Tivoli software

IBM

Pete Soto Jr
03/30/09
Rev. 1.2.1

Whitepaper

sym_update.sh documentation

This paper explains the function of the *sym_update.sh* script, requirements and operation, to facilitate required procedures for a compromised (corrupt) Tivoli Workload Scheduler Symphony file on a TWS agent.

Utility Overview

The *sym_update.sh* utility script is designed to run from the Tivoli Workload Scheduler Version 8.2.0 or greater workstation with the following UNIX operating systems:

- AIX
- HPUX
- Linux_i386
- Solaris

The *sym_update.sh* is used to generate an updated Sinfonia file that may be sent to a UNIX or Windows TWS agent that has a compromised (corrupt) Symphony file. The TWS agent will receive the updated Sinfonia file as a Symphony file. The *sym_update.sh* utility script must run from the UNIX prompt as the “TWS” user, and not as “root” user, on either TWS Master Domain Manager (MDM), active Backup Domain Manager (BKM) or Domain Manager (DM) workstations.

Important: The *sym_update.sh* may be run on backup master (active BKM) to send an updated Symphony file to the original master only if a TWS "switchmgr" command has been issued making it the active master. The script may then be used to send and updated Symphony file to the original master from the active backup master.

Script allows TWS agent to resume operations quickly, minimize loss of job or job stream information, minimize time consuming task of cancelling jobs or job streams which were requirements of previous solutions. User will need to verify information in Symphony file received by TWS agent before resuming operations.

Caution: This procedure involves renaming TWS files and should only be attempted if Symphony file has been compromised on TWS agent. Contact L2 support if you are not sure of status of Symphony file.

The *sym_update.sh* script displays status of completed tasks that are being performed. See Figure 1

If an error is encountered in performing a task, the script will echo the error and exit the script. The error message will also be displayed after the completed task list. See Figure 1.

If *sym_update.sh* script does not complete successfully, the task list and error message will assist in troubleshooting by identifying the last completed task and task that failed or generated an error.

```
Logon user should not be root               [passed]
Valid OS                                    [passed]
TWS Agent specified                         [passed]
twsuser m83rtm is valid                     [passed]
CPU taos for m83rtm is a MDM or DM          [passed]
TWSCPUNAME different from TWSAGENT CPU      [passed]
twsagent data exists in current Symphony    [failed]
Set limit for TWSAGENT to 0 and stop TWS    [ ]
TWS processes not active                    [ ]
Original Sinfonia file renamed              [ ]
Copy Symphony as Sinfonia                   [ ]
Netman started                              [ ]
TWS processes started                       [ ]
Link command issued for agents              [ ]
Start command issued for TWS agent          [ ]
Exiting /tmp/utils/sym_update.sh            [ ]

Error:  The twsagent data does not exist in current Symphony.
Specify an agent that exists in the Symphony file.
```

Figure 1 Sample completed task list

sym_update.sh Requirements

The /tmp directory, the directory for temporary files, must exist where the *sym_update.sh* script will be executed.The temporary files will have a prefix of **sym**_ and include the process id **.<pid>** as an extension. These files will be removed when script ends normally.

Temporary files will default to the /tmp directory. The script will terminate if the /tmp directory does not exist. The directory for temporary files may be customized (discussed shortly).

Attention: Temporary files will not be removed if the *sym_update.sh* is killed or terminated while still running. The temporary files in the /tmp directory will need to be removed manually.

The *sym_update.sh* utility script must run from the UNIX prompt as the “TWS” user, and not as “root” user, on either TWS Master Domain Manager (MDM), active Backup Domain Manager (BKM) or Domain Manager (DM) workstations.

The *sym_update.sh* requires specifying the TWS agent workstation name that will receive the updated Symphony file. The agent name is required since the script will set the cpu limit to “0” before performing the Symphony update procedures. If multiple TWS agents have a compromised Symphony file, user will need to set the cpu job limit of TWS agent(s) to “0” to all but one of the TWS agents (discussed later) before executing the script.

Setting the cpu job limit to “0” is necessary to prevent jobs or job streams from executing until the user has verified that TWS operations may resume on the TWS agent(s).

The cpu job limit must be raised for the TWS agent before normal operations may resume on the TWS agent.

Attention: The cpu job limit must be raised for TWS agent before normal operations may resume on the TWS agent. This may be performed via Job Scheduling Console (JSC) or conman.

The Symphony file must be compromised (corrupt) on a TWS agent before implementing this option. TWS level 2 should be contacted if unsure that the Symphony file has been compromised.

The sym_update.sh must run after TWS agent with compromised Symphony file has stopped TWS processes and renamed Sinfonia, Symphony, Jobtable, TWSHome/*.msg and TWSHome/pobox/*.msg files.

Attention: The StartUp command on TWS agent with compromised Symphony file may only be executed after the *sym_update.sh* script has completed successfully on MDM, active BKM or DM.

The “thiscpu” variable in the TWSHome/localopts and entries in the /etc/TWS/TWSRegistry.dat file for the TWS user must be correct and valid.

The *sym_update.sh* script should considered a last resort option since it involves unlinking from all TWS managed agents, shutting down TWS processes on a TWS MDM, active BKM or DM, renaming of current Sinfonia file and copying Symphony as Sinfonia.

The *sym_update.sh* script will minimize the time that TWS processes are down. Currently executing jobs on the current MDM, active BKM or DM will not be affected and jobs will resume launching once the TWS processes on the MDM, active BKM or DM have been restarted by the script.

The script is run once even if multiple TWS agents have compromised Symphony files at the same time but additional procedures will be required on the MDM, active BKM or DM (discussed shortly).

The script must be run again and procedures repeated if another TWS agent has a compromised Symphony file later in the day. This procedure is necessary since the updated Symphony file is generated for a specific time frame.

The *sym_update.sh* may be run on backup master (active BKM) to send an updated Symphony file to the original master only if a TWS “switchmgr” command has been issued making it the active master. The script may then be used to send and updated Symphony file to the original master from the backup master.

The script may exist in any directory, must be executed on MDM, active BKM or DM workstation as the MDM, BKM or DM TWS user and script must have execute permissions. This requirement is necessary since TWS operations performed by the script require TWS user access.

Attention: The *sym_update.sh* must be executed on MDM, active BKM or DM workstation by the MDM, BKM or DM TWS user and may be run on a backup domain manager only if the switchmgr command has been issued making backup master the active master. The script will terminate if executed by a user other than the MDM, BKM or DM TWS user.

The *sym_update.sh* script is **not** designed to be run as a TWS job since the job would show a state of “EXEC” even though it may have completed successfully or abended. Furthermore the script requires a reply to a prompt and thus may not executed as a TWS job. Executing as a TWS job will cause *sym_update.sh* script to terminate. See Figure 2.

Attention: The *sym_update.sh* will terminate if it is executed as a TWS job. The script is not designed to be run as a TWS job.

```
Logon user should not be root               [ ]
Valid OS                                    [ ]
TWS Agent specified                         [ ]
TWSUSER is valid                            [ ]
CPU TWSCPUNAME for twsuser is a MDM or DM   [ ]
TWSCPUNAME different from TWSAGENT CPU      [ ]
TWSAGENT exists in current Symphony         [ ]
Set limit for TWSAGENT to 0 and stop TWS    [ ]
TWS processes not active                    [ ]
Original Sinfonia file renamed              [ ]
Copy Symphony as Sinfonia                   [ ]
Netman started                              [ ]
TWS processes started                       [ ]
Link command issued for agents              [ ]
Start command issued for TWS agent          [ ]
Exiting /tmp/utils/sym_update.sh            [ ]

Error: Script /tmp/utils/sym_update.sh was executed as a TWS job,
this is not permitted, exiting utility.
```

Figure 2 Invalid script execution, script executed as a TWS job

sym_update.sh Script Customizable Option

Variable: TMP_DIR

The sym_update.sh has one customizable option, the directory for temporary files. Temporary files will default to the /tmp directory. The script will terminate if the /tmp directory does not exist.

The temporary files will have a prefix of **sym**_ and include the process id .<**pid**> as an extension. These files will be removed when script ends normally.

The variable “TMP_DIR” for the default directory may be edited to specify a different directory. The variable is in the “Customizable Options” section of the script. See Figure 3

The path specified for the variable must be enclosed by double quotes. The user executing script must have permissions to create files in the specified directory.

```
#**************** Customizable Options *****************
#
#Directory location for temporary file. Directory must exist
otherwise script will terminate
#Default TMP_DIR="/tmp";
TMP_DIR="/tmp";export TMP_DIR
```

Figure 3 Customizable directory for temporary files

sym_update.sh Options

The *sym_update.sh* has one required option and one optional option.

Optional Option: Debug

The debug option will provide a more verbose output of all tasks performed by the script and is invoked by including the “-debug” text after the agent name.

```
sym_update.sh <agent_name> -debug
```

The verbose output will default to stdout.

When *sym_update.sh* is executed from the UNIX prompt, stdout may be re-directed to a file by performing the following steps, see Figure 4.

```
1) Issue the script command and specify a file that will contain the
output.
   Example:
   script -a filename

2) Execute the script:
   Example:
   sym_update.sh <agent_name> -debug

3) Once script completes, issue a <ctrl> d to terminate capturing of
stdout to file.
   Example:
   <ctrl> d
4) Review the contents of filename specified.
```

Figure 4 Capturing stdout to file

Script Execution Option

The *sym_update.sh* script has one required option, <agent_name>, the agent that will receive the updated Symphony file. See Figure 5.

```
sym_update.sh syntax:
sym_update.sh <agent_name> -debug (optional)
```

Figure 5 sym_update.sh syntax

The script verifies that the TWS agent was provided. If the TWS agent was not provided will exit and display an error referencing the *sym_update.sh* script syntax option. See Figure 6.

```
Logon user should not be root               [passed]
Valid OS                                    [passed]
TWS Agent specified                         [failed]
TWS user is valid                           [ ]
CPU TWSCPUNAME for twsuser is a MDM or DM   [ ]
TWSCPUNAME different from TWSAGENT CPU      [ ]
TWSAGENT exists in current Symphony         [ ]
Set limit for TWSAGENT to 0 and stop TWS    [ ]
TWS processes not active                    [ ]
Original Sinfonia file renamed              [ ]
Copy Symphony as Sinfonia                   [ ]
Netman started                              [ ]
TWS processes started                       [ ]
Link command issued for agents              [ ]
Start command issued for TWS agent          [ ]
Exiting /tmp/utils/sym_update.sh           [ ]

Error: TWS Agent not provided.   USAGE: sym_update.sh <agent_name>
```

Figure 6 Invalid script option example

Script Operation

The *sym_update.sh* script renames original Sinfonia file, copies Symphony file as Sinfonia and restarts TWS processes. These operations must be performed by the MDM active BKM or DM TWS user and not “root” or other user.

The *sym_update.sh* may be run on backup master (active BKM) to send an updated Symphony file to the original master only if a TWS “switchmgr” command has been issued making it the active master. The script may then be used to send and updated Symphony file to the original master from the backup master.

The script will verify that user executing script is not the “root” user and is a TWS user on the master backup, active backup master or domain manager workstation before proceeding with the execution of the script otherwise the script will terminate. See Figure 7 and Figure 8.

Attention: The *sym_update.sh* must be executed on MDM, active BKM or DM workstation by the MDM, BKM or DM TWS user and may be run on a backup domain manager only if the “switchmgr” command has been issued making backup master the active master. The script will terminate if executed by a user other than the MDM, BKM or DM TWS user.

```
Logon user should not be root               [failed]
Valid OS                                    [ ]
TWS Agent specified                         [ ]
TWSUSER is valid                            [ ]
CPU TWSCPUNAME for twsuser is a MDM or DM   [ ]
TWSCPUNAME different from TWSAGENT CPU      [ ]
TWSAGENT exists in current Symphony         [ ]
Set limit for TWSAGENT to 0 and stop TWS    [ ]
TWS processes not active                    [ ]
Original Sinfonia file renamed              [ ]
Copy Symphony as Sinfonia                   [ ]
Netman started                              [ ]
TWS processes started                       [ ]
Link command issued for agents              [ ]
Start command issued for TWS agent          [ ]
Exiting /tmp/utils/sym_update.sh            [ ]

Error: You must not be root when executing this utility.
```

Figure 7 Invalid user, script executed as root

```
Logon user should not be root               [passed]
Valid OS                                    [passed]
TWS Agent specified                         [passed]
twsuser m83rtm is valid                     [passed]
CPU taos for m83rtm is a MDM or DM          [passed]
TWSCPUNAME different from TWSAGENT CPU      [ ]
TWSAGENT exists in current Symphony         [ ]
Set limit for TWSAGENT to 0 and stop TWS    [ ]
TWS processes not active                    [ ]
Original Sinfonia file renamed              [ ]
Copy Symphony as Sinfonia                   [ ]
```

Figure 8 Valid TWS user on a MDM or DM workstation

The *sym_update.sh* script will verify that the current workstation is not the same as specified TWS agent otherwise it will terminate. See Figure 9.

```
Logon user should not be root                [passed]
Valid OS                                     [passed]
TWS Agent specified                          [passed]
twsuser m83rtm is valid                      [passed]
CPU taos for m83rtm is a MDM or DM           [passed]
TWSCPUNAME different from TWSAGENT CPU       [failed]
TWSAGENT exists in current Symphony          [ ]
Set limit for TWSAGENT to 0 and stop TWS     [ ]
TWS processes not active                     [ ]
Original Sinfonia file renamed               [ ]
Copy Symphony as Sinfonia                    [ ]
Netman started                               [ ]
TWS processes started                        [ ]
Link command issued for agents               [ ]
Start command issued for TWS agent           [ ]
Exiting /tmp/utils/sym_update.sh             [ ]

Error: CPU taos for TWS user m83rtm may not be the same as TWSAGENT
CPU (taos), CPU that will receive updated Symphony file. Specify a
different TWS agent.
```

Figure 9 Invalid TWS agent, TWS workstation name same as DMD or DM workstation

The *sym_update.sh* script will validate the TWS agent and will terminate if the TWS agent does not exist in the current Symphony file. See Figure 10.

```
Logon user should not be root                [passed]
Valid OS                                     [passed]
TWS Agent specified                          [passed]
twsuser m83rtm is valid                      [passed]
CPU taos for m83rtm is a MDM or DM           [passed]
TWSCPUNAME different from TWSAGENT CPU       [passed]
twsagent data exists in current Symphony     [failed]
Set limit for TWSAGENT to 0 and stop TWS     [ ]
TWS processes not active                     [ ]
Original Sinfonia file renamed               [ ]
Copy Symphony as Sinfonia                    [ ]
Netman started                               [ ]
TWS processes started                        [ ]
Link command issued for agents               [ ]
Start command issued for TWS agent           [ ]
Exiting /tmp/utils/sym_update.sh             [ ]

Error:  The twsagent data does not exist in current Symphony.
Specify an agent that exists in the Symphony file.
```

Figure 10 Invalid TWS agent, specified TWS workstation does not exist in current Symphony file

The script will generate temporary files in the /tmp directory. The generated files will have a prefix of **sym**_ and include the process id **.<pid>** as an extension. These generated files will be removed as soon as the script completes

The script will display the status for each task performed. The status will either be passed, failed, yes, no or blank.

The *sym_update.sh* will prompt the TWS user if they wish to continue before performing operations on the MDM, active BKM or DM. If user replies “n”, script will terminate and perform no actions on MDM, active BKM or DM workstation. See Figure 11

If user replies “y”, script will rename SInfonia as Sinfonia.mmddyy_hhmm prior to copy of Symphony as Sinfonia on MDM or DM.

The renamed Sinfonia.mmddyy_hhmm does not need to be renamed when script completes and may be removed at a later date.

```
Warning this utility will shut TWS processes on MASTER taos to
generate and send an updated Sinfonia file to FT-AGENT fta83rtm
Do you wish to continue? (Enter y, n) -> n
Will exit /tmp/utils/sym_update.sh per user request.

Logon user should not be root               [passed]
Valid OS                                    [passed]
TWS Agent specified                         [passed]
twsuser m83rtm is valid                     [passed]
CPU taos for m83rtm is a MDM or DM          [passed]
TWSCPUNAME different from TWSAGENT CPU      [passed]
twsagent fta83rtm exists in current Symphony [passed]
Set limit for TWSAGENT to 0 and stop TWS    [ ]
TWS processes not active                    [ ]
Original Sinfonia file renamed              [ ]
Copy Symphony as Sinfonia                   [ ]
Netman started                              [ ]
TWS processes started                       [ ]
Link command issued for agents              [ ]
Start command issued for TWS agent          [ ]
Exiting /tmp/utils/sym_update.sh            [ ]

Warning:  Exiting script per user request
```

Figure 11 Continue script execution prompt

sym_update.sh Implementation Procedures

The script **must** be executed soon after the TWS agent(s) with compromised Symphony file has been stopped, Sinfonia, Symphony, Jobtable, TWSHome/*.msg and TWSHome/pobox/*.msg files have been renamed.

The script is run once even if multiple TWS agents have compromised Symphony files at the same time but additional procedures will be required on the MDM, active BKM or DM (discussed shortly).

The script must be run again and procedures repeated if another TWS agent has a compromised Symphony file later in the day, since the updated Symphony file was generated for a specific time frame.

Attention: Do not issue a TWSHome/StartUp on TWS agent(s) until after the *sym_update.sh* completes successfully on MDM, active BKM or DM.

TWS Agent Procedures

The following procedures will be performed on TWS agent(s) with compromised Symphony file.

1. Unlink TWS agent(s) from master.
2. Issue TWSHome/bin/conman “stop;wait” onTWS agent.
3. Issue TWSHome/bin/conman “shut;wait” on TWS agent.
4. Verify that TWS batchman, mailman, netman and writer processes are no longer active. Active jobman processes may exist and **should not be** terminated.
5. Rename TWSHome/Symphony, TWSHome/Sinfonia, TWSHome/Jobtable, TWSHome/*.msg and TWSHome/pobox/*.msg file on TWS agent with compromised Symphony file.
6. Wait for successful completion of *sym_update.sh* script on MDM, active BKM or DM.
7. Issue TWSHome/StartUp script on TWS agent after successful completion of *sym_update.sh* script on MDM, active BKM or DM.

MDM/DM Procedures To Execute From UNIX Prompt:

If multiple TWS agents have compromised Symphony files, set cpu limit to “0” for those TWS agent(s) except for TWS agent that will be specified with script prior to executing *sym_update.sh* script (see below).

1. Verify that the “**TWS Agent Procedures**” steps 1 thru 5 have been performed.
2. Verify that *sym_update.sh* script has execute permissions.
3. Issue an unlink command for TWS agent(s) with compromised Symphony file via JSC or conman from MDM or DM.

Attention: If more than one TWS agents has a compromised Symphony file, set cpu limit to “0” for those TWS agent(s) except for agent that will be specified with script .

4. (Optional) If multiple TWS agents have a compromised Symphony file, set cpu limit for those TWS agent(s) to “0” except for TWS agent specified with script .
5. Execute the *sym_update.sh* script.

   sym_update.sh <***agent_name***> (See Figure 12).

```
sym_update.sh syntax:
sym_update.sh <agent_name> -debug (optional)

Where: agent_name is workstation not specified in step 4, that will
receive updated Symphony file.

sym_update.sh example:
sym_update.sh m83a
```

Figure 12 sym_update.sh syntax and example

6. Verify that s*ym_update.sh* script completes successfully. If not, refer to error message, make corrections and rerun script.
7. Issue TWSHome/StartUp script on TWS agent(s) with compromised Symphony file.
8. *The sym_update.sh* script issues a conman “link” command from MDM, active BKM or DM for all TWS agent(s) in current domain to establish communications and send Symphony file to TWS agents that do not have one. An additional link command to TWS agents with compromised Symphony file may be necessary from MDM, active BKM or DM.
9. *The sym_update.sh* script will issue a conman “start” command from MDM or DM for TWS agent specified when script was executed.
10. If multiple TWS agents had a compromised Symphony file, verify that these agents have not already started their TWS processes before issuing a TWS “start” command for TWS agent(s) via JSC or conman that were not specified in step 5.
11. Perform Required TWS Agent Post Operations (see below).
12. Verify that the job and job stream status on TWS agent(s) is correct.

Attention: Since .msg files were renamed on the TWS agent, information on jobs or job streams on the TWS agent may have been lost and not be reflected on the updated Symphony file received from MDM, active BKM or DM. Verifying the status of scheduled jobs and job streams, adhoc submitted jobs and job streams is necessary to prevent jobs from running more than once or not launching.

13. Raise TWS agent(s) cpu job limit to greater than “0” either via JSC or conman.

Required TWS Agent Post Operations

The *sym_update.sh* script will issue a conman “link” command to all agents from the MDM, active BKM or DM.

1. After the *sym_update.sh* completes successfully, the TWS user must execute the TWSHome/StartUp script on the TWS agent(s) to start the netman process. After netman has started on the TWS agent it will receive the updated Sinfonia file as Symphony and have a cpu job limit of “0”.
2. Once the TWS agent receives the Symphony file, the user will need to verify that jobs or job streams on TWS agent that received the updated Symphony file are correct before raising the cpu job limit for the TWS agent. Some jobs or job streams may need to be cancelled or adhoc jobs re-submitted on local TWS agent if updates did not get included in the local Symphony.

Attention: Since .msg files were renamed on the TWS agent, information on jobs or job streams on the TWS agent may have been lost and not be reflected on the updated Symphony file received from MDM, active BKM or DM. Verifying the status of scheduled jobs and job streams, adhoc submitted jobs and job streams is necessary to prevent jobs from running more than once or not launching.

Normal processing may resume after verifying information that exists on the TWS agent Symphony file.

Attention: The job limit must be raised for TWS agent before normal operations may resume on the TWS agent. This may be performed via JSC or conman.

Troubleshooting

The script has specific requirements for execution. Review and verify the following:

1. The /tmp directory must exist.
2. Script must not be executed as “root”.
3. Script must not be executed as a TWS job.
4. Script must be executed as TWS user from a MDM, active BKM or DM workstation.
5. Script *sym_update.sh* must be executable.
6. A valid TWS agent must be specified and exist in the current Symphony file on MDM, active BKM or DM
7. The /tmp directory must exist.
8. The /etc/TWSRegistry.dat file must exist and be valid and current for the TWS user.
9. TWS file system and /tmp directory must has sufficient disk space.
10. If problems still exist, invoke script with “-debug” option as outlined previously and send output to TWS L2 support.