Installing OMERO.server on Microsoft Windows

Błażej Pindelski
Open Microscopy Environment
Agenda

Avoiding common pitfalls
Prerequisites (installation and verification)
Filesystem layout
DB scaffolding
Server configuration (web deployment)
Windows service configuration
Live demo
Avoiding common pitfalls

Windows restrictions
- run Command Prompt as superuser
- verify user account permissions (LDAP/AD limitations)
- Windows not seen often as production server OS

Walkthrough and help available
- server installation documentation
- hardware requirements
- community resources
- feedback welcome
Prerequisites

Installation time ~10 minutes

Can be installed in any order

- exception: Python libraries

Safe to use installation defaults

- simplify target paths (e.g. C:\ice)
Prerequisites - limitations

Python - version numbers, bitness, dependencies

- no PIL for 64-bit Python, PyTables:
  - Numexpr 1.4 works, 2.x doesn’t...

Some prerequisites have their own limitations

- Ice 3.4 - Python 2.6 / Ice 3.3 - Python 2.5

Python 2.6 (32 bit) and Ice 3.4 setup preferred

- using newest PostgreSQL is fine
- other configurations possible
Prerequisites - verification

Set `PATH` and `PYTHONPATH` first

```
C:\Users\ome>python
Python 2.6.6 (r266:84297, Aug 24 2010, 18:46:32) [MSC v.1500 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import tables
>>> tables.test()
```

```
C:\Users\ome>icegridadmin -v
3.4.2
```

```
C:\Users\ome>java -version
java version "1.7.0_21"
Java(TM) SE Runtime Environment (build 1.7.0_21-b11)
Java HotSpot(TM) Client VM (build 23.21-b01, mixed mode, sharing)
```
Filesystem layout

Default use case

- server binaries in `C:\OMERO.server`
- repository in `C:\OMERO`
- binary repository explained

OMERO writes to user's home directory

- i.e. `C:\Users\<name>\omero`
- can be changed by setting `OMERO_TEMPDIR`

Unzip server code, run diagnostics

- `C:\OMERO.server\bin\omero admin diagnostics`
DB scaffolding

Do not use db_user and db_password!

- usernames and passwords explained

Create DB user, set a password

- as DB superuser in pgsql
- configuring PostgreSQL

Create tables using bin\omero db script output

- as DB OMERO user in pgsql
- usually the pgpqlsql language is already created with the new DB
- COMMIT as final output line means success
Server configuration

Use `bin\omero config set`, minimally configure

- `omero.db.name`
- `omero.db.user`
- `omero.db.pass`
- `etc/omero.properties` for defaults set and other options

If needed, set up LDAP authentication

- `setting up LDAP`

`bin\omero admin start` should succeed

- and your mem/CPU usage should go up
Server configuration (web deployment)

Make sure you have IIS installed

- OMERO.web requires ISAPI_WSGI with IIS 6 compatibility and extra settings
- Unzip isapi_wsgi-0.4.2 and run python setup.py install
- Feedback on IIS configuration?

bin\omero config set

- omero.web.session_engine
- omero.web.cache_backend

Finally bin\omero web iis

- Do not use web start (unless with dev web server)
Server configuration

Minimal settings to be up and running

C:\OMERO.server>bin\omero config get
omero.db.name=omero
omero.db.pass=omero
omero.db.user=omero
omero.web.cache_backend=file://C:/windows/temp/
omero.web.session_engine=django.contrib.sessions.backends.cache

http://localhost/omero
Windows service configuration

Windows service installed during start-up

- omero.windows.user and omero.windows.pass for configuring credentials for service user
- can also be configured as startup params (-u and -w)

Some account limitations can be solved this way
Live demo
Thank you
Server heap memory

Server has 512 MiB set by default

- see etc/grid/templates.xml

General rule for memory allocation

- depends on your largest image size
- 2 copies of that image present in RAM
- 2 GB medium, 3 GB in other cases
- JCB DataViewer uses 4/8 GB RAM
Filesystm I/O latency

NFS increases the latency

Lock management

- distributed locking over NFS is subject to many variables

Considerations

- NFS vs. CIFS
- NAS embedded locking management
- "mount lost" recovery scenario
- mount health monitoring
Security

Overview

Out of the box

- encryption of all passwords between client and server via SSL
- full encryption of all data when requested via SSL
- limited visible TCP ports to ease firewalling
- escaping and bind variable use in all SQL interactions performed via Hibernate
Data backup/restore

DB and binary repository go together!

→ do not back up only one or the other

→ walkthrough

PostgreSQL

→ pg_dump and pg_restore are helpful