Dos and Don’ts for Virtual Appliance Preparation

Boris Parák\textsuperscript{1}  Enol Fernández\textsuperscript{2}

\textsuperscript{1}CESNET

\textsuperscript{2}EGI.eu

Nov 10, 2015
Introduction
Virtual Appliance

- a set of one or more virtual resource descriptors
- in the cloud context, virtual resource == virtual machine
- metadata & binary data of included virtual machines
- pre-installed and *partially* pre-configured software
- simplifies and speeds up resource deployment for users
- for the purposes of this talk, one resource per appliance
Life-cycle

Create → Publish → Use → Update → Expire → Finish → Discontinue

Dos and Don’ts for Virtual Appliance Preparation
Maintenance
Pros & Cons

+ easier for users, no installation guides
+ faster instance convergence, *provisioning* $\rightarrow$ *work*
+ fine-grained environment control (version, configuration)

- time-consuming preparation of appliances
- issues with interoperability across different platforms
- challenging security-related aspects
EGI.eu VAs

- virtual appliances provided & endorsed by EGI.eu
- available in The EGI Applications Database
- mostly so-called base appliances containing a clean operating system
- providing useful jumping-off points for new users
- making sure users do not repeat the same mistakes
- isolating users from technical details of our infrastructure
Community VAs

- virtual appliances provided by the community
- available in (& endorsed by) selected virtual organizations
- available in The EGI Applications Database
- contain end-user applications & computational frameworks
- responsibility of the community and endorsing VO managers
Should You Have One?

- depends on your workflow and applications
- use generic/base appliances as much as possible
- consider utilizing contextualization tools with base appliances
- create your own, if you really have to and know how
- a virtual appliance is your **pet**, for as long as it lives
Hands-on Interlude
Instructions

Prerequisites:

1. Install & test VirtualBox
2. Install & test Packer
3. Install & test OVFTool

Steps:

1. Get the tutorial package (flash drive or link)
2. Run the automated build
3. Package the resulting appliance as OVA
https://goo.gl/Md7zJp

$ cd fedcloud-userinterface-packer

$ packer build fedcloud-userinterface.json

VA Preparation Basics
Think First

• Why am I doing this? Do I have to?
• How can I distribute my application in the best way possible?
• Which operating system should I choose?
• Do I know how to configure the chosen operating system?
• How should I (pre-)configure my application?
• Can I easily adjust my application’s configuration for each instance?
Installation

General:
• minimalistic OS installation
• basic configuration for remote access
• contextualization support (e.g., cloud-init)
• guest utilities or agents, not good for portability
• integration with 3rd party services

Linux-specific:
• no GUI, no desktop applications, no network managers
• up-to-date kernel & modules, ideally v3+
• avoid complex partition layouts
• make sure /etc/fstab and grub use labels or UUIDs
Contextualization

- every appliance must support contextualization
  → metadata-based configuration on boot
- **cloud-init** is the *de facto* standard
- YAML-based configuration file containing user data
- helps with credentials, user accounts, mounts, package installation, writing files, adding repositories, ...
- do **NOT** forget to remove `/var/lib/cloud` before distributing the appliance
Configuration

- always keep in mind “less is better”
- no unnecessary services or user accounts
- be careful with publicly visible listeners
- no password-based authentication, if possible
- no pre-installed credentials or “backdoor” utils
- let contextualization do the work
Security

- minimize the attack surface for running instances
- always use an up-to-date system, enable security updates if possible
- do **NOT** use plain authentication methods
- always check your newly created appliance
  - `nmap`, `netstat`, `lynis`, `ssh`
  - `/etc/passwd`, `/etc/shadow`, `/etc/sudoers`
  - `/etc/ssh/sshd_config`, `~/.ssh/authorized_keys`
- think about runtime patch status monitoring
$ apt-get update && apt-get dist-upgrade
# or
$ yum update

$ ssh -o PreferredAuthentications=none localhost

$ lynis audit system
# or
$ lynis --auditor system

$ nmap -sS localhost

$ netstat -tapn
$ netstat -uapn
Image & Appliance Formats

- every virtualization platform uses a different native disk image format
- commonly used: qcow(2), vdi, vmdk, raw
- compression optional, reduces size → reduces performance
- on top of that, appliance “envelopes” → OVF/OVA
- Open Virtualization Format → Open Virtual Appliance
  - OVF appliance descriptor (metadata)
  - disk image(s), usually vmdk
Distribution w/ The EGI AppDB

→ Demo Later
Advanced Topics
Format Conversion

Tools:
- QEMU utilities (for disk image conversion)
- VMWare OVFTool (for OVF/OVA transformation)

```sh
## convert formats
$ qemu-img convert -f vmdk -O qcow2 Appliance.vmdk Appliance.qcow2
$ qemu-img convert -f qcow2 -O vmdk Appliance.qcow2 Appliance.vmdk
$ qemu-img convert -f qcow2 -O raw Appliance.qcow2 Appliance.raw

## package
$ ovftool MyFirstAppliance.ovf MyFirstAppliance.ova

## unpack
$ tar xvf MyFirstAppliance.ova
```
• changing hardware (CPU, memory, PCI devices)
  → most systems can cope with that
• dynamically adding/removing disks
  → most systems can cope with that
• dynamically adding/removing NICs
  → disable `udev` rules generator
• different network topologies
  → most systems can cope with that
• different methods of contextualization
  → `cloud-init v0.7.5+`
• different supported appliance/image formats
  → use one of the most popular formats and hope for the best
Automation

- identify repetitive tasks, don’t perform them manually
- good places to start looking
  - OS installation (building)
  - configuration (provisioning)
  - publishing appliances (distribution)
- automation saves your time and makes the process more reliable
Automating Builds

Tools:
- Packer → https://www.packer.io/
- VeeWee → https://github.com/jedi4ever/veewee

Sample Packer Templates:
- https://github.com/shiguredo/packer-templates
- https://github.com/joefitzgerald/packer-windows
- Notice: builders, provisioners, post-processors

$ packer build template.json
Automating Provisioning

- usual suspects
  - scripting (shell, powershell, ...)
  - ansible, saltstack, chef
  - puppet
- select one based on the complexity of your application
- use in Packer provisioners
Automating Distribution

EGI Federated Cloud:
• The EGI AppDB (w/ custom built tools)

Out There:
• Project Raindrops http://projectraindrops.net/
• Atlas by HashiCorp https://atlas.hashicorp.com/
The EGI AppDB
Quick AppDB Summary

1. SSO Account Registration
2. First AppDB Login (w/ SSO account)
3. Quick AppDB Orientation
4. Registering a VA
5. Asking for VO Endorsement
Quick AppDB Summary

Example VA: https://goo.gl/WqEKsq

Register VA: https://goo.gl/2cCCjl

Populate VA: https://goo.gl/YikgZN

Notify VO(s): https://goo.gl/OxUTz7
Overall Summary

(Credit: Toronja Azul via Creative Commons)

...and automate ... A lot!
– That’s All Folks! –

...
Do you have any questions?

• ask NOW!
• ask us directly at parak@cesnet.cz or enol.fernandez@egi.eu
• send your questions to ucst@egi.eu