гарантировалось соответствие уровню безопасности C2. В настоящее время (точнее, в 1999 г.) сертифицирована версия NT 4 с Service Pack 6a с использованием файловой системы NTFS в автономной и сетевой конфигурации. Следует помнить, что этот уровень безопасности не подразумевает защиту информации, передаваемой по сети, и не гарантирует защищенности от физического доступа.

Компоненты защиты NT частично встроены в ядро, а частично реализуются подсистемой защиты. Подсистема защиты контролирует доступ и учетную информацию. Кроме того, Windows NT имеет встроенные средства, такие как поддержка резервных копий данных и управление источникамибесперебойного питания, которые не требуются "Оранжевой книгой", но в целом повышают общий уровень безопасности.

ОС Windows 2000 сертифицирована по стандарту Common Criteria. В дальнейшем линейку продуктов Windows NT/2000/XP, изготовленных по технологии NT, будем называть просто Windows NT.

Ключевая цель системы защиты Windows NT - следить за тем, кто и к каким объектам осуществляет доступ. Система защиты хранит информацию, относящуюся к безопасности для каждого пользователя, группы пользователей и объекта. Единообразие контроля доступа к различным объектам (процессам, файлам, семафорам и др.) обеспечивается тем, что с каждым процессом связан маркер доступа, а с каждым объектом - дескриптор защиты. Маркер доступа в качестве параметра имеет идентификатор пользователя, а дескриптор защиты - списки прав доступа.

ОС может контролировать попытки доступа, которые производятся процессами прямо или косвенно инициированными пользователем.

Windows NT отслеживает и контролирует доступ как к объектам, которые пользователь может видеть посредством интерфейса (такие, как файлы и принтеры), так и к объектам, которые пользователь не может видеть (например, процессы и именованные каналы). Любопытно, что, помимо разрешающих записей, списки прав доступа содержат и запрещающие записи, чтобы пользователь, которому доступ к какому-либо объекту запрещен, не смог получить его как член какой-либо группы, которой этот доступ предоставлен.

Система защиты ОС Windows NT состоит из следующих компонентов:

- Процедуры регистрации (Logon Processes), которые обрабатывают запросы пользователей на вход в систему. Они включают в себя начальную интерактивную процедуру, отображающую начальный диалог с пользователем на экране и удаленные процедуры входа, которые позволяют удаленным пользователям получить доступ с рабочей станции сети к серверным процессам Windows NT;
- Подсистемы локальной авторизации (LocalSecurity Authority, LSA), которая гарантирует, что пользователь имеет разрешение на доступ в систему. Этот компонент - центральный для системы защиты Windows NT. Он порождает маркеры доступа, управляет локальной политикой безопасности и предоставляет интерактивным пользователям аутентификационные услуги. LSA также контролирует политику аудита и ведет журнал, в котором сохраняются сообщения, порождаемые диспетчером доступа;
- Менеджера учета (Security Account Manager, SAM), который управляет базой данных учета пользователей. Эта база данных содержит информацию обо всех пользователях и группах пользователей. SAM предоставляет услуги по легализации пользователей, применяющиеся в LSA;
- Диспетчера доступа (Security Reference Monitor, SRM), который проверяет, имеет ли пользователь право на доступ к объекту и на выполнение тех действий, которые он пытается совершить. Этот компонент обеспечивает легализацию доступа и политику аудита, определяемые LSA. Он предоставляет услуги для программ супервизорного и пользовательского режимов, для того чтобы гарантировать, что пользователи и процессы, осуществляющие попытки доступа к объекту, имеют необходимые права. Данный компонент также порождает сообщения службы аудита, когда это необходимо.

Microsoft Windows NT - относительно новая ОС, которая была спроектирована для поддержки разнообразных защитных механизмов, от минимальных до С2, и

безопасность которой наиболее продумана. Дефолтный уровень называется минимальным, но он легко может быть доведен системным администратором до желаемого уровня.

Заключение

Стремительное развитие информационных технологий привело к формированию информационной среды, оказывающей влияние на все сферы человеческой деятельности. Однако с развитием информационных технологий возникают и стремительно растут риски, связанные с их использованием, появляются совершенно новые угрозы, с последствиями, от реализации которых человечество раньше не сталкивалось.

Одним из главных инструментов для реализации конкретных информационных технологий являются информационные системы, задача обеспечения безопасности которых является приоритетной, так как от сохранения конфиденциальности, целостности и доступности информационных ресурсов зависит результат деятельности информационных систем.

Операционная система является важнейшим программным компонентом любой вычислительной машины, поэтому от уровня реализации политики безопасности в каждой конкретной операционной системе во многом зависит и общая безопасность информационной системы.

В связи с этим знания в области современных методов и средств обеспечения безопасности операционных систем являются необходимым условием для формирования специалиста по информационной безопасности.

Список используемой литературы:

1. Основы сетей передачи данных, В.Г. Олифер, Н.А. Олифер, 2005, 176 с, ISBN 5-9556-0035-3.

2. Основы информационной безопасности, 2-е издание, В.А. Галатенко, 2004, 264 с, ISBN 5-9556-0015-9.

3. Основы микропроцессорной техники, 2-е издание, Ю.В. Новиков, П.К. Скоробогатов, 2004, 440 с, ISBN 5-9556-0016-7.

4. Стандарты информационной безопасности, В.А. Галатенко, 2004, 328 с, ISBN 5-9556-0007-8.

5. Основы операционных систем, 2-е издание, В.Е. Карпов, К.А. Коньков, 2004, 536 с, ISBN 5-9556-0044-2.

6. Архитектуры и топологии многопроцессорных вычислительных систем, А.В. Богданов, В.В. Корхов, В.В. Мареев, Е.Н. Станкова, 2004, 176 с, ISBN 5-9556-0018-3.
7. Операционная система UNIX, Г.В. Курячий, 2004, 320 с, ISBN 5-9556-0019-1.

8. Основы сетевой безопасности: криптографические алгоритмы и протоколы взаимодействия, О.Р. Лапонина, 2005, 608 с, ISBN 5-9556-0020-5.

9. Основы локальных сетей, Ю.В. Новиков, СВ. Кондратенко, 2005, 360 с, ISBN 5-9556-0032-9.

10. Операционная система Linux, Г.В. Курячий, К.А. Маслинский, 2005, 392 с, ISBN 5-9556-0029-9.

11. Проектирование информационных систем, В.И. Грекул и др., 2005, 296 с, ISBN 5-9556-0033-7.