Debian Appliance Builder

Revision as of 19:31, 22 October 2015 (Just before LXC page transition)

Introduction

Creating high quality appliances is a difficult task and requires deep knowledge of the underlying operating system. So we created the 'Debian Appliance Builder' to simplify that task. 'dab' is a script to automate the creation of OpenVZ appliances. It is basically a rewrite of debootstrap in perl, but uses OpenVZ instead of chroot and generates OpenVZ and now also LXC templates. Another difference is that it supports multi-stage building of templates. That way you can execute arbitrary scripts between package installation steps to accomplish what you want.

Furthermore, some common tasks are fully automated - like setting up a database server (mysql or postgres). To accomplish minimal template creation time, packages are cached to a local directory, so you do not need a local Debian mirror (although this would speed up the first run). All generated templates includes an appliance description file. Those can be used to build appliance repositories.

'dab' is licensed under GPLv2 (Open source).

Main features

- Fully automated build process with standard 'make' scripts
- Local package cache
- Multi-stage build process
- Creates and includes appliance description files
- Simplifies various tasks like database installation
- Appliances can be certified and integrated into the Proxmox VE download directory
- Supports Debian 5, 6 and 7 based appliances
- Generates 32 or 64 bit appliances
- Automatic dependency resolution

Target audience

'dab' is an easy way for software vendors and open source project to package their product into a "ready-to-run" appliance. Appliances can be published into the Proxmox VE appliance download directory to reach the target customer. See Download templates (Video).

In order to work with 'dab', basic Linux knowledge is needed.

DAB for LXC in PVE 4+

DAB can now be used to create LXC templates where the SSH keys are regenerated at creation / cloning time.

Updates to DAB

DAB is developed in the Git Repo at:

https://git.proxmox.com/?p=dab.git;a=summary

The DAB binary is located at:

`/usr/share/perl5/PVE/DAB.pm`

An unofficial backported DAB for PVE 3.x is discussed here.

OpenVZ **conf** directory in PVE 1.x is at `/etc/vz/conf` whilst in PVE 3 it is in `/etc/pve/openvz/` and is set in **DAB.pm** as my `$cfgdir = "/etc/vz/conf";` or my `$cfgdir = "/etc/pve/openvz";` around line 312.

Installation

The script needs to be run as root, so it is not recommended to start it on a production machine with running containers.

System requirements

Prepare your operating system - you can install 'dab' on the following operating systems:

- Proxmox VE (fastest)
- Proxmox VE inside Proxmox VE (KVM) - (most comfortable)
- Proxmox VE inside VMWare/XEN 64bit virtual machine
- Debian 64/Ubuntu 64 with OpenVZ Kernel (not tested)

'dab' installation

Log into Proxmox VE, and install 'dab' using apt:

```
apt-get install dab
```

Or download and install it manually:

```
wget http://download.proxmox.com/debian/dists/squeeze/pve/binary-amd64/dab_1.2-3_all.deb
```

The bleeding edge known to work and is available at:

```
wget http://download.proxmox.com/debian/dists/squeeze/pvetest/binary-amd64/dab_1.2-4_all.deb
```

Both `ftp://` and `http://` work for the **wget** URLs above.

'dab' sample templates

Check for current examples: Latest build files: https://git.proxmox.com/?p=dab-pve-appliances.git;a=summary

Older examples are on ftp://download.proxmox.com/sources/

```
wget http://download.proxmox.com/sources/dab-pve-appliances_2011-09-13.tar.gz
```

Unpack with tar:

```
tar -zxvf dab-pve-appliances_2011-09-13.tar.gz
```

'dab' manual pages

```
dab(1)                    User Contributed Perl Documentation                    dab(1)

NAME
       dab - Debian OpenVZ Appliance Builder

SYNOPSIS
       dab command [OPTIONS]
       dab init
           Downloads the package descriptions form the repository. Also truncates the
           "logfile".

       dab bootstrap
           Bootstrap a debian system and allocate a temporary container (we use IDs
           90000 and above).

           --exim
               Use exim as MTA (we use postfix by default)

           --minimal
               Do not install standard packages.

       dab veid
           Print used container ID.

       dab basedir
           Print container private directory.

       dab packagefile
           Print the appliance file name.

       dab install pkg ...
           Install one or more packages. pkg can also refer to a file named
           "xyz.pkglist" which contains a list of packages. All dependencies are
           automatically installed.

       dab unpack pkg ...
           Unpack one or more packages. pkg can also refer to a file named "xyz.pkglist"
           which contains a list of packages. All dependencies are automatically
           unpacked.

       dab exec CMD ARGS
           Executes command CMD inside the container.

       dab enter
           Calls "vzctl enter CTID" - this is for debugging only.

       dab task mysql
```

```
            Install a mysql database server. During appliance generation we use "admin"
            as mysql root password (also stored in /root/.my.cnf).

            --password=XXX
                Specify the mysql root password. The special value "random" can be use to
                generate a random root password when the appliance is started first time
                (stored in /root/.my.cnf)

            --start
                Start the mysql server (if you want to execute sql commands during
                appliance generation).

        dab task postgres
            Install a postgres database server.

            --version=XXX
                Select Postgres version. Posible values are 7.4, 8.1 and 8.3 (depends on
                the selected suite).

            --start
                Start the postgres server (if you want to execute sql commands during
                appliance generation).

        dab task php
            Install php5.

            --memlimit=i
                Set the php memory_limit.

        dab finalize
            Cleanup everything inside the container and generate the final appliance
            package.

            --keepmycnf
                Do not delete file "/root/.my.cfg" (mysql).

        dab list
            List installed packages.

            --verbose
                Also print package versions.

        dab clean
            Remove all temporary file and destroy the used OpenVZ container.

        dab dist-clean
            Like clean, but also removes the package cache (except when you specified
            your own cache directory in the config file)

DESCRIPTION
        dab is a script to automate the creation of OpenVZ appliances. It is basically a
        rewrite of debootstrap in perl, but uses OpenVZ instead of chroot and generates
        OpenVZ templates. Another difference is that it supports multi-stage building of
        templates. That way you can execute arbitrary scripts between to accomplish what
        you want.

        Furthermore some common tasks are fully automated, like setting up a database
        server (mysql or postgres).

        To accomplish minimal template creation time, packages are cached to a local
        directory, so you do not need a local debian mirror (although this would speed up
        the first run).

        See http://pve.proxmox.com/wiki/Debian_Appliance_Builder for examples.

        This script need to be run as root, so it is not recommended to start it on a
        production machine with running container. Proxmox VE (http://pve.proxmox.com) is
        the preferred environment, because it is able to log the console output when a
        container starts. You wont be able to detect errors during container startup when
        running on standard OpenVZ. So many people run Proxmox VE inside a KVM or VMWare
        64bit virtual machine to build appliances.
```

All generated templates includes an appliance description file. Those can be used to build appliance repositories.

CONFIGURATION

Configuration is read from the file "dab.conf" inside the current working directory. The files contains key value pairs, separated by colon.

Suite: etch|lenny|squeeze|hardy|intrepid|jaunty
  The Debian or Ubuntu suite.

Source: URL [components]
  Defines a source location. By default we use the following for debian:

```
Source: http://ftp.debian.org/debian SUITE main contrib
Source: http://security.debian.org SUITE/updates main contrib
```

  Note: SUITE is a variable and will be substituted.

  There are also reasonable defaults for Ubuntu. If you do not specify any source the defaults are used.

Depends: dependencies
  Debian like package dependencies. This can be used to make sure that speific package versions are available.

CacheDir: path
  Allows you to specify the directory where downloaded packages are cached.

Mirror: SRCURL => DSTURL
  Define a mirror location. for example:

```
Mirror: http://ftp.debian.org/debian => ftp://mirror/debian
```

All other settings in this files are also included into the appliance description file.

Name: name
  The name of the appliance.

  Appliance names must consist only of lower case letters (a-z), digits (0-9), plus (+) and minus (-) signs, and periods (.). They must be at least two characters long and must start with an alphanumeric character.

Architecture: i386|amd64
  Target architecture.

Version: upstream_version[-build_revision]
  The version number of an appliance.

  =item: Section: section

  This field specifies an application area into which the appliance has been classified. Currently we use the following section names: system, admin, www

Maintainer: name <email>
  The appliance maintainer's name and email address. The name should come first, then the email address inside angle brackets <> (in RFC822 format).

Infopage: URL
  Link to web page containing more informations about this appliance.

Description: single line synopsis
  extended description over several lines (indended by space) may follow.

Appliance description file

All generated templates includes an appliance description file called

```
/etc/appliance.info
```

this is the first file inside the tar archive. That way it can be easily

```
       exctracted without scanning the whole archive. The file itself contains
       informations like a debian "control" file. It can be used to build appliance
       repositories.

       Most fields are directly copied from the configuration file "dab.conf".

       Additionally there are some auto-generated files:

       Installed-Size: bytes
           It gives the total amount of disk space required to install the named
           appliance. The disk space is represented in megabytes as a simple decimal
           number.

       Type: type
           This is always "openvz".

       OS: [debian-4.0|debian-5.0|ubuntu-8.0]
           Operation system.

       Appliance repositories usually add additional fields:

       md5sum: md5sum
           MD5 checksum

FILES
       The following files are created inside your working directory:

        dab.conf          appliance configuration file

        logfile           contains installation logs

        .veid             stores the used container ID

        cache/*           default package cache directory

        info/*            package information cache

AUTHOR
       Dietmar Maurer <dietmar@proxmox.com>

       Many thanks to Proxmox Server Solutions (www.proxmox.com) for sponsoring this
       work.

COPYRIGHT AND DISCLAIMER
       Copyright (C) 2007-2012 Proxmox Server Solutions GmbH

       Copyright: dab is under GNU GPL, the GNU General Public License.

       This program is free software; you can redistribute it and/or modify it under the
       terms of the GNU General Public License as published by the Free Software
       Foundation; version 2 dated June, 1991.

       This program is distributed in the hope that it will be useful, but WITHOUT ANY
       WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A
       PARTICULAR PURPOSE.  See the GNU General Public License for more details.

       You should have received a copy of the GNU General Public License along with this
       program; if not, write to the Free Software Foundation, Inc., 51 Franklin St,
       Fifth Floor, Boston, MA 02110-1301, USA.

1.2                                   2012-12-31                                dab(1)
```

Build Virtual Appliances

We published several examples to show how to work with 'dab'. To start, just build a first OS template based on our examples. If you understand the basics, you can start customizing your own appliance (based on a standard OS template).

Basic recommendations

- Choose the best suitable base OS (squeeze)
- Choose the right architecture (i383 or amd64)
- Always try to use deb packages for applications (instead of tar.gz files)
- Pre-configure applications as much as possible (PHP, Apache, MySql, Postgresql, etc.) - based on the recommendations of the software
- Use unique keys and passwords (auto generate them during first start-up)
- Define the update procedure for users running the appliance in production: if you use Debian packages, update is easy (apt-get update)
- Contact the Proxmox support team for help Proxmox Support Forum
- And finally: Get certified! See Certification Overview

Debian 6 Standard (Squeeze, i386)

This example shows the basic steps when creating appliances with 'dab'. First step is to create a temporary working directory and cd into that directory (or just cd into the previously un-tared examples):

```
mkdir debian-6.0-standard
cd debian-6.0-standard
```

Now you need to create/edit the configuration file called 'dab.conf':

```
Suite: lenny
Architecture: i386
Name: debian-6.0-standard
Version: 6.0-1
Section: system
Maintainer: Proxmox Support Team <support@proxmox.com>
Infopage: http://pve.proxmox.com/wiki/Debian_6.0_Standard
Description: Debian 6.0 (standard)
 A small Debian Lenny system including all standard packages.
```

Next step: Download the available package lists:

```
dab init
```

Start the creation process with:

```
dab bootstrap
```

Finally build the final appliance package with:

```
dab finalize
```

Thats all. Simply upload the final template to your Proxmox VE server (/var/lib/vz/template/cache/) and test what you created. Sure, always check if everything went well before releasing a template. For this purpose all output is logged to a file called 'logfile'.

When finished you may want to cleanup your working directory with:

```
dab clean
```

or use 'dist-clean' if you also want to erase the package cache directory:

```
dab dist-clean
```

You can automate above steps using make. Simply create a file called '`Makefile`' (here is an example):

```
BASEDIR:=$(shell dab basedir)

all: info/init_ok
        dab bootstrap
        dab finalize

info/init_ok: dab.conf
        dab init
        touch $@

.PHONY: clean
clean:
        dab clean
        rm -f *~

.PHONY: dist-clean
dist-clean:
        dab dist-clean
        rm -f *~
```

To start template creation you can now simply type:

```
make
```

and

```
make clean
```

to cleanup the directory.

The **Build your first DAB Appliance Template** page shows how to practically setup **DAB** and get along building templates.

Other OS sample Appliances

These example OS templates are included in the dab-pve-appliances

- Debian 6.0 Standard (i386)
- Debian 6 Standard (amd64)

Sample Application Appliances

These example application templates are included in the dab-pve-appliances

- Drupal (Debian Squeeze, i386)
- MediaWiki (Debian Squeeze, i386)
- Joomla! (Debian Squeeze, i386)
- Wordpress (Debian Squeeze, i386)
- SugarCRM (Debian Squeeze, i386)
- RT Request Tracker (Debian Squeeze, i386)

Howto certify a Virtual Appliance

See Certification Overview

Get support

Get support including commercial support offering concerning 'dab'.

Related Forum Posts

- ATUIN's generic DAB File Template
- Oct 2014 mods to DAB.pm based on the DAB Git Repo - Cache

Categories: Virtual Appliances | HOWTO