**ПЕРЕДАЧА УПРАВЛЕНИЯ МЕЖДУ УРОВНЯМИ ПРИВИЛЕГИЙ**

**Напомним основные моменты защиты по привилегиям, рассмотренные в предыдущих разделах:**

**- процессор i86 аппаратно поддерживает работу на любом из четырех уровней привилегий;**

**- текущий уровень привилегий CPL определяется выполняемым кодом, который, в свою очередь, задается полем DPL дескриптора сегмента кода;**

**- процессор жестко ограничивает передачи управления в пределах текущего уровня привилегий.**

**Передача управления между уровнями привилегий в первую очередь связана с доступом пользовательских программ (PL3-программ) к обслуживаниям операционной системы, процедуры которых работают на уровне PL1. Об объемах услуг операционной системы свидетельствует, например, тот факт, что операционная система представляет пользователям более 150 разнообразных обслуживаний.**

**Имеются два ответа на поставленный выше вопрос. Первый их них (простой) заключается в применении подчиненных сегментов кода. Второй (сложный) – в использовании специальных дескрипторов, называемых шлюзами вызова.**

**ПОДЧИНЕННЫЕ СЕГМЕНТЫ КОДА.**

**Представим себе, что в системе имеется процедура (т.е. сегмент кода) для преобразования двоичных целых чисел в символьный код ASCII, который используют программы на каждом уровне привилегий. Так как вызовы менее привилегированных процедур процессор запрещает, процедура должна находиться в кольце 0, чтобы ее можно было вызвать из этого кольца. Обычно такая процедура требует достаточных привилегий только для обращения к своему параметру (двоичному числу) и возвращения результата (символьного кода), т.е. таких же привилегий, что и вызывающая программа. Именно для таких ситуаций в процессоре i86 предусмотрены подчиненные (точнее, подчиняемые) или согласованные сегменты кода.**

**Сегмент кода определяется, как подчиненный установкой бита 2 (бит C – Conforming) в байте прав доступа AR дескриптора сегмента – типы сегментов 6 и 7. В таких сегментах может быть разрешена или запрещена операция считывания. Обычные правила защиты по значениям CPL и DPL не действуют, если бит C=1. В этом случае вводятся другие правила.**

**С подчиненными сегментами кода не ассоциируется конкретный уровень привилегий, так как они подчиняются уровню привилегий того вида, который передает им управление с помощью команд CALL или JMP. Другими словами, подчиненные сегменты кода никогда не требуют больше привилегий, чем код, передающий им управление. Если, например, PL3-программа передает управление подчиненному сегменту кода, то он работает с CPL = 3; если же этот сегмент вызывает PL0-программа, то он выполняется с CPL = 0. Из-за такой подвижности уровня привилегий подчиненного сегмента кода в нем не должны содержаться приведенные выше привилегированные команды, например, команды ввода-вывода.**

**Когда управление передается подчиненному сегменту кода, биты поля RPL регистра CS не изменяются на поле DPL дескриптора нового сегмента кода, как это обычно бывает, а сохраняют значения DPL последнего выполнявшегося неподчиненного сегмента кода. Только в этой единственной ситуации биты RPL регистра CS не соответствуют битам DPL дескриптора текущего выполняющегося сегмента кода.**

**Хотя с подчиненным сегментом кода не ассоциируется никакой конкретный уровень привилегий, применительно к нему все же действует одно ограничение. Оно заключается в том, что значение DPL дескриптора подчиненного сегмента кода всегда должно быть меньше или равно текущему значению CPL. Другими словами, передача управления разрешается только во внутренние, более защищенные кольца. Передавать управление сегменту с DPL > CPL (т.е. менее привилегированному, чем текущая программа) никогда не разрешается.**

**Чтобы разобраться в упомянутом ограничении, следует рассмотреть ситуацию, когда процедура в подчиненном сегменте закончилась и необходимо возвращать управление на уровень привилегий вызвавшего кода. Если для перехода к подчиненному сегменту кода применялась команда JMP (или команда CALL), то возвращающая управление команда JMP (или команда CALL) должна передавать управление более привилегированному и, главное, неподчиненному сегменту кода. Такие передачи управление запрещаются, что, собственно, лежит в основе механизма защиты по привилегиям в процессоре i86.**

**ШЛЮЗЫ ВЫЗОВА. Наличие подчиненных сегментов кода обеспечивает некоторую свободу передачи управления с одного уровня привилегий на другой. Хотя подчиненный код фактически не изменяет текущий уровень привилегий, он позволяет программам на различных уровнях привилегий разделять общий код. Обычно им являются разделенные библиотеки, а идея подчиненного кода не требует наличия копий библиотек на каждом уровне привилегий.**

**Для реализации фактического изменения уровней привилегий привлекаются особые системные объекты, называемые шлюзами вызова. (В переводной литературе употребляется менее удачный термин «вентиль вызова».)**

**В дополнение к дескрипторам сегментов кода и данных прикладных программ процессор i86 имеет дескрипторы для системных сегментов и шлюзов. Эти структуры данных привлекаются для управления задачами, а также особыми случаями и прерываниями. В таблице 1 приведены все типы дескрипторов, определенные для системных сегментов и шлюзов. Отметим, что не все дескрипторы определяют сегменты; например, в дескрипторах шлюзов содержатся точки входа процедур.**

**Таблица 1. Типы системных сегментов и шлюзов**

|  |  |
| --- | --- |
| **Тип** | **Описание** |
| **0**  **1**  **2**  **3**  **4**  **5**  **6**  **7**  **8**  **9**  **10**  **11**  **12**  **13**  **14**  **15** | **Зарезервирован**  **Доступный TSS процессора 80286**  **Таблица LDT**  **Занятый TSS процессора 80286**  **Шлюз вызова**  **Шлюз задачи**  **Шлюз прерывания процессора 80286**  **Шлюз ловушки процессора 80286**  **Зарезервирован**  **Доступный TSS процессора i486**  **Зарезервирован**  **Занятый TSS процессора i86**  **Шлюз вызова процессора i86**  **Зарезервирован**  **Шлюз прерывания процессора i86**  **Шлюз задачи процессора i86** |

**Как системный объект, шлюз вызова имеет собственный дескриптор:**

**Как видно из формата, 48 бит в дескрипторе шлюза вызова определяют полный указатель селектор:смещение точки входа той процедуры (назначения), которой шлюз передаёт управление. Дескриптор шлюза вызова действует как посредник или своеобразный «интерфейсный слой» между сегментами кода, находящимися на различных уровнях привилегий. Шлюзы вызова идентифицируют разрешенные точки входа в более привилегированном коде, которым может быть передано управление, и являются единственным средством смены уровней привилегий. Т.е. шлюз вызова – системный объект, который адресует процедуру в сегменте кода. Шлюз имеет уровень привилегий, отличный от уровня привилегий вызываемого сегмента кода.**

**В отличии от дескрипторов сегментов кода, данных или стека, а также дескрипторов таблиц LDT , дескрипторы шлюзов вызова не определяют никакого адресного пространства, поэтому в них нет полей базы и предела. По существу дескрипторы шлюзов вызова не являются дескрипторами, но их удобно размещать в дескрипторных таблицах. Селекторы, которые выбирают дескрипторы шлюзов вызова можно загружать только в сегментный регистр CS для передачи управления сегменту кода на другом уровне привилегий. Адресовать шлюз вызова разрешается только в команде межсегментного вызова FAR CALL, а команды безусловного перехода JMP запрещены.**