

IBM Lotus Domino-based IBM Sametime server for IBM i: Diagnosing and troubleshooting common problems

Le Chang
Chen Yun

November 08, 2012

This article summarizes the common problems for IBM® Lotus® Domino®-based IBM Sametime® servers on IBM i and introduces some of the methods that can help a Sametime Server Administrator to diagnose and troubleshoot the problems. It also suggests methods for gathering useful information that might require IBM support to address the problems.

Overview

IBM Sametime is an instant communications application, usually deployed in medium- or large-sized enterprises. It supports multiple operating system platforms, including IBM i. Because of its stability and integration, many customers prefer IBM i as their Sametime server.

In this article, we will talk about how to collect the most useful diagnostic information which can improve the efficiency of working with an IBM support engineer. We will also introduce some common problems and show how to diagnose and troubleshoot them on IBM i. This helps Sametime administrators to solve some simple problems and quickly recover their Sametime server, instead of seeking support from an IBM engineer. This article provides information about the following topics:

- Enabling diagnostic files and gathering the traces and logs.
- How to analyze diagnostic files
- Typical symptoms and how to resolve common problems.

Enabling diagnostic files and gathering the traces and logs

Note: All of the files mentioned are in the server data directory.

Traces and logs are important for addressing or debugging a problem when the software does not work as expected. Lotus Sametime also has its own diagnostic mechanism. If we can enable the diagnostic trace when a problem occurs, combined with Lotus Domino diagnostic traces and IBM i system job logs, then it becomes more flexible and easier to address the problems.

1. Enabling Sametime diagnostic files

Make sure that the console log is enabled in the *Notes.ini* file

```
CONSOLE_LOG_ENABLE=1
```

This is already enabled by default, but you need to check that it is not disabled. Value 0 means disabled and 1 means enabled.

- Enable community server diagnostics.
For community server jobs, the easiest way to turn on all diagnostics is by setting `VP_TRACE_ALL` to 1, in the *Sametime.ini* file.

```
[Debug]
VP_TRACE_ALL=1
```

Refer to *How to enable Sametime Component debugging* to find more information about enabling each component trace individually.

This creates trace files in the *Trace* directory. You will get a different trace file for each community job. Every trace file will create a new trace file after reaching 2 megabytes (MB), so that each individual file does not get too large to manage.

- Enable meeting server diagnostics.

Meeting server diagnostics are enabled through the *MeetingServer.ini* file. This is the relevant section:

```
[SOFTWARE\Lotus\Sametime\MeetingServer\Diagnostics]
LogPrintLevel=4
ModuleLogEnabled=0
SummaryLogEnabled=1
SummaryLogFilename=summary.diag
```

`LogPrintLevel`: 4 is the default value. Generally, when tracing a problem, set it to 16.

`ModuleLogEnabled`: This value determines if each meeting server job will create its own trace file or not. This is **off** by default. Set it to 1.

`SummaryLogEnabled`: This value determines if all the meeting server jobs will write their traces out to the same file or to. This is **on** by default. You usually need not change the value.

`SummaryLogFilename`: This is the name of the summary trace file. You usually need not change the value.

Note that you can have both **ModuleLogEnabled** and **SummaryLogEnabled** turned on.

- Enable servlets diagnostics

The HTTP servlets diagnostics are controlled by the *SametimeDiagnostics.properties* file. Generally, to turn these on, you just edit the following last few lines of this file:

```
# The default VM File level is set to 30, so all Warnings,
#Error and Criticals are shown,
# to change the level of prints that are shown to VM File,
#uncomment the following
# line and enter the new level:
# Levels: Off(0), Critical(10), Error(20), Warning(30), Notify(40), Trace(50)
com=20
com.lotus=30
com.ibm=30
```

Change these log levels to 50.

These trace files are recorded in the *Trace* directory in the *servlets_<timestamp>.java.diag* file.

2. Gathering the traces and logs.

- Using STDIAGZIP to collect Sametime diagnostics traces
STDIAGZIP is a tool for collecting the Sametime server trace and log files.
Run the following command:

```
CALL QSAMETIME/STDIAGZIP ServerName
```

This command collects all the information in the *Trace* directory, all the Network Shared Disks (NSDs), and some configuration files, and will finally put them into a zip file. It also generates an HTML file filled with system information that might help you diagnose problems. This HTML file shows the Transmission Control Protocol (TCP) configuration, program temporary fixes (PTFs) installed, and other information. The file name has the following format: *ST_Snapshot_20101029_04.47.56.html*

The information that the **STDIAGZIP** command collects is actually determined by the *stdiagzip.properties* file.

After you run the command, the zip file is created in the *Trace* directory.

- Other logs that might be needed due to different problems.
 - If the Sametime server encounters a Java™ virtual machine (JVM) problem, such as java.lang.NullPointerException, java.lang.OutOfMemoryError and so on, the error message might or might not be displayed on the console or in any log files, but some JVM CoreDump files will be generated in the server data directory, and those logs also need to be collected. They will look similar to the following message:

```
core.201111201.212143.7279.0001.dmp
javacore.20111129.194902.5824.0003.txt
```

- For other specific problems, the following files might also need to be gathered. All are in the server data directory.

```
log.nsf
da.nsf
names.nsf
```

- Job logs and spooling files
The IBM i system uses job logs to record information for every job. The job log records the time when the job started, the time when the job ended, the commands that the job called, error information (if any), and so on. For certain types of errors and exceptions, you might need to check the job logs to address the problems. There are many ways to check job logs.

- Using the green screen
 - Use the **WRKJOBLOG** or **DSPJOBLOG** command to see the job. log file.
 - Use the **WRKSPLF** or **DSPSPLF** command.
If the job has already stopped, and the job log has not been printed, use the **WRKSPLF** or **DSPSPLF** command to view the spooling files. For example, **WRKSPLF SELECT (QNOTES)**
 - Using IBM iSeries® Navigator
Click **iSeries Navigator -> My connections -> Connection -> Work Management -> Active Jobs directory** or **Server Jobs directory**. Then, right-click one of the job's icon and then click **Job Log**.

Analyzing diagnostic files

After collecting the traces and logs, we can start analyzing them. But, as so many files are already gathered, what are these traces for? Which one should we start with? Understanding how to read them is a problem. The section will not tell you more about analyzing a detailed technical problem, but provides information about how the logs are classified, and the general steps to analyze when we collected these traces.

Logs classification

- Logs under the *IBM_TECHNICAL_SUPPORT* directory:
Logs under this directory are generated by the Domino server. The *console.log* file and the *NSD* file would contain some information about the Sametime server. When you debug a hang or deadlock issue in the Sametime jobs, the Semaphore debug files will also be generated to this directory. For more Domino debug parameters, refer to *Turning on semaphore debugging parameters in notes.ini for Domino*.
- Logs under the *Trace* directory:
Logs under this directory are generated by the Sametime jobs. These logs can be classified as:
 - Summarized logs, such as *Sametime.log*, *summary_*.diag*, *stlinks.txt* and so on.
 - Configuration related information, such as *communityConfig.txt*, *ST_Snapshot_*.html* and so on.
 - Each module's log.
 - Some statistics logs, such as *OfflineMessages_Statistics_*.csv* and so on.
- Useful files under the server data directory:
The most useful files under this directory are *Notes.ini*, *Sametime.ini*, *Meetingserver.ini*, *StConfig.nsf*, and *StLog.nsf*.
If there are Java core dump logs generated, they should be under this directory also.
 - Job logs, spooling files, and others:
Usually these files are not collected by default, but are gathered for specific problems. Refer to *Spooling files and job logs*.

How to check the logs and trace?

Usually, we follow this sequence to check these log files.

1. Go through the *console.log* and *Sametime.log* files to identify the error type, for example, a startup issue, shutdown issue, a crash happened, a hang occurred, or a JVM exception generated, and so on.
2. Check the configuration related logs, for example *CommunityConfig.txt*, *StConfig.nsf*, *ST_Snapshot_*.html* and use the check list of Chapter 2 to identify if there is a configuration issue. You may also need to check the *StLinks.txt* file, unified with the *ST_Snapshot_*.html* file to see if there is a connection issue.
3. Check if there are any meaningful NSDs generated under *IBM_TECHNICAL_SUPPORT*, this can help you find the issued job and the call stack.
4. Address a specific issue in the corresponding module according to that module or job's trace.

5. Check other logs such as job logs, spooling files, Java core dump files, and so on, if needed.

Addressing a problem shown in the logs

In general, we use keywords and timestamps, combined with technical notes published by the IBM Sametime team to address and solve most of the common problems. These problems are usually caused by an incorrect configuration, unfulfilled system requirements, or a network issue.

By using the keyword in the log file, we can search in the technical notes to see if it is a known problem.

You can use the timestamp to identify the problem in the different trace files. For example, if you found a suspicious error message in *Summary_*.diag* at 12:01:55, you can check other related logs around that time to see what happened.

In these logs, there might be some error or exception messages, but many of these can be ignored, as explained in the following points.

- *Summary_*.diag*

```
STMTGSRV/QNOTES/212374 | stnsc |T: 0016| 12/31/10 14:49:10.5007
|[ Error]| getaddrinfo()
failed Error: 6, host=IPV6-LOCALHOST port=9092
STMTGSRV/QNOTES/212374 | stnsc |T: 0016| 12/31/10 14:49:23.0075
|[ Error]| getaddrinfo()
failed Error: 6, host=IPV6-LOCALHOST port=9092
```

When the Sametime meeting server jobs start up, the code that determines if we are using IPv4 or IPv6 (or both) does many calls to `getaddrinfo` on the known addresses, such as the system host name and `localhost` (IPv4 and IPv6). If IPv6 is not set up, then this error is normal.

```
IPv4 Enabled = TRUE
IPv6 Enabled = FALSE
Fallback Enabled = FALSE
Local Host Test = PASS
Any Address Test = PASS

Hostname: LP10UT21.RCHLAND.IBM.COM
Hostname Address Test = PASS
Address Type: IPv4: 1 IPv6: 0 Length: 16 IP: 9.5.36.31
```

This part in *summary_*.diag* can be a bit confusing, as the host name listed here will always be the SYSTEM host name and not the SERVER host name. So, do not assume that there is a configuration problem somewhere.

- *servlets_*.java.diag*

```
12/31/10 14:48:17.622 [Crit ] [ Thread-13 ]
[ com.lotus.sametime.statistics.ParticipantThreshold ]
Initialize() failed, Event System may not be running: java.lang.NullPointerException
12/31/10 14:48:27.300 [Warn ] [ Thread-10 ]
[ com.lotus.sametime.statistics.VPPublisher ]
Initialize() failed, Event System may not be running: Try 2 of 10
12/31/10 14:48:27.529 [Crit ] [ Thread-11 ]
[ com.lotus.sametime.statistics.StatisticsClient ] IOException
java.net.SocketException: Protocol family unavailable
at java.net.PlainSocketImpl.doConnect(PlainSocketImpl.java:352)
```

```

at java.net.PlainSocketImpl.connectToAddress(PlainSocketImpl.java:214)
at java.net.PlainSocketImpl.connect(PlainSocketImpl.java:201)
at java.net.SocksSocketImpl.connect(SocksSocketImpl.java:378)
at java.net.Socket.connect(Socket.java:528)
at java.net.Socket.connect(Socket.java:477)
at java.net.Socket.<init>(Socket.java:374)
at java.net.Socket.<init>(Socket.java:217)
at com.lotus.sametime.event.DBEventRequestServiceTCPProxy.getSocket(Unknown Source)
at com.lotus.sametime.event.DBEventRequestServiceTCPProxy.getProxySocket(Unknown Source)
at com.lotus.sametime.event.DBEventRequestServiceTCPProxy.initialize(Unknown Source)
at com.lotus.sametime.event.DBEventRequestServiceTCPProxy.<init>(Unknown Source)
at com.lotus.sametime.statistics.StatisticsClient.initializeEventController(Unknown Source)
at com.lotus.sametime.statistics.StatisticsServlet.run(Unknown Source)
at java.lang.Thread.run(Thread.java:736)

```

This error is also normal, it is just the servlet job trying to communicate with the Sametime event server job. The servlets start up before Sametime, so they will generate this error, until the STEVETSVR job is started up.

```

11/11/10 10:46:20.864 [Warn ] [ main ]
[ com.lotus.sametime.admin.configuration.NotesDataSource ]
Directory assistance database was referenced but not found.
11/11/10 10:46:20.987 [Error ] [ main ]
[ com.lotus.sametime.configuration.PropertyFileXMLDataSource ]
Moving on, but IO Exception loading resource:
domino/html/sametime/javaconnect/stconnectver.txt
java.io.IOException: Unable to load resource,
resourceName = [domino/html/sametime/javaconnect/stconnectver.txt]

```

This error message is shown because we did not install the Java Client Connector. It does not affect any Sametime server-side function.

- *gwcontroller_*.diag*

```

11/09/10 10:39:00.618 [Error ] [ main ]
[ com.lotus.sametime.broadcastcontroller.GWGatewayAttributes ]
GWGatewayAttributes: number f
java.lang.NumberFormatException: For input
string: "false"
at java.lang.NumberFormatException.forInputString(NumberFormatExce
ption.java:63)
at java.lang.Integer.parseInt(Integer.java:481)
at java.lang.Integer.parseInt(Integer.java:531)
at com.lotus.sametime.broadcastcontroller.GWGatewayAttributes.getCon
figurationBoolean
(GWGatewayAttributes.java:548)
at com.lotus.sametime.broadcastcontroller.GWGatewayAttributes.<init>
(GWGatewayAttributes.java:286)
at com.lotus.sametime.broadcastcontroller.GWController.loadGatewayList
(GWController.java:253)
at com.lotus.sametime.broadcastcontroller.GWController.process
(GWController.java:135)
at com.lotus.sametime.broadcastcontroller.GWController.main
(GWController.java:81)
11/09/10 10:41:02.657 [Error ] [ TCP Listener Thread ]
[ com.lotus.sametime.broadcastcontroller.GWConferenceAttributes ]
MaterialsNa

```

This is a normal error, caused by a data type in a database not quite matching something that was expected. It does not cause any problems though.

- *eventserver_***_java.diag*

```

11/21/11 14:31:37.261 [Crit ] [ TCP Request Server Thread ]
[ com.lotus.sametime.event.DBEventRequestServiceTCPServer ]
Failed to accept client socket.
java.net.SocketException: Socket closed
at java.net.PlainSocketImpl.socketAccept(Native Method)
at java.net.PlainSocketImpl.accept(PlainSocketImpl.java:457)
at java.net.ServerSocket.implAccept(ServerSocket.java:473)
at java.net.ServerSocket.accept(ServerSocket.java:444)
at com.lotus.sametime.event.DBEventRequestServiceTCPServer.run(DBEventRequest
ServiceTCPServer.java:414)
at java.lang.Thread.run(Thread.java:810)
11/21/11 14:31:37.297 [Crit ] [ main ] [ com.lotus.sametime.event.
DBEventListenerTCP ] shutdown exception
java.net.SocketException: Socket closed
at java.net.SocketOutputStream.socketWrite(SocketOutputStream.java:112)
at java.net.SocketOutputStream.write(SocketOutputStream.java:149)
at java.io.DataOutputStream.writeInt(DataOutputStream.java:210)
at com.lotus.sametime.event.DBEventRequestFactory.writeGUID(DBEventRequestFactory.ja
va:241)
at com.lotus.sametime.event.DBEventRequest.writeObject(DBEventRequest.java:161)
at com.lotus.sametime.event.DBEventListenerTCP.shutdown(DBEventListenerTCP.java:245)
at com.lotus.sametime.event.DBEventRequestServiceTCPServer.shutdown(DB
EventRequestServiceTCPServer.java:261)
at com.lotus.sametime.event.DBEventRequestService.handle
ControllerEvent(DBEventRequestService.java:223)
at com.lotus.sametime.event.DBEventController.fireControllerEvent(DBEventContr
oller.java:1483)
at com.lotus.sametime.event.DBEventController.fireControllerShutdownEvent(DBEventCont
roller.java:1453)
at com.lotus.sametime.event.DBEventController.shutdown(DBEventController.java:125)
at com.lotus.sametime.eventserver.DBEventServer.shutdown(DBEventServer.java:74)
at com.lotus.sametime.eventserver.DBEventServer.shutdownEventServer(DBEventSe
rver.java:128)
at com.lotus.sametime.eventserver.DBEventServer.main(DBEventServer.java:162)

```

This exception is caused by the socket having been already shut down before the EventServer was shut down.

- *Sametime_*.log*

```

W FileTransfer 09/Nov/10, 10:38:24
Virus Scan BB Dll was not found or error occured in mapping API methods
W FileTransfer 09/Nov/10, 10:38:24
Virus Scan BB Dll is not loaded, mode is RELAX

```

This is normal. We do not actually ship a Virus Scan DLL. We leave that for third parties to develop. So they will get this message unless they install a third-party virus scanner.

- WSASend/WSARecv returned SOCKET_ERROR and WSA_IO_PENDING.
These error messages might be found in many log files, and it confuses a lot of customers, because they may think there is a socket error, but actually it is normal. The error messages are:

```

WSASend returned SOCKET_ERROR for sId=6. RequestedBytes=106, SentBytes=0
WSASend returned WSA_IO_PENDING for sId=6. RequestedBytes=106, SentBytes=0

```

On the IBM i platform, IBM i OS does not have WSASend/WSARecv and TCP SOCKET does not implement WSA_IO_PENDING. So, we implemented WSASend and WSARecv by using the IBM i system APIs, QsoStartSend and QsoStartRecv. It is not a real issue, but an artifact.

Spooling files and job logs

You need not be concerned about some *kill* keywords and *Escape* messages in some job logs and spooling files.

That is a normal system behavior, as the spooling file records the start, run, and end information for every job. During shutdown, there are always some Java/Portable Application Solution Environment (PASE) errors in the job log because of some Sametime applications using JNI try to load Java classes when you shut down the server. A *kill* signal will end the JVM. That is the design of the Sametime exit mechanism. It does not affect any function.

You might see the following error job logs and spooling files:

1. Diagnostic message for checking the Java Virtual Machine environment variable.

```
JVAB302 Diagnostic 10 08/17/11 15:41:09.294887 QJVAJVMXIF QSYS *STMT QJVAJVMXIF QSYS *STMT
From module . . . . . : QJVAJVMXIF
From procedure . . . . . : sendMessage__10QjvaJvmXifFPcPviT1
Statement . . . . . : 10
To module . . . . . : QJVAJVMXIF
To procedure . . . . . : check_J9_Envvar__10QjvaJvmXifFv
Statement . . . . . : 178
Thread . . . . . : 000002D0
Message . . . . . : Java Virtual Machine is IBM Technology for Java.
PID(56145)
Cause . . . . . : JAVA_HOME environment variable is
/QOpenSys/QIBM/ProdData/JavaVM/jdk50/32bit
```

2. Escape message when a PASE job ended.

```
CPFB9C6 Escape 40 08/17/11 15:44:12.309941 QP2SHELL2 QSYS *STMT QXJ9VM QJVM50 *STMT
From module . . . . . : QP2SHELL2
From procedure . . . . . : send_message__FPcT1PvUi
Statement . . . . . : 11
To module . . . . . : QXJ9STRTPA
To procedure . . . . . : paseMainThread__13Qxj9StartPaseFPv
Statement . . . . . : 41
Thread . . . . . : 000002D7
Message . . . . . : PASE for i ended for signal 30, error code 0.
Cause . . . . . : The PASE for i program ended because of PASE for i signal
30. Error code 1 indicates a core file was written in the current directory.
The signal may have been produced for an exception message that appears in
the job log. Recovery . . . : Correct any error and then try the request
again. Technical description . . . . . : If a core file was written,
examine it with the PASE for i 'dbx' command. PASE for i commands can be
entered on the command line displayed by calling program QP2TERM in an
interactive job.
```

3. Escape message when a Java job ended.

```
MCH74A5 Escape 40 08/17/11 15:44:12.310578 QXJ9UTLJVM QJVM50 *STMT XJ9UTLJVM QJVM50 *STMT
From module . . . . . : QXJ9UTLJVM
From procedure . . . . . : JvaSendMsg
Statement . . . . . : 16
5770SS1 V7R1M0 100423 Job Log DOMPRD02 08/17/11 15:44:42 Page 2
Job name . . . . . : STUSERINFO User . . : QNOTES Number . . . . : 817575
Job description . . . . . : LSTP01A Library . . . . . : QUSRNOTES
MSGID TYPE SEV DATE TIME FROM PGM LIBRARY INST TO PGM LIBRARY INST
To module . . . . . : QXJ9UTLJVM
To procedure . . . . . : JvaSendMsg
Statement . . . . . : 16
```



```

Thread . . . . : 000002D0
Message . . . . : The Java Virtual Machine has ended.
Cause . . . . : Java Virtual Machine 1 has ended because of reason 1. The
reason codes are defined as follows: 01- A Java program called the
java.lang.System.exit method with a zero status code. 02- A Java program
called the java.lang.System.exit method with a non zero status code of 0.
03- An unexpected system error was detected. 04- A critical Java Virtual
Machine thread has ended and processing cannot continue. Recovery . . . :
If the Java Virtual Machine ended because of reason code 03, an internal
error has occurred. Contact your service representative. Information about
the error was saved in the Licensed Internal Code log.

```

Something you should know about the NSD files

A Sametime C++ job creates an NSD file if it fails from an uncaught exception.

When a C++ child process fails, one of the following two things happens, depending on the type of job:

- If it is a Community Server job, the main Community Server job (STCOMMUNITY) will notice a child process failing and restarts that child. It performs this activity for unlimited number of times. However, if the child process fails again several times within a small window of time, it will stop restarting the child process.
- If the job that failed is a Legacy Meetingserver job, then when the main Meetingserver job (STMTGSRV) realizes that a job has failed, it will have to shut down the entire server. The Meetingserver has no ability to automatically restart jobs. When the Legacy Meetingserver shuts down due to a child process failing, it almost always hits its own exception during shutdown. This will create a second *NSD*. This second *NSD* is nothing to worry about, as it almost always happens, and does not have anything to do with the real problem.

The following example shows how a second *NSD* might appear. Remember, this *NSD* is not the problem you are trying to solve.

```

JOB:887378/QNOTES/STMTGSRVTHREAD:0xf47
_CXX_PEP__Fv0QP0ZPCP2
QP0ZPCP2
Qp0zNewProcess266QP0ZPCPN
QP0ZPCPN
InvokeTargetPgm__FP11qp0z_pcp_cb210
_CXX_PEP__Fv6MAIN
STMTGSRV
main106
OS400DominoCancelHandler1STUB
LIBNOTES
OS400DominoCancelHandler5BREAK
fatal_error34
OSFaultCleanup1CLEANUP
OSFaultCleanupExt91
OSRunExternalScript40
__system_a2STDLIB_A
LIBCAW
system6QC2SYS
QC2SYS
303
QCMDXC
_C_peg0NSDNSD
main

```

If a Java job fails, you will likely see an exception in its trace file in the *Trace* directory, and possibly, some kind of exceptions or errors in its job logs.

Typical symptoms and the corresponding solution for common problems

1. First, check for the following problems when the Sametime server is not working well. There are several common situations that are responsible for most of the problems with Domino-based Sametime servers running on IBM i. Most (but not all) of them are setup or configuration problems. The *Sametime for IBM i: Most Common Problems* document explains the following problems:
 - System requirements not met or insufficient system resources
 - Java installation is incomplete or out of date
 - An important fix is not installed
 - Not using a fully qualified host name to access server
 - Have not installed or configured optional components for slides or whiteboard area of the meeting room
 - Server document is incorrect
 - Server host name configuration is incorrect
 - Sametime server encountering TCP/IP port conflicts

Refer to *Sametime for IBM i: Most Common Problems* for more detail information at the following link:

2. Sametime server startup failure due to a timeout problem. There are many reasons that can cause Sametime server startup failure (and this is explained in step 3). If the Sametime server fails to start, check the messages in the Domino server console. The symptom is that you cannot find the following messages:

```
<date> 02:23:49 PM Sametime: All services started successfully.  
<date> 02:23:49 PM Sametime: Server startup successful.  
<date> 02:24:25 PM Sametime Server: Running
```

Usually, when a startup failure occurred, you should first follow the instructions mentioned in the step 1 (*Check for the basic problems first when the Sametime server is not working well*) to exclude the possibility of configuration issues. This is important.

The most common causes for IBM i Sametime server startup failures are configuration problems or that the process is taking too long to start, causing an internal timeout to occur. Typically, the problem can be solved by correcting the configuration problems, adding additional resources (disk and memory) to the system, or by updating the Sametime server configuration to increase its internal timeout values and allow more time for the startup to complete.

Symptom

If the server is taking too long to start and the process is timing out, you will typically find a Domino console message stating that **Service failed to initialize in the allowed time**:

```

<date> 09:55:14 AM Sametime: Starting service Event Bridge.
<date> 09:55:15 AM Sametime: Service started successfully.
<date> 09:55:15 AM Sametime: Starting service Logger.
<date> 09:55:15 AM Sametime: Service started successfully.
<date> 09:55:15 AM Sametime: Starting service Token Server.
<date> 09:55:20 AM Sametime: Service started successfully.
<date> 09:55:20 AM Sametime: Starting service T.120 MCU.
<date> 09:58:23 AM Sametime: Service failed to initialize in allowed time.
<date> 09:58:24 AM Sametime: Service failed to start.
<date> 09:58:24 AM Sametime: One or more services failed to start.
<date> 09:58:24 AM Sametime: Server startup failed. Shutting down...
<date> 09:58:24 AM Sametime: Stopping services...

```

Reason

There are several known issues that can result in timeouts during the server startup process. Some of the issues are explained in this section.

- **Unsupported model or insufficient resources (memory, disk space)**

First you should ensure that your system meets the minimum hardware requirements as documented in *Sametime for IBM i: System Requirements*. Startup issues can occur if you are not using one of the supported system models and processor features.

The number of disk drives and the amount of available memory are of particular importance during server startup. We recommend that you have at least four disk drives, 2 GB of free disk space for each Sametime server (1 GB minimum) and at least 1 GB available memory for each Sametime and Domino server on the system. Increasing these resources should result in faster startup.

While you may be able to temporarily work around the resource constraints by increasing the internal timeout values (as described in the following section) or starting the Sametime server when the other system activity is low, you might still be dissatisfied with the length of time needed to start the server. The right solution is to upgrade the server to a supported configuration.

- **Insufficient available resources (resources are allocated to other applications)**

Startup issues might occur if the system meets the minimum resource requirements for Sametime, but the resources are being used by other applications.

You can use the online workload estimator to determine an appropriately sized system to run all of your applications. Refer to *IBM Systems Workload Estimator* at for more information. Again, you may be able to temporarily work around the resource constraints by increasing internal timeout values (as described in the following section).

How to resolve?

Increasing the internal timeout values in the Meetingserver.ini file allows the server to take more time to start up before signaling an error. Normally, this should be treated as a temporary workaround.

Find the following sections and increase the values for the ConfigWaitTime and StartWaitTime parameters.

```

[SOFTWARE\Lotus\Sametime\MeetingServer]
ConfigWaitTime=60000

[SOFTWARE\Lotus\Sametime\MeetingServer\Services]
StartWaitTime=180000

```

The value is in milliseconds, so the default value is 3 minutes. Increase the value.

Note: There is another StartWaitTime parameter in the [SOFTWARE\Lotus\Sametime\MeetingServer\Services\CommunityServer] section of the Meetingserver.ini file for which the default value is 0. **Do not** change this value.

If you are running a Sametime 8.5 (or later) Entry server, also refer to *Sametime on IBM i: Entry server shuts down during startup*.

3. Sametime server startup failure due to an exception that occurred.

Symptom

The Sametime server shuts down immediately after the server startup successful message is displayed, followed by an error message, *Sametime: Meeting server: child process died! Ending server.* The following logs appear in the Domino console:

```
Sametime: All services started successfully.  
Sametime: Server startup successful.  
Sametime: Meeting server: child process died! Ending server.  
Sametime: Shutdown detected.  
Sametime: Stopping services...
```

If you look at the *IBM_TECHNICAL_SUPPORT* directory, there would be two different NSDs generated. Refer to *Important things to know about the NSD files* to find the meaningful NSD that would be better for the problem probe.

How to resolve?

Most of the time, this problem is due to a network configuration error or because you missed installing some important PTFs. Refer to *Check first when the Sametime server is not working well* in step 1 to exclude the possibility of configuration issues first, and then restart the Sametime server to try again.

If you look at the Sametime.log file and find that STLINKS is the first task to end, make sure that your network configuration has no problem. Another tip that might help resolve the connection problem is to replace all of the host names in the Sametime.ini and Meetingserver.ini files with the actual IP address.

If the issue still exists, check the call stack in the meaningful NSDs to locate the Sametime job that has the exception, and find out that module's trace to check what has happened. Collect the traces and send them to an IBM engineer for support.

4. Sametime server start failures due to orphaned shared memory or semaphores.

When the Domino or Sametime server runs, it uses interprocess communication (IPC) resources that require proper termination, and then the server goes down.

Symptom

Sametime services cannot start up because the Domino server's jobs do not start up successfully.

Reason

In cases where a clean shutdown is not performed, especially after a Domino or Sametime server crashes, stale IPC resources may remain, and the next Sametime or Domino server start might fail, crash, or hang.

How to resolve?

Use the DLTDOMSMEM and DLTDOMSEM commands to clean up the orphaned shared memory or semaphores after a Domino server crash.

Use the DLTSTSHMEM command to clean up the orphaned shared memory after a Sametime server crashes.

5. Sametime server startup failure due to other possible reasons.

- **Very large Domino directory or directory structure**

Sametime startup issues can occur if the Domino directory is very large (for example, 800 MB) or the directory structure includes cascading address books or directory assistance.

As it starts, Sametime attempts to read the entire directory or directory structure into memory. If Sametime is unable to read all the information into memory before the specified timeout period, the startup issues described earlier might occur.

The solution is to increase the internal timeout values in the MeetingServer.ini file (described earlier).

- **Sametime server has not been properly configured to work with Secure Sockets Layer (SSL).**

If this is a Sametime server, enabled for SSL, and the specific service that fails to start is the configuration bridge, then it is likely related to SSL configuration problems. Refer to *Sametime server Configuration Bridge service fails to start* for additional information

- **Domino Servlet Manager is not enabled in the server document.**

This is likely the problem if the only Sametime job running is STADDIN2. Also, the Domino console indicates that none of the Sametime servlets were initialized, and there is no indication of any attempt to start Sametime services in the console. You do not see the following messages in the Domino console:

```
05/16/2003 07:59:08 AM Sametime Server: Starting services
05/16/2003 07:59:25 AM Sametime: Building service start order from
```

You need to perform the following steps to correct the problem:

1. Open the server document in the Domino directory (*names.nsf*).
2. Select the **Internet Protocol** tab and then select the **Domino Web Engine** tab.
3. In the **Java Servlets** section, set the Java Servlet support field to **Domino Servlet Manager**.
4. Save the server document and restart the server.

- **Extraneous entries in the server task list.**

Assuming that your Domino server is dedicated to running Sametime (as recommended), you should check the **ServerTasks** entry in the notes.ini file to ensure you are not starting unnecessary tasks such as QNNINADD and LEI. Make sure that your notes.ini file does not include the following entries:

```
NSF_HOOKS=QNNDIHK
EXTMGR_ADDINS=DECSEXT
```

- **Problem with PASE (Portable Application Solution Environment) installation.**

Refer to the *Sametime IBM i: Sametime server fails to start* technote for more information.

6. Sametime server shutdown issue.

Symptom

In rare cases, the Sametime Community Server might crash during server shutdown with one or more of the Sametime jobs still active. In this case, you may need to manually end some of the server jobs before you can restart the server.

How to resolve?

If you need to manually end the remaining active Sametime jobs and then restart the server, following the following steps:

- a. Run the `WRKDOMSVR` command, and enter option 6 (shutdown server) on your Sametime server.
- b. Wait a few minutes for any jobs to shut down. Use option 9 (work with server jobs) to check the status of the Sametime jobs.
- c. If there are still Sametime jobs that do not end, run the following command to end the server immediately:

```
ENDDOMSVR <servername> *IMMED
```

Again, wait a few minutes and then use `WRKDOMSVR` option 9 to check the status of the jobs. If there are still Sametime jobs running, you will have to end them abnormally. The `ENDJOBABN` command can be used to kill jobs that will not shut down using other methods, but you must wait for at least 10 minutes after attempting to end the server using the `*IMMED` command.

If you need to end any Sametime jobs abnormally, perform the following steps:

1. Run the `WRKDOMSVR` command and enter option 9 (work with server jobs) on your Sametime server.
2. Enter option 5 (work with job) on any of the remaining jobs. The **Work with Job** menu is displayed. The top of the menu contains information similar to the one shown in Figure 1.

Figure 1.



```

Work with Job
Job:  STCOMMUNITY  User:  QNOTES  Number:  110377  System:  Q6060A1C
  
```

3. On the command line at the bottom of the menu, run the `ENDJOBABN` command, specifying the job number, the user, and the job name. For this example, you can run the following command:

```
ENDJOBABN 110377/QNOTES/STCOMMUNITY
```

4. Repeat steps 2 and 3 for each remaining server job. After you have verified that all the server jobs have ended, restart the Sametime Community Server.
7. Sametime server hang or crash

Symptom

- **Server hang**

The Sametime server can not accept any event requests such as login, chat, or meeting, and so on. And when you type the `show task` command in the Domino console, the server has no response.

- **Server crash**

Usually you will see the message `Sametime: Meeting server: child process died! Ending server.` in the Domino console, the and NSDs will be generated in the `IBM_TECHNICAL_SUPPORT` directory.

How to resolve?

- **Server hang**

Usually, the reason for server hang is a temporary network or performance issue. Restarting the server might resolve the issue.

Check to see if there are any exceptions or JVM errors in the trace files, or if some Java Coredump files were generated. Reach out to an IBM engineer for further support.

If the *javacore_*.txt* shows the following message:

```
Dump Event "systhrow" (00040000) Detail "java/lang/OutOfMemoryError" received
```

then increase the value that follows `-Xmx` in the `Sametime.ini` and `Meetingserver.ini` files to see whether the problem can be resolved.

- **Server crash**

Often, for this kind of error, you should contact an IBM engineer for help if you have already ruled out the configuration issue mentioned in step 1, *Check first when the Sametime server is not working well*.

But if you can distinguish whether the errors are from the Domino or Sametime server side, you might be able to resolve the problem more efficiently. We will talk about this in the following section.

8. Errors returned from the Domino application programming interface (API) cause a Sametime server failure.

As the Domino-based Sametime is an add-in task of the Domino, it has a close relationship with the Domino server. For example, it uses Domino APIs to operate on all the *NSF* and *NTF* files, it uses some information from `Notes.ini`, and it uses Domino APIs to write or read these values, and so on.

But deciding whether the error is from the Domino or Sametime server is not easy. You will need a lot of development or support experience to determine this. If you have tried all of the solutions you know, but cannot resolve the Sametime server's problem, note that the issue might arise from Domino, and not Sametime.

Some obvious problems from Domino but not Sametime:

When you look at NSD's call stack, the issued function or module has the `NSFxxx` or `NIFxxx` prefix, and belongs to the `LIBNOTES` service program. Here is an example of a typical call stack:

```
<@@ Notes Process Info -> Call Stack for Process @@>
```

```
JOB: 847412/QNOTES/STLOGGER    THREAD: 0x133
```

| | | | |
|---------------------------------------------------------|-----|-------------------------|-----------------------|
| <code>_CXX_PEP_Fv</code> | 0 | <code>QP0ZPCP2</code> | <code>QP0ZPCP2</code> |
| <code>Qp0zNewProcess</code> | 264 | <code>QP0ZPCPN</code> | <code>QP0ZPCPN</code> |
| <code>InvokeTargetPgm__FP11qp0z_pcp_cb</code> | 181 | | |
| <code>_CXX_PEP_Fv</code> | 6 | <code>VPLOGGER</code> | <code>STLOGGER</code> |
| <code>main</code> | 44 | | |
| | | | |
| | | | |
| <code>onStartExecute__11VpDbRequestFv</code> | 11 | <code>DBREQUEST</code> | |
| <code>prepareViewFolder__9VpDbNotesFR6SnViewPCcb</code> | 5 | <code>DBNOTES</code> | |
| <code>openDatabase__9VpDbNotesFv</code> | 10 | | |
| <code>open__10SnDatabaseFPCc</code> | 6 | <code>SNDATABASE</code> | <code>STNTSLYR</code> |
| <code>NSFDbOpen</code> | 2 | <code>STUB</code> | <code>LIBNOTES</code> |

```

NSFDbOpen                1  DBOPEN2
NSFDbOpenExtended        5  DBOPEN
NSFDbOpenExtended3       2
NSFDbOpenExtended4       1
NSFDbOpenExtended5       2255
.....
.....
Handle_Unexpected_Exceptions_
system                    1  VPLOGGER  STLOGGER
                        13  QC2SYS    QC2SYS
303  QCMDEXC
_CXX_PEP__Fv              0  STUBPGM  NSD
main                      27
Call0Parm__FPcT1iPPc     27
_C_pep

```

In some trace files, you can see the following message:

```

111012_131928.639,INF,StNotesL,
Failed to open a collection for view [By Form] in database [stconfig],
notes C api error [33557]
[Someone else deleted this index while you were updating it.]
111012_131928.640,INF,Notes ,Error, failed to open view [By Form]:
Someone else deleted this index while you were updating it.

```

Summary

This article first described how to enable and collect the Sametime diagnosis traces, and then explained how to read these trace files. At the end, it listed some common problems, including their symptoms, reasons, and the way to resolve them. It can be difficult and time-consuming to retrace all of the steps you performed to configure a Sametime server. We hope these topics we have discussed will help our customers to resolve some simple and common problems, and to find the source of the problems.

References

Here are a few materials that can help you to get a better understanding of this article:

- [Turning on semaphore debugging parameters in notes.ini for Domino](#)
- [Sametime for IBM i: Most Common Problems for more detail information](#)
- [Sametime for IBM i: System Requirements](#)
- [IBM Systems Workload Estimator](#)
- [Sametime on IBM i: Entry server shuts down during startup](#)
- [Sametime server Configuration Bridge service fails to start](#)
- [Sametime IBM i: Sametime server fails to start](#)
- [How to enable Sametime Component debugging](#)

© Copyright IBM Corporation 2012

(www.ibm.com/legal/copytrade.shtml)

Trademarks

(www.ibm.com/developerworks/ibm/trademarks/)