University of Colorado Libraries

University Libraries

Jul 26, 2023

CONTENTS

Documentation covering Core Tech & Apps(CTA) Products and Policies.

CHAPTER

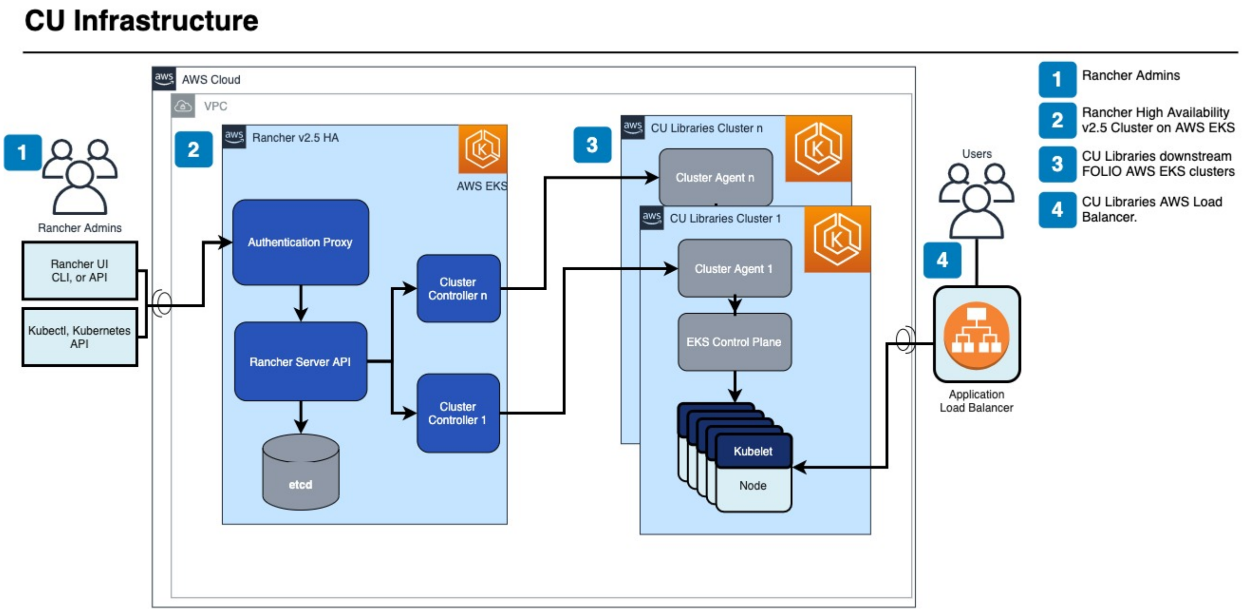
ONE

TABLE OF CONTENTS

1.1 Frequently Asked Questions

1.2 Infrastructure

CU Libraries infrastructure is leveraging AWS Cloud Services.



1.2.1 Kubernetes

Production products are deployed in containers(micro-service) within a kubernetes cluster. Rancher is used for cluster management and deployments.

1. Highly available
2. Horizontal Scalable: Products have the ability to scale up or down based on demand.

AWS EKS

The production clusters are using AWS EKS infrastructure with compute nodes using AWS EC2 instances.

Note: AWS EKS runs the Kubernetes control plane across multiple Availability Zones, automatically detects and replaces unhealthy control plane nodes, and provides on-demand, zero downtime upgrades and patching. EKS offers a 99.95% uptime SLA. At the same time, the EKS console provides observability of your Kubernetes clusters so you can identify and resolve issues faster.

EKS Worker Nodes

The EKS worker nodes are rotated on an annual basis. EC2 instance provides on-demand security patches.

1.3 AWS Tips and Tricks

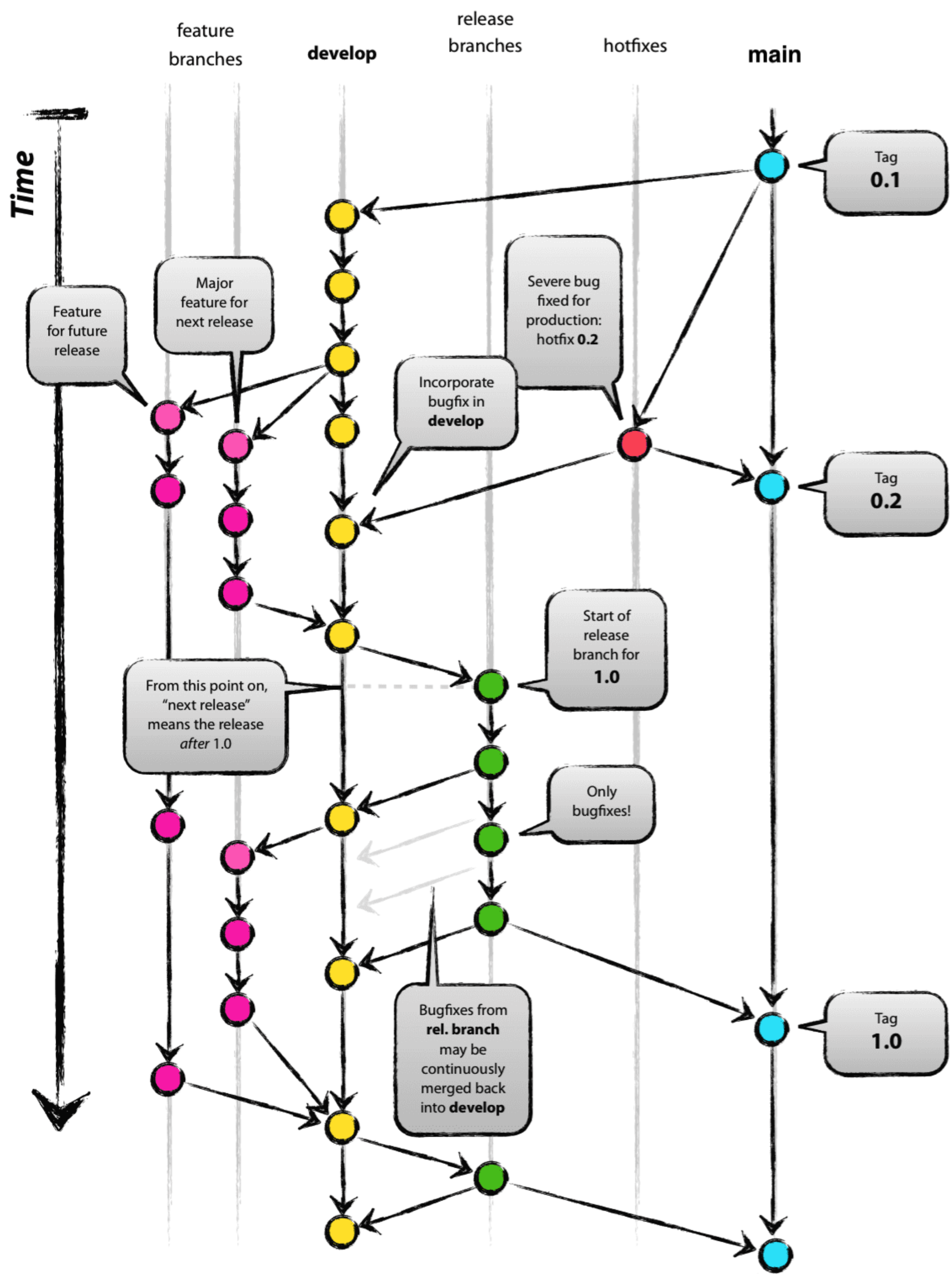
Login to AWS Console via OIT FedAuth using bit.ly/OIT-AWS

1.4 Version Control

CU Libraries uses git as its version control system. All repositories are stored remotely within Github

1.4.1 Branch Management

Git Flow model for branch management.


feature branches
develop
release branches
hotfixes
main
Time
Tag 0.1
Feature for future release
Major feature for next release
Severe bug fixed for production: hotfix 0.2
Incorporate bugfix in develop
Tag 0.2
Start of release branch for 1.0
From this point on, "next release" means the release after 1.0
Only bugfixes!
Bugfixes from rel. branch may be continuously merged back into develop
Tag 1.0

Main Branches

1. main
2. develop

Supporting Branches

1. Feature branches
2. Release branches
3. Hot-fix branches

Pull Request with Code Review

A Pull Request(PR) is required to merge code into the `main` branch. All PRs to the `main` branch require a code review.

1.5 Data Backup Policy

CU Libraries Core Tech and Apps provides backup of all production applications. The policy changes with the format of the data. Production applications are deployed using AWS services.

1.5.1 Data Files

Data files are backed up using the 3-2-1 rule.

- 3 – Keep 3 copies of any important file: 1 primary and 2 backups.
- 2 – Keep the files on 2 different media types to protect against different types of hazards.
- 1 – Store 1 copy offsite (e.g., outside your home or business facility).

1. S3

AWS S3 is preferred storage for production data files. AWS s3 can be tuned to provide automatic replication.

1. Elastic Block Store

AWS EBS snapshots are held for a rolling 30 day period. This provides quick restoration of application components. For example the Solr index EBS blocks can be quickly restored for minimal downtime.

1. PetaLibrary

The PetaLibrary is a University of Colorado Boulder Research Computing service support the storage, archival, and sharing of research data.

CU Libraries stores copies of production data files as the offsite copy within the PetaLibrary. All other copies are located with the AWS infrastructure.

1.5.2 Relational Database

Production Relational Databases are backed up daily via Amazon Web Services RDS. These backups are held for a rolling 30-day period. Additionally, AWS RDS is built on distributed, fault-tolerant, self-healing Aurora storage with 6-way replication to protect against data loss.

1.6 FTP Server

The **FTP server** is used by CTA for transferring files to/from University Libraries using FTP or secure file transfer. The service is deployed on a dedicated EC2 instance.

Supported use cases include:

- Transfer of ETD submissions from ProQuest (SSH file transfer)
- Transfer of bursar out files from Sierra (FTP)

As ProQuest uses secure file transfer, its public key is contained in the authorized_keys file. It's corresponding user account (proquest) does not use a password (this is the preferred method to connect to an EC2 instance).

Sierra relies on FTP to transfer files and therefore has a password-enabled user account (sierra). There is no expiry date on the password. Credentials are stored in Keepass. Note that transfer of files is initiated through the Sierra console and is usually done weekly by the Fin Clerk responsible for patron fines processing.

1.6.1 Infrastructure

The FTP service is hosted on a single AWS EC2 instance (ftp-prod) contained within the US West 2 production VPC. Its public-facing IP address is 54.187.105.7. Refer to the AWS console for other instance details.

External traffic to/from the instance is controlled by the security group cubl-ftp-sg.

File storage is provided by an EFS network file service (FTPFS). Each use case is provided its own directory for file storage, e.g., /data/proquest and /data/sierra. Further delineation between production and test is provided by corresponding access points. For example, in a production environment, the access point /prod is defined and the local mount point is /data. The access point for testing purposes is /test. Access to/from EFS is controlled by security group cubl-ftpfs-sg.

1.6.2 FTP service

While SSH file transfer is the preferred method to send files from an external service provider, there may be situations where FTP is the only option. In that case, an FTP daemon is installed on this server to handle file transfers. This service is **vsftpd** or Very Secure FTP Daemon. The behavior of the service is controlled by a single configuration file located at `/etc/vsftpd/vsftpd.conf`. Refer to https://github.com/culibraries/ftp-server for the current configuration settings.

The preferred FTP method is explicit FTP over TLS, which is more secure than plain FTP. This method ensures that traffic between the client and server is encrypted and secure. The support of encrypted connections requires an SSL certificate. This can be generated using openssl or one can be generated/purchased elsewhere. For this installation, the SSL key is placed at `/etc/ssl/private/vsftpd.key` and the certificate is located at `/etc/ssl/certs/vsftpd.crt`. The SSL settings in vsftpd.conf handle the rest. Refer to Configure VSFTPD with an SSL for details on how to set this up.

NOTE: The certificate was recently renewed (10/25/2021) for a two-year period.

Refer to the manpage for the various configuration options and allowable values. Another good resource (with examples) is available at the Ubuntu Community Help Wiki.

1.7 Rancher (Original)

The original deployment of rancher was on EC2 instance. Single docker container running 2.2.x version of Rancher.

1.7.1 Access TO EC2 Instance

1. Public Key(ec2-user@libops.colorado.edu)
2. Vida is currently the only CTA member

1.7.2 System Operations

```
service-rancher
Usage: /opt/bin/service-rancher {status|start|stop|restart}
```

1.7.3 Backup

The data volume for the EC2 instance has automatic snapshots for 15 days.

1.7.4 Administration

The production cluster is a single docker container that deployed a Rancher RKE cluster. If a node is terminated or has pressure(Disk/Memory/CPU), the node requires manual interaction. Spot instance terminated, delete the node in Rancher. If a node has pressure, you can download keys and ssh into the EC2 Instance. I find it easier to delete the node and generate a replacement.

1.7.5 Historical Error and Resolution

Local Certs Expired

The Rancher UI did not come up and kept restarting itself due to accessing the API server under localhost:6443.

Error

```
localhost:6443: x509: certificate has expired or is not yet valid
```

Solution

- Rancher Single Node setup on libops.colorado.edu
- Rancher docker container was running v2.2.8 at the time the error was generated
- Container had local file folder mount -v /opt/rancher/etcd:/var/lib/rancher

```
$ sudo su -
$ cd /opt/rancher/etcd/management-state/tls/
# Check expiration of cert
$ openssl x509 -enddate -noout -in localhost.crt
  notAfter=Apr  8 17:27:09 2020 GMT
$ mv localhost.crt localhost.crt_back
$ exit

$ service-rancher restart
```

- The docker container restarted and the system updated the certificate. I also updated the container to v2.2.10.

Notes

This error was difficult to track down. Found solution at the end of this [https://github.com/rancher/rancher/issues/20011#issuecomment-608440069 Rancher/Rancher Issues]. The issue resulted in about 12 hours of downtime from the single node rancher deployment. The Kubernetes production and test clusters continued to run without interruption.

Unable to add etcd peer to cluster

This error occurred within the test cluster when a spot instance was terminated.

Rancher Error within UI

- Failed to reconcile etcd plane: Failed to add etcd member [xxx-xxx-xxx-xxx] to etcd cluster

Solution

- Logged into Kubernetes Node
- Rancher UI Cluster > Nodes … Download Keys

```
ssh -i id_rsa rancher@<ip of node>
docker logs etcd
...
2020-04-16 02:37:26.327849 W | rafthttp: health check for peer 5f0cd4c2c1c93ea1 could␣
↪not connect: dial tcp 172.31.30.190:2380: i/o timeout (prober "ROUND_TRIPPER_SNAPSHOT")
...

docker exec -it etcd sh
etcdctl member list
5f0cd4c2c1c93ea1, started, etcd-test-usw2b-spot-1, https://172.31.30.190:2380, https://
↪172.31.30.190:2379,https://172.31.30.190:4001
88d3ad844b3306a5, started, etcd-test-usw2c-spot-1, https://172.31.11.25:2380, https://
↪172.31.11.25:2379,https://172.31.11.25:4001
b0c2cb2c8e55611f, started, etcd-test-usw2c-spot-2, https://172.31.4.219:2380, https://
↪172.31.4.219:2379,https://172.31.4.219:4001
efb9f597e4952edb, started, etcd-test-usw2c-spot-3, https://172.31.8.159:2380, https://
↪172.31.8.159:2379,https://172.31.8.159:4001
```

The problem was that the spot node was terminated but the etcd cluster did not release the node. Checked Rancher UI for IPs and 172.31.30.190 was no longer available.

```
etcdctl member remove 5f0cd4c2c1c93ea1
Member 5f0cd4c2c1c93ea1 removed from cluster c11cbcba5f4372cf
etcdctl member list
88d3ad844b3306a5, started, etcd-test-usw2c-spot-1, https://172.31.11.25:2380, https://
↪172.31.11.25:2379,https://172.31.11.25:4001
b0c2cb2c8e55611f, started, etcd-test-usw2c-spot-2, https://172.31.4.219:2380, https://
↪172.31.4.219:2379,https://172.31.4.219:4001
efb9f597e4952edb, started, etcd-test-usw2c-spot-3, https://172.31.8.159:2380, https://
↪172.31.8.159:2379,https://172.31.8.159:4001
```

Returned to UI and added new etcd instances. etcd nodes most always be an odd number. Nodes vote stuck in split-brain or etcd cluster had 4 nodes and was waiting on non-existent node to vote.

Unable to mount volume

If all nodes are not in a subnet that contains a volume. Deployment will fail with volume mount.

1.8 Enterprise Logging

The logging stack consists of AWS Open Search , OpenSearch Dashboards, and Fluent Bit.

1.8.1 Installation

The installation was directly copied from AWS EKS Workshop with one exception. The first item is to create an OIDC identity provider. This had already been done when installing the AWS load balancer.

1.8.2 Components

1. Fluent Bit: an open source and multi-platform Log Processor and Forwarder which allows you to collect data/logs from different sources, unify and send them to multiple destinations. It’s fully compatible with Docker and Kubernetes environments.
2. Amazon OpenSearch Service: OpenSearch is an open source, distributed search and analytics suite derived from Elasticsearch. Amazon OpenSearch Service offers the latest versions of OpenSearch, support for 19 versions of Elasticsearch (1.5 to 7.10 versions), and visualization capabilities powered by OpenSearch Dashboards and Kibana (1.5 to 7.10 versions).
3. OpenSearch Dashboards: OpenSearch Dashboards, the successor to Kibana, is an open-source visualization tool designed to work with OpenSearch. Amazon OpenSearch Service provides an installation of OpenSearch Dashboards with every OpenSearch Service domain.

1.8.3 CU Boulder Initial Domain

OpenSearch Dashboards URL

Username and Password in Keepass

Domain endpoint

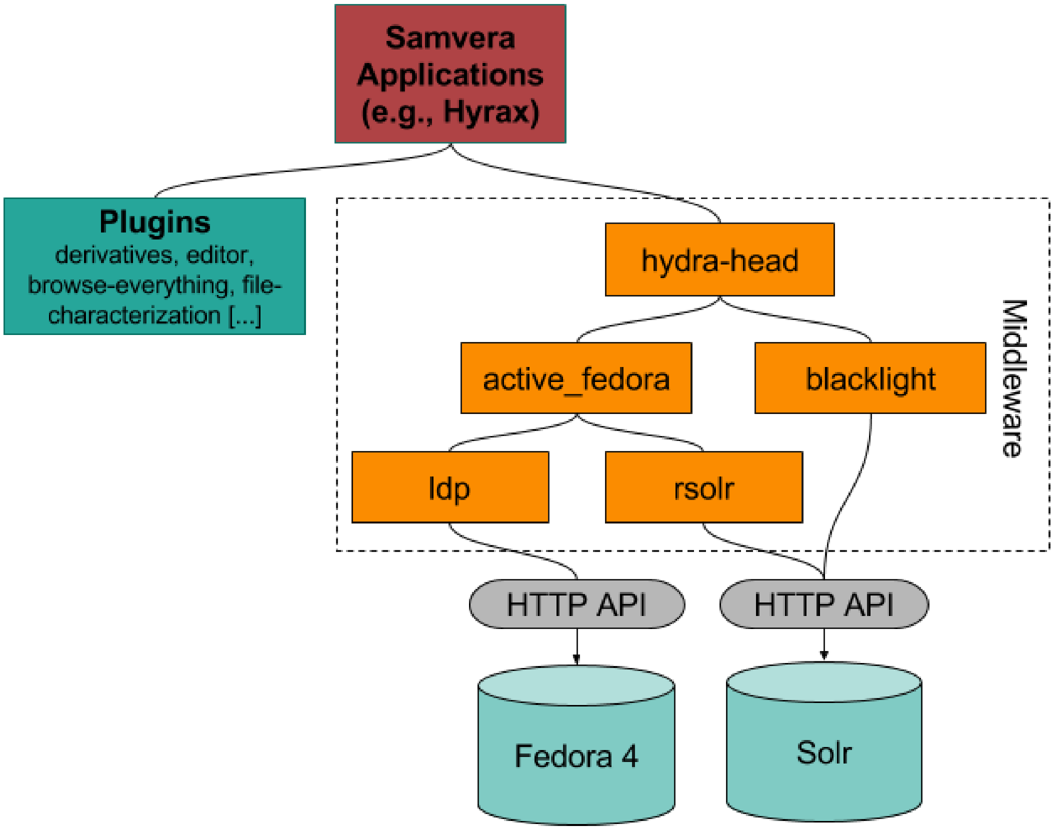
Issues

1. Multiline logs are split into single line logs. Therefore, we need to configure the parser for multiline logs within Fluent Bit.

1.9 CU Scholar

CU Scholar is the University of Colorado Libraries Institutional Repository. This repository serves as a platform for preserving and providing public access to the research activities of members of the CU Boulder community. The repository is using Samvera an open source repository framework.

1.9.1 Samvera Technical Stack


Samvera Applications (e.g., Hyrax)
Plugins
derivatives, editor, browse-everything, file-characterization [...]
hydra-head
active_fedora
blacklight
ldp
rsolr
Middleware
HTTP API
HTTP API
Fedora 4
Solr

1.9.2 Infrastructure

CU Scholar utilizes the CU Library *infrastructure*.

1.9.3 CU Scholar Components

1. Hyrax https://github.com/culibraries/ir-scholar
2. Ruby gems https://github.com/culibraries/ir-scholar/blob/master/Gemfile
3. Fedora 4.7 https://duraspace.org/fedora/
    - Fedora features
    - Metadata stored within Mysql Relational Database - AWS RDS
4. Solr 6.x https://solr.apache.org/

Metadata

CU Scholar metadata is stored within the Flexible Extensible Digital Object Repository Architecture(fedora). The metadata utilizes the Mysql AWS RDS database. Fedora logs all changes and stores metadata changes within the database. *Backup Policy* (CTS 4)

Data Files

CU Scholar data files are stored within the Flexibile Extensible Digital Object Repository Architecture(fedora). The production data files are stored within AWS S3 object storage service. *Backup Policy* (CTS 3: offsite copy to CU Boulder PetaLibrary(implementation phase))

Data File Checks

1. All files uploaded undergoes virus scan.(CTS 4)
2. Fixity checksums are performed at a regular interval (Quarterly). (CTS 3)

1.10 COUNTER

COUNTER stands for Counting Online Usage of NeTworked Electronic Resources. It is both a standard and the name of the governing body responsible for publishing the related code of practice (CoP). The governing body represents a collaborative effort of publishers and librarians whose collective goal is to develop and maintain the standard (CoP) for counting the use of electronic resources in library environments.

This document refers to the online tool developed by Libraries IT that simplifies the aggregation and reporting of electronic resource usage data for University Libraries.

1.10.1 Overview of Loading Process

The following describes the steps to load COUNTER data from Excel spreadsheets. These spreadsheets, or reports, are downloaded from the various platform sites by e-resources staff and are stored on the Q: drive (typically Q:\SharedDocs\Usage Stats) in year-specific folders. The product development team is notified by email that new reports are available for processing and importing into the COUNTER database.

Workflow steps:

1. Copy new reports to remote server.
2. Run preprocessing/renaming script.
3. Replicate production database on staging.
4. Run loading script.
5. Restore production database from staging.
6. Archive reports to AWS S3.

Each of these steps will be described in further detail later.

1.10.2 Staging Infrastructure

The COUNTER staging infrastructure consists of an EC2 instance (counter-staging) with MySQL 5.7 installed. This approach removes the loading workload to AWS from your local desktop/laptop. The staging server also facilitates access to both the test and production databases in RDS. Details of the instance can be found in the AWS console.

Loading scripts and associated modules can be copied to the staging server by cloning the Github repo (assuming you are starting at the home directory):

```
$ git clone https://github.com/culibraries/counter-data-loader.git
```

After cloning the repo, you will need to copy the `config.py` file to the /counter-data-loader/dataloader directory to enable a connection to the local MySQL database. The config file is available in KeePass in the MySQL folder.

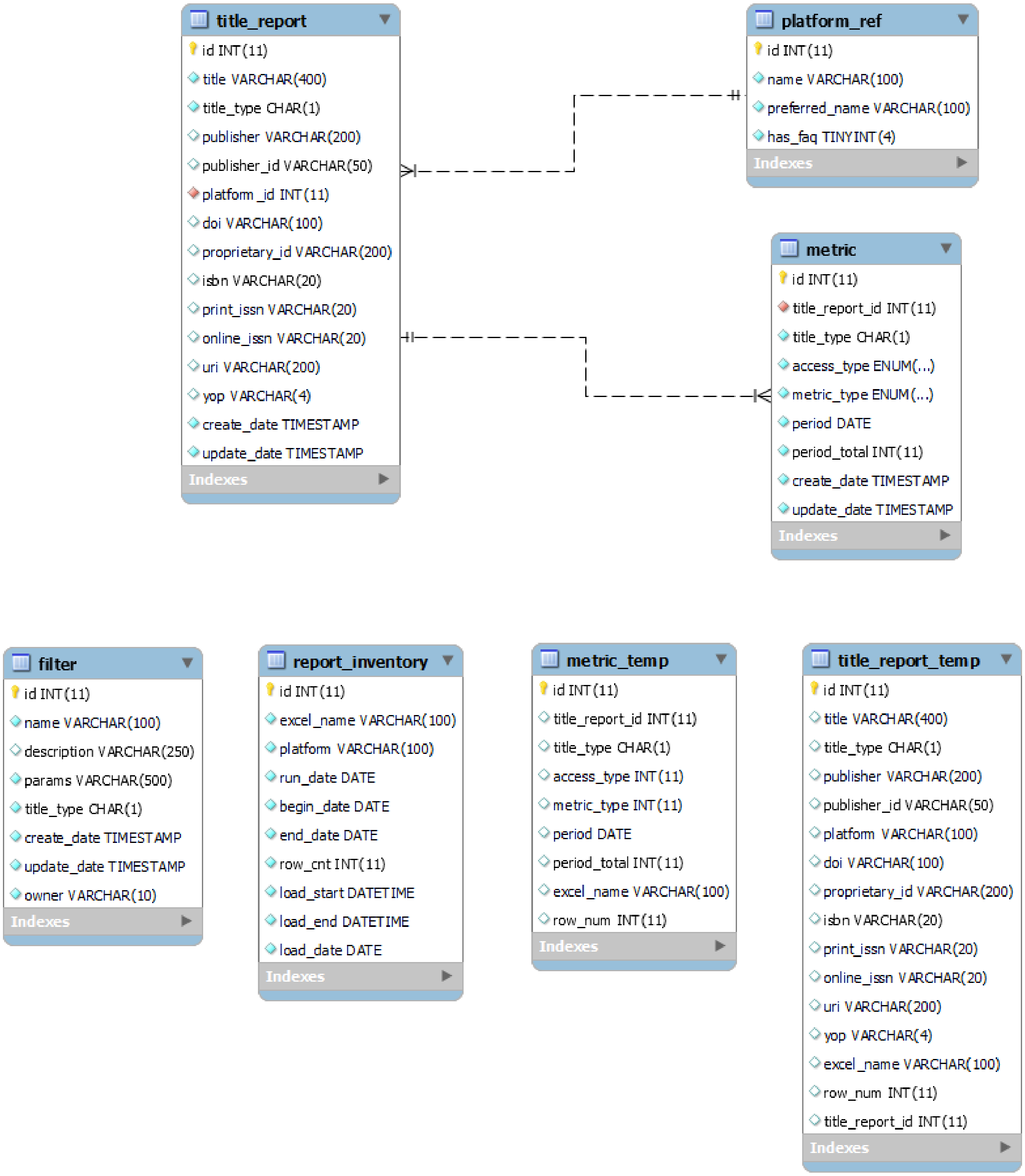
All data (including the MySQL database) is stored on an attached volume (/dev/sdf) currently sized at 50 GiB.

In addition to MySQL, the staging server requires the following software components:

- Python 3.x
- openpyxl 3.0.9
- mysql-connector-python 8.0.27
- boto3 1.19.7
- botocore 1.22.7

Versions are minimum requirements. Updated modules are acceptable.

1.10.3 Database Schema



1.10.4 Details of the Loading Process

Copy New Reports to Remote Server

Copy all files to be processed from the Q: drive to the remote server. The working directory for all source files is /data/counter.

Run Preprocessing/Renaming Script

Run the following command:

```
python3 preprocess-source-files.py <report directory>
```

This script will rename all files in the specified working directory to a common format. Refer to the comments in the code for a description of the naming convention.

If errors are raised, they will be recorded in an error log.

Replicate Production Database on Staging

The starting point for loading new COUNTER reports is the current production database. To replicate the production database on staging, run the following commands:

```
mysqldump --databases counter5 -h cudbcluster.cluster-cn8ntk7wk5rt.us-west-2.rds.
↪amazonaws.com -u dbmuser -p --add-drop-database -r /data/backups/20220329-counter-prod.
↪sql
mysql -u dbmuser -p < /data/backups/20220329-counter-prod.sql
```

When prompted, enter the password for dbmuser (available in KeePass). Be patient as the dump and load can take a bit of time depending on the size of the production database. While the dump is fairly quick (~30-45s), the load can take upwards of 8-10 minutes.

NOTE: Use the current date-time stamp as the file prefix.

To improve loading performance, drop all indexes:

```
$ mysql counter5 -u dbmuser -p < sql/drop-indexes.sql
```

At this point, the staging database is ready for loading the new files.

Run Loading Script

The loading process is a multistep process:

- Read the title and usage data in the source Excel spreadsheet.
- Generate CSV files from the spreadsheet representing title information and corresponding metrics.
- Import CSV files into temporary tables.
- Do inserts/updates in title and metric tables.
- Log the spreadsheet as processed.

This is an iterative process that is performed for every spreadsheet to the loaded.

The entire sequence of steps as outlined above are initiated and executed from a single “controller” file (loader.py). The process is started by entering the following command:

```
python3 loader.py <report directory> <year>
```

The report directory parameter is the location of the prepared Excel files. The year parameter is the 4-digit year that corresponds to the usage data, e.g., for a report containing usage data for 2021, this parameter value would be “2021” (without the quotes).

Refer to the source code comments for further details.

Restore Database to Test/Production

Once all spreadsheets have been loaded, the database on the staging server can be restored to the test RDS cluster for acceptance testing:

```
mysqldump --databases counter5 -u dbmuser -p --add-drop-database -r /data/backups/
↪20220329-counter-staging.sql
mysql -h test-dbcluster.cluster-cn8ntk7wk5rt.us-west-2.rds.amazonaws.com -u dbmuser -p <␣
↪/data/backups/20220329-counter-staging.sql
```

Next recreate the indexes in the test environment:

```
mysql counter5 -h test-dbcluster.cluster-cn8ntk7wk5rt.us-west-2.rds.amazonaws.com -u␣
↪dbmuser -p < sql/create-indexes.sql
```

With the test database restored, the designated product team can begin acceptance testing. For this step, it is recommended that a handful of spreadsheets be compared to the data returned from the UI. On completion of testing, the updated database can be restored to the production environment:

```
mysql -h cudbcluster.cluster-cn8ntk7wk5rt.us-west-2.rds.amazonaws.com -u dbmuser -p < /
↪data/backups/20220329-counter-staging.sql
mysql -h cudbcluster.cluster-cn8ntk7wk5rt.us-west-2.rds.amazonaws.com -u dbmuser -p <␣
↪sql/create-indexes.sql
```

Archive Reports to AWS S3

The last step in the process is to archive all of the processed spreadsheets by moving them to AWS S3. Do this by running the following command:

```
aws s3 mv /data/counter/ s3://cubl-backup/counter-reports/ --recursive --storage-class␣
↪ONEZONE_IA
```

1.10.5 Other Considerations

Running Loading Script in Screen Mode

It is recommended that the loading script be run in a Linux `screen` session. Using this approach will enable the script to run in the background while disconnected from the remote host.

To start a screen session, just type `screen` at the command prompt. This will open a new screen session. From this point forward, enter commands as you normally would. To return to the default terminal window, enter `ctrl+a d` to detach from the screen session. The program running in the screen session will continue to run after you detach from the session.

To resume the screen session, enter `screen -r` at the command prompt.

Further information about the Linux screen command is available at How To Use Linux Screen.

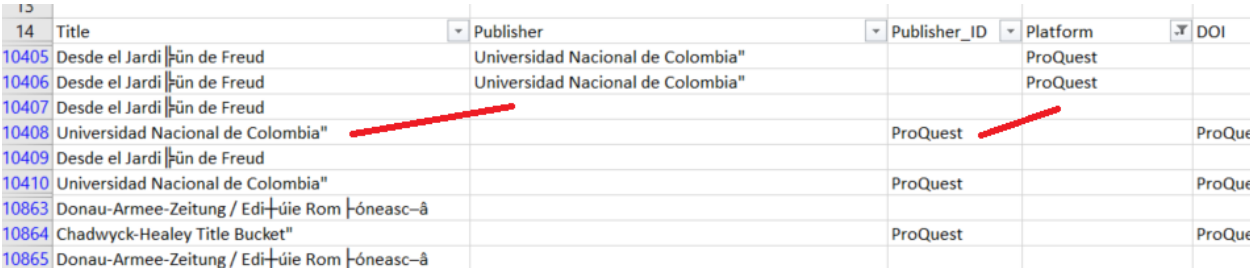
When Errors Occur During Loading

The loading process will raise an error (and log it) if the source spreadsheet cannot be loaded. Errors typically occur when the spreadsheet does not adhere to the COUNTER specification. For example, sometimes there will be a blank row at the top of the spreadsheet. Other formatting issues may also cause errors and two common problems are:

- A platform name is not referenced in the `platform_ref` table.
- ProQuest data consistently presents problems that preclude a clean load.

For the platform name issue, either add the missing platform data to the `platform_ref` table or update the spreadsheet (the platform column) to reflect a known reference value. Save the changes and reload the spreadsheet.

For the Proquest issue, same title entries spill over into two rows (see example below), a situation that will cause the load process to fail. In these cases (it's usually 12 or so rows), the simplest approach (though tedious) is to manually fix the offending rows, save the changes, and then reload the spreadsheet.



14	Title	Publisher	Publisher_ID	Platform	DOI
10405	Desde el Jardi╠ün de Freud	Universidad Nacional de Colombia"		ProQuest	
10406	Desde el Jardi╠ün de Freud	Universidad Nacional de Colombia"		ProQuest	
10407	Desde el Jardi╠ün de Freud				
10408	Universidad Nacional de Colombia"		ProQuest		ProQue
10409	Desde el Jardi╠ün de Freud				
10410	Universidad Nacional de Colombia"		ProQuest		ProQue
10863	Donau-Armee-Zeitung / Edi┼úie Rom├óneasc─â				
10864	Chadwyck-Healey Title Bucket"		ProQuest		ProQue
10865	Donau-Armee-Zeitung / Edi┼úie Rom├óneasc─â				

In these cases, the loading script will skip the spreadsheet and move on to the next one in the queue. An entry will also be written to a log file. After loading has finished, these Excel files should be examined for any obvious formatting errors and, if found, these can be rectified and the loading script rerun. If errors persist, let the Product Owner know.

Updating the platform_ref Table

If a new platform needs to be added to the `platform_ref` table, enter the following command in the MySQL environment:

```
mysql> INSERT INTO platform_ref VALUES (id, name, preferred_name, has_faq);
```

where

- id = the next id in the sequence (do a select max(id) to find the max)
- name = the name contained in the spreadsheet that needs to be reference

- preferred_name = the common or preferred name for the platform (consult with the PO as needed)
- has_faq = is always 0 (reserved for future use)

Using a Virtual Environment

TBD

1.11 Room Reservation Tablets

1.11.1 Overview

We were tasked to deploy a study room reservation system at the Engineering, Math and Physics Library (Gemmill) to enable patrons to check room availability and make reservations at each designated study room location. This project is intended to enhance the user experience of reserving study rooms by providing the capability at the point of need.

This project was also intended to serve a prototype to inform a more comprehensive and common solution to meet the needs of all branches of University Libraries.

The Libraries also produced a video about the project

1.11.2 System Specification

Application Features

- Check room (current or other rooms in the same locations) availability for current and future dates.
- Reserve room using BuffOne card or email.
- Send a reservation confirmation and cancel a reservation via email (Handled by LibCal)

Hardware Specification

- Tablet from Mimo Monitors Company
  - Model is `Mimo Adapt-IQV 10.1" Digital Signage Tablet Android 6.0 - RK3288 Processor MCT-10HPQ`
  - Exact model
  - We do not need PoE (Power over Ethernet)
  - Support via techsupport@mimomonitors.com
- Magnetic Swipe Card : IDTech Company
  - Suggested by the Buff OneCard Office
  - Contact the Buff OneCard office for support

1.11.3 Starting a tablet for the first time

Preparation

Login to Libcal

- Get `Location ID` from LibCal.
  - Admin => Equipment & Space => Looking at the column `ID` for `location ID`
- Get `Hours View ID` from LibCal.
  - Admin => Hours => Widgets => JSON-LD Data => Select `Library` => Click `Generate Code/Previews` => Looking at the `lid number` in `Embed Code` section
- Get `Space ID` from LibCal.
  - Admin => Equipment & Space => In column `Spaces` => Click the space number (Ex: 3 spaces, 5 spaces) => Looking at the column `ID` for `space ID`

Software Setup

Only users who have administrative permission is able to follow the steps below to setup the software for the application.

1. Login to room booking admin
   - Production https://libapps.colorado.edu/room-booking-admin
   - Test https://test-libapps.colorado.edu/room-booking-admin
2. Click "New Device".
3. Click "Generate" to generate new id for the device.
4. Fill out all of the information including:
   - Name
   - Note (optional)
   - `Location ID`
   - `Hours View ID`
   - `Space ID`
5. Click "Submit"

Warning: Note : The users are not able to Activate/Deactivate/Delete the device at their computer, they must activate it at the device where it is to be installed.

Wireless Setup

1. Connect the tablet to `UCB Wireless` WiFi. The tablet should be able to connect to the network but not authenticate onto the WiFi.
2. Find the device's WiFi MAC address
    1. Unlock the tablet
        1. Touch/Hold the top-right corner of the tablet for 10+ seconds then a password dialog will pop-up
        2. Enter the pin (In Keepass: `Study Room Application/Hardware Tablet Pin`).
    2. Use the `Home/House` icon in the Mimo app to get to the "Desktop"
    3. Slide open the top menu from the upper right corner and click the `gear` icon to access the settings
    4. Scroll to the bottom and click `About tablet`
    5. Click `Status`
    6. Find `Wi-Fi MAC address`
3. Leave the tablet connected to the network
4. Open a ServiceNow ticket with Dedicated Desktop Support ticket (DDS)

```
Please add the following MAC addresses to SafeConnect with the user `libnotify@colorado.edu`. These are for android tablets that we are deploying to support an in-place room reservation system in the Libraries.

ma:c1
ma:c2
ma:c3

Thank you!
CTA
```

5. Wait 20 minutes after DDS adds the devices to SafeConnect for the configuration to propagate.
6. "Forget" the existing "UCB Wireless" configuration and reselect "UCB Wireless".

The device should connect and you should be able to access webpages via a browser.

Device Setup

In the Android Settings menus:

1. Time/Date Setup
    1. Settings > Date & Time
    2. Turn off "Automatic time zone"
    3. "Select time zone" to "Denver GMT-07:00"
    4. Turn on "Automatic time zone"
2. Text-correction Setup
    1. Settings > Language & Input > Android Keyboard (AOSP) > Text correction
    2. Switch all off except "Block offensive words"

Application Setup

1. Unlock the tablet
   1. Touch/Hold the top-right corner of the tablet for 10+ seconds then a password dialog will pop-up
   2. Enter the pin (In Keepass: `Study Room Application/Hardware Tablet Pin`).
2. Open "MLock" application
3. Press "Cookie" then turn on Cookie State
   - Pink is ON, Gray is OFF
   - Press "< Cookie Setting" to go back.
4. Press "Default App" and turn on "Auto Start"
   - Pink is ON, Gray is OFF
   - Press "< Default App" to go back
5. Press "Playback Setting" > "Web URL" and change the url
   - Production https://libapps.colorado.edu/room-booking-admin
   - Test https://test-libapps.colorado.edu/room-booking-admin
   - Press the "go back" icon on the top-left
6. Login using the admin user credential in (Software setup) section above, then activate the device that you want to set up in the list by turn on the toggle button.
   - Pink is Activated, Gray is Deactivated
7. Plug Magnetic Swipe Card device to the USB port

Warning: In production, only undergraduate students can reserve the rooms. Faculty, Staff, or graduate student will receive an error message. In Test, all staff can reserve the room.

1.11.4 Code Repositories

- Room Booking application (Main Interface)
- Room Booking admin (Admin Interface
- API (Cybercomm)

Admin User

In order to access to Room Booking admin application, you need to create a local user in Cybercomm with a password and add group `study-room-admin` to that user as the permission. Store the username and password in a safe place. You will need to use that to set up the tablet.

Basically, The tablet will interact with API (Cybercomm) using the Token from this admin user, which is stored in localStorage in each device. In Application Setup -> Step 5 (in the section above), The reason you need to go to room-booking-admin is to get the Token and store it in each tablet for future API call.

Room Booking Admin

This application is to manage information in each tablet. You can add/delete/modify the information of each room which corresponding to each tablet in front of the room. However, you are not able to activate unless you access to this application in a tablet.

Do not forget to Generate Unique ID for each tablet.

Room Booking

Local Development

1. Create a testing room in Libcal and get all of information including: location_id, space_id, hours_view_id.
2. Get Token string from admin user above, and libcal_token (It will be expired after 60 minutes.)
3. Store all of these variable in LocalStorage of the web browser.
4. Go to auth-guard.service.ts and remove line 26,27. (Don't forget to bring it back when deploy to TEST or PRODUCTION.

You can plugging the Magnetic Swipe Card into your computer via USB port and use it as a testing device.

API (Cybercom)

- Get libcal_token from Libcal using LIBCAL_CLIENT_ID and LIBCAL_CLIENT_SECRET (in Libops).
- Get information after swipe action using sierra api.

Logs

All the log will be uploaded to S3 at cubl-log/room-booking as CSV format around midnight everyday. Terms explanation in the log:

- refresh libcal_token: Since the token only good for 60 minutes, so it will need to recreated when it is expired.
- app starting…: this happen when the application refresh the page.
- card - PType - Room (email and time slots): when these 3 messages go together. It means someone successfully book the room.
- the library is closed: It means the tablet is not display for booking anymore since the library is closed. Please looking the github code for more log information.

1.11.5 Support Documentation For Circ Desk Staff

Note: Bring a USB keyboard with you to debug. Disconnect the USB card reader in order to connect the keyboard.

1. What should I do if the message "SYSTEM ERROR" appears?
   - If a student reports to Circ Desk about the error above:
     1. Go to the tablet and press "Press here to reload". If the problem isn't fixed, continue.
     2. Reboot the tablet.

- If the error persists, manually reserve the room for that student via libcal, and submit a ticket to University Libraries - Core Tech & Apps.

2. What should I do if there is an internet connection issue?
    - If internet connection is active, submit a ticket to University Libraries - Core Tech & Apps
    - If there is no internet connectivity in the building, contact OIT for a network connection issue.
3. How can I reboot the tablet?
    - There is a power switch underneath the tablet on the left side.
    - You can turn the tablet off and on to reboot it.
4. What should I do if the screen turns black?
    - Touch the screen to see if its back on.
    - Check to see if there is any power outage.
    - Reboot the tablet.
    - If the issue is not resolved, submit a ticket to University Libraries - Core Tech & Apps.
5. Why student successfully reserve a room through Libcal but it is not showing on the tablet?
    - It can take 45 seconds to 60 seconds for the tablet pull information from Libcal. If the student has an email confirmation from Libcal it should update.
6. For any other issue, begin with rebooting the tablet.
    - As a workaround, manually reserve the room for the student via Libcal
    - Submit a ticket to University Libraries - Core Tech & Apps.

1.12 Self-checkout

Procedure to update Sierra self-checkout machines

1. Request OIT move machines from limited access OU via ServiceNow ticket.
2. Update Sierra client through Software Center or download from https://libraries.colorado.edu/web/installers/
3. Get information from old shortcut
    1. Type `Win+r` to open a cmd prompt
    2. Type `shell:common startup` to open the startup folder.
4. Open Sierra shortcut and copy “`program=milselfcheck username=xxxnor[1,2,3,4] password==xxxnor[1,2,3,4]`
5. Edit new desktop shortcut
    1. Add `"program=milselfcheck” username=xxxnor[1,2,3,4] password==xxxnor[1,2,3,4]` to the shortcut target.
    2. Click Apply
6. The shortcut can be tested at this point if desired, but you will need admin credentials for Sierra to close self-checkout window)
7. Delete the old shortcut from the startup folder and then copy the new desktop shortcut to the startup folder. (Requires super-user credentials requested from OIT via ServiceNow)

8. Restart the machine and confirm self-check program auto-starts
9. Notify OIT to move machines back to limited access OU.

1.13 BitCurator

There is a desktop in E1B25A (the Digital Archives Lab) running a distro based on an LTS release of Ubuntu in E1B25A. This is for BitCurator, a tool used for digital forensics.

Walker Sampson is the primary contact for the machine. It is primarily used Monday 9:00-11:00, Tuesday 2:00-4:00, Wednesday 9:00-11:00, Thursday 2:00-4:00, and Friday 2:30-4:30.

If updates are required you can use the Software Update application in Ubuntu or using `apt` via the command line.

1.14 Cert Manager

All CTA Infrastructure & Applications certificates are issued with the cert-manager application in the production kubernetes cluster.

1.14.1 DNS Registration

1. CU Boulder OIT has CNAMEs that are directed to our AWS Application Load Balancer
   - cubl-load-balancer (arn:aws:elasticloadbalancing:us-west-2:735677975035:loadbalancer/app/cubl-load-balancer/3039b8466406df2c)
   - If deleted will have to register all domains with CU Boulder OIT and point all CNAMEs to new load balancer

1.14.2 Network Configuration

Certificates are held within a secret on production cluster. The following network configuration is important to keep Target groups up to date with current Worker nodes in the cluster. If target group does not have a worker nodes the certificate will fail.

ALB Listeners

1. Http: 80
   - RULEs Path is /.well-known/acme-challenge/* ====> k8s-nodes Otherwise redirect Https(443)
2. Https: 443
   - Rules are setup for each domain. Test domains are usually locked to campus and VPN IPs.

Target Groups

1. http 80 => k8s-nodes
2. https 443 => k8s-nodes-https

1.14.3 Lets Encrypt Certificate and Issuer

- Github repository for certificates and issuer yaml file. culibraries/cert-manager
- Daily Cronjob runs to update certificate in AWS Certificate Manager. culibraries/k8s-cronjob-tasks

1.14.4 Example Certificate Request Failure

Recently, DNS (folio.colorado.edu) was transferred to the FOLIO team. As a result, my certificate request failed because the folio.colorado.edu domain name was routed to a different AWS Load Balancer.

Corrective Actions

1. If you think your certificate is correct, check certificate request.

```
kubectl get certificaterequest -n cert-manager
NAME                              READY    AGE
cubl-lib-colorado-edu-99lwk       True     5d16h
cubl-lib-colorado-edu-9d98s       False    5d16h
...

kubectl describe certificaterequest cubl-lib-colorado-edu-9d98s -n cert-manager
```

2. Update Certificate and Delete Failed Request Update Certificate

```
kubectl delete certificaterequest cubl-lib-colorado-edu-9d98s -n cert-manager
kubectl apply -f certs/cubl-lib-colorado-edu-main.yaml -n crontab
```

1.15 Patron Fines

The patron fines workflow provides a means for Sierra generated fines file to be accessed by library personnel.

1.15.1 Overview

Contact: Mee Chang

1. Library Personnel submit a request to generate export
2. Sierra exports file to FTP server
3. Data file is stored in AWS EFS filesystem
4. Cronjob runs every 2 mins (mounts EFS filesystem)
5. Job runs transforms
6. Job uploads file to S3(cubl-patron-fines)

7. Library personnel access through Cloud-Browser

1.15.2 Transform

Transform repository: https://github.com/culibraries/patron-fines

1.15.3 Kubernetes Cronjob

Dockerfile and deploy YAML: https://github.com/culibraries/k8s-cronjob-tasks/tree/main/patron-fine

1.16 ETD Loader Process

It is stored inside IR project on GitHub

1.16.1 Folder Structure

All the scripts running are stored inside IR container. However, all of the ETD files are store at: `/efs/prod/proquest/` or `/efs/test/proquest`. Access scholar-worker image with Kubectl. Then go to `/efs/prod/proquest` or `/efs/test/proquest` to see those files.

- `.zip`: new zip files, have not processed.
- `.zip.proccessed`: files have been unzipped and processed
- `logs/` : folder to store log files
- `proccessing_folder/`: folder to store files after unzip (.zip files) to process.
- `rejected/`: folder to store rejected files(.zip). If there is an error happen during the process. The script will move .zip error file to this folder.
- `unaccepted/`: folder to store unaccepted files(.zip). If the ETD item is not allow to load to IR. It will be moved to this folder.

1.16.2 Execute Script

1. scholar-worker

   ```
   kubectl exec -it scholar-worker-78f7c8646-mztqv -n scholar -- bash
   ```

2. At /app run command below:

   - In TEST:

     ```
     python3 etd-loader/main.py /efs/test/proquest/ /efs/test/proquest/processing_
     ↪folder/ number_item
     ```

   - In PRODUCTION:

     ```
     python3 etd-loader/main.py /efs/prod/proquest/ /efs/prod/proquest/processing_
     ↪folder/ number_item
     ```

   - `number_item`: is number of zip file that you want to process. You can put any number as long as it is easy to keep track. Less than 20 is the suggestion.
   - `etd-loader/main.py`: access to script.
   - `/efs/test/proquest/`: where the ETD .zip files store.
   - `/efs/prod/proquest/processing_folder/`: where ETD .zip files extract.

3. Go to scholar website to make sure its loaded including any upload files. Check logs and folders to see if is there any rejected files, unaccepted files or any error might happen.

1.17 PetaLibrary S3 Glacier

Documentation for the Globus Online transfer of Petalibrary data to S3 Bucket.

1.17.1 Configuration

1. Discussion with Research Computing (Jason Armbruster) to set up Globus Online endpoint (cubl-petalibrary-archive).
2. RC set up a globus endpoint "S3 prototype CU Boulder Libraries"
3. When you open that collection you're going to get prompted to authenticate with Boulder Identikey first, then once that's successful, you'll have to also authenticate with an AWS key/secret pair for a user who has access to the S3 bucket.
4. User needs to have UC Boulder Research Computing Account with access to "dulockgrp" group. The group will give access to "libdigicoll"
5. Path /pl/archive/libdigicoll/ to access UC Boulder Library data

1.17.2 Trial S3 Data Transfer

1. CTA (Vida) is currently awaiting UC Boulder Research Computing account with access to Library data.
2. Data Trial: Contact Michael Dulock

```
/pl/archive/libdigicoll/libimage-bulkMove/
(2TB)

/pl/archive/libdigicoll/libstore-bulkMove/
(4.4TB)

/pl/archive/libdigicoll/libberet-bulkMove/RFS/DigitalImages/
(Important)

/pl/archive/libdigicoll/libberet-bulkMove/RFS/
(23TB)
```

1.17.3 TODO

1. If trial successful add new globus endpoints
2. The CU Scholar archive is currently being manually moved. This would cut out the middle step and provide direct access from S3 to PetaLibrary.
3. Transfer S3 bucket(cubl-ir-fcrepo) ==> /pl/archive/libdigicoll/dataSets/cu_scholar/cubl-ir-fcrepo
4. The above actions will allow for 3 copies with one copy in a different geolocation.
5. This is part of the Core Trust Seal actions needed for CU Scholar.
6. AWS Lambda to move IR files on demand

1.17.4 Manual backup to Petalibrary

1. Sync S3 Bucket to local drive

   ```
   cd { data download directory }
   aws s3 sync s3://cubl-ir-fcrepo .
   ```

2. Install Globus Connect Personal
3. Create Endpoint on System
4. Use the Web Interface to start a transfer from Laptop Endpoint to Petalibrary
   - Laptop endpoint where the AWS sync happened
   - Petalibrary Endpoint `/pl/archive/libdigicoll/dataSets/cu_scholar/`
   - select `cubl-ir-fcrepo`

1.18 Kubernetes Cronjob Tasks

This document describes the individual Cronjob Tasks associated with our current kubernetes infrastructure. The backup cronjobs to S3 bucket(cubl-backup) have a 30 day lifecycle rule. Github Repository

1.18.1 Active Tasks

1. alb-targetgroup-update This task updates the instances within the target group. The cta-test cluster is 100% spot instances, and the task updates the target group with new spot instances. When cta-prod is moved to AWS EKS, the cluster could be 100% spot instances. This task would need modification to include the cta-prod cluster.
2. cert-aws-upload The CTA Infrastructure and Application section uses a cert-manager to produce HTTPS (SSL/TLS) certificates. This nightly task uploads the production cluster certificate to the AWS Certificate Manager service.
3. cybercom-db-backup This cronjob task is for backing up MongoDB for the production cluster. Once backed up, file uploaded to S3 cubl-backup/cybercom/mongo/
4. deployment-restart Cronjob is used to restart multiple deployments. This action is no longer needed with the update of nginx resolver.
5. ir-reports Cronjob for creating IR Reports for metadata. Library personnel use the Cloud-Browser to access reports. ir-exportq celery queue used to run report.

6. ir-s3-sync Cronjob used to sync FCRepo S3 Bucket.
7. patron-fine Cronjob to check for new parton fine files. Once a new file the job will transform and upload to S3(cubl-patron-fines).
8. solr Solr backup of IR solr index and cloud configuration. Backup uploaded to S3 cubl-backup/solr bucket.
9. solr-json Cronjob takes a dump of all documents within the IR solr index. The export is in JSON.
10. survey Cronjob that checks the mongo collection that holds the survey schedule. If survey is scheduled, ENV variable updated and a restart of deployment.

1.18.2 Inactive Tasks

1. folio-db-backup Honeysuckle version backup of postgres in cluster DB. This task is retired
2. gatecount Retired task that was an example task that collected gatecount numbers
3. wekan Retired backup of Wekan Mongo DB instance. This was used for Libkan(Retired project boards)

1.19 Environment Data Logger

This documentation is for the environmental monitors. The operational manuals are linked below with links to the Github repository.

1.20 Deprecated

This service was turned off on 7/26/2023 and is no longer needed. Pinnacle environmental monitors have been replaced with Conserv environmental monitors which have their own web portal and data logging service.

1.20.1 Documentation

- Pinnacle Data Loggers

1.20.2 Cronjob

Update Cronjobs with the correct paths. See data logger repository

1.20.3 Cloud Front Address

https://d3v45qbh6kyrnc.cloudfront.net/

1.21 Static Web Server

The static web server is configured behind CU Boulder federated SSO.

1.21.1 Configuration

1. Cybercom API handles the SAML Service Provider
2. cubl_static
3. Deployment yaml provided within repostitory
4. Nginx uses DNS Resolver within Kubernetes cluster. New cluster will need to check the IP of kube_dns.

```
cat /etc/resolv.conf
nameserver 10.43.0.10
search prod-cybercom.svc.cluster.local svc.cluster.local cluster.local
options ndots:5
```

5. Add nameserver ip to default config

1.21.2 Current Applications

1. Static Web
2. Print Purchase
3. LibBudget

1.22 IR CU Scholar Development

IR Development process and common problems with CU Scholar. Documents the current status and the Docs for development process.

Contact: Andrew Johnson

1.22.1 Local Development

The Samvera community has good documentation regarding local development and dependencies.

Samvera Docs

- Double Check: solr_wrapper and fcrepo_wrapper require `java 1.8`

Development Documentation

Configuration

- Build local Gems

```
bundle install
```

- Start local rails server

```
bin/rails hydra:server
```

- Create local sqlite3 DB

```
bin/rails db:migrate RAILS_ENV=development
```

- Create CU Scholar Admin User and Admin Sets

```
build/firstrun.sh
```

After this process the Solr and FCRepo is setup along with local admin user.

1.22.2 Groups and Permission

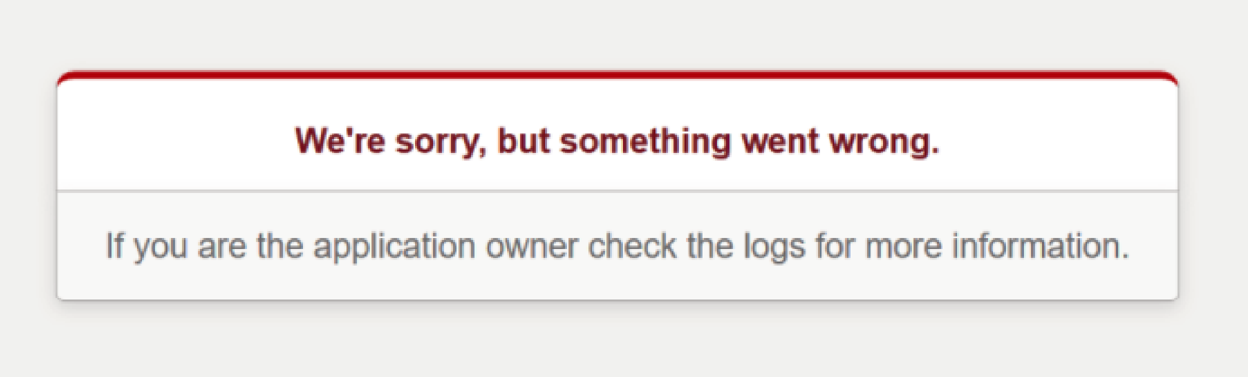
Admin user can update roles.

https://scholar.colorado.edu/roles

1.22.3 Common Errors

1.22.4 Permission errors with Solr

- User delete/upgrade version of main file. Updated file does not get the permissions set within solr.



We're sorry, but something went wrong.

If you are the application owner check the logs for more information.

- Solution: Add test file and mark individual file as private
- example

1.22.5 Encoding Error

- This error is difficult to track down. If an illegal character(non-UTF-8). The Fedora Commons Repository stores the characters, but on the Solr side will not store correctly. Data will be different on the front end compared to the data in the edit form.
- Solution: Use editor, which shows encoding. I use Microsoft Code and copy each item to the editor and view encoding. Once found, update the form item. The Abstract field is usually the culprit.

1.22.6 SOLR (Primary Concern)

The Solr instance needs to be even distributed cores for each node. Our current production cluster has two Cores with three nodes. I believe this is what is causing our problems. I have worked on the test cluster with multiple configurations. The original design was similar to ElastiSearch, which handles the multiple core configuration. I assumed that Solr Cloud handles this as well; I was wrong. The last configuration has only two replicas deployed with two cores on each node. It appears to work. I have not deployed to production.

Example Configuration Changes

1. Create Backup of current collection

```
curl "http://solr-svc:8983/solr/admin/collections?action=BACKUP&name=2022-03-14-
↪testSolrBackup&collection=hydra-prod5&location=/backup/test"
```

2. Delete Collection

```
curl "http://solr-svc:8983/solr/admin/collections?action=DELETE&name=hydra-prod5"
```

3. Perform configuration action

```
kubectl -n scholar scale statefulsets solr --replicas=2
```

4. Restore from backup

```
curl "http://solr-svc:8983/solr/admin/collections?action=RESTORE&name=2022-03-14-
↪testSolrBackup&collection=hydra-prod5&location=/backup/test"
```

1.23 GeoLibrary Development

CU GeoLibary is deploy from the GeoBlacklight open-source project.

Contact Phil White

1.23.1 Code Repositories

1. GeoLibrary https://github.com/culibraries/geo-geolibrary
2. GeoDataLoader https://github.com/culibraries/geo-data-loader
3. geoBlacklightq https://github.com/culibraries/geo-blacklightq
4. GeoServer:A docker container that runs GeoServer influenced by this docker recipe

1.23.2 Local Development

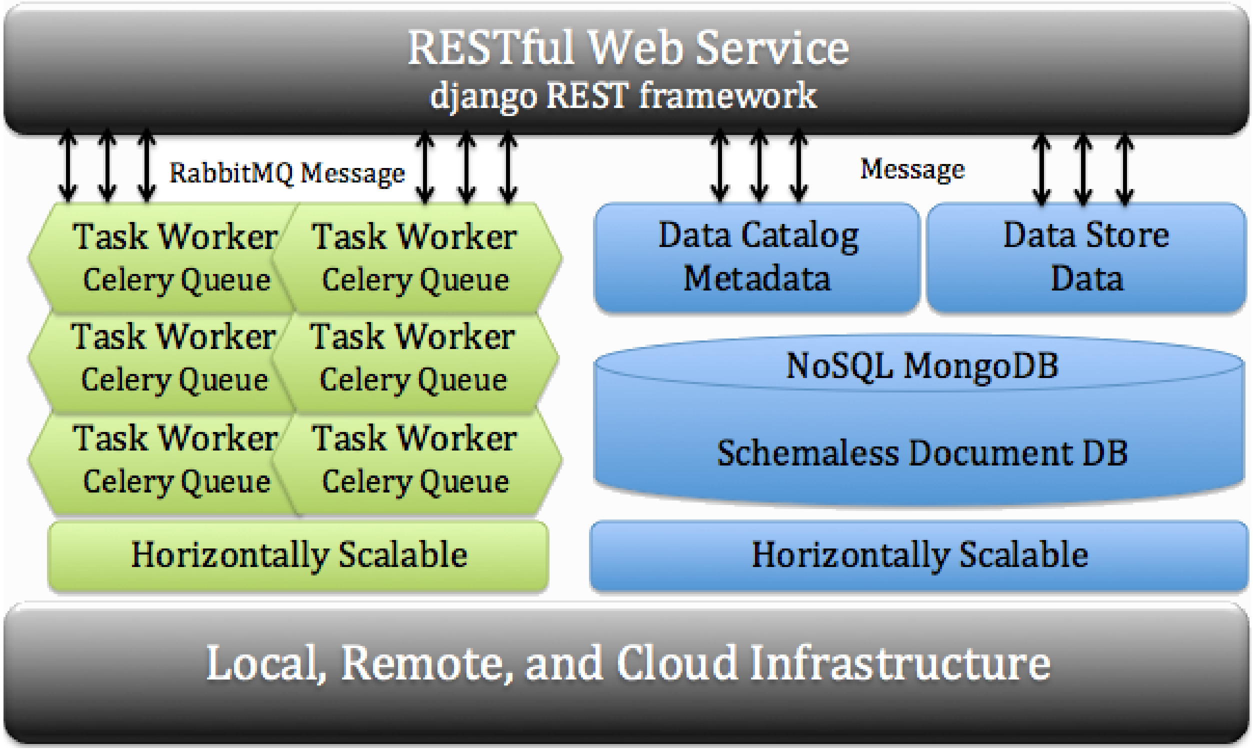
1. GeoBlacklight Guides
2. GeoBlacklight online Tutorial
3. With the dependency with GeoServer, Solr, and Data Loader, I have found it easier to build code and deploy it to the Test cluster. Dockerfile
4. Celery, GeoServer, and Data Loader application mount AWS EFS fileshare. This is the data store for geosever and for the application to load data into the Geoserver application.

1.24 Ansible Development

Docs for development process.

1.25 Cybercom API

Cybercomm API is based from open-source project. CU Boulder library modified the API using federated SSO and security groups merged from local and grouper groups.



1.25.1 Containers

1. API django application dockerfile
2. Celery dockerfile
3. Docker Hub: RabbitMQ - rabbitmq:3.6
4. Docker Hub: Mongo - mongo:4.2.10
5. Docker Hub: Memcache - memcached:latest

1.25.2 Configuration

Refer to cybercommons for system configuration documentation. This documentation assumes you are not working in kubernetes.

Changes with Kubernetes:

1. Secret(cybercom) contains all secrets
2. Encrypted communication through self signed certificates stored in Secret(cybercom)
3. Within container certs are mounted from secret and located `/ssl` directory.
4. Certificates are valid

```
cat /ssl/server/mongodb.pem | openssl x509 -noout -enddate
notAfter=Sep 10 19:12:02 2029 GMT
```

5. Federated SSO certificates are stored in Secret(cybercom)
6. SAML Service Provider

1.25.3 Catalog and Data Store

The Catalog and Data Store are using MongoDB for the backend. The API leverages the pymongo query language, including aggregation and distinct queries. Documentation

1.25.4 Applications API SSO Authentication

1. Authentication configuration within Nginx conf file
2. LibBudget Example

```
server {
    listen 80;
    server_name libapps.colorado.edu;
    resolver 10.43.0.10;
    index index.php index.html;
    auth_request /user;

    location / {
    root /usr/share/nginx/html/;
    autoindex on;
    }

    location = /user {
        internal;
        set $upstream_user https://libapps.colorado.edu/api/user/;
        proxy_pass $upstream_user?app=libbudget;

        proxy_read_timeout 3600;
        proxy_pass_request_body off;
        proxy_set_header Content-Length "";
        proxy_set_header X-Original-URI $request_uri;
        proxy_set_header X-Original-METHOD $request_method;
    }

    error_page 401 = @error401;
    location @error401 {
        set_escape_uri $request_uri_encoded $request_uri;
        set $saml_sso https://libapps.colorado.edu/api/api-saml/sso/saml;
        return 302 $saml_sso?next=$request_uri_encoded;

    }

    # redirect server error pages to the static page /50x.html
    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
        root /usr/share/nginx/html/;
    }

    location ~\.php$ {
        root /usr/share/nginx/html/;

        fastcgi_split_path_info ^(.+?\.php)(/.*)$;
```

(continues on next page)

(continued from previous page)

```
        if (!-f $document_root$fastcgi_script_name) {
        return 404;
        }
        fastcgi_param HTTP_PROXY "";

        fastcgi_pass libbudget-php-service:9000;
        fastcgi_index index.php;
        include fastcgi_params;
        fastcgi_read_timeout 300s;
        fastcgi_send_timeout 300s;
        fastcgi_connect_timeout 70s;

        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
    }
}
```

1.25.5 Possible Errors

1. Certificate Expiration: Will see logs with certificate expiration. Current certificates expiration `Sep 10 19:12:02 2029 GMT`
2. Upgrading dependencies: API <===> RabbitMQ <===> Celery(kombu)
3. Additionally, TSL arguments are changing on Mongo URI from ssl to tls
4. Mongo unable to connect to volume. Volume assigned to subnet. Spot instances occassional do not have capacity in specific subnet.
5. Celery Queue build missing requirement

1.25.6 Applications

Application	Auth	Django Apps	Celery	Mongo
LibBudget	Yes		emailCULibq	
Print Purchase	Yes		emailCULibq,ppodq	
Cloud Browser	Yes	cloud-browser-django-app	thumbnailq	
Room Booking (tablet only)	Yes	Room Booking		Yes
Room Booking Admin	Yes	Room Booking		Yes
Survey	No			Yes
Counter	Yes	counter-django-app	counterq	
ARK Server , info	Yes	ark-django-app		Yes
Static (NYTimes,thumbnails)	Yes			
GeoLibrary Data Loader	Yes		geo-blacklightq	Yes
IR Scholar Export Report	Yes		ir-exportq	
Email Service	Yes		emailCULibq	
Thumbnail Creation	Yes		thumbnailq	

1.25.7 Inactive Applications

1. Information Survey
2. Gate Count Celery Queue

1.26 CU Library Read the Docs

ReadtheDocs is CU Boulder Libraries Core Tech & Apps approved method for documentation.

1. Github Markdown Guides
2. Read the Docs Documentation

1.26.1 Requirements

1. Python 3.3 or greater

1.26.2 Installation

1. Clone Repository

   ```
   git clone git@github.com:culibraries/documentation.git

   or

   git clone https://github.com/culibraries/documentation.git
   ```

2. Create Virtual Environment

   NOTE: Win variations assume cmd.exe shell

   ```
   cd documentation
   python3 -m venv venv (Win: python -m venv <dir>)
   . venv/bin/activate (Win: venv\Scripts\activate.bat)
   pip install -r requirements.txt
   ```

3. Create HTML

   ```
   cd docs
   make html
   ```

4. New Terminal - Web server

   ```
   . venv/bin/activate
   cd docs/_build/html
   python -m http.server
   Serving HTTP on :: port 8000 (http://[::]:8000/) ...
   ```

5. Open Browser http://localhost:8000

1.26.3 Add new documentation

1. `git checkout -b new_docs`
2. Edit/Add documentation (Markdown)
3. `make html`
4. add new pages to toctree (index.rst)

1.26.4 Pull Request to main branch

CU Boulder Libraries' regular activity is to create a PR from the `release` branch with a code review. The `documentation` repository is slightly different. Perform a PR from the feature branch to the `main` branch. Add a code review before merge to main.

1.26.5 View Build Process on ReadtheDocs

1. Merge to main required before ReadtheDocs build process will start.
2. ReadtheDocs View builds
3. After successful build: https://cu-boulder-libraries.readthedocs.io/en/latest/