DAITSS Production Ingest Quick Start Guide

1 Login to darchive using personal account.

- If accessing darchive outside of the FCLA subnet, initiate a FCLA VPN connection.
- From a Unix or Linux based computer, open a terminal and type: `ssh <username>@darchive.fcla.edu`
- From a Windows computer, open a secure shell client (such as Open SSH), create a profile for host darchive.fcla.edu, and login using that profile.
- You will be prompted to enter your password.

2 Check to see if ingest is already running in production.

- `ps aux | grep "daitss .*ingest" | wc -l`
- A return value of 2 or higher means that ingest is already running in production. Do not continue.
- A return value of 1 means that ingest is not running. Continue.

3 Check to see if ingest is running in other regions.

- Contention for TSM can lead to significantly lengthened storage cycles.
  - If ingest is running in other regions, make sure there will not be contention for TSM tape drives.
  - Once the FDA has a dedicated production machine, this step will not be necessary.

3.1 Check if ingest is running in test.

- `ps aux | grep "daitss_test .*ingest" | wc -l`
  - A return value of 1 means that ingest is not running in test.
  - A return code of 2 or greater means that ingest is running in test.
    - Make sure that the `WRITE_TO_STORAGE` property for test is set to false.
      - `grep "^WRITE_TO_STORAGE=" /daitss/test/daitss/config/daitss.properties`
      - If `WRITE_TO_STORAGE` is true, there will be contention for TSM. Do not continue.
      - If `WRITE_TO_STORAGE` is false, there will not be contention for TSM. Continue.

3.2 Check if ingest is running in dev.

- `ps aux | grep "daitss_dev .*ingest" | wc -l`
  - A return value of 1 means that ingest is not running in dev.
  - A return code of 2 or greater means that ingest is running in dev.
    - Make sure that the `WRITE_TO_STORAGE` property for dev is set to false.
      - `grep "^WRITE_TO_STORAGE="`

`/daitss/dev/daitss/config/daitss.properties`

- If `WRITE_TO_STORAGE` is true, there will be contention for TSM. Do not continue.
- If `WRITE_TO_STORAGE` is false, there will not be contention for TSM. Continue.

4 Switch user to the DAITSS production user account.

- `su – daitss`
- Enter the password.

5 (Optional) Verify that a DAITSS distribution exists.

- `ll $DAITSS_HOME/dist/daitss.jar`

6 (Optional) Check status of OIDServer and rmiregistry and initiate them if necessary.

- `ps aux | grep prod.*OIDServer | wc -l`
- A return value of 2 means the production OIDServer is running. Go to Step 7.
- A return value of 1 means it is not running.
  - Check to see if the rmiregistry is running.
    - `ps aux | grep rmiregistry | wc -l`
    - A return value of 2 means the rmiregistry is running.
    - A return value of 1 means the rmiregistry is not running.
      - Start the rmiregistry.
        - `rmiregistry &`
  - Start the OIDServer.
    - `cd ~/scripts`
    - `./startOID 2>&1 > /dev/null &`

7 Examine reject directories.

7.1 Note prep rejects and clear them from rejects directories.

- `ll ~/data/prep/reject`
- If the result shows any packages, note them and then delete them.
  - `rm -r ~/data/prep/reject/*`

7.2 Note ingest rejects and clear them from rejects directories.

- `ll ~/data/ingest/reject`
- If the result shows any packages, note them and then delete them.
  - `rm -r ~/data/ingest/reject/*`

7.3 For all rejected packages, determine reason for rejection.

- Find the most recent log file.

- prep
  - `ll ~/data/logs/preprocess`
- ingest
  - `ll ~/data/logs/ingest`
- The most recent log file is listed last

- Search the log file for reason for rejection.
  - `grep -B5 "<package name>.*SIP rejected" ~/data/logs/ingest/<log file name>`
    - To read more log entries before the point of rejection, increase the number for B

- If the package was rejected due to a system error (with a possibility of being ingested successfully on a subsequent ingest), leave the package in the input directory.
  - Examples are failure during storage and errors communicating with the database.

- If the package was rejected due to an error with the package itself (and will not be ingested successfully on a subsequent ingest), move the package from the input directory to a rejects holding area.
  - Examples are mismatch in message digest (checksum) values, error during PDF normalization, and unknown AP code.
  - prep
    - `mv ~/data/prep/in/<package name> <path to prep rejects storage directory>`[1], or
  - ingest
    - `mv ~/data/ingest/in/<package name> <path to ingest rejects storage directory>`

8 Populate prep input directory.

- There are two ways to fill the input directory, with packages that have descriptors or with packages that do not have descriptors.
- Once a package has been successfully processed by prep, it will be moved to the ingest input directory.

8.1 Populating with packages that do not have descriptors.

- Make sure all packages are supposed to have the same account, project, and subaccount.

1 For more uses of the mv command, type man mv

- Set configuration properties[2]
  - Open the config file
    - `nano $DAITSS_HOME/config/daitss.properties`
  - Set the `GENERATE_DESCRIPTORS` property to true
    - Find the property
      - `ctrl+w`
      - `GENERATE_DESCRIPTORS <enter>`
    - Set the property to true
      - The line must read: `GENERATE_DESCRIPTORS=true`
  - Set the `DEFAULT_ACCOUNT`
    - The account code must exist in the ACCOUNT table.
    - Set the `DEFAULT_ACCOUNT` property to the desired account code.
  - Set the `DEFAULT_PROJECT`
    - The project code must exist in the PROJECT table.
    - The combination of account and project must exist in the ACCOUNT_PROJECT table.
    - Set the `DEFAULT_PROJECT` property to the desired project code.
  - Set the `DEFAULT_SUBACCOUNT`
    - If no sub-account is needed, the code will be “”
    - If a sub-account is needed, the code must exist in the SUB_ACCOUNT table and it must be associated with the account defined in `DEFAULT_ACCOUNT`.
    - Set the `DEFAULT_SUBACCOUNT` property to the desired sub-account code.
  - Close the config file
    - `ctrl+x`
    - If you have made changes to the document, you will be prompted to save those changes before the properties file is closed.

8.2 Populating with packages that have descriptors.

- Set the `GENERATE_DESCRIPTORS` property to false.
- Set the `SUPPLY_ACCOUNT_INFO` property to false.[3]

---

2 To view a property value without opening the file for editing, type: `grep "^<property name>=" $DAITSS_HOME/config/daitss.properties`

3 In some cases, it may be necessary to programatically supply account information to existing descriptors that are missing

8.3 Move packages to prep input directory (on the reservoir)

- `mv <package_path1> [<package_path2> ... <package_pathN>] /res/daitss/prod/data/prep/in`

9 Prep data

- Count number of packages in input directory.
  - `ls -1 /res/daitss/prod/data/prep/in | wc -l`
  - The number of packages will be used later to determine prep progress.
- `cd $DAITSS_HOME`
- `ant -f util.xml prep 2>&1 > prep.out &`
- To determine when prep is done, see Step 11: Check for completion.
- Logout of daitss user account.
  - `ctrl+d`

10 Ingest data

- Wait until prep has finished (see Step 11: Check for completion).
- Count number of packages in input directory.
  - `ls -1 /daitss/prod/data/ingest/in | wc -l`
  - The number of packages will be used later to determine prep progress.
- Login to daitss user account
  - `su - daitss`
- Make sure the `WRITE_TO_STORAGE` property is true.
- `cd $DAITSS_HOME`
- `ant -f util.xml ingest > ingest.out 2>&1 &`
- Logout of daitss user account.
  - `ctrl+d`

11 Check for completion

- Do not login as the daitss user to check for progress and completion.

11.1 Examine output file.

- `cd /daitss/prod/daitss`
- Read the end of the outputfile.

---

it. This should only be done when descriptors are missing the information and when all such descriptors need the same account, project, and subaccount. When setting SUPPLY_ACCOUNT_INFO to true, the DEFAULT_ACCOUNT, PROJECT, and SUBACCONT properties must be set according to the same rules as when GENERATE_DESCRIPTORS is set to true.

    - prep
        - `tail prep.out`
    - ingest
        - `tail ingest.out`
- If the end of the output file contains the message “BUILD SUCCESSFUL”, processing is done.
- If the end of the output file does not contain the “BUILD SUCCESSFUL” message, processing has not finished, processing has ended abnormally (halted), or processing has hung.
    - Check for halted process
        - See Step 2 to determine if ingest is running in production.
        - An ingest or prep process that has ended abnormally will show a Java stack trace near the end of the output file
            - It may be necessary to search the last 50 (or so) lines of the output file
            - `tail -50 <output file>`

    - Check for hung processing
        - Hung processes are generally the result of errors related to TSM
        - Many messages written to the output file will contain a timestamp. If the last message is over an hour old, it is likely that processing has hung, especially if the message is for writing to tape.
            - If the message does not contain a timestamp, it will be necessary to check the log file which automatically applies a timestamp to all entries. Since all logs contain a timestamp in their name, the current log will be the last log listed in the logs directory.
                - List all prep logs
                    - `ll ~/data/logs/prep`
                - List all ingest logs
                    - `ll ~/data/logs/ingest`
                - View the end of the log
                    - `tail <path to log file>`
        - It is possible that some functions can take an unexpectedly long time to complete due to data characteristics. Familiarity with DAITSS processing will be helpful in determining whether a process is active or hung. It is difficult to account for all possibilities in this guide, but one example is creating TIFF images from a PDF during normalization. The larger the PDF, the longer the TIFF generation.
        - If the process has not halted and is not hung, go to the next step: Check progress

11.2 Check progress

- The number of packages left to process is the difference between the number of packages in the input directory minus the number of packages in the rejects directory.
  - prep
    - Number of packages in input directory
      - `ls -1 /res/daitss/prod/data/prep/in | wc -l`
    - Number of packages in reject directory
      - `ls -1 /res/daitss/prod/data/prep/reject | wc -l`
  - ingest
    - Number of packages in input directory
      - `ls -1 /daitss/prod/data/ingest/in | wc -l`
    - Number of packages in reject directory
      - `ls -1 /daitss/prod/data/ingest/reject | wc -l`