vms Software

SQL Relay Client for OpenVMS I64

January 2021

1. Introduction

Thank you for your interest in this port of the SQL Relay client API to VSI OpenVMS I64. The current release of the SQL Relay client API or OpenVMS is based on the SQL Relay 1.9.0 open source distribution.

SQL Relay is an open source database connection management solution that resides between your application and the database, providing functionality not typically provided by the database directly, including persistent database connection pooling, proxing, throttling, high availability, query routing, query filtering, query translation, and connection scheduling. Of particular interest from an OpenVMS perspective are the proxying capabilities of SQL Relay, which can be used to facilitate access to databases from unsupported platforms. Databases that can be accessed via SQL Relay using the client API include Oracle, Sybase, Microsoft SQL Server, IBM DB2, MySQL, MariaDB, PostgreSQL, Firebird, and SQLite, as well as ODBC data sources. For more information see http://sqlrelay.sourceforge.net/index.html.

This OpenVMS port of the SQL Relay client API includes all functionality provided by the Open Source release, including SSL/TLS support (based on OpenSSL 1.1.1g). Also included is a wrapper API that makes it easier to use the SQL Client with OpenVMS programming languages other than C/C++, such as COBOL, FORTRAN, Pascal, and BASIC. Additionally, the kit provides several SQL Relay command line tools, including `sqlrsh` (an interactive tool similar to Oracle SQL*Plus), and `sqlr-export` and `sqlr-import`, which can be used to export and import data from XML files on a per-table basis.

2. Acknowledgements

VMS Software Inc. would like to acknowledge the work of David Muse and the firstworks SQL Relay development team (http://www.firstworks.com/index.html) for their ongoing efforts in developing and supporting this open source software.

3. What's new in this release

For a detailed description of the features and bug fixes included in this release, please refer to the `ChangeLog` file in the source repository (git://git.code.sf.net/p/sqlrelay/sqlrelay).

4. Requirements

The kit you are receiving has been compiled and built using the operating system and compiler versions listed below. While it is highly likely that you will have no problems installing and using the kit on systems running higher versions of the operating system or products listed, we cannot say for sure that you will be so lucky if your system is running older versions.

- VSI OpenVMS Version 8.4-1H1 or higher

- VSI TCP/IP, HPE TCP/IP Services for OpenVMS, or MultiNet TCP/IP
- C or other language compiler (depending upon which language or languages you intend to use to develop applications using the SQL Relay client API)
- OpenSSL 1.1.1 (statically linked into the supplied SQL Relay client API shareable images)

  Note that if you wish to statically link application code requiring with the supplied object libraries and require SSL/TLS support, it will be necessary to link with a comparable OpenSSL distribution.

In addition to the above requirements, it is assumed that the reader has a good knowledge of OpenVMS and of software development in the OpenVMS environment.

5. Recommended reading

It is recommended that developers and administrators read the extensive documentation provided on the SQL Relay web site (http://sqlrelay.sourceforge.net/documentation.html) and that developers carefully examine the provided sample programs before using the software.

6. Installing the kit

The kit is provided as an OpenVMS PCSI kit (`VSI-I64VMS-SQLRELAY-V0109-0A-1.PCSI`) that can be installed by a suitably privileged user using the following command:

```
$ PRODUCT INSTALL SQLRELAY
```

The installation will then proceed as follows (output may differ slightly from that shown):

```
Performing product kit validation of signed kits ...

The following product has been selected:
    VSI I64VMS SQLRELAY V1.9-0A            Layered Product

Do you want to continue? [YES]

Configuration phase starting ...

You will be asked to choose options, if any, for each selected
product and for any products that may be installed to satisfy
software dependency requirements.

Configuring VSI I64VMS SQLRELAY V1.9-0A

    VMS Software Inc.

* This product does not have any configuration options.

Execution phase starting ...

The following product will be installed to destination:
    VSI I64VMS SQLRELAY V1.9-0A            DISK$I64SYS:[VMS$COMMON.]

Portion done: 0%...20%...50%...70%...80%...90%...100%

The following product has been installed:
    VSI I64VMS SQLRELAY V1.9-0A            Layered Product
```

```
VSI I64VMS SQLRELAY V1.9-0A

    Post-installation tasks are required.


    To enable SQLRelay at system boot time, add the following lines
    To SYS$MANAGER:SYSTARTUP_VMS.COM:

        $ file := SYS$STARTUP:SQLRELAY$STARTUP.COM
        $ if f$search("''file'") .nes. "" then @'file'

    To disable SQLRelay at system shutdown, add the following lines
    To SYS$MANAGER:SYSHUTDWN.COM:

        $ file := SYS$STARTUP:SQLRELAY$SHUTDOWN.COM
        $ if f$search("''file'") .nes. "" then @'file'
```

Note: In addition to installing the SQL Relay client API, on OpenVMS it is also necessary to install and configure the SQL Relay server on the Linux or Windows server hosting the target database. A detailed description of installing and configuring the SQL Relay server software is beyond the scope of this document and the reader should refer to the documentation available on the SQL Relay web site (http://sqlrelay.sourceforge.net/documentation.html).

6.1. Post-installation steps

After the installation has successfully completed, include the commands displayed at the end of the installation procedure into `SYSTARTUP_VMS.COM` to ensure that the logical names required in order for developers to use the software are defined system-wide at start-up.

In addition to the logical name `SQLR$ROOT` (which points to the root of the SQL Relay client software installation), the logical name `SQLR$SHR` is also defined. This logical name points to the shareable image `SQLR$ROOT:[LIB]SQLR$SHR.EXE`, which can be linked with application code that uses the SQL Relay client API. Alternatively, it is possible to statically link application code with the object libraries found in the `SQLR$ROOT:[LIB]` directory.

Other logical names defined by the SQL Relay start-up script are as follows:

Logical name	Purpose
`SQLR$BASIC`	Location of the include file `sqlrdef.bas` for use with BASIC applications.
`SQLR$BIN`	Location of SQL Relay command line utilities (`sqlr-export.exe`, `sqlr-import.exe`, `sqlrsh.exe`)
`SQLR$COBOL`	Location of the include file `sqlrdef.cob` for use with COBOL applications.
`SQLR$EXAMPLES`	Location of example programs in BASIC, FORTRAN, Pascal, COBOL, and C.
`SQLR$FORTRAN`	Location of the include file `sqlrdef.for` for use with FORTRAN applications.
`SQLR$LIB`	Location of SQL Relay client API object libraries and shareable image.

SQLR$PASCAL	Location of the include file sqlrdef.pas for use with Pascal applications.

From a development perspective, for C/C++ programs it should be noted that symbols in the shareable image and object libraries are mixed-case, and developers should therefore use the C and C++ compiler option /NAMES=(AS_IS,SHORTENED) or include in their code appropriate #pragma directives (C only) to ensure that symbols are correctly resolved when linking. Developers will also need to include in their code the appropriate language header file, from SQLR$ROOT:[INCLUDE]. Symbols for the wrapper API that can be used with other OpenVMS programming languages are all upper-case.

6.2. Privileges and quotas

Generally speaking there are no special quota or privilege requirements for applications developed using the SQL Relay client API, although a high BYTLM is recommended, and SYSPRV, BYPASS, or OPER privilege will be required if applications developed using the library need to utilise privileged ports (ports below 1024).

The following quotas should be more than adequate for most purposes:

```
Maxjobs:          0  Fillm:       256  Bytlm:       128000
Maxacctjobs:      0  Shrfillm:      0  Pbytlm:           0
Maxdetach:        0  BIOlm:       150  JTquota:       4096
Prclm:           50  DIOlm:       150  WSdef:         4096
Prio:             4  ASTlm:       300  WSquo:         8192
Queprio:          4  TQElm:       100  WSextent:     16384
CPU:         (none)  Enqlm:      4000  Pgflquo:     256000
```

7. Sample applications

The directory SQLR$ROOT:[EXAMPLES] contains several simple example programs written in C, COBOL, FORTRAN, BASIC, and Pascal that serve to illustrate the usage of the API. The command procedure SQLR$ROOT:[EXAMPLES]BUILD_EXAMPLES.COM can be used to build all of the example programs, and assumes that you have all of the relevant language compilers installed on the system in question. If this is not the case, it will be necessary to modify the command procedure to remove or comment out any language examples that you do not wish to build.

These simple example programs are intended to provide an introduction to the SQL Relay API and to hopefully serve as a basis for the development of more sophisticated applications. Note that it will be necessary to have available an appropriately configured SQL Relay server environment in order to run the example programs, and the example programs will need to be modified to specify the correct username, password, network address, and database connection details for your environment.

8. What's missing?

The supplied kit for VSI OpenVMS includes all functionality supported by the open source SQL Relay client API. In addition, the port includes a language-agnostic API that makes it

straightforward to write applications using 3GL languages such as COBOL, FORTRAN, Pascal, and BASIC.

Subsequent releases of the SQL Relay client for OpenVMS will include addition development tools and utilities, including embedded SQL processors for various programming languages and an enhanced interactive query tool. It is anticipated that the inclusion of these facilities and the scope of the functionality they provide will evolve over the course of several product releases.