Note sulla sicurezza dei sistemi

19 marzo 2007

Alessio Checcucci

Riccardo Aldinucci

Q.It Universita' degli Studi di Siena

1 Note sulla sicurezza

Lo scopo di queste brevi note non e' quello di fornire una visione esaustiva del problema della sicurezza dei sistemi Unix, ma solo di dare all'amministratore ed anche al comune utente che si trovi ad installare una macchina, delle linee guida per una configurazione di base che non lasci aperte tutte le possibili vie di intrusione al sistema.

La crescente popolarita' dei sistemi Unix, grazie soprattutto alla diffusione di GNU/Linux e dei sistemi BSD sul fronte server, ha portato ad un piu' rapido riconoscimento delle falle di sicurezza e ad una piu' rapida soluzione, grazie alla filosofia *release often release early* e alla *full disclosure*. Essendo Unix nato come un progetto di ricerca, la filosofia *securiry through obscurity* e' sempre stata ignota.

Dall'altro lato la diffusione crescente dei sistemi Unix ha anche allargato la "piattaforma d'attacco".

Il sistema Unix non e' stato progettato con criteri di sicurezza militare in mente. Anzi, fino a circa 15 anni fa' esistevano una quantita' enorme di metodi per introdursi nel sistema e ogni buon amministratore ne era a conoscenza. Lo sviluppo e la presa di coscienza dei problemi concernenti la sicurezza ha portato a realizzare sistemi operativi capaci di superare le strette specifiche della NSA (National Security Agency).

Il punto cardinale attorno a cui ruota la debolezza della sicurezza di Unix e' il potere illimitato che viene conferito al superutente (*root*) e ai suoi processi, che possono interagire con qualsiasi entita' e ogni file (in Unix tutto e' un file) presente sul sistema senza restrizioni. Questo tipo di controllo della sicurezza viene definito DAC (Discrectionary Access Control).

I progetti piu' avanzati nell'ottica della sicurezza sono tesi a modificare il comportamento di Unix (in particolare del kernel) per mettere in piedi dei meccanismi obbligatori che limitino cio' che root e i suoi processi possono fare. Questo tipo di approccio si chiama MAC (Mandatory Access Control).

Quello che segue e' un breve elenco di passi che si possono seguire appena installato un sistema e non esauriscono certo il campo delle cose che possono e dovrebbero essere fatte su un server Unix prima che questo venga "esposto al mondo".

1.1 Generalita'

1.1.1 Definizione di Sicurezza

Un computer e' sicuro se si puo' essere certi che esso e il suo software si comportano nel modo che ci aspettiamo.

1.1.2 La rete

Come primo passo, il sistema dovrebbe possibilmente essere installato completamente sconnesso da qualsiasi rete. In realta' se l'installer e' ben fatto non ci dovrebbero essere problemi. Ma e' sempre consigliato essere prudenti, specialmente se siamo su reti che non conosciamo bene o se siamo su VLAN non protette o pubbliche.

La macchina potra' essere connessa alla rete al termine dei successivi passi. La prima operazione da fare sara' poi quella di passare tutti gli aggiornamenti

disponibili. Esistono oggi un sacco di tool automatizzati che scaricano i pacchetti dalla rete, verificano la loro autenticita' per mezzo di chiavi crittografiche e li installano. Ma se la sicurezza deve essere un requisito stringente, allora e' meglio scaricare gli aggiornamenti su un'altra macchina verificare tutte le hash (MD5 o SHA1) crittografiche e installarli con la macchina ancora staccata da ogni rete. Solo termine sara' consigliato connettersi con ragionevole sicurezza.

1.1.3 Sicurezza del BIOS

Sebbene sia impossibile garantire la sicurezza di un qualsiasi oggetto a cui si possa accedere fisicamente, e' sempre importante limitare le possibilita' di interagire con le nostre macchine. Soprattutto perche' i possibili intrusori potrebbero avere un tempo breve per agire e quindi rendere piu' difficile l'accesso e' un buon sistema. E soprattutto, se si e' amministratori, non fidarsi mai di nessuno e non tenere le password a disposizione in luoghi ben conosciuti.

Se fosse possibile aprire o asportare un elaboratore la nostra sicurezza sarebbe nulla, ma questo non puo' essere un problema che deve risolvere l'amministratore ;-). Occupiamoci invece di porre un limite a come puo' essere cambiata la configurazione del sistema. Ovvero restringiamo tramite password l'accesso al software di configurazione del CMOS (o Flash-RAM), quella parte programmabile del BIOS ovvero la porzione di codice che viene caricata al boot del sistema e viene memorizzata in dispositivi non volatili. In questa maniera si impedisce che possa essere banalmente alterato il dispositivo da cui il sistema si avvia.

Quindi **impostare sempre una (o piu' se sono previste) password del BIOS**.

Naturalmente e' una metodica di sicurezza piena di limitazioni, infatti anche senza poter accedere fisicamente all'hardware della macchina e prevalicare la programmazione del CMOS, esistono (per ogni venditore di BIOS) una serie di password standard che permettono l'accesso ai menu' di configurazione in ogni situazione. Quanto meno ci vuole il tempo per provarle...

1.1.4 Sicurezza del boot loader

Ogni sistema operativo moderno ha necessita' di un boot loader per poter caricare il codice del kernel all'avvio della macchina. In GNU/Linux i due boot loader piu' utilizzati sono GRUB (GRand Unified Booloader) e LILO (LInux LOader). Per architetture non-Intel esistono degli oggetti equivalenti denominati SILO (Sun Sparc), MILO (DEC Alpha), PALO (HP-PA).

E' sempre consigliato **impostare una password per il boot loader**.

Questo ha l'effetto di impedire ad un soggetto che non la conosca di passare parametri al boot loader e quindi modificare, anche profondamente, il caricamento del sistema operativo. Passando parametri al boot loader rende l'intrusione in un sistema GNU/Linux una questione di pochi secondi.

LILO Il file di configurazione in genere si trova in */etc/lilo.conf*, le seguenti opzioni andrebbero sempre aggiunte:

time-out=00
resticted
password=<passwd>

La password in questo caso non e' criptata, per cui il file deve essere protetto da letture indesiderate (*chmod 600 /etc/lilo.conf*) e reso possibilmente immutabile (*chattr +i /etc/lilo.conf*). A questo punto occorre rilanciare lilo per rendere effettive le modifiche.

GRUB In questo caso la password puo' essere specificata in maniera criptata, come hash MD5.

Per prima cosa occorre generare una password o direttamente dalla linea di comando di grub, oppure attraverso l'utilita' *grub-md5-crypt* che genera la hash md5. A questo punto e' sufficiente specificare nel file di di configurazione grub.conf (presente in */etc*, */etc/grub* oppure */boot/grub*) la linea:

password –md5 <hash MD5>

La configurazione di grub e' dinamica, per cui non c'e' nessuna necessita' di lanciare alcun comando. Anche in questo caso come nel caso di lilo e' opportuno rendere il file non leggibile dai comuni utenti e immutabile.

1.2 Gli utenti

1.2.1 Disabilitare gli account speciali

Ogni installazione Unix crea una serie di account utente e gruppi che vengono utilizzati, per la maggior parte, per far girare i vari servizi del sistema. Non e' poi detto che questi servizi vengano installati o utilizzati sul server e in questo caso tutti questi account utente e gruppo dovrebbero essere disabilitati, anche se sono associati ad una shell non valida (tipicamente nologin).

Per fare questo gli account dovranno essere rimossi o con la gui fornita, oppure con i classici comandi:

userdel <username>
groupdel <groupname>

1.2.2 Rafforzare il meccanismo delle password

Per default la maggior parte delle distribuzioni implementano un meccanismo di verifica della forza delle password per mezzo di una serie di moduli PAM. Pero' molto spesso la lunghezza ammessa delle password e' troppo breve (in genere 5 caratteri). Il minimo richiesto per un sistema sicuro e' di 8 caratteri. Per fare questa modifica e' necessario modificare il file */etc/login.defs*, impostando il parametro:

PASS_MIN_LEN 8

In questo modo il programma login viene configurato opportunamente.

Shadow Password In tutti i sistemi dovrebbe essere impostato il meccanismo delle shadow password. Ovvero password cripatate presenti nel file */etc/shadow* che vengono associate agli account utente elencati in */etc/passwd*.

Limitare il comando su Il comando *su* permette di cambiare l'utente con cui si lavora. Se non sono stati posti particolari limiti, ogni utente puo' diventare root (previo inserimento della password). E' al contrario possibile, per mezzo di un modulo PAM, limitare gli utenti che possono eseguire il comando su. Fornendo un ulteriore layer di sicurezza. In genere questi utenti privilegiati vengono associati ad un gruppo (tipicamente wheel). Per mettere in piedi questo meccanismo e' necessario modificare il file PAM di configurazione del comando *su*, in genere presente in */etc/pam.d* e aggiungere o decommentare la riga:

auth required pam_wheel.so use_uid

Fatto questo gli utenti che devono poter eseguire *su* verranno aggiunti al gruppo wheel.

1.2.3 Configurare l'account root

L'account root e' il piu' privilegiato in un sistema Unix. Se l'amministratore si dimentica di fare logout e lascia la macchina senza lock il sistema diventa completamente vulnerabile. Per questo e' necessario fare in modo che la shell faccia un logout automatico trascorso un certo intervallo di tempo (che viene espresso in secondi).

Oltre a questo e' buona norma non conservare mai nella history la lista dei comandi che la root ha impartito.

Per ottenere questo nella bash shell e' sufficiente impostare due variabili built-in:

HISTSIZE=0
TMOUT=<timeout>

Un'altra accortezza e' quella di pulire lo schemo da tutto l'output al logout. Questo puo' essere ottenuto inserendo il comando *clear* nel file di configurazione *.bash_logout* di ogni utente. In questo file potrebbe anche essere prevista la rimozione della history con un: *rm -f .bash_history*

1.2.4 Disabilitare il console-quivalent access per gli utenti comuni

Gli utenti comuni non dovrebbero poter eseguire comandi che influiscono sul funzionamento di tutto il sistema (*halt, shutdown, reboot).* E' necessario quindi impostare i permessi di questi file in modo molto ristretto. Oltre a questo e' possibile rimuovere il console-equivalent access per tutta una serie di programmi:

*rm -f /etc/security/console.apps/**

Nel caso si utilizzi XDM per il login grafico, potrebbe essere necessario lasciare l'X server nella directory. Altrimenti solo la root potra' eseguirlo.

1.2.5 Impedire il root login da console diverse

Il file */etc/securetty* permette di specificare su quali dispositivi TTY l'utente root puo' fare login. E' buona norma commentare tutte le console di cui non abbiamo bisogno. In genere due sono piu' che sufficienti.

1.3 Configurazioni varie

1.3.1 Disabilitare e disinstallare tutti i servizi inutili

Sebbene sia possibile disabilitare i servizi che girano sul sistema andando ad agire sul file di startup/shutdown presenti nelle directory */etc/rc<runlevel>.d/* nei sistemi con inizializzazione SysV oppure in */etc/rc.d/rc.S* in quelli con inizializzazione BSD. Oppure utilizzare opportune interfacce piu' o meno grafiche fornite da ogni distribuzione. Tutti i file di startup/shutdown devono appartenere alla root e avere permessi ristretti (*0700*).

Il metodo migliore e' invece quello di disinstallare i servizi inutili, in base al vecchio motto degli elettronici: "tutto quello che non c'e' non si rompe e non fa rumore".

Oltre ai servizi che vengono avviati standalone, esistono quelli gestiti dal superserver inetd e piu' recentemente xinetd. In questo caso e' sufficiente:

- Nel caso di inetd, commentare (#) ogni servizio che non debba essere avviato nel file *inetd.conf* e poi riavviare il demone (*kill -HUP*)
- Nel caso di xinetd. Agire come al precedente punto se la configurazione e' monolitica (*xinetd.conf*) oppure porre la riga *disable=yes* in ogni file di interesse presente nella directory di configurazione */etc/xinetd.d* e poi riavviare il demone, se la configurazione e' distinta per ogni singolo servizio.

I file di configurazione dei servizi dovrebbero appartenere a root e avere permessi minimi indispensabili all'esecuzione e/o alla lettura da parte della root stessa.

Come ulteriore misura di sicurezza tutti i file che mandano in esecuzione servizi potrebbero essere resi immutabili a livello di filesystem (*chattr +i*).

1.3.2 Eliminare compilatori e pacchetti di sviluppo

Da un server di produzione dovrebbe essere eliminata la parte grafica (a meno che non sia utilizzato come terminal server), tutti i compilatori e i pacchetti di sviluppo, compreso il kernel. Infatti un possibile intrusore potrebbe trovarsi la strada spianata dalla possibilita' di compilare ed eseguire software. Oltre a questo sarebbe buona norma limitare al mount le operazioni che possono essere eseguite sui filesystem. Una classica opzione che si utilizza e' *noexec* da porre nella fstab in corrispondenza di tutti i filesystem che devono contenere solo dati e non codice eseguibile.

1.3.3 Connessione al sistema (SSH)

Ogni sistema orientato alla sicurezza deve prevenire ogni connessione con protocolli che prevedano lo scambio di password non criptate (telnet, ftp, snmp, ...), se non preventivamente tunnellizzati.

Per la connessione in console l'alternativa **obbligatoria** a telnet e' la suite ssh. Che oltre allo shell connection replacement, fornisce anche un utilita' di secure copy (scp) e un sub-server per il file transfer criptato (stfp). In questa disamina ci interessa analizzare la configurazione lato server. Il file di configurazione in genere si trova in */etc/ssh/sshd_ config*.

Vale la pena di far notare come sia necessario, salvo esigenze particolari, utilizzare solo il protocollo SSH v.2. La versione 1 soffre infatti per progetto

di una debolezza (nota come CRC32 compensation/insertion attack) intrinseca. Il server versione 2 puo' mettere in esecuzione la versione 1 solo se e' stato configurato in questo senso e se la versione 1 e' contemporaneamente installata.

Le linee che dovrebbero sempre essere presenti nel file sono:

Port 22
Protocol 2
ListenAddress <address>

- se si vuole limitare la connettivita' ad un particolare IP, altrimenti per default il listening e' su tutte le interfacce

PermitRootLogin no
LoginGraceTime <timeout>
StrictModes yes
PubkeyAuthentication yes
HostbasedAuthentication no
PasswordAuthentication yes

- sarebbe preferibile tenerla disabilitata in una macchina con indirizzo pubblico

PermitEmptyPasswords no
UsePAM yes
AllowTcpForwarding no
GatewayPorts no
X11Forwarding no
PrintMotd no
PrintLastLog no
ShowPatchLevel no
UsePrivilegeSeparation yes
PermitUserEnvironment no
Compression yes
UseDNS yes
PidFile /var/run/sshd.pid
MaxStartups <number>
Subsystem sftp /usr/sbin/sftp-server

Il metodo di connessione piu' sicuro e' quello con scambio di chiavi pubblica. In questo modo una passphrase viene utilizzata solo per decrittare la chiave privata sull'host da cui si origina la connesione e non passa mai in alcun modo sulla rete. Il tool ssh-agent permette di gestire il caching delle chiavi decrittate. Un minimo di lavoro preliminare garantisce pero' un'ottima sicurezza.

Qualora sia possibili si dovrebbe sempre evitare l'autenticazione con password. Sebbene lo scambio sia crittato con chiave simmetrica, un debolezza del protocollo potrebbe permette l'intercettazione della stessa.

1.3.4 TCP Wrappers

Sebbene i tcp wrappers siano meno popolari di un tempo. Quasi tutti i demoni possono essere compilati con il supporto ai tcp wrappers e garantire un

minimo di sicurezza base a livello di socket per tutti i servizi. I tcp wrappers (che non sono altro che una shared object library linkata all'eseguibile del demone) vengono configurati per mezzo di una coppia di file */etc/hosts.deny* e */etc/hosts.allow*. In un sistema in cui vengano implementati criteri minimi di sicurezza la politica deve essere di *default deny*. Ovvero negare l'accesso ad ogni servizio e poi aprire esplicitamente solo quelli necessari.

Il file *hosts.deny* dovra' contenere allora un'unica direttiva:

ALL: ALL@ALL, PARANOID

verra' cosi' negato l'accesso ad ogni servizio per qualsiasi utente proveniente da ogni host.

Nel file *hosts.allow* potranno essere abilitati i servizi a cui vogliamo concedere l'accesso, inserendovi righe della forma:

daemon_list: user@host

1.3.5 Modificare il file host.conf

E' il file (*/etc/host*.conf) *che* specifica come la resolving library risolve i nomi in indirizzi e, qualora di possegga un name server trusted interno alla struttura, come dovrebbe essere in ogni area server, deve contenere le seguenti direttive:

order bind, hosts
multi on
nospoof on

Il resolving viene allora eseguito prima rivolgendosi al dns e poi al file hosts. Se quest'ultimo fosse modificato, tutti i servizi che facessero reverse path resolving delle connessioni sarebbero protetti.

1.3.6 Proteggere il file services

Il file */etc/services* specifica l'associazione del nome dei servizi alle porte TCP e UDP. Questo file dovrebbe essere non alterabile. Questo puo' essere fatto a livello di filesystem (*chattr +i*).

1.3.7 Disabilitare la sequenza CTRL-ALT-DEL

Il processo init generalmente interpreta la sequenza CTRL-ALT-DEL come un segnale per passare al runlevel 6, ovvero quello utilizzato per il reboot del sistema. Naturalmente per un server che deve essere attivo 24/7 non e' ammesso che un utente distratto possa fare il riavvio da console con tanta facilita'. Per eliminare questo rischio e' necessario commentare la linea:

ca:ctrlaltdel:<reboot_program>

anteponendo un carattere #, nel file */etc/inittab*.

1.3.8 Disabilitare i programmi SUID/SGID non utilizzati

Il rischio e' che vari eseguibili nel sistema sono SUID root come default. Un utente che fa girare questi file lo fa con i privilegi del superutente. La maggior parte di questi software non sono necessari su un server e devono essere disabilitati i SUID e SGID bit che li caratterizzano. E' possibile crearne una lista con il comando:

find / -type f \(-perm 040000 -o -perm -02000 \) > testfile.txt

A questo punto si fa un analisi e si disabilitano i bit SUID e SGID con i comandi:

chmod u-s
chmod g-s

Un'altra possibilita' e' fornita dall'opzione di montare un intero filesystem in modo che i bit SUID e SGID vengano disabilitati per tutti i file. L'opzione da specificare nel file */etc/fstab* o alla linea di comando e' *nosuid*.

References

[1] Kapil Sharma - Linux Security Tips - Linux Gazette n.58

[2] O'Reilly - Simson Garfinkel, Alan Schwartz, Gene Spafford - Practical Unix & Internet Security, 3rd Edition

[3] The Linux Documentation Project - Linux Security HOWTO (http://www.tldp.org/HOWTO/html_single/Security-HOWTO)

Contents