



# IVD

# INFINISTORE VirtualDisk

LogAnalyzer

## Administrator Guide

**GRAU**  
DATA STORAGE

Version 2.3

# INFINISTORE VirtualDisk

---

## Imprint

This manual has been prepared with the utmost care. Nevertheless, there may still be mistakes in form and content.

Protected trademarks are not marked as such in this Manual. The fact that these trademarks are not shown does not imply that the trade names are free for use.

All rights, including those arising from applications for proprietary rights, withheld. The publisher retains all rights of disposition, such as copying or distribution.

Subject to changes without notice.

This Manual is related to Version 2.3.1 (Build: 4)

Publisher: GRAU Data Storage AG,  
Marie-Curie-Straße 19 , D-73529 Schwäbisch Gmünd, Germany.

Editors: GRAU Data Storage AG,  
Am Weinberg 20, D-86732 Oettingen i. Bay., Germany.

© 2005 by GRAU Data Storage AG,  
Marie-Curie-Straße 19 , D-73529 Schwäbisch Gmünd, Germany.

6. reworked Edition in December 2005

# INFINISTORE VirtualDisk

---

## Contents

---

---

Imprint .....	3
---------------	---

---

Contents .....	5
----------------	---

---

1. General .....	7
------------------	---

---

2. Installation .....	8
2.1 Installation on a Windows System.....	8
2.2 Installation on a Linux System.....	9

---

3. Configuration .....	11
3.1 The <i>LogAnalyzer.config</i> Configuration File.....	12
3.2 The <i>LogAnalyzerRules.py</i> Configuration File .....	16
3.2.1 Describing a Filter.....	17
3.2.2 Calling Actions.....	20
3.2.3 Example of a <i>LogAnalyzerRules.py</i> Configuration File .....	22
3.3 MIB Files .....	23

---

4. Employment on a Linux System .....	25
4.1 The <i>LogAnalyzer_messages.config</i> Configuration File .....	26
4.2 The <i>LogAnalyzerRules_messages.py</i> Configuration File.....	27

---

5. GRAU Service Center.....	29
-----------------------------	----

---

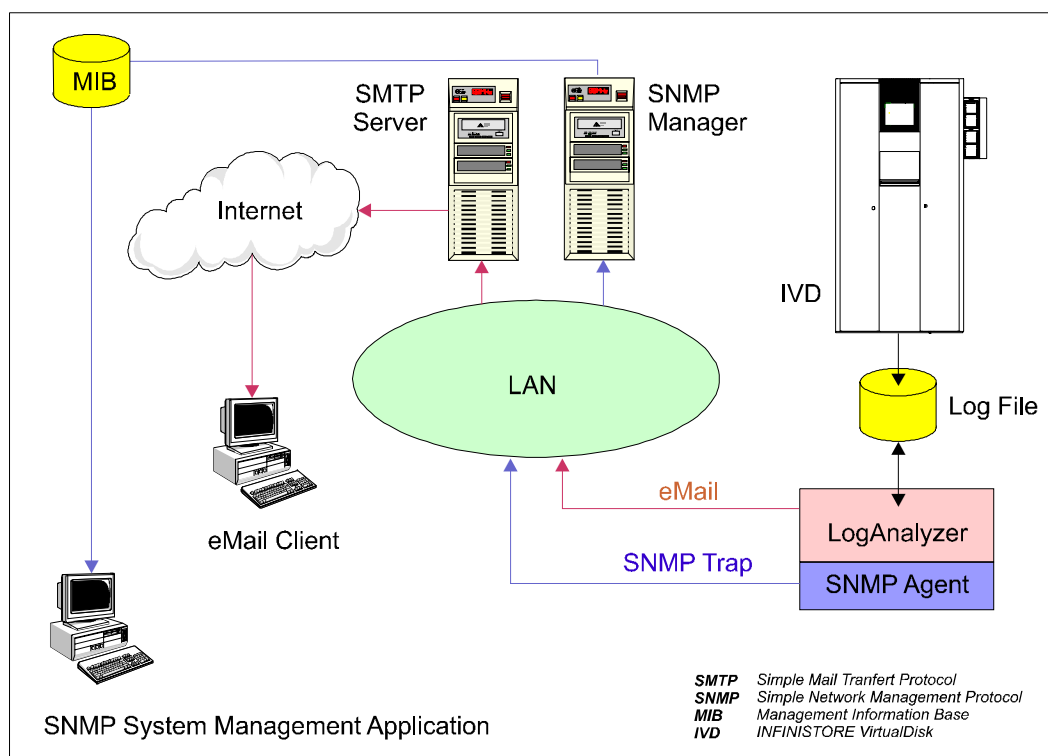
6. Software License Agreement .....	31
-------------------------------------	----

---

# INFINISTORE VirtualDisk

## 1. General

The *IVD LogAnalyzer* can be used to permanently watch messages in any kind of log file. Usually, it is used to dispatch emails or SNMP (Simple Network Management Protocol) traps based on error and warning messages out of the watched log file.



The messages are collected in a package and sent to previously defined destination addresses at specified intervals. Emails and/or SNMP traps are generated on base of messages which have to match user defined rules.

---

## 2. Installation

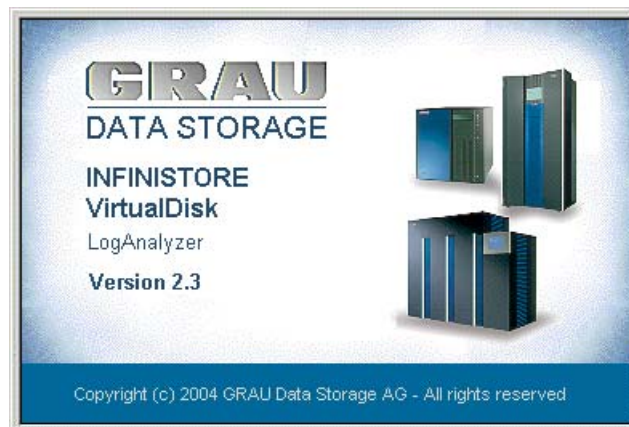
---

---

### 2.1 Installation on a Windows System

---

You find the setup program of the *IVD LogAnalyzer* on the installation CD in the `\Win32\Tools\LogAnalyzer` directory. To install the *IVD LogAnalyzer*, you have to start the program *setup.exe* from CD. Follow the instructions appearing on the screen until installation is finished.



After having completely adapted the configuration files *LogAnalyzerRules.py* and *LogAnalyzer.config* (see page 11), the *IVD LogAnalyzer* is ready for use.

#### De-installation

- ☒ If the *IVD LogAnalyzer* was started as service, stop the service *GRAU IVD LogAnalyzer*.
- ☒ Remove the product *GRAU IVD LogAnalyzer* using the software manager of Microsoft Windows.



## 2.2 Installation on a Linux System

The *IVD LogAnalyzer* installation requires the installation of the packages mentioned below. Whether these packages are installed is checked by *rpm* during the installation of the *IVD LogAnalyzer*.

☒ compat version 2002.8.15

The rpm packages for the installation of the *IVD LogAnalyzer* on a Linux system is located in the following directories of the installation CD.

SuSE:        */sles8/ia32/tools/LogAnalyzer*

RedHat:      */rhel3/ia32/tools/LogAnalyzer*

**Call:**

```
rpm -i LogAnalyzer-2.3-X.i386.rpm
```

x            = Revision number of the *IVD LogAnalyzer*

**Example:**

```
rpm -i /usr/src/packages/RPMS/i386/LogAnalyzer-2.3-1.i386.rpm
```

**Check of installation:**

```
rpm -qa | grep LogAnalyzer
```

A positive check shows the installed rpm package of the *IVD LogAnalyzer* (see example).

```
LogAnalyzer-2.3-1
```

The following files are created through the *IVD LogAnalyzer* installation:

- /opt/ivd/sbin/loganalyzer	- /usr/local/lib/_random.so
- /opt/ivd/sbin/loganalyzer_messages	- /usr/local/lib/array.so
	- /usr/local/lib/cStringIO.so
- /opt/ivd/doc/LogAnalyzer23DE.pdf	- /usr/local/lib/fcntl.so
- /opt/ivd/doc/LogAnalyzer23UK.pdf	- /usr/local/lib/math.so
	- /usr/local/lib/pwd.so
- /etc/opt/ivd/loganalyzerrules.py	- /usr/local/lib/struct.so
- /etc/opt/ivd/loganalyzer.config	- /usr/local/lib/time.so
- /etc/opt/ivd/loganalyzerrules_messages.py	- /usr/local/lib/_socket.so
- /etc/opt/ivd/loganalyzer_messages.config	- /usr/local/lib/binascii.so
- /etc/opt/ivd/gds.mib	- /usr/local/lib/collections.so
- /etc/opt/ivd/gds_ivd.mib	- /usr/local/lib/itertools.so
	- /usr/local/lib/md5.so
- /etc/init.d/loganalyzer	- /usr/local/lib/strop.so
- /etc/init.d/loganalyzer_messages	- /usr/local/lib/termios.so
- /usr/local/lib/operator.so	- /usr/local/lib/zlib.so

After having completely adapted the configuration files *loganalyzerrules.py* and *loganalyzer.config* (see page 11), the *IVD LogAnalyzer* is ready for use.

The files *loganalyzer\_messages*, *loganalyzerrules\_messages.py* and *loganalyzer\_messages.config* can be used to monitor a second log file (for example the Linux log file */var/log/messages*). The structure of these files are equals structure of the files *loganalyzer*, *loganalyzerrules.py* and *loganalyzer.config*. For further information see page 25.

### De-installation

Use the following rpm command, if you wish to de-install the *IVD LogAnalyzer*.

```
rpm -e LogAnalyzer-2.3-1
```

### Start as Process

If the *IVD LogAnalyzer* is to be started as process, the environment variable *LD\_LIBRARY\_PATH* must be set before (see description below).

```
export LD_LIBRARY_PATH=/usr/local/lib
```

### Start as Daemon

```
usage: /etc/init.d/loganalyzer {start|stop|status|restart}
usage: /etc/init.d/loganalyzer_messages {start|stop|status|restart}
```

### 3. Configuration

Before you are able to use the *IVD LogAnalyzer*, you have to individually adjust the following configuration files:

☒ **LogAnalyzer.config**

Serves to configure the mail and SNMP dispatch

☒ **LogAnalyzerRules.py**

Serves to configure the log analysis

Irrespective of the platform used (Windows or Linux), you can start the *IVD LogAnalyzer* as a process from the command line. The parameter to be entered here is the name of the configuration file.

```
LogAnalyzer LogAnalyzer.config
```

This is helpful, if, depending on the situation, the *IVD LogAnalyzer* shall be started with different configuration files or if several log files are to be monitored.

When calling the *IVD LogAnalyzer* without parameters as process, the following help text appears:

```
LogAnalyzer
Version 2.3.1.0004 (C) Copyright GRAU DataStorage AG, 2001-2005.
All rights reserved.
usage: LogAnalyzer [-h|--help|-d|--debug|-s|smtp_debug|-v] <ConfigFile>
-h | --help          shows the usage information
-d | --debug         enables the debug mode
-s | --smtp_debug    enables smtp debug mode
-v                  shows the version information
```

On a Windows system, the *IVD LogAnalyzer* is additionally installed as a service that can be started via the Service Manager after the configuration is finished. In order to automatically activate the *IVD LogAnalyzer* after each system start, change the start type of the *IVD LogAnalyzer* to *automatic*. This is done in the Service Manager.

As for Linux systems, the *IVD LogAnalyzer* is started as daemon. How to start and stop a daemon is described in your Linux documentation.



If the *IVD LogAnalyzer* is started with the option *-d* or *--debug*, all operations, together with the triggering rule, are displayed at the screen. For further information about rules see page 16.

### 3.1 The *LogAnalyzer.config* Configuration File

The installation program creates a default configuration file. The default values have to be adjusted according to the given requirements (see example).

```
# =====
# Template for the configurationfile of the LogAnalyzer Version 2.3.1
# Created: 25.10.05
# Author: GDS Development
# =====

# === USED TERMS ===
# AnalysisFile: The file which analyzes the LogAnalyzer
# Package:      The file, produced by the LogAnalyzer.
#               This is the file which will be attached to the email.

[LogAnalyzer]
# =====
# This section describes internal settings of the LogAnalyzer
# =====

# --- SleepTime ---
# After "SleepTime" the LogAnalyzer checks the AnalysisFile
# again for new Messages
SleepTime = 0.1

# --- Debug ---
# Debug = 0      ==> No Debuginformation is shown (Default)
# Debug = 1      ==> Shows Debuginformation ( Rules, Actions, etc.)
Debug = 0

# --- ProcessFileFromBeginning ---
# ProcessFileFromBeginning = 0 ==> Processing the AnalysisFile at the end
#                               ( Default, Waiting for new messages )
# ProcessFileFromBeginning = 1 ==> Processing the AnalysisFile from the
#                               beginning. ( For debugging purposes )

ProcessFileFromBeginning = 0

# --- RulesFile ---
# The rules, which the LogAnalyzer will load on startup
# NOTE: Must be fully qualified pathname !
RulesFile = C:\Program Files\GDS\LogAnalyzer\LogAnalyzerRules.py

# --- TmpDir ---
# Directory where temporary files are stored.
# If not set, the System-Temporary-Directory will be used.
# NOTE: Must be fully qualified pathname !
TmpDir = C:\Program Files\GDS\LogAnalyzer\LogTmp

# --- ActivityLog ---
# In this file the LogAnalyzer logs its activities.
# NOTE: Must be fully qualified pathname !
ActivityLog = C:\Program Files\GDS\LogAnalyzer\Analyzer.log

# --- AnalysisFile ---
# The file which is analyzed by the LogAnalyzer
AnalysisFile = C:\Program Files\GDS\LogAnalyzer\logfile2.txt
```

```

[Mailer]
# =====
# This section describes the settings for the MAIL-features of the LogAnalyzer
# =====

# --- ServerName ---
# Name of the SMTP-Server, which will be used to send the mails
ServerName = mailgds1.graustorage.de

# --- SMTP Authentication ---
UseSMTPAuthentication = 0
#Username = mustermann
#Password = bux91

# --- Text ---
# This text will be used in the body of the mail
# NOTE: For multiple lines the continuation-lines must start with a whitespace!
MailText =
    This is a package from your LogAnalyzer
    -----
    ...
    Your LogAnalyzer
    -----

# --- Subject ---
# Subject-Part of the mailheader
Subject = IVD-Mail

# --- Subject Prefix---
# Used for MailCurrentLogMessages("Sample Subject")
# If used, this text will be prepended to the text "Sample Subject"
SubjectPrefix = IVD001

# --- From ---
# From-Part of the mailheader
From = IVD001@company.com

# --- To ---
# To-Part of the mailheader
To = service@GrauDataStorage.de mueller@company.com maier@company.com

# --- AttachmentPrefix ---
# The Packagefilename is composed by: <AttachmentPrefix>_<YYMMDD_HHMMSS>.log.zip
# Example : IVD_001_040517_174320.log.zip
AttachmentPrefix = IVD_001_

# --- Zip Attachment ---
ZipAttachment = 1

# --- Slot ---
# Time interval to send the package (in minutes)
Slot = 1

```

```
[SNMP]
# =====
# This section describes the settings for the SNMP-features of the LogAnalyzer
# =====

# --- DestinationIP ---
# The IP-Address of the management unit
DestinationIP = 127.0.0.1

# CommunityName
# The community name of the machine
CommunityName = public

# --- SenderOID ---
# The SenderOID. This value depends on the product, that will be used
SenderOID = 1.3.6.1.4.1.18945.1.1.0
```

Adjust the following data in the sections of the configuration file:

Key	Check/Adjustment
<b>SleepTime</b>	Waiting time in seconds between the analysis cycles.
<b>Debug</b>	Debug mode 0 = Debug mode off (default), 1 = Debug mode on.
<b>ProcessFileFromBeginning</b>	Analysis mode 0 = Analysis of the newly-added log messages (default) 1 = Analysis of the entire log file (debug purpose)
<b>RulesFile</b>	Fully qualified path name of the file containing the analysis rules.
<b>TmpDir</b>	Fully qualified directory name for saving temporary LogAnalyzer files. (Default: system temp directory).  Make sure that this directory exists, otherwise, it must be created prior to the first start.
<b>ActivityLog</b>	Fully qualified path name of the LogAnalyzer log file.
<b>AnalysisFile</b>	Fully qualified path name of the log file which is to be analyzed.
<b>ServerName</b>	Name or IP address of the own SMTP server.

Key	Check/Adjustment
<b>UseSMTPAuthentication</b>	<p>Authentication mode</p> <p>0 = Disables authentication (default)</p> <p>1 = Enables authentication</p> <p>If smtp authentication is enabled, the following keys are also necessary:</p> <p style="padding-left: 40px;"><i>Username</i></p> <p style="padding-left: 40px;"><i>Password</i></p> <p>In this case remove the comment symbols.</p>
<b>MailText</b>	Text contained in the mail. If the text consists of several lines, the lines following must start with a blank.
<b>Subject</b>	Mail subject.
<b>SubjectPrefix</b>	The text, which is defined in this key, is placed in front of the subject of the <i>MailCurrentLogMessage</i> action (see page 20).
<b>From</b>	Mail address of sender.
<b>To</b>	One or more mail addresses of receiver delimited by space.
<b>AttachmentPrefix</b>	Prefix for the attached file (analysis results) which avoids that this file is continuously overwritten at the receiver.
<b>ZipAttachment</b>	<p>Compression mode</p> <p>0 = Disables compression of attachments</p> <p>1 = Enables compression of attachments (default)</p>
<b>Slot</b>	Interval for the dispatch of message packages (all filtered log messages) in minutes.
<b>DestinationIP</b>	IP address of the SNMP server.
<b>CommunityName</b>	A name used to group SNMP hosts. By default, all hosts belong to the <i>public</i> community, that is the standard name for the common community of all SNMP hosts.
<b>SenderOID</b>	<p>Object ID (OID) of the sender system. The OID depends on the particular GRAU product:</p> <p>IVD =1.3.6.1.4.1.18945.1.1.0</p> <p>ITL =1.3.6.1.4.1.18945.1.2.0</p> <p>IAF =1.3.6.1.4.1.18945.1.3.0</p> <p>FMA =1.3.6.1.4.1.18945.1.4.0</p> <p>FSR =1.3.6.1.4.1.18945.1.5.0</p>

### 3.2 The *LogAnalyzerRules.py* Configuration File

The *LogAnalyzerRules.py* file contains the rules according to which the *IVD LogAnalyzer* checks the defined log file. These rules are formulated in the form of regular expressions. To say it more simplified: these rules are search patterns which are applied to the string of a log message.

If the search patterns match the string of a log message, the *IVD LogAnalyzer* starts the necessary action, that is a previously defined procedure. Thus, a rule consists of two components: one component which analyzes the string of a log message (filter) and a second component which defines the procedure to be executed (see page 20).

#### The rule syntax:

```
RULES = \  
[  
    {  
        'filter': STRING,  
        'action': PROCEDURE  
    }  
]
```

#### Example:

```
RULES = \  
[  
    {  
        'filter': ".*errors.*",  
        'action': "MailLogMessages() "  
    }  
]
```



### 3.2.1 Describing a Filter

In the example described above, a log message appears containing the string „errors“. As a consequence, the action *MailLogMessages()* is started causing the dispatch of a mail. In the filter, the character combination „\*“ (wildcard) stands in front of and after the actual search string „errors“. This wildcard represents any character string.

A separate rule has to be created for every filter constellation. In the following example, a mail shall be dispatched whenever the terms „error“ or „failer“ appear in a log message.

```
RULES = \
[
  {
    'filter': ".*error.*",
    'action': "MailLogMessages() "
  },
  {
    'filter': ".*failer.*",
    'action': "MailLogMessages() "
  }
]
```

The Exclude instruction can be used to define the search inquiries in the rules in a way that particular messages are excluded even if the filter condition is fulfilled for this log message.

Let's have a look at an example to make the use of the Exclude instruction more clear. A mail shall be generated when finding the string „errors“ in a log message. When the message says „No errors found“, however, a mail should not be generated.

```
RULES = \
[
  {
    'filter': ".*error.*",
    'exclude': ".*No errors found.*",
    'action': "MailLogMessages() "
  }
]
```

All IVD log messages have the same structure like the message in the following example (error number 15001).

```
[2004/03/12 23:12:15, infinistore, LAS, GTL-340i, 20040312000002, 000021]
ERROR: Move: D=1 -> S=3 FAILED (15001: Invalid barcode in slot or drive.
'Drive # 1 bc: expected: 000021')
```

For further information about the structure and the error codes of log messages, please read the *IVD User's Guide*.

If you want the *IVD LogAnalyzer* to search for the string „[E]“ which describes a log message as error message, you have to define hard characters (here „[“ and „]“) in the filter. This might be difficult. When specifying hard characters in a filter, a backslash is written in front of the hard characters (see example).

```
RULES = \
[
  {
    'filter': ".*\\[E\\].*",
    'action': "MailLogMessages()"
  }
]
```

The search string „[E]“ will then be „\\[E\\“. The wildcard „.\*“ which stands for any character string, has already been explained above.

#### Hard characters:

Character	becomes ...
[	\\[
]	\\]
{	\\{
}	\\}
'	\\'
"	\\"
\\	\\\\

#### Meta characters:

Character	stands for ...
. *	no or repeated occurrence of any character.
.	a single character.
^	the beginning of a string.
\$	the end of a string.
	Separates alternatives (A B = character 'A' or 'B').

*Regular Expressions* (RE) can be applied in every search string. That means that the filter is activated whenever a substring defined by *Regular Expressions* corresponds to the respective log message. Let's have a look at the following example which illustrates the effect of the *Regular Expressions*.

Log messages:

```
[FSC1111][E][2003/12/16 - 18:39:36,292][00000001][0448:061c] ML2-Partition ...  
[FSC1111][I][2003/12/16 - 18:39:36,292][00000001][0448:061c] Switch to Part. [E] ...
```

Rule with RE

```
RULES = \  
[  
  {  
    'filter': "^\\[. {7}\\]\\[E\\]",  
    'action': "MailLogMessages() "  
  }  
]
```

The rule shown above filters only the first one of the two log messages, since there are exactly seven characters (`{7}`) between the square brackets at the string beginning (`^\\[`) before an `[E]` (`\\[E\\]`) occurs.

For further information on *Regular Expressions* please read the relevant technical literature.

### 3.2.2 Calling Actions

The following procedures are available as actions:

#### Mail dispatch and saving log messages

Procedure	Description
<code>MailLogMessages()</code>	The filtered log messages are sent as email to the address defined in the <i>LogAnalyzer.config</i> file.
<code>MailLogMessages('subj')</code>	The filtered log messages are sent as email to the address defined in the <i>LogAnalyzer.config</i> file. The subject used, however, is the contents of <i>subj</i> . (See example below)
<code>MailCurrentLogMessage('subj')</code>	The currently filtered log message is sent as email to the address defined in the <i>LogAnalyzer.config</i> file. The subject used, is the contents of <i>subj</i> .
<code>SaveCurrentLogMessage()</code>	The currently filtered log message is saved by the Log Analyzer in a temporary file.
<code>DisableActions()</code>	When calling this procedure, all following procedure calls are suppressed.
<code>EnableActions()</code>	When calling this procedure, suppression of the procedures is cancelled (see <i>DisableActions</i> ).

#### Example:

```

RULES = \
[
  {
    'filter': "^.*\[E\].*",
    'action': "MailLogMessages('Error Message from system IVD001')"
  }
]

```

### Sending SNMP traps

Procedure	Description
<code>SendTrap (Trap)</code>	The log message found is sent as SNMP trap, depending on the set trap <i>Trap</i> .

Trap	Category	Description
I	Info	An information SNMP trap is sent.
W	Warning	A warning SNMP trap is sent.
E	Error	An error SNMP trap is sent.
C	Critical	A critical SNMP trap is sent.

### Example:

```

RULES = \
[
  {
    'filter': ".*\[E\].*",
    'action': "SendTrap('E')"
  }
]

```

Additional it is possible to call more than one procedure per action. In this case the procedures must be separated by a comma.

### Example:

```

RULES = \
[
  {
    'filter': ".*\[E\].*",
    'action': "MailCurrentLogMessage('Critical Message'), SendTrap('C')"
  }
]

```

### 3.2.3 Example of a *LogAnalyzerRules.py* Configuration File

```
# =====
# Sample Rulesfile for LogAnalyzer Version 2.3.1
# Created: 25.10.05
# Author: GDS Development
# =====

RULES = \
[
    {
        'filter': "Execution stack dump",
        'action': "DisableActions()"
    },

    {
        'filter': "End of stack dump",
        'action': "EnableActions()"
    },

    {
        'filter': "^\\[{7}\\]\\[E\\]",
        'action': "SaveCurrentLogMessage()"
    },

    {
        'filter': "^\\[{7}\\]\\[E\\]",
        'action': "SendTrap('E')"
    },

    {
        'filter': ".*",
        'exclude': "(^\\[{7}\\]\\[E\\]|End of stack dump|^$)",
        'action': "SaveCurrentLogMessage()"
    },

    {
        'filter': ".*So what.*",
        'action': "MailLogMessages()"
    },

    {
        'filter': "Critical",
        'action': "MailCurrentLogMessage('Critical message found')"
    },

    {
        'filter': "Critical",
        'action': "SendTrap('C')"
    }
]
```

### 3.3 MIB Files

MIB files serve to integrate a software product into an SNMP management system. The MIB files relevant for the *IVD LogAnalyzer* are:

- ☒ *GDS.MIB*
- ☒ *GDS\_IVD.MIB*

These MIB files are required only, if the SNMP traps generated by the *IVD LogAnalyzer* are to be processed by the SNMP management system. In this case, they are imported by the SNMP management system.

Through the installation of the *IVD LogAnalyzer* the above indicated MIB files are saved in the following directories:

<b>Windows:</b>	Installation directory of the <i>IVD LogAnalyzer</i> Default: <i>C:\Program Files\GDS\LogAnalyzer</i>
<b>Linux:</b>	<i>/etc/opt/ivd</i>

For further information on MIB files, please read the documentation of your SNMP management system.



Use a SNMP trap watcher for testing the SNMP configuration. A suitable SNMP trap watcher is available in internet on page

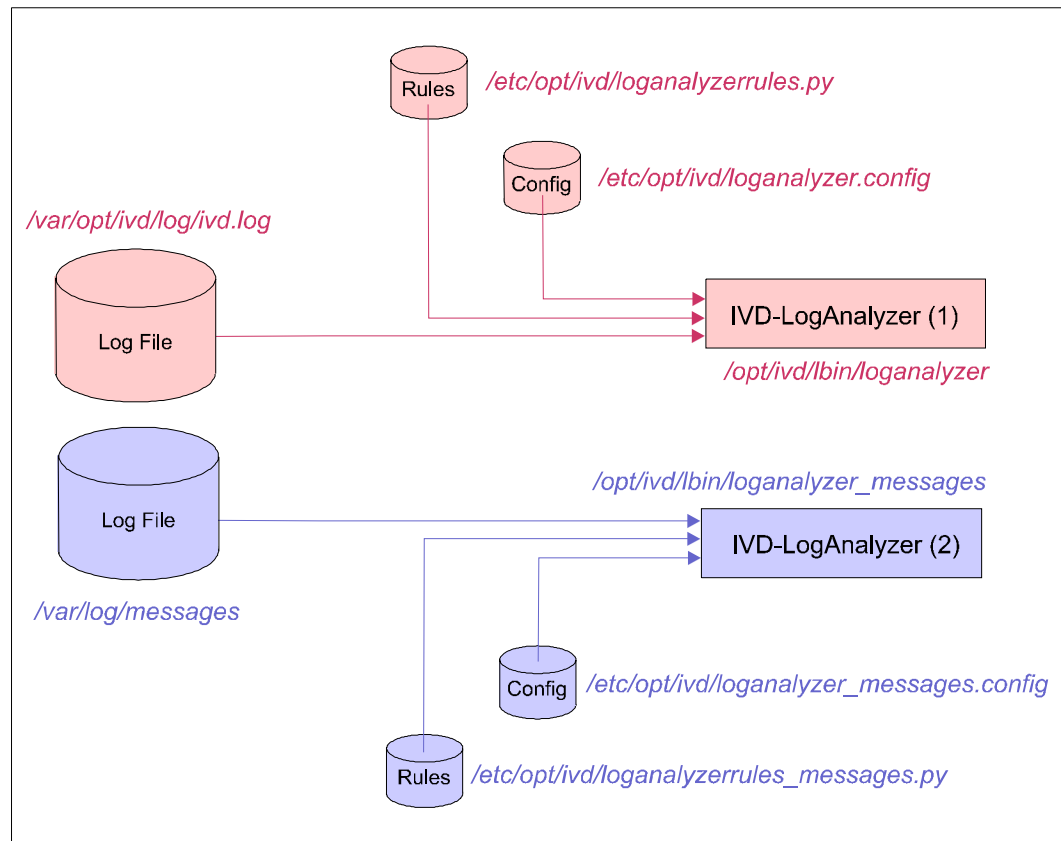
<http://www.bttsoftware.co.uk/snmptrap.html>

# INFINISTORE VirtualDisk



## 4. Employment on a Linux System

On Linux systems it might be useful to monitor the Linux system log file `/var/log/messages` in addition to the IVD log file `/var/opt/ivd/log/ivd.log`. The *IVD-LogAnalyzer* must be started therefore twice with different configuration files.



The appropriate templates are created during the installation (see page 10).

#### 4.1 The *LogAnalyzer\_messages.config* Configuration File

For the monitoring of a second log file (here */var/log/messages*) the installation program creates a second standard configuration file (*LogAnalyzer\_messages.config*). Please modify the default values to your needs (see example below).

```
.
.
# --- RulesFile ---
# Path to the rules the LogAnalyzer will load on startup
# NOTE: Must be full qualified!
RulesFile = /etc/opt/ivd/loganalyzerrules_messages.py

# --- TmpDir ---
# Directory to write temporary files.
# If not set, the System-Temporary-Directory will be used.
# NOTE: Must be full qualified!
# TmpDir = d:\MyTmp\LogTmp

# --- ActivityLog ---
# In this file the LogAnalyzer writes down his own activities.
# NOTE: Must be full qualified!
ActivityLog = /var/opt/ivd/log/loganalyzer_messages.log

# --- AnalysisFile ---
# The file to be analyzed by the LogAnalyzer
AnalysisFile = /var/log/messages
.
.
```

The structure is exactly the same as of file *LogAnalyzer.config*. For further informations about the structure of this configuration file see chapter „*The LogAnalyzer.config Configuration File*“ on page 12.

## 4.2 The *LogAnalyzerRules\_messages.py* Configuration File

For the monitoring of the Linux log file */var/log/messages* the installation program creates a standard configuration file (*LogAnalyzerRules\_messages.py*). The structure is exactly the same as of the file *LogAnalyzerRules.py* (see page 16).

```
.  
.   
RULES = \  
[  
    {  
        'filter': "kernel.*error",  
        'action': "SaveCurrentLogMessage()"   
    },  
    {  
        'filter': "CRITICAL",  
        'action': "MailCurrentLogMessage('Critical')"  
    },  
]  
.  
.
```

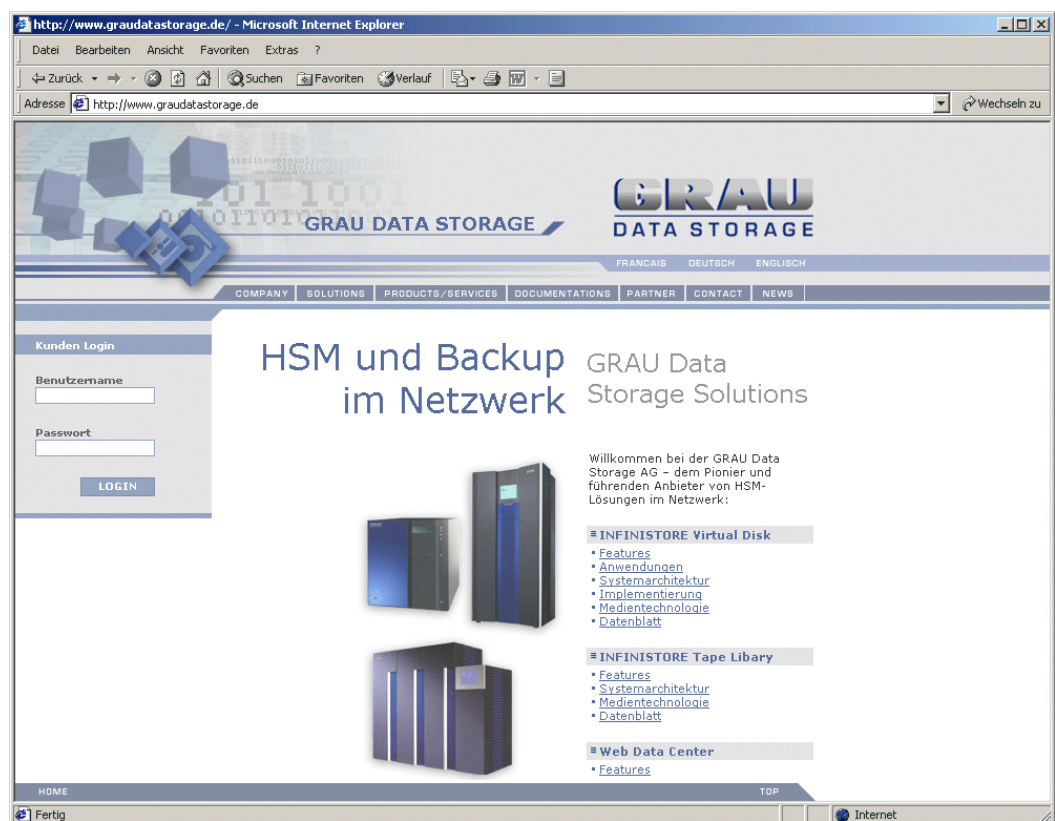
For further informations about the structure of this configuration file see chapter „*The LogAnalyzerRules.py Configuration File*“ page 16.

# INFINISTORE VirtualDisk

## 5. GRAU Service Center

Technical Support	
Telephone:	+49 (0)7171 / 187-333
Telefax:	+49 (0)7171 / 187-250
E-mail:	support@graudatastorage.de
Internet:	<a href="http://www.graadatastorage.de">www.graadatastorage.de</a>

Sales & Marketing	
Telephone:	+49 (0)7171 / 187-212
Telefax:	+49 (0)7171 / 187-250
E-mail:	info@graudatastorage.de
Internet:	<a href="http://www.graadatastorage.de">www.graadatastorage.de</a>



# INFINISTORE VirtualDisk

## 6. Software License Agreement

### **Attention:**

The Use of this Software is subject to the enclosed license agreement from GRAU Data Storage AG. By usage of the software, you demonstrate your agreement to the terms of the following license.

If you have not signed any other agreement with GRAU Data Storage AG, the following terms of license determine the user rights for this software.

### **GRANT OF LICENSE**

GRAU Data Storage AG grants you a non-exclusive, non-transferable license and right to use the software. You are not allowed to modify the software or to deactivate the license or control functions.

If you possess a multiple user license, not more than the maximal number of authorized users are allowed to use the software simultaneously.

Licensee assumes responsibility for the selection of the program to achieve intended results, and for the installation, use, and results obtained from the program. All rights not specifically granted in this license are expressly reserved by GRAU Data Storage AG.

### **RIGHT OF OWNERSHIP**

You agree that this software remains the property of GRAU Data Storage AG and that you only have a license to use this software. GRAU Data Storage AG retains title to all of the Software, Software Copies and related materials.

### **COPIES AND ADAPTIONS**

You may make and maintain one backup copy of the Software, provided it is used only for backup or archive purposes and you keep possession of the backup copy. In addition, all the information appearing on the original disk labels (including name of Software, serial number and any copyright notices) must be copied onto the backup labels.

You are not permitted to copy the Software on a public network or to rent, lease, sub-license time-share or sell the Software or the manual.

### **NO REVERSE ENGINEERING**

Modification, reverse engineering, reverse compiling, or disassembly of the Software is expressly prohibited without written permission from GRAU Data Storage AG. In some countries a limited disassembly or reverse compiling is allowed without permission. Please ask the legal department of GRAU Data Storage AG for more information about this.

You are not permitted to decode the Software unless the decoding it is part of software functionality.

### **ASSIGNMENT**

You may not transfer or assign the Software and/or this License Agreement to another party without the prior written consent of GRAU Data Storage AG. If such consent is given and you transfer or assign the Software and/or this License Agreement, then you must at the same time transfer any copy of the Software as well as the supporting documentation to the same party and the recipient must agree to this License Agreement.

#### **TERMINATION**

This License Agreement shall automatically terminate if you breach any of the terms and conditions of this License Agreement. If for any reason your right to use the software terminates you must remove and destroy all copies of this software and any documentation provided to you.

#### **EXPORT RESTRICTIONS**

You agree that you will not export or re-export the Software or accompanying documentation or any products utilizing the Software or such documentation in violation of any applicable laws or regulations.

Failure to comply with any term of this Agreement automatically terminates your license and may make you liable for legal damages to GRAU Data Storage AG.

#### **LIMITED WARRANTY AND LIMITATION OF REMEDIES**

The Software is provided "as is" without any warranty of any kind, either expressed or implied, including but not limited to implied warranties or merchantability and fitness for a particular purpose. Without limiting the foregoing, no warrant is given that the software will meet your requirements, or be compatible with other software or hardware. In no event shall GRAU Data Storage AG be liable for loss of profit, business interruption, loss of business information, or other pecuniary loss, including but not limited to special incidental consequential or other damages. Some States do not allow the exclusion of limitation of liability or incidental or consequential damages, in which case this limitation may not apply to you.

The software was exclusively developed on private costs. The Software and the Documentation are defined and licensed as "Commercial Computer Software".

INFINISTORE is a registered trademark of the GRAU Data Storage AG.