

Re: There seems to be a memory leak in srv_paraminfo

Source:

<http://www.tech-archive.net/Archive/SQL-Server/microsoft.public.sqlserver.programming/2004-08/0262.html>

From: Gert E.R. Drapers (*GertD_at_SQLDev.Net*)

Date: 08/02/04

Date: Mon, 2 Aug 2004 23:09:58 +0200

How did you measure this? Which library version did you compile with? The one that comes with SQL Server 2000, the one that comes with the latest service pack?

GertD@SQLDev.Net

Please reply only to the newsgroups.

This posting is provided "AS IS" with no warranties, and confers no rights.

You assume all risk for your use.

Copyright © SQLDev.Net 1991-2004 All rights reserved.

"Daniel" <softwareengineer98037@yahoo.com> wrote in message
news:efOQNVqdEHA.2752@TK2MSFTNGP12.phx.gbl...

>I called my own extended stored procedure 1000 times to make sure there was
> no memory leak but the sqlservr.exe memory usage went up so i thought
> there
> might be a memory leak. I commented out all the code in the extended
> stored
> procedure but the call to srv_paraminfo and it seems that srv_paraminfo,
> when called w/ null to get parameter lenght etc. leaks some memory.
> perhaps
> something in the internals of srv_paraminfo. If an extended stored
> procedure
> is called every 10 milliseconds for 24 hours and all it does is use
> srv_paraminfo to get the size of the incoming parameter but never takes
> the
> parameter, the sqlservr.exe memory usage will increase by about 30
> megabytes
> a day. This worries me, will i have to restart sql server 2000 now and then
> if i use srv_paraminfo in a stored procedure that is called frequently? I
> thought maybe it was just somethign wrong w/ MY code, so i ran the sane
> test
> w/ a sample extended stored procedure that comes on the SQL Server 2000 CD
> called: xp_srv_paraminfo_sample and the memory climbed at the same rate.
> It

microsoft.public.sqlserver.programming: Re: There seems to be a memory leak in srv_paraminfo

> doesnt seem to matter what i send into an extended stored procedure or how
> large it is, just seems that when ever srv_paraminfo is used to get the
> size
> of data, it leaks memory.
>
> Here is the sample on the SQL server 2000 CD that seems to leak about 30
> megabytes a day if run every 10 milliseconds for 24 hours:
>
> http://msdn.microsoft.com/library/default.asp?url=/library/en-us/odssql/ods_6_sam_01_4yel.asp
>
> Here is the code in the sample extended stored procedure on the SQL Server
> 2000 CD:
>
> /*****
> Copyright (c) 2000, Microsoft Corporation
> All Rights Reserved.
> *****/
> // This is an example of an extended procedure DLL built with Open Data
> // Services. The function within the DLL can be invoked by using the
> // extended stored procedures support in SQL Server.
> //
> // For more information on Open Data Services refer to the Microsoft Open
> // Data Services Programmer's Reference.
> //
> // xp_srv_paraminfo_sample accepts parameters from a Transact-SQL
> // statement.
> // It analyzes each parameter then posts a result set to the client
> // containing information about each parameter and the parameter's value.
> //
> // The Transact-SQL script xp_param.sql installs and exercises the
> // extended
> // stored procedure.
> #include <stdlib.h>
> #include <stdio.h>
> #include <string.h>
> #include <ctype.h>
> #include <windows.h>
> #include <srv.h>
> #include <time.h>
>
> // Macros -- return codes
> #define XP_NOERROR 0
> #define XP_ERROR 1
>
> #define MAX_SERVER_ERROR 20000
> #define XP_PARAM_ERROR MAX_SERVER_ERROR+1
>
> void printError (SRV_PROC *pSrvProc, CHAR* szErrorMsg);
> void printUsage (SRV_PROC *pSrvProc);
>
> // It is highly recommended that all Microsoft® SQL Server (7.0

Re: There seems to be a memory leak in srv_paraminfo

microsoft.public.sqlserver.programming: Re: There seems to be a memory leak in srv_paraminfo

```
> // and greater) extended stored procedure DLLs implement and export
> // __GetXpVersion. For more information see SQL Server
> // Books Online
> ULONG __GetXpVersion()
>
> {
> return ODS_VERSION;
> }
>
>
> SRVRETCODE xp_srv_paraminfo_sample
> (
> SRV_PROC* pSrvProc
> )
> {
> char acHeader[24];
> BYTE bType;
> long cbMaxLen;
> long cbActualLen;
> PBYTE* ppbData;
> BOOL fNull;
> int nParams;
> int nParam;
> SRVRETCODE rc = XP_NOERROR;
>
> #ifdef _DEBUG
> // In a debug build, look up the data type name for assistance.
> DBCHAR* pdbcDataType;
> int cbDataType;
> #endif
>
> // Count up the number of input parameters
> nParams = srv_rpcparams(pSrvProc);
> if (nParams == -1)
> {
> printUsage (pSrvProc);
> return (XP_ERROR);
> }
>
> // Build an array of data pointers for the input parameters.
> ppbData = (PBYTE*) malloc(nParams * sizeof(PBYTE));
> if (ppbData == NULL)
> {
> printError (pSrvProc, "Memory allocation error.");
> return (XP_ERROR);
> }
> memset(ppbData, (int) NULL, nParams * sizeof(PBYTE));
>
> for (nParam = 0; nParam < nParams; nParam++)
> {
> // Use srv_paraminfo to get data type and length information. Ask
```

Re: There seems to be a memory leak in srv_paraminfo

microsoft.public.sqlserver.programming: Re: There seems to be a memory leak in srv_paraminfo

```
> // for the data later using a non-null ppbData argument in a second
> // srv_paraminfo call.
> if (srv_paraminfo(pSrvProc, nParam+1, &bType, &cbMaxLen,
> &cbActualLen,
> NULL, &fNull) == FAIL)
> {
> rc = XP_ERROR;
> break;
> }
>
> // Describe the paramter. The column header indicates whether
> // the parameter is input or output.
> sprintf(acHeader, "Parameter %d: %s", nParam+1,
> ((srv_paramstatus(pSrvProc, nParam+1) & SRV_PARAMRETURN) ?
> "Output" : "Input"));
> srv_describe(pSrvProc, nParam+1, acHeader, SRV_NULLTERM, bType,
> cbActualLen, bType, cbActualLen, NULL);
>
> // If there's data, then dynamically allocate memory and retrieve
> it.
> if (fNull == 0)
> {
> ppbData[nParam] = malloc(cbActualLen);
> if (ppbData[nParam] == NULL)
> {
> rc = XP_ERROR;
> break;
> }
>
> if (srv_paraminfo(pSrvProc, nParam+1, &bType, &cbMaxLen,
> &cbActualLen, ppbData[nParam], &fNull) == FAIL)
> {
> rc = XP_ERROR;
> break;
> }
>
> #ifdef _DEBUG
> // A debugging aid. Get the name of the data type of the parameter.
> pdbcDataType = srv_symbol(SRV_DATATYPE, (int) bType, &cbDataType);
> #endif
>
> // Set the column's data;
> if (srv_setcoldata(pSrvProc, nParam+1, ppbData[nParam]) == FAIL)
> {
> rc = XP_ERROR;
> break;
> }
> }
>
> // Send the row to the client.
```

Re: There seems to be a memory leak in srv_paraminfo

microsoft.public.sqlserver.programming: Re: There seems to be a memory leak in srv_paraminfo

```
> if (rc == XP_NOERROR)
> {
> if (srv_sendrow(pSrvProc) == FAIL)
> {
> rc = XP_ERROR;
> }
> }
>
> // Free dynamically allocated memory.
> for (nParam = 0; nParam < nParams; nParam++)
> {
> if (ppbData[nParam])
> {
> free(ppbData[nParam]);
> }
> }
> free(ppbData);
>
> // Indicate that we're done.
> if (rc == XP_NOERROR)
> {
> srv_senddone(pSrvProc, (SRV_DONE_COUNT | SRV_DONE_MORE), 0, 1);
> }
> else if (rc == XP_ERROR)
> {
> printError (pSrvProc, "XP encountered an error.");
> }
>
> return (rc);
> }
>
> // send szErrorMsg to client
> void printError (SRV_PROC *pSrvProc, CHAR* szErrorMsg)
> {
> srv_sendmsg(pSrvProc, SRV_MSG_ERROR, XP_ERROR, SRV_INFO, 1,
> NULL, 0, (DBUSMALLINT) __LINE__,
> szErrorMsg,
> SRV_NULLTERM);
>
> srv_senddone(pSrvProc, (SRV_DONE_ERROR | SRV_DONE_MORE), 0, 0);
> }
>
> // send XP usage info to client
> void printUsage (SRV_PROC *pSrvProc)
> {
> // usage: exec xp_srv_paraminfo_sample [@param1, [@param2 output],...]
>
> srv_sendmsg(pSrvProc, SRV_MSG_ERROR, XP_PARAM_ERROR, SRV_INFO, 1,
> NULL, 0, (DBUSMALLINT) __LINE__,
> "usage: exec xp_srv_paraminfo_sample [@param1, [@param2
```

Re: There seems to be a memory leak in srv_paraminfo

microsoft.public.sqlserver.programming: Re: There seems to be a memory leak in srv_paraminfo

```
> output],...]";  
> SRV_NULLTERM);  
> srv_senddone(pSrvProc, (SRV_DONE_ERROR / SRV_DONE_MORE), 0, 0);  
>  
> }  
>  
>
```