Kapitel 18

Nagios och driftövervakning

18.1 Driftövervakning

När kunder eller kollegor ringer och klagar på att tjänster eller servrar är nere är det bra att, ärligt, kunna säga ”Jag fick varning om det nyss och jobbar på att lösa problemet”. Alla problem kan inte förutses, men många saker kan övervakas automatiskt. Till din hjälp finns program som Nagios, Cacti och Zabbix. Men jag börjar med att beskriva några enklare saker.

18.2 cronskript

Som du nog vet finns det en daemon som heter crond, med den kan man schemalägga regelbunden körning av program och skript. Det finns tre vanliga sätt att lägga upp cronjobb. Vill du köra dem som en vanlig användare, vilket ofta är bra, så kör du kommandot `crontab -e`. Då startas en editor (ofta vim eller nano) och du kan lägga till rader i den användarens crontab. Syntaxen finns beskriven i `man 5 crontab`. Ordningen för kolumnerna är minut (0–59), timme (0–59), månadsdag (1–31), månad (1–12 eller förkortat

namn), veckodag (0–7 där både 0 och 7 betyder söndag, eller de engelska trebokstavsförkortningarna). En * i något fält betyder alla av den sorten. Därefter kommer kommandot som ska köras.

Det kan bli tydligare med några exempel:

```
00 23 * * *   echo "klockan är 23:00"
01 00 1 * *   echo "En minut in på den nya månaden,
                    skicka tidrapport till chefen"
00 15 24 12 * echo "Kalle Anka"
00,30 7-19 * * fri   ps aux|grep quake
# kollar varje halvtimme mellan 7 och 19 på
# fredagar om nån spelar quake.
```

Kommandot som körs kan vara lite av varje, men tänk på att vissa miljövariabler, såsom $PATH, kan vara annorlunda för cronjobb. Så ange helst full sökväg, och lägg aldrig till cronjobb som kör skript som andra kan förändra (kataloger med 777, filer med 666 eller liknande rättigheter).

Ger skriptet du kör ingen utdata till STDOUT eller STDERR händer inget mer än att kommandot körs. Men kommer det utdata så är det så fiffigt anordnat att du får texten i ett mejl. Det bör du utnyttja. Det finns några standardprogram som körs via cron, den första jag beskriver är Logwatch.

18.2.1 Logwatch

För att börja använda Logwatch bör du bara se till att mejl till root kommer till någon användare som läser dem. Se sidan 114 om du inte redan har satt upp rätt alias.

Därefter är det bara att installera Logwatch som behöver göras. På RH kör du `yum install logwatch` och på Debian `apt-get install logwatch`. Därefter får du mejl varje natt med föregående dygns viktigaste händelser på datorn. Logwatch kollar diverse filer under `/var/log/` och sammanfattar det viktigaste.

Dessa mejl bör läsas regelbundet, och de är ganska snabblästa när man har vant sig vid vad som brukar stå i dem. Du ser intrångsförsök, vilka paket som har installerats och om datorn har startat om. Att läsa logwatch nästan dagligen är ett bra sätt att hålla koll på maskinerna.

18.2.2 kolladf

Här är en annan favorit som jag kallar kolladf:

```
#!/bin/bash
/bin/df -h |egrep '(9[89]|10.)%'
```

Den är tyst om ingen disk börjar bli full. På Debian/Ubuntu kan den behöva avslutas med en rad med `exit 0`.

Vill du bara kolla vissa filsystem anger du vilka efter `df`, och vill du ha varning tidigare eller senare än 98% så ändrar du regexpen. Detta skript kan givetvis köra som vanlig användare, med en egen rad i användarens crontab. Vill du köra den varje timme kan du lägga den i `/etc/cron.hourly`. Saker som ska köras varje timme, dag, vecka eller månad kan man lägga i de katalogerna, bara gör dem exekverbara så körs de av root vid respektive tidpunkt.

18.2.3 kolla_pgrep

Här är ett lite mer avancerat skript som bör specialanpassas lite för varje maskin:

```
#!/bin/bash
dator=zeus
for ps in syslogd klogd sendmail sshd ntpd named \
          dhcpd httpd
do
  if pgrep $ps >/dev/null; then
    :
  else
    echo "$ps kör ej på $dator!"
  fi
done
```

Det kollar så vissa processer finns. Det ska normalt sett vara tyst, och kan vara lämpligt att köra varje timme. Ändra namnet på datorn och listan efter `in` till de tjänster du kör. Utöver de ovan nämnda kan man kolla till exempel postgres och mysqld. Tänk på att processer heter lite olika på

olika system, Apache heter ju `httpd` på RH och `apache2` på Debian.

Du bör även kontrollera tjänsterna ”utifrån”, så de svarar på rätt port och ger rätt svar. Men detta lilla skript är ett bra komplement. Några processer som ibland kan ”försvinna” är ntpd (om klockan går för mycket fel) och imap (t.ex. vid tidshopp bakåt).

18.2.4 nyttetc

Även det här skriptet är oftast tyst:

```
sudo cat /root/bin/nyttetc
#!/bin/bash
find /etc -mtime -1 |egrep -v '^/etc$'|
  egrep -v '^/etc/prelink.cache$'|
  egrep -v '^/etc/ircd/links.txt$'
```

De flesta filer i `/etc` ändras sällan, så det kan vara bra att få en daglig lista de gånger något har ändrats. Jag brukar köra detta vid 18:00 och det bör köras av root. Det går att luras med mtime, så räkna inte med `nyttetc` som viktig säkerhetskontroll, mer för att se normala förändringar: om någon annan root eller sudoer har ändrat något, om någon har bytt sitt lösenord (`shadow`), uppdateringar, reboot (`mtab`) osv. Du kan behöva anpassa vilka filer den inte ska visa, dessa är bara ett förslag.

18.2.5 Talande ping

Med ping kan man kolla om datorer är nåbara, se stycke 6.5 på sidan 51. Ping går att använda på många sätt för driftövervakning, här är ett kombinerat med talsyntes:

```
#!/bin/bash
for d in zeus hera dionysos apollon hermes \
         poseidon einstein.df.lth.se
do
  if ! ping -c5 -w10 $d |
    grep "bytes from" &>/dev/null; then
```

```
    echo "down, down. $d is down"|
      /usr/bin/festival --tts
    echo "$d is down"
  fi
done
```

Man ska alltså köra den på en klient, inte på varje server. För att undvika falsklarm är det bra om man har stabil internetanslutning på den dator där man kör det, och den måste givetvis ha ljudkort och köra något Linuxliknande operativsystem. Jag kör ett liknande skript via cron varje minut på min hemmadator. Begreppet ”tyst skript” blir här bokstavligt. Det händer att jag blir väckt av orden ”down, down. einstein.df.lth.se is down” eller motsvarande för andra datorer (inte de namnen här ovan). Tidigare hade jag $d först i det den säger, men lade till två down för att få lite förvarning när den börjar prata. Den andra echo-satsen som inte pajpas till festival gör att man även får mejl.

Prova gärna att köra `echo datornamn|festival –tts` för dina datorers namn så du får höra om den kan uttala dem. Programmet festival finns i paketet med samma namn på både Red Hat och Debian. En kul bug/feature är att den uttalar .se som dot-south-east, men det är ju början av domännamnet som är mest relevant.

18.3 Nagios

Nagios är ett bra system för driftövervakning. Man listar datorer och tjänster och får information om problem via mejl och webbsidor. Nagios hette tidigare NetSaint och har funnits sen 1999.

Första gången som jag installerade Nagios så slog jag ihop alla direktiv till en enda fil, men nu har jag insett fördelarna med att behålla det uppdelat. Man har datornamn i t.ex. `/etc/nagios/objects/datorer.cfg`. Raderna för varje server kan se ut så här:

```
define host{
        use             linux-server
        host_name       www
        address         85.235.7.95
        }
```

Den ärver standardinställningarna för ”linux-server”, som jag har definierat i filen `templates.cfg`. Dessa rader kan se ut ungefär så här:

```
define host{
        name                    linux-server
        use                     generic-host
        check_period            24x7
        check_interval          5
        retry_interval          1
        max_check_attempts      10
        check_command           check-host-alive
#ao130113 Alltid
#       notification_period     workhours
        notification_interval   120
        notification_options    d,u,r
        contact_groups          admins
#ao130113 lade till grupp
        hostgroups              linux-servers
        register                0
        }
```

Det är standardinställningarna, utom för de två som jag kommenterat. Fördelen med att ha en ”template” är att du inte behöver upprepa för varje ”host”.

Varje tjänst som du vill kontrollera lägger du till i din `datorer.cfg`. Även services ärver standardinställningar från `templates.cfg`, med namn som generic-service. En tjänst som kollar ping av alla definierade ”hosts” kan se ut så här:

```
define service{
    use                 generic-service
    host_name           *
    service_description PING
check_command       check_ping!300.0,20%!500.0,60%
    }
```

Och en som kollar HTTP-anrop till två specifika värdar så här:

```
define service{
    use                 generic-service
    host_name           www, einstein.df.lth.se
    service_description http
    check_command       check_http
    }
```

Varje kommando som du använder vid ”check_command” i service-definitionen måste definieras i `commands.cfg`. Där används ofta plugins som du kan installera med `yum install nagios-plugins-all`. Exempel på rader i `commands.cfg`:

```
# 'check_http' command definition
define command{
    command_name  check_http
command_line $USER1$/check_http -I $HOSTADDRESS$ $ARG1$
        }
# 'check_local_load' command definition
define command{
    command_name  check_local_load
command_line $USER1$/check_load -w $ARG1$ -c $ARG2$
    }
```

Dessa kommandon kan få gränserna för Warning och Critical från service-definitionen. Text efter första ! blir $ARG1$ och det efter nästa utropstecken blir $ARG2$. Se raden med check_ping på förra sidan. Även $HOSTADDRESS$ får sitt rätta värde för varje anrop. Bland alla plugins kan dessa vara lämpliga att börja med att använda: `check_ping`, `check_http`, `check_imap`, `check_smtp` och `check_tcp` (för godtyckliga protokoll och portar). Man kan även kolla status via SNMP, och via SSH kan man köra kommando på respektive server.

Om du vill kolla vilka argument som går att ha till en plugin så kör du t.ex:

```
/usr/lib64/nagios/plugins/check_imap -h
```

Det går även att provköra plugins direkt, innan du lägger in dem i Nagios-konfigurationen.

Bästa sättet att få varningar från Nagios är via mejl. I `contacts.cfg` så ändrar du raden med ”email” till en mejladress till dig, root, ett alias eller en Mailman-lista.

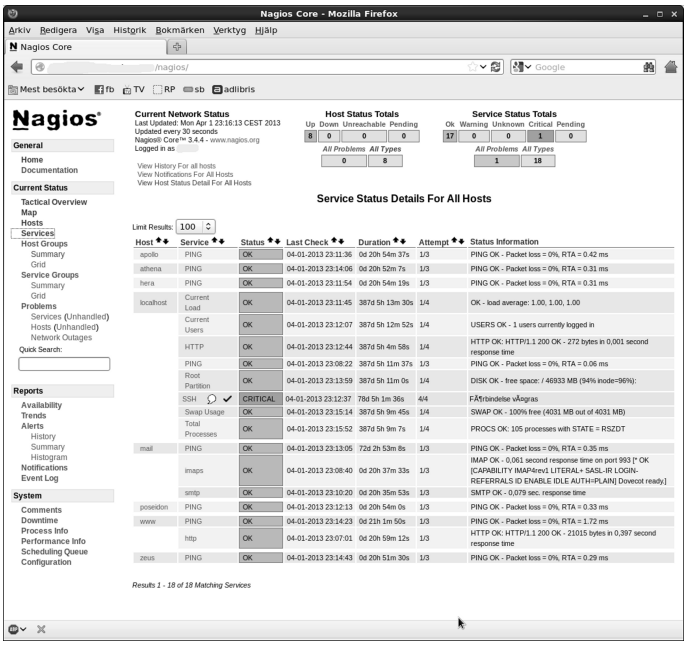
I `timeperiods.cfg` kan du ange vilka tider som det är ok att kontakta, men jag kör med 24x7 för alla.

Vid nyinstallation av Nagios bör du börja med att kolla och ändra i `nagios.cfg` Bland annat lägga till

cfg_file=/etc/nagios/objects/datorer.cfg

Filen `nagios.cfg` är väl dokumenterad, och standardvärdena brukar fungera bra. I `cgi.cfg` bör du ändra till en annan användare i authorized_for_all_services= och authorized_for_all_hosts=

I figur 18.1 kan du se hur Nagios kan se ut.



Figur 18.1: *Nagios*

18.4 Lästips

En grundläggande, men lite föråldrad, bok om Nagios är
Wolfgang Barth (2006), *Nagios*. No Starch Press
Nyare, men ganska avancerad, är
Schubert et al (2008), *Nagios 3 Enterprise Network Monitoring*. Syngress.