Глава 2. ЗАМКНУТАЯ ПРОГРАММНАЯ СРЕДА

2.1 Основные определения

Определение 12. КС называется замкнутой по порождению субъектов, если в ней действует МБС, разрешающий порождение только фиксированного конечного подмножества субъектов для любых объектов-источников, рассматриваемых для фиксированной декомпозиции компьютерной системы на субъекты и объекты [5].

При рассмотрении вопросов реализации защищенных сред будет рассматриваться термин «замкнутая программная среда», который по существу эквивалентен приведенному определению.

Однако замкнутости КС по порождению субъектов недостаточно для описания свойств системы в части защищенности, поскольку необходимо обеспечить корректность порождаемых МБС субъектов относительно его самого и МБО. Механизм замкнутой программной среды сокращает множество возможных субъектов до некоторого множества фиксированной мощности, но при этом допускает существование некорректных субъектов, включенных в замкнутую среду.

Сформулируем определение изолированности КС.

Определение 13. Множество субъектов КС называется изолированным (абсолютно изолированным), если в ней действует МБС и субъекты из порождаемого множества корректны относительно друг друга и МБС.

С учетом ограничения множества субъектов за счет применения механизма МБС можно сформулировать утверждение о достаточном условии гарантированного выполнения политики безопаности по иному: если в абсолютно изолированной КС существует МБО и порождаемые субъекты абсолютно корректны относительно МБО, а также МБС абсолютно корректен относительно МБО, то в такой системе реализуется только доступ, описанный в правилах разграничения доступа.

При рассмотрении технической реализации изолированности субъектов в КС будет употребляться термин «изолированная программная среда» (ИПС), который описывает механизм реализации изолированности.

Определение 13. Операция порождения субъекта ***Create (*$\mathbf{S_k}$, $\boldsymbol{O_m}$*)->*** $\mathbf{S_i}$ называется порождением с контролем неизменности объекта, если для любого момента времени t>t0, в который активизирована операция порождения ***Create***, порождение субъекта $\mathbf{S_i}$ возможно только при тождественности объектов **Om**[t0] и **Om**[t].

Следствие. В условиях определения 13 порожденные субъекты Si[t1] и Si[t2] тождественны, если t1>t0 t2>t0. При t1= t2 порождается один и тот же субъект.

При порождении субъектов с контролем неизменности объекта в КС допустимы потоки от субъектов к объектам-источникам, участвующим в порождении субъектов, с изменением их состояния.

Утверждение 2 (*базовая теорема ИПС).*

Если в момент времени t0 в изолированной КС действует только порождение субъектов с контролем неизменности объекта и существуют потоки от любого субъекта к любому объекту, не противоречащие условию корректности субъектов, то в любой момент времени t>t0 КС также остается изолированной (абсолютно изолированной).

Доказательство.

По условию утверждения в КС возможно существование потоков, изменяющих состояние объектов, не ассоциированных в этот момент времени с каким-либо субъектом. Если объект с измененным состоянием не является источником для порождения субъекта, то множество субъектов изолированной среды нерасширяемо, в противном случае (измененный объект является источником для порождения субъекта) по условиям утверждения (порождение субъекта с контролем) порождение субъекта невозможно. Следовательно, мощность множества субъектов не может превышать той, которая была зафиксирована до изменения состояния любого объекта. По следствию из определения 13 (о замкнутости множества субъектов в ИПС с невозрастанием мощности множества субъектов) получим, что множество субъектов КС изолировано. Утверждение доказано.

Можно сформулировать методологию проектирования гарантированно защищенных КС. Сущность данной методологии состоит в том, что при проектировании защитных механизмов КС необходимо опираться на совокупность приведенных выше в ут-

верждениях достаточных условий, которые должны быть реализованы для субъектов, что гарантирует защитные свойства, определенные при реализации МБО в КС (т.е. гарантированное выполнение заданной МБО политики безопасности).

Рассмотренная концепция изолированной программной среды является расширением зарубежных подходов к реализации ядра безопасности.

Обычно модель функционирования ядра безопасности изображается в виде следующей схемы, показанной на рис. 3.

На рис. 3 «база данных защиты» означает объект, содержащий в себе информацию о потоках множества L (защита по «белому списку» – разрешение на потоки) или N (защита по «черному списку» – запрещение на потоки).

Для учета влияния субъектов в КС необходимо рассматривать расширенную схему взаимодействия элементов системы реализации и гарантирования ПБ.

Рис. 4 наглядно поясняет взаимодействие элементов ядра безопасности с учетом контроля порождения субъектов. На рис. 4 подчеркнута роль монитора безопасности при порождении субъектов из объектов. Взаимодействие субъектов и объектов при порождении потоков уточнено введением ассоциированных с субъектом объектов.

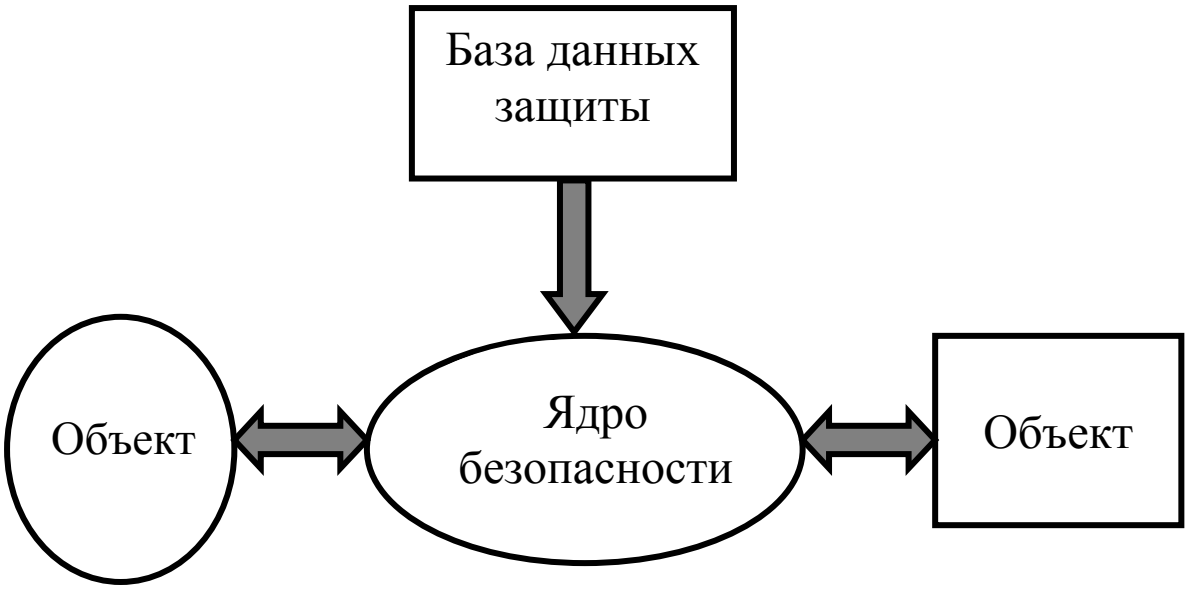

Рис. 3 – Классическая модель ядра безопасности

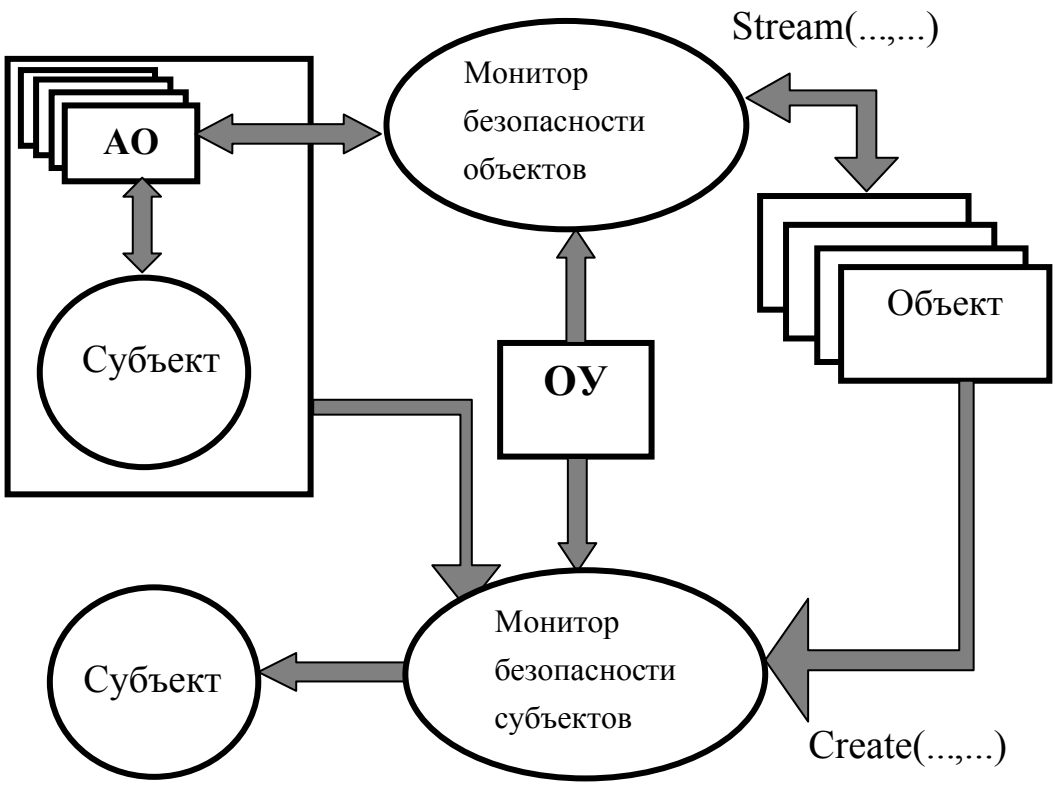

Рис. 4 – Ядро безопасности с учетом контроля порождения субъектов

Конструкция ОУ на схеме обозначает объект управления, т.е. объект, содержащий информацию о разрешенных значениях ***Stream*** (об элементах множества L или N) и ***Create*** (элементы множества E). Объект управления может быть ассоциирован (ассоциированный объект-данные) как с МБО, так и с МБС.

Перейдем к описанию практических методов построения изолированной программной среды. Целью рассмотрения практических подходов является иллюстрация утверждения о том, что достаточные условия гарантированной защищенности могут быть практически выполнены в реальных КС.

2.2 Метод генерации изолированной программной среды при проектировании механизмов гарантированного поддержания политики безопасности

Опираясь на базовую теорему ИПС, опишем метод субъектно-объектного взаимодействия в рамках ИПС для более конкретной архитектуры КС [5].

Из теоремы следует, что для создания гарантированно защищенной КС в смысле выполнения политики безопасности необходимо:

1. Убедиться в попарной корректности субъектов, замыкаемых в ИПС (либо убедиться в корректности любого субъекта относительно МБО и МБС).

2. Спроектировать и реализовать программно (или программно-аппаратно) МБС так, чтобы:

- для любого субъекта и любого объекта производился контроль порождения субъектов (т.е. чтобы реализация МБС соответствовала его определению;
- порождение любого субъекта происходило с контролем неизменности объекта-источника.

3. Реализовать МБО в рамках априорно сформулированной политики безопасности.

Надо заметить, что приводимые выше утверждения верны только тогда, когда описанная и реализованная ПБ не нарушает их условий (проверка данного факта зависит от модели ПБ и является отдельной весьма важной задачей).

Кроме того, необходимо обратить внимание на следующее. Объект управления, который является ассоциированным объектом МБС (обычно ассоциированный объект-данные), играет решающую роль в проектировании ИПС. При возможности изменения состояния объекта управления потенциально возможно «размыкание» программной среды, т.е. добавление к множеству разрешенных субъектов дополнительных, реализующих злоумышленные функции. С другой стороны, процесс управления безопасностью подразумевает возможность изменения объекта управления. Возможность изменения объекта управления (реализация потока ***Stream*** (субъект управления, АО объекты субъек-

та управления)->ОУ) должна присутствовать для выделенных субъектов.

Важную роль при проектировании ИПС играет свойство КС, заключающееся в поэтапной активизации субъектов из объектов различного уровня представления информации. Рассмотрим в таблице 1 иерархию уровней при загрузке операционной системы.

В таблице выделен термин «сектор» для обозначения представления объекта аппаратно-программного уровня. Он обозначает непрерывную последовательность элементов хранения (байт) на материальном носителе, характеризуемую местом расположения.

Термин «файл» обозначает абстрактный объект, построенный по списочной структуре из объектов «сектор». Объекты типа «файл» и «сектор» выделены исключительно исходя из типовой структуры объектов КС.

Таблица 1 – Иерархия уровней при загрузке ОС

Уровень	Субъект	Локализация	Представление информации	Через какие функции реализуются потоки
0	Субъект аппаратно-программного уровня	ПЗУ (Bios)	сектора	через микропрограммы ПЗУ
1	Субъект уровня первичной загрузки	Загрузчик ОС	сектора	через Bios или первичный загрузчик
2	Субъект уровня вторичного загрузчика (драйвер)	Драйверы ОС	сектора	через Bios или первичный загрузчик
3	Субъект уровня ОС	Ядро ОС	файлы	через драйверы
4	Субъект пользовательского уровня	Прикладные приложения	файлы	через ядро ОС

С учетом иерархической структуры представления объектов можно говорить о том, что в начальные этапы активизации КС декомпозиция на субъекты и объекты динамически изменяется. Следовательно, основная теорема ИПС может быть применима только на отдельных интервалах времени, когда уровень представления объектов постоянен и декомпозиция фиксирована. Можно утверждать, что ИПС, действующую от момента активизации до момента окончания работы КС, невозможно сформировать в начальный момент активизации КС.

Пусть в КС выделяется конечное число уровней представления объектов U={0,…,R}, где R – максимальный уровень представления объекта.

С точни зрения базовой теоремы ИПС имело бы смысл говорить о некотором «стационарном состоянии КС, когда в отображениях ***Stream*** и ***Create*** участвуют только объекты уровня R. Тогда реализация МБС может быть значительно упрощена (в том смысле, что все аргументы-объекты операции ***Create*** имеют тот же уровень). Необходимо обратить внимание на то, что такое требование, с одной стороны, может накладывать ограничительные условия на свойства прикладного ПО (невозможность инициирования потоков, включающих объекты уровня менее R, прикладными программами, а с другой стороны, быть следствием проектировочных решений реализации субъекта, локализованного в ядре операционной системы (примером является ОС Windows NT 4.0, запрещающая операции ниже уровня «файл» со стороны субъектов прикладного уровня).

Практическая реализация всех операционных систем позволяет выделить две фазы их работы: активизация субъектов с ростом уровня представления объектов (фаза загрузки или начальная фаза) и фаза стационарного состояния (когда уровень представления объектов не увеличивается). Конечно, необходимо сделать оговорку, касающуюся возможности реализации потоков к объектам нижнего уровня (операционные системы типа DOS, в которых возможна операция с любым объектом нижнего уровня (сектор) из программ прикладного уровня).

Тогда практическая реализация ИПС может состоять из двух этапов: предопределенное выполнение начальной фазы, включающее в себя момент активизации МБС (и МБО), и работа

в стационарной фазе в режиме ИПС (возможно, с контролем неизменности объектов-источников).

Введем понятие последовательности активизации компонент КС. Смысл вводимых понятий и формулируемых ниже утверждений состоит в необходимости приведения субъектов КС в одно и то же состояние после активизации первичного субъекта аппаратно-программного уровня или, иначе говоря, в задании предопределенной последовательности активизации субъектов КС.

Обозначим: ZL – последовательность пар (i,j)t (t=0,1,2,…,**l**-1 – моменты времени) длины **l**, такие, что ***Create(Si,Oj)***[t]->**Sm**[t+**l**].

Обозначим также:

Sz – множество всех субъектов, включенных в последовательность Z_L;

O_Z – множество всех объектов, включенных в последовательность Z_L.

Для многопотоковых КС можно рассматривать несколько (возможно, независимых друг от друга) последовательностей Z_Lи соответственно множеств Sz и O_Z.

Определение 14. Состоянием КС в момент времени t называется упорядоченная совокупность состояний субъектов.

Напомним, что каждый объект есть слово в априорно определенном языке, а понятие состояния субъекта сформулировано выше.

Утверждение 3 (*условие одинакового состояния КС*).

Состояние КС в моменты времени tx1 и tx2 (tx1 и tx2 исчисляются для двух отрезков активности КС от нулевого момента активизации КС to1 и to2 – например, включения питания аппаратной части) одинаково, если:

1) tx1=tx2;

2) тождественны субъекты Si[to1] и Si[to2];

3) неизменны все объекты из множества O_Z;

4) неизменна последовательность Z_L.

Доказательство по принципу математической индукции.

Верность утверждения при t=1 следует из определения тождественности субъектов.

Пусть утверждение верно для t=k<1.

Тогда в момент времени k+1 могут быть порождены только тождественные субъекты, поскольку тождественны активизирующие субъекты (по предположению индукции) и по условию утверждения неизменны элементы множества O_Z.

Длина l последовательности Z_L определяется:

1. По признаку невозможности управления субъектами, принадлежащими множеству Sz, со стороны пользователя (в противном случае последовательность активизации субъектов может быть изменена).

2. По признаку доступности для контроля неизменности всех объектов из множества O_Z.

3. По признаку невозрастания уровня представления информации (в данном случае имеется в виду, что существует момент времени tx такой, что для любого t>tx объект-аргумент O_j операции ***Create(Si,Oj)*** принадлежит одному уровню представления).

Необходимо заметить, что последовательность Z_L локализуется в некотором объекте либо совокупности объектов (например, для DOS последовательность активизации субъектов предопределена содержанием файлов AUTOEXEC.BAT и CONFIG.SYS) и неизменность последовательности Z_L тождественна неизменности указанных объектов, для ОС Windows NT последовательность активизации компонент определена содержанием соответствующих ключей реестра (registry).

Пусть в последовательности Z_L можно выделить z_i такое, что для любого z_k при k>i отображения ***Create*** и ***Stream*** используют только объекты уровня R. Другими словами, с момента времени i наступает стационарная фаза функционирования КС.

В этих условиях, а также при попарной корректности субъектов и действии МБС с контролем неизменности объектов-источников на уровне R с момента времени m>k верно:

Утверждение 4 *(достаточное условие ИПС при ступенчатой загрузке).*

При условии неизменности Z_L и неизменности объектов из O_Z в КС с момента времени установления неизменности Z_L и O_Z действует ИПС. Доказательство не приводим.

Утверждение 5 *(требования к субъектному наполнению ИПС).*

Для того, чтобы ИПС поддерживалась в течение всего времени активности КС, достаточно, чтобы в составе программного обеспечения, могущего быть инициированным в ИПС, не было функций порождения субъектов и прекращения их работы, кроме заранее предопределенных при реализации МБС, и не существовало возможностей влияния на среду выполнения (под средой выполнения понимается множество ассоциированных объектов) любого процесса, а также инициирования потоков к объектам логического уровня менее R.

Утверждение 6 *(достаточное условие чтения реальных данных).*

Если субъект, обслуживающий процесс чтения данных (т.е. указанный субъект инициируется запрашивающим данные субъектом и участвует в потоке) содержал только функции тождественного отображения данных на ассоциированные объекты-данные любого субъекта, инициирующего поток чтения, и целостность объекта-источника для этого субъекта зафиксирована, то при его последующей неизменности чтение с использованием порожденного субъекта будет чтением реальных данных.

2.3 Формирование и поддержка изолированной программной среды

Предположим, что в КС работают N субъектов-пользователей, каждый i-й из которых характеризуется некоторой персональной информацией Ki, не известной другим пользователям и хранящейся на некотором материальном носителе. Существует также выделенный субъект − администратор системы, который знает все Ki. Администратор КС присваивает i-му пользователю полномочия, заключающиеся в возможности исполнения им только заданного подмножества программ

$$Ti=\{Pi1, Pi2, \ldots Pit\}.$$

Несанкционированным доступом является использование имеющихся на жестком диске ПЭВМ программ либо субъектом, не входящим в N допущенных, либо i-м пользователем вне подмножества своих полномочий Ti. Субъект, пытающийся проделать данные действия, называется злоумышленником. НСД осу-

ществляется обязательно при помощи имеющихся на ПЭВМ или доставленных злоумышленником программных средств (в данном случае не рассматривается возможность нарушения целостности аппаратных средств компьютера). НСД может носить непосредственный и опосредованный характер. При непосредственном НСД злоумышленник, используя некоторое ПО, пытается непосредственно осуществить операции чтения или записи (изменения) интересующей его информации. Если предположить, что в Ti нет программ, дающих возможность произвести НСД (это гарантирует администратор при установке полномочий), то НСД может быть произведен только при запуске программ, не входящих в Ti. Опосредованный НСД обусловлен общностью ресурсов пользователей и заключается во влиянии на работу другого пользователя через используемые им программы (после предварительного изменения их содержания или их состава злоумышленником). Программы, участвующие в опосредованном НСД, будем называть разрушающими программными воздействиями (РПВ) или программными закладками. РПВ могут быть внедрены i-м пользователем в ПО, принадлежащее j-му пользователю только путем изменения программ, входящих в Tj. Следовательно, система защиты от НСД должна обеспечивать контроль за запуском программ, проверку их целостности и активизироваться всегда для любого пользователя. Выполнение контроля целостности и контроля запусков ведется на основе Ki для каждого пользователя. При этом внедренный в КС защитный механизм должен обеспечивать следующее:

- в некоторый начальный момент времени требовать у субъекта предъявления аутентифицирующей информации и по ней однозначно определять субъекта и его полномочия Ti;

- в течение всего времени работы пользователя i должен обеспечивать выполнение программ только из подмножества Ti;

- пользователь не должен иметь возможности изменить подмножество Ti и/или исключить из дальнейшей работы защитный механизм и его отдельные части.

Положим, что в ПЗУ (BIOS) и операционной среде (в том числе и в сетевом ПО) отсутствуют специально интегрированные в них возможности НСД. Пусть пользователь работает с программой, в которой также исключено наличие каких-либо

скрытых возможностей (проверенные программы). Потенциально злоумышленные действия могут быть такими:

1. Проверенные программы будут использованы на другой ПЭВМ с другим BIOS и в этих условиях использоваться некорректно.

2. Проверенные программы будут использованы в аналогичной, но не проверенной операционной среде, в которой они также могут использоваться некорректно.

3. Проверенные программы используются на проверенной ПЭВМ и в проверенной операционной среде, но запускаются еще и не проверенные программы, потенциально несущие в себе возможности НСД.

Тогда НСД в КС гарантированно невозможен, если выполняются условия:

У1. На ПЭВМ с проверенным BIOS установлена проверенная операционная среда.

У2. Достоверно установлена неизменность DOS и BIOS для данного сеанса работы.

У3. Кроме проверенных программ в данной программно-аппаратной среде не запускалось и не запускается никаких иных программ, проверенные программы перед запуском контролируются на целостность.

У4. Исключен запуск проверенных программ в какой-либо иной ситуации, т.е. вне проверенной среды.

У5. Условия У1-4 выполняются в любой момент времени для всех пользователей, аутентифицированных защитным механизмом.

При выполнении перечисленных условий программная среда называется изолированной (далее будем использовать термин ИПС – изолированная программная среда). Функционирование программ в изолированной программной среде (ИПС) существенно ослабляет требования к базовому ПО. В самом деле, ИПС контролирует активизацию процессов через операционную среду, контролирует целостность исполняемых модулей перед их запуском и разрешает инициирование процесса только при одновременном выполнении двух условий – принадлежности к разрешенным и неизменности.

В таком случае от базового ПО требуется только:

1. Невозможность запуска программ помимо контролируемых ИПС событий.

2. Отсутствие в базовом ПО возможностей влиять на среду функционирования уже запущенных программ (фактически это требование невозможности редактирования оперативной памяти).

Все прочие действия, являющиеся нарушением Условий 1-3, в оставшейся их части будут выявляться и блокироваться. Таким образом, ИПС существенно снижает требования к ПО в части наличия скрытых возможностей.

Основным элементом поддержания изолированности среды является контроль целостности. При этом возникает проблема чтения реальных данных, так как контроль целостности всегда сопряжен с чтением данных (по секторам, по файлам и т.д.). В процессе чтения РПВ может навязывать вместо одного сектора другой или редактировать непосредственно буфер памяти. С другой стороны, даже контроль самого BIOS может происходить «под наблюдением» какой-либо дополнительной программы («теневой BIOS») и не показать его изменения. Аналогичные эффекты могут возникать и при обработке файла. Таким образом, внедренное в систему РПВ может влиять на процесс чтения-записи данных на уровне файлов или на уровне секторов и предъявлять системе контроля некоторые другие вместо реально существующих данных. Этот механизм неоднократно реализовывался в STELTH-вирусах. Однако верно утверждение: если программный модуль, обслуживающий процесс чтения данных, не содержал РПВ и целостность его зафиксирована, то при его последующей неизменности чтение с использованием этого программного модуля будет чтением реальных данных. Из данного утверждения следует способ ступенчатого контроля.

Алгоритм ступенчатого контроля для создания ИПС на примере DOS

При включении питания ПЭВМ происходит тестирование ОП, инициализация таблицы прерываний и поиск расширений BIOS. При их наличии управление передается на них. После отработки расширений BIOS в память считывается первый сектор дискеты или винчестера (загрузчик) и управление передается на него, код загрузчика считывает драйверы DOS, далее выполня-

ются файлы конфигурации, подгружается командный интерпретатор и выполняется файл автозапуска.

С учетом этого механизма для реализации ИПС предварительно фиксируется неизменность программ в основном и расширенных BIOS, далее, используя функцию чтения в BIOS (для DOS int 13h), читаются программы обслуживания чтения (драйверы DOS), рассматриваемые как последовательность секторов, и фиксируется их целостность. Далее, используя уже файловые операции, читаются необходимые для контроля исполняемые модули (командный интерпретатор, драйверы дополнительных устройств, .EXE и .COM – модули и т.д.). При запуске ИПС таким же образом и в той же последовательности выполняется контроль целостности. Этот алгоритм можно обобщить на произвольную операционную среду. Для контроля данных на i-м логическом уровне их представления для чтения требуется использование предварительно проверенных на целостность процедур i-1-го уровня. В случае описанного механизма загрузки процесс аутентификации необходимо проводить в одном из расширений BIOS (чтобы минимизировать число ранее запущенных программ), а контроль запуска программ включать уже после загрузки DOS (иначе DOS определяет эту функцию на себя). При реализации ИПС на нее должна быть возложена функция контроля за запуском программ и контроля целостности.

Реализация ИПС с использованием механизма расширения BIOS

Рассмотрим 2 этапа – этап установки ИПС и этап эксплуатации ИПС. Предположим существование N пользователей, каждый i-й из которых характеризуется некоторой персональной информацией Ki, не известной другим пользователям и хранящейся на некотором материальном носителе (например, устройстве типа Touch Memory). Существует также администратор КС, который знает все Ki и единолично проводит этап установки. Пользователи же участвуют только в этапе эксплуатации.

Процесс установки ИПС состоит из следующих действий:

1. В ПЭВМ устанавливается плата, включающая в себя устройства и программы ПЗУ данного устройства, реализующие:

- чтение Ki;

- идентификацию пользователя с номером i по введенному Ki;

- чтение массива данных, содержащего множество доступных для выполнения пользователем i задач Pi1, Pi2, ..., Pim и информации Mi1, Mi2, ... Mim, фиксирующей целостность файлов Fi1, ..., Fim каждой задачи.

Описанное устройство должно активизироваться сразу после включения питания, отработки процедур самотестирования и инициализации системы прерываний.

Для ПЭВМ типа IBM PC для этой цели необходимо использовать механизм расширения BIOS.

2. Администратор определяет для пользователя i набор задач и соответствующих задачам исполняемых файлов {Pit, Fit}, t=1,...,mi; i=1, ..., N, где mi - число разрешенных к запуску задач для i-го пользователя.

3. Администратор формирует (и заносит на носитель) или считывает с носителя для i-го пользователя его Ki и вычисляет значения для последующего контроля целостности Mir=f(Ki, Fir, Pir), где f – функция фиксации целостности (хэш-функция).

4. Администратор проделывает действия 2 и 3 для всех N пользователей.

5. Администратор устанавливает в программную среду модуль активизации ИПС и фиксирует его целостность. Фиксируется также целостность файлов операционной среды Foc (в которые входят файлы DOS, драйверы и сетевое ПО).

Процесс эксплуатации состоит из следующих действий.

1. Включение питания и активизация расширенного BIOS:

а) идентификация пользователя по его Ki (при успехе п. б);

б) проверка целостности всех включенных в ПЭВМ BIOS (при положительном исходе п. в);

в) чтение по секторам файлов DOS и проверка их целостности;

г) чтение как файлов Рипс, так и сетевого ПО (с помощью функций операционной среды) и проверка его целостности;

д) активизация сетевого ПО;

е) активизация процесса контроля Рипс;

ж) запуск избранной задачи i-го пользователя.

2. Работа в ИПС.

Запуск каждого процесса Ps сопровождается проверками:

а) принадлежит ли Fs к множеству разрешенных для i (Ti), если да, то п. б), иначе запуск игнорируется;

б) совпадает ли G=f(Ki,Fs,Ps) с M=f(Ki,Fs,Ps), вычисленной администратором;

в) при положительном исходе проверки б) задача запускается.

Легко видеть, что условия изолированности среды (У1-5) в данном случае выполнены.

Пункт У1 гарантируется при установке системы администратором.

Пункты У2, У4 и У5 обеспечиваются платой (загрузка собственной DOS и сетевой среды с дискеты невозможна, поскольку расширенный BIOS активен раньше и направляет загрузку на винчестер; пользователь допускается к работе только при проверке Ki).

Пункт У3 реализован программным модулем контроля запусков и контроля целостности задач, входящим в состав ИПС. Кроме того, в данном случае реализован механизм ступенчатого контроля, обеспечивающий чтение реальных данных.