Analytics Stack Improvements

Discovery

***Authors**: Tim Krones <tim@opencraft.com>, Braden MacDonald <braden@opencraft.com>*

Approach

edX Solutions has a client interested in Insights/Analytics and is looking at making some contributions to edX Insights to make it more useful for that client, future Solutions and white label clients, as well as the community at large.

This document presents a list of pain points from multiple stakeholders (Analytics team, OpenCraft, the edX community), as well as suggestions for addressing them and improving the the Analytics pipeline as a whole. By bringing this information together we hope to be able to jointly agree on a prioritized list of things to work on, and to get feedback on possible solutions for individual tasks.

Timeframe

We need to get the pipeline for the solutions client running in/around April.

Parties involved

Analytics team, edx-solutions team, OpenCraft

Goals

- Make analytics devstack and production easier to setup and maintain
- Reduce costs for general maintenance and setup
- Reduce barriers to contributions such as implementation of new features

Pain points & suggestions for improvements

From the Analytics Team

Source: Analytics Operations (Google doc), OLIVE-1

1. Maintaining jobs on the scheduler is a highly manual and rather difficult process
    a. At a minimum it would be great if we could check the jobs in somewhere
        i. Additional note from Gabe's comment on OLIVE-1: “Also, if/when we decide to tackle the automation of the scheduler, definitely get in touch with Ben Patterson about committing jenkins jobs to source control. I think we have standardized on the Job DSL plugin to handle this type of thing.”
    b. Our jenkins box is rather out of date, in part, due to its fragility
    c. Ad-hoc backups are unreliable
    d. How do we draw the line between config-file defined parameters and command-line defined parameters? Should we formalize our de-facto standard here?
    e. Can we inherit from other jobs? or build dependency chains easily?
        i. For example: deploy cluster, run task, teardown cluster
2. Jobs fail periodically, we should identify all common causes and resolve them
    a. It will be difficult for a non-analytics team member to diagnose and resolve some of the issues
    b. Many can be resolved by simply restarting the job that failed - typically S3 timeouts
3. Schema changes are very painful (see the process above)
    a. Improvements are needed here, some might be “free” or very low cost if the above work is completed first. In particular “job templates” or “job inheritance”.
4. The AWS configuration is rather complex and difficult to replicate
    a. We should consider developing terraform scripts that completely specify the components and their relationships
    b. We could also use Ansible or cloud formation to provision all of the AWS resources.
    c. Note that this tool would just create all of the resources in AWS, we would continue to use ansible for the configuration management once the resources were created (for example: installing software, writing config files etc).
5. The pipeline should be installed like every other component in the edX infrastructure. Currently it is not.
    a. This branch implements this (partially): https://github.com/edx/configuration/compare/hack2015/gabe/analytics-hadoop
    b. As part of this, we will need tests to fail in edx/configuration if/when people commit breaking changes to it. This applies both to changes that would break the analytics stack and changes that would break edx-platform deployment.
6. We should seriously consider deprecating edx-analytics-configuration and just merging it into the edx/configuration monolith.
    a. Some roles are duplicated between these code bases (yuck!)
    b. Note that edx-analytics-configuration uses a fairly old version of ansible, so some changes may need to be made to support the newer version that edx/configuration uses.

7. The analyticstack (devstack) lags behind quite a bit and takes some manual intervention to generate new versions of. It also doesn't support Elasticsearch 1.5, which is used by currently-in-development features in Insights. We'd like to move this into Docker.
    a. A path forward here is to keep many services running in the normal edX devstack and run some set of services in docker containers that just connect to the devstack via exposed ports.
    b. The more services we moved to docker, the more effort required. But this seems to be the architectural direction the organization is moving toward.
    c. See this branch for a working version of edx-analytics-pipeline on docker: https://github.com/edx/configuration/compare/hack2015/gabe/analytics-hadoop
8. Centralize event collection. We should probably be using Kafka or something similar.
9. Non-AWS configuration is rather complex and difficult to setup, which is very painful for the open source community.

From OpenCraft

10. Lack of documentation
11. Problems setting up edX Analytics Devstack (process took a long time, was impossible to complete for one team member; overall complexity of the stack made it difficult to distribute work to additional team members as needed)
12. Problems with Hadoop version conflicts (fixed at the time via a couple of PRs: #128, #127), not really an issue anymore
13. No (straightforward) way to run acceptance tests for edx-analytics-pipeline
14. Using Analytics in production:
    a. Many steps required to install the stack (partly due to Ansible scripts making assumptions about, e.g., AWS regions)
    b. Many steps required to configure Jenkins (manually creating jobs and setting parameters/interval for each Analytics task, etc.)
15. The number of PRs required to implement major changes slows work down (these types of changes often require PRs in four different repos; see "Dependencies" in this example)
16. Not being able to merge PRs implementing work done for clients; having to maintain changes separately
17. Deciding where to add different types of functionality (instructor dashboard vs. insights) was not straightforward in some cases

From the Community

18. TBD during mailing list discussion

Decisions Needed

D1: **Should we use Docker to isolate the major analytics components from the devstack VM**?

a. The software required by the analytics stack (Hadoop etc.) uses a lot of resources and struggles to coexist with the LMS/Studio devstack installation.
b. In addition, if analytics requires a different version of some component such as ElasticSearch than the LMS uses, conflicts will arise.
c. One approach to improving this situation is to move components like ElasticSearch, Hadoop, pipeline, etc. out of the devstack VM and into Docker containers.
    i. this seems to be the architectural direction the edX organization is moving toward:
        1. https://openedx.atlassian.net/wiki/display/OpenOPS/Open+edX+on+Docker
        2. https://github.com/edx/configuration/tree/master/docker
        3. eCommerce team is using a similar approach for ElasticSearch
    ii. See this branch for a working version of edx-analytics-pipeline on docker: https://github.com/edx/configuration/compare/hack2015/gabe/analytics-hadoop
    iii. However, *if* Docker is not being used for production, then this may only serve to make the development environment more complex and to add yet another difference between devstack and production setups.
    (e.g. to develop on Mac, you end up running two VMs anyways, one to host the Docker containers, and one vagrant VM to host the devstack)
    iv. Is using Docker in production an option?
d. An alternate approach is simply to deploy the analytics software in a second vagrant devstack VM, so that developers run two VMs at once: one for the LMS/Studio (optional), and one for the analytics stack (optional). This approach was tested with the first iteration of the analytics devstack and worked fine.

Proposed Changes and Improvements

S1: **Fix the Analytics devstack to unblock developer onboarding**. (OLIVE-5 / OC-1323)

a. Currently, the analytics devstack does not work. It needs some bugfixes and an updated base/box image.
b. approach for now is per AN-6654 option 1
c. Add the “src” vagrant mount dir as seen in Dogwood devstack
d. This is currently in progress: https://github.com/edx/configuration/pull/2753 and https://github.com/edx/edx-analytics-pipeline/pull/199

S2: **Create/update documentation on available pipeline tasks and parameters**.

a. AFAIK, there is currently no authoritative reference that lists the various analytics pipeline tasks and documents their parameters.
b. Partial documentation is at https://github.com/edx/edx-analytics-pipeline/wiki/Tasks-to-Run-to-Update-Insights
c. Gabe: “worth exploring is generating the docs from docstrings. Ideally it would be aware of inheritance of parameters etc.”

S3: **Make pipeline acceptance tests easier to run on devstack** (OLIVE-6 / OC-1324)

a. Convert the steps listed in AN-5750 to ansible tasks, and ensure those tasks run as part of the devstack provisioning
b. Annotate each test to indicate whether it can run on devstack, on Jenkins acceptance test environment, or both
c. Create a wrapper script (e.g. “run_acceptance.sh”) that can run the whole test suite or just one specific task.
d. Document how to run acceptance tests on devstack within the edx-analytics-pipeline README

S4: **To facilitate OpenCraft’s development, set up an OpenCraft analytics “sandbox”**

a. In order to work on the various tasks below, we’ll need a single-server edxapp deployment, valid sample data on the edxapp instance, and a working production analytics deployment using EMR.

S5: **Make pipeline so it is installed like any other component**

a. Finish https://github.com/edx/configuration/compare/hack2015/gabe/analytics-hadoop
b. Remove the ansible playbook that lives within edx-analytics-pipeline
c. Make corresponding updates to analyticsstack and test with the AWS sandbox production setup

S6: **Fix analyticsstack resource limit issues**:

a. split the install out to external Docker containers or a separate vagrant VM?

S7: **Write ansible scripts to set up a working installation of Jenkins that schedules each of the analytics pipeline tasks**

a. Use the Job DSL plugin. edX devops has some examples (not public at the moment).
b. include default parameter values (for number of Hadoop nodes, task intervals, etc.) that are reasonable for a small installation
c. make it possible to enable/disable tasks and override parameters using Ansible vars.

S8: **Generalize and contribute the AWS deployment fixes made by Matjaz**
   a. https://github.com/rue89-tech/edx-analytics-configuration/commits/rue89

S9: **Refactor edx-analytics-dashboard to be plugin-based**
   a. This allows external plugins to be installed, helpful for custom/proprietary analytics as well as to allow community development of Insights features
   b. Ideally, the same ansible vars that enable/disable jobs in Jenkins can also enable/disable the corresponding report plugin in the dashboard

S10: **Combine edx-analytics-configuration into edx-configuration**
   a. Would be nice if the analytics team could take this one on? (We're happy to do so, it just might take us longer than it would the analytics team)

S11: **Contribute Amazon EMR support to Terraform**
   a. https://github.com/hashicorp/terraform/issues/2098 has details and an outline of how to do this

S12: **Create Terraform scripts that can provision the AWS infrastructure needed for analytics**
   a. Would be nice if the analytics team could take this one on? (I believe their AWS setup is more complex than any OpenCraft has used, and we don't have visibility into it so we'll need help to ensure compatibility and best practices)