Hardening di un Sistema Operativo

Un computer, sistema operativo più hardware, assolutamente *sicuro* non esiste. Nemmeno un computer spento, disconnesso e con i dischi azzerati può essere considerato assolutamente sicuro. Non solo, la definizione stessa di *sicurezza* è relativa a cosa si vuole proteggere. In pratica più un sistema è sicuro, più è difficile il suo utilizzo poiché le procedure di sicurezza si scontrano spesso frontalmente con la fruibilità. Questo implica anche che la *sicurezza* costa e riduce la produttività. D'altra parte, la *sicurezza* riduce i rischi.

La morale di tutto ciò è che invece di partire da un sistema molto sicuro per renderlo anche utilizzabile, la prassi è di partire da un sistema facilmente utilizzabile e cercare di renderlo più sicuro. Questo processo è spesso indicato con il nome di *Hardening del Sistema Operativo.*[1] In questo articolo verrà fatto un breve riassunto dei passi organizzativi, tecnici e pratici che l'esperienza insegna aiutano a rendere più sicuro un server. Anche se le indicazioni pratiche sono indirizzate a sistemi operativi di tipo Unix, le procedure sono generali. Rimane ovviamente il fatto che ogni realizzazione pratica richiederà comunque un approfondimento degli argomenti o la collaborazione di un esperto.

Bisogna sottolineare come l'*Hardening del Sistema Operativo* sia, da un punto di vista teorico, la cosa sbagliata da fare. Cercare di mettere in sicurezza qualche cosa che non è stato progettato con criteri di sicurezza è sempre un compito molto arduo. Il livello di sicurezza che si ottiene con questo procedimento è di norma relativamente basso. La possibilità di fare degli errori nella fase di *Hardening* è molto grande ed è quasi sicuro di dimenticarsi di mettere in sicurezza qualche programma, file ecc. Quello che si può ottenere è un sistema *ragionevolmente sicuro*, in cui la maggior parte dei possibili punti di intrusione sono bloccati e vi sono *ragionevoli* meccanismi di verifica, controllo (inclusi identificazione, autorizzazione ed accounting), alerting, recovering e backup.

Va inoltre ricordato che per molti sistemi operativi esistono programmi o procedure automatiche di *Hardening*, e pubblicazioni, anche degli stessi vendor, che ne descrivono l'approccio ed i dettagli.

1 Anche se questo argomento non rientra in maniera esplicita tra quelli di questa rubrica, il suo uso diffuso e l'importanza pratica che ricopre come punto di partenza per quasi ogni altra applicazione, ne impone una se pur breve trattazione.

Supponiamo di dover attivare un servizio informatico e per questo di dover acquistare e configurare un server apposito. I passi fondamentali devono essere fatti ancora prima dell'acquisto. Prima di tutto:

- il server deve essere utilizzato solo per il servizio informatico scelto, e non anche per altre attività che nulla hanno a che fare con questo;
- l'hardware deve essere scelto in modo che sia proporzionato solo al servizio che deve offrire, anche per non indurre in tentazioni future di *aggiungere* servizi diversi, ma ovviamente con un po' di spazio per la crescita del servizio stesso (nuove release di software, dati che si accumulano, aumento del traffico ecc.);
- bisogna valutare attentamente le questioni di ridondanza dell'hardware e le conseguenze di possibili guasti, pensare quindi se è necessario avere server in load−balancing, hot−standby, macchine di riserva, multipli dischi (RAID), doppie reti e così via.

La scelta del sistema operativo è più complicata:[2]

- il sistema operativo deve garantire prestazioni e stabilità per le applicazioni che deve sostenere;
- il vendor deve avere una nota e sperimentata politica di assistenza, tempi certi per la release dei patch di sicurezza e possibilmente una e−mailing list per i clienti tramite la quale vengano inviati gli annunci;
- è auspicabile che i patch di sicurezza e gli upgrade siano firmati dal vendor con chiavi PGP o simili, od almeno che siano disponibili le hash MD5; le chiavi PGP e gli hash MD5 devono essere ottenuti indipendentemente dai pacchetti e devono essere controllati periodicamente; nessun pacchetto deve essere installato se prima non sono state verificate le firme digitali e/o gli hash; anche recentemente si sono verificati casi di pacchetti in cui erano state inserite backdoor, la verifica dell'hash ha permesso di scoprire le modifiche in tempi molto brevi;
- è molto utile iscriversi ad e−mailing list quali quelle del CERT, BUGTRAQ o SANS per tenersi al corrente delle ultime vulnerabilità e di possibili rimedi temporanei in attesa dei patch dei vendor;
- infine, ovviamente, è necessario disporre di personale qualificato a gestire tale sistema operativo.

Una volta scelto il sistema operativo, prima di passare alla fase di installazione bisogna progettare la partizione del o dei dischi. Partizionare correttamente i dischi è un esercizio difficile ma che aiuta sia a proteggersi dalle intrusioni che a gestire meglio la macchina. In linee molto generali, un file−system può essere diviso in tre grandi zone:

2 In questo articolo non verranno presi in considerazione molti degli aspetti che fanno parte di un progetto globale di sicurezza, dai backup alla sicurezza fisica, alla progettazione della rete ecc.

1. file che non devono cambiare mai, eccetto che in caso di upgrade o patch del sistema: in questa classe sono tutti gli eseguibili di sistema e degli applicativi, ma anche i file di configurazione;
2. file che cambiano frequentemente ma in modo prevedibile: in questa classe sono tutti i file di log del sistema e degli applicativi, e tutti i dati degli applicativi stessi;
3. file temporanei che possono essere creati da diverse applicazioni in diverse condizioni: il tipico esempio è tutto quanto compare in `/tmp`, `/var/spool` e `/var/tmp` nel file−system unix.

Inoltre i file possono essere programmi eseguibili, file di dati, file speciali quali device, link, socket ecc., oppure programmi eseguibili che permettono di cambiare l'identità mentre girano, ovvero che sono lanciati da un utente ma girano con i permessi di un altro. Questi ultimi sono i più pericolosi e sono spesso indicati genericamente con il termine di `eseguibili SUID`. La soluzione migliore sarebbe dedicare una partizione apposta ad ogni tipo di file, dando ad ogni partizione il minimo di permessi necessari per i file che contiene. Ad esempio una partizione che contiene solo file di configurazione potrebbe essere montata come `nosuid, nodev, noexec` (ovvero che i file presenti nella partizione non possono essere SUID, non possono essere dei device e non possono essere eseguiti) e nel caso anche come `read−only` (questo però implica che per cambiare una configurazione è necessario ri−montare la partizione `read−write`). Analogamente una partizione per file temporanei può essere montata `nosuid, noexec`. Attenzione, il fatto di montare le partizioni con queste proprietà non garantisce che siano impossibili le intrusioni, infatti un cracker che sia in grado di ottenere il permesso di ri−montare le partizioni o ha accesso diretto al device fisico, può facilmente cambiare questi permessi. Questa configurazione rende ovviamente la vita più difficile ad un cracker poiché non gli sarebbe possibile scrivere od eseguire programmi dove gli piacerebbe senza ad esempio ri−montare le partizioni, ed aiuta in caso di problemi software e nella gestione dei backup e dei restore. Una applicazione impazzita che cerca di cancellare tutto il contenuto del disco si bloccherebbe di fronte ad una partizione montata `read−only`, oppure se una applicazione scrive i propri dati in una partizione a lei riservata anche se l'applicazione impazzisce e riempie la partizione, il sistema operativo non ne dovrebbe risentire.

In pratica è però impossibile partizionare i dischi in modo perfetto, e bisogna scegliere la soluzione più ragionevole per il sistema operativo e gli applicativi del caso.

Finalmente si può passare ad installare il sistema operativo:

1. bisogna procurarsi non solo il sistema operativo e gli applicativi, ma anche tutti i patch e gli upgrade del software sino al giorno corrente, controllare le firme digitali e/o gli hash;

2. predisporre la macchina da installare completamente disconnessa dalla rete; se per l'installazione è necessario connettersi ad una macchina remota, bisogna preparare la macchina remota, disconnetterla da ogni rete e creare una LAN solo per queste due macchine; questo punto va ribadito con forza, anche 10 minuti connessi ad internet con una macchina senza gli ultimi patch di sicurezza possono essere fatali;
3. installare il sistema operativo, gli applicativi e tutti gli upgrade e patch.

A questo punto abbiamo un sistema operativo con applicativi funzionante e in linea con tutti gli update e patch di sicurezza. Dobbiamo ora passare all'*Hardening* vero e proprio. I punti essenziali sono i seguenti:

1. verifica di cosa è installato e di quali programmi sono attivi;
2. disinstallazione dei programmi non utilizzati;
3. verifica e modifica della configurazione, dei permessi dei programmi installati;
4. installazione di sistemi di verifica dello stato del sistema, di protezione e di allarme.

Il primo punto è semplice, anche se noioso. Una volta attivato il sistema bisogna fare l'elenco dei servizi attivi (i comandi unix `ps, top` e `netstat` possono essere utili per ciò) e fare l'elenco del software installato. Se il software è installato in pacchetti, il programma che li gestisce è di solito in grado di offrire una lista di pacchetti installati. Ad esempio molte distribuzioni Linux usano `rpm` ed il comando `rpm –qa` da l'elenco di tutti i pacchetti installati. E' sempre comunque utile fare una lista di tutti i file eseguibili installati sulla macchina, per far questo si può utilizzare ad esempio il comando

```
find / –type f –perm +0111 –print
```

(una forma alternativa del comando è:

```
find / –type f –perm +u=x,g=x,o=x –print
```

si raccomanda inoltre di eseguire questo comando solo sulle partizioni locali e non su partizioni montate da altre macchine modificandolo nel caso come segue

```
find / \( –local –o –prune \) –type f –perm +0111 –print
```

oppure indicando esplicitamente tutte le cartelle da esaminare invece di /). Si dovrebbe in questo modo creare una lista di programmi installati ma non utilizzati. Una lista indicativa di programmi che spesso sono installati nelle piattaforme Unix ma che possono essere rimossi se non utilizzati, è:

- nfs
- portmap e procedure rpc (Sun)
- server di posta elettronica, pop, imap (di norma una macchina che non funziona da server di posta elettronica non dovrebbe avere bisogno di avere il relativo programma attivo, anche se

spesso per la configurazione scelta dal vendor non è facile disattivarlo; nel caso si può pensare a sostituire `sendmail` con `qmail`, `postfix` od altri programmi più sicuri)

- rsh, rlogin server (da sostituire con `ssh`)
- telnet server (da sostituire con `ssh`)
- ftp server (da sostituire con `ssh`)
- DNS server
- news (NNTP) server
- programmi per iirc, chat
- X11 od ambiente grafico (l'ambiente grafico non è di solito necessario al funzionamento di un server, anche se in alcuni sistemi operativi non è possibile disinstallarlo o anche solo fermarlo)
- auth (identd)
- finger server
- snmp server
- tftp server
- rexd server
- uucp

Ovviamente conviene disinstallare tutto ciò che non viene utilizzato, ma alle volte questo non è possibile e bisogna quindi solamente disattivare i programmi. Per far ciò bisogna configurare il sistema operativo in modo tale che questi programmi non siano attivati. Programmi sempre attivi (`daemon`) sono di solito attivati dai file di inizializzazione in `/etc/rc*`, bisogna perciò ispezionare questi file e le procedure che li controllano e quelle da essi avviate e disattivare i programmi non utilizzati.

Molti programmi sono però attivati a richiesta dal network daemon `inet`. E' fondamentale esaminare la configurazione di `inetd` di solito in `/etc/inetd.conf` ed eliminare tutti i programmi non utilizzati, in modo particolare `chargen, systat, echo, discard` ecc. Se alla fine nessun programma è attivato da `inetd` conviene disinstallarlo o disattivarlo del tutto. In caso contrario, se si ha la versione classica di `inetd` conviene integrarla con i `tcp_wrappers` di Wietse Venema, oppure sostituire `inetd` con `xinetd` od un'altro programma che abbia già integrata la possibilità di filtrare l'accesso e le condizioni di attivazione dei servizi.

Per dare un'idea quantitativa di questo lavoro basta citare che una installazione normale di una

distribuzione Linux (Suse, RedHat, Mandrake ecc.) occupa da 1GB a 2GB di spazio disco. Con una selezione accurata dei programmi e dei pacchetti è possibile scendere a 300MB. Con un po' più di sforzo (inclusa la ricompilazione del kernel) si può scendere sotto i 100MB ed in alcuni casi anche sotto i 32MB.

A questo punto solo i servizi necessari dovrebbero essere attivi o attivabili. Bisogna passare alla verifica della configurazione dei programmi. Oltre a verificare che tutti i file di configurazione siano corretti, bisogna fare delle verifiche più generali quali ad esempio:

- cercare tutti i programmi SUID e GUID (il comando

  ```
  find / -perm +u=s,g=s -print
  ```

  oppure

  ```
  find / \(-perm -2000 -o -perm -4000\) -print
  ```

  permette di elencare tutti questi file) e rimuovere il bit di *set user or group Id* a tutti quei programmi per i quali si può limitare l'accesso al solo amministratore;
- verificare che sia in uso il file di password `shadow`, che questo file abbia le protezioni corrette e che, se possibile, le password utilizzino MD5 e non DES come algoritmo di crittazione;
- verificare che non vi siano cartelle, programmi o file scrivibili da tutti gli utenti ad eccezione delle aree di scrittura temporanea (il comando

  ```
  find / -type l -prune -o -perm +o=w -print
  ```

  oppure

  ```
  find / -type l -prune -o -perm +0002 -print
  ```

  permette di elencare tutti questi file) ;
- verificare che i programmi abbiano i permessi necessari per il loro funzionamento e non oltre; per quanto possibile evitare che i programmi siano eseguiti come l'utente `root` (ovvero l'amministratore del sistema);
- verificare l'elenco degli utenti che hanno accesso alla macchina ovvero la lista di username presenti nel file di password, impedire l'accesso all'utente `root` (l'amministratore) da remoto ma permetterlo solo da console o da un altro utente della macchina (se necessario creare un utente da utilizzare per accedere alla macchina da remoto);
- verificare che le password utilizzate siano ben scelte utilizzando un programma di verifica (ad esempio `crack`) delle stesse e che nessun utente abbia accesso alla macchina senza una password; bisogna verificare anche che non sia possibile accedere ad applicazioni sia da locale che da remoto senza password o con le password di default inserite dal vendor, vedi ad esempio *`snmp`*, l'accesso a database, interfacce web di manutenzione ecc.;
- verificare che le cartelle in cui risiedono i programmi, i file di configurazione dei programmi

stessi ed i file di configurazione negli account degli utenti abbiano le protezioni corrette; si consiglia anche di creare file di configurazione vuoti e di bloccarli cambiandone il proprietario a quello di un altro utente (di solito `root`), il tipico esempio è il file `~/.forward` utilizzato dai server di posta elettronica per reindirizzare la posta ma che può anche essere usato per eseguire programmi da remoto ogni qual volta un certo messaggio arriva a quell'utente; creando un file vuoto e non modificabile in nessun modo dall'utente si può bloccare questo trucco; altri file di questo tipo sono ad esempio `host.equiv, .rhosts, .netrc, .exrc` ecc.; in maniera analoga se ad esempio il server httpd viene eseguito come l'utente `httpd` mentre tutte le pagine ed i file di configurazione del server stesso sono dell'utente `html` (e non sono scrivibili da tutti gli utenti) anche se un cracker riuscisse a pentrare nella macchina attraverso il server httpd non sarebbe in grado di modificare alcun file, ma solo di scrivere nelle aree temporanee che però, come detto precedentemente, dovrebbero essere in partizioni `noexec` e `nosuid` (a meno ovviamente che il cracker riesca anche ad ottenere le priorità di `root` o di un altro utente);

L'ultimo passo è l'installazione di sistemi di verifica dello stato del sistema, di protezione e di allarme. Tralasciando del tutto l'argomento di firewall, N−IDS ecc., consideriamo solamente il nostro server. E' necessario:

- avere un sistema di backup di tutti i file presenti sulla macchina, con copie integrali ad una data fissata mantenute per lunghi periodi di tempo off−site in luoghi sicuri;
- avere un sistema di logging sia interno alla macchina che su (almeno due) server remoti; installare sistemi di analisi dei log sia sulla macchina che sui server remoti che allertino gli operatori nel caso di problemi (per questo scopo vi sono molti programmi disponibili sia gratuiti che commerciali per tutte le piattaforme, e nei casi più semplici non è difficile realizzarne uno in proprio);
- installare sulla macchina almeno un sistema di verifica dell'integrità dei file, vedi ad esempio `tripwire` o `aide`; data una lista di file e cartelle sulla macchina, questi programmi ne memorizzano le proprietà (hash, dimensioni, permessi ecc.) e permettono poi di verificare se i file sono stati modificati, il tutto è fatto in sicurezza utilizzando algoritmi e chiavi crittografiche;
- installare sulla macchina un firewall locale (Personal Firewall); anche se la macchina si trova in una rete considerata sicura ed, ad esempio, si sono installati i `tcp_wrappers`, è assolutamente consigliato installare un firewall locale e configurarlo in modo che solo i protocolli e le porte veramente utilizzati siano aperti sia in INPUT che in OUTPUT.

In ogni caso, la protezione più importante è sempre quella di mantenere la macchina aggiornata con i più recenti upgrade e patch di sicurezza, e di utilizzarla esclusivamente per lo scopo originario.

Purtroppo, anche se quanto scritto sinora sembra molto, sono stati tralasciati molti possibili miglioramenti quali ad esempio:

- l'utilizzo di sistemi avanzati di indentificazione, autorizzazione ed accounting, quali `kerberos`, `one−time−password (S/KEY, OPIE)`, `Smart−Card` ecc.;
- l'utilizzo di sistemi di port anti−scanning ed in generale H−IDS;
- l'utilizzo di kernel *hardened* e di sistemi per limitare o prevenire i `buffer overflow`;
- l'utilizzo di sistemi MAC o RBAC.

Andrea Pasquinucci
Consulente di Sicurezza Informatica
pasquinucci@ucci.it

Per approfondimenti:
moltissima documentazione è reperibile sia sul Web che direttamente dai vendor; in particolare può essere utile consultare i documenti preparati dall'AUSCERT (Australian Computer Emergency Response Team) disponibili sul sito http://www.auscert.org.au/render.html?cid=1937