**Защита.**

Глава 1. Проверки, обеспечивающие защиту. http://top.list.ru/counter?id=213421;js=13;r=http%3A//sasm.narod.ru/docs/pm/pm_pr/main.htm;j=false;s=1280*1024;d=24;rand=0.06580230254041175

        Когда используется механизм защиты, каждое обращение к памяти должно удовлетворять целому ряду условий защиты. Все проверки выполняются до начала обращения к памяти и любые нарушения приводят к генерации исключений. Эти проверки происходят одновременно с трансляцией адреса и не приводят к потере производительности.   
        Проверки защиты делятся на следующие категории:

|  |  |
| --- | --- |
|  | Проверка предела. |
|  | Проверка типа. |
|  | Проверка уровня привилегий. |
|  | Ограничение адресного пространства. |
|  | Ограничение входных точек процедуры. |
|  | Ограничение набора команд. |

        В данном разделе рассматривается механизм защиты, применяемый в защищённом режиме. Защита в режиме реальных адресов и виртуального 8086 рассматривается в разделе "Режим виртуального процессора 8086".

**1.1. Разрешение и запрещение защиты сегментов и страниц.**

        Установка флага PE в регистре CR0 заставляет процессор переключиться в защищённый режим, который в свою очередь автоматически обеспечивает механизм защиты, причём та его часть, которая отвечает за защиту по привилегиям, может вообще не использоваться, если программа, работающая в защищённом режиме использует все дескрипторы и селекторы на нулевом уровне привилегий.   
        Защита на уровне страниц автоматически обеспечивается при включении страничного преобразования установкой бита PG в CR0. Эту защиту можно отключить, выполнив следующие операции:

|  |  |
| --- | --- |
|  | Сбросить флаг WP в регистре CR0. |
|  | Установить флаги R/W и (U/S) в каждом элементе каталога и таблиц страниц. |

        Эти действия делают каждую страницу доступной для чтения и записи с любого уровня привилегий, что отменяет защиту на уровне страниц.

**1.2. Поля и флаги, используемые для защиты на уровне сегментов и страниц.**

|  |  |
| --- | --- |
|  | **Флаг типа дескриптора (S)** - (4-й бит байта прав доступа дескриптора). Определяет вид дескриптора - если 0, то он описывает системный объект, если 1, то сегмент кода или данных. |
|  | **Поле типа дескриптора** - (Биты 0..3 байта прав доступа). Определяет тип системного объекта, сегмента кода или данных. |
|  | **Поле предела дескриптора** - (Биты 0..15 и 48..51 в дескрипторе). Определяют размер сегмента в единицах размера (байтах или 4Кб страницах). |
|  | **Флаг гранулярности G** - (Бит 55 дескриптора). Определяет единицы измерения размера сегмента: 0 - сегмент измеряется в байтах, 1 - в 4Кб страницах. |
|  | **Флаг E** - (Бит 2 байта прав доступа дескриптора сегмента данных). Определяет расширение сегмента: 0 - адреса растут вверх (как в обычном сегменте), 1 - вниз (т.е. наоборот). |
|  | **Поле уровня привилегий дескриптора (DPL)** - (Биты 5 и 6 байта прав доступа). Определяют уровень привилегий сегмента. |
|  | **Поле запрошенного уровня привилегий (RPL)** - (Биты 0 и 1 любого селектора сегмента). Содержат запрашиваемый уровень привилегий при обращении к дескриптору. |
|  | **Поле текущего уровня привилегий (CPL)** - (Биты 0 и 1 регистра CS). Содержат уровень привилегий текущей процедуры. |
|  | **Флаг пользователя/системы (User/Supervisor) (U/S)** - (Бит 2 элемента каталога или таблицы страниц). Определяют тип страницы: 0 - системный, 1 - пользовательский. |
|  | **Поле чтения/записи (Read/write) (R/W)** - (Бит 1 элемента каталога или таблицы страниц). Определяет тип доступа к странице: 0 - только чтение, 1 - чтение и запись. |

**1.3. Проверка предела.**

        Проверка предела дескриптора сегмента не позволяет программе обращаться за пределы сегмента. Значение предела зависит от флага гранулярности (G) дескриптора. Предел определяет максимальное значение смещения в сегменте.   
        Для сегментов данных, расширяющихся вниз, предел определяет последний адрес, доступ к которому запрещён внутри сегмента. Допустимыми будут адреса в диапазоне от (предел + 1) до FFFFh, если флаг D=0 и от (предел + 1) до FFFFFFFFh, если D=1. Максимальный размер такие сегменты имеют с пределом, равным нулю.   
        Процессор также проверяет пределы дескрипторных таблиц GDT, IDT, LDT и текущего сегмента TSS, не позволяя обращаться за их пределы.

**1.4. Проверка типа**

        Дескриптор сегмента содержит информацию о типе в двух элементах:

|  |  |
| --- | --- |
|  | Флаг S (тип дескриптора). |
|  | Поле типа. |

        Процессор использует эту информацию для определения ошибочных действий программ, когда они пытаются использовать сегмент или шлюз неправильным или несоответствующим образом.   
        Флаг S определяет, описывает дескриптор системный объект или сегмент кода или данных. Поле типа содержит 4 дополнительных бита, определяя различные типы дескрипторов.   
        В таблице 1-1 приведены значения поля типа дескрипторов, в таблице 1-2 приведены значения полей для системных дескрипторов.

        Процессор проверяет значение типа несколько раз, когда оперирует селекторами и дескрипторами. Далее приведены примеры типичных операций, где происходит проверка типов:

|  |  |
| --- | --- |
| - | В регистр CS можно загрузить только селектор сегмента кода. |
| - | Селектор сегмента нечитаемого кода нельзя загружать в сегментные регистры данных (DS, ES, FS и GS). |
| - | В регистр SS можно загружать только селектор сегмента данных, разрешённого для записи. |
| - | В LDTR можно загрузить можно загрузить только селектор для LDT. |
| - | В TR можно загрузить можно загрузить только селектор для TSS. |
| - | Нельзя записывать в сегмент кода. |
| - | Нельзя записывать в сегмент данных, предназначенный только для чтения. |
| - | Нельзя читать из сегмента кода, предназначенного только для выполнения. |
| - | Команда LAR должна обращаться к дескриптору сегмента или шлюза для сегментов LDT, TSS, шлюзу вызова, шлюзу задачи, сегменту кода или данных. |
| - | Команда LSL должна обращаться к дескриптору сегмента LDT, TSS, кода или данных. |
| - | Элементом IDT может быть только шлюзы прерывания, ловушки или вызова. |
| - | При выполнении команд FAR CALL и FAR JMP процессор проверяет тип дескриптора, селектор которого содержится в адресе назначения этих команд. Если это дескриптор сегмента кода или шлюза вызова, то происходит передача управления по этому адресу; если дескриптор описывает TSS или шлюз задачи, то происходит переключение задач. |

**Таблица 1-1. Тип сегмента**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Бит # | 11 | 10 | 9 | Тип |  |
| Название |  | E | W |
|  | 0 | 0 | 0 | Данные | Только чтение |
| 0 | 0 | 1 | Данные | Чтение и запись |
| 0 | 1 | 0 | Данные | Только чтение, расширяется вниз |
| 0 | 1 | 1 | Данные | Чтение и запись, расширяется вниз |
|  | | | | | |
|  | 1 | 0 | 0 | Код | Только выполнение |
| 1 | 0 | 1 | Код | Только выполнение |
| 1 | 1 | 0 | Код | Только выполнение, согласованный |
| 1 | 1 | 1 | Код | Выполнение и считывание, согласованный |

**Таблица 1-2. Типы системных объектов**

|  |  |  |
| --- | --- | --- |
| **Значе- ние** | **Поле типа1  биты: 3 2 1 0** | **Системный объект** |
| 0 | 0 0 0 0 | Зарезервировано |
| 1 | 0 0 0 1 | 16-разрядный TSS (Свободный)2 |
| 2 | 0 0 1 0 | LDT |
| 3 | 0 0 1 1 | 16-разрядный TSS (Занятый)2 |
| 4 | 0 1 0 0 | 16-разрядный шлюз вызова (call gate) |
| 5 | 0 1 0 1 | Шлюз задачи (task gate) |
| 6 | 0 1 1 0 | 16-разрядный шлюз прерывания (interrupt gate) |
| 7 | 0 1 1 1 | 16-разрядный шлюз ловушки (trap gate) |
| 8 | 1 0 0 0 | Зарезервировано |
| 9 | 1 0 0 1 | 32-разрядный TSS (Свободный)2 |
| 10 | 1 0 1 0 | Зарезервировано |
| 11 | 1 0 1 1 | 32-разрядный TSS (Занятый)2 |
| 12 | 1 1 0 0 | 32-разрядный шлюз вызова (call gate) |
| 13 | 1 1 0 1 | Зарезервировано |
| 14 | 1 1 1 0 | 32-разрядный шлюз прерывания (interrupt gate) |
| 15 | 1 1 1 1 | 32-разрядный шлюз ловушки (trap gate) |

Примечания:

|  |  |
| --- | --- |
| **1.** | Полем типа в дескрипторе системного объекта являются 4 бита в 5-м байте дескриптора или его битами 40..43. |
| **2.** | Для дескрипторов TSS второй по счёту бит в поле типа называется B от слова "Busy" и этот бит отражает занятость задачи: 1/0 - задача занята / свободна. |

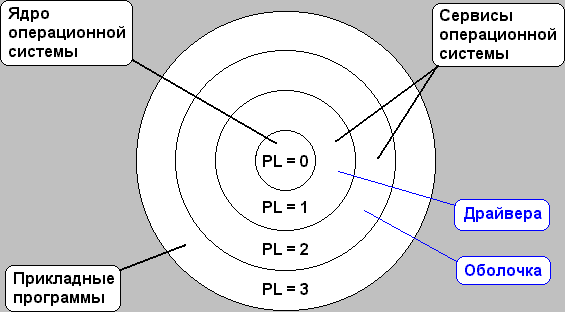
**1.5. Проверка нулевого селектора сегмента.**

        Попытка загрузки нулевого селектора в регистр CS или SS приводит к генерации исключение общей защиты (#GP). Этот селектор можно загружать в регистры DS, ES, FS или GS, но тогда обращение через них к памяти приводит к генерации такого же исключения.

Глава 2. Использование уровней привилегий. http://top.list.ru/counter?id=213421;js=13;r=http%3A//sasm.narod.ru/docs/pm/pm_pr/chap_1.htm;j=false;s=1280*1024;d=24;rand=0.5350673325688855

**2.1. Уровни привилегий.**

        Механизм защиты использует 4 уровня привилегий - от 0 до 3. Большее значение соответствует меньшим привилегиям. На рис. 2-1 показано, как эти уровни привилегий могут быть интерпретированы как кольца защиты. Центр, предназначенный для наиболее привилегированного кода, данных и стека, используется для сегментов, содержащих наиболее важные компоненты системы, такие, как ядро операционной системы. Внешние кольца используются для менее важных частей системы. Для систем, использующих только два из четырёх уровней привилегий, рекомендуется использовать уровни 0 и 3.

  
**Рисунок 2-1. Кольца защиты.**

Примечания:

|  |  |
| --- | --- |
| **1.** | "PL" означает "уровень привилегий" (сокращение от англ. Privilege Level). |
| **2.** | Синим цветом выделено рекомендуемое расположение компонентов операционной системы |

        Когда процессор обнаруживает нарушение использования уровней привилегий, он генерирует исключение общей защиты. Процессор использует три обозначения уровней привилегий:

|  |  |
| --- | --- |
|  | **Текущий уровень привилегий - CPL** (Current privilege level). CPL - это уровень привилегий текущего исполняемого кода. Он хранится в битах 0 и 1 регистров CS и SS. Обычно, CPL равен уровню привилегий сегмента кода, из которого выбираются команды. Процессор меняет CPL, когда управление передаётся сегменту кода с другим уровнем привилегий. CPL интерпретируется немного иначе, когда управление передаётся подчинённому сегменту кода. Подчинённый сегмент кода доступен с тех уровней привилегий, которые численно не меньше, чем DPL подчинённого сегмента кода. CPL не меняется, когда происходит передача управления на подчинённый сегмент кода, имеющий уровень привилегий, отличный от CPL. |
|  | **Запрашиваемый уровень привилегий - RPL** (Requested privilege level). RPL - это уровень привилегий, назначаемый селекторам сегментов. Он хранится в битах 0 и 1 селектора сегмента. Процессор проверяет RPL совместно с CPL, определяя возможность доступа к сегменту. Даже если программа или задача, запрашивающая доступ к сегменту, имеет достаточные привилегии для доступа, доступ будет запрещён, если RPL не имеет достаточных привилегий. Таким образом, если RPL селектора сегмента численно превосходит CPL, то значение RPL будет решающим и наоборот. RPL можно использовать для обеспечения того, что привелигерованный код не обратится к сегменту от имени прикладной программы, пока эта программа не получит достаточных на то привилегий. |
|  | **Уровень привилегий дескриптора - DPL** (Descriptor privilege level). DPL - это уровень привилегий сегмента или шлюза. Он хранится в поле DPL дескриптора сегмента или шлюза. При попытке доступа из текущего сегмента кода к другому сегменту кода или шлюзу, DPL целевого дескриптора сравнивается с CPL и RPL. В зависимости от типа сегмента или шлюза, DPL интерпретируется следующим образом:   |  |  | | --- | --- | | **-** | **Сегмент данных.** DPL определяет наибольший номер уровня привилегий, который программа или задача может иметь для доступа к этому сегменту. Например, если DPL сегмента данных равен 1, то только программы, работающие на уровнях 0 и 1 (т.е. имеющие CPL 0 или 1) могут обращаться к этому сегменту. | | **-** | **Неподчинённый (т.е. обычный) сегмент кода, без использования шлюза вызова.** DPL определяет уровень привилегий, который должна иметь программа или задача для доступа к этому сегменту. Например, если DPL неподчинённого сегмента кода равен 0, то только программа, работающая на CPL = 0 может обратиться к этому сегменту. | | **-** | **Шлюз вызова.** DPL определяет номер наибольшего уровня привилегий, который может иметь текущая программа или задача для доступа к этому шлюзу вызова (правило доступа - такое же, как и для сегмента данных). | | **-** | **Подчинённый или неподчинённый сегмент кода, доступный через шлюз вызова.** DPL определяет наименьший номер уровня привилегий, который должна иметь программа или задача для доступа к этому сегменту. Например, если DPL подчинённого сегмента кода равен 2, то к нему будут иметь доступ только программы с CPL 2 или 3. | | **-** | **TSS.** DPL определяет наибольший номер уровня привилегий, с которого программа или задача может обратиться к этому TSS (правила доступа - такие же, как и для сегмента данных). | |

        Уровни привилегий проверяются, когда селектор дескриптора сегмента загружается в сегментный регистр. Проверки, используемые для доступа к данным отличаются от проверок, используемых для передачи управления другим сегментам кода. Эти два типа проверок рассматриваются далее отдельно.

**2.2. Проверка уровня привилегий при доступе к сегментам данных.**

        Для доступа к операнду в сегменте данных, необходимо загрузить в сегментный регистр (DS, ES, FS или GS) селектор дескриптора сегмента данных. Для этого предназначены команды MOV, POP, LDS, LES, LFS и LGS.   
        Перед тем, как загрузить селектор в сегментный регистр, процессор проверяет уровень привилегий, сравнивая уровень привилегий текущего кода (CPL), RPL селектора и DPL дескриптора. Загрузка селектора производится, когда DPL больше либо равен CPL или RPL, иначе загрузка не произойдёт и процессор сгенерирует исключение общей защиты.

        Адресное пространство процедуры или задачи зависит от значения её CPL. Когда CPL = 0, доступны сегменты данных на всех уровнях привилегий, при CPL = 1 - на уровнях 1, 2 и 3, при CPL = 3 - только на 3-м уровне.   
        Прикладная программа может изменить RPL селектора, например, установить его в 0, и тогда проверка доступа будет осуществляться только по CPL.

**2.3. Доступ к данным в сегменте кода.**

        В некоторых случаях может понадобиться доступ к данным (точнее - чтение), содержащимся в сегменте кода; для этого возможны следующие способы:

|  |  |
| --- | --- |
|  | Загрузить в сегментный регистр данных селектор сегмента кода, разрешённого для чтения. |
|  | Использовать префикс замены сегмента (CS) для чтения сегмента кода, разрешённого для чтения, чей селектор уже загружен в регистр CS. |

**2.4. Проверка уровня привилегий при загрузке регистра SS.**

        Уровень привилегий стека должен быть таким же, как и у кода, т.е. RPL селектора сегмента стека должен быть равен его DPL и CPL сегмента кода. В противном случае, генерируется исключение общей защиты.

Глава 3. Межсегментная и межуровневая передача управления. http://top.list.ru/counter?id=213421;js=13;r=http%3A//sasm.narod.ru/docs/pm/pm_pr/chap_2.htm;j=false;s=1280*1024;d=24;rand=0.949226079622671

        Для передачи управления из одного сегмента кода в другой, селектор целевого сегмента должен быть загружен в регистр CS. При загрузке процессор проверит в дескрипторе этого сегмента его предел, тип и уровень привилегий. Если проверка была успешной, то селектор будет загружен и произойдёт передача управления.   
        Передача управления осуществляется командами JMP, CALL, RET, SYSENTER, SYSEXIT, INT n и IRET.   
        Команда JMP или CALL может ссылаться на другой сегмент любым из 4-х способов:

|  |  |
| --- | --- |
|  | Операнд команды содержит селектор целевого сегмента. |
|  | Операнд указывает на дескриптор шлюза вызова, который содержит селектор целевого сегмента. |
|  | Операнд указывает на TSS, в котором содержится целевой сегмент кода. |
|  | Операнд указывает на шлюз задачи, который указывает на TSS, в котором содержится целевой сегмент кода. |

        Команды SYSENTER и SYSEXIT предназначены для быстрого вызова и возврата системных процедур; их действия описаны в главе 5.   
        Близкие формы команд JMP, CALL и RET передают управление внутри текущего сегмента кода и проверка уровня привилегий не производится. Дальние формы этих команд передают управление в другой сегмент и при этом происходит проверка уровня привилегий.   
        При передаче управления в другой сегмент кода без перехода через шлюз вызова, процессор проверяет 4 типа уровней привилегий и тип сегмента:

|  |  |
| --- | --- |
|  | CPL сегмента кода, из которого происходит передача управления. |
|  | DPL дескриптора сегмента кода, в который происходит передача управления. |
|  | RPL селектора целевого сегмента. |
|  | Флаг подчинения C в дескрипторе целевого сегмента кода ( C=1/0 - сегмент подчинённый / неподчинённый). |

        Правила, по которым процессор проверяет CPL, RPL и DPL зависят от значения флага C.

**Доступ к неподчинённому сегменту кода.**

        При доступе к неподчинённому сегменту кода, CPL вызывающей процедуры должен быть равен DPL целевого сегмента кода, иначе процессор сгенерирует исключение общей защиты.   
        RPL селектора неподчинённого сегмента кода должен быть численно не больше CPL текущего кода. Когда такой селектор загружается в CS, поле привилегий CPL не меняется, даже если RPL имеет другое значение.

**Доступ к подчинённому сегменту кода.**

        При доступе к подчинённому сегменту кода, CPL вызывающей процедуры должен быть численно не меньше DPL целевого сегмента, иначе процессор генерирует исключение общей защиты. Для подчинённого сегмента поле RPL селектора не учитывается.   
        Для подчинённого сегмента кода DPL показывает численно наименьший уровень привилегий, которая может иметь вызывающая процедура для доступа в него.   
        При передаче управления подчинённому сегменту кода, CPL не меняется, даже если DPL целевого сегмента меньше, чем CPL. Эта ситуация - единственная, когда CPL отличается от DPL. Кроме того, т.к. CPL не меняется, то не происходит смены стека.   
        Подчинённые сегменты используются для таких модулей, как математические библиотеки и обработчики исключений, которые поддерживают приложения, но не требуют доступа к защищённым ресурсам системы. Эти модули являются частью ОС, но они могут выполняться на численно большем уровне привилегий (т.е. менее привилегированном). То, что CPL не меняется при переходе на подчинённый сегмент, ограничивает прикладную программу от доступа к более привилегированным ресурсам (коду и данным).   
        Большая часть сегментов кода - не подчинённые и в них передавать управление можно только на их уровне привилегий, кроме случаев перехода через шлюзы вызова, описанного в следующей главе.

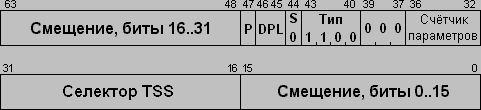
Глава 4. Шлюз вызова. http://top.list.ru/counter?id=213421;js=13;r=http%3A//sasm.narod.ru/docs/pm/pm_pr/chap_3.htm;j=false;s=1280*1024;d=24;rand=0.01391704176355979

        Для обеспечения контролируемого доступа к сегменту кода на другом уровне привилегий, процессор предоставляет 4 специальных типа дескрипторов, так называемых шлюзов:

|  |  |
| --- | --- |
|  | Шлюз вызова (Call gate) |
|  | Шлюз ловушки (Trap gate) |
|  | Шлюз прерывания (Interrupt gate) |
|  | Шлюз задачи (Task gate) |

        Шлюзы ловушки и прерывания используются в дескрипторной таблице прерываний (IDT) и описаны в разделе ["Прерывания в защищённом режиме"](http://sasm.narod.ru/docs/pm/pm_int/main.htm). Шлюз задачи используется для передачи управления задаче и подробно описан в разделе ["Мультизадачность"](http://sasm.narod.ru/docs/pm/pm_tss/main.htm). Здесь же описан только шлюз вызова.

        Шлюз вызова позволяет передавать управление с одного уровня привилегий на другой. Также шлюз вызова полезен при передаче управления между 16- и 32-разрядным кодом.   
        На рис. 4-1 приведен формат дескриптора шлюза вызова.

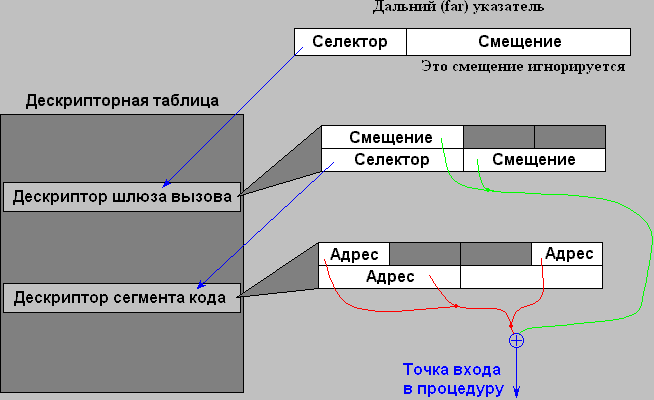
  
**Рисунок 4-1. Формат дескриптора шлюза вызова**

        Этот дескриптор можно размещать в GDT и LDT, но не в IDT. Дескриптор шлюза вызова выполняет 6 функций:

|  |  |
| --- | --- |
|  | Определяет сегмент кода, к которому будет доступ. |
|  | Определяет точку входа в этом сегменте. |
|  | Определяет уровень привилегий, который должна иметь вызывающая процедура. |
|  | При смене стека, он указывает число необязательных параметров, копируемых процессором из одного стека в другой. |
|  | Определяет размер значений, записываемых в целевой стек: 16-разрядный шлюз использует 16-разрядные значения, 32-разрядный - 32-разрядные. |
|  | Определяет присутствие шлюза вызова. |

        Поле селектора в шлюзе вызова определяет целевой сегмент кода, смещение задаёт точку входа. DPL определяет уровень привилегий шлюза вызова, который должна иметь процедура, чтобы передать управление через него.   
        Флаг P определяет присутствие дескриптора шлюза вызова в памяти. Поле счётчика определяет число параметров (16- или 32-разрядных, в зависимости от разрядности шлюза), копируемых из стека вызывающей процедуры в стек вызываемой процедуры (если происходит смена стека).   
        Как правило, флаг P (присутствие дескриптора) всегда установлен, если же он сброшен (т.е. равен 0), то при обращении к шлюзу процессор генерирует исключение неприсутствующего сегмента (#NP). Используя бит P система может определить число переходов через этот шлюз. Для этого бит присутствия шлюза сбрасывается, а функции обработчика исключения #NP сводятся к увеличению на 1 счётчика переходов через шлюз и установке бита P. После возврата из такого обработчика процессор снова попытается передать управление через шлюз и сделает это, если больше никаких нарушений в нём не обнаружит.

        Шлюз вызова по сути определяет дальнюю точку входа (селектор:смещение) для процедуры. При передаче управления через шлюз вызова, вызывающая процедура должна использовать далёкий адрес, селектор которого указывает на шлюз, смещение при этом будет проигнорировано процессором (см. рис. 4-2). Селектор, указанный в шлюзе вызова, может выбирать дескриптор из GDT или LDT.

  
**Рисунок 4-2. Использование шлюза вызова**

        При передаче управления через шлюз вызова, используются 4 типа уровней привилегий:

|  |  |
| --- | --- |
|  | CPL, |
|  | RPL селектора шлюза вызова, |
|  | DPL дескриптора шлюза вызова, |
|  | DPL дескриптора целевого сегмента кода. |

        Также учитывается флаг C (подчинение) в дескрипторе сегмента кода.

        Правила проверки уровня привилегий отличаются, в зависимости от того, какой командой было передано управление (см. таб. 4-1):

**Таблица 4-1. Правила проверки уровня   
привилегий для шлюза вызова.**

|  |  |
| --- | --- |
| **Команда** | **Правила проверки привилегий** |
| CALL | CPL ≤ DPL шлюза вызова  RPL ≤ DPL шлюза вызова  DPL целевого подчинённого сегмента ≤ CPL  DPL целевого неподчинённого сегмента ≤ CPL |
| JMP | CPL ≤ DPL шлюза вызова  RPL ≤ DPL шлюза вызова  DPL целевого подчинённого сегмента ≤ CPL  DPL целевого неподчинённого сегмента = CPL |

        Поле DPL дескриптора шлюза вызова определяет наибольший номер уровня привилегий, с которого вызывающая процедура может обращаться к шлюзу. Таким образом, для доступа к шлюзу CPL вызывающей процедуры должен быть не больше DPL шлюза.   
        RPL селектора шлюза вызова должен удовлетворять тем же условиям, что и CPL вызывающей процедуры, т.е. RPL должен быть не больше DPL шлюза.

        Если проверка уровня привилегий между вызывающей процедурой и шлюзом вызова успешна, то процессор проверяет DPL дескриптора сегмента кода, сравнивая его с CPL вызывающей процедуры.   
        Правила проверки уровня привилегий отличаются для команд JMP и CALL. Только команда CALL может использовать шлюз вызова для передачи управления более привилегированному неподчинённому сегменту кода (т.е., чей DPL ≤ CPL). Команда JMP может использовать шлюз вызова только для передачи управления в неподчинённый сегмент кода с DPL = CPL. Команды JMP и CALL обе могут передавать управление более привилегированному подчинённому сегменту кода (т.е., чей DPL ≤ CPL).   
        Если вызов производится на более привилегированный неподчинённый сегмент кода, то CPL опускается до значения DPL и производится переключение стека. Если же передача управления происходит в подчинённый сегмент кода, то CPL не меняется и стек не переключается.

        Шлюзы вызова позволяют одному сегменту кода иметь доступ к процедурам, располагающимся на разных уровнях привилегий. Например, операционная система, находящаяся в сегменте кода, может иметь некоторые сервисы, которые могут использоваться как ею самой, так и прикладными программами (например, процедуры по обработке ввода/вывода символов).

**5.1. Переключение стека.**

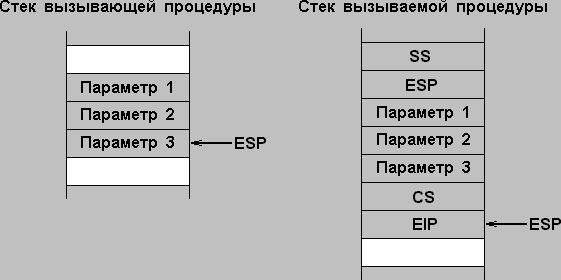
        Когда шлюз вызова используется для передачи управления более привилегированному неподчинённому сегменту кода (т.е., чей DPL < CPL), процессор автоматически переключает стек на другой, находящийся на уровне привилегий вызываемого сегмента. Благодаря переключению стека такая ситуация, как нехватка стека, не произойдёт для привилегированной процедуры, к тому же это защищает привилегированную процедуру от вмешательства в её стек со стороны вызывающей процедуры.   
        Каждая задача должна определить до 4-х стеков: один - для прикладного кода, работающего на уровне привилегий 3, и один для каждого уровня привилегий - 0, 1 и 2. Если используются только два уровня - 0 и 3, то нужно определить только стек для 0-го. Каждый из этих стеков (селектор и смещение) определён в сегменте состояния задачи. Их значения предназначены только для чтения и процессор не меняет их в процессе выполнения задачи; они предназначены для создания новых стеков при вызове более привилегированных процедур. Значения, описанные в полях для SS:EIP соответствуют 3-му уровню привилегий. Для такого стека не выделено отдельное поле, потому что процессор не допустит передачи управления с уровня 0..2 на 3 (разве что, только при возврате из процедуры).   
        Каждый раз, при переключении на меньший уровень привилегий, процессор создаёт новый стек, используя значения из TSS и после возврата из привилегированной процедуры, этот стек больше не используется (обычно - уничтожается).   
        О создании и удалении стеков должна заботиться операционная система. Стеки должны быть достаточного размера, чтобы хранить:

|  |  |
| --- | --- |
|  | Содержимое регистров вызывающей процедуры: SS, ESP, CS и EIP. |
|  | Параметров и временных переменных, требуемых вызванной процедуре. |
|  | Регистра EFLAGS и кода ошибки, если есть явный вызов обработчика исключения или прерывания. |

        Стек также должен быть достаточно большой, чтобы учитывать вызовы внутри вызванной процедуры, вложенные процедуры и задачи и обработку прерываний и исключений.   
        Если операционная система не использует мультизадачность, она всё равно должна определить как минимум один TSS для переключения стеков.

        Когда процедура вызывает через шлюз вызова другую, более привилегированную процедуру, процессор выполняет следующие действия:

|  |  |
| --- | --- |
| **1.** | Используя DPL целевого сегмента кода (это - будет новым значением CPL), выбирает из TSS указатель нового стека SSi:ESPi. |
| **2.** | Проверяет селектор и смещение нового стека, при этом любые нарушения приводят к генерации исключения недопустимого TSS (#TS). |
| **3.** | Проверяет дескриптор сегмента стека на соответствующие привилегии и его тип. В случае нарушения - генерирует исключение недопустимого TSS. |
| **4.** | Временно сохраняет текущие значения SS и ESP. |
| **5.** | Загружает селектор сегмента и смещение нового стека в SS:ESP. |
| **6.** | Помещает в новый стек временно сохранённые значения для SS:ESP (см. рис. 5-1) |
| **7.** | Копирует указанное в шлюзе вызова число параметров из стека вызывающей процедуры в новый стек. Если счётчик параметров равен 0, то параметры не копируются. |
| **8.** | Помещает в новый стек адрес возврата в вызывающую процедуру (т.е. текущее значение CS:EIP). |
| **9.** | Загружает новые значения CS:EIP из шлюза вызова и начинает выполнение вызванной процедуры. |

  
**Рисунок 5-1. Переключение стека при межуровневом вызове.**

        Параметр "счётчик" в шлюзе вызова указывает число элементов данных (от 0 до 31), которые процессор должен скопировать из стека вызывающей процедуры в стек вызываемой процедуры. Размер параметров зависит от разрядности шлюза вызова.

**5.2. Возврат из вызванной процедуры.**

        Команда RET может использоваться для выполнения близкого возврата и дальнего возврата на тот же или другой уровень привилегий. Близкий возврат возвращает управление внутри текущего сегмента кода и при этом процессор проверяет только предел сегмента.   
        При дальнем возврате на тот же уровень привилегий, процессор проверяет селектор, дескриптор, на который он указывает и новое значение EIP (чтобы оно было в пределах сегмента).   
        Дальний возврат со сменой уровня привилегий разрешается только на менее привилегированный уровень.   
        Процессор использует поле RPL из сохранённого в стеке значения CS (см. рис. 5-1), чтобы определить, нужен ли возврат на менее высокий уровень привилегий. Если это значение RPL численно больше CPL, то происходит возврат на другой уровень привилегий.

**5.3. Быстрый вызов системных процедур командами SYSENTER и SYSEXIT.**

        Команды SYSENTER и SYSEXIT появились в процессоре Pentium II и они предназначены для быстрого вызова системных процедур. Команда SYSENTER позволяет процедуре на не нулевом уровне привилегий вызывать системную процедуру, находящуюся на уровне привилегий 0. Команда SYSEXIT позволяет быстро вернуть управление с 0-го уровня привилегий на любой другой.   
        Эти две команды используются совместно, они действуют не так как команды вызова и возврата и не сохраняют значений в стеке. Параметры этих команд указываются не в операндах, а в регистрах MSR и регистрах общего назначения. Для команды SYSENTER параметры указываются следующим образом:

|  |  |
| --- | --- |
|  | Целевой сегмент кода - в SYSENTER\_CS\_MSR. |
|  | Целевой указатель команд - в SYSENTER\_EIP\_MSR. |
|  | Сегмент стека - вычисляет команда, добавляя 8 к значению в SYSENTER\_CS\_MSR. |
|  | Указатель стека - в SYSENTER\_ESP\_MSR. |

        Для SYSEXIT параметры задаются так:

|  |  |
| --- | --- |
|  | Целевой сегмент кода - значение из SYSENTER\_CS\_MSR + 16. |
|  | Целевой указатель кода - в EDX. |
|  | Сегмент стека - значение из SYSENTER\_CS\_MSR + 24. |
|  | Указатель стека - в ECX. |

        Значения, находящиеся в регистрах MSR, процессором проверяются при их загрузке; при выполнении команд SYSENTER и SYSEXIT проверки сведены к минимуму, обращение к памяти в дескрипторы не происходит и передача управления происходит быстрее.   
        Прочие параметры, используемые при передаче управления и возврате этими двумя командами, должны передаваться программой явно.

Глава 6. Привилегированные команды. http://top.list.ru/counter?id=213421;js=13;r=http%3A//sasm.narod.ru/docs/pm/pm_pr/chap_5.htm;j=false;s=1280*1024;d=24;rand=0.5923637466965156

      Некоторые команды, так называемые привилегированные команды, запрещены для использования прикладными программами. Эти команды управляют системными функциями, например, загрузкой системных регистров и они могут выполняться только на нулевом уровне привилегий (т.е. когда CPL = 0). При попытке выполнить их на другом уровне привилегий, генерируется исключение общей защиты.

      Привилегированными являются следующие команды:

|  |  |
| --- | --- |
|  | LGDT - Загрузка регистра GDTR (Load GDT register). |
|  | LLDT - Загрузка регистра LDTR (Load LDT register). |
|  | LTR - Загрузка регистра TR (Load Task Register). |
|  | LIDT - Загрузка регистра IDTR (Load IDT register). |
|  | MOV (в регистр управления) - Чтения и запись регистра управления. |
|  | LMSW - Загрузка младшей половины регистра CR0 (Load Machine Status Word). |
|  | CLTS - Сброс флага переключения задачи в регистре CR0 (Clear Task-Switched flag). |
|  | MOV (в регистр отладки) - Чтение и запись в регистр отладки. |
|  | INVD - Сброс кэша (объявление его недостоверным), без обратной записи. |
|  | WBINVD - Сброс кэша (объявление его недостоверным), с обратной записью. |
|  | INVLPG - Сброс (invalidate) элемента TLB. |
|  | HLT - Остановка процессора. |
|  | RDMSR - Чтение из регистра спецификации модели MRS (Read Model-Specific Register). |
|  | WRMSR - Запись в регистр спецификации модели MSR (Write Model-Specific Register). |
|  | RDPMC - Чтение счётчика мониторинга производительности (Read Performance-Monitoring Counter). |
|  | RDTSC - Чтение счётчика тактов процессора (Read Time-Stamp Counter). |

      Флаги PCE и TSD в регистре CR4 (биты 4 и 2), могут разрешить выполнение команд RDPMC и RDTSC на любом уровне привилегий.

      При работе в защищённом режиме, процессор проверяет все указатели для обеспечения защиты сегментов и поддержки изоляции на разных уровнях привилегий. Проверка указателя состоит из следующих действий:

|  |  |
| --- | --- |
| **1.** | Проверка прав доступа. |
| **2.** | Проверка доступа чтения/записи. |
| **3.** | Проверка предела. |
| **4.** | Проверка возможности доступа к сегменту "владельца" указателя. |
| **5.** | Проверка выравнивания. |

      При выполнении команды процессор автоматически выполняет действия 1, 2 и 3. Программа может явно использовать проверку 4, используя команду ARPL. Пятая проверка выполняется автоматически на уровне привилегий 3, если включена проверка выравнивания.

**Проверка прав доступа (команда LAR).**

      Когда процессор обращается к сегменту через дальний указатель, он выполняет проверку прав доступа целевого дескриптора, чтобы определить, является ли допустимым для данной операции использование этого сегмента (по его типу и уровню привилегий). Если процессор обнаруживает нарушение, то он генерирует исключение.   
      Чтобы избежать исключения, программа перед обращением к сегменту, может проверить его права доступа командой LAR (Load Access Rights).

**Проверка доступа чтения/записи (команды VERR и VERW).**

      При обращении к сегменту, процессор проверяет возможность чтения и записи для данного сегмента. Программа может сделать это явно, используя команды VERR - проверка на чтение (VERify for Reading) и VERW - проверка на запись (VERify for Writing).

**Проверка предела (команда LSL).**

      При обращении к сегменту, процессор проверяет проверку его предела, чтобы определить, является ли допустимым заданное значение смещения. Программа может выполнить эту проверку, используя команду LSL - загрузить предел сегмента (Load Segment Limit).

**Проверка привилегии доступа вызывающей процедуры (команда ARPL).**

      Поле запрошенного уровня привилегий RPL в селекторе содержит уровень привилегий вызывающей процедуры, т.е. её CPL. Прикладная процедура может вызывать системные процедуры (если это позволено) и передавать селекторы сегментов, к которым ей нужно обращаться.   
      Программа может установит значение RPL любым, но при доступе к сегменту вместе с RPL будет учитываться CPL. То значение, которое будет больше, определит итоговый уровень привилегий, с которого обращается программа за доступом к сегменту.   
      Прикладная программа не может обратиться к системным сегментам, даже если она установит значение RPL в 0 - не позволит CPL. Однако, прикладная программа может передать системной процедуре селектор с нулевым RPL и таким образом воспользоваться системным сегментом.   
      Команда ARPL (Adjust RPL - выровнять RPL) может использоваться для того, чтобы скорректировать значение RPL переданного селектора. Если RPL будет меньше, чем CPL вызывающей процедуры, то эта команда установит RPL равным этому CPL. Значение CPL вызывающей команды можно взять из стека - там будет храниться селектор кода вызывающей процедуры (он входит в состав дальнего адреса возврата из процедуры).

**Проверка выравнивания.**

      Когда CPL = 3, можно проверять выравнивание при доступе к памяти - это делается установкой флага AM в CR0 и AC в EFLAGS. Невыравненное обращение к памяти генерирует исключение выравнивания (#AC). На уровнях 0, 1 и 2 это исключение не генерируется. См. описание этого исключения, где приведена таблица с правилами, нарушение которых приводит к исключению выравнивания.

**Свойства системных команд.**

**Таблица 6-1. Список свойств системных команд**

|  |  |  |  |
| --- | --- | --- | --- |
| **Команда** | **Описание** | **Полезна для прикладной программы** | **Защищена от прикладной программы** |
| LLDT | Загрузка регистра LDTR | Нет | Да |
| SLDT | Сохранение регистра LDTR | Нет | Нет |
| LGDT | Загрузка регистра GDTR | Нет | Да |
| SGDT | Сохранение регистра GDTR | Нет | Нет |
| LTR | Загрузка регистра задачи TR | Нет | Да |
| STR | Сохранение регистра задачи | Нет | Нет |
| LIDT | Загрузка регистра IDTR | Нет | Да |
| SIDT | Сохранение регистра IDTR | Нет | Нет |
| MOV CRi | Загрузка и сохранение регистров управления | Нет | Да |
| SMSW | Сохранение MSW | Да | Нет |
| LMSW | Загрузка MSW | Нет | Да |
| CLTS | Сброс флага TS в CR0 | Нет | Да |
| ARPL | Коррекция поля RPL | Да[1](http://sasm.narod.ru/docs/pm/pm_pr/chap_6.htm#notes) | Нет |
| LAR | Загрузка прав доступа | Да | Нет |
| LSL | Загрузка предела сегмента | Да | Нет |
| VERR | Проверка на чтение | Да | Нет |
| VERW | Проверка на запись | Да | Нет |
| MOV DBi | Загрузка и сохранение регистров отладки | Нет | Да |
| INVD | Сброс кэша без обратной записи | Нет | Да |
| WBINVD | Сброс кэша с обратной записью | Нет | Да |
| INVLPG | Сброс элемента TLB | Нет | Да |
| HLT | Остановка процессора | Нет | Да |
| LOCK | Префикс блокировки шины | Да | Нет |
| RSM | Возврат из режима системного управления | Нет | Да |
| RDMSR[3](http://sasm.narod.ru/docs/pm/pm_pr/chap_6.htm#notes) | Чтение регистра MSR | Нет | Да |
| WRMSR[3](http://sasm.narod.ru/docs/pm/pm_pr/chap_6.htm#notes) | Запись регистра MSR | Нет | Да |
| RDPMC[4](http://sasm.narod.ru/docs/pm/pm_pr/chap_6.htm#notes) | Чтение счётчика мониторинга производительности | Да | Да[2](http://sasm.narod.ru/docs/pm/pm_pr/chap_6.htm#notes) |
| RDTSC[3](http://sasm.narod.ru/docs/pm/pm_pr/chap_6.htm#notes) | Чтение счётчика тактов | Да | Да[2](http://sasm.narod.ru/docs/pm/pm_pr/chap_6.htm#notes) |

Примечания:

|  |  |
| --- | --- |
| **1.** | Полезна для программ, работающих на уровнях привилегий 1 и 2. |
| **2.** | Флаги TSD и PCE в регистре CR4 управляют доступом к этим командам для программ, работающих на 3-м уровне привилегий. |
| **3.** | Эти команды впервые появились в процессоре Pentium. |
| **4.** | Эта команда появились в процессоре Pentium Pro и Pentium MMX. |

Глава 7. Защита на уровне страниц. http://top.list.ru/counter?id=213421;js=13;r=http%3A//sasm.narod.ru/docs/pm/pm_pr/chap_6.htm;j=false;s=1280*1024;d=24;rand=0.9306873783967442

      При использовании защиты на уровне страниц, каждое обращение к памяти проверяется до того, как оно произойдёт и при выявлении нарушений, генерируется исключение страничного нарушения. Эта проверка происходит параллельно трансляции адреса и не приводит к потере производительности.   
      Процессор выполняет две проверки защиты на уровне страниц:

|  |  |
| --- | --- |
|  | Ограничение адресного пространства (системный и пользовательский режимы). |
|  | Тип страницы (только чтение или чтение/запись). |

      Защитная информация для страниц хранится в двух флагах каждого элемента каталога или таблицы страниц:

|  |  |
| --- | --- |
|  | флаг чтения/записи (бит 1 элемента), |
|  | флаг система/пользователь (user/supervisor) (бит 2 элемента). |

**Ограничение адресного пространства.**

      Защитный механизм страниц, позволяющий ограничить доступ к страницам, основан на двух уровнях привилегий:

|  |  |
| --- | --- |
|  | **Режим системы** (supervisor mode - флаг U/S равен 0) - Наиболее привилегированный. Для ядра операционной системы и защищённых страниц кода (например, драйверы) и данных (например, таблицы страниц). |
|  | **Режим пользователя** (user mode - флаг U/S равен 1) - Не привилегированный. Для прикладного кода и данных. |

      Если процессор находится на нулевом уровне привилегий, то он работает в режиме системы, если на любом другом уровне (1, 2 или 3) - то в режиме пользователя.   
      В режиме системы процессору доступны все страницы, в режиме пользователя - только пользовательские.   
      Если операционная система использует плоскую модель памяти, в которой сегменты могут перекрываться, то защита на уровне сегментов не всегда действует, т.к. сегмент прикладной программы может быть отображён на системный сегмент и единственной защитой по привилегиям остаётся защита на уровне страниц.

**Тип страницы.**

      Есть два типа страниц:

|  |  |
| --- | --- |
|  | С доступом только для чтения (флаг R/W равен 0). |
|  | С доступом по чтению и записи (флаг R/W равен 1). |

      Когда процессор находится в режиме системы и флаг WP в CR0 сброшен (это его состояние после инициализации), все страницы являются доступными для чтения и записи (защита от записи игнорируется). Когда процессор находится в режиме пользователя, он может записывать только в пользовательские страницы с доступом по чтению и записи. Системные страницы для процессора в таком режиме не доступны и при попытке обращения к ним, процессор генерирует исключение страничного нарушения.   
      Процессоры семейства P6, Pentium и Intel486 позволяют защитить пользовательские страницы от записи процессором, находящимся с режиме системы. Это делается установкой в 1 флага WP в регистре CR0.

      Атрибуты защиты для страниц в таблице страниц, могут отличаться от атрибутов, установленных в самой таблице страниц. Совместное действие этих атрибутов, при сброшенном флаге WP, показано в таблице 4-2.

**Совместная защита на уровне сегментов и страниц.**

      При разрешённом страничном преобразовании, процессор сначала проверяет защиту на уровне сегментов, а потом - на уровне страниц. Если обнаруживается ошибка, процессор не производит обращения к памяти и генерирует исключение.   
      Защита на уровне страниц не перекрывает защиту на уровне сегментов. Например, сегмент кода по определению на записываемый; установка страниц, на которые он отображён, как страницы с доступом по чтению и записи всё равно не делает возможным запись в сегмент кода.

**Таблица 4-2. Совместная защита каталога и таблиц страниц.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Элемент каталога страниц** | | **Элемент таблицы страниц** | | **Совместный эффект** | |
| **Приви- легия** | **Тип доступа** | **Приви- легия** | **Тип доступа** | **Приви- легия** | **Тип доступа** |
| User | Только чтение | User | Только чтение | User | Только чтение |
| User | Только чтение | User | Чтение и запись | User | Только чтение |
| User | Чтение и запись | User | Только чтение | User | Только чтение |
| User | Чтение и запись | User | Чтение и запись | User | Чтение и запись |
|  | | | | | |
| User | Только чтение | System | Только чтение | System | Чтение и запись\* |
| User | Только чтение | System | Чтение и запись | System | Чтение и запись\* |
| User | Чтение и запись | System | Только чтение | System | Чтение и запись\* |
| User | Чтение и запись | System | Чтение и запись | System | Чтение и запись\* |
|  | | | | | |
| System | Только чтение | User | Только чтение | System | Чтение и запись\* |
| System | Только чтение | User | Чтение и запись | System | Чтение и запись\* |
| System | Чтение и запись | User | Только чтение | System | Чтение и запись\* |
| System | Чтение и запись | User | Чтение и запись | System | Чтение и запись\* |
|  | | | | | |
| System | Только чтение | System | Только чтение | System | Чтение и запись\* |
| System | Только чтение | System | Чтение и запись | System | Чтение и запись\* |
| System | Чтение и запись | System | Только чтение | System | Чтение и запись\* |
| System | Чтение и запись | System | Чтение и запись | System | Чтение и запись |

Примечание:

|  |  |
| --- | --- |
| \* | Если флаг WP в CR0 установлен, тип доступа определяется флагом R/W элемента каталога или таблиц страниц. |