

Watch_sys User Documentation

V 1.7

Watch_sys is written in DCL.

The purpose of the procedure is to monitor things on OpenVMS systems. Having a background task that monitors some items on a system is sometimes useful to catch errors quickly, without the hassle of watching continuously the output of Opcom or other monitoring tools.

The procedure sends its alarms via email (smtp) and/or adds entries in the operator.log file. Alarms are also sent to the terminal (interactive run) or in a log file (procedure running in batch mode).

It has been written in such a way that the main and unique procedure does not need to be adapted to fit your needs. All variables, parameters are declared in external data files and it's a quite fast and easy task to change the list of things to be monitored, even when the procedure runs.

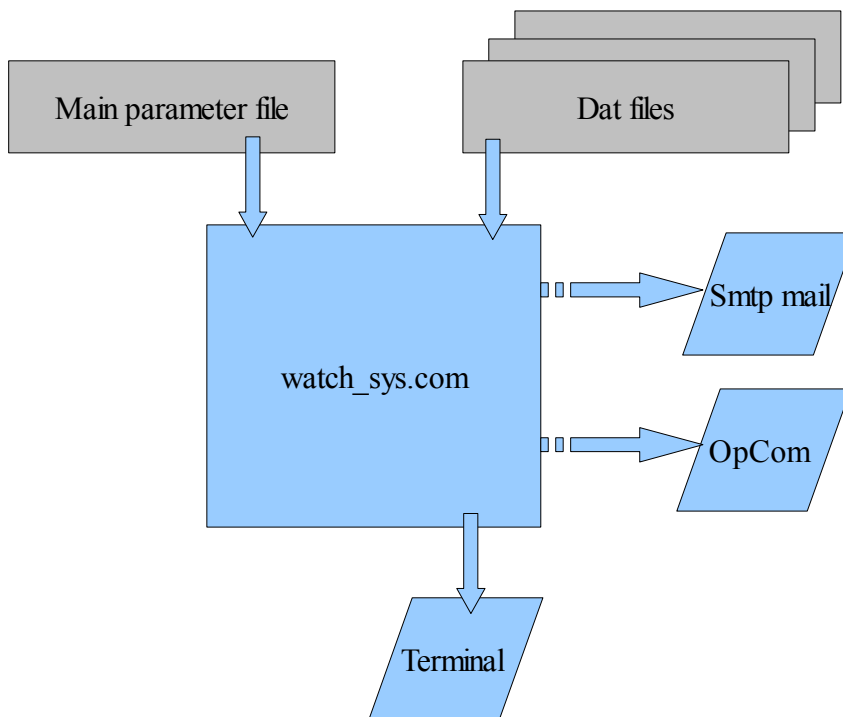
The procedure runs continuously, at an interval you specify.

Watch_sys is composed of :

- A dcl procedure

- One parameter file (that can be changed on the fly)

- Some dat files for specific monitoring tasks (also modifiable on the fly).



This version (1.7) checks the following items :

New hardware errors. Increments of device errors counters after the initial scan of the procedure.

Host ip ping tests.

Batch and print queues status.

Batch and print jobs status.

Processes that have to exist on the system.

Files that should/shouldn't reside on the system.

Free disk space.

Mean idle cpu.

Maximum processes allowed.

Process COM queue.

Process in SUSP states.

Intrusions.

Shadow sets state.

Cluster members entering/leaving the cluster.

Searching patterns in files.

Running external procedures.

Watch some internal hardware raid devices via msa\$util or sas\$util.

Heartbeat. An email can be sent everyday to be sure that the procedure runs.

Possibility to run scans only during periods of a day.

Pagefile usage.

The main parameter file specification is also a variable, so you will be able to monitor different things in different circumstances.

It is also possible to run the procedure for "one scan" only.

Detailed description.

Files

WATCH_SYS.COM

WATCH_SYS.PAR

WATCH_SYS_DISK_FREE_SPACE_EXCEPTIONS.DAT

WATCH_SYS_EXTERNAL.DAT

WATCH_SYS_FILES.DAT

WATCH_SYS_IP_REACHABILITY_HOSTS.DAT

WATCH_SYS_MONITORED_PROCESS.DAT

WATCH_SYS_SEARCH_FILES.DAT
WATCH_SYS_QUEUE_IGNORED.DAT
WATCH_SYS_SEARCH_FILES.DAT
MSA\$COMMANDS.COM
MSA\$STRINGS.TXT

All parameter and data files follow the same rules. Files are composed of free text, variables are defined inside the files with a dot «.» in column 0.

Here follows an extract of watch_sys.par file :

Notifications can be emails and / or OPCOM messages

```
.send_mail = n  
.notify_opcom = Y
```

Multiple destinations. Add one addressee line per destination. The limit is the maximum characters a DCL line may have.

And an extract of WATCH_SYS_MONITORED_PROCESS.DAT

This file contains the list of processes that should be present on the system.

Just enter the name of the process preceded with a ".".

Wildcard is allowed. Two parameters are allowed in that case, the number of processes expected and a kind of threshold. This threshold could be :

=, <, > or a range [x,y]

Eg

```
Th%spro*g|=4
```

Means 4 processes should exist on the system, not more, not less.

```
This*user|[3,7]
```

Means 3 to 7 process may exists

```
.swapper|=1
```

Watch_sys.com procedure.

There is nothing really special to say about it. The procedure accepts two parameters :

P1 : oneshot or ?/help/-h to get some infos about P1 and P2

P2 : the file specification of the main parameter file if the default one is not used.

If P1 contains the string oneshot, the procedure does one scan and then stops.

@watch_sys oneshot

To start the procedure in continuous run mode, with an alternate main parameter file, launch it with the command :

@watch_sys "" alternate_file_specification

It creates some temporary files that are cleaned automatically every 10 scans. These files will be created in sys\$login.

The main parameter file, watch_sys.par

By default, watch_sys.com searches this file in your Sys\$Login directory. Its location can also be specified as a parameter (P2).

@watch_sys P1 watch_sys_par_file_location

Parameters description :

scan_interval = hh:mm:ss

Defines the time interval the procedure waits before starting a new scan.

send_mail = n

notify_opcom = y

Notifications can be sent by email and / or OPCOM messages.

addressee = root@centos.itphonic.org

addressee = pierre@centos.itphonic.org

Multiple email destinations. Add one addressee line per destination. The limit is the maximum characters a DCL line may have.

check_memory = n (to be implemented)

check_errors = y

New hardware errors detection. All items reporting error in the errorlog will be monitored.

check_traffic_ip = n (to be implemented)

check_traffic_decnet_osi = n (to be implemented)

check_traffic_lat = n (to be implemented)

check_print_queue = y

check_batch_queue = y

Queues and jobs monitoring. Queue status are checked against the following status :
STOPPED/PAUSED/PAUSING/STALLED/STARTING/STOP_PENDING/STOPPING/CLOSED

Jobs status are checked against the status :

RETAINED/ABORTING/SUSPENDED

queue_ignored = y

queue_ignored_list = Sys\$Login:watch_sys_queue_ignored.dat

It's possible to ignore the monitoring of some queues.

check_ip_reachability = y

check_ip_reachability_hosts = Sys\$Login:watch_sys_ip_reachability_hosts.dat

Specify with these parameters ping tests to perform.

check_process = y

process_list = sys\$Login:watch_sys_monitored_process.dat

Specify the process that are required to run on the system. Wildcard names are allowed, also exact or range values are possible.

check_specific_files = y

check_specific_files_list = Sys\$Login:watch_sys_files.dat

Specify here files that should / shouldn't reside on the system.

check_shadows_state = y

check_free_disk_space = y

disk_free_space_value = 30%

disk_free_space_exceptions = sys\$login:watch_sys_disk_free_space_exceptions.dat

Parameters to test shadows (dsa devices) and free disk space. Once `disk_free_space_exceptions` is used, all disks to be monitored should be listed in that file. The free disk space values can be different for each day of the week.

check_cluster_votes = y

cluster_quorum = 2

Watch cluster votes. Get an alarm when a transition of the total number of votes in the cluster is detected. You can add an information about the quorum value the cluster needs to stay up & running. Give a value of 0 to this parameter to ignore it

check_logfail = y

Gives an alarm if « \$ show intrusion » reports something.

check_search_files = y

check_search_files_list = sys\$login:watch_sys_search_files.dat

Search string patterns in files.

external_proc = y

external_proc_files = Sys\$Login:watch_sys_external.dat

It is possible to start your own procedure(s) from `watch_sys`.

show_nodename = y

If the procedure runs on multiple nodes, add the nodename to messages.

watch_free_cpu = y

watch_free_cpu_value = 50

Monitor mean idle cpu. The system will be monitored during 30 seconds and an alarm will be issued when the mean idle cpu drops below watch_free_cpu_value.

check_susp_states = y

susp_loop_time = 00:00:30

check_com_states = y

max_com_proc = 2

Process in SUSP states can be monitored. Be warned that SUSP states includes all RW states too. When a process is found in this state, the procedure waits susp_loop_time seconds and re-test the state of the process. If the state is still SUSP a message will be issued if this process has not consumed cpu during the susp_loop_time seconds interval between the tests. Susp_loop_time should be expressed as hh:mm:ss.

Process in com sates can be watched too. Specify the maximum number of process waiting for the cpu the system may have before issuing a message.

check_max_process = y

max_process = 10

Max processes that are expected to be on the system. This test is valid if the mean free cpu test is performed because the number of processes comes from the data collected during 30 seconds (monitor) used to get the mean idle cpu %. Max process will be the mean value detected during the same "monitor system" of 30 seconds. Alarm issued if the mean value is bigger than the variable max_process.

msa\$util_use = y

msa\$strings = Sys\$Login:msa\$strings.txt

msa\$commands = Sys\$Login:msa\$commands.com

Monitor internal scsi/msa/sas devices. To use sys\$system:msa\$util.exe, specify in msa\$strings the patterns you would like search from the output of msa\$commands. Msa\$commands may contain commands for sas\$util..

Up to you to determine if one of the two utilities detects your internal raid controller and use the relevant one.

Heartbeat = y

heartbeat_time = 18:00

Watch_sys can send an email to addressees list everyday at ~ heartbeat_time.

run_forever = n

allowed_runtime = 09/10/11/12/13/14/16/17/18/19/20

The procedure scans can be disabled at some times. If you want to "freeze" the scans some hours, set the value run_forever to no, then specify in the variable allowed_runtime the hours when the procedure is allowed to run. The hours are separated by a "/" to form a list. If you want that the scans are allowed between 06:00 and 12:00 and 14:00 till 18:00, define allowed_runtime as 06/07/08/09/10/11/14/15/16/17

Watch_pagefile = y

Free_pagefile = 50%

Set parameter watch_pagefile to « y » if you want a monitoring of the pagefile usage. The procedure issues messages when the sum of all free pages is less than free_pagefile parameter expressed as the % of the sum of all installed pagefiles.

Other parameter files

Please see «templates»/exemple files coming with the procedure. Each file contains a little explanation and syntax rule(s).