Instalación de Nagios core 4, pnp4nagios, check_mk y Nagvis en Debian 8 Jessie.

Introducción.

Ya tenemos fresquita la nueva versión de Debian y nada mejor para estrenar que probar nuestro paquete favorito de software de monitorización en Debian 8: Nagios core 4, pnp4nagios, check_mk y Nagvis. Todo en sus últimas versiones a fecha de publicación de este artículo.
Ya tratamos previamente la Instalación de estos componentes de monitorización con Nagios Core 4 en Debian 7 (sin Nagvis), en Redhat/Centos 6 y en Redhat/Centos 7 así como con Nagios 3 en todos ellos. En este caso nos toca actualizar la guiá de instalación para Debian 8.

Software usado en la instalación

- S.O. Debian 8 Jessie
- Nagios Core 4.1.0rc1 (DIY Source)
- Monitoring-plugins v 2.1.1 (debian 8 package)
- PNP4Nagios 0.6.25
- Check_mk 1.2.6p2
- Nagvis 1.8.2

Aviso. Si no quieres complicarte la vida y tener todo este software ya empaquetado e instalado en cinco minutos puedes probar OMD Distro que es un paquete de software que trae todo listo y configurado para usar en un "pis pas" (aunque no con las últimas versiones habitualmente). Te preguntarás cual es entonces la razón de complicarse la vida instalando todo esto por separado... Básicamente tener las últimas versiones de producto y la capacidad de actualizar cada componente por separado, lo cual no es sencillo con OMD. Vamos... tener el control. En cualquier caso si eres nuevo y quieres cacharrerar sin duda te recomiendo mejor omdistro. Y si puedes pagar pues quizá la mejor opción es directamente la versión de pago OMD.

Consideraciones importantes.
En esta guiá se van a realizar solo dos cambios importante en cuanto a las instalaciones por defecto de cada software:

- Se va a intentar que todos creen los ficheros de configuración en el directorio /etc/producto. Nagios y pnp4nagios se le indicará en la instalación. Check_mk lo hace por defecto. Nagvis lo dejaremos en su ubicación "normal". Si quieres instalar todo por defecto puedes prescindir en las instalaciones de indicarle la ruta a /etc.
- La URI de acceso a cada sitio se modificará para que sea http://server/site01/producto. (se le añadirá a esta el "/**site01/")**. Detrás de esta configuración hay una razón importante relacionada con el uso de múltiples sitios desde la consola CMK Multiste y el uso de apache mod_proxy para acceder a diferentes sitios enlazados con una única consola MK Multisite. Con este tipo de instalación ya estaremos preparados para su uso con MK Multisite. Donde pone "site01" puedes poner cualquier cosa que se ajuste más a tus necesidades: la ciudad, el pais,... lo que identifique una instalación concreta. En posteriores artículos trataremos este tema.

Debian 8 Jessie.

La instalación inicial de Debian 8 es la realizada por defecto desde una ISO sencilla "netinst". En el momento de instalación se desmarca la opción de entorno de escritorio. Se dejan solo las opciones:

- SSH server
- Utilidades estándar del sistema

Nagios core 4.

La versión de Nagios que incluye Debian 8 es aún Nagios 3. Como lo que queremos es Nagios 4 debido a que incluye importantes mejoras de rendimiento vamos a tener que compilar desde fuentes.

Requisitos previos.

Instalamos algunos paquetes que vamos a necesitar y activamos la ejecución de cgi para Nagios y el módulo rewrite de apache que lo usará posteriormente Pnp4nagios

```
apt-get install build-essential apache2 libapache2-mod-php5 libgd2-xpm-dev unzip
a2enmod rewrite
a2enmod cgi
```

Creamos cuenta para Nagios (por defecto crea el grupo) y asignamos password:

```
useradd -m -s /bin/bash nagios
passwd nagios
```

Creamos un grupo nagcmd que usaremos luego para los comandos externos entre otras cosas y metemos en dicho grupo a nuestro usuario Nagios y al usuario que usa apache.

```
/usr/sbin/groupadd nagcmd
/usr/sbin/usermod -a -G nagcmd nagios
/usr/sbin/usermod -a -G nagcmd www-data
```

Instalación de Nagios Core 4.

Bajamos Nagios 4 (core) en su versión "free", descomprimimos y compilamos (asignándole el grupo). Nos irá guiando por todos los pasos.

```
./configure --with-command-group=nagcmd --with-httpd-conf=/etc/apache2/sites-enabled
--sysconfdir=/etc/nagios
make all
make install
make install-init
make install-commandmode
make install-config
make install-webconf
make install-exfoliation
# Establecemos password para usuario nagiosadmin
htpasswd -c /etc/nagios/htpasswd.users nagiosadmin
service apache2 restart
```

El script de inicio que nos instala no es para Debian sino para Redhat y similar. Si intentamos activarlo para que arranque al inicio vemos que nos dará un error::

```
systemctl enable nagios.service
```

```
Synchronizing state for nagios.service with sysvinit using update-rc.d...
.....
Executing /usr/sbin/update-rc.d nagios enable
update-rc.d: error: nagios Default-Start contains no runlevels, aborting.
```

No le demos vueltas. Le ponemos en un script de inicio nagios 4 para Debian 8. Otra opción sería bajar al paquete de Nagios 3 para Debian 8, extraer el fichero, modificarlo / ajustarlo, probarlo y colocarlo. Pero eso es precisamente lo que hice para Debian 7 y está perfecto para Debian 8. Lo bajas y sustituyes (previa copia) el instalado.

```
cp nagios-daemon-debian8 /etc/init.d/nagios
```

Ahora podemos configurarlo para que arranque en el inicio del sistema.

```
systemctl enable nagios.service
systemctl start nagios.service
```

Podemos ya acceder a Nagios: http://server/nagios/
Si no usamos /etc/nagios para los ficheros de configuración **revisa las rutas del script**.

Veremos los servicios de localhost con el error "(No output on stdout) stderr: execvp(/usr/local/nagios/libexec/check_users, ...) failed. errno is 2: No such file or directory". Es normal ya que no hemos instalado ni configurado los plugins así que no los encuentra en esa ruta.

Nagios Plugins

Las opciones de instalación de plugins son varias:

- Instalar los plugins integrados por paquetes en Debian 8 (que son monitoring-plugins).
- Instalar monitoring-plugins en su versión actual pero compilando.
- Instalar nagios-plugins compilando.

Si no entiendes que es eso de nagios-plugins y monitoring-plugins leete el artículo "¿Revolución en la comunidad Nagios?"

Optamos por comodidad por la primera opción. La versión de monitoring-plugins es la 2.1.1 y no tenemos mucho que perder con las pequeñas modificaciones en las últimas versiones de los plugins.

```
apt-get install monitoring-plugins
```

IMPORTANTE: Dado que no hemos instalado la distribución oficial de los plugins de Nagios debemos decirle a este donde localizar los plugins. Para ellos editamos /etc/nagios/resource.cfg y cambiamos la ubicación de los plugins como vemos:

```
#$USER1$=/usr/local/nagios/libexec
$USER1$=/usr/lib/nagios/plugins
```

Probamos ahora a reiniciar el demonio de Nagios

```
systemctl restart nagios.service
```

Accedemos a Nagios y vemos que al rato ya funcionan correctamente los chequeos de localhost.

Host	Service	Status	Last Check	Duration	Attempt	Status Information
localhost	Current Load	OK	05-01-2015 18:10:44	0d 0h 0m 16s	1/4	OK - load average: 0.01, 0.07, 0.05
	Current Users	OK	05-01-2015 18:10:51	0d 0h 0m 9s	1/4	USERS OK - 1 users currently logged in
	HTTP	OK	05-01-2015 18:10:57	0d 0h 0m 3s	1/4	HTTP OK: HTTP/1.1 200 OK - 11378 bytes in 0,025 second time
	PING	OK	05-01-2015 18:10:26	0d 0h 0m 34s	1/4	PING OK - Packet loss = 0%, RTA = 0.07 ms
	Root Partition	OK	05-01-2015 18:09:22	0d 0h 1m 38s	1/4	DISK OK - free space: / 7061 MB (78% inode=88%):
	SSH	OK	05-01-2015 18:10:00	0d 0h 1m 0s	1/4	SSH OK - OpenSSH_6.7p1 Debian-5 (protocol 2.0)
	Swap Usage	OK	05-01-2015 18:09:37	0d 0h 1m 23s	1/4	SWAP OK - 100% free (465 MB out of 465 MB)
	Total Processes	OK	05-01-2015 18:10:15	0d 0h 0m 45s	1/4	PROCS OK: 27 processes with STATE = RSZDT

PNP4Nagios

Requisitos previos.

Instalamos unos paquetes necesarios previamente.

```
apt-get install php5-gd librrds-perl rrdtool
```

Instalación y configuración inicial.

Bajamos pnp4nagios e instalamos. Durante la instalación nos indicará las rutas de directorios. Conviene tomar nota.

```
tar zxfv pnp4nagios-0.6.25.tar.gz
cd pnp4nagios-0.6.25/
./configure --with-httpd-conf=/etc/apache2/sites-enabled --sysconfdir=/etc/pnp4nagios --with-base-url=/site01/pnp4nagios
make all
make fullinstall
```

Editamos el fichero /etc/apache2/sites-enabled/pnp4nagios.conf y cambiamos a la ruta correcta del fichero de autentificación de Apache:

```
        AuthUserFile /etc/nagios/htpasswd.users
```

Reiniciamos apache

```
systemctl reload apache2.service
```

Accedemos a la URL de pnp4nagios para verificar la configuración:
http://server/site01/pnp4nagios/

Realiza los chequeos del entorno y debe aparecer todo en verde (OK). Nos indica:

"Your environment passed all requirements. Remove or rename the /usr/local/pnp4nagios/share/install.php file now."

Le hacemos caso y renombramos dicho fichero:

```
mv /usr/local/pnp4nagios/share/install.php /usr/local/pnp4nagios/share/install.php.ORI
```

Recargamos la página de pnp4nagios y obtenemos un error: Please check the documentation for information about the following error.perfdata directory "/usr/local/pnp4nagios/var/perfdata/" is empty. Please check your Nagios config. Read FAQ online

En este punto de la instalación es normal. Necesitamos aún realizar la configuración final.

Configuración de pnp4nagios.

Pnp4nagios permite varios opciones para configurar la integración con Nagios Core. Vamos a usar la opción denominada “Bulk-mode” ya que es la mejor de las disponibles para Nagios 4. La opción “bulk mode with npcdmod” es más sencilla pero no es valida para la versión 4 de Nagios ya que usa un broker que no es compatible con los nuevo modulos de broker de Nagios 4.

Tenemos ejemplos de configuración listos para copy / paste en /etc/pnp4nagios

Del fichero nagios.cfg-sample copiamos SOLAMENTE (mira bien lo que copias) el siguiente texto al fichero de configuración de nagios, /etc/nagios/nagios.cfg (al final p.e.)

```
#
# Bulk / NPCD mode
#

process_performance_data=1

# *** the template definition differs from the one in the original nagios.cfg
#
service_perfdata_file=/usr/local/pnp4nagios/var/service-perfdata
service_perfdata_file_template=DATATYPE::SERVICEPERFDATA\tTIMET::$TIMET$\tHOSTNAME::
$HOSTNAME$\tSERVICEDESC::$SERVICEDESC$\tSERVICEPERFDATA::
$SERVICEPERFDATA$\tSERVICECHECKCOMMAND::$SERVICECHECKCOMMAND$\tHOSTSTATE::
$HOSTSTATE$\tHOSTSTATETYPE::$HOSTSTATETYPE$\tSERVICESTATE::
$SERVICESTATE$\tSERVICESTATETYPE::$SERVICESTATETYPE$
service_perfdata_file_mode=a
service_perfdata_file_processing_interval=15
service_perfdata_file_processing_command=process-service-perfdata-file

# *** the template definition differs from the one in the original nagios.cfg
#
host_perfdata_file=/usr/local/pnp4nagios/var/host-perfdata
host_perfdata_file_template=DATATYPE::HOSTPERFDATA\tTIMET::$TIMET$\tHOSTNAME::
$HOSTNAME$\tHOSTPERFDATA::$HOSTPERFDATA$\tHOSTCHECKCOMMAND::
$HOSTCHECKCOMMAND$\tHOSTSTATE::$HOSTSTATE$\tHOSTSTATETYPE::$HOSTSTATETYPE$
host_perfdata_file_mode=a
host_perfdata_file_processing_interval=15
host_perfdata_file_processing_command=process-host-perfdata-file
```

Del fichero misccommands.cfg-sample copiamos SOLAMENTE la siguiente configuración al fichero de config. de nagios /etc/nagios/objects/commands.cfg. **Descomenta las líneas!!**

```
# Bulk with NPCD mode
#
define command {
        command_name    process-service-perfdata-file
        command_line    /bin/mv /usr/local/pnp4nagios/var/service-perfdata /usr/local/pnp4nagios/var/spool/service-
perfdata.$TIMET$
```

```
}

define command {
      command_name    process-host-perfdata-file
      command_line    /bin/mv /usr/local/pnp4nagios/var/host-perfdata /usr/local/pnp4nagios/var/spool/host-perfdata.
$TIMET$
}
```

Configuramos npcd para inicio automático y reiniciamos:

```
systemctl enable npcd.service
systemctl start npcd.service && systemctl restart nagios.service
```

Ya debería funcionar perfectamente el entorno de pnp4nagios.

http://server/site01/pnp4nagios/

Tendremos que esperar o forzar algunos de los chequeos de servicios de localhost p.e. para que empiece a generar gráficas.

Configurar enlaces a gráficas.

Podemos configurar un icono de acceso directo a la gráficas de host / servicios desde estos en Nagios. Incluso un gráfico flotante. Realmente mi intención es no usar el interface gráfico de Nagios si no el de Check_mk. Este último configura la integración con pnp4nagios de forma automática así que no es necesario.

En cualquier caso si quieres configurar esta integración con Nagios se explicaba en el artículo "Nagios Core 4 + PNP4Nagios. Instalación y configuración desde fuentes en Debian 7 (wheezy)."

Check_mk

Requisitos previos.

Instalamos sudo y el módulo de python necesario para Apache.

```
apt-get install libapache2-mod-python sudo
```

Instalación.

Ojo. Para compilar check_mk tendrás que tener al menos 700Mb de RAM. Si no es así podremos obtener un error del tipo: g++: internal compiler error: Killed (program cc1plus).

Bajamos check_mk, decscomprimos e instalamos. Antes de ejecutar la instalación tenemos que asegurarnos de que el demonio de Nagios 4 está iniciado ya que el setup lo busca y configura adecuadamente ciertas opciones si lo encuentra.

```
tar zxfv check_mk-1.2.6p2.tar.gz
cd check_mk-1.2.6p2
./setup.sh
```

Las opciones que tuvimos que cambiar fueron las siguientes (el resto por defecto):

Nagios command pipe: /usr/local/nagios/var/rw/nagios.cmd
URL Prefix for Web addons: /site01/
Apache config dir: /etc/apache2/sites-enabled
Install Event Console: **yes**

Una vez que instala crea un fichero de respuestas en nuestro home ".check_mk_setup.conf". Dicho fichero lo usa para en la siguiente instalación / actualización tener las respuestas previas ya por defecto y sugeríendolas. De hecho si lo copias antes de instalar también te funcionará.

Activamos el demonio de eventos de CMK y reiniciamos apache, nagios y mkeventd

```
systemctl enable mkeventd
systemctl restart apache2 && systemctl restart nagios &&  systemctl restart mkeventd
```

Una vez finalice la instalación verificamos que la instalación añadió las siguientes líneas al final del fichero de configuración de nagios "nagios.cfg" (por defecto lo hace pero...)

```
# Load Livestatus Module
broker_module=/usr/lib/check_mk/livestatus.o /usr/local/nagios/var/rw/live
event_broker_options=-1
# added by setup.sh of check_mk
cfg_dir=/usr/local/nagios/etc/check_mk.d
```

Reiniciamos todo

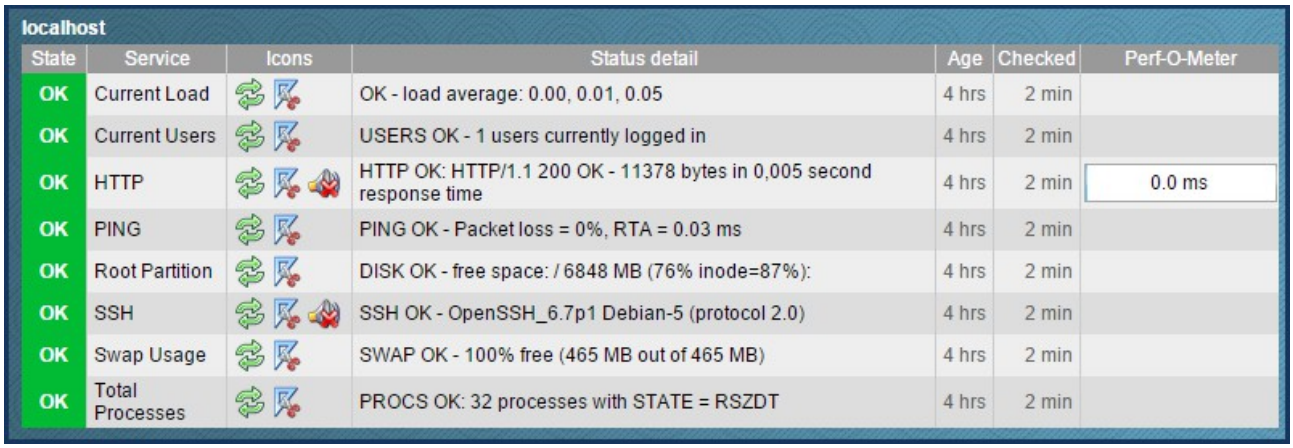
```
service apache2 restart $$ service npcd restart && service nagios restart
```

Echamos un ojo rápido al log de Nagios para ver si al reiniciar cargó correctamente el broker de check_mk livesatus como vemos:

```
> tail -f /usr/local/nagios/var/nagios.log
[1430464208] Event broker module '/usr/lib/check_mk/livestatus.o' initialized successfully.
```

Podemos acceder ya a check_mk: http://server/site01/check_mk/

Ejemplo de localhost en check_mk.



localhost

State	Service	Icons	Status detail	Age	Checked	Perf-O-Meter
OK	Current Load		OK - load average: 0.00, 0.01, 0.05	4 hrs	2 min	
OK	Current Users		USERS OK - 1 users currently logged in	4 hrs	2 min	
OK	HTTP		HTTP OK: HTTP/1.1 200 OK - 11378 bytes in 0,005 second response time	4 hrs	2 min	0.0 ms
OK	PING		PING OK - Packet loss = 0%, RTA = 0.03 ms	4 hrs	2 min	
OK	Root Partition		DISK OK - free space: / 6848 MB (76% inode=87%):	4 hrs	2 min	
OK	SSH		SSH OK - OpenSSH_6.7p1 Debian-5 (protocol 2.0)	4 hrs	2 min	
OK	Swap Usage		SWAP OK - 100% free (465 MB out of 465 MB)	4 hrs	2 min	
OK	Total Processes		PROCS OK: 32 processes with STATE = RSZDT	4 hrs	2 min	

Ojo que este host viene configurado en ficheros de Nagios no en “estilo” check_mk.

Instalación de cliente check_mk en Debian.

Para poder probar de formá rápida las bondades de check_mk instalaremos un cliente check_mk en nuestro propio server de monitorización. Necesitamos tener instalado xinetd, copiar el agente de cmk (script), generar un fichero de ejecución externa para xinetd y abrir el puerto 6556.

Primero necesitamos instalar xinetd. Realmente si instalamos luego el agente empaquetado debería instalar xinetd por dependencias.

```
apt-get install xinetd
```

Opción 1. Instalación mediante paquete RPM o DEB.

En el directorio /usr/share/check_mk/agents tenemos todos los agentes para numerosos S.O, además de el RPM para Redhat / Centos / Suse …. el DEB para Debian / Ubuntu... y el EXE importante para los Windows.

```
dpkg -i /usr/share/check_mk/agents/check-mk-agent_1.2.6p2-1_all.deb
```

Cambiamos en /etc/xinetd/check_mk

```
        only_from      = 127.0.0.1
```

Y reiniciamos xinetd

```
systemctl restart xinetd
```

Opción 2. Instalación manual.

Será más sencillo siempre instalar el paquete pero aunque no la usaremos habitualmente, la instalación manual nos vale para cualquier linux con python y xinetd. Es bueno saber lo que estamos haciendo con el RPM realmente que no deja de ser esto.

Luego crearemos el fichero /etc/xinetd.d/check_mk con el contenido:

```
service check_mk
{
        type          = UNLISTED
        port          = 6556
        socket_type    = stream
        protocol       = tcp
        wait          = no
        user          = root
        server        = /usr/bin/check_mk_agent

        # If you use fully redundant monitoring and poll the client
        # from more then one monitoring servers in parallel you might
        # want to use the agent cache wrapper:
        #server        = /usr/bin/check_mk_caching_agent
```

```
        # configure the IP address(es) of your Nagios server here:

    only_from      = 127.0.0.1

        # Don't be too verbose. Don't log every check. This might be
        # commented out for debugging. If this option is commented out
        # the default options will be used for this service.
        log_on_success =
        disable        = no
}
```

Y copiamos el script del directorio de agentes de cmk al sitio esperado y reiniciamos xinetd:

```
cp /usr/share/check_mk/agents/check_mk_agent.linux  /usr/bin/check_mk_agent
systemctl restart xinetd
```

El RPM además de hacer esto nos crea unos directorios vacios en /usr/lib/check_mk_agent para poder luego copiar plugins en estos.

Verificación de funcionamiento.

Vemos que nuestro puerto 6556 está a la escucha:

```
netstat -putaven | grep 6556
tcp        0      0 0.0.0.0:6556            0.0.0.0:*
```

Y que el agente responde en el puerto:

```
 telnet localhost 6556
```

Vamos a acceder al cliente a través de localhost pero si fueramos acceder desde CMK en otro servidor tendríamos que facilitar en el FW el acceso a dicho puerto.

Configuración en CMK WATO de nuestro host.

Accedemos a CMK y desde el ménu WATO Configuration / Hosts / New Host añadimos el nuestro:

- Hostname: srv-monitorizacion (ya tenemos un localhost)
- IP addres: 127.0.0.1

Save & go to Services (ya veremos los sevicios que va chequear)

Save manual check configuration

Salvamos los cambios: 2 Changes → Activate changes

Ya tenemos nuestro servidor en las vistas de hosts, servicios, ...

srv-monitorizacion

State	Service	Icons	Status detail	Age	Checked	Perf-O-Meter
OK	Check_MK		OK - Agent version 1.2.6p2, execution time 0.0 sec	2 min	50 sec	0.0 s
PEND	Check_MK Discovery			-	-	
OK	CPU load		OK - 15min load 0.05	2 min	50 sec	0.0
OK	CPU utilization		OK - user: 0.8%, system: 0.5%, wait: 0.3%, total: 1.6%	2 min	50 sec	1%
OK	Disk IO SUMMARY		OK - 2.71 kB/sec read, 133.36 kB/sec write, IOs: 5.27/sec	109 sec	50 sec	0.00 M/s 0.13 M/s
OK	Filesystem /		OK - 27.8% used (2.58 of 9.27 GB), (levels at 80.00/90.00%), trend: +12.77 MB / 24 hours, inodes available 547k/87.45%	2 min	50 sec	27.82 %
OK	Interface 2		OK - [eth0] (up) speed unknown, in: 97.85 B/s, out: 101.16 B/s	2 min	50 sec	97.9B/s 101.2B/s
OK	Kernel Context Switches		OK - 115/s	109 sec	50 sec	115.4/s
OK	Kernel Major Page Faults		OK - 0/s	109 sec	50 sec	0.1/s
OK	Kernel Process Creations		OK - 2/s	109 sec	50 sec	1.8/s
OK	Memory used		OK - 0.34 GB used (0.33 RAM + 0.00 SWAP + 0.01 Pagetables, this is 35.0% of 0.96 RAM (0.46 total SWAP)), 0.0 mapped, 0.5 committed, 0.0 shared	2 min	50 sec	34%
OK	Mount options of /		OK - mount options exactly as expected	2 min	50 sec	
OK	Number of threads		OK - 110 threads	2 min	50 sec	110
OK	TCP Connections		OK - ESTABLISHED: 5, TIME_WAIT: 2	2 min	50 sec	
OK	Uptime		OK - up since Fri May 1 18:15:14 2015 (0d 02:19:19)	2 min	50 sec	00d 02h 19m

El icono de la estrella naranja nos enlaza directamente con las gráficas de PNP4Nagios para ese servicio.

Nagvis.

Requisitos previos.

Instalamos los paquetes necesarios. Algunos ya los tendremos instalados.

```
apt-get install rsync php5-common libapache2-mod-php5 php5-cli php-gettext php5-cgi graphviz
sqlite sqlite3 php5-sqlite libjson-xs-perl
```

Instalación.

```
tar zxfv nagvis-1.8.2.tar.gz
cd nagvis-1.8b3
./install.sh -W /site01/nagvis -w /etc/apache2/sites-enabled
```

Vemos durante la instalación que descubre y valida los módulos de software que necesita y nos preguntará una serie de rutas y valores que podemos dejar por defecto.

El instalador nos dejará el fichero de configuración de apache en "/etc/apache2/conf-available/nagvis.conf " (incluso aunque intentemos pesarle el directorio adecuado con -w).

Primero debemos editarlo para hacer algunos cambios para nuestra versión de Apache ya que si no lo hacemos no nos funcionará el site (nos dará un error de permisos)

```
<Directory "/usr/local/nagvis/share">
 Require all granted
 Options FollowSymLinks
 AllowOverride None
 #Order allow,deny
 #Allow from all
```

Crearemos también un enlace directo desde sites-enabled donde están el resto de nuestros ficheros de configuración de Apache.

```
ln -s /etc/apache2/conf-available/nagvis.conf /etc/apache2/sites-enabled/nagvis.conf
systemctl restart apache2.service
```

Y accedemos a nagvis: http://server/site01/nagvis (admin / admin por defecto)

Nos encontraremos con un montón de ejemplos de mapas para poder ver las posibles opciones de estos. Una vez hayamos jugado con estos podremos borrarlos por medio de los menús.

Prueba rápida.

- Menu Options / Manage Backends / Backend por defecto → live_1 (Guardar)
- Menu Options / Mange Maps / Create Map → Pepito (Regular map) (Guardar)

- Con pepito seleccionado, Edit Map / Add Icon / Servicio. Soltamos la cruz en el mapa y al seleccionar en “Host_name” listará nuestros hosts en “service_description” los servicios de este. Como de momento solo tenemos “localhost” y nuestro servidor (con el cliente de check_mk) no da para mucho :-)
- Desativamos el modo edición del mapa y nuestro icono deberá reflejar fielmente el estado del servicio (usando MK Livestatus)