# NAGIOS: Una rete sotto controllo
## Parte 1: Installazione e configurazione

Controllare e monitorare adeguatamente un sistema informativo non è mai molto semplice, ma con Nagios è possibile farlo anche nel caso di reti complesse. Questo è il primo di una serie di articoli con cui Raoul Scarazzini ci mostra come procedere

**di Raoul Scarazzini**

Tutto quello di cui un amministratore di sistema necessita per svolgere in maniera ottimale il proprio lavoro è il controllo di tutte le risorse che deve gestire. Però ottenere il controllo totale risulta particolarmente complicato all'interno di situazioni complesse, come nel caso di sistemi estesi in cui operano decine di PC, ciascuno con una propria specifica e critica funzione.

Si potrebbe creare un elenco scritto delle cose da controllare ogni mattina appena arrivati in ufficio o creare degli script specifici che partano automaticamente e che generino delle notifiche, ma sarebbe meglio essere sempre previdenti e agire appena un problema si rivela, senza magari ricevere prima una telefonata che segnala un disservizio.

Cerchiamo di capire con questo primo articolo come si possa implementare un complesso sistema di controllo e monitoraggio delle funzioni critiche di un sistema informativo servendosi del software Open Source NAGIOS.

### Perché usare NAGIOS

Il mercato mette a disposizione delle aziende parecchie soluzioni commerciali per il controllo dei sistemi: Openview, distribuito da HP, dispone di un sistema di auto-rilevamento molto ben fatto, è un programma modulare (nel senso che a seconda di ciò che si vuole controllare, viene caricato un determinato "modulo" senza appesantire il sistema) e destinato ad un livello di utilizzo professionale, soprattutto a causa del costo. Presenta però alcuni lati negativi come la mancanza di un' interfaccia web e l'obbligo di visionare lo stato dei sistemi solo dal PC sul quale è installato.

Un'altra alternativa è IBM Tivoli-netview che per funzionalità e capacità non è da meno ad Openview, ma anch'esso risulta molto costoso e presenta un ulteriore lato negativo: gira solo su sitemi Unix AIX (che ne aggravano ulteriormente il costo).

Si fa presto a capire che seppur buoni, questi (come altri) programmi incidono sul bilancio aziendale in maniera consistente. Ecco perché ancora una volta l'alternativa più interessante è offerta dal mondo Open Source, precisamente da NAGIOS.

NAGIOS è un programma che consente di tenere sotto controllo servizi di rete, monitorare le risorse dei server come i carichi di lavoro e la capienza dei dischi, ricevere in tempo reale notifiche di malfunzionamenti via mail o addirittura SMS, possiede una interfaccia web completa, intuitiva e comprensibile, gira su tutti i sistemi unix-like (Linux in testa), è gratuito e rilasciato sotto licenza GPL (General Public License, http://www.gnu.org/copyleft/gpl.html).

Implementare NAGIOS in modo da monitorare un complesso sistema informativo consente di avere una efficiente alternativa ai software commerciali sopraelencati senza spendere nulla, a parte il tempo necessario per configurare a dovere questo utile software.

### Concetto e requisiti di NAGIOS

NAGIOS è un demone, un programma sempre attivo e residente in memoria. In base al contenuto dei file di configurazione effettua tutti i controlli che sono stati decisi ed in caso di malfunzionamenti invia le notifiche agli utenti interessati. Per effettuare le interrogazioni ai servizi, NAGIOS si serve dei cosiddetti "plugin" i quali altro non sono che dei programmi il cui unico scopo, una volta lanciati, è quello di interrogare questo o quel servizio e restituire un codice di uscita che NAGIOS interpreterà per effettuare le notifiche del caso.

NAGIOS possiede inoltre una interfaccia web (figura 1 e 2) che illustra, con una quantità considerevole di dettagli, lo stato di tutti i servizi, dei server e di quanto altro abbiamo deciso debba essere controllato.

A livello di requisiti, ciò di cui si necessita per compilare e far funzionare NAGIOS sono un compilatore C, che ci consentirà di compilare programma e plugin relativi, un web-server da configurare affinché visualizzi l'interfaccia web e la libreria "gd library" che consentirà all'interfaccia web di generare grafici e disegni. Ad ogni modo durante la fase di configurazione del programma, qualsiasi mancanza del sistema ci verrà segnalata in modo da potervi porre rimedio.

### Installare NAGIOS

Descriveremo l'installazione partendo dai file sorgente in modo da rendere l'articolo il più generale possibile. Sono disponibili comunque i pacchetti binari per le più comuni distribuzioni, ma come sempre, luoghi e posizioni dei file di configurazione ed eseguibili potrebbero variare sensibilmente. Un elenco dei pacchetti disponibili lo si può trovare a questo indirizzo: http://dag.wieers.com/packages/nagios/

A questo punto, per proseguire, è necessario prelevare i file sorgenti dal sito ufficiale di NAGIOS http://www.nagios.org. I file che ci interessano sono due, quello relativo al nucleo del programma: http://prdownloads.sourceforge.net/nagios/nagios-1.2.tar.gz?download e

quello relativo ai plugin: http://prdownloads.sourceforge.net/nagiosplug/nagios-plugins-1.3.1.tar.gz?download al momento in cui scrivo le versioni "stabili" dei pacchetti sono quelle rappresentate dal relativo nome di file: 1.2 per quanto riguarda il programma effettivo ed 1.3.1 per quanto riguarda i suoi plugin.

Prima di iniziare, è necessario creare un utente denominato "nagios" affinché la nostra compilazione vada a buon fine. Questo utente sarà colui il quale avvierà il demone. Non è obbligatorio che questo si chiami "nagios" (è possibile modificarlo in fase di configurazione), ma torna utile in quanto è lo standard di installazione:

```
# adduser nagios
```

Quindi, una volta scaricati i file procediamo con la decompressione del primo:

```
# cd /usr/src
# tar -xzvf nagios-1.2.tar.gz
# cd nagios-1.2
```

Questo comando creerà la directory /usr/src/nagios-1.2 nella quale dovremo entrare per far partire la configurazione, lanciando il comando ./configure con alcune opzioni che indicano il percorso nel quale vogliamo che il programma venga installato, la directory relativa ai CGI (i programmi che consentono all'interfaccia web di funzionare a dovere), la directory base del sito, l'utente con il quale nagios verrà eseguito ed il suo gruppo:

```
# ./configure –prefix=/usr/local/nagios –with-
cgiurl=/nagios/cgi-bin –with-htmurl=/nagios/ –with-
nagios-user=nagios –with-nagios-grp=nagios
```

È da notare che tutte le opzioni passate al comando qui sopra sono il default e che quindi sarebbe bastato digitare un ./configure per ottenere lo stesso risultato, in questo modo però, si capisce come personalizzare ulteriormente la propria installazione.

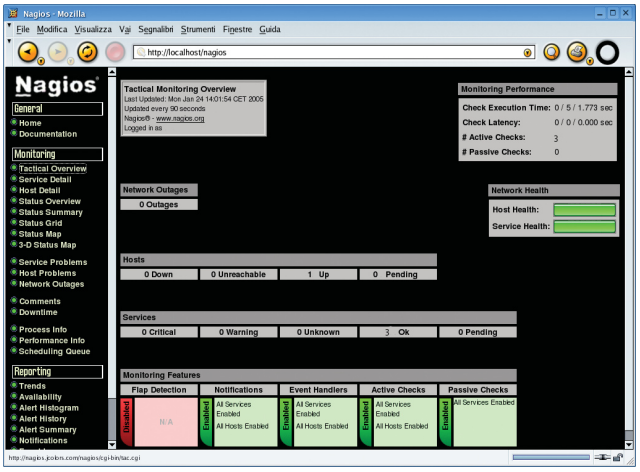


Figura 1

Completata la configurazione e verificato che tutto sia andato bene (in caso non lo fosse, ad esempio per problemi di mancanza di pacchetti, è necessario rimediare installandoli e rilanciare il comando) si può procedere con la compilazione effettiva seguita dall'installazione all'interno del nostro sistema dei file del programma:

```
# make all
# make install
```

Si procede poi con l'installazione dello script di init, che sarà necessario per avviare automaticamente il programma durante il boot del sistema e controllarne poi il funzionamento (in pratica per controllare le operazioni di start, stop, restart e reload del servizio).

Lo script nelle più comuni distribuzioni verrà posto nella directory /etc/rc.d/init.d:

```
# make install-init
```

Per concludere, verranno installati i file "sample" (di esempio) necessari per orientarsi durante la prima configurazione di NAGIOS:

```
# make install-config
```

Completiamo la fase di installazione installando i plugin con i seguenti comandi:

```
# cd /usr/src
# tar -xzvf nagios-plugins-1.3.1.tar.gz
# cd nagios-plugins-1.3.1
# ./configure  –prefix=/usr/local/nagios –with-
nagios-user=nagios –with-nagios-group=nagios
# make all
# make install
```

A questo punto l'installazione può dirsi terminata e la directory /usr/local/nagios/ avrà questo contenuto:

```
# cd /usr/local/nagios/
# ls
bin  etc  libexec  sbin  share  var
```

Tra le sottodirectory, bin/ contiene il file eseguibile nagios, /usr/local/nagios/libexec/ i file eseguibili dei plugin, etc/ i file di esempio e conterrà i file effettivi di configurazione, sbin/ gli script CGI per l'interfaccia web, share/ le pagine web "statiche" e la documentazione mentre var/ conterrà le informazioni di esecuzione di NAGIOS (file di lock e via dicendo).

Tutti i plugin contenuti in /usr/local/nagios/libexec/ sono file eseguibili da linea di comando. Questa qualità di NAGIOS (operare su eseguibili di sistema) ne estende le possibilità all'infinito. Qualsiasi sia la nostra esigenza, qualsiasi sia il servizio da monitorare, possiamo costruirci per conto nostro un plugin che ne verifichi il funzionamento.

A questo punto NAGIOS è pronto da configurare.

## Configurare NAGIOS

NAGIOS necessita di un'approfondita configurazione che, se fatta a regola d'arte, può soddisfare le più svariate richieste. Va aggiunto però, che imparare a padroneggiare NAGIOS in un solo articolo è impossibile, pertanto è bene definire cosa ci proponiamo di implementare in modo da capire le basi del programma ed acquisire conoscenze che ci consentano in breve di approfondire lo studio e di sperimentare.

Possiamo quindi limitarci all'obiettivo di configurare NAGIOS affinché monitori lo stato di una sola risorsa, il server "local_pc": se questo sia "vivo" (cioè se risponde ai ping), quanto spazio è presente sul disco e se il servizio HTTP è attivo. In caso di qualsiasi malfunzionamento, verrà allertato l'utente "lan_user_1".

Per ciascun file di configurazione ci baseremo sui sample che il programma di installazione ha creato per noi, copiandoli in un nuovo file senza il suffisso "-sample", ad esempio per il file hosts.cfg faremo come segue:

```
# cd /usr/local/nagios/etc/
# cp hosts.cfg-sample hosts.cfg
```

Editando il file ottenuto:

```
hosts.cfg
```

All'interno di questo file vanno poste le definizioni di ciascun host che intendiamo monitorare. Il formato degli inserimenti di questo file prevede la clausola define, seguita dal nome host e dalla definizione dei suoi parametri racchiusi tra due parentesi graffe.

La prima definizione, riguardante "generic-host", riguarda il template (modello) su cui si baseranno gli inserimenti e va perciò mantenuta. Per il resto, dovendo configurare un solo host, utilizzeremo la seguente dichiarazione:

```
# 'local_pc' host definition
define host{
  use  generic-host  ;Name of host template to use
       host_name                local_pc
       alias                    Server locale
       address                  192.168.1.1
       check_command            check-host-alive
       max_check_attempts       10
       notification_interval    120
       notification_period      24x7
       notification_options     d,u,r
       }
```

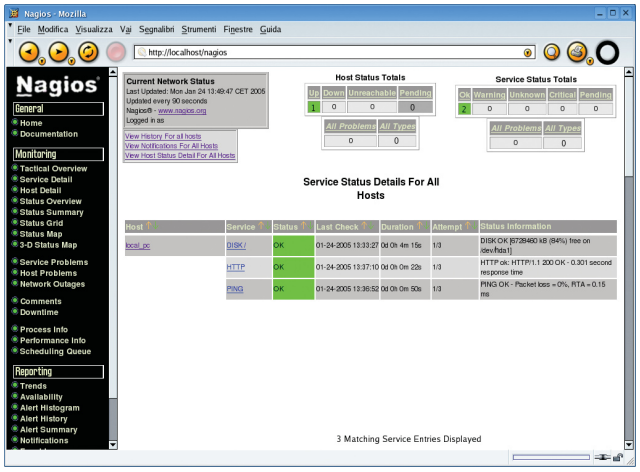


Figura 2

La definizione risulta abbastanza intuitiva in quanto ciascun parametro è auto-esplicativo. Il parametro più importante è "address" che identifica l'indirizzo della macchina, il quale potrebbe essere anche un sito internet, ad esempio www.esempio.com.

```
hostgroups.cfg
```

In questo file vengono definiti i gruppi di host e soprattutto il gruppo di contatto (che vedremo di seguito) ad essi associato. I membri del gruppo di contatto saranno allertati in caso di malfunzionamento. Nel nostro caso, il file conterrà:

```
# 'lan-servers' host group definition
define hostgroup{
       hostgroup_name  lan-servers
       alias           Gruppo gestione LAN
       contact_groups  lan-admins
       members         local_pc
       }
```

Abbiamo quindi configurato il gruppo "lan-servers" al quale il nostro host appartiene ed indicato per questo gruppo, i contatti di riferimento "lan-admins" che verranno dichiarati nel file che segue...

```
contactgroups.cfg

# 'lan-admins' contact group definition
define contactgroup{
       contactgroup_name       lan-admins
       alias                   LAN Administrators
       members                 lan_user_1
       }
```

Qui vengono indicati i gruppi di contatto, nel nostro caso, abbiamo dichiarato "lan-admins" nel quale sarà incluso l'utente deciso in partenza, "lan_user_1", i cui dettagli verranno dichiarati in...

```
contacts.cfg
```

All'interno di questo file possiamo definire i dettagli relativi ai contatti, nel nostro caso il solo contatto che vogliamo dichiarare è "lan_user_1":

```
# 'lan_user_1' contact definition
define contact{
 contact_name   lan_user_1
 alias          Network Administrator
 service_notification_period    24x7
 host_notification_period       24x7
 service_notification_options   w,u,c,r
 host_notification_options      d,u,r
 service_notification_commands  notify-by-email
 host_notification_commands     host-notify-by-email
 email          lan_user_1@localhost.localdomain
 }
```

services.cfg

Ultimo ma non per importanza, il file relativo ai servizi. Anche per questo è presente una template definita come "generic-service" che va obbligatoriamente inclusa in testa al file. Vanno poi incluse in questo file tutte le dichiarazioni dei servizi che vogliamo monitorare. Come deciso, controlleremo che la macchina sia attiva e raggiungibile dai ping, che non abbia il disco pieno e che su di essa sia attivo il servizio HTTP.

```
# Service definition
define service{
  use           generic-service
  host_name             local_pc
  service_description   PING
  is_volatile           0
  check_period          24x7
  max_check_attempts    3
  normal_check_interval 5
  retry_check_interval  1
  contact_groups        lan-admins
  notification_interval 120
  notification_period   24x7
  notification_options  c,r
  check_command      check_ping!200.0,20%!500.0,60%
  }

# Service definition
define service{
  use                   generic-service
  host_name             local_pc
  service_description   DISK /
  is_volatile                   0
  check_period          24x7
  max_check_attempts    3
  normal_check_interval 5
  retry_check_interval  1
  contact_groups        lan-admins
  notification_interval 120
  notification_period   24x7
  notification_options  w,u,c,r
  check_command  check_local_disk!20%!10%!/dev/hda1
  }
```

```
# Service definition
define service{
  use                   generic-service
  host_name             local_pc
  service_description   HTTP
  is_volatile              0
  check_period             24x7
  max_check_attempts       3
  normal_check_interval    5
  retry_check_interval     1
  contact_groups            lan-admins
  notification_interval    120
  notification_period      24x7
  notification_options     w,u,c,r
  check_command             check_http
  }
```

Osservando le dichiarazioni di questi servizi si fa presto a capire come possa essere regolato qualsiasi dettaglio, il periodo di tempo (la cui definizione è indicata nel file timeperiods.cfg il quale, a meno di casi particolari, è bene lasciare con il contenuto di default) di ventiquattro ore per sette giorni, il numero massimo di tentativi prima di inviare segnalazioni di malfunzionamento e così via.

Il parametro che indica il comando da eseguire per effettuare il controllo è check_command, di fianco al quale va dichiarato il nome del plugin da utilizzare. Se il comando necessita di parametri, ed è il caso del comando check_ping, questi vanno indicati di seguito, separati con un punto esclamativo.

I comandi si trovano nel file checkcommand.cfg ed è proprio qui che, nell'eventualità in cui si vogliano costruire plugin per proprio conto, ne vanno indicate le dichiarazioni.

Per quel che ci riguarda, checkcommand.cfg può rimanere com'è, così come resource.cfg.

cgi.conf

All'interno di questo importantissimo file, vi sono indicate le opzioni per la gestione dei CGI, fondamentali per l'utilizzo dell'interfaccia web. I CGI sono una serie di programmi che NAGIOS utilizza per visualizzare nelle pagine web le più svariate informazioni. Questi programmi, si trovano nella directory /usr/local/nagios/sbin.

In questo file, vanno abilitate le linee relative ai permessi di accesso. Nel nostro caso sarà utile definire che l'utente "local_user_1" potrà accedere tramite browser alla consultazione di tutte le informazioni di NAGIOS, pertanto, nelle righe relative andrà indicato come nell'esempio riportato:

```
authorized_for_system_information=local_user_1
```

Più avanti vedremo come configurare il web server affinché l'interfaccia web funzioni a dovere.

nagios.cfg

L'ultimo file da tenere in considerazione, è nagios.cfg. Questo file contiene tutte le impostazioni generali di programma, a partire dalla posizione dei file di configurazione fino ad arrivare ai permessi di accesso dell'interfaccia web. Le impostazioni di default sono in generale ideali ed in caso di esigenze specifiche si può ottenere il risultato desiderato facendo riferimento ai commenti.

Solo un'opzione ci interessa abilitare ai fini del nostro progetto:

```
use_authentication=1
```

Quest'indicazione obbligherà NAGIOS a richiedere l'autenticazione per l'accesso ai sopracitati CGI.

Tra i file rimanenti, dependencies.cfg contiene le dipendenze dei diversi servizi ed al suo interno va indicato ad esempio che la vitalità del sito dipende dall'esecuzione del web server e senza di questo non può funzionare. Nel nostro caso, questo file può rimanere vuoto, così come il file escalations.cfg in cui si possono definire ad esempio il numero totale di volte in cui una notifica deve essere segnalata.

### Configurare Apache affinché supporti NAGIOS

Prima di procedere con il primo avvio del programma, è bene predisporre Apache, il nostro web server, per il funzionamento di NAGIOS. All'interno del file httpd.conf vanno perciò immesse queste righe:

```
<Directory /usr/local/nagios/sbin>
AllowOverride AuthConfig
order allow,deny
allow from all
Options ExecCGI
</Directory>

<Directory /usr/local/nagios/share>
AllowOverride AuthConfig
order allow,deny
allow from all
</Directory>
```

In questo modo, oltre ad essere dichiarate, le directory di NAGIOS saranno anche protette (AllowOverride AuthConfig).

Per completare il discorso autenticazione, nella directory /usr/local/nagios/cgi, andrà creato un file denominato .htaccess con questo contenuto:

```
AuthName "Nagios Access"
AuthType Basic
AuthUserFile /usr/local/nagios/etc/htpasswd.users
require valid-user
```

La consultazione dei dettagli relativi allo stato dei servizi e degli host, raggiungibili attraverso l'indirizzo http://localhost/nagios, sarà protetta: cioè all'accesso verrà richiesto di indicare un nome utente ed una password.

Il file .htaccess indica ad Apache che i dati di accesso sono inclusi nel file /usr/local/nagios/etc/htpasswd.users. Questo file viene creato con il comando htpasswd in questo modo:

```
# htpasswd -c /usr/local/nagios/etc/
 htpasswd.users local_user_1
New password: ******
Re-type new password: ******
Adding password for user local_user_1
```

All'interno di htpasswd.users verranno registrati i dati di accesso l'utente "local_user_1" abilitato in precedenza nel file /usr/local/nagios/etc/cgi.conf.

Per creare altri utenti basterà rilanciare il comando senza l'opzione -c.

Nel caso in cui si volesse bloccare completamente l'accesso all'interfaccia web, sarà necessario inserire il file .htaccess anche nella directory /usr/local/nagios/share .

A questo punto, per applicare le modifiche è necessario avviare (o riavviare) il servizio di Apache in questo modo:

```
# /etc/init.d/httpd start
```

La fase di configurazione può dirsi completata.

### Avviare NAGIOS

Per avviare il demone di nagios è sufficiente lanciare il seguente comando:

```
# /etc/init.d/nagios start
Starting network monitor: nagios
  PID TTY          TIME CMD
22037 ?        00:00:00 nagios
```

Se l'output è come quello mostrato, NAGIOS è in esecuzione e non resta che aprire il browser all'indirizzo http://localhost/nagios per verificare che tutto sia come ci aspettiamo: in figura 1 è riportata la schermata definita "Tactical Overview" in cui c'è il riepilogo generale del sistema, mentre in figura 2, "Service Detail", è rappresentato il dettaglio dei servizi configurati.

### Conclusioni

È facile capire come configurando attentamente in NAGIOS la nostra rete, potremo avere decine e decine di host costantemente sotto controllo e ricevere in caso di qualsiasi malfunzionamento un'istantanea notifica.

Nel prossimo articolo vedremo come monitorare risorse remote che necessitano dell'esecuzione di plugin locali tramite NRPE: Nagios Remote Plugin Executor.

**Raoul Scarazzini -** rascasoft@tiscali.it
http://web.tiscali.it/rascasoft