УДАЛЕННО УПРАВЛЯЕМ BIOS SETUP

КРИС КАСПЕРСКИ

Каждый из вас хотя бы раз в жизни сталкивался с необходимостью войти в BIOS Setup и слегка его «подкрутить» или починить «рухнувшую» Windows NT, Linux/FreeBSD. Традиционно эта задача решается при помощи мыши и клавиатуры, но что делать, если сервер физически недоступен?

Компьютеры семейства IBM PC долгое время рассматривались как недорогие рабочие станции и сервера на их основе начали строить лишь недавно. Разработчики увеличили количество процессоров, добавили поддержку коррекции памяти, отказоустойчивые дисковые массивы и прочие прелести, однако полное превращение в сервер так и не наступило. В частности, сохранилась проблема удаленного администрирования. Операционные системы семейства Windows NT поддерживают удаленный контроль лишь формально. Даже такие программы, как Remote Admin, выполняют ограниченный спектр простейших операций, и на полноценное обслуживание сервера по сети не способны. В мире UNIX дела обстоят чуть-чуть получше, но проблемы все равно есть.

Вот, например, BIOS отказывается грузиться, предлагая нажать <F1> для входа в BIOS Setup или <F2> для загрузки с параметрами по умолчанию (см. **рис. 1**). Но сервер находится в другом конце города, да еще в помещении, ключей от которого у администратора нет. Знакомая ситуация, не правда ли? Другой вариант: после установки очередного пакета обновления операционная система «умерла», стала жертвой хакерской атаки или просто зависла. Во всех этих случаях стандартные средства удаленного управления уже не работают и приходится приближаться к серверу вплотную, что достаточно затруднительно. Даже если сервер расположен на соседнем этаже, намного предпочтительнее управлять им без отрыва от своего любимого кресла, чем бегать с дискетами (лазерными дисками) туда-сюда.

И это действительно можно сделать! Существуют по меньшей мере три пути, о которых я и хочу рассказать.

Удаленный контроль за BIOS

Порядок загрузки BIOS в общих чертах выглядит так. Первым получает управление BOOT-block (загрузочный блок или первичный загрузчик, не путать с boot-сектором!). Он выполняет инициализацию основного оборудования (оперативная память, контроллер прерываний, системный таймер и т. д.), сканирует ISA-шину и подключает BIOS всех обнаруженных устройств (например, SCSI-контроллеров, видео-, сетевых карт и т. д.). Перед завершением своей работы BOOT-block распаковывает основной код BIOS (так называемый BIOS extensions, или вторичный загрузчик) и передает ему управление. Вторичный загрузчик сканирует PCI-шину и выполняет окончательную инициализацию оборудования – распознает IDE-диски, при необходимости выводит интерактивный редактор BIOS Setup, распределяет системные ресурсы между PnP-устройствами и, наконец, считывает boot-сектор с гибкого или жесткого диска.

Таким образом, BIOS, установленные на картах расширения, получают управление на самой ранней стадии инициализации, задолго до того, как начинается подсчет контрольной суммы CMOS или распаковка вторичного загрузчика. Кстати говоря, большинство утилит «прожига» BIOS не трогают BOOT-block и даже если прожиг прошел неудачно, ISA-слоты расширения все-таки инициализируются. С PCI-слотами все обстоит намного сложнее, и в общем случае они доступны только из вторичного загрузчика (а он гибнет при неудачном прожиге). Некоторые производители, например ASUS, включают в BOOT-block специальный драйвер для работы с

PCI-шиной, чтобы материнская плата могла инициализировать видеокарту и хоть что-то вывести на экран, даже если основной код BIOS повержен. Но мне не известен ни один BIOS, BOOT-block которого мог бы работать с шиной AGP или PCI-express.

Следовательно, все, что нам нужно, – изготовить «фиктивную» ISA- или PCI-карту, установить на ней «свой» BIOS и запрограммировать его на удаленное управление. Когда-то я «дорабатывал» древние сетевые карты (которые просто выбрасывались), превращая их в «пульт» удаленного управления, позволяющий редактировать настройки BIOS по локальной сети. Это совсем несложно сделать! Достаточно уметь программировать на Ассемблере и чуть-чуть разбираться в архитектуре «железа» (см. **рис. 2**).

Впрочем, корпеть над отладчиком совсем необязательно, все можно купить и готовое. Такие платы (они называются Remote Boards) выпускает множество фирм. Обычно они представляют из себя стандартную VGA-карту с интегрированным COM-портом, к которому подключается внешний модем. В некоторых моделях имеется Ethernet-порт. Его можно воткнуть в DSL-модем или соединить со Switch. Через эти порты передается копия экрана на удаленный монитор и принимаются команды от клавиатуры, в результате чего IBM PC превращается в самый настоящий «мейнфрейм» и физического доступа к нему уже не требуется (см. **рис. 3**)!

Большой популярностью пользуется модель Remote Insight от Hewlett-Packard, которая вставляется в PCI-слот и управляется через 10/100 Мбит Ethernet-порту. Она поддерживает как текстовые, так и графические режимы (вплоть до 1280x1024/256 цветов), питается от внешнего источника, что позволяет ей «нажимать» на кнопки «Power» и «Reset». В дополнение к удаленной мыши и клавиатуре имеется возможность подключать удаленный дисковод и привод CD-ROM, без которых не обходится ни одна переустановка системы. Это просто фантастика! Всегда можно загрузиться с Live CD и посмотреть, что случилось с сервером и сохранить уцелевшие данные на любой носитель, который только будет под рукой. Это усиливает безопасность системы, поскольку сервер, оснащенный «Remote Insight», может вообще не иметь никаких съемных носителей!

Кстати, о безопасности. Remote Insight поддерживает SSL и 128-битное шифрование, что позволяет ему функционировать даже на незащищенных каналах (а других каналов в распоряжении рядового администратора зачастую просто не оказывается).

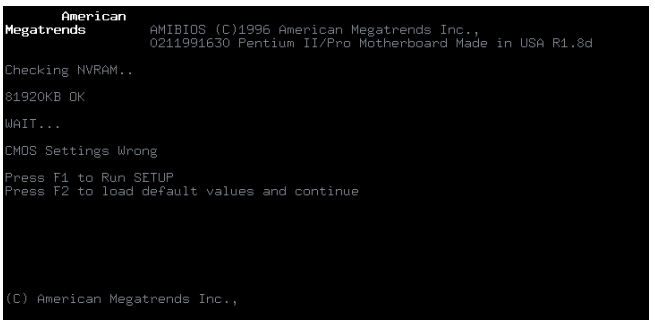

Рисунок 1. BIOS отказывается грузиться до тех пор, пока не будет нажата клавиша <F1> или <F2>

Все управление происходит либо через telnet, либо через веб-браузер. Как будет удобнее администратору. На сервере может быть установлена практически любая операционная система: Windows 2000/2003 (Advanced Server, Data Center, Terminal Server, Standard или Enterprise Edition), Novell NetWare 5.1, 6.0, Red Hat Advanced Server2.1, Red Hat Linux 7.3/8.0, SuSE Linux Enterprise Server V7/V8 и некоторые другие (см. **рис. 4**).

Карту можно приобрести в магазине или заказать по Интернету непосредственно в самой Hewlett-Packard. Она обойдется в $399, которые явно стоят того! В принципе можно найти производителя и подешевле, но в отношении цена/функциональность этой карте равных нет, тем не менее она далека от идеала. Исходных текстов прошивки нам никто не даст, и доработать «напильником» под свои конкретные нужны ее не удастся (теоретически это возможно, но очень затруднительно). К тому же качество реализации протоколов шифрования находится под большим вопросом. Возможно, в карте присутствуют отладочные люки или переполняющиеся буферы, которые позволят атакующему захватить штурвал управления в свои руки (см. **рис. 5**)!

Этих недостатков лишена PC Weasel 2000 от одноименной компании. Вместе с самой платой покупатель получает полный исходный код прошивки и лицензию на право его изменения. Это все та же самая VGA-плата, только вместо Ethernet-порта на ней находится контроллер UART (он же стандартный COM-порт типа 16550). К сожалению, ее функциональность намного беднее. Поддерживаются только текстовые видеорежимы и отсутствуют удаленные приводы, правда, сохраняется возможность «нажать» серверу на «Reset» или посмотреть POST-коды, чтобы сразу оценить масштабы неисправности (см. **рис. 6**).

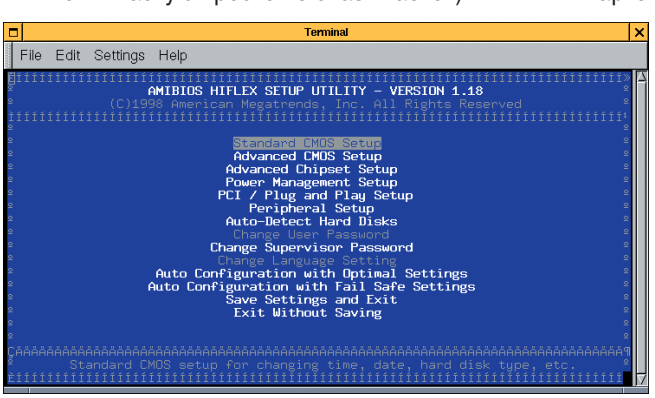

Рисунок 2. Удаленное редактирование настроек BIOS Setup по терминалу – это реальность!

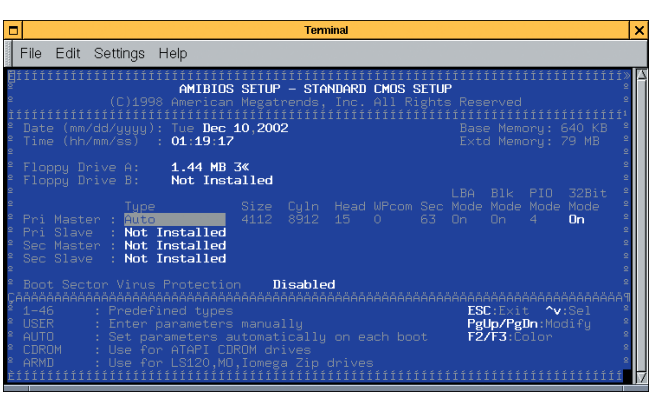

Рисунок 3. Удаленная настройка дисков

Рисунок 4. Плата удаленного управления Remote Insight от Hewlett-Packard

Рисунок 5. Еще одна плата удаленного управления – PC Weasel 2000

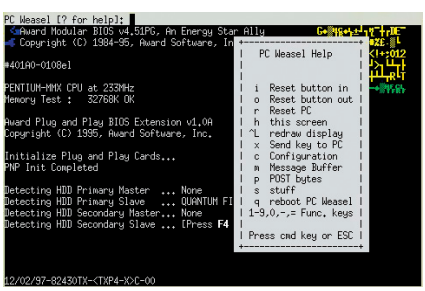

Рисунок 6. Инженерное меню, высвечиваемое PC Weasel 2000

ISA-вариант обойдется вам в $250, а PCI – во все $350. Не слишком ли большая цена за открытую лицензию при урезанной функциональности? Не торопитесь с выводами. Исходные тексты – великая штука! Можно купить одну плату и установить ее на неограниченном количестве машин. Клонировать аппаратное обеспечение нам не понадобится. Если слегка переделать прошивку можно обойтись и стандартными компонентами, но об этом – чуть позже. Сначала познакомимся с диаметрально противоположным классом устройств удаленного управления, среди которых, возможно, притаилось устройство вашей мечты (см. **рис. 7**).

KVM, или Удаленный контроль продолжается

Главный недостаток VGA-плат с модифицированным BIOS состоит в том, что они требуют вскрытия корпуса сервера, что не всегда желательно. К тому же техника перехвата изображения и эмуляция клавиатурного ввода далека от идеала и чрезвычайно конфликтна. KVM-коммутаторы исповедуют совершенно иной подход. Свое название они получили по трем первым буквам: Keyboard, Video-monitor и Mouse. Коммутатор представляет собой автономное устройство, подключаемое к компьютеру через стандартные PS/2 и DB-15 VGA-коннекторы. Их сигнал преобразуется в цифровой поток и передается на соседний KVM-терминал, подключенный к удаленному компьютеру. Грубо говоря, мы как бы подключаем клавиатуру, мышь и монитор очень длинными кабелями (см. **рис. 8**).

Можно настраивать BIOS Setup или рассматривать Windows, свалившуюся в синий экран, но ни удаленных дисководов, ни даже возможности нажать на Reset у нас нет, то есть иллюзия полного физического доступа оказывается не такой уж и полной. Зато поддерживаются практически все видеорежимы и в код BIOS не вносится никаких изменений, а в критических инфраструктурах это очень актуально. Внедрять посторонний эмулятор в банковский компьютер нам попросту не дадут, поскольку эта технология не сертифицирована, а вот у KVM-коммутаторов все необходимые сертификаты, как правило, имеются (см. **рис. 9**).

Подавляющее большинство моделей рассчитано на управление несколькими серверами с одного терминала, при этом сигнал пускается по экранированной витой паре с максимальной длиной в несколько сотен метров. Это совсем не Ethernet и в сетевой концентратор его вставлять нельзя! Для реального удаленного управления по Интернету или модему нам потребуется установить дополнительный компьютер, принимающий KVM-сигнал и с помощью специального программного обеспечения ретранслирующий его в «удобоваримую» сетевую форму. А это нехорошо! К счастью, некоторые модели поддерживают работу по модему или локальной сети. Такой тип KVM-коммутаторов называется «over IP», хотя здесь не обходится без вариаций. Просто загляните в спецификацию: если там встретится что-то похожее на LAN или Dial-Up, это то, что нам нужно (**рис. 10**)!

Довольно хорошо зарекомендовала себя фирма Minicom, в ассортименте которой можно обнаружить по меньшей мере две подходящие модели – Phantom Dial-Up Remote Access и Smart IP Extender Switch Over IP. Первая стоит в районе $800, вторая… – $3500. Для банков и прочих денежных учреждений такая сумма, может быть, и подойдет, но вот для мелкой конторы – навряд ли. Кончено, порывшись в магазинах, можно найти KVM-коммутатор и подешевле, но лучше собрать систему удаленного управления самостоятельно.

Как это работает, или Удаленный контроль своими руками!

Для создания собственной системы удаленного управления нам понадобится любая PCI-карта и материнская плата, поддерживающая работу с PCI-шиной через BOOT-block (например, ASUS). На борту карты обязательно должна присутствовать «кроватка» с BIOS. На худой конец BIOS может находиться в отдельной микросхеме, которую несложно выпаять с платы и воткнуть в программатор. К сожалению, сетевые карты с «внешним» BIOS выходят из употребления и найти их становится все сложнее и сложнее. Современные Ethernet-контроллеры интегрируют BIOS в микросхему чиспета, и мы уже не можем ничего с ним сделать (только не перепутайте BIOS с панельной для Boot-ROM, это совсем не одно и то же!).

Вот и приходится пересаживаться на SCSI-контроллеры, цены на которые упали до 10-14. Разумеется, речь идет о простейших моделях, но

Рисунок 7. Плата удаленного управления типа eRIC enhanced Remote Management Card

Рисунок 8. KVM-коммутатор за работой

ведь нам ничего, кроме BIOS, не нужно! Поэтому, даже дешевая модель будет работать ничуть не хуже дорогой. Заботиться о сохранении работоспособности контроллера не обязательно. Намного проще переписать BIOS с чистого листа, чем добавлять свои собственные модули в уже существующий (однако при желании это можно сделать) (см. **рис. 11**).

Дополнительный UART-контроллер приобретать не нужно. Лучше воспользоваться тем, что встроен в материнскую плату, а при желании можно задействовать еще и интегрированный Ethernet или любое другое средство коммуникации.

Разработка прошивок обычно ведется на Ассемблере, но при желании можно использовать и высокоуровневые языки типа Си/Си++. Только ни в коем случае не используйте стандартные библиотеки ввода/вывода и прикажите линкеру отключить Start-Up. Для этого достаточно переименовать функцию main в нечто вроде MyMain. Поскольку Си не поддерживает базирования, откомпилированный код должен быть полностью перемещаем (то есть выполняться независимо от базового адреса загрузки в память). Этого можно добиться, отказавшись от глобальных переменных и выключив все опции компилятора, которые могут генерировать неперемещаемый код, о котором мы даже не подозреваем (например, контроль «срыва» стека). Если вы не уверены, что хорошо знаете «задний двор» компилятора, – не используйте его! Программируйте на Ассемблере. Он не подведет!

Код прошивки исполняется в 16-разрядном сегменте реального режима, однако никто не запрещает нам переходить в защищенный режим и выходить оттуда, правда, не совсем понятно, зачем это нужно. Использовать служебные функции BIOS недопустимо, поскольку часть аппаратуры еще не инициирована, да и сам BIOS еще не распакован. Работайте только через порты ввода/вывода, однако перед этим не забудьте, что оборудование должно быть инициализировано вручную. В частности, интегрированный COM-порт еще не имеет ни базового адреса, ни IRQ, ведь PnP-менеджер, распределяющий системные ресурсы, еще не получил управления! Приходится открывать документацию на южный мост чипсета и программировать все железо с нуля. Это самый низкий уровень «общения» с аппаратурой! Необычайно сложный, но в то же время захватывающе интересный! К счастью, серверный мост уже частично инициализирован, поэтому настраивать контроллер памяти не обязательно.

Теперь поговорим о методиках эмуляции и перехвата. Для вывода информации на экран BIOS использует свою собственную сервисную службу INT 10h. Она же используется на стадии первичной загрузки операционных систем семейства Windows и UNIX. Перехватив это прерывание, мы сможем

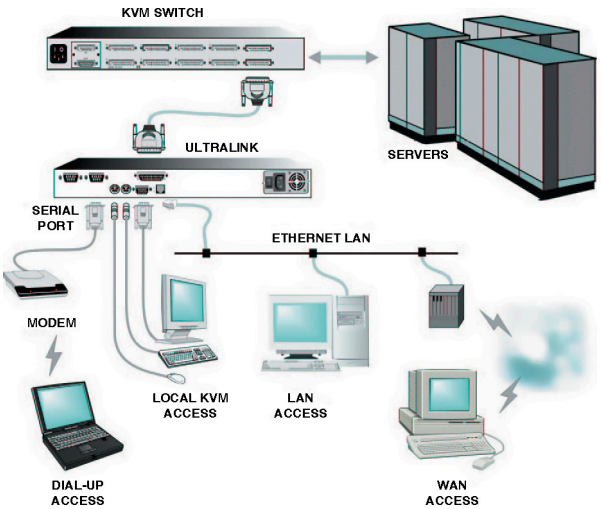

Рисунок 9. Схема подключения KVM-коммутатора для удаленного управления через Интернет или по модемному соединению

Рисунок 10. Внешний вид некоторых KVM-коммутаторов

Рисунок 11. SCSI-контроллер с несъемным BIOS (слева), он нам не подходит. Справа SCSI-контроллер со съемным BIOS, который легким движением руки превращается в плату удаленного управления

грабить весь вывод на экран и передавать его на удаленный компьютер («грабить» – вполне легальный термин, позаимствованный у англоязычных инженеров, которые говорят в этом случае «grab», звучит грубовато, зато почестному).

Разумеется, без сложностей здесь не обходится. Поскольку в процессе инициализации BIOS вектора прерывания могут переустанавливаться многократно, одной лишь модификации таблицы прерываний (т.е. классического способа перехвата) будет явно недостаточно. Да, мы можем изменить far-указатель по адресу: 0000h:10h*sizeof(DWORD) == 0000h:0040h,перенаправив его на свой собственный обработчик, но… через некоторое время контроль за INT 10h будет утерян. Чтобы этого избежать, необходимо установить аппаратную точку останова на запись этой ячейки памяти. В этом нам помогут отладочные регистры семейства DRx. Регистры Dr0-Dr3 хранят линейный физический адрес точки останова, а Dr7 определяет условия, при которых она срабатывает, заставляя процессор генерировать прерывание INT 01h, на котором должен находиться наш обработчик, выполняющий повторную «экспроприацию» INT 10h у системы.

Пример работы с отладочными регистрами приведен ниже.

Листинг 1. Перехватчик передает управление нашему коду в момент загрузки Boot-сектора

```
; перехватываем INT 01h
MOV ax, CS
XOR bx,bx
MOV DS,bx
; смещение нашего обработчика
MOV [bx], offset our_vx_code
; относительно сегмента 0000h
MOV [bx+2],bx
MOV DS, ax

; устанавливаем точку останова
; на исполнение
MOV eax,302h
; линейный физический адрес
; точки останова
MOV ebx,7C00h

; Заносим значения в отладочные
; регистры
MOV dr7,eax
mov dr0,ebx
```

Прерывание INT 10h поддерживает свыше сотни различных функций, номер которых передается в регистре AH. В частности, 02h управляет курсором, а 09h печатает символ. Естественно, чтобы грабить вывод на экран, необходимо уметь отличать одну функцию от другой и знать, чем именно каждая из них занимается. Описание функций можно найти либо в технической документации на конкретную видеокарту (а если карта встроена в материнскую плату, то в документации на серверный мост чипсета), либо в знаменитом Interrupt List Ральфа Брауна, правда, он уже давно не обновлялся и сильно устарел. Последняя версия датируется летом 2000 года. С тех пор вышло множество новых карт! Впрочем, базовые видео-функции не претерпели никаких изменений, и если отбросить нестандартные видеорежимы, все будет работать на ура.

Текстовые режимы грабятся просто замечательно, а вот графические в пропускную способность аналоговых модемов уже не вмещаются, и передаваемую информацию приходится както сжимать. Самое простое – передавать только изменения, предварительно упаковав их по gzip-алгоритму, для работы с которым существует множество готовых библиотек.

Правда, с переходом операционной системы в защищенный режим, весь наш перехват будет «подавлен», и удаленный компьютер отобразит унылый застывший экран. В принципе с этим можно и смириться. Главное, что нам подконтролен BIOS Setup и начальная стадия загрузки оси, а там можно и стандартным telnet воспользоваться, если, конечно, на середине загрузки Windows не выбросит синий экран.

В своих первых моделях систем удаленного управления я поступал так: отслеживал попытку перехода в защищенный режим (а отследить ее можно с помощью все тех же отладочных регистров), переходил в защищенный режим сам, устанавливал свои обработчики прерывания и отдавал управление операционной системе, не позволяя ей ничего менять. Это работало! Хотя и сбоило тоже. Универсального перехватчика создать не получилось, и пришлось учитывать особенности реализации всех операционных систем. В конце концов я махнул рукой и написал обыкновенный драйвер-фильтр, работающий, как VGA-miniport, и пересылающий экранный вывод на «нашу» карту расширения (**рис. 12**).

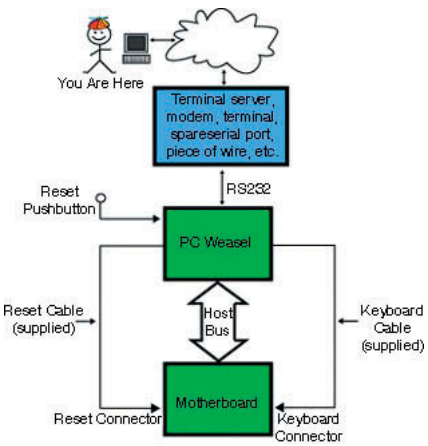

Рисунок 12. Принцип работы платы удаленного управления

Некоторые системы удаленного контроля (например, уже упомянутый комплекс PC Weasel 2000) вместо перехвата INT 10h просто грабят видеобуфер, что на первый взгляд существенно упрощает реализацию. Не нужно возиться с отладочными регистрами, рыться в Interrupt List и т. д. На самом деле даже в текстовом режиме имеется множество экранных страниц, а уж про графический мы вообще молчим! Причем совершенно неясно, как синхронизовать экранный вывод с его перехватом. Сканировать видеопамять с частой 50-60 Гц вполне реально, но вот запихать награбленные данные в модемный канал получится едва ли. А как это дело будет тормозить! Неудивительно, что PC Weasel 2000 работает только с текстовыми режимами!

Теперь перейдем к эмуляции ввода с клавиатуры. Мышь рассматривать не будем, поскольку нормальные администраторы свободно обходятся и без нее. Весь клавиатурный сервис сосредоточен в прерывании INT 16h, которое мы должны перехватить. Когда программа (и в частности, BIOS Setup) ожидает нажатия на клавишу, она обнуляет регистр AH и вызывает INT 16h. Конечно, существуют и другие варианты, но этот – самый популярный. В этом случае наш обработчик прерывания должен поместить ASCII-код символа, нажатого на удаленной клавиатуре, в регистр AL и возвратить управление. Естественно, все это будет работать только до перехода операционной системы в защищенный режим, а после – придется подгружать свой драйвер, «садящийся» поверх стандартного клавиатурного драйвера и эмулирующего ввод.

Удаленные диски реализуются совсем тривиально. За это отвечает прерывание INT 13h. Функция 02h обеспечивает чтение сектора, 03h – его запись. Номер сектора передается в регистрах CX и DX в CHS-формате. Удаленный CD-ROM реализуется чуть-чуть сложнее. Если вы не сильны в системном программировании, на первых порах лучше ограничиться виртуальными дискетами. Между прочим, использовать физические дискеты совсем не обязательно – удаленная машина может работать с их образом, записанным на жестком диске в виде файла. Для удаленной переустановки Windows NT этот прием вполне подходит. А смену виртуальных дискет автоматизировать совсем нетрудно.

В результате мы получим довольно могучий комплекс удаленного управления, и самое главное – очень дешевый. Конечно, наше время тоже что-то стоит (а времени на разработку и пуско-наладку уйдет много), но если такие комплексы изготавливать под заказ, они быстро себя окупят, тем более что на них наблюдается устойчивый спрос, ведь западные аналоги большинству просто не по карману.

Для завершения картины остается сущая мелочь – удаленный Reset, без которого наше творение будет неполноценно. Ну тут все просто. Достаточно подключить к LPT-порту реле, ведущее к «заветной» кнопке, и проблема будет решена. Из прошивки SCSI-контроллера мы можем управлять LPT-портом, конечно, не забыв, что перед этим его нужно инициализировать.

Один маленький трюк напоследок. Если полноценная система удаленного управления вам не нужна и всего лишь требуется запретить BIOS требовать нажатия на клавишу при загрузке, то без дополнительного оборудования легко обойтись. Достаточно загрузить прошивку основного BIOS в дизассемблер и найти все «ругательные» сообщения. Перекрестные ссылки приведут нас к машинному коду, который эти строки и выводит. Там же будет код, ожидающий нажатия на клавишу, который мы должны удалить. Прямой вызов INT 16h используется редко.

Скорее всего, мы увидим что-то вроде CALL xxx, где xxx – адрес функции-обертки. Для достижения задуманного мы должны заменить CALL xxx на «MOV AX,scan-code», указав скэн-код требуемой клавиши. Например, клавиша <F2> в большинстве BIOS означает «загрузку с настройками по умолчанию», однако в некоторых случаях может потребоваться нажать <Enter> или <Esc>.

Проблема в том, что основной образ BIOS упакован и защищен контрольными суммами. Практически все разработчики BIOS распространяют утилиты для распаковки/упаковки и пересчета контрольных сумм, однако никакой гарантии, что модифицированный BIOS будет исправно работать, у нас нет. Ошибки могут появляться в самых неожиданных местах. Работа системы становится нестабильной, материнская плата без всяких видимых причин начинает зависать и т. д. Разумеется, для серверов это неприемлемо, поэтому приходится идти другим путем.

Вместо того чтобы модифицировать упакованный образ основного кода BIOS, мы возьмем неупакованный BOOT-block и добавим в него автоматический патчер, правящий нужные байты прямо в памяти, когда распаковка уже завершена. Поскольку основной код BIOS распаковывается в RAM, никаких проблем с его исправлением не возникает. Главное – определить нужные адреса. В этом нам поможет тот факт, что сам BIOS свой образ не затирает и в момент загрузки boot-сектора он присутствует в памяти. Достаточно написать крошечную ассемблерную программу, считывающую первые 640 Кб нижней памяти и записывающую их на гибкий диск, а затем внедрить ее в boot-сектор. После перезагрузки системы мы станем обладателями распакованного BIOS, лежащего по своим «родным» адресам.

Остается только прожечь обновленный BOOT-block и можно наслаждаться бесперебойной работой сервера!

Заключение

Полноценный удаленный контроль за системой – это реальность! Ассортимент возможных решений необычайно широк: от готовых (и весьма дорогостоящих!) KVM-устройств до более дешевых, но вместе с тем и более функциональных (!) плат расширения, которые большинство программистов легко изготовят самостоятельно. Физический доступ к серверу будет требоваться только при его ремонте (здесь без него никак не обойтись, ведь плоскогубцы с отверткой по модему не передашь), однако фатальные отказы происходят не так уж и часто.

Ссылки:

1. Remote Insight «Lights Out» boards – обзор систем удаленного управления (на англ.): http://www.paul.sladen.org/lights-out/riloe.html;
2. Remote Insight Lights-Out Edition II – описание платы удаленного управления от Hewlett-Packard с возможностью заказа по Интернету (на англ.): http://h18004.www1.hp.com/products/servers/management/riloe2/server-slot-matrix.html;
3. PC Weasel 2000 – описание альтернативной платы удаленного управления, микрокод, который распространяется по открытой лицензии (на англ.): http://www.realweasel.com/intro.html;
4. Технические характеристики огромного количества систем удаленного управления (преимущественно KVM-коммутаторов, на англ.): http://www.kvms.com;
5. Raritan IP-Reach TR364 – описание KVM-коммутатора TR364 (на англ.): http://www.42u.com/tele-reach_bk.htm;
6. Архитектура ввода-вывода персональных ЭВМ IBM PC – электронная версия книги, посвященной устройству IBM PC, которую настоятельно рекомендуется прочитать перед разработкой собственной системы удаленного управления (на русском языке): http://redlib.narod.ru/asmdocs/asm_doc_07.zip;
7. Ralf Brown Interrupt List – электронный справочник по всем прерываниям, портам ввода/вывода, «волшебным» адресам памяти, включая нестандартные расширения и недокументированные возможности (на англ.): http://www.ctyme.com/rbrown.htm.