Plugging OpenVMS into the Cloud

General comments and examples

Brett Cameron September 2014

Abstract

This short talk provides an overview of how OpenVMS can interact with and utilize cloud-based services. General techniques for interacting with cloud-based services from OpenVMS will be considered, with particular attention being given to interfacing with the OpenStack Object Storage service and a prototype RESTful cloud-based message queuing service.

AGENDA

- Interfacing with cloud-based services
- Example plugging OpenVMS into the OpenStack Object Storage service
- Example message queuing as a service
- Summary
- Questions

Interfacing with cloud-based services

- Interfaces to many cloud services are implemented using HTTP(S)/REST API's
 - Representational state transfer (REST)
 - A fancy (and somewhat miss-used) term for a fairly simple concept
 - See my talk with Jeff Allen "OpenVMS and web services: theory and practise" for more on this topic
- Typical examples of such API's would be:
 - Identity Service API
 - One-stop authentication service for all services within the cloud
 - Object Storage API
 - Store, retrieve, delete, copy, and examine objects (files)
 - Compute API
 - Create, destroy, reboot, and rebuild virtual servers
 - Block Storage API
 - Provides a means of enabling additional storage volumes for compute instances
 - Monitoring, Load Balancing, DNaaS, DBaaS, SDN, and any other services that may be provided by the cloud in question

Interfacing with cloud-based services

- These API's generally provide a means to perform administrative operations relating to the service
- For some services there may also be additional (non-HTTP/REST-based) interfaces
- And some services may provide HTTP/REST-based interfaces to programmatic (non-administrative) functions
- Cloud providers generally provide high-level command-line tools and bindings for various languages that use these API's to interface with the associated services
 - Most are not particularly OpenVMS-friendly
 - Tend to be written in languages such as Ruby or Python
 - Not readily callable from OpenVMS 3GL programs
- Click here to take a look at a typical REST API definition (hopefully I'm hooked up to the internet)

Interfacing with cloud-based services

- Interfacing with cloud-based services is for the most part quite straightforward
 - Dealing with XML or JSON-formatted request and response messages is generally the most problematical aspect
- Things like cURL and libcURL are your friends!
 - You can in fact use cURL to issue many service calls manually (although dealing with some of the responses may be fun)
 - A command line tool like cURL can be very useful for basic testing and prototyping
- From a programmatic perspective you will typically need:
 - An HTTP(S) client API
 - libcURL is a good choice (<u>http://curl.haxx.se/libcurl/</u>)
 - Either an XML or a JSON parser/API
 - Most HTTP/REST API calls return JSON or XML responses
 - $\,\circ\,\,$ You can specify which format you want to use
 - I prefer JSON
 - $\circ~$ It tends to be a little much more efficient and somewhat easier to work with (for me anyway)
 - The json-c library is easily ported to OpenVMS (see <a href="https://github.com/json-c/json

AGENDA

- Interfacing with cloud-based services
- Example plugging OpenVMS into the OpenStack Object Storage service
- Example message queuing as a service
- Summary
- Questions

OpenStack Object Storage

- A redundant, scalable, and dynamic storage service
- A safe, secure, network-accessible way to store data
- Can store an essentially unlimited quantity of objects (files)
 - Each file can be up to 5GB
 - With segmented objects, it is possible upload and store objects of virtually any size
- Allows users to store and retrieve files via a simple HTTP/REST interface
 - See http://docs.openstack.org/api/openstack-object-storage/1.0/content/ for API details
 - It could be likened to a sort of primitive Dropbox-like facility
- An HTTP/REST API is by itself is generally not all that useful, but it is ubiquitous
 - Could use something like cURL, but would typically want to implement a higher-level API on top of the REST API
 - The higher level API can then be used by other programs
 - As a proof-of-concept exercise we've developed a simple high-level API and CLI utility for OpenVMS...

High-level prototype API routines

• API functions:

```
hpc$init()
hpc$authenticate()
hpc$set_proxy()
hpc$list_container()
hpc$free_object_list()
hpc$create_container()
hpc$delete_object()
hpc$get_statistics()
hpc$get_object()
hpc$get_usage()
hpc$put_object()
```

A simple set of functions to handle authentication, the creation, deletion, and browsing of containers, and the uploading and downloading of files (objects). Very much like FTP.

- Currently really only callable from C code, but straightforward to make language-neutral
- Leverages libcURL for HTTP(S) client API
- Leverages json-c for JSON parsing

CLI utility \$ define hpc\$proxy "16.172.48.10:8080" \$ run hpc set noverify create container "xyz" put/log "bootcamp.com" "xyz" list container "xyz" get/log "bootcamp.com" "xyz" create container "zzz" delete object "bootcamp.com" "xyz" delete object "bootcamp.com" "zzz" list container "zzz" delete container "xyz" delete container "zzz" exit Ś \$ exit





12

Potential enhancements

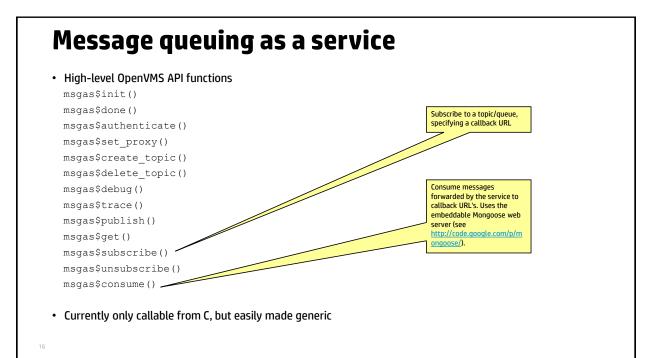
- Functionality to backup/stream files to object storage containers
- API needs to be made language-neutral
- Include support for other services (particularly Compute)
- Understanding and handling of OpenVMS file types and RMS attributes

AGENDA

- Interfacing with cloud-based services
- Example plugging OpenVMS into the OpenStack Object Storage service
- Example message queuing as a service
- Summary
- Questions

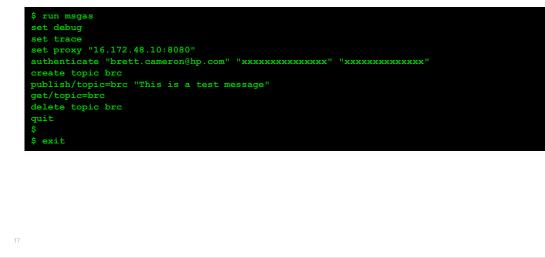
Message queuing as a service

- Prototype cloud-based message queuing service developed last year for HP Cloud
- May or may not be production-ized
- Supports point-to-point and publish/subscribe messaging operations
- Can be multiple subscribers to a topic/queue
- Highly scalable, fault tolerant, secure, ...
- · Can be used in-cloud or over WAN/internet links
- HTTP(S)/REST interface
 - Somewhat similar to Amazon's SQS and SNS services and Iron.MQ (http://www.iron.io/mg)
- Again, we've developed a simple prototype high-level API and CLI utility for OpenVMS...



CLI utility

Simple example using the CLI to authenticate, create a topic/queue, publish a message, get a message, and delete the topic/queue:



AGENDA

- Interfacing with cloud-based services
- Example plugging OpenVMS into the OpenStack Object Storage service
- Example message queuing as a service
- Summary
- Questions

Summary

- Interfaces to cloud-based services are typically HTTP(s)/REST-based
- These cloud-based services are readily accessible from OpenVMS
- Prototype high-level OpenVMS 3GL-friendly API's and utilities have been developed for object storage and message queuing services
- I'll talk more about some of this stuff later on when we are discussing NoSQL databases, and I will be discussing REST in a little more detail in my session with Jeff Allen "*OpenVMS and web services; theory and practice*"

AGENDA

- Interfacing with cloud-based services
- Example plugging OpenVMS into the OpenStack Object Storage service
- Example message queuing as a service
- Summary
- Questions

