|  |
| --- |
| **Government of Ontario** |
| **Eraser™: User Manual** |
| **Technical Support Office** |

|  |
| --- |
| **Version: 1.2**  **4/8/2013** |

©2013 The Technical Support Office. All rights reserved.

This documentation is furnished *AS IS* and is subject to change without notice and therefore should not be construed as a commitment by the Technical Support Office. This documentation may *not* be copied, modified or distributed in any way without the express authorization of the Technical Support Office. This documentation may be used only in connection with the Technical Support Office’s products and services. Use, duplication, reproduction, release, modification, disclosure or transfer of this documentation is restricted in accordance with the Government of Ontario’s policies and regulations. This documentation qualifies as computer software documentation and any use shall be governed solely by these terms. All other use is strictly prohibited. The Technical Support Office assumes no responsibility or liability for any errors or inaccuracies that appear in this documentation.

# Revisions

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Description | Owner (initials) |
| December 7, 2012 | 0.1 | Initial version | TSO (hp) |
| January 22, 2013 | 0.5 | Updated content | TSO (hp) |
| February 20, 2013 | 0.9 | Updated content | TSO (hp) |
| March 07, 2013 | 1.0 | Ready to be published | TSO (hp) |
| March 13, 2013 | 1.1 | Updated content (config.properties samples) | TSO (hp) |
| April 08, 2013 | 1.2 | Changed from /opt/tso/sfw to /opt/tso/opt | TSO (hp) |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Contents

[Revisions 2](#_Toc351365263)

[Introduction 4](#_Toc351365264)

[User Manual 4](#_Toc351365265)

[Features 4](#_Toc351365266)

[Installation 5](#_Toc351365267)

[Instructions 5](#_Toc351365268)

[Configuration 6](#_Toc351365269)

[The Global Definition 7](#_Toc351365270)

[Global Definition Reference Table 8](#_Toc351365271)

[The Job Definition 9](#_Toc351365272)

[Job Definition Reference Table 10](#_Toc351365273)

[Deployment 12](#_Toc351365274)

[Content Package 12](#_Toc351365275)

[Requirements 12](#_Toc351365276)

[Typical Deployment 12](#_Toc351365277)

[Examples 13](#_Toc351365278)

[Shared Log Directory with Similar Log Formats 13](#_Toc351365279)

[Individual Log Directories with Unique Log Formats 16](#_Toc351365280)

[Logging 20](#_Toc351365281)

[Example 21](#_Toc351365282)

[Resources and References 23](#_Toc351365283)

[Regular Expressions Reference Table 23](#_Toc351365284)

[Special Characters Reference Table 24](#_Toc351365285)

# Introduction

## User Manual

This user manual contains instructions for the configuration and deployment of the TSO’s in-house **Eraser** software. This user manual describes the functionality of, and uses output and screenshots from version 3.4 of the software.

There are several versions of the *“User Manual”*, therefore, you should refer to the version appropriate for the Eraser variant you are configuring and deploying.

## Features

Eraser is a flexible log management and rotation tool that is suitable for running from crontab. Notable features include:

* Flexible configuration
  + Global and local directives concept
  + Job definition isolation
* Utilize powerful regular expression for filenames matching
* Can be executed as a non-privileged user
  + Files can be compressed under the original owner by utilizing sudo
* Produce verbose output (for logging and auditing purpose)
* Utilize GNU Parallel for concurrency and speed
  + Fully take advantage of multiple threads architect
* Minimum requirements: only require Perl and Korn shell.
  + Should work on multiple UNIX variants with minimum changes

# Installation

Installing Eraser is as simple as following the three steps below:

* Extract the installation package to a directory
* Editing the config.properties file to fit your requirements
* Schedule a job in the O/S crontab for the automatic execution of run.sh

## Instructions

1. Upload the installation package to the target UNIX server (at the time this documentation is written, the latest Eraser package is at version 3.4; hence, the full filename is **eraser\_3.4.tar.gz**)
2. Extract the installation package.

|  |
| --- |
| gzip -dc eraser\_3.4.tar.gz|tar xvfp - |
| x eraser, 0 bytes, 0 tape blocks  x eraser/VERSION, 3 bytes, 1 tape blocks  x eraser/RELEASE, 976 bytes, 2 tape blocks  x eraser/parallel.pl, 157885 bytes, 309 tape blocks  x eraser/run.sh, 18563 bytes, 37 tape blocks  x eraser/README, 579 bytes, 2 tape blocks |

1. Verify that both the files **run.sh** and **parallel.pl** have executable bits set.

|  |
| --- |
| ls -l eraser |
| total 563  -rw-r--r-- 1 tsoadm tsoadm 579 Dec 19 14:51 README  -rw-r--r-- 1 tsoadm tsoadm 976 Dec 21 15:37 RELEASE  -rw-r--r-- 1 tsoadm tsoadm 3 Dec 21 15:37 VERSION  -rwxr-xr-- 1 tsoadm tsoadm 157885 Aug 22 2012 parallel.pl  -rwxr-xr-- 1 tsoadm tsoadm 18563 Dec 21 15:35 run.sh |

1. Invoke touch to create an empty file called **config.properties**. This file will contain the configuration settings of the software.

|  |
| --- |
| touch config.properties |

1. Follow the [**Configuration**](#_Configuration) chapter to modify the **config.properties** file to meet your requirements. (The next chapter describes how to create and edit the configuration file in details)
2. Modify the crontab to schedule daily execute of the software.

For example, to set Eraser to run at 4am every morning, add the following entries to the crontab.

|  |
| --- |
| # perform log rotation  0 4 \* \* \* cd /opt/eraser && ./run.sh >log/eraser\_cronjob\_`date +\%Y\_\%m\_\%d-\%T` 2>&1 |

# Configuration

The **config.properties** file is the main configuration file of the Eraser software. It comprises two primary sections:

1. [*The Global Definition*](#_The_Global_Definition)
2. [*The Job Definition*](#_The_Job_Definition)

The Global Definition defines the default behaviour of the software during start-up, whereas the Job Definition defines the jobs to be executed during runtime. A sample template of the **config.properties** file looks similar to the one below:

|  |
| --- |
| ######################  ## GLOBAL DEFINITON ##  ######################  >> global directives go here |
| ###############  ## FIRST RUN ##  ###############  . $RUN --first-time |
| ###################  ## JOB DEFINITON ##  ###################  >> a new job definition goes here  . $RUN --execute-job  ###################  ## JOB DEFINITON ##  ###################  >> another job definition goes here  . $RUN --execute-job |
| ##############  ## LAST RUN ##  ##############  . $RUN --last-time |

## The Global Definition

The global directives defined within Global Definition set the default behaviour of the software during start-up. However, these directives can be individually overridden by the job directives defined within the Job Definition section (this will be explained later in greater details.)

For example, if the directive **JOB\_DRY\_RUN** is set to *true* and **GLOBAL\_DRY\_RUN** is set to *false,* then final resultant behaviour of the Job Definition in question will use the value defined by **JOB\_DRY\_RUN**, which is true.

The **Global Definition** **Reference Table** below lists all of the possible directives and the associated values and descriptions available to the users.

Note the following:

* If the global directive is marked as “*(mandatory)*” then it has to be defined prior to execution.
* If the global directive is marked as “*(optional)*” then it does *not* have to be defined prior to execution; the default behaviour – which is noted in under **Descriptions** – will be used.

### Global Definition Reference Table

|  |  |  |
| --- | --- | --- |
| Global Directives | Possible Values | Descriptions |
| GLOBAL\_RET\_DELETE | **(mandatory)** [int]  1 or greater | Set the numbers of days (retention period) before deletion is performed. |
| GLOBAL\_RET\_COMPRESS | **(mandatory)** [int]  1 or greater | Set the numbers of days (retention period) before compression is performed. |
| GLOBAL\_PARALLEL\_BIN | **(mandatory)** [string]  path to binary | Set the path to the GNU Parallel binary. |
| GLOBAL\_VERBOSE\_LOGGING | **(optional)** [int]  1 to 3 | If set greater than 0, then enable verbose logging; bigger number increases the verboseness of the logs. Default to 0. |
| GLOBAL\_DRY\_RUN | **(optional)** [boolean]  true or false | If set to true, do not perform any actions against the files; only output what will be done during the actual run. Default to false. |
| GLOBAL\_CHILD\_PROCESS | **(optional)** [int]  1 or greater | Set the number of child process to spawn during the compression process. Default to 4. |
| GLOBAL\_NICENESS | **(optional)** [int]  0 to 19 | Set the niceness or *“processing priority”* of the software during execution (0 for normal priority and 19 for the lowest priority). Default to 19. |
| GLOBAL\_RECURSIVE | **(optional)** [boolean]  true or false | If set to true, enable recursive search (please only enable if you know what are you doing!); does not follow symbolic link. Default to false. |
| GLOBAL\_SUDO\_BIN | **(optional)** [string]  path to binary | Set the path to the SUDO binary. Default to “sudo”. |
| GLOBAL\_USE\_SUDO | **(optional)** [boolean]  true or false | If set to true, enable the use of the SUDO. Default to false. |
| GLOBAL\_ENABLED | **(optional)** [boolean] true or false | If set to true, the defined jobs to be enabled by default. Default to true. |
| GLOBAL\_FLUSH | **(optional)** [boolean]  true or false | If set to true, enable system flushing prior to execution. Default to true. |
| GLOBAL\_DIR\_QUEUE | **(optional)** [string]  One directory or a list of a directories | If defined, set a list of directories where the log files will be searched. Optional since this value can be defined per job definition |
| GLOBAL\_FILE\_PATTERN | **(optional)** [string]  A regular expression or a list of regular expressions (also accept extended RE) | If defined, set a list of regular expressions to be used to search for the log files. The regular expressions will be applied against not only the filename but also the file’s path as well. Optional since this value can be defined per job definition |

## The Job Definition

Each job definition must be defined *individually*. The first job definition should be defined between the $RUN --first-time and the $RUN --execute-job lines; the next job definition follows and so on and so forth. Each job definition is separated by the $RUN --execute-job line. The end of the **config.properties** file is marked by the $RUN --last-time line as illustrated below.

|  |
| --- |
| . $RUN --first-time  **# define a new job definition here**  . $RUN --execute-job  **# another job definition should go here**  . $RUN --execute-job  . $RUN --last-time |

During execution, the job definitions are independent of each other: overridden global directives are only retained for the duration of their job definition only. This architect gives the users the greatest flexibility because each job definition or a set of job definitions can be disabled or set with different behaviours without affecting the neighbouring job definitions.

There is no limit to the amount of jobs can be defined in the **config.properties** file. Obviously, more job definitions will result in longer execution.

The **Job Definition** **Reference Table** below lists all of the possible directives, the associated values and descriptions available to the users.

Note the following:

* If the job directive is marked as “*(mandatory)*” then it has to be defined prior to execution.
* If the job directive is marked as “*(optional)*” then it does *not* have to be defined prior to execution; the default behaviour – which is noted in under the **Descriptions** column – will be used.

### Job Definition Reference Table

|  |  |  |
| --- | --- | --- |
| Job Definition Directives | Possible Values | Descriptions |
| JOB\_USER | **(mandatory)** [string] username | When defined, only the log files owned by JOB\_USER will be searched.  Additionally, if the SUDO feature is enabled (either globally or per job definition), the software will be executed as defined by JOB\_USER (via SUDO). |
| JOB\_DIR\_QUEUE | **(mandatory)** [string]  One directory or a list of a directories | Set a list of directories where the log files will be searched. Mandatory if not set in the Global Section |
| JOB\_FILE\_PATTERN | **(mandatory)** [string]  A regular expression or a list of regular expressions (also accept extended RE) | When defined, set a list of regular expressions to be used to search for the log files. The regular expressions will be applied against not only the filename but also the file’s path as well. Mandatory if not set in the Global Section |
| JOB\_PARALLEL\_BIN | **(optional)** [string]  path to binary | Set the path to the GNU Parallel binary. |
| JOB\_RET\_DELETE | **(optional)** [int]  1 or greater | Set the numbers of days (retention period) before deletion is performed. Optional: if set, this value will override GLOBAL\_RET\_DELETE. |
| JOB\_RET\_COMPRESS | **(optional)** [int]  1 or greater | Set the numbers of days (retention period) before compression is performed. Optional: if set, this value will override GLOBAL\_RET\_COMPRESS. |
| JOB\_VERBOSE\_LOGGING | **(optional)** [int]  1 to 3 | If set greater than 0, then enable verbose logging; bigger number increases the verboseness of the logs. Default set by GLOBAL\_VERBOSE\_LOGGING |
| JOB\_DRY\_RUN | **(optional)** [boolean] true or false | If set to true, do not perform any actions against the files; only output what will be done during an actual run. Default set by GLOBAL\_DRY\_RUN |
| JOB\_CHILD\_PROCESS | **(optional)** [int]  1 or greater | Set the number of child process to spawn during the compression process. Default set by GLOBAL\_CHILD\_PROCESS |
| JOB\_NICENESS | **(optional)** [int]  0 to 19 | Set the niceness or *“processing priority”* of the software during execution (0 for normal priority and 19 for the lowest priority). Default set by GLOBAL\_NICENESS |
| JOB\_RECURSIVE | **(optional)** [boolean]  true or false | If set to true, enable recursive search (please only enable if you know what are you doing!); does not follow symbolic link. Default set by GLOBAL\_RECURSIVE |
| JOB\_SUDO\_BIN | **(optional)** [boolean]  true or false | Set the path to the SUDO binary. Default set by GLOBAL\_SUDO\_BIN |
| JOB\_USE\_SUDO | **(optional)** [boolean] true or false | If set to true, enable the use of the SUDO. Default set by GLOBAL\_USE\_SUDO |
| JOB\_ENABLED | **(optional)** [boolean] true or false | If set to true, the defined jobs to be enabled. Default set by GLOBAL\_ENABLE |

# Deployment

## Content Package

The Eraser software comprises three primary parts, or files:

|  |  |
| --- | --- |
| File | Descriptions |
| run.sh | * Executable binary * Read-only |
| parallel.sh | * Helper binary * Read-only |
| config.properties | * Configuration File * To be modified by users |

During deployment, you should only modify the **config.properties** file; the other two files are not meant to be edited, unless under special circumstances such as in the event that the **ksh** and the **perl** binaries are located in somewhere (non-POSIX standard) other than the /usr/bin directory.

## Requirements

Perl 5.0 or newer and Korn88 or newer are required to run Eraser. The software assumes that the **perl** and **korn** binaries are both located under the /usr/bin directory. In the event that this is not the case, you will have to manually modify the run.sh and config.properties files to reflect the special requirements.

## Typical Deployment

In order to fully utilize the full potential of Eraser, it is highly recommended that the software should be deployed under an UNIX account that has the ability to *“sudo”* to other account or accounts it is configured to monitor. Typically, the software should reside in the /opt directory to comply with the POSIX guidelines.

We also recommend that the **sudoers** rules to be configured with the *NOPASSWD* option. This software will not function properly if a password input is prompted during execution, especially without a tty or an interactive terminal (e.g. invoked by cron).

Prior to running the software the first time, remember to set either **GLOBAL\_DRY\_RUN** or the **JOB\_DRY\_RUN** to true; enabling this feature allows you to test the software first before any permanent changes are made.

# Examples

Two real world scenarios will be presented in this chapter which, hopefully, will get you up and running quickly.

## Shared Log Directory with Similar Log Formats

In our first scenario, there are a total of three application users – **appone**, **apptwo**, **appthree** – that share the same parent, log directory, /opt/logs, with similar log formats.

1. Within the parent directory /opt/logs, **appone**, **apptwo** and **appthree** have their own log directories, /opt/logs/appone, /opt/logs/apptwo and /opt/logs/appthree, respectively.
2. The logs produced by the three respective applications have the same format: **stdout\_<app\_id>.log\_<YYY><MM><DD>**
   * For examples:
     + /opt/logs/appone/stdout\_appone.log\_20121242
     + /opt/logs/apptwo/stdout\_apptwo.log\_20110512

The *requirements* for our illustrated scenario are as follow:

1. The log files need to be compressed after 30 days
2. The log files need to be deleted after 60 days
3. Enable sudo to only compress files are owned by the original owners
4. Enable recursion since the log files share the same parent directory

To tackle the above requirements, our **config.properties** file should look similar to the example below:

[Sample Configuration A-1]

|  |
| --- |
| ######################  ## GLOBAL DEFINITON ##  ######################  # (int) delete if older than GLOBAL\_RET\_DELETE days  GLOBAL\_RET\_DELETE=60  # (int) compress if older than GLOBAL\_RET\_COMPRESS days  GLOBAL\_RET\_COMPRESS=30  # (string) path to the GNU parallel binary  GLOBAL\_PARALLEL\_BIN="$CUR\_DIR"/parallel.pl  # (boolean) enable/disable the use of sudo  GLOBAL\_USE\_SUDO=true  # (directory) directory where all of the jobs defined below should search against  GLOBAL\_DIR\_QUEUE='  /opt/logs  '  # (regex) regex to be globally applied against the full path and file names  GLOBAL\_FILE\_PATTERN='  stdout\_app(one|two|three)\.log\_\d{8}(\.gz)?$  '  ###############  ## FIRST RUN ##  ###############  RUN="$0"  . $RUN --first-time  ###################  ## JOB DEFINITION ##  ####################  JOB\_USER=appone  . $RUN --execute-job  ####################  ## JOB DEFINITION ##  ####################  JOB\_USER=apptwo  . $RUN --execute-job  ####################  ## JOB DEFINITION ##  ####################  JOB\_USER=appthree  . $RUN --execute-job  ##############  ## LAST RUN ##  ##############  . $RUN --last-time |

Most of the directives presented in the sample **config.properties** file such as **GLOBAL\_RET\_DELETE** and **GLOBAL\_RET\_COMPRESS** are self-explanatories. However, if the details are required, you can refer to the [Global Definition Reference Table](#_Global_Definition_Reference) chapter:

* The **GLOBAL\_DIR\_QUEUE** has one directory listed: /opt/logs
  + Since **GLOBAL\_RECURSIVE** is set to true (to be enabled), we do not need to list the subdirectories (e.g. /opt/logs/appone, /opt/logs/apptwo, etc…)
* The **GLOBAL\_FILE\_PATTERN** contains the regular expression of the log file format.

Below, we will briefly explain the regular expression found in the example:

|  |  |
| --- | --- |
| Example Log File Format | Regular Expression |
| stdout\_appone.log\_20121242 | stdout\_app(one|two|three)\.log\_\d{8}(\.gz)?$ |

* 1. The phrase follows **“stdout\_app”,** can either be **one**, **two** or **three**; we achieve this match by forming the following expression: (one|two|three)
  2. The dot or period (.) is escaped properly since it is a meta-character (as per the [Regular Expression Reference Table](#_Regular_Expressions_Reference) below, the dot represent any character (except newline)
  3. The phrase follows **“log\_”**, contains a total of 8 digits; we achieve this match by forming the following expression: \d{8}
  4. The last regular expression, (\.gz)?, indicates that after the log file has been compressed, there will be a **“.gz”** extension at the end of. We need to be able to tag this change so that the target log file can either be compressed (when the .gz extension exists) or deleted (when .gz extension does not exist).
  5. The dollar sign ($) at the end of the regular expression indicates the end of the file name match

## Individual Log Directories with Unique Log Formats

In our second scenario, there are a total of three application users – **appone**, **apptwo**, **appthree** – that do not share the same parent log directory. In addition, their log files have unique formats.

1. The application users, **appone**, produces logs under /opt/appone/log; **apptwo** produces logs under /var/log; and **appthree** produces logs under /export/home/appthree/log
2. The example file formats of the three applications are as follow:

|  |  |
| --- | --- |
| App ID | Sample Log File |
| appone | /opt/appone/logs/stdout\_appone.log\_20121242 |
| apptwo | /var/log/apptwo.1 |
| appthree | /export/home/appthree/log/trace.2013-03-28 |

The *requirements* for our illustrated scenario are as follow:

For **appone** and **apptwo**:

* The log files need to be compressed after 10 days
* The log files need to be deleted after 365 days

For **appthree**:

* The log files need to be compressed after 30 days
* The log files need to be deleted after 60 days

For **all** jobs:

* Enable sudo to only compress files are owned by the original owners
* Do not enable recursion

To tackle the above requirements, our **config.properties** file should look similar to the example below:

[Sample Configuration A-2]

|  |
| --- |
| ######################  ## GLOBAL DEFINITON ##  ######################  GLOBAL\_RET\_DELETE=365  GLOBAL\_RET\_COMPRESS=10  GLOBAL\_PARALLEL\_BIN="$CUR\_DIR"/parallel.pl  GLOBAL\_USE\_SUDO=true  ###############  ## FIRST RUN ##  ###############  RUN="$0"  . $RUN --first-time  ###################  ## JOB DEFINITION ##  ####################  JOB\_USER=appone  JOB\_DIR\_QUEUE='  /opt/appone/log  '  JOB\_FILE\_PATTERN='  stdout\_appone\.log\_\d{8}(\.gz)?$  '  . $RUN --execute-job  ####################  ## JOB DEFINITION ##  ####################  JOB\_USER=apptwo  JOB\_DIR\_QUEUE='  /var/log  '  JOB\_FILE\_PATTERN='  apptwo\.\d(\.gz)?$  '  . $RUN --execute-job  ####################  ## JOB DEFINITION ##  ####################  JOB\_USER=appthree  JOB\_DIR\_QUEUE='  /export/home/appthree/log  '  JOB\_FILE\_PATTERN='  trace\.log\.\d{4}-\d{2}-\d{3}(\.gz)?$  '  JOB\_RET\_DELETE=60  JOB\_RET\_COMPRESS=30  . $RUN --execute-job  ##############  ## LAST RUN ##  ##############  . $RUN --last-time |

Most of the directives presented in the sample **config.properties** file are self-explanatories. However, if the details are required, you can refer to the [Global Definition Reference Table](#_Global_Definition_Reference) chapter:

Unlike the previous example (Sample Configuration A-1), both the directives, **GLOBAL\_DIR\_QUEUE** and **GLOBAL\_FILE\_PATTERN** do not exist since the three follow job definitions do not share anything in common

Below, we will briefly explain the three regular expressions found in the sample:

|  |  |
| --- | --- |
| Example Log File Format | Regular Expression |
| stdout\_appone.log\_20121242 | stdout\_appone\.log\_\d{8}(\.gz)?$ |

1. The dot or period (.) is escaped properly since it is a meta-character (as per the [Regular Expression Reference Table](#_Regular_Expressions_Reference) below, the dot represent any character (except newline)
2. The phrase follows **“log\_”**, contains a total of 8 digits; we achieve this match by forming the following expression: \d{8}
3. The last regular expression, (\.gz)?, indicates that after the log file has been compressed, there will be a **“.gz”** extension at the end of. We need to be able to tag this change so that the target log file can either be compressed (when the .gz extension exists) or deleted (when .gz extension does not exist).
4. The dollar sign ($) at the end of the regular expression indicates the end of the file name match

|  |  |
| --- | --- |
| Example Log File Format | Regular Expression |
| apptwo.1 | apptwo\.\d(\.gz)?$ |

1. The dot or period is escaped properly since it is a meta-character
2. The phrase follows **“apptwo\.”** contains a single digit: we achieve this match by forming the following expression: \d
3. The last regular expression, (\.gz)?, indicates that after the log file has been compressed, there will be a **“.gz”** extension at the end of. We need to be able to tag this change so that the target log file can either be compressed (when the .gz extension exists) or deleted (when .gz extension does not exist).
4. The dollar sign ($) at the end of the regular expression indicates the end of the file name match

|  |  |
| --- | --- |
| Example Log File Format | Regular Expression |
| trace.2013-03-28 | trace\.log\.\d{4}-\d{2}-\d{3}(\.gz)?$ |

1. The dot or period is escaped properly since it is a meta-character
2. The phrase follows **“log\.”** contains multiple digits in a format of YYY-MM-DD: we achieve this match by forming the following expression: \d{4}-\d{2}-\d{3}
3. The last regular expression, (\.gz)?, indicates that after the log file has been compressed, there will be a **“.gz”** extension at the end of. We need to be able to tag this change so that the target log file can either be compressed (when the .gz extension exists) or deleted (when .gz extension does not exist).
4. The dollar sign ($) at the end of the regular expression indicates the end of the file name match

At the end of the sample, notice that the job definition for **appthree** contains two extra directives: **JOB\_RET\_DELETE** and **JOB\_RET\_COMPRESS**. They exist for the sole purpose of overriding the deletion and compression retentions original set in the global definition.

# Logging

The chapter briefly explains the meaning of the logs produced by the Eraser software and how to interpret them for troubleshooting purpose.

During deployment, it is recommended that you redirect both of the software’s **stdout** and **stderr** outputs to a log file. Below is an example of an already configured line in crontab:

|  |
| --- |
| # perform log rotation  0 4 \* \* \* cd /opt/eraser && ./run.sh >log/eraser\_cronjob\_`date +\%Y\_\%m\_\%d-\%T` 2>&1 |

## Example

In our case above, both output are redirected to a log file called **eraser\_cronjob\_YYY\_MM\_DD\_(%time)**. After the initial run, view the said log should give you an output similar to the excerpts below:

|  |
| --- |
| <Mar 13, 2013 04:00:00 AM EDT> <Notice> <server123> Starting TSO Log Management utility version 3.4.  <Mar 13, 2013 04:00:00 AM EDT> <Notice> <server123> The directive GLOBAL\_RET\_DELETE is set to "365".  <Mar 13, 2013 04:00:00 AM EDT> <Notice> <server123> The directive GLOBAL\_RET\_COMPRESS is set to "120".  **1**  <Mar 13, 2013 04:00:00 AM EDT> <Notice> <server123> The directive GLOBAL\_DRY\_RUN is set to "false".  <Mar 13, 2013 04:00:00 AM EDT> <Notice> <server123> The directive GLOBAL\_CHILD\_PROCESS is set to "4".  <Mar 13, 2013 04:00:00 AM EDT> <Notice> <server123> The directive GLOBAL\_PARALLEL\_BIN is set to "./parallel.pl".  <Mar 13, 2013 04:00:00 AM EDT> <Notice> <server123> The directive GLOBAL\_NICENESS is set to "19".  <Mar 13, 2013 04:00:00 AM EDT> <Notice> <server123> The directive GLOBAL\_RECURSIVE is set to "true".  <Mar 13, 2013 04:00:00 AM EDT> <Notice> <server123> The directive GLOBAL\_SUDO\_BIN is set to "/usr/local/bin/sudo".  <Mar 13, 2013 04:00:00 AM EDT> <Notice> <server123> The directive GLOBAL\_USE\_SUDO is set to "true".  <Mar 13, 2013 04:00:00 AM EDT> <Notice> <server123> The directive GLOBAL\_ENABLED is set to "true".  <Mar 13, 2013 04:00:00 AM EDT> <Notice> <server123> The directive GLOBAL\_VERBOSE\_LOGGING is set to "0".  <Mar 13, 2013 04:00:01 AM EDT> <Notice> <server123> The directive GLOBAL\_FLUSH is set to "true".  … (more GLOBAL directives go here) |
| <Mar 13, 2013 04:00:01 AM EDT> <Notice> <server123> Forcing system to flush unwritten buffers (this may take a couple of seconds).  <Mar 13, 2013 04:00:06 AM EDT> <Notice> <server123> <job:1> <appone> Override found: the new value for JOB\_USER is now "appone".  **2**  <Mar 13, 2013 04:00:06 AM EDT> <Notice> <server123> <job:1> < appone> Processing job number 1 for "appone " user.  <Mar 13, 2013 04:00:06 AM EDT> <Notice> <server123> <job:1> < appone> <delete> Searching directory "/opt/logs"...  <Mar 13, 2013 04:00:06 AM EDT> <Notice> <server123> <job:1> < appone> <delete> No match (RC=0). Skipping to the next directory in the list.  <Mar 13, 2013 04:00:06 AM EDT> <Notice> <server123> <job:1> < appone> <compress> Searching directory "/opt/logs"...  <Mar 13, 2013 04:00:07 AM EDT> <Notice> <server123> <job:1> < appone> <compress> Applying the matching patterns against the files in the directory...  <Mar 13, 2013 04:00:11 AM EDT> <Notice> server123> <job:1> < appone> <compress> Found 1 file(s) that are older than 120 days. Proceed to compress the files...  … (more JOB results go here) |
| <Mar 13, 2013 04:01:10 AM EDT> <Notice> <server123> No more items in the queue.  <Mar 13, 2013 04:01:10 AM EDT> <Notice> <server123> Forcing system to flush unwritten buffers (this may take a couple of seconds).  **3**  <Mar 13, 2013 04:01:14 AM EDT> <Notice> <server123> A total of 5 jobs ran.  <Mar 13, 2013 04:01:14 AM EDT> <Notice> <server123> A total of 1000 files have been deleted.  <Mar 13, 2013 04:01:14 AM EDT> <Notice> <server123> A total of 500 files have been compressed.  <Mar 13, 2013 04:01:14 AM EDT> <Notice> <server123> A total of 700000kB have been freed from deletion.  <Mar 13, 2013 04:01:14 AM EDT> <Notice> <server123> A total of 350000kB have been freed from compression.  <Mar 13, 2013 04:01:14 AM EDT> <Notice> <server123> A total of 346407kB have been freed from ALL methods.  <Mar 13, 2013 04:01:14 AM EDT> <Notice> <server123> This script ran for approximately 74 second(s).  <Mar 13, 2013 04:01:14 AM EDT> <Notice> <server123> Finished. |

The log can be categorized into three parts:

* The first part (1) of the log (**in light blue**), contains information such as the version of the Eraser software and the default values of the GLOBAL directives.
* The second part (2) of the log (**in light orange**), contains the results of all of the jobs that have been executed
* The last part (3) of the log (**in light purple**), contains the statistics of the execution. Information such as the number of jobs ran, the number of files compressed or deleted, the amount of space freed and the execution duration of the job are presented here.

Each line of the log contains the following headers:

* The date and time of the event (e.g. <Mar 13, 2013 04:01:14 AM EDT>)
* The type of error; the possible types are **Notice**, **Warn** and **Error**
* The hostname of the server (e.g. server123)
* (Optional) If the event is part of a job definition, the job number is shown here, sorted from low to high (e.g. <job:1>)
* The details of the event

# Resources and References

## Regular Expressions Reference Table

A regular expression is a specific pattern that provides concise and flexible means to "match" (specify and recognize) strings of text, such as particular characters, words, or patterns of characters.

The table below lists commonly used regular expression meta-characters. Please note this table only scratches the surface of the **Perl Regular Expression** library (which currently powers Eraser). The full Perl regular expression documentation can be found here: <http://perldoc.perl.org/perlre.html>

|  |  |  |  |
| --- | --- | --- | --- |
| Meta-characters | Descriptions | Code | Result Examples |
| ^ | Start of a string | ^abc | abc, abcdefg, abc123, … |
| $ | End of a string | abc$ | abc, endsinabc, 123abc, … |
| . | Any character (except newline) | a.c | abc, aac, acc, adc, aec, … |
| | | Alternation | bill|ted | ted, bill |
| {…} | Explicit quantifier notation | ab{2}c | abbc |
| […] | Explicit set of characters to match | a[bB]c | abc, aBc |
| (…) | Logical grouping of part of expression | (abc){2} | abcabc |
| \* | 0 or more of previous expression | ab\*c | ac, abc, abbc, abbbc, … |
| + | 1 or more of previous expression | ab+c | abc, abbc, abbbc, … |
| ? | 0 or 1 of previous expression | ab?c | ac, abc |
| \ | Preceding one of the above, it makes it a literal instead of a special character | a\sc | a c |

A number of special characters or meta characters are used to denote actions or delimit groups; but it is possible to force these special characters to be interpreted as normal characters by preceding them with a defined escape character, usually the backslash "\". For example, a dot is normally used as a "wild card" metacharacter to denote any character, but if preceded by a backslash it represents the dot character itself. The pattern c.t matches "cat", "cot", "cut", and non-words such as "czt" and "c.t"; but c\.t matches only "c.t". The backslash also escapes itself, i.e., two backslashes are interpreted as a literal backslash character.

## Special Characters Reference Table

|  |  |
| --- | --- |
| Character Classes | Descriptions |
| \c | Control character |
| \s | White space |
| \S | Not white space |
| \d | Digit |
| \D | Not digit |
| \w | Word |
| \W | Not word |
| \x | Hexadecimal digit |
| \O | Octal digit |
| \n | New line |
| \r | Carriage return |
| \t | Tab |
| \v | Vertical tab |
| \f | Form feed |
| \xxx | Octal character xxx |
| \xhh | Hex character hh |