**Средства безопасности операционных систем**

***Внешние условия, требующие принятия дополнительных мер безопасности***

***Угрозы***

Безопасность информационных систем разбита на три компонента: конфиденциальность, целостность и доступность (Confidentiality, Integrity, Availability). Позже было решено, что трех основных свойств для всех возможных сценариев недостаточно, и были добавлены дополнительные свойства, такие как аутентичность, идентифицируемость, неотвергаемость, закрытость и др.

* Конфиденциальность – направлено на сохранение секретности данных. Точнее говоря, если владелец неких данных решил, что эти данные могут быть доступны строго определенному кругу лиц, система должна гарантировать невозможность допуска к данным лиц, не имеющих на это права. Как минимум владелец должен иметь возможность определить, кто и что может просматривать, а система должна обеспечить выполнение этих требований, касающихся в идеале отдельных файлов.
* Целостность - означает, что пользователи, не обладающие необходимыми правами, не должны иметь возможности изменять какие-либо данные без разрешения их владельцев. Изменение данных включает в себя не только внесение в них изменений, но и их удаление или добавление в них ложных данных. Если система не может гарантировать, что заложенные в нее данные не будут подвергаться изменениям, пока владелец не решит их изменить, то она утратит свою роль хранилища данных.
* Доступность - никто не может нарушить работу системы и вывести ее из строя. Атаки, вызывающие отказ от обслуживания (denial of service, (DOS)), приобретают все более распространенный характер. Например, если компьютер работает в роли интернет-сервера, то постоянное забрасывание его запросами может лишить его работоспособности, отвлекая все рабочее время его центрального процессора на изучение и отклонение входящих запросов. Если на обработку входящего запроса на чтение веб-страницы уходит, скажем, 100 мкс, то любой, кто сможет отправлять 10 000 запросов в секунду, способен вывести сервер из строя.

Системы находятся под постоянной угрозой, исходящей от взломщиков. Существует множество способов, воспользовавшись которыми посторонний человек может атаковать систему. Многие атаки в наши дни поддерживаются весьма совершенными инструментальными средствами и службами. Например, взломщик может привязаться к трафику локальной сети и нарушить конфиденциальность информации, особенно если в протоколе обмена данными не используется шифрование. Также злоумышленник может взломать систему управления базами данных и удалить или изменить некоторые записи, нарушив целостность базы. И наконец, хитроумно размещенные атаки, вызывающие отказ от обслуживания, могут нарушить доступность одной или нескольких компьютерных систем.

Многие инструменты используются как тестировщиками безопасности, так и взломщиками. Например, nmap (посредством сканирования портов (portscan) выявляет сетевые услуги, предлагаемые компьютерной системой) или dsniff (предлагает различные способы отслеживания сетевого трафика и перенаправления сетевых пакетов).

Киберпреступники также предлагают большой выбор служб для своих грязных целей. Основная часть преступных действий в Интернете построена на инфраструктурах, известных как ботнеты, или бот-сети, которые состоят из множества зараженных компьютеров.

***Стоит задать вопрос: можно ли создать полностью защищённую систему?***

Сегодняшние системы не являются защищенными, но пользователи не собираются от них отказываться.

Единственный известный способ создать защищенную систему заключается в поддержании ее простоты. Функциональные возможности являются врагом безопасности. Пользователям хочется все больше функциональных возможностей, еще более широких и более качественных, следовательно, сложность систем повышается, растёт объём кода, что приводит к росту числа дефектов и ошибок в безопасности системы.

***Высоконадежная вычислительная база***

Надёжные системы – системы, имеющие официально установленные требования к безопасности и соблюдающие эти требования. В основе такой находится минимальная база TCB (Trusted Computing Base — высоконадежная вычислительная база), состоящая из аппаратного и программного обеспечения, необходимого для принудительного выполнения всех мер безопасности. Если высоконадежная вычислительная база работает в соответствии с техническими условиями, безопасность системы не может быть нарушена ни при каких других неблагоприятных обстоятельствах.

TCB состоит в основном из аппаратного обеспечения, части ядра операционной системы и большей части или всех пользовательских программ, обладающих полномочиями привилегированного пользователя. Функции ОС, которые являются частью TLB: создание процесса, переключение процессов, управление отображением памяти, а также часть управления файлами и вводом-выводом данных. Одной из целей некоторых современных исследований в сфере безопасности является сокращение объема высоконадежной вычислительной базы с нескольких миллионов строк кода до всего лишь десятков тысяч.

Важную часть высоконадежной вычислительной базы составляет монитор обращений - он принимает все системные вызовы, имеющие отношение к безопасности.

***Доступ к ресурсам***

Безопасности проще достичь, если есть четкая модель того, что должно быть защищено и кому и что разрешено делать.

***Домен защиты***

Компьютерная система содержит множество ресурсов или объектов, нуждающихся в защите. У каждого объекта есть уникальное имя, по которому к нему можно обращаться, и конечный набор операций, которые процессы могут выполнять в отношении этого объекта. Очевидно, что нужен способ запрещения процессам доступа к тем объектам, к которым у них нет прав доступа. Этот механизм должен также предоставлять возможность при необходимости ограничивать процессы поднабором разрешенных операций.

Домен - множество пар типа объект – право доступа. Каждая пара определяет объект и некоторое подмножество операций, которые могут быть выполнены в отношении этого объекта. Права доступа означают в данном контексте разрешение на выполнение той или иной операции. Зачастую домен соотносится с отдельным пользователем, сообщая о том, что может, а что не может сделать этот пользователь, но он может также иметь и более общий характер, распространяясь не только на отдельного пользователя. Распределение объектов по доменам зависит от особенностей того, кому и о чем нужно знать. Одним из фундаментальных понятий является принцип минимальных полномочий, или принцип необходимого знания. В общем, безопасность проще соблюсти, когда у каждого домена имеется минимум объектов и привилегий для работы с ними и нет ничего лишнего.

В любой момент времени каждый процесс работает в каком-нибудь домене защиты. Во время работы процессы могут также переключаться с одного домена на другой.

***Перечни возможностей***

Для определения доступа к объектам используются матрицы защиты, в которых строки - домены, а колонки — объекты. Практическое применение нашли два метода: хранение матрицы по столбцам или по строкам, а затем хранение только заполненных элементов.

В первом методе с каждым объектом ассоциируется упорядоченный список ACL (Access Control List — список управления доступом), содержащий все домены, которым разрешен доступ к данному объекту, а также тип доступа. У каждого файла есть связанный с ним ACL.

Во втором методе с каждым процессом будет связан список объектов, к которым может быть получен доступ, а также информация о том, какие операции разрешены с каждым объектом, иными словами, с ним будет связан его домен защиты. Такой список называется перечнем возможностей (capability list, C-list), а его элементы — возможностями. Перечни возможностей сами являются объектами, и на них могут указывать другие перечни возможностей, облегчая, таким образом, совместное использование субдоменов.

Очевидно, что перечни возможностей должны быть защищены от подделки со стороны пользователей. Есть три способа защиты:

Для первого способа требуется теговая архитектура (аппаратная конструкция, в которой каждое слово памяти имеет дополнительный (теговый) бит, сообщающий, содержит данное слово памяти элемент перечня возможностей или нет). Теговый бит не используется в обычных командах, к которым относятся арифметические действия, сравнения или им подобные команды, и может быть изменен только программами, работающими в режиме ядра (то есть операционной системой).

Второй способ заключается в хранении перечня возможностей внутри операционной системы. При этом к элементам перечня возможностей можно обращаться по их позиции в перечне.

Третий способ заключается в хранении перечня возможностей в пространстве пользователя, но в зашифрованном виде, так чтобы пользователь не смог подделать эту информацию. Такой подход особенно хорош для распределенных систем. Когда клиентский процесс отправляет сообщение удаленному серверу, например, файловому серверу, чтобы для него был создан объект, сервер создает объект, а также генерирует сопутствующее ему большое случайное число, используемое как контрольное поле. В таблице файлов сервера для объекта резервируется запись, в которой контрольное поле хранится наряду с адресами дисковых блоков. В понятиях системы UNIX контрольное поле хранится на сервере в i-узле. Оно не отправляется пользователю и вообще никогда не попадает в сеть. Затем сервер формирует и передает пользователю элемент перечня возможностей. Возвращаемый пользователю элемент перечня возможностей содержит идентификатор сервера, номер объекта (индекс в таблицах сервера, по сути, — номер i-узла), а также права доступа, хранящиеся в виде битового массива. У только что созданного объекта все биты прав доступа установлены в единицу, поскольку само собой разумеется, что владелец может делать все, что ему заблагорассудится. Последнее поле представляет собой конкатенацию объекта, прав и контрольного поля, пропущенную через криптостойкую одностороннюю функцию f (y = f(x)), позволяющую при наличии x легко получить y, но обратный процесс — при наличии y получить x — невозможен. Достаточно знать, что при наличии хорошей односторонней функции даже самый настойчивый взломщик не сможет разгадать содержимое контрольного поля, даже если ему будет известно содержимое всех остальных полей возможности. Когда пользователь хочет получить доступ к объекту, он отправляет элемент перечня возможностей серверу в своем запросе. Сервер извлекает из него номер объекта и использует его в качестве индекса для поиска объекта в своих таблицах. Затем он вычисляет f(Objects, Rights, Check), взяв первые два параметра для этой функции из присланного ему пользователем элемента перечня возможностей, а третий параметр — из своих таблиц. Если результат совпадает с четвертым полем элемента перечня возможностей, запрос удовлетворяется, в противном случае он отклоняется. Если пользователь пытается получить доступ к чужому объекту, он не сможет подделать четвертое поле, так как не знает значения контрольного поля, и его запрос будет отклонен. Пользователь может потребовать у сервера создания элемента перечня возможностей с меньшими правами, например, позволяющего только читать объект. Первым делом сервер проверяет достоверность элемента перечня возможностей. Если проверка проходит успешно, он вычисляет f(Objects, New\_rights, Check) и создает новый элемент перечня возможностей, помещая полученное значение в четвертое поле. Обратите внимание на то, что при этом используется исходное значение контрольного поля Check, поскольку от него зависят другие, неизмененные элементы перечня возможностей. Этот новый элемент перечня возможностей отправляется обратно запрашивающему процессу. Теперь пользователь может передать его своему другу, отправив, к примеру, этот элемент в сообщении. Если друг попытается включить какой-нибудь бит прав, который должен быть выключен, сервер обнаружит это при попытке воспользоваться возможностью, поскольку значение функции f не будет соответствовать сфальсифицированному полю прав. Поскольку другу не известно значение поля проверки достоверности, он не сможет подделать элемент перечня возможностей так, чтобы тот соответствовал фальсифицированным битам прав.

Вдобавок к специфическим правам, зависящим от характера объекта, таким как право на чтение и исполнение, элементы перечня возможностей (как содержащиеся в ядре, так и защищенные шифрованием) содержат, как правило, общие права (generic rights), применимые ко всем объектам.  
Примерами общих прав являются:

* копирование элемента возможностей — создание нового элемента для того же самого объекта;
* копирование объекта — создание дубликата объекта с новым элементом возможностей;
* удаление элемента возможностей — удаление записи из перечня возможностей без оказания какого-либо влияния на объект;
* удаление объекта — полное удаление объекта и связанных с ним возможностей.

Версии, управляемой на уровне ядра, отмена доступа к объекту сильно затруднена. Системе трудно найти все незадействованные элементы возможностей для какого-нибудь объекта, чтобы их отобрать, поскольку они могут храниться в перечнях возможностей, размещенных в разных местах по всему диску. Один из подходов заключается в том, чтобы каждая возможность указывала на косвенный объект, а не на сам объект как таковой. При наличии косвенного объекта, указывающего на реальный объект, система всегда может разорвать эту связь, аннулируя возможности.

***Формальные модели систем безопасности***

Матрицы защиты часто подвергаются изменениям при создании новых объектов, уничтожении старых объектов и принятии решения владельцами о расширении или сужении круга тех, кто пользуется их объектами.

Есть шесть элементарных операций в матрице защиты, которые могут послужить основой модели любой системы защиты: *создание объекта, удаление объекта, создание домена, удаление домена, вставка права и удаление права.* Эти операции могли объединяться в команды защиты, которыми пользовательские программы могут изменять матрицу защиты. Они могут и не выполнять примитивы непосредственно. Например, у системы может быть команда для создания нового файла, которая будет проверять, не существует ли уже такой файл, и, если его не существует, создавать новый объект и предоставлять владельцу все права на этот объект.

В любой момент времени согласно матрице определяется, что процесс в любом домене может делать, а не на что ему были предоставлены права. Стоит сказать, что матрица ведется системой, а предоставление прав должно относиться к политике управления.  
Пример: рассмотрим простую систему, где домены соотносятся с пользователями. Допустим один из пользователей нашел способ выдачи команды на изменение матрицы. Следовательно, он совершать действия, на которые у него нет полномочий (несанкционированные), т.е. этот пользователь находится в несанкционированном состоянии. Множество всех возможных матриц должно быть разделено на два разобщенных подмножества: подмножество санкционированных состояний и подмножество несанкционированных состояний.

***Многоуровневая защита***

Многоуровневая система безопасности – схема, в которой несколько уровней безопасности.

Разграничительным управлением доступом – политика поведения ОС, при которой отдельным пользователям позволено определять, кто может читать и записывать их файлы и другие объекты.

Для некоторых сред (таких как военные организации, корпоративные патентные отделы и лечебные учреждения) требуется принудительное управление доступом –гарантия, что установленная политика безопасности реализуется системой. Такое управление доступом регулирует поток информации, гарантируя отсутствие непредвиденной ее утечки.

Существует несколько моделей многоуровневой защиты:

***Модель Белла — Лападулы***

Имеет наиболее широкое распространение. Была разработана для обеспечения военной системы безопасности, но она подходит и для других организаций. Объекты (документы) обладают грифом секретности, например: «несекретный», «для служебного пользования», «секретный» и «совершенно секретный». Пользователи получают форму допуска, определяющую, какие документы им разрешено просматривать. Пользовательский процесс приобретает уровень безопасности этого пользователя.

Данная модель имеет правила организации информационного потока:

* Простое свойство безопасности - процесс, запущенный на уровне безопасности, может проводить операцию чтения только в отношении объектов своего или более низкого уровня. (Генерал может читать документы лейтенанта, но лейтенант не может читать генеральские документы.)
* Свойство - процесс, работающий на определённом уровне безопасности, может вести запись только в объекты своего или более высокого уровня. (Лейтенант может добавить сообщение в генеральский почтовый ящик, докладывая обо всем, что ему известно, но генерал не может добавить сообщение в лейтенантский почтовый ящик, сообщая о том, что известно ему, поскольку генерал может быть ознакомлен с совершенно секретными документами, содержание которых не должно доводиться до лейтенанта.)

***Модель Биба***

Для гражданский (коммерческий и т.п) целей.   
Свойства данной модели (противоположны модели Белла-Лападулы):

* Простое свойство целостности - процесс, работающий на каком-то уровне безопасности, может записывать только в объекты своего или более низкого уровня (никакой записи наверх).
* Свойство целостности - процесс, работающий на каком-то уровне безопасности, может читать из объектов своего или более высокого уровня (никакого чтения из нижних уровней).

***Тайные каналы***

На практике, даже в системе, в основу которой заложена надлежащая модель безопасности, прошедшая апробацию и реализованная по всем правилам, могут все же происходить утечки секретных сведений.  
Модель Лэмпсона: три процесса – клиент, сервер, сотрудник. Клиент, желающий, чтобы второй процесс, сервер, выполнил некоторое задание. Клиент и сервер не вызывают друг у друга доверия. (Клиенты опасаются, что сервер тайно запишет их финансовую информацию, к примеру сведения о том, кто сколько зарабатывает, а затем продаст эту информацию. Сервер опасается, что клиенты попытаются украсть ценную программу подсчета налогов.) Третий процесс является сотрудником, вступающим в сговор с сервером именно для того, чтобы похитить конфиденциальные данные клиента. Сотрудник и сервер обычно принадлежат одному и тому же лицу.

С точки зрения разработчика системы, задача заключается в инкапсуляции, или ограждении сервера таким образом, чтобы он не мог передать информацию сотруднику.

Для утечки информации могут использоваться менее заметные каналы. Например, сервер может передавать следующий поток двоичных битов. Для отправки единичного бита он может заниматься вычислением на полную мощность в течение определенного интервала времени, а для передачи нулевого бита он на тот же период времени может прекращать вычислительный процесс. Сотрудник может попытаться обнаружить поток битов за счет тщательного отслеживания времени отклика этого процесса. Подобные каналы связи называют тайными. Конечно, тайный канал относится к зашумленным каналам, содержащим множество посторонней информации, но информация может надежно отправляться по зашумленному каналу за счет использования кода с коррекцией ошибок (код Хемминга).   
Существует множество способов установить тайный канал. Очевидно, ни одна из моделей защиты, основанной на матрице объектов и доменов, не в состоянии перекрыть подобный канал утечки информации. Как пишет Танненбаум: «Само обнаружение всех тайных каналов, не говоря уже об их блокировке, и то является почти безнадежным делом.»

***Стеганография***

Полный сообщения составляет N байт. Все это сначала было сжато до размера около 274 Кбайт (столько информации можно разместить в младших разрядах каждого цветового значения пикселя изображения размера 1024x768 пикселей) с использованием стандартного алгоритма сжатия. Затем сжатый результат был зашифрован и вставлен в биты младшего разряда каждой цветовой составляющей. Можно убедиться в том, что присутствие информации абсолютно не заметно. Ее присутствие не заметно также на большой полноцветной версии фотографии. Человеческий глаз не в состоянии так просто отличить 7-разрядный цвет от 8-разрядного. Как только этот файл с изображением благополучно пройдет цензуру (которая осуществляется «вручную»), получатель просто извлечет все биты младших разрядов, применит алгоритмы дешифрации и распаковки и восстановит исходные 734 891 байт. Такое сокрытие присутствия информации называется стеганографией (steganography, от греческого слова «тайнопись»). Данный метод может использоваться также для защиты водяными знаками музыки, фильмов и т. д.