How to…

Monitor TREX with alertMonitor.py

RETRIEVAL AND CLASSIFICATION (TREX) 6.0

PUBLIC

ASAP "How to…" Paper

Applicable Releases: TREX 6.0 SP1

July 2002

Copyright

Contents

1 ALERTMONITOR.PY – A MONITORING TOOL FOR TREX 3

2 FEATURES IN DETAIL 3

3 LOCATION OF THE PYTHON SCRIPT 4

4 OPTIONS AND HELP 4

5 CONFIGURATION OF INI FILE *ALERTMONITOR.INI* 4

6 START ALERTMONITOR.PY 5

7 EXAMPLES OF LOGGING AND TRACING OUTPUT 6

7.1 Example of Command Line Output 6

7.2 Example of E-Mail Sent by alertMonitor 7

1 alertMonitor.py – A Monitoring Tool for TREX

The Python script alertMonitor.py is a tool that consolidates the checking and control of multiple TREX servers and processes, and that checks TREX trace files for errors. It works under Windows 2000 and UNIX.

AlertMonitor.py reports the results in the command line and sends reports by e-mail. For archiving purposes, alertMonitor.py can also produce its ***own*** trace files. The information on available TREX servers, execution options, and so on, is read from a separate INI file. Detailed information on how to modify the provided alertMonitor.ini prototype for customizing alertMonitor.py can be found below.

2 Features in Detail

- Checks the availability of multiple **index servers**
- Checks one **HTTP server** using a URL such as http://localhost:8353/TREXHttpServer/TrexHttpServer.dll?CMD=PING
- Checks the availability of multiple **name servers**
- Checks the availability of multiple **queue servers**
- Checks the availability of multiple **preprocessor servers**
- Executes a **search** on one dedicated INI file-specified server with an INI file-specified index, query, maxresults parameter, and search timeout
- Executes a **ping** on multiple servers, using the ping command on every supported OS
- Inspects TREX **trace files** that have been modified since the last script run (“cycle”). Reports any errors with error occurrence dates since the last script run.
- Checks all **queues** on one dedicated INI file-specified queue server. If it finds queues in one of the statuses specified in the INI file, alertMonitor.py reports an error. Alternatively, queues can be specified if you do not want to inspect all queues on the queue server.
- **TREX OS processes** with INI file-specified names are periodically checked (“running or not”)
- The checking interval can be saved in the INI file, as well as the behavior of the script such as
    - Whether or not to attach the log(s) (the first log has a detailed list of error lines in trace files, and the second log is the trace file of alertMonitor itself)
    - E-mail address specifications for sending report e-mails
    - The behavior for sending mails: `Always` (default), `error,` or `error_and_once_a_day`
- The number of alertMonitor logs stored on the computer and the folder for storing them can be customized in the INI file. It is also possible to suppress the generation of logs completely.
- Some important parts of the options in the INI file are available using command line parameters that override INI file entries. See alertMonitor.py help (calling the script and command line options are described below in detail).

3 Location of the Python Script

You can find the Python script in the current TREX installation under:

<TREX directory>/python_support/text_tools/lib

- alertMonitor.py
- alertMonitor.ini

4 Options and Help

The command **`alertMonitor --help`** lists the available command line options. Alternatively, use **`python alertMonitor.py --help`**.

5 Configuration of INI File *alertMonitor.ini*

Before changing the alertMonitor.ini, make a backup copy so that you can switch to the default. Also make sure that you enter your e-mail address in the section [mail], parameter address.

Description of Sections From INI file *alertMonitor.ini*

Section	Description
[checks]	The **`checks`** section turns single tests **on** and **off**.
[servers]	The **`servers`** section lists the machines, separated by commas. 'localhost:<port>' entries are recognized. The servers under the **`server`** entry specify machines that will be pinged using the ping call on the command line.
[search]	There is a separate **`search`** section for a multi-threaded search on a single server, and the **`timeout`** (in seconds) is the maximum time span given to a thread to complete its search. It is very important to specify an existing **`index`**, as alertMonitor does not create indexes.
[queues]	From the **`queueserver`** entry, the first server is for finding queues in **`errorstatus`** (such as **`delayed`** or **`failed`**; comma-separated multiple status entries are permitted). If **`queuelist`** entry has no value, all queues are considered.
[process]	Enter **full** names of TREX OS processes here (such as **`ims_server_regis.exe`**, **`trexdaemon.exe`**). Note that Windows 2000, for example, truncates process names longer than 15 characters in the Task Manager and is case-sensitive to process names. AlertMonitor.py converts process names from OS and the INI file to lower case to make sure capitalization does not change the result of the check.
[tracefile]	The **`tracefile`** section is made of two entries: **`tracepath`** (paths with spaces are OK) and **`tracefile`** (restricts the inspection to trace files with names starting with given prefixes). Note that only trace files that changed within the "check interval" time span are checked, and only lines that include dates within this time span are parsed and included in the report.

Section	Description
[mail]	The **mail** section allows multiple comma-separated e-mail addresses (an invalid single address is recognized and does not affect other recipients). The **checkinterval** value is interpreted in minutes, and values such as 0.5 (= 30 seconds) are allowed. If the script completes the whole run ("cycle") in less than the check interval, it "sleeps" for the remaining time. During this time, it can be stopped with the "Ctrl-C" Python keyboard interrupt stroke. The **sendmail** flag should be **always, error,** or **error_and_once_a_day**. The **attachlog** flag only has an impact if an e-mail is sent, the **makelog** flag in the **logging** section (see below) is set to **yes,** and a valid **logpath** was entered either by command line or into the INI file (see below)
[logging]	The **logpath** value in the **logging** section allows backslashes and blanks. If omitted while **makelog** is set to **yes,** no logs are written. The default **logpath** value in the alertMonitor.ini file is C:\Program Files\SAP\TREX_6\trace\ without quotation marks (as only one value is expected for **logpath**, blanks are permitted).

6 Start alertMonitor.py

After modifying alertMonitor.ini, **execute** alertMonitor**: `alertMonitor.py`**

The lines produced by alertMonitor are preceded by identifiers in squared brackets but **without commas**. Identifiers help to localize the execution phase in which any possible errors have occurred. You will not see most of alertMonitor.py output if you have called alertMonitor with **`--showoutput=0`**.

Note that alertMonitor.py runs ("cycles") are separated by horizontal lines.

7 Examples of Logging and Tracing Output

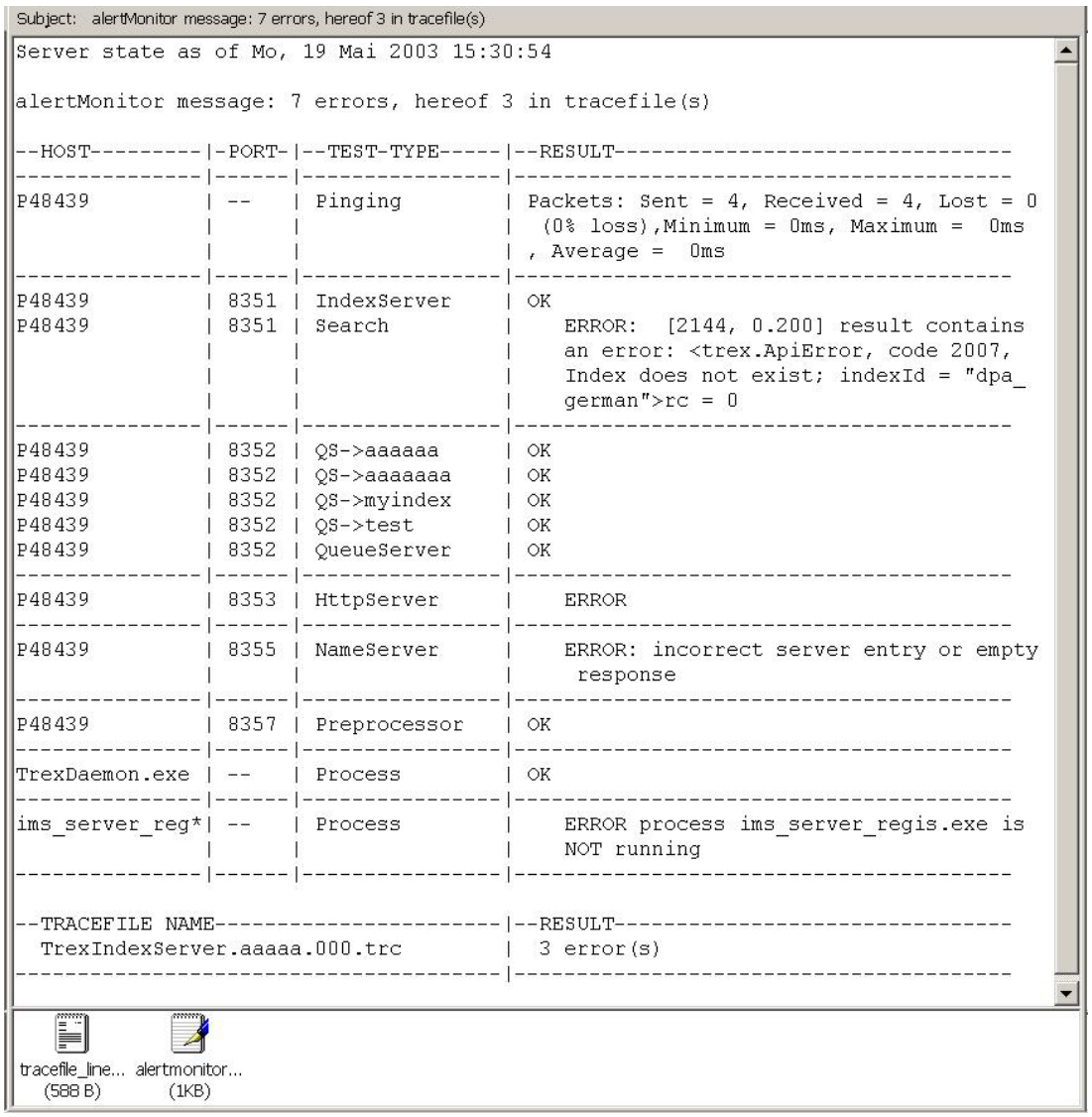
7.1 Example of Command Line Output

The following example shows how the output should look (execution was stopped with Ctrl-C after the first cycle):

```
C:\perforce\workplace\ims\dev\nutest\lib>alertMonitor --inifile=alertMonitor_01.ini
[2496, 0.050] >> starting alertMonitor (at Mon May 19 15:25:19 2003)
[2496, 0.050] args: ()
[2496, 0.060] keys: {'inifile': 'alertMonitor_01.ini'}
[test002b]  using INI file C:\perforce\workplace\ims\dev\nutest\lib\alertMonitor_01.ini
[iniF002 ]  extracted checking interval from ini file; its value's 0.5 minutes
[iniF003A]  time given to search thread to complete: 5 (self.timeout)
[iniF007 ]  1. searchServer:P48439:8351; 1. searchIndex:dpa_german
[iniF008 ]  search.maxresults in ini file:random, finally computed to 55
[iniF010 ]  search cmd.:"search.py --indexServer=P48439:8351 --query=* -
                       -maxResults=55 --cmethod=t --indexId=dpa_german"
------------------------------------------------------------
[http_e2 ]  ERROR with monitoring httpserver
http://P48439:8353/TREXHttpServer/TrexHttpServer.dll/?CMD=PING
[inds002 ]  indexserver: P48439:8351 OK
[names002]  nameserver: P48439:8355 ERROR: incorrect server entry or empty response
[prepr001]  preprocessor: P48439:8357 OK
[proces_2]  process TrexDaemon.exe is running
[proces_2]  ERROR process ims_server_regis.exe is NOT running
[ques002 ]  queueserver: P48439:8352 OK
[thrd0   ]  created, time: Mo, 19 Mai 2003 15:25:24 +0000
[search1 ]  thread0 started, given 5 seconds to complete
[search3 ]  [2492, 0.201] result contains an error: <trex.ApiError, code 2007, Index does not
exist; indexId = "dpa_german">
[search4 ]  search on  : P48439:8351                          : problems
[ping001 ]  ping P48439
                Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
                Minimum = 0ms, Maximum =  0ms, Average =  0ms
----------SENDING MAIL----------
[mail001 ]  sendmailFlag set to always
[mail008 ]  sending to myaccount@mycompany.com successful
[sleep03 ]  sleeping 16 s  after work (default check interval is 30 s )
------------------------------------------------------------
[sleep_E2]  exiting (keyboard interrupt while waiting for next check cycle)
```

Of course, your command line will differ in content and values. However, the e-mail report provides a better and more convenient overview of results and logs.

7.2 Example of E-Mail Sent by alertMonitor



Subject: alertMonitor message: 7 errors, hereof 3 in tracefile(s)

```
Server state as of Mo, 19 Mai 2003 15:30:54

alertMonitor message: 7 errors, hereof 3 in tracefile(s)

--HOST---------|-PORT-|--TEST-TYPE-----|--RESULT-------------------------------
---------------|------|----------------|---------------------------------------
P48439         | --   | Pinging        | Packets: Sent = 4, Received = 4, Lost = 0
               |      |                |  (0% loss),Minimum = 0ms, Maximum =  0ms
               |      |                | , Average =  0ms
---------------|------|----------------|---------------------------------------
P48439         | 8351 | IndexServer    | OK
P48439         | 8351 | Search         |    ERROR:  [2144, 0.200] result contains
               |      |                |    an error: <trex.ApiError, code 2007,
               |      |                |    Index does not exist; indexId = "dpa_
               |      |                |    german">rc = 0
---------------|------|----------------|---------------------------------------
P48439         | 8352 | QS->aaaaaa     | OK
P48439         | 8352 | QS->aaaaaaa    | OK
P48439         | 8352 | QS->myindex    | OK
P48439         | 8352 | QS->test       | OK
P48439         | 8352 | QueueServer    | OK
---------------|------|----------------|---------------------------------------
P48439         | 8353 | HttpServer     |    ERROR
---------------|------|----------------|---------------------------------------
P48439         | 8355 | NameServer     |    ERROR: incorrect server entry or empty
               |      |                |     response
---------------|------|----------------|---------------------------------------
P48439         | 8357 | Preprocessor   | OK
---------------|------|----------------|---------------------------------------
TrexDaemon.exe | --   | Process        | OK
---------------|------|----------------|---------------------------------------
ims_server_reg*| --   | Process        |    ERROR process ims_server_regis.exe is
               |      |                |    NOT running
---------------|------|----------------|---------------------------------------

--TRACEFILE NAME-----------------------|--RESULT-------------------------------
  TrexIndexServer.aaaaa.000.trc        |  3 error(s)
---------------------------------------|---------------------------------------
```

tracefile_line... (588 B) alertmonitor... (1KB)

As well as the information contained in the e-mail body, the e-mail has the following two attachments:

- A text file that contains lines describing errors from TREX trace file(s) modified during the “checkinterval” time since the start of the alertMonitor run summarized by the e-mail in question
- The log of the run that produced the results (this means that you can reexamine an execution cycle without scrolling through the command line output)

Note the following about the example e-mail

- The sender (“From” field entry) indicates the PC (“P48439”) on which the script was executed
- The subject (“TO DO”) provides information on the success/failure of the run
- Results are sorted in descending order by host, then by port, then by test type
- The first attachment is named tracefile_lines_with_errors.txt
- The second attachment is the log of the summarized alertMonitor cycle, not the log of the entire script since it was started
  The extension is **.amt.trc**, whereby amt means alertMonitor trace. The name is structured as follows: `<day><month><year>_<hour>.<minute>.<second>.amt.trc scheme`. You can open the **trc** extension with any text editor: alertMonitor tracefiles are raw text.

Technical example: If the check interval corresponds to 10 minutes and the alertMonitor starts its 5th cycle at 11:10:09 am (local time), the e-mail summarizing this 5th cycle includes trace files checked if they were modified after 11:00:09, and the e-mail attachment only includes the extracted trace file lines that have a timestamp after 11:00:09 local time.