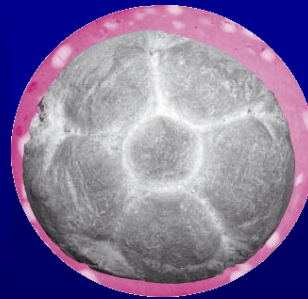


Il software libero dal produttore al consumatore

Gianni Bianchini - FLUG, AsSoLi

`giannibi@firenze.linux.it`



La mortadella e l'etica hacker
Firenze, Facoltà di Ingegneria, 5 Dicembre 2002

Copyright ©2002 Gianni Bianchini

La copia, la modifica e la redistribuzione di questo documento sono consentite nei termini della GNU Free Documentation License

<http://www.gnu.org/licenses/fdl.txt>

Produttore e consumatore: chi eran costoro?

(no, la programmazione concorrente non c'entra)

Nel software **proprietario**...

- I due ruoli sono molto ben definiti
- Le modalità di fruizione sono ristrette e limitate
(*questo ce lo avete già detto, passiamo al tecnico...*)
- Sviluppo: modello a *cattedrale*
 - ★ Gruppo di sviluppo ristretto
 - ★ Nel caso di tecnologie proprietarie, spesso vincolato al segreto

Produttore e consumatore: chi eran costoro?

Nel software **proprietario**...

- Limitate possibilità delle informazioni di ritorno (*feedback*)
 - ★ Sorgente **chiuso**
 - * *Non funziona*
 - * Richiesta nuove caratteristiche
 - * Scoperta e segnalazione bug su base esperimento
 - * In casi estremi (e spesso non tanto legali) *reverse engineering*
 - ★ Sorgente **aperto**
 - * Maggiore flessibilità data dalla possibilità di esame ed auditing del codice (fondamentale in alcune applicazioni)
 - * Il *futuro* del lavoro del consumatore è soggetto all'approvazione (e spesso all'appropriazione) da parte del produttore
- Creazione di alibi distorti nel software a sorgente chiuso
 - ★ Security through obscurity come alternativa ad auditing e correzione di vulnerabilità (brrr...)

Produttore e consumatore: chi eran costoro?

Nel software libero...

- I ruoli non sono affatto ben definiti, anzi...
- Modello a *cattedrale* ed a *bazaar*
 - ★ Evoluzione dall'uno all'altro modello nella realizzazione di un progetto
 - ★ Sviluppo collaborativo (traduzioni, documentazione, sistemi di gestione delle versioni e di *bug tracking*, *mailing list* di discussione)
- Potenzialità del feedback sfruttate al massimo
 - ★ Esame, modifica, distribuzione (quasi) incondizionati del codice
 - ★ *Quick fixes* e *patches*
 - ★ Reazione positiva (*boom!*)
- Un progetto può nascere dalle più svariate esigenze ed avere evoluzioni diverse ed anche imprevedibili...
- ...o morire di morte naturale (*non violenta*), ma non sempre tutto è perduto

E gli hacker che cosa c'entrano?

Si potrebbe dire in modo semplice che sono coloro che si identificano sia nel ruolo di produttore sia in quello di consumatore (*attivo!*) secondo il modello precedente

In modalità produttore

Come nasce un progetto di software libero?

- *Tradizionalmente per passione*
 - ★ Imparare (sfruttando l'esperienza di altri!)
Assai importante la fase di *ricerca*
- Per necessità...
 - ★ di fare qualcosa di nuovo (avendo già tanti *tool* di uso generale a disposizione)
 - ★ di migliorare qualcosa di già esistente (anche qui *succhiando* tutto il possibile dal lavoro pregresso)
- L'idea commerciale
 - ★ I modelli di business sul software libero sono *work in progress*
- *Libera(lizza)zione* di versioni di software proprietario
 - ★ Il caso *OpenOffice.org*

Gli strumenti liberi

- I sistemi operativi
 - ★ Per lo più *Unix-like* - GNU/Linux, (Free|Open|Net)BSD
- I linguaggi di programmazione ed i compilatori
 - ★ Linguaggi compilati: C, C++, Objective-C, Java, Fortran, Ada ...
 - ★ Interpretati: Perl, PHP, Python, Tcl/Tk, Unix shell ...
 - ★ Il compilatore per eccellenza: GCC (*The GNU Compiler Collection*)
 - * *Front end* per C, C++, Objective-C, Fortran, Ada, Java
 - * *Multi- e cross-platform*
 - * Conforme agli standard ANSI/ISO
- Le librerie di sviluppo
 - ★ Insieme pressoché infinito
 - ★ Librerie grafiche (e non solo)
 - * Gtk/Gnome (*The Gimp Toolkit*), Gnome
 - * Qt/KDE
 - ★ Interfacce di programmazione ad applicativi

Gli strumenti liberi

- Gli ambienti di sviluppo integrati (IDE)
 - ★ Kdevelop (Qt/KDE-oriented)
 - ★ Anjuta (Gtk-oriented)
- I debuggers (GDB e relativi front-end)
- I sistemi di gestione dello sviluppo concorrente e di manutenzione
 - ★ CVS (*Concurrent Versioning System*)
- I tool di localizzazione e internazionalizzazione
 - ★ Gettext
 - ★ Pango
- I sistemi di documentazione
 - ★ Man
 - ★ Texinfo

Gli strumenti liberi

- I formati di documentazione liberi
- I sistemi di gestione dei contenuti (CMS)
- I sistemi di bug-tracking
 - ★ Bugzilla
- I servizi per l'indicizzazione e l'hosting di progetti in rete
 - ★ Sourceforge (hosting)
 - ★ Freshmeat (indicizzazione)
 - ★ I LUG!

Un nuovo progetto di software libero

(un applicativo scritto in C, tanto per chiarire le idee)

- Preliminari

- ★ L'albero dei sorgenti

```
mortadella-x.y.z/  
  \__ AUTHORS  
     COPYING  
     INSTALL  
     README  
     TODO  
     doc/  
     \__ info/  
         man/  
     intl/  
     src/  
     ...
```

- ★ I numeri di versione

```
mortadella-<major>.<minor>.<patchlevel>-<altro>
```

- * Standard del kernel linux: minor dispari = versione in sviluppo

Strumenti di manutenzione (GNU coding standard)

- Automattizzazione del processo di configurazione e compilazione
 - ★ Make
 - * Automattizzazione dei passi di compilazione (`Makefile`)
 - * Tracciamento di modifiche e dipendenze tra moduli
 - * Uso ricorsivo
 - ★ Automake
 - * Generazione di Makefile complessi (`Makefile.in`) da macro (`Makefile.am`)
 - ★ Autoconf
 - * Creazione di uno script di autoconfigurazione (`configure`) da macro (`configure.in`)
 - Adattamento del software al sistema che lo esegue (portabilità!)
 - Verifica ambiente di compilazione e librerie
 - Creazione dei Makefile finali
 - ★ Altri (`libtool`, `autoheader`, `aclocal`...)
- Gli IDE (es. Kdevelop) automatizzano l'uso di questi strumenti (*wizard*)

Ed ora al lavoro!

- Codifica
- Documentazione
- Traduzioni
- Distribuzione
- Recepimento del feedback degli utenti, bug tracking
- Manutenzione
 - `while (1) {`
 - ★ Correzione dei bug
 - ★ Aggiunta nuove funzionalità
 - ★ Testing e distribuzione nuova versione
 - ★ Feedback (suggerimenti, patches)
 - `}`

In modalità consumatore

- I sistemi operativi liberi (per lo più **nix flavours*)
 - ★ GNU/Linux. Disponibile in numerosi gusti (o *distribuzioni*)
 - * Debian
 - * Slackware
 - * Red Hat
 - * Mandrake
 - * ...
 - ★ *BSD
 - * FreeBSD
 - * OpenBSD
 - * NetBSD
- Gli applicativi liberi
 - ★ Le distribuzioni escono con grandi quantità di software (libero o anche non libero) precompilato e *pacchettizzato*
 - ★ Sistemi di gestione/installazione/configurazione pacchetti con controllo versioni e dipendenze (da citare: APT di Debian)

In modalità consumatore

- Gli aggiornamenti
 - ★ Le maggiori distribuzioni mettono a disposizione i pacchetti corretti in corrispondenza di bug e/o vulnerabilità
 - * Sistemi di aggiornamento automatico integrati nella gestione pacchetti
 - * Da citare: *security.debian.org*. Tempo medio (dichiarato) di pubblicazione del fix dalla segnalazione della vulnerabilità: circa 48 ore
 - ★ Aggiornamenti molto specifici
 - * A livello di singole applicazioni/moduli/librerie compresi nella distribuzione, non solo a livello di nucleo ed utility di sistema
 - * BTW quanto è grosso l'ultimo ServicePack(TM) di Windows XP(TM)?
 - * Ogni quanto escono questi "pacchi"? <G>

Non sono contento, mi manca ancora qualcosa...

E allora compila!

- Necessità dei sorgenti per l'utente
 - ★ Esame del codice
 - * Può essere fondamentale in contesti critici (es. applicazioni crittografiche per autenticazione, privacy e integrità dei dati)
 - ★ Curiosità!
 - ★ Personalizzazione
 - ★ Incorporazione di procedure di terze parti
 - ★ Mancanza o incompatibilità dei binari rilasciati
 - ★ Ottimizzazione per l'hardware e il software ospite
 - ★ Esigenze di sicurezza per cui si rendano troppo lunghi i tempi fisiologici di pubblicazione del pacchetto corretto
 - * Nelle mailing list o gruppi di discussione vengono spesso pubblicati quick fixes e patches contestualmente all'annuncio della vulnerabilità
 - ★ *Perché io uso la slackware e mi piace compilare! Ovvvia!*

Configurazione, compilazione e installazione

- Procurarsi i sorgenti
 - ★ Ricerca su *freshmeat.net*
 - ★ Ricerca mirata su motori di uso generale
 - ★ Distribuzione ufficiale
 - ★ Direttamente dal *repository* CVS del progetto
- Sistemi Unix-like: la *tarball*

```
mortadella-0.0.1.tar.gz
```

- La tarball “esplosa”

```
$ ls ~/mortadella-0.0.1/  
AUTHORS      Makefile.am      TODO            configure       install-sh  
COPYING      Makefile.dist    acconfig.h      configure.in     missing  
ChangeLog    Makefile.in      aclocal.m4      mortadella/     mkinstalldirs  
INSTALL      README           config.h.in     mortadella.lsm  stamp-h.in
```

Configure, make, make install!

- Leggere il file `INSTALL!`
- Configure

```
$ ./configure <opzioni>
```

- ★ In caso di errori modificare l'ambiente (!) e riprovare finché tutto va a buon fine

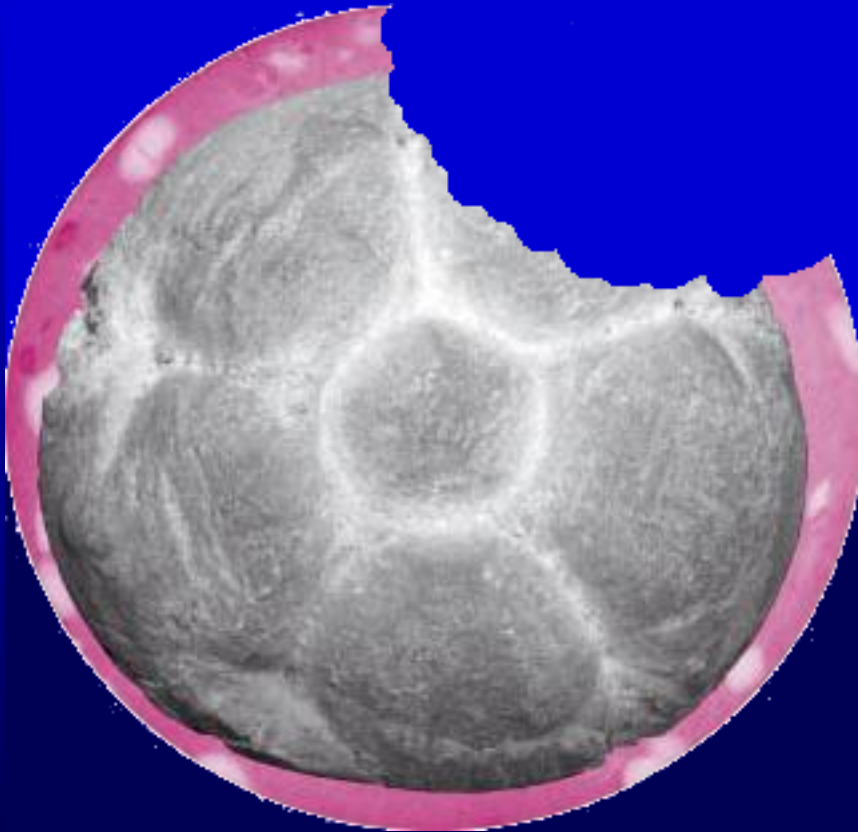
- Make

```
$ make all
```

- ★ Se ci sono errori qui le cose si mettono molto, ma molto peggio

- Make install

```
$ make install
```



Enjoy free software!!!

/giannibi

Appendice. Le mailing list del FLUG

- Flug - Discussione generale
<http://lists.firenze.linux.it/mailman/listinfo/flug>
- Primipassi - Tecnica, per utenti alle prime armi
<http://lists.firenze.linux.it/mailman/listinfo/primipassi>
- Tech - Discussioni tecniche
<http://lists.firenze.linux.it/mailman/listinfo/tech>

Riferimenti

- E. Raymond, *The Cathedral and the Bazaar*,
<http://www.tuxedo.org/~esr/writings/cathedral-bazaar>
- R. Chassel, *Un'economia del software libero: vantaggi e pericoli*,
<http://www.softwarelibero.it/altri/economia-sl.shtml>
- OpenOffice.org
<http://www.openoffice.org>
- Il progetto GNU
<http://www.gnu.org>
- Il kernel Linux
<http://www.kernel.org>
- Gnu Compiler Collection
<http://gcc.gnu.org>

Riferimenti

- Perl
<http://www.cpan.org>
- Python
<http://www.python.org>
- The Gimp Toolkit
<http://www.gtk.org>
- Gnome
<http://www.gnome.org>
- The Qt Toolkit
<http://www.troltech.no>
- KDE Desktop Environment
<http://www.kde.org>

Riferimenti

- CVS, Concurrent Versioning System
<http://www.cvshome.org>
- Bugzilla
<http://www.mozilla.org/projects/bugzilla>
- Sourceforge
<http://www.sourceforge.net>
- Freshmeat
<http://www.freshmeat.net>

Riferimenti

- Debian GNU/Linux
<http://www.debian.org>
- Slackware
<http://www.slackware.com>
- Red Hat
<http://www.redhat.com>
- Mandrakesoft
<http://www.mandrakesoft.com>

Ack

- Szymon Stefanek a.k.a. Pragma
<http://www.kvirc.net>
- Firenze Linux User Group - FLUG
<http://www.firenze.linux.it>
- Associazione Software Libero - AsSoLi
<http://www.softwarelibero.it>

Typeset using \LaTeX