Package management

Deepak Ravi

Overview

- Package management
  - Different kinds of package management systems
  - Lab: Configure client to use a package repository
  - Lab: What is in a packagename.deb?
  - Lab: Create your own deb file
  - Lab: Create your package repository
  - Lab: Install Samba, cups, and configure them.

Package Management

Need for package?

- Our software:
  - bin/workshop
  - bin/bash
  - lib/libxyz.so
  - doc/documentation.txt
  - etc/workshop.cfg
- How to distribute the packages?
  - One simple way: Just zip and ship it!
  - And client just unzip it to some folder and use it!
- Issue?

Need for package?

- Our software:
  - bin/workshop
  - bin/bash
  - lib/libxyz.so
  - doc/documentation.txt
  - etc/workshop.cfg
- How to distribute the packages?
  - One simple way: Just zip and ship it!
  - And client just unzip it to some folder and use it!
- Issue?
  - Every package has it's own version of libraries, and its dependency
  - Large file size
- How to fix this issue?

Need for package?

- Our software:
  - bin/workshop
  - bin/bash
  - lib/libxyz.so
  - doc/documentation.txt
  - etc/workshop.cfg
- Approach 2:
  - Don't ship dependencies which are shared.
  - Only allow one version of package at a time.
  - Just mention that we need another package in some special file, say control.
  - Use a tool to ensure that dependencies are met or not.
- Issue?
  - We need a tool to ensure dependency are met (dpkg)
  - To install a package user has to manually download dependency packages
  - Only one version of package at a time.

Need for package:

- Issue?
  - Only one version of package at a time. (ignored by debian. solved by nix)
  - We need a tool to ensure dependency are met (dpkg)
  - To install a package user has to manually download dependency packages
- How to solve issue of manually downloading dependency packages..
  - New tool which will maintain a list of
    - all available packages that can be downloaded
    - and their dependencies
    - and from where they can be downloaded
  - If a user wants to install a package:
    - recursively download the dependencies
    - download the requested package
    - ask dpkg to install downloaded files.
- Issue? Constraints. User wants to say need gcc ($>= 6.0$)

Need for package:

- Issue?
  - Only one version of package at a time. (ignored by debian. solved by nix)
  - We need a tool to ensure dependency are met (dpkg)
  - To install a package user has to manually download dependency packages
- How to solve issue of manually downloading dependency packages..
  - New tool which will maintain a list of
    - all available packages that can be downloaded
    - and their dependencies
    - and from where they can be downloaded
  - If a user wants to install a package:
    - recursively download the dependencies
    - download the requested package
    - ask dpkg to install downloaded files.
- Issue? Constraints. User wants to say need gcc ($>= 6.0$)
- Implement constraint solving in apt.

Summary so far:

- deb: file format
  - Why a new format? We need to specify the dependencies as well.
- dpkg: tool
  - To ensure the dependencies are met.
  - Install the deb
  - To manage the installed files(so no conflict, and can be uninstalled)
- apt: tool
  - Knows a list of all packages
  - Can download dependencies automatically

How to specify the list of available packages

- /etc/apt/sources.list
  - format:
    - deb uri suite/distribution components
  - ex:
    - deb http://httpredir.debian.org/debian/ unstable main
    - deb http://localhost/debian unstable main
  - doc:
    - see man sources.list

apt

- To update the list of packages:
  - sudo apt update
- Search for a package:
  - apt search keyword
- How to install a package:
  - sudo apt install package
- Download a deb file:
  - apt download package
  - sudo apt -d install package

Install deb directory

- sudo dpkg -i deb_file

format of deb:

- deb: ar format (similar to zip)
  - data.tar.gz
  - control.tar.gz
  - debian-binary
- ar t deb
- ar x deb

Create your own deb:

- usr/bin/workshop : your software executables
- DEBIAN/control : To mention dependency

```
Package: workshop
Version: 1.0-1
Section: base
Priority: optional
Architecture: i386
Depends: bash (>= 1.0)
Maintainer: Deepak Ravi <deepak.ravi@gmail.com>
Description: A simple tool to print hello
 This is our first debian package.
```

- dir:
  - workshop-1.0-1/usr/bin/workshop
  - workshop-1.0-1/DEBIAN/control
- dpkg-deb -b ./workshop-1.0-1

Lab:

- create different deb files
  - mention unmet constraints
  - create two dependency file one depending on another
  - What if two deb files try to overwrite same file?

Create your own repository:

- Tool: reprepro
- create config file for reprepro: conf/distributions

```
Origin: Debian
Label: Debian-All
Suite: unstable
Codename: sid
Architectures: i386 amd64
Components: main non-free contrib
Description: Debian unstable
SignWith: 59BF40EC1EB1E240A908571577A92009B7A9F876
```

- man reprepro for more info
- reprepro includedeb distribution deb_file
- serve using an http server

Install samba

- Install: sudo apt install samba smbc smbclient
- Configure: edit /etc/samba/smb.conf

```
[test]
   comment = Test Public
   browseable = yes
   path = /home/deepakravi/Public
   guest ok = yes
   read only = yes
```

- Restart samba service: sudo service smbd restart
- In your file manager(nautilus),
  - Click on:connect to server
  - Enter smb://localhost

Install cups:

- Install: sudo apt install cups cups-pdf
- Print file to PDF printer
- Printed file will be saved in $HOME/PDF