Importing data directly into XNAT

These procedures are useful when you have a large amount of data on the same machine or local network as your XNAT installation. These procedures presume that you can access both your XNAT data directories and the location of the data you want to import via standard local file access (including performant network-mounted file systems such as NFS or Samba) instead of requiring HTTP or FTP access. This avoids sending a large amount of data over the network and through your Tomcat server, which makes it much quicker to get your data into XNAT.

There are two main ways you can accomplish this task:

- Copying files directly into the prearchive and creating a row in the prearchive table for the data
- Using a script to send the data through XNAT's DICOM SCP receiver

There are pros and cons to both approaches, which are discussed along with the procedures.

You can get the study instance UID from a DICOM study through a number of different ways. A commonly used tool for this sort of thing is **dcmdump**, which is part of dcmtk. You can dump the specific tag for the study instance UID with the following command:

```
[xnat@xnatdev prearchive]$ dcmdump +P 0020,000d case001_MR1/case001.MR.NeuroMRT.10.19.n0ti51.dcm
(0020,000d) UI [1.2.840.113654.2.45.2.108105] # 28, 1 StudyInstanceUID
```

Note that you can do this with any file from a session: the study instance UID must be the same for all files from a single DICOM study.

Copying files directly into the prearchive

This procedure takes advantage of how XNAT stages data for processing in the prearchive. Normally, when you send data into XNAT via the applet or some other upload or import functionality, the data is handled by an importer process. The importer places the incoming files into a folder in XNAT's prearchive location, sorting the files into the various series that compose the DICOM study. Upon receiving the first file for a new session, the importer creates a row in the database in a table named **xdat_search.prearchive**, setting the status to **RECEIVING**, indicating that data is still be collected for the session. For every file that arrives for the same session, the **timestamp** column of the database entry is updated. There's a scheduled process that runs and checks the prearchive table for sessions that haven't been updated in more than some set amount of time (by default XNAT uses 5 minutes). Once the latency time has passed, the prearchive entry is updated to **READY** status and an initial session XML document is built based on the DICOM metadata.

Working prearchive table insert

```
INSERT INTO xdat_search.prearchive (project, timestamp, lastmod, uploaded, scan_date,
                                    scan_time, subject, foldername, name, tag, status,
                                    autoarchive, prevent_anon, prevent_auto_commit, url)
     VALUES ('prj001', '20140818_144524024', '2014-08-18 11:33:30.621', '2014-08-18 14:45:24.024',
             '2013-12-11 00:00:00', '15:18:18', 'prj001_001', 'prj001_001_MR1', 'prj001_001_MR1',
             '1.2.840.113654.2.45.2.108105', 'RECEIVING', 'Manual', 'f', 'f',
             '/data/xnat/prearchive/prj001/20140818_144524024/prj001_001_MR1');
```

This procedure basically fakes that process by putting the data directly into the prearchive process:

1. Copy or move your data into a folder within XNAT's prearchive folder. The folder you put the data in must have the following path (note that *prearchive* indicates the location of your XNAT's prearchive folder):
   - If you know the project into which you want to archive your data, put the files into the folder *prearchive/projId/timestamp/sessionId*. The *timestamp* must have a value like *YYYYMMDD_HHmmsssss* (note that *sssss* is seconds and milliseconds), for example 20140818_144524024.
   - If you don't know the project, put the files into the folder *prearchive/timestamp/sessionId*.
2. Insert a new row into the prearchive table using the query format in the code sample above:
   - The value for **project** should be the project ID, *not* the project label or description (if you didn't use a project in step 1, then just leave the project value empty)
   - The **timestamp** value must be the same value for the timestamp column *and* the folder name you used in step 1
   - The value for **subject** must be the subject ID (if you don't know the subject, leave the subject value empty)
   - The value for **foldername** is the same as the session label and should the same as the folder you used for *sessionId* in step 1
   - The value for **tag** must be the study instance UID for the session (see the tip above for a method to get the study instance UID)
3. Rebuild the session. You can either:
   - Go to the prearchive page in the XNAT application, select your incoming session, and click **Rebuild**
   - POST to the prearchive rebuild REST function (/data/services/prearchive/rebuild), with the form variable **src** set to the value **/prearchive/projects/*projId*/*timestamp*/*sessionId***

Once you've completed these steps, your session should be ready to be archived into the XNAT system. You can just select the session in the prearchive interface and click **Archive** or **Review and Archive**.

As noted earlier, there are pros and cons to each of these methods. This method has a number of good points. It's very quick and continues to leverage the data processing steps of the archiving process. The primary drawback with this approach is that the data is not structured by the DICOM importer process. Data that goes through the DICOM importer is placed into a folder structure that organizes the data within the study by series or scan. The overall session XML is located at the top of the session structure. Files for each of the scans are placed into folders organized by scan number, along with the XML document that contains the manifest for each scan. If your data comes into XNAT without that structure, it will stay unstructured. As long as your data is only accessed directly by the XNAT application, this is usually OK. If your data is ever given over to users to work with, it can make the session data unwieldy, since a single DICOM study may comprise thousands or even tens of thousands of files.

Sending data to the DICOM SCP receiver

The XNAT DICOM SCP receiver is a standard DICOM C-STORE end-point service. You can leverage the DICOM receiver by using any standard C-STORE sender, such as dcmtk's **storescu** or XNAT's DicomBrowser application.

The main information you'll need is the IP or hostname of your XNAT instance and the AE title and port of the DICOM SCP receiver configured on your XNAT. For command-line applications such as **storescu** or **DicomRemap** (a command-line utility that comes as part of the DicomBrowser package), you usually need to send each file separately, using **find** or a similar utility to locate all of the files in your session. For example:

```
find . -name *.dcm -exec storescu --aetitle LOCAL --caller XNAT xnatdev.foo.com 8104 {} \;
```

Note that the value for the **--aetitle** option can be any random string: XNAT's DICOM receiver accepts all incoming data without regard to the originating application entity. However, the argument for the **--caller** parameter must match the configured AE title for your XNAT's DICOM SCP receiver.

storescu also has an option for scanning directories recursively for input files, **+sd** or **--scan-directories**.

You can also use a number of GUI applications to send DICOM data directly to XNAT, including **DicomBrowser** and **Osirix**.

The advantage of this method is that it can handle quite a lot of incoming data, while avoiding the web application and overhead associated with HTTP transactions. It doesn't require you to mess around with finding the study instance UID or other specific data. And because this method goes through the SCP receiver, it also receives the data structuring from the XNAT importer that is missed when you use the direct-to-prearchive method described above. However, this method is not nearly as fast as the direct-to-prearchive method.

Related articles

- Importing data directly into XNAT
- Clearing stuck sessions from the prearchive
- Uploading Zip Archives to XNAT