Linux e Software libero (autore: Vittorio Albertoni)

Premessa

Il mondo che, in maniera abbreviata, chiamiamo Linux si articola su tre componenti: il kernel, il sistema operativo e la distribuzione.

Linux, il nome con il quale tendiamo impropriamente a battezzare tutto, è, in realtà il kernel, il nocciolo di tutto il sistema: possiamo concepirlo come l'interfaccia tra il software e l'hardware.
E' lui stesso un software, scritto in linguaggio C inizialmente da Linus Torvalds, ma da solo non fa nulla di umanamente tangibile.

Il sistema operativo è l'interfaccia tra l'utente e l'hardware, cioè è il software che rende disponibile il kernel a ciò che vuole fare l'utente, prevedendo una serie di comandi destinati a produrre file, elaborarli, archiviarli, spostarli, ecc.
Dalla sua nascita Linux è stato integrato nel progetto di sistema operativo GNU (GNU Not Unix) di Richard Stallman ed è nato il sistema operativo propriamente chiamato GNU/Linux.

Dopo una breve fase di vita totalmente pionieristica sono nate le così dette distribuzioni, in gergo chiamate distro, per diffondere il sistema GNU/Linux offrendo anche ad utenti poco esperti, insieme al kernel Linux, il sistema operativo, più o meno legato al progetto GNU, comunque arricchito da una serie di software di produttività e da interfacce grafiche variamente concepite, destinate a facilitare il dialogo tra utente e macchina.
Red Hat Enterprise, Debian, SUSE, Fedora, Ubuntu, tanto per citare le più note e diffuse, sono distribuzioni, tutte basate sul kernel Linux e che, oltre a software derivato dal progetto GNU, possono contenere anche arricchimenti e personalizzazioni tipiche di ciascuna distribuzione.

Per notizia, rammento che anche il sistema Android utilizza il kernel Linux.

Fatta questa premessa, vediamo come si rapporta il sistema Linux al mondo del software libero, che offre una infinita varietà di programmi soprattutto e innanzi tutto destinati proprio a Linux, che è a sua volta software libero.

Rammento che il software libero ha la prerogativa di rendere disponibile il codice sorgente: da qui l'altra definizione che identifica il software libero come software open source. Il codice sorgente, cioè lo scritto da cui compilare l'eseguibile, è reso disponibile affinché chiunque possa modificarlo per renderlo più funzionale al proprio fabbisogno o per arricchirlo a vantaggio della comunità. In quest'ultimo caso deve essere reso disponibile anche il codice sorgente modificato, in modo che questo possa a sua volta essere modificato o arricchito.
Dal momento che i contributi alla produzione, al mantenimento ed al perfezionamento del software libero sono volontaristici, quasi e praticamente sempre il software libero è anche gratuito, specie se l'utente si limita ad utilizzarlo a proprio rischio e pericolo, senza chiedere assistenze o garanzie di alcun tipo.

All'opposto abbiamo il così detto software proprietario, quello di cui è disponibile, quasi sempre con acquisto a pagamento, il solo eseguibile. In questo caso il codice sorgente rimane segreto e di proprietà del programmatore, in modo che nessuno possa copiarlo e servirsene per produrre altro software.

Indice

1 Tarball

Il codice sorgente viene reso disponibile attraverso i così detti Tarball.

Si tratta di archivi compressi (generalmente file con estensione `.tar.gz`) che, oltre al sorgente, contengono uno script di configurazione dove sono elencate le dipendenze necessarie perché il programma possa funzionare. Le dipendenze riguardano le librerie che sono richiamate nel sorgente e che sono necessarie per la compilazione o software ausiliari che sono necessari per il funzionamento del programma.

Disporre del codice sorgente significa avere la possibilità di modificarlo e di ottimizzarlo per il proprio fabbisogno ma significa anche doversi far carico della compilazione dell'eseguibile.

Nel mondo Linux, dove tradizionalmente si crea il software utilizzando il linguaggio di programmazione C, il sistema operativo GNU è dotato per default di un compilatore per il linguaggio C (il compilatore GCC, GNU C Compiler) e il procedimento tradizionale per la compilazione del software e la sua installazione partendo dal tarball è il seguente:

. si copia il tarball preferibilmente nella directory `/usr/local/src`,

. posizionati con il terminale in quella directory, si estrae il tarball con il comando

```
tar -zxvf <tarball>
```

. ci si posiziona nella directory in cui si trova il tarball estratto e si dà il comando

```
./configure
```

. se lo script `configure` rileva dipendenze non soddisfatte si installano i file mancanti,

. si prosegue poi con il comando

```
make
```

. infine, come superutente, si dà il comando

```
make install
```

A questo punto il software è installato e possiamo cancellare dalla directory `/usr/local/src` il tarball e la directory in cui è stato estratto.

Praticamente con lo stesso procedimento potremmo installare un kernel Linux di versione diversa da quella che abbiamo installato o, se ne siamo capaci, modificato a nostro uso e consumo, procurandoci il relativo tarball all'indirizzo *https://www.kernel.org/*.
Nel momento in cui scrivo (aprile 2021) il tarball più aggiornato è `linux-5.11.11.tar.xz`.

Tutto ciò a dimostrazione di come, nel mondo Linux, tutto possa ridursi ai minimi termini. Addirittura è possibile sostituire il cuore del sistema operativo, il kernel, con lo stesso procedimento con il quale si installa un qualsiasi programmino.

Ma è una semplicità solo apparente.

In realtà sono cose da professionisti che, trattate da incompetenti, possono causare disastri.

Rimane da riconoscere che con questi strumenti semplici ma pericolosi il professionista ha la possibilità di

. mantenere elevata la performance di sistema attraverso ottimizzazioni e razionalizzazioni,

. personalizzare finemente il sistema,

. eseguire aggiornamenti nella maniera più rapida possibile.

Ma da qui in poi vediamo solo cose a portata di dilettante.

2 Package

Un package, in italiano pacchetto, è un file archivio contenente un insieme di file destinati a fare svolgere al computer un compito specifico: quello che si dice un programma o, con terminologia iniziata da Steve Jobs, una app.

Se installiamo un package otteniamo lo stesso risultato che otterremmo sottoponendo un tarball del codice sorgente al processo di compilazione e installazione che abbiamo visto nel capitolo precedente, senza però passare attraverso la configurazione e la compilazione.

Per installare il package utilizziamo un software, detto package manager, di cui ogni distribuzione Linux è dotata, e il package contiene anche tutti i metadati di cui il package manager ha bisogno per svolgere il suo compito.

Dal momento che ogni distribuzione ha un suo modo di organizzare le proprie risorse, esistono vari tipi di package manager ed esistono pertanto vari tipi di package, che si identificano attraverso l'estensione del file che li contiene:

. l'estensione `.deb` identifica i package per i sistemi Debian e derivati come Ubuntu e Linux Mint;

. l'estensione `.rpm` identifica i package per i sistemi Red Hat, Fedora e SUSE;

. l'estensione `.pkg` identifica i package per il sistema Arch Linux;

. l'estensione `.apk` identifica i package per il sistema Android.

Sembra tutto molto complicato, ma è il prezzo della libertà.

Nel mondo libero di Linux anche l'unica cosa invariabile, massimo comune denominatore, il kernel Linux, è personalizzabile. Figuriamoci cosa può fare la libertà su tutto il resto.

Nessuna meraviglia, quindi, se ci troviamo di fronte questa variegatura.

Come vedremo subito, comunque, tutto si semplifica una volta che siamo allocati sulla distribuzione che abbiamo scelto.

3 Repository

Una distribuzione Linux, in genere, oltre al sistema operativo, comprende una serie di programmi per svolgere i più ricorrenti compiti che richiediamo a un computer, in modo che, una volta installata, potremmo non avere bisogno di altro.

Abbiamo infatti a disposizione sicuramente un word processor, un foglio di calcolo, un browser web, qualche lettore di file multimediali, ecc.

Esistono anche distribuzioni specificamente orientate a settori di interesse, come Ubuntu Studio e AVLinux per la mutimedialità, Edubuntu per la scuola, ecc.

Ma il mondo del software libero ci offre una miriade di altri programmi, anche per svolgere i compiti più impensati e ciascuna distribuzione ci mette a disposizione sul web un repository (deposito, ripostiglio) che contiene i package che possono funzionare con il proprio sistema operativo, in modo che possiamo arricchire il nostro computer con ulteriori funzionalità, ivi compresa quella di produrre noi stessi altro software con linguaggi diversi dal C, di casa per default su Linux, come Pascal, Python, Java, ecc.

L'accesso al repository è possibile attraverso il package manager della distribuzione, variamente denominato (Gestore pacchetti, Software center), che in molti casi ha una gra-

devole interfaccia utente grafica che ci presenta i vari pacchetti, suddivisi per materia, e ci consente di installare quelli che ci servono con un semplice click su un'icona.

Essendo, ovviamente, collegati a Internet in qualche secondo avremo sul nostro computer il programma contenuto nel package.

Praticamente tutto il software che troviamo nei repository lo possiamo in qualche modo trovare anche attraverso ricerche in rete, sotto forma di tarball o package. Nel caso di package dobbiamo semplicemente porre attenzione a scegliere il package adatto al nostro sistema badando all'estensione del file, secondo quanto indicato nel precedente Capitolo. Una volta scaricato il package, con doppio click sul file si avvia il package manager che provvede all'installazione.

Il vantaggio dell'installazione con ricorso al repository sta nell'avere la certezza di installare un software esente da virus e sicuramente funzionante con la nostra versione del sistema operativo.
Può infatti accadere, se abbiamo un sistema operativo vecchio di alcuni anni, che il package che troviamo in rete richieda un sistema operativo più recente.

* * *

Il sistema dei package e dei repository, come ho già detto, è abbastanza complicato, soprattutto in presenza di una tradizione nella produzione del software libero che tende a sfruttare il più possibile librerie già liberamente disponibili, in quanto a loro volta software libero, in alleggerimento del lavoro di programmazione.

Ciò crea il problema delle dipendenze, cui abbiamo accennato nei capitoli precedenti, e, aumentando la varietà di software disponibile, si corre il rischio di installare software che entrano in conflitto tra loro a motivo della condivisione di talune dipendenze.

Il tutto con il vantaggio di risparmiare qualche megabyte di memoria sul disco fisso, vantaggio che, con la capienza e il relativamente basso costo degli attuali supporti di memorizzazione, è molto relativo.

Posto che agli utenti esperti rimane sempre il modo di procurarsi il tarball con il codice sorgente e lavorare con quello, a favore degli utenti meno esperti si sono fatti strada alcuni metodi alternativi a quello del package e del repository e li vediamo nei prossimi Capitoli.

4 Snappy

Snappy è un sistema di distribuzione di software e di gestione di pacchetti realizzato da Canonical, la società che cura la distribuzione Ubuntu.

Di casa in Ubuntu e derivate, è stato presto adottato anche da altre distribuzioni (Debian, Mint, Fedora, Arch Linux, openSUSE, Red Hat, Manjaro, ecc.).

I pacchetti di Snappy, chiamati snaps, contengono il programma con tutte le librerie e le dipendenze, fatta eccezione per alcune componenti di base che sono allocate in un framework comune.
L'installazione avviene in una directory riservata, chiamata `snap`, che è un ambiente completamente separato dal resto del sistema, addirittura separato anche all'interno dello stesso ambiente. Al punto che sarebbe possibile installare sullo stesso computer la versione più recente di un programma mantenendovi installata la precedente.

In questo modo, infatti, i programmi vengono installati in totale indipendenza e si elimina il problema di possibili conflitti tra dipendenze.

Per utilizzare gli snaps occorre avere installato il framework, contenuto nel pacchetto `snapd` presente nel repository delle distribuzioni che adottano Snappy e installabile attraverso il gestore programmi della distribuzione. Chi usa Ubuntu lo trova probabilmente preinstallato: per verificarlo basta scrivere a terminale il comando `snap`.

Il sito di Snappy è all'indirizzo *https://snapcraft.io/*.
Nella pagina DOCS ▷ INSTALL THE DAEMON troviamo le istruzioni per installare `snapd` sulle distribuzioni che lo supportano.
Nella pagina STORE troviamo il catalogo dei pacchetti disponibili, suddiviso per argomenti, con esaurienti descrizioni. Cliccando sull'icona del pacchetto che ci interessa apriamo una pagina ulteriormente descrittiva che contiene il pulsante INSTALL. Cliccando su questo pulsante otteniamo le istruzioni sul comando da dare a terminale per installare il pacchetto.

Con il comando a terminale `snap list` otteniamo un listato degli snaps installati.

Con il comando a terminale `snap remove <nome_pacchetto>` disinstalliamo uno snap.

5 Flatpak

Concettualmente simile a Snappy ne è una valida alternativa che ci viene proposta dalla comunità di sviluppatori del progetto Gnome, supportata da Red Hat.

Anche in questo caso si tratta di pacchetti contenenti programma e dipendenze, fatta eccezione per l'ambiente di runtime, Gnome 3, che deve essere altrimenti presente sul computer.
Anche in questo caso l'installazione avviene in maniera separata, anche se in modo più criptico rispetto al caso Snappy.

Per utilizzare i pacchetti Flatpak occorre installare innanzi tutto il software secondo le istruzioni che troviamo all'indirizzo *https://flatpak.org/* cliccando sul pulsante GET SET UP. Nella pagina che si apre clicchiamo sul simbolo della nostra distro Linux ed eseguiamo a terminale quanto ci viene suggerito.

A piede di questa pagina, o di qualsiasi altra pagina del sito, clicchiamo sulla scritta BROWSE APPS per aprire il catalogo dei pacchetti, che ce ne mostra le icone suddivise per argomento.
Possiamo raggiungere lo stesso catalogo all'indirizzo *https://flathub.org/home*.

Per installare il pacchetto che ci interessa clicchiamo sulla sua icona e scriviamo a terminale i comandi che ci sono suggeriti nella parte inferiore della pagina che si apre.

Se non abbiamo sul computer il sistema di runtime Gnome, quando installiamo il primo pacchetto verrà installato anche questo (bene sapere che pesa circa 400 MB ed occorre qualche minuto per l'installazione).

Con il comando a terminale `flatpak list` otteniamo un listato dei pacchetti installati.

Con il comando a terminale `flatpak uninstall <nome_pacchetto>` disinstalliamo un pacchetto.

6 AppImage

Attraverso Snappy e Flatpak si semplifica notevolmente il ricorso al software libero da parte degli utenti Linux, ma è sempre richiesto di installare qualche cosa da far poi fun-

zionare per procurarsi e per utilizzare i programmi.

Con la soluzione AppImage non si deve fare assolutamente nulla, si deve semplicemente scaricare un file, che ha l'estensione `.AppImage`, collocarlo nella directory che più ci fa comodo e renderlo eseguibile con il comando a terminale `chmod 555` oppure attraverso l'attivazione dell'opzione CONSENTIRE L'ESECUZIONE DEL FILE COME PROGRAMMA che si trova nella scheda PERMESSI del gestore di file.

Il file contiene il programma e tutto quanto serve per farlo girare e per lanciarlo basta un doppio click sulla sua icona nella directory dove è collocato il file.

All'indirizzo *https://appimage.org/* troviamo l'indicazione di quali AppImage sono distribuite direttamente sui siti dei produttori di software e troviamo un link (`here`) per accedere ad un catalogo.
Cliccando su questo link accediamo a *https://bintray.com/probono/AppImages* dove abbiamo un ricco elenco alfabetico di AppImage disponibili con una breve descrizione. Cliccando sul nome della AppImage che ci interessa apriamo una pagina dedicata ad una sua più dettagliata descrizione e aprendo la scheda FILES di questa pagina troviamo il file da scaricare (quello con la semplice estensione `.AppImage`).

Altro catalogo si trova in *https://appimage.github.io/apps/*, con elenco alfabetico di AppImage disponibili. Cliccando sul nome di quella che ci interessa veniamo dirottati sul sito da cui possiamo scaricare il file `.AppImage`.

Dal momento che il file AppImage non richiede alcuna installazione, il relativo programma si toglie dal computer semplicemente cancellando il file che lo contiene.