MSILO Module

Daniel-Constantin Mierla
FhG FOKUS

Edited by

Daniel-Constantin Mierla

MSILO Module

Edited by and Daniel-Constantin Mierlaand Daniel-Constantin Mierla

Copyright © 2003 FhG FOKUS

Revision History

Revision $Revision: 1.2 $ $Date: 2003/07/16 15:17:04 $

Table of Contents

1. User's Guide **1**
1.1. Overview 1
1.2. Dependencies 1
1.2.1. SER modules 1
1.2.2. External libraries or applications 1
1.3. Exported Parameters 1
1.3.1. `db_url` (string) 2
1.3.2. `db_table` (string) 2
1.3.3. `registrar` (string) 2
1.3.4. `expire_time` (int) 2
1.3.5. `check_time` (int) 3
1.3.6. `clean_period` (int) 3
1.3.7. `use_contact` (int) 3
1.4. Exported Functions 4
1.4.1. `m_store(mode)` 4
1.4.2. `m_dump()` 4
1.5. Installation & Running 4
1.5.1. Database setup 4
1.5.2. SER config file 5
2. Developer's Guide **9**
3. Frequently Asked Questions **10**

List of Examples

1-1. Set the “db_url” parameter 2
1-2. Set the “db_table” parameter 2
1-3. Set the “registrar” parameter 2
1-4. Set the “expire_time” parameter 3
1-5. Set the “check_time” parameter 3
1-6. Set the “clean_period” parameter 3
1-7. Set the “param_name” parameter 3
1-8. msilo sql script 4
1-9. SER config script - sample msilo usage 5

Chapter 1. User's Guide

1.1. Overview

This modules provides offline message storage for the SIP Express Router. It stores received messages for an offline user and sends them when the user becomes online.

For each message, the modules stores "Request-URI" ("R-URI") only if it is a complete address of record ("username@hostname"), URI from "To" header, URI from "From" header, incoming time, expiration time, content type and body of the message. If "R-URI" is not an address of record (it might be the contact address for current SIP session) the URI from "To" header will be used as R-URI.

When the expiration time passed, the message is discarded from database. Expiration time is computed based on incoming time and one of the module's parameters.

Every time when a user registers with SER, the module is looking in database for offline messages intended for that user. All of them will be sent to contact address provided in REGISTER request.

It may happen the SIP user to be registered but his SIP User Agent to have no support for MESSAGE request. In this case it should be used the "failure_route" to store the undelivered requests.

1.2. Dependencies

1.2.1. SER modules

The following modules must be loaded before this module:

- *database module* - mysql, dbtext or other module that implements the "db" interface and provides support for storing/receiving data to/from a database system.
- *TM*--transaction module--is used to send SIP requests.

1.2.2. External libraries or applications

The following libraries or applications must be installed before running SER with this module:

- *none*.

1.3. Exported Parameters

1.3.1. `db_url` (string)

Database URL.

Default value is "sql://root@localhost/msilo".

Example 1-1. Set the "db_url" parameter

```
...
modparam("msilo", "db_url", "sql://user:passwd@host.com/dbname")
...
```

1.3.2. `db_table` (string)

The name of table where to store the messages.

Default value is "silo".

Example 1-2. Set the "db_table" parameter

```
...
modparam("msilo", "db_table", "silo")
...
```

1.3.3. `registrar` (string)

The SIP address used to inform users that destination of their message is not online and the message will be delivered next time when that user goes online. If the parameter is not set, the module will not send any notification. All requests intended for this SIP address will not be stored for lately delivery.

Default value is "NULL".

Example 1-3. Set the "registrar" parameter

```
...
modparam("msilo", "registrar", "sip:registrar@iptel.org")
...
```

1.3.4. `expire_time` (int)

Expire time of stored messages - seconds. When this time passed, the message is silently discarded from database.

Default value is "259200 (72 hours = 3 days)".

Example 1-4. Set the "expire_time" parameter

```
...
modparam("msilo", "expire_time", 36000)
...
```

1.3.5. `check_time` (int)

Timer interval to check if dumped messages are sent OK - seconds. The module keeps each request send by itself for a new online user and if the reply is 2xx then the message is deleted from database.

Default value is "30".

Example 1-5. Set the "check_time" parameter

```
...
modparam("msilo", "check_time", 10)
...
```

1.3.6. `clean_period` (int)

Number of "check_time" cycles when to check if there are expired messages in database.

Default value is "5".

Example 1-6. Set the "clean_period" parameter

```
...
modparam("msilo", "clean_period", "3")
...
```

1.3.7. `use_contact` (int)

Turns on/off the usage of the Contact address to send notification back to sender whose message is stored by MSILO.

Default value is "1 (0 = off, 1 = on)".

Example 1-7. Set the "param_name" parameter

```
...
modparam("msilo", "use_contact", 0)
...
```

1.4. Exported Functions

1.4.1. `m_store(mode)`

The method stores certain parts of the current SIP request (it should be called when the request type is MESSAGE and the destination user is offline or his UA does not support MESSAGE requests). If the user is registered with a UA which does not support MESSAGE requests you should not use mode="0" if you have changed the request uri with the contact address of user's UA.

Meaning of the parameters is as follows:

- *mode* - specifies what to save as R-URI.
  - "0" - first check if new_uri is an address of record. If yes, then use it and store it as R-URI, otherwise look at R-URI and, if necessary, at URI from "To" header.
  - "1" - look first at R-URI and then at URI from "To" header.
  - "2" - look only at URI form "To" header.

1.4.2. `m_dump()`

The method sends stored messages for the SIP user that is going to register to his actual contact address. The method should be called when a REGISTER request is received and the "Expire" header has a value greater than zero.

1.5. Installation & Running

1.5.1. Database setup

Before running SER with msilo, you have to setup the database table where the module will store the message. For that, if the table was not created by the installation script or you choose to install everything by yourself you can use the following SQL script (good for MySQL) as template. Database and table name can be set with module parameters so they can be changed, but the name of the columns must be as they are in the next SQL script.

Example 1-8. msilo sql script

```
...

-- SQL script for MSILO module

DROP DATABASE IF EXISTS msilo;

-- create a database for storage
CREATE DATABASE msilo;

USE msilo;

-- create the table
CREATE TABLE silo(
      -- unique ID per message
    mid INTEGER NOT NULL AUTO_INCREMENT PRIMARY KEY,
      -- src address - From URI
    src_addr VARCHAR(255) NOT NULL DEFAULT "",
      -- dst address - To URI
    dst_addr VARCHAR(255) NOT NULL DEFAULT "",
      -- Request-URI
    r_uri VARCHAR(255) NOT NULL DEFAULT "",
      -- incoming time
    inc_time INTEGER NOT NULL DEFAULT 0,
      -- expiration time
    exp_time INTEGER NOT NULL DEFAULT 0,
      -- content type
    ctype VARCHAR(32) NOT NULL DEFAULT "text/plain",
      -- body of the message
    body BLOB NOT NULL DEFAULT ""
);


...
```

1.5.2. SER config file

Next picture displays a sample usage of msilo.

Example 1-9. SER config script - sample msilo usage

```
...
#
# MSILO usage example
#
#


debug=9           # debug level (cmd line: -dddddddddd)
fork=no           # don't fork
log_stderror=yes  # log to stderr (cmd line: -E)
```

```
children=2        # number of children
check_via=no      # (cmd. line: -v)
dns=off           # (cmd. line: -r)
rev_dns=off       # (cmd. line: -R)
port=5060

listen=10.0.0.2   # listen address

# ------------------ module loading ----------------------------------

loadmodule "../sip_router/modules/print/print.so"
loadmodule "../sip_router/modules/textops/textops.so"

loadmodule "../sip_router/modules/sl/sl.so"
loadmodule "../sip_router/modules/mysql/mysql.so"
loadmodule "../sip_router/modules/maxfwd/maxfwd.so"
loadmodule "../sip_router/modules/msilo/msilo.so"
loadmodule "../sip_router/modules/tm/tm.so"
loadmodule "../sip_router/modules/registrar/registrar.so"
loadmodule "../sip_router/modules/usrloc/usrloc.so"

# ----------------- setting module-specific parameters ---------------

# -- registrar params --

modparam("registrar", "default_expires", 120)

# -- registrar params --

modparam("usrloc", "db_mode", 0)

# -- msilo params --

modparam("msilo","db_url","sql://user:xxx@127.0.0.1/msilo")
modparam("msilo","registrar","sip:registrar@mydomain.com")

# -- tm params --

modparam("tm", "fr_timer", 10 )
modparam("tm", "fr_inv_timer", 15 )
modparam("tm", "wt_timer", 10 )


route{
    if ( !mf_process_maxfwd_header("10") )
    {
        sl_send_reply("483","To Many Hops");
        drop();
    };


    if (uri==myself) {
    {
        # for testing purposes, simply okay all REGISTERs
        if (method=="REGISTER")
        {
            save("location");
```

```
        log("REGISTER received -> dumping messages with MSILO\n");

        # MSILO - dumping user's offline messages
        if (m_dump())
        {
            log("MSILO: offline messages dumped - if they were\n");
        }else{
            log("MSILO: no offline messages dumped\n");
        };
        break;
    };

    # domestic SIP destinations are handled using our USRLOC DB

    if(!lookup("location"))
    {
        if (! t_newtran())
           {
            sl_reply_error();
            break;
           };
        # we do not care about anything else but MESSAGEs
        if (!method=="MESSAGE")
        {
            if (!t_reply("404", "Not found"))
            {
                sl_reply_error();
            };
            break;
        };
        log("MESSAGE received -> storing using MSILO\n");
        # MSILO - storing as offline message
        if (m_store("0"))
        {
            log("MSILO: offline message stored\n");
            if (!t_reply("202", "Accepted"))
            {
                sl_reply_error();
            };
        }else{
            log("MSILO: offline message NOT stored\n");
            if (!t_reply("503", "Service Unavailable"))
            {
                sl_reply_error();
            };
        };
        break;
    };
    # if the downstream UA does not support MESSAGE requests
    # go to failure_route[1]
    t_on_failure("1");
    t_relay();
    break;
};

# forward anything else
t_relay();
```

```
}

failure_route[1] {
    # forwarding failed -- check if the request was a MESSAGE
    if (!method=="MESSAGE")
    {
        break;
    };

    log(1,"MSILO:the downstream UA doesn't support MESSAGEs\n");
    # we have changed the R-URI with the contact address, ignore it now
    if (m_store("1"))
    {
        log("MSILO: offline message stored\n");
        t_reply("202", "Accepted");
    }else{
        log("MSILO: offline message NOT stored\n");
        t_reply("503", "Service Unavailable");
    };
}



...
```

Chapter 2. Developer's Guide

The module does not provide any sort of API to use in other SER modules.

Chapter 3. Frequently Asked Questions

1. Where can I find more about SER?

Take a look at http://iptel.org/ser.

2. Where can I post a question about this module?

First at all check if your question was already answered on one of our mailing lists:

- http://mail.iptel.org/mailman/listinfo/serusers
- http://mail.iptel.org/mailman/listinfo/serdev

E-mails regarding any stable version should be sent to <serusers@iptel.org> and e-mail regarding development versions or CVS snapshots should be send to <serdev@iptel.org>.

If you want to keep the mail private, send it to <serhelp@iptel.org>.

3. How can I report a bug?

Please follow the guidelines provided at: http://iptel.org/ser/bugs