Aide: un semplice HIDS

Anche se non si tratta di difesa attiva, un Host Intrusion Detection System (HIDS) è oggi fondamentale in qualunque elaboratore, visto che anche elaboratori a prima vista considerati senza valore contengono informazioni importantissime per un attaccante. Un HIDS deve svolgere come minimo[1] un lavoro molto semplice, verificare che file importanti non vengano modificati.[2] Ad esempio l'esperienza recente del problema al kernel Linux (Novembre 2003) e le relative intrusioni in macchine Debian, Rsync e FSF (vedi [1]) ha mostrato ancora una volta come un efficace HIDS può rilevare molto velocemente una intrusione e prevenire danni maggiori. Anche se è un tool per alcuni versi un po' rudimentale ma molto efficace, mostriamo qui come utilizzare *Aide* (Advanced Intrusion Detection Environment)[3] che ben si presta a indicare alcune caratteristiche salienti di un HIDS.

1. Installazione

L'installazione è molto semplice. Per alcuni sistemi operativi o distribuzioni di Linux esistono pacchetti già compilati, altrimenti bisogna scaricare il sorgente di Aide e della libreria *mhash* di hash crittografici. La compilazione sia per mhash, da fare per primo, che per Aide, si riduce a tre semplici comandi

```
./configure /configure –prefix=/usr/local
make
make install
```

Per la compilazione di Aide sono a disposizione altre opzioni per il comando `./configure`, ad esempio `–with-zlib` che permette di usare file compressi o la possibilità di scrivere i dati in un database SQL, una lista di opzioni si ottiene dando il comando `./ configure –help.`

E' da notare come l'eseguibile di Aide generato dalla compilazione sia statico e non legato a librerie dinamiche, quindi il programma è estremamente portatile e per utilizzarlo necessita di soli tre file: l'eseguibile, il file di configurazione ed il database di verifica.

2. Configurazione

Il file di configurazione di Aide ed il database ci faranno capire il funzionamento del programma. Il

1 In realtà un HIDS potrebbe/dovrebbe svolgere molte altre funzioni ma in questo articolo ci limitiamo a discutere solo questa problematica.

2 In questo caso un nome più preciso sarebbe *File Integrity Checker.*

3 Aide è scritto in C ed gira su vari sistemi operativi di tipo Unix: AIX, BSD, Linux, SunOS/Solaris ecc.

file di configurazione, di solito `aide.conf`, è diviso in tre sezioni: la prima di definizioni generali, la seconda di *Macro* ovvero definizioni da utilizzare nel file di configurazione stesso (questa sezione non verrà considerata in questo articolo), e per ultima la sezione di selezione dei file. Nella Tabella 1 è riportato un esempio della prima parte del file di configurazione. Il parametro `database` indica il nome del file che contiene il database di verifica, mentre `database_out` il nome del file in cui verrà nel caso creato un nuovo database, i due nomi devono essere distinti. E' possibile poi comprimere i database riducendo lo spazio utilizzato (non compressi i database di norma utilizzano dai 5MB ai 30MB) ma aumentando l'utilizzo di risorse durante la creazione e verifica del database. Le righe più interessanti sono però le `Check Rules` che definiscono quali proprietà dei file devono essere memorizzate nel database e controllate nelle verifiche. Nella Tabella 2 sono indicati i significati di queste regole ed alcune regole create automaticamente da Aide. Ad esempio, la regola R, utilizzata molto spesso, dice che per ogni file a cui questa regola è applicata vanno verificate le proprietà `p+i+n+u+g+s+m+c+md5` cioè i permessi, il numero di inode, il numero di hard-links ... ed il checksum md5 del file, e se uno di questi parametri è modificato va segnalato un allarme. In Tabella 1 sono stati ad esempio anche definiti `Dev`, `Mtab` e `Logs` che possono essere utilizzati per i file di tipo *device*, per `/etc/mtab` e per i file di log che continuano a crescere.

A questo punto bisogna selezionare i file di cui vogliamo verificare le proprietà indicandoli nella terza sezione del file di configurazione. La sintassi è quella delle *regular expressions*, ma per un uso iniziale non vi è bisogno di approfondire questo argomento. In pratica ogni riga ha due colonne: nella prima si indica il path assoluto[4] di un file o di una directory od anche solo la prima parte di esso, in altre parole la regola va interpretata come "*tutto quello che comincia con*"; se però l'espressione termina con il carattere "$" viene interpretata come "*il file che ha esattamente questo nome*". Nella seconda colonna vi è la regola che specifica le proprietà da verificare. Invece se una riga comincia con il carattere "!" vuol dire di non fare alcuna verifica su "*tutto quello che comincia con*", oppure su "*il file che ha esattamente questo nome*" se l'espressione termina con "$";[5] in questo caso la seconda colonna è assente.

Bisogna fare molta attenzione all'utilizzo delle regular expressions poiché anche se sembrano molto semplici è facile farsi prendere la mano e costruire espressioni complicate che alla fine *non* fanno quello che ci si aspetta. Un esempio semplice e solamente indicativo di questa sezione del file di configurazione è riportato in Tabella 3 mentre in Tabella 4 è riportato un esempio del database interno.

---

4 Nel linguaggio delle regular expressions ad ogni riga viene preposto implicitamente un "^".

5 Si suggerisce di utilizzare "!" insieme a "$" per evitare che un attaccante possa nascondere file all'interno di rami ignorati di Aide.

3. Uso

E' chiaro quindi quale è il funzionamento di Aide: inizialmente vengono registrati nel database tutte le proprietà indicate per i file selezionati, poi per ogni file viene eseguito un controllo di queste proprietà. Se vi è discrepanza viene segnalato un allarme.

Nell'uso quindi vi sono le seguenti fasi: per prima cosa bisogna creare il database iniziale, usando il comando `aide --config=/dir/aide.conf –init;` bisogna poi copiare il `database_out` in quello utilizzato per la verifica. A questo punto si può regolarmente dare il comando `aide –config=/dir/aide.conf –check` per verificare se qualche file è stato modificato. Una semplice, sicura ma un poco dispendiosa procedura è quella di partire ad esempio con la sola regola "`/ R`", vedere cosa Aide segnala nelle verifiche, decidere se la segnalazione non indica un problema e cambiare la configurazione di conseguenza. Dopo qualche iterazione di questa procedura si può giungere ad una configurazione stabile. Va notato che periodicamente vi sono delle modifiche da ritenersi normali, quali la rotazione dei file di log o l'aggiornamento di programmi ecc. In questi casi va ricreato il database usando l'opzione `–init` oppure anche durante una verifica con l'opzione `–update`, altrimenti dopo poco la lista di segnalazioni non problematiche può diventare molto lunga e poco utile.

Se la gestione di Aide è molto semplice, potente ed intuitiva, quello che è lasciato totalmente all'utente è la gestione del database e del file di configurazione. E' ovvio che se un attaccante riesce ad accedere alla macchina, la prima cosa che cercherà di fare è di modificare o cancellare il database di Aide in modo da cancellare le proprie tracce. Per evitare questo una possibilità ad esempio è di mettere l'eseguibile, il file di configurazione e il database su di un supporto non riscrivibile, ad esempio un CDR, ed eseguirli periodicamente in modo automatico e manuale. Ovviamente il cambio di database richiede la creazione di un nuovo supporto non riscrivibile. Un altro approccio è di conservare l'eseguibile, il file di configurazione e il database su di una macchina sicura, copiarli sulla macchina da verificare, eseguire la verifica e poi cancellarli. Se questo è fatto con le opportune precauzioni, diventa difficile per un attaccante accorgersi immediatamente della presenza di Aide in quanto il programma è presente sulla macchina solo per pochi minuti.

Andrea Pasquinucci
Libero Professionista in Sicurezza Informatica
pasquinucci@ucci.it

Riferimenti Bibliografici

[1] http://www.wiggy.net/debian/ http://rsync.samba.org/ http://www.fsf.org/

[2] http://sourceforge.net/projects/aide http://www.cs.tut.fi/~rammer/aide/manual.html

[3] http://mhash.sourceforge.net/

```
# aide.conf
database=file:/dir/aide.db
database_out=file:/dir/aide.db.new
verbose=20
#gzip_dbout=yes
report_url=stdout
#check rules
All=R+a+sha1+rmd160+tiger
Norm=s+n+b+md5+sha1+rmd160+tiger
Dev=p+i+n+u+g+s+md5
Mtab=p+n+u+g+s+md5
Logs=p+i+n+u+g+S
```

Tabella 1. Esempio di definizioni generali in aide.conf

```
p:   permissions
i:   inode
n:   number of links
u:   user
g:   group
s:   size
b:   block count
m:   mtime
a:   atime
c:   ctime
S:   check for growing size
md5: md5 checksum
sha1: sha1 checksum
rmd160: rmd160 checksum
tiger: tiger checksum
crc32: crc32 checksum
haval: haval checksum
gost: gost checksum
R:   p+i+n+u+g+s+m+c+md5
L:   p+i+n+u+g
E:   Empty group
>:   Growing logfile: p+u+g+i+n+S
```

Tabella 2. Proprietà dei file

```
/bin R
/boot R
!/dev/pts
!/dev/shm
/dev/log$ p+n+u+g+s+md5
/dev Dev
/etc/mtab$ Mtab
!/etc/sysctl.conf$
!/etc/.aumixrc$
!/etc/adjtime$
/etc$ Dev
/etc R
/initrd R
/lib R
/lost+found R
/misc R
/mnt R
/root R
/sbin R
/usr R
!/var/lock
!/var/run
!/var/tmp
!/var/spool
!/var/cache
/var/log Logs
!/var/lib$
!/var/lib/ntp$
!/var/lib/ntp/drift$
!/var/lib/slocate$
!/var/lib/slocate/slocate.db$
/var R
```

Tabella 3. Esempio di selezione dei file

```
@@begin_db
# This file was generated by Aide, version 0.9
# Time of generation was 2004-01-14 10:46:44
@@db_spec name attr perm uid gid size mtime ctime inode lcount md5
/boot 3005 40755 0 0 2048 MTA3MzIwNjIzNQ== MTA3MzIwNjIzNQ== 2 4 0
/dev 2621 40755 0 0 118784 0 0 64001 21 0
/var 3005 40755 0 0 4096 MTA3MzIwNjU4NQ== MTA3MzIwNjU4NQ== 2 24 0
/etc/fstab 7101 100644 0 0 614 MTA3MjUyMzEyNA== MTA3MjUyMzEyNQ==
  64224 1 i7Xt+hWiHKn7MMquNq2otw==
/etc/mtab 6205 100644 0 0 211 0 0 0 1 KP/Xe95Bo7WguO9kplxLtw==
/etc/modules.conf 7101 100644 0 0 210 MTA3MjA0NDYwMQ==
  MTA3MjA0NDYwMQ== 64230 1 S5hgIeTQyGPB1TZ/fMnwcw==
@@end_db
```

Tabella 4. Esempio del contenuto del database