Segurança em Sistemas de Login: Proteção Contra SQL Injection

27 dez
2010 **5 Comentários** (http://rberaldo.com.br/seguranca-em-sistemas-de-login-protecao-contra-sql-injection/#comments) Postado por **Beraldo** (http://rberaldo.com.br/author/admin/)

Dando continuação ao artigo sobre Segurança em Sistemas de login, hoje mostrarei como se defender de ataques por SQL Injection. Foco MySQL e PostgreSQL, os principais SGBDs gratuitos da atualidade.

Caso não tenha visto a primeira parte do artigo, leia-a aqui:
Segurança em SIstemas de Login: Senhas e Cookies (http://www.rberaldo.com.br/blog/2010/12/22/seguranca-em-sistemas-de-login-senhas-e-cookies/)

Sumário

1. Introdução (#intro)
2. Magic Quotes (#magic)
3. Soluções Especificas para cada SGBD (#solucoes)
3.1. MySQL (#mysql)
3.2. PostgreSQL (#postgre)
3.3. Exemplos de códigos para MySQL e PostgreSQL (#exemplos)
4. Apenas Isso Não Basta (#apenas)
5. Conclusão (#conclusao)

1. Introdução

Neste segundo artigo sobre **segurança** em sistemas de login, abordarei formas de proteção contra **SQL Injection**

(http://pt.wikipedia.org/wiki/Inje%C3%A7%C3%A3o_de_SQL) .

Existem muitas discussões na Internet, em listas de discussão e fóruns, sobre qual seria a função perfeita para impedir ataque por SQL Injection SQL Injection. Alguns programadores até criam funções que removem, por segurança, palavras-chave da linguagem SQL, como SELECT, DROP, DELETE. Isso pode até resolver, mas não podemos danificar a informação; se permitirmos que o usuário escreva informações em nosso site, devemos permitir-lhe escrever
SELECT, DROP e DELETE também. Imagine, por exemplo, um fórum sobre programação: como poderíamos postar códigos SQL se o fórum removesse as palavras SELECT, DELETE etc? Logo, não podemos remover essas palavras.

A soluçãoo é muito simples! Sim, é simples, mesmo. Muitos querem complicar à toa, porém é muito simples: **escapar caracteres especiais**.

Esses caracteres especiais podem variar conforme o **SGBD** (http://pt.wikipedia.org/wiki/Sistema_de_gerenciamento_de_banco_de_dados) que se está utilizando. Normalmente são aspas simples e duplas, as quais delimitam strings em um comando SQL.

Vamos a um exemplo. Considere a SQL abaixo:

```
$sql = "SELECT id, nome, sobrenome FROM autores WHERE nome = '" . $nome . "'
                AND sobrenome = '" . $sobrenome . "'";
```

Supondo que $nome contenha **jo'sé**, e $sobrenome, **silva**, a SQL ficará assim:

```
SELECT id, nome, sobrenome FROM autores WHERE nome = 'jo'se' AND sobrenome = 'silva';
```

Isso gera um erro de sintaxe, sem comprometer o banco de dados. Porém, se mantivermos $sobrenome e definirmos $nome com o valor **jo'; DROP TABLE autores ; –**, teremos:

```
SELECT id, nome, sobrenome FROM autores WHERE nome = 'jo'; DROP TABLE autores ;
--'AND sobrenome = 'silva';
```

Dessa forma, selecionam-se os registros com nome igual a "jo", remove-se a tabela "autores" e considera-se **' AND sobrenome = 'silva';** como comentário. Isso caracteriza um ataque por SQL Injection.

2. Magic Quotes

Face aos possíveis grandes danos que SQL Injection pode causar, o PHP possui um mecanismo nativo automático para escapar caracteres especiais: o *magic quotes*. Porém, esse é um mecanismo genérico, que não pode ser aplicado a todos os SGBDs. Logo, não o utilize!

O próprio Manual do PHP não recomenda seu uso:

```
There is no reason to use magic quotes because they are no longer a supported
part of PHP. However, they did exist and did help a few beginners blissfully
and unknowingly write better (more secure) code. But, when dealing with code
that relies upon this behavior it's better to update the code instead of
turning magic quotes on. So why did this feature exist? Simple, to help prevent
SQL Injection. Today developers are better aware of security and end up using
database specific escaping mechanisms and/or prepared statements instead of
relying upon features like magical quotes.
```

Fonte: **http://br3.php.net/manual/en/security.magicquotes.why.php** (http://br3.php.net/manual/en/security.magicquotes.why.php)

Como citado no trecho, é preferível adaptar seus códigos a fim de torná-los seguros e não vulneráveis a SQL

Injection a habilitar o magic quotes. Portanto mantenha a diretiva **magic_quotes_gpc**, do , em **off**! Dê preferência a funções específicas para cada SGBD.

Leia o capítulo sobre Magic Quotes, do }Manual do PHP, no link abaixo:
http://br3.php.net/manual/en/security.magicquotes.php (http://br3.php.net/manual/en/security.magicquotes.php)

3. Soluções Especificas para cada SGBD

3.1. MySQL

Vamos ao exemplo mais comum: MySQL: existe uma função específica do PHP para escapar caracteres especiais do MySQL: **mysql_real_escape_string (http://br3.php.net/manual/pt_BR/function.mysql-real-escape-string.php)** .

Ela deve ser usada com magic_quotes_gpc em off. Caso seu servidor mantenha essa diretiva ativa, **desabilite-a por meio de htaccess (http://php.net/manual/pt_BR/security.magicquotes.disabling.php)** ou, caso isso não seja possível, certifique-se de usar **stripslashes (http://php.net/manual/en/function.stripslashes.php)** antes de aplicar essa função. Veja o exemplo abaixo:

```
if ( get_magic_quotes_gpc() )
{
    $name = stripslashes( $name );
}
$name = mysql_real_escape_string( $name );
mysql_query( "SELECT * FROM users WHERE name=$name" );
```

Esse trecho de código e outras dicas sobre prevenção de SQL Injection com MySQL podem ser vistas no link abaixo, do próprio Manual do MySQL:
http://dev.mysql.com/tech-resources/articles/guide-to-php-security.html (http://dev.mysql.com/tech-resources/articles/guide-to-php-security.html)

3.2. PostgreSQL

O escape de caracteres no PostgreSQL não é feito com barra invertida; é feito com aspas simples. Ou seja, addslashes não funcionaria aqui.

O PHP também tem uma função específica para escape de caracteres especiais para PostgreSQL:
pg_escape_string (http://php.net/manual/en/function.pg-escape-string.php) .

Mais informações sobre prevenção de SQL Injection em PostgreSQL podem ser vistas no link abaixo, do Wiki do PostgreSQL:
http://wiki.postgresql.org/wiki/Sql_injection (http://wiki.postgresql.org/wiki/Sql_injection)

3.3. Exemplos de códigos para MySQL e PostgreSQL

```
mysql_connect( 'localhost', 'usuario', 'senha' );
$str = "There's no place like 127.0.0.1, the \"localhost\"";
echo "String:   " . $str . "<br />";
echo "MySQL:    " . mysql_real_escape_string( $str ) . "<br />";
echo "Postgre:  " . pg_escape_string( $str ) . "<br />";
```

* Para usar mysql_real_escape_string, é necessário uma conexão MySQL ativa.\

Saída:

```
String: There's no place like 127.0.0.1, the "localhost"
MySQL:  There\'s no place like 127.0.0.1, the \"localhost\"
```

```
Postgre:        There''s no place like 127.0.0.1, the "localhost"
```

4. Apenas Isso Não Basta

Apenas escapar caracteres não é suficiente, uma vez que não existem apenas strings. Também temos dados numéricos, como inteiros, floats e outros tipos de ponto flutuante, que não são envolvidos por aspas em consultas SQL.

Considere a seguinte SQL:

```
$sql = "SELECT id, nome, sobrenome FROM autores WHERE id=" . $id;
```

Se $id tiver o valor **0; DROP TABLE autores; –**, a SQL final será:

```
SELECT id, nome, sobrenome FROM autores WHERE id=0; DROP TABLE autores; --;
```

Isso removeria a tabela “autores”.

A solução é, novamente, muito simples: basta fazer ***casting*** **(http://en.wikipedia.org/wiki/Cast_(computer_science))**, ou coerção, convertendo o parâmetro para um tipo numérico.

No exemplo acima, bastaria isto:

```
$id = (int) $id;
```

O mesmo vale para float, double e os demais tipos de dados.

5. Conclusão

SQL Injection é um problema muito grave, que muitos programadores iniciantes deixam passar despercebido, principalmente por falta de conhecimento.

Apesar disso, sua prevenção é muito simples. Basta entender o funcionamento do ataque para saber como se defender dele.

Abraços,

Beraldo

Contribua com o Blog do Beraldo

Gostou do conteúdo e deseja contribuir para a manutenção e para o crescimento do Blog do Beraldo? Se você deseja ajudar a equipe, pode fazer uma doação pelo PagSeguro.

Postado em: Banco de Dados (http://rberaldo.com.br/category/banco-de-dados/) , PHP (http://rberaldo.com.br/category/programacao/php/) , Programação (http://rberaldo.com.br/category/programacao/) , Segurança (http://rberaldo.com.br/category/seguranca/) - **Tags:** escape (http://rberaldo.com.br/tag/escape/) , injection (http://rberaldo.com.br/tag/injection/) , login (http://rberaldo.com.br/tag/login/) , magic (http://rberaldo.com.br/tag/magic/) , magic_quotes (http://rberaldo.com.br/tag/magic_quotes/) , mysql_real_escape_string (http://rberaldo.com.br/tag/mysql_real_escape_string/) , pg_escape_string (http://rberaldo.com.br/tag/pg_escape_string/) , PHP (http://rberaldo.com.br/tag/php/) , Programação (http://rberaldo.com.br/tag/programacao/) , quotes (http://rberaldo.com.br/tag/quotes/) , segurança (http://rberaldo.com.br/tag/seguranca-2/) , sql (http://rberaldo.com.br/tag/sql/) , stripslashes (http://rberaldo.com.br/tag/stripslashes/)

DevMedia Posts:

Posts Relacionados

- Server Push: Long Polling usando PHP (http://rberaldo.com.br/server-push-long-polling-php-ios/) — *Há situações em que precisamos obter uma resposta de um s...*
- Como usar o servidor nativo do PHP 5.... (http://rberaldo.com.br/como-usar-o-servidor-nativo-do-php-5-4/) — *Até o PHP 5.3.*, era necessário instalar um Servidor Web,...*
- Desenvolvimento iOS: como realizar op... (http://rberaldo.com.br/desenvolvimento-ios-como-realizar-operacoes-multi-thread-com-nsoperation/) — *Sempre esperamos que nossos dispositivos (principalmente ...*
- Desenvolvimento iOS: iniciando com Ta... (http://rberaldo.com.br/desenvolvimento-ios-iniciando-com-tableview/) — *Para iniciar a série de tutoriais sobre Desenvolvimento i...*
- PHP 5.4 lançado oficialmente (http://rberaldo.com.br/php-5-4-lancado-oficialmente/) — *Ontem, dia 1º de março, o lançamento do PHP 5.4 foi anunc...*

« **Segurança em sistemas de login: senhas e cookies (http://rberaldo.com.br/seguranca-em-sistemas-de-login-senhas-e-cookies/)**
» **Write in C (http://rberaldo.com.br/write-in-c/)**

4 Comments

(http://rberaldo.com.br/seguranca-em-sistemas-de-login-protecao-contra-sql-injection/feed/)

1. ***Andrey Knupp (http://dev.mbiosinformatica.com.br)***

13/02/2011 at 21:07 | **Permalink (http://rberaldo.com.br/seguranca-em-sistemas-de-login-protecao-contra-sql-injection/comment-page-1/#comment-3428)**

Otimo artigo , devia publicar no forum em alguns sistemas de login que o pessoal não usa uma funçao ou algum meio de evitar
esses problemas , parabens , seguindo suas postagens aqui , ^^

ja foi pros favoritos xD

Abraços

Reply (/seguranca-em-sistemas-de-login-protecao-contra-sql-injection/?replytocom=3428#respond)

2. *Gabriel Beraldo*

17/02/2011 at 11:09 | **Permalink (http://rberaldo.com.br/seguranca-em-sistemas-de-login-protecao-contra-sql-injection/comment-page-1/#comment-3476)**

Obrigado Primo,

tambem sou TI e descobri um pessoal de sistemas detonando meu ambiente de backup com um injection.

Abraço,

Gabriel Beraldo

Reply (/seguranca-em-sistemas-de-login-protecao-contra-sql-injection/?replytocom=3476#respond)

3. *Anderson*

07/04/2012 at 15:02 | **Permalink (http://rberaldo.com.br/seguranca-em-sistemas-de-login-protecao-contra-sql-injection/comment-page-1/#comment-7668)**

Olá , Cara sou meio leigo em php e estou com um problema de Injection no PHP onde vc colocar ‘or’ 1 no lugar do login e senha a pessoa tem acesso total a tudo …. se vc poder me ajudar com esse script eu fico muito agradecido … Obrigado

```
<?php
if ($acao == "validar_acesso"){
$sql="select * from usuario where (login='$edt_login') and (senha='$edt_senha')";
$exe=mysql_query($sql);
$num=mysql_num_rows($exe);
if ($num <=0){
echo "window.alert(‘ERRO. Acesso nao permitido’);";
echo “window.location=’index.php’;”;
} else {
$tmp=mysql_fetch_array($exe);
$usuario_situacao=$tmp[situacao];
if ($usuario_situacao == 0){
echo “window.alert(‘ERRO. Login Inativo. Favor contactar o Administrador’);”;
echo “window.location=’index.php’;”;
```

```
}
if ($usuario_situacao == 9){
echo "window.alert('ATENCAO. Login Bloqueado. Favor contactar o Administrador');";
echo "window.location='index.php';";
}
if ($usuario_situacao == 1){
$_SESSION['acesso_idusuario']=$tmp[idusuario];
$_SESSION['acesso_idempresa']=$tmp[idempresa];
```

Reply (/seguranca-em-sistemas-de-login-protecao-contra-sql-injection/?replytocom=7668#respond)

- ***Beraldo (http://rberaldo.com.br)***

07/04/2012 at 15:48 | **Permalink (http://rberaldo.com.br/seguranca-em-sistemas-de-login-protecao-contra-sql-injection/comment-page-1/#comment-7669)**

O meu post aborda justamente isso. O que você não entendeu?

Reply (/seguranca-em-sistemas-de-login-protecao-contra-sql-injection/?replytocom=7669#respond)

1. **Andrey Knupp » Sistema De Login & Senha Usando MD5 & MySQLi – jQuery ~ Ajax (http://www.mbiosinformatica.com.br/blog/sistema-de-login-senha-usando-md5-mysqli-jquery-ajax/)** on **24/02/2011 at 16:02 (http://rberaldo.com.br/seguranca-em-sistemas-de-login-protecao-contra-sql-injection/comment-page-1/#comment-3571)**

Deixe um Comentário

O seu endereço de email não será publicado Campos obrigatórios são marcados *

Nome *

Email *

Site

Comentário

Você pode usar estas tags e atributos de HTML: `<a href="" title=""> <abbr title=""> <acronym title=""> <b> <blockquote cite=""> <cite> <code> <del datetime=""> <em> <i> <q cite=""> <strike> <strong> <pre lang="" line="" escaped="" highlight="">`

Publicar comentário

Publicidade

Categorias

Selecionar categoria

Arquivo

Selecionar o mês

Divulgue o Blog do Beraldo

(http://rberaldo.com.br)

```
<a href="http://rberaldo.com.br"
title="Link para o Blog do
Beraldo"><img
src="http://rberaldo.com.br/img/log
o_blog.jpg" alt="Link para o Blog
```

Contribua

Gostou do conteúdo e deseja contribuir para a manutenção e para o crescimento do Blog do Beraldo? Se você deseja ajudar a equipe, pode fazer uma doação pelo PagSeguro.

Parceiros

(http://ogordo.com) (http://www.silasjr.com) (http://piadasnerds.com)

(http://www.boipassarim.com/)

Links

- Dolce Arte – Doces Finos Artesanais (http://www.dolcearte.com.br)
- Gerencie Você Mesmo (http://gerencievocemesmo.com.br/site/)

- **Música Viciante (http://www.lalalalalalalalalalalalalalalala.com)**

Blog do Beraldo (http://rberaldo.com.br)
Sempre Berando o Conhecimento