IBC 2013 EBU demo

Version 04 July 2013

Title: EBU Media Cloud Orchestrating

Subtitle: Open Source Cloud Infrastructure for Encoding and Distribution (OSCIED)

Github: From the opening of the IBC the OSCIED software is available on the EBU GitHub repository.

Description: The demo will show elastic scaling of virtualised encoding and distribution services in a hybrid cloud. A broadcaster is envisaged that needs to efficient allocate recourses to 3 main content flows related to online delivery:

1. Encoding of a constants content flow of live and on-demand files
2. A variable amount concurrent users during the day
3. Transcoding of an archive library for on-demand services

The first operation requires a constant capacity, the second typically peaks in the evening hour while the third one is not a daily routine but more an onetime effort with a separate allocated budget. Even though one can efficiently allocate recourses to a constant process, this is not the case when one has to permanently install servers for a one time action or to handle a short traffic peak.

Virtualisation of services using a hybrid cloud setup will optimise recourses and minimise operational costs by elastically changing the amount of encoding or distribution nodes in a private and public cloud. The system is hybrid as there is a payoff between temporally rented and structural allocated capacity. The flexibility of the cloud approach also allows to adapt sudden growth of popularity of a service as one can temporally upscale the capacity.

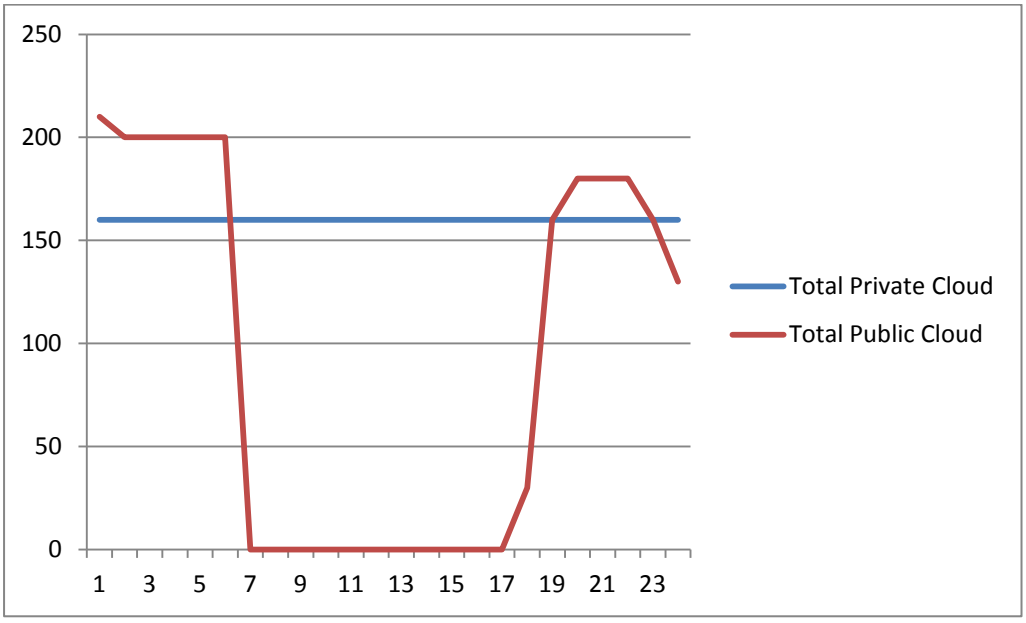


Illustration 1: While the capacity of the local cloud can be accessed constantly the public cloud can be used to offload the peaks.

Example:

Let's say a broadcaster operates a private cloud of 10 machines supporting 160 encoding or distribution nodes as visualised in the illustrations. This rack should be located close to the master control room in order to reduce latency and costs for transporting a constant feed the high quality source material (live and on-demand) over the local IP-network*. In this example 40 nodes are needed as minimum during broadcast hours to encode the constant flow of linear programmed content. During the day the distribution nodes are up-scaled conform the activity patterns of the audience. Normally concurrent use grows slowly during the day while peaking around 8 pm with also a small temporal increase at 8 am. As is it is a shared environment rest capacity can be used all times to transcode the archive.

When content is extra popular, this can be the peak time of the day or a program that goes viral, the public cloud is addressed to temporally step in. For example peak offload in between 17:00 and 01:00 hours. Scaling distribution nodes in the cloud can also be helpful if it improves the peering with certain ISPs. While competing cloud provider have different price levels and some of them use different pricings for time slots during the day one could also use the cloud to budget efficient speed up transcoding of an archive. In the illustration add 200 nodes between 1 and 7 AM, none between 7 AM and 7 pm and 100 after that hour and twelve o'clock in the night.

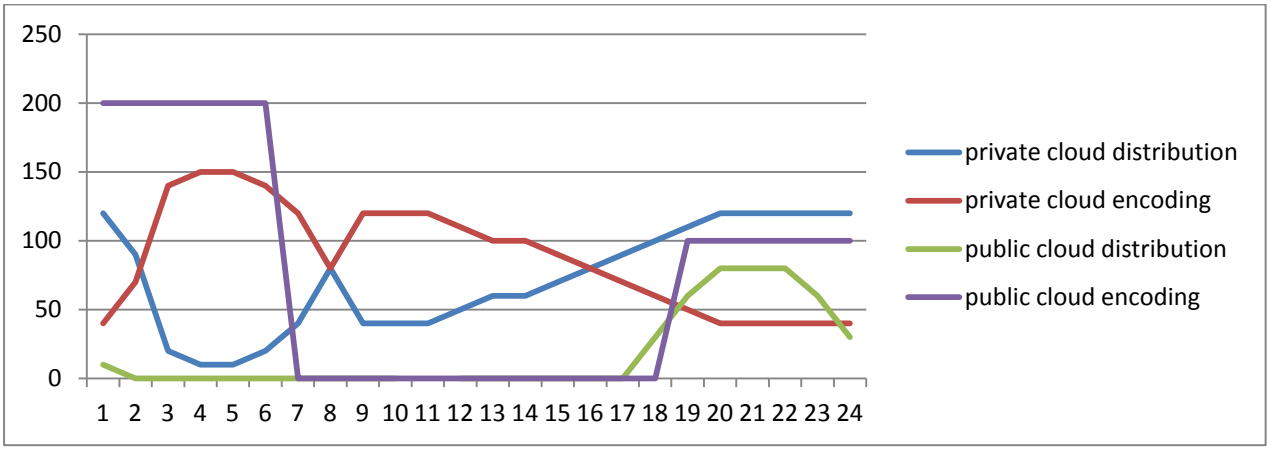


Illustration 2: The total constant capacity of the private cloud can be used by different virtualised services during the day. The public cloud is only used for offloading distribution peaks or cost efficient upscaling of transcoding jobs.

*The private cloud can also be located in gateways close to important POPs and connected via a fiber-ring. This scenario is not reflected in the demo but would only set some parameters mainly related to the bitrate of the contribution files.

2 virtualised services in OSCIED:

1. Encoding: Archive encoding where you have a large media set that needs to be transcoded to different output file formats. Capacity needs to be balanced between the time it takes to transcode the files and the related cost.
2. Distribution: Live encoding is a constant dataflow optimised for low latency and an audience viewing at different times during the day with different devices. Therefore encoding capacity

is almost constant and done in the private cloud while distribution is variable and automatically scaled up in the cloud.

Questions and answers

An encoding node is a virtualised transoding server, in OSCIED based on ffmpeg/DashCast. The output of one node depends on the allocated CPU (perhaps also GPU support?), RAM and IO-speed (data speed from storage to and from CPU). Q: How does this work in a cloud?

In our setup we allocate with MAAS

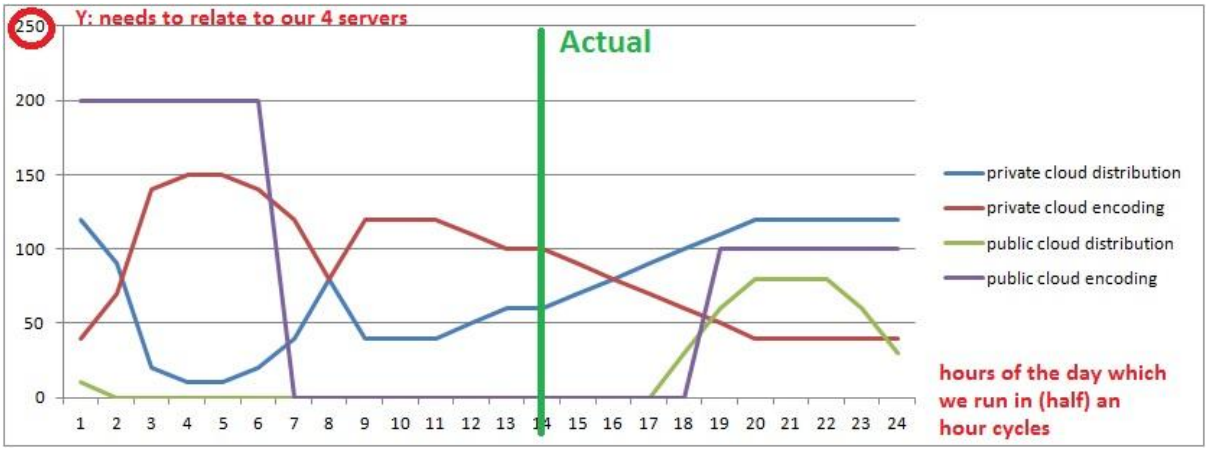
A distribution node is in OSCIED a virtualised Apache webserver using Unified Streaming MOD264 module. The output is less intensive for CPU, RAM and IO-speed but needs to output over IP the concurrent streams to the end user. Normally a server with a dual Gigabit card can serve 2500 concurrent streams of 800kbs. Q: How does this work in a cloud?

Questions

1. What it the optimal output of a node?
2. How do we calculate the cost of a node in private and public cloud?
3. How to identify what the ideal state is between private and public cloud?
4. How to calculate the data costs from the private to the public cloud?

Physical setup and logistics:

- One big screen showing the private and public cloud being used at one moment. We simulate an amount of nodes during the day but run this cycle every (half) hour according.



- 4 servers for demo (can we see them through a window?) connected with a GB wireless router.
- An internet connection with maximum upload and download capacity (budget dependant).
- 1 laptop that is used to demonstrate the user interface and IO. This laptop will also be used to show the software layers of this project.

- 1 iPad and 1 Android machine for playout of files on distribution nodes of the private and public cloud

OSCIED setup

Definitions

An OSCIED service can be either scalable TRANSFORMATION (encoding/transcoding), DISTRIBUTION (webserver), STORAGE unit (also called node or charm) or the ORCHESTRATOR, WEBUI unit which normally are fixed to one location.

TRANSFORMATION service: This is the virtualised service of OSCIED that contains one or a collection of transform units (encoders) located in private and / or public clouds.

Transformation unit: A single Transform unit is a Worker dedicated to encoding or transcoding files from a source to an output. A Transform unit can harbour different software encoders. In the current version we adopted FFMPEG and DashCast but this could be other encoding software like GStreamer, VLC, etc..

Transformation profile: Predefined encoder options that define which encoder should be addressed and what variables should be used, for example the 'DASH vod_x264_HbbTV' profile contains the parameters for the DashCast encoder to output optimised files for playout to HbbTV services.

Task Queue: Mechanism to distribute work across threads or virtual machines. Routing tasks of different processes in OSCIED, for example to send jobs in TRANSFORM or DISTRIBUTION units. (Flexibility of queues and business rules .....)

Business rule: External parameter for the scheduling algorithm

Scheduling algorithm:

Job: Synonym used for Task queue input (Celery vocabulary) which is a unit of work that needs to be done by a Worker.

Worker: This is either a TRANSFORMATION or a DISTRIBUTION unit that constantly monitor the Task queues for pending jobs.

There are **4 deployment scenarios** for installing and using OSCIED in private and public clouds (more options not excluded):

1. Private OSCIED cloud only
2. Private OSCIED cloud plus extended virtual encoding / distribution OSCIED services in public clouds
3. Public OSCIED cloud only
4. Private OSCIED cloud plus Public OSCIED clouds in parallel

You can integrate OSCIED in your own infrastructure on different levels:

1. By connecting your own SAN instead of using OSCIED STORAGE service
2. Using the API to connect to your own MAM system
3. More options?

Flow Charts

1. System flow charts (physical and software layers private and public cloud scenarios)
2. Data flow chart (how the data can travel through the system)
3. Cloud stack (showing what OSCIED does and OpenStack can do)
4. Connection points (API flow chart)

OSCIED API

OSCIED can be fully operated and monitored via RESTful API calls for either create, read, update and delete actions (CRUD). There are different subclasses of the API that address different services and functionalities.

1. UserAPI: Management of users that can access the platform (define users with different right levels; in the IBC13 version this is restricted only to ADMIN that can change everything from anybody and other users that can only change their own assets/jobs).
2. MediaAPI: Manage the files in the STORAGE service (register or delete the media files and add/change the metadata of the asset in the database).
3. EnvironmentAPI: Manage different environments which can be targets for deployments of the services / units / Workers. This can be a local setup or one in a specific cloud (the IBC13 version only supports the list option).
4. TransformProfileAPI: Manage the profiles you have in your library (only add, read and delete profiles as update can potentially create conflicts in current encoding jobs)
5. TransformQueueAPI: Ask what encoder queues exist.
6. TransformJobAPI: Manage encoding/transcoding jobs (add a job by specifying options what the source file (input media), name of the output file (output media), which profile should be used, what descriptive metadata of the asset should be added to the database, specify to which queue the job is added)
7. TransformUnitAPI: Check active Transform units and ask to deploy or remove units in the different environments (for emaple private cloud, MAAS setup, the Amazon cloud, etc.).
8. PublishJobAPI: Publish assets on the DISTRIBUTION service allowing assets to be located in different available distribution nodes located either in the private or public clouds. You can specify Task queues that place assets on specific Distribution nodes or a Task queue that places the asset on all available Distribution nodes.
9. PublishQueueAPI: Ask what distribution queues exist.
10. UnplublishAPI: revoke assets from DISTRIBUTION service.

Installation guide for OSCIED Scenario's

This is a proven step by step installation description. It is not the only way to deploy OSCIED in your own infrastructure.

A: Install physical setup and operating system components needed for private cloud setup (scenario 1,2 and 4):

1. Install servers (profile advise)
    a. Allocate servers in one network
    b. Configure BIOS for WAKE-ON LAN, Virtualisation Extension etc (list of BIOS setup)
2. Use a separate laptop/desktop machine (any OS) that can access the servers from step 1 for:
    a. Download / clone OSCIED repository from GITHUB/EBU
    b. Download UBUNTU server LIVE CD
    c. Download and setup VIRTUAL BOX (VM-ware Workstation, KVM or any other)
3. Start VIRTUAL BOX and create a virtual machine (VM)
    a. Configure the VM to boot from UBUNTU LIVE CD image
    b. Start the VM and choose in UBUNTU setup (Splash menu) MAAS controller to start up Maas controller as a guest of your computer
4. Follow installation guide from ....(To DO) until MAAS is configured
5. Boot your physical severs and accept them in MAAS webUI as a cluster (or add to cluster)
6. Now you can use the script to setup OSCIED (manual install is possible just follow what the script does)

B: Install OSCIED in private cloud

Maas is the provisioning tool for Ubuntu setup on specific machines

Open STack

Juju is the deployment tool (alternative of Slipstream) for services (service orchestration

1. Installation with Maas
    a.