Advanced Linux System Administration

Topic 4. Software management

Pablo Abad Fidalgo
José Ángel Herrero Velasco

Departamento de Ingeniería Informática y Electrónica

Index

Introduction

- Distribution alternatives:
  - **Proprietary** software:
    - Usually employs its own installation tools (automatized).
  - **Free** software:
    - From its source code (always).
    - From packages (usually).
    - From repository (depends on the distribution).
- Two simple recommendations about software installation:
  - Check before committing (**testing** and more testing). A twin system machine (virtual or real) could be an interesting option...
  - Security patches. Always try to be updated, especially in network-exposed systems.

Index

Installation from Source Code

- The source code of any Free Software is publicly available:
  - Mandatory for Free software licensing: GPL, BSD, etc.
- Advantages:
  - Optimize software for our hardware (compilation options).
  - Have more freedom concerning versions or software to install.
- Disadvantages:
  - Tough installation process (building and dependencies).
  - Easier to disorganize our system installation:
    - The installed software is not labeled in any database.
    - Recommended to make use of special directories. /usr/local/, /opt/, /usr/src/.
- Available formats:
  - Pre-built packages (tar.gz, tar.bz2…).
  - Software repositories (git, hg…).

Installation from Source Code

- Installation Steps:
  - **[previous-1]** Install **compilation**/building tools:
    - gcc, g++, autotools, cmake, scons…
  - **[previous-2]** Install **dependencies:**
    - External libraries (.so, .a) and other tools.
  - **[1] Download** the software (format .tar.gz, .tar.bz2):
    - cd /opt/prebuilds && wget http://www.python.org/ftp/python/2.7.6/Python-2.7.6.tgz.
  - **[2] Uncompress:**
    - tar –xzvf Python-2.7.6.tgz.
  - **[3]** Read README/INSTALL. Pre-configure the Makefiles (paths) and **resolve** the possible **dependencies** (previous software):
    - cd /opt/prebuilds/Python-2.7.6/ && ./configure –prefix=/usr/local/.
  - **[4] Compile** the packet and install it in a different directory (/usr/local/):
    - Make –j <num cores> && make install.

Installation from Source Code

- Not all free software is available through .tar.gz packages.
- **DCVS** Systems (Distributed Concurrent Versioning Systems) are becoming the standard for this labor:
  - Distributed versioning systems (avoid dependency on the server). Employed for **collaborative software** projects (like the linux kernel).
  - Software versions (code modifications/fixing/improvements) maintained through a revision tree.
  - Examples: **git**, **mercurial** (A nice starting point: https://try.github.io/).
  - How is download performed? (Example with Xen, virtualization sw):
    - Apt-get update && apt-get install git.
    - Git clone git://xenbits.xen.org/xen.git.

Index

Installation from Packages

- A software package contains:
  - The source code or compiled binaries.
  - Scripts for pre and post-installation (location control, dependencies…).
- Advantages:
  - Unified and organized administration of installed software (Database).
  - Simplifies installation process (no compiler required, pre/post install…).
- Source (who provides the package?):
  - If the developers of a UNIX/Linux distribution support the software, they usually provide it pre-packaged.
  - The developer can also provide the package.
- Each distribution has its own format:
  - RedHat and derivatives (Suse, Centos, Fedora…): .rpm.
  - Debian and derivatives (Ubuntu): .deb.

Installation from Packages

- .deb packages (Debian):
  - Binary package: binary, configuration file, man pages, copyright…
  - Source package:
    - File .dsc: package descriptor.
    - File .orig.tar.gz: original code, no modification.
    - File .diff.gz: modifications performed by Debian to the original code.
  - Naming: [name]_[version-code]_[Debian-revision]_[arch].deb.
  - More info:
    - https://debian.org/doc/manuals/debian-faq/ch-pkg_basics.en.html.
- Associated files and directories:
  - /etc/dpkg/: configuration file (dpkg.cfg).
  - /var/lib/dpkg/: information about available/installed packages.

Installation from Packages

- Command **dpkg:** packet management in Debian:
  - Format: dpkg --<options> [packet]:
    - Option **–i (--install):** install a downloaded package.
    - Option **–r (-P purge):** uninstall a package (purge also removes configuration files).
    - Option **–c:** show the contents of the package.
    - Option **–b (--build):** compile a package if it's source code.
    - Option **–l(--list):** list all the packages available. The second character shows the status of the package: [i-installed], [n-not installed], [c-only configuration files]…
  - Example: installation of python 2.7 for Debian Wheezy (precompiled amd64):
    - `wget http://ftp.us.debian.org/debian/pool/main/p/python2.7/python2.7_2.7.3-6_amd64.deb`
    - `dpkg –i python2.7_2.7.3-6_amd64.deb`
  - In most cases it is not so easy (example GDM3):
    - `wget http://ftp.us.debian.org/debian/pool/main/g/gdm3/gdm3_3.4.1-8_amd64.deb`
    - `dpkg: dependency problems prevent configuration of gdm3:`
    - `gdm3 depends on libaccountsservice0 (>= 0.6.8); however:`
    - `Package libaccountsservice0 is not installed.`
    - …

More than 30 dependencies that must be resolved manually.

Index

Installation from Repository

- Debian introduced the use of an autoçmatized tool to simplify the installation process:
  - Automatic maintenance of versions.
  - Automatic resolution of package dependencies.
- **APT: Advanced Packaging Tool:**
  - Connects transparently the package management tool (dpkg) with external repositories.
  - Searches in the repositories, downloads the package, manages dependencies, installs and finally configures (all made transparent to the used).
  - Management/Resolution of dependencies (pre-installation).
- Other distributions have their own package management tools:
  - Yum (Red-Hat), Yast2 (Suse).

Installation from Repository

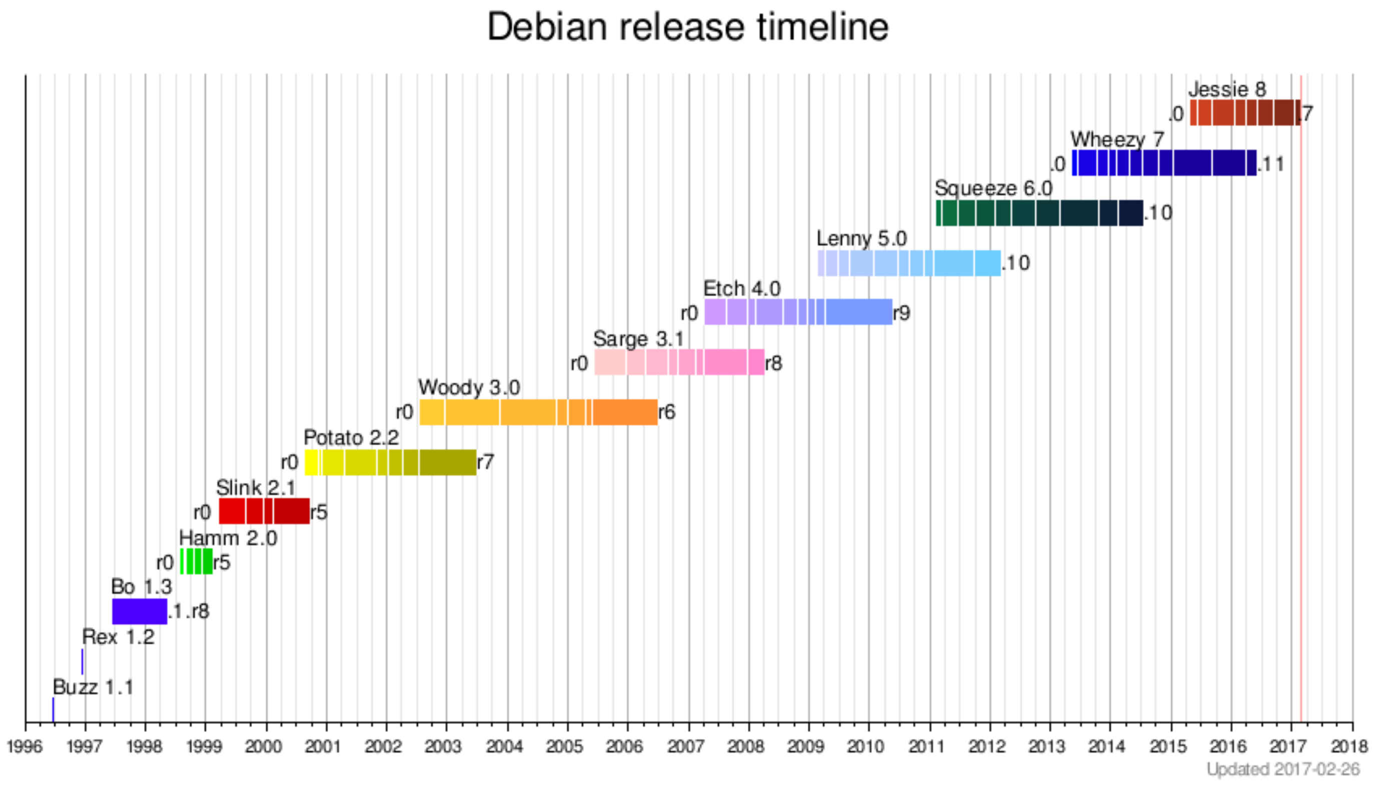
- Command **apt-get:** command-line interface for APT:
  - Format: apt-get <option> [package]:
    - Option **update:** update the list of known packages. (regular use recommended).
    - Option **upgrade:** update all the packages in the system.
    - Option **install:** install a package and all the dependencies.
    - Option **remove (purge):** remove a package (purge: + configuration files).
    - Option **clean:** remove the .deb files downloaded for installation.
- Cache of contents:
  - A copy of installed packages is kept in /var/cache/apt.
- Command **apt-cache**: tool for package searching:
  - Format: apt-cache <option> [word/package]:
    - Option **search:** (apt-cache search wrd) find all the packages with the word “wrd”.
    - Option **show:** shows information about a package.
    - Option **depends:** shows the dependencies of a package.

Installation from Repository

- What about the repository?: configuration in /etc/apt/**sources.lst:**
  - Syntax: [file_type] [URL] [distribution] [components]:
    - File type: can be deb or deb-src. deb indicates that the repository contains binary packages, deb-src indicates source-code packages.
    - URL: link to the repository from where packages are downloaded. (mirrors).
    - Distribution: name (alias) of the distribution (squeeze, wheezy, jessie, stretch) or kind of distribution (oldstable, stable, testing, unstable).
    - Components: 3 kinds of packages: main, contrib, non-free:
      - Main: packages that meet Debian requirements (OpenSource).
      - Contrib: contains OpenSource software but some dependencies are not OpenSource.
      - Non-free: non-OpenSource software.
    - Example:

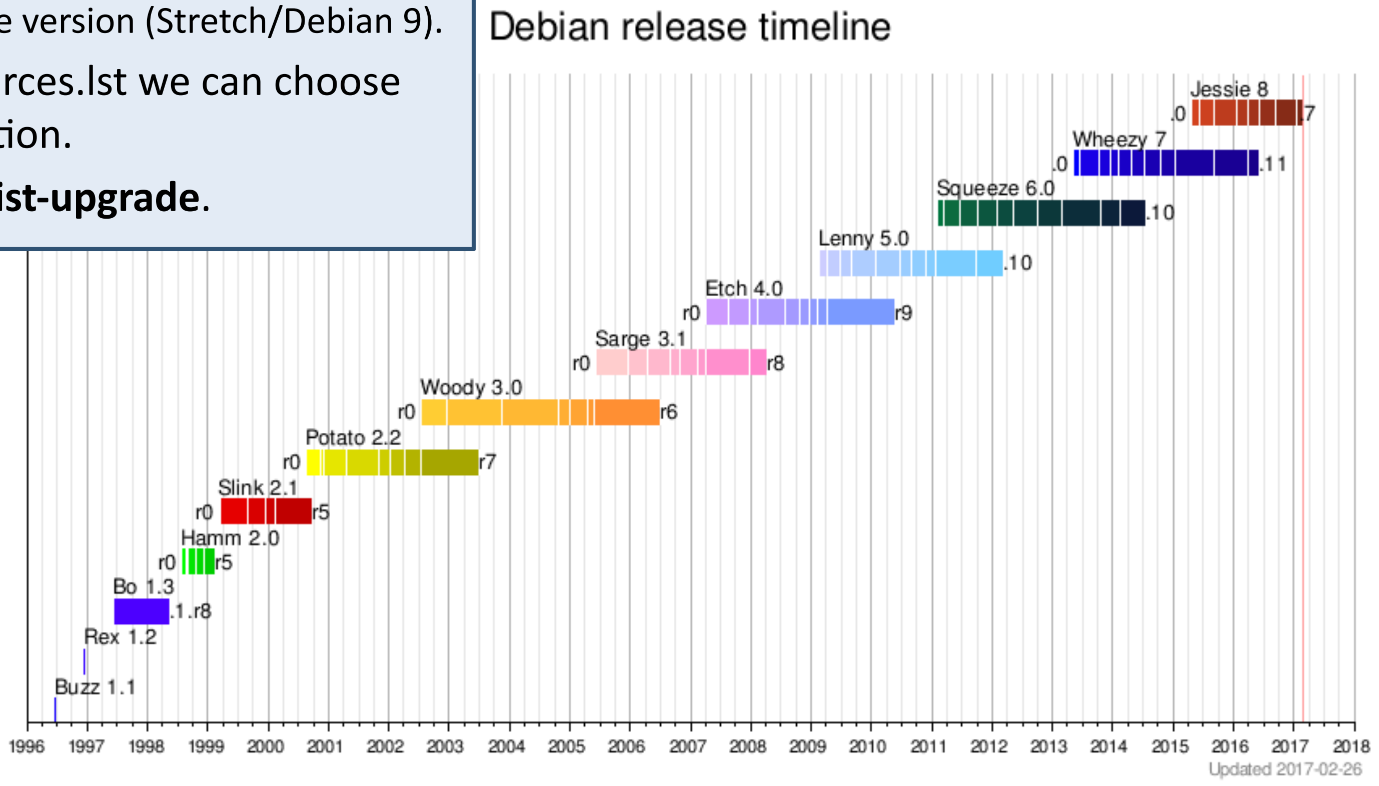
```
deb http://cdn.debian.net/debian/ wheezy main non-free contrib
deb-src http://cdn.debian.net/debian wheezy main non-free contrib

deb http://security.debian.org/ wheezy/updates main contrib non-free
deb-src http://security.debian.org/ wheezy/updates main contrib non-free
```

Installation from Repository



Installation from Repository

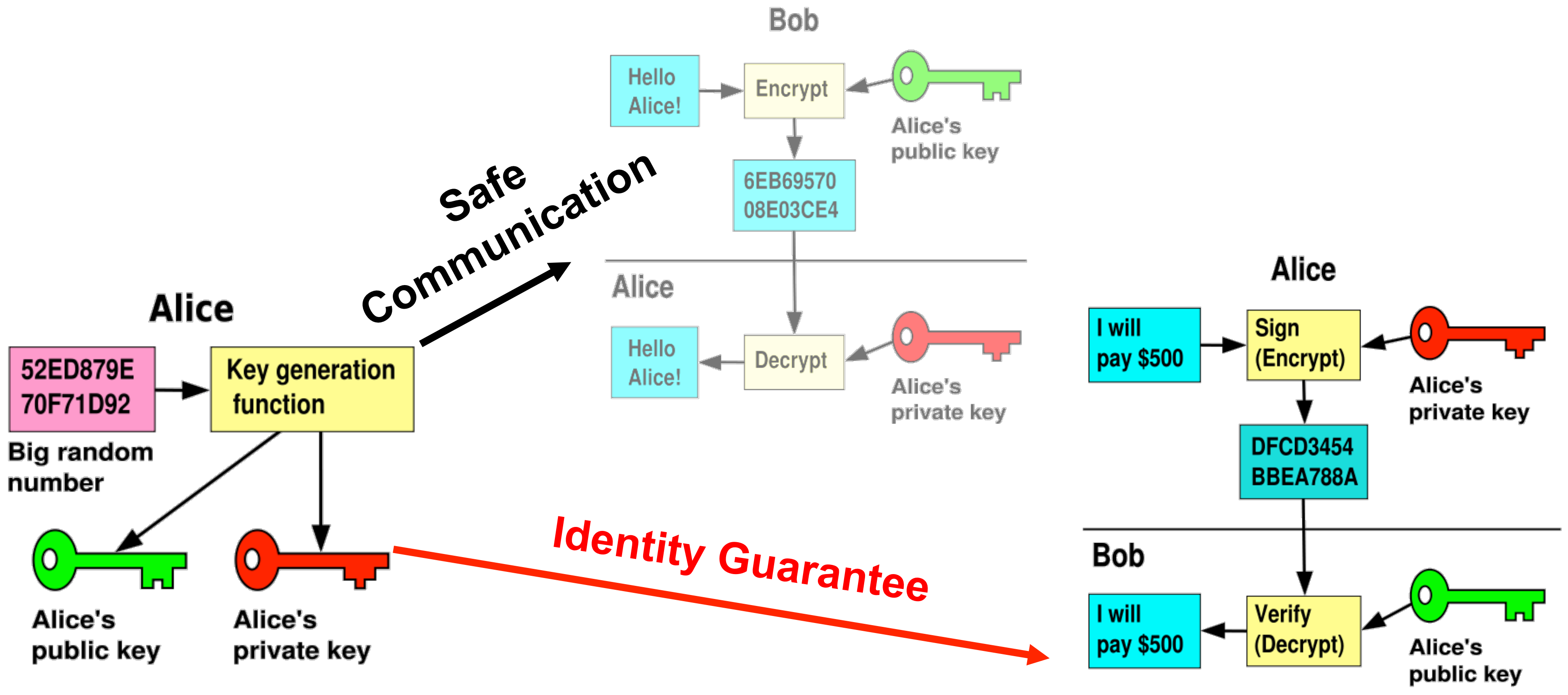
- Keeping the distribution up-to-date…:
  - Debian keeps 3 OS versions alive:
    - Oldstable: old, corresponding to old machines (Wheezy/Debian 7).
    - Stable: current stable version (Jessie/Debian 8).
    - Testing: future stable version (Stretch/Debian 9).
  - Through the file sources.lst we can choose the kind of distribution.
  - Updating: **apt-get dist-upgrade**.



Index

Security

- Previous: Asymmetric Cryptography concept:
  - Public-key – private-key.



Security

- Never blindly trust an APT repository:
  - Do not apply automatic updates (not supervised).
  - Read always before updating.
  - **Verify software authenticity**.
- Debian keeps software authenticity through asymmetric cryptography:
  - Each repository has a pair of keys. The private one remains in the repository, the public one is spread to everybody.
  - All distribution packages are signed (private signature).
  - Any package illegally modified violates its authenticity, because signing key is not known by the attacker.
  - Locally, a “keyring” with public keys is maintained to authenticate the origin of the packages.

Security

- Command **apt-key**: APT key management:
  - Format: apt-key <--keyring file> [action]:
    - Option **keyring:** indicates the key file where action is performed. Default: /etc/apt/trusted.gpg.
    - action **add filename:** add a new key to the keyring file. The key is read form filename.
    - action **list:** list all trusted keys.
    - action **del keyid:** remove the keyid key from the keyring file.
- Adding a new repository:
  - Example: VirtualBox repository (http://www.virtualbox.org):
    - Look for the public key (https://www.virtualbox.org/download/oracle_vbox.asc).
    - Download it and add to our keyring: apt-key add oracle_vbox.asc.
    - Check it is in the list of trusted keys: apt-key list.
    - Add the repository to the file sources.list:
      - Echo “deb http://download.virtualbox.org/virtualbox/debian wheezy contrib” >> /etc/apt/sources.list.
    - Now we can install VirtualBox: apt-get install virtualbox-4.3.