Как обеспечить безопасность данных

Фленов Михаил http://www.vr-online.ru

То, что данные нужно защищать, понятно даже ежику в тумане. В особенности это касается баз данных, потому что в корпоративных хранилищах очень часто хранится вся жизнь фирмы. Но защищать нужно не только от хакеров, но и от особо одаренных чайников, которые своими нелепыми ручонками вечно удаляют не то, что нужно.

Супер админ

Во время установки MS SQL Server 7.0 и младше имя администратора по умолчанию выбиралось sa (System Administrator). Пароль можно было не указывать, и система абсолютно без проблем хавала что угодно. В этом MS как всегда выделялась, хотя на каждом углу весят плакаты, что нельзя устанавливать пустые пароли, особенно админским учетным записям.

Начиная с MS SQL Server 2000, установщик уже предупреждает о возможных проблемах, если не указать пароль. Наконец-то кто-то увидел надпись на заборе, что нельзя выбирать простые пароли, а пустой пароль это вообще пробоина в безопасности, как дыра в корпусе Титаника. С такими паролями тонут в первые же дни плаванья.

Когда устанавливаешь базу данных, и инсталлятор предложит выбрать пароль, обязательно указывай ычто-нибудь не короче, чем 8 символов и сложное для подбора. Это значит, что нельзя выбирать в качестве пароля читаемые слова или даты, такие вещи подбираются за пять сек. Лучше выбрать что-то нечитаемое. Лично я всегда наугад набираю что-нибудь на клавиатуре, а потом просто сохраняю эту ерунду в секретном файле, который защищаю от таких хакеров как ты.

> В настоящее время ни один сайт и ни одна контора не может прожить без удобного хранилища для данных, в качестве которого чаще всего выступают базы данных. В истории были случаи, когда уничтожение каких-то данных приводило к банкротству фирм из-за дилетантства админов.

Хочется заметить, что во время установки MySQL до сих пор в качестве админа используется учетная запись root без пароля. Эта запись не связана с пользователем root из ОС, поэтому пароль необходимо поменять сразу после установки сервера. Для его смены надо выполнить команду:

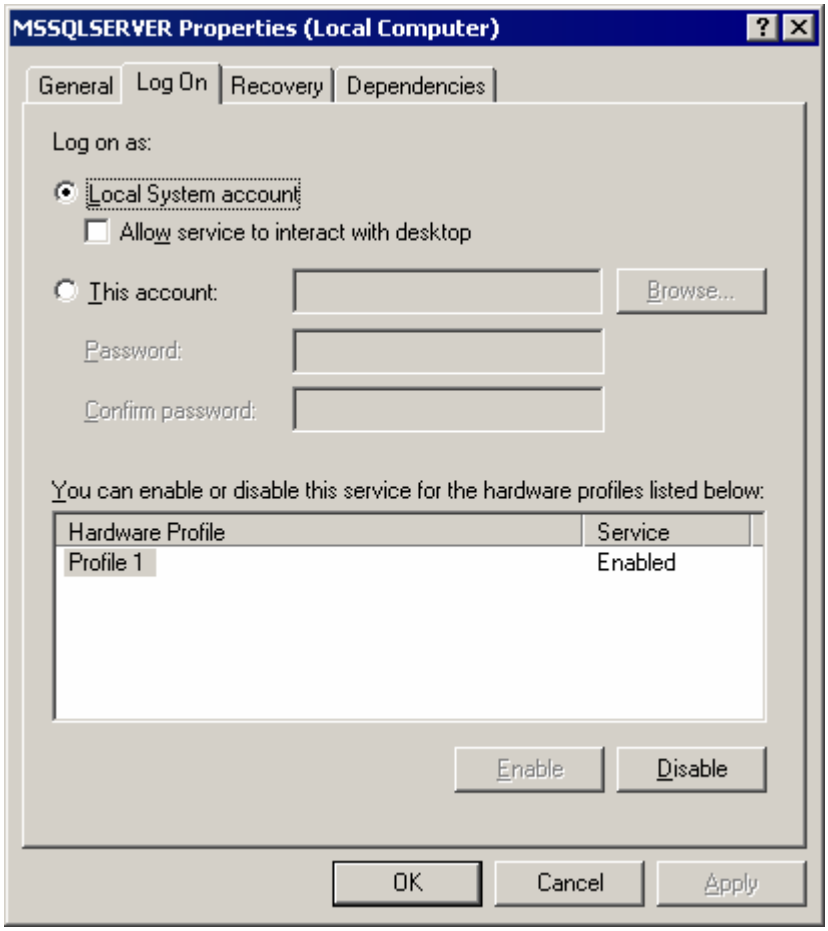
```
/usr/bin/mysqladmin –uroot password newpass
```

Вместо newpass нужно указать новый пароль пользователя root.

Работа сервиса

Следующие замечания касаются только Windows баз данных, потому что все они работают в системе как службы. По умолчанию все службы в Windows работают под системной учетной записью. Но у нее слишком много прав и если хакер через баг сможет проникнуть в сервер базы данных, то сможет выполнять команды в системе от имени локального пользователя. Чтобы ограничить права, нужно изменить пользователя, от имени которого стартует служба. Для этого войди в Панель управления/Администрирование/Службы и найди здесь службы своего сервера. Для

SQL Server это: MSSQLServer и SQLServerAgent. Дважды щелкаем по обеим записям и в появившемся окне свойств переходим на закладку Log on (Вход в систему).

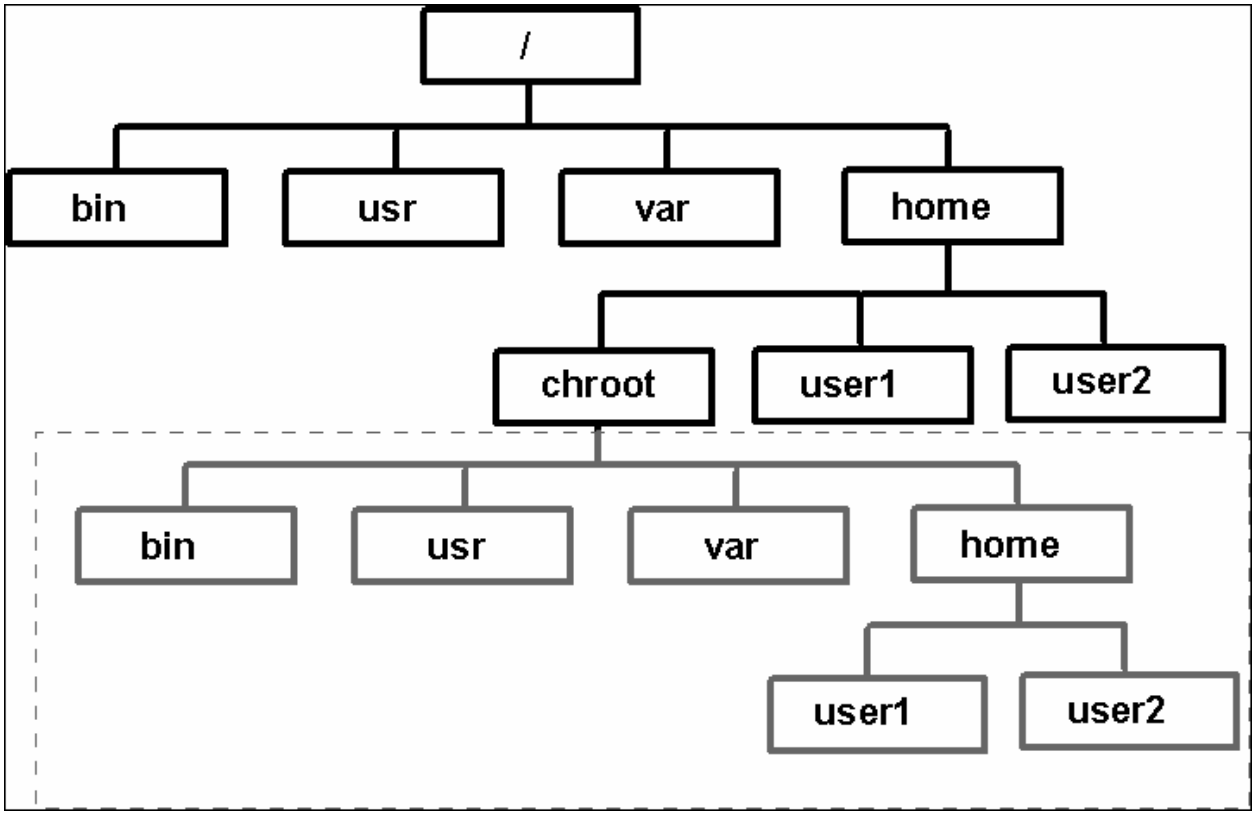


Настройка учетной записи службы

Теперь выбираем пункт This account (с учетной записью) и указываем имя и пароль нужного пользователя. В идеале, необходимо создать в системе новую учетную запись, которой предоставлены только те права, которые реально нужны этому сервису. Ничего лишнего давать нельзя.

Работа демона

В Linux дела обстоят немного сложнее, но намного безопаснее. Здесь нужно создать виртуальную директорию, которая будет являться корневой для сервиса. Для этого я рекомендую скачать утилиту jail с сайта http://www.jmcresearch.com/projects/jail/. Пример работы утилиты рассмотреть не могу, потому что сковывают объемы журнала, а тут нужна отдельная статья, поэтому расскажу только принцип. За более подробной инфой можешь обратиться к справочным файлам или купить книгу «Linux глазами Хакера», которая выйдет летом 2005-го года, где по полочкам разложена вся основная информация по безопасности ОС Linux.



Виртуальная директория chroot

Итак, служба базы данных в Linux должна работать в своей виртуальной директории. Выше это директории программа попасть не может. Посмотрим на схему виртуальной директории. Здесь показана часть файловой системы Linux. Во главе всего стоит корневая директория /. В ней находятся /bin, /etc, /home, /usr и т.д. В /home расположены каталоги пользователей системы. Мы создаем здесь новую директорию, для примера назовем ее chroot и она будет является корнем для службы. В ней будут свои каталоги /bin, /usr и т.д. и служба будет работать с ними, а все, что выше /home/chroot оказывается недоступным. Просто служба будет считать, что /home/chroot – это и есть корень файловой системы.

Одной из причин падения акций фирм, расположенных в знаменитых башнях США после 11 сентября была потеря корпоративных данных. А ведь в башнях были офисы крупных корпораций и много серверов с секретными и особо важными данными.

На рисунке в рамку обведены папки, которые будут видны службе. Именно в этом пространстве будет работать сервер баз данных и будет считать, что это и есть реальная файловая система сервера.

Если хакер проникнет в систему через защищенную службу и захочет просмотреть каталог /etc, то он увидит каталог /home/chroot/etc, но никак не системный /etc. Чтобы взломщик ничего не заподозрил, в каталоге /home/chroot/etc можно расположить все необходимые файлы, но содержащие некорректную информацию. Взломщик, запросив файл /etc/passwd через уязвимую службу, получит доступ к /home/chroot/etc/passwd, потому что служба видит его системным.

Так, например, файл /home/chroot/etc/passwd может содержать неверные пароли. На работу системы в целом это не повлияет, потому что система будет брать пароли из файла /etc/passwd, а службе реальные пароли системы не нужны, поэтому в файл /home/chroot/etc/passwd можно засунуть что угодно.

Типы аутентификации

Теперь переходим к знакомству с пользователями базы данных. В большинстве баз данных учетные записи пользователей хранятся в самой базе (в виде системных таблиц или настроечных файлов). В SQL Server 2000 пошли дальше. Здесь может быть два типа аутентификации – Windows и Смешанная.

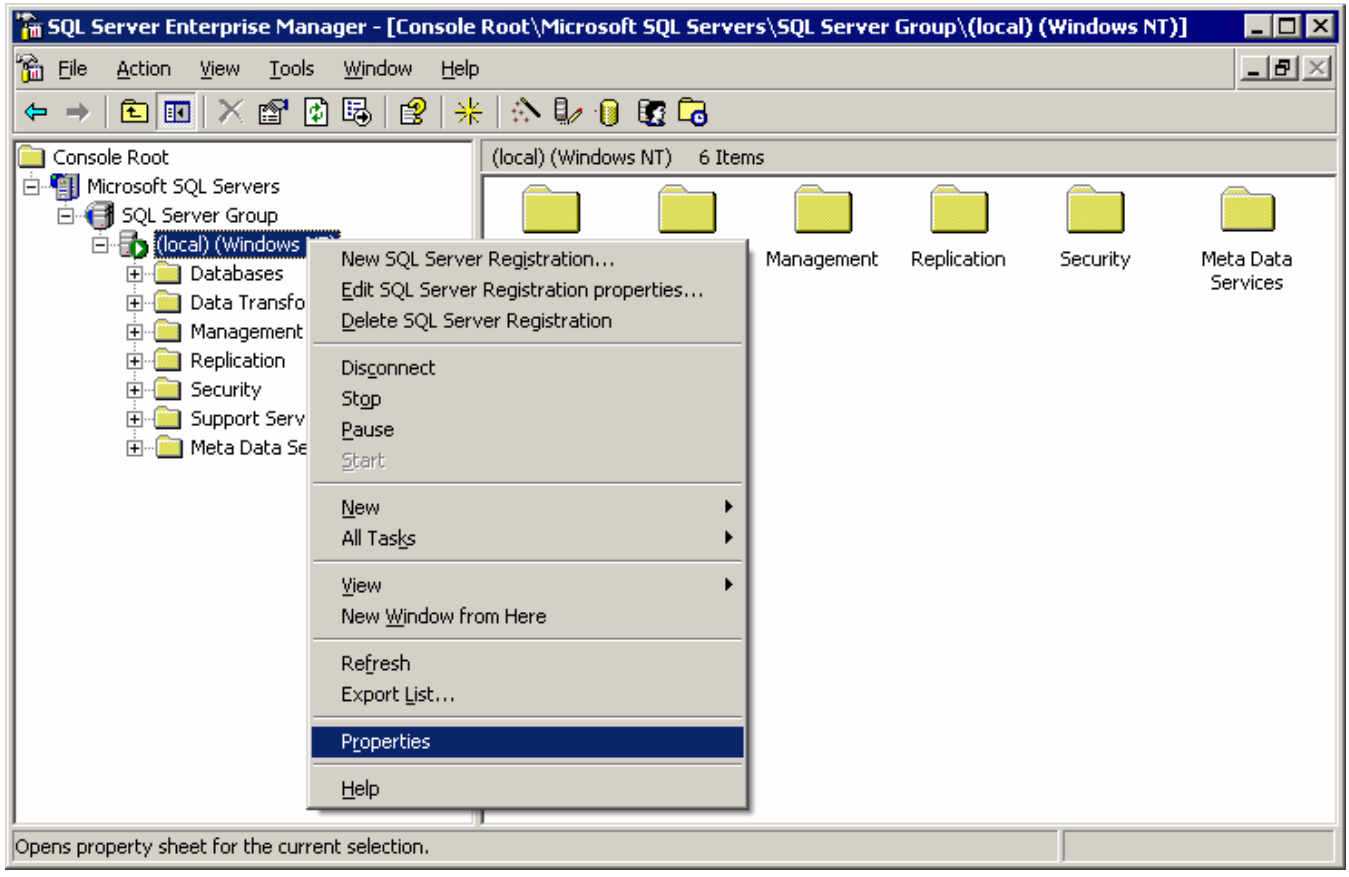
Если выбрана аутентификация Windows, то для проверки пользователей используются учетные записи Windows и ее встроенные механизмы проверки подлинности. Именно этот метод я рекомендую использовать, потому что в нынешних дистрибутивах для аутентификации используется Kerberos, который достаточно надежен и проверен временем в *nix подобных системах.

В случае смешанного режима, можно создавать пользователей, информация о которых будет храниться SQL сервером в системных таблицах. А это уже не есть хорошо по следующим причинам:

- нужно управлять двумя базами пользователей. В большинстве случаев заниматься этим лень, поэтому чаще всего все юзеры работают под одной учетной записью или записи соответствуют тем, что заведены для них в ОС Windows. Таким образом, взломав SQL Server, взломщик получает доступ к паролю, который открывает все двери в системе.

- пользователям нужно знать два пароля – на вход в Windows сервер для работы с файлами и на SQL Server. Конечно же, если юзеру нужен доступ только к базе данных, то для работы нужен будет только один пароль, с правами доступа к SQL Server.

- ОС хранит свои пароли более надежно, с хорошим шифрованием и с запретом на чтение файле. В MS SQL сервере защита записей проще и все записи при наличии прав админа легко прочитать в таблице sysusers базы данных Master.



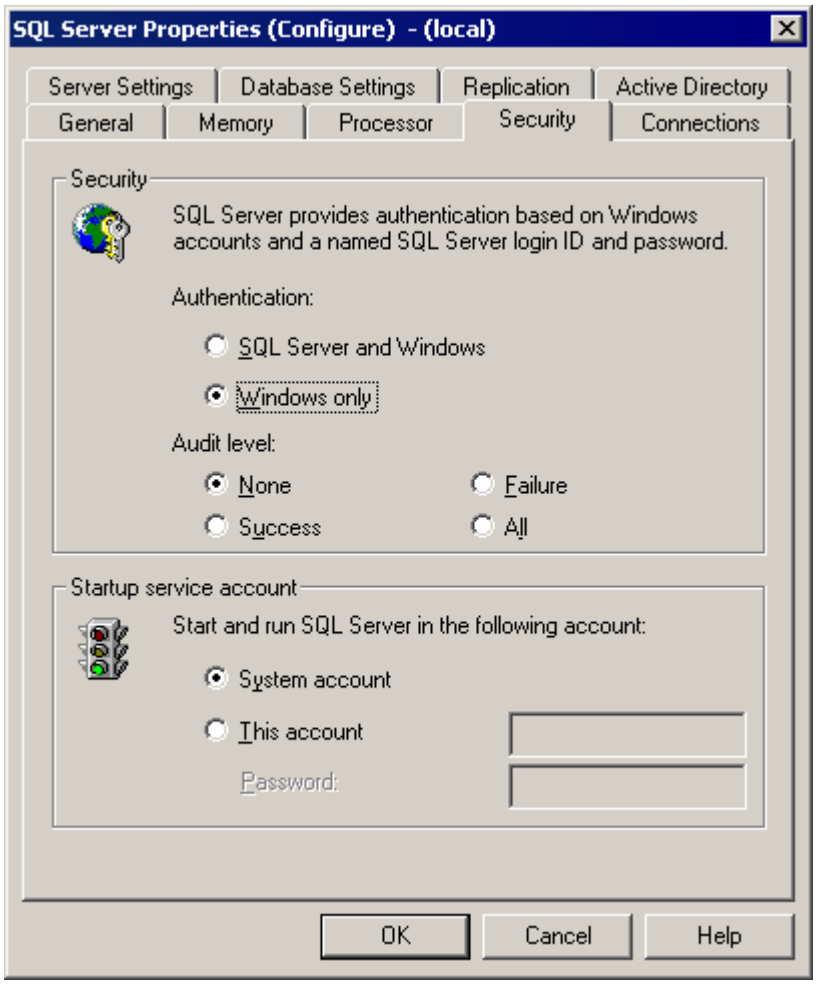
Окно управление сервером в Enterprise Manager

В данной статье я буду рассматривать оба варианта хранения паролей, потому что не все базы данных (я имею ввиду, отличные от MS) поддерживают аутентификацию Windows.

Аутентификация MS SQL

В MS SQL Server все настройки происходят в SQL Enterprise Manager. Запусти эту программу и перед тобой откроется окно, рабочая область которого будет поделена на две части: слева дерево объектов, справа будет отображаться то, что содержит выделенный в дереве объект.

Открой ветку Microsoft SQL Servers. В ней содержаться группы серверов. По умолчанию создается группа с именем SQL Server Group. Выдели группу, и в ней увидишь все сервера. Если есть локальный сервер, то он будет там единственный, пока не зарегистрируешь другие серверы баз данных (удаленные или локальные). Щелкни по имени сервера и в появившемся меню выбери пункт Properties. Перед нами открывается окно свойств сервера. Перейди на закладку Security и здесь можно будет увидеть переключатель между режимами.



Настройка аутентификации

Здесь же можно выбрать уровень аудита (Audit Level). По умолчанию выбран None, а значит, сервер не будет сохранять в логах информацию об удачных или неудачных входах в систему. Все знают, что в продуктах MS настройки по умолчанию далеки от идеала, но то, что в логах не будет инфы о входах пользователей, это подобно

катастрофе. Срочно переключай аудит на All, чтобы можно было контролировать, кто и когда входил или пытался войти, но неудачно.

Внешние ключи

Как ключи могут повлиять на безопасность? Ведь это всего лишь связь между двумя таблицами. Все очень просто. Чаще всего связь построена по принципу главный-подчиненный (один ко многим). В одной таблице находится главная строка, а в другой множество подчиненных строк. Например, допустим, что у тебя есть две таблицы – одна для хранения списка сотрудников (People), а другая с их зарплатами за каждый месяц (Salary). Если попытаться удалить запись из таблицы Peoples, для которой есть подчиненные записи в Salary, то произойдет ошибка. Сначала нужно удалить все подчиненные записи.

Ты еще не видишь преимущество вторичных ключей? А я вижу. Таблицы сотрудников могут быть защищены не сильно, потому что с ними работает множество народа и текущий список может быть доступен через WEB. А вот зарплата всегда защищается, чуть ли не с пулеметом. Если хакер получит доступ к Peoples и попытается удалить все записи, то ничего не выйдет. Внешние ключи не дадут врагу сделать свое черное дело, пока не будут удалены соответствующие записи из Salary. А так как там защита лучше, то и сделать это будет сложно.

Вот тебе решение простейшей задачи защиты от удаления. Если есть публичная таблица, из которой нельзя удалять (Public), создай для нее подчиненную таблицу (Slave), защищенную по полной программе, и свяжи их внешним ключом. При создании новой записи в главной таблице, в подчиненную должна добавляться связанная строка. Эта связь сделает удаление из Public невозможным, пока хакер не найдет закрытую от всеобщего взора таблицу Slave.

Триггеры

Не менее интересным способом обеспечения безопасности являются триггеры. Это код, похожий на процедуры, хранимые на сервере. Такой код нельзя вызвать напрямую и он выполняется в ответ на определенные события (Вставка, Изменение и Удаление строк). Внутри триггера можно проверить корректность выполняемых действий. Если хакер попытается испортить данные, то в триггере можно увидеть этот косяк.

> Дополнительную безопасность могут обеспечить constraint – ограничение на допустимые значения. Например, колонку для хранения пола человека можно ограничить вводом только значениями эМ и Же, тогда ламеры и хакеры будут иметь меньше возможностей навредить.

Рассмотрим пример защиты таблицы от изменений через триггер. Для защищенной таблицы заводим поле Security. В этом поле должен хранится код, который вычисляется известным только тебе способом, например расчет контрольной суммы всех полей. Если пользователь изменил значения какой-либо строки с помощью программы, то она автоматически пересчитывает контрольную сумму. Если строка изменена напрямую, то в поле Security будет некорректное значение, которое легко определить в триггере (который должен выполняться на события изменения данных) и откатить злостное изменение.

Точно также можно защищать таблицы не только от изменения, но и от вставки (защита от флуда на базу данных) и удаление (попытки уничтожить важные данные).

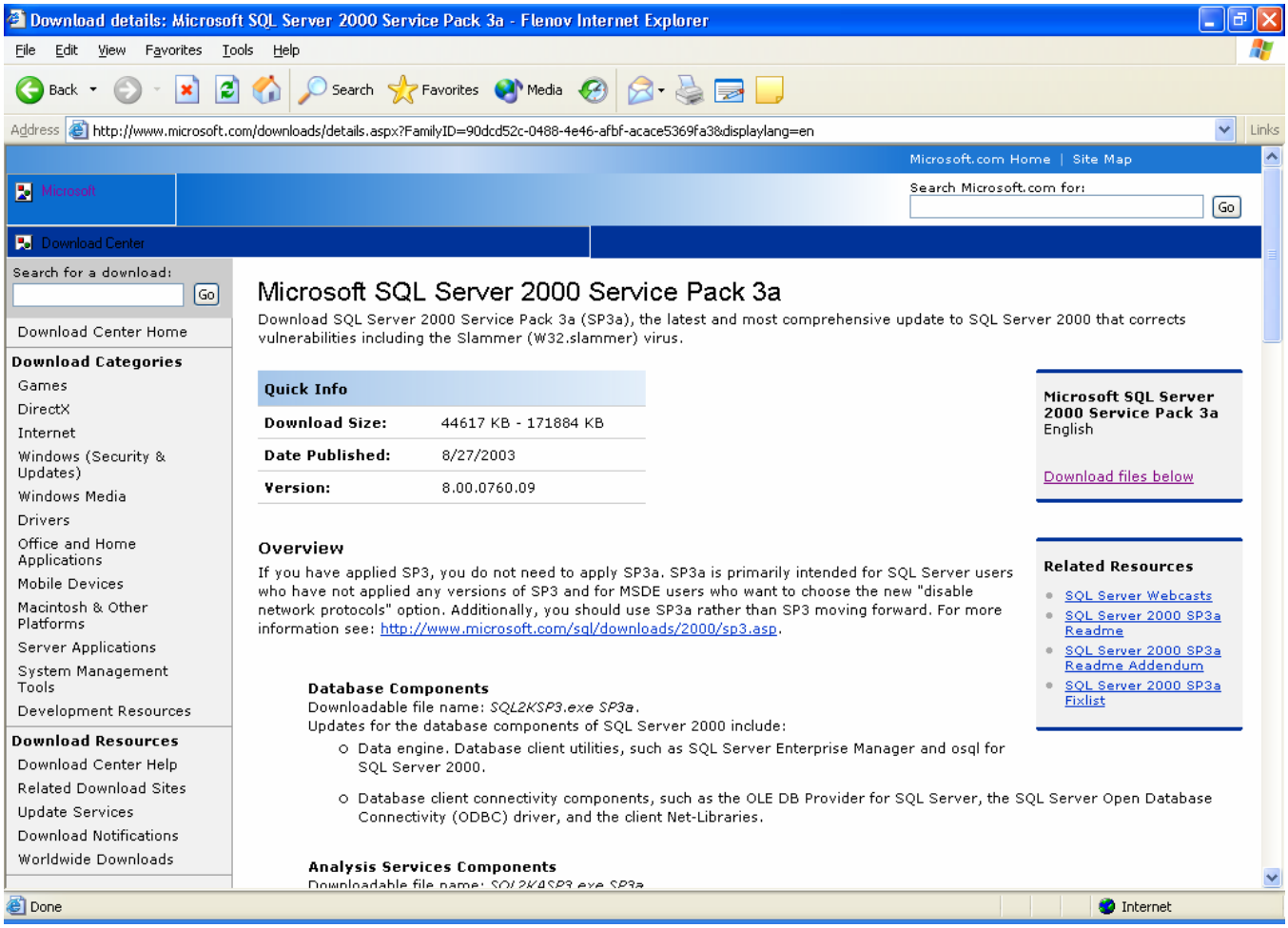
Права доступа

Любые попытки отконфигурировать базу данных на полную безопасность превратятся в пустую трату времени, если ты неправильно настроишь права доступа на объекты. Если все объекты базы данных и сами данные светятся в Интернете как гирлянда на Кремлевской елке, то работа админа бесполезна. Мы должны первым делом правильно

настроить права доступа. В этом случае, даже если хакер проникнет в систему, у него может не хватить прав на доступ ко всем секретам. О правах доступа мы поговорим более подробно в отдельной статье.

Обновление

В любой программе есть ошибки и производитель или тип тут не играет роли. Спроси любого хакера – какое ядро Linux самое безопасное? Ответ очевиден – самое последнее. Но это не значит, что это ядро не содержит ошибок, просто о них еще никто не знает. Повтори свой вопрос через пол года, и то ядро, которое называлось ранее, хакер может назвать самым дырявым в истории Linux.



Страница MS с сервис паком для SQL Server 2000

Вот такая природа человечества и программного обеспечения. Ошибки есть всегда и везде. Как только появляется критическая ошибка в какой-либо базе данных, так сразу у админов по всему миру начинается черный день, потому что в первые дни после выхода эксплоита хакеры ломают все подряд.

Основная задача специалиста по безопасности вовремя выявлять эти ошибки и исправлять раньше, чем взломщик воспользуется уязвимостью. Для этого нужно быть подписанным на все основные BagTraq и регулярно следить за выходами обновлений твоей базы данных. По моей практике могу сказать, что лучше всего на это дело реагируют Oracle и MS. Их патчи выходят достаточно быстро и если сразу обновить, то вероятность взлома уменьшается.

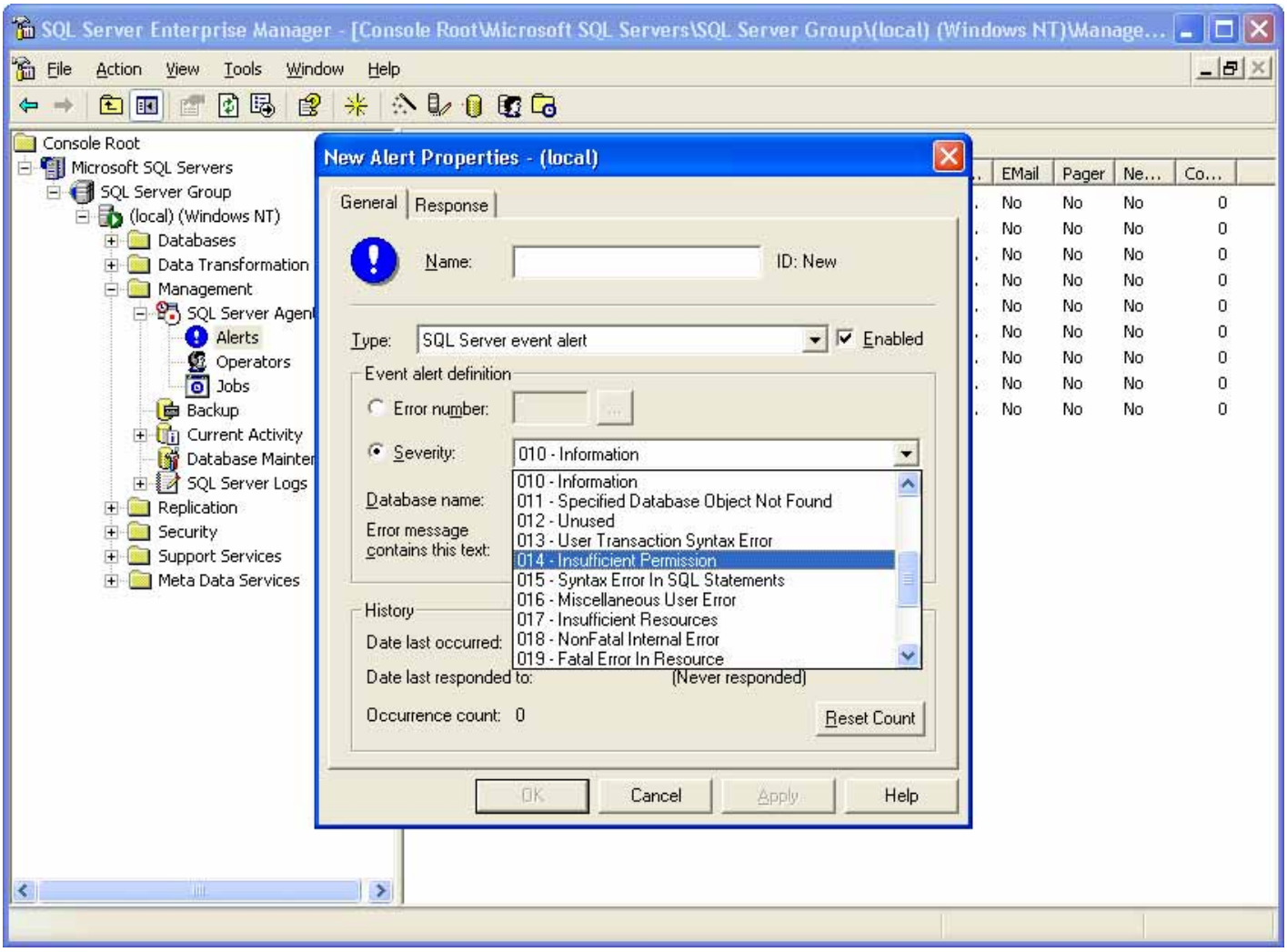
Помни, что основными причинами взлома являются неправильное распределение прав и не вовремя пропатченный софт.

Моментальная реакция

Между взломщиками и специалистами по безопасности идет самая настоящая война. В ней побеждает тот, кто больше знает и кто быстрее реагирует. Если не хочешь оказаться среди проигравших, то научись быстро реагировать на происходящее. В базах данных для этого есть множество удобных прибамбасов, и в этом отношении одним из лучших является MS SQL Server. В этом сервере есть очень удобный помощник – события.

Сервер баз данных может ловить достаточно много событий. Наиболее интересным с точки зрения безопасности может быть Insufficient permission (не достаточные права). Допустим, что хакер пытается проникнуть в систему и удалить все данные. На каком-то этапе исследования он узнает пароль доступа одного из пользователей и запускает команду DELETE FROM DatabaseName. Если прав не достаточно, то хакер будет искать другую учетную запись и пароль к ней, и так, пока не найдет нужную жертву.

Задача защищающей стороны вовремя определить попытку взлома и в этом помогают события. Когда хакер неудачно выполнил команду, система генерирует ошибку Insufficient permission, и чем быстрее мы узнаем, что эта ошибка произошла, тем быстрее сможем принять меры. Например, узнав об ошибке, можно тут же добавить в сетевой экран фильтр и запретить любое подключение с IP адреса злоумышленника. Таким образом, можно выиграть время, пока хакер будет обходить правила сетевого экрана. А начинающего хакера такие вещи пугают и он убежит сломя голову и больше не вернется.



Окно создания нового события

Как создаются события? В Enterprise Manager открываем ветку Management/SQL Server Agent/Alerts. Здесь щелкаем правой кнопкой и в появившемся меню выбираем New Alert. Перед нами открывается окно создания нового события. Здесь нужно заполнить следующие поля:

- name – имя, которое может быть любым;

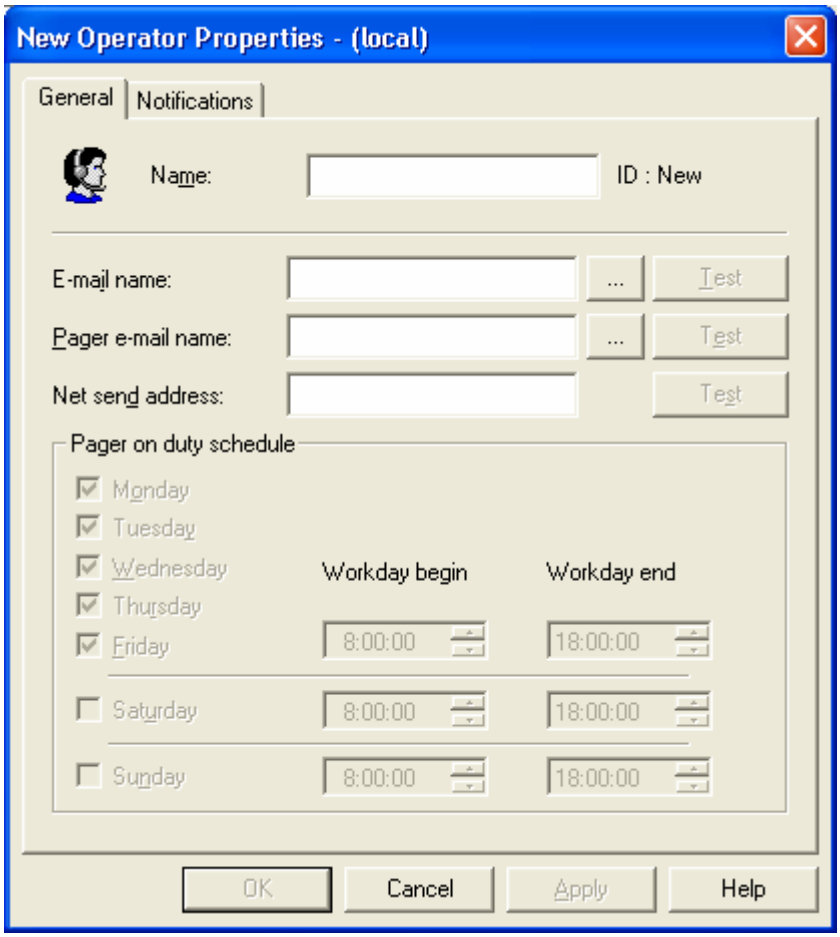
- type – тип события может быть event alert (здесь все основные события) и performance condition alert (события производительности);

- Severity – здесь нужно указать конкретное событие, которое вы хотите отлавливать;

На закладке Response можно указать операторов, которым нужно отсылать сообщения (e-mail, net send или на пейджер), о возникновении события.

Обязательно используй внешние ключи для объединения таблиц. Эти ключи помогут сохранить целостность данных и не позволят удалить строки, если есть существующие связи. Так хакер не сможет очистить главную таблицу, не подчистив подчиненную, на которую может и не быть прав.

Операторы – это просто контактная информация людей, ответственных за работу сервера. Например, вы можете указать себя и свой e-mail и при возникновении события, на ящик будет падать письмо с информацией об ошибке. Таким образом, как только возникает критическое событие, вы первым узнаете об этом, и не надо будет лишний раз осматривать весь журнал безопасности.



Окно создания нового оператора

Итого

В данной статье мы рассмотрели основы безопасности и средства, которые предоставляют базы данных. Но нельзя забывать, что уязвимыми могут быть не только настройки, но и сама ОС или программы базы данных. Ошибки есть в любом софте, поэтому не забывай следить за сообщениями об ошибках и обновлять сервер. Надежда на то, что тебя не взломают – рисковое дело. Когда-нибудь найдется человек, который просто от скуки или в отместку напишет DROP DATABASE и можно будет распрощаться с многолетними трудами.

В защите данных неплохо помогают триггеры – функции, которые выполняются на определенные действия (вставка, изменение, удаление). В них можно проверять правильность выполняемых действий и если заметить что-то подозрительное, то можно забить тревогу и отменить злостные действия откатом транзакции.