OLC and Other OLxx Services*

Albert Dvornik

last RCS change: January 14, 1999

Contents

0 About This Document **3**

1 Functional Overview **3**
1.1 olcd . . . 3
1.2 rpd . . . 3
1.3 polld . . . 3
1.4 lumberjack . . . 4
1.5 topic archives . . . 4
1.6 mail feeds . . . 4
1.7 compiling statistics . . . 4
1.8 Rebecca . . . 5
1.9 other gunk . . . 5

2 Server Characteristics **5**
2.1 Compatibility . . . 5
2.2 Security . . . 5
2.2.1 Potential Targets . . . 5
2.2.2 Vulnerabilities . . . 5
2.3 Resource Loading . . . 6
2.4 Policy . . . 6

3 Requirements and Dependencies **6**
3.1 Required . . . 6
3.2 Required for Interoperability with Earlier Versions . . . 7
3.3 Not Required . . . 7

*The source for this document is in `/mit/olcdev/doc/ops/operation.tex`

4 Dependents 7

5 Impacts of Service Failures 8

6 Building 8

7 Installation 9

8 Maintenance 10
8.1 Relocation 10
8.2 Reconfiguration 11
8.3 Restarts 12
8.4 Updates 12
8.5 Shutdowns 12
8.6 Removal from Service 12
8.7 Backups and Restores 12

9 Troubleshooting 13
9.1 Diagnostics 13
9.2 Known/Predicted Failure Modes 13
9.3 Service 13

10 State Information 13

0 About This Document

As of the most recent revision, this document now covers the setup and operation of all the OL*xx* (OLC, OLTA, OWL) services. However, porting and development work tends to use OLC as a model. Therefore, this document will tend to become somewhat OLC-centric, and may be less than fully accurate when it comes to other OL*xx* services (OLTA and OWL). Please exercise extra caution when trying to upgrade or tweak the OLTA/OWL server, and update this document with any insights you gain into their operation. (In general, whenever you think of a way to improve this document, do it.)

1 Functional Overview

The software that may run on the OLC server can be divided in several major components:

1.1 olcd

The OLC daemon, `olcd`, is the central part of the OLC system. It talks to the user and consulting client binaries and performs almost all operations associated with the operation of the OLC system.

Currently, a separate `olcd` binary most be compiled for each of the OL*xx* services, but the changes are minimal and restricted to a small number of header files.

The OLTA and OWL services are currently run on the same machine. To avoid namespace collision, the olcd binaries for these services are named `oltad` and `owld` respectively.

1.2 rpd

The replay daemon, `rpd`, is used by the `olist/oreplay/oshow` client and by `xolc` to read some of the log information. This functionality is already offered by `olcd`, but since `rpd` doesn't lock the queue, it offers better performance. The `rpd` functions are currently primarily used by the Emacs OLCR interface.

The idea for the implementation of `rpd` originated in a time when hardware was rather expensive and running a complicated networking server taxed the available resources. It is unlikely that this will ever be a problem again. As a result, designs for the future of OLC should consider the option of entirely scrapping the support for `rpd`; of course, such a change must take into account all of the OLC system, including the more "exotic" bits.[1]

For the OLTA/OWL server, the two rpd binaries are named `rpd.olta` and `rpd.owl` to distinguish between them.

[1] Such as, for example, Emacs OLCR, `xolc` and MacOLX.

1.3 polld

`polld` is a program that runs periodically to determine which of the users with questions in the queue are currently logged in, and where. This helps consultants to determine the priority of various questions.

Despite its name, `polld` does not normally run as a daemon, but is periodically invoked from `cron`. The `-cycle` command-line option turns on stand-alone operation, which may be buggy (I haven't tested it).

For the OLTA/OWL server, the two polld binaries are named `polld.olta` and `polld.owl` to distinguish between them.

1.4 lumberjack

The logs of resolved questions are archived by `lumberjack`, which is invoked by `olcd` whenever needed. If `lumberjack` can't be run, the logs will collect dust in `/var/athena/olxx/donelogs` until the next sucessful invocation of `lumberjack`.

For the OLTA/OWL services running on the same machine, two lumberjack binaries are needed. They are named `lumberjack.olta` and `lumberjack.owl`.

1.5 topic archives

`lumberjack` stores the logs of answered questions in Discuss meetings. Each question topic has a separate meeting. For OLC and OLTA, the short name[2] consists of the letter `o` followed by the topic name: for example, `olatex` or `oother`. For the OWL service, the short name of each meeting consists of the string `or` followed by the topic name. In addition, the meeting named `ooga` should exist to store questions resolved as "unanswered."

Note that other Discuss meetings may exist on the OL*xx* server, but they shouldn't be directly associated with the operation of the OL*xx* software.

1.6 mail feeds

OLC currently supports asking questions by e-mail. (To the best of my knowledge, OLTA and OWL do not.) Athena mailhubs alias `olc-`*topic* to `olc-`*topic*`@matisse.local`, and Matisse's sendmail daemon is configured to turn each such piece of mail into a question with appropriate *topic*. The actual work is performed by a program named `olcm`, which is invoked from the `aliases` file similarly to `dsmail`:

```
olc-other@matisse.LOCAL: "|/usr/athena/etc/olcm -s matisse -t other
-k olc.matisse"@matisse.LOCAL
```

Operation of the mail feed system will not be further covered in this document.

[2] If you don't know what I'm talking about, read some Discuss documentation, which will hopefully explain it.

1.7 compiling statistics

OLC server currently calculates weekly statistics for answering questions. Neither OLTA nor OWL calculate weekly statistics, although applying the OLC scripts with appropriate modifications would not be difficult. The statistics generation uses a set of Perl scripts that are not strictly a part of the OLC system. The scripts are periodically run from `cron`.

Statistics generation will not be further covered in this document.

1.8 Rebecca

The OLC server currently runs a daemon named Rebecca,[3] which listens to several common Zephyr classes and instances. Whenever people complain about `olcd` or `rpd`, Rebecca checks if the daemon is running and restarts it if needed. OLTA and OWL do not run this daemon.

Rebecca will not be further covered in this document.

1.9 other gunk

The OLC server will, without a doubt, also be running a lot of what can best be described as "other gunk". Most of it will be wholly useless, such as an unrelated service installed five or ten years ago by some forgotten maintainer for a completely incomprehensible reason. Migration is a fine time to nuke this from the proverbial orbit.

2 Server Characteristics

2.1 Compatibility

There is currently no conceptual reason why two fully functional OL*xx* services couldn't be run on the same host. In fact, OLTA and OWL do just that. Until recently, the port used by `rpd` was hard coded into the binary, so an rpd was only run for OLTA. Now, however, `rpd` is run for both OLTA and OWL, with the port to be used determined by Hesiod service entries for `olta-query` and `owl-query`.

The OL*xx* services can be expected to peacefully coexist with any well-behaved non-OL*xx* service, but are traditionally kept on a separate server for other reasons.

2.2 Security

2.2.1 Potential Targets

The OL*xx* servers store active user questions and Discuss logs of old questions. Both should be considered private, although not critically confidential. Users are normally prevented

[3] "Rebecca" is short for "Rebecca P. Davis", which is long for RPD.

from making unauthorized requests (both to access this data and to change the server state) through use of access control and Kerberos authentication.

There is no other known sensitive data related to the OL*xx* servers.

2.2.2 Vulnerabilities

The network connections between the OLC clients and the server are currently not encrypted, and are consequently vulnerable to passive network attacks.

The server still uses some static-sized buffers, and a user with access to Kerberos tickets in the Athena realm might conceivably be able to use a custom-designed client to overflow an unprotected buffer. A denial of service could certainly result from this; at worst, the attacker may be able to gain super-user access to the server. This problem could be remedied by rewriting all of the source to use dynamic allocation, but this would be a rather large project for relatively little immediate gain.

2.3 Resource Loading

While CPU loading may have been a concern once upon a time, it is not expected to be an issue on a modern architecture (such as a Sparc 5). Network load from the OL*xx* services is negligible.

2.4 Policy

Traditionally, root login privileges on the OLC server have been granted to the full time consulting staff and some student consultants. This constiutes a further reason against using the machine(s) in question to run other operational services.

3 Requirements and Dependencies

3.1 Required

- *Kerberos* — The client and the server require Kerberos.
- *srvtab* — Requires an `olc` srvtab entry
- *root access* — Currently needs to run as root. [change?]
- *Athena* — Current installation requires either Athena 8.1 or 8.2.
- *Platforms* — Runs under Sun/Solaris 2.5.1 or Solaris 2.6. A small amount of porting may be required to make it work on other POSIX and XPG3 compliant platforms.

- *Hesiod* — To be accessed via the UNIX `olc/olcr` client, `ol`*xx* `sloc` and `ol`*xx* `service` should exist in Hesiod. (Alternatively, the fallback `sloc` information can be retrieved from the client configuration file, and the `service` information from the Athena-wide `/etc/services` file.)

  To be accessed via the UNIX `oreplay/olist/oshow` client, `ol`*xx* `sloc` and `ol`*xx*`-query service` should exist in Hesiod. (The fallbacks are as above.)

  In addition to those already mentioned, access via the Macintosh client is believed to depend on `ol`*xx*`-locking service` entry.

- *name service* — Clients use name service in connecting to the server.

- *syslog* — Uses syslogging. (A running `syslogd` is not strictly required for operation, but in the Athena environment the data provided is extremely useful for diagnosing and correcting problems.)

- *zhm* — OLC uses zephyr to notify users and consultants of various events.

3.2 Required for Interoperability with Earlier Versions

- *Hesiod* — Old UNIX clients (Release 8.0 and earlier) used the `ols service` Hesiod entry (instead of `olc-query`). When those clients are fully desupported, and after a final `grep` through the source tree, this entry can be removed.

- *moon* — The machine must take an accidental update when the Lunar cycles say so.

3.3 Not Required

- *disk space* — The queue takes up relatively little space (¡100MB). On the other hand, the Discuss archives have the ability to grow indefinitely. The disks should not be allowed to fill during operation.

- *swap space/real memory* — Has no notable memory requirements, except that the swap space should not be allowed to fill during operation.

- *peripherals* — Requires no special peripherals; an external disk may be used to preform regular local backups.

- */srvd* — RVD need not be attached.

- *ethernet* — Does not depend on ethernet address.

4 Dependents

The direct dependents of the OLC system are the On-Line Consultants. However, the entire Athena user community uses the OLC service to ask (and answer) questions. Finally, since problems are more likely to get noticed when reported to the consultants, and since noticed problems are easier to correct in a timely fashion, successful operation of OLC helps Athena Operations in their eternal vigilance.

The direct dependents of the OLTA system are the students of the classes using the OLTA system and the TA's and professors answering those questions. At the time of this writing, 10.001, 6.170, and 10.301 are the classes most often using OLTA.

The direct dependents of the OWL system are the Librarians of the MIT Libraries, and the students and staff members using the OWL service to seek help with their research.

5 Impacts of Service Failures

1. If `olcd` is not operating properly, users will be unable to ask questions and consultants will be unable to answer them.

   If the daemon is not running, it can get restarted by Rebecca (sec. 1.8).

2. If `rpd` is not operating properly, consultants will be unable to use the Emacs OLCR client. However, they can still use the prompt-based client, and users will still be able to ask questions. Most consultants normally use EOLCR, so this will decrease their efficiency at answering questions.

   If the daemon is not running, it can get restarted by Rebecca (sec. 1.8).

3. If `polld` is not operating properly, the data on who is currently logged in will be out of date or incorrect.

4. If `lumberjack` is not operating properly, the resolved logs will not be placed in archives, but will accumulate in the `donelogs` subdirectory. Note that this is sometimes desirable behavior, such as during migrations; you may wish to make the lumberjack binary to become unavailable in those situations by renaming it.

5. If the Discuss daemons aren't operating properly, consultants will be unable to read logs of past questions and extract the wisdom of the ages from them.

6. If the local sendmail daemon isn't running correctly, users will be unable to ask questions by e-mail, and consultants will be unable to correct previous answers by mail. If the mailhubs aren't operating properly, in addition to the above, consultants will be unable to use e-mail to communicate to logged-out users.

7. If Rebecca isn't operating properly, consultants without root login privileges on the server won't be able to restart `olcd` and `rpd`. Furthermore, we'll be deprived of her wit and humor.

8. If `zhm` is not running, `olcd` will fall back to a networked version of `write`.

There are doubtlessly more failure modes that have not been mentioned here, but one should hope their consequences are less significant than the ones noted above.

6 Building

First and foremost: if you ever build OLC on Athena, you have a moral obligation to ensure that any changes you make get incorporated back into a central source tree, which is[4] currently a part of the Athena release.

[NOTE: these instructions aren't intended to work with the source tree from the release yet. That should be fixed once the release has a useful source tree.]

The simplest way to build the OLC source tree is to find the corresponding copy of the `ramirez` script and invoke it as `./ramirez -X -Wall -DLOG`. (`-DLOG` adds debugging syslogs for each type of request.) For building OLTA/OWL binaries, add `-DOLTA` or `-DOWL` to the flags given, as appropriate. Alternatively, try the following:

- Create a symlink farm into the source tree. Change your working directory to the top level of the link farm. (You could just build in the source tree, but someone will hate you for it, I bet.)
- Run `imake -Iconfig -DUseWclFromLocker`.
- Change to `./server` and run `make all`.
- Copy the server binaries somewhere useful.

7 Installation

The following steps should be followed to install a new server:

Note that all instances of "olxx" below should be replaced with the name of the appropriate service (olc, olta, or owl).

- If creating an OLC server, run `mkserv ops olc`.[5] (This will list many of the instructions that follow.)

[4] Well, it will soon be, as of release 8.3. Someone get rid of this footnote as soon as it's actually true.

[5] Actually, currently you need to run `mkserv ops olcdev:/mit/olcdev/mkserv olc`, until the OLC mkserv scripts are installed in the mkserv locker.

Note that `mkserv ops` will automatically invoke `remote`, while `olc` will automatically invoke `mail` and `discuss`.

- If creating an OLTA/OWL server, run only `mkserv ops mail discuss`. `mkserv olc` is still OLC specific, so you'll need to use the seperate mkserv scripts and do some things by hand.

  The most important thing will be to build the appropriate binaries. OLTA/OWL binaries can be built using the build system described in this document, using `-DOLXX` as appropriate.

- Install a `olc.hostname` srvtab in /etc/athena/ol*xx*/srvtab.
- Create a list of topics in /etc/athena/ol*xx*/topics.
- Create the `.acl` files under /etc/athena/ol*xx*/specialties/.
- Create and populate the `.acl` files under /etc/athena/ol*xx*/acls/.
- Examine other files in /etc/athena/ol*xx*, and edit if needed.
- Set up Discuss archives.
- Set up olcm mail feeds.
- Make sure all of the relevant `cron` jobs are installed. Currently:

```
# polld locates users and updates their OLC login status (by talking to olcd)
1,11,21,31,41,51 * * * *        /usr/athena/etc/polld
# Incrementals to /backup/inc
#  Note: medusa runs remote UFS backups via the network.  Ideally, these
#    happen ~3 times a week.  The following line makes a daily incremental
#    dump (relative to the last medusa dump) and stores it on an external
#    drive.  This should happen daily.  --bert 12feb1998
55 17 * * *     /var/ops/bin/backup_inc 7
# Copy OLC and Discuss acls from Moira lists.
19 7,22 * * *   /u1/moira-sync/syncer.pl >> /u1/moira-sync/log.stdout
# The following jobs deal with generating OLC statistics reports
0 0 * * *       /var/ops/bin/gen_olc_stats.pl -a /var/athena/olc/stats/ask_stats\
 -r /var/athena/olc/stats/res_stats
0 10 * * 1      /var/ops/bin/olc-report.pl -w; /var/ops/bin/olc-report.pl -w -c
```

  Again, this is OLC specific. For the OLTA/OWL server, only the polld cron job is required. However, both copies of polld (`polld.olta` and `polld.owl`) will need to be run from cron.

- Make sure to install all of the scritps that `cron` will try to run. Many of these are located in /mit/ops/services/olc/scripts.
- For the OLC server, set up Rebecca. She lives in /mit/ops/services/olc/rebecca, and /var/ops/bin on the server. Her init script should be linked to /etc/rc2.d/S95rebecca.

8 Maintenance

8.1 Relocation

The following steps should be followed to migrate a server. It is assumed that the new server will, at the end of migration, have the new server's hostname and IP address.

On the new server:

- Run `mkserv ops olc` or `mkserv ops mail discuss` as appropriate.[6]
- Copy the contents of /etc/athena/ol*xx*. (This needs to be encrypted, since it contains the srvtab!)
- Set up olcm mail feeds.
- Copy the `/.klogin`.
- Set up other related stuff (Rebecca, stats, backups...). (sec. 7)

Shortly in advance of the move, on the old server:

- Change the OLxx motd.
- Disable `lumberjack` by renaming the binary in /usr/athena/etc.
- Migrate the Discuss archives.

This is a good point to consider turning back. =)

To start the move, on the old server:

- Restart `olcd` in the maintenance mode, by killing the old daemon and restarting it with the `-maint` option.
- Change the OLxx motd to indicate that a move is in progress.

[6] Once again, currently `mkserv olcdev:/mit/olcdev/mkserv olc`.

- Copy all queue data (from /var/athena/ol*xx*, or on old servers, from /usr/spool/ol*xx*) to the new server.
- Bring down the old server.
- Restart the new server with the old server's hostname and IP address.
- Make sure everything is running as expected.
- Optionally, change the OLxx motd to indicate that the move succeeded.
- Optionally, bring up the old server on a different IP address.

If the move fails in some non-salvageable way, the old server should be brought back up on its old address and `lumberjack` re-enabled, until the problem can be fixed.

8.2 Reconfiguration

Most configuration changes that are expected under normal operation of OL*xx* can be made using the administrative functions of `olcr`. Please avoid making those changes by hand on the server.

[Make a list of things that can't?]

8.3 Restarts

The simplest way of restarting `olcd` and `rpd` on the OLC server is by using Rebecca. (For more information, try `zwrite olc.matisse -m "Rebecca, help"`.)

Otherwise, `olcd` and `rpd` will shut down cleanly when killed with SIGHUP, SIGINT or SIGTERM, and can be restarted by simply running /usr/athena/etc/olcd and /usr/athena/etc/rpd, respectively.

8.4 Updates

If the OL*xx* server software is updated, any number of things described in this document could change. Therefore, whoever is updating a server should make sure to update this document as needed.

Normally, updates will be tied to Athena version updates. Therefore, unless the data formats change, you can just use normal procedures for Athena version upgrades. However, I would heartily recommend trying the upgrade (with live data) on a different machine first.

8.5 Shutdowns

`olcd` and `rpd` will shut down cleanly when killed with SIGHUP, SIGINT or SIGTERM. However, it is often preferable to leave `rpd` running and restart `olcd` in maintenance mode (with the `-maint` command-line argument), which will enable the users to see the MOTD explaining the problem.

8.6 Removal from Service

If any of the OL*xx* services is to be taken out of service, the Athena user community should be notified and given time to complain far in advance.

If there is to be an Irish wake, make sure to invite the full-time consultants and all past OLC developers (myself included). Please be aware that letting MIT know about any Irish wakes for servers may be a violation of MIT's alcohol policies. Don't tell them when OLC code drives you to drink, either. [It will. =)]

8.7 Backups and Restores

The OLC server currently backs up its disk daily at dump level 7, to supplement the regular level 0 network backups from Medusa. This uses standard Ops scripts for the purpose. The OLTA/OWL server is backed up to Medusa with the same schedule, but does not perform daily backups to an alternate local disk.

9 Troubleshooting

9.1 Diagnostics

The bad news regarding diagnostics is that the best available diagnostic tool is currently the source code. The good news regarding diagnostics is that the best available diagnostic tool is currently the source code.

Many server messages are syslogged using the `local6` facility. Those messages will by default be logged to /var/athena/ol*xx*/admin/syslog, and should prove insight into most service failures (we hope). You may wish to also forward those messages elsewhere to keep an eye on the state of the server.

Coredumps will be found in /var/athena/ol*xx*/core.*date*.

9.2 Known/Predicted Failure Modes

- *olcd died* — The command-line clients will display the message
  `Unable to connect to the OLC daemon.  Please try again later.`
  Other clients will display similar errors.

- *rpd died* — The EOLCR list of questions will fail to be displayed. `olist`, `oshow` and `oreplay` will show messages like
  `Thu 05-Mar-98 8:09pm connect:  Connection refused`
- *other errors* — It is unclear how gracefully the system will deal with most other errors, especially those that may occur in a large number of places (eg, filling disk or swap). Many errors may result in the daemon dying, with the results described above.

9.3 Service

Bug reports should be sent to <`bug-olc@mit.edu`> for posterity. Prompt responses should not be expected; outside of periods of testing, there is no one responsible for fixing critical problems in a timely fashion.

10 State Information

All current server configuration state is stored under /etc/athena/ol*xx*/. All current queue and log data is stored under /var/athena/ol*xx*/. All past OLC/OLTA question data is stored in Discuss meetings whose names start with `o`. All past OWL question data is stored in Discuss meetings whose names start with `or`.