Gestione del software

Installazione e gestione repository

Andrea Gussoni
andrealinux1@gmail.com

Corsi Gnu/Linux Avanzati 2014

Introduzione

- Breve introduzione agli argomenti del talk
- Domande preliminari?

La Compilazione

- Pro: ottimizzazione, personalizzazione, possibile eventuale controllo sui sorgenti
- Contro: dispendiosa in termini di tempo e risorse, “frammentazione” del software nel pc, non risolve automaticamente le dipendenze
- Come risolvere?

I Pacchetti Precompilati

- I principali e più utilizzati formati
  - .deb (formato di Debian)
  - .rpm (formato di Red Hat)
- Altri formati
  - .tgz (formato di Slackware)
  - .pkg.tar.xz (formato di Arch Linux)

Principali Operazioni

- Installazione di un pacchetto
  - *dpkg -i pacchetto.deb*
  - *rpm -i pacchetto.rpm*
- Rimozione di un pacchetto
  - *dpkg -r pacchetto.deb*
  - *rpm -r pacchetto.rpm*

Ancora non basta

- Ora l'installazione non richiede la compilazione
- Possiamo facilmente gestire i programmi installati, rimuovendo all'atto della disinstallazione tutta la "sporcizia"
- Però:
  - Abbiamo bisogno dei binari per la nostra architettura
  - Le dipendenze vanno ancora soddisfatte manualmente
  - All'uscita di una nuova versione di un programma dobbiamo scaricare di nuovo il pacchetto e procedere al suo aggiornamento manualmente

I Package Manager

- Apt per gestire il formato ".deb", Yum per gestire il formato ".rpm"
- Cos'è e come funziona un repository
- Le principali distribuzioni offrono i propri repository ufficiali
  - Ubuntu ~47000 pacchetti deb
  - Debian ~37000 pacchetti deb
  - Fedora ~22000 pachheti rpm
  - OpenSuse ~40000 pacchetti rpm
- Si possono anche aggiungere dei repository "fidati" aggiuntivi al proprio sistema

Adesso Ci Siamo

- Unico “posto” dove cercare il software che ci serve
- Dipendenze automaticamente risolte
- Gestione centralizzata e automatica degli aggiornamenti
- Software compilato per la nostra architettura e verificato dai mantainer della nostra distribuzione

Apt

- “Advanced Packaging Tool”
- Usato dalle principali distribuzioni Debian Based (Ubuntu, Linux Mint)
- Esiste anche “Aptitude” per i nostalgici dell’interfaccia grafica

Comandi

- Prima di tutto dobbiamo aggiornare il database dei pacchetti
  - *apt-get update*
- Installazione
  - *apt-get install pacchetto*
- Rimozione
  - *apt-get remove pacchetto*
- Ricerca di un pacchetto nel database
  - *apt-cache search pacchetto*
- Informazioni aggiuntive sul paccheto
  - *apt-cache show pacchetto*

Comandi

- Aggiornamento dei pacchetti
  - *apt-get upgrade*
  - *apt-get dist-upgrade*
  - il primo comando esegue solo l'aggiornamento alle nuove versioni dei pacchetti disponibili senza però aggiornare i pacchetti che richiedono installare o rimuovere altri pacchetti, mentre dist-upgrade aggiorna anche questi ultimi

Comandi

- Altri comandi utili:
  - *apt-get autoremove* – elimina tutti i pacchetti non necessari al corretto funzionamento del sistema
  - *apt-get clean* – pulisce la cache (utile per liberare un po di spazio in /var)
  - *apt-get purge* – rimuove un pacchetto andando però ad eliminare anche i file di configurazione ad esso collegato
  - *apt-get --no-install-recommends* – installa un pacchetto evitando però di installare i pacchetti consigliati, ad esempio la documentazione eccetera...

Comandi

- Altri comandi utili basati su dpkg:
  - *dpkg -l pacchetto* – simile ad *apt-cache search* ma in più mostra se il pacchetto è installato o no
  - *dpkg -L pacchetto* – elenca i file installati nel sistema da un dato pacchetto
  - *dpkg -S* nomefile – cerca il file e dice a quale pacchetto appartiene

Approfondimenti – /var/cache/apt

- Apt tiene una repository locale dei pacchetti che vengono installati in /var/cache/apt, e nel caso abbiate una /var su partizione separata questa si può riempire facilmente, potete ovviare a questo con il comando citato precedentemente. Questa cache però in condizioni normali può essere utile per velocizzare le operazioni di installazione e rimozione di grandi pacchetti

Approfondimenti – unattended-upgrades

- Esiste la possibilità di far eseguire automaticamente gli aggiornamenti di sicurezza al momento della loro pubblicazione nei repository
- Molto utile per le macchine server, perchè abbiamo le vulnerabilità fixate senza il bisogno di un nostro intervento manuale
- Maggiori informazioni [1]

[1] https://wiki.debian.org/UnattendedUpgrades

Apt-File

- Utilizzo:
  - *apt-file update* – aggiorna il database del programma
  - *apt-file search nomefile* – restituisce il pacchetto a cui appartiene il file passato come argomento
  - *apt-file list nomepacchetto* – elenca tutti i file che appartengono ad un certo pacchetto

I Repository

- il file /etc/apt/sources.list

/home/andrea/Pictures/Screenshot from 2014-03-14 21:52:02.png

I Repository

- Demo

I Repository

- Possiamo aggiungere altri repository
  - se il repository si trova su launchpad basta usare:
    - *add-apt-repository ppa:utente/nome-ppa*
  - altrimenti dobbiamo aggiungere "a mano" al file sources.list l'indirizzo del repository, ricordandoci di aggiungere anche la chiave per verificare la firma dei pacchetti [2]
  - ricordarsi sempre di aggiornare il database dei pacchetti dopo aver aggiunto un repository, altrimenti il package manager non "vedrà" la nuova risorsa

[2] http://wiki.ubuntu-it.org/Repository/RigaDiComando

Yum

- Yellowdog Updater, Modified
- Package manager standard di molte distribuzione rpm-based (RHEL, CentOS, Fedora, OpenSuse)
- A differenza di Apt non richiede ogni volta l'aggiornamento esplicito del database dei pacchetti

Comandi

- Installazione
  - *yum install pacchetto*
- Rimozione (rimuove anche i file di configurazione, come *apt-get purge*)
  - *yum remove pacchetto*
- Ricerca nel database
  - *yum search pacchetto*
- Informazioni aggiuntive
  - *yum info pacchetto*

Comandi

- Aggiornamento dei pacchetti
  - *yum update*
  - *yum distro-sync*
- Altri comandi utili
  - *yum clean packages :* Pulizia della cache
  - *yum autoremove :* Rimozione dei pacchetti non più necessari

I Repository

- In questo caso i file che contengono le informazioni sui repository si trovano in */etc/yum.repos.d* ed hanno estensione ".repo"
- Un esempio : un file chiamato "example.repo" contiene al suo interno

```
[examplerepo]
name=Example Repository
baseurl=http://mirror.cisp.com/CentOS/6/os/i386/
enabled=1
gpgcheck=1
gpgkey=http://mirror.cisp.com/CentOS/6/os/i386/RPM-GPG-KEY-
CentOS-6
```

Alien

- Esiste eventualmente anche la possibilità di convertire da formato "deb" a formato "rpm" e viceversa i pacchetti tramite Alien, la sintassi è :
  - *alien --to-deb pacchetto.rpm (*da rpm a deb*)*
  - *alien --to-rpm pacchetto.deb (viceversa)*

Tutto Qui? Ni

- I package manager e i comandi che abbiamo visto in precedenza sono assolutamente sufficienti per la normale amministrazione di un server, poichè tutti i pacchetti di maggiore utilità sono inclusi nei repository, ed i manutentori di tutte le distro che abbiamo citato sono affidabili per quanto riguarda gli aggiornamenti di sicurezza
- Ovviamente esistono gli equivalenti per le distro che non abbiamo visto in dettaglio, dato che non hanno una innata orientazione server, anche se in fondo nessuno vi vieta di usare la vostra distribuzione preferita per il vostro server, in fondo il bello di Gnu/Linux è proprio questo
- Nel caso foste interessati vi invito a cercare nelle documentazione della vostra distribuzione, e come "cheatsheet" vi consiglio la "Stele Di Rosetta" che trovate sulla wiki di Arch [3]

[3] https://wiki.archlinux.org/index.php/Pacman_Rosetta

La Via Più Ripida

- Esiste anche la possibilità di compilare il software direttamente dai sorgenti (Gentoo usa questa filosofia per l'intera distribuzione)
- Solitamente è l'ultima via che viene considerata, perchè come detto in precedenza compilare da sorgenti significa non avere nessun controllo o aiuto da parte del sistema, e quindi dover sistemare eventuali dipendenze e incompatibilità a mano, oltre a dover poi rimuovere tutti i file che vengono installati se non dovessimo più avere bisogno del software
- In alcuni casi (software particolarmente nuovi o di nicchia) la compilazione da sorgenti può essere l'unica soluzione, cercherò quindi di darvi qualche indicazione di massima per indirizzarvi sulla strada giusta

I Grandi Classici

- La compilazione da sorgenti spesso viene ridotta ai seguenti comandi
  - *./configure*
  - make
  - make install
- Spesso non è esattamente così, per ogni software che si va a compilare e altamente consigliata la lettura del file README all'interno dell'archivio, che può illuminare sulla procedura da seguire, e spesso chiarisce anche le dipendenze necessarie
- Infatti generalmente il primo passo della procedura è volto proprio a verificare requisiti e dipendenze, ed in caso di errore sarete voi a dover interpretare l'output del programma per capire dove e perchè si è fermato, procedendo quindi a risolvere il problema

Checkinstall

- Permette di installare da sorgenti tenendo traccia però dell'installazione in modo da rimuovere facilmente in futuro
- *checkinstall* al posto di *make install*
- più informazioni a [4]

[4] http://wiki.ubuntu-it.org/Programmazione/Checkinstall

Domande?

Demo!

Domande?

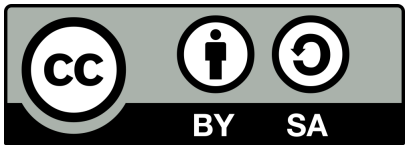
Demo!

Fonti

- Unix and Linux System Administration Handbook (Fourth Edition)
- Ubuntu-Wiki
- Debian-Wiki
- Fedora Project-Wiki

License

Grazie per l'attenzione!



http://www.poul.org