



Universidad
Politécnica
Internacional

Universidad Politécnica Internacional
Organización de Archivos
Julio, 2025

**Proyecto de Investigación y PoC de Nagios
San José, Costa Rica**

Benavides Robles Daniel
Céspedes Alfaro Ariana
López Pastora Kenia
Pérez Bejarano Sergio

Escuela de Ingeniería Informática, Universidad Politécnica Internacional

90503: Organización de Archivos

Segundo Cuatrimestre de 2025

Tema: Nagios para monitoreo de servicios

¿Qué es nagios?

Nagios es un software de código abierto diseñado para la monitorización continua de sistemas, redes e infraestructuras de TI. Su objetivo principal es supervisar hosts (equipos, servidores, dispositivos de red) y servicios (HTTP, SMTP, uso de CPU, espacio en disco, entre otros), generando alertas proactivas cuando se detectan anomalías o fallos.

Funcionalidades y características

- Monitorización en tiempo real: comprueba periódicamente el estado de hosts y servicios.
- Alertas personalizables: notificaciones por correo electrónico, SMS u otros canales cuando se superan umbrales predefinidos.
- Escalabilidad: gestiona desde unas pocas decenas hasta miles de dispositivos sin perder rendimiento.
- Personalización: reglas, umbrales y acciones pueden adaptarse a requerimientos específicos.
- Informes y gráficos: análisis de tendencias de uso de recursos y capacidad de planificación proactiva.

A continuación se presentan los elementos que puede monitorear Nagios:

- **Servidores:** uso de CPU, memoria, espacio en disco, procesos críticos en ejecución.
- **Redes:** routers, switches y firewalls; disponibilidad, tráfico y errores en interfaces.
- **Aplicaciones y servicios:** servidores web (HTTP), correo (SMTP/IMAP), bases de datos (MySQL, PostgreSQL), colas de mensajería, etc.
- **Hardware diverso:** impresoras, UPS, sistemas de almacenamiento (NAS/SAN), sensores de temperatura/humedad.
- **Procesos y transacciones:** monitorización de flujos de negocio y servicios web críticos mediante plugins especializados³.

Beneficios de usar Nagios

- Detección y resolución proactiva de incidentes antes de que impacten a usuarios finales.
- Reducción de tiempos de inactividad y mejora en el cumplimiento de acuerdos de nivel de servicio (SLA).
- Planificación de capacidad facilitada por datos históricos y gráficos de tendencias.
- Gran comunidad y ecosistema de más de 1 000 plugins que amplían su funcionalidad.

- Versión empresarial (Nagios XI) con asistentes, paneles avanzados y soporte oficial para medianas y grandes organizaciones.

Problema a resolver

El problema a resolver y por consiguiente a desarrollar para este trabajo es implementar un monitoreo básico de servicios, puertos o recursos del sistema para mejor visibilidad de los recursos de los servidores.

Pasos de implementación y requisitos

1. Requisitos Previos

Antes de comenzar, se debe asegurar que se tienen instalados:

- **Docker:** Para crear y gestionar los contenedores.
- **Docker Compose:** Para definir y ejecutar aplicaciones Docker multi-contenedor. Esto es crucial para la automatización.
- **Git:** Para clonar repositorios si se utilizan ejemplos o plantillas existentes.

2. Estructura de Directorios y Archivos Esenciales

Una organización clara de los archivos es vital, especialmente al trabajar con volúmenes de Docker en Windows.

- **Creación de la Estructura de Directorios:**
 - **Función:** Aloja todos los archivos de configuración y el **docker-compose.yml**.
 - **Dónde se corre/configura:** Se abre la terminal de Windows (PowerShell o CMD) y se ejecutan los siguientes comandos para crear la carpeta principal y la subcarpeta de configuraciones de Nagios:

```
mkdir P:\nagios_docker
cd P:\nagios_docker
mkdir nagios_cfg
```

Esto crea la estructura **C:\nagios_docker\nagios_cfg**, donde **nagios_cfg** contendrá los archivos de configuración específicos de Nagios.

- **Archivo docker-compose.yml:**

- **Función:** Este archivo es la pieza central para la orquestación. Define los servicios (contenedores) que se van a ejecutar, sus imágenes, puertos, volúmenes para persistencia de datos y dependencias.
- **Dónde se configura:** Se crea un archivo llamado **docker-compose.yml** directamente dentro de la carpeta **P:\nagios_docker**. Se recomienda usar Notepad++
- **Comandos:**

```
services:
  nagios:
    image: jasonrivers/nagios:latest # Se utiliza una imagen comunitaria estable de Nagios Core
    container_name: nagios_server
    hostname: nagios_host # Nombre de host dentro de la red Docker
    ports:
      - "8080:80" # Se mapea el puerto 80 del contenedor al puerto 8080 de la máquina Windows
    volumes:
      - ./nagios_cfg:/opt/nagios/etc/servers # Se monta la carpeta 'nagios_cfg' para las configuraciones personalizadas
      - ./nagios_cfg/nagios.cfg:/opt/nagios/etc/nagios.cfg # Monta del archivo de configuración principal de Nagios
      - nagios_data:/opt/nagios/var # Volumen persistente para logs, historial y datos de Nagios
    environment:
      # Variables de entorno para la configuración inicial del usuario administrador de Nagios
      - NAGIOSADMIN_USER=nagiosadmin
      - NAGIOSADMIN_PASS=Password123 # ¡Cambiar por una contraseña fuerte!
    networks:
      - nagios_network # Se asigna el contenedor a una red Docker definida

apachel:
  image: httpd:latest # Se utiliza la imagen oficial de Apache
  container_name: apachel
  hostname: apachel_host # Nombre de host que Nagios usará para comunicarse
```

```

ports:
  - "8081:80" # Se expone en un puerto diferente para evitar conflictos
networks:
  - nagios_network

apache2:
  image: httpd:latest
  container_name: apache2
  hostname: apache2_host
  ports:
    - "8082:80"
  networks:
    - nagios_network

volumes:
  nagios_data: # Se define un volumen con nombre para la persistencia de datos de Nagios

networks:
  nagios_network:
    driver: bridge # Se define una red tipo bridge para la comunicación entre los contenedores
    attachable: true
    external: false

```

- **Archivos de Configuración de Nagios (.cfg):**

- **Función:** Estos archivos definen qué se monitorea (hosts), cómo se monitorea (comandos) y los servicios específicos en cada host.
- **Dónde se configura:** Se crean estos archivos dentro de la carpeta P:\nagios_docker\nagios_cfg.
- **hosts.cfg (dentro de P:\nagios_docker\nagios_cfg):** Define los dispositivos o servidores a monitorear:

```

# Se define una plantilla para el template de apache server
define host{
    name                apache-server-template    ; Nombre de la plantilla
    use                 generic-host             ; Se basa en la plantilla genérica de
Nagios
    check_period        24x7                     ; Período de verificación
    check_interval      5                       ; Intervalo de verificación (minutos)
    retry_interval      1                       ; Intervalo de reintento (minutos)
    max_check_attempts  5                       ; Número máximo de intentos antes de
cambiar de estado
    check_command       check-host-alive        ; Comando para verificar si el host
está vivo
    notification_period 24x7                     ; Período para enviar notificaciones
    notification_interval 30                     ; Intervalo entre notificaciones
    notifications_enabled 1                     ; Notificaciones habilitadas
    contact_groups      admins                   ; Grupo de contactos a notificar
    register            0                       ; Esto es una plantilla, no un host real
}

# Se define el host para el servidor Apache 1
define host{
    use                 apache-server-template
    host_name           apache1_host
    alias               Apache Server 1
    address             apache1_host
    contact_groups      admins
}

# Se define el host para el servidor Apache 2
define host{
    use                 apache-server-template
    host_name           apache2_host
    alias               Apache Server 2
    address             apache2_host
    contact_groups      admins
}

```

- **services.cfg (dentro de P:\nagios_docker\nagios_cfg):** Define los servicios específicos a verificar en los hosts.

```
# Se define una plantilla para el template de apache server
define service{
    name                                apache-server-template          ; Nombre de la
plantilla
    active_checks_enabled                1                            ; Verificaciones activas
habilitadas
    passive_checks_enabled               1                            ; Verificaciones pasivas
habilitadas
    parallelize_check                    1                            ; Paralelizar
verificaciones
    obsess_over_service                 1                            ; Obsesión sobre el
servicio (si Nagios es el único que lo verifica)
    check_freshness                      0                            ; No se verifica la
frescura de los datos
    notifications_enabled                1                            ; Notificaciones
habilitadas
    event_handler_enabled                1                            ; Manejador de eventos
habilitado
    flap_detection_enabled                1                            ; Detección de
'flapping' habilitada
    failure_prediction_enabled           1                            ; Predicción de fallos
habilitada
    process_perf_data                    1                            ; Procesar datos de
rendimiento
    retain_status_information             1                            ; Retener información de
estado
    retain_nonstatus_information          1                            ; Retener información no
de estado
    is_volatile                          0                            ; No es volátil (su
estado no cambia aleatoriamente)
    check_period                          24x7                        ; Período de
verificación
    max_check_attempts                   3                            ; Número máximo de
```

```

intentos antes de cambiar de estado
    normal_check_interval          5                ; Intervalo de
verificación normal (minutos)
    retry_interval                  1                ; Intervalo de reintento
(minutos)
    notification_period             24x7             ; Período para enviar
notificaciones
    notification_interval           60               ; Intervalo entre
notificaciones
    first_notification_delay        0                ; Retraso para la
primera notificación
    #contact_groups                 admins            ; Grupo de contactos (se
hereda del host o se define aquí)
    register                        0                ; Esto es una plantilla,
no un servicio real
}

# Servicios para el Servidor Apache 1
define service{
    use                             apache-server-template
    host_name                       apache1_host
    service_description             Ping Check
    check_command                   check_ping!100.0,20%!500.0,60%
    notifications_enabled           1
    contact_groups                  admins
}

define service{
    use                             apache-server-template
    host_name                       apache1_host
    service_description             HTTP Web Service
    check_command                   check_http!-H apache1_host
    notifications_enabled           1
    contact_groups                  admins
}

# Servicios para el Servidor Apache 2

```



```

define service{
    use                apache-server-template
    host_name          apache2_host
    service_description Ping Check
    check_command       check_ping!100.0,20%!500.0,60%
    notifications_enabled 1
    contact_groups      admins
}

define service{
    use                apache-server-template
    host_name          apache2_host
    service_description HTTP Web Service
    check_command       check_http!-H apache2_host
    notifications_enabled 1
    contact_groups      admins
}

```

- Se crea el archivo **nagios.cfg** (dentro de **P:\nagios_docker\nagios_cfg**): Para definir las configuraciones de hosts y services personalizadas, ingresamos las siguientes líneas para este caso:

```

#####
#
# NAGIOS.CFG - Sample Main Config File for Nagios 4.1.1
#
# Read the documentation for more information on this configuration
# file.  I've provided some comments here, but things may not be so
# clear without further explanation.
#
#
#####

```

```
# LOG FILE
# This is the main log file where service and host events are logged
# for historical purposes. This should be the first option specified
# in the config file!!!

log_file=/opt/nagios/var/nagios.log


# OBJECT CONFIGURATION FILE(S)
# These are the object configuration files in which you define hosts,
# host groups, contacts, contact groups, services, etc.
# You can split your object definitions across several config files
# if you wish (as shown below), or keep them all in a single config file.

# You can specify individual object config files as shown below:
cfg_file=/opt/nagios/etc/objects/commands.cfg
cfg_file=/opt/nagios/etc/objects/contacts.cfg
cfg_file=/opt/nagios/etc/objects/timeperiods.cfg
cfg_file=/opt/nagios/etc/objects/templates.cfg


# Definitions for monitoring the local (Linux) host
cfg_file=/opt/nagios/etc/objects/localhost.cfg


cfg_file=/opt/nagios/etc/servers/hosts.cfg
cfg_file=/opt/nagios/etc/servers/services.cfg


# Definitions for monitoring a Windows machine
#cfg_file=/opt/nagios/etc/objects/windows.cfg


# Definitions for monitoring a router/switch
#cfg_file=/opt/nagios/etc/objects/switch.cfg


# Definitions for monitoring a network printer
#cfg_file=/opt/nagios/etc/objects/printer.cfg
```

```
# You can also tell Nagios to process all config files (with a .cfg
# extension) in a particular directory by using the cfg_dir
# directive as shown below:
```

```
#cfg_dir=/opt/nagios/etc/servers
#cfg_dir=/opt/nagios/etc/printers
#cfg_dir=/opt/nagios/etc/switches
#cfg_dir=/opt/nagios/etc/routers
```

OBJECT CACHE FILE

```
# This option determines where object definitions are cached when
# Nagios starts/restarts. The CGIs read object definitions from
# this cache file (rather than looking at the object config files
# directly) in order to prevent inconsistencies that can occur
# when the config files are modified after Nagios starts.
```

```
object_cache_file=/opt/nagios/var/objects.cache
```

PRE-CACHED OBJECT FILE

```
# This options determines the location of the precached object file.
# If you run Nagios with the -p command line option, it will preprocess
# your object configuration file(s) and write the cached config to this
# file. You can then start Nagios with the -u option to have it read
# object definitions from this precached file, rather than the standard
# object configuration files (see the cfg_file and cfg_dir options above).
# Using a precached object file can speed up the time needed to (re)start
# the Nagios process if you've got a large and/or complex configuration.
# Read the documentation section on optimizing Nagios to find our more
# about how this feature works.
```

```
precached_object_file=/opt/nagios/var/objects.precache
```

```
# RESOURCE FILE
# This is an optional resource file that contains $USERx$ macro
# definitions. Multiple resource files can be specified by using
# multiple resource_file definitions. The CGIs will not attempt to
# read the contents of resource files, so information that is
# considered to be sensitive (usernames, passwords, etc) can be
# defined as macros in this file and restrictive permissions (600)
# can be placed on this file.
```

```
resource_file=/opt/nagios/etc/resource.cfg
```

```
# STATUS FILE
# This is where the current status of all monitored services and
# hosts is stored. Its contents are read and processed by the CGIs.
# The contents of the status file are deleted every time Nagios
# restarts.
```

```
status_file=/opt/nagios/var/status.dat
```

```
# STATUS FILE UPDATE INTERVAL
# This option determines the frequency (in seconds) that
# Nagios will periodically dump program, host, and
# service status data.
```

```
status_update_interval=10
```

```
# NAGIOS USER
# This determines the effective user that Nagios should run as.
```

```
# You can either supply a username or a UID.

nagios_user=nagios


# NAGIOS GROUP
# This determines the effective group that Nagios should run as.
# You can either supply a group name or a GID.

nagios_group=nagios


# EXTERNAL COMMAND OPTION
# This option allows you to specify whether or not Nagios should check
# for external commands (in the command file defined below). By default
# Nagios will *not* check for external commands, just to be on the
# cautious side. If you want to be able to use the CGI command interface
# you will have to enable this.
# Values: 0 = disable commands, 1 = enable commands

check_external_commands=1


# EXTERNAL COMMAND FILE
# This is the file that Nagios checks for external command requests.
# It is also where the command CGI will write commands that are submitted
# by users, so it must be writeable by the user that the web server
# is running as (usually 'nobody'). Permissions should be set at the
# directory level instead of on the file, as the file is deleted every
# time its contents are processed.

command_file=/opt/nagios/var/rw/nagios.cmd
```

```
# QUERY HANDLER INTERFACE
# This is the socket that is created for the Query Handler interface

#query_socket=/opt/nagios/var/rw/nagios.qh


# LOCK FILE
# This is the lockfile that Nagios will use to store its PID number
# in when it is running in daemon mode.

lock_file=/opt/nagios/var/nagios.lock


# TEMP FILE
# This is a temporary file that is used as scratch space when Nagios
# updates the status log, cleans the comment file, etc. This file
# is created, used, and deleted throughout the time that Nagios is
# running.

temp_file=/opt/nagios/var/nagios.tmp


# TEMP PATH
# This is path where Nagios can create temp files for service and
# host check results, etc.

temp_path=/tmp


# EVENT BROKER OPTIONS
# Controls what (if any) data gets sent to the event broker.
# Values:  0          = Broker nothing
```

```

#          -1          = Broker everything
#          <other>     = See documentation

event_broker_options=-1


# EVENT BROKER MODULE(S)
# This directive is used to specify an event broker module that should
# be loaded by Nagios at startup. Use multiple directives if you want
# to load more than one module. Arguments that should be passed to
# the module at startup are separated from the module path by a space.
#
#!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
# WARNING !!! WARNING !!! WARNING !!! WARNING !!! WARNING !!! WARNING
#!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
#
# Do NOT overwrite modules while they are being used by Nagios or Nagios
# will crash in a fiery display of SEGFAULT glory. This is a bug/limitation
# either in dlopen(), the kernel, and/or the filesystem. And maybe Nagios...
#
# The correct/safe way of updating a module is by using one of these methods:
#   1. Shutdown Nagios, replace the module file, restart Nagios
#   2. Delete the original module file, move the new module file into place,
#   restart Nagios
#
# Example:
#
#   broker_module=<modulepath> [moduleargs]

#broker_module=/somewhere/module1.o
#broker_module=/somewhere/module2.o arg1 arg2=3 debug=0


# LOG ROTATION METHOD
# This is the log rotation method that Nagios should use to rotate

```

```
# the main log file. Values are as follows..
#   n   = None - don't rotate the log
#   h   = Hourly rotation (top of the hour)
#   d   = Daily rotation (midnight every day)
#   w   = Weekly rotation (midnight on Saturday evening)
#   m   = Monthly rotation (midnight last day of month)

log_rotation_method=d

# LOG ARCHIVE PATH
# This is the directory where archived (rotated) log files should be
# placed (assuming you've chosen to do log rotation).

log_archive_path=/opt/nagios/var/archives

# LOGGING OPTIONS
# If you want messages logged to the syslog facility, as well as the
# Nagios log file set this option to 1.  If not, set it to 0.

use_syslog=1

# NOTIFICATION LOGGING OPTION
# If you don't want notifications to be logged, set this value to 0.
# If notifications should be logged, set the value to 1.

log_notifications=1

# SERVICE RETRY LOGGING OPTION
# If you don't want service check retries to be logged, set this value
```



```
# to 0. If retries should be logged, set the value to 1.
```

```
log_service_retries=1
```

```
# HOST RETRY LOGGING OPTION
```

```
# If you don't want host check retries to be logged, set this value to  
# 0. If retries should be logged, set the value to 1.
```

```
log_host_retries=1
```

```
# EVENT HANDLER LOGGING OPTION
```

```
# If you don't want host and service event handlers to be logged, set  
# this value to 0. If event handlers should be logged, set the value  
# to 1.
```

```
log_event_handlers=1
```

```
# INITIAL STATES LOGGING OPTION
```

```
# If you want Nagios to log all initial host and service states to  
# the main log file (the first time the service or host is checked)  
# you can enable this option by setting this value to 1. If you  
# are not using an external application that does long term state  
# statistics reporting, you do not need to enable this option. In  
# this case, set the value to 0.
```

```
log_initial_states=0
```

```
# CURRENT STATES LOGGING OPTION
```

```
# If you don't want Nagios to log all current host and service states
```

```
# after log has been rotated to the main log file, you can disable this
# option by setting this value to 0. Default value is 1.
```

```
log_current_states=1
```

```
# EXTERNAL COMMANDS LOGGING OPTION
```

```
# If you don't want Nagios to log external commands, set this value
# to 0. If external commands should be logged, set this value to 1.
# Note: This option does not include logging of passive service
# checks - see the option below for controlling whether or not
# passive checks are logged.
```

```
log_external_commands=1
```

```
# PASSIVE CHECKS LOGGING OPTION
```

```
# If you don't want Nagios to log passive host and service checks, set
# this value to 0. If passive checks should be logged, set
# this value to 1.
```

```
log_passive_checks=1
```

```
# GLOBAL HOST AND SERVICE EVENT HANDLERS
```

```
# These options allow you to specify a host and service event handler
# command that is to be run for every host or service state change.
# The global event handler is executed immediately prior to the event
# handler that you have optionally specified in each host or
# service definition. The command argument is the short name of a
# command definition that you define in your host configuration file.
# Read the HTML docs for more information.
```

```
#global_host_event_handler=somecommand
```

```
#global_service_event_handler=somecommand
```

```
# SERVICE INTER-CHECK DELAY METHOD
```

```
# This is the method that Nagios should use when initially  
# "spreading out" service checks when it starts monitoring. The  
# default is to use smart delay calculation, which will try to  
# space all service checks out evenly to minimize CPU load.  
# Using the dumb setting will cause all checks to be scheduled  
# at the same time (with no delay between them)! This is not a  
# good thing for production, but is useful when testing the  
# parallelization functionality.
```

```
# n = None - don't use any delay between checks
```

```
# d = Use a "dumb" delay of 1 second between checks
```

```
# s = Use "smart" inter-check delay calculation
```

```
# x.xx = Use an inter-check delay of x.xx seconds
```

```
service_inter_check_delay_method=s
```

```
# MAXIMUM SERVICE CHECK SPREAD
```

```
# This variable determines the timeframe (in minutes) from the  
# program start time that an initial check of all services should  
# be completed. Default is 30 minutes.
```

```
max_service_check_spread=30
```

```
# SERVICE CHECK INTERLEAVE FACTOR
```

```
# This variable determines how service checks are interleaved.  
# Interleaving the service checks allows for a more even  
# distribution of service checks and reduced load on remote  
# hosts. Setting this value to 1 is equivalent to how versions  
# of Nagios previous to 0.0.5 did service checks. Set this
```

```
# value to s (smart) for automatic calculation of the interleave
# factor unless you have a specific reason to change it.
#      s      = Use "smart" interleave factor calculation
#      x      = Use an interleave factor of x, where x is a
#                number greater than or equal to 1.
```

```
service_interleave_factor=s
```

```
# HOST INTER-CHECK DELAY METHOD
# This is the method that Nagios should use when initially
# "spreading out" host checks when it starts monitoring. The
# default is to use smart delay calculation, which will try to
# space all host checks out evenly to minimize CPU load.
# Using the dumb setting will cause all checks to be scheduled
# at the same time (with no delay between them)!
#      n      = None - don't use any delay between checks
#      d      = Use a "dumb" delay of 1 second between checks
#      s      = Use "smart" inter-check delay calculation
#      x.xx   = Use an inter-check delay of x.xx seconds
```

```
host_inter_check_delay_method=s
```

```
# MAXIMUM HOST CHECK SPREAD
# This variable determines the timeframe (in minutes) from the
# program start time that an initial check of all hosts should
# be completed. Default is 30 minutes.
```

```
max_host_check_spread=30
```

```
# MAXIMUM CONCURRENT SERVICE CHECKS
# This option allows you to specify the maximum number of
```

```
# service checks that can be run in parallel at any given time.
# Specifying a value of 1 for this variable essentially prevents
# any service checks from being parallelized. A value of 0
# will not restrict the number of concurrent checks that are
# being executed.
```

```
max_concurrent_checks=0
```

```
# HOST AND SERVICE CHECK REAPER FREQUENCY
# This is the frequency (in seconds!) that Nagios will process
# the results of host and service checks.
```

```
check_result_reaper_frequency=10
```

```
# MAX CHECK RESULT REAPER TIME
# This is the max amount of time (in seconds) that a single
# check result reaper event will be allowed to run before
# returning control back to Nagios so it can perform other
# duties.
```

```
max_check_result_reaper_time=30
```

```
# CHECK RESULT PATH
# This is directory where Nagios stores the results of host and
# service checks that have not yet been processed.
#
# Note: Make sure that only one instance of Nagios has access
# to this directory!
```

```
check_result_path=/opt/nagios/var/spool/checkresults
```

```
# MAX CHECK RESULT FILE AGE
```

```
# This option determines the maximum age (in seconds) which check  
# result files are considered to be valid. Files older than this  
# threshold will be mercilessly deleted without further processing.
```

```
max_check_result_file_age=3600
```

```
# CACHED HOST CHECK HORIZON
```

```
# This option determines the maximum amount of time (in seconds)  
# that the state of a previous host check is considered current.  
# Cached host states (from host checks that were performed more  
# recently than the timeframe specified by this value) can immensely  
# improve performance in regards to the host check logic.  
# Too high of a value for this option may result in inaccurate host  
# states being used by Nagios, while a lower value may result in a  
# performance hit for host checks. Use a value of 0 to disable host  
# check caching.
```

```
cached_host_check_horizon=15
```

```
# CACHED SERVICE CHECK HORIZON
```

```
# This option determines the maximum amount of time (in seconds)  
# that the state of a previous service check is considered current.  
# Cached service states (from service checks that were performed more  
# recently than the timeframe specified by this value) can immensely  
# improve performance in regards to predictive dependency checks.  
# Use a value of 0 to disable service check caching.
```

```
cached_service_check_horizon=15
```

```
# ENABLE PREDICTIVE HOST DEPENDENCY CHECKS
```

```
# This option determines whether or not Nagios will attempt to execute  
# checks of hosts when it predicts that future dependency logic test  
# may be needed. These predictive checks can help ensure that your  
# host dependency logic works well.
```

```
# Values:
```

```
# 0 = Disable predictive checks
```

```
# 1 = Enable predictive checks (default)
```

```
enable_predictive_host_dependency_checks=1
```

```
# ENABLE PREDICTIVE SERVICE DEPENDENCY CHECKS
```

```
# This option determines whether or not Nagios will attempt to execute  
# checks of service when it predicts that future dependency logic test  
# may be needed. These predictive checks can help ensure that your  
# service dependency logic works well.
```

```
# Values:
```

```
# 0 = Disable predictive checks
```

```
# 1 = Enable predictive checks (default)
```

```
enable_predictive_service_dependency_checks=1
```

```
# SOFT STATE DEPENDENCIES
```

```
# This option determines whether or not Nagios will use soft state  
# information when checking host and service dependencies. Normally  
# Nagios will only use the latest hard host or service state when  
# checking dependencies. If you want it to use the latest state (regardless  
# of whether its a soft or hard state type), enable this option.
```

```
# Values:
# 0 = Don't use soft state dependencies (default)
# 1 = Use soft state dependencies

soft_state_dependencies=0


# TIME CHANGE ADJUSTMENT THRESHOLDS
# These options determine when Nagios will react to detected changes
# in system time (either forward or backwards).

#time_change_threshold=900


# AUTO-RESCHEDULING OPTION
# This option determines whether or not Nagios will attempt to
# automatically reschedule active host and service checks to
# "smooth" them out over time. This can help balance the load on
# the monitoring server.
# WARNING: THIS IS AN EXPERIMENTAL FEATURE - IT CAN DEGRADE
# PERFORMANCE, RATHER THAN INCREASE IT, IF USED IMPROPERLY

auto_reschedule_checks=0


# AUTO-RESCHEDULING INTERVAL
# This option determines how often (in seconds) Nagios will
# attempt to automatically reschedule checks. This option only
# has an effect if the auto_reschedule_checks option is enabled.
# Default is 30 seconds.
# WARNING: THIS IS AN EXPERIMENTAL FEATURE - IT CAN DEGRADE
# PERFORMANCE, RATHER THAN INCREASE IT, IF USED IMPROPERLY

auto_rescheduling_interval=30
```



```
# AUTO-RESCHEDULING WINDOW
# This option determines the "window" of time (in seconds) that
# Nagios will look at when automatically rescheduling checks.
# Only host and service checks that occur in the next X seconds
# (determined by this variable) will be rescheduled. This option
# only has an effect if the auto_reschedule_checks option is
# enabled. Default is 180 seconds (3 minutes).
# WARNING: THIS IS AN EXPERIMENTAL FEATURE - IT CAN DEGRADE
# PERFORMANCE, RATHER THAN INCREASE IT, IF USED IMPROPERLY
```

```
auto_rescheduling_window=180
```

```
# TIMEOUT VALUES
# These options control how much time Nagios will allow various
# types of commands to execute before killing them off. Options
# are available for controlling maximum time allotted for
# service checks, host checks, event handlers, notifications, the
# oosp command, and performance data commands. All values are in
# seconds.
```

```
service_check_timeout=60
host_check_timeout=30
event_handler_timeout=30
notification_timeout=30
oosp_timeout=5
perfdata_timeout=5
```

```
# RETAIN STATE INFORMATION
# This setting determines whether or not Nagios will save state
# information for services and hosts before it shuts down. Upon
```

```
# startup Nagios will reload all saved service and host state
# information before starting to monitor. This is useful for
# maintaining long-term data on state statistics, etc, but will
# slow Nagios down a bit when it (re)starts. Since its only
# a one-time penalty, I think its well worth the additional
# startup delay.
```

```
retain_state_information=1
```

```
# STATE RETENTION FILE
```

```
# This is the file that Nagios should use to store host and
# service state information before it shuts down. The state
# information in this file is also read immediately prior to
# starting to monitor the network when Nagios is restarted.
# This file is used only if the retain_state_information
# variable is set to 1.
```

```
state_retention_file=/opt/nagios/var/retention.dat
```

```
# RETENTION DATA UPDATE INTERVAL
```

```
# This setting determines how often (in minutes) that Nagios
# will automatically save retention data during normal operation.
# If you set this value to 0, Nagios will not save retention
# data at regular interval, but it will still save retention
# data before shutting down or restarting. If you have disabled
# state retention, this option has no effect.
```

```
retention_update_interval=60
```

```
# USE RETAINED PROGRAM STATE
```

```
# This setting determines whether or not Nagios will set
```

```
# program status variables based on the values saved in the
# retention file.  If you want to use retained program status
# information, set this value to 1.  If not, set this value
# to 0.
```

```
use_retained_program_state=1
```

```
# USE RETAINED SCHEDULING INFO
```

```
# This setting determines whether or not Nagios will retain
# the scheduling info (next check time) for hosts and services
# based on the values saved in the retention file.  If you
# If you want to use retained scheduling info, set this
# value to 1.  If not, set this value to 0.
```

```
use_retained_scheduling_info=1
```

```
# RETAINED ATTRIBUTE MASKS (ADVANCED FEATURE)
```

```
# The following variables are used to specify specific host and
# service attributes that should *not* be retained by Nagios during
# program restarts.
```

```
#
```

```
# The values of the masks are bitwise ANDs of values specified
# by the "MODATTR_" definitions found in include/common.h.
# For example, if you do not want the current enabled/disabled state
# of flap detection and event handlers for hosts to be retained, you
# would use a value of 24 for the host attribute mask...
```

```
# MODATTR_EVENT_HANDLER_ENABLED (8) + MODATTR_FLAP_DETECTION_ENABLED (16) = 24
```

```
# This mask determines what host attributes are not retained
retained_host_attribute_mask=0
```

```
# This mask determines what service attributes are not retained
retained_service_attribute_mask=0
```

```
# These two masks determine what process attributes are not retained.
# There are two masks, because some process attributes have host and service
# options. For example, you can disable active host checks, but leave active
# service checks enabled.
retained_process_host_attribute_mask=0
retained_process_service_attribute_mask=0
```

```
# These two masks determine what contact attributes are not retained.
# There are two masks, because some contact attributes have host and
# service options. For example, you can disable host notifications for
# a contact, but leave service notifications enabled for them.
retained_contact_host_attribute_mask=0
retained_contact_service_attribute_mask=0
```

```
# INTERVAL LENGTH
# This is the seconds per unit interval as used in the
# host/contact/service configuration files. Setting this to 60 means
# that each interval is one minute long (60 seconds). Other settings
# have not been tested much, so your mileage is likely to vary...
```

```
interval_length=60
```

```
# CHECK FOR UPDATES
# This option determines whether Nagios will automatically check to
# see if new updates (releases) are available. It is recommend that you
# enable this option to ensure that you stay on top of the latest critical
# patches to Nagios. Nagios is critical to you - make sure you keep it in
# good shape. Nagios will check once a day for new updates. Data collected
# by Nagios Enterprises from the update check is processed in accordance
# with our privacy policy - see https://api.nagios.org for details.
```

```
check_for_updates=1
```

```
# BARE UPDATE CHECK
# This option determines what data Nagios will send to api.nagios.org when
# it checks for updates. By default, Nagios will send information on the
# current version of Nagios you have installed, as well as an indicator as
# to whether this was a new installation or not. Nagios Enterprises uses
# this data to determine the number of users running specific version of
# Nagios. Enable this option if you do not want this information to be sent.
```

```
bare_update_check=0
```

```
# AGGRESSIVE HOST CHECKING OPTION
# If you don't want to turn on aggressive host checking features, set
# this value to 0 (the default). Otherwise set this value to 1 to
# enable the aggressive check option. Read the docs for more info
# on what aggressive host check is or check out the source code in
# base/checks.c
```

```
use_aggressive_host_checking=0
```

```
# SERVICE CHECK EXECUTION OPTION
# This determines whether or not Nagios will actively execute
# service checks when it initially starts. If this option is
# disabled, checks are not actively made, but Nagios can still
# receive and process passive check results that come in. Unless
# you're implementing redundant hosts or have a special need for
# disabling the execution of service checks, leave this enabled!
# Values: 1 = enable checks, 0 = disable checks
```

```
execute_service_checks=1
```

```
# PASSIVE SERVICE CHECK ACCEPTANCE OPTION
# This determines whether or not Nagios will accept passive
# service checks results when it initially (re)starts.
# Values: 1 = accept passive checks, 0 = reject passive checks
```

```
accept_passive_service_checks=1
```

```
# HOST CHECK EXECUTION OPTION
# This determines whether or not Nagios will actively execute
# host checks when it initially starts. If this option is
# disabled, checks are not actively made, but Nagios can still
# receive and process passive check results that come in. Unless
# you're implementing redundant hosts or have a special need for
# disabling the execution of host checks, leave this enabled!
# Values: 1 = enable checks, 0 = disable checks
```

```
execute_host_checks=1
```

```
# PASSIVE HOST CHECK ACCEPTANCE OPTION
# This determines whether or not Nagios will accept passive
# host checks results when it initially (re)starts.
# Values: 1 = accept passive checks, 0 = reject passive checks
```

```
accept_passive_host_checks=1
```

```
# NOTIFICATIONS OPTION
# This determines whether or not Nagios will sent out any host or
# service notifications when it is initially (re)started.
# Values: 1 = enable notifications, 0 = disable notifications
```

```
enable_notifications=1
```

```
# EVENT HANDLER USE OPTION
```

```
# This determines whether or not Nagios will run any host or  
# service event handlers when it is initially (re)started. Unless  
# you're implementing redundant hosts, leave this option enabled.  
# Values: 1 = enable event handlers, 0 = disable event handlers
```

```
enable_event_handlers=1
```

```
# PROCESS PERFORMANCE DATA OPTION
```

```
# This determines whether or not Nagios will process performance  
# data returned from service and host checks. If this option is  
# enabled, host performance data will be processed using the  
# host_perfdata_command (defined below) and service performance  
# data will be processed using the service_perfdata_command (also  
# defined below). Read the HTML docs for more information on  
# performance data.  
# Values: 1 = process performance data, 0 = do not process performance data
```

```
process_performance_data=1
```

```
# HOST AND SERVICE PERFORMANCE DATA PROCESSING COMMANDS
```

```
# These commands are run after every host and service check is  
# performed. These commands are executed only if the  
# enable_performance_data option (above) is set to 1. The command  
# argument is the short name of a command definition that you  
# define in your host configuration file. Read the HTML docs for  
# more information on performance data.
```

```
#host_perfdata_command=process-host-perfdata
#service_perfdata_command=process-service-perfdata


# HOST AND SERVICE PERFORMANCE DATA FILES
# These files are used to store host and service performance data.
# Performance data is only written to these files if the
# enable_performance_data option (above) is set to 1.


#host_perfdata_file=/opt/nagios/var/host-perfdata
service_perfdata_file=/opt/nagios/var/perfdata.log


# HOST AND SERVICE PERFORMANCE DATA FILE TEMPLATES
# These options determine what data is written (and how) to the
# performance data files. The templates may contain macros, special
# characters (\t for tab, \r for carriage return, \n for newline)
# and plain text. A newline is automatically added after each write
# to the performance data file. Some examples of what you can do are
# shown below.


host_perfdata_file_template=empty
service_perfdata_file_template=$LASTSERVICECHECK$||$HOSTNAME$||$SERVICEDESC$||$SERVICEOUTPUT$||$SERVICEPERFDATA$


# HOST AND SERVICE PERFORMANCE DATA FILE MODES
# This option determines whether or not the host and service
# performance data files are opened in write ("w") or append ("a")
# mode. If you want to use named pipes, you should use the special
# pipe ("p") mode which avoid blocking at startup, otherwise you will
# likely want the default append ("a") mode.


host_perfdata_file_mode=w
service_perfdata_file_mode=a
```



```
# HOST AND SERVICE PERFORMANCE DATA FILE PROCESSING INTERVAL
# These options determine how often (in seconds) the host and service
# performance data files are processed using the commands defined
# below. A value of 0 indicates the files should not be periodically
# processed.

#host_perfdata_file_processing_interval=0
service_perfdata_file_processing_interval=10


# HOST AND SERVICE PERFORMANCE DATA FILE PROCESSING COMMANDS
# These commands are used to periodically process the host and
# service performance data files. The interval at which the
# processing occurs is determined by the options above.

#host_perfdata_file_processing_command=process-host-perfdata-file
service_perfdata_file_processing_command=process-service-perfdata


# HOST AND SERVICE PERFORMANCE DATA PROCESS EMPTY RESULTS
# These options determine whether the core will process empty perfdata
# results or not. This is needed for distributed monitoring, and intentionally
# turned on by default.
# If you don't require empty perfdata - saving some cpu cycles
# on unwanted macro calculation - you can turn that off. Be careful!
# Values: 1 = enable, 0 = disable

#host_perfdata_process_empty_results=1
#service_perfdata_process_empty_results=1


# OBSESS OVER SERVICE CHECKS OPTION
# This determines whether or not Nagios will obsess over service
```

```
# checks and run the ocsp_command defined below. Unless you're
# planning on implementing distributed monitoring, do not enable
# this option. Read the HTML docs for more information on
# implementing distributed monitoring.
# Values: 1 = obsess over services, 0 = do not obsess (default)

obsess_over_services=0


# OBSESSIVE COMPULSIVE SERVICE PROCESSOR COMMAND
# This is the command that is run for every service check that is
# processed by Nagios. This command is executed only if the
# obsess_over_services option (above) is set to 1. The command
# argument is the short name of a command definition that you
# define in your host configuration file. Read the HTML docs for
# more information on implementing distributed monitoring.

#ocsp_command=somecommand


# OBSESS OVER HOST CHECKS OPTION
# This determines whether or not Nagios will obsess over host
# checks and run the ochp_command defined below. Unless you're
# planning on implementing distributed monitoring, do not enable
# this option. Read the HTML docs for more information on
# implementing distributed monitoring.
# Values: 1 = obsess over hosts, 0 = do not obsess (default)

obsess_over_hosts=0


# OBSESSIVE COMPULSIVE HOST PROCESSOR COMMAND
# This is the command that is run for every host check that is
# processed by Nagios. This command is executed only if the
```

```
# obsess_over_hosts option (above) is set to 1. The command
# argument is the short name of a command definition that you
# define in your host configuration file. Read the HTML docs for
# more information on implementing distributed monitoring.
```

```
#ochp_command=somecommand
```

```
# TRANSLATE PASSIVE HOST CHECKS OPTION
```

```
# This determines whether or not Nagios will translate
# DOWN/UNREACHABLE passive host check results into their proper
# state for this instance of Nagios. This option is useful
# if you have distributed or failover monitoring setup. In
# these cases your other Nagios servers probably have a different
# "view" of the network, with regards to the parent/child relationship
# of hosts. If a distributed monitoring server thinks a host
# is DOWN, it may actually be UNREACHABLE from the point of
# this Nagios instance. Enabling this option will tell Nagios
# to translate any DOWN or UNREACHABLE host states it receives
# passively into the correct state from the view of this server.
# Values: 1 = perform translation, 0 = do not translate (default)
```

```
translate_passive_host_checks=0
```

```
# PASSIVE HOST CHECKS ARE SOFT OPTION
```

```
# This determines whether or not Nagios will treat passive host
# checks as being HARD or SOFT. By default, a passive host check
# result will put a host into a HARD state type. This can be changed
# by enabling this option.
# Values: 0 = passive checks are HARD, 1 = passive checks are SOFT
```

```
passive_host_checks_are_soft=0
```

```
# ORPHANED HOST/SERVICE CHECK OPTIONS
```

```
# These options determine whether or not Nagios will periodically  
# check for orphaned host service checks. Since service checks are  
# not rescheduled until the results of their previous execution  
# instance are processed, there exists a possibility that some  
# checks may never get rescheduled. A similar situation exists for  
# host checks, although the exact scheduling details differ a bit  
# from service checks. Orphaned checks seem to be a rare  
# problem and should not happen under normal circumstances.  
# If you have problems with service checks never getting  
# rescheduled, make sure you have orphaned service checks enabled.  
# Values: 1 = enable checks, 0 = disable checks
```

```
check_for_orphaned_services=1
```

```
check_for_orphaned_hosts=1
```

```
# SERVICE FRESHNESS CHECK OPTION
```

```
# This option determines whether or not Nagios will periodically  
# check the "freshness" of service results. Enabling this option  
# is useful for ensuring passive checks are received in a timely  
# manner.  
# Values: 1 = enabled freshness checking, 0 = disable freshness checking
```

```
check_service_freshness=1
```

```
# SERVICE FRESHNESS CHECK INTERVAL
```

```
# This setting determines how often (in seconds) Nagios will  
# check the "freshness" of service check results. If you have  
# disabled service freshness checking, this option has no effect.
```

```
service_freshness_check_interval=60
```

```
# SERVICE CHECK TIMEOUT STATE
# This setting determines the state Nagios will report when a
# service check times out - that is does not respond within
# service_check_timeout seconds. This can be useful if a
# machine is running at too high a load and you do not want
# to consider a failed service check to be critical (the default).
# Valid settings are:
# c - Critical (default)
# u - Unknown
# w - Warning
# o - OK

service_check_timeout_state=c


# HOST FRESHNESS CHECK OPTION
# This option determines whether or not Nagios will periodically
# check the "freshness" of host results. Enabling this option
# is useful for ensuring passive checks are received in a timely
# manner.
# Values: 1 = enabled freshness checking, 0 = disable freshness checking

check_host_freshness=0


# HOST FRESHNESS CHECK INTERVAL
# This setting determines how often (in seconds) Nagios will
# check the "freshness" of host check results. If you have
# disabled host freshness checking, this option has no effect.

host_freshness_check_interval=60
```

```
# ADDITIONAL FRESHNESS THRESHOLD LATENCY
# This setting determines the number of seconds that Nagios
# will add to any host and service freshness thresholds that
# it calculates (those not explicitly specified by the user).
```

```
additional_freshness_latency=15
```

```
# FLAP DETECTION OPTION
# This option determines whether or not Nagios will try
# and detect hosts and services that are "flapping".
# Flapping occurs when a host or service changes between
# states too frequently. When Nagios detects that a
# host or service is flapping, it will temporarily suppress
# notifications for that host/service until it stops
# flapping. Flap detection is very experimental, so read
# the HTML documentation before enabling this feature!
# Values: 1 = enable flap detection
#         0 = disable flap detection (default)
```

```
enable_flap_detection=1
```

```
# FLAP DETECTION THRESHOLDS FOR HOSTS AND SERVICES
# Read the HTML documentation on flap detection for
# an explanation of what this option does. This option
# has no effect if flap detection is disabled.
```

```
low_service_flap_threshold=5.0
high_service_flap_threshold=20.0
low_host_flap_threshold=5.0
high_host_flap_threshold=20.0
```

```
# DATE FORMAT OPTION
# This option determines how short dates are displayed. Valid options
# include:
#   us      (MM-DD-YYYY HH:MM:SS)
#   euro    (DD-MM-YYYY HH:MM:SS)
#   iso8601 (YYYY-MM-DD HH:MM:SS)
#   strict-iso8601 (YYYY-MM-DDTHH:MM:SS)
#
```

```
date_format=us
```

```
# TIMEZONE OFFSET
# This option is used to override the default timezone that this
# instance of Nagios runs in. If not specified, Nagios will use
# the system configured timezone.
#
# NOTE: In order to display the correct timezone in the CGIs, you
# will also need to alter the Apache directives for the CGI path
# to include your timezone. Example:
#
#   <Directory "/usr/local/nagios/sbin/">
#       SetEnv TZ "Australia/Brisbane"
#       ...
#   </Directory>
```

```
#use_timezone=US/Mountain
#use_timezone=Australia/Brisbane
```

```
# ILLEGAL OBJECT NAME CHARACTERS
```

```
# This option allows you to specify illegal characters that cannot
# be used in host names, service descriptions, or names of other
# object types.
```

```
illegal_object_name_chars=~!$%^&*|'"<>?,()=
```

```
# ILLEGAL MACRO OUTPUT CHARACTERS
```

```
# This option allows you to specify illegal characters that are
# stripped from macros before being used in notifications, event
# handlers, etc. This DOES NOT affect macros used in service or
# host check commands.
```

```
# The following macros are stripped of the characters you specify:
```

```
# $HOSTOUTPUT$
# $LONGHOSTOUTPUT$
# $HOSTPERFDATA$
# $HOSTACKAUTHOR$
# $HOSTACKCOMMENT$
# $SERVICEOUTPUT$
# $LONGSERVICEOUTPUT$
# $SERVICEPERFDATA$
# $SERVICEACKAUTHOR$
# $SERVICEACKCOMMENT$
```

```
illegal_macro_output_chars=~$&|'"<>
```

```
# REGULAR EXPRESSION MATCHING
```

```
# This option controls whether or not regular expression matching
# takes place in the object config files. Regular expression
# matching is used to match host, hostgroup, service, and service
# group names/descriptions in some fields of various object types.
# Values: 1 = enable regexp matching, 0 = disable regexp matching
```

```
use_regexp_matching=0
```



```
# "TRUE" REGULAR EXPRESSION MATCHING
# This option controls whether or not "true" regular expression
# matching takes place in the object config files. This option
# only has an effect if regular expression matching is enabled
# (see above). If this option is DISABLED, regular expression
# matching only occurs if a string contains wildcard characters
# (* and ?). If the option is ENABLED, regexp matching occurs
# all the time (which can be annoying).
# Values: 1 = enable true matching, 0 = disable true matching

use_true_regexp_matching=0


# ADMINISTRATOR EMAIL/PAGER ADDRESSES
# The email and pager address of a global administrator (likely you).
# Nagios never uses these values itself, but you can access them by
# using the $ADMINEMAIL$ and $ADMINPAGER$ macros in your notification
# commands.

admin_email=nagios@localhost
admin_pager=pagenagios@localhost


# DAEMON CORE DUMP OPTION
# This option determines whether or not Nagios is allowed to create
# a core dump when it runs as a daemon. Note that it is generally
# considered bad form to allow this, but it may be useful for
# debugging purposes. Enabling this option doesn't guarantee that
# a core file will be produced, but that's just life...
# Values: 1 - Allow core dumps
#          0 - Do not allow core dumps (default)
```

```
daemon_dumps_core=0
```

```
# LARGE INSTALLATION TWEAKS OPTION
```

```
# This option determines whether or not Nagios will take some shortcuts  
# which can save on memory and CPU usage in large Nagios installations.  
# Read the documentation for more information on the benefits/tradeoffs  
# of enabling this option.
```

```
# Values: 1 - Enabled tweaks
```

```
#          0 - Disable tweaks (default)
```

```
use_large_installation_tweaks=0
```

```
# ENABLE ENVIRONMENT MACROS
```

```
# This option determines whether or not Nagios will make all standard  
# macros available as environment variables when host/service checks  
# and system commands (event handlers, notifications, etc.) are  
# executed.
```

```
# Enabling this is a very bad idea for anything but very small setups,  
# as it means plugins, notification scripts and eventhandlers may run  
# out of environment space. It will also cause a significant increase  
# in CPU- and memory usage and drastically reduce the number of checks  
# you can run.
```

```
# Values: 1 - Enable environment variable macros
```

```
#          0 - Disable environment variable macros (default)
```

```
enable_environment_macros=0
```

```
# CHILD PROCESS MEMORY OPTION
```

```
# This option determines whether or not Nagios will free memory in  
# child processes (processed used to execute system commands and host/  
# service checks). If you specify a value here, it will override
```

```
# program defaults.
# Value: 1 - Free memory in child processes
#         0 - Do not free memory in child processes

#free_child_process_memory=1


# CHILD PROCESS FORKING BEHAVIOR
# This option determines how Nagios will fork child processes
# (used to execute system commands and host/service checks). Normally
# child processes are fork()ed twice, which provides a very high level
# of isolation from problems. Fork()ing once is probably enough and will
# save a great deal on CPU usage (in large installs), so you might
# want to consider using this. If you specify a value here, it will
# program defaults.
# Value: 1 - Child processes fork() twice
#         0 - Child processes fork() just once

#child_processes_fork_twice=1


# DEBUG LEVEL
# This option determines how much (if any) debugging information will
# be written to the debug file. OR values together to log multiple
# types of information.
# Values:
#         -1 = Everything
#         0 = Nothing
#         1 = Functions
#         2 = Configuration
#         4 = Process information
#         8 = Scheduled events
#        16 = Host/service checks
#        32 = Notifications
#        64 = Event broker
```

```
#      128 = External commands
#      256 = Commands
#      512 = Scheduled downtime
#      1024 = Comments
#      2048 = Macros
```

```
debug_level=256
```

```
# DEBUG VERBOSITY
```

```
# This option determines how verbose the debug log out will be.
```

```
# Values: 0 = Brief output
```

```
#      1 = More detailed
```

```
#      2 = Very detailed
```

```
debug_verbosity=1
```

```
# DEBUG FILE
```

```
# This option determines where Nagios should write debugging information.
```

```
debug_file=/opt/nagios/var/nagios.debug
```

```
# MAX DEBUG FILE SIZE
```

```
# This option determines the maximum size (in bytes) of the debug file.  If
```

```
# the file grows larger than this size, it will be renamed with a .old
```

```
# extension.  If a file already exists with a .old extension it will
```

```
# automatically be deleted.  This helps ensure your disk space usage doesn't
```

```
# get out of control when debugging Nagios.
```

```
max_debug_file_size=1000000
```

```
# Should we allow hostgroups to have no hosts, we default this to off since
# that was the old behavior

allow_empty_hostgroup_assignment=0


# Normally worker count is dynamically allocated based on 1.5 * number of cpu's
# with a minimum of 4 workers. This value will override the defaults

#check_workers=3


# DISABLE SERVICE CHECKS WHEN HOST DOWN
# This option will disable all service checks if the host is not in an UP state
#
# While desirable in some environments, enabling this value can distort report
# values as the expected quantity of checks will not have been performed

#host_down_disable_service_checks=0


# EXPERIMENTAL load controlling options
# To get current defaults based on your system issue a command to
# the query handler. Please note that this is an experimental feature
# and not meant for production use. Used incorrectly it can induce
# enormous latency.
# #core loadctl
#   jobs_max - The maximum amount of jobs to run at one time
#   jobs_min - The minimum amount of jobs to run at one time
#   jobs_limit - The maximum amount of jobs the current load lets us run
#   backoff_limit - The minimum backoff_change
#   backoff_change - # of jobs to remove from jobs_limit when backing off
#   rampup_limit - Minimum rampup_change
```

```
#   rampup_change - # of jobs to add to jobs_limit when ramping up
# NOTE: The backoff_limit and rampup_limit are NOT used by anything currently,
#       so if your system is under load nothing will actively modify the jobs
#       even if you have these options enabled, they are for external
#       connector information only.  However, if you change the jobs_max or
#       jobs_min manually here or through the query handler interface that
#       WILL affect your system
#loadctl_options=jobs_max=100;backoff_limit=10;rampup_change=5
cfg_dir=/opt/nagios/etc/conf.d
cfg_dir=/opt/nagios/etc/monitor
use_timezone=UTC
```

- **commands.cfg, contacts.cfg, timeperiods.cfg:** Para un monitoreo básico, Nagios ya incluye plantillas predefinidas que son referenciadas por las imágenes Docker. Sin embargo, para personalizar contactos o comandos avanzados, se crearían estos archivos en P:\nagios_docker\nagios_cfg y se añadirían las definiciones.
 - **commands.cfg:** Aquí se definirían comandos personalizados si los que vienen por defecto no son suficientes. Ejemplo:

```
define command{
    command_name    check_my_custom_script
    command_line    $USER1$/check_custom_script.sh -H $HOSTADDRESS$
                    -p $ARG1$
}
```

- **contacts.cfg:** Para definir a quién notificar.

```
define contact{
    contact_name    nagiosadmin
    use             generic-contact
```

```

    alias                Nagios Admin
    email                su_email@ejemplo.com
    # Se pueden agregar más métodos de notificación
}

define contactgroup{
    contactgroup_name    admins
    alias                Nagios Administrators
    members              nagiosadmin
}

```

- **timeperiods.cfg:** Para definir cuándo se envían notificaciones o se realizan verificaciones. Nagios ya tiene 24x7 y workhours definidos.

```

define timeperiod {
    timeperiod_name 24x7
    alias           24x7
    sunday          00:00-24:00
    monday          00:00-24:00
    tuesday         00:00-24:00
    wednesday       00:00-24:00
    thursday        00:00-24:00
    friday          00:00-24:00
    saturday        00:00-24:00
}

```

3. Iniciar los Contenedores Docker

Una vez que los archivos están configurados, se procede al despliegue.

- **Ejecución de Docker Compose:**
 - **Función:** Este comando lee el `docker-compose.yml` y levanta todos los servicios definidos en él.
 - **Dónde se corre:** Se abre la terminal de Windows (PowerShell o CMD), se navega a la carpeta donde se encuentra el `docker-compose.yml` (es decir, `P:\nagios_docker`) y se ejecuta:

```
docker-compose up -d
```

```

PS C:\Users\k1987\Desktop\nagios_docker> docker-compose up -d
time="2025-08-09T19:48:39-06:00" level=warning msg="C:\\Users\\k1987\\Desktop\\nagios_docker\\docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Running 3/3
 ✓ Container apache2      Started
 ✓ Container apache1      Started
 ✓ Container nagios_server Started

```

- up: Crea y levanta los contenedores.
- -d: Ejecuta los contenedores en modo "detached" (en segundo plano), lo que permite seguir usando la terminal.

• Verificación del Estado de los Contenedores:

- **Función:** Se comprueba que los contenedores se hayan iniciado correctamente.
- **Dónde se corre:** En la misma terminal, se ejecuta:

`docker-compose ps`

```

PS C:\Users\k1987\Desktop\nagios_docker> docker-compose ps
time="2025-08-09T19:50:51-06:00" level=warning msg="C:\\Users\\k1987\\Desktop\\nagios_docker\\docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
NAME                IMAGE              COMMAND                SERVICE  CREATED   STATUS    PORTS
apache1             httpd:latest      "httpd-foreground"    apache1  2 minutes ago Up 2 minutes 0.0.0.0:8081->80/tcp
apache2             httpd:latest      "httpd-foreground"    apache2  2 minutes ago Up 2 minutes 0.0.0.0:8082->80/tcp
nagios_server       jasonrivers/nagios:latest "/usr/local/bin/star...  nagios   6 minutes ago Up 2 minutes 5667/tcp, 0.0.0.0:8080->80/tcp
PS C:\Users\k1987\Desktop\nagios_docker>

```

Esto mostrará el estado de los servicios definidos en el `docker-compose.yml`. Se debería ver el `nagios_server` con un estado Up.

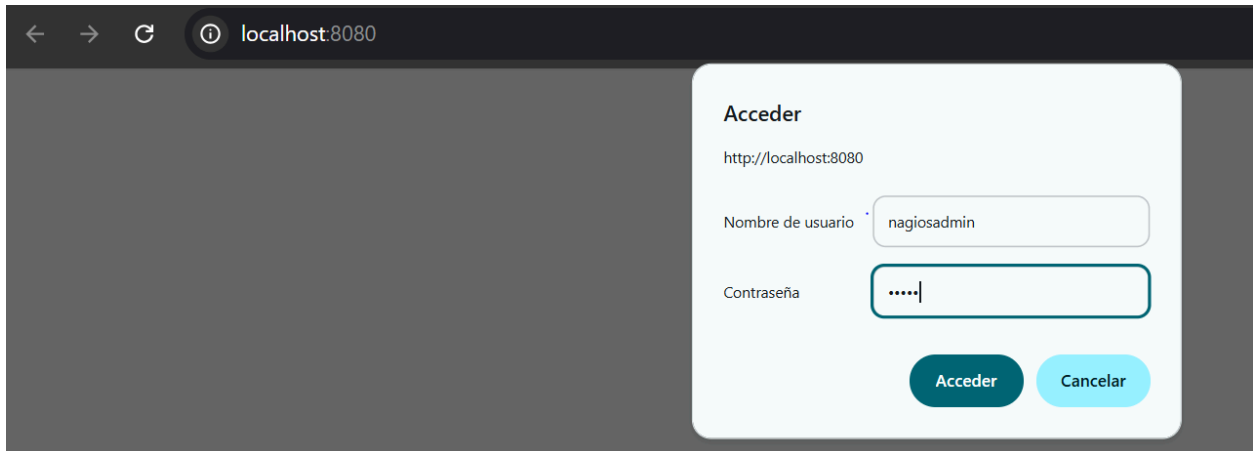
4. Acceso a la Interfaz Web de Nagios

Con los contenedores en funcionamiento, es momento de acceder a la interfaz gráfica.

• Acceso al Navegador:

- **Función:** Permite interactuar con Nagios, ver el estado de los hosts y servicios, y acceder a los informes.
- **Dónde se corre/configura:** Se abre un navegador web (Chrome, Firefox, Edge) y se ingresa la siguiente **URL: `http://localhost:8080/nagios`**

- **Credenciales:** Se solicitarán credenciales. Se utilizan las definidas en el `docker-compose.yml`:
 - **Usuario:** nagiosadmin
 - **Contraseña:** La que se haya configurado en la variable NAGIOSADMIN_PASS (ejemplo. SuContrasenaSeguraAqui).



Monitoreo

Dentro de Nagios se encuentran diferentes dashboards de monitoreo e información.

La sección de **“Tactical Overview”**, proporciona una vista consolidada en tiempo real de todos los hosts y servicios monitoreados, lo que permite una evaluación rápida del estado general del sistema y de posibles problemas. Permite a los usuarios ver el estado de todos los hosts y servicios, identificar problemas que requieren atención y realizar acciones rápidas, como reconocer problemas o desactivar notificaciones.

En este caso, se tienen 2 servicios monitoreados, que corresponden a los servidores Apache_1 y Apache_2.

Monitoring Performance	
Service Check Execution Time:	0.01 / 0.01 / 0.007 sec
Service Check Latency:	0.00 / 0.15 / 0.074 sec
Host Check Execution Time:	4.17 / 4.21 / 4.190 sec
Host Check Latency:	0.00 / 0.04 / 0.020 sec
# Active Host / Service Checks:	2 / 2
# Passive Host / Service Checks:	0 / 0

Network Outages

0 Outages

Network Health

Host Health:

Service Health:

Monitoring Features					
Flap Detection	Notifications	Event Handlers	Active Checks	Passive Checks	
✓ <div>All Services Enabled</div> <div>No Services Flapping</div> <div>All Hosts Enabled</div> <div>No Hosts Flapping</div>	✓ <div>All Services Enabled</div> <div>All Hosts Enabled</div>	✓ <div>All Services Enabled</div> <div>All Hosts Enabled</div>	✓ <div>All Services Enabled</div> <div>All Hosts Enabled</div>	✓ <div>All Services Enabled</div> <div>All Hosts Enabled</div>	✓ <div>All Services Enabled</div> <div>All Hosts Enabled</div>

La sección de “**Network Map for All Hosts**”, proporciona una representación visual de la jerarquía de su red, mostrando los hosts y sus estados (activo, inactivo, inalcanzable) en un diseño circular u otro.

Los usuarios pueden elegir cómo visualizar su infraestructura con la opción de layout method o método de diseño y dar click en update para las diferentes vistas.

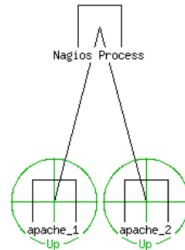
Network Map For All Hosts
 Last Updated: Sun Aug 10 04:14:47 UTC 2025
 Updated every 90 seconds
 Nagios® Core™ 4.5.7 - www.nagios.org
 Logged in as [nagiosadmin](#)

[View Status Detail For All Hosts](#)
[View Status Overview For All Hosts](#)

Layout Method: Balanced tree Scaling factor: 0.0

Drawing Layers: 0 Layer mode: ☐ Include ☒ Exclude

Suppress popups: ☐ Update



En la sección de Servicios, “**current network status**” o “estado actual de la red”, los usuarios pueden ver de manera rápida, el estado de todos los hosts y servicios que se monitorean.

Host Status Totals para ver la cantidad de hosts en cada estado:

- **Up**: número de hosts activos (funcionando bien).
- **Down**: hosts caídos.
- **Unreachable**: no se puede acceder al host, pero la causa puede ser un problema de red intermedio.
- **Pending**: Nagios aún no ha hecho la primera comprobación de ese host.

Service Status Totals: estado de los servicios monitoreados.

- **Ok**: funcionando correctamente.
- **Warning**: advertencia (por ejemplo, uso de disco > 80%).
- **Unknown**: Nagios no puede determinar el estado.
- **Critical**: servicio caído o fuera de parámetros críticos.
- **Pending**: aún no evaluado.

Current Network Status
 Last Updated: Sun Aug 10 04:18:48 UTC 2025
 Updated every 90 seconds
 Nagios® Core™ 4.5.7 - www.nagios.org
 Logged in as [nagiosadmin](#)

[View Service Status Detail For All Host Groups](#)
[View Status Overview For All Host Groups](#)
[View Status Summary For All Host Groups](#)
[View Status Grid For All Host Groups](#)

Host Status Totals

Up	Down	Unreachable	Pending
2	0	0	0

[All Problems](#) [All Types](#)

0	2
---	---

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
2	0	0	0	0

[All Problems](#) [All Types](#)

0	2
---	---

Host Status Details For All Host Groups

Limit Results: 100

Host	Status	Last Check	Duration	Status Information
apache_1	UP	08-10-2025 04:15:17	0d 0h 23m 31s	PING OK - Packet loss = 0%, RTA = 0.10 ms
apache_2	UP	08-10-2025 04:17:46	0d 0h 21m 2s	PING OK - Packet loss = 0%, RTA = 0.11 ms

Results 1 - 2 of 2 Matching Hosts

Tabla con detalle por host

- **Host:** nombre del host monitoreado (ej. apache_1, apache_2).
- **Status:** estado actual del host (verde = UP).
Last Check: última vez que Nagios lo verificó.
- **Duration:** cuánto tiempo lleva en ese estado.
- **Status Information:** resultado del último chequeo (en este caso, un ping con 0% pérdida de paquetes y latencia promedio).

Se puede hacer click en los servidores para ver más detalles. En el caso de apache_1 y apache_2, el tipo de servicio monitoreado es HTTP, muestra información sobre el último chequeo que se realizó, la duración, número de intentos e información de estado.

Limit Results: 100

Host	Service	Status	Last Check	Duration	Attempt	Status Information
apache_2	HTTP	OK	08-10-2025 04:29:52	0d 0h 31m 57s	1/3	HTTP OK: HTTP/1.1 200 OK - 289 bytes in 0.001 second response time

Limit Results: 100

Host	Service	Status	Last Check	Duration	Attempt	Status Information
apache_1	HTTP	OK	08-10-2025 04:25:16	0d 0h 29m 14s	1/3	HTTP OK: HTTP/1.1 200 OK - 289 bytes in 0.002 second response time

Host Information

Last Updated: Sun Aug 10 04:48:41 UTC 2025
Updated every 90 seconds
Nagios® Core™ 4.5.7 - www.nagios.org
Logged in as nagiosadmin

View Status Detail For This Host

View Alert History For This Host

View Trends For This Host

View Alert Histogram For This Host

View Availability Report For This Host

View Notifications For This Host

Host

Servidor Web Apache 1
(apache_1)

Member of

No hostgroups

apache_1

Host State Information

Host Status:

UP (for 0d 0h 53m 24s)

Status Information:

PING OK - Packet loss = 0%, RTA = 0.10 ms

Performance Data:

rta=0.097000ms;3000.000000;5000.000000;0.000000 pl=0%;80;100;0

Current Attempt:

1/3 (HARD state)

Last Check Time:

08-10-2025 04:45:17

Check Type:

ACTIVE

Check Latency / Duration:

0.000 / 4.213 seconds

Next Scheduled Active Check:

08-10-2025 04:50:17

Last State Change:

08-10-2025 03:55:17

Last Notification:

N/A (notification 0)

Is This Host Flapping?

NO (0.00% state change)

In Scheduled Downtime?

NO

Last Update:

08-10-2025 04:48:15 (0d 0h 0m 26s ago)

Active Checks:

ENABLED

Passive Checks:

ENABLED

Obsessing:

ENABLED

Notifications:

ENABLED

Event Handler:

ENABLED

Flap Detection:

ENABLED

Host Commands

Locate host on map

Disable active checks of this host

Re-schedule the next check of this host

Submit passive check result for this host

Stop accepting passive checks for this host

Stop obsessing over this host

Disable notifications for this host

Send custom host notification

Schedule downtime for this host

Schedule downtime for all services on this host

Disable notifications for all services on this host

Enable notifications for all services on this host

Schedule a check of all services on this host

Disable checks of all services on this host

Enable checks of all services on this host

Disable event handler for this host

Disable flap detection for this host

Clear flapping state for this host

Host Comments

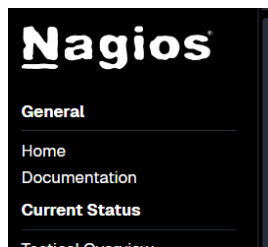
Add a new comment

Delete all comments

Reportes Disponibles

Reporte de Disponibilidad: Se puede acceder a los Informes de Disponibilidad de Nagios y generarlos a través de la interfaz de Nagios XI, lo que proporciona información sobre el tiempo de actividad y de inactividad de hosts y servicios. Estos informes ofrecen una representación visual, a menudo mediante gráficos circulares, del porcentaje de tiempo que un sistema estuvo en diferentes estados (p. ej., correcto, advertencia, crítico, desconocido, inactivo, inaccesible) durante un período específico.

Para configurar, primero seleccionar el tipo de reporte, en este caso se tiene un grupo de hosts para monitorear la disponibilidad de Apache_1 y Apache_2.



Availability Report
Last Updated: Sun Aug 10 04:34:42 UTC 2025
Nagios® Core™ 4.5.7 - www.nagios.org
Logged in as *nagiosadmin*

Step 1: Select Report Type

Type:

Se seleccionan todos los grupos de hosts.

Step 2: Select Hostgroup

Hostgroup(s):

El usuario puede seleccionar entre diferentes opciones para crear reportes de disponibilidad, por ejemplo el período de tiempo a reportar, puede elegir rangos específicos de fechas, coleccionar datos 24x7, o por horas de trabajo, o excluir días feriados. Además, los resultados pueden exportarse en un archivo csv si se requiere.

Step 3: Select Report Options

Report Period:	Last 7 Days ▼		
If Custom Report Period...			
Start Date (Inclusive):	August ▼	1	2025
End Date (Inclusive):	August ▼	10	2025
Report time Period:	24x7 ▼		
Assume Initial States:	Yes ▼		
Assume State Retention:	Yes ▼		
Assume States During Program Downtime:	Yes ▼		
Include Soft States:	No ▼		
First Assumed Host State:	Current State ▼		
First Assumed Service State:	Unspecified ▼		
Backtracked Archives (To Scan For Initial States):	4		
Output in CSV Format:	<input checked="" type="checkbox"/>		
<button>Create Availability Report!</button>			

Trends

En la sección de trends, los usuarios pueden usar el reporte de Host State Trends para de manera histórica, ver cómo ha cambiado el estado de un host a lo largo del tiempo.

Dentro de la organización, es importante para que los equipos que manejan infraestructura puedan verificar:

- ¿Cuántas veces se ha caído este host?
- ¿Cuánto tiempo estuvo caído en total en el último mes?
- ¿Qué porcentaje del tiempo estuvo disponible?

De esta manera, y haciendo uso de logs se pueden determinar patrones de fallos, verificar si estos incidentes son recurrentes o aislados, mantener documentación de incidentes para propósitos de auditorías, o evaluaciones y mejoras posibles dentro de la infraestructura para disminuir y prevenir incidentes.

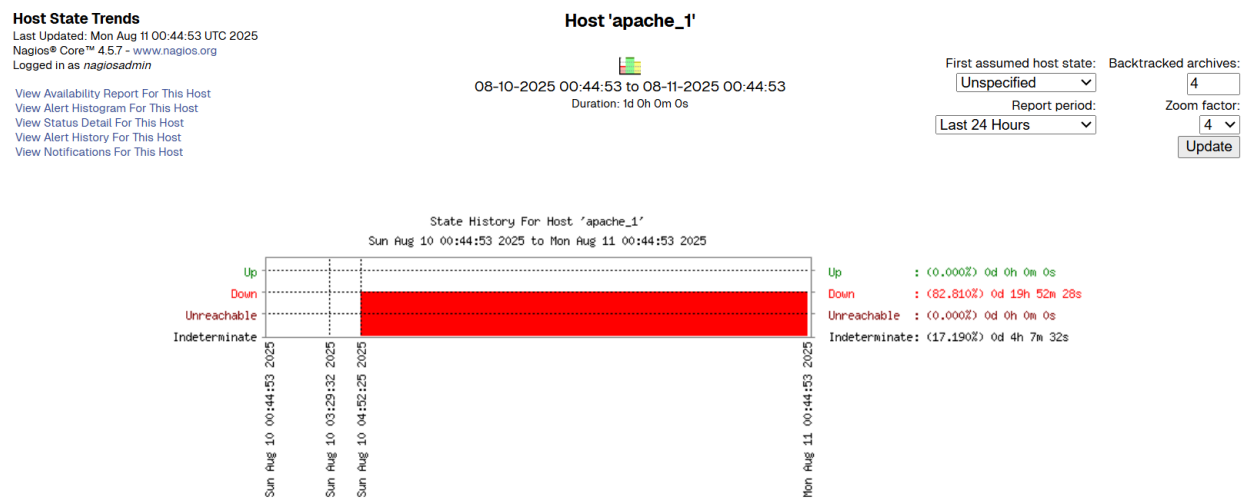
En el ejemplo simulado de fallo, este reporte muestra un gráfico de líneas que indican los estados de cambio del host (up, down, unreachable) con base en el periodo de tiempo a seleccionar.

En este caso el host Apache_1, estuvo caído por 19 horas, 52 minutos y 28 segundos durante el periodo analizado (24 horas). Esto puede brindar información sobre posibles problemas con el servidor, al no estar encendido no podría responder al ping/chequeo definido.

El estado indeterminado corresponde a 4 horas y 7 minutos en los que Nagios no tiene datos para determinar el estado (puede ser porque el servicio de Nagios estuvo apagado o el host no tenía aún chequeos configurados).

Además, muestra una tabla detallada con:

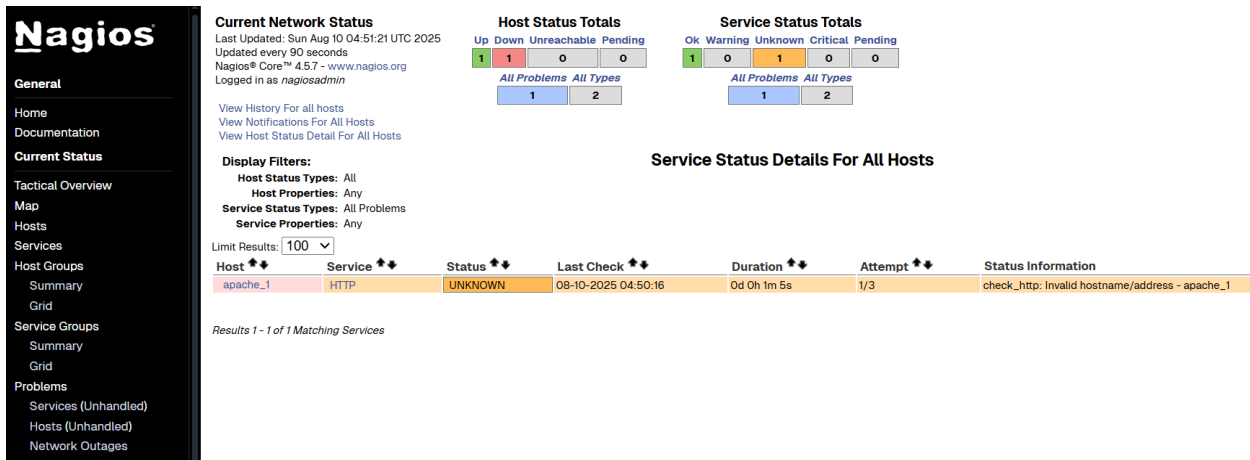
- **Start Time / End Time:** cuándo empezó y terminó cada estado.
- **Duration:** cuánto duró cada evento.
- **State:** estado durante ese período.
- **Type:** si fue un estado confirmado (HARD) o temporal (SOFT).
- **Attempt:** cuántos intentos de verificación hizo Nagios antes de confirmar el estado.



Problemas

En Nagios, "hosts no gestionados" se refiere a hosts que se encuentran actualmente en estado problemático (por ejemplo, inactivos o inaccesibles) y que no han sido reconocidos ni programados como para estar en un estado de mantenimiento. Este estado indica un problema activo que requiere la atención de un administrador.

La imagen indica que el host Apache_1 es el servicio monitoreado, con HTTP el tipo de servicio que Nagios intenta comprobar. En este caso el estado Unknown indica que Nagios no puede determinar si el servicio está OK, Warning o Critical. Además, da información sobre la última vez que se intentó verificar el servicio, además de la duración que lleva en el estado actual y la cantidad de intentos hechos antes de considerar el estado como "hard", en este caso lleva 1 intento de un máximo de 3. Y finalmente, la parte de status information, lo que indica es que el comando de chequeo de HTTP falló porque el nombre de host o dirección no es válido o no se puede resolver.



Notificaciones

Esta sección de Contact Notifications en Nagios muestra el historial de notificaciones enviadas a los contactos cuando ha ocurrido un problema. Las columnas indican lo siguiente:

- **Host:** apache_1 es el host que presentó el problema.
- **Service:** N/A indica el estado del host.
- **Type:** HOST DOWN (en rojo) indica que el host fue marcado como caído.

- **Time:** 08-10-2025 04:52:27 y 04:52:25 indica la hora en que se enviaron las notificaciones.
- **Contact:** nagiosadmin es el contacto que recibió la alerta.
- **Notification Command:** notify-host-by-email fue el comando usado para enviar la notificación (en este caso, correo electrónico).
- **Information:** check_ping: Invalid hostname/address - apache_1 Esto indica que el comando check_ping no pudo resolver o encontrar el host apache_1.

En resumen, esta sección indica que el host apache_1 no se puede resolver (no está en DNS, /etc/hosts o red Docker compartida). Debido a lo anterior, esto provocó que el host se considere DOWN y que se enviaran notificaciones al contacto configurado.

Contact Notifications

Last Updated: Sun Aug 10 04:54:46 UTC 2025
Nagios® Core™ 4.5.7 - www.nagios.org
Logged in as nagiosadmin

All Contacts

Notification detail level for all contacts:

All notifications

Older Entries First:

☐

Update

Latest Archive



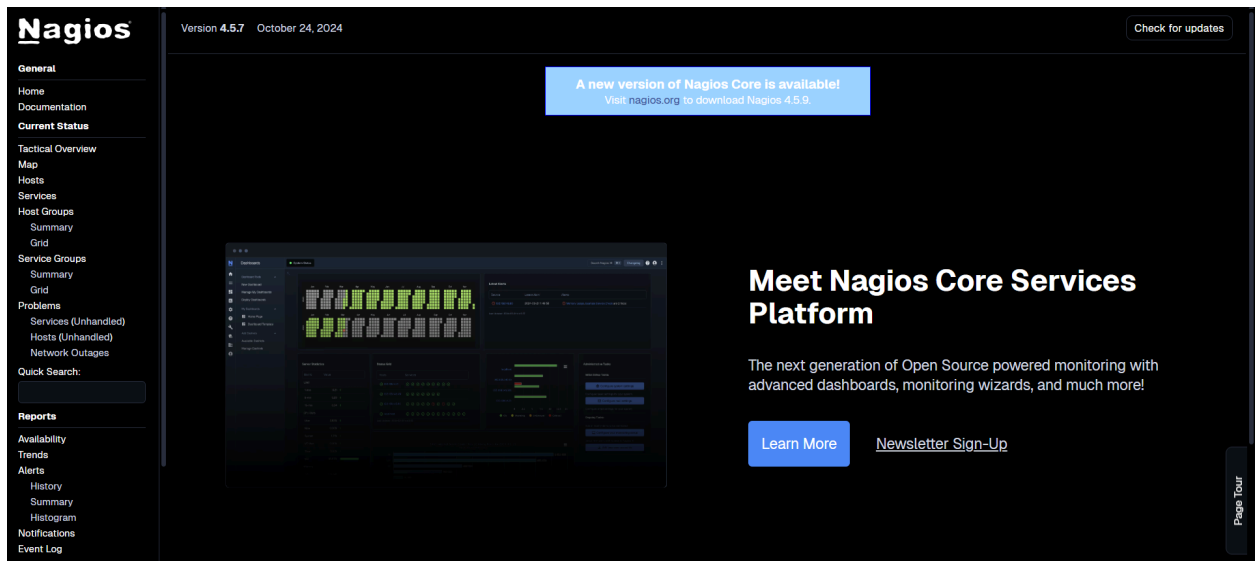
Log File Navigation

Sun Aug 10 00:00:00 UTC 2025
to
Present.

File: /opt/nagios/var/nagios.log

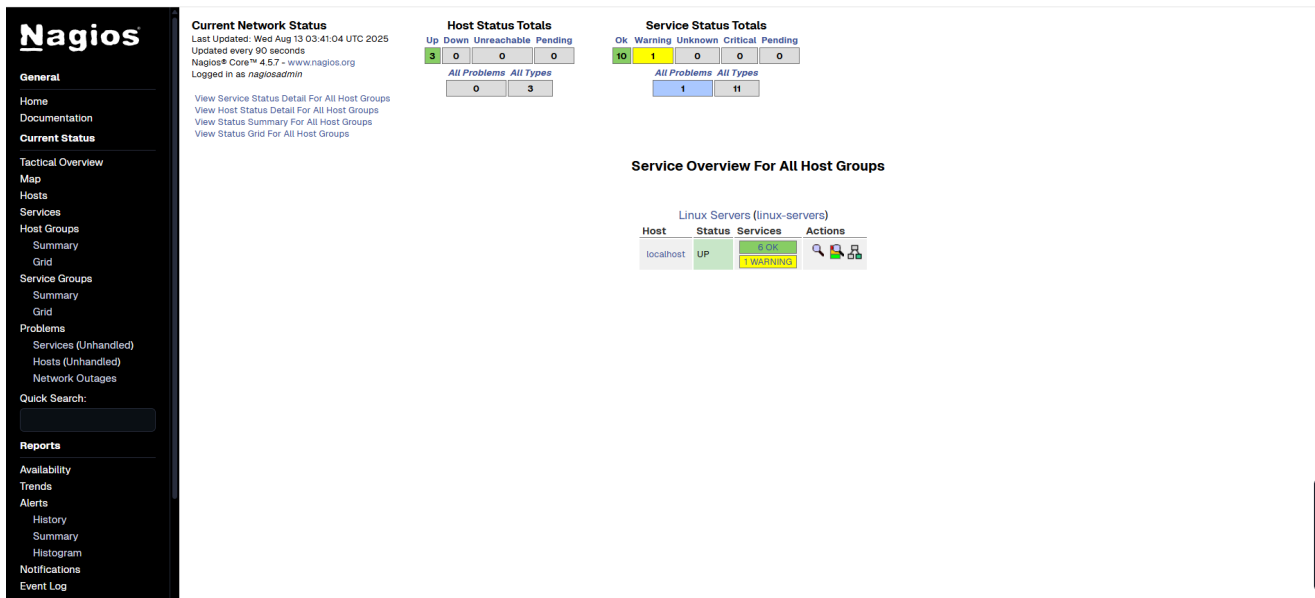
Host	Service	Type	Time	Contact	Notification Command	Information
apache_1	N/A	HOST DOWN	08-10-2025 04:52:27	nagiosadmin	notify-host-by-email	check_ping: Invalid hostname/address - apache_1
apache_1	N/A	HOST DOWN	08-10-2025 04:52:25	nagiosadmin	notify-host-by-email	check_ping: Invalid hostname/address - apache_1

Home



- Suele mostrar un mensaje de bienvenida, versión instalada, noticias y enlaces a documentación oficial

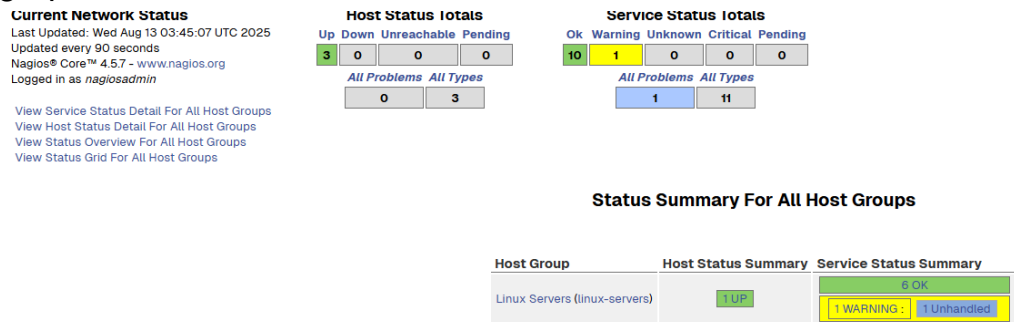
Host Groups



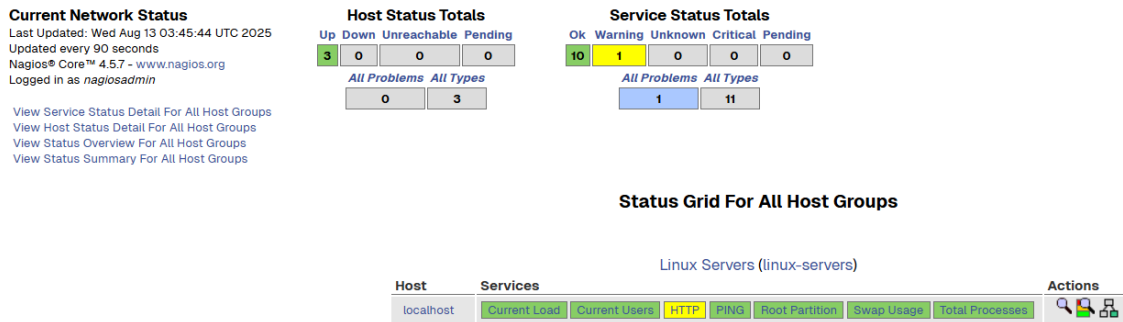
- Se pueden asignar *checks* o *servicios* a todo un grupo, sin tener que configurar host por host.
- Esto ahorra tiempo cuando hay decenas o cientos de hosts similares.
- Los reportes de disponibilidad y tendencias pueden generarse por grupo.
- Permite filtrar alertas solo para un grupo específico.

En tu panel hay dos subopciones:

- **Summary** → Muestra un resumen del estado de todos los hosts en ese grupo.



- **Grid** → Presenta una vista en cuadrícula, útil para identificar rápidamente problemas.



Problem Services

En Nagios, un **servicio** es algo específico que se monitorea dentro de un host, por ejemplo:

- Estado de un puerto HTTP
- Disponibilidad de un servicio SSH
- Espacio en disco
- Uso de CPU

Si es un chequeo que requiere usuario/contraseña → Configura las credenciales en el comando de chequeo HTTP de Nagios.

Si quieres marcarlo como “atendido” → Haz clic en el icono de **check** o en el nombre del servicio y selecciona **Acknowledge this service problem**.

Si es un falso positivo → Ajusta el comando o los umbrales del chequeo.

General

Home

Documentation

Current Status

Tactical Overview

Map

Hosts

Services

Host Groups

Summary

Grid

Service Groups

Summary

Grid

Problems

Services (Unhandled)

Hosts (Unhandled)

Network Outages

Quick Search:

Current Network Status

Last Updated: Wed Aug 13 03:46:24 UTC 2025

Updated every 90 seconds

Nagios® Core™ 4.5.7 - www.nagios.org

Logged in as nagiosadmin

View History For all hosts

View Notifications For All Hosts

View Host Status Detail For All Hosts

Host Status Totals

Up	Down	Unreachable	Pending
3	0	0	0

All Problems

All Types

0	3
---	---

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
10	1	0	0	0

All Problems

All Types

1	11
---	----

Display Filters:

Host Status Types: All

Host Properties: Any

Service Status Types: All Problems

Service Properties: Any

Limit Results: 100

Host	Service	Status	Last Check	Duration	Attempt	Status Information
localhost	HTTP	WARNING	08-13-2025 03:45:21	0d 0h 36m 34s	4/4	HTTP WARNING: HTTP/1.1 401 Unauthorized - 695 bytes in 0.000 second response time

Results 1 - 1 of 1 Matching Services

Service Status Details For All Hosts

¿Qué es un *Network Outage* en Nagios?

- Es una interrupción total de conectividad hacia un host o grupo de hosts.
- Se produce cuando un host que está como “padre” (host de nivel superior) está caído, y como consecuencia todos sus hosts “hijos” quedan inaccesibles.
- En lugar de mostrar muchos errores repetidos, Nagios los agrupa y los presenta como un solo evento de caída de red.

General

Home

Documentation

Current Status

Tactical Overview

Map

Hosts

Services

Host Groups

Summary

Grid

Service Groups

Summary

Grid

Problems

Services (Unhandled)

Hosts (Unhandled)

Network Outages

Quick Search:

Network Outages

Last Updated: Wed Aug 13 03:50:10 UTC 2025

Updated every 90 seconds

Nagios® Core™ 4.5.7 - www.nagios.org

Logged in as nagiosadmin

Blocking Outages

Severity	Host	State	Notes	State Duration	# Hosts Affected	# Services Affected	Actions
0 Blocking Outages Displayed							

Event log

En el *Event Log* se ve la secuencia:

- 03:06 – 03:09: Nagios marca el HTTP como SOFT WARNING → aún en fase de intentos.

- 03:09: Se convierte en HARD WARNING porque falló en el número de intentos configurado (4/4 en tu caso).
- El mensaje “HTTP/1.1 401 Unauthorized – 695 bytes” confirma que la conexión fue rápida, pero el servidor exigió autenticación.
- También aparecen advertencias de configuración:
 - normal_check_interval está obsoleto, se recomienda usar check_interval.
 - failure_prediction_enabled ya no tiene efecto en la versión actual.

Current Event Log

Last Updated: Wed Aug 13 03:51:39 UTC 2025
Nagios® Core™ 4.5.7 - www.nagios.org
Logged in as nagiosadmin

Latest Archive



Log File Navigation

Wed Aug 13 00:00:00 UTC 2025
to
Present.

☐ Older Entries First

[Update](#)

File: /opt/nagios/var/nagios.log

August 13, 2025 03:00

```
[08-13-2025 03:09:50] SERVICE ALERT: localhost:HTTP-WARNING:HARD:4;HTTP WARNING: HTTP/1.1 401 Unauthorized - 695 bytes in 0.000 second response time
[08-13-2025 03:08:50] SERVICE ALERT: localhost:HTTP-WARNING:SOFT:3;HTTP WARNING: HTTP/1.1 401 Unauthorized - 695 bytes in 0.000 second response time
[08-13-2025 03:07:50] SERVICE ALERT: localhost:HTTP-WARNING:SOFT:2;HTTP WARNING: HTTP/1.1 401 Unauthorized - 695 bytes in 0.001 second response time
[08-13-2025 03:06:51] SERVICE ALERT: localhost:HTTP-WARNING:SOFT:1;HTTP WARNING: HTTP/1.1 401 Unauthorized - 695 bytes in 0.001 second response time
[08-13-2025 03:02:20] Successfully launched command file worker with pid 60
[08-13-2025 03:02:19] WARNING: The normal_check_interval attribute is deprecated and will be removed in future versions. Please use check_interval instead.
[08-13-2025 03:02:19] Warning: failure_prediction_enabled is obsolete and no longer has any effect in service type objects (config file /opt/nagios/etc/servers/services.cfg, starting at line 2)
[08-13-2025 03:02:19] wproc: Registry request: name=Core Worker 51;pid=51
[08-13-2025 03:02:19] wproc: Registry request: name=Core Worker 52;pid=52
[08-13-2025 03:02:19] wproc: Registry request: name=Core Worker 47;pid=47
[08-13-2025 03:02:19] wproc: Registry request: name=Core Worker 50;pid=50
[08-13-2025 03:02:19] wproc: Registry request: name=Core Worker 53;pid=53
[08-13-2025 03:02:19] wproc: Registry request: name=Core Worker 54;pid=54
[08-13-2025 03:02:19] wproc: Registry request: name=Core Worker 48;pid=48
[08-13-2025 03:02:19] wproc: Registry request: name=Core Worker 49;pid=49
[08-13-2025 03:02:19] wproc: Registry request: name=Core Worker 46;pid=46
[08-13-2025 03:02:19] wproc: Registry request: name=Core Worker 45;pid=45
[08-13-2025 03:02:19] wproc: Registry request: name=Core Worker 41;pid=41
[08-13-2025 03:02:19] wproc: Registry request: name=Core Worker 43;pid=43
[08-13-2025 03:02:19] wproc: Registry request: name=Core Worker 44;pid=44
[08-13-2025 03:02:19] wproc: Registry request: name=Core Worker 42;pid=42
[08-13-2025 03:02:19] wproc: Registry request: name=Core Worker 38;pid=38
[08-13-2025 03:02:19] wproc: Registry request: name=Core Worker 39;pid=39
[08-13-2025 03:02:19] wproc: Registry request: name=Core Worker 37;pid=37
[08-13-2025 03:02:19] wproc: Registry request: name=Core Worker 40;pid=40
[08-13-2025 03:02:19] wproc: Successfully registered manager as @wproc with query handler
[08-13-2025 03:02:19] qh: help for the query handler registered
[08-13-2025 03:02:19] qh: echo service query handler registered
[08-13-2025 03:02:19] qh: core query handler registered
[08-13-2025 03:02:19] qh: Socket /opt/nagios/var/rw/nagios.qh successfully initialized
[08-13-2025 03:02:19] LOG VERSION: 2.0
[08-13-2025 03:02:19] Local time is Wed Aug 13 03:02:19 UTC 2025
[08-13-2025 03:02:19] Nagios 4.5.7 starting... (PID=20)
```

Process info

- **Program Version:** 4.5.7 → Versión actual de Nagios Core.
- **Program Start Time:** 08-13-2025 03:02:19 → Hora en que se inició el proceso.
- **Total Running Time:** 0h 54min 19s → Tiempo que lleva activo.
- **Nagios PID:** 20 → ID de proceso en el contenedor.
- **Notificaciones / Checks:** Todas las casillas de **YES** indican que las verificaciones y notificaciones están habilitadas.

Qué significa cada indicador en verde

- **Notifications Enabled:** Nagios enviará alertas si detecta un problema.
- **Service Checks Being Executed:** Se están ejecutando verificaciones activas de servicios.
- **Passive Service Checks Being Accepted:** Se aceptan resultados enviados por otros sistemas o scripts externos (modo pasivo).
- **Host Checks Being Executed:** Se están ejecutando verificaciones activas de hosts.
- **Passive Host Checks Being Accepted:** Se aceptan resultados pasivos para hosts.

Comandos de control (lado derecho)

- Los botones con **X roja** significan que esas funciones están activas y si presionas el enlace, las desactivarías.
- Los botones con **flecha azul** como *Restart the Nagios process* permiten reiniciar Nagios desde la interfaz.

Process Information		Process Commands
Program Version:	4.5.7	Shutdown the Nagios process
Program Start Time:	08-13-2025 03:02:19	Restart the Nagios process
Total Running Time:	0d 0h 54m 19s	Disable notifications
Last Log File Rotation:	N/A	Stop executing service checks
Nagios PID	20	Stop accepting passive service checks
Notifications Enabled?	YES	Stop executing host checks
Service Checks Being Executed?	YES	Stop accepting passive host checks
Passive Service Checks Being Accepted?	YES	Disable event handlers
Host Checks Being Executed?	YES	Start obsessing over services
Passive Host Checks Being Accepted?	YES	Start obsessing over hosts
Event Handlers Enabled?	Yes	Disable flap detection
Obsessing Over Services?	No	Disable performance data
Obsessing Over Hosts?	No	
Flap Detection Enabled?	Yes	
Performance Data Being Processed?	Yes	

Scheduling Queue

Host → El nombre del host monitorizado (por ejemplo, `localhost`, `apache1_host`, `apache2_host`).

Service → El servicio específico que se está chequeando (HTTP, Ping, carga de CPU, uso de swap, etc.).

Last Check → Fecha y hora en que se hizo la última comprobación.

Next Check → Fecha y hora en que se hará la siguiente comprobación.

Active Checks → Si está en **ENABLED**, Nagios ejecutará ese chequeo automáticamente de forma periódica.

Actions → Botones para detener o reprogramar el chequeo manualmente.

Entries sorted by next check time (ascending)

Host	Service	Last Check	Next Check	Type	Active Checks	Actions
apache2_host		08-13-2025 03:55:21	08-13-2025 04:00:21	Normal	ENABLED	
apache1_host		08-13-2025 03:55:21	08-13-2025 04:00:21	Normal	ENABLED	
localhost	Total Processes	08-13-2025 03:55:21	08-13-2025 04:00:21	Normal	ENABLED	
localhost	Root Partition	08-13-2025 03:55:21	08-13-2025 04:00:21	Normal	ENABLED	
localhost	PING	08-13-2025 03:55:21	08-13-2025 04:00:21	Normal	ENABLED	
localhost	HTTP	08-13-2025 03:55:21	08-13-2025 04:00:21	Normal	ENABLED	
localhost	Current Load	08-13-2025 03:55:21	08-13-2025 04:00:21	Normal	ENABLED	
apache1_host	Ping Check	08-13-2025 03:55:21	08-13-2025 04:00:21	Normal	ENABLED	
apache1_host	HTTP Web Service	08-13-2025 03:55:21	08-13-2025 04:00:21	Normal	ENABLED	
localhost	Current Users	08-13-2025 03:55:29	08-13-2025 04:00:29	Normal	ENABLED	
apache2_host	Ping Check	08-13-2025 03:55:45	08-13-2025 04:00:45	Normal	ENABLED	
localhost	Swap Usage	08-13-2025 03:55:57	08-13-2025 04:00:57	Normal	ENABLED	
apache2_host	HTTP Web Service	08-13-2025 03:56:24	08-13-2025 04:01:24	Normal	ENABLED	
localhost		08-13-2025 03:56:52	08-13-2025 04:01:52	Normal	ENABLED	

Services groups summary

Muestra un resumen del estado de todos los grupos de servicios configurados en el sistema.

Esto permite a los administradores de sistemas obtener una visión general rápida del estado de salud de los servicios críticos de la infraestructura de red, lo que les indica si estos operan correctamente o si presentan problemas.

Nagios

General

Home

Documentation

Current Status

Tactical Overview

Map

Hosts

Services

Host Groups

Summary

Grid

Service Groups

Summary

Grid

Problems

Services (Unhandled)

Hosts (Unhandled)

Network Outages

Quick Search:

Reports

Availability

Trends

Alerts

History

Summary

Heatmap

Notifications

Event Log

System

Comments

Downtime

Process Info

Performance Info

Scheduling Queue

Configuration

Current Network Status

Last Updated: Thu Aug 14 17:06:32 UTC 2025

Updated every 90 seconds

Nagios® Core™ 4.5.7 - www.nagios.org

Logged in as nagiosadmin

View Service Status Detail For All Service Groups

View Status Overview For All Service Groups

View Service Status Grid For All Service Groups

Host Status Totals

Up Down Unreachable Pending

3 0 0 0

All Problems All Types

0 3

Service Status Totals

Ok Warning Unknown Critical Pending

10 1 0 0 0

All Problems All Types

1 11

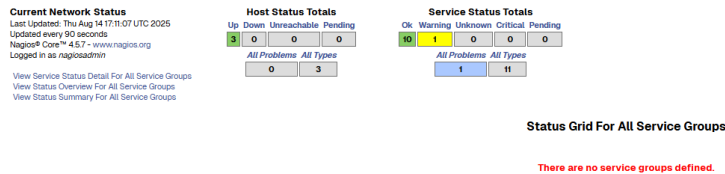
Status Summary For All Service Groups

Service Group Host Status Summary Service Status Summary

There are no service groups defined.

Services groups grid

Es una vista que muestra el estado de todos los grupos de servicios definidos en la configuración de Nagios en una sola tabla o cuadrícula. Permite a los administradores de sistemas monitorear rápidamente el estado de múltiples servicios y hosts agrupados lógicamente, lo que facilita la identificación de problemas y la gestión de la infraestructura de red.



Alertas

La historia de alertas de Nagios registra los eventos de alerta que han ocurrido en el sistema, proporcionando un historial de las notificaciones generadas.

Alert History

Last Updated: Thu Aug 14 17:54:45 UTC 2025

Nagios® Core™ 4.5.7 - www.nagios.org

Logged in as nagiosadmin

View Status Detail For All Hosts

View Notifications For All Hosts

All Hosts and Services

Latest Archive

Log File Navigation

Thu Aug 14 00:00:00 UTC 2025

to

Present.

File: /opt/nagios/var/nagios.log

State type options:

All state types

History detail level for all hosts:

All alerts

☐ Hide Flapping Alerts

☐ Hide Downtime Alerts

☐ Hide Process Messages

☐ Older Entries First

Update

August 14, 2025 17:00

08-14-2025 17:00:20 Nagios 4.5.7 starting... (PID=20)

August 14, 2025 16:00

08-14-2025 16:59:50 Caught SIGTERM, shutting down...

08-14-2025 16:48:23 Nagios 4.5.7 starting... (PID=22)

August 12, 2025 03:00

08-12-2025 03:58:28 Nagios 4.5.7 starting... (PID=23)

08-12-2025 03:58:21 Caught SIGTERM, shutting down...

08-12-2025 03:56:29 Nagios 4.5.7 starting... (PID=28)

08-12-2025 03:56:07 Caught SIGTERM, shutting down...

08-12-2025 03:54:04 Nagios 4.5.7 starting... (PID=27)

08-12-2025 03:47:08 Bailing out due to one or more errors encountered in the configuration files. Run Nagios from the command line with the -v option to verify your config before restarting. (PID=8905)

08-12-2025 03:47:08 Nagios 4.5.7 starting... (PID=8906)

08-12-2025 03:47:07 Bailing out due to one or more errors encountered in the configuration files. Run Nagios from the command line with the -v option to verify your config before restarting. (PID=8882)

08-12-2025 03:47:07 Nagios 4.5.7 starting... (PID=8883)

08-12-2025 03:47:06 Bailing out due to one or more errors encountered in the configuration files. Run Nagios from the command line with the -v option to verify your config before restarting. (PID=8859)

08-12-2025 03:47:06 Nagios 4.5.7 starting... (PID=8859)

08-12-2025 03:47:05 Bailing out due to one or more errors encountered in the configuration files. Run Nagios from the command line with the -v option to verify your config before restarting. (PID=8836)

08-12-2025 03:47:05 Nagios 4.5.7 starting... (PID=8836)

08-12-2025 03:47:04 Bailing out due to one or more errors encountered in the configuration files. Run Nagios from the command line with the -v option to verify your config before restarting. (PID=8813)

08-12-2025 03:47:04 Nagios 4.5.7 starting... (PID=8813)

08-12-2025 03:47:03 Bailing out due to one or more errors encountered in the configuration files. Run Nagios from the command line with the -v option to verify your config before restarting. (PID=8790)

08-12-2025 03:47:03 Nagios 4.5.7 starting... (PID=8790)

08-12-2025 03:47:02 Bailing out due to one or more errors encountered in the configuration files. Run Nagios from the command line with the -v option to verify your config before restarting. (PID=8767)

08-12-2025 03:47:02 Nagios 4.5.7 starting... (PID=8767)

08-12-2025 03:47:01 Bailing out due to one or more errors encountered in the configuration files. Run Nagios from the command line with the -v option to verify your config before restarting. (PID=8744)

08-12-2025 03:47:00 Nagios 4.5.7 starting... (PID=8744)

08-12-2025 03:46:59 Bailing out due to one or more errors encountered in the configuration files. Run Nagios from the command line with the -v option to verify your config before restarting. (PID=8721)

08-12-2025 03:46:59 Nagios 4.5.7 starting... (PID=8721)

08-12-2025 03:46:58 Bailing out due to one or more errors encountered in the configuration files. Run Nagios from the command line with the -v option to verify your config before restarting. (PID=8698)

08-12-2025 03:46:58 Nagios 4.5.7 starting... (PID=8698)

08-12-2025 03:46:57 Bailing out due to one or more errors encountered in the configuration files. Run Nagios from the command line with the -v option to verify your config before restarting. (PID=8675)

08-12-2025 03:46:57 Nagios 4.5.7 starting... (PID=8675)

08-12-2025 03:46:56 Bailing out due to one or more errors encountered in the configuration files. Run Nagios from the command line with the -v option to verify your config before restarting. (PID=8652)

08-12-2025 03:46:56 Nagios 4.5.7 starting... (PID=8652)

08-12-2025 03:46:55 Bailing out due to one or more errors encountered in the configuration files. Run Nagios from the command line with the -v option to verify your config before restarting. (PID=8629)

08-12-2025 03:46:55 Nagios 4.5.7 starting... (PID=8629)

08-12-2025 03:46:54 Bailing out due to one or more errors encountered in the configuration files. Run Nagios from the command line with the -v option to verify your config before restarting. (PID=8605)

08-12-2025 03:46:54 Nagios 4.5.7 starting... (PID=8605)

Alertas summary

La vista Alerts Summary muestra un resumen estadístico de todas las alertas registradas en un período de tiempo específico.

Básicamente, sirve para que los administradores vean:

Muchas alertas han ocurrido.

Qué servicios u hosts han generado más problemas.

La frecuencia y severidad de los incidentes (por ejemplo, CRÍTICO, ADVERTENCIA, DESCONOCIDO, ABAJO).

Tendencias de fallos para identificar puntos débiles en la infraestructura.

Standard Reports:
Report Type: 25 Most Recent Hard Alerts ▾
[Create Summary Report!](#)

Custom Report Options:
Report Type: Most Recent Alerts ▾
Report Period: Last 7 Days ▾
If Custom Report Period:
Start Date (Inclusive): August ▾ 1 2025
End Date (Inclusive): August ▾ 14 2025

Limit To Hostgroup: ** ALL HOSTGROUPS ** ▾
Limit To Servicegroup: ** ALL SERVICEGROUPS ** ▾
Limit To Host: ** ALL HOSTS ** ▾
Alert Types: Host and Service Alerts ▾
State Types: Hard and Soft States ▾
Host States: All Host States ▾
Service States: All Service States ▾
Max List Items: 25
[Create Summary Report!](#)

Alertas Histogram

Es una vista que muestra las alertas en forma de gráfico de barras, organizadas por intervalos de tiempo.

Sirve para:

- **Visualizar cuándo ocurren más alertas** (por hora, día, semana, etc.).
- Detectar patrones o picos de fallos en determinados momentos.
- Comparar la cantidad de alertas entre distintos periodos.

Comentarios

La sección Comments muestra y gestiona las notas asociadas a hosts o servicios.


Sirve para que los administradores o técnicos:

- Añadan información contextual sobre un problema.
- Coordinar tareas entre turnos de trabajo.
- Dejen un registro escrito que no se pierde aunque el estado del servicio cambie.
- Se pueden añadir, editar o eliminar desde la interfaz web.
- Cada comentario queda asociado a un host o servicio específico.
- Los comentarios pueden ser temporales o permanentes.

All Host and Service Comments
 Last Updated: Thu Aug 14 17:58:19 UTC 2025
 Updated every 90 seconds
 Nagios® Core™ 4.5.7 - www.nagios.org
 Logged in as nagiosadmin


[Host Comments | Service Comments]

Host Comments

 [Add a new host comment](#)

Host Name	Entry Time	Author	Comment	Comment ID	Persistent	Type	Expires	Actions
There are no host comments								

Service Comments

 [Add a new service comment](#)

Host Name	Service	Entry Time	Author	Comment	Comment ID	Persistent	Type	Expires	Actions
There are no service comments									

Downtime

Downtime es una función que permite programar períodos en los que un host o servicio estará fuera de servicio de forma planificada, para que durante ese tiempo:

- No se generen alertas.
- No se envíen notificaciones a los contactos.
- El estado aparece como *Scheduled Downtime* en lugar de un fallo real.

Se usa, por ejemplo, cuando:

- Vas a hacer mantenimiento a un servidor.
- Realizas actualizaciones de software o hardware.
- Sabes que un servicio estará temporalmente inactivo.

Una vez finalizado el período de downtime, Nagios vuelve a monitorear normalmente y enviará alertas si el servicio sigue caído.

All Host and Service Scheduled Downtime
 Last Updated: Thu Aug 14 17:59:58 UTC 2025
 Updated every 90 seconds
 Nagios® Core™ 4.5.7 - www.nagios.org
 Logged in as nagiosadmin


[Host Downtime | Service Downtime]

Scheduled Host Downtime

 [Schedule host downtime](#)

Host Name	Entry Time	Author	Comment	Start Time	End Time	Type	Duration	Downtime ID	Trigger ID	Actions
There are no hosts with scheduled downtime										

Scheduled Service Downtime

 [Schedule service downtime](#)

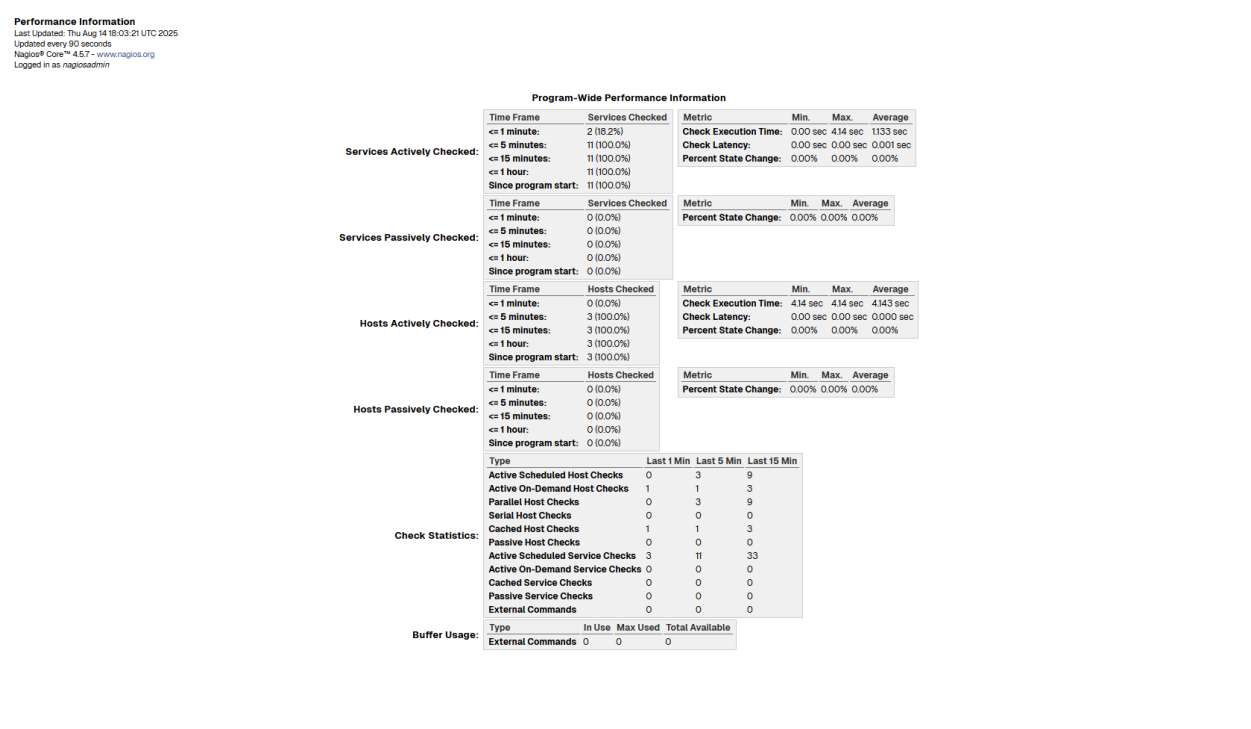
Host Name	Service	Entry Time	Author	Comment	Start Time	End Time	Type	Duration	Downtime ID	Trigger ID	Actions
There are no services with scheduled downtime											

Performance information

La pestaña Performance Info muestra información relacionada con el rendimiento de los hosts y servicios monitoreados.

Su función principal es:

- Registrar métricas de desempeño reportadas por los plugins de Nagios, como tiempos de respuesta, uso de CPU, memoria, espacio en disco, etc.
- Permitir análisis más detallados sobre la eficiencia y carga de los sistemas.
- Servir como base para generar gráficos de tendencias en herramientas externas como PNP4Nagios o Grafana.



Configuracion

Es la sección de la interfaz web donde los administradores pueden ver y validar cómo está definida la infraestructura que Nagios monitorea.

Su función principal es:

- Revisar la configuración cargada: hosts, servicios, contactos, grupos, comandos, plantillas, etc.
- Verificar errores: mostrar advertencias o problemas en los archivos de configuración antes de reiniciar Nagios.
- Validar cambios: asegurarse de que la sintaxis de los archivos **.cfg** es correcta y que Nagios puede cargar la configuración sin fallos.

En otras palabras, no se usa para cambiar directamente la configuración, sino para auditar, inspeccionar y validar la configuración activa del sistema.

Configuration

Last Updated: Thu Aug 14 18:01:24 UTC 2025

Nagios® Core™ 4.5.7 - www.nagios.org

Logged in as nagiosadmin

Object Type:

Hosts

Show Only:

Update

Hosts																								
Host Name	Alias/Description	Address	Importance (Host)	Importance (Host + Services)	Parent Hosts	Max. Check Attempts	Check Interval	Retry Interval	Host Check Command	Check Period	Obsess Over	Enable Active Checks	Enable Passive Checks	Check Freshness	Freshness Threshold	Default Contacts/Groups	Notification Interval	First Notification Delay	Notification Options	Notification Period	Event Handler	Event Handler Period	Enable Event Handler	Stalk Option
apache1_host	Apache Server 1	apache1_host	0	0		5	0h 5m 0s	0h 1m 0s	check-host-alive	24x7	Yes	Yes	Yes	No	Auto-determined value	admins	0h 30m 0s	0h 0m 0s	Down, Unreachable, Recovery, Flapping, Downtime	24x7			Yes	None
apache2_host	Apache Server 2	apache2_host	0	0		5	0h 5m 0s	0h 1m 0s	check-host-alive	24x7	Yes	Yes	Yes	No	Auto-determined value	admins	0h 30m 0s	0h 0m 0s	Down, Unreachable, Recovery, Flapping, Downtime	24x7			Yes	None
localhost	localhost	1270.0.1	0	0		10	0h 5m 0s	0h 1m 0s	check-host-alive	24x7	Yes	Yes	Yes	No	Auto-determined value	admins	2h 0m 0s	0h 0m 0s	Down, Unreachable, Recovery	workhours			Yes	None

