NORDUGRID-XXXXXXX-NN
25/7/2008

ARC Web Services Quick Usage Guide

Contents

1 ARC Middleware

The Advanced Resource Connector (ARC) middleware [1], introduced by NorduGrid [2], is an open source software solution enabling production quality computational and data grids.

Since the first release (May 2002) the middleware has been deployed and been used in production environments. Emphasis is put on scalability, stability, reliability and performance of the middleware. A growing number of grid projects, like NDGF, Swegrid or Swiss National Grid has chosen ARC as their middleware.

This document provides a quick usage guide to the new Web Service (WS) components of ARC, often referred to as ARC1 components. WS design of ARC is based on a service container – the Hosting Environment Daemon (HED), and different grid capabilities are implemented as Web Services residing in HED.

Currently included components are:

- an OGSA BES [3] compliant execution service – *ARC Resource-coupled EXecution service (A-REX)*
- *an echo service (for testing purposes)*

2 Dependencies

The core part of middleware is written in C/C++. Building the software from source or installing a precompiled binary requires different external packages, furthermore the client and server packages have different dependencies too. Below a list of the explicit requirements is shown:

- Mandatory (on client as well as server side):
  - gnu make, autotools (autoconf $\geq$ 2.56, automake $\geq$ 1.8) (build)
  - C++ compiler and library (build)
  - libtool (build)
  - pkg-config (build)
  - gthread-2.0 $\geq$2 .4.7 (build, run)
  - glibmm-2.4 $\geq$ 2.4.7 (build, run)
  - libxml-2.0 $\geq$ 2.4.0 (build, run)
  - openssl $\geq$ 0.9.7a (build, run)
  - e2fsprogs (build, run)
  - doxygen (build)
- Optional (mainly applicable on server side):
  - Swig $\geq$ 1.3.28 (build)
  - Java SDK $\geq$ 1.4 for Java bindings (build, run)
  - Python for Python bindings (build, run)
  - Grid Packaging Tools (GPT) [4] (build)
  - Globus Toolkit® 4 [5] which contains (build, run):
    - Globus RLS client
    - Globus FTP client
    - Globus RSL
  - LHC File Catalog (LFC) [6] (build, run)
  - CppUnit for unit testing (build)
  - Berkeley DB C++ interface (build, run)

Please note that depending on operating system distribution in order to build ARC1 you may need to install development versions of mentioned packages.

3 Getting the software

The software is free to deploy anywhere by anybody. Pre-built binaries for a dozen of Linux platforms can be downloaded from the NorduGrid software repository at `http://download.nordugrid.org/` ("arc1" repository).

The software is released under the GNU General Public License (GPL) (see the COPYING file distributed with the software).

You can get the latest source code for ARC1 from the Subversion repository. See `http://svn.nordugrid.org` for more details.

The NorduGrid software repository hosts the source code, and provides most of the required external software which are not part of a standard Linux distribution.

There are also nightly code snapshots available at:

```
http://download.nordugrid.org/software/nordugrid-arc1/nightly/
```

Choose the latest date available and download snapshot tarball – for example `nordugrid-arc1-200802201038-snapshot.t`

4 Building and Installation

Building from source is currently the recommended way to install ARC Web Services. If you downloaded the tarball, unpack it and traverse into the created directory:

```
tar -zxvf nordugrid-arc1-200802201038-snapshot.tar.gz
cd nordugrid-arc1-200802201038
```

If you obtained the code from the Subversion repository, use the "trunk" directory:

```
cd trunk
```

Now configure the obtained code with:

```
./autogen.sh
./configure --prefix=PLACE_TO_INSTALL_ARC
```

Choose installation prefix wisely and according to the requirements of your OS and personal preferences. ARC1 services should function properly from any location. By default installation goes into `/usr/local` if you omit the "`--prefix`" option. For some modules of ARC1 to work properly you may need to set up the environment variable after installation:

```
export ARC_LOCATION=PLACE_TO_INSTALL_ARC
```

On some systems "`autogen.sh`" may produce few warnings. Ignore them as long as "`configure`" passes without errors. But in case of problems during configure or compilation, collect them and present while reporting problems.

If the previous commands finish without errors, compile and install ARC1 services:

```
make
make install
```

Depending on chosen installation location you may need to run the last command from `root` account. That should install the following components:

- `sbin/arched` - server executable
- `bin/` - user tools and command line clients
- `lib/` - common libraries used by clients, server and plugins
- `lib/arc/` - plugins implementing Message Chain, Service and Security components
- `include/arc/` - C++ headers for application development
- `libexec/` - additional modules used by ARC1 services - currently only A-REX
- `share/doc/arc` - configuration examples/templates and documentation
- `share/locale` - internationalization files - curently very limited support
- `share/man` - manual pages for various utilities

5 Security and Authorization

ARC1 sevices implement security related features through set of Security Handler and Policy Decision Point components. Security Handler components are attached to message processing components. Each Security Handler takes care of processing own part of security information. Currently ARC comes with the following Security Handlers:

- `identity.map` – associates client's identity with local (UNIX) identity. It uses PDP components to choose local isentity and/or identity mapping algorithm.
- `arc.authz` – calls PDP components and combines obtained authorization decisions.
- `delegation.collector` – parses proxy policy from remote proxy certificate. This Security Handler should be configured under TLS MCC component.
- `usernametoken.handler` – implement the functioanlity of WS-Security Usernametoken profile. It will generate Usernametoken into SOAP header, or extract Usernametoken out of SOAP header and do authentication based on the extracted Usernametoken.

Among available PDP components there are:

- `allow` – always returns positive result
- `deny` – always returns negative result
- `simplelist.pdp` – compares DN of user to those stored in a file.
- `arc.pdp` – compares request information parsed from message and policy information specified in this PDP.
- `pdpservice.invoker` – composes the request, puts request into SOAP message, and invokes the remote PDP service to get the response SOAP which includes authorization decision. The PDP service has similar functionality with `arc.pdp`.
- `delegation.pdp` – compares request information parsed from message and policy information specified in proxy certificate from remote side.

There are examples of A-REX service and Echo service with Security Handlers being used. They may be found in `$ARC_LOCATION/share/doc/arc/arex_secure.xml` and `$ARC_LOCATION/share/doc/arc/echo.xml`.

There is also a PDP service which implements the same functionality as `arc.pdp`. See `src/service/pdp/README`.

Specifically for `arc.pdp` and `pdpservice`, a formatted policy with specific schema should be managed, see `$ARC_LOCATION/share/doc/arc/pdp_policy.xml.example` and `$ARC_LOCATION/share/doc/arc/Policy.xsd` for details.

For usernametoken handler, there is example about configuration on service side in `$ARC_LOCATION/share/doc/arc/echo.` you can run Echo service by using this configuration file with usernametoken sechandler configuered. For client side, the echo client (`src/client/echo`)can use usernametoken sechandler to authenticate against echo service (see README under `src/client/echo`); there is also a test program in `src/tests/echo/test_clientinterface` which can be compiled and tested against Echo service with usernametoken sechandler configured.

5.1 X509 Certificates

Most of planned and existing ARC1 services use HTTPS as transport protocol so they require proper setup of X509 security infrastructure [7]. Minimal requirements are:

- Host certificate aka public key in PEM format
- Corresponding private key
- Certificate of the Certification Authority (CA) which was used to sign the host certificate
- Certificates of CAs of clients which are going to send requests to services (unless of course clients use the same CA as the server).

More information about X509 certificates and their usage in Grid environement can be found in:

`http://www.nordugrid.org/documents/certificate_howto.html`
`http://www.nordugrid.org/documents/ng-server-install.html#security`.

For testing purposes you can use the pre-generated certificates and keys available in the code repository:

`http://svn.nordugrid.org/trac/nordugrid/browser/arc1/trunk/doc/sec/TestCA`

Alternatively you may choose to use KnowARC Instant CA service available at:

`https://vls.grid.upjs.sk/CA/instantCA`

The latter is especially usefull if you want to test installation consisting of multiple hosts.

Please remember that it is not safe to use these keys in publicly accessible insallations of ARC. Make sure that even the CA certificate is removed before making your services available to the outside world.

You can put host certificates and private keys anywhere, their location is configurable. Common locations for servers running from root account are:

`/etc/grid-security/hostcert.pem`
and
`/etc/grid-security/hostkey.pem`

respectively.

> Since services have no way to ask for passwords, the content of private key must not be encrypted, neither it should be protected by password. So make sure it is properly protected by means of the file system.

It is possible to configure ARC1 server to accept either a single CA certificate or multiple CA certificates located in the specified directory. The latter option is recommended. The common location is `/etc/grid-security/certificates/` . In that case names of certificate files have to follow hash values of the certificates. These are obtainable by running the command:

```
openssl x509 -hash -noout -in path_to_certificate
```

The corresponding file name for the certificate should be `<hash_value>.0`. The hash value for the pre-generated CA certificate is `4457e417`.

Please make sure the chosen location of certificates is correctly configured in the service configuration file. The configuration for the certificate for TLS MCC should look like this:

```
<KeyPath>/etc/grid-security/hostkey.pem</KeyPath>
<CertificatePath>/etc/grid-security/hostcert.pem</CertificatePath>
<CACertificatesDir>/etc/grid-security/certificates</CACertificatesDir>
```

or

```
<CACertificatePath>/etc/grid-security/ca.pem</CACertificatePath>
```

The same requirements are valid for the client tools for ARC1. You may use the pre-generated user certificate and key located at the same place. Locations of the credentials are configurable, or can be provided to the client tools from the command line.

The set of pre-generated keys and certificates also includes a user certificate in PKCS12 format which you can import into your browser for accessing ARC1 services capable of producing HTML output.

ARC comes with utility `approxy` which generates proxy credentials from certificate/private key pair. It provides only basic functionality and is meant for testing purposes only.

IMPORTANT: If during configuration stage you see a message

"OpenSSL contains no support for proxy credentials"

then you won't be able to use proxy credentials generated by utilities like `grid-proxy-init`, `voms-proxy-init` or `approxy`. Because of that all user private keys will have to be kept unencrypted.

To avoid providing credential information on command line it is possible to have user configuration file with predefined values. Default location for this file is `~/.arc/client.xml`. There is an example installed in `$ARC_LOCATION/share/doc/arc/client.xml`. Please edit before copying it into your home directory.

5.2 Proxy Certificate Generation and Usage

As metioned above, ARC comes with proxy generation utility – `approxy`, installed in `$ARC_LOCATION/bin`. The usage of `approxy` is like:

```
$ARC_LOCATION/bin/approxy -P proxy.pem -C cert.pem -K key.pem
  -c validityStart=2008-05-29T10:20:30Z
  -c validityEnd=2008-06-29T10:20:30Z
  -c proxyPolicyFile=delegation_policy.xml
```

By using argument "-c", some constraints can be specified for proxy certificate. Currently, the life-time can be specified by using "`-c validityStart=...`" and "`-c validityEnd=...`", or "`-c validityStart=...`" and "`-c validityPeriod=...`"; the proxy policy can be specified by using "`-c proxyPolicyFile=...`" or "`-c proxyPolicy=...`".

If proxy certificate is used, in the configuration file for service side or client side, the configuration for the certificate for TLS MCC should look like this:

```
<KeyPath>./proxy.pem</KeyPath>
<CertificatePath>./proxy.pem</CertificatePath>
<CACertificatePath>./ca.pem</CACertificatePath>
```

Because normally a proxy certificate file includes the proxy certificate and private key corresponding to the proxy certificate, `<KeyPath/>` and `<CertificatePath/>` are configured to be the same.

Proxy policy can be spefified as constraint. Proxy policy is for constraining identity delegation. Currently, the supported policy is ARC specific policy. Proxy policy is inserted into proxy certificate's "proxy cert info" extenstion in RFC3820's policy language "`NID_id_ppl_anyLanguage`".

6 ARC Server Setup and Configuration

The configuration of the ARC server is specified in an XML file, the location of which is specified as a command line argument with the -c option of "`arched`" daemon. Examples of configuration files with comments describing various elements are available in directory `/usr/share/doc/nordugrid-arc<version>` corresponding to the ARC1 installation.

7 The Echo Service

The *Echo* service is offered purely for testing purposes. It is "atomic" and has no additional dependencies other than what is provided by the Hosting Environment Daemon (HED). An example of an Echo service configuration can be found in `/usr/share/doc/nordugrid-arc<version>/echo.xml`.

7.1 The Echo Client

The configuration of the ARC Echo client is specified in an XML file. The location of the configuration file is specified by the environment variable `$ARC_ECHO_CONFIG`. If there is no such environment variable, the configuration file is assumed to be `echo_client.xml` in the current working directory.

To use the Echo client, type

```
$ARC_LOCATION/bin/apecho <message>
```

where `<message>` is the message which the Echo service should return.

8 The A-REX Service

ARC1 comes with OGSA BES compliant Grid job management service called A-REX. To deploy A-REX use example configuration files available in `/usr/share/doc/nordugrid-arc<version>`:

- `arex.xml` – configuration for arched server. Read comments inside this file.
- `arc-arex.conf` – legacy configuration for Grid Manager part of A-REX. This file defines how jobs are managed by A-REX locally. Read and edit it. and edit it to fit your installation. This file defines WS interface of A-REX.

For more detailed information please read Grid Manager documentation [8]. Grid Manager runs as part of A-REX service. There is no need to run any additional executable. But you still need to setup its infrastructure as long as you are going to have anything more sophisticated than described in the example configuration. For more information read the previously mentioned document.

Currently, a proper functioning A-REX requires environment variable `$ARC_LOCATION` to be set to the installation prefix of ARC.

A-REX uses HTTPS as transport protocol (although one can reconfigure it to use plain HTTP) so it requires proper setup of X509 security infrastructure. See Section 5.1 for instructions.

Copy example configuration files to some location and edit them. Make sure all paths to X509 certificates and Grid Manager configuration are set correctly. Start server with command:

```
$ARC_LOCATION/sbin/arched -c path_to_edited_arex.xml
```

Look into log file specified in `arex.xml` for possible errors. You can safely ignore messages like *"Not a '...' type plugin"* and *"Unknown element ... - ignoring"*.

If you compiled ARC1 services with Globus support and you see complaints about "libglobus..." and that it cannot open a shared object file, try to add `/opt/globus/lib` to your `$LD_LIBRARY_PATH`:

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/globus/lib
```

8.1 Testing and Using A-REX (clients)

Now you may use the command line utility "`apinfo`" to obtain a service description. Make sure user key and certificate and CA certificate(s) are readable from account you use, and run something like:

```
./apinfo -d VERBOSE -C $HOME/usercert.pem -K $HOME/userkey.pem
 -A /etc/grid-security/certificates https://localhost:60000/arex 2>error.log
```

This should produce XML output describing resource A-REX represents. Listing 1 shows an example of proper output. In case of error see `error.log` file and report problems to ARC development team. Include content of `error.log` and server log.

Listing 1: Example of a good apinfo output

```
<n:nordugrid xmlns:n="urn:nordugrid" xmlns:M0="urn:Mds"
xmlns:nc0="urn:nordugrid-cluster">
  <nc0:name>mobile2.home.net</nc0:name>
  <nc0:aliasname>MINIMAL Computing Element</nc0:aliasname>
  <nc0:comment>This is a minimal out-of-box CE setup</nc0:comment>
  <nc0:issuerca>/O=Grid/O=Test/CN=CA</nc0:issuerca>
  <nc0:issuerca-hash>4457e417</nc0:issuerca-hash>
  <nc0:trustedca>/O=Grid/O=NorduGrid/CN=NorduGrid Certification
Authority</nc0:trustedca>
  <nc0:trustedca>/O=Grid/O=Test/CN=CA</nc0:trustedca>
  <nc0:contactstring>gsiftp://mobile2.home.net:2811/jobs</nc0:contactstring>
  <nc0:lrms-type>fork</nc0:lrms-type>
  <nc0:architecture>i686</nc0:architecture>
  <nc0:homogeneity>TRUE</nc0:homogeneity>
  <nc0:nodeaccess>inbound</nc0:nodeaccess>
  <nc0:nodeaccess>outbound</nc0:nodeaccess>
  <nc0:totalcpus>1</nc0:totalcpus>
  <nc0:usedcpus>0</nc0:usedcpus>
  <nc0:cpudistribution>1cpu:1</nc0:cpudistribution>
  <nc0:prelrmsqueued>0</nc0:prelrmsqueued>
  <nc0:totaljobs>0</nc0:totaljobs>
  <nc0:sessiondir-free>8469</nc0:sessiondir-free>
  <nc0:sessiondir-total>43865</nc0:sessiondir-total>
  <nc0:sessiondir-lifetime>10080</nc0:sessiondir-lifetime>
  <M0:validfrom>20080220142836Z</M0:validfrom>
  <M0:validto>20080220143036Z</M0:validto>
  <nq0:name xmlns:nq0="urn:nordugrid-queue" name="fork">
    <nq0:name>fork</nq0:name>
```

```
    <nq0:status>inactive, grid-manager is down</nq0:status>
    <nq0:comment>This queue is nothing more than a fork host</nq0:comment>
    <nq0:schedulingpolicy>FIFO</nq0:schedulingpolicy>
    <nq0:homogeneity>TRUE</nq0:homogeneity>
    <nq0:nodecpu>Intel(R) Celeron(R) M CPU        420  @ 1.60GHz @ 1596.048
MHz</nq0:nodecpu>
    <nq0:architecture>i686</nq0:architecture>
    <nq0:maxcputime>unlimited</nq0:maxcputime>
    <nq0:maxwalltime>unlimited</nq0:maxwalltime>
    <nq0:running>0</nq0:running>
    <nq0:gridrunning>0</nq0:gridrunning>
    <nq0:localqueued>0</nq0:localqueued>
    <nq0:gridqueued>0</nq0:gridqueued>
    <nq0:prelrmsqueued>0</nq0:prelrmsqueued>
    <nq0:totalcpus>1</nq0:totalcpus>
    <IM0:validfrom>20080220142836Z</M0:validfrom>
    <M0:validto>20080220143036Z</M0:validto>
    <nig0:name xmlns:nig0="urn:nordugrid-info-group" name="jobs">
      <nig0:name>jobs</nig0:name>
      <M0:validfrom>20080220142836Z</M0:validfrom>
      <M0:validto>20080220143036Z</M0:validto>
      <nj0:globalid xmlns:nj0="urn:nordugrid-job"
name="gsiftp://mobile2.home.net:2811/jobs/636312034687631804
289383">

<nj0:globalid>gsiftp://mobile2.home.net:2811/jobs/6363120346876318042893 83</nj0:
globalid>
        <nj0:globalowner>/O=Grid/O=Test/CN=user</nj0:globalowner>
        <nj0:jobname>TEST</nj0:jobname>
        <nj0:execcluster>mobile2.home.net</nj0:execcluster>
        <nj0:execqueue>fork</nj0:execqueue>
        <nj0:submissionui>127.0.0.1:59790</nj0:submissionui>
        <nj0:submissiontime>20080220005243Z</nj0:submissiontime>
        <nj0:sessiondirerasetime>20080227005843Z</nj0:sessiondirerasetime>
        <nj0:completiontime>20080220005843Z</nj0:completiontime>
        <nj0:cpucount>1</nj0:cpucount>
        <nj0:comment>GM: The grid-manager is down</nj0:comment>
        <nj0:usedcputime>0</nj0:usedcputime>
        <nj0:usedwalltime>0</nj0:usedwalltime>
        <nj0:errors>Job submission to LRMS failed</nj0:errors>
        <nj0:status>FAILED</nj0:status>
        <nj0:rerunable>SUBMIT</nj0:rerunable>
        <M0:validfrom>20080220142836Z</M0:validfrom>
        <M0:validto>20080220143036Z</M0:validto>
      </nj0:globalid>
    </nig0:name>
    <nig0:name xmlns:nig0="urn:nordugrid-info-group" name="users">
      <nig0:name>users</nig0:name>
      <M0:validfrom>20080220142836Z</M0:validfrom>
      <M0:validto>20080220143036Z</M0:validto>
    </nig0:name>
  </nq0:name>
</n:nordugrid>
```

Please note that you can run similar `apinfo` request against any ARC1 service except Echo service. Output should always be XML description of service.

A-REX accepts jobs described in JSDL language. Example JSDL jobs are provided in `$ARC_LOCATION/share/doc/` in files `jsdl_simple.xml` and `jsdl_stage.xml`. To submit job to A-REX service one may use `apsub` com-

mand:

```
$ARC_LOCATION/bin/apsub https://localhost:60000/arex
 /usr/local/share/doc/arc/jsdl_simple.xml id.xml
```

If everything goes properly somewhere in it's output there should be a message "Submitted the job!". Obtained job identifier is an XML document and is stored in id.xml file. It may then be used to query job state with `apstat` utility:

```
$ARC_LOCATION/bin/apstat id.xml 2>/dev/null
Job status: Running/Submiting

$ARC_LOCATION/bin/apstat id.xml 2>/dev/null
Job status: Running/Finishing

$ARC_LOCATION/bin/apstat id.xml 2>/dev/null
Job status: Finished/Finished
```

Full set of A-REX client tools consists of `apsub`, `apstat`, `apkill`, `asstat`, `apget` and `apclean` commands. For more information please see man pages ot those utilities.

9 Finding more information

Many information about functionality and configuration of various components may be found inside corresponding configuratrion XML schemas.

There are several API documents available as well.

10 Contributing

The open source development of the ARC middleware is coordinated by the NorduGrid Collaboration. Currently, the main contributor is the KnowARC project (`www.knowarc.eu`), but the Collaboration is open to new members. Contributions from the community to the software and the documentation is welcomed. Sources can be downloaded from the software repository at `download.nordugrid.org` or the Subversion code repository at `svn.nordugrid.org`.

The technical coordination group defines outstanding issues that have to be addressed in the framework of the ARC development. Feature requests and enhancement proposals are recorded in the Bugzilla bug tracking system at `bugzilla.nordugrid.org`. For a more detailed description, write access to the code repository and further questions, write to the `nordugrid-discuss` mailing list (see `www.nordugrid.org` for details). Ongoing and completed Grid Research projects and student assignments related to the middleware are listed on the NorduGrid Web site as well.

11 Support, documentation, mailing lists, contact

User support and site installation assistance is provided via the request tracking system available at `nordugrid-support@no`
In addition, NorduGrid runs a couple of mailing lists, among which the nordugrid-discuss mailing list is a general forum for all kind of issues related to the ARC middleware.

Feature and enhancement requests, as well as discovered problems, should be reported in the Bugzilla problem tracking system `bugzilla.nordugrid.org`.

Research papers, overview talks, reference manuals, user guides, installation instructions, conference presentations, FAQ and even tutorial materials can be fetched from the documentation section of `www.nordugrid.org`.

Contact information is kept updated on the `www.nordugrid.org` Web site.

References

[1] M. Ellert *et al.*, "Advanced Resource Connector middleware for lightweight computational Grids," *Future Gener. Comput. Syst.*, vol. 23, no. 1, pp. 219–240, 2007.

[2] The NorduGrid Collaboration. [Online]. Available: http://www.nordugrid.org

[3] I. Foster *et al.* (2007, August) OGSA™ Basic Execution Service Version 1.0. GFD-R-P.108. [Online]. Available: http://www.ogf.org/documents/GFD.108.pdf

[4] "Grid Packaging Tools." [Online]. Available: http://www.gridpackagingtools.org

[5] I. Foster and C. Kesselman, "Globus: A Metacomputing Infrastructure Toolkit," *International Journal of Supercomputer Applications*, vol. 11, no. 2, pp. 115–128, 1997.

[6] LHC File Catalog. [Online]. Available: https://savannah.cern.ch/projects/jra1mdw/

[7] Public-Key Infrastructure (X.509) (PKI), Proxy Certificate Profile. [Online]. Available: http://rfc.net/rfc3820.html

[8] A. Konstantinov. The ARC Computational Job Management Module - A-REX. NORDUGRID-TECH-14.