

OPeNDAP Server

Neil Killeen

May 2005

Contents

1	Introduction	2
2	OPeNDAP Resources	2
2.1	Useful Web Sites	2
2.2	Mail list	2
2.3	Documentation	2
2.4	Downloads	2
3	Unix Administration	3
4	Configuration	3
4.1	Basic	3
4.2	HPSC	4
5	Build	5
5.1	Build from Source	5
5.2	Configure Template OPeNDAP Server	6
5.2.1	dods.rc	6
5.2.2	nph-dods	6
5.2.3	DODS_Dispatch.pm	7
6	Install	9
6.1	Data Area	9
6.2	OPeNDAP Server	10
6.3	Configure Webserver	10
6.3.1	Authentication of the OPeNDAP server	10
6.3.2	Setting up the alias	11
6.3.3	Configuring the Reverse-proxy webserver	12
6.4	Naming Consistency	12
7	Operations	12
7.1	New Project	12
7.2	Test Project	13

1 Introduction

This document describes information relevant to the installation and operation of the OpenDAP server at CSIRO-HPSC. OPeNDAP is an open-source project. OPeNDAP is often referred to as DODS, which I think is what OPeNDAP was created from.

The intended audience is for HPSC staff who need to administer OPeNDAP, not for end users. The reader of this document should already be familiar with OPeNDAP documentation available from the website (see § 2.1).

2 OPeNDAP Resources

2.1 Useful Web Sites

- www.opendap.org - This is the home web site

2.2 Mail list

There are several mail lists which you can subscribe to from the web site. The main one is called `dods@unidata.ucar.edu`. The mail list is active, responsive and helpful.

2.3 Documentation

The OPeNDAP web site has quite good FAQs plus various specific guides. Some of it is out of date, but still close enough to the current software that this is not a big problem.

2.4 Downloads

You must build a server for each data type you wish to support. Each server comes with its set of tar files to download. Most of the servers require the same common infrastructure.

HPSC is supporting data types/servers:

- netCDF
- FreeForm

requiring common infrastructure

- `DODS-asciival-3.4.3.tar.gz`
- `DODS-www-interface-3.4.3.tar`
- `DODS-dap-3.4.8.tar`
- `DODS-packages-3.4.4.tar.gz`

and server-specific software

- DODS-ff-dods-3.4.3.tar.gz
- DODS-nc3-dods-3.4.7.tar.gz

3 Unix Administration

OPeNDAP was built only on host 'cherax' under my account **kil064**; there is no special account. The source resides in `/tools/OPeNDAP` (which I will refer to as `$OPeNDAP`). There are a number of subdirectories which I have created by hand (do NOT delete them !).

- `Downloads` - zipped tar files and modern `configure` files.
- `Doc` - where I have put the source for any HPSC documentation (including this document)
- `cgi-bin` - where I copy, save and modify the server scripts before installing them
- `SRB` - where modifications to integrate with SRB can be found (not installed at HPSC as yet)

The OPeNDAP server is accessed via the host webserver. The server is not running all the time; it consists of scripts that are executed as required.

4 Configuration

4.1 Basic

OPeNDAP is designed basically to have one server to serve public data directly from a visible web server host. The OPeNDAP server is placed in the `cgi-bin` tree of the webserver and the user constructs a URL pointing at scripts in that directory which access data visible to the root document directory of the webserver.

It can also be set up so that authentication is required for the server.

However, the OPeNDAP server can reach any data that is visible from the root document directory. The OPeNDAP server bypasses any webserver authentication apart from that set up explicitly on the OPeNDAP server directory.

The main OPeNDAP script operates by handing off functionality to support programs such as `curl` and its own programs such as `www_int`. Some of these access the data via the webserver, some go straight to the data and some get to it via the OPeNDAP server itself.

OPeNDAP does not pass any authentication on to these support programs and so if you operate with authentication enabled you must take with your

webserver configuration so that the support programs can run without requiring authentication.

This is a security hole in the design because it means **any** process operating on the server machine can access the data without authentication. For example, if you run a browser or client program on the same host as the OPeNDAP server, all authentication is bypassed.

Some of the support programs also constructs URLs embedded in HTML fragments. As discussed below, these URLs may be incorrect if your environment does not match the expected one (e.g. you are behind a firewall and proxy server).

4.2 HPSC

At HPSC we are dealing with an OPeNDAP server (in host 'cherax.hpsc.csiro.au') that is behind a firewall and accessed via a proxy server ('hpsc.csiro.au'). In addition, we wish to protect data so that users can access only their data. This required some configuration changes and modifications to the OPeNDAP scripts since this is not the environment for which it is designed.

The configuration we are using at HPSC is to have one OPeNDAP server per project and users are required to authenticate that server with a unique password. Each server has access only to one specific data area where both the data and the authentication password file reside. This data area is not in the webserver root document tree and the password file is controlled by the user.

This setup would provide total data security. However, OPeNDAP uses **curl** to access the data area to make directory listings. **curl** itself uses the webserver and so therefore we must define a webserver alias for each data directory. Unfortunately, this allows navigation to the data directory from the root document tree via the webserver (not via the OPeNDAP server). Therefore, as well as authenticating the server, this data directory also requires authentication, should a user endeavour to navigate there via the webserver alone (unlikely since they won't know about the alias).

The server directory setup looks like :

```
/var/www/cgi-bin/OpenDAP
    /Project1
        nph-dods                # Server files
        DODS_Dispatch.pm
        ...
    /Project2
        nph-dods                # Server files
        DODS_Dispatch.pm
        ...
```

etc. with one server directory per project. The data directories are equivalently named

```
/cs/datastore/OpenDAP
```

```
/Project1
.password          # Authentication
/Project2
.password
```

etc. For each project there is a webserver alias named e.g. `Project1`, `Project2` etc. to access the data areas from the webserver.

The modifications that were required and the webserver setup are described in detail in the following sections. Briefly, the modifications that are required are to allow

- Data access out of the webserver document root
- Data protection
- Correct URLs in HTML fragments (allowing for the proxy)

5 Build

5.1 Build from Source

- Place the tar files in the `$OPENDAP` directory and unpack them. This will create a `DODS` subdirectory tree.
- Remove any spurious (wrong operating system) pre-existing object files in the tree. You can find them with

```
% find . -name '*.o' -print
```

- Copy the modern autoconf files (recognizing the SGI altix machine) from `$OPENDAP/Downloads/config.guess,sub` to `$OPENDAP/etc`
- Configure and build

```
% cd $OPENDAP
% ./configure
% make World
```

Note that if you clean the build via

```
% make distclean
```

you will need to re-run the `configure` script as the `distclean` target deletes some critical files.

5.2 Configure Template OPeNDAP Server

After installation, each server will be self-contained in a subdirectory of the webserver cgi directory. As a staging or template area, I use the directory `$OPENDAP/cgi-bin`. This is where I copy the server files from the build tree, and make basic modifications for a template project. If you are upgrading, you better save this directory first before you delete or overwrite its contents!

There are copies of the unmodified files in here as well (extension `.orig`). The best way to understand the modifications is to diff the files, but I will describe them here anyway incase disaster ensues and you start from scratch.

You can copy the server to this directory by running `$OPENDAP/DODS/etc/installServer` and answering the questions in the obvious way.

Now you must modify three files in this directory. Now, since this is a template setup, let's say we are modifying the server for a project called `TestProject` (see § 4).

5.2.1 dods.rc

Modify `dods.rc` so that

```
cache_size 1000                                # say
cache_dir /work/tmp
```

but these are not critical changes.

5.2.2 nph-dods

This is the main driving script. It makes an object Perl object of class `DODS_Dispatch` and calls its functions. The main function is `command` which prepares a command to invoke one of the support programs like `curl` and `www_int`.

`www_int` is used to generate an HTML form describing a data file. It contains a URL to provide access to that data file. The URL will contain 'cherax' which is no good because it is not externally visible. Therefore we must parse the HTML and replace 'cherax' by the calling server (usually 'hpsc').

So replace

```
if ($command[0] ne "") {                        # if no error...
    exec(@command);
} else {
    $dispatch->print_error_msg($bad_command, 1);
}
```

by

```
if ($command[0] ne "") {
    if ($command[0] eq "./www_int") {           # Trap command used for .html
        my $h1 = "cherax.hpsc.csiro.au";
        my $h2 = $dispatch->server_name_orig();
    }
}
```

```

#
    open(WWW_INT, "-|", @command);
    while (<WWW_INT>) {
        s/$h1/$h2/g;                # Replace
        print;
    }
    close(WWW_INT);
} else {
    exec(@command);
}
} else {
    $dispatch->print_error_msg($bad_command, 1);
}

```

5.2.3 DODS_Dispatch.pm

The changes to this file are so that

1. The log files are named per project
2. The server can access its root data tree
3. Proxy URLs are not incorrectly used when invoking the support programs that OPeNDAP uses (e.g. `curl`, `www_int` etc.).

Modify `DODS_Dispatch.pm` (which is critical) as follows (easiest to diff the existing one with the original really).

1. Logfile - specify where the log file should go. Modify the open statement to something like

```
open(DBG_LOG, ">> /work/tmp/OpenDAP-TestProject.log") if $debug > 0;
```

In this way we have one log file per server.

If you actually want the log file to be written you need to set the variable `$debug` to some value above 0 (I use 2). You might do this when you initially set the system up to make sure it is functioning, and then later turn it off.

2. Root document tree - You must tell the server where the root directory tree is that it is allowed to navigate. This can be achieved by modifying the value of the environment variable `$PATH_TRANSLATED` according to

```
$PATH_TRANSLATED = "/cs/datastore/OpenDAP/TestProject".$ENV{PATH_INFO};
```

This means this specific OPeNDAP server can only see the data in this tree. Put this line immediately after the log file is opened in the script.

3. Data alias - this variable will be used later on so that the data can be accessed by the webserver

```
my $dataAlias = "/TestProject";
```

Put this line after the previous addition.

4. Server name - we are operating OPeNDAP on 'cherax'. However, it will generally be accessed via the reverse-proxy webserver, 'hpsc.csiro.au'. We need to modify the function `server_name` to supply 'cherax' as the server name and add another function (called `server_name_orig`) to supply the caller server name.

```
sub server_name {                                     # Modify this function
    return "cherax.hpsc.csiro.au";
}

sub server_name_orig {                               # New function
    my $self = shift;
    return $self->{server_name};
}
```

5. Curl URL - The OPeNDAP server hands some of its jobs off to support programs. For example, it uses the program called `curl` to return the contents of a directory. However, `curl` operates by using the underlying webserver and therefore it can only access directories under the root document tree of the webserver. To solve this I have defined a webserver alias for each project (see § 6.3). `DODS_Dispatch.pm` then needs to be modified to use this alias when it uses `curl`. Locate the subroutine called `sub`. Then modify the lines

```
    } elsif ($self->ext() eq "/") {
use FilterDirHTML; # FilterDirHTML is a subclass of HTML::Filter
my $url = "http://" . $self->server_name() . $self->port()
    . $self->path_info();
```

(comments excluded) to be

```
    } elsif ($self->ext() eq "/") {
use FilterDirHTML; # FilterDirHTML is a subclass of HTML::Filter
my $url = "http://" . $self->server_name() . $self->port()
    . $dataAlias . $self->path_info();
```

thus inserting the `$dataAlias` that we defined earlier into the URL.

Now once `curl` has been called, the directory listing will be formatted in some HTML with URLs for the files it found. Those URLs must refer to the calling server (usually the proxy) not to 'cherax' because 'cherax' is not visible externally.

In the same section of the code modify the URL construction so that

```
# Parse the HTML directory page
# Build URL with CGI in it but remove ?M=A type query expression.
my $server_url = "http://" . $self->server_name() . $self->port()
                . $self->request_uri();
```

is replaced by

```
# Parse the HTML directory page
# Build URL with CGI in it but remove ?M=A type query expression.
my $server_url = "http://" . $self->server_name_orig() . $self->port()
                . $self->request_uri();
```

where we are using the new function `server_name_orig` we defined earlier.

6 Install

6.1 Data Area

You must create the data area and give ownership to the user who will manage it. E.g.

```
% cd /cs/datastore/OpenDAP
% mkdir TestProject
% chown -R kil064:csssg TestProject
```

Then you must create the password file that will be used by the webserver to authenticate the server with. Use a default password the same as the project name and then the user can change it. E.g.

```
htpasswd /cs/datastore/OpenDAP/TestProject/.password TestProject
```

will create the password file for username `TestProject` with the specified (you will be prompted for it) password encrypted.

6.2 OPeNDAP Server

In the previous section, we saw how to generate a template server setup for the project called `TestProject`. Now we need to install it in the webserver tree. We need one server per project. The servers are stored in subdirectories of the same name as the project in `/var/www/cgi-bin/OpenDAP`.

Apart from `TestProject`, my convention is to call them `Project1`, `Project2` etc.

These subdirectories could be full copies or partial copies using symbolic links. I am using the former as the overhead is not so great to make the extra bother of symbolic links worth the effort.

So all you need do is copy the contents of `$OPENDAP/cgi-bin` to the appropriate subdirectory in `/var/www/cgi-bin/OpenDAP`; in this case this subdirectory is called `TestProject`.

Since the files in `$OPENDAP/cgi-bin` are just a template, you must then edit `DODS.Dispatch.pm` in situ in the obvious way for whatever project you are creating. E.g. you might be editing `$OPENDAP/cgi-bin/Project3/DODS.Dispatch.pm` so that wherever it says “`TestProject`” in the template script, you would use “`Project3`”.

6.3 Configure Webserver

So now you have configured the OPeNDAP server. Now you must configure the webserver on cherax too.

6.3.1 Authentication of the OPeNDAP server

As we have mentioned, the only way to control access to data is to provide one OPeNDAP server per project, allow that server access only to a specified data area and authenticate on that server.

To authenticate the server area you need commands in the webserver (on ‘cherax’) config file `/etc/httpd/conf/httpd.conf` like

```
<Directory "/var/www/cgi-bin/OpenDAP/TestProject">
  Options ExecCGI Indexes FollowSymLinks

  Order deny,allow
  Deny from all
  # ALLOW SERVER (IP OF SERVER) MACHINE TO REQUEST DATA ITSELF
  Allow from cherax
  Require valid-user
  # ALL VISITORS NEED USERNAME AND PASS BUT NOT SERVER
  Satisfy any

  AuthType Basic
  AuthUserFile /cs/datastore/OpenDAP/TestProject/.password
```

```
    AuthName "OpenDAP TestProject Server"  
</Directory>
```

This tells the webserver that when the user tries to access the server directory `/var/www/cgi-bin/OpenDAP/TestProject` that it must authenticate with the username and password stored in `/cs/datastore/OpenDAP/TestProject/.password`. Thus, the password is in the user's control.

In addition, it tells the server to allow any access from host 'cherax' (note that for name resolution reasons 'cherax' should not be qualified further) directly. This is needed to get the support programs that the OPeNDAP server uses to work because the authentication is not passed on to them. It is a security hole as it means if you run something directly on 'cherax' (e.g. a client or browser) you bypass the whole authentication process. We would need to run the server on a different machine to solve that.

6.3.2 Setting up the alias

Because OPeNDAP uses `curl` to create directory listings, and `curl` uses the webserver, it means we have to make an alias from the root document tree of the webserver so that `curl` can access the data (this is a great pity as it's the only thing that needs this). Again you must edit the webserver config file (on 'cherax') `/etc/httpd/conf/httpd.conf`. Add lines like

```
Alias /TestProject /cs/datastore/OpenDAP/TestProject
```

to provide the alias. This directory also needs access rights and needs to demand authentication. For example, if someone happens to know that the alias exists (they don't ever actually need to use it directly) and tries to access the data area directly with the webserver, they will still be required to authenticate. In this way we have secured the data area (as well as the server).

Add lines like

```
<Directory "/cs/datastore/OpenDAP/TestProject">  
    Options ExecCGI Indexes FollowSymLinks  
  
    Order deny,allow  
    Deny from all  
    # ALLOW SERVER (IP OF SERVER) MACHINE TO REQUEST DATA ITSELF  
    Allow from cherax  
    Require valid-user  
    # ALL VISITORS NEED USERNAME AND PASS BUT NOT SERVER  
    Satisfy any  
  
    AuthType Basic  
    AuthUserFile /cs/datastore/OpenDAP/TestProject/.password  
    AuthName "OpenDAP TestProject Data"  
</Directory>
```

6.3.3 Configuring the Reverse-proxy webserver

The OPeNDAP server on 'cherax' will actually be accessed via the reverse-proxy server, 'hpscworld.hpsc.csiro.au' (or 'hpsc.csiro.au' or 'www.hpsc.csiro.au'). Now OPeNDAP uses some illegal URLs with square brackets in them. So the reverse-proxy must be told to not check URLs for legality. This means edit the file `/usr/local/etc/pound.cfg` so that the line

```
CheckURL 0
```

is present.

6.4 Naming Consistency

Splattered throughout the previous sections, the places which need to be named consistently (e.g. `Project3`) are the data area (e.g. `/cs/datastore/OpenDAP/Project3`), the server area (e.g. `/var/www/cgi-bin/OpenDAP/Project3`) and the webserver alias (e.g. `Project3`) pointing to the data area,

7 Operations

7.1 New Project

If someone requests OPeNDAP access you must

- Assign a project name. My convention is `ProjectN` so just pick the next available `N` such as `Project10`.
- Create the data area and password file for the project (see § 6.1 for details).
- Create the OPeNDAP server for this project by copying it from the template area to the webserver cgi area. For example

```
% cp /tools/OpenDAP/cgi-bin/* /var/www/cgi-bin/OpenDAP/Project10
```

and then modifying the scripts as needed for this project (see § 6.2 for details).

- Now configure the webserver for this project (see § 6.3 for details).
- Tell the user the password and how to change it (see § 6.1).
- The user can now put data in their `Project` data area and access it

7.2 Test Project

I use `TestProject` for testing. There is a little bit of data in it.

People can try out `OPeNDAP` with this project. Example queries from a browser are

```
http://hpsc.csiro.au/cgi-bin/OpenDAP/TestProject/nph-dods/mytest.nc.info
http://hpsc.csiro.au/cgi-bin/OpenDAP/TestProject/nph-dods/mytest.nc.dds
http://hpsc.csiro.au/cgi-bin/OpenDAP/TestProject/nph-dods/mytest.nc.das
http://hpsc.csiro.au/cgi-bin/OpenDAP/TestProject/nph-dods/mytest.nc.html
http://hpsc.csiro.au/cgi-bin/OpenDAP/TestProject/nph-dods/mytest.nc.asc?lat
http://hpsc.csiro.au/cgi-bin/OpenDAP/TestProject/nph-dods/test
http://hpsc.csiro.au/cgi-bin/OpenDAP/TestProject/nph-dods/
```

and the first time they will be prompted to authenticate with both the username and password `TestProject`.

You can see that the URL is constructed with the specific `OpenDAP` server and that it knows where to find the data file `mytest.nc` which resides in the datastore in `/cs/datastore/OpenDAP/TestProject`.