NAME

argusi − Argus Eye: Argus Data Collection Controller

SYNOPSIS

/afs/athena.mit.edu/astaff/project/argus/@sys/bin/argusi

DESCRIPTION

argusi uses a window interface which operates under the X11 window system. It allows users to set specifications for collecting data about their use of the *Argus Message Filter* (see *argus(1)).* This document assumes familiarity with the *Argus Message Filter.*

argusi allows users to set collection times for three types of data: *rule activity*, *rule life*, and *message traffic.* Users can set the start and stop times for data collection on each of these categories. The user may also shut of data collection completely, regardless of the collection times, for any or all of the data collection categories.

Collected Data for the three categories are stored in three separate files. By default the three files are under the *data* subdirectory of the user's default *lens* directory. The default names of the three files are *lens.ra.log,* for *rule activity* data, *lens.rl.log* for *rule life* data, and *msg.log* for *message traffic* data. All the above defaults can be reset in the user's *.lens.profile* (see *argus(1)).*

The start and stop times that the user provides *argusi* define a time interval for data collection. There are two types of data collection intervals: *renewable* and *non-renewable.* Data generated for a category with a *non-renewable* time interval will be collected only if it is generated within the specified start and stop times. Under these circumstances the data is stored directly to disk (with some formating) directly after it is generated. However, data generated for a category with a *renewable* time interval will be collected if it is generated after the start time. In this case the collected data is not stored directly to the collection file. It is stored internally until the time interval expires, at which point a statistical overview of the data collected for that time interval is written to the data collection file for that category. At this point the time interval is *renewed* and data collection continues. Thus, data collection for a category with a renewable time interval can continue past the stop time for that category.

The type of time interval for a given data collection category is invariable and cannot be set or changed by the user.

Below is a description of each of the three data categories, outlining what data are collected for each category, and which type of time interval is associated to which category.

rule activity
: Data of this type is generated each time *runrules(1)* evaluates a rule against a message, that is, each time the conditions of a rule are compared to a message. Time intervals for this category are *renewable,* so data is collected over a period of time, then statistical information about the data is written to the rule activity data collection file. The statistics for a given rule's activity within the time interval are written hierarchically, with the rule at the top of the hierarchy. The rule is identified by its last update time in the format returned by *ctime(3).* Underneath the rule identifier appears each ruleset from which the rule was called. The ruleset is indentified by its name. Underneath the ruleset identifier appears the actual data: the number of messages the given rule in the given ruleset was evaluated against, and the number of messages of this set for which the evaluation returned true (that is, for which the rule fired). The above data is subcategorized according to the *type* of the message.

rule life
: Data of this type is generated when the user creates, deletes, or changes rules or rulesets.

message activity
: Data of this type is generated when *runrules(1)* processes a message. Time intervals for this category are renewable. The information stored in the data collection file at the end of the time interval is categorized by the *type* of the message. Under the individual headings for each message type appear the number of messages processed, subcategorized by direction (incoming, outgoing, other).

FILES

There is a bug in the internal date parser. This may lead to warnings certain circumstances.

SEE ALSO

argus(1), runrules (1), rulebrowser (1), ruleeditor (1), alcomp (1), alinc (1), mteditor (1)

AUTHOR

Toby Mintz, MIT Project Athena and Digital Equipment Corporation

RESTRICTIONS