

# InfoSphere Information Server DataStage Version 8

## How to trace running parallel jobs



© 2011 IBM Corporation

This presentation explains how to trace running parallel DataStage® jobs in version 8.

## Objectives

- Why and when you should use tracing
- How to prepare parallel job for tracing
- How to use trace job from command line
- How to analyzing trace output

The objectives of this presentation are to explain why and when you should use tracing, how to prepare a parallel job for tracing, how to use trace from a command and how to analyze the trace output.

## Why and when you should use tracing

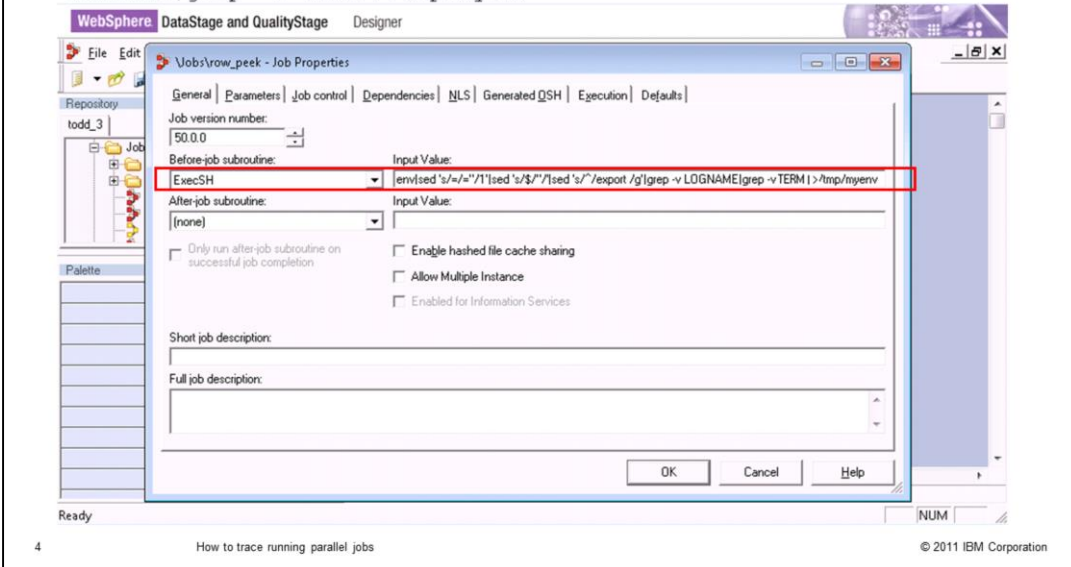
- Trace is report of system calls made by program
- Useful to troubleshoot run-time issues reproduced
- Trace started simultaneously with job to get complete information

A trace is a report with detailed information of the system calls a process invokes while running. Some examples of the data you can capture include: environment variables, parameter values, library locations and return codes. With regards to DataStage parallel jobs, a trace is recommended when you have a job that is showing a consistent problem at run-time and for which the messages in the log are not enough. Although a trace can be performed on a running job that was started from Designer or Director, it is better if you start the trace at the same time the job starts. This will allow you to capture a complete report of the job's activities since the main process was created and you will not have to manually identify the *pids* of the job's processes. This presentation shows you how to run your job from a command line so it can be easily traced.

## Prepare parallel job to be traced (1 of 2)

- Add before-job subroutine with input value:

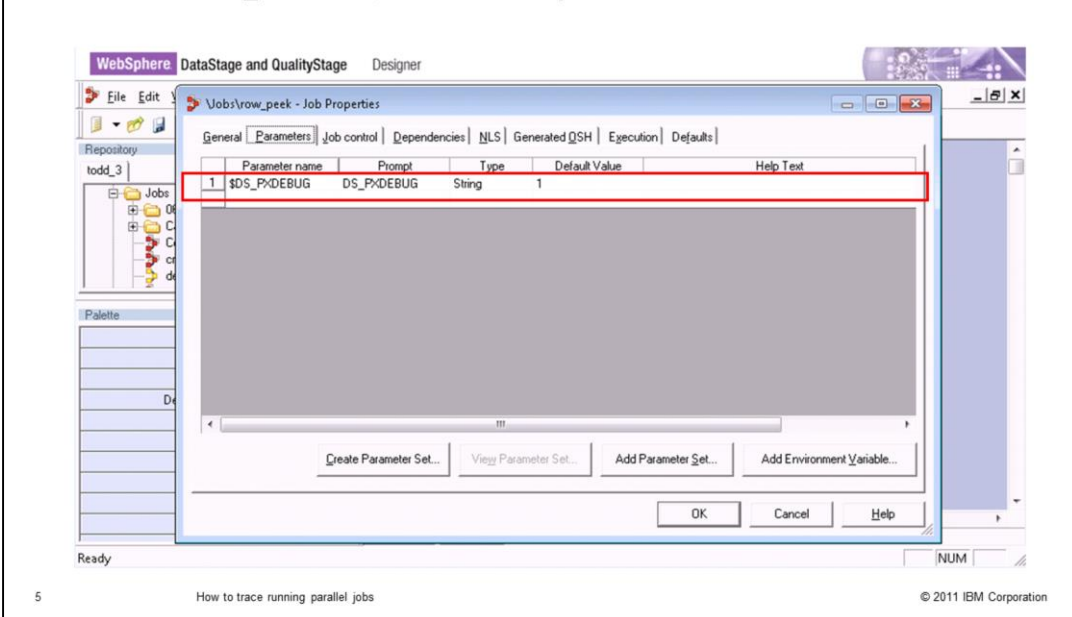
```
env|sed 's/=/="/1'|sed 's/$/"/'|sed 's/^/export /g'|grep -v LOGNAME|grep -v TERM >/tmp/myenv
```



To ensure that the execution of the job from the command line reproduces the same environment of the job started from Designer or Director, set the environment variables of your terminal session so they have the same values as those used by the job during runtime. To capture the value of these variables you can add a before sub-routine under Job Properties with the input value displayed on this slide. This will create a file called `/tmp/myenv` that you can source from your terminal session. You can change the path of the output file as needed.

## Prepare parallel job to be traced (2 of 2)

- Add variable DS\_PXDEBUG, set to 1 and run job



To run a job from the command line you also need some auxiliary files that can be obtained by turning debugging on for the PXEngine. To do this, add the user-defined environment variable DS\_PXDEBUG under Job Properties and set it to one. If you have never used this variable, you need to create it for this project from the DataStage Administrator client. When creating this variable, always set its default value to zero at the project level as this variable may have an impact on the performance of your jobs and should only be used to troubleshoot specific problems. Add the variable as a job parameter for this job and set the value to one. Now you can save, compile and run the job. This will create a folder under `Projects/<YourProjectName>/Debugging/<YourJobName>` where `<YourProjectName>` is the name of your project and `<YourJobName>` is the name of this job.

## Verify you can run job from osh command (1 of 2)

- Login as same user that runs job
- Source dsenv file
- Source myenv file created in previous slide

```
$ cd /opt/IBM/InformationServer/Server/DSEngine
$ . ./dsenv
$ . /tmp/myenv
```

- Go to Debugging folder created for this job:

```
$ cd ../Projects/<YourProjectName>/Debugging/<YourJobName>
$ osh -f OshScript.osh -pf jpfile
```

Next, open a terminal session and login using the same user that runs the job. Go to the directory where DSEngine is located and source the dsenv file. Next, source the file /tmp/myenv created on slide 4. In this example, it is in /tmp. If you obtain errors while sourcing /tmp/myenv then you will have to manually modify the file to remove or edit those variables that are preventing you from sourcing it.

Next, go to the Debugging folder created under this project folder and validate if you can run the job from the command line by using the osh command displayed on this slide.

## Verify you can run job from osh command (2 of 2)

Check end of output and validate output of job:

```
...  
##I IIS-DSEE-TOIX-00059 14:12:31(000) <APT_RealFileExportOperator in  
Sequential_File,0> Export complete; 10000 records exported  
successfully, 0 rejected.  
##I IIS-DSEE-TFSC-00010 14:12:31(000) <main_program> Step execution  
finished with status = OK.  
##I IIS-DSEE-TCOS-00026 14:12:31(001) <main_program> Startup time,  
0:00; production run time, 0:00.
```

Finally, check the end of the output and validate that you are reproducing the same behavior as when you run the job from Director. In this example, the job ends successfully in Director and you also see that the job ends with "status = OK" using the command line. If your job is hanging or aborting with an error, you will also get the same error or hanging condition using this command line.

## Tracing job

- To get trace of job run command:

```
$ truss -o truss.output osh -f OshScript.osh -pf jpfile
```

- File truss.output will have output of the trace
  - Inspect file using text editor
- View IBM Education Assistant module on operating system level tracing for more information

[http://publib.boulder.ibm.com/infocenter/ieduasst/imv1r0/topic/com.ibm.iea.infosphere\\_is/infosphere\\_is/8.5/ds\\_gs\\_ProblemDetermination.html](http://publib.boulder.ibm.com/infocenter/ieduasst/imv1r0/topic/com.ibm.iea.infosphere_is/infosphere_is/8.5/ds_gs_ProblemDetermination.html)

Once you have validated that you can reproduce the same behavior of the job using the command line, you are ready to trace the job. To do this, use the command `truss` (or the equivalent in the operating system that you are using) before the `osh` command. For example:

```
truss -o truss.output osh -f OshScript.osh -pf jpfile
```

This will create a file called `truss.output` with the output of the trace. You can use different options of `truss` depending what type of information you want to trace. Check the main page of `truss` to learn more about this command.

The `truss` utility is included by default in AIX®. Other operating systems have other trace utilities that can be used instead of `truss`. In Linux® or Solaris, you can use `strace` and in HP-UX you can use `tusc`. If the utility is not present you need to download it. You can also view the IBM Education Assistant module on operating system level tracing for more information.



## Examples of trace output (1 of 2)

```
$ truss -c -o truss.summary osh -f OshScript.osh -pf jpfile
```

```
$ cat truss.summary
```

```
signals -----
```

```
...
```

```
syscall          seconds   calls  errors
```

```
...
```

```
open             4.828   133922 129527
```

```
close            3.297   10429   4
```

```
unlink           21.392    781   190
```

```
...
```

```
sys totals:      2.997  136640  58825
```

```
usr time:        .935
```

```
elapsed:        25.130
```

Using the option `-c`, `truss` generates a summary of the time spent for each type of system call. This type of tracing can be used to identify bottlenecks in the execution of a job. In this example, the output shows that 21 seconds of the total job execution were spent in `unlink` system calls, which are part of the file deletion process. This might indicate an I/O problem possibly related to the use of a non-local system. If the job is using NFS files, the next test is to run the same job pointing to local files and compare times for this call.

## Examples of trace output (2 of 2)

```
$ truss -ae -o truss.detail osh -f OshScript.osh -pf jpfile
$ cat truss.detail
...
statx("/InformationServer/Server/PXEngine/etc/operator.apt",
0x0FFFFFFFFFFFF9CC8, 176, 0) = 0
access("/InformationServer/Server/PXEngine/etc/operator.apt", 04)
Err#13 EACCES
statx("/InformationServer/Server/Projects/dstage1/buildop",
0x0FFFFFFFFFFFA150, 176, 0) = 0
statx("/InformationServer/Server/Projects/dstage1/buildop/operator.ap
t", 0x0FFFFFFFFFFFF9CC8, 176, 0) Err#2 ENOENT
sbrk(0x0000000000000000) = 0x00000000101B4B80
times(0x000000011017D2B0) = 1011039719
* * *
```

How to trace running parallel jobs

© 2011 IBM Corporation

Using the option `-ae`, you can obtain a more detailed output including the parameters passed for each system call function. This type of output is useful to identify problems caused by missing libraries or lack of permissions.

In this example, you see that the file `operator.apt` was found but the process was not able to access. The next step is to validate that users have the right permissions to access this file.

## Trademarks, disclaimer, and copyright information

IBM, the IBM logo, ibm.com, AIX, DataStage, and InfoSphere are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of other IBM trademarks is available on the web at "[Copyright and trademark information](http://www.ibm.com/legal/copytrade.shtml)" at <http://www.ibm.com/legal/copytrade.shtml>

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. Other company, product, or service names may be trademarks or service marks of others.

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS OR SOFTWARE.

© Copyright International Business Machines Corporation 2011. All rights reserved.