PALMS: Programs, Applications and Libraries Management and Setup

Introduction

PALMS is a system to install and manage software in a CVMFS repository such as OASIS. It simplifies the packaging and installation of a software in different versions for different platforms and it helps users to setup the correct and desired environment (applications and libraries). With OASIS and PALMS, OSG can therefore offer not only a central location to install and store software, but also tools and procedures for VO librarians (software managers) to install and maintain the software. And once the software is distributed, PALMS will help VO users access the software. By using PALMS the VOs will follow best practices and common guidelines set by OSG and enforced by PALMS.

PALMS is a software and a set of procedures that OSG may offer to smaller VOs together with OASIS to assist with the distribution of VO application software. This service would not be mandatory, but we believe would complement greatly OASIS in significantly lower the barrier in becoming an OSG VO.

The requirements outlined in this document cover what we believe is needed for the first version of PALMS. We highlight requirements that may be done in the future.

Background

OSG has always valued the ability to provide a consistent environment to ease the task of researchers using all its resources. As mentioned in the OASIS documents [1,2], OSG has always encouraged the concept of software portability: a user's scientific application should be able to run in as many operating system environments as possible. However, the heterogeneity of the systems and the increasing size of a scientist's software stack make this difficult.

Also in the Linux and Open Source communities we start to see different approaches, in part thanks to the needs of Cloud service providers, in part by the comparison with other OSes like Windows and Mac OS X. The now traditional system using native packages is well supported with management tools taking care of package management and dependencies. This approach has well known advantages like easy access to source files, centrally distributed binary packages, and maintaining a coherent system where the applications are seen as part of the distribution. On the other side it requires administrative privileges to install

new software, it provides a single environment on each host, difficult portability across different base systems, little isolation for the applications and a single application version for each operating system version.

Since its first release, the OSG CE has provided a directory to VOs, referred to as “OSG APP” (for the job environment variable, $OSG_APP, that points to its runtime location), which is used for application software installations. The recently deployed OSG Application Software Installation Service (OASIS) addresses some shortcomings of the previous solution like the absence of enforcement or allocation mechanisms, relying on the sites to provide consistency and mostly the necessity to install the software on all sites separately. With OASIS the VO librarians are be able to login and install their application software in a central location and OSG is responsible for distributing the VO software to as many sites as possible.

However, even with this new capability there are open problems:

- VO librarians need more guidance to install the software. The current Linux software distribution philosophy, adopted also by OSG Software, is to provide native packages that are tailored to the platform and take care of dependencies. OASIS provides a file system propagated to OSG resources but librarians would have to build and install a relocatable VO software stack and include support for multiple platforms and often multiple software versions.
- Researchers would benefit from more help to access the software from the OSG resources. Once the jobs land on a site, the user’s job must easily select the correct software version for the platform and the job requirements.

Requirements for PALMS v1

PALMS requires the availability of OASIS or at least a setup including access to a CVMFS server to write in a central location and read from ideally all OSG sites.

For librarians (software managers) installing and managing the VO software:

- Help packaging an application and deploying it on OASIS (or CVMFS)
- Allow installs, updates and removals of applications and libraries
- Allow distribution of multiple versions for distinct platforms
- Allow distribution of multiple versions for the same platform
- Does not require privileged access (root) on the OASIS server or OSG resources
- Help adapting and installing native packages

For researchers using the VO software on OSG:

- Help selecting the correct software version for the platform
- Provide a default software version but allow choosing a different one
- Setup the correct environment for the user shell
- Works automatically with different shells
- No performance degradation compared with plain use of OASIS

Possible Future Requirements

For librarians (software managers) installing and managing the VO software:

- Can manage and solve dependencies and conflicts
- Help adapting and installing native packages

For researchers using the VO software on OSG:

- Help locating all the installed VO software

Technologies and State of The Art

PALMS aims to simplify the installation of the software and allow distinct users to see different environments: all the components that they or the underlying platform need, e.g. the desired combination of programs and libraries in the desired version, and no interferences with and from other components.

In the traditional approach each system deployment has a single status where the software installed is managed centrally and all users and programs see the same system. Alternatives range from changes in the environment, to isolation of processes, file system and networking, to full blown virtual machines. The rest of this section will first describe some of the underlying technologies that facilitate these different approaches and then it will survey some of the existing systems.

The runtime environment determines the available executables and the libraries loaded dynamically by the programs. In Linux this can be controlled in many ways: by changing environment variables like PATH, LD_LIBRARY_PATH, PYTHONPATH and other application paths; by setting aliases; by setting RUNPATH and RPATH at compile time.

Application virtualization [6] is a software technology that encapsulates application software from the underlying operating system on which it is executed. A fully virtualized application is not installed in the traditional sense, although it is still executed as if it were. The application behaves at runtime like it is directly interfacing with the original operating system and all the resources managed by it, but can be isolated or sandboxed to varying degrees. Here are some enabling

technologies, providing different levels of isolation in different system components:

- Overlay file systems, like UnionFS (Union File System) or aufs (Another Union File System), allow to view as a single file system multiple file system layers stacked on top of each other, some of which may also be read only or provide copy on write. Overlay file systems are extensively used in the deployments from read only system images like Live DVDs and support reliably hundreds of layers
- Namespaces are capabilities of the Linux kernel that allow isolating processes. pid namespaces create a hierarchy where processes in parent namespaces can see and affect children while the ones down in the hierarchy cannot see processes in parent or sibling namespaces. net namespaces have different network interfaces connected using virtual switches. ipc namespaces for independent inter process communication. mnt namespaces to allow different sets of mounted file systems and root directories (more generic than chroot). Uts namespaces to let the processes see different hostnames.
- Control groups, or “cgroups”, are a set of mechanisms to measure and limit resource usage for groups of processes
- lxc, a set of convenience scripts to simplify the creation of Linux Containers, a technology providing resource management through cgroups and resource isolation through namespaces and other mechanisms.

A full virtualization (hardware virtualization) provides full-blown Virtual Machines (guests), sometime emulating up to the hardware, sometime sharing hardware access via para-virtualization. Virtual Machines (VM) allow a completely independent environment, simulating enough hardware to allow an unmodified "guest" OS. There are many virtualization software such as VirtualBox, Xen, QEMU, KVM and there are tools like Vargant[4] that ease the process of building VMs. OSG could consider hosting on OASIS one or more VM images for each VO but the image files would likely be very big and virtualization tools would have to be installed on all OSG sites, which is beyond PALMS requirements

Here after are discussed some technologies, guidelines and programs that provide different level of flexibility and virtualization in the deployment of software.

RPM (Red Hat Package Manager) is well known and used to package a lot of programs, including all OSG software. RPM has the ability to give users some latitude in deciding where packages are to be installed on their systems. However, packages must be built to allow that. These packages are called relocatable RPMs [3] and pose additional constraints to both the package builder and the software. Relocatable RPMs are a step in the right direction but still present some inconveniences: they may be difficult to build (depending on the application), once installed the path cannot be changed, they create a directory

tree that may cause duplication and inefficiencies, and they don't solve the problem of supporting multiple versions of the same software.

Docker[5] is an open-source engine which automates the deployment of applications as highly portable, self-sufficient containers which are independent of hardware, language, framework, packaging system and hosting provider. Containers are lighter to deploy than virtual machines but Dockers itself would have to be installed and it supports only the latest Ubuntu Linux distribution so it is not something that could be deployed on OSG. Anyway the mnt namespace used by Docker could be useful in PALMS if aufs is an acceptable requirement.

Zero Install (0install) [8] is a decentralized cross-platform software installation system intended to complement, rather than replace, the operating system's package management. Its packages are portable applications and never interfere with those provided by the distribution. 0install uses mechanisms similar to CVMFS to propagate the software from the publishing sites to the host where they are run. Generating portable software that can be deployed via Zero Install still requires an effort similar to relocatable RPMs. Anyway few packages are available and 0install provides a useful comparison for PALMS.

RUNZ [9] is another framework for portable applications. This one is packaging all the necessary components inside a ".runz" file that contains a file system image. ".runz" files are very similar to the ".exe" files in Windows or the Application directories in Mac OS X. RUNZ works only on Ubuntu 8 or 9 and the project seems abandoned. It is another project that could serve as comparison with PALMS.

The PortableLinuxApps project [10] is another portable application projects developed on Debian Linux. This one provides a specification for the AppImage, the image file used to distribute software. It provides also AppImageToolkit, a series of tools that help with the building and packaging of portable images. Although developed for ".deb" packages they could probably be adapted also for RPM packages and become part of the toolset available in PALMS.

ATLASLocalRootBase is a successful example of centrally managed portable software. It is used by ATLAS, one of OSG VOs, and provides a consistent "look and feel" to ATLAS users on any platform or site where it is installed. It provides a set of carefully bundled and tested applications that can be selected by the user. This approach is very specific and not scalable but it is an useful comparison for PALMS.

Pilot project

The pilot project would include the development of the software, both the part handling software management and the part handling the job environment; the

documentation of the procedures for librarians; as an initial demonstrator, serve as software librarian for the OSG VO and deploy an initial set of applications.

The librarian tasks would include activities like maintaining in OSG OASIS a set of packages for the community, decide (with input from OSG VOs and the CIC) which software is in the library, install and update the packages, provide instructions/documentation on available packages.

Project milestones:

- PALMS project proposal (this document) – May 12
- Rapid prototyping of the PALMS v1 software in four 2 week cycles (software deliverable, evaluation, new coding and documentation) starting June 3
- Delivery of PALM v1 at the and of the four cycles – July 15

Resources:

- Marco Mambelli – project lead and main developer
- Rob Gardner – project supervision
- Help of friendly users in the OSG production and CIC communities that may take advantage from early adoption. Lincoln Bryant (UC3 and MWT2) expressed interest in the project.

References

[1] OASIS Requirements – OSG Blueprint meeting, April 2012 https://twiki.grid.iu.edu/twiki/pub/Blueprint/BlueprintFNALApril2012/OASIS_requirements.docx

[2] Proposal for OSG Application Software Installation Service (OASIS) - http://www.usatlas.bnl.gov/~caballer/files/oasis/oasis.pdf

[3] Relocatable RPMs from the Maximum RPM guid - http://www.rpm.org/max-rpm/ch-rpm-reloc.html

[4] Docker https://github.com/dotcloud/docker

[5] Vagrant - http://www.vagrantup.com/

[6] Application virtualization as defined in Wikipedia - http://en.wikipedia.org/wiki/Application_virtualization

[7] PortableLinuxApps - http://portablelinuxapps.org/

[8] Zero Install – http://0install.net/

[9] RUNZ framework - http://hacktolive.org/wiki/RUNZ_framework

[10] PortableLinuxApps - http://portablelinuxapps.org/

[11] ATLASLocalRootBase - https://twiki.atlas-canada.ca/bin/view/AtlasCanada/ATLASLocalRootBase