LogTrend's Complex Alarms Manual

LogTrend

LogTrend's Complex Alarms Manual

by LogTrend

Revision History

Revision 1.0September, 2001

First DocBook version

Table of Contents

Chapter 1. LogTrend's Complex Alarms

1.1. Alarms in LogTrend

There are two types of alarms in LogTrend : agents' binary alarms and complex alarms. Agents' alarms are generated during the data collection. These alarms are useful to know the state of a system in real time, but not sufficient to carry out trends analysis. Trend analysis can be done by complex alarms. These alarms are described by logical equations like : " if the cpu load is superior to 90 % during two hours EXCEPT monday between 9 and 12."

These equations are logical operations (AND, OR, NOT) between functions, called detection functions, applied to variables :

```
superior(cpu, 90, 3600) AND inferior(cpu_system, 10, 3600)
```

`superior` and `inferior` are detection functions. `cpu` and `cpu_system` are variables.

You can found a description of the standard detection functions in Section 2.3.

1.2. Complex Alarms configuration file

Like all LogTrend config files, Complex Alarms comfiguration file is an XML file :

```
<ComplexAlarm>
   <DataBase Name="logtrend" Host="server.domain.fr" Port="5432"
             User="logtrend" Password="logtrend" />
   <Source>5</Source>
   <Agent>7</Agent>
   <Number>1</Number>
   <Name>CPU overload detection</Name>
   <Level>
     <Warning />
       </Level>
   <Action>
     <Xmessage Display=":0.0">Alarm</Xmessage>
     <Mail Address="me@mydomain.com" Subject="CPU Overloaded">
       The CPU is overloaded on myclient.mydomain.com
     </Mail>
   </Action>
   <Equation>
     (superior(cpu, 90) AND inferior(cpu_system, 10)) AND duration(3600)
   </Equation>
 </ComplexAlarm>
```

1.2.1. DataBase (required)

```
<DataBase Name="logtrend" Host="server.domain.fr" Port="5432"
          User="logtrend" Password="logtrend" />
```

The DataBase tag is used to set up LogTrend database connection parameters. This tag has 5 attributes and contains no data.

Attributes (all is required, no default values) :

- Name : Name of the database
- Host : Name of the host where is the database
- Port : Port of the database on the server
- User : User name to connect to the database
- Password : Password of this user

1.2.2. Source, agent and number (required)

```
<Source>5</Source>
<Agent>7</Agent>
<Number>1</Number>
```

Each complex alarm is identifed by a source, an agent and a number. Each complex alarm of an agent must have a different number.

1.2.3. Alarm level (required)

```
<Level>
   <Warning />
</Level>
```

There are three levels for alarms :

- Info : Not really an alarm, just an information
- Warning : Not critical alarm.
- Error : Critical alarm.

1.2.4. Action (optional)

```
<Action>
  <Xmessage Display=":0.0">Alarm</Xmessage>
  <Mail Address="me@mydomain.com" Subject="CPU Overloaded">
    The CPU is overloaded on myclient.mydomain.com
  </Mail>
</Action>
```

This tag describe actions launched when the alarm is raised. There four standard actions. But you can develop customized action.

Available actions are :

1.2.4.1. Mail

```
<Mail Address="me@mydomain.com" Subject="CPU Overloaded" Sender="sender@mydomain.com" SMTPServ
  The CPU is overloaded on myclient.mydomain.com
</Mail>
```

Send a mail to `Address` with subject `subject`. If specified, SMTPServer is used. Sender is used in the From field of the mail. By default, Sender is equal to Address.

1.2.4.2. SMS

```
<SMS NationalPhone="0612345678" InternationalPhone="+33612345678">
   System overloaded
</SMS>
```

Send a message to a cellular phone. Phone number must be specified in national and international format.

Note: Currently, SMS Sender uses free web sites to send messages. This method does not really work because it depends of web site structure that change frequently. This action will be soon reimplemented to used more reliable methods like GSM modem.

1.2.4.3. Syslog

```
<Syslog>System overloaded</Syslog>
```

Add a message in system logs with Syslog

1.2.4.4. Execute

```
<Execute>/usr/local/bin/myscript.pl</Execute>
```

Run an external command line.

1.2.4.5. Xmessage

```
<Xmessage Display=":0.0">Alarm</Xmessage>
```

Display a message in a X Window. This action is very usefull while debugging.

1.2.5. Equation (required)

```
<Equation>
   (superior(cpu, 90) AND inferior(cpu_system, 10)) AND duration(3600)
</Equation>
```

Contains the complex alarm equation, see Chapter 2 for more detail.

Chapter 2. Complex Alarm Equations

As I said above, complex alarm equations are logical operations between detection functions :

```
(superior(cpu, 90) AND inferior(cpu_system, 10)) AND duration(3600)
```

Above equation stand for : "if the cpu is up to 90 % and cpu used by the system inferior to 10 % during 1 hour." It’s possible to do more complex equation with parenthesis like :

```
(f1() AND f2()) OR (NOT f4() OR f5() AND f6())
```

2.1. Logical operators

There are three logical operators available : AND, OR and NOT. Priorities are classical : NOT has priority on AND and AND has priority on OR.

Detection functions return tabulars of data, one tabular by variable. In this tabular, there are couple of (value, date). AND, OR and NOT do intersection, union and inverting of these tabulars. An equation is true if the result is a non empty tabular.

> **About NOT operator:** Note that the NOT does not work like a boolean NOT but like a binary NOT. It return all values which was in initial values tabular and NOT in the result return by the function.

2.2. Detection functions

Detection functions can take several arguments :

```
function_name(arg1, arg2, arg3)
```

Some of these arguments can be *variable names*. Variable names are name of agents data in lowercase with _ instead of spaces and wihout characters between parenthesis.

Examples :

- "Cpu System" became cpu_system
- "Received bytes on eth0 (192.168.10.47)" became received_bytes_on_eth0

You can have the list of agents data names with GetListfVariables (see Appendix A for more details on Complex Alarm tools).

It’s possible to build equation using variable of differents agents :

```
var_name[source_number, agent_number]
```

Default values for source and agent are value of <Source> and <Agent> tags of the complex alarm description file.

Example :

```
function1(var1[2, 5]) AND function2(var2[9, 1]) AND function3(var3)
```

You can do arithmetic operations between variables or number :

```
function1(var1 + var2, (10+8)/5, 15) AND function2((var3+var4[5, 6])/2)
```

2.3. Standard detection function

`superior``( variable limit [duration] );`

Return value of `variable` superior to `limit`. If `duration` is present, `superior` return all values suites superior to `limit` during at least `duration` seconds.

`inferior``( variable limit [duration] );`

Return value of `variable` inferior to `limit`. If `duration` is present, `inferior` return all values suites inferior to `limit` during at least `duration` seconds.

`equal``( variable limit [duration] );`

Return value of `variable` equal to `limit`. If `duration` is present, `equal` return all values suites equal to `limit` during at least `duration` seconds.

`different``( variable limit [duration] );`

Return value of `variable` not equal to `limit`. If `duration` is present, `different` return all values suites not equal to `limit` during at least `duration` seconds.

`between``( variable lower_bound upper_bound [duration] );`

Return value of `variable` between `lower_bound` and `upper_bound`. If `duration` is present, `between` return all values suites between `lower_bound` and `upper_bound` during at least `duration` seconds.

average(*variable duration comp_func comp_func_args*);

Return the average on `duration` seconds for each data of variables, and apply the comparison function `comp_func` to the result. `comp_func` should be one of `superior`, `inferior`, `equal`, `different` and `between`.

duration(*duration*);

This function is very particular. `duration` is not apply to a particular variable but to all variables used in the equation. `duration` look for a suite of value with continuous date. The sentence which describe when this function can be used is : "during X seconds".

Example :

```
(superior(cpu, 90) AND inferior(cpu_system, 10)) AND duration(3600)
```

This equation stand for : "when cpu is greater than 90 % AND cpu_system lower than 10 % during 3600 seconds.

Warning

This is not the same thing than :

```
superior(cpu, 90, 3600) AND inferior(cpu_system, 10, 3600)
```

In this equation, it's not mentionned that `superior` and `inferior` must be true during the same 3600 seconds

There are more examples with duration in Section 2.3.1.

time_interval(*start_date end_date*);

Return the data having a date between `start_date` and `end_date` Dates are strings like "friday 14 september 2001 13 hours 00 minutes 00 seconds" or "fri 14 sep 2001 13 h 00 m 00 s" or "14/09/2001 13:00:00". Hours is in 24 hours formats.

Dates are not necessary absolute. Examples :

- "Tue 24 april 2001 9 hours 24 min 30 sec"
- "24 april" : each 24 april
- "24" : the 24th of each month
- "24 10 hours" : the 24th of each month at 10
- "2000" : the data of the year 2000
- "9 h 24 m" : each day at 9h 24

Note: Unlike `duration` you can write `NOT time_interval(date1, date2)` to exclude a time interval

2.3.1. Examples of equation with `duration`

- `f1(var1) AND duration(120) AND f2(var2)` : duration just affect f1 and not f2. It's equivalent to : `(f1(var1) AND duration(120)) AND f2(var2)`
- `f1(var1) AND f2(var2) AND duration(120)` : duration just affect f2 and not f1. It's equivalent to : `(f2(var2) AND duration(120)) ANDf1(var1)`
- `(f1(var1) AND f2(var2)) AND duration(120)` : Thank to parenthesis, duration is applyed to `f1(var1) AND f2(var2)`
- `duration(120) AND f2(var2)` : This does not make any sense. There is nothing to compare where duration is ran.
- `(superior(cpu, 90) AND inferior(cpu_system, 10)) AND NOT duration(3600)` : This does not make any sense. You should write : `NOT ((superior(cpu, 90) AND inferior(cpu_system, 10)) AND duration(3600))`
- `(superior(cpu, 90) AND inferior(cpu_system, 10)) OR duration(3600)` : This does not make any sense. OR cancel duration.

Appendix A. Complex Alarm tools

BuildDescFileSquel [--database *database_name*] [--host *host_name*] [--port *server_port*] *filename.xml*
[--help]

Tool to build new complex alarm description files.

ComplexAlarm [--register] [--frequency *freq*] [--daemon] *filename*...
[--help]

Run or register complex alarms.

- With `--register`, register complex alarms in database.
- `--frequency` specify the time in minutes between 2 alarms check.
- `filename` can be a directory name

GetSourcesList [--database *database_name*] [--host *host_name*] [--port *server_port*] [--help]

Return the list of the source on the specified database.

GetListOfAgentsOnSource --source *source_number* [--database *database_name*] [--host *host_name*] [--port *server_port*] [--help]

Return the list of agents of a source.

GetListOfVariables --source *source_number* --agent *agent_number* [--database *database_name*] [--host *host_name*] [--port *server_port*] [--help]

Return the variable names list of an agent.