

# il **GiornaLinux**

del  **POUL**  
have a lot of fun



Mess with the best...  
die like the rest!



- ~ Software Patents
- ~ GNU/Linux [in]security
- ~ Linux-HA
- ... ed altro ancora!

#### INTRODUZIONE:

Il POuL e' un'associazione studentesca Libera e senza fini di lucro, principalmente composta da studenti del Politecnico di Milano. Si propone di sviluppare, attorno allo studio e all'hacking dei sistemi \*nix, una comunita' solida e collaborativa, esente da pregiudizi di sesso, razza, religione, regione e fede politica. Costituita nel dicembre 2001 da 10 soci fondatori, il POuL condivide le medesime finalita' e linee guida che hanno portato alla stesura dello statuto dell'associazione culturale OpenLabs (<http://www.openlabs.it>), al quale il presente e' solo un modulo aggiuntivo (i lettori sono pertanto invitati a prendere visione del citato statuto). Il POuL non e' un'entita' fiscale/legalmente riconosciuta, e si appoggia pertanto ad OpenLabs per il proprio sviluppo.

#### MECCANISMI DI FUNZIONAMENTO:

Il POuL (d'ora in avanti, per comodita', "associazione") e' guidato da un direttivo, composto da 7 membri dell'associazione ed aventi i requisiti per essere eletti, e rimane in carica per 1 anno accademico, salvo scioglimento da parte dell'assemblea degli aventi diritto al voto (d'ora in avanti, per comodita', "assemblea"). Il direttivo viene eletto dall'assemblea, tramite votazione espressa nel seguente modo: una volta raccolte le candidature per il direttivo, ogni avente diritto al voto esprime al piu 3 preferenze tra i canditiati, in modo totalmente anonimo. Vengono eletti nel direttivo i 7 candidati che avranno raccolto il maggior numero di voti. In caso di ballottaggio, il candidato che otterra' il 50% + 1 dei voti della votazione di ballottaggio verra' eletto. In caso di ulteriore parita', sar  il direttivo uscente a decidere sul da farsi.

Compongono l'assemblea degli aventi diritti al voto tutti i membri iscritti alla lista pubblica dell'associazione che abbiano partecipato ad almeno 1 riunione indetta dal direttivo, e che possano certificare la propria iscrizione al Politecnico di Milano.

In questa fase dell'associazione non verranno raccolti contributi d'iscrizione, ma per poter essere eletti nel direttivo, sara' necessario essere iscritti ad OpenLabs almeno come "socio semplice". I membri del direttivo dell'associazione dovranno necessariamente avere lo stesso potere decisionale. Per garantire una corretta comunicazione con OpenLabs, il direttivo dovra' eleggere a suffragio universale, entro il termine di 2 settimane dall'elezione del nuovo direttivo, un referente che si occuper  delle relazioni con la citata associazione. Le decisioni che prevedono una modifica del presente statuto devono essere votate all'unanimita dal direttivo dell'associazione, e confermate da almeno il 50% + 1 partecipanti all'assemblea, entro il termine di 2 settimane. Se entro tale termine la decisione non sara' stata confermata dall'assemblea, sara' da ritenersi decaduta e necessitante di una nuova votazione all'unanimita' da parte del direttivo. Le decisioni che invece non prevedono una modifica statutaria possono essere confermate direttamente dal direttivo tramite voto del 50% + 1 dei suoi membri, purch  queste decisioni vengano poi rese note durante la prima assemblea. L'assemblea degli iscritti puo', se le circostanze lo rendono necessario, sciogliere il direttivo. Per fare cio', dovra' essere presentata una ragione valida di fronte ad un'assemblea che veda presenti almeno il 90% degli iscritti all'associazione aventi diritto di voto, e la mozione dovra' essere votata da almeno il 75% + 1 degli aventi diritto di voto.

Il direttivo del POUL si riunisce almeno una volta al mese. Le modalita' ed il luogo di riunione vengono decise dal direttivo stesso, e comunicati anche in lista pubblica. Eventuali spostamenti di data, dovuti ad eventi o manifestazioni, vengono votati a maggioranza dal direttivo stesso, e vengono comunicati anche pubblicamente.

#### ORGANIGRAMMA:

I membri dell'attuale direttivo, eletto il 16 aprile 2005 (e valido quindi sino al 16 aprile 2006), e' composto da:

- \* Mangano Giuseppe (Silvero)
- \* Primier Corrado (bARDO)
- \* Puglisi Silvia (Hiromi)
- \* Rizzo Giacomo (alt-os)
- \* Scabini Igor (Fer)
- \* Serra Guido (Zeph)
- \* Sia Danilo (Birdack)

( ^  
// \c{ }  
V \_/\_

Sono inoltre definiti i seguenti incarichi:

Referente presso OpenLabs: Guido Serra  
Tesoriere: Sia Danilo  
Magazziniere: Rizzo Giacomo  
Addetto stampa: Primier Corrado  
Addetto sito Web: Mangano Giuseppe

-0)  
// \c{ }  
V \_/\_

Altri eventuali incarichi verranno decisi e comunicati in caso di necessita'.

N.B. Lo statuto attuale del POuL e' un modulo aggiuntivo dello statuto di OpenLabs, associazione della quale il POuL   satellite.

# INDICE

:: Lo statuto del POuL .....	pag. 0010
:: POuL: Politecnico Open unix Labs .....	pag. 0100
:: Security Workshop, 7 aprile 2005 .....	pag. 0101
:: Linux in uno studio professionale .....	pag. 0110
:: Come funzionano i Software Patents .....	pag. 1000
:: L'effettività giuridica delle licenze .....	pag. 1001
:: Cluster ad alta affidabilità: Linux-HA .....	pag. 1010
:: GNU Linux, libertà e diffusione .....	pag. 1100
:: GNU/Linux [in]security .....	pag. 1110

La versione PDF e' diponibile sul sito del POuL -> <http://www.poul.org>

## Copyright Notice

Quest'opera è rilasciata sotto la licenza: Creative Commons BY-NC-SA 2.0  
Questo significa che sei libero di riprodurre, distribuire, comunicare al pubblico, esporre in pubblico, rappresentare, eseguire o recitare l'opera e creare opere derivate alle seguenti condizioni:



**Attribuzione.** Devi riconoscere il contributo dell'autore originario.



**Non commerciale.** Non puoi usare quest'opera per scopi commerciali.



**Condividi allo stesso modo.** Se alteri, trasformi o sviluppi quest'opera, puoi distribuire l'opera risultante solo per mezzo di una licenza identica a questa.

In occasione di ogni atto di riutilizzazione o distribuzione, devi chiarire agli altri i termini della licenza di quest'opera. Se ottieni il permesso dal titolare del diritto d'autore, è possibile rinunciare ad ognuna di queste condizioni.

Le tue utilizzazioni libere e gli altri diritti non sono in nessun modo limitati da quanto sopra

Questo è un riassunto in linguaggio accessibile a tutti del Codice Legale:  
<http://creativecommons.org/licenses/by-nc-sa/2.0/it/legalcode>



# POuL: Politecnico Open unix Labs

In quattro anni, quante cose!

POuL? che cavolo di nome! Non ricordo più come è venuto fuori questo acronimo assurdo 4 anni fa quando siamo nati. Tutto è partito da alcuni frequentatori di una chat chiamata #polimi sulla rete IRCnet. Dei "veci" ne sono rimasti pochi, ormai davvero pochi. Eppure sono già passati 4 anni. All'interno dell'associazione alcune cose sono cambiate, altre no. Non sono cambiati i nostri intenti, non sono cambiati i valori in cui crediamo, non è cambiata la voglia di fare, e questo giornalino lo dimostra. E nonostante questi 4 anni, il POuL è ancora un gruppo giovane. Siamo tutti studenti del nuovo ordinamento, col tempo si è aggregato qualche dottorando, qualcuno è passato alla specialistica, qualcuno si sta laureando e ci sta lasciando, qualcuno magari ci raggiungerà, invogliato da questo giornalino :).

Vi state ancora chiedendo cosa sia il POuL? :) Beh, è presto detto. Il POuL vuole essere prima di ogni altra cosa un gruppo di amici. Un gruppo di persone che condividono dei valori, degli ideali, delle curiosità. Un gruppo di persone che si incontrano, cercando di allietare, gli uni con gli altri, la nostra (vostra) permanenza in questa grigia città. Fare hacking è bello, ma farlo in compagnia, è divino.

Su questa filosofia, anni fa è nato il Politecnico Open unix Labs. Non siamo un LUG (Linux User Group), ma non siamo nemmeno un'associazione di retrocomputing, siamo un hacklab, con un occhio per tutte le piattaforme unix-like.

Alcuni di voi ci hanno conosciuto di recente alle conferenze sulla Sicurezza Informatica, altri ci conoscevano già, per sentito dire, per passaparola, per google... Altri, non ci conoscevano, e non ci conoscono tuttora, nonostante abbiano in mano questo giornalino. E allora diciamolo. Cosa fa il POuL?

Beh, qualcosa di definito, il POuL non lo fa. Si fa un po' di tutto, lasciandosi guidare dalla propria fantasia e spirito di iniziativa. Come abbiamo appena detto, quest'anno abbiamo organizzato (grazie anche al sostegno offertoci dal Politecnico di Milano) un workshop sulla sicurezza, che ha visto una partecipazione incredibile (si diceva che manco a lezione...). Abbiamo tirato in piedi questo giornalino. Stiamo organizzando la Giornata delle Libertà Digitali (la quinta edizione, ormai...). Partecipiamo ad eventi milanesi, provinciali, nazionali, internazionali (sto parlando delle giornate di "Fa' la cosa giusta", a cui abbiamo partecipato assieme all'associazione culturale Openlabs, sto parlando dei vari Webbit [01, 02, 03, 04, non so se faremo lo '05...], dei LinuxDay [02,03,04] e degli Hackmeeting [Bologna '03, Madrid '03, Genova '04 e quest'anno Napoli]...).



Vi manca qualcosa all'appello? Volete partecipare? L'iscrizione consiste nell'iscriversi ad una mailing list. La trovate sul nostro sito ([www.poul.org](http://www.poul.org)). Fatto questo, sarete informati di tutto quello che l'associazione organizza, e potrete partecipare alle interessanti discussioni che si svolgono quotidianamente in lista. Per partecipare alle manifestazioni, si chiede solo di dividere le spese. Non c'è scopo di lucro, solo voglia di stare insieme. Se avete voglia di imparare, condividere, mettervi in gioco, o solo farvi una pizzata con dei geek, bene, siamo il gruppo che fa per voi.

Il presidente  
Guido Serra, a.k.a. Zeph  
[zeph@poul.org](mailto:zeph@poul.org)

# Security Workshop, 7 aprile 2005

Cronaca di un evento del POuL

Il 7 aprile 2005, presso l'aula Il2 del Politecnico di Milano (sede di Piazza Leonardo), si è svolto un workshop sulla sicurezza informatica, organizzato dal POuL (Politecnico Open unix Labs) e patrocinato dal Politecnico di Milano.

L'incontro, durato la bellezza di 4 ore (dalle 9:15 alle 13:15, altro che lezioni da 45 minuti :D), ha visto la partecipazione di oltre 300 studenti, con picchi massimi di presenza in aula di oltre 280 persone, intente a seguire i diversi interventi.

Tra i relatori, da notare la partecipazione dell'Ing. Stefano Zanero, dottorando presso il nostro ateneo, e sicuramente personalità di spicco nel panorama della sicurezza informatica in Italia, che ha presentato il suo più recente lavoro sulle "tecniche d'apprendimento non supervisionate" per gli Intrusion Detection Systems (IDS). L'interessantissimo speech ha incluso anche una generale presentazione degli IDS, strumenti ormai indispensabili nella gestione della sicurezza informatica dei sistemi informativi "critical". Le successive presentazioni, preparate da membri dello staff del POuL, hanno visto la trattazione di altri argomenti cruciali nel mondo della sicurezza informatica.



Giacomo Rizzo (a.k.a. alt-os) ha presentato uno speech di ampio respiro sulle tecniche di attacco nelle reti LAN, con particolare attenzione agli attacchi di tipo Man In The Middle, di cui ha potuto illustrare alcuni esempi.

Danilo Sia (a.k.a. Birdack), incaricato del difficile compito di preparare il "talk tecnico" dell'evento, ha presentato alcuni esempi di cattiva programmazione, con relativi exploit, volti ad ottenere l'accesso root sulla macchina bersaglio. Nonostante parte della platea abbia comprensibilmente fatto un po di fatica a seguire, molti sono stati coloro che si sono complimentati con Danilo per la qualità' del suo intervento.

Giuseppe Mangano (a.k.a. SilveRo), infine, ha parlato della situazione attuale della sicurezza delle reti wireless, con particolare attenzione al territorio milanese. Dopo una breve introduzione al WiFi, sono stati trattati aspetti interessanti del mondo "senza fili", come il WarDriving ed il "modus operandi" delle intrusioni che le reti wireless rendono possibili, con alcuni accenni alle contromisure che la tecnologia mette a disposizione a chi, da queste intrusioni, si deve difendere.



Nel suo complesso, l'intero evento è stato seguito con notevole interesse. L'affluenza ha colto impreparati gli organizzatori, non abituati ad una simile partecipazione, che è risultata superiore a quella delle manifestazioni precedenti. Complice di questa notevole affluenza sicuramente è stato il patrocinio dell'Ateneo, che ha messo a disposizione i suoi strumenti informatici (tra cui la mailing list a cui sono iscritti tutti gli studenti) al fine di pubblicizzare l'evento. Durante l'evento sono state fatte registrazioni audio degli interventi che, con le relative slides, sono disponibili sul sito del POuL, dove è possibile vedere anche le foto fatte quella mattina.

# Linux in uno studio professionale

E' così semplice scegliere una distribuzione?

Quando si pensa a Linux, molti si immaginano il solito server oppure il più blasonato firewall aziendale, ma quello che non si immagina mai è il suo utilizzo all'interno di uno studio, e non per il suo essere server, ma per la sua nuova dimensione Desktop.

Partiamo però da un concetto principale: Linux non è ancora pronto per riuscire ad entrare nelle case di tutti gli utenti e prendere il posto del sistema operativo di casa Microsoft; e questo non perché sia poco affidabile oppure complicato, ma solamente perché gli utenti non sono ancora pronti a fare un qualche cosa che sia diverso dal solito "per installare clicca sempre su avanti". Sotto Linux è ancora necessario pensare a quello che si sta facendo, doversi ricordare più di una password, oh quante password ci si deve ricordare in questo periodo storico, e poi decidere cosa installare e cosa no. Per me che utilizzo Linux da moltissimi anni questo è solamente un piccolo scoglio, diciamo una cunetta, ma per l'utente che si avvicina a linux per la prima volta, tutto diventa difficile ed insuperabile. D'altra parte, diciamo anche che i passi avanti che sono stati fatti sono grandi, per non dire enormi; utilizzare ora Linux, anche per i compiti come lo scrivere questo articolo oppure leggere il giornale online, è notevolmente più semplice rispetto ad alcuni anni fa, ed i vari gestori di desktop come KDE e Gnome, solo per nominare i due più famosi, aiutano.

Fatta questa piccola premessa, vediamo di capire quale può essere il tipico computer da lavoro per un utente che vuole utilizzare Linux come Desktop e non come Server. Prima di tutto la distribuzione da utilizzare, una delle scelte più importanti perché deve essere utilizzabile sia per l'utente di tutti i giorni, il desktop-user che si aspetta di cliccare e aspettare poco tempo per iniziare a lavorare, sia per l'utente più raro, il sistemista che non può perdere troppo tempo per risolvere un problema o aggiornare tutto il sistema.

Avendo provato un po' di distribuzioni la mia scelta, e permettetemi di sottolineare come questa scelta sia nel pieno rispetto dell'IMHO (In My Humble Opinion), è caduta sulla Novell Suse 9.2 Professional. Questa scelta è stata motivata da una scelta ancora più a monte: il sistema operativo da utilizzare sul server, che dovrà essere il collante che permetterà allo studio di funzionare al meglio, aumentandone anche la produttività. In questo caso le distribuzioni si equivalgono, ed ognuna ha i suoi pregi ed i suoi difetti.

La cosa che si deve tenere in considerazione per effettuare la scelta migliore non è la mole di documentazione ed informazioni che si possono trovare su internet oppure in libreria, ma la nostra conoscenza della distribuzione e la possibilità che il computer che farà da server sia utilizzabile con la distribuzione scelta. Prendiamo due casi distinti e partiamo da un server nuovo di zecca da comprare ex-novo. In questo caso possiamo scegliere la distribuzione che più ci aggrada e che meglio conosciamo, come ad esempio la Debian, per la sua ottima reputazione di distribuzione "da server", oppure la Fedora Core, per la sua ottima reputazione di distribuzione adattabile a moltissimi nuovi server ed alle tecnologie di ultima generazione grazie al supporto di RedHat. Prendiamo ora il caso contrario: si arriva in uno studio dove è già presente un server da formattare ed utilizzare per i nostri scopi, e le cose si complicano. Perché si complicano? Semplice: prima di tutto si deve controllare che l'hardware che compone il server sia utilizzabile sotto Linux, e già questo restringe le possibili distribuzioni ad un numero molto più piccolo rispetto ad un server ex-novo. L'hardware magari è dell'ultimissima generazione, oppure è talmente "esoterico", passatemi il termine, che solamente una distribuzione può essere utilizzata.

Ma se dovessimo scegliere la distribuzione solamente partendo dall'hardware del server saremmo in errore: la distribuzione non deve essere scelta solamente in funzione del server, ma anche in funzione di quello che dovrà fare il server: stiamo parlando di gestire uno studio con Linux, ed in altre parole far sì che un computer rotto non significhi fermare il lavoro di un professionista perché i suoi dati risiedevano proprio sul computer guasto; questo vuole dire che il server dovrà contenere le home degli utenti, e quindi la distribuzione che andremo a scegliere dovrà riuscire a gestire sia le home degli utenti che avranno un Desktop Linux, quindi NFS (con tutto quello che ne consegue), sia quelli che magari dovranno utilizzare Windows, quindi SMB (con tutto quello che ne consegue). A questo punto si potrebbe pensare che la scelta della distribuzione è fatta: basta far sì che supporti NFS e SMB, e che l'hardware in



nostro possesso la supporti. Fosse così sarebbe semplice, ma in verità se le cose si devono fare, devono essere fatte per bene, e non alla buona. Prima di tutto, come si gestiranno gli utenti? Direttamente con le soluzioni di base della nostra distribuzione oppure con qualche cosa di più evoluto, come ad esempio LDAP? Se utilizziamo LDAP, qual'è la distribuzione migliore per riuscire a gestire al meglio questa scelta? Se utilizziamo la soluzione più semplice, quella degli utenti direttamente sulla macchina, non è che ci perdiamo la possibilità di utilizzare delle funzioni avanzate che la nostra scelta non ci concede? Queste piccole ma difficili domande sono solamente la punta dell'iceberg di tutto quello che il sistemista si deve "chiedere" affinché la propria scelta non si riveli sbagliata dopo poco tempo.

A questo punto ci troviamo in mano la scelta della distribuzione giusta per il nostro server: supporta l'hardware che dobbiamo utilizzare? Ha tutte le cose che ci interessano, come ad esempio LDAP oppure la gestione intuitiva del protocollo SMB, ed è giunto quindi il momento di installarla? Ma, e c'è sempre un ma, siamo sicuri che sia la scelta giusta? Se vi state chiedendo a cosa mi possa riferire, visto che ho appena detto che è la distribuzione giusta per il nostro sever, mi sto proprio riferendo al server, e non in quanto distribuzione, ma in quanto al lavoro che dovrà svolgere. Come interagirà questa distribuzione con quella che noi andremo ad adottare sui client? Non si è ancora scelta la distribuzione sui client? Ottimo. Pensate che stia sbagliando? Vediamo di farvi capire il mio pensiero. Se da una parte abbiamo una distribuzione-server che fa egregiamente il proprio dovere per quanto riguarda gestire le nostre necessità "da server", ed ha pieno supporto per il protocollo LDAP, dall'altra parte abbiamo la necessità di trovare una distribuzione che sia altrettanto valida per le necessità "da desktop" e, sottolineando in rosso la lettera 'e', deve essere anche semplice da agganciare, passatemi il termine, alla nostra distribuzione-server. Che senso ha prendere la distribuzione X per il server e la distribuzione Y per i client/desktop se poi queste due distribuzioni sono difficilmente interagibili? Se vogliamo basare tutto su LDAP e la distribuzione Y non lo supporta se non attraverso una serie di artifici, magari difficili da ricreare, come possiamo pensare di riuscire ad ottenere un ufficio che sia funzionale al 100%? Come vedete, la scelta di una distribuzione non è poi così semplice.

E finalmente siamo giunti ad avere la nostra scelta: distribuzione X per il server e distribuzione Y per il client/desktop, ma la nostra scelta si è basata sulla sola considerazione che free è bello oppure sulla considerazione di quello che dobbiamo fare (gestire uno studio professionale)?. Vi state chiedendo cosa intenda? Semplice: avete tenuto in considerazione anche le distribuzioni a pagamento? No? Sbagliato. Cerchiamo di non essere di mente ristretta pensando di essere nel giusto: noi dobbiamo gestire tutte le necessità di uno studio professionale, e nel contempo far vedere che la scelta di Linux non è stata solamente dettata dal fatto che è gratis, ma dal fatto che è una scelta valida per le necessità dello studio. Se tramite la distribuzione "libera" impieghiamo 20 minuti per fare una operazione da ripetere su N client, mentre con la distribuzione a pagamento che implementa una gestione più oculata della problematica che dobbiamo risolvere, ne impieghiamo solamente 5, secondo voi 20\*N è meglio di 5\*N solamente perché si sta utilizzando una distribuzione "libera"? Magari per voi sì, ma per la persona che vi paga no.

E con questo ho concluso, sperando di aver fatto capire come l'affermare "installo la distribuzione X" non sia sempre la scelta corretta se dietro di essa non vi è un ragionamento accurato di tutti i possibili scenari. La prossima volta, se ce ne sarà la possibilità, vedremo la scelta delle distribuzioni dedicate ai desktop, altro punto "caldo" di Linux in ambito lavorativo.

**Davide Michel Morelli, a.k.a. ZioBudda**  
 di [ZioBudda.net](http://ZioBudda.net) Italian Linux Portal  
[michel@ziobudda.net](mailto:michel@ziobudda.net)

(o\_      (o\_      (o\_      //\\_  
 (/)\_    (/)\_    (\)\_    v/\_-

# Come funzionano i Software Patents

Ecco cosa ci aspetta se passa la direttiva europea

Da promotori del free software vi aspetterete un articolo diverso da quanto stiamo per presentarvi. Magari vi aspetterete che vi parliamo di come i brevetti siano "un tradimento all'intera comunita' di sviluppatori", per usare le parole di Stallman. Quello che invece vogliamo presentarvi e' una piccola realta' imprenditoriale, in un settore, l'ICT appunto, che e' stato, fin dalla nascita, particolarmente adatto allo sviluppo di questo tipo di aziende; basti pensare a come hanno iniziato grandi corporation quali Apple o Microsoft.

Un programmatore puo' tranquillamente scrivere dell'ottimo software semplicemente facendo leva sulle sue abilita', e naturalmente avendo a disposizione dei computer, che al giorno d'oggi sono molto piu' facili da reperire ed acquistare di quanto lo fossero venti anni fa. Il suo lavoro e la sua attivita' sono automaticamente protette dalle leggi sul copyright, senza alcun bisogno di registrare il prodotto o pagare delle commissioni. Non vi serve una laurea in economia per capire quanto tutto questo generi un mercato di libera concorrenza, spinga all'innovazione, e fornisca piu' scelta al consumatore, abbattendo i costi. Cosa accadrebbe, pero', se qualsiasi parte del tuo lavoro originale inciampasse su un brevetto software?

Richard Stallman definisce i software patents dei campi minati per i programmatori, infatti in qualsiasi momento del processo di sviluppo potresti calpestare un brevetto e rendere inutilizzabile il tuo prodotto, o peggio ancora distruggere la tua attivita'. Un brevetto, infatti, concede un monopolio esclusivo, non su un'opera specifica, ma su una singola idea, per 20 anni. Inoltre, per ottenere un singolo brevetto in Europa, i costi si aggirano intorno alle decine di migliaia di euro, mentre per difendersi o contestarne uno le cifre sono molto piu' alte e per molti insostenibili. Dovrete inoltre considerare che, a differenza di quel che succede coi copyright, registrare e gestire dei brevetti richiede ricerche complesse e costose.

Dal canto loro, le grandi multinazionali evitano la minaccia di lunghe e costose cause legali tramite quello che in inglese viene chiamato cross-licensing. In pratica, si scambiano reciprocamente i diritti d'uso di gruppi di brevetti, escludendo tutti gli altri dall'accedervi. Quindi, un possessore di brevetti puo' non far altro che andare in giro accusando la gente di violazione della propria "intellectual property", e poiche' licensing e cross-licensing sono meno costosi di andare in tribunale, indifferentemente da quanto possano essere infondate le accuse, molti sono spinti a pensare di aver bisogno dei brevetti solo per sopravvivere alla minacce di azioni legali. Inoltre il software differisce da qualsiasi altro prodotto industriale; infatti, quando programmi, scrivi una serie di istruzioni e le dai in pasto al computer per farle eseguire. E' come un modello di business, o un gioco, o qualsiasi altra attivita' astratta che necessita' di regole ben definite. Stai semplicemente dando in pasto delle istruzioni, non stai inventando!

Se invece sei uno dei tanti sviluppatori che in tutto il mondo stanno lavorando ad un grande progetto come GNU/Linux, devi sapere che i software patents potrebbero essere usati da alcune grandi corporation per minacciare sia il Free che l'Open Source software, come farebbero con i concorrenti piu' piccoli; questo e' quanto sostengono molte persone che lavorano nel campo della Information Technology.

Venti anni fa infatti, Microsoft Windows, gli Apple Mac ed altre multinazionali erano appena nati, e non avevano alcun bisogno di brevetti software, ma anzi li temevano come li temiamo noi oggi, tanto da spingere niente meno che Bill Gates, nel 1991, a dichiarare: "Se quando la maggior parte delle idee di oggi sono state inventate tutti avessero saputo come i brevetti sarebbero stati concessi, e avessero quindi richiesto di brevettarle, l'industria odierna si troverebbe in uno stato di immobilita'".

Brevettare il software dovrebbe ricompensare e promuovere l'innovazione per il beneficio di sviluppatori, aziende e consumatori; in realta', i brevetti software escludono le piccole aziende, congelano l'innovazione e la concorrenza, e trasformano la pubblicazione di software in un privilegio per pochi.

(\*  
//\c{}  
V\_\_)

The Drunk Penguin

# L'effettività giuridica delle licenze

Da quando mi occupo degli aspetti giuridici legati alle licenze copyleft, la domanda che più spesso mi sento porre è: "ma queste licenze hanno validità legale?". Tale interrogativo, posto in questi termini, agli orecchi di chiunque abbia un'infarinatura di concetti giuridici, suona come una goffaggine, ma cela una legittima serie di dubbi che adesso andremo a discutere.

Innanzitutto l'apparente goffaggine deriva dal fatto che, essendo tali licenze attinenti all'ambito contrattuale, non si può parlare di una loro invalidità giuridica a priori. Il contratto è, di fatto, uno strumento di diritto privato con cui le parti regolano una certa situazione di rilevanza giuridica da cui si generano degli obblighi reciproci. Il diritto pubblico non può far altro che regolare le forme con cui questa potestà contrattuale privata va esercitata e al massimo indicare quali siano i vizi che determinano l'invalidità del contratto (o di alcune sue parti). Questo per dire che le licenze copyleft, siccome rientrano nell'ambito del diritto privato, non possono essere considerate "legali o illegali"; è invece da verificare se le principali licenze free/open nate negli Stati Uniti non contengano alcune clausole che, applicate in un contesto europeo, risultino invalide. I sistemi giuridici anglo-americano ed europeo-continentale, denotano alcune rilevanti differenze proprio nel trattamento dei diritti d'autore, ed in un certo senso anche nella definizione del tipo contrattuale in questione: mi riferisco alla disputa sul fatto che tali licenze siano da considerare atti unilaterali oppure atti bilaterali. Tale disputa è però più di natura giurico-dottrinale che pratica e per questo compete più agli accademici che ai consulenti o agli avvocati. A costoro si pone più che altro il problema dell'interpretazione (in senso giuridico) delle disposizioni contenute in queste licenze; problema che si pone però solamente nel momento in cui si instauri una controversia civile nata dall'applicazione della licenza. Questo vuol dire che ogni licenza è giuridicamente perfetta finché qualcuno che abbia avuto a che fare con essa non intenda contestarla di fronte a un giudice o in sede stragiudiziale. E' a questo punto che gli avvocati delle parti coinvolte e poi il giudice cercheranno di entrare nel merito del significato giuridico delle clausole della licenza, verificando quali di esse siano da considerare eventualmente invalide oppure a quali di esse attribuire una certa interpretazione piuttosto che un'altra.

Ecco perchè si sente spesso parlare della necessità di test giudiziali sulle licenze, ovvero di reali pronunce di organi giurisdizionali da cui poter trarre dei principi guida per l'enforcement di questi strumenti di tutela. Considerate che - nel sistema delle fonti del diritto -, in mancanza di precise disposizioni di legge, un rilevante ruolo è giocato dalle sentenze delle supreme corti che con la loro giurisprudenza possono sgombrare il campo dai più pregnanti dubbi interpretativi. Attualmente, in Europa l'unico punto di riferimento è una sentenza (dell'aprile 2004) di un tribunale tedesco il quale ha condannato al risarcimento del danno una società che aveva reso proprietarie parti di codice rilasciate sotto licenza GPL. E' ancora troppo poco per poter avere dei riferimenti giurisprudenziali rilevanti, soprattutto perchè trattasi di pronuncia di primo grado suscettibile di revisione in appello; ma è comunque un inizio. Alcuni malignamente (e miopemente) sostengono che la scarsità di pronunce giurisdizionali sulla GPL (o su licenze simili) derivi da una sorta di paura delle associazioni come Free Software Foundation a far valere in giudizio le proprie ragioni o da una pretesa mancanza di organizzazione delle stesse. Ma Eben Moglen (consulente legale di FSF) nel suo articolo "Enforcing the GNU GPL" ha tenuto a precisare: "La verità è esattamente il contrario: non ci siamo mai trovati a portare la GPL in tribunale perché mai nessuno ha voluto correre il rischio di contestarla in quella sede."

Sicuramente rimangono aperti molti problemi legati alla tutela giuridica delle licenze copyleft, e tra questi ne cito principalmente due: uno è l'individuazione dei soggetti che hanno titolo per difendere in giudizio la licenza. Il software sviluppato col modello free/open è un'opera con un numero indefinito di autori, dunque è difficile individuare chi tra essi (tutti? alcuni?) abbia titolarità giuridica per attivarsi in sede legale. E l'altro è relativo alla gestione pratica dell'attività legale: cause di questo tipo infatti posso risultare piuttosto onerose e le fondazioni/associazioni (a volte piuttosto piccole) che raccolgono gli sviluppatori di questi software potrebbero incontrare la resistenza di imprese (a volte piuttosto grandi) dotate di mezzi incomparabili. E' per questo motivo che a New York è stato costituito il Software Freedom Law Center, un ente di consulenza e assistenza legale attivo in ambito copyleft e coordinato da giuristi di spicco come lo stesso Moglen e Lawrence Lessig (Creative Commons).

Gran parte delle informazioni qui riportate sono approfondite e sviluppate all'interno del mio libro "Copyleft & opencontent - l'altra faccia del copyright": informazioni sul libro e approfondimenti sul fenomeno copyleft al sito [www.copyleft-italia.it](http://www.copyleft-italia.it).

# Cluster ad Alta Affidabilità: Linux-HA

Quando l'unione fa la forza!

In una società sempre più informatizzata, è di vitale importanza garantire la continuità del servizio ad applicazioni software da cui dipendono le nostre attività quotidiane legate a servizi di rete: pensiamo ai server di posta, DNS, Web-server, Database, ecc..

La soluzione più vantaggiosa per ripristinare un servizio interrotto da una anomalia di un server (non importa di che natura sia: hardware o software) è senza dubbio avere la possibilità di 'dirottare' gli utenti del servizio (in maniera loro trasparente) su un server secondario che eroghi il medesimo servizio del primario.

L'insieme dei due server si può indicare come cluster ad alta disponibilità (intesa come disponibilità del servizio erogato).

Prima di affrontare il tema dell'alta disponibilità, è meglio chiarire il concetto di cluster. Un cluster è un insieme di elaboratori che concorrono ad un unico obiettivo. Ne esistono 2 tipi fondamentali:

1) Cluster di calcolo: tutti i computer condividono le loro risorse per effettuare calcoli scientifici (es. simulazioni fluidodinamiche, chimica molecolare) o commerciali (es. produzioni cinematografiche, ricerche petrolifere, simulazioni di crash-test). Una soluzione per i sistemi Linux è OpenMosix.

2) Cluster ad alta affidabilità: i computer si sorvegliano fra loro e intervengono per garantire la continuità del servizio erogato.

Scopo di questo articolo è illustrarvi il progetto Linux-HA, la cui homepage è all'indirizzo: [www.linux-ha.org](http://www.linux-ha.org)

L'obiettivo principale di Linux-HA è:

"fornire una soluzione di clustering che offra affidabilità e disponibilità attraverso lo sforzo di una comunità di sviluppatori"

( )  
\_o) BOFH  
/\ \\  
\\_V->

Il software sviluppato prende il nome di HEARTBEAT, letteralmente tradotto in "battito cardiaco", concetto che rende bene la stretta relazione che intercorre tra i nodi del cluster. Questo software fa parte ormai di molte distribuzioni tra cui Debian, RedHat, Conectiva, SUSE e Mandrake. Sebbene chiamato Linux-HA, è stato portato e funziona bene anche su FreeBSD e Solaris.

Heartbeat non è progettato per monitorare servizi, ma solo la disponibilità di un nodo, intesa come sua raggiungibilità in una rete. Grazie al monitoraggio costante ("battito cardiaco"), i nodi, al verificarsi di certi eventi, possono lanciare dei servizi e/o script. Attraverso tre semplici file di configurazione, si può customizzare completamente il cluster, ottenendo una notevole flessibilità. Il protocollo di rete di Heartbeat, particolarmente valido, è stato riutilizzato da altri progetti di clustering come LVM (Linux Virtual Machine) e Ultramonkey.

Il cluster più semplice (ed il più diffuso per questo progetto) è costituito da due pc con Linux installato (esistono pacchetti rpm e deb) collegati tra loro da un cavo di rete incrociato, che connette i due pc tramite due NIC 'riservate'. Gli sviluppatori consigliano inoltre di collegare i due PC tramite via seriale con un cavo null-modem, per scongiurare falsi allarmi dovuti a deficienze della rete.

Tra i nodi del cluster bisogna stabilire una gerarchia: un nodo sarà il master, e normalmente erogherà i servizi da garantire, e l'altro si occuperà di intervenire se non 'sentirà' più il master. L'intervento consisterà nell'avviare i servizi che gestiva il nodo principale sul nodo secondario e/o lanciare degli script le cui potenzialità spettano solo allo sviluppatore.

Nel pacchetto scaricabile dal sito del progetto vi sono inclusi alcuni script utilissimi, capaci, ad esempio, di far migrare l'indirizzo IP virtuale del server pubblico su un'altra macchina. Quest'ultimo script permette di assegnare dinamicamente un indirizzo IP al nodo che in quel momento viene eletto master (IP Takeover). Grazie a questo concetto potrete avere un server sempre raggiungibile anche se fisicamente saranno diversi PC, poiché lo raggiungerete grazie ad un IP flottante.

Per una più chiara visione di questo script, analizziamo adesso il traffico di rete generato dal cluster. Ogni server disporrà di due interfacce di rete, una connessa alla rete accessibile dagli altri host e un'altra riservata.

Ecco un esempio di cluster HA: ogni pc possiede due schede di rete, ed eth1 (la rete privata) collegherà i PC con un cavo cross:

Server 1 (Master):		Server 2 (Slave):
eth0 indirizzo privato		eth0 indirizzo privato
eth0:0 indirizzo IP *flottante*		
eth1 indirizzo rete HA		eth1 indirizzo rete HA

Sulle schede di rete che hanno un indirizzamento privato viaggia il protocollo di Heartbeat. La notifica del corretto funzionamento utilizza il protocollo UDP sulla porta 694 in broadcast, ed è possibile utilizzare il multicast se sulla stessa rete ci sono più di due nodi. In particolare, verrà inviato un pacchetto ogni intervallo di tempo settato nei file di configurazione, ed un nodo verrà ritenuto "morto" se gli altri nodi non riceveranno il suo pacchetto entro un periodo di tempo fissato. Questa rete inoltre sarà usata per sincronizzare tutti i file necessari ai servizi che migreranno (configurazioni, siti, DB) tramite, ad esempio, Rsync.

Sulla rete pubblica (nell'esempio eth0) si avvicenderà l'IP gestito da Ipaddr, il master sarà raggiungibile dall'esterno e lo slave prenderà l'IP pubblico (diventando così visibile agli utenti) solo se il master non sarà disponibile. Una nota di fondamentale importanza: l'indirizzo flottante DEVE essere gestito solo da Ipaddr, e non dal sistema (/etc/network).

Data la criticità dell'operazione che compie (assegnamento di IP con conseguente attività ARP e binding di servizi), è necessario analizzare molto bene l'architettura di rete nelle quale si intende lavorare.

Heartbeat può gestire (avvio e arresto) tutti i servizi tipici di Linux che risiedono in /etc/rc.d/init.d e quelli propri del progetto situati in /etc/ha.d/resource.d

Ovviamente si possono realizzare dei propri script e metterli in una delle due directory; l'unico vincolo è che si avviino tramite la dicitura "servizio start", e si terminino con "servizio stop". Tipici esempi sono l'avvio di Apache e Qmail per gli script di sistema, MailTo e AudibleAlarm per gli script del progetto.

Nella cartella /etc/ha.d risiedono i tre file di configurazione necessari per il corretto funzionamento di Heartbeat. Ognuno ha un preciso scopo, illustrato a fianco:

- 1) ha.cf: specifica tutti i settaggi di comunicazione del cluster (IP, nodo master)
- 2) haresources: specifica le risorse che verranno avviate sui nodi (i servizi)
- 3) authkeys: determina in quale modo i nodi si autenticano tra loro

Infine, eccovi una breve panoramica dei plugin-addons per Linux-HA:

STONITH: è una interfaccia per il riavvio da remoto dei nodi del cluster  
IPFAIL: permette di monitorare le connessioni verso l'esterno per reinstradare le comunicazioni

Altri progetti correlati sono:  
DRDB: progetto per fare il mirror via rete dei dischi, rimpiazzando i 'shared storage devices' comuni a molte soluzioni di clustering di vari Vendor  
MON: demone per il monitoraggio dei servizi  
Esiste anche un modulo per Webmin che permette di configurare Heartbeat comodamente via Web.

Spero di avervi dato una buon punto di partenza per cimentarvi nella realizzazione del vostro cluster ad alta disponibilità.

Mauro Rappa - OpenLabs  
mauro@crezine.com

# GNU Linux, libertà e diffusione

No, non sarò imparziale in questo articolo, sia ben chiaro fin dall'inizio. Se avessi voluto essere imparziale, non avrei scritto un articolo di questo tipo. Se quella che cercate è l'imparzialità o l'oggettività, allora questo non è l'articolo che fa per voi. Se volete o avete bisogno di costruirvi un'opinione, non fate finta di cercare parcondicio di sorta: quella che volete non è l'imparzialità. Cercate persone schierate, estremisti, tutti gli estremisti che potete trovare, indipendentemente dalle idee, dal credo e da ciò che professano, e confrontatevi con le loro opinioni. Senza mezzi termini, senza mezze parole, e soprattutto senza false ipocrisie e senza la presunzione di poter essere imparziali... Bene, state parlando con un'estremista. Non dei peggiori (o dei migliori?), è chiaro, ma sicuramente un estremista. Più che parlando, state leggendo l'articolo che ha scritto, probabilmente in un attimo di particolare slancio GNUPolitico. Forse il primo che possa ricordare. "GNUPolitica?? Ma siamo pazzi?? Ne abbiamo abbastanza dei mali che affliggono il mondo, di politici dagli interessi sempre ambigui... siamo geek, smanettiamo con un computer, ci piace passare le ore davanti a un monitor, non tirate in ballo la politica anche qua... per favore, non infilate la politica nell'informatica..."

Ok, d'accordo. Per questa volta sarete accontentati (sto cercando di convincervi ad arrivare fino in fondo? Può darsi, ma avete un solo modo per togliervi questa curiosità). Evitiamo argomenti pericolosi, non parliamo di GNUPolitica. Parliamo di GNU, Linux e distribuzioni, come il titolo sembrava promettere. Non so chi siate voi: tecnici? semplici curiosi? sviluppatori linux? ricercatori? Beh, se non avete mai sentito parlare di Linux e state leggendo questo articolo, probabilmente siete fuori strada. Prendete google, cercate "storia di linux" e mettetelo da parte. Se non sapete cos'è GNU, siete quasi giustificati: se ne parla poco, almeno tra i non addetti ai lavori. Beh, Linux deve molto a GNU, al progetto GNU. E se la diatriba è ancora aperta, se non altro gli deve l'idea: quella di un sistema operativo libero, completamente modificabile e distribuibile da chiunque, che garantisca all'utente la libertà di poter usufruire del suo computer come meglio crede, senza dovere niente a nessuno e senza correre il rischio di commettere un crimine o di essere chiamato "pirata" (e sì, lo so... tutti noi, come persone, istituzioni o aziende, compriamo regolarmente i software che utilizziamo o li disinstalliamo dopo i 30 giorni stabiliti dalle licenze shareware... e sì, dobbiamo essere fieri di essere stati in grado di scaricare un dato software da un sito warez, abilità che non tutti posseggono).

"Ma andiamo! Stiamo parlando di un computer... di tre scatole collegate tra di loro... non di diritti umani, non dei problemi di una comunità o di una nazione... come si può parlare di libertà in riferimento a delle scatole? come si può parlare di crimini o di 'pirati'? smettetela di giocare, ed iniziate a preoccuparvi di cose serie...". Forse è vero, sarebbe meglio che ci dedicassimo ad altro (esistono francobolli bellissimi, mi dicono) e ci dimenticassimo di quelle scatolette appoggiate sulle scrivanie di tutti gli uffici... quelle scatolette che gestiscono i nostri conti correnti, quelle scatolette che consentono all'ufficio anagrafe di funzionare, quelle scatolette che controllano il traffico aereo, quelle scatolette... quelle scatolette a cui ogni giorno deleghiamo sempre più compiti ed affidiamo sempre più dati, i nostri dati. Eh, sì. Questo è il bello dell'era dell'informazione. Ognuna di queste scatolette diventa giorno per giorno sempre più importante. Certo, potrete dire, "la scatoletta sulla mia scrivania non fa nulla di tutto ciò, l'affare non mi riguarda, i dati più sensibili che ho sono le mail della mia fidanzata...". Beh, non vi devo e non vi voglio convincere a passare a GNU Linux, ad abbandonare la stabilità del software che fino ad oggi avete utilizzato, non sono qui per questo. Non vi voglio annoiare con i soliti argomenti di cui si sente parlare. Non voglio fare discorsi abusati sulla maggiore stabilità di GNU Linux, sulle migliori prestazioni, sulla sua gratuità o sul fatto che, utilizzandolo, solo difficilmente prenderete "virus"... queste cose dovrete averle già sentite. Il discorso che voglio fare è un altro, un discorso che spesso si dimentica parlando di GNU Linux. Come credete che possa aver fatto un sistema come questo a raggiungere questo grado di evoluzione in così poco tempo? Come ha fatto in così pochi anni a diffondersi ed a guadagnarsi la fiducia dei suoi utenti in questo modo, tanto da convincere persone come me a scrivere articoli come questo, senza alcun profitto? O migliaia (eh, sì, migliaia) di persone a lavorare, dedicando buona parte del loro tempo libero ad una cosa che non gli appartiene e che non gli dà alcuna remunerazione? Qualche sorta di illusione collettiva? Può sembrare strano, ma anche i geek, gli hackers, o i ragazzini che scorrazzano per le nostre reti avrebbero cose ben più divertenti o piacevoli da fare nel loro tempo libero. Come ha fatto un sistema a raggiungere una tale dimensione da poter intimorire giganti che da anni sono

sul mercato? No, GNU Linux non è il prodotto di una particolare società, non è una questione di investimenti, di business plan o prospettive di guadagno... dite che è una questione di stabilità? di pochi virus? di sicurezza? di gratuità? Non esistevano già prima alternative sicure, stabili e/o poco costose?

Per uno che se lo sente dire per la prima volta, suona sicuramente un po' strano: alcuni dicono che GNU Linux sia il frutto del lavoro di una comunità, la comunità del software libero. Forse la causa di tutto ciò è un'utopia: l'utopia di poter avere un sistema su misura per ognuno di noi, con tutte quelle feature che ognuno di noi ha sempre sognato, senza alcuna limitazione legale al suo impiego o alla sua diffusione. Non ti piace come funziona quella cosa in quel programma? E' software libero: prendi i sorgenti e modificali. Se altri hanno la tua stessa opinione, probabilmente adotteranno la tua versione, ed il software continuerà ad evolversi. Vuoi dimostrare che quel tuo algoritmo, quella tua idea, è migliore di altre? Non hai bisogno di riscrivere il programma per intero, non devi farti in quattro con una società perché introduca la tua soluzione. Se è software libero, puoi prenderne i sorgenti, modificarli, applicare la tua idea e dimostrare di avere ragione. Ti piacerebbe un software nuovo che ancora non esiste? Inizia a scriverlo, lancia l'idea in rete. Se l'idea è utile e buona, qualcun altro ti aiuterà... Con il software libero, la qualità stessa del software che usi dipende da te. E se non hai le competenze per lavorare direttamente sul codice, non ti lamentare con i tuoi amici: scrivi bug report, parla con coloro che giorno per giorno lavorano sul software libero, dà loro dei suggerimenti... da spettatore passivo, se lo vuoi, puoi diventare attore ed avere un ruolo nello sviluppo di quel software che tanto ti piace, con la certezza che continuerà a fare ciò per cui ti è utile nel migliore dei modi e nella massima trasparenza per te e per tutti coloro che si affidano ai tuoi sistemi...

L'idea alla base del progetto GNU, la GPL, o GNU General Public License, è stata forse il motore di questo movimento: utilizzare il copyright, il diritto d'autore, ciò che di solito viene sfruttato per limitare i vostri diritti, per garantire agli utenti o a chiunque la libertà di usufruire di un programma, di modificarlo, di distribuirlo, a patto che le modifiche introdotte siano a loro volta rilasciate sotto questa stessa licenza, e quindi continuando a garantire la libertà del risultato. Non posso quindi che chiedervi un favore: quando parlate di Linux, di GNU, non pensate solo ad un comodo rimedio al problema delle licenze. Non pensate solo che è l'unico sistema in grado di far funzionare quella vecchia scatoletta borchiata "sun" su uno scaffale in fondo alla vostra camera, non pensate solo alle caratteristiche tecniche di ciò che avete di fronte... pensate soprattutto alla libertà che guadagnate come utenti ed alla libertà che contribuite a creare collaudando, provando o migliorando il software libero. Ricordatevi che state utilizzando il prodotto del lavoro collettivo di migliaia di persone, volontari e non, persone come voi, che hanno deciso di dedicare del tempo a creare quella cosa a cui anche voi state contribuendo.

Ok, anche se breve, direi che per oggi il mio slancio GNUpolitico è terminato. Tornate pure a compilare il vostro nuovo driver nVidia per giocare all'ultima versione di Unreal Tournament, perché altrimenti quel miliardo di transistor frutto dell'evoluzione tecnologica degli ultimi 20 anni che avete pagato a caro prezzo sarebbe inutile. Riavviate in windows ed usate i programmi che avete sempre usato, perché è quello che fanno tutti e poi quel programma è proprio bello, non se ne può fare a meno (e no, scaricare dai peer to peer non è immorale e non dovrebbe essere reato...). Compriamo Mac OS, che è bello, colorato, stabile ed accattivante. E quando installate GNU Linux, quando scegliete una distribuzione, l'importante è che riconosca bene il vostro winmodem e che sia facile da usare. Poco importa come è stata creata, il modello di sviluppo che è stato utilizzato o il modello etico alla base della sua evoluzione e diffusione. Quando scrivete software, tenete i sorgenti nascosti, non sia mai che qualcuno possa avere un'idea più utile della vostra guardandoli, o rilasciateli sotto licenza BSD: date ancora maggiore libertà, consentite anche a coloro che si sono sempre opposti alla diffusione del software libero di usare il vostro codice e di includerlo nelle loro applicazioni e nei loro sistemi, con giusto quelle migliorie indispensabili a rendere il prodotto commercializzabile, migliorie di cui non vedrete mai il codice. Tanto non è la libertà quella di cui abbiamo bisogno: vogliamo solo dimenticarci di quelle scatolette che stanno sui nostri tavoli. Tanto non è la libertà quella di cui abbiamo bisogno: vogliamo solo dimenticarci di tutto quello che ci sta attorno, e dedicarci a fare solo ciò che ci piace, alle cose "vere" della vita. E poi, è faticoso partecipare in ciò che ci circonda, e non ci possono essere questioni etiche dietro un computer...

Carlo Contavalli  
ccontavalli@commedia.it

# GNU/Linux [in]security

## I. Introduzione

I sistemi operativi GNU/Linux, così come altri sistemi della grande famiglia \*nix (\*BSD, Solaris e molti altri), sono spesso citati come esempio di sicurezza. Questa affermazione è indubbiamente opinabile, anzi, è ormai universalmente riconosciuto che i sistemi \*nix non sono sicuri per default e che la robustezza di un sistema dipende più di tutto dal suo amministratore. La sicurezza, però, non è una procedura, né una feature intrinseca in un kernel, o nel sistema operativo. La sicurezza è fatta di tanti piccoli accorgimenti e modifiche alle configurazioni di base dei nostri applicativi e ovviamente dello stesso kernel.

In questo articolo presenteremo alcune delle tecniche di base per mettere in sicurezza il nostro sistema. La sicurezza assoluta, beninteso, non esiste; ci si può solo avvicinare ad essa. Oltretutto, l'argomento è incredibilmente vasto, quindi l'obiettivo che ci prefiggiamo è quello di indirizzarvi su questa strada e di stimolarvi a documentarvi, imparare e capire: Google è vostro amico :)

## II. Amministrazione del sistema

Cominciamo col dire che il software del vostro sistema dovrebbe essere sempre aggiornato all'ultima versione disponibile: nella maggior parte dei casi, il rilascio di un aggiornamento è dovuto alla correzione di errori nel codice, oltre all'introduzione di nuove feature, e anche se nuovo codice può significare nuovi errori, ci vorrà più tempo prima che un eventuale attaccante li scopra e capisca come sfruttarli. Sperando che nel frattempo gli sviluppatori li correggano in una nuova release.

Un'altra ottima abitudine è quella di non utilizzare l'utente root (o qualunque altro utente con privilegi alti) se non in caso di necessità. Un programma vulnerabile permette infatti di ottenere i permessi di chi lo esegue: se si tratta di un utente è quantomeno possibile limitare i danni. A questo proposito è bene accennare anche la gestione dei permessi: a meno di particolari situazioni è buona norma inserire (o modificare, se già esistente) nel file `/etc/profile` la riga:

```
umask = 027
```

In questo modo, ogni volta che un nuovo file viene creato, i suoi permessi saranno impostati in modo che il proprietario possa farne ciò che desidera, gli utenti del suo gruppo di default (leggere `'man group'` per informazioni) potranno leggerlo ed eseguirlo e tutti gli altri non potranno toccarlo. Ulteriori dettagli disponibili in `man umask`.

## III. Servizi

Se la vostra macchina offre servizi (web, ftp, terminale o altri) è importante che questi siano adeguatamente protetti. Se sono disponibili, è bene sfruttare protocolli di comunicazione cifrati, come https, sftp, ssl e ssh. La stessa cosa vale anche quando agite da client: verificate, ad esempio, se il vostro provider vi dà la possibilità di scaricare e inviare la posta tramite i protocolli apop e ssl.

## IV. Firewall

Il kernel Linux offre un firewall integrato molto potente, che lavora a filtro di pacchetto e permette la *stateful inspection*. Il suo nome è netfilter, ed è possibile attivarlo direttamente nella configurazione del kernel; nei kernel della serie 2.6 è reperibile sotto *Device drivers -> Networking support -> Networking options -> Network packet filtering*. Ricordate di caricare i moduli che vi servono se non compilate il supporto staticamente. Sarà oltretutto necessario installare nel sistema il programma iptables, che agisce da interfaccia tra l'utente e netfilter.

Una configurazione di base potrebbe essere la seguente:

```
iptables -P INPUT DROP
iptables -P OUTPUT ACCEPT
iptables -P FORWARD DROP
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Questi sono comandi da eseguire in shell ad ogni avvio del sistema, quindi è necessario inserirli in uno script che venga eseguito al boot.

Le prime tre righe definiscono il comportamento di default del firewall, le regole che saranno applicate in mancanza di altre più specifiche: rifiutare qualsiasi pacchetto in entrata o da inoltrare ad altri computer e permettere a qualunque pacchetto di uscire dalla nostra macchina. La regola successiva stabilisce di accettare tutti i pacchetti provenienti dall'interfaccia di loopback (il nostro computer), l'ultima permette di far passare i pacchetti appartenenti o relativi a connessioni già iniziate. Questo

!! >0)  
 ^ \  
 \_ \  
 - \_ V

significa che le uniche connessioni che funzioneranno saranno quelle iniziate da noi, e non ne accetteremo nessuna proveniente dall'esterno. In questo modo non potremo però fornire alcun servizio. Se volessimo ad esempio aprire a connessioni esterne la porta 22 per il servizio ssh, dovremmo aggiungere una linea come questa:

```
iptables -A INPUT -p tcp --dport 22 -j ACCEPT
```

Da questa regola è semplice dedurre altre che facciano lo stesso lavoro su altre porte. L'opzione `--dport` (utilizzabile se si è specificato il protocollo in uso con `-p tcp`) indica ovviamente la porta di destinazione della connessione; si può specificare una porta sorgente con `--sport`.

Queste sono solo alcune delle potenzialità di iptables; il sito ufficiale (<http://netfilter.org>) fornisce comunque grandi quantità di documentazione, anche in italiano. Un ottimo script, piuttosto complesso ma molto personalizzabile, è quello di MonMotha (<http://monmotha.mplug.org>). Molto utile è anche la pagina di Linux-Sec.net dedicata ai firewall (<http://www.linux-sec.net/Firewall>); nel resto del sito troverete comunque trattato l'argomento "sicurezza" più in dettaglio, da molti punti di vista. Può essere un buon punto di partenza.

## V. Kernel patching

A volte, per rendere sicura una macchina, è necessario agire a basso livello, modificando lo stesso cuore del sistema operativo, in modo da integrare nel kernel funzionalità orientate alla sicurezza, spesso scomode per un utente desktop ma indispensabili per chi del proprio sistema ha fatto un server.

Nei sistemi Linux si possono ottenere risultati molto soddisfacenti con due gruppi di patch, entrambi molto famosi: si tratta di GrSecurity (<http://www.grsecurity.net>) e di SELinux (<http://www.nsa.gov/selinux>). GrSecurity in particolare presenta molte interessanti feature, di cui citiamo le più importanti:

- **Role-Based Access Control (RBAC)**: a differenza del largamente usato *Discretionary Access Control* (DAC), in cui il proprietario del file decide i permessi sul file, si definiscono dei ruoli e le autorizzazioni vengono assegnate in base al ruolo che i singoli utenti ricoprono. Questo sistema si contrappone anche al *Mandatory Access Control* (MAC), usato da SELinux, consigliato soprattutto in organizzazioni di stampo militare, che prevede il controllo degli accessi limitato ad un'unica unità centrale che supervisiona l'intero funzionamento del sistema
- **Random PID**: il PID dei processi viene assegnato casualmente, rendendo di fatto impossibile conoscere a priori l'identificativo di un processo non ancora lanciato
- Lo stack non è eseguibile, rendendo quindi estremamente difficile qualsiasi attacco tramite *buffer overflow* (o comunque complicando la scrittura di tali exploit)
- Non è possibile inserire codice nella memoria in cui sta girando un processo
- La base degli indirizzi allocati dalla funzione `mmap()` è completamente casuale; l'operazione di *address-guessing* durante un exploit è pertanto molto più complicata
- Si ha una comoda protezione verso le "race conditions"
- La macchina non risponde ai pacchetti ICMP e gli *OS guessing* sono impossibili; sono inoltre disponibili varie configurazioni in risposta ai port-scanning

Installare una qualsiasi patch per il kernel risulta una operazione banale che si riduce all'esecuzione di un unico, semplice comando. Ad esempio, nel caso di GrSecurity, sarà sufficiente eseguire

```
patch -p0 < ./grsecurity.x.x-y.y.yy.patch
```

dove x è la versione della patch GrSecurity e y quella del kernel di linux. Successivamente, non si dovrà far altro che eseguire i normali passi di configurazione e compilazione senza alcuna variazione rispetto al solito.

## VI. Integrity checking

In una strategia di sicurezza è buona norma tenere traccia delle modifiche effettuate ai singoli file, o almeno a quelli maggiormente critici.

Questo per il semplice motivo che una volta compromesso un sistema è naturale che il malintenzionato sia portato ad effettuare modifiche atte a fare in modo che i suoi successivi accessi siano "facilitati". In questa analisi ci viene incontro una particolare categoria di software definita di *integrity checking*, che non fanno altro che analizzare il contenuto dei file ed estrapolare un identificativo univoco del suo contenuto. In tal modo possiamo assicurarci che ogni modifica ai file non passi insosservata. Ottimi esempi automatizzati di questo tipo di programmi sono *tripwire* (reperibile su <http://www.tripwire.org>), *integrit* (<http://integrit.sourceforge.net>) e *aide* (<http://www.cs.tut.fi/~rammer/aide.html>).

Daniilo Sia (a.k.a. Birdack) <[birdack@gmail.com](mailto:birdack@gmail.com)>  
Corrado Premier (a.k.a. BARDO) <[ilbardo@gmail.com](mailto:ilbardo@gmail.com)>

