

# Re: how to decrypt an encrypted stored proc in 2005

---

*Source:* <http://www.derkeiler.com/Newsgroups/microsoft.public.sqlserver.security/2006-08/msg00307.html>

---

- *From:* DallasBlue <DallasBlue@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
  - *Date:* Thu, 24 Aug 2006 17:28:01 -0700
- 

awesome. Thank you so very much. and thanks a million :-)

sysobjvalues .. that did the trick , UUUUUUhhhhhh I did not find anywhere.

"Dr. Network" wrote:

SQL 2005 still obfuscates object encryption in a similar manner to SQL 2000. It is simply an XOR of the encrypted bytes against a known sample to dump out the real unencrypted bytes. The trick is getting to the the encrypted bytes. In SQL 2005, instead of querying the SYSCOMMENTS view to get the binary bytes to decrypt, you need to run the following query from the dedicated admin connection (DAC) to get the bytes to decrypt:

```
SELECT imageval FROM sys.sysobjvalues WHERE id = object_id(@procedure) AND  
valclass = 1 AND subobjid = 1
```

where @procedure is the name of the object that you want to decrypt. The table sys.sysobjvalues is one of those new system tables that is normally hidden. It is actually easier in SQL 2005 because you don't have to string together 4000 character blocks from SYSCOMMENTS records like you had to do in SQL 2000. Code to do the XOR on SQL 2000 is well documented on the web. I'm including my work below.

This is code that I quickly rolled together from my old SQL 2000 decrypt code. It isn't perfect yet but should get you down the line. It only does procedures. It actually wraps the ALTER PROCEDURE in a rolled-back transaction so it shouldn't be too problematic. Then again my local copy of SQL Server 2005 only starts with the -f flag this morning ;-). Have fun.

Chuck

```
CREATE PROCEDURE dbo.sp__procedure$decrypt  
(@procedure sysname = NULL, @revfl int = 1)  
AS  
SET NOCOUNT ON
```

Re: how to decrypt an encrypted stored proc in 2005

```
IF @revfl = 1
BEGIN
PRINT 'CAUTION: THIS PROCEDURE DELETES AND REBUILDS THE ORIGINAL
STORED
PROCEDURE.'
PRINT ' MAKE A BACKUP OF YOUR DATABASE BEFORE RUNNING THIS
PROCEDURE.'
PRINT ' IDEALLY, THIS PROCEDURE SHOULD BE RUN ON A NON-PRODUCTION
COPY
OF THE PROCEDURE.'
PRINT ' To run the procedure, change the @revfl parameter to 0'
RETURN 0
END
```

```
DECLARE @intProcSpace bigint, @t bigint, @maxColID smallint, @intEncrypted
tinyint, @procNameLength int
select @maxColID = max(colid), @intEncrypted = encrypted FROM
sys.sysobjvalues WHERE id = object_id(@procedure)
GROUP BY encrypted
```

```
--select @maxColID as 'Rows in sys.sysobjvalues'
select @procNameLength = datalength(@procedure) + 29
```

```
DECLARE @real_01 nvarchar(max)
```

```
DECLARE @fake_01 nvarchar(max)
```

```
DECLARE @fake_encrypt_01 nvarchar(max)
```

```
DECLARE @real_decrypt_01 nvarchar(max), @real_decrypt_01a nvarchar(max)
```

```
select @real_decrypt_01a = "
```

```
-- extract the encrypted imageval rows from sys.sysobjvalues
SET @real_01=(SELECT imageval FROM sys.sysobjvalues WHERE id =
object_id(@procedure) and valclass = 1 and subobjid = 1 )
```

```
-- create this table for later use
create table #output ( [ident] [int] IDENTITY (1, 1) NOT NULL ,
[real_decrypt] NVARCHAR(MAX) )
```

```
-- We'll begin the transaction and roll it back later
```

```
BEGIN TRAN
```

```
-- alter the original procedure, replacing with dashes
```

```
SET @fake_01='ALTER PROCEDURE '+ @procedure +' WITH ENCRYPTION AS
'+REPLICATE('-', 40003 - @procNameLength)
```

```
EXECUTE (@fake_01)
```

```
-- extract the encrypted fake imageval rows from sys.sysobjvalues
```

```
SET @fake_encrypt_01=(SELECT imageval FROM sys.sysobjvalues WHERE id =
```

## Re: how to decrypt an encrypted stored proc in 2005

object\_id(@procedure) and valclass = 1 and subobjid = 1)

```
SET @fake_01='CREATE PROCEDURE '+ @procedure +' WITH ENCRYPTION AS  
'+REPLICATE('-', 40003 - @procNameLength)
```

```
--start counter
```

```
SET @intProcSpace=1
```

```
--fill temporary variable with with a filler character
```

```
SET @real_decrypt_01 = replicate(N'A', (datalength(@real_01) /2 ))
```

```
--loop through each of the variables sets of variables, building the real  
variable
```

```
--one byte at a time.
```

```
SET @intProcSpace=1
```

```
-- Go through each @real_xx variable and decrypt it, as necessary
```

```
WHILE @intProcSpace<=(datalength(@real_01)/2)
```

```
BEGIN
```

```
--xor real & fake & fake encrypted
```

```
SET @real_decrypt_01 = stuff(@real_decrypt_01, @intProcSpace, 1,
```

```
NCHAR(UNICODE(substring(@real_01, @intProcSpace, 1)) ^
```

```
(UNICODE(substring(@fake_01, @intProcSpace, 1)) ^
```

```
UNICODE(substring(@fake_encrypt_01, @intProcSpace, 1))))
```

```
SET @intProcSpace=@intProcSpace+1
```

```
END
```

```
-- Load the variables into #output for handling by sp_helptext logic
```

```
insert #output (real_decrypt) select @real_decrypt_01
```

```
-- select real_decrypt AS '#output chek' from #output -- Testing
```

```
-----  
-- Beginning of extract from sp_helptext  
-----
```

```
declare @dbname sysname
```

```
,@BlankSpaceAdded int
```

```
,@BasePos int
```

```
,@CurrentPos int
```

```
,@TextLength int
```

```
,@LineId int
```

```
,@AddOnLen int
```

```
,@LFCHR int --lengths of line feed carriage return
```

```
,@DefinedLength int
```

```
,@SyscomText nvarchar(4000)
```

```
,@Line nvarchar(255)
```

```
Select @DefinedLength = 255
```

```
SELECT @BlankSpaceAdded = 0 --Keeps track of blank spaces at end of lines.
```

```
Note Len function ignores trailing blank spaces
```

Re: how to decrypt an encrypted stored proc in 2005

```
CREATE TABLE #CommentText
(LineId int
,Text nvarchar(255) collate database_default)

-- use #output instead of sys.sysobjvalues
DECLARE ms_crs_syscom CURSOR LOCAL
FOR SELECT real_decrypt from #output
ORDER BY ident
FOR READ ONLY

-- Else get the text.

SELECT @LFCR = 2
SELECT @LineId = 1

OPEN ms_crs_syscom

FETCH NEXT FROM ms_crs_syscom into @SyscomText

WHILE @@fetch_status >= 0
BEGIN

SELECT @BasePos = 1
SELECT @CurrentPos = 1
SELECT @TextLength = LEN(@SyscomText)

WHILE @CurrentPos != 0
BEGIN
--Looking for end of line followed by carriage return
SELECT @CurrentPos = CHARINDEX(char(13)+char(10), @SyscomText,
@BasePos)

--If carriage return found
IF @CurrentPos != 0
BEGIN
--If new value for @Lines length will be > then the
--set length then insert current contents of @line
--and proceed.

While (isnull(LEN(@Line),0) + @BlankSpaceAdded +
@CurrentPos-@BasePos + @LFCR) > @DefinedLength
BEGIN
SELECT @AddOnLen = @DefinedLength-(isnull(LEN(@Line),0) +
@BlankSpaceAdded)
INSERT #CommentText VALUES
( @LineId,
isnull(@Line, N'') + isnull(SUBSTRING(@SyscomText,
@BasePos, @AddOnLen), N''))
```

Re: how to decrypt an encrypted stored proc in 2005

```
SELECT @Line = NULL, @LineId = @LineId + 1,
@BasePos = @BasePos + @AddOnLen, @BlankSpaceAdded = 0
END
SELECT @Line = isnull(@Line, N") +
isnull(SUBSTRING(@SyscomText, @BasePos, @CurrentPos-@BasePos + @LFCR), N")
SELECT @BasePos = @CurrentPos+2
INSERT #CommentText VALUES( @LineId, @Line )
SELECT @LineId = @LineId + 1
SELECT @Line = NULL
END
ELSE
--else carriage return not found
BEGIN
IF @BasePos <= @TextLength
BEGIN
--If new value for @Lines length will be > then the
--defined length
--
While (isnull(LEN(@Line),0) + @BlankSpaceAdded +
@TextLength-@BasePos+1 ) > @DefinedLength
BEGIN
SELECT @AddOnLen = @DefinedLength -
(isnull(LEN(@Line),0) + @BlankSpaceAdded)
INSERT #CommentText VALUES
( @LineId,
isnull(@Line, N") + isnull(SUBSTRING(@SyscomText,
@BasePos, @AddOnLen), N"))
SELECT @Line = NULL, @LineId = @LineId + 1,
@BasePos = @BasePos + @AddOnLen, @BlankSpaceAdded =
0
END
SELECT @Line = isnull(@Line, N") +
isnull(SUBSTRING(@SyscomText, @BasePos, @TextLength-@BasePos+1 ), N")
if LEN(@Line) < @DefinedLength and charindex(' ',
@SyscomText, @TextLength+1 ) > 0
BEGIN
SELECT @Line = @Line + ' ', @BlankSpaceAdded = 1
END
END
END
END

FETCH NEXT FROM ms_crs_syscom into @SyscomText
END

IF @Line is NOT NULL
INSERT #CommentText VALUES( @LineId, @Line )

select Text from #CommentText order by LineId

CLOSE ms_crs_syscom
```

Re: how to decrypt an encrypted stored proc in 2005

```
DEALLOCATE ms_crs_syscom
```

```
DROP TABLE #CommentText
```

```
-----  
-- End of extract from sp_helptext  
-----
```

```
-- Drop the procedure that was setup with dashes and rebuild it with the  
good stuff
```

```
-- Version 1.1 mod; makes rebuilding hte proc unnecessary
```

```
ROLLBACK TRAN
```

```
DROP TABLE #output
```

```
GO
```

```
SET QUOTED_IDENTIFIER OFF
```

```
GO
```

```
SET ANSI_NULLS ON
```

```
GO
```

"Puneet" <Puneet@xxxxxxxxxxxxxxxxxxxxxxxxxxxx> wrote in message  
news:CB5448F0-0CD4-4C61-BF66-C48CE0156414@xxxxxxxxxxxxxxxxxxxx

There's a software, which claims it does it, you can download it from  
<http://www.elitude.net/>

However, not sure, if it's going to mess up your system even further or  
how

it will do it. and seems like you got to purchase it for full version.

Cheers,

Puneet.

"DallasBlue" wrote:

I actually compiled my proc with encryption and now i dont  
have src code

so I

was trying to get it back... there sure must be a way as I saw  
some tools

doing it.

Any thots, hints , pointers will be appreciated.

thanks,

Re: how to decrypt an encrypted stored proc in 2005

GA

"Sue Hoegemeier" wrote:

That won't show it either. You can't see the definition of encrypted stored procedures – that would kind of defeat the purpose of encrypting them in the first place. You need to have the original source code of stored procedures created with the encryption option.

–Sue

On Wed, 23 Aug 2006 11:41:02 –0700,  
DallasBlue  
<DallasBlue@xxxxxxxxxxxxxxxxxxxxxxxxxxxx>  
wrote:

Hi,  
can I get back the source code of the stored procedure which is created with 'with encryption' option.  
sys.sql\_modules/sys.all\_sql\_modules/syscomments dosent show it even after