УДК 681.391.1

МОДЕЛЬ ПОЛИТИКИ БЕЗОПАСНОСТИ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

Д. А. Подкорытов,
аспирант
Санкт-Петербургский государственный университет аэрокосмического приборостроения

В статье дается обзор применяемых моделей политики безопасности. Показано, что традиционный способ назначения меток безопасности для объектов системы и прав доступа к ним пользователя не обеспечивает гарантированного режима безопасности информации и может привести к ее компрометации. Предлагается принцип построения подсистемы безопасности операционной системы, позволяющий решить эту проблему, а также значительно снизить риск от человеческого фактора при настройке прав доступа к объектам системы и эксплуатации такой системы. Обосновывается целесообразность структурного разделения модели политики безопасности на три подсистемы.

This paper starts with a review of security policies for computing systems. It is shown that the traditional way of assigning security labels and user access rights to the objects in a system does not provide a guaranteed mode of information security. To solve this problem, we propose a construction of the security subsystem in the operating system which will considerably reduce the human factor risk in the assignment of access rights to system's objects. Also, we justify the structural subdivision of our security model into three subsystems.

Обзор применяемых моделей политики безопасности

Для разграничения доступа в современных многопользовательских операционных системах широко используется принцип назначения меток безопасности для каждого объекта системы. Кроме этого, применяется принцип объединения пользователей в группы по классу решаемых задач и необходимости предоставления тех или иных полномочий. Наличие групп доступа существенно упрощает администрирование системы.

Каждому пользователю или группе (субъектам), а также каждому ресурсу (объекту) ставятся в соответствие метки – идентификаторы, позволяющие управлять доступом субъектов к объектам согласно политике безопасности.

В настоящее время под управлением метками обычно понимается выполнение этой операции вручную в командном режиме, администратором системы для каждого субъекта или объекта, согласно собственному опыту и исходя из политики безопасности организации, эксплуатирующей систему.

На взгляд автора, в современных моделях безопасности не уделяется должного внимания политике управления метками, так как управление метками осуществляется в ручном (командном), не автоматическом режиме и является более организационным, нежели техническим вопросом эксплуатации.

Тем не менее, возможен принцип, основанный на автоматическом или полуавтоматическом (интерактивном) назначении меток, например, в зависимости от содержимого документов.

В этом случае метки безопасности, присваиваемые субъектам и объектам, вычисляются исходя из некоторого набора правил, а не назначаются командами администратора системы.

Современные информационные системы имеют в своем составе сложные объекты, обладающие множеством возможных состояний и переходов и некоторыми реактивными характеристиками. Субъекты, принимающие участие в деятельности такой системы, обладают также некоторой динамикой, поскольку взаимодействие субъектов и объектов разворачивается во времени. Это говорит о том, что при оценке безопасности информационной системы необходимо учитывать и динамические свойства системы, и субъектов, с ней взаимодействующих. Пренебрежение учетом динамических свойств системы приводит в некоторых современных системах к их уязвимости для DoS атак и делает возможным существование скрытых по времени каналов.

Отличие предлагаемого принципа от традиционного принципа предоставления полномочий хочется продемонстрировать примерами правил доступа.

В традиционной политике предоставления доступа правила доступа формулируются исходя из статического состояния системы: «Через эту дверь аутентифицированный субъект проходить может». По мнению автора, необходимо учитывать и динамику: «Через эту дверь проходить можно, но не более одного раза в секунду для каждого субъекта, при

условии, что информационная система находится в некотором состоянии S».

Исходя из вышесказанного, предлагается структурно разделять модель политики безопасности на модели политик управления метками, предоставления доступа и существования объектов и субъектов.

Модель политики управления метками – это набор правил, согласно которым динамически или итеративно (с участием администратора) вычисляется соответствие того или иного объекта или субъекта идентификатору его уровня безопасности (так называемой метке). Решение о соответствии объекта или субъекта той или иной метке принимается исходя из этих правил и его текущего состояния всякий раз перед попыткой взаимодействия вида субъект–объект. Частный случай модели политики управления метками – это традиционное назначение меток безопасности субъектов и объектов только в командном режиме. В этом случае метки постоянны во времени и не зависят от содержимого объекта или деятельности субъекта.

Модель политики предоставления доступа – это набор правил, согласно которым предоставляется доступ субъектов к объектам в зависимости от их динамически вычисленных или назначенных меток-идентификаторов уровня безопасности.

Модель политики существования – это набор правил, действующих на протяжении всего времени присутствия субъекта в системе и на протяжении всего жизненного цикла объекта системы и учитывающих динамику их поведения. Субъекты и объекты должны следовать этим правилам для того, чтобы иметь право на существование в системе. Объекты и субъекты, чья деятельность выходит за рамки правил этой модели, блокируются.

Например, метка субъекта, ведущего себя подозрительно, может быть изменена и, таким образом, субъект будет понижен в привилегиях доступа к системе.

В настоящее время известны несколько моделей безопасности. Согласно введенной терминологии, они в основном относятся к политикам предоставления доступа, и лишь некоторые из них отчасти захватывают вопросы управления метками – в смысле предоставления прав субъекту манипулировать метками безопасности вверенных ему в управление субъектов и объектов.

Рассмотрим известные модели политик безопасности информационных систем с тем, чтобы определить, какой политике должна следовать каждая из трех компонент модели.

Принцип дискреционного доступа Харрисона–Руззо–Ульмана [7]

Этот принцип реализует управление доступом объектов к субъектам и контроль за распространением прав доступа. В рамках идеологии этой системы существуют объекты O (ресурсы системы), субъекты S (активная сущность, производящая доступ к информации), множество прав доступа R. В целом информационная база системы – это матрица $O \times S$, содержащая на пересечении столбца и строки право доступа $r[j,i]$ i-го субъекта к j-му объекту. Политика позволяет предоставлять субъекту право манипулировать метками безопасности вверенных ему в управление субъектов и объектов.

Однако Харрисон, Руззо и Ульман доказали, что в общем случае не существует алгоритма, который может для произвольной системы, ее начального состояния и правил доступа решить, является ли система безопасной.

Дискреционная модель доступа Харрисона–Руззо–Ульмана в своей общей постановке не дает гарантий безопасности системы, так как не учитывает безопасности переходов из одного состояния в другое, а также не содержит способа контроля информационных потоков и «троянских коней» [5].

Типизированные матрицы доступа (TAM & MTAM) [14]

Эта политика предоставления доступа развивает дискреционную модель Харрисона–Руззо–Ульмана, дополняя ее концепцией типов, что позволяет несколько смягчить условия, для которых возможно доказательство безопасности системы.

Формальное описание модели TAM содержит:

1. Конечный набор прав доступа $R = \{r_1, ..., r_j\}$.
2. Конечный набор типов $T = \{t_1, ..., t_g\}$.
3. Конечные наборы субъектов и объектов $S = \{s_1, ..., t_n\}$, $O = \{o_1, ..., o_m\}$.
4. Матрицу M, содержащую права доступа субъектов к объектам и ее начальное состояние M_0.
5. Конечный набор команд оперирования над матрицей доступа $C = \{\alpha_1(...), ..., \alpha_k(...)\}$, включающий условия выполнения команд и их интерпретацию в терминах элементарных операций.

Рави Садху введено понятие монотонной TAM (MTAM).

MTAM не содержит немонотонных операций *delete, destroy subject, destroy object* и упрощает процесс доказательства безопасности матрицы доступа, но сложность доказательства безопасности такой модели является NP полной задачей.

Рассматривается еще один частный случай – тернарная MTAM (TMTAM) с системой команд C, в которой $\alpha_k(...)$ содержит не более трех аргументов. Для TMTAM доказательство безопасности выполняется за полиномиальное время.

Политика безопасности MAC (мандатного контроля доступа Белла и Ла Падулы BLP) [5]

Основная задача этой политики – обеспечение конфиденциальности. Она использует как прототип принципы работы с секретными документами в государственных учреждениях. Основной принцип заключается в том, что существует иерархия нескольких уровней безопасности (совершенно секретно, секретно, для служебного пользования и т. д.).

Объектам присваивается метка-идентификатор, показывающая уровень секретности хранимой в них информации. Субъектам также присваивается метка, показывающая их уровень допуска к информации. Используются два основных правила, обеспечивающих конфиденциальность:

1) субъект не может читать данные с верхнего по отношению к нему уровня допуска;

2) субъект не может передавать данные на нижний по отношению к нему уровень допуска.

Эта политика запрещает доступ на запись к объектам системы с более низким уровнем доступа, чем у субъекта. Такая политика надежно предот-

вращает миграцию данных с более высоких уровней секретности на более низкие. К сожалению, лицам, имеющим высокий уровень доступа к секретным документам, часто приходится решать и задачи с меньшим уровнем секретности; также возможна ситуация, в которой необходимо организовать двухсторонний обмен информацией между двумя достаточно высокими уровнями безопасности. Поэтому эту политику применяют в сочетании с политикой дискреционного доступа. На стыке этих политик возможны логические бреши и уязвимости реализаций.

Политика безопасности «Китайская стена» (WC)

Политика предложена Бревером и Нэшем [3] и ориентирована на сферу бизнеса. Эта политика учитывает конфликты бизнес-интересов фирм-конкурентов, пользующихся услугами одних и тех же консультантов. Политика служит для реализации общепринятого в сфере бизнес-консалтинга правила: внешний аналитик имеет возможность получить информацию о любой фирме из секции ее критических бизнес интересов Q_i. Получив однажды эту информацию об одной компании, он не может получить информацию об интересах других компаний в этой критической области Q.

Политика безопасности Кларка и Вильсона [2]

Основная задача этой политики – предохранение от неавторизованных изменений информации авторизованными субъектами. Кларком и Вильсоном введены понятия: детально описанные транзакции (WFT), обеспечивающие целостность обрабатываемой информации, и принцип разделения полномочий.

Из задач, решаемых этой политикой, следует:

1) субъекты могут иметь доступ к объектам только через авторизованные программы;

2) обязанности субъектов и объектов разделяются, возможно выполнение одной функции несколькими разными людьми и представление одним человеком нескольких функций;

3) предусмотрен аудит событий.

В модели описаны две системы безопасности – для военных и для коммерческих целей. В этой политике каждому субъекту и объекту присваиваются иерархические уровни безопасности. Также как и в мандатной политике, BLP объекты с нижних, по отношению к субъекту, уровней безопасности доступны только для чтения, что предотвращает компрометацию информации. Модель используется как формальный базис для построения военных систем компьютерной безопасности.

Модель Biba [5]

Основная задача этой политики – обеспечение целостности информации. Эта политика является, по сути, противоположной политике мандатного доступа. В ней используются два основных правила, обеспечивающих целостность:

1) субъект не может читать данные с нижнего относительно своего уровня допуска;

2) субъект не может передавать данные на верхний по отношению к своему уровень допуска.

В рамках этой политики все операции чтения-записи производятся в пределах одного уровня безопасности. Для миграции данных между уровнями необходимо предусмотреть членство объектов, для которых разрешен обмен информацией в нескольких уровнях безопасности.

Такая политика наиболее проста в реализации и, видимо, поэтому послужила прообразом для построения систем безопасности в современных операционных системах.

Мандатная модель Мак-Лина [12]

Эта политика является расширенной модификацией мандатной политики BLP. Специфика политики заключается в том, что она базируется не на безопасных состояниях, как BLP, а на безопасных переходах. В модели введено понятие «функции безопасного перехода». В рамках модели введены «Уполномоченные субъекты», которым разрешено инициировать функции изменения уровня безопасности у сущностей системы. Политика предусматривает модель совместного доступа при работе субъектов в группах и с общими разделяемыми объектами.

Политика безопасности Диона [6]

Данная политика уделяет внимание как контролю целостности, так и конфиденциальности. Модель Диона обобщает более известные модели безопасности Biba и Bell-LaPadula. В модели Диона с каждым субъектом (объектом) ассоциируется три метки конфиденциальности и три метки целостности информации:

1) абсолютная метка конфиденциальности (ACL) – присваивается объекту во время его создания и остается постоянной в течение всего времени его существования. Обычно это метка пользователя-инициатора процесса;

2) метка конфиденциальности чтения (RCL) – минимальный уровень конфиденциальности, с которого субъекту разрешено чтение;

3) метка конфиденциальности записи (WCL) – минимальный уровень конфиденциальности, на который субъекту разрешена запись;

4) абсолютная метка целостности (AIL);

5) метка целостности чтения (RIL) – минимальный уровень целостности, с которого субъекту разрешено чтение;

6) метка целостности записи (WIL) – минимальный уровень целостности, на который субъекту разрешена запись;

Для каждого объекта s должны выполняться правила:

$$WCL(s)<=ACL(s)<=RCL(s);$$
$$RIL(s)<=AIL(s)<=WIL(s).$$

С каждым объектом ассоциируется также три метки конфиденциальности и три метки целостности информации:

1) абсолютная метка конфиденциальности (ACL) – уровень конфиденциальности хранимой информации;

2) метка конфиденциальности чтения (RCL) – максимальный уровень конфиденциальности, на который могут мигрировать данные объекта;

3) метка конфиденциальности записи (WCL) – минимальный уровень конфиденциальности, с которого может производиться запись в объект;

4) абсолютная метка целостности (AIL);

5) метка целостности чтения (RIL) – минимальный уровень целостности, на который могут мигрировать данные, хранящиеся в объекте;

6) метка целостности записи (WIL) – максимальный уровень целостности, с которого данные могут записываться в объект.

Для каждого объекта *o* должны выполняться правила:

$$WCL(o) <= ACL(o) <= RCL(o);$$
$$RIL(o) <= AIL(o) <= WIL(o).$$

Модель Диона предусматривает возможность организовать однонаправленный канал передачи информации от одного объекта к другому:

$$ACL(o1) <= RCL(s);$$
$$RIL(s) <= AIL(o1);$$
$$WCL(s) <= ACL(o2);$$
$$AIL(o2) <= WIL(s).$$

Эта модель обладает наибольшей полнотой и универсальностью.

Политика описания информационных потоков

Описывает информационные потоки, а не права доступа субъектов к объектам. В рамках этой политики указывается источник информации, приемник и права субъекта. Идея достаточно интересна, так как позволяет более точно, чем, например, дискреционная политика, описывать права доступа к информационным потокам. Данная политика может быть реализована средствами политики Диона и является ее подмножеством. Такой принцип контроля и ограничения информационных потоков в том или ином виде применяется в современных сетевых защитных экранах (*firewall*).

Политика заявок участия в аукционе (SB) [11]

Служит для обеспечения безопасности сделки, производимой по принципам аукциона. Политика использует следующие принципы:

каждый аукцион имеет предопределенные временные рамки;

каждая выставленная заявка не может быть отвергнута аукционом, но может быть снята субъектом, ее выставившим (аукционером);

победителем аукциона является заявка с максимальной ценой;

после выставления заявка не может быть переоценена, даже аукционером;

проигравшие заявки не приводят к потере денег аукционерами;

аукционеры могу участвовать одновременно в нескольких аукционах.

Ролевая политика RC (Role Compatibility) [16]

В рамках этой модели вводятся такие термины, как цели и запросы. Субъекты (процессы) делают запросы на доступ к цели (объекту). Определяются RC-роли и RC-типы, а затем определяется, что может делать та или иная роль с тем или иным типом. Таким образом, создается некоторая абстрактная модель, которая затем привязывается к реальным пользователям, программам и файлам. Независимость модели от реальных субъектов и объектов позволяет производить мгновенную перенастройку политики безопасности быстрым перепривязыванием ролей и/или типов. Кроме того, это очень удобно для создания готовых решений, например, распределение ролей и типов для защиты содержимого страниц Web-узла. Интересной особенностью является возможность запускать программы с ролью, отличной от роли пользователя, производящего запуск. В результате, можно, например, произвести такие настройки, что прямой доступ к диску будут иметь только разрешенные программы, а все остальные пользователи системы (включая администратора) будут лишены такой возможности.

Запросы могут быть самые разнообразные и для их выполнения должны совпадать с правами ACL.

Целями могут быть: ***File*** (файл), ***DIR*** (каталог), ***DEV*** (устройство), ***IPC*** (объект IPC – семафоры, разделяемая память и т. д.), ***SCD*** – Системные данные (имя машины, системный журнал), как правило, только для чтения, ***User*** (пользователи), ***Process*** (процессы), ***None*** (пустой объект), ***FD*** (файловый дескриптор).

Каждый тип цели может иметь свой подтип. Например, файл может быть обычный, секретный или системный. Роль определяет некий класс субъектов, задавая права, которые имеют члены этого класса по отношению к определенным подтипам цели и другим классам. Рассмотрим эти параметры более подробно:

- *Name* – название роли;
- *Role Comp* – совместимые роли, на которые данная роль может переключиться без смены владельца;
- *Admin Roles* – роли, которые данная роль может администрировать;
- *Assign Roles* – роль, которая может назначаться пользователям или процессам;
- *Type comp FD* – здесь указывается, какие ACL-права имеет данная роль при обращении к тому или иному подтипу типа FD;
- *Type comp DEV* – аналогично для типа DEV;
- *Type comp Process* – аналогично для типа Process;
- *Type comp IPC* – аналогично для типа IPC;
- *Type comp SCD* – аналогично для типа SCD;
- *Admin Type* – устаревший параметр, указывающий тип этой роли: Системный Администратор, Ролевой Администратор (Офицер безопасности) или Простой Пользователь (можно вместо него пользоваться первыми четырьмя параметрами);
- *Default FD Create Type* – при создании объекта типа FD будет использован соответствующий подтип, например, может быть роль, создающая только секретные файлы и каталоги. Пользоваться ими смогут только роли с соответствующими ACL-правами;
- *Default Process CreateType* – аналогично для создаваемых (клонируемых) процессов;
- *Default Process Chown Type* – подтип процесса после смены владельца (setuid);
- *Default Process Execute Type* – подтип процесса после запуска;
- *Default IPC Create Type* – подтип новых IPC каналов, семафоров и т. д.

Такое количество настроек для роли позволяет гибко менять систему безопасности, достигая при этом фантастических результатов.

Например, можно настроить систему так, чтобы администратор мог добавлять пользователей, задавать им пароли, удалять их и при этом не мог вручную отредактировать */etc/passwd* или */etc/shadow*. Такой прием может быть полезен и для организации Web-сайта. После установки соответствующих прав на домашний каталог сервера никто кроме него самого не сможет работать с файлами из этого каталога. Даже прорвавшись в систему, злоумышленник не сможет поменять первую страницу сайта.

Угрозы моделям безопасности

Для каждой из представленных в этой статье политик предоставления доступа существуют следующие угрозы:

скрытые каналы – способы передачи информации помимо механизма политики безопасности и авторизации. Известно два основных типа: скрытые каналы по времени и скрытые каналы по данным;

черные ходы, заложенные разработчиками;

вопросы изучения времени реакции системы;

переполнение буферов;

сбои оборудования и программного обеспечения.

Кроме этого, при ручном (командном) способе назначения атрибутов безопасности объектов система имеет и другие недостатки.

1. Субъективный и рутинный характер работ по назначению прав доступа. Решение по назначению атрибутов безопасности принимает человек, который управляет системой. А люди, как известно, иногда ошибаются.

Кроме человеческого фактора, действуют и другие факторы, а именно: фактор времени (администратор может не успеть выполнить необходимую настройку атрибутов безопасности, и далеко не всегда по своей вине – его могут просто не успеть поставить в известность); фактор информированности (администратор может не знать тонких деталей, специфики обрабатываемой пользователем информации и специфики человеческих отношений внутри группы, решающей задачу).

2. С ростом объема обрабатываемой информации, количества пользователей и решаемых ими задач сложность администрирования растет.

3. Широко применяется принцип разделения доступа по директориям для разных пользователей (групп) и принцип наследования прав доступа к нижележащим элементам файловой системы. При этом получается позиционная зависимость, очень часто не имеющая ничего общего с необходимым уровнем безопасности файла. Например, ничто не мешает файлу с грифом «Совершенно секретно» находиться в каталоге с общими правами доступа.

4. Поскольку правила назначения меток безопасности отданы на откуп сфере организационных вопросов и процесс их назначения не автоматизирован, то они пишутся на естественном языке, более того, в некоторых организациях они или существуют в виде устных рекомендаций, или подразумевают, что «администратор сам должен знать, что делать». Подобный подход чреват неоднозначностью толкования этих правил и их противоречивостью, особенно явно проявляющейся в больших организациях с развитой инфраструктурой и сложной иерархией.

Далее рассматривается политика безопасности, способная предотвратить некоторые из представленных угроз.

Политика существования субъекта, ориентированная на предотвращение сбоев в работе оборудования, программного обеспечения, а также ошибок управления

Согласно данным о причинах разрушения информации [15], 52 % составляют непредумышленные действия персонала, 15 % – пожары, 10 % – умышленные действия персонала, 10 % – отказ оборудования, 10 % – затопление, 3 % – прочее.

В итоге общая доля действий персонала при разрушении информации составляет 62 %.

В работе [15] также приведен процентный состав кибер-взломщиков: 81 % – персонал учреждений, 13 % – посторонние люди, 6 % – бывшие сотрудники.

Сочетание этих статистических данных говорит о необходимости учета человеческого фактора и принятия мер по снижению его влияния.

Описанные модели политик ориентированы в основном на предоставление доступа субъекта к объекту и не позволяют распознавать подозрительное поведение субъекта после получения доступа к объекту, а также не позволяют системе выполнять действия, направленные на ликвидацию связанных с этим угроз. Именно по этой причине в большинстве современных моделей политик безопасности возможны атаки вида «отказ в обслуживании» (DoS).

Ролевая модель политики, на взгляд автора, наиболее близка для реализации модели политики существования объекта. Но она нуждается в некоторых дополнениях.

Предлагается дополнить ролевую модель следующими принципами.

1. Принцип безопасного коридора для каждого субъекта в системе.

Состояния атрибутов дескриптора субъекта (процесса), изменяющихся в некотором диапазоне, должны иметь нижнюю и верхнюю допустимые границы. Выход за них недопустим. Принцип нужен для контроля нагрузки на подсистемы операционной среды. Например, загрузка ЦПУ от процесса пользователя не должна быть более 99,9 %.

2. Реляционная полнота описания системы.

3. Периодический контроль допустимости существования субъекта в системе.

4. Машиноориентированность системы. Субъектом системы является не пользователь, а процесс пользователя, так как это более близкий к ядру системы и более информативный, чем пользователь, дескриптор. Идентификатор пользователя – это один из сотни атрибутов процесса. При реализации необходимо учитывать архитектурные особенности операционной системы и оборудования.

5. Максимальная детализация при описании и обработке системой ее служебных состояний.

6. Принцип нескольких коридоров безопасности.

Остановимся подробнее на предлагаемых принципах.

Принцип безопасного коридора

Под безопасным коридором будем понимать безопасное с точки зрения аппаратуры, программного обеспечения, состояния операционной среды (объектов) состояние, имеющее нижнюю и верхнюю допустимые границы и учитывающее текущие

состояния всех субъектов системы. Например, если в операционной системе класса Windows нет пространства для увеличения файла свопинга, а потребность пользователей в ресурсах растет, то такое состояние опасно.

По сложности реализации и по требуемым ресурсам возможно несколько уровней реализации (детализации) коридора. Старшие уровни включают в себя младшие:

1) следование процессом собственным min-max-параметрам коридора;

2) учет темпов изменения параметров процесса во времени;

3) учет темпов изменения параметров всех процессов системы во времени;

4) учет процессом min-max-параметров всех других процессов системы;

5) учет процессом собственной истории поведения;

6) учет истории поведения всей системы.

Реляционная полнота описания системы

Опасное состояние должно быть: 1) распознаваемо; 2) недопустимо; 3) протоколируемо; 4) система должна иметь стратегию выхода из такого состояния.

Должно быть распознаваемо любое состояние системы. К сожалению, этот принцип легче декларировать, чем реализовать, так как он требует: 1) детального описания деятельности каждого процесса; 2) знание всей истории поведения каждого процесса в системе.

Поскольку в результате своей деятельности процесс изменяет состояние памяти, то для накопления всей истории его поведения нужны гигантские объемы памяти. Возможно, с появлением реверсивных вычислительных систем, способных возвращаться к любому из своих предыдущих состояний, это станет реализуемым. Пока же это проблема носит теоретический характер из области теории вычислений.

Периодичность и постоянство контроля допустимости существования субъекта в системе

Традиционно один из принципов обеспечения безопасности компьютерной системы – это предоставление субъекту тех или иных прав доступа к объекту. Такой принцип хорош при идентификации субъекта, а также для принятия решения о предоставлении субъекту доступа к объекту, но он не учитывает динамики поведения субъекта и не позволяет делать количественной оценки адекватности его поведения при пользовании ресурсом. Например, необходимо предоставить право субъекту проходить сквозь дверь, а он, желая создать трудности другим, может начать это делать с частотой 10 раз в секунду. Таким образом, пользование этим ресурсом для других субъектов будет ограничено.

Большинство современных DoS атак возможно именно потому, что политики безопасности не учитывают динамики поведения системы. Таким образом, существует необходимость в постоянном контроле поведения субъекта для проверки соответствия его поведения политике системы.

Машиноориентированность системы

Субъектом системы является не пользователь, а процесс пользователя. Процесс системы является более близким для системы и, главное, более информативным. Пользователь – это всего лишь один из атрибутов процесса, тогда как количество других атрибутов у процесса очень значительно. Деятельность пользователя легко выражается в деятельности процессов, ему принадлежащих.

Максимальная детализация при описании и обработке системой ее служебных состояний

Максимальная детализация необходима для обеспечения полноты информации о состоянии системы. Учету должны подлежать все атрибуты процесса. Система должна распознавать любую их комбинацию, а также в идеале помнить всю предысторию своих переходов. К сожалению, это требует или ресурсов памяти, недоступных в настоящее время, или принципов вычислений, значительно отличающихся от традиционных. Сегодня возможность реализации этого принципа в полном объеме находится за рамками вычислительных возможностей. Поэтому необходимо учитывать динамику поведения наиболее важных атрибутов процесса. Вопрос о важности учета тех или иных атрибутов и их динамики будет краеугольным камнем при реализации такой политики на практике.

Принцип нескольких коридоров безопасности

Предлагается использовать несколько цветовых коридоров, отражающих степень доверия субъекту (процессу) или объекту (ресурсу). Файлы входящей почты, Internet-кеша, сторонних организаций должны иметь привилегии, соответствующие красному коридору.

Временные файлы, файлы *core* представляют большую угрозу, так как обычно располагаются в директориях с минимальными правами доступа. Предлагается ввести атрибут файла под названием *временный*. Установка этого атрибута означает, что файл может существовать, только пока существует процесс, его создавший. При загрузке операционной системы все временные файлы, порожденные ядром, должны уничтожаться. При любом, даже аварийном завершении процесса все открытые им временные файлы должны уничтожаться и все открытые им ресурсы – утилизироваться.

Для реализации этого принципа пригодны и мандатные модели, но, кроме этого, крайне желателен триггер-режим для коридоров, заключающийся в том, что после принятия решения о понижении уровня коридора субъекта или объекта он не может быть автоматически возвращен.

Политика управления метками, основанная на автоматическом назначении меток в зависимости от содержимого

Как видно из вышесказанного, при эксплуатации системы учет человеческого фактора не менее важен, чем контроль несанкционированного доступа или вирус-контроль. Снизить влияние человеческо-

го фактора можно при помощи предлагаемых способов пассивного и активного контроля содержимого информационного потока.

Пассивный принцип контроля содержимого. Каждый пользователь в многозадачной операционной системе работает с некоторыми информационными потоками и имеет определенные привилегии по доступу к ним. Пользователь работает в некотором информационном пространстве. Поскольку система многопользовательская, ее пользователи работают не изолированно друг от друга, а обмениваются данными, решая общие задачи, т. е. информационные пространства часто пересекаются. Такие области пересечения являются потенциально опасным пространством. Так, в рамках дискреционной модели режим безопасности информации из файла может понизить любой пользователь, имеющий к нему доступ на чтение. Для этого достаточно лишь, чтобы у него были права на запись в каталог с более низким, чем у пользователя, уровнем доступа. В моделях, обеспечивающих конфиденциальность, пользователь в состоянии неоправданно повысить степень конфиденциальности, а следовательно, доверия к объекту. В случае, если этот объект оказывается «троянской» программой – исход очевиден.

В современных системах эта проблема решается организационными способами, которые далеко не всегда эффективны, а также применением комбинаций политик безопасности, обеспечивающих целостность и конфиденциальность.

На стыке этих политик могут возникать логические уязвимости. Поэтому в рамках данного метода предлагается анализировать содержание информационного потока (памяти, файла, сетевого потока данных) и исходя из его содержания назначать ему уровень (метку) безопасности. Например, комбинация слов в файле: «Подводная лодка», «ГРИФ: Совершенно секретно» и «Исполнитель: Иванов», «Отдел перспективных разработок», «Проект: Альфа» может совершенно однозначно использоваться для разграничения доступа пользователей и групп пользователей к такому файлу. Преимущество в данном случае – позиционная независимость уровня доступа к файлу от его положения на файловой системе. Файл может находиться где угодно в системе, и он будет иметь позиционно независимый режим доступа. Контролю в этом случае подлежит содержимое, а не атрибуты файла.

Особенностью этого метода является то, что пользователь в состоянии повысить уровень секретности данных введением в него соответствующих слов, т. е. послать «троянского коня» со своего уровня на более высокий уровень безопасности. Для предотвращения этого можно использовать секретный код-идентификатор в теле документа. Например «Класс документа: XDFWE34A».

Пользователь, имеющий полномочия доступа на чтение, в состоянии понизить уровень секретности удалением или заменой (кодированием, шифрованием) ключевых слов и записью послания в другой файл на более низком уровне безопасности. Таким образом, мандатный принцип здесь не применим в его классическом смысле. Этих уязвимостей можно избежать в одном из случаев:

1) уровни безопасности не будут пересекаться;

2) применение активного принципа контроля содержимого, при котором содержимое будет шифроваться множественными ключами по принципу, описанному в работе [1];

3) применение контроля потока, описанного в модели Диона [6] с правилами, запрещающими в общем случае понижать или повышать уровень безопасности потока для рядовых пользователей и разрешающими эти процедуры для пользователей с соответствующими правами;

4) сочетание автоматического назначения меток с мандатной политикой безопасности разрешает проблему несанкционированного изменения пользователем идентификационных меток документа, так как при их кодировании документ автоматически переходит в более низкий класс допуска, запись в который, согласно политике BLP, запрещена, а для передачи на более высокий уровень необходимо знать секретный код-ключ.

Функции перехода объекта из состояния в состояние станут также безопасными и, таким образом, критические замечания по дискреционной модели Кларка и Вильсона будут учтены. Сочетание автоматического назначения меток с политикой безопасности мандатного типа позволит сделать систему менее зависимой от человеческого фактора и как следствие – более защищенной. Возможны следующие режимы назначения меток: динамический, статический, итерактивный. Наиболее прост статический метод. В этом случае субъекту и объекту метки присваиваются в командном режиме. Метки, приписанные субъектам и объектам, неизменны на протяжении их существования. Для изменения меток необходимо вмешательство администратора. При динамическом методе метки безопасности вычисляются и присваиваются исходя из некоторой целевой функции автоматически. В итерактивном режиме метки вычисляются автоматически (например, исходя из содержимого файла). При присвоении этой метки субъекту или объекту система итерактивно предлагает присвоить субъекту/объекту вычисленную метку или другую по списку. Таким образом, этот итерактивный вариант является промежуточным между статическим и динамическим. Операторское вмешательство администратора необходимо, но процесс до некоторой степени автоматизирован.

Активный принцип контроля содержимого. Дальнейшее развитие принципа контроля содержимого потока данных заключается в динамическом изменении содержимого файла в зависимости от его содержимого и метки субъекта, осуществляющего к нему доступ. Именно это и будем называть активным принципом контроля.

Под активным контролем будем понимать правила преобразования информации в зависимости от контекста пользовательского обработчика потока (динамически вычисляемой метки безопасности).

Отметим, что традиционный антивирусный мониторинг является подмножеством системы с активным контролем содержимого, но он не учитывает контекст процесса. Так, антивирусная программа может пытаться внести изменения в файл, доступ к которому на запись запрещен.

Активный принцип контроля необходим в ряде случаев: если мы хотим не только распознавать опасный код вирусов, но и производить лечение файлов на «лету», для организации антивирусной защиты; организовать контекстную обработку потока данных, когда функция преобразования входного

потока в выходной зависит от атрибутов процесса, производящего доступ к файлу. Таким образом, можно организовать объектно-ориентированный принцип доступа к ресурсам системы.

Одно из частных решений контекстной обработки потока данных – контекстное шифрование «на лету», когда ключ шифрования вычисляется как функция от меток объекта и субъекта, производящего доступ к объекту. В этом случае можно организовать такой режим шифрования, когда ключ зависит и от идентификатора носителя. Тогда для чтения информации, скопированной на сменный носитель (дискету или CD), нужен другой ключ.

Второе возможное решение – формирование одного потока как набора разных сообщений для разных корреспондентов, с применением алгоритма, описанного в работе [2]. Такой принцип шифрования информации позволит:

сократить объем передаваемых данных;

при криптоанализе такой информации она будет обладать столькими смыслами, сколько у нее корреспондентов.

Возможно предусмотреть разрушение при попытке несанкционированного доступа, а также контекстно-зависимое шифрование.

Итак, несмотря на то что традиционно подобие такой системы выполняется в виде антивирусного контроля, предлагаемый подход более полный, так как позволяет контролировать не только исполняемый код, но и данные, а также то, каким пользователем и с какими привилегиями он обрабатывается.

Преимущества предлагаемого подхода.

1. Обеспечивается позиционная независимость прав доступа пользователя к информации.

2. Снимается проблема транспорта метки уровня секретности информации.

3. Снижается риск от человеческого фактора. Человек, выполняющий надзор за техническим состоянием объекта (так называемый администратор системы или сети) при таком подходе может быть лишен возможности вмешиваться в уровень безопасности информационных потоков в системе. Это свойство уникально и недостижимо в системах с ручным назначением меток или в системах, где возможен уровень доступа пользователя, эквивалентный ядру ОС.

4. Легко тиражируются правила доступа. В случае большой организации это правило может иметь очень значительный вес при решении вопроса о выборе той или иной модели безопасности. Например, проще написать набор правил для организации масштаба транснациональной корпорации и затем заниматься его простым тиражированием, чем вменять в обязанность администраторам системы назначать права доступа к файлам на тысячах компьютерах организации.

5. Возможен реальный лексический, а вероятно, и семантический контроль содержимого файла, а не его атрибутов.

6. Антивирусный контроль должен быть одной из задач, решаемых системой.

7. Шифрование правил доступа позволит снизить уровень информированности администратора системы о специфике решаемых пользователями задач.

8. Предлагаемый принцип позволит разделять полномочия администраторов безопасности и администраторов системы на два непересекающихся класса субъектов. Более того, класс администраторов безопасности становится возможным разделить функционально на три вида администраторов:

1) администраторы политики управления метками;

2) администраторы политики назначения меток;

3) администраторы политики существования субъектов.

Каждый из этих видов администраторов имеет административные полномочия только в своем виде. Доступ к свойствам политик вне его сферы управления или ограничен до уровня чтения, или отсутствует.

Подобное разделение обязанностей хорошо тем, что ограничивает безраздельную власть администратора над безопасностью системы и снижает его уровень информированности до минимально необходимого для осуществления его полномочий, распространяя таким образом и на администраторов принцип предоставления минимальных полномочий в системе.

9. Метки безопасности могут инкапсулироваться в существующие транспортные протоколы, в том числе и в протоколы класса IP, и служить для управления уровнем безопасности сетевого трафика.

10. Возможно применение контекстно-зависимого шифрования потока информации.

11. Такая система безопасности будет обладать большей предсказуемостью поведения, так как все компоненты гарантированно имеют одинаковые настройки, т. е. система предсказуема на макроуровне.

12. Как следствие всех преимуществ – более высокий уровень безопасности системы.

Недостатки предлагаемого подхода.

1. Более высокие требования к производительности системы.

2. Необходимость описывать все информационные потоки и режимы доступа к ним настолько подробно, насколько это возможно, вплоть до допустимого содержания потоков.

3. Необходимость производить не только контроль содержимого, но и контроль информационных потоков.

4. Для снижения риска компрометации информации о решаемых в системе задачах необходимо подвергать шифрованию правила доступа при передаче и хранении.

5. В случае сети необходимо тиражировать правила доступа к информации по всему пути следования документа.

6. Метки для объектов не постоянны, что нарушает требования стандарта IEEE.

7. К сожалению, «макро» предсказуемость системы – это меч с двухсторонней заточкой, так как ее поведение предсказуемо не только для персонала кампании, но и для злоумышленников.

Это является еще одной причиной, по которой не рекомендуется применять такую систему в чистом виде, без других средств защиты.

Выводы

Политику безопасности целесообразно разделять на политику управления метками, политику управления доступом и политику существования субъекта.

Предлагается политика управления метками, использующая анализ содержимого файлов, памяти и сетевого трафика. Исходя из содержимого пользо-

вателям и группам пользователей автоматически предоставляются соответствующие права доступа.

В качестве модели политики управления доступом рекомендуется использовать модель Диона, так как она является одной из наиболее полных.

Предлагается политика существования субъекта, основанная на периодической проверке поведения субъекта и соответствия этого поведения заданным коридорам.

Политика управления метками и политика существования субъекта могут значительно повысить безопасность вычислительной системы в совокупности с моделью предоставления доступа Диона.

Литература

1. **Ерош И. Л.** Разграничение доступа к ресурсам в системах коллективного пользования // Информационно-управляющие системы. – 2003. – № 2–3. – С. 63–66.
2. **Ерош И. Л.** Защита информационных потоков в системах распределенного контроля и управления // Информационно-управляющие системы. – 2002. – № 1. – С. 40–46.
3. **Clark, D. D., Wilson, D. R.** A comparison of commercial and military computer security policies // Proceedings of the 1987 IEEE Symposium on Security and Privacy. May 1987. P. 184–194.
4. **Brewer D. F., Nash M. J.** The chinese wall security policy // IEEE Symposium on Security and Privacy. – 1989. – P. 51–68.
5. **Зегжда Д. П., Ивашко А. М.** Основы безопасности информационных систем. – М.: Горячая линия – Телеком, 2000. – 425 с.
6. **Dion L. C.** A complete protection model // Proceedings of the 1981 IEEE Symposium on Security and Privacy. – April 1981. – P. 49–55.
7. **Harrison M., Ruzzo W., Ulman J.** Protection in operating systems // Communication of the ACM. – 1976. – P. 28–37.
8. **ГОСТ Р ИСО/МЭК 15408-1–2002.** Методы и средства обеспечения безопасности информации. Критерии оценки безопасности информационных технологий. – Ч.1. Введение и общая модель. – М.: ИПК Изд-во стандартов. – 2002.
9. **Галатенко В. А.** Информационная безопасность – основы. http://www.osp.ru/dbms/1996/01/49.htm
10. **Trusted** extension of the FreeBSD operating system. http://www.trustedbsd.org
11. **Franklin M., Reiter M.** The design and implementation of a secure auction service // In Processing of the IEEE Symposium on Security and Privacy. – May 1995. – P. 2–14.
12. **McLean J.** Security models // Encyclopedia of software engineering. – 1994. – P. 246.
13. **Царегородцев А. В.** Информационная безопасность в распределенных управляющих системах: Монография. – М.: Изд-во РУДН, 2003. – 217 с.
14. **Ravi S. Sandhu** The typed access matrix model // Proceeding of IEEE Symposium on Security and Privacy. – Oakland, California. – May 4–6,1992. – P. 122–136.
15. **Семко Ю., Прохоров А.** Internet-отмычка для компьютера // Компьютер-пресс. – 2002 – № 3. – С. 38–41.
16. **ALTLinux** Castle. Общие сведения. http://castle.altlinux.ru/White-Paper.html