



MÄK Data Logger 4.4.1 Release Notes

This document provides the following release-specific information 4.4.1:

Systems Supported	2
Logger API	2
Qt Toolkit Compatibility	2
Compiler Compatibility on Windows	2
License Manager.....	3
ODBC Driver Support.....	3
Using Libraries and Binaries Built with Visual Studio 2005 and Later	3
Patch Required for AMD Dual-processor Windows PCs	4
Network Compatibility	4
RTI Support	5
FOM Support.....	5
New Features	5
Documentation Updates	6
Bug Fixes	6
Known Problems	7
Playing HLA 1.3 Logger Files in HLA 1516 Federations	7

Copyright © 2010 VT MÄK, 68 Moulton St., Cambridge, MA 02138 All rights reserved.
MÄK Technologies®, VR-Forces®, RTIsPY®, B-HAVE®, and VR-Link® are registered trademarks of VT
MÄK. Document ID: LOG-4.4.1-2-101026

Systems Supported

Table 1 lists the platforms currently supported by MÄK Logger 4.4.1. Application code must be built with the indicated compilers in order to link to Logger libraries.

Table 1: Platforms supported

Platform	Compiler
Red Hat Enterprise Linux Workstation 4.0, and 5.0	Default compiler
PC with Windows XP	Microsoft Visual C++ 7.1, 8.0, and 9.0
PC with Windows Vista or Windows 7	Microsoft Visual C++ 8.0 and 9.0



You must be an administrator to install MÄK products on Windows Vista.

Logger API

You must have VR-Link 3.13.2 and its required compiler versions to link the Logger libraries, but you do not need it to run the executables. Please see VR-Link documentation for information about required compilers.

You must have a VR-Link developer's license to compile plug-in or applications using the Logger API.

Qt Toolkit Compatibility

The Logger GUI is built using the Qt Toolkit, a cross-platform GUI toolkit. Logger 4.4.1 uses Qt 4.5. You do not need Qt to do development that does not affect the GUI.

Compiler Compatibility on Windows

Current and prior releases of MÄK products have been compiled using different versions of the Visual C++ compiler. When you run MÄK products together, for example, the Logger and the Stealth, we strongly recommend that you run versions created with the same compiler. Mixing products compiled with different compilers can result in program instability.

License Manager

To run the licensed version of Logger, you must install license management software. Logger 4.4.1 uses FLEXlm 11.8 for all versions except the Windows VC++ 7.1 version, which continues to use FLEXlm 11.6. If you are upgrading from a version of the MÄK Data Logger that used an older version of FLEXlm, you must upgrade your license management files. You do not need a new license. Licenses are forward compatible.

The License Manager files are not part of the Logger installer. You can download them at:

- Windows: <ftp://ftp.mak.com/out/flexlm11-8-win-setup.exe>
- Linux: <ftp://ftp.mak.com/out/flexlm11-8-linux.tar.gz>

A license manager FAQ is available at:

<http://www.mak.com/support/faq.php#license>

ODBC Driver Support

The Logger database export feature supports the following ODBC drivers:

- MySQL ODBC 3.51
- Microsoft Access.

Using Libraries and Binaries Built with Visual Studio 2005 and Later

All MÄK products built with Microsoft Visual Studio require the C Runtime Library to function. The C runtime libraries have always been available from Microsoft for download, they are also installed on a user's machine when a Microsoft compiler is installed. The C runtime libraries are not part of the normal Windows installation. For customers who plan to use MÄK products on machines that do not have a compiler installed, MÄK has historically distributed a copy of the C Runtime Libraries with MÄK products. These libraries were put in the *bin* directory used by the MÄK products. MÄK products would then use the libraries in the *bin* directory and customers would not have a problem if copies of the libraries were not already installed.

Unfortunately, with the release of the new C Runtime Libraries required by Microsoft Visual Studio 2005 (MSVC++8.0) and later, the libraries can no longer just be copied into the *bin* directory of an application. The libraries need to be installed correctly into Windows system folders. (The process is actually a little more complicated, a manifest file needs to be created to tell Windows where to find the libraries.)

To accommodate this change, MÄK is distributing the Windows installer for the C runtime libraries with all MÄK products released for MSVC++8.0 and later. The 32-bit installer is named *vcredist_x86.exe*; the 64-bit installer (if supported) is named *vcredist_x64.exe*. They are in the base directory of any installed MÄK product that requires them.

Running the installer requires Administrator privileges for the machine the installer is run on. MÄK has chosen to not integrate the MÄK installer and the Microsoft installer so as not to require users to have Administrator privileges to install MÄK products. Therefore, if you who do not have a compiler installed, or get error messages like “Software has not been installed correctly, please re-install”, you must apply the patch.

For more information see this Microsoft URL:

<http://msdn2.microsoft.com/en-us/library/ms235299.aspx>



You must ensure that the preprocessor defines `_SECURE_SCL=0`, and `_HAS_ITERATOR_DEBUGGING=0` are set for MSVC++8.0 and MSVC++9.0 builds. If these are not set, random crashes and assertions may be encountered during runtime.

Patch Required for AMD Dual-processor Windows PCs

VR-Link-based products use a high resolution counter for time calculations on Windows PCs. Customers who are running Windows on PCs with multiple AMD Athlon 64-bit processors may notice clock jitter, which may cause time in MÄK products to run backwards. This occurs when the Windows scheduler changes the CPU the MÄK process is using. If the high resolution counters on each processor are not synchronized, the application may witness a decrease in the high resolution counter value stored in the processor causing an incorrect time calculation. To fix this problem customers, apply the AMD Dual-Core Optimizer patch provided by AMD. You can get the patch at:

http://www.amd.com/us-en/Processors/TechnicalResources/0,,30_182_871_9706,00.html



If you get an error when you try to access this URL, reload the page.

Network Compatibility

HLA only

Logger 4.4.1 was built against VR-Link 3.13.2 and is compliant with:

- ♦ RPR-FOM 1.0 and a subset of 2.0 (draft 6, 14, and 17)
- ♦ MÄK RTI 3.x.x (please see “[Known Problems](#),” on page 1-7.)

DIS only

Logger 4.4.1 supports DIS 4, 5, and 6.

RTI Support

We recommend using the latest version of the MÄK RTI. However, the Logger should work with other RTIs that conform to the HLA 1.3 specification or the IEEE 1516 SISO DLC API and are built using the same compiler as your version of the Logger.

When using the MÄK RTI, remember to make sure that the Logger can find your FED file, and optional *rid.mtl*, file by either putting them in the directory from which you are running, or by setting the environment variable the directory that contains them.

FOM Support

Logger 4.4.1 has built-in support for versions 0.5, 0.7, 0.8, 1.0, and 2.0, drafts 6, 14, and 17, of the RPR FOM. It also supports VR-Link's ability to support alternative FOMs through the FOM Mapper. By default, Logger 4.4.1 uses RPR FOM 1.0.

For information about FOM mapping and selecting the correct FOM Configuring the Logger for HLA in Chapter 2, "Installing and Configuring the Logger", in *MÄK Data Logger Users Guide*.

New Features

Logger 4.4.1 is a maintenance release. This release is due, in part, to a need to relink with the MÄK RTI (version 4.0.1) due to problems with Linux library versioning in MÄK RTI 4.0. This release has the following new features:

- ♦ The limit on site and applications IDs in the DIS connection dialog box has been increased from 255 to 65535.
- ♦ The `SetRemoteControlId id` parameter has been renamed `remoteControlId`. (This parameter was incorrectly documented as `identifier` in *MÄK Data Logger Users Guide*.)
- ♦ Two new command line arguments have been added:
 - `--remoteControlID ID`. Specifies the remote control ID. Range: 0-65535.
 - `--allowAllRemoteControl {True | False}`. Determines whether remote control PDUs are filtered based on the Logger's site-application-entityNum triplet. For details, please the description of the `allowAllRemoteControl` parameter of the `SetRemoteControlSettings` command.

Documentation Updates

The description of the `allowAllRemoteControl` of the `SetRemoteControlSettings` command has changed (Table 11-5 in *MÄK Data Logger Users Guide*), as follows:

`allowAllRemoteControl` determines whether remote control PDUs are filtered based on the Logger's site-application-entityNum triplet, where the Logger's remote control ID is used for the entityNum. By default this parameter is disabled and all remote control PDUs are accepted. When enabled, the Logger accepts remote control PDUs when the receiver ID matches the Logger. The receiver ID can have wildcard values (0xFFFF or unsigned int = 65535) in any of the triplet values (for example, 1:1:65535 matches any Logger with site = 1 and application ID = 1). Remote control PDUs are filtered when they do not match. By default, the Logger's remote control ID equals its site ID. The remote control ID can be changed using the `SetRemoteControlId` command.

Bug Fixes

The following bugs (with defect tracking numbers) have been fixed in this release:

- The RPR FOM formatting of the `isPartOf` attribute was incorrect. It did not insert the appropriate number of padding bytes.
- The `SetRemoteControlSettings` command `recordRemoteControl` parameter did not work. Setting this value to false now causes the Logger to engage the filter for the remote control PDU or interaction class as appropriate.
- This release changes how the Logger filters remote control PDUs in DIS when the `SetRemoteControlSettings` command `allowAllRemoteControl` parameter is disabled. Instead of using only the site field of the remote control receiver ID, the Logger now uses the site-application-entityNum triplet, where the Logger's remote control ID is used for the entityNum. The Logger accepts remote control PDUs when the receiver ID matches the Logger. The receiver ID can have wildcard values (0xFFFF or unsigned int = 65535) in any of the triplet values (for example, 1:1:65535 matches any Logger with site = 1 and application ID = 1).

Known Problems

This section lists known problems with Logger functionality:

- ♦ The HLA 1516 Logger crashes on exit if you use the MÄK RTI 2.4.2 through 3.1.1. This is due to a problem in the RTI.
- ♦ Logger files that include the VR-Forces embarkation feature in DIS exercises, such as *embarkdemoDIS.lgr*, do not reset embarkation status after time jumps. To correct the embarkation state of entities, stop playback and start playing the file from the beginning.
- ♦ If you are running the Logger in HLA 1516 with the MÄK RTI, and the RTI parameter `RTI_enableLrcWebService` is enabled, the Logger might pause when starting or stopping recording. This pause can be 30 seconds or longer and the Logger might not be responsive during this time.

Playing HLA 1.3 Logger Files in HLA 1516 Federations

In general, the Logger uses the same file format for HLA 1.3 and HLA 1516 Logger files. You can record a Logger file from an HLA 1.3 federation, and play it back into an HLA 1516 federation (and vice versa). However, you must use the same FOM representation during record and playback. For example, if you use an HLA-1.3-style FED file during recording, you must use an HLA-1.3 style FED file during playback, even if you are using the IEEE 1516 version of the Logger and an IEEE 1516 RTI. (VR-Link and the MÄK RTI allow you to use HLA-1.3-style FED files with IEEE-1516-based federates, and vice versa).

The main reason that consistency in FOM format is necessary is that IEEE 1516 uses different names for the "Root" classes of the Object class hierarchies. A 1.3-style FED file requires a Root class a 1516-style XML files requires a Root class called *HLAObjectRoot*. The reason this can be a problem is that if the Logger is playing an HLA-1.3-based Logger file into a federation that is using a 1516-based XML file it might come across an instance of a class called, for example, "ObjectRoot.Vehicle". If it tries to register an object of this class, the RTI will complain that no such class exists. There might be a class called *HLAObjectRoot.Vehicle* in the current FOM, but the RTI or Logger does not know that this is actually the same class.

