The RSPLIB RSerPool Implementation Handbook

Thomas Dreibholz
Institute for Experimental Mathematics
Ellernstraße 29, 45326 Essen, Germany
dreibh@iem.uni-due.de

November 23, 2010

Contents

1 Introduction

This is the documentation for the RSPLIB RSerPool package. It contains information how to install and make use of RSPLIB. For a detailed introduction to RSerPool and its concepts itself, see [2]. For questions about RSerPool and RSPLIB, see our mailing lists at [4].

2 What is RSerPool?

Figure 1 provides an illustration of the Reliable Server Pooling (RSerPool) architecture as defined by the Internet Draft [23]; the protocol stack of RSerPool is shown in figure 2. An RSerPool scenario consists of three component classes [23]: servers of a pool are called *pool elements* (PE). Each pool is identified by a unique *pool handle* (PH) in the handlespace, which is the set of all pools.

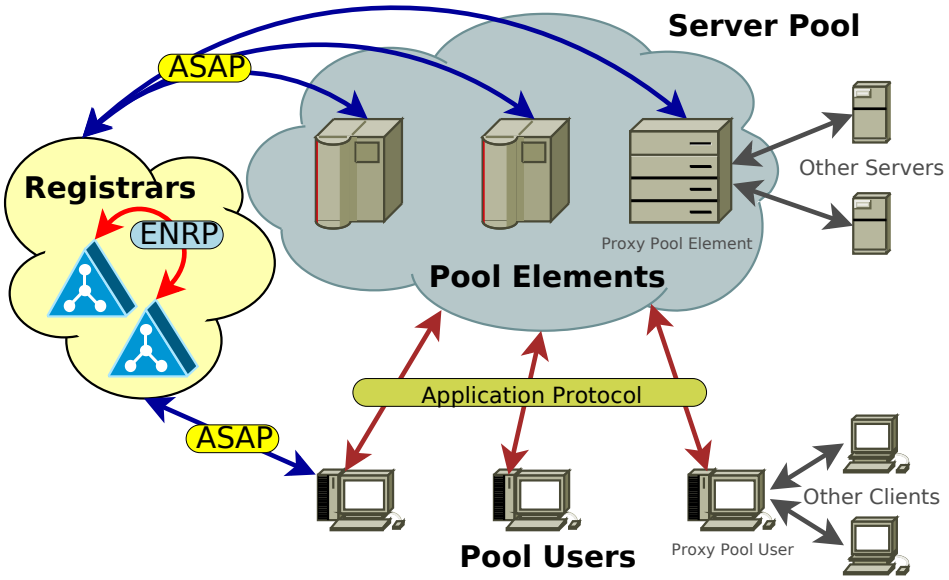

Figure 1: The RSerPool Architecture

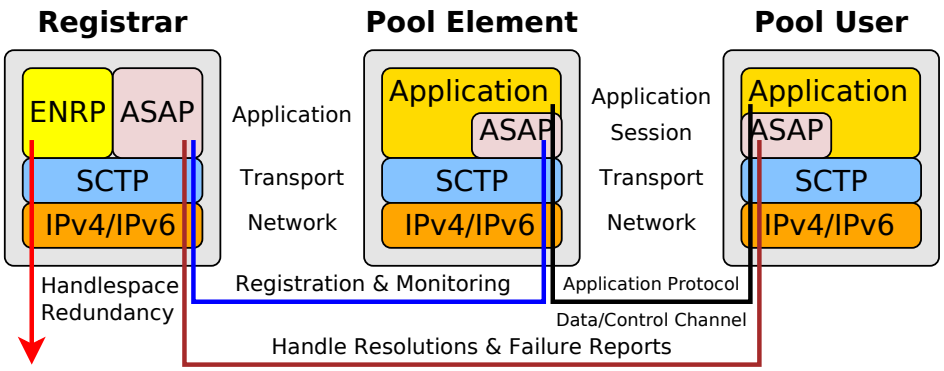

Figure 2: The RSerPool Protocol Stack

The handlespace is managed by *pool registrars* (PR). PRs of an *operation scope* (the domain which is covered by the handlespace, e.g. an organization or building) synchronize their view of the handlespace using the Endpoint haNdlespace Redundancy Protocol (ENRP [30, 16, 26]), transported via SCTP [24, 19, 21, 20].

Unlike already available solutions in the area of GRID and high-performance computing, the fundamental property of RSerPool is to be"lightweight", i.e. it must also be usable low-performance devices like telecommunications equipment or routers. This property restricts the RSerPool architecture to the management of pools and sessions only, but on the other hand makes a very efficient realization possible [11, 8, 5]. In particular, an operation scope has a limited range, e.g. a company or organization; RSerPool does not intend to scale to the whole Internet. Nevertheless, it is assumed that PEs can be distributed globally, for their service to survive localized disasters [9].

PEs choose an arbitrary PR to register into a pool by using the Aggregate Server Access Protocol (ASAP [25, 3, 26]), again transported via SCTP. Upon registration at a PR, the chosen PR becomes the Home-PR (PR-H) of the newly registered PE. A PR-H is responsible for monitoring its PEs' availability by using ASAP Endpoint Keep-Alive messages (to be acknowledged by the PE within a given timeout) and propagates the information about its PEs to the other PRs of the operation scope via ENRP Update messages.

A client is called *pool user* (PU) in RSerPool terminology. To access the service of a pool given by its PH, a PE has to be selected. This selection procedure – called *handle resolution* – is performed by an arbitrary PR of the operation scope. A PU can request a handle resolution from a PR using the ASAP protocol. The PR selects PE identities by using a pool-specific server selection rule, denoted as *pool policy*. A set of adaptive and non-adaptive pool policies is defined in [14, 15]; for a detailed discussion of these policies, see [2, 6, 7, 17, 34, 12, 31, 33, 32].

3 Installation

In this section, the installation of the RSPLIB package is described.

3.1 Installation of the SCTP Protocol

The first step is to decide which SCTP implementation should be used. You have the choice between kernel SCTP and userland SCTP. In most cases, you probably want kernel SCTP. It is most efficient and the implementation should be sufficiently stable.

3.1.1 Installation of Kernel SCTP for Linux

A SCTP kernel module is already provided by all major Linux distributions. To load it into the kernel, call:

```
sudo modprobe sctp
```

In order to load it permanently, add a line "sctp" to /etc/modules. After that, the module will be loaded automatically at boot time. Also make sure that the SCTP include files are installed (in particular: /usr/include/netinet/sctp.h). If they are not installed, install the package libsctp-dev (or similar name). For Debian/Ubuntu Linux, you can use:

```
sudo apt-get install libsctp-dev
```

3.1.2 Installation of Userland SCTP SCTPLIB/SOCKETAPI

If you decide to use our userland SCTP implementation SCTPLIB [27], the following steps have to be performed. The SCTPLIB userland SCTP implementation consists of two packages: SCTPLIB containing the actual SCTP implementation and SOCKETAPI containing a BSD sockets API for SCTPLIB. You need both. Get the latest versions from [4] (http://tdrwww.iem.uni-due.de/dreibholz/rserpool http://tdrwww.iem.uni-due.de/dreibholz/rserpool). First, unpack, configure and installSCTPLIB:

```
tar xzf sctplib-<version>.tar.gz
cd sctplib-<version>
./configure <Options>
make
sudo make install
```

Useful options are "`--enable-static --disable-shared --enable-maintainer-mode`" to generate a static library with debug symbols. This is useful for debugging purposes (e.g. memory leak detection using Valgrind [29]). To use SCTP over UDP (defined in [28]), use "`--enable-sctp-over-udp`". The next step is to install the socketapi package:

```
tar xzf socketapi-<version>.tar.gz
cd socketapi-<version>
./configure <Options>
make
sudo make install
```

Again, you can use "`--enable-static --disable-shared --enable-maintainer-mode`" to generate a static library with debug symbols. If you want to use kernel SCTP and SCTP over UDP simultaneously, add "`--enable-sctp-over-udp`". In this case, the socketapi will not abort if it finds a loaded kernel SCTP module.

3.2 Installation of the RSPLIB Package

After installing SCTP support, the RSPLIB package can be installed.

3.2.1 Preparation Work

In order to prepare your system for the installation of the RSPLIB package, it is recommended to do the following tasks:

- Add the environment variable QTDIR, setting the path to the installation of Qt. Qt is a GUI library and required for the Fractal PU example. Without Qt and properly set QTDIR, the client's compilation can be skipped. The ./configure script expects the environment variable QTDIR set to the Qt directory. You may need to set it appropriately, e.g. "export QTDIR=/usr/share/qt4" for Ubuntu Linux. This setting depends on your distribution; use "locate qwidget.h" to find out the directory.

- The Component Status Protocol (CSP) can be used to send status messages of PRs, PEs and PUs to a central monitoring program (cspmonitor, to be explained later). This is a helpful feature to keep an overview of large, distributed test setups. You can set a default address and report interval for CSP by defining two environment variables:
  `CSP_SERVER=<Address>:<Port>`
  `CSP_INTERVAL=<Report interval in milliseconds>`
  Useful settings are CSP_SERVER=127.0.0.1:2960 and CSP_INTERVAL=333.

- For debugging, it is useful to turn on the generation of core dumps. Under the bash shell, this can be done by"ulimit -c unlimited".

To make all settings above permanent, you can append them to your shell configuration (usually ~/.bashrc). Example (your settings may be different!):

```
...
export QTDIR=/usr/share/qt4
export CSP_SERVER=127.0.0.1:2960
export CSP_INTERVAL=333
ulimit -c unlimited
```

In order to use RSerPool, your host needs at least one multicast-capable network interface with at least a private IP address (i.e. 192.168.x.y; 10.a.b.c; 172.16.n.m - 172.31.i.j). If your host is already connected to a network and has an IP address, everything should be fine. For testing with a non-connected host, you can just set up a dummy interface:

```
sudo ifconfig dummy0 10.255.255.1 netmask 255.255.255.0 broadcast 10.255.255.255
up multicast
```

In order to permanently set up a dummy interface, you can add the following lines to /etc/network/interfaces (Debian/Ubuntu Linux; may be different for other distributions!):

```
auto dummy0
iface dummy0
inet static address 172.31.249.1 netmask 255.255.255.252
```

(You may need to change this!)

```
post-up ip link set dummy0 up multicast on
pre-down ip link set dummy0 up multicast off
```

After appending these lines, they will be loaded automatically each time a new bash shell is started. Your system should now be ready to install the RSPLIB package.

3.2.2 Configuration and Installation

In order to install RSPLIB, get the latest version from [4] (http://tdrwww.iem.uni-due.de/dreibholz/rserpool), unpack, configure and compile it:

```
tar xzf rsplib-<version>.tar.gz
cd rsplib-<version>
./configure <Options>
make
sudo make install
```

(This step is optional and not needed to run the examples!)

You may use the following options:

`--enable-kernel-sctp` Enables usage of kernel SCTP (**default**).

`--disable-kernel-sctp` Use userland SCTP (i.e. SCTPLIB/SOCKETAPI) instead of kernel SCTP.

`--enable-qt` Enables Qt usage; this is necessary for the Fractal Generator client. Without Qt, the client's compilation will be skipped. The ./configure script expects the environment variable QTDIR set to the Qt directory. You may need to set it appropriately, e.g. "export QTDIR=/usr/share/qt4" for Ubuntu Linux. This setting depends on your distribution; use "locate qwidget.h" to find out the directory. It is recommended to add the setting of QTDIR to your shell configuration (usually ~/.bashrc). (**default**)

`--disable-qt` Disables Qt usage; the Fractal Generator client will**not** be available.

`--enable-shared` Create a shared library.

`--disable-shared` Disable the creation of a shared library (recommended for debugging).

`--enable-static` Create a static library (recommended for debugging).

`--disable-static` Disable the creation of a static library.

`--enable-maintainer-mode` Enable the creation of debug symbols (-g) and turn off compiler optimization. Necessary for debugging, but makes programs much larger and slower. (**default**)

`--disable-maintainer-mode` Disables maintainer mode. The code will be optimized and without debugging symbols. Node: Please turn on the maintainer mode if you discover any problems and report bugs to us!

`--with-max-loglevel` Allows for reduction of the maximum logging verbosity. Setting a lower value here makes the programs smaller, at cost of reduces logging capabilities (**default: 9**).

`--enable-registrar-statistics` Adds registrar option to write statistics file.

`--disable-registrar-statistics` Do not compile in the statistics option. In this case, the dependency on LIBBZ2 is removed.

`--enable-hsmgtverify` Enable Handlespace Management verification. This is useful for debugging only; it makes the very PR slow!

`--disable-hsmgtverify` Turns off Handlespace Management verification. (**default**)

`--enable-csp` Enable the Component Status Protocol support (strongly recommended!) (**default**)

`--disable-csp` Turns the Component Status Protocol support off.

For learning how to work with RSPLIB, the following options are highly recommended:
`--enable-kernel-sctp --disable-shared --enable-static --enable-qt --enable-csp --enable-maintainer-mode`
The RSPLIB package also provides filter and coloring rules for WIRESHARK [22]. You can find them in the rsplib/wireshark/ subdirectory. Just copy the files "dfilters" and "colorfilters" to your WIRESHARK settings directory: ~/.wireshark (Ubuntu/Debian) or /root/.wireshark (you need root permission to do so, i.e. use "sudo"!). Optionally, you can also copy the file "preferences" (if you do not have your own preferences configured yet, otherwise this would overwrite them!). You do **not** need to install the provided dissectors, they have already contributed to the WIRESHARK developers and are included already!

3.3 Testing the Installation

To perform a test of the installation, start the following programs in the rsplib/ subdirectory of the RSPLIB package:

1. Start the CSP monitor, it will print out useful information about the components started:
   `./cspmonitor`

2. First, start a registrar:
   `./registrar`

3. Start a PE for the Fractal Generator service:
   `./server -fractal`
   The PE should find the PR and show its PE ID upon startup. When it shows the ID, it has successfully registered. If something does not work, use the parameter -loglevel=5 to increase the verbosity of the log output. Also refer to subsection 3.2.1 to check your system configuration. Have a look at the CSP monitor output. It should show the PR and PE.

4. Start a PU for the Fractal Generator service:
   `./fractalpooluser`
   You should now see the calculation progress in the PU's window. Also have a look at the CSP monitor output; it should show the PU.

5. Start more PEs, PUs and PRs. You can turn on the "unreliable mode" of the PE using the parameter -fgpfailureafter=20. When all PEs are in unreliable mode, you should see the failovers. You can also abort and restart the PRs. Also have a look at the CSP monitor output.

6. Start WIRESHARK, sniff on the "lo" (loopback, only local traffic) or the "any" interface. If you have set up the filter and coloring rules (see subsection 3.3), you can select some useful filters and get the RSerPool traffic nicely colorized.

4 The Programs

All installable programs in the rsplib/ subdirectory also have a manual page (suffix: .8). You can view the manual page in the rsplib/ directory using

```
man ./<program name>.8
```

After installation (make install, see subsection 3.2.2), the manual pages will also be available directly.

The programs included in rsplib/ subdirectory have the following purposes:

registrar The PR implementation.

server A PE which provides multiple services. The actual service started is given by command-line parameter.

cspmonitor The CSP monitor program to view status information of running components.

terminal A simple PU for services like Echo, Discard, Daytime and CharGen.

hsdump A ENRP-based test utility to dump the handlespace of a PR.

pingpongclient A simple PU for a request-response example service with cookie-based failover [1, 13].

calcappclient The PU for the CalcApp service used for performance measurements (see [2, 9, 18]).

fractalpooluser The PU for the FractalGenerator service.

scriptingclient The PU for the scripting service (remote script execution with input/output data transfer; see also [10]).

5 The RSPLIB API

5.1 Initialization/Clean-Up

5.1.1 rsp_initinfo()

5.1.2 rsp_freeinfo()

5.1.3 rsp_initarg()

5.1.4 rsp_initialize()

5.1.5 rsp_cleanup()

5.2 Basic Mode API

5.2.1 rsp_pe_registration()

5.2.2 rsp_pe_deregistration()

5.2.3 rsp_pe_failure()

5.2.4 rsp_getaddrinfo()

5.2.5 rsp_freeaddrinfo()

5.3 Enhanced Mode API Socket Functions

5.3.1 rsp_socket()

5.3.2 rsp_update_session_parameters()

5.3.3 rsp_bind()

5.3.4 rsp_listen()

5.3.5 rsp_getsockname()

5.3.6 rsp_getpeername()

5.3.7 rsp_close()

5.3.8 rsp_poll()

5.3.9 rsp_select()

5.3.10 rsp_getsockopt()

5.3.11 rsp_setsockopt()

5.4 Enhanced Mode API Pool Element Functions

5.4.1 rsp_register()

5.4.2 rsp_deregister()

5.4.3 rsp_accept()

5.4.4 rsp_connect()

5.5 Enhanced Mode API Pool User Functions

5.5.1 rsp_has_cookie()

5.5.2 rsp_forcefailover()

5.5.3 rsp_sendmsg()

5.5.4 rsp_send_cookie()

27th IEEE Local Computer Networks Conference (LCN) (Tampa, Florida/U.S.A., Oct. 2002), pp. 348–352. ISBN 0-7695-1591-6.

[2] DREIBHOLZ, T. *Reliable Server Pooling – Evaluation, Optimization and Extension of a Novel IETF Architecture*. PhD thesis, University of Duisburg-Essen, Faculty of Economics, Institute for Computer Science and Business Information Systems, Mar. 2007.

[3] DREIBHOLZ, T. Handle Resolution Option for ASAP. Internet-Draft Version 07, IETF, Individual Submission, July 2010. draft-dreibholz-rserpool-asap-hropt-07.txt, work in progress.

[4] DREIBHOLZ, T. Thomas Dreibholz's RSerPool Page, 2010.

[5] DREIBHOLZ, T., AND RATHGEB, E. P. Implementing the Reliable Server Pooling Framework. In *Proceedings of the 8th IEEE International Conference on Telecommunications (ConTEL)* (Zagreb/Croatia, June 2005), vol. 1, pp. 21–28. ISBN 953-184-081-4.

[6] DREIBHOLZ, T., AND RATHGEB, E. P. On the Performance of Reliable Server Pooling Systems. In *Proceedings of the IEEE Conference on Local Computer Networks (LCN) 30th Anniversary* (Sydney/Australia, Nov. 2005), pp. 200–208. ISBN 0-7695-2421-4.

[7] DREIBHOLZ, T., AND RATHGEB, E. P. The Performance of Reliable Server Pooling Systems in Different Server Capacity Scenarios. In *Proceedings of the IEEE TENCON '05* (Melbourne/Australia, Nov. 2005). ISBN 0-7803-9312-0.

[8] DREIBHOLZ, T., AND RATHGEB, E. P. An Evaluation of the Pool Maintenance Overhead in Reliable Server Pooling Systems. In *Proceedings of the IEEE International Conference on Future Generation Communication and Networking (FGCN)* (Jeju Island/South Korea, Dec. 2007), vol. 1, pp. 136–143. ISBN 0-7695-3048-6.

[9] DREIBHOLZ, T., AND RATHGEB, E. P. On Improving the Performance of Reliable Server Pooling Systems for Distance-Sensitive Distributed Applications. In *Proceedings of the 15. ITG/GI Fachtagung Kommunikation in Verteilten Systemen (KiVS)* (Bern/Switzerland, Feb. 2007), pp. 39–50. ISBN 978-3-540-69962-0.

[10] DREIBHOLZ, T., AND RATHGEB, E. P. A Powerful Tool-Chain for Setup, Distributed Processing, Analysis and Debugging of OMNeT++ Simulations. In *Proceedings of the 1st ACM/ICST International Workshop on OMNeT++* (Marseille/France, Mar. 2008). ISBN 978-963-9799-20-2.

[11] DREIBHOLZ, T., AND RATHGEB, E. P. An Evaluation of the Pool Maintenance Overhead in Reliable Server Pooling Systems. *SERSC International Journal on Hybrid Information Technology (IJHIT) 1*, 2 (Apr. 2008), 17–32. ISSN 1738-9968.

[12] DREIBHOLZ, T., AND RATHGEB, E. P. Reliable Server Pooling – A Novel IETF Architecture for Availability-Sensitive Services. In *Proceedings of the 2nd IEEE International Conference on Digital Society (ICDS)* (Sainte Luce/Martinique, Feb. 2008), pp. 150–156. ISBN 978-0-7695-3087-1.

[13] DREIBHOLZ, T., AND RATHGEB, E. P. Overview and Evaluation of the Server Redundancy and Session Failover Mechanisms in the Reliable Server Pooling Framework. *International Journal on Advances in Internet Technology (IJAIT) 2*, 1 (June 2009), 1–14. ISSN 1942-2652.

[14] DREIBHOLZ, T., AND TÜXEN, M. Reliable Server Pooling Policies. RFC 5356, IETF, Sept. 2008.

[15] DREIBHOLZ, T., AND ZHOU, X. Definition of a Delay Measurement Infrastructure and Delay-Sensitive Least-Used Policy for Reliable Server Pooling. Internet-Draft Version 06, IETF, Individual Submission, July 2010. draft-dreibholz-rserpool-delay-06.txt, work in progress.

[16] DREIBHOLZ, T., AND ZHOU, X. Takeover Suggestion Flag for the ENRP Handle Update Message. Internet-Draft Version 04, IETF, Individual Submission, July 2010. draft-dreibholz-rserpool-enrp-takeover-04.txt, work in progress.

[17] DREIBHOLZ, T., ZHOU, X., AND RATHGEB, E. P. A Performance Evaluation of RSerPool Server Selection Policies in Varying Heterogeneous Capacity Scenarios. In *Proceedings of the 33rd IEEE EuroMirco Conference on Software Engineering and Advanced Applications* (Lübeck/Germany, Aug. 2007), pp. 157–164. ISBN 0-7695-2977-1.

[18] DREIBHOLZ, T., ZHOU, X., RATHGEB, E. P., AND DU, W. A PlanetLab-Based Performance Analysis of RSerPool Security Mechanisms. In *Proceedings of the 10th IEEE International Conference on Telecommunications (ConTEL)* (Zagreb/Croatia, June 2009), pp. 213–220. ISBN 978-953-184-131-3.

[19] JUNGMAIER, A. *Das Transportprotokoll SCTP*. PhD thesis, Universität Duisburg-Essen, Institut für Experimentelle Mathematik, Aug. 2005.

[20] JUNGMAIER, A., RATHGEB, E. P., AND TÜXEN, M. On the Use of SCTP in Failover-Scenarios. In *Proceedings of the State Coverage Initiatives, Mobile/Wireless Computing and Communication Systems II* (Orlando, Florida/U.S.A., July 2002), vol. X, pp. 363–368. ISBN 980-07-8150-1.

[21] JUNGMAIER, A., SCHOPP, M., AND TÜXEN, M. Das Simple Control Transmission Protocol (SCTP) – Ein neues Protokoll zum Transport von Signalisierungsmeldungen über IP-basierte Netze. *Elektrotechnik und Informationstechnik – Zeitschrift des Österreichischen Verbandes für Elektrotechnik 117*, 6 (2000), 381–388.

[22] LAMPING, U., SHARPE, S., AND WARNICKE, E. Wireshark User's Guide, 2006.

[23] LEI, P., ONG, L., TÜXEN, M., AND DREIBHOLZ, T. An Overview of Reliable Server Pooling Protocols. Informational RFC 5351, IETF, Sept. 2008.

[24] STEWART, R. Stream Control Transmission Protocol. Standards Track RFC 4960, IETF, Sept. 2007.

[25] STEWART, R., XIE, Q., STILLMAN, M., AND TÜXEN, M. Aggregate Server Access Protcol (ASAP). RFC 5352, IETF, Sept. 2008.

[26] STEWART, R., XIE, Q., STILLMAN, M., AND TÜXEN, M. Aggregate Server Access Protocol (ASAP) and Endpoint Handlespace Redundancy Protocol (ENRP) Parameters. RFC 5354, IETF, Sept. 2008.

[27] TÜXEN, M. The sctplib Prototype, 2010.

[28] TÜXEN, M., AND STEWART, R. UDP Encapsulation of SCTP Packets. Internet-Draft Version 05, IETF, Individual Submission, July 2010. draft-tuexen-sctp-udp-encaps-05.txt, work in progress.

[29] VALGRIND DEVELOPERS. Valgrind Home, 2010.

[30] XIE, Q., STEWART, R., STILLMAN, M., TÜXEN, M., AND SILVERTON, A. Endpoint Handlespace Redundancy Protocol (ENRP). RFC 5353, IETF, Sept. 2008.

[31] ZHOU, X., DREIBHOLZ, T., AND RATHGEB, E. P. A New Approach of Performance Improvement for Server Selection in Reliable Server Pooling Systems. In *Proceedings of the 15th IEEE International Conference on Advanced Computing and Communication (ADCOM)* (Guwahati/India, Dec. 2007), pp. 117–121. ISBN 0-7695-3059-1.

[32] ZHOU, X., DREIBHOLZ, T., AND RATHGEB, E. P. Evaluation of a Simple Load Balancing Improvement for Reliable Server Pooling with Heterogeneous Server Pools. In *Proceedings of the IEEE International Conference on Future Generation Communication and Networking (FGCN)* (Jeju Island/South Korea, Dec. 2007), vol. 1, pp. 173–180. ISBN 0-7695-3048-6.

[33] ZHOU, X., DREIBHOLZ, T., AND RATHGEB, E. P. Improving the Load Balancing Performance of Reliable Server Pooling in Heterogeneous Capacity Environments. In *Proceedings of the 3rd Asian Internet Engineering Conference (AINTEC)* (Nov. 2007), vol. 4866 of *Lecture Notes in Computer Science*, Springer, pp. 125–140. ISBN 978-3-540-76808-1.

[34] ZHOU, X., DREIBHOLZ, T., AND RATHGEB, E. P. A New Server Selection Strategy for Reliable Server Pooling in Widely Distributed Environments. In *Proceedings of the 2nd IEEE International Conference on Digital Society (ICDS)* (Sainte Luce/Martinique, Feb. 2008), pp. 171–177. ISBN 978-0-7695-3087-1.