Hammr
Building Consistent Machine Images for the Cloud

Alexandre Lefebvre

alexandre.lefebvre@usharesoft.com
@alefebvr

www.usharesoft.com
@usharesoft

OW2con'14

PARIS OPENSTACK SUMMIT NOVEMBER 2014

Industry Problem

Industry Problem: Cloud Software Onboarding

Application Focused

Machine images are created and managed manually	Limited control of software stacks threatens software governance
Limited self-service tools to enable the ecosystem	Application portability difficult A pre-cursor for hybrid cloud

Infrastructure Focused

To harness the full agility of Cloud, software onboarding requires to be automated and have self-service

Required Building Blocks for Hybrid Cloud Management

Networking

- VPN
- Network Management (SDN)

Single Pane of Glass Management & Monitoring

- Self-Service Provisioning
- Workflow Process
- Capacity and Usage
- IT Process Orchestration

Software On-boarding & Management

- Self-Service Marketplace
- Machine image creation
- Migration Services
- Patch management
- Audit

Required Building Blocks for Hybrid Cloud Management

Networking

- VPN
- Network Management (SDN)

Single Pane of Glass Management & Monitoring

- Self-Service Provisioning
- Workflow Process
- Capacity and Usage
- IT Process Orchestration

Software On-boarding & Management

- Self-Service Marketplace
- Machine image creation
- Migration Services
- Patch management
- Audit

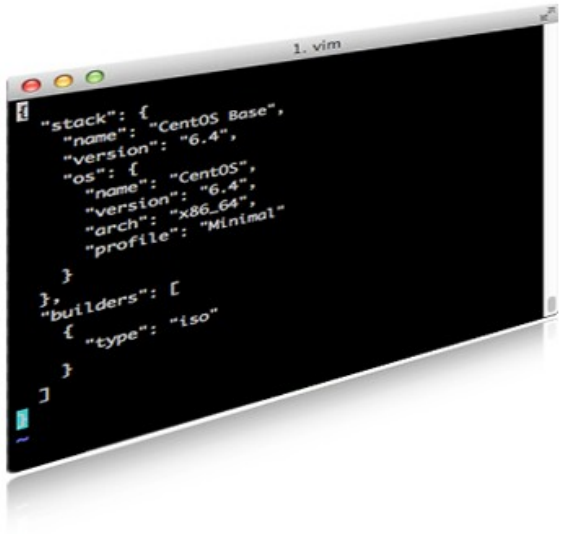
hammr

What is hammr ?

Hammr Command-Line Tool

Hammr Command-line Tool: Consistent Machine Images from a Single Configuration File

- Open source OW2 project
- command-line tool written in Python
- www.ow2.org/ActivitiesDashboard/hammr – www.hammr.io
- Github: https://github.com/usharesoft/hammr



build

amazon web services™ vmware EUCALYPTUS openstack VAGRANT nimbula cloudstack flexiant™ your cloud simplified Windows Azure

Installing Hammr

Installing Hammr

Install Hammr system dependencies first, then

```
$ pip install hammr
$ hammr –v
hammr version '0.2.3'
```

Help Menu

```
$ hammr --help
```

Installing Hammr: Authenticating with UForge

Authentication Information* (in credentials JSON file or as command-line parameters)

```
$ cd ~/.hammr
$ vi credentials.json
{
  "user" : "alex",
  "password" : "password",
  "url" : "https://factory.usharesoft.com/ufws-3.3"
}
```

* Your UForge account.

Get a free account at http://www.usharesoft.com

Creating & Managing Your Stacks

Creating and Managing Your Stacks

Generate and Publish

Generate a machine image for any of the leading hypervisors, or cloud formats.
Publish the machine image directly to your target environment using your credentials.

Model Your Software Stack

Speed up and simplify the way you deliver IT software.
Model and maintain the entire software stack as an appliance template including OS packages, middleware, application software and configuration.

Share with Colleagues

Export your templates and share with colleagues

Getting Started with Hammr

- Template file: JSON configuration file
- Describe your application stack using the `stack` keyword
- Generate machine images by defining `builders`
- Publish the generated machine images using your cloud credentials

Defining your Stack: Example nginx

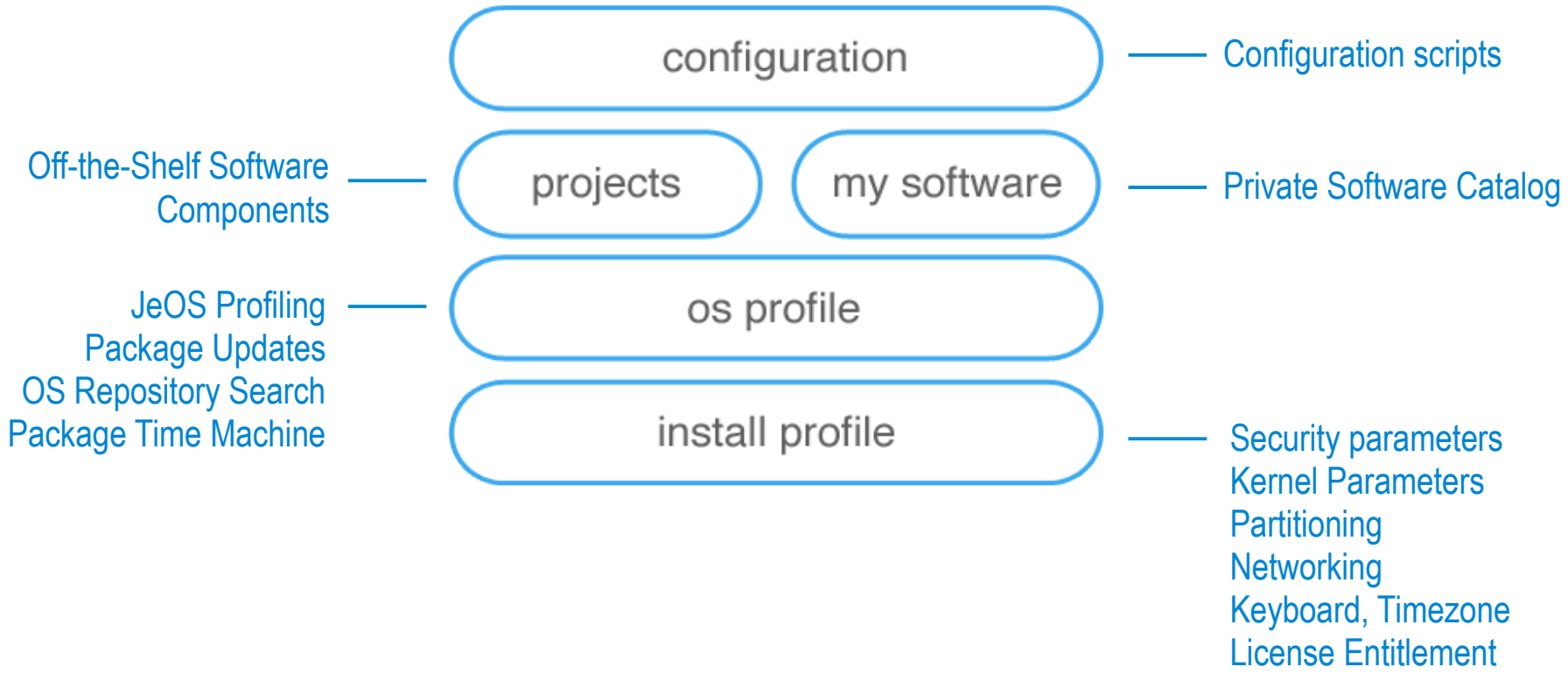
```
$ vi nginx-template.json
{
   "stack": {
       "name": "nginx",
       "version": "1.0",
       "os": {
       "name": "Ubuntu",
           "version": "12.04",
           "arch": "x86_64",
           "profile": "Minimal",
           "pkgs": [
               {
                   "name": "nginx"
               }
           ]
       },
       "installation": {
           "diskSize": 12288
       }
   }
}
```

`stack` keyword defining the content of your stack

`os` keyword defining the os profile and packages

`installation` keyword defining the “install” parameters

Hammr: Modeling the Stack from a Single Config File



Creating the Template: Example nginx

Verify the JSON syntax: `template validate`

```
$ hammr template validate --file nginx-template.json
Validating the template file [/Users/james/nginx-template.json] ...
OK: Syntax of template file [/Users/james/nginx-template.json] is ok
```

Create the Template: `template create`

```
$ hammr template create --file nginx-template.json
Validating the template file [/Users/james/nginx-template.json] ...
OK: Syntax of template file [/Users/james/nginx-template.json] is ok
Creating template from temporary [/var/folders/f6/8kljm7cxgn/T/hammr-15888/archive.tar.gz] archive ...
100%|################################################################################|
OK: Template create: DONE
Template URI: users/root/appliances/898
Template Id : 898
```

Listing Your Created Templates

List created templates: `template list`

```
$ hammr template list
+-----+----------------------+---------+--------------------+---------------------+---------------------+--------+---------+
| Id  |         Name         | Version |         OS         |       Created       |    Last modified    | # Imgs | Updates |
+=====+======================+=========+====================+=====================+=====================+========+=========+
 683 | nginx                | 1.0     | Ubuntu 12.04 x86_64 | 2014-05-02 13:59:25 | 2014-05-02 13:59:27 | 0      | 0       |
+-----+----------------------+---------+--------------------+---------------------+---------------------+--------+---------+
Found 1 templates
```

Defining a Builder: Machine Image Format

```
$ vi nginx-template.json
...stack section omitted
{
  "builders": [
    {
      "type": "openstack",
      "account": {
        "file": "/home/joris/accounts/openstack-
account.json"
      },
      "tenant": "opencloudware",
      "image-name": "test-nginx",
      "description": "joris test nginx image"
    }
  ]
}
```

`builders` keyword defining all the machine images to build

Image format to build

Cloud account credentials

Machine image registration information

Defining a Builder: Machine Image Format

```
$ vi nginx-template.json
...stack section omitted
{
  "builders": [
    {
      "type": "openstack",
      "account": {
        "file": "/home/joris/accounts/openstack-
account.json"
      },
      "tenant": "opencloudware",
      "image-name": "test-nginx",
      "description": "joris test nginx image"
    }
  ]
}
```

← Cloud account credentials

```
$ vi openstack-account.json
{
  "accounts": [
    {
      "type": "openstack",
      "name": "My OpenStack Account",
      "endpoint": "http://ow2-04.xsalto.net:9292/v1",
      "keystoneEndpoint":
            "http://ow2-04.xsalto.net:5000/v2.0",
      "username": "test",
      "password": "password"
    }
  ]
}
```

Generating Machine Image(s)

Create any cloud account first: `account create`

```
$ hammr account create --file openstack-account.json
Validating the template file [openstack-account.json] ...
OK: Syntax of template file [openstack-account.json] is ok
Create account for 'ow2stack'...
OK: Account create successfully for [openstack]
```

Generate the machine images: `template build`

```
$ hammr template build --file nginx-template.json
Validating the template file [nginx-template.json] ...
Generating 'openstack' image (1/1)
|>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>| 100%: Done, created on ... |<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<|
OK: Generation 'openstack' ok
Image URI: users/root/appliances/21/images/47
Image Id : 47
```

Image ID: 47, used to register this generated machine image to the target cloud environment

Publishing Machine Image(s)

Publishing the generated machine image: `image publish`

```
$ hammr image publish --id 47 --file nginx-template.json
Publishing 'openstack' image (1/1)
|>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>| 100%: Done, published o... |<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<|
OK: Publication to 'ow2stack' is ok
Cloud ID : 37d20dba-7a1c-43f9-9c77-c05f60d3094c
```

Cloud ID: used for provisioning instances on OpenStack

Listing Your Images: `image list`

```
$ hammr image list
```

Supported Target Machine Image Formats

Physical

- ISO

Virtual

- Hyper-V
- KVM
- Raw
- QCOW2
- Vagrant
- VirtualBox
- VHD
- VMware Workstation
- VMware ESXi
- VMware vCenter vSphere
- Xen

Cloud

- Abiquo
- AWS (AMI)
- CloudStack
- Eucalyptus (EMI)
- Flexiant
- Google Compute Engine
- Microsoft Azure
- Nimbula
- OpenStack
- VMware VCD

Migration

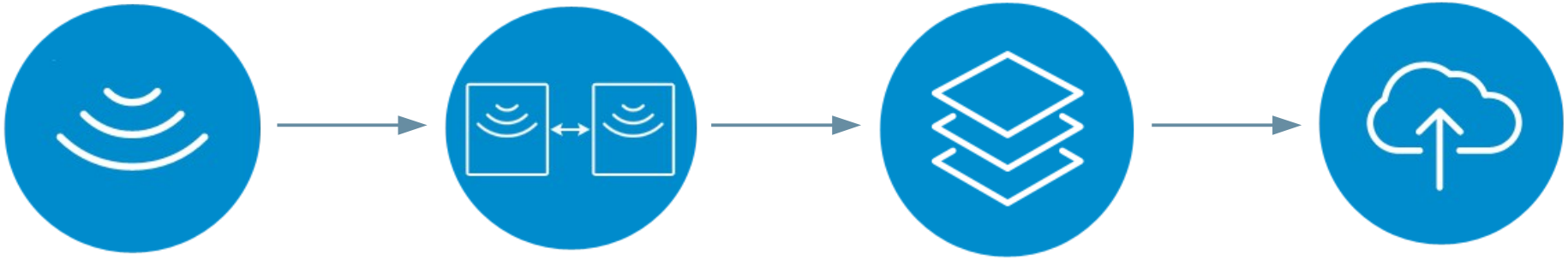
Migration Overview

Compare Scans

Compare any two scans to determine their differences. This can be an incremental scan from the same machine to detect server drift or from two different machines to determine any configuration changes.

Migrate To Target Environment

Generate a machine image for any of the leading hypervisors, or cloud formats. Publish the machine image directly to your target environment using your credentials.
This is the final migrated system.



"Deep Scan" the Workload

All native packages, files and configuration of the system is detected and reported as a scan report.
Scanning the system multiple times creates incremental scans.

Make Changes (Whitebox Migration)

Import your scan as an appliance template allowing you to update or change the contents of the scan prior to migrating.

Scanning the Source Workload

Deep scan the system: `scan run`

```
$ hammr scan run --ip 192.0.2.0 --login LOGIN --name scan-name
Password for root@192.0.2.0:
... uforge-scan v2.54 (Feb 18 2014 13:16:37) (SVN Revision: 21664)
... Distribution: Debian / 6.0.9 / x86_64
... Current System Name: Linux
... Node Name: test-deb-1-0rev2-vbox
... Release: 2.6.32-5-amd64
... Version: #1 SMP Tue May 13 16:34:35 UTC 2014
... Machine: x86_64
... Domain: (none)
... Server URL: http://192.168.10.141/ufws-3.3
... User: root
... Testing connection to the service...
... SUCCESS! ...
...  Searching scan on uforge
... |>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>| 100%: Successfully scanned |<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<| OK: Scan successfully
```

Migrate Using the Scan

List Your Scans to get the Id: `scan list`

```
$ hammr scan list
Getting scans for [root] ...
+-----+----------------------------+--------+-----------------+
| Id  | Name                       | Status | Distribution    |
+=====+============================+========+=================+
| 133 | scanExample                |        | Debian 6 x86_64 |
+-----+----------------------------+--------+-----------------+
| 149 | scanExample Scan #1        | Done   |                 |
+-----+----------------------------+--------+-----------------+
Found 1 scans
```

Generate and Publish (using a `builder`): `scan build, publish`

```
$ hammr scan build --id 149 --file openstack-builder.json
OK: Syntax of template file [openstack-builder.json] is ok
Generating 'openstack' image (1/1)
|>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>| 100%: Done, created on ... |<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<| OK: Generation 'openstack' ok
$ hammr publish --id 45
```

Whitebox Migration

Import a Scan as a Template: `scan import`

```
$ hammr scan import --id 123 --name "MyScan" --version "1.0"
Import scan id [123] ...
|>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>| 100%: Imported 28 May 2... |<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<| OK: Importing ok
```

This is now a template (stack), that can be changed prior to generating and publishing

hammr demo

hammr

machine image builder for the cloud

It's an open source OW2 project
It's OpenStack friendly

Use it! Contributions welcome!

hammr.io
ow2.org/ActivitiesDashboard/hammr
https://github.com/usharesoft/hammr

www.usharesoft.com

@usharesoft

OW2con'14

PARIS OPENSTACK SUMMIT NOVEMBER 2014