копирование без границ
или передовые методики защиты CD
или CD неподвластный копированию

крис каперски ака мыщъх

копировщики лазерных дисков совершенствуется с каждым днем, но и разработчики защитных механизмов не дремлют, тем не менее, новые защиты тут же ломаются. почему? вашему вниманию предлагается обзор наиболее популярных ошибок и конструктивных просчетов с рекомендациями по их устранению, а так же описывается законченный алгоритм чрезвычайно стойкой защиты, не копируемой никаким копировщиком.

введение

Хакерская мудрость гласит: "взломать можно все, это только вопрос времени". Валовый программный продукт, ориентированного на массовый рынок, такая ситуация вполне устраивает. Какой-то процент пользователей покупает программу, какой-то нет. Но для специализированных программных комплексов (расчет прецизионного литья или звездных спектров), количество пользователей которых измеряется какими-то тысячами, такой подход уже неприемлем. Потенциальный рынок настолько мал, что каждая нелегальная копия чувствуется весьма болезненно. Значит, надо защищать так, чтобы не взломали, но как?

У нас две новости — хорошая и не очень. Надежные защиты все-таки существуют, но их разработка требует больших усилий и знания реалий. Это довольно заковыристая предметная область и с одной лишь теоретической подготовкой в нее не войдешь. Тем не менее, спрос рождает предложение и автор статьи делиться своим многолетним опытом по созданию стойких защитных комплексов.

основные концепции защиты

Главное свойство защиты — это надежность (не путать со стойкостью ко взлому). Защита должна уверенно работать на всем спектре программно-аппаратного обеспечения с заранее непредсказуемыми свойствами. Антиотладочные приемы следует использовать с большой осторожностью или не использовать вообще, поскольку от них слишком много проблем. К тому же защита должна быть проста в реализации и отладке. Например, если часть функционала вынесена в микроконтроллер, смонтированный на отдельной печатной плате, подключаемой через PCI-шину или COM/USB/LPT-порт, взлом становится не только нерентабельным, но и практически невозможным, конечно, при условии, что микроконтроллер не позволяет считывать ПЗУ (некоторые приемы аппаратного взлома можно найти на страничке Сергея Скоробогатого: http://www.cl.cam.ac.uk/~sps32/), однако, отладить такой комплекс будет намного сложнее, чем взломать, поэтому для практического применения он непригоден. В идеале, защита вообще не должна требовать никакого "внешнего" оборудования, а это значит, что ее код заведомо будет заведомо доступен для анализа и модификации.

Рисунок 1 электронный ключ…

Рисунок 2 …и его взлом, путем считывая содержимого EEPROM под микроскопом с увеличением 500x

Защита не может отталкиваться ни от серийных номеров, ни от ключевых файлов, поскольку всегда найдется пользователь, пожелавший разделить себя с миром и, в общем-то по своему он будет прав. Черные списки, "засвеченных" серийных номеров и прочие организационные меры, как показывает практика, недостаточно эффективны и со своей задачей не справляются.

Так же недопустимо привязываться к оборудованию, поскольку потребители очень не любят, когда ограничивают их свободу, к тому же, если стоимость защищенного комплекса составляет хотя бы 500$, хакеры могут клонировать весь компьютер целиком, особенно если это

будут китайские хакеры (цены на оборудование в Азии намного ниже европейских). Наиболее популярным объектом привязки служит серийный номер жесткого диска и MAC-адрес сетевой карты, которые легко изменить (для жесткого диска можно воспользоваться комплексом PC-3000 от ACE Laboratory — www.acelab.ru, а для сетевой карты — утилитой ipconfig, запущенной из-под LINIX). Остальные же характеристики "железа" еще менее уникальны и в пределах одной партии практически идентичны друг другу. К тому же, нельзя забывать про виртуальные машины (VMWare, Virtual PC и т. д.), привязка к которым лишена всякого смысла. Можно, конечно, распознать наличие виртуальной машины и отказаться работать под ней, только это не выход. Виртуальных машин существует ~~огромное~~ великое множество и каждая из них детектируется по-своему, к тому-то для большинства из них существуют специальные "патчи", предотвращающие детектирование. В общем, вложенные в защиту усилия, окажутся нерентабельными.

Привязываться можно только к ***носителю*** — дискете, лазерному или DVD-диску, карте FLASH-памяти и т. д. Это в наименьшей степени ущемляет права потребителей и практически не создает непреодолимых неудобств, к тому же, качественно защищенный носитель чрезвычайно трудно скопировать. Существует множество хакерских групп, специализирующихся на взломе программного кода, но очень немногие разбираются в устройстве носителей на профессиональном уровне.

Весь вопрос в том — какой носитель выбрать? Дискеты отбросим сразу. Их время уже прошло. DVD-диски только набирают силу и требовать обязательно наличия DVD привода на целевом компьютере по меньшей мере негуманно. Тоже самое относится и к FLASH-картам. Остается только CD. Вот к нему-то мы и будем привязываться.

>>> врезка: детектирование VMWare

Автору известны по меньшей мере три способа обнаружения VM-Ware. Во-первых, по оборудованию: виртуальные машины несут на своем борту довольно специфический набор железа, практически не встречающийся в живой природе. Это:

- видеокарта VMware Inc [VMware SVGA II] PCI Display Adapter;
- сетевая карта: Advanced Micro Devices [AMD] 79c970 [PCnet 32 LANCE] (rev 10);
- жесткие диски: VMware Virtual IDE Hard Drive и VMware SCSI Controller.

Опросив конфигурацию оборудования, защищаемая программа сразу поймет, куда ее занесло.

Во-вторых, виртуальные сетевые карты имеют довольно предсказуемый диапазон MAC-адресов, а именно: 00-05-69-xx-xx-xx, 00-0C-29-xx-xx-xx и 00-50-56-xx-xx-xx. Защите достаточно выполнить команду "`arp -a`", чтобы распознать хакерские планы.

В-третьих, VM-Ware имеет коварный backdoor, оставленный разработчиками для служебных целей и управляемый через порт 5658h, при этом в регистре EAX должно содержатся "магическое" число 564D5868h. Ниже приведен фрагмент кода червя Agobot, определяющий версию VM-Ware:

```
mov eax, 564D5868h ; VMWARE_MAGIC
mov ecx, 0Ah ; Get VMware version
mov edx, 5658h ; VMWARE_PORT
in eax, dx
```

Листинг 1 определение версии VM Ware

Для маскировки виртуальной машины, Костей Кортчинским (Kostya Kortchinsky) был написан специальный патч, изменяющий идентификационные строки оборудования, MAC-адреса и магический номер backdoor'a (http://honeynet.rstack.org/tools/vmpatch.c).

Подробнее о способах детектирования виртуальных машин можно прочитать в подборке статей **Know your Enemy** (на английском языке): www.honeynet.org/misc/files/papers.tar.gz;

обзор популярных средств защиты

Механизмы привязки к CD-ROM можно разделить на три большие группы. В первую (и наиболее древнюю) попадают защиты, внедряющие ключевую метку в служебные структуры данных, не копируемые штатными копировками. Это может быть и область пред-зазора первого трека (first pre-gap), и субканальные данные, так же называемые данными подканалов

(subchannel data) и т. д. Достоинства — высокая совместимость с различными моделями приводов и предельная простота программной реализации. Недостатки — при подготовке диска к тиражированию придется долго объяснятся с сотрудниками завода. Обычно им передают готовый диск или образ, содержащий только пользовательские данные (он же образ типа ISO9660, хотя это название не вполне корректно), здесь же требуется "сырой" (RAW) образ в формате 2352/96, который поддерживает далеко не всякое оборудование! Зачастую, производитель игнорирует наши требования и вопреки всем обещаниям и договорам с "конвейера" сходят диски, записанные стандартным образом, то есть без ключевых меток, а для заказчика это катастрофа! (Поверьте, моему горькому опыту, это довольно часто встречается). Но это все равно напрасно, поскольку такие защиты уже давно копируются специализированными копировщиками. Исключение составляет ключевая метка в Q-подканале аудио-трека. Без спецоборудования она не копируется в принципе! Как-нибудь мы об этом поговорим.

Вторая группа защитных механизмов основана на нестандартных форматах диска (необычная длина сектора, нестандартные номера треков, искаженные заголовки и т. д.). Все они крайне конфликтны и отказываются идти на многих моделях приводов, к тому же при тиражировании дисков возникают очень серьезные проблемы, намного более серьезные, чем в первом случае. Как правило, требуется специальное (и весьма дорогостоящее!) оборудование, оправдывающее себя только на больших тиражах. Но эти вложения вряд ли окупятся, поскольку копировщики защищенных дисков уже давно справились с нестандартными форматами, так что не будем на них останавливаться и двинемся дальше.

Третью группу возглавляют защиты, привязывающиеся к физической структуре носителя. Их можно разделить на две подгруппы. Первая выделяет на диске некоторые более или менее уникальные характеристики, к которым, собственно, она и привязывается. Вторая же, не собираясь ждать милости от природы, самостоятельно формирует трудновоспроизводимые дефекты на диске. Возьмем известную защиту Laser Lock, которую легко опознать по наличию крошечной "дырки", проделанной лазером точно посередине спиральной дорожки (аналогичный способ был широко известен еще во времена дискет, причем их дырявили не только лазером, но еще и гвоздем). На первый взгляд тут должен образоваться BAD-сектор, однако, практика показывает, что диск читается без проблем. А как же дыра? Все дело в кодах Рида-Соломона, корректирующей способности которых вполне хватает для исправления "дыры". Но если отключить коррекцию ошибок, то в секторе сразу же обнаружатся "дефективные" биты, причем начало разрушенной области будет соответствовать позиции дырки в секторе. Это если объяснять на пальцах.

На самом деле все намного интереснее и сложнее. Структура хранения данных на лазерном диске такова, что физически смежные биты расположены значительном удалении друг от друга, зачастую даже в различных секторах! Информация как бы "размазывается" вдоль спиральной дорожки, чтобы противодействовать царапинам и дефектам. Дело в том, что корректирующие коды Рида-Соломона отлично справляются с одиночными ошибками, но намного хуже с групповыми. Чтобы ослабить влияние дефектов пришлось прибегнуть к перемешиванию. А это значит, что "дыра" затрагивает не один сектор, а целую группу секторов! Это дает возможность распознавать "виртуальные" BAD-сектора, имитируемые копировщиками. Поразительно, но ни одна из коммерческих защит такой проверки не выполняет! Тем не менее, копировщики защищенных дисков уже давно научились обходить Laser-Lock, имитируя даже такие "тонкие" эффекты как увеличение времени чтения "продырявленных" секторов и т. д., так что такой прием подходит только для борьбы со штатными копировщиками. К тому же, очень трудно найти завод, располагающий соответствующим защитным оборудованием. Вообще-то, в отсутствии лазера, можно воспользоваться и обыкновенным маркером, однако, он подходит только для крошечных тиражей.

Рисунок 3 лазерный диск, защищенный Laser Lock ("снимок" получен сканером HP 1200)

Остается последний тип защиты — измерение физических характеристик спиральной дорожки (так же называемый снятием топологии). По такому принципу, в частности, работают CD-Cops, SecureROM 4x, Star-Force и некоторые другие защиты. Методика отработанная, можно даже сказать вылизанная до зеркального блеска, и неплохо себя зарекомендовавшая. Судите сами. В образ защищенного диск не вносится никаких изменений и для его тиражирования можно использовать абсолютно любое оборудование, в том числе и бытовой CD-R/RW рекордер. Скопировать физическую структуру спиральной дорожки нереально (на

CD-R/RW дисках уже нанесена предварительная разметка, причем у каждого типа болванок своя), и хотя ее можно проэмулировать, от эмуляторов легко защититься (чуть позже мы покажем как). По правде говоря, существует возможность подбора болванки с похожей спиральной структурой, однако, если привязываться не к одной, а нескольким физическим характеристикам, вероятность подобрать "правильный" диск будет крайне мала.

В принципе, можно воспользоваться готовым защитный пакетом, но во-первых, за него придется платить. Во-вторых, все вышеперечисленные защиты легко копируются в режиме эмуляции копировщиками Clone CD и Alcohol 120%. Исключение составляет Star-Force Professional Edition, непосредственно скопировать который еще никому не удалось, однако (и это в третьих!), защита слишком агрессивно вгрызается в операционную систему, вызывая множество проблем у легальных пользователей. Разработчики характеризуют себя как людей с хакерским прошлым, сильных в системном программировании. Что касается прошлого — с этим можно согласиться. Операционную систему они знают лучше, чем свой задний двор. Но вот программировать умеют едва ли. Программирование — это в первую очередь проектирование. А проектирование — это учет рисков. Никакой конструктор не позволит себе стоить мост по непроверенным формулам или проводить на нем научные эксперименты, гадая — произойдет обрушение на этот раз или не произойдет. Программа, ориентированная на массовое применение, просто не может пользоваться недокументированными возможностями и прочими приемами нетрадиционного программирования (в народе именуемых хаками). У себя в заднем дворе делайте, что хотите, но вот пользователю требуется нормальный продукт.

А давайте запрограммируем защиту самостоятельно! Ядро измерителя структуры спиральной дорожки занимает всего несколько строк на Си. Вместе с обвязочным кодом выходит около десятка… Полный цикл разработки вместе с отладкой легко укладывается в пару недель. Так стоит ли за это платить?

star-force своими руками

Спиральная дорожка лазерных дисков очень похожа на грампластинку, только начинается не снаружи, а изнутри, наматываясь от центра к краю. Оптическая головка, удерживаемая в магнитном поле "звуковой катушки", движется на "салазках" поперек спиральной дорожки. Сама дорожка состоит из секторов с данными и каналов подкода. Номера секторов находятся как в заголовках самих секторов, так и в каналах подкода, "размазанных" вдоль спиральной дорожки. Для грубой наводки не требуемый сектор используются салазки и каналы подкода, а для точной — отклонение в магнитном поле и секторные заголовки.

Просто взять и измерять структуру спиральной дорожки нельзя, но можно сделать вот что: допустим, головка считывает сектор X, а следом за ним сектор Y. Если угол XOY, образованный центром (O) диска, секторами X и Y составляет порядка ~15 град, а сами сектора расположены на соседних витках спирали, то приводу достаточно всего лишь немного отклонить головку и через мгновение сектор Y сам падает в руки, как перезревшее яблоко — диск ведь вращается! Если же угол XOY составляет менее ~15 град, тогда за время перемещения головки, сектор Y уже "уплывет" и приводу придется ждать целый оборот лазерного диска, пока он не достигает оптической головки!

Рисунок 4 когда угол между секторами X и Y составляет ~15 град. при переходе на соседний виток, сектор Y сразу же "подлетает" к оптической головке (рисунок слева), при меньшем значении угла сектор Y успевает уплыть и головка вынуждена ждать целый виток

Замеряя время чтения различных пар секторов, мы можем приблизительно определить их взаимное расположение на спиральной дорожке. У каждой партии диска для заданных секторов X и Y оно будет своим (ведь степень "закрутки" спирали неодинакова и варьируется от одного производителя к другому). Чтобы побороть упреждающее считывание (которым "страдают" многие приводы), защита должна читать сектора в порядке убывания их LBA-адресов. Так же она должна измерять скорость вращения привода, чтобы во-первых, определить постоянство временных замеров (пляшут ли они как пьяные человечки или нет), а во-вторых скорректировать формулу для вычисления угла, ведь как легко показать, чем быстрее вращается диск, тем скорее "уплывает" сектор.

Исходный текст "измеряющей" программы приведен ниже:

```
//-[чтение сектора с диска]-------------------------------------------------------
// ARG:
//      CD      указатель на строку с именем провода (например, "TEAC"),
//              адрес на ASPI-шине (например, "1.1") или имя диска("\\.\G:");
//              первые два варианта работают через ASPI, последний через SPTI;
//
//      buf     указатель на буфер SECROR_SIZE*2
//
//      sector  номер сектора в LBA-формате
//
// RETURN:
//       0      успешно
//      -1      ошибка
read_from_cd(char *CD, unsigned char *buf, long sector)
{
        int stat;
        stat=cd_raw_sector_read(CD, buf, SECTOR_SIZE, sector, ONE_SECTOR, W_USER_DATA);
        if (stat == SCSI_OK) return 0; return -1;
}

//-[чтение TSC-счетчика]----------------------------------------------------------
unsigned int A()
{
        __asm{
        _emit   0xF     ; RDTSC
        _emit   0x31
        }
}

#define argCD          v[1]

// КОНФИГУРАЦИЯ
//-----------------------------------------------------------------------------

// номер первой точки измерения (LBA-адрес)
// данная утилита измеряет топологию только по одной точке,
// что не есть хорошо, т.к. легко подобрать похожий диск
// для уверенности следует выбрать несколько точек:
// в начале, середине и конце диска
#define _CFG_BGN_SEC_  17699

// кол-во секторов для измерения
// должно быть не меньше утроенного кол-ва секторов на виток в данной
// точке измерения (см. _CFG_BGN_SEC_ )
// число витков спирали N с поперечной плотностью D витков/мм
// от радиуса R1 до радиуса R2 определяется формулой: N = (R2- R1) * D
#define _CFG_LEN_SEC_  0x669

// максимальный шаг приращения
// в принципе должен быть равен удвоенному кол-ву секторов
// на данном витке спирали, что увеличивает точность измерений
// но можно использовать и  значение _CFG_LEN_SEC_
#define _CFG_LEN_DEL_  _CFG_LEN_SEC_

// начальный шаг приращения (должен быть по возможности мал)
#define _CFG_BGN_DEL_  0x2

// приблизительное кол-во секторов на данном витке спирали
// (в данной версии программы это значение мало на что влияет)
#define _CFG_xWHELL_   27

// конечный сектор для проверки
#define _END_SEC_ (_CFG_BGN_SEC_+_CFG_LEN_SEC_)

// конечный шаг
#define _END_DEL_ (_CFG_BGN_DEL_+_CFG_LEN_DEL_)

#define FB(b) (##b = (##b + 1) % _END_DEL_);         // приращение шага

// шапка цикла
#define FH(a,b) for (##a=_END_SEC_,##b=_CFG_BGN_DEL_; ##a > _CFG_BGN_SEC_; ##a-=##b)

main(int c, char** v)
{
        int a, b; int x=0; int i=0; int A1, A2;
        unsigned char buf[SECTOR_SIZE];
```

```
        // проверка аргументов командной строки
        if (c < 2) {
                fprintf(stderr,"USAGE:sf.exe CD\n\n");
                printf( "       SCSI_INQUITY via ASPI32\n"\
                        "-----------------------------------\n");
                        read_from_cd("?.?", buf,0); return 0;
        }

        // этап первый
        //-------------------------------------------------------------------------
        // читаем случайные сектора для разгона привода
        fprintf(stderr,"%s\n",_TEXT_SPINEUP_);
        for (a = 0; a < 0x69; a++)
        {
                read_from_cd(argCD, buf,rand()%_END_SEC_);
                fprintf(stderr,"\r%02d%%",a*100/0x69);
        }

        // этап второй
        //-------------------------------------------------------------------------
        // определяем кол-во секторов на дорожке и стабильность вращения привода
        // алгоритм определения кол-ва секторов: читаем сектора задом наперед,
        // с циклически увеличивающимся шагом, наименьшее значение шага
        // при котором время чтения секторов будет минимальным - и будет равно
        // кол-ву секторов на данном витке спирали
        // (fixit: пока не реализовано, кол-во секторов взято на глазок)
        //
        // алгоритм определения стабильности: читаем сектора с шагом, равным
        // кол-ву секторов на данном витке спирали, и оцениваем разброс;
        // если разброс будет слишком большим (превышает 10%-15%), следует
        // уменьшить скорость привода (как это сделать показано в CD.snail.c)
        fprintf(stderr,"\r%s\n",_TEXT_TEST_);
        for (a = _END_SEC_; a > _CFG_BGN_SEC_; a-=_CFG_xWHELL_)
        {
                A1=A(&c);read_from_cd(argCD, buf,a);A2=A(&c);
                fprintf(stderr,"\r%02d%%",(_END_SEC_-a)*100/_CFG_LEN_SEC_);
        }

        // этап третий (важнейший!)
        //-------------------------------------------------------------------------
        // производим, собственно, измерения
        // fix1: добавить "сглаживание" полученных данных
        fprintf(stderr,"\r%s\n",_TEXT_ANGLE_);

        //  выводим шапку таблицы
        printf("delta:"); FH(a,b) { printf("\t%d",a); FB(b); } printf("\ntime:");

        // измеряем и тут же выводим результаты
        FH(a,b)
        {
                A1=A(&c); read_from_cd("TEAC", buf,a); A2=A(&c);
                printf("\t%d",(A2-A1)/100);
                fprintf(stderr,"\r%02d%%",(_END_SEC_-a)*100/_CFG_LEN_SEC_);
                FB(b);
        } printf("\n");

        // всему конец
        fprintf(stderr,"\r%s\n",_TEXT_END_); return 0;
}
```

Листинг 2 макет программы sf.c для снятия топологий

Программа использует библиотечку SCSIlib, разработанную автором для низкоуровневого управления приводами с прикладного уровня. Ее можно бесплатно скачать с ftp-сайта автора.

запуск программы

При запуске без аргументов, программа выдаст краткую справку по ключам, а при наличии ASPI-дравйера автоматически просканирует системную шину и выведен адреса и названия всех обнаруженных приводов.

Это может выглядеть, например, так:

```
>sf.exe
```

```
USAGE:sf.exe CD

       SCSI_INQUITY via ASPI32
-----------------------------------
0.0 <-- ELBY    DVD-ROM         1.0  (5)
1.0 <-- ST380011A               3.06 (0)
2.0 <-- IBM-DTLA-307015         TX2O (0)
2.1 <-- TEAC    CD-W552E        1.09 (5)
3.0 <-- AXV     CD/DVD-ROM      2.2a (5)
3.1 <-- AXV     CD/DVD-ROM      2.2a (5)
3.2 <-- AXV     CD/DVD-ROM      2.2a (5)
```

Листинг 3 результат запуска программы без ключей

Ключ "CB", отвечающий за выбор привода, задает не только сам привод, но и интерфейс взаимодействия.

Если имя привода выглядит как название устройства (т. е. начинается с префикса "\\.\"), то управление будет осуществляться через интерфейс SPTI. Для этого вы должны иметь Windows NT/2000/XP и права администратора.

Если имя выглядит как адрес устройства на шине или как часть идентификационной строки привода, то управление будет осуществляться через интерфейс ASPI, для работы через которой необходимо установить ASPI-драйвер (его можно бесплатно скачать с сервера копании Adaptec — www.adaptec.com).

Например, "`sf.exe TEAC`" (или "`sf.exe 2.1`") , заставляет программу работать с приводом TEAC (адрес на ASPI-шине — 2.1) через ASPI-интерфейс, а "`sf.exe \\.\G:`" — с приводом "G:" через SPTI-интерфейс.

Программа выводит данные в форме таблицы, предназначенной для импорта в MSGraph, причем, вывод оптимизирован для перенаправления в файл (на экране все выглядит кошмарно), поэтому правильный вызов выглядит примерно так: "`sf.exe \\.\G: > C:\1.txt`".

По окончании работы программы запускаем MS Word, открываем меню "Вставка", там будет "Рисунок" и "Диаграмма". В меню "Правка" находим "Импорт", "тип файлов" – "*.txt", "формат данных" — "с разделителем", "начать импорт" со строки 1, "далее >>", "Символом разделителя является": "[x] символ табуляции", "далее >>", "формат данных" — "общий" и жмем кнопку "готово". Далее действуем по своему вкусу, то есть по обстановке.

сбор топологий испытания защиты и обсуждение результатов

Вставляем подопытный диск в привод (пусть это будет, например, диск, прилагаемый к журналу "Компьютер пресс" 2005/07, условно обозначенный нами как диск "A") и снимаем с него топологию (см. рис. 5). Мы получаем характерную "пилу", точки минимума и максимума которой указывают на то, что данная пара секторов лежит на одной прямой.

Рисунок 5 топология диска "A"

Очевидно, что на других дисках те же самые сектора будут иметь совсем другой угол, поэтому минимумы и максимумы сместятся на некоторое расстояние. Берем диск Microsoft Visual Basic 2005 (обозначенный нами как "B"), прилагаемый к тому же самому журналу и запускаем программу еще раз (см. рис. 6).

Пилообразная кривая действительно сместилась приблизительно на половину периода. Так же слегка изменилась и амплитуда колебаний, но нас она не интересует. Будем отталкивается не от абсолютных, а от относительных значений, т. е. от LBA-адресов "изломов" кривой (абсолютные значения находятся в прямой зависимости от "окружающей среды" и потому ненадежны). Учитывая, что положения максимумов/минимумов зависят не только от топологии диска, но еще и от скорости вращения привода, необходимо расширить доверительный интервал до нескольких секторов.

Рисунок 6 топология диска "B"

А теперь возьмем диск с другого номера "Компьютер Пресс" (диск "C") и сравним его топологию с диском "A" (см. рис. 07).

7 сравнение топологий дисков "A" и "C"

Топологии обоих дисков полностью совпадают! Это означает, что диски "Компьютер Пресс" штамповались на одном заводе, а диск с Visual Basic на другом! Любопытное, наблюдение, не правда ли? С помощью этой программы мы можем не только защищаться от копирования, но и проводить интересные исследования (кстати, эта программа была специально написана мной по заказу американского полицейского управления, занимающегося борьбой с пиратством. Обычная история — продавец заказывает партию лицензионных дисков и затем перемешивает их с "пираткой". Как установить факт обмана? Вот тут-то структура спиральной дорожки и выручает!).

Демонстрационная программа снимает топологию только в одной точке (на участке между адресами 27532 и 18082. Разумеется, это ненадежно и при желании можно подобрать диск с похожей топологией (для этого достаточно взять приблизительно 10 болванок от разных производителей). Для усиления защиты настоятельно рекомендуется снимать топологию по меньшей мере в трех точках — начале, конце и середине диска. В этом случае, найти похожий диск будет намного труднее.

>>> врезка полезный совет

Перед снятием топологии скорость привода рекомендуется уменьшить хотя бы до 16x-24x. Как это сделать показано в утилите CD.snail.c, исходный текст которой можно бесплатно скачать с ftp автора.

защита от анализа

Существует по меньшей мере два способа взлома: копирование диска специализированными копировщиками или анализ защитного кода с последующий модификацией (он же bit hack). Создать некопируемый диск это только полдела. Еще необходимо защитить свою программу от анализа.

Обычно анализу противостоят шифровкой кода/данных, однако, это не слишком-то удачное решение, ведь перед выполнением программы ее все равно приходится расшифровывать. Хакеру остается всего лишь дождаться этого момента и снять дамп. Можно, конечно, расшифровывать программу по частям или использовать несколько независимых расшифровщиков, но трудоемкость разработки защиты в этом случае практически не отстает от сложности взлома.

В последнее время большое распространение получил некрасивый, но убойный подход, основанный на генерации "мусорного кода", внедряемого в защищаемую программу. Похожая техника используется во многих полиморфных вирусах, из которых можно "выдрать" уже готовые "движки" (engine). Мусорный код чрезвычайно затрудняет анализ, делая его практически невозможным. Допустим, ключевая функция программы компилируется в тысячу машинных команд. Исходя из "крейсерской" скорости дизассемблирования 1 команда в секунду, хакер сможет "прочесть" (только прочесть! не проанализировать!) весь код за ~15 минут, но после разбавления функции миллионом мусорных инструкций, чтение листинга потребует свыше 10 суток напряженной работы, а полный анализ растянется на долгие годы.

Звучит прекрасно, но без подводных камней не обходится. Во-первых, если перестараться, то скорость программы упадет в разы, причем, нужно учитывать, что далеко не у всех стоит Pentium-4, поэтому "замусоривать" можно только редко вызываемые функции. Во-вторых, мусорный код должен быть достаточно нетривиальным, чтобы его "мусорность" не бросалась в глаза. Использование XCHG EBX,EBX или MOV EAX,EAX легко отсекается анализаторами. В-третьих, в некоторых случаях анализ защитного алгоритма необязателен и хакер может взломать программу и так (см. "защита от мониторов шины"), поэтому, линия обороны должна быть хорошо продуманной и однородной на всем своем протяжении. Бронебойные ворота бесполезны, если вокруг них стен.

Еще лучше применять P-код, реализовав "виртуальную машину" и написав свой собственный интерпретатор. Поскольку, программировать в P-коде очень непроизводительно и неудобно, рекомендуется реализовать транслятор, "пережевывающий" исходный текст, написанный на Паскале или Си и выдающий последовательность инструкций Машины Тьюринга, Сетей Петри, Стрелки Пирса и т. д. Чем ниже уровень абстракции, тем лучше для нас и хуже для хакера. Программа, состоящая из нескольких сотен строк, превращается в десятки тысяч или даже миллионы инструкций Тьюринга, декомпиляторов с которой не существует!

Как компромиссный вариант можно использовать Форт, транслирующий исходный текст в шитый код. Это легко (и с комфортом) программируется, быстро работает, но долго ломается. Тем не менее, трудоемкость взлома порядка на два ниже, чем у Машины Тьюринга или Стрелки Пирса.

>>> врезка ссылки по теме запутывания кода

- **Анализ запутывающих преобразований программ:**
  - теоретическая статья, рассматривающая основные методы "замусоривания" кода и проблемы его "прополки" (на русском языке): http://www.citforum.ru/security/articles/analysis/;
- **A Taxonomy of Obfuscating Transformations:**
  - еще одна статья, посвященная "замусориванию" кода с большим количеством примеров (на английском языке): www.cs.arizona.edu/~collberg/Research/Publications/CollbergThomborsonLow97a/
- **Virtual Machine Design and Implementation in C/C++ by Bill Blunden:**
  - реализация виртуальных машин на Си/Си++ — отличная бумажная книга на английском языке, электронная версия по-видимому недоступна;

защита от эмуляторов

Структуру спиральной дорожки невозможно скопировать, но легко проэмулировать. Вместе с копировщиками защищенных дисков, как правило, поставляются программы-эмуляторы, создающие виртуальный CD-ROM/DVD привод. Копировщик проделывает ту же самую операцию, что и защита – снимает топологию с защищенного диска и передает ее эмулятору.

Можно ли это противостоять? Разработчики Star-Force использовали прямой доступ к IDE-контроллеру в обход всех установленных драйверов, в том числе и драйвера-эмулятора. А как быть, если у пользователя установлен не IDE-привод? Тогда Star-Force вынуждена работать через уже установленные драйвера. Чтобы обойти защиту достаточно выдернуть шлейф со своего привода или отключить соответствующий канал IDE-контроллера на "лету" и Star-Force падет. Не очень-то сильная защита, да к тому же трудно реализуемая.

Рисунок 8 виртуальный диск Virtual Clone CD в "Моем Компьютере"

Мы же пойдет другим путем. Если открыть стандарт по SCSI или ATAPI устройства (их черновые версии лежат на www.t10.org и www.t13.org соответственно), можно обнаружить десятки "мультимедийных" команд, поддерживаемыми практически всеми современными приводами. В эмуляторах же реализована лишь малая часть. Поэтому, чтобы защититься от эмуляторов, достаточно задействовать хотя бы половину SCSI/ATAPI-команд (попутно это "убьет" мониторы шины и прочие анализаторы).

Даже если последующие версии эмуляторов поумнеют и будут имитировать весь "лексикон" привода целиком, они навряд ли смогут воспроизвести все особенности каждой команды и их комбинаций. Впрочем, не будем забегать вперед. Эмуляторы совершенствуются медленно и качественных рывков с их стороны пока не ожидается.

защита от мониторов шины

Для взлома защиты не всегда требуется дизассемблировать код. Вместо этого можно запустить монитор шины для перехвата обмена программы с приводом. Анализируя последовательность вызовов SCSI/ATAPI команд, вполне реально разгадать алгоритм защиты, определив к каким именно характеристикам диска она привязалась.

Большой популярностью пользуется монитор Bus Hound, ознакомительную версию которого можно бесплатно скачать с сайта http://www.perisoft.net/bushound. Он не только позволяет перехватывать USB 1.0/2.0, SCSI/ATAPI/IDE/SATA, FireWire, Bluetooth, Fibre Channel порты, но еще и отображает относительное и абсолютное время выполнения запросов, что существенно облегчает исследование защит, основанных на временных характеристиках.

Наша защита никак не противодействует мониторам шины (Star-Force, кстати говоря, тоже), поэтому все команды видны как на ладони и алгоритм привязки становится понятным с первого взгляда (см. рис. 10).

Рисунок 9 исследование защиты с помощью монитора шины

Как "ослепить" монитор? Можно, конечно, использовать прямой доступ к IDE-контроллеру или разнообразные антиотладочные (точнее, "анти-мониторные") приемы, только… все это сложно, потенциально конфликтно и вообще ненадежно. Лучше "разбавлять" наши команды большим количеством "мусорных" SCSI/ATAPI-команд, чтобы затеряться на их фоне. Алгоритм станет неочевидным и анализ потребует намного больше времени и усилий. Один нюанс: "мусорные" команды ни в коем случае нельзя генерировать на случайной основе, в противном случае, хакер снимет несколько дампов, выделит в них постоянную часть, а все остальное отбросит за ненадобностью. А вот основные команды лучше выбирать случайным образом. В частности, для чтения сектора можно использовать и READ 10/12, и READ CD и READ CD MSF, и некоторые другие.

удаленный прожиг

Ключевые лазерные диски хорошо подходят для "коробочного" ПО, но для программ, распространяемых преимущественно через Интернет, они создают множество трудностей. Рассылка дисков это огромная головная боль, но ведь не передавать же лазерный диск по модему. А почему бы и нет? Современные технологии еще и не такое способы!

Пользователь скачивает клиентскую программу, которая просит вставить чистый CD-R диск в пишущий привод, прожигает одну сессию, заполненную незначащими данными, снимает топологию и по криптографическому протоколу передает эту информацию серверу. Сервер "зашивает" топологию внутрь защищаемой программы и отсылает ее клиентской программа, которая либо записывает программу на диск, либо устанавливает на винчестер.

Конечно, такая схема взаимодействия существенно ослабляет защищенность программы. Ведь во всей партии болванок структура спиральной дорожки одинакова и потому, оплатив только одну копию, нечестный пользователь сможет тиражировать ее чуть ли не в промышленном масштабе. Чтобы этого не произошло, следует использовать аддитивную защиту, комбинирующую привязку к топологии с нестандартным форматом диска, например. Конечно, нестандартный формат — плохая штука (и об этом мы уже говорили в начале статьи), но с некоторыми предосторожностями использовать его все-таки можно. Тем более, что прожиг на CD-R снимает проблему тиражирования — клиентская программа самостоятельно управляет пишущим приводом и может свободно использовать все нестандартные режимы на которые он только способен. Большое количество разнообразных защитных приемов описано в моей книге "Техника защиты лазерных дисков от копирования", демонстрационную версию которой можно найти на моем ftp.

>>> врезка: полезные советы

- положите на ключевой диск файл с заманчивым названием user_name.key и выполняйте над ним различные запутанные операции, это слегка умерит прыть хакера и остановит его на какое-то время;
- вносите на диск несколько ***принципиально различных*** ключевых меток, но проверяйте только часть из них, а часть — оставьте для последующих версий программы (или ее обновлений), тогда хакеру придется каждый раз придется проводить утомительные исследования, подолгу зависая над монитором шины, дизассемблером и отладчиком;
- помните, что даже тщательно оттестированная защита в силу некоторых причин может не сработать, "обругав" легального пользователя; поэтому, необходимо использовать по меньшей мере три независимых защитных механизма, основанных на различных характеристиках носителя, и если срабатывают хотя бы два из них, проверка считается пройденной; в данной статье мы рассмотрели всего лишь один такой механизм, остальные — в следующий раз.

>>> врезка nezimi ftp

Мыщъх поднял экспериментальный ftp-сервер, раздающий свои авторские материалы:
IP-адрес: 83.239.33.46;
доменное имя: nezumi.org.ru;
порт: 21 (стандартный),
логин: WASM,

пароль: не требуется;
папка: /pub,
канал: 500 мегабит;
приблизительное время работы: с 14:00 до 06:00 (мыщъх ночной зверь), при возникновении проблем рекомендуется установить пассивный режим ftp-клиента;

заключение

Разработка собственных защит от копирования — это реальность. Забудьте о широко разрекламированных готовых пакетах. Всякая серийная защита притягивает внимание сотен тысяч хакеров со всего света, которые быстро находят более или менее универсальный способ взлома, такой, что им может воспользоваться даже простой обыватель. Стойкость несерийных защит (даже спроектированных непрофессионалами) зачастую оказывается намного выше, потому что никто не хочет с ними связываться. Единичный взлом себя не окупит!

Надеюсь, что эта статья поможет вам избежать ошибок и разработать защиту свой мечты самостоятельно.