



Monitorización empresarial en entorno sanitario con Nagios Core

José Manuel Méndez Torres

Grado Ingeniería Informática

Administración de redes y sistemas operativos

Mario Prieto Vega

David Bañeres Besora

Montse Bizern Serra

Última entrega 16 de junio, 2024



Esta obra está sujeta a una licencia de
Reconocimiento-NoComercial-
SinObraDerivada [3.0 España de Creative
Commons](https://creativecommons.org/licenses/by-nc-nd/3.0/es/)

B) GNU Free Documentation License (GNU FDL)

Copyright © 2024 Jose Manuel Méndez Torres.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.

A copy of the license is included in the section entitled "GNU Free Documentation License".

C) Copyright

© (el autor/a)

Reservados todos los derechos. Está prohibido la reproducción total o parcial de esta obra por cualquier medio o procedimiento, comprendidos la impresión, la reprografía, el microfilme, el tratamiento informático o cualquier otro sistema, así como la distribución de ejemplares mediante alquiler y préstamo, sin la autorización escrita del autor o de los límites que autorice la Ley de Propiedad Intelectual.

FICHA DEL TRABAJO FINAL

Título del trabajo:	<i>Monitorización empresarial en entorno sanitario con Nagios Core.</i>
Nombre del autor:	<i>José Manuel Méndez Torres.</i>
Nombre del consultor/a:	<i>Mario Prieto Vega.</i>
Nombre del PRA:	<i>David Bañeres Besora y Montse Bizern Serra</i>
Fecha de entrega (mm/aaaa):	<i>06/2024.</i>
Titulación:	<i>Grado Ingeniería Informática.</i>
Área del Trabajo Final:	<i>Administración de redes y sistemas operativos.</i>
Idioma del trabajo:	<i>Castellano.</i>
Palabras clave	<i>Nagios Core, Monitorización</i>
<p>Resumen del Trabajo (máximo 250 palabras): <i>Con la finalidad, contexto de aplicación, metodología, resultados i conclusiones del trabajo.</i></p>	
<p><i>El proyecto tiene como finalidad mejorar la monitorización para permitir respuestas inmediatas a diferentes tipos de incidencias. Se propone configurar un servidor Ubuntu con Nagios Core, un software de monitorización.</i></p> <p><i>El contexto es un consorcio sanitario con múltiples centros de atención médica. Se pretende tener un servidor operativo al finalizar el proyecto, con diversos clientes funcionales a escoger para integrar nuevos monitoreos posteriormente.</i></p> <p><i>Se llevará a cabo la instalación y configuración paso a paso del servidor Ubuntu y Nagios Core, seguido de la exploración y explicación detallada de sus características y funcionalidades. Se realizarán pruebas para garantizar el correcto funcionamiento de los agentes y la monitorización de los equipos. Además, se llevará a cabo una investigación para abordar los objetivos parciales relacionados la seguridad.</i></p> <p><i>Los resultados del proyecto serán la instalación del servidor Ubuntu con Nagios Core, la configuración segura del software de monitorización y la implementación de diversos clientes para la monitorización de equipos. Se proporcionará una documentación detallada que incluirá explicaciones sobre el funcionamiento de Nagios Core, protocolos de monitorización, configuración de plantillas, agentes seguros y procedimientos de respaldo.</i></p> <p><i>Las conclusiones destacarán la importancia de contar con una monitorización de calidad en entornos como el consorcio sanitario descrito. Se resaltarán los beneficios de utilizar Nagios Core para este propósito y se discutirán posibles áreas de mejora o investigación futura.</i></p>	

Abstract (in English, 250 words or less):

The purpose of the project is to enhance monitoring to allow for immediate responses to different types of incidents. It is proposed to set up an Ubuntu server with Nagios Core, a monitoring software. The context is a healthcare consortium with multiple medical centers. The aim is to have an operational server by the end of the project, with various functional clients to choose from for integrating new monitoring subsequently.

The installation and configuration of the Ubuntu server and Nagios Core will be carried out step by step, followed by the exploration and detailed explanation of their features and functionalities. Tests will be conducted to ensure the proper functioning of the agents and equipment monitoring. Additionally, research will be conducted to address partial objectives related to security.

The project outcomes will include the installation of the Ubuntu server with Nagios Core, secure configuration of the monitoring software, and implementation of various clients for equipment monitoring. Detailed documentation will be provided, including explanations of Nagios Core operation, monitoring protocols, template configuration, secure agents, and backup procedures.

The conclusions will emphasize the importance of having quality monitoring in environments such as the described healthcare consortium. The benefits of using Nagios Core for this purpose will be highlighted, and potential areas for improvement or future research will be discussed.

Índice

1. Introducción.....	1
1.1 Contexto y justificación del Trabajo.....	1
1.2 Objetivos del Trabajo	1
1.3 Enfoque y método seguido.....	2
1.4 Planificación del Trabajo	2
1.5 Breve resumen de productos obtenidos.....	3
1.6 Breve descripción de los otros capítulos de la memoria.....	5
2. Herramientas de monitorización:	6
2.1 Estudio del producto Nagios Core.....	6
2.2 Alternativas en el mercado	7
3. Instalación del Nagios Core y configuración:	9
3.1 Instalación de recursos necesarios para Nagios Core	9
3.2 Instalación de Nagios Core.....	10
3.3 Configuración inicial de Nagios Core	13
3.4 Certificación del sitio web de Nagios Core	19
3.5 Limitación de acceso web a determinados clientes al portal Web de Nagios Core	25
3.6 Uso de Postfix para el envío de alertas en Nagios Core.....	26
4. Investigación y pruebas:.....	30
4.1 Investigación y explicación de protocolos de monitorización utilizados por Nagios Core	30
4.2 Investigación de software para clientes y explicar cuáles son más seguros - Nagios Core	35
4.3 Investigar y explicar el funcionamiento de plantillas: contactos, servicios, comandos, etc.	38
4.4 Configuración cliente para Nagios Core en entornos Windows.	41
4.5 Configuración clientes para Nagios Core en entornos UNIX.....	49
4.6 Monitorización de servicios que no requieran de clientes.	56
4.7 Monitorización con SNMP en Switch	61
4.8 Investigar monitorear CRC Switch e implementar	65
5. Configuración de Nagios Core en entorno sanitario:	68
5.1 Crear plantillas pertinentes.	68
5.2 Configuración de clientes.	68
5.3 Configurar copias de seguridad de los ficheros de configuración. ...	81
6. Conclusiones.....	85
7. Glosario.....	86
8. Bibliografía.....	87
9. Anexos.....	91
9.1 Instalación de Ubuntu Server.....	91
9.2 Configuración de Ubuntu Server.....	93
9.3 Instalación de un servidor de correo electrónico en Ubuntu Server	95
9.4 Instalación de clientes para Nagios Core en entornos Windows.	97
9.5 Instalación de clientes para Nagios Core en entornos UNIX.....	104

Lista de figuras

- Figura 1.** Esquema de la arquitectura de Nagios Core. Fuente: [enlace](#)
- Figura 2.** Make install daemoninit Nagios Core. Fuente: imagen propia.
- Figura 3.** Make install commoandmode Nagios Core. Fuente: imagen propia.
- Figura 4.** Acceso web Nagios Core. Fuente: imagen propia.
- Figura 5.** Estado Nagios Core. Fuente: imagen propia.
- Figura 6.** Servicios Nagios Core. Fuente: imagen propia.
- Figura 7.** Versión Nagios Core. Fuente: imagen propia.
- Figura 8.** Instalación y descarga MIBS. Fuente: imagen propia.
- Figura 9.** Instalación NRPE. Fuente: imagen propia.
- Figura 10.** Apertura puertos NRPE. Fuente: imagen propia.
- Figura 11.** Habilitación NRPE. Fuente: imagen propia.
- Figura 12.** Permisos uso NRPE. Fuente: imagen propia.
- Figura 13.** Estado servicio NRPE. Fuente: imagen propia.
- Figura 14.** Copia de ficheros NSCA. Fuente: imagen propia.
- Figura 15.** Asignación de permisos NSCA. Fuente: imagen propia.
- Figura 16.** Añadir IP servidor de monitorización NSCA. Fuente: imagen propia.
- Figura 17.** Ejecución de comando. Fuente: imagen propia.
- Figura 18.** Apertura de puertos NSCA. Fuente: imagen propia.
- Figura 19.** Verificación apertura de puertos del servicio NSCA. Fuente: imagen propia.
- Figura 20.** Adición del servicio NSCA. Fuente: imagen propia.
- Figura 21.** Copia del fichero nsca.xinetd. Fuente: imagen propia.
- Figura 22.** Adición de clientes para la escucha NSCA. Fuente: imagen propia.
- Figura 23.** Instalación NCPA. Fuente: imagen propia.
- Figura 24.** Creación enlace simbólico Python NCPA. Fuente: imagen propia.
- Figura 25.** Comandos necesarios para completar instalación NRPD. Fuente: imagen propia.
- Figura 26.** Adición tokens NRPD. Fuente: imagen propia.
- Figura 27.** Copia del sitio web de NRPD a Apache. Fuente: imagen propia.
- Figura 28.** Acceso web NRDP. Fuente: imagen propia.
- Figura 29.** Creación registro DNS. Fuente: imagen propia.
- Figura 30.** Verificación mediante ICMP resolución del registro DNS. Fuente: imagen propia.
- Figura 31.** Servidor web sin certificado. Fuente: imagen propia.

Figura 32. Creación plantilla para generar CSR. Fuente: imagen propia.

Figura 33. Generación del CSR. Fuente: imagen propia.

Figura 34. Impresión por pantalla del CSR. Fuente: imagen propia.

Figura 35. Proceso de solicitud de certificado. Fuente: imagen propia.

Figura 36. Generación del certificado. Fuente: imagen propia.

Figura 37. Subida de ficheros mediante WinSCP. Fuente: imagen propia.

Figura 38. Copia y permisos del certificado y clave privada. Fuente: imagen propia.

Figura 39. Habilitación de SSL en Apache2 y reinicio del servicio. Fuente: imagen propia.

Figura 40. Apuntar al certificado y clave privada en Apache. Fuente: imagen propia.

Figura 41. Redirección de HTTP a HTTPS. Fuente: imagen propia.

Figura 42. Habilitación escucha HTTPS. Fuente: imagen propia.

Figura 43. Reinicio Apache2. Fuente: imagen propia.

Figura 44. Apertura de puertos HTTPS. Fuente: imagen propia.

Figura 45. Verificación instalación del certificado. Fuente: imagen propia.

Figura 46. Autorización acceso web determinados clientes. Fuente: imagen propia.

Figura 47. Demostración acceso clientes autorizados. Fuente: imagen propia.

Figura 48. Definición de equipo. Fuente: imagen propia.

Figura 49. Activación de plantilla. Fuente: imagen propia.

Figura 50. Definición de contactos. Fuente: imagen propia.

Figura 51. Definición de intervalo. Fuente: imagen propia.

Figura 52. Deshabilitación de interfaz de red. Fuente: imagen propia.

Figura 53. Verificación del servicio. Fuente: imagen propia.

Figura 54. Verificación de alerta por correo. Fuente: imagen propia.

Figura 55. Verificación de alerta por correo. Fuente: imagen propia.

Figura 56. Configuración de Postfix. Fuente: imagen propia.

Figura 57. Verificación de alerta por correo. Fuente: imagen propia.

Figura 58. Diagrama básico de MIBS. Fuente: [enlace](#).

Figura 59. Diagrama árbol MIBS. Fuente: [enlace](#).

Figura 60. Imagen funcionamiento NRPE. Fuente: [enlace](#).

Figura 61. Imagen funcionamiento NSCA. Fuente: [enlace](#).

Figura 62. Imagen funcionamiento Check_NT. Fuente: [enlace](#).

Figura 63. Imagen funcionamiento NRPE. Fuente: [enlace](#).

Figura 64. Imagen funcionamiento NSCA. Fuente: [enlace](#).

Figura 65. Compatibilidad NCPA. Fuente: [enlace](#).

Figura 66. Definición plantillas. Fuente: imagen propia.

Figura 67. Definición servicio a monitorear. Fuente: imagen propia.

Figura 68. Definición contactos. Fuente: imagen propia.

Figura 69. Definición grupo de contactos. Fuente: imagen propia.

Figura 70. Definición ruta de la plantilla de Windows. Fuente: imagen propia.

Figura 71. Definición de cliente. Fuente: imagen propia.

Figura 72. Definición de cliente parental. Fuente: imagen propia.

Figura 73. Enlace parental de clientes. Fuente: imagen propia.

Figura 74. Definición grupo Windows Servers. Fuente: imagen propia.

Figura 75. Definición de clientes. Fuente: imagen propia.

Figura 76. Definición de servicio grupal. Fuente: imagen propia.

Figura 77. Configuración NSClient++. Fuente: imagen propia.

Figura 78. Definición de cliente de monitorización. Fuente: imagen propia.

Figura 79. Definición de servicio de versión. Fuente: imagen propia.

Figura 80. Definición de servicio de tiempo de encendido. Fuente: imagen propia.

Figura 81. Definición de servicio de carga de CPU. Fuente: imagen propia.

Figura 82. Definición de servicio de uso de memoria RAM. Fuente: imagen propia.

Figura 83. Definición de servicio de espacio de disco C: imagen propia.

Figura 84. Reinicio de Nagios Core. Fuente: imagen propia.

Figura 85. Visualización de servicios Nagios Core. Fuente: imagen propia.

Figura 86. Verificación comunicación con Wireshark. Fuente: imagen propia.

Figura 87. Comprobación NRPE de cliente. Fuente: imagen propia.

Figura 88. Verificación comunicación con Wireshark. Fuente: imagen propia.

Figura 89. Definición de servicios monitoreo. Fuente: imagen propia.

Figura 90. Visualización de servicios monitoreados Nagios. Fuente: imagen propia.

Figura 91. Alias monitorización cliente NSCA. Fuente: imagen propia.

Figura 92. Definición servicio y checks pasivos. Fuente: imagen propia.

Figura 93. Definición alias servicio NSCA. Fuente: imagen propia.

Figura 94. Definición servicio y checks pasivos. Fuente: imagen propia.

Figura 95. Definición servicio y checks pasivos. Fuente: imagen propia.

Figura 96. Definición servicio y checks pasivos. Fuente: imagen propia.

Figura 97. Definición servicio y checks pasivos. Fuente: imagen propia.

Figura 98. Definición servicio y checks pasivos. Fuente: imagen propia.

Figura 99. Autorización IP cliente NSCA. Fuente: imagen propia.

Figura 100. Visualización servicios monitoreados Nagios. Fuente: imagen propia.

Figura 101. Definición cliente Nagios Core. Fuente: imagen propia.

Figura 102. Check del disco NCPA. Fuente: imagen propia.

Figura 103. Check de la CPU NCPA. Fuente: imagen propia.

Figura 104. Check de la memoria NCPA. Fuente: imagen propia.

Figura 105. Check total procesos NCPA. Fuente: imagen propia.

Figura 106. Estado servicios Nagios Core. Fuente: imagen propia.

Figura 107. Estado servicios Nagios Core. Fuente: imagen propia.

Figura 108. Definición check pasivo cliente NRPD. Fuente: imagen propia.

Figura 109. Visualización servicios monitoreados NRPD. Fuente: imagen propia.

Figura 110. Definición intervalo NRPD. Fuente: imagen propia.

Figura 111. Verificación servicios. Fuente: imagen propia.

Figura 112. Verificación servicios. Fuente: imagen propia.

Figura 113. Verificación NRPE. Fuente: imagen propia.

Figura 114. Unidades almacenamiento cliente. Fuente: imagen propia.

Figura 115. Descomentar línea comando check disck. Fuente: imagen propia.

Figura 116. Definición host y servicio. Fuente: imagen propia.

Figura 117. Definición comando NRPE. Fuente: imagen propia.

Figura 118. Verificación monitorización. Fuente: imagen propia.

Figura 119. Definición cliente. Fuente: imagen propia.

Figura 120. Definición de servicio. Fuente: imagen propia.

Figura 121. Verificación de servicio. Fuente: imagen propia.

Figura 122. Ejecución comando. Fuente: imagen propia.

Figura 123. Verificación de servicio. Fuente: imagen propia.

Figura 124. Script creado para monitorear. Fuente: imagen propia.

Figura 125. Verificación de servicio Warning. Fuente: imagen propia.

Figura 126. Script creado para monitorear. Fuente: imagen propia.

Figura 127. Verificación de servicio Critical. Fuente: imagen propia.

Figura 128. Creación ejecutable .sh Script. Fuente: imagen propia.

Figura 129. Permisos ejecutable .sh. Fuente: imagen propia.

Figura 130. Crontab para ejecutable. Fuente: imagen propia.

Figura 131. Verificación funcionamiento. Fuente: imagen propia.

Figura 132. Definición de host. Fuente: imagen propia.

Figura 133. Listado para monitorear NCPA. Fuente: imagen propia.

Figura 134. Definición de servicio NCPA. Fuente: imagen propia.

Figura 135. Definición de servicio NCPA. Fuente: imagen propia.

Figura 136. Verificación servicios NCPA. Fuente: imagen propia.

Figura 137. Servicios para check pasivos NRPD Fuente: imagen propia.

Figura 138. Definición servicio pasivo NRPD. Fuente: imagen propia.

Figura 139. Verificación estado servicio NRPD. Fuente: imagen propia.

Figura 140. Verificación estado servicio NRPD. Fuente: imagen propia.

Figura 141. Definición comando FTP. Fuente: imagen propia.

Figura 142. Uso comando FTP. Fuente: imagen propia.

Figura 143. Definición comando SNMP. Fuente: imagen propia.

Figura 144. Uso comando SNMP. Fuente: imagen propia.

Figura 145. Definición comando HTTP. Fuente: imagen propia.

Figura 146. Uso comando HTTP. Fuente: imagen propia.

Figura 147. Definición comando SSH. Fuente: imagen propia.

Figura 148. Uso comando SSH. Fuente: imagen propia.

Figura 149. Definición comando PING. Fuente: imagen propia.

Figura 150. Uso comando PING. Fuente: imagen propia.

Figura 151. Definición comandos POP y IMAP. Fuente: imagen propia.

Figura 152. Uso comandos POP y IMAP. Fuente: imagen propia.

Figura 153. Definición comando SMTP. Fuente: imagen propia.

Figura 154. Uso comando SMTP. Fuente: imagen propia.

Figura 155. Definición comandos TCP y UDP. Fuente: imagen propia.

Figura 156. Uso comandos TCP y UDP. Fuente: imagen propia.

Figura 157. Uso de MIB Browser. Fuente: imagen propia.

Figura 158. Monitorización SNMP. Fuente: imagen propia.

Figura 159. Uso de MIB Browser para obtener OID. Fuente: imagen propia.

Figura 160. Monitorización interface SNMP. Fuente: imagen propia.

Figura 161. Monitorización interface SNMP. Fuente: imagen propia.

Figura 162. Monitorización interface SNMP. Fuente: imagen propia.

Figura 163. Definición comando SNMP V1. Fuente: imagen propia.

Figura 164. Definición comando SNMP V2. Fuente: imagen propia.

Figura 165. Visualización servicios Nagios Core. Fuente: imagen propia.

Figura 166. Visualización servicios Nagios Core. Fuente: imagen propia.

Figura 167. Visualización CRC interfaz Swtich. Fuente: imagen propia.

Figura 168. Identificación OID mediante MIB Browser. Fuente: imagen propia.

Figura 169. IRTable MIB Browser. Fuente: imagen propia.

Figura 170. Búsqueda CRC MIB Browser. Fuente: imagen propia.

Figura 171. Búsqueda CRC MIB Browser. Fuente: imagen propia.

Figura 172. Búsqueda CRC MIB Browser. Fuente: imagen propia.

Figura 173. Búsqueda CRC MIB Browser. Fuente: imagen propia.

Figura 174. Búsqueda CRC MIB Browser, Fuente: imagen propia.

Figura 175. Búsqueda finalizada MIB Browser. Fuente: imagen propia.

Figura 176. Plantillas Nagios Core. Fuente: imagen propia.

Figura 177. Plantilla Linux Servers. Fuente: imagen propia.

Figura 178. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 179. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 180. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 181. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 182. Visualización equipos plantilla Routers. Fuente: imagen propia.

Figura 183. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 184. Visualización equipos plantilla DC_Primary. Fuente: imagen propia.

Figura 185. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 186. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 187. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 188. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 189. Summary Routers Primary. Fuente: imagen propia.

Figura 190. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 191. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 192. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 193. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 194. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 195. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 196. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 197. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 198. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 199. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 200. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 201. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 202. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 203. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 204. Summary Host Switch Primary. Fuente: imagen propia.

Figura 205. Servicios cliente en Nagios Core. Fuente: imagen propia.

Figura 206. Summary Host Switch. Fuente: imagen propia.

Figura 207. Carpeta para recurso. Fuente: imagen propia.

Figura 208. Creación de recurso. Fuente: imagen propia.

Figura 209. Permisos acceso recurso. Fuente: imagen propia.

Figura 210. Instalación NFS. Fuente: imagen propia.

Figura 211. Creación de carpeta para mapeo de recurso. Fuente: imagen propia.

Figura 212. Montaje recurso. Fuente: imagen propia.

Figura 213. Edición FSTAB. Fuente: imagen propia.

Figura 214. Ejecución comando “mount -a”. Fuente: imagen propia.

Figura 215. Script para el backup. Fuente: imagen propia.

Figura 216. Modificación Crontab. Fuente: imagen propia.

Figura 217. Máquina virtual creada. Fuente: imagen propia.

Figura 218. Arranque imagen ISO. Fuente: imagen propia.

Figura 219. Configuración interfaz de red. Fuente: imagen propia.

Figura 220. Modificación parámetros de red. Fuente: imagen propia.

Figura 221. Modificación nombre de host y usuario. Fuente: imagen propia.

Figura 222. Instalación de openSSH. Fuente: imagen propia.

Figura 223. Permisos Root SSH. Fuente: imagen propia.

Figura 224. Permisos Root SSH. Fuente: imagen propia.

Figura 225. Permisos Root SSH. Fuente: imagen propia.

Figura 226. Permisos Root SSH. Fuente: imagen propia.

Figura 227. Reinicio del servicio SSH. Fuente: imagen propia.

Figura 228. Cambio contraseña Root. Fuente: imagen propia

Figura 229. Cambio zona horaria. Fuente: imagen propia.

Figura 230. Instalación Postfix. Fuente: imagen propia.

Figura 231. Configuración nombre host Postfix. Fuente: imagen propia.

Figura 232. Reinicio Postfix. Fuente: imagen propia.

Figura 233. Instalación mailutils. Fuente: imagen propia.

Figura 234. Envío correo de prueba. Fuente: imagen propia.

Figura 235. Recepción de correo de prueba. Fuente: imagen propia.

Figura 236. Fuente: imagen propia.

Figura 237. Instalación NSClient++. Fuente: imagen propia.

Figura 238. Variable USER1. Fuente: imagen propia.

Figura 239. Generación CSR. Fuente: imagen propia.

Figura 240. Descarga del certificado emitido por la CA. Fuente: imagen propia.

Figura 241. Uso certificado NSClient++. Fuente: imagen propia.

Figura 242. Descarga del certificado. Fuente: imagen propia.

Figura 243. Subida del certificado. Fuente: imagen propia.

Figura 244. Edición parámetros SSL. Fuente: imagen propia.

Figura 245. Edición parámetros SSL. Fuente: imagen propia.

Figura 246. Verificación del servicio NRPE. Fuente: imagen propia.

Figura 247. Definición del comando NRPE. Fuente: imagen propia.

Figura 248. Instalación NCPA. Fuente: imagen propia.

Figura 249. Configuración NRPD. Fuente: imagen propia.

Figura 250. Fase instalación NCPA. Fuente: imagen propia.

Figura 251. Instalación a nivel de equipo NCPA. Fuente: imagen propia.

Figura 252. Definición comando NCPA. Fuente: imagen propia.

Figura 253. Servicios NRPD. Fuente: imagen propia.

Figura 254. Instalación NRPE. Fuente: imagen propia.

Figura 255. Permisos uso NRPE Nagios Core. Fuente: imagen propia.

Figura 256. Copia de ficheros certificación para NRPE. Fuente: imagen propia.

Figura 257. Instalación certificado CA. Fuente: imagen propia.

Figura 258. Definición parámetros SSL. Fuente: imagen propia.

Figura 259. Reinicio servicio NRPE. Fuente: imagen propia.

Figura 260. Definición comando NRPE. Fuente: imagen propia.

Figura 261. Permisos uso NSCA. Fuente: imagen propia.

Figura 262. Definición parámetros NSCA. Fuente: imagen propia.

Figura 263. Arranque y verificación del servicio NSCA. Fuente: imagen propia.

Figura 264. Definición token NSCA. Fuente: imagen propia.

Figura 265. Autorización uso NSCA Nagios Core. Fuente: imagen propia.

Figura 266. Definición parámetros NCPA. Fuente: imagen propia.

Figura 267. Reinicio servicio NCPA. Fuente: imagen propia.

Figura 268. Definición parámetros NRDP. Fuente: imagen propia.

Figura 269. Definición parámetros NRDP. Fuente: imagen propia.

Figura 270. Creación fichero NRDP. Fuente: imagen propia.

1. Introducción

1.1 Contexto y justificación del Trabajo

La motivación de este proyecto surge de la necesidad de aumentar la calidad de la monitorización para de este modo poder actuar de forma inmediata ante cualquier incidencia de gravedad que así lo requiera. Es vital para cualquier entorno empresarial disponer de un sistema que alerte si algún servicio o equipo presenta incidencias.

Actualmente se dispone de diversos softwares de monitorización obsoletos, desactualizados y con simples características.

Se pretende configurar un servidor Ubuntu donde se le instalará un software de monitorización llamado Nagios Core. Se pretende explicar el funcionamiento, configurarlo de la forma más segura posible, explicar que clientes se pueden encontrar, como funcionan sus plantillas, que protocolos se utilizan para comunicarse con los clientes, monitorear equipos, explicar que se puede llegar a monitorear, establecer alertas, así como explicarlas, configurar copias de seguridad de las plantillas, etc.

Se trata de un desarrollo real para mi entorno laboral. Trabajo en un consorcio sanitario con un gran número de centros de atención primaria, dos hospitales, dos centros de servicios de rehabilitación comunitaria, un centro de radioterapia y centros de salud mental. Pretendo que al finalizar de este proyecto este el servidor en funcionamiento con los diferentes tipos de clientes operativos para poder ir integrando clientes que requieran ser monitorizados y que no puedan dar tiempo a integrarlos durante el transcurso de este proyecto.

1.2 Objetivos del Trabajo

Los objetivos de este proyecto consisten en explicar el funcionamiento de Nagios Core y los protocolos utilizados para monitorear, así como diferentes clientes utilizados y su funcionamiento. Se pretende lograr:

- Instalación del sistema operativo Ubuntu Server 22.04.
- Instalación de Nagios Core y actualización a su versión más reciente.
- Certificar el sitio web de Nagios Core con la entidad certificadora de un dominio.
- Limitar el acceso al sitio web de Nagios Core mediante equipos autorizados.
- Explicar los diferentes protocolos de monitorización: SNMP, NCPA, NRPE, etc.
- Explicar el funcionamiento de plantillas: contactos, servicios, comandos, etc.

- Explicar que agentes son más seguros (cuales tienen una comunicación encriptada, cuales usan contraseñas para la autenticación, etc.).
- Instalar agentes y probarlos. Explicando algunos servicios que se puedan monitorear y cómo funciona.
- Monitorear equipos en la red con SNMP (no requiere de ningún agente).
- Configurar copias de seguridad de los ficheros de configuración.

Como objetivos parciales que requieren de una mayor investigación y que no tengo claro si podré conseguir de los anteriormente descritos son los siguientes:

- Lograr monitorear con SNMP CRC de las interfaces de los Switch. No sé si será posible, requerirá de investigación. Nunca he trabajado con Nagios Core y no sé si mediante SNMP se puede lograr obtener toda la información necesaria.
- Encontrar que cliente es más seguro y configurarlo de la forma más segura para garantizar una comunicación encriptada y bajo autenticación cliente-servidor.

1.3 Enfoque y método seguido

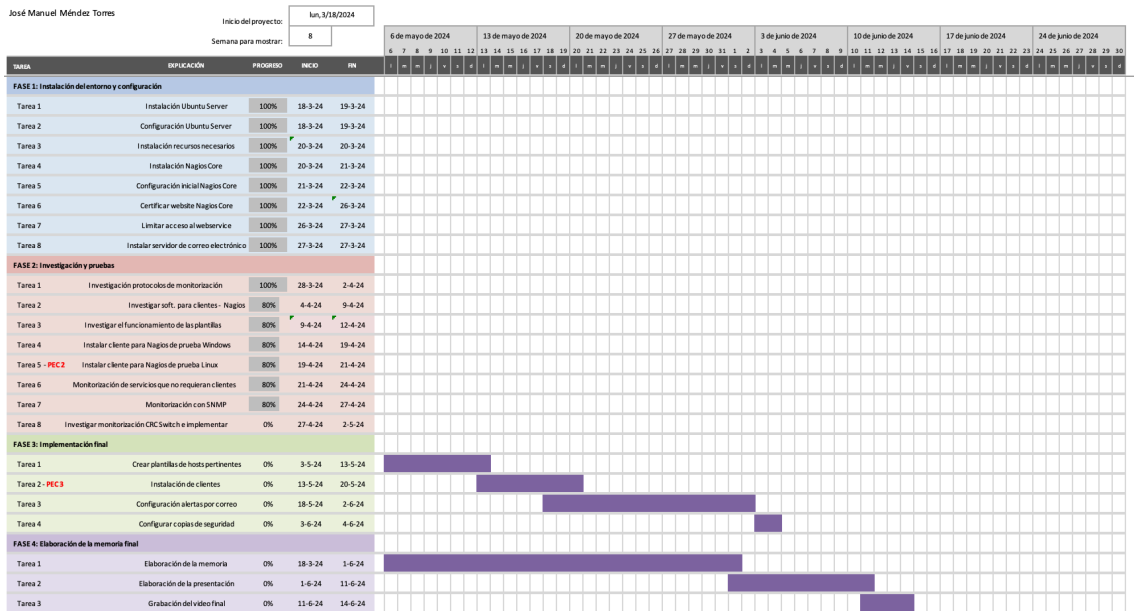
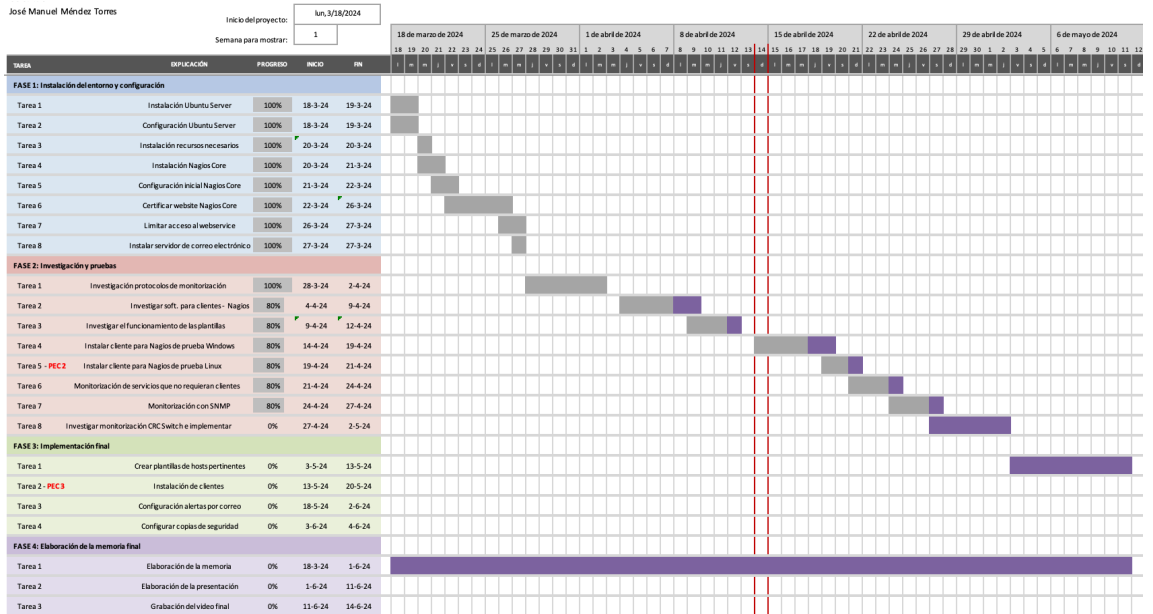
La estrategia de este proyecto se basa en implementar y desarrollar un nuevo producto desde cero en un entorno sanitario para monitorear equipos y servicios. Se deberá aprender cómo funcionan los clientes de monitorización y cómo configurar el servidor de monitorización. Se considera que esta opción es la más adecuada debido a que Nagios Core es una herramienta de monitorización de código abierto, gratuita, flexible y basada en plugins.

1.4 Planificación del Trabajo

Para el desarrollo de este proyecto requerirá de:

- Hacer uso de nuestro vSphere para crear una nueva máquina virtual donde se instalará una ISO de Ubuntu Server y posteriormente el software de monitorización Nagios Core con sus requisitos previos: compilador del lenguaje C gcc, openssl, vim, curl, apache2, php etc.
- Utilizar algunos de los servidores virtualizados y físicos que tenemos para poder monitorizarlos.
- Utilizar algunos de los Switches que tenemos para poder monitorizarlos.
- Utilizar nuestros enrutadores para poder monitorizarlos.
- Utilizar alguna de nuestras cabinas de almacenamiento para efectuar copias de seguridad.

Se presenta el siguiente diagrama con la planificación del proyecto:



1.5 Breve resumen de productos obtenidos

Hardware:

- vSphere: Plataforma de virtualización de VMWare que permite instalar y ejecutar sistemas operativos mediante la creación de máquinas virtuales. En este entorno se instalará el sistema operativo Ubuntu Server mediante el cual ejecutaremos el software de motorización Nagios Core.

Software:

Se va a describir de forma resumida el software principal que será utilizado para la elaboración de este proyecto:

- Ubuntu Server: Sistema operativo que se administra mediante línea de comandos y que proporciona un entorno estable para instalar y ejecutar variedad de servicios y aplicaciones. Se instalará Nagios Core.
- Apache2: Servidor web de código abierto que albergará Nagios Core y uno de los servidores web más populares en uso en la actualidad.
- Nagios Core: Software de monitorización de código abierto y gratuito el cual se encarga de monitorear equipos y servicios de una infraestructura la cual se puede administrar desde el propio equipo y vía interfaz web.
- NSClient++: Software diseñado para ser utilizado como agente de monitorización entornos Windows. Este agente puede ser utilizado para check_nt, NRPE Servidor y NSCA.
- NCPA: Software diseñado para ser utilizado como agente de monitorización en gran diversidad de sistemas operativos.
- NRPD: Servidor NRPD que permite que los clientes envíen resultados de sus chequeos pasivos. De este modo, se evita que Nagios Core efectúe las comprobaciones de forma activa.
- NRPE: Es un protocolo propio de Nagios. Las siglas significan: Nagios Remote Plugin Executor. Este protocolo permite el uso de plugins de Nagios en máquinas remotas con el fin de supervisar los recursos locales de estos. Su uso es más para comprobaciones activas, pero también se permiten comprobaciones pasivas, donde los clientes envían los resultados del servidor Nagios.
- NSCA: Es un protocolo propio de Nagios. Las siglas significan: Nagios Service Check Acceptor. Este protocolo es utilizado para que los propios servidores o clientes envíen la información de sus comprobaciones activas al servidor de monitorización Nagios. Por lo tanto, son comprobaciones pasivas. Para las comprobaciones pasivas se requiere crear un directorio en el servidor monitorizado y en él se ejecutará por medio de crontab los chequeos requeridos.
- MIBS: Archivos de datos que contienen información estructurada y jerárquica utilizada para gestionar dispositivos de red. Estos archivos definen los objetos gestionados que pueden ser supervisados y controlados en un dispositivo de red mediante un protocolo de gestión de red como SNMP.
- Postfix: Servidor de correo que proporciona un sistema de entrega de correo electrónico eficiente y confiable, soportando protocolos

estándar como SMTP (Simple Mail Transfer Protocol). Se instalará en el sistema operativo Ubuntu Server.

1.6 Breve descripción de los otros capítulos de la memoria

Capítulo 2: *Herramientas de monitorización.* Primer análisis de Nagios Core y alternativas en el mercado.

Capítulo 3: *Instalación del entorno y configuración.* Instalación del software de monitorización Nagios Core, configuración inicial del producto, certificación del sitio web, restricción de acceso a clientes autorizados.

Capítulo 4: *Investigación y pruebas.* Investigación, explicación e instalación de los diferentes protocolos y clientes de monitorización para Nagios Core.

Capítulo 5: Implementación final. Instalación de clientes, monitorización de servicios necesarios, creación de plantillas definitivas de equipos, configuración de alertas por correo y configuración de copias de seguridad de plantillas.

Capítulo 6: Conclusiones del proyecto.

Capítulo 7: Glosario del proyecto.

Capítulo 8: Bibliografía del proyecto.

Capítulo 9: Instalación del sistema operativo, servidor de correo electrónico y clientes de monitorización en entornos Windows y Linux para Nagios Core.

2. Herramientas de monitorización:

2.1 Estudio del producto Nagios Core

Nagios Core [2] es una herramienta de monitorización de código abierto y gratuita la cual se encarga de monitorear equipos y servicios de una infraestructura la cual se puede administrar desde el propio equipo y vía interfaz web.

La arquitectura de Nagios es basada en plugins, los cuales son programas independientes que se pueden utilizar para realizar comprobaciones específicas del estado de servicios o dispositivos. Estos plugins pueden ser escritos por los propios usuarios o pueden ser plugins predefinidos que vienen con la instalación de Nagios Core.

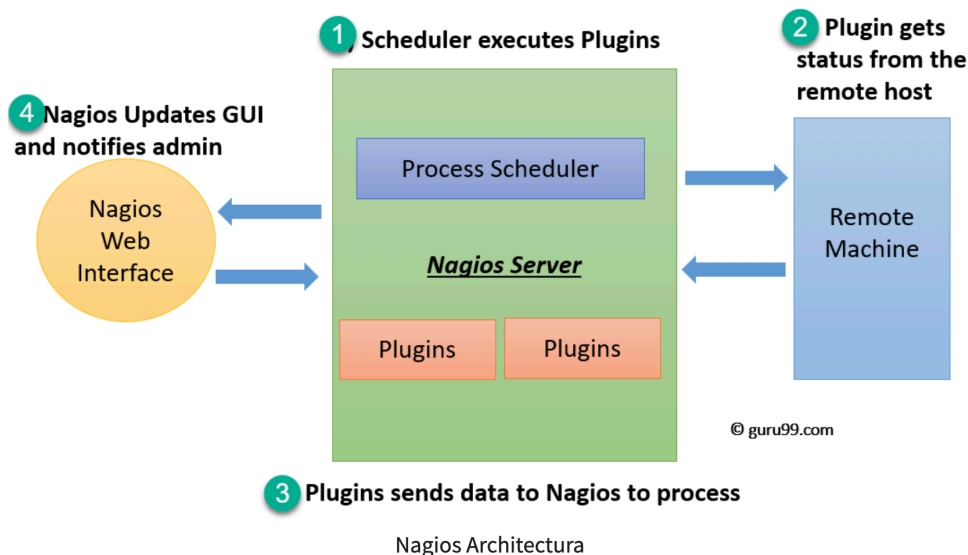


Figura 1

Con Nagios Core se pueden programar alertas y envíos de estas mismas por diferentes medios.

Es una herramienta muy utilizada en entornos empresariales para garantizar la disponibilidad y el rendimiento de los sistemas y servicios críticos.

La principal característica de Nagios Core es su flexibilidad, permite que un administrador utiliza la herramienta de diferentes formas para administrar sus equipos y servicios de su infraestructura.

Otra de las ventajas que ofrece Nagios Core es el sistema de dependencia. Este sistema está basado en niveles. Es decir, si un host depende de otro y este sufre una caída el host del cual depende no será monitoreado.

Por otro lado, se pueden definir periodos de mantenimiento en los hosts monitoreados para que de esta forma se puedan hacer intervenciones y no se notifique de su estado.

2.2 Alternativas en el mercado

Existen gran variedad de alternativas en el mercado de soluciones de monitorización de equipos y servicios, entre ellas podemos encontrar:

- Zabbix [\[3\]](#): Es una solución de software de código abierto diseñada para monitorear y controlar redes y sistemas informáticos.
 - Amplia cobertura de monitorización: tiene la capacidad de supervisar una amplia variedad de dispositivos y sistemas, que abarcan desde servidores hasta conmutadores, enrutadores, aplicaciones y servicios.
 - Interfaz intuitiva: proporciona una interfaz web intuitiva que simplifica la configuración, visualización y análisis de los datos de monitoreo, haciendo que estas tareas sean más accesibles y comprensibles.
 - Flexibilidad para el crecimiento: se adapta fácilmente a entornos empresariales de gran envergadura y puede expandirse según las necesidades específicas de la organización.
 - Configuración de alertas: se permite configurar alertas personalizadas.

Como desventajas podemos encontrar:

- En los clientes se tiene que instalar un agente.
 - La configuración inicial puede llegar a requerir mucho tiempo y formación.
 - Al ser de código abierto no se puede obtener soporte oficial.
-
- Zenoss [\[4\]](#): Es una herramienta adaptable de seguimiento que simplifica la adición de nuevas capacidades y puede supervisar una variedad de dispositivos como Linux, Windows, switches y routers. Además, cuenta con una interfaz web extremadamente intuitiva para su uso. Como ventajas de este producto podemos destacar las siguientes:
 - Amplia cobertura de monitorización: tiene la capacidad de supervisar una amplia variedad de dispositivos y sistemas, que abarcan desde servidores hasta conmutadores, enrutadores, aplicaciones y servicios.
 - Interfaz intuitiva: proporciona una interfaz web intuitiva que simplifica la configuración, visualización y análisis de los datos de monitoreo, haciendo que estas tareas sean más accesibles y comprensibles.
 - Flexibilidad para el crecimiento: se adapta fácilmente a entornos empresariales de gran envergadura y puede

expandirse según las necesidades específicas de la organización.

- Configuración de alertas: se permite configurar alertas personalizadas.
- No requiere instalar agentes en los clientes.
- Al existir versión de pago se puede conseguir soporte oficial ante cualquier incidencia.

Como desventajas podríamos encontrar las siguientes:

- Dificultad en la configuración: su configuración no es sencilla y requiere de formación previa.
- Integración con terceros: debería tener una mayor integración con herramientas de rendimiento de aplicaciones de terceros que facilitarían el uso de la herramienta.

3. Instalación del Nagios Core y configuración:

En este apartado se va a describir como instalar y configurar el entorno para adaptarlo a las necesidades del proyecto. Véase información más detallada sobre la instalación y configuración del sistema operativo en los anexos de la memoria.

3.1 Instalación de recursos necesarios para Nagios Core

Para que el sistema operativo Ubuntu Server pueda hacer funcionar correctamente Nagios Core primero de todo se debe de actualizar los repositorios de software con el comando “sudo apt-get update”.

Posteriormente, deberemos de instalar las siguientes dependencias para que se pueda compilar e instalar Nagios Core [\[5\]](#):

- Autoconf: Herramienta para generar scripts de configuración para programas de software. Nagios Core lo requiere para configurar su compilación en el sistema.
- GCC: El compilador de GNU de C. Es necesario para compilar el código fuente de Nagios Core en binarios ejecutables.
- Libc6: Biblioteca C estándar de GNU. Casi todas las aplicaciones de Linux dependen de esta biblioteca, incluido Nagios Core.
- Make: Utilidad para automatizar el proceso de compilación de programas. Nagios Core utiliza Makefile para compilar su código fuente en binarios ejecutables.
- Wget: Herramienta para descargar archivos desde la web. Nagios Core requiere de esto para descargar paquetes de su sitio web o de repositorios en línea.
- Unzip: Herramienta para descomprimir archivos zip. Algunas distribuciones de Nagios Core se distribuyen en archivos zip que necesitan ser descomprimidos antes de la instalación.
- Apache2: El servidor web Apache. Nagios Core se integra con Apache para proporcionar una interfaz web para su panel de control y visualización de informes.
- PHP: Lenguaje de programación de servidor utilizado para la creación de páginas web dinámicas. Nagios Core utiliza PHP para generar la interfaz web y los informes.
- Libapache2-mod-php7.4: Módulo para Apache que permite la ejecución de scripts PHP en el servidor web Apache.

- Libgd-dev: Biblioteca de gráficos para la creación dinámica de imágenes. Nagios Core utiliza esta biblioteca para generar gráficos y visualizaciones en su interfaz web.
- OpenSSL y Libssl-dev: Se requiere para habilitar la comunicación segura (a través de HTTPS) entre el navegador web y el servidor web donde se ejecutará Nagios Core. OpenSSL es una biblioteca de software ampliamente utilizada para implementar protocolos de cifrado de datos seguros en aplicaciones y servicios de red.

Para su instalación se debe de ejecutar el siguiente comando: “sudo apt-get install -y autoconf gcc libc6 make wget unzip apache2 php libapache2-mod-php7.4 libgd-dev” y “sudo apt-get install openssl libssl-dev”.

3.2 Instalación de Nagios Core

Para la instalación de Nagios Core en Ubuntu Server se procede a seguir la correspondiente documentación: <https://support.nagios.com/kb/article/nagios-core-installing-nagios-core-from-source-96.html#Ubuntu> [5]

Primero de todo, se debe de descargar en un directorio temporal la versión 4.4.13, para ello se debe de hacer uso del siguiente comando: “wget -O nagioscore.tar.gz <https://github.com/NagiosEnterprises/nagioscore/archive/nagios-4.4.14.tar.gz>”.

Una vez descargado, se procede a descomprimir con el comando “tar xzf nagioscore.tar.gz” y a compilarlo con “sudo ./configure --with-httpd-conf=/etc/apache2/sites-enabled” y “sudo make all”.

Una vez compilado, se debe de crear el usuario y el grupo necesarios, y añadir el usuario Nagios al grupo www-data con los siguientes comandos “sudo make install-groups-users” y “sudo usermod -a -G nagios www-data”. Después de ello se debe instalar los binarios con el comando “make install”. Una vez instalados los binarios, se debe de crear un archivo de servicio systemd para gestionar el servicio Nagios.

Se crea el script systemd init con el siguiente comando “make install-daemoninit”:

```
root@monsrv01:/tmp/nagioscore-nagios-4.4.14# sudo make install-daemoninit
/usr/bin/install -c -m 755 -d -o root -g root /lib/systemd/system
/usr/bin/install -c -m 755 -o root -g root startup/default-service /lib/systemd/system/nagios.service
Created symlink /etc/systemd/system/multi-user.target.wants/nagios.service → /lib/systemd/system/nagios.service.

*** Init script installed ***

root@monsrv01:/tmp/nagioscore-nagios-4.4.14#
```

Figura 2

Con el comando “make install-commandmode” establecemos el permiso adecuado en el directorio de instalación de Nagios Core:

```
root@monsrv01:/tmp/nagioscore-nagios-4.4.14# sudo make install-commandmode
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/var/rw
chmod g+s /usr/local/nagios/var/rw

*** External command directory configured ***

root@monsrv01:/tmp/nagioscore-nagios-4.4.14#
```

Figura 3

Posteriormente, se debe de crear un archivo de configuración de Nagios de ejemplo con el siguiente comando “make install-config” e instalar la interfaz web de Nagios con los siguientes comandos “make install-webconf”, “a2enmod rewrite cgi” y “sudo a2enmod cgi”.

Una vez instalado lo anterior, se debe permitir el tráfico entrante por el puerto 80 en el firewall local para poder acceder a la interfaz web de Nagios Core con los comandos “sudo ufw allow Apache”, “sudo ufw reload” y crear la cuenta de nagiosadmin para acceder al sitio web con el comando “sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin”.

Una vez habilitada la interfaz web, se debe de reiniciar el servicio apache con el comando: “systemctl restart apache2” e iniciar los servicios de Nagios con los siguientes comandos “sudo systemctl start nagios.service”. Posteriormente se tendrá disponible el sitio web, en este caso a través de la URL:

<http://172.31.3.74/nagios/>

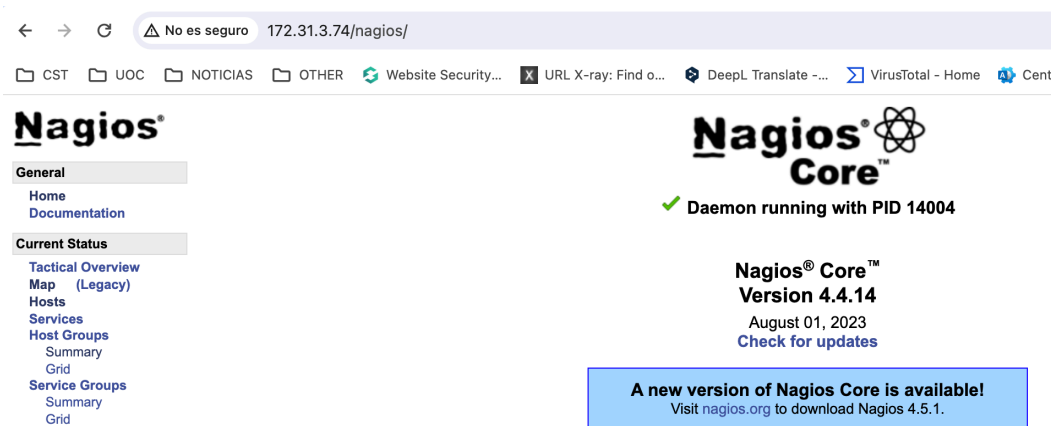


Figura 4

Como se puede apreciar, se indica que existe una nueva versión. Una vez se termine de configurar esta versión se procederá a instalar la nueva versión.

Se puede consultar su estado con el siguiente comando “systemctl status nagios”:

```

root@monsrv01:/tmp/nagioscore-nagios-4.4.14# systemctl status nagios
● nagios.service - Nagios Core 4.4.14
   Loaded: loaded (/lib/systemd/system/nagios.service; enabled; vendor preset: enabled)
   Active: active (running) since Sat 2024-03-09 08:54:49 UTC; 3min 15s ago
     Docs: https://www.nagios.org/documentation
   Process: 14002 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.c
   Process: 14003 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
   Main PID: 14004 (nagios)
    Tasks: 8 (limit: 4557)
   Memory: 6.8M
     CPU: 2.103s
   CGroup: /system.slice/nagios.service
           └─14004 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             └─14005 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
               └─14006 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                 └─14007 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                   └─14008 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                     └─14009 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                       └─14010 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                         └─14011 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

mar 09 08:54:51 monsrv01 nagios[14004]: Successfully launched command file worker with pid 140
mar 09 08:54:51 monsrv01 nagios[14004]: HOST ALERT: localhost;DOWN;SOFT;1;(No output on stdout>
mar 09 08:55:26 monsrv01 nagios[14004]: SERVICE ALERT: localhost;Current Load;CRITICAL;HARD;1;>
mar 09 08:55:51 monsrv01 nagios[14004]: HOST ALERT: localhost;DOWN;SOFT;2;(No output on stdout>
mar 09 08:56:04 monsrv01 nagios[14004]: SERVICE ALERT: localhost;Current Users;CRITICAL;HARD;1>
mar 09 08:56:41 monsrv01 nagios[14004]: SERVICE ALERT: localhost;HTTP;CRITICAL;HARD;1;(No outp>
mar 09 08:56:51 monsrv01 nagios[14004]: HOST ALERT: localhost;DOWN;SOFT;3;(No output on stdout>
mar 09 08:57:19 monsrv01 nagios[14004]: SERVICE ALERT: localhost;PING;CRITICAL;HARD;1;(No outp>
mar 09 08:57:55 monsrv01 nagios[14004]: HOST ALERT: localhost;DOWN;SOFT;4;(No output on stdout>
mar 09 08:57:56 monsrv01 nagios[14004]: SERVICE ALERT: localhost;Root Partition;CRITICAL;HARD;>
lines 1-30/30 (END)

```

Figura 5

Posteriormente, se debe de instalar los complementos para que pueda funcionar correctamente Nagios Core, ejecutamos el siguiente comando para garantizar que tenemos los paquetes necesarios “sudo apt-get install -y autoconf gcc libc6 libmccrypt-dev make libssl-dev wget bc gawk dc build-essential snmp libnet-snmp-perl gettext” accedemos a un directorio temporal y descargamos la fuente con el siguiente comando “wget --no-check-certificate -O nagios-plugins.tar.gz <https://github.com/nagios-plugins/nagios-plugins/archive/release-2.4.6.tar.gz>” y se procede con la descompresión con el comando “tar xzf nagios-plugins.tar.gz”.

Después de tener la descompresión efectuada, procedemos con la compilación e instalación con los siguientes comandos: “cd /tmp/nagios-plugins-release-2.4.6/”, “sudo ./tools/setup”, “sudo ./configure”, “sudo make”, “sudo make install” y si accedemos al sitio web veremos el equipo local monitorizado:

Host	Service	Status
localhost	Current Load	OK
	Current Users	OK
	HTTP	OK
	PING	OK
	Root Partition	OK
	SSH	OK
	Swap Usage	OK
	Total Processes	OK

Figura 6

Una vez tenemos completamente funcional Nagios, se va a actualizar su versión para tener la más reciente, para ello, se debe de descargar desde la página oficial: <https://www.nagios.org/downloads/nagios-core/> [6] Y tendremos como referencia el manual oficial <https://assets.nagios.com/downloads/nagioscore/docs/nagioscore/4/en/upgrading.html> [7].

Una vez descargada la subimos mediante WinSCP y la descomprimos mediante el comando “tar xzf nagios-4.5.1.tar.gz”. Accedemos al directorio para ejecutar el script de configuración de Nagios Core, pasando el nombre del grupo utilizado para controlar los permisos del archivo del comando externo con el siguiente comando “./configure --with-command-group=nagcmd”, posteriormente compilamos con “make all” e instalamos los binarios con el comando “make install”, por último se debe de reiniciar el servicio de nagios core con “/sbin/service nagios restart” y si accedemos al sitio web verificamos que se ha efectuado la actualización correctamente:

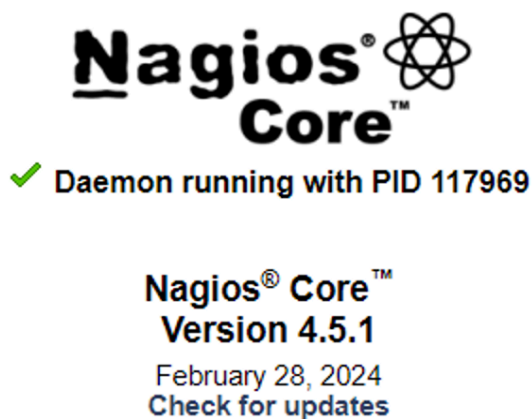


Figura 7

3.3 Configuración inicial de Nagios Core

Una vez tenemos Nagios Core en funcionamiento se debe de configurar con los plugins necesarios para que se puedan monitorear correctamente los clientes. En este apartado se va a explicar cómo se instalan y más adelante se explicará su funcionamiento y uso:

- SNMP [8]: Primero de todo deberemos de instalar el paquete y los MIBS:

```
root@monsrv01:~# apt-get install snmp-mibs-downloader
root@monsrv01:~# download-mibs
```

Figura 8

Por un error de MIBS en Ubuntu Server se debe de editar el archivo “/usr/share/snmp/mibs/ietf/SNMPv2-PDU” ya que contiene errores tipográficos. Por lo tanto, se debe editar el archivo borrando todo su

contenido y añadiendo el que encontramos en el siguiente enlace:
<https://pastebin.com/raw/p3QyuXzZ> [9]

- NRPE: Para instalar NRPE se hará uso del siguiente manual oficial: <https://support.nagios.com/kb/article.php?id=515#Ubuntu> [10], debemos de descargarlo ejecutando el siguiente comando: “wget --no-check-certificate -O nrpe.tar.gz <https://github.com/NagiosEnterprises/nrpe/archive/nrpe-4.1.0.tar.gz>”, posteriormente se descomprime: “tar xzf nrpe.tar.gz” y lo compilamos accediendo a su directorio “/tmp/nrpe-nrpe-4.1.0/” y ejecutando el comando: “sudo ./configure --enable-command-args --with-ssl-lib=/usr/lib/x86_64-linux-gnu/”, posteriormente, ejecutamos “sudo make all” y después, se debe ejecutar “sudo make install” para instalar los binarios.

Una vez instalados, instalamos los ficheros de configuración mediante el siguiente comando: “sudo make install-config”:

```
root@monsrv01:/tmp/nrpe-nrpe-4.1.0# sudo make install-config
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/etc
/usr/bin/install -c -m 644 -o nagios -g nagios sample-config/nrpe.cfg /usr/local/nagios/etc
root@monsrv01:/tmp/nrpe-nrpe-4.1.0#
```

Figura 9

Ahora, editamos el fichero de los servicios para añadir el servicio y el puerto que utiliza:

```
root@monsrv01:/tmp/nrpe-nrpe-4.1.0# sudo sh -c "echo >> /etc/services"
sudo sh -c "sudo echo '# Nagios services' >> /etc/services"
sudo sh -c "sudo echo 'nrpe 5666/tcp' >> /etc/services"
root@monsrv01:/tmp/nrpe-nrpe-4.1.0#
```

Figura 10

Posteriormente, se instala el demonio con el comandos: “sudo make install-init” y habilitamos el servicio de nrpe: “sudo systemctl enable nrpe.service” y, se abren los puertos utilizados en el firewall local:

```
root@monsrv01:/tmp/nrpe-nrpe-4.1.0# sudo mkdir -p /etc/ufw/applications.d
sudo sh -c "echo '[NRPE]' > /etc/ufw/applications.d/nagios"
sudo sh -c "echo 'title=Nagios Remote Plugin Executor' >> /etc/ufw/applications.d/nagios"
sudo sh -c "echo 'description=Allows remote execution of Nagios plugins' >> /etc/ufw/applications.d/nagios"
sudo sh -c "echo 'ports=5666/tcp' >> /etc/ufw/applications.d/nagios"
sudo ufw allow NRPE
sudo ufw reload
Skipping adding existing rule
Skipping adding existing rule (v6)
Firewall reloaded
root@monsrv01:/tmp/nrpe-nrpe-4.1.0#
root@monsrv01:/tmp/nrpe-nrpe-4.1.0#
```

Figura 11

Editamos el fichero de configuración ubicado en “/usr/local/nagios/etc/nrpe.cfg” y añadimos la IP local del servidor de monitorización en la línea “Allowed host”:

```
allowed_hosts=127.0.0.1,::1,172.31.3.74
```

Figura 12

Arrancamos el demonio con el comando “sudo systemctl start nrpe.service” y verificamos su estado:

```

root@monsrv01:/tmp/nrpe-nrpe-4.1.0# service nrpe.nagios status
Unit nrpe.nagios.service could not be found.
root@monsrv01:/tmp/nrpe-nrpe-4.1.0# service nrpe status
● nrpe.service - Nagios Remote Plugin Executor
   Loaded: loaded (/lib/systemd/system/nrpe.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2024-03-29 18:35:24 CET; 46min ago
     Docs: http://www.nagios.org/documentation
   Main PID: 928 (nrpe)
    Tasks: 1 (limit: 4557)
   Memory: 1.8M
      CPU: 39ms
   CGroup: /system.slice/nrpe.service
           └─928 /usr/local/nagios/bin/nrpe -c /usr/local/nagios/etc/nrpe.cfg -f

mar 29 18:35:24 monsrv01 systemd[1]: Started Nagios Remote Plugin Executor.
mar 29 18:35:25 monsrv01 nrpe[928]: Starting up daemon
mar 29 18:35:25 monsrv01 nrpe[928]: Server listening on 0.0.0.0 port 5666.
mar 29 18:35:25 monsrv01 nrpe[928]: Server listening on :: port 5666.
mar 29 18:35:25 monsrv01 nrpe[928]: Listening for connections on port 5666
mar 29 18:35:25 monsrv01 nrpe[928]: Allowing connections from: 127.0.0.1,::1,172.31.3.74
root@monsrv01:/tmp/nrpe-nrpe-4.1.0#

```

Figura 13

- NSCA: Para instalar este NSCA se hace uso del siguiente manual: <https://www.pluralsight.com/cloud-guru/labs/aws/configure-nagios-server-to-accept-passive-check-results-via-nasca> [11]. Primero de todo se debe descargar mediante el siguiente comando: “wget <http://prdownloads.sourceforge.net/sourceforge/nagios/nsca-2.9.1.tar.gz>”. Una vez descargado, se descomprime con el comando tar xvzf nsca-2.9.1.tar.gz y se configura con el comando “./configure --with-nasca-user=nagios --with-nasca-grp=nagcmd”. Posteriormente, se compila con el comando “sudo make all” y copiamos los siguientes archivos en las rutas correspondientes:

```

root@monsrv01:/tmp/nsca-2.9.1# cp sample-config/nsca.cfg sample-config/send_nasca.cfg /usr/local/nagios/etc/
root@monsrv01:/tmp/nsca-2.9.1/src# cp nsca.c /usr/local/nagios/bin
root@monsrv01:/tmp/nsca-2.9.1/src# cp send_nasca.c /usr/local/nagios/bin

```

Figura 14

Y cambiamos los permisos y propietario:

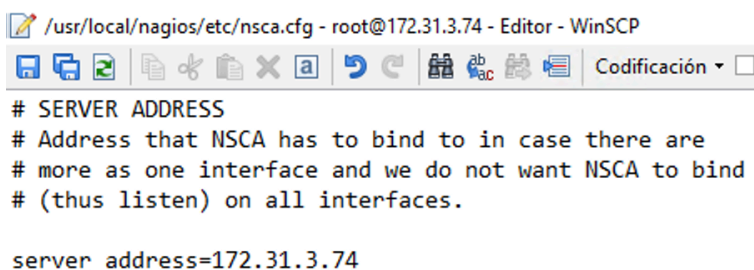
```

root@monsrv01:/tmp/nsca-2.9.1/src# chown nagios:nagios /usr/local/nagios/bin/nsca /usr/local/nagios/bin/send_nasca
root@monsrv01:/tmp/nsca-2.9.1/src# chown nagios:nagcmd /usr/local/nagios/etc/nsca.cfg /usr/local/nagios/etc/send_nasca.cfg
root@monsrv01:/tmp/nsca-2.9.1/src# chmod g+r /usr/local/nagios/etc/nsca.cfg

```

Figura 15

Posteriormente, se debe de editar el fichero “/usr/local/nagios/etc/nsca.cfg” para añadir el servidor de monitorización:



```

# SERVER ADDRESS
# Address that NSCA has to bind to in case there are
# more as one interface and we do not want NSCA to bind
# (thus listen) on all interfaces.

server_address=172.31.3.74

```

Figura 16

Ejecutamos el siguiente comando:

```
root@monsrv01:/# sudo /usr/local/nagios/bin/nsca -c /usr/local/nagios/etc/nsca.cfg
```

Figura 17

Y se abren los puertos:

```
root@monsrv01:~# sudo ufw allow 5667/tcp
Rules updated
Rules updated (v6)
```

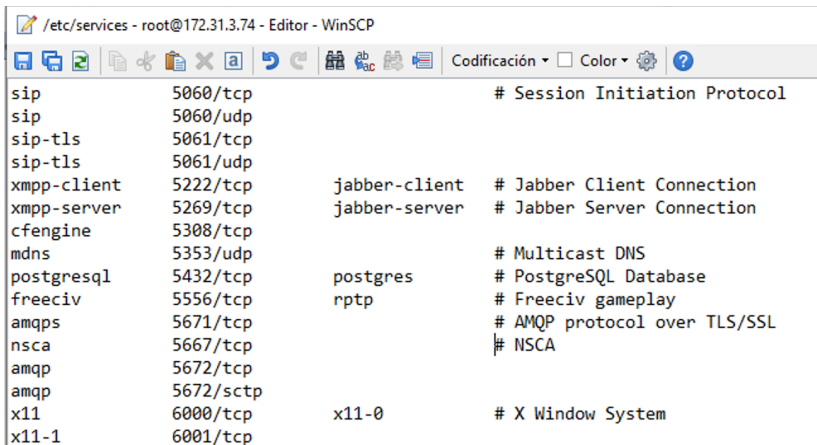
Figura 18

Y se verifica que el servicio este en escucha:

```
root@monsrv01:/tmp/nsca-2.9.1/src# ps -ef | grep -v grep | grep -i nsca
nagios 6198 1 0 14:27 ? 00:00:00 /usr/local/nagios/bin/nsca -c /usr/local/nagios/etc/nsca.cfg
root@monsrv01:/tmp/nsca-2.9.1/src# netstat -planet | grep 5667
tcp 0 0 0.0.0.0:5667 0.0.0.0:* LISTEN 1001 36076 6198/nsca
root@monsrv01:/tmp/nsca-2.9.1/src#
```

Figura 19

Añadimos el servicio en el archivo “/etc/services” de este modo se simplifica la administración de la configuración de red y mejora la legibilidad y comprensión del entorno de red en un sistema:



```
/etc/services - root@172.31.3.74 - Editor - WinSCP
Codificación Color
sip 5060/tcp # Session Initiation Protocol
sip 5060/udp
sip-tls 5061/tcp
sip-tls 5061/udp
xmpp-client 5222/tcp jabber-client # Jabber Client Connection
xmpp-server 5269/tcp jabber-server # Jabber Server Connection
cfengine 5308/tcp
mdns 5353/udp # Multicast DNS
postgresql 5432/tcp postgres # PostgreSQL Database
freeciv 5556/tcp rtp # Freeciv gameplay
amqps 5671/tcp # AMQP protocol over TLS/SSL
nsca 5667/tcp # NSCA
amqp 5672/tcp
amqp 5672/sctp
x11 6000/tcp x11-0 # X Window System
x11-1 6001/tcp
```

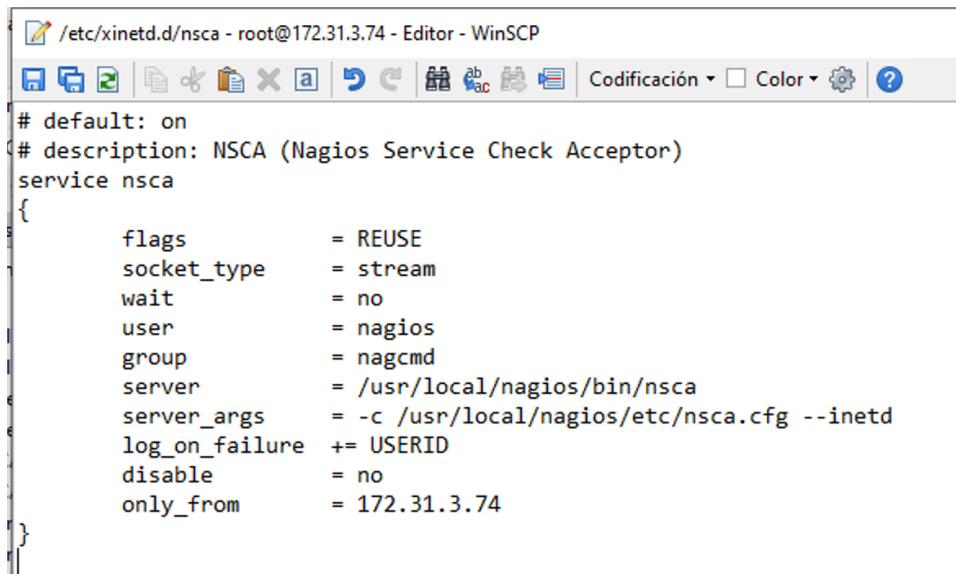
Figura 20

Copiamos el archivo de ejemplo nsca.xinetd dentro del directorio /etc/xinetd.d/nsca:

```
root@monsrv01:/tmp/nsca-2.9.1# cp sample-config/nsca.xinetd /etc/xinetd.d/nsca
root@monsrv01:/tmp/nsca-2.9.1# rm /var/run/nsca.pid
```

Figura 21

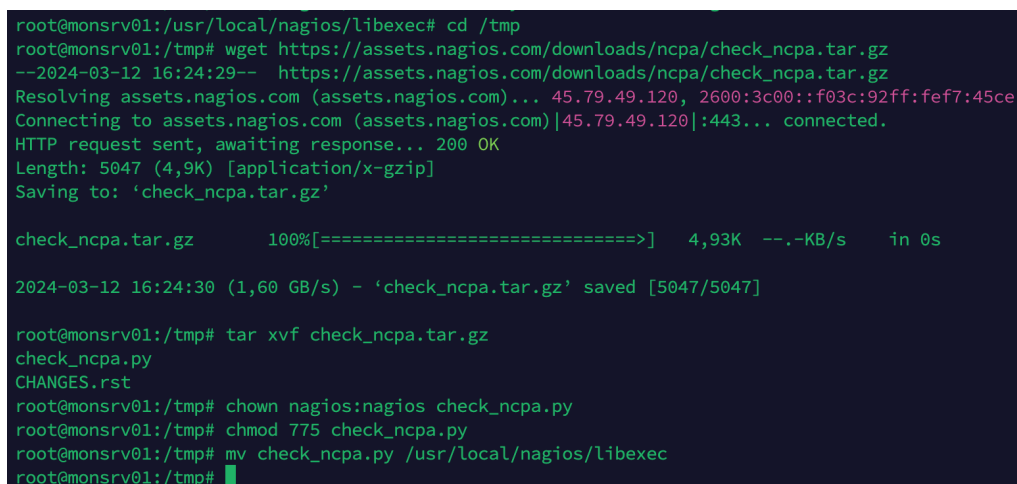
Y para finalizar, configuramos el archivo `/etc/xinetd.d/nsca` para añadir los equipos que serán escuchados para el servicio NSCA en la línea “`only_from`” y por ahora añadimos al propio servidor de monitorización:



```
#!/etc/xinetd.d/nsca - root@172.31.3.74 - Editor - WinSCP
# default: on
# description: NSCA (Nagios Service Check Acceptor)
service nsca
{
    flags          = REUSE
    socket_type    = stream
    wait          = no
    user          = nagios
    group         = nagcmd
    server        = /usr/local/nagios/bin/nsca
    server_args    = -c /usr/local/nagios/etc/nsca.cfg --inetd
    log_on_failure += USERID
    disable       = no
    only_from     = 172.31.3.74
}
```

Figura 22

- NCPA: Para instalar el plugin NCPA, se hace uso del siguiente manual <https://assets.nagios.com/downloads/nagioscore/docs/nagioscore/3/en/monitoring-windows.html> [12]. Lo descargamos mediante el siguiente comando: “`wget https://assets.nagios.com/downloads/ncpa/check_ncpa.tar.gz`”, posteriormente descomprimos el paquete con el comando: “`tar xvf check_ncpa.tar.gz`”, establecemos los permisos con “`chown nagios:nagios check_ncpa.py`” y “`chmod 775 check_ncpa.py`”. Posteriormente movemos el fichero al directorio de los scripts de nagios con el comando “`mv check_ncpa.py /usr/local/nagios/libexec`”



```
root@monsrv01:/usr/local/nagios/libexec# cd /tmp
root@monsrv01:/tmp# wget https://assets.nagios.com/downloads/ncpa/check_ncpa.tar.gz
--2024-03-12 16:24:29-- https://assets.nagios.com/downloads/ncpa/check_ncpa.tar.gz
Resolving assets.nagios.com (assets.nagios.com)... 45.79.49.120, 2600:3c00::f03c:92ff:fef7:45ce
Connecting to assets.nagios.com (assets.nagios.com)|45.79.49.120|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5047 (4,9K) [application/x-gzip]
Saving to: 'check_ncpa.tar.gz'

check_ncpa.tar.gz      100%[=====] 4,93K  --KB/s   in 0s

2024-03-12 16:24:30 (1,60 GB/s) - 'check_ncpa.tar.gz' saved [5047/5047]

root@monsrv01:/tmp# tar xvf check_ncpa.tar.gz
check_ncpa.py
CHANGES.rst
root@monsrv01:/tmp# chown nagios:nagios check_ncpa.py
root@monsrv01:/tmp# chmod 775 check_ncpa.py
root@monsrv01:/tmp# mv check_ncpa.py /usr/local/nagios/libexec
root@monsrv01:/tmp#
```

Figura 23

Al intentar comprobar la versión, nos percatamos de que no encuentra Python, por lo tanto, deberemos de crear un enlace simbólico:

```
root@monsrv01:~# /usr/local/nagios/libexec/check_ncpa.py -V
/usr/bin/env: 'python': No such file or directory
root@monsrv01:~# whereis python3
python3: /usr/bin/python3 /usr/lib/python3 /etc/python3 /usr/share/python3 /usr/share/man/man1/
python3.1.gz
root@monsrv01:~# sudo ln -s /usr/bin/python3 /usr/bin/python
root@monsrv01:~# /usr/local/nagios/libexec/check_ncpa.py -V
check_ncpa.py, Version 1.2.4
root@monsrv01:~#
```

Figura 24

- NRPD: Para instalar NRPD, se hace uso del siguiente manual: <https://support.nagios.com/kb/article/nrdp-installing-nrdp-from-source-602.html#Ubuntu> [13]. Debemos de descargarlo mediante el siguiente comando: “wget -O nrdp.tar.gz <https://github.com/NagiosEnterprises/nrdp/archive/1.5.1.tar.gz>.”

Posteriormente, lo descomprimos con el comando “tar xzf nrdp.tar.gz”. Una vez descomprimido, accedemos al recurso y se debe crear un directorio para almacenar los archivos php de NDRP donde copiaremos los archivos y estableceremos los permisos con los siguientes comandos: “sudo mkdir -p /usr/local/nrdp”, “sudo cp -r clients server LICENSE* CHANGES* /usr/local/nrdp” y “sudo chown -R nagios:nagios /usr/local/nrdp”:

```
root@monsrv01:/tmp# tar xzf nrdp.tar.gz
root@monsrv01:/tmp# cd /tmp/nrdp-1.5.1/
root@monsrv01:/tmp/nrdp-1.5.1# sudo mkdir -p /usr/local/nrdp
root@monsrv01:/tmp/nrdp-1.5.1# sudo cp -r clients server LICENSE* CHANGES* /usr/local/nrdp
root@monsrv01:/tmp/nrdp-1.5.1# sudo chown -R nagios:nagios /usr/local/nrdp
root@monsrv01:/tmp/nrdp-1.5.1#
```

Figura 25

Posteriormente se deben definir los tokens editando el siguiente fichero “/usr/local/nrdp/server/config.inc.php”:

```
/usr/local/nrdp/server/config.inc.php - root@172.31.3.74 - Editor - WinSCP
Codificación Color
<?php
//
// NRDP Config File
//
// Copyright (c) 2010-2017 - Nagios Enterprises, LLC.
// License: Nagios Open Software License <http://www.nagios.com/legal/licenses>
//

// An array of one or more tokens that are valid for this NRDP install
// a client request must contain a valid token in order for the NRDP to response or honor the request
// NOTE: Tokens are just alphanumeric strings - make them hard to guess!
$config['authorized_tokens'] = array(
    // "mysecrettoken", // <-- not a good token
    // "90dfs7jwn3", // <-- a better token (don't use this exact one, make your own)
    "PasswordSecure",
```

Figura 26

Copiamos el nuevo sitio a apache:

```
root@monsrv01:/tmp/nrdp-1.5.1# sudo cp nrdp.conf /etc/apache2/sites-enabled/  
root@monsrv01:/tmp/nrdp-1.5.1#
```

Figura 27

Y podremos acceder al nuevo sitio web:

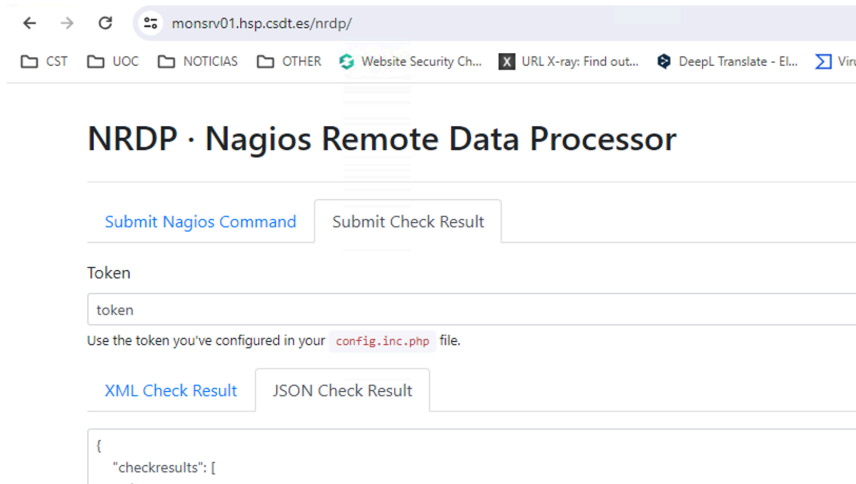


Figura 28

3.4 Certificación del sitio web de Nagios Core

Para certificar el sitio web del servidor Ubuntu Server, debemos de seguir el correspondiente manual oficial: <https://support.nagios.com/kb/article/nagios-core-configuring-ssl-tls-595.html#Ubuntu> [14]

Primero de todo, se procede a crear un registro DNS en nuestro dominio que apunte a nuestro servidor Ubuntu Server 22.04:

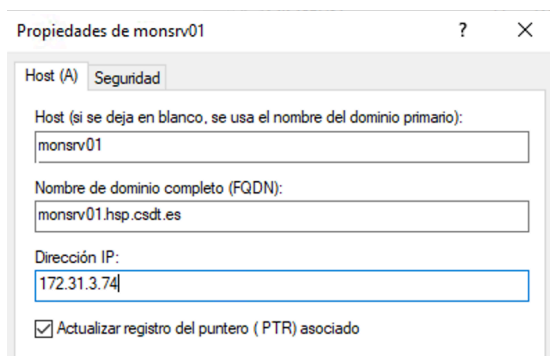


Figura 29

Se verifica el correcto funcionamiento mediante un ping al hostname ya que resuelve correctamente:

```
C:\Users\27537>ping monsrv01.hsp.csdt.es

Haciendo ping a monsrv01.hsp.csdt.es [172.31.3.74] con 32 bytes de datos:
Respuesta desde 172.31.3.74: bytes=32 tiempo<1m TTL=64
Respuesta desde 172.31.3.74: bytes=32 tiempo<1m TTL=64
Respuesta desde 172.31.3.74: bytes=32 tiempo<1m TTL=64
Respuesta desde 172.31.3.74: bytes=32 tiempo<1m TTL=64

Estadísticas de ping para 172.31.3.74:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
              (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
    Mínimo = 0ms, Máximo = 0ms, Media = 0ms
```

Figura 30

Se accede al sitio web y se verifica que no se dispone de certificación:

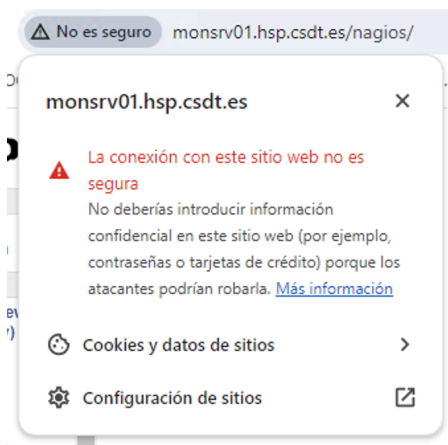


Figura 31

Posteriormente, se crea el siguiente archivo en /etc/ssl/openssl-csr.conf con los dos nombres de DNS para poder certificarlos con la entidad certificadora del dominio con extensiones de nombre alternativo del sujeto. Ya que es requisito de los actuales navegadores que se tengan dos nombres de sujeto. Para ello, se ha seguido la siguiente documentación: <https://netassured.co.uk/openssl-certificate-signing-request-subject-alternative-name/> [15]

```
/etc/ssl/openssl-csr.conf - root@172.31.3.74 - Editor - WinSCP

[ req ]
default_bits          = 2048
distinguished_name    = req_distinguished_name
req_extensions        = req_ext
[ req_distinguished_name ]
countryName           = ES
stateOrProvinceName   = Barcelona
localityName           = Terrassa
organizationName       = HSP
commonName             = MONSRV01.hsp.csdt.es
[ req_ext ]
subjectAltName = @alt_names
[alt_names]
DNS.1 = monsrv01.hsp.csdt.es
DNS.2 = nagios01.hsp.csdt.es
```

Figura 32

Y emite el correspondiente certificado:

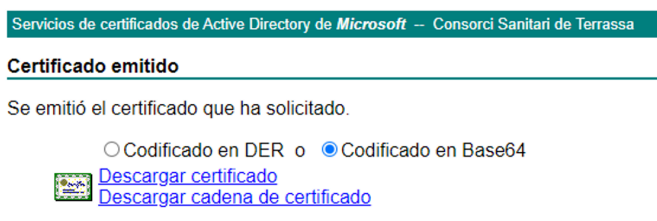


Figura 36

Mediante WinSCP se sube el nuevo certificado generado .crt que contiene un certificado digital codificado en Base64:

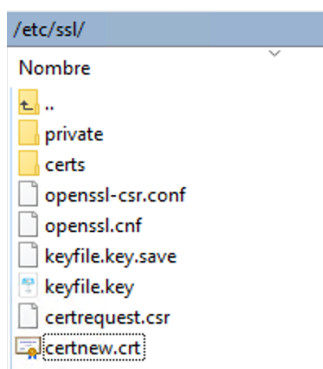


Figura 37

Posteriormente, se deben de copiar los ficheros en los siguientes directorios (que vienen a ser la clave publica junto sus datos como la información del sujeto, periodo de validez, etc y la clave privada) y establecer los permisos correspondientes con los siguientes comandos: “sudo cp certnew.crt /etc/ssl/certs/”, “sudo cp keyfile.key /etc/ssl/private/”, “sudo chmod go-rwx /etc/ssl/certs/certnew.crt”, “sudo chmod go-rwx /etc/ssl/private/keyfile.key”:

```
root@monsrv01:/etc/ssl# sudo cp certnew.crt /etc/ssl/certs/  
root@monsrv01:/etc/ssl# sudo cp keyfile.key /etc/ssl/private/  
root@monsrv01:/etc/ssl# sudo chmod go-rwx /etc/ssl/certs/certnew.crt  
root@monsrv01:/etc/ssl# sudo chmod go-rwx /etc/ssl/private/keyfile.key  
root@monsrv01:/etc/ssl#
```

Figura 38

Después del paso anterior, se debe de habilitar el mod ssl de apache, para ello, se hacen uso de los siguientes comandos: “sudo a2enmod ssl” y “sudo a2enmod rewrite” y se debe de reiniciar el servicio de apache2:

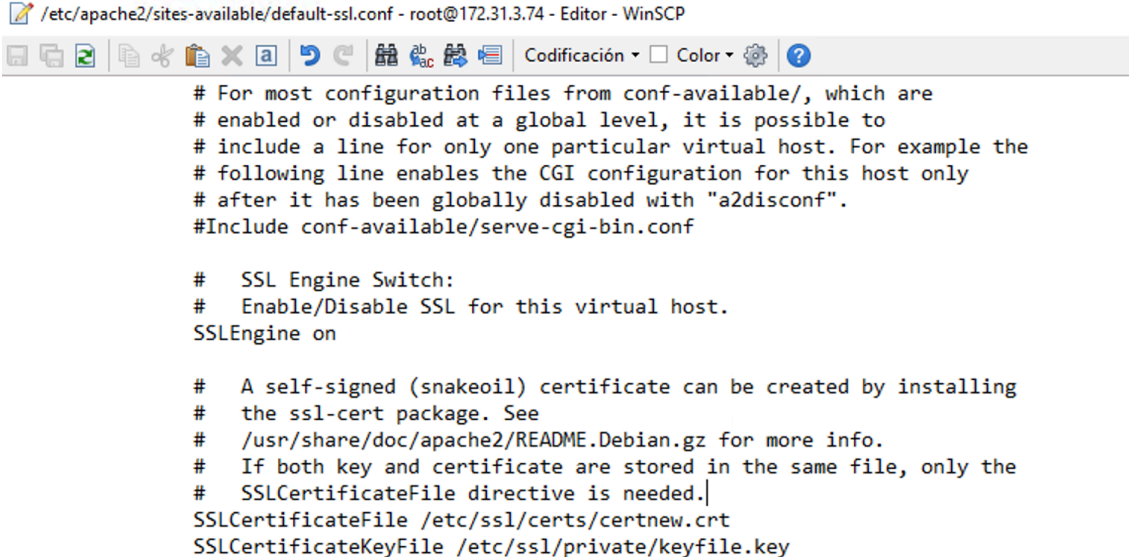
```

root@monsrv01:/etc/ssl# sudo a2enmod ssl
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and create self-signed certificates.
To activate the new configuration, you need to run:
  systemctl restart apache2
root@monsrv01:/etc/ssl# sudo a2enmod rewrite
Module rewrite already enabled
root@monsrv01:/etc/ssl# systemctl restart apache2
root@monsrv01:/etc/ssl#

```

Figura 39

Tras el reinicio, se debe de indicar al servidor web donde encontrar el nuevo certificado, para ello, se debe de editar el fichero “/etc/apache2/sites-available/default-ssl.conf” modificando las dos siguientes líneas: “SSLCertificateFile /etc/ssl/certs/certfile.crt” y “SSLCertificateKeyFile /etc/ssl/private/keyfile.key” por “SSLCertificateFile /etc/ssl/certs/certfile.crt” y “SSLCertificateKeyFile /etc/ssl/private/keyfile.key”:



```

# For most configuration files from conf-available/, which are
# enabled or disabled at a global level, it is possible to
# include a line for only one particular virtual host. For example the
# following line enables the CGI configuration for this host only
# after it has been globally disabled with "a2disconf".
#Include conf-available/serve-cgi-bin.conf

# SSL Engine Switch:
# Enable/Disable SSL for this virtual host.
SSLEngine on

# A self-signed (snakeoil) certificate can be created by installing
# the ssl-cert package. See
# /usr/share/doc/apache2/README.Debian.gz for more info.
# If both key and certificate are stored in the same file, only the
# SSLCertificateFile directive is needed.
SSLCertificateFile /etc/ssl/certs/certnew.crt
SSLCertificateKeyFile /etc/ssl/private/keyfile.key

```

Figura 40

Posteriormente, se edita el fichero “/etc/apache2/sites-available/000-default.conf” para redirigir todas las peticiones al protocolo HTTPS para que la comunicación del navegador con el servidor de monitorización vaya encriptada mediante SSL/TLS. Se deben de añadir las siguientes líneas:

```

RewriteEngine On
RewriteCond %{HTTPS} off
RewriteRule (.*) https://%{HTTP\_HOST}%{REQUEST\_URI}

```

```
/etc/apache2/sites-available/000-default.conf - root@172.31.3.74 - Editor - WinSCP
Codificación Color
# redirection URLs. In the context of virtual hosts, the ServerName
# specifies what hostname must appear in the request's Host: header to
# match this virtual host. For the default virtual host (this file) this
# value is not decisive as it is used as a last resort host regardless.
# However, you must set it for any further virtual host explicitly.
#ServerName www.example.com

ServerAdmin webmaster@localhost
DocumentRoot /var/www/html

# Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
# error, crit, alert, emerg.
# It is also possible to configure the loglevel for particular
# modules, e.g.
#LogLevel info ssl:warn

ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined

# For most configuration files from conf-available/, which are
# enabled or disabled at a global level, it is possible to
# include a line for only one particular virtual host. For example the
# following line enables the CGI configuration for this host only
# after it has been globally disabled with "a2disconf".
#Include conf-available/serve-cgi-bin.conf

RewriteEngine On
RewriteCond %{HTTPS} off
RewriteRule (.*) https://%{HTTP_HOST}%{REQUEST_URI}
|
</VirtualHost>
```

Figura 41

Después de la redirección, se debe de habilitar lo que permite que el servidor web escuche y responda a las solicitudes HTTPS mediante el siguiente comando: “sudo a2ensite default-ssl.conf”:

```
root@monsrv01:/etc/ssl# sudo a2ensite default-ssl.conf
Site default-ssl already enabled
root@monsrv01:/etc/ssl#
```

Figura 42

Se debe de reiniciar el servidor apache2 mediante el siguiente comando: “sudo systemctl reload apache2.service”

```
root@monsrv01:/etc/ssl# sudo systemctl reload apache2.service
root@monsrv01:/etc/ssl#
```

Figura 43

Y finalmente, se deben de abrir los puertos en el firewall del https con los siguientes comandos “sudo ufw allow https” y “sudo ufw reload”:

```
root@monsrv01:/etc/ssl# sudo ufw allow https
Rules updated
Rules updated (v6)
root@monsrv01:/etc/ssl# sudo ufw reload
```

Figura 44

Se verifica el correcto funcionamiento del nuevo certificado, la redirección https y que la comunicación entre el cliente (navegador) y servidor va encriptada:

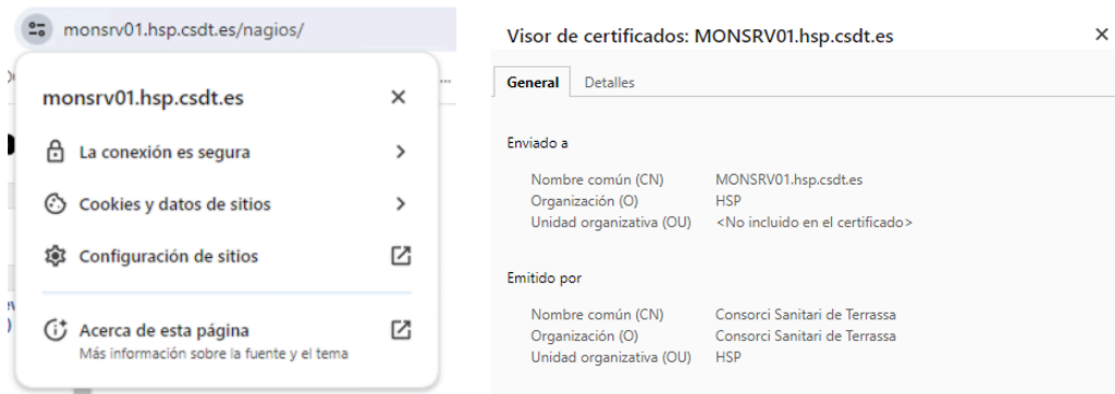


Figura 45

3.5 Limitación de acceso web a determinados clientes al portal Web de Nagios Core

Se va a limitar el acceso al sitio web de Nagios Core a un único equipo, para ello se debe de editar el fichero ubicado en "/etc/apache2/sites-enabled/nagios.conf" que originalmente está como en la imagen de la izquierda y se quedará como en la imagen de la derecha, después de ello se debe de reiniciar el servicio de apache:

```
<Directory "/usr/local/nagios/share">
# SSLRequireSSL
Options None
AllowOverride None
<IfVersion >= 2.3>
  <RequireAll>
    Require all granted
    Require host 127.0.0.1

    AuthName "Nagios Access"
    AuthType Basic
    AuthUserFile /usr/local/nagios/etc/htpasswd.users
    Require valid-user
  </RequireAll>
</IfVersion>
<IfVersion < 2.3>
  Order allow,deny
  Allow from all
# Order deny,allow
# Deny from all
# Allow from 127.0.0.1

  AuthName "Nagios Access"
  AuthType Basic
  AuthUserFile /usr/local/nagios/etc/htpasswd.users
  Require valid-user
</IfVersion>
</Directory>
```

```
<Directory "/usr/local/nagios/share">
# SSLRequireSSL
Options None
AllowOverride None
<IfVersion >= 2.3>
  <RequireAll>
    Require ip 172.31.154.216
    Require host 127.0.0.1
    AuthName "Nagios Access"
    AuthType Basic
    AuthUserFile /usr/local/nagios/etc/htpasswd.users
    Require valid-user
  </RequireAll>
</IfVersion>
<IfVersion < 2.3>
  # Order allow,deny
  # Allow from all
  Order deny,allow
  Deny from all
  Allow from 172.31.154.216

  AuthName "Nagios Access"
  AuthType Basic
  AuthUserFile /usr/local/nagios/etc/htpasswd.users
  Require valid-user
</IfVersion>
</Directory>
```

Figura 46

De este modo, se autoriza únicamente a la IP 172.31.154.216 para el acceso al sitio web. Procedemos a su verificación (cliente autorizado a la izquierda, cliente no autorizado a la derecha):

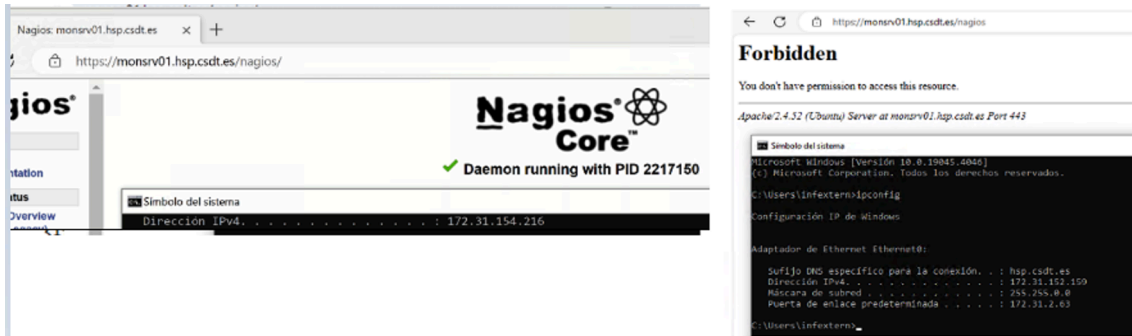


Figura 47

3.6 Uso de Postfix para el envío de alertas en Nagios Core

Una vez se tiene funcionamiento Postfix (véase en anexos la instalación), se procede a configurar un host de prueba y validar el funcionamiento de las alertas de Nagios Core. Para ello, lo incluimos en la siguiente plantilla:

```

/usr/local/nagios/etc/objects/windows.cfg - root@172.31.3.74 - Editor - WinSCP
define host {
    use                windows-server          ; Inherit default values from a template
    host_name          VDICST63                ; The name we're giving to this host
    alias              VDICST63               ; A longer name associated with the host
    address            172.31.153.202         ; IP address of the host
}

```

Figura 48

Se debe de descomentar la siguiente línea para activar la plantilla:

```

/usr/local/nagios/etc/nagios.cfg - root@172.31.3.74 - Editor - WinSCP
# Definitions for monitoring the local (Linux) host
cfg_file=/usr/local/nagios/etc/objects/localhost.cfg

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

```

Figura 49

Posteriormente, se le debe de indicar a quien se le enviarán las notificaciones:


```

/usr/local/nagios/etc/objects/contacts.cfg - root@172.31.3.74 - Editor - WinSCP
#####

#####
# CONTACTS
#
#####

# Just one contact defined by default - the Nagios admin (that's you)
# This contact definition inherits a lot of default values from the
# 'generic-contact' template which is defined elsewhere.

define contact {

    contact_name    nagiosadmin        ; Short name of user
    use              generic-contact    ; Inherit default values from generic-contact template (defined above)
    alias            Nagios Admin      ; Full name of user
    email            jmendez@cst.cat| ; <<***** CHANGE THIS TO YOUR EMAIL ADDRESS *****
}

```

Figura 50

Se reinicia el servicio del Nagios y modificamos la siguiente plantilla para establecer que haga un check cada minuto en check_interval y que nos reenvíe las alertas cada 15 minutos:

```

/usr/local/nagios/etc/objects/templates.cfg - root@172.31.3.74 - Editor - WinSCP
#####

# Windows host definition template
# This is NOT a real host, just a template!

define host {

    name                windows-server        ; The name of this host template
    use                 generic-host         ; Inherit default values from the generic-host template
    check_period        24x7                ; By default, Windows servers are monitored round the clock
    check_interval      1                   ; Actively check the server every 5 minutes
    retry_interval      1                   ; Schedule host check retries at 1 minute intervals
    max_check_attempts  1                   ; Check each server 10 times (max)
    check_command        check-host-alive   ; Default command to check if servers are "alive"
    notification_period 24x7                ; Send notification out at any time - day or night
    notification_interval 15               ; Resend notifications every 15 minutes
    notification_options d,u,r             ; Only send notifications for specific host states
    contact_groups      admins             ; Notifications get sent to the admins by default
    hostgroups          windows-servers    ; Host groups that Windows servers should be a member of
    register            0                   ; DON'T REGISTER THIS - ITS JUST A TEMPLATE
}

```

Figura 51

Se accede al equipo cliente, y se le deshabilita la interfaz de red para ver si recibe la pertinente notificación:

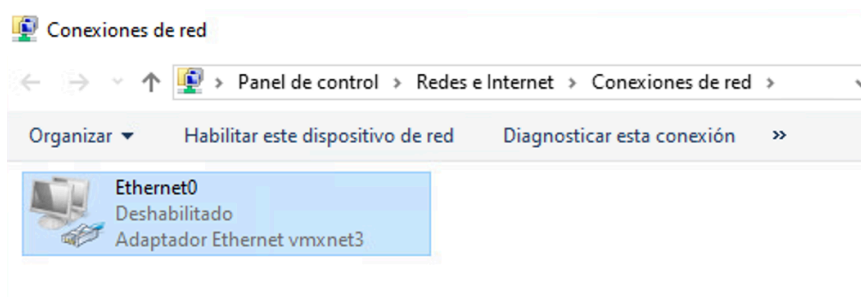


Figura 52

Se visualiza que en el sitio web del Nagios Core ya no comunica:

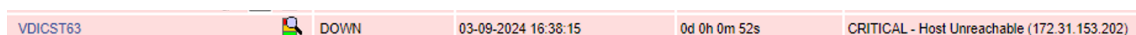


Figura 53

Y se recibe la correspondiente alerta:

**** PROBLEM Host Alert: VDICST63 is DOWN ****

 nagios@monsrv01
Per a: Mendez Torres, Jose Manuel

PRECAUCIÓ: aquest correu electrònic s'ha originat fora de l'organització reconegueu el remitent i que el contingut sigui segur.

***** Nagios *****

Notification Type: PROBLEM
Host: VDICST63
State: DOWN
Address: 172.31.153.202
Info: CRITICAL - Host Unreachable (172.31.153.202)

Figura 54

Restablecemos la interfaz para verificar que recibimos alerta indicando que se ha restablecido y recibimos la pertinente confirmación de restablecimiento del equipo:

 nagios@monsrv01
Per a: Mendez Torres, Jose Manuel

PRECAUCIÓ: aquest correu electrònic s'ha originat fora de l'organització. NO FEU CLIC a enllaços reconegueu el remitent i que el contingut sigui segur.

***** Nagios *****

Notification Type: RECOVERY
Host: VDICST63
State: UP
Address: 172.31.153.202
Info: PING OK - Packet loss = 0%, RTA = 0,64 ms

Figura 55


Se debe de configurar postfix para que envíe de forma encriptada por SMTP, para ello, haremos uso de los certificados generados anteriormente. Se debe de editar el siguiente fichero para especificar las rutas “/etc/postfix/main.cf”:

```
# TLS parameters
smtpd_tls_cert_file=/etc/ssl/certs/certfile.crt
smtpd_tls_key_file=/etc/ssl/private/keyfile.key
smtpd_tls_security_level=may

smtp_tls_CApath=/etc/ssl/certs
smtp_tls_security_level=may
smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache
```

Figura 56

Reiniciamos el servicio con el comando “sudo systemctl restart postfix” y se verifica el correcto funcionamiento:

 **nagios@monsrv01**
Per a: Mendez Torres, Jose Manuel

PRECAUCIÓ: aquest correu electrònic s'ha originat fora de l'organització. NO FEU CLIC a enllaços: reconegueu el remitent i que el contingut sigui segur.

***** Nagios *****

Notification Type: RECOVERY
Host: VDICST63
State: UP
Address: 172.31.153.202
Info: PING OK - Packet loss = 0%, RTA = 0.64 ms

Figura 57

4. Investigación y pruebas:

4.1 Investigación y explicación de protocolos de monitorización utilizados por Nagios Core

Nagios Core admite una gran variedad de protocolos para monitorear diferentes servicios. Algunos de estos protocolos de monitorización son propios de Nagios y otros no. Nagios Core puede utilizar los siguientes, aunque puede utilizar más que no están mencionados en este listado:

- ICMP: Permite verificar si los equipos están accesibles en la red.
- SNMP: Permite la monitorización de equipos en la red. Se utiliza para recopilar información sobre su estado y rendimiento.
- HTTP: Protocolo utilizado para acceder a páginas web. En Nagios Core se utiliza para realizar peticiones HTTP a servidores web para verificar si están accesibles.
- HTTPS: Protocolo utilizado para acceder a páginas web. En Nagios Core se utiliza para realizar peticiones HTTP a servidores web para verificar si están accesibles.
- SMTP: Protocolo utilizado para el envío de correos electrónicos. En Nagios Core se utiliza para monitorear si el servicio SMTP de un servidor de correo electrónico está funcionando. Se pueden enviar correos electrónicos de prueba.
- POP/IMAP: Son protocolos de recepción de correo electrónicos. POP es utilizado para descargar el correo en local y al poco tiempo se elimina de un servidor de correo electrónico. IMAP se utiliza para mantener sincronizados los correos tanto en local como en el servidor de correo y de este modo que sea accesible desde diferentes equipos. Se utilizan para monitorear si los servicios POP, IMAP de un servidor de correo electrónico está funcionando. Se pueden efectuar conexiones por su puerto para verificar que este accesible en la red.
- SSH: Es un protocolo utilizado para acceder de forma remota a un equipo mediante un túnel cifrado. En Nagios Core se utiliza para que se pueda ejecutar comandos remotos en servidores para verificar su estado y recopilar información.
- DNS: Es un protocolo de resolución de nombres de dominio. En Nagios Core se utiliza para verificar la disponibilidad de servidores DNS y la resolución de nombres.

- LDAP: Es un protocolo estándar de acceso a directorios. Se utiliza para monitorear los servidores LDAP y directorios de acceso.
- FTP: Es un protocolo para el envío y recepción de ficheros de un cliente-servidor sin encriptación. En Nagios Core se utiliza para verificar la disponibilidad de ese servicio.
- SFTP: Al igual que FTP es un protocolo para el envío y recepción de ficheros de un cliente-servidor, pero de forma encriptada. En Nagios Core se utiliza para verificar la disponibilidad de ese servicio.
- TCP/UDP: Son protocolos de transmisión de datos donde en TCP se tiene control sobre toda la trama (descarta la trama si los paquetes no están completos) y en UDP no se tiene ese control tan exhaustivo. En Nagios Core se utiliza para verificar que puertos específicos de diferentes servicios estén operativos.

Existen de otros protocolos que son propios de Nagios:

- NRPE [\[16\]](#): Es un protocolo propio de Nagios. Las siglas significan: Nagios Remote Plugin Executor. Este protocolo permite el uso de plugins de Nagios en máquinas remotas con el fin de supervisar los recursos locales de estos. Su uso es más para comprobaciones activas, pero también se permiten comprobaciones pasivas, donde los clientes envían los resultados del servidor Nagios.
- NSCA [\[17\]](#): Es un protocolo propio de Nagios. Las siglas significan: Nagios Service Check Acceptor. Este protocolo es utilizado para que los propios servidores o clientes envíen la información de sus comprobaciones activas al servidor de monitorización Nagios. Por lo tanto, son comprobaciones pasivas. Para las comprobaciones pasivas se requiere crear un directorio en el servidor monitorizado y en él se ejecutará por medio de crontab los chequeos requeridos.

A continuación, procederemos a desarrollar algunos de estos protocolos que requieren de mayores conocimientos para su uso en Nagios Core:

SNMP

SNMP [\[18\]](#) es un protocolo simple para la administración de red. es un protocolo de capa de aplicación definido para intercambiar información de administración entre los diferentes dispositivos de red. Forma parte del conjunto de protocolos Protocolo de control de transmisión/Protocolo de Internet (TCP/IP).

SNMP es reconocido como uno de los protocolos más utilizados para la gestión y supervisión de dispositivos de red. La mayoría de los dispositivos de red de alta gama están equipados con un agente SNMP integrado, los cuales necesitan ser activados y ajustados para poder interactuar con el sistema de gestión de red (NMS).

SNMP está compuesto por:

- Administrador SNMP: En esta composición entraría nuestro software de monitorización “Nagios Core”, ya que sería el encargado de hacer de agente de consultas, obtener respuestas de agentes, establecer variables con agentes y reconocer eventos asincrónicos de agentes.
- Dispositivos administrados: clientes requeridos de monitorización.
- Agente SNMP: Se trata de un componente de software que es ejecutado en dispositivos de red. Sus funciones se basan en:
 - o Recopilación de información de administración local.
 - o Respuesta de solicitudes de gestión por parte de un administrador de SNMP.
 - o Generar trampas (*traps*) para notificar al administrador de SNMP eventos importantes de forma asíncrona.
 - o MIB (Base de Datos de Información): las MIB son archivos que contienen identificadores de objetos (OID) definidos de manera precisa. Cada objeto dentro de una MIB posee una descripción que detalla sus características en el dispositivo que se está administrando. Para interactuar con estos objetos, se utiliza el protocolo SNMP.

En la siguiente imagen se puede apreciar el diagrama básico de comunicación:

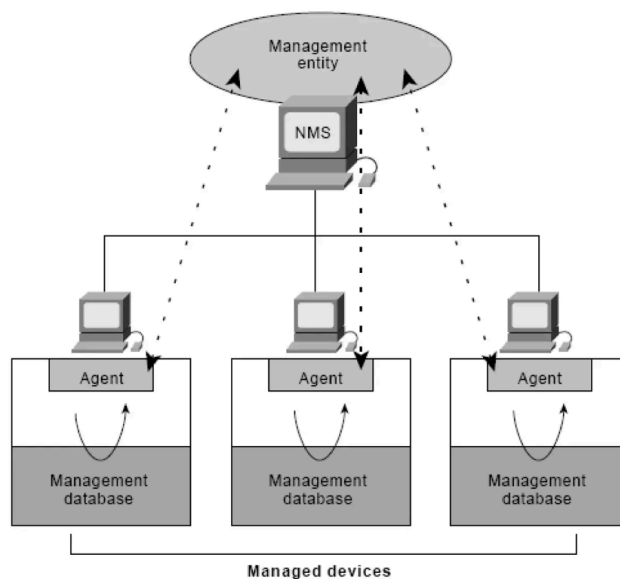


Figura 58

En la siguiente imagen podemos ver un diagrama del árbol MIB donde hay dos tipos de objetos:

- Escalar: Nombre del proveedor del dispositivo, el resultado solo puede ser uno.
- Tabular: El objeto tabular define varias instancias de objetos relacionados que se agrupan en tablas MIB

A continuación, se muestra el diagrama del árbol MIB:

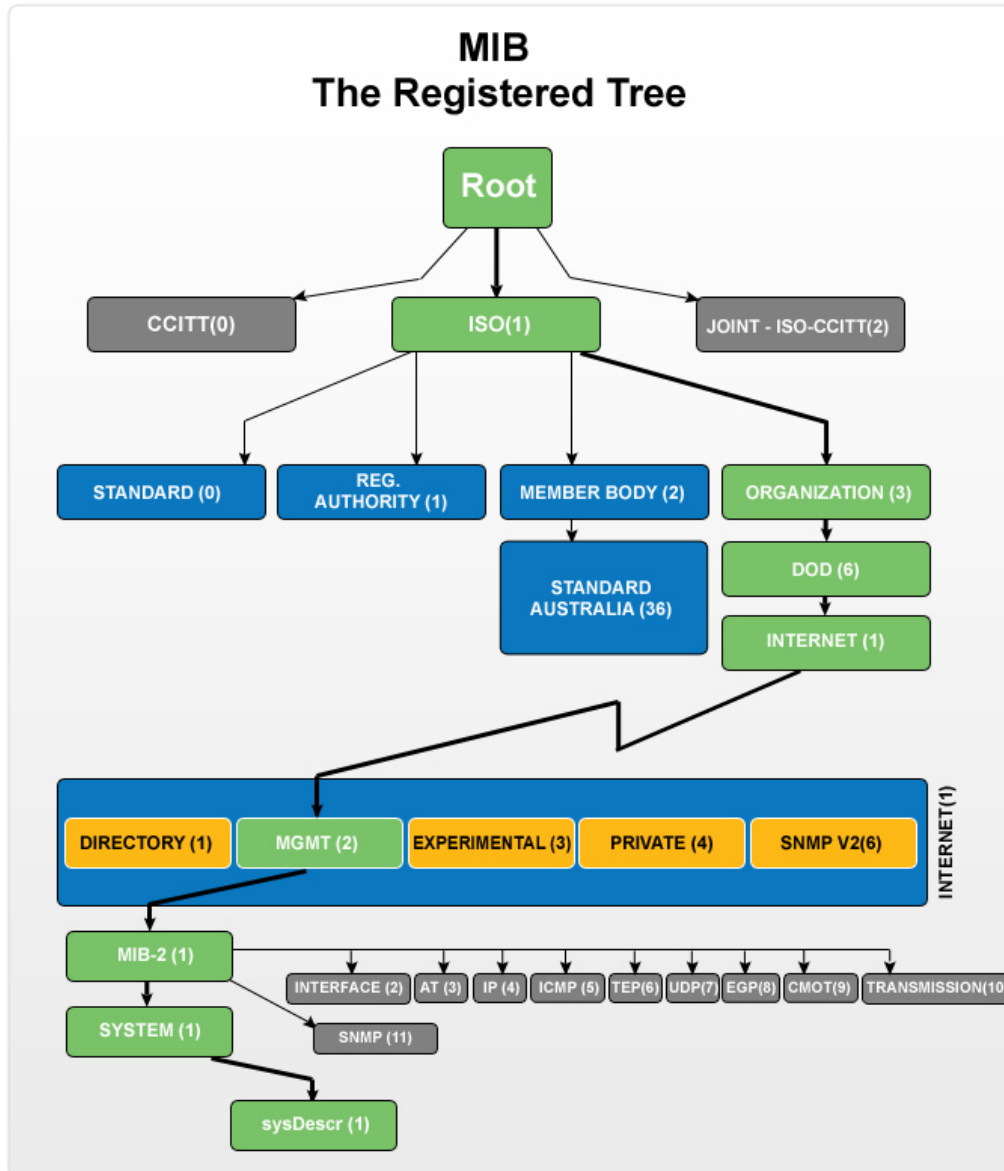


Figura 59

SNMP tiene 3 versiones:

- SNMPv1: Esta versión es relativamente fácil de utilizar para la monitorización de equipos y al no establecerse una comunicación cifrada no tiene demasiada sobrecarga. La principal restricción radica en su diseño de registro de 32 bits, el cual se queda corto para las demandas de las redes actuales, las cuales manejan cantidades de datos de un gigabyte o más.
- SNMPv2: Esta versión introduce una estructura MIB mejorada, permitiendo una organización más eficiente y con mayor

representatividad de los datos de gestión. Además, esta versión, añade características como el Modelo de Seguridad SNMPv2 (SNMPv2-SMI), que ofrece autenticación mejorada y opciones de privacidad, como el uso de contraseñas más seguras y la posibilidad de cifrar datos sensibles e incluso es compatible con SNMPv1.

- SNMPv3: incluye las ventajas de SNMPv2 y aporta soluciones de seguridad como cuentas de usuario, autenticación y cifrado de paquetes de datos opcional para garantizar la privacidad de los datos. También es compatible con la versión 1 y 2.

Estas tres versiones utilizan los mismos conceptos de Object Identifiers (OIDs) y Management Information Bases (MIBs), aunque puede haber algunas diferencias en la forma en que se implementan o gestionan.

SNMP utiliza el protocolo UDP como su transporte subyacente para enviar mensajes entre la estación de administración y el agente SNMP. UDP es un protocolo sin conexión que proporciona una comunicación rápida, pero no garantiza la entrega de mensajes ni el orden de llegada. En SNMPv2 se introdujo la opción de utilizar TCP como alternativa a UDP, pero UDP sigue siendo el protocolo predeterminado a utilizar.

NRPE

El protocolo NRPE [16] permite la ejecución de plugins en máquinas remotas para poder monitorear sus recursos locales de forma activa. Nagios ejecuta el comando `check_nrpe` con los argumentos requeridos para monitorear ese servicio y esta petición llega al cliente de forma cifrada y autenticada (si se configura) y ejecuta el comando pasado por argumento mediante el servidor de monitorización y se ejecuta un script local en el cliente que devuelve el resultado. Véase en la siguiente ilustración:

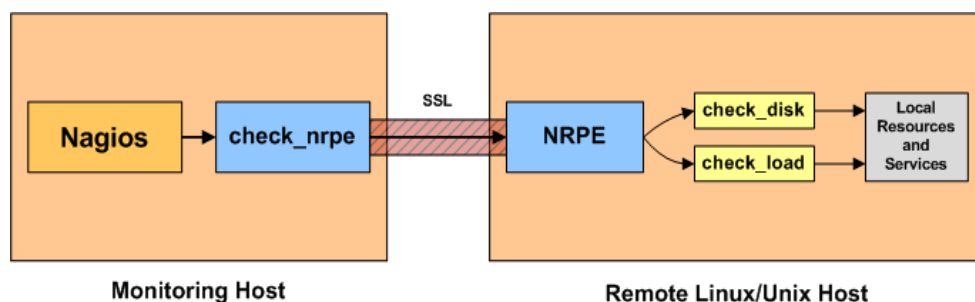


Figura 60

NSCA

El protocolo NSCA [17] permite que los clientes monitorizados envíen información de sus comprobaciones locales al servidor Nagios. Esto se logra mediante un script local que se ejecuta bajo una programación horaria (con `crontab` por ejemplo) y una vez ejecutado el Script envía al servidor monitorización la información de su comprobación, y el tráfico no va encriptado véase la siguiente ilustración:

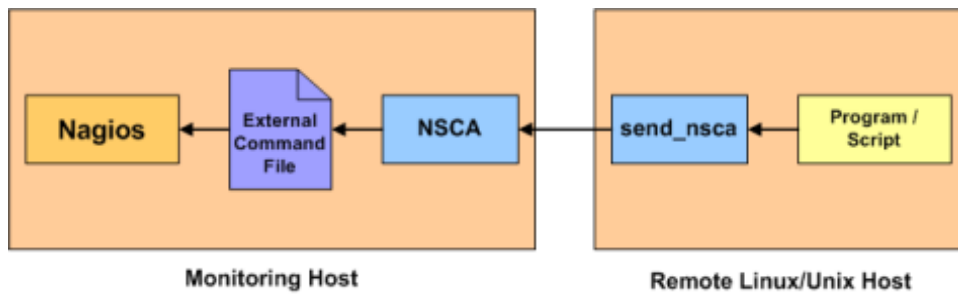


Figura 61

4.2 Investigación de software para clientes y explicar cuáles son más seguros - Nagios Core

A continuación, se va a explicar algunos de los clientes de monitorización que se pueden utilizar en entornos Windows y UNIX para Nagios Core.

NSClient++

NSClient++ es un software diseñado para ser utilizado como agente de monitorización entornos Windows. Se puede descargar desde el siguiente enlace: <https://github.com/mickem/nscp/releases>

Este agente puede ser utilizado para `check_nt`, NRPE Servidor y NSCA. A continuación, se va a detallar el funcionamiento:

- `Check_nt` [19]: Se trata de uno de los comandos principales utilizados para efectuar comprobaciones activas en el equipo local. El servidor de monitorización lanza el comando y se recibe en el cliente, este ejecuta el complemento o Script específico designado en la configuración de NSClient++. Véase en la siguiente ilustración:

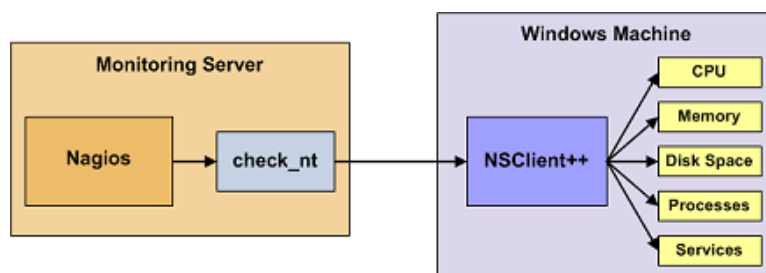


Figura 62

La comunicación entre el cliente y el servidor de monitorización no va cifrada, pero si se puede configurar una autenticación.

- NRPE Servidor [20]: NRPE Servidor utiliza el protocolo NRPE de Nagios descrito anteriormente. A diferencia de Check_nt cuando se hace uso de este plugin la comunicación se efectúa cifrada (siempre y cuando no se haga uso de una versión antigua del complemento). Para esta comunicación entre cliente y servidor se pueden seleccionar estas opciones:
 - o Modo heredado inseguro: no se requieren métodos de autenticación.
 - o Modo seguro: se utilizan certificados para la encriptación.
 - o Seguro: se utilizan certificados para la autenticación de ambos extremos de la conexión y la comunicación va cifrada.
 Véase la siguiente ilustración para entender su funcionamiento:

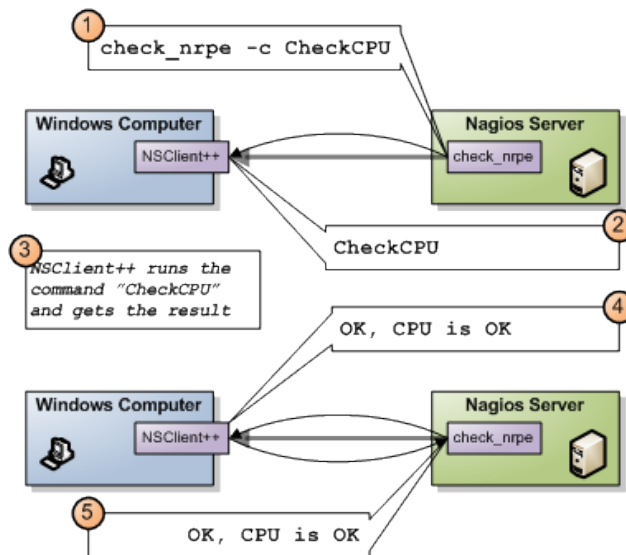


Figura 63

- NSCA Client [21]: Utiliza el protocolo NSCA de Nagios descrito anteriormente. Con esta opción se habilita al agente a actuar como cliente de monitorización y enviar resultados de comprobaciones de forma pasiva al servidor de monitorización. Se puede establecer una comunicación que requiera de autenticación. Véase la siguiente ilustración sobre su funcionamiento:

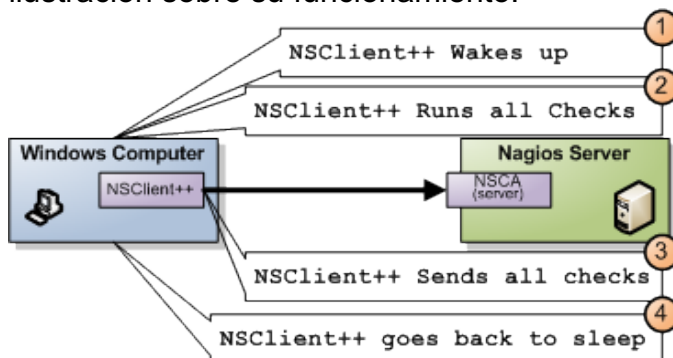


Figura 64

Plugin NRPE UNIX Client

En los entornos UNIX se puede instalar el plugin NRPE. Para descargarlo se puede hacer uso del siguiente enlace:

<https://assets.nagios.com/downloads/nagiosxi/agents/linux-nrpe-agent.tar.gz> [22]. El funcionamiento es el descrito anteriormente para este protocolo. [20]

Cuando se instala el agente, se añaden una serie de directorios con los plugins que se ejecutaran en local bajo petición del servidor de monitorización. Se deberá de configurar la encriptación y autenticación en el archivo de configuración del cliente.

Plugin NSCA UNIX Client - Nagios Service Check Acceptor

En los entornos UNIX se puede instalar el plugin NSCA. El funcionamiento es el descrito anteriormente para este protocolo [21]. Se puede descargar desde el siguiente enlace: <https://github.com/NagiosEnterprises/nsca/releases> [23]. Se debe de crear un directorio donde se descargarán las fuentes de NSCA que permitirán efectuar los chequeos pasivos. Para ello, se deberá de copiar el fichero de configuración principal al servidor de monitorización donde entre otros, en su interior aparece la dirección IP o nombre de host del cliente. Una vez hecho esto, se ha de configurar en el cliente cada cuando ejecutar sus Scripts de monitoreo local que se enviarán al servidor de monitorización.

NCPA – Nagios Cross-Platform Agent

En Windows se puede descargar desde el siguiente enlace:

<https://www.nagios.org/ncpa/> [24] y en UNIX a través de añadir las siguientes fuentes: `https://repo.nagios.com/deb/$(lsb_release -cs) /" > /etc/apt/sources.list.d/nagios.list.`

Este cliente utiliza TLS 1_2 para encriptar las comunicaciones, se le define un token al cliente para validarse con Nagios (que tiene definido el mismo token) y de esta forma se establece una autenticación. También existe la posibilidad de instalar el servidor NRPD en Nagios y de esta forma con el cliente efectuar chequeos pasivos si se configura dicha opción.

Una vez analizados los diferentes clientes se considera que NCPA es la mejor opción para este proyecto. En entornos UNIX cuando se instala este cliente automáticamente se habilita TLS y se instala un certificado adhoc para el cifrado de los datos, en cambio con NRPE se ha de configurar manualmente. el cliente NCPA es altamente compatible con diversos sistemas operativos, más flexible y versátil. Véase en la siguiente ilustración:

Agent Comparison:

Features \ Edition	NCPA	NRDS Agent	NSClient ++	NRPE
Installs on Linux	✓	✓		✓
Installs on Windows	✓	✓	✓	
Installs on Mac OSX	✓	✓		✓
Graphical User Interface	✓			
Active Check Metrics	✓		✓	✓
Passive Check Capabilities	✓	✓	✓	
Flexible API Access	✓			
Seamless Integration with Nagios XI	✓		✓	✓
Integration with Nagios Core via NRDP	✓	✓	✓	
Official Nagios Enterprises Monitoring Agent	✓			
Pre-configured Monitoring Metrics	✓		✓	✓
Integration with NRDS Configuration Protocol		✓		

Figura 65

NRPE ya sea con NSClient++ o el plugin en UNIX también es un protocolo y cliente seguro si se configura adecuadamente, pero esta más enfocado a entornos UNIX.

4.3 Investigar y explicar el funcionamiento de plantillas: contactos, servicios, comandos, etc.

Los archivos de configuración de Nagios Core están en el directorio: “/usr/local/nagios/etc/”. En este archivo se especifican las diferentes rutas de las diferentes plantillas [\[25\]](#):

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

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

Figura 66

Se va a explicar el funcionamiento de cada plantilla:

- **Commands.cfg:** Es el fichero donde se definen los diferentes comandos para la monitorización, véase la siguiente ilustración de ejemplo:

```
define command {
    command_name    check_local_disk
    command_line    $USER1$/check_disk -w $ARG1$ -c $ARG2$ -p $ARG3$
}
```

Figura 67

- **Contacts.cfg:** Es el fichero donde se definen los contactos y se agrupan en diferentes contactgroups. Serán los grupos de contactos los que relacionaremos con los hosts y servicios a fin de notificar sobre sus alertas. Cuando se produzca una alarma sobre un host o servicio se notificará a todos los miembros del o de los contactgroups que tenga definido ese host o servicio en cuestión.

```
#####
#
# CONTACTS
#
#####

# Just one contact defined by default - the Nagios admin (that's you)
# This contact definition inherits a lot of default values from the
# 'generic-contact' template which is defined elsewhere.

define contact {

    contact_name      nagiosadmin      ; Short name of user
    use                generic-contact  ; Inherit default values from generic-contact template (defined above)
    alias              Nagios Admin    ; Full name of user
    email              jmendez@cst.cat ; <<***** CHANGE THIS TO YOUR EMAIL ADDRESS *****

}

```

Figura 68

```
#####
#
# CONTACT GROUPS
#
#####

# We only have one contact in this simple configuration file, so there is
# no need to create more than one contact group.

define contactgroup {

    contactgroup_name  admins
    alias              Nagios Administrators
    members             nagiosadmin

}

```

Figura 69

- **Timeperiods.cfg:** Es el fichero en el que se definen las franjas horarias en las que realizaremos los “checks” y enviaremos las notificaciones.
- **Templates.cfg:** Este fichero sirve para estandarizar y simplificar la configuración al proporcionar un conjunto común de atributos que se pueden aplicar a múltiples objetos de monitoreo.

Si queremos generar una nueva plantilla de objetos debemos de editar el archivo “nagios.cfg” para incluir la nueva ubicación:

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

```

Figura 70

Posteriormente, creamos el archivo en la ruta indicada y lo primero que se debe de definir es el host que se quiere monitorear, por ejemplo:

```
define host {
    use                windows-server        ; Inherit default values from a template
    host_name          SHPCLI01              ; The name we're giving to this host
    alias              SHPCLI01              ; A longer name associated with the host
    address             172.31.154.188       ; IP address of the host
}
```

Figura 71

También, podemos añadir otro objeto que sea dependiente del anterior con “parents”:

```
define host {
    use                windows-server        ; Inherit default values from a template
    host_name          AF30219              ; The name we're giving to this host
    alias              W10_Manu              ; A longer name associated with the host
    address             172.31.150.109       ; IP address of the host
    parents             SHPCLI01
}
```

Figura 72

De este modo, una vez reiniciados los servicios, si vamos al mapa de Nagios podremos ver la dependencia:

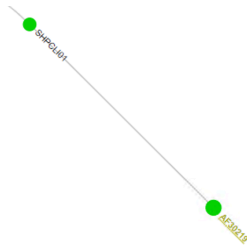


Figura 73

Posteriormente, en la misma plantilla se puede definir un hostgroup para de este modo a través de la definición de un único servicio monitorear a todos los miembros:

```
#####
#
# HOST GROUP DEFINITIONS
#
#####

# Define a hostgroup for Windows machines
# All hosts that use the windows-server template will automatically be a member of this group

define hostgroup {
    hostgroup_name     windows-servers      ; The name of the hostgroup
    alias               Windows Servers     ; Long name of the group
}
```

Figura 74

Posteriormente, a los hosts definidos anteriormente, se les debería de añadir al grupo creado:

```
define host {
    use                windows-server        ; Inherit default values from a template
    host_name          SHPCLI01             ; The name we're giving to this host
    hostgroups         windows-servers
    alias              SHPCLI01             ; A longer name associated with the host
    address            172.31.154.188      ; IP address of the host
}

define host {
    use                windows-server        ; Inherit default values from a template
    host_name          AF30219             ; The name we're giving to this host
    hostgroups         windows-servers
    alias              W10_Manu            ; A longer name associated with the host
    address            172.31.150.109     ; IP address of the host
    parents            SHPCLI01
}
```

Figura 75

Y podemos definir un servicio que monitoree a los miembros del grupo:

```
define service {
    use                generic-service      ; Inherit values from a template
    hostgroup          windows_servers      ; The name of the host the service is associated with
    service_description PING               ; The service description
    check_command      check_ping!200.0,20%!600.0,60% ; The command used to monitor the service
    check_interval     2                   ; Check the service every 5 minutes under normal conditions
    retry_interval     1                   ; Re-check the service every minute until its final/hard state is determined
}
```

Figura 76

4.4 Configuración cliente para Nagios Core en entornos Windows.

A continuación, se va a explicar cómo se configuran los diferentes clientes en entornos Windows. Véase en anexos de la memoria como se instalan.

NSCLIENT++

Una vez se dispone de NSClient++ instalado (véase en anexos), deberemos detener el agente y acceder al directorio de instalación “C:\Program Files\NSClient++” para editar el fichero de configuración nsclient.ini y habilitar los siguientes módulos [\[26\]](#):

```
; Undocumented key
CheckDisk = enabled

; Undocumented key
CheckSystem = enabled
```

Figura 77

Una vez habilitados, volvemos a iniciar el agente y se debe de acceder a nuestro servidor de monitorización y editar la plantilla con la que se gestionará este equipo para añadirlo:

```
define host {  
  
    use                windows-server        ; Inherit default values from a template  
    host_name          VDICST63              ; The name we're giving to this host  
    alias              VDICST63              ; A longer name associated with the host  
    address             VDICST63.hsp.csdt.es ; IP address of the host  
}
```

Figura 78

Una vez definido el equipo, ya podemos añadir en el mismo archivo los servicios a monitorear:

CHECK_NT

Primero de todo monitorearemos con check_nt, para ello definimos los servicios:

- Versión del cliente:

```
define service {  
  
    use                generic-service  
    host_name          VDICST63  
    service_description NSClient++ Version  
    check_command      check_nt!CLIENTVERSION  
}
```

Figura 79

- Tiempo encendido:

```
define service {  
  
    use                generic-service  
    host_name          VDICST63  
    service_description Uptime  
    check_command      check_nt!UPTIME  
}
```

Figura 80

- La carga de CPU:

```
define service {  
  
    use                generic-service  
    host_name          VDICST63  
    service_description CPU Load  
    check_command      check_nt!CPULOAD!-1 5,80,90  
}
```

Figura 81

- El uso de memoria:

```
define service {
    use                generic-service
    host_name          VDICST63
    service_description Memory Usage
    check_command      check_nt!MEMUSE!-w 80 -c 90
}
```

Figura 82

- El espacio en el disco:

```
define service {
    use                generic-service
    host_name          VDICST63
    service_description C:\ Drive Space
    check_command      check_nt!USEDISKSPACE!-l c -w 80 -c 90
}
```

Figura 83

Una vez efectuados los cambios correspondientes, se debe de reiniciar el servicio Nagios en nuestro servidor de monitorización con el comando "systemctl restart nagios.service":

```
root@monsrv01:~# sudo systemctl restart nagios.service
root@monsrv01:~#
```

Figura 84

Una vez reiniciado, si accedemos al portal web podremos ver el estado de los diferentes servicios:

Host	Service	Status	Last Check	Next Check	Output	Performance Data
VDICST63	C:\ Drive Space	OK	03-25-2024 18:47:13	0d 0h 0m 36s	1/3	c - total: 59,36 Gb - used: 38,87 Gb (65%) - free 20,50 Gb (35%)
	CPU Load	OK	03-25-2024 18:47:35	0d 0h 0m 14s	1/3	CPU Load 15% (5 min average)
	Memory Usage	OK	03-25-2024 18:45:57	0d 0h 3m 52s	1/3	Memory usage: total 5567,10 MB - used: 3037,58 MB (55%) - free: 2529,52 MB (45%)
	NSClient++ Version	OK	03-25-2024 18:45:53	0d 0h 11m 56s	1/3	NSClient++ 0.5.2.41 2018-04-26
	Uptime	OK	03-25-2024 18:47:20	0d 0h 0m 29s	1/3	System Uptime - 4 day(s) 1 hour(s) 50 minute(s)

Figura 85

Si hacemos uso de la herramienta Wireshark en el cliente podremos apreciar que la comunicación con nuestro servidor de monitorización no va encriptada:

No.	Time	Source	Destination	Protocol	Length	Info
4661	17.307242	172.31.153.202	172.31.3.74	TCP	54	12489 → 45122 [ACK] Seq=8 Ack=18 Win=2102272 Len=0
7808	32.240708	172.31.3.74	172.31.153.202	TCP	74	47010 → 12489 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=71262938 TSecr=0 WS=128
7809	32.240782	172.31.153.202	172.31.3.74	TCP	66	12489 → 47010 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 MS=256 SACK_PERM
7810	32.240951	172.31.3.74	172.31.153.202	TCP	60	47010 → 12489 [ACK] Seq=1 Ack=1 Win=64256 Len=0
7811	32.240997	172.31.3.74	172.31.153.202	TCP	72	47010 → 12489 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=18
7812	32.242227	172.31.153.202	172.31.3.74	TCP	56	12489 → 47010 [PSH, ACK] Seq=1 Ack=19 Win=2102272 Len=2
7813	32.242303	172.31.153.202	172.31.3.74	TCP	54	12489 → 47010 [FIN, ACK] Seq=3 Ack=19 Win=2102272 Len=0
7814	32.242383	172.31.3.74	172.31.153.202	TCP	60	47010 → 12489 [ACK] Seq=19 Ack=3 Win=64256 Len=0
7815	32.242412	172.31.3.74	172.31.153.202	TCP	60	47010 → 12489 [FIN, ACK] Seq=19 Ack=4 Win=64256 Len=0
7816	32.242428	172.31.153.202	172.31.3.74	TCP	54	12489 → 47010 [ACK] Seq=4 Ack=20 Win=2102272 Len=0
10610	48.535139	172.31.3.74	172.31.153.202	ICMP	98	Echo (ping) request id=0xca04, seq=1/256, ttl=64 (reply in 10611)
10611	48.535220	172.31.153.202	172.31.3.74	ICMP	98	Echo (ping) reply id=0xca04, seq=1/256, ttl=128 (request in 10610)
10833	49.539926	172.31.3.74	172.31.153.202	ICMP	98	Echo (ping) request id=0xca04, seq=2/512, ttl=64 (reply in 10834)
10834	49.540811	172.31.153.202	172.31.3.74	ICMP	98	Echo (ping) reply id=0xca04, seq=2/512, ttl=128 (request in 10833)

Figura 86

NRPE [27]

Desde el servidor Nagios Core comprobamos si el comando check_nrpe funciona con nuestro cliente:

```

root@monsrv01:/usr/local/nagios/libexec# ./check_nrpe -H vdicst63 -2
I (0.5.2.41 2018-04-26) seem to be doing fine...
root@monsrv01:/usr/local/nagios/libexec#

```

Figura 87

Y si se hace uso de la herramienta Wireshark en el cliente podremos apreciar que la comunicación con nuestro servidor de monitorización va encriptada mediante TLSv1.2 (ya que ha sido configurada, véase en anexos):

No.	Time	Source	Destination	Protocol	Length	Info
652	4.982598	172.31.3.74	172.31.153.202	TLSv1.2	571	Client Hello
653	4.984354	172.31.153.202	172.31.3.74	TLSv1.2	1601	Server Hello, Certificate, Server Key Exchange, Server Hello Done
656	4.992939	172.31.3.74	172.31.153.202	TLSv1.2	372	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
657	4.998037	172.31.153.202	172.31.3.74	TLSv1.2	280	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
658	4.998419	172.31.3.74	172.31.153.202	TLSv1.2	1119	Application Data
659	4.998612	172.31.153.202	172.31.3.74	TLSv1.2	1119	Application Data

Figura 88

Para finalizar, creamos dos servicios para monitorizar, en este caso CPU y tiempo de encendido:

```

define service {
    host_name          VDICST63
    service_description CPU
    check_command      check_nrpe_windows!check_cpu
    max_check_attempts 5
    check_interval     1
    retry_interval     1
}

define service {
    host_name          VDICST63
    service_description UPTIME_VDI
    check_command      check_nrpe_windows!check_uptime
    max_check_attempts 5
    check_interval     1
    retry_interval     1
}

```

Figura 89

Y podremos visualizarlo en el website:

VDICST63	CPU	OK	03-28-2024 14:34:21	0d 0h 5m 29s	1/5	OK: CPU load is ok.
	UPTIME_VDI	WARNING	03-28-2024 14:34:01	0d 0h 2m 49s	3/5	WARNING: uptime: 1d 22:54h,

Figura 90

NSCA

Para hacer uso de NSCA seguimos la siguiente documentación: <https://nagiosenterprises.my.site.com/support/s/article/Using-NSClient-for-Passive-Checks-68988c19> [28], se debe de definir el intervalo con el que el cliente enviara sus chequeos al servidor de monitorización y los comandos que ejecutara con sus alias:

```
[/settings/scheduler/schedules/default]
interval=1m
[/settings/scheduler/schedules]
cpu=alias_cpu
mem=alias_mem
disk=alias_disk
```

Cabe destacar que es altamente editable, por ejemplo, podemos añadir los siguientes comandos adicionales:

```
Uptime = CheckUptime MinCrit=12h ShowAll
Print Spooler Service = check_service service=spooler
```

Resultado:

```
[/settings/scheduler/schedules]
cpu = alias_cpu
mem = alias_mem
disk = alias_disk
Uptime = CheckUptime MinCrit=12h ShowAll
Print Spooler Service = check_service service=spooler
```

Figura 91

Posteriormente, de debe de indicar el nombre del equipo local, la dirección del servidor de monitorización y la clave para la autenticación:

```
[/settings/NSCA/client]
hostname=VDICST63
[/settings/NSCA/client/targets/default]
address=172.31.3.74
encryption=3
password=PasswordSecure
```

Una vez configurado, procedemos a definir el host en nuestro servidor de monitorización indicándole que no se efectuarán chequeos activos y si pasivos:

```
define host {
    use windows-server
    host_name VDICST63
    alias VDICST63
    address VDICST63
    active_checks_enabled 0 ; Active host checks are enabled
    passive_checks_enabled 1 ; Passive host checks are enabled/accepted
}
```

Figura 92

Y definimos los servicios que queremos monitorear. En service_description se debe de poner el nombre del servicio y en check_comand el comando a utilizar y pasarle como argumento el alias o el comando. Por ejemplo, tenemos cpu=alias_cpu:

```
[/settings/scheduler/schedules]
cpu=alias_cpu
```

Figura 93

El servicio se debe de configurar así:

```
define service {
    use                generic-service
    host_name          VDICST63
    service_description cpu
    check_command      check_nrpe_windows!alias_cpu
    active_checks_enabled 0 ; Active service checks are enabled
    passive_checks_enabled 1 ; Passive service checks are enabled/accepted
}
```

Figura 94

Configuramos el resto de los servicios:

```
define service {
    use                generic-service
    host_name          VDICST63
    service_description mem
    check_command      check_nrpe_windows!alias_mem
    active_checks_enabled 0 ; Active service checks are enabled
    passive_checks_enabled 1 ; Passive service checks are enabled/accepted
}
```

Figura 95

```
define service {
    use                generic-service
    host_name          VDICST63
    service_description disk
    check_command      check_nrpe_windows!alias_disk
    active_checks_enabled 0 ; Active service checks are enabled
    passive_checks_enabled 1 ; Passive service checks are enabled/accepted
}
```

Figura 96

```
define service {
    use                generic-service
    host_name          VDICST63
    service_description Uptime
    check_command      check_nrpe_windows!CheckUptime MinCrit=12h ShowAll
    active_checks_enabled 0 ; Active service checks are enabled
    passive_checks_enabled 1 ; Passive service checks are enabled/accepted
}
```

Figura 97

```
define service {
    use                generic-service
    host_name          VDICST63
    service_description Print Spooler Service
    check_command      check_nrpe_windows!check_service service=spooler
    active_checks_enabled 0 ; Active service checks are enabled
    passive_checks_enabled 1 ; Passive service checks are enabled/accepted
}
```

Figura 98

Finalmente, se debe de autorizar la IP del cliente o el nombre de equipo a comunicarse con el protocolo NSCA:

```

/etc/xinetd.d/nsca - root@172.31.3.74 - Editor - WinSCP
# default: on
# description: NSCA (Nagios Service Check Acceptor)
service nsca
{
    flags          = REUSE
    socket_type    = stream
    wait          = no
    user           = nagios
    group         = nagcmd
    server        = /usr/local/nagios/bin/nsca
    server_args   = -c /usr/local/nagios/etc/nsca.cfg --inetd
    log_on_failure += USERID
    disable       = no
    only_from     = 172.31.3.74 172.31.153.202
}

```

Figura 99

Se reinician servicios de Nagios y NSCA y se verifica correcto funcionamiento. Los símbolos “?” indican que son chequeos pasivos y cada minuto se actualizan según lo configurado:

VDICST63	?	Print Spooler Service	?	OK	03-29-2024 18:45:53	0d 0h 0m 22s+	1/3	OK: All 1 service(s) are ok.
		Uptime	?	OK	03-29-2024 18:45:53	0d 0h 0m 22s+	1/3	OK: uptime: 3d 03:0h. boot: 2024-03-26 14:39:05 (UTC)
		cpu	?	OK	03-29-2024 18:45:53	0d 0h 3m 41s	1/3	OK: CPU load is ok.
		disk	?	OK	03-29-2024 18:45:53	0d 0h 3m 41s	1/3	OK: All 2 drive(s) are ok.
		mem	?	OK	03-29-2024 18:45:53	0d 0h 3m 41s	1/3	OK: committed = 3.045GB, physical = 2.751GB

Figura 100

NCPA

Una vez instalado NCPA (véase en anexos), se procede a seguir la documentación para su configuración: <https://assets.nagios.com/downloads/nagioscore/docs/nagioscore/3/en/monitoring-windows.html> [29].

En Nagios Core definimos el host y los servicios a monitorear en la correspondiente plantilla:

```

define host {
    use                windows-server          ; Inherit default values from a template
    host_name          VDICST14                ; The name we're giving to this host
    alias              VDICST14                ; A longer name associated with the host
    address            VDICST14.hsp.csdt.es    ; IP address of the host
}

```

Figura 101

Check del disco:

```

define service {
    use                generic-service
    host_name          VDICST14
    service_description C:\ Drive Space
    check_command      check_ncpa!-t 'PasswordSecure' -P 5693 -M 'disk/logical/C:/free' --warning 10: --critical 5: -u G
}

```

Figura 102

Check de la CPU:

```
define service {
    use                generic-service
    host_name          VDICST14
    service_description CPU Usage
    check_command      check_ncpa!-t 'PasswordSecure' -P 5693 -M cpu/percent -w 20 -c 40 -q 'aggregate=avg'
    max_check_attempts 5
    check_interval     5
    retry_interval     1
    check_period       24x7
    notification_interval 60
    notification_period 24x7
    contacts           nagiosadmin
    register           1
}
```

Figura 103

Check de de la memoria:

```
define service {
    use                generic-service
    host_name          VDICST14
    service_description Memory Usage
    check_command      check_ncpa!-t 'PasswordSecure' -P 5693 -M memory/virtual -w 50 -c 80 -u G
    max_check_attempts 5
    check_interval     5
    retry_interval     1
    check_period       24x7
    notification_interval 60
    notification_period 24x7
    contacts           nagiosadmin
    register           1
}
```

Figura 104

Check del contador de procesos:

```
define service {
    use                generic-service
    host_name          VDICST14
    service_description Process Count
    check_command      check_ncpa!-t 'PasswordSecure' -P 5693 -M processes -w 350 -c 400
    max_check_attempts 5
    check_interval     5
    retry_interval     1
    check_period       24x7
    notification_interval 60
    notification_period 24x7
    contacts           nagiosadmin
    register           1
}
```

Figura 105

Está pendiente:

VDICST14	C:\ Drive Space	PENDING	N/A	0d 0h 0m 13s+	1/3	Service check scheduled for Wed Apr 3 18:42:18 CEST 2024
	CPU Usage	PENDING	N/A	0d 0h 0m 13s+	1/5	Service check scheduled for Wed Apr 3 18:41:27 CEST 2024
	Memory Usage	PENDING	N/A	0d 0h 0m 13s+	1/5	Service check scheduled for Wed Apr 3 18:42:21 CEST 2024
	Process Count	PENDING	N/A	0d 0h 0m 13s+	1/5	Service check scheduled for Wed Apr 3 18:44:41 CEST 2024

Figura 106

Y ya tenemos los servicios monitoreados:

VDICST14	C:\ Drive Space	OK	04-03-2024 18:42:18	0d 0h 3m 51s+	1/3	OK: Free was 23.43 GB
	CPU Usage	OK	04-03-2024 18:42:40	0d 0h 2m 9s	1/5	OK: Percent was 0.00 %
	Memory Usage	WARNING	04-03-2024 18:43:21	0d 0h 0m 15s	1/5	WARNING: Memory usage was 69.90 % (Available: 1.29 GB, Total: 4.29 GB, Free: 1.29 GB, Used: 3.00 GB)
	Process Count	OK	04-03-2024 18:42:40	0d 0h 3m 51s+	1/5	OK: Process count was 195

Figura 107

Una vez explicado, procedemos con NRDP. Para hacer uso, se ha de modificar el servicio para deshabilitar los checks activos y habilitar los pasivos de servicio “CPU Usage” por ejemplo:

```
define service {
    use                generic-service
    host_name          VDICST14
    service_description CPU Usage
    check_command      check_ncpa!-t 'PasswordSecure' -P 5693 -M cpu/percent -w 20 -c 40 -q 'aggregate=avg'
    active_checks_enabled 0 ; Active service checks are enabled
    passive_checks_enabled 1 ; Passive service checks are enabled/accepted
}
```

Figura 108

Una vez reiniciamos NCPA del cliente y el servicio de Nagios en el servidor de monitorización ya pasa a funcionar. De forma predeterminada se envía un check cada 5 minutos:

VDICST14	C:\ Drive Space	OK	04-06-2024 10:08:59	2d 15h 30m 53s	1/3	OK: Free was 23.37 GB
	CPU Usage	?	04-06-2024 10:11:09	0d 0h 23m 57s	1/3	OK: Percent was 39.40 %

Figura 109

Si queremos que este envío pasivo se haga a cada minuto, por ejemplo, se debe modificar el archivo anterior i poner el tiempo “60” (de 60 segundos):

```
#
# AUTO GENERATED NRDP CONFIG FROM WINDOWS INSTALLER
#

[passive checks]

# Host check - This is to stop "pending check" status in Nagios
%HOSTNAME%|__HOST__ = system/agent_version

# Service checks
%HOSTNAME%|CPU Usage |60 = cpu/percent --warning 80 --critical 90 --aggregate avg
%HOSTNAME%|Disk Usage = disk/logical/C:/used_percent --warning 80 --critical 90 --units Gi
%HOSTNAME%|Swap Usage = memory/swap --warning 60 --critical 80 --units Gi
%HOSTNAME%|Memory Usage = memory/virtual --warning 80 --critical 90 --units Gi
%HOSTNAME%|Process Count = processes --warning 300 --critical 400
```

Figura 110

Posteriormente reiniciamos el cliente NCPA y verificamos el funcionamiento:

VDICST14	C:\ Drive Space	OK	04-06-2024 10:08:59	2d 15h 34m 33s	1/3	OK: Free was 23.37 GB
	CPU Usage	?	04-06-2024 10:16:40	0d 0h 0m 11s	1/3	CRITICAL: Percent was 100.00 %

Figura 111

Y verificamos que se efectúa:

VDICST14	C:\ Drive Space	OK	04-06-2024 10:08:59	2d 15h 35m 22s	1/3	OK: Free was 23.37 GB
	CPU Usage	?	04-06-2024 10:17:37	0d 0h 0m 3s	1/3	OK: Percent was 3.00 %

Figura 112

4.5 Configuración clientes para Nagios Core en entornos UNIX

A continuación, se va a explicar cómo se configuran los diferentes plugins en entornos Linux. Véase la instalación en anexos.

NRPE [20]

Desde el servidor de monitorización se comprueba la comunicación con el nuevo cliente instalado:

```
root@monsrv01:/usr/local/nagios/libexec# ./check_nrpe -H ubuntu manu
NRPE v4.0.3
```

Figura 113

Se va a monitorear el espacio en el disco. Para ello, primero tenemos que ver que disco queremos monitorear de nuestro cliente Linux, se ejecuta “df -h /” y nos proporciona nombre del disco que es “/dev/mapper/Ubuntu--vg-ubuntu--lv”:

```
root@ubuntumanu:/# df -h /
Filesystem                Size      Used Avail Use% Mounted on
/dev/mapper/ubuntu--vg-ubuntu--lv 19G  7,9G  9,8G  45% /
```

Figura 114

Posteriormente, se edita el fichero de configuración del cliente nrpe para descomentar la línea del comando check_disk y añadirle los parámetros del disco que se quiere monitorear:

```
command[check_hda1]=/usr/lib/nagios/plugins/check_disk -w 20% -c 10% -p /dev/mapper/ubuntu--vg-ubuntu--lv
```

Figura 115

Una vez modificado esto, debemos de reiniciar el servicio con el comando “sudo systemctl restart nagios-nrpe-server” y se procede a definir el host y el servicio a monitorear para comprobar el correcto funcionamiento en una de las plantillas:

```
define host {
    use linux-server ; Inherit default values from a template
    host_name ubuntu manu ; The name we're giving to this host
    alias ubuntu manu ; A longer name associated with the host
    address ubuntu manu ; IP address of the host
}

define service {
    use local-service
    host_name ubuntu manu
    service_description Check SPACE NRPE
    check_command check_nrpe!check_hda1 15% 10% /
    check_interval 1
    retry_interval 1
    contact_groups admins
}
}
```

Figura 116

Una vez definidos el host y el servidor, se debe de crear el correspondiente comando NRPE en la plantilla de comandos de nuestro servidor de monitorización:

```
define command {
    command_name check_nrpe
    command_line $USER1$/check_nrpe -H $HOSTADDRESS$ -c $ARG1$ $ARG2$
}
}
```

Figura 117

Y se verifica la monitorización:

```
ubuntumanu Check SPACE NRPE OK 03-29-2024 11:42:10 0d 0h 57m 44s 1/4 DISK OK - free space: /var/tmp 9943 MB (55% inode=90%):
```

Figura 118

NSCA [30]

Definimos el host cliente en la plantilla correspondiente del servidor de monitorización:

```
define host {  
  
    use                linux-server          ; Inherit default values from a template  
    host_name          ubuntu              ; The name we're giving to this host  
    alias              ubuntu              ; A longer name associated with the host  
    address            ubuntu              ; IP address of the host  
  
}
```

Figura 119

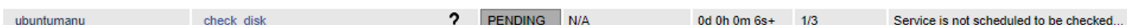
Una vez definido el host, definimos el servicio de ejemplo que vamos a monitorear:

```
define service {  
    use                generic-service  
    host_name          ubuntu  
    service_description check_disk  
    check_command      check_nrpe!check_disk -w 20% -c 10%  
    active_checks_enabled 0 ; Active service checks are enabled  
    passive_checks_enabled 1 ; Passive service checks are enabled/accepted  
  
}
```

Figura 120

Hemos definido el comando `check_nrpe` porque de este modo si queremos activar el check activo podemos hacerlo ya que el cliente tiene instalado el plugin `nrpe`.

Y ya tenemos al servidor de monitorización pendiente de recibir respuesta por parte del cliente:



ubuntumanu	check_disk	?	PENDING	N/A	0d 0h 0m 6s+	1/3	Service is not scheduled to be checked...
------------	------------	---	---------	-----	--------------	-----	---

Figura 121

Una vez llegados a este punto, para que desde el cliente se envíe el estado del servicio se puede ejecutar el siguiente comando: “echo

```
"ubuntumanu;check_disk;0;$(/usr/lib/nagios/plugins/check_disk -w 20% -c 10%)"| /usr/sbin/send_nsc -d ';' -c /etc/send_nsc.cfg -H 172.31.3.74”
```

Donde lo que se hace es:

- Echo:
 - “ubutumanu”: imprime el nombre de host que hemos definido en el servidor de monitorización.
 - “check_disk”: imprime el nombre del servicio que hemos definido en el servidor de monitorización.
 - “0”: Este parámetro indica si el estado es normal, warning o critical. Por ahora enviaremos el estado normal que es el 0.
 - `$(/usr/lib/nagios/plugins/check_disk -w 20% -c 10%)`: ejecutamos el check del disco para poder enviar el resultado.
- `|usr/sbin/send_nsc -d ';' -c /etc/send_nsc.cfg -H 172.31.3.74`: concatenamos la impresión anterior para el envío al servidor de monitorización.

Ejecutamos el comando anterior:

```
root@ubuntumanu:/# echo "ubuntumanu;check_disk;0;$(/usr/lib/nagios/plugins/check_disk -w 20% -c 10%)" | /usr/sbin/send_nsca -d ';' -c /etc/send_nsca.cfg -H 172.31.3.74
1 data packet(s) sent to host successfully.
```

Figura 122

Como podemos apreciar, se notifica de que se ha enviado. Si comprobamos la página web del servidor de monitorización:

ubuntumanu	check_disk	?	OK	03-31-2024 17:42:25	0d 0h 4m 58s	1/3	DISK OK - free space: /dev 1899 MB (100% inode=99%); / 993 MB (55% inode=90%); /boot 1576 MB (86% inode=99%);
------------	------------	---	----	---------------------	--------------	-----	---

Figura 123

El problema de este comando es el estado, que siempre enviamos que está correcto y puede que no lo esté. Para ello lo podemos solucionar mediante el siguiente script que he diseñado:

```
echo "ubuntumanu;check_disk;$(if echo
"$(/usr/lib/nagios/plugins/check_disk -w 20% -c 10%)" | grep -q
"WARNING"; then
    echo 1
elif echo "$(/usr/lib/nagios/plugins/check_disk -w 20% -c 10%)" | grep
-q "CRITICAL"; then
    echo 2
else
    echo 0
fi);$(/usr/lib/nagios/plugins/check_disk -w 20% -c
10%)"|/usr/sbin/send_nsca -d ';' -c /etc/send_nsca.cfg -H 172.31.3.74
```

Vamos a simular un warning modificando los parámetros del check_disk:

```
echo "ubuntumanu;check_disk;$(if echo
"$(/usr/lib/nagios/plugins/check_disk -w 100% -c 10%)" | grep -q
"WARNING"; then
    echo 1
elif echo "$(/usr/lib/nagios/plugins/check_disk -w 100% -c 10%)" |
grep -q "CRITICAL"; then
    echo 2
else
    echo 0
fi);$(/usr/lib/nagios/plugins/check_disk -w 100% -c
10%)"|/usr/sbin/send_nsca -d ';' -c /etc/send_nsca.cfg -H 172.31.3.74
```

Resultado:

```
root@ubuntumanu:/# echo "ubuntumanu;check_disk;$(if echo "$(/usr/lib/nagios/plugins/check_disk -w 100% -c 10%)" | grep -q "WARNING"; t
hen
    echo 1
elif echo "$(/usr/lib/nagios/plugins/check_disk -w 100% -c 10%)" | grep -q "CRITICAL"; then
    echo 2
else
    echo 0
fi);$(/usr/lib/nagios/plugins/check_disk -w 100% -c 10%)"|/usr/sbin/send_nsca -d ';' -c /etc/send_nsca.cfg -H 172.31.3.74
1 data packet(s) sent to host successfully.
root@ubuntumanu:/#
```

Figura 124

ubuntumanu	check_disk	?	WARNING	03-31-2024 17:46:43	0d 0h 0m 23s	1/3	DISK WARNING - free space: /dev 1899 MB (100% inode=99%); 9936 MB (55% inode=90%); /boot 1576 MB (86% inode=99%);
------------	------------	---	---------	---------------------	--------------	-----	---

Figura 125

Y ahora lo mismo, pero con critical:

```
echo "ubuntumanu;check_disk;$(if echo
"$(/usr/lib/nagios/plugins/check_disk -w 20% -c 100%)" | grep -q
"WARNING"; then
    echo 1
elif echo "$(/usr/lib/nagios/plugins/check_disk -w 20% -c 100%)" |
grep -q "CRITICAL"; then
    echo 2
else
    echo 0
fi);$(/usr/lib/nagios/plugins/check_disk -w 20% -c
100%)"|/usr/sbin/send_nsc -d ';' -c /etc/send_nsc.cfg -H
172.31.3.74
```

Resultado:

```
root@ubuntumanu:/# echo "ubuntumanu;check_disk;$(if echo "$(/usr/lib/nagios/plugins/check_disk -w 20% -c 100%)" | grep -q "WARNING"; t
hen
    echo 1
elif echo "$(/usr/lib/nagios/plugins/check_disk -w 20% -c 100%)" | grep -q "CRITICAL"; then
    echo 2
else
    echo 0
fi);$(/usr/lib/nagios/plugins/check_disk -w 20% -c 100%)"|/usr/sbin/send_nsc -d ';' -c /etc/send_nsc.cfg -H 172.31.3.74
1 data packet(s) sent to host successfully.
root@ubuntumanu:/#
```

Figura 126

ubuntumanu	check_disk	?	CRITICAL	03-31-2024 17:48:00	0d 0h 0m 27s	2/3	DISK CRITICAL - free space: /dev 1899 MB (100% inode=99%); 9936 MB (55% inode=90%); /boot 1576 MB (86% inode=99%);
------------	------------	---	----------	---------------------	--------------	-----	--

Figura 127

Una vez explicado, se procede a crear un ejecutable con el script en el cliente:

```
/usr/local/check_disk.sh - root@172.31.150.65 - Editor - WinSCP
Codificación - Color - ?
echo "ubuntumanu;check_disk;$(if echo "$(/usr/lib/nagios/plugins/check_disk -w 20% -c 10%)" | grep -q "WARNING"; then
    echo 1
elif echo "$(/usr/lib/nagios/plugins/check_disk -w 20% -c 10%)" | grep -q "CRITICAL"; then
    echo 2
else
    echo 0
fi);$(/usr/lib/nagios/plugins/check_disk -w 20% -c 10%)"|/usr/sbin/send_nsc -d ';' -c /etc/send_nsc.cfg -H 172.31.3.74
```

Figura 128

Establecemos permisos y ejecutamos para verificar el correcto funcionamiento:

```
root@ubuntumanu:/usr/local# chmod 777 check_disk.sh
root@ubuntumanu:/usr/local# ./check_disk.sh
1 data packet(s) sent to host successfully.
```

Figura 129

Y programamos un check a cada minuto por ejemplo con crontab:

```

/etc/crontab - root@172.31.150.65 - Editor - WinSCP
Codificación Color
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab`
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
# You can also override PATH, but by default, newer versions inherit it from the environment
#PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# Example of job definition:
# ----- minute (0 - 59)
# | | | | | ----- hour (0 - 23)
# | | | | | | | ----- day of month (1 - 31)
# | | | | | | | | | ----- month (1 - 12) OR jan,feb,mar,apr ...
# | | | | | | | | | ----- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# | | | | | | | | |
# * * * * * user-name command to be executed
17 * * * * root cd / && run-parts --report /etc/cron.hourly
25 6 * * * root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6 * * 7 root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6 1 * * root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
* * * * * root /usr/local/check_disk.sh
#

```

Figura 130

Se verifica correcto funcionamiento:

```

ubuntumanu check_disk ? OK 03-31-2024 18:02:10 0d 0h 0m 59s 1/3 DISK OK - free space: /dev 1899 MB (100% inode=99%); / 993 MB (55% inode=90%); /boot 1576 MB (86% inode=99%);

```

Figura 131

NCPA [31]

Definimos el host en nuestro servidor de monitorización:

```

define host {
    use linux-server ; Inherit default values from a template
    host_name ubuntumanu ; The name we're giving to this host
    alias ubuntumanu ; A longer name associated with the host
    address ubuntumanu ; IP address of the host
}

```

Figura 132

Para construir los servicios que queremos monitorear, desde nagios podemos ejecutar desde la ubicación del script "/usr/local/nagios/libexec#" el comando: ". /check_ncpa.py -H ubuntumanu -t 'PasswordSecure' -P 5693 -list" y podremos ver que podemos monitorear. En este caso vamos al apartado del disco:

```

"disk": {
  "mount": {
    "|run|snapd|ns|lxd.mnt": {
      "device_name": [
        "nsfs"
      ],
      "fstype": "nsfs",
      "opts": "rw"
    }
  },
  "logical": {
    "|": {
      "used_percent": [
        40.5,
        "%"
      ],
      "used": [
        7.12,
        "GiB"
      ],
      "max_file_length": 255,
      "inodes_used": [
        86005,
        "inodes"
      ],
      "free": [
        10.45,
        "GiB"
      ],
    }
  }
}

```

Figura 133

Y podemos construir su comando para monitorear ese servicio: “check_ncpa!-t ‘token’ -P 5693 -M ‘disk/logical/|/free’ --warning 10: --critical 5: -u G”

```

define service {
    use generic-service
    host_name ubuntu manu
    service_description Check DISC NCPA Free Space
    check_command check_ncpa!-t 'PasswordSecure' -P 5693 -M 'disk/logical/|/free' --warning 10: --critical 5: -u G
}

```

Figura 134

Y lo mismo para la memoria, por ejemplo:

```

define service {
    use local-service ; Name of service template to use
    host_name ubuntu manu
    service_description Check NCPA Memory
    check_command check_ncpa!-t 'PasswordSecure' -P 5693 -M memory/virtual -w 50 -c 80 -u G
}

```

Figura 135

Y podremos apreciar su estado:

ubuntumanu	Check DISC NCPA Free Space	OK	04-03-2024 19:03:31	0d 0h 1m 31s	1/3	OK: Free was 10.40 GB
	Check NCPA Memory	OK	04-03-2024 19:02:59	0d 0h 3m 3s	1/4	OK: Memory usage was 13.40 % (Available: 3.55 GB, Total: 4.10 GB, Free: 2.97 GB, Used: 0.31 GB)

Figura 136

Una vez explicado el envío de checks activos, vamos a explicar cómo configurar los pasivos en un entorno UNIX. Para ello, primero de todo, editamos el fichero “usr/local/ncpa/etc/ncpa.cfg.d/nrdp” Y le establecemos el siguiente contenido, donde enviaremos el check pasivo de CPU Usage cada 60 segundos:

```

/usr/local/nagios/etc/nagios.cfg.d/nrdp.cfg - root@172.31.150.65 - Editor - WinSCP
#
#      Basic example of Passive checks being defined
#
[passive checks]

%HOSTNAME%|__HOST__ = system/agent_version
%HOSTNAME%|Disk Usage = disk/logical/C:|/used_percent --warning 80 --critical 90 --units Gi
%HOSTNAME%|CPU Usage |60 = cpu/percent --warning 60 --critical 80 --aggregate avg
%HOSTNAME%|Swap Usage = memory/swap --warning 60 --critical 80 --units Gi
%HOSTNAME%|Memory Usage = memory/virtual --warning 80 --critical 90 --units Gi
%HOSTNAME%|Process Count = processes --warning 300 --critical 400

```

Figura 137

Posteriormente, reiniciamos los servicios y en el servidor de monitorización Nagios, en su correspondiente plantilla definimos el servicio CPU Usage habilitando los checks pasivos y deshabilitando los pasivos:

```

define service {
    use                generic-service
    host_name          ubuntu manu
    service_description CPU Usage
    check_command      check_ncpa!-t 'PasswordSecure' -P 5693 -M cpu/percent -w 20 -c 40 -q 'aggregate=avg'
    active_checks_enabled 0 ; Active service checks are enabled
    passive_checks_enabled 1 ; Passive service checks are enabled/accepted
}

```

Figura 138

Reiniciamos servicios y verificamos:

ubuntumanu	CPU Usage	?	OK	04-06-2024 11:23:36	0d 0h 44m 17s+	1/3	OK: Percent was 55.45 %
------------	-----------	---	----	---------------------	----------------	-----	-------------------------

Figura 139

Y al minuto vuelve a enviar los resultados:

ubuntumanu	CPU Usage	?	OK	04-06-2024 11:24:36	0d 0h 1m 13s	1/3	OK: Percent was 1.00 %
------------	-----------	---	----	---------------------	--------------	-----	------------------------

Figura 140

4.6 Monitorización de servicios que no requieran de clientes.

Con Nagios Core se pueden monitorizar gran cantidad de protocolos que no requieren tener agentes instalados. Para ello se hace uso de comandos que verifican el estado de puertos o servicios de equipos que deberían de estar abiertos. En este caso, podríamos monitorear si el puerto 443 por TCP está abierto (puerto web HTTPS), o el SSH (por el puerto 22).

Para poder monitorear todo esto, disponemos del fichero “usr/local/nagios/etc/objects/commands.cfg”, donde de forma predeterminada, en el fichero comandos de Nagios Core tenemos esto comandos predefinidos para monitorear protocolos, donde se hace la llamada al script que verifica el servicio y se le pasan como parámetros el equipo a monitorear y los argumentos requeridos.

Los Scripts están ubicados en “/usr/local/nagios/libexec” y para efectuar las comprobaciones, vamos a hacer uso de los scripts directamente:

- FTP: El protocolo FTP (File Transfer Protocol) se utiliza para la transferencia de ficheros entre cliente y servidor.

Véase en la siguiente ilustración como viene definido el monitoreo del servicio en Nagios Core:

```
define command {  
  
    command_name    check_ftp  
    command_line    $USER1$/check_ftp -H $HOSTADDRESS$ $ARG1$  
}
```

Figura 141

Véase la siguiente ilustración de ejemplo de uso del Script de monitorización con un servidor FTP instalado en un cliente de prueba:

```
root@monsrv01:/usr/local/nagios/libexec# ./check_ftp AF30219  
FTP OK - 0,008 second response time on port 21 [220-FileZilla Server 1.7.2  
220 Please visit https://filezilla-project.org/]|time=0,007557s;;;0,000000;10,000000  
root@monsrv01:/usr/local/nagios/libexec#
```

Figura 142

- SNMP: El protocolo SNMP (Simple Network Management Protocol) es utilizado para administración y monitorización de dispositivos de red:

Véase en la siguiente ilustración como viene definido el monitoreo del servicio en Nagios Core:

```
define command {  
  
    command_name    check_snmp  
    command_line    $USER1$/check_snmp -H $HOSTADDRESS$ $ARG1$  
}
```

Figura 143

Véase la siguiente ilustración de ejemplo de uso del Script de monitorización con un cliente SNMP de la comunidad “public” donde queremos visualizar el tiempo de encendido:

```
root@monsrv01:/usr/local/nagios/libexec# ./check_snmp HSP001-1 -C public -o sysUpTime.0  
SNMP OK - Timeticks: (570546159) 66 days, 0:51:01.59 | DISMAN-EXPRESSION-MIB::sysUpTimeInstance=570546159
```

Figura 144

- HTTP: El protocolo HTTP (Hypertext Transfer Protocol) es un protocolo de comunicación utilizado para transferir información en la World Wide

Web. Este protocolo no utiliza SSL/TLS para la encriptación de los datos de navegación:

Véase en la siguiente ilustración como viene definido el monitoreo del servicio en Nagios Core:

```
define command {  
  
    command_name    check_http  
    command_line    $USER1$/check_http -I $HOSTADDRESS$ $ARG1$  
}
```

Figura 145

Véase la siguiente ilustración de ejemplo de uso del Script de monitorización con un servidor HTTP instalado en un cliente:

```
root@monsrv01:/usr/local/nagios/libexec# ./check_http monsrv01  
HTTP OK: HTTP/1.1 302 Found - 465 bytes in 0,001 second response time |time=0,000987s;;;0,000000 size=465B;;;0
```

Figura 146

- SSH: El protocolo SHH (Secure Shell) es utilizado para establecer comunicaciones encriptadas con clientes.

Véase en la siguiente ilustración como viene definido el monitoreo del servicio en Nagios Core:

```
define command {  
  
    command_name    check_ssh  
    command_line    $USER1$/check_ssh $ARG1$ $HOSTADDRESS$  
}
```

Figura 147

Véase la siguiente ilustración de ejemplo de uso del Script de monitorización con un servidor SSH instalado en un cliente:

```
root@monsrv01:/usr/local/nagios/libexec# ./check_ssh monsrv01  
SSH OK - OpenSSH_8.9p1 Ubuntu-3ubuntu0.6 (protocol 2.0) | time=0,025626s;;;0,000000;10,000000
```

Figura 148

- Ping: Se hace uso del protocolo ICMP para verificar la conectividad en la red de un equipo:

Véase en la siguiente ilustración como viene definido el monitoreo del servicio en Nagios Core:


```

define command {
    command_name    check_ping
    command_line    $USER1$/check_ping -H $HOSTADDRESS$ -w $ARG1$ -c $ARG2$ -p 5
}

```

Figura 149

Véase la siguiente ilustración de ejemplo de uso del Script de monitorización:

```

root@monsrv01:/usr/local/nagios/libexec# ./check_ping -H monsrv01 -w 3000.0,80% -c 5000.0,100% -p 5
PING OK - Packet loss = 0%, RTA = 0.09 ms|rta=0.086000ms;3000.000000;5000.000000;0.000000 pl=0%;80;100;0

```

Figura 150

- Protocolos de recepción de correo IMAP/POP: Los protocolos IMAP y POP (Internet Message Access Protocol y Post Office Protocol) son protocolos de recepción de correo electrónico. La principal diferencia entre ellos es que IMAP sincroniza los correos electrónicos (de cualquier carpeta) con el servidor de correo y POP no. Con POP los correos se bajan en local y posteriormente se eliminan del servidor de correo al momento o a los pocos días.

Véase en la siguiente ilustración como vienen definidos el monitoreo de los servicios en Nagios Core:

```

define command {
    command_name    check_pop
    command_line    $USER1$/check_pop -H $HOSTADDRESS$ $ARG1$
}

define command {
    command_name    check_imap
    command_line    $USER1$/check_imap -H $HOSTADDRESS$ $ARG1$
}

```

Figura 151

Véase la siguiente ilustración de ejemplo de uso del Script de monitorización:

```

root@monsrv01:/usr/local/nagios/libexec# ./check_imap exsrv01
IMAP OK - 0,004 second response time on port 143 [* OK The Microsoft Exchange IMAP4 service is ready.][time=0,003595s;;;0,000000;10,000000]
root@monsrv01:/usr/local/nagios/libexec# ./check_pop exsrv01
POP OK - 0,004 second response time on port 110 [+OK The Microsoft Exchange POP3 service is ready.][time=0,004231s;;;0,000000;10,000000]
root@monsrv01:/usr/local/nagios/libexec#

```

Figura 152

- SMTP: el protocolo SMTP (Simple Mail Transfer Protocol), es el protocolo utilizado para el envío de correos electrónicos:

Véase en la siguiente ilustración como viene definido el monitoreo del servicio en Nagios Core:

```
define command {
    command_name    check_smtp
    command_line    $USER1$/check_smtp -H $HOSTADDRESS$ $ARG1$
}
```

Figura 153

Véase la siguiente ilustración de ejemplo de uso del Script de monitorización con un servidor SMTP instalado en un servidor:

```
root@monsrv01:/usr/local/nagios/libexec# ./check_smtp exsrv01
SMTP OK - 0,004 sec. response time|time=0,003743s;;;0,000000
```

Figura 154

- Protocolos de comunicación (TCP y UDP): Los protocolos TCP y UDP (Transmission Control Protocol y User Datagram Protocol) son los protocolos más utilizados para el transporte de los datos. Son la base. La principal diferencia es que TCP tiene control de los paquetes y UDP no. UDP se utiliza frecuentemente en telefonía (ya que no se necesita verificar la integridad de los paquetes) y TCP por ejemplo en la descarga de un fichero.

Véase en la siguiente ilustración como vienen definidos el monitoreo de los servicios en Nagios Core:

```
define command {
    command_name    check_tcp
    command_line    $USER1$/check_tcp -H $HOSTADDRESS$ -p $ARG1$ $ARG2$
}

define command {
    command_name    check_udp
    command_line    $USER1$/check_udp -H $HOSTADDRESS$ -p $ARG1$ $ARG2$
}
```

Figura 155

Véase la siguiente ilustración de ejemplo de uso del Script de monitorización:

```
root@monsrv01:/usr/local/nagios/libexec# ./check_udp -H 172.31.2.4 -p 53 -s "google.es A" -e "172.217.19.35"
UDP WARNING - Unexpected response from host/socket: google.es A|time=2,001932s;;;0,000000;10,000000
root@monsrv01:/usr/local/nagios/libexec# ./check_tcp -H monsrv01 -p 443
TCP OK - 0,000 second response time on monsrv01 port 443|time=0,000294s;;;0,000000;10,000000
root@monsrv01:/usr/local/nagios/libexec# ./check_udp -H 172.31.2.4 -p 53 -s "google.es A" -e "172.217.19.35"
CRITICAL - Socket timeout
root@monsrv01:/usr/local/nagios/libexec#
```

Figura 156

4.7 Monitorización con SNMP en Switch

Para monitorear mediante SNMP en un Switch [33], se puede hacer uso de la herramienta MIB Browser para identificar los OID y ver que se puede monitorear. Se puede descargar la aplicación desde la siguiente web: <https://www.ireasoning.com/mibbrowser.shtml>. Una vez descargado e instalado el software, podemos conectarnos a un equipo indicándole la comunidad y versión y establecemos la conexión:

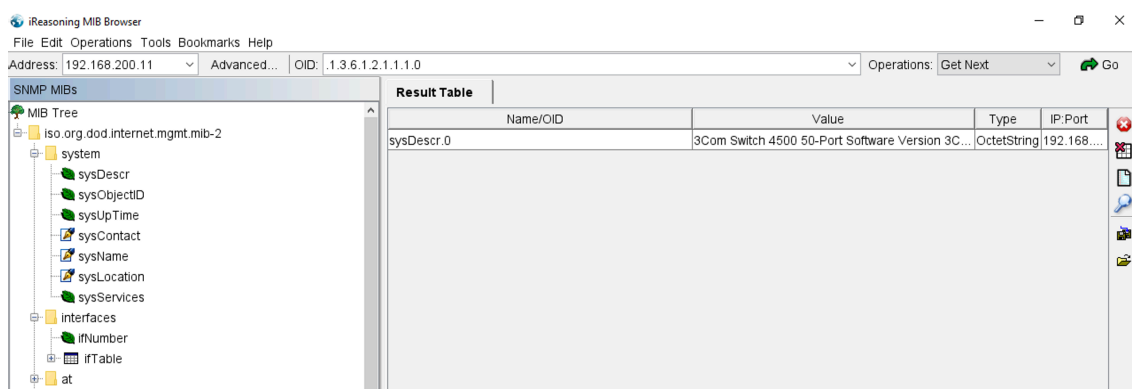


Figura 157

Por ejemplo, desde Nagios Core podemos monitorear el sysDescr.0 (como aparece en el MIB Browser) y nos proporcionará la misma información:

```
root@monsrv01:/usr/local/nagios/libexec# ./check_snmp 192.168.200.11 -C public -o sysDescr.0
SNMP OK - 3Com Switch 4500 50-Port Software Version 3Com OS V3.03.02s168p21 | SNMPv2-MIB::sysDescr.0=3
```

Figura 158

A partir del uso de esta herramienta podemos monitorear cualquier parámetro que aparezca en la tabla. Por ejemplo, si queremos identificar el OID de un puerto, podemos hacer uso del apartado ifDescr para localizarlo:

Address: 192.168.200.11 Advanced... OID: 1.3.6.1.2.1.2.2.1.2 Operations: Get Next

SNMP MIBs

- interfaces
 - ifNumber
 - ifTable
 - ifEntry
 - ifIndex
 - ifDescr**
 - ifType
 - ifMtu
 - ifSpeed
 - ifPhysAddress
 - ifAdminStatus
 - ifOperStatus
 - ifLastChange
 - ifInOctets
 - ifInUcastPkts
 - ifInNUcastPkts
 - ifInDiscards
 - ifInErrors
 - ifInUnknownProtos
 - ifOutOctets
 - ifOutUcastPkts

Name/OID	Value	Type
ifDescr.503316714	Ethernet8/0/27	OctetString
ifDescr.503316722	Ethernet8/0/28	OctetString
ifDescr.503316730	Ethernet8/0/29	OctetString
ifDescr.503316738	Ethernet8/0/30	OctetString
ifDescr.503316746	Ethernet8/0/31	OctetString
ifDescr.503316754	Ethernet8/0/32	OctetString
ifDescr.503316762	Ethernet8/0/33	OctetString
ifDescr.503316770	Ethernet8/0/34	OctetString
ifDescr.503316778	Ethernet8/0/35	OctetString
ifDescr.503316786	Ethernet8/0/36	OctetString
ifDescr.503316794	Ethernet8/0/37	OctetString
ifDescr.503316802	Ethernet8/0/38	OctetString
ifDescr.503316810	Ethernet8/0/39	OctetString
ifDescr.503316818	Ethernet8/0/40	OctetString
ifDescr.503316826	Ethernet8/0/41	OctetString
ifDescr.503316834	Ethernet8/0/42	OctetString
ifDescr.503316842	Ethernet8/0/43	OctetString
ifDescr.503316850	Ethernet8/0/44	OctetString
ifDescr.503316858	Ethernet8/0/45	OctetString
ifDescr.503316866	Ethernet8/0/46	OctetString
ifDescr.503316874	Ethernet8/0/47	OctetString
ifDescr.503316882	Ethernet8/0/48	OctetString
ifDescr.503316889	GigabitEthernet8/0/49	OctetString
ifDescr.503316897	GigabitEthernet8/0/50	OctetString
ifDescr.503316905	GigabitEthernet8/0/51	OctetString
ifDescr.503316913	GigabitEthernet8/0/52	OctetString

Figura 159

Una vez tenemos el OID ya podemos monitorear por ejemplo si está activo:

```
root@monsrv01:/usr/local/nagios/libexec# ./check_snmp 192.168.200.11 -C public -o ifDescr.503316913 -r 1 -m RFC1213-MIB
SNMP CRITICAL - *GigabitEthernet8/0/52* |
```

Figura 160

Podemos revisar también si el puerto está en UP o Down (en este caso en down):

```
root@monsrv01:/usr/local/nagios/libexec# ./check_snmp 192.168.200.11 -C public -o ifOperStatus.503316913 -r 1 -m RFC1213-MIB
SNMP CRITICAL - *down(2)* |
```

Figura 161

Así se vería un puerto el UP:

```
root@monsrv01:/usr/local/nagios/libexec# ./check_snmp 192.168.200.11 -C public -o ifOperStatus.4228017 -r 1 -m RFC1213-MIB
SNMP OK - up(1) |
root@monsrv01:/usr/local/nagios/libexec#
```

Figura 162

Una vez entendidos estos conceptos, declaramos el comando para las versiones 1 y 2 en el archivo "usr/local/nagios/etc/objects/commands.cfg":

Versión 1:

```

define command {
    command_name    check_smtp
    command_line    $USER1$/check_smtp -H $HOSTADDRESS$ $ARG1$
}

```

Figura 163

Versión 2:

```

define command {
    command_name    check_snmpv2
    command_line    $USER1$/check_snmp -H $HOSTADDRESS$ -P2c $ARG1$
}

```

Figura 164

Una vez definidos los comandos, se ha creado una plantilla denominada Switch.cfg en la ruta “usr/local/nagios/etc/objects” donde definimos estos el equipo a monitorear por la versión 1 y los servicios de ejemplo:

- Definición del equipo a monitorear:

```

define host {
    use                generic-switch
    host_name          HSP001-1
    alias              HSP001-1
    address            10.10.20.15
    hostgroups         switches
    parents            CORE_CST
}

```

- Monitorear si llegamos por ICMP al equipo:

```

define service {
    use                generic-service
    host_name          HSP001-1
    service_description PING
    check_command      check_ping!200.0,20%!600.0,60%
    check_interval     1
    retry_interval     1
}

```

- Monitorear el tiempo de encendido del Switch:

```

define service {
    use                generic-service
    host_name          HSP001-1
    service_description Uptime
}

```

```

    check_command      check_snmp!-C public -o sysUpTime.0
}

```

- Monitorear si el puerto esta UP o DOWN (de dos puertos en este caso):

```

define service{
    use                generic-service
    host_name          HSP001-1
    service_description Port 1/0/50 To Core Link Status
    check_command      check_snmp!-C SNMPub -o
ifOperStatus.4228017 -r 1 -m RFC1213-MIB
    contact_groups    admins
}

```

```

define service{
    use                generic-service
    host_name          HSP001-1
    service_description Port 5/0/14 Link Status
    check_command      check_snmp!-C SNMPub -o
ifOperStatus.289407106 -r 1 -m RFC1213-MIB
    check_interval     1
    retry_interval     1
    contact_groups    admins
}

```

- Monitorear si el puerto tiene errores:

```

define service{
    use                generic-service
    host_name          HSP001-1
    service_description Port 1/0/50 Check Port Errors
    check_command      check_snmp!-C SNMPub -o
ifInErrors.4228017 -r 1 -m RFC1213-MIB -w 170 -c 200
    check_interval     1
    retry_interval     1
    contact_groups    admins
}

```

Y así se ve en el portal web de Nagios Core:

HSP001-1	PING	OK	04-21-2024 10:42:21	3d 17h 7m 43s	1/3	PING OK - Packet loss = 0%, RTA = 2.76 ms
	Port 1/0/50 Check Port Errors	OK	04-21-2024 10:42:20	20d 10h 21m 33s	1/3	SNMP OK - 0
	Port 1/0/50 To Core Link Status	OK	04-21-2024 10:38:21	32d 16h 30m 40s	1/3	SNMP OK - up(1)
	Port 5/0/14 Link Status	CRITICAL	04-21-2024 10:42:21	5d 10h 37m 1s	3/3	SNMP CRITICAL - "down(2)"
	Port 8/0/46 Check Port Errors	WARNING	04-21-2024 10:42:21	28d 10h 25m 44s	3/3	SNMP WARNING - "90"

Figura 165

Ahora hacemos lo mismo, pero para probar los equipos que utilizan versión2:
10.10.10.10

```

define host {
    use                generic-switch
    host_name          HSL001
    alias              HSL001
    address            10.10.10.10
}

```

```

hostgroups      Switch_Primary
parents         RouterHSL_Virtual
}

define service {
    use          generic-service
    host_name    HSL001
    service_description Uptime
    check_command check_snmpv2!-C PSCom -o sysUpTime.0
    check_interval 3
    retry_interval 1
}

```

Y apreciamos su estado en el portal web de Nagios Core:

HSL001	PING	OK	04-21-2024 11:48:21	8d 3h 40m 12s	1/3	PING OK - Packet loss = 0% RTA = 7.92 ms
	Uptime	OK	04-21-2024 11:50:41	0d 0h 6m 18s	1/3	SNMP OK - Timeticks: (1562961025) 180 days, 21:33:30.25

Figura 166

4.8 Investigar monitorear CRC Switch e implementar

Para ello nos conectamos a un Switch e identificamos una interfaz que tenga CRC. En este caso la interfaz 1/0/1 tiene 33 CRC:

```

<HSP001-1>dis int Ethernet 1/0/1
Ethernet1/0/1 current state : UP
IP Sending Frames' Format is PKTFMT_ETHNT_2, Hardware address is 0016-e009-cdc3
Media type is twisted pair, loopback not set
Port hardware type is 100_BASE_TX
100Mbps-speed mode, full-duplex mode
Link speed type is autonegotiation, link duplex type is autonegotiation
Flow-control is not enabled
The Maximum Frame Length is 1522
Broadcast MAX-pps: 1000
Unicast MAX-ratio: 100%
Multicast MAX-pps: 1000
Forbid jumbo frame to pass
PVID: 1
Mdi type: auto
Port link-type: access
Tagged VLAN ID : none
Untagged VLAN ID : 1
Last 300 seconds input: 11 packets/sec 2058 bytes/sec
Last 300 seconds output: 292 packets/sec 245071 bytes/sec
Input(total): 104950744 packets, 32772108065 bytes
2457068 broadcasts, 244288 multicasts, 0 pauses
Input(normal): - packets, - bytes
- broadcasts, - multicasts, - pauses
Input: 5146 input errors, 0 runts, 0 giants, - throttles, 33 CRC
6 frame, - overruns, 5107 aborts, 0 ignored, - parity errors
Output(total): 1929751308 packets, 1801320702425 bytes
450352319 broadcasts, 192895618 multicasts, 0 pauses
Output(normal): - packets, - bytes
- broadcasts, - multicasts, - pauses
Output: 54224546 output errors, - underruns, - buffer failures
54224546 aborts, 0 deferred, 0 collisions, 0 late collisions
0 lost carrier, - no carrier
<HSP001-1>

```

Figura 167

Una vez hemos identificado una interfaz con CRC, abrimos MIB Browser y buscamos el OID de la interfaz:

Result Table	
Name/OID	Value
ifDescr.14	NULL0
ifDescr.16	InLoopBack0
ifDescr.31	Vlan-interface1
ifDescr.63	Vlan-interface5
ifDescr.1623	Vlan-interface200
ifDescr.4227614	Aux1/0/0
ifDescr.4227626	Ethernet1/0/1

Figura 168

Una vez tenemos el OID, podemos ejecutar los parámetros de la tabla que consideremos para listarlos y mediante el OID buscamos si encontramos el valor 33, por ejemplo:

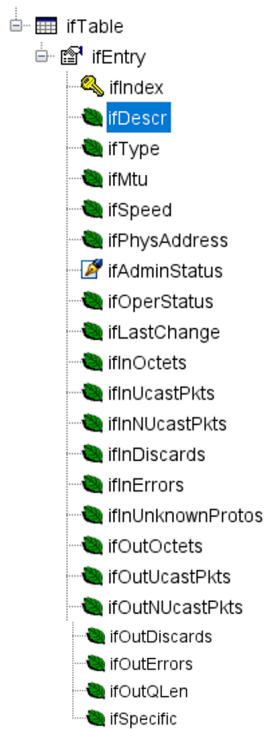


Figura 169

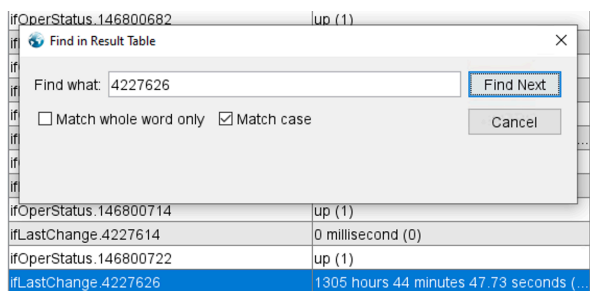


Figura 170

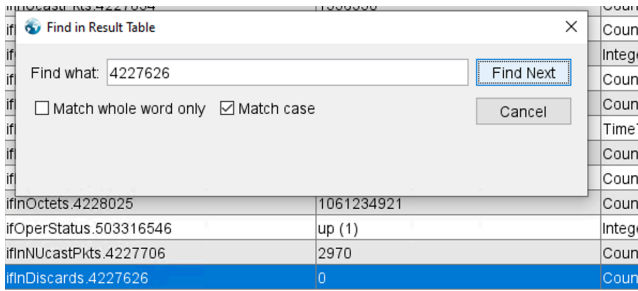


Figura 171

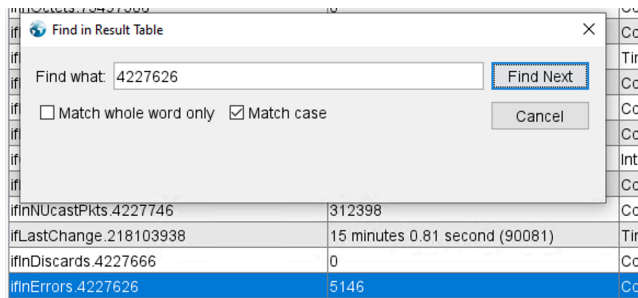


Figura 172

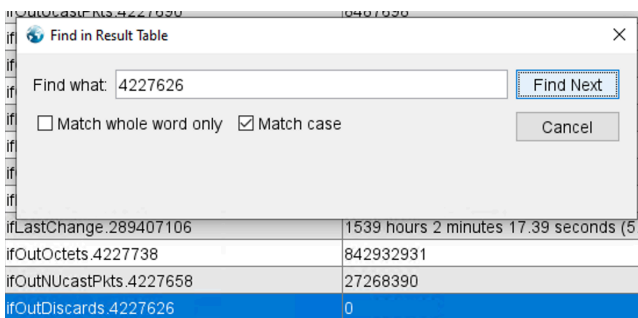


Figura 173

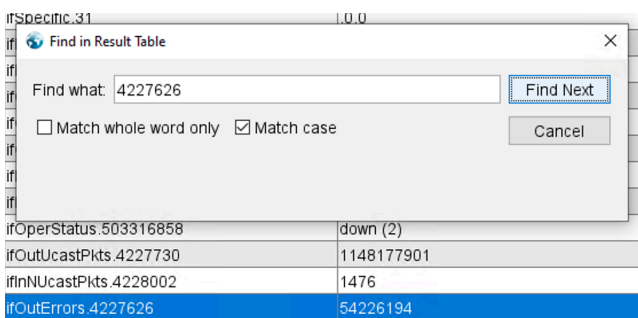


Figura 174

Y se ha finalizado la búsqueda:

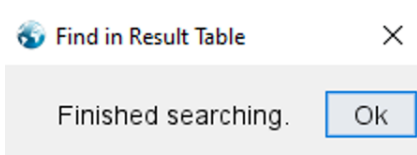


Figura 175

En este caso, no ha sido posible localizar la cantidad de CRC que tiene una interfaz, por lo tanto, no es posible su monitorización.

5. Configuración de Nagios Core en entorno sanitario:

5.1 Crear plantillas pertinentes.

En el siguiente apartado se van a generar las plantillas pertinentes para monitorear los equipos correspondientes.

Para ello, primero de todo se debe definir las rutas y nombres de los nuevos archivos en el siguiente fichero “/usr/local/nagios/etc/nagios.cfg”. Para el siguiente proyecto se harán uso de las siguientes:

```
cfg_file=/usr/local/nagios/etc/objects/windows.cfg
cfg_file=/usr/local/nagios/etc/objects/linux.cfg
cfg_file=/usr/local/nagios/etc/objects/services.cfg
cfg_file=/usr/local/nagios/etc/objects/routers.cfg
cfg_file=/usr/local/nagios/etc/objects/CAPS/DC.cfg
cfg_file=/usr/local/nagios/etc/objects/CAPS/routers.cfg
cfg_file=/usr/local/nagios/etc/objects/CAPS/switchs.cfg
cfg_file=/usr/local/nagios/etc/objects/switch.cfg
```

Figura 176

Una vez definido, se deben crear los archivos en las rutas mencionadas con sus correspondientes extensiones y ya podremos comenzar a instalar clientes en caso necesario y comenzar con la monitorización pertinente.

5.2 Configuración de clientes.

En el siguiente apartado, se procederá a configurar los clientes que deben de ser monitoreados en sus respectivas plantillas:

- Linux: Para monitorear equipos Linux se debe instalar el agente NCPA. Una vez instalado y configurado ya se pueden construir los respetivos comandos de monitorización.

Primero de todo se debe de definir el host en la correspondiente plantilla:

```
define host {
use          linux-server
host_name    USISRV01_Kayako
alias        USISRV01_Kayako
address      usisrv01.hsp.csdt.es
parents      CORE_CST
hostgroups   linux-servers
}
```

Como podemos apreciar, hacemos que este objeto sea dependiente del CORE_CST. Core CST es el Switch principal de donde radican todas las comunicaciones. Por otro lado, se hace uso del grupo predefinido Linux-servers, se puede ver su contenido en “templates.cfg”:

```
define host {
    name                linux-server                ; The name of this host template
    use                 generic-host                ; This template inherits other values from the generic-host template
    check_period        24x7                        ; By default, Linux hosts are checked round the clock
    check_interval      5                          ; Actively check the host every 5 minutes
    retry_interval      1                          ; Schedule host check retries at 1 minute intervals
    max_check_attempts  10                         ; Check each Linux host 10 times (max)
    check_command        check-host-alive          ; Default command to check Linux hosts
    notification_period workhours                  ; Linux admins hate to be woken up, so we only notify during the day
                                                    ; Note that the notification_period variable is being overridden from
                                                    ; the value that is inherited from the generic-host template!

    notification_interval 0                        ; Resend notifications every 2 hours
    notification_options  d,u,r                    ; Only send notifications for specific host states
    contact_groups        admins                   ; Notifications get sent to the admins by default
    register              0                        ; DON'T REGISTER THIS DEFINITION - ITS NOT A REAL HOST, JUST A TEMPLATE!
}
}
```

Figura 177

Como podemos apreciar en la plantilla, se definen los intervalos de monitorización, los intentos y las notificaciones (que por defecto se utiliza los contactos “admins”).

Una vez se ha definido el host, monitoreamos los servicios que nos interesan:

Monitoreamos mediante ICMP que el servidor este operativo:

```
define service {
    use                 generic-service
    host_name          USISRV01_Kayako
    service_description PING
    check_command       check_ping!100.0,20%!500.0,60%
}
}
```

Monitoreamos mediante el cliente NCPA el uso del porcentaje utilizado de la CPU, en el caso de que supere el 80% alertaremos con un Warning, si se utiliza el 95% la alerta será del tipo “Critical” y con el uso de “G” nos proporcionará el valor en GB en vez de Bytes:

```
define service {
    use                 generic-service
    host_name          USISRV01_Kayako
    service_description Check CPU percentatge utilitzat
    check_command       check_ncpa!-t 'SecurePassword' -P 5693 -M
cpu/percent -w 80 -c 95 -u G
}
}
```

Monitoreamos mediante el cliente NCPA el uso del porcentaje utilizado de la memoria virtual, en el caso de que supere el 50% alertaremos con un Warning, si se utiliza el 80% la alerta será del tipo “Critical” y con el uso de “G” nos proporcionará el valor en GB en vez de Bytes:

```

define service {
    use                generic-service
    host_name          USISRV01_Kayako
    service_description Check Memoria Virtual
    check_command      check_ncpa!-t 'SecurePassword' -P 5693 -M
memory/virtual -w 50 -c 80 -u G
}

```

Monitoreamos mediante el cliente NCPA el uso del porcentaje utilizado del disco lógico, en el caso de que supere el 85% alertaremos con un Warning, si se utiliza el 90% la alerta será del tipo “Critical” y con el uso de “G” nos proporcionará el valor en GB en vez de Bytes:

```

define service {
    use                generic-service
    host_name          USISRV01_Kayako
    service_description Check Disk percentatge utilitzat
    check_command      check_ncpa!-t 'SecurePassword' -P 5693 -
M'disk/logical//used_percent' -w 85 -c 90 -u G
}

```

Monitoreamos mediante el protocolo HTTP que el servidor este operativo por el puerto TCP 80:

```

define service{
    use                generic-service
    host_name          USISRV01_Kayako
    service_description Kayako http
    check_command      check_tcp!80
}

```

Una vez se reinician los servicios, podremos apreciar su estado en el sitio web de Nagios:

USISRV01_Kayako	Check CPU percentatge utilitzat	OK	05-11-2024 10:40:34	0d 22h 43m 11s	1/3	OK: Percent was 0.00 %, 2.00 %, 0.00 %, 0.00 %
	Check Disk percentatge utilitzat	OK	05-11-2024 10:40:35	1d 21h 42m 10s	1/3	OK: Used_percent was 79.60 %
	Check Memoria Virtual	OK	05-11-2024 10:40:35	1d 21h 44m 34s	1/3	OK: Memory usage was 20.00 % (Available: 13.33 GB, Total: 16.66 GB, Free: 2.21 GB, Used: 2.09 GB)
	Kayako http	OK	05-11-2024 10:40:35	1d 21h 17m 20s	1/3	TCP OK - 0.018 second response time on usisrv01.hsp.csdt.es port 80
	PING	OK	05-11-2024 10:40:35	1d 21h 39m 23s	1/3	PING OK - Packet loss = 0%, RTA = 0.93 ms

Figura 178

Una vez explicado como configurar un cliente de monitorización, se van a presentar capturas de los servicios monitoreados del resto de clientes:

TOMCAT01_intranet	Check CPU percentatge utilitzat	OK	05-11-2024 10:46:44	0d 3h 44m 11s	1/3	OK: Percent was 0.00 %, 1.90 %, 0.00 %, 0.00 %, 0.00 %, 0.00 %
	Check Disk percentatge utilitzat	WARNING	05-11-2024 10:40:34	1d 21h 41m 30s	3/3	WARNING: Used_percent was 93.90 %
	Check Memoria Virtual	OK	05-11-2024 10:40:36	1d 21h 40m 7s	1/3	OK: Used memory was 30.50 % (Available: 11.73 GB, Total: 16.87 GB, Free: 0.24 GB, Used: 5.11 GB)
	Intranet http	OK	05-11-2024 10:40:35	1d 21h 12m 54s	1/3	TCP OK - 0.011 second response time on tomcat01.hsp.csdt.es port 80

Figura 179

WEBSRV01	Check CPU percentatge utilitzat	OK	05-11-2024 10:40:34	1d 21h 42m 4s	1/3	OK: Percent was 10.00 %
	Check Disk percentatge utilitzat	OK	05-11-2024 10:40:35	1d 21h 40m 5s	1/3	OK: Used_percent was 72.80 %
	Check Memoria Virtual	OK	05-11-2024 10:40:35	1d 21h 46m 2s	1/3	OK: Used memory was 41.10 % (Available: 1.25 GB, Total: 2.12 GB, Free: 0.31 GB, Used: 0.85 GB)
	PING	OK	05-11-2024 10:40:35	1d 21h 45m 41s	1/3	PING OK - Packet loss = 0%, RTA = 2.06 ms

Figura 180

- Services: En esta plantilla se van a proceder a definir servicios a monitorear de servidores y servicios de los cuales no podemos gestionar. Definimos uno de los hosts:

```
define host {
    use                generic-service-host
    host_name          hccc.salut.gencat.cat
    alias              hccc.salut.gencat.cat
    address            hccc.salut.gencat.cat
    parents            MACROLAN
}
```

Y posteriormente, su servicio a monitorear, en este caso HTTPS:

```
define service{
    use                generic-service
    host_name          hccc.salut.gencat.cat
    service_description Check HCCC port https
    check_command      check_tcp!443
}
```

Y podremos visualizar su estado en Nagios Core:

hccc.salut.gencat.cat	Check HCCC port https	OK	05-12-2024 09:34:10	0d 6h 55m 48s	1/3	TCP OK - 0,006 second response time on hccc.salut.gencat.cat port 443
-----------------------	-----------------------	----	---------------------	---------------	-----	---

Figura 181

- Routers: Procedemos con la definición de uno de los Routers en su correspondiente plantilla:

```
define host {
    use                generic-routers
    host_name          MACROLAN;
    alias              MACROLAN
    address            192.168.10.2
    hostgroups         routers
    parents            CORE_CST
}
```

Como se puede apreciar, forma parte del grupo routers, de este modo establecemos un único servicio para monitorear mediante ICMP un conjunto de equipos:

```
define service {
    use                generic-service
```

```

hostgroup      routers
service_description  PING
check_command   check_ping!200.0,20%!600.0,60%
check_interval  1
retry_interval  1
}

```

Y podremos ver su estado en la página web de Nagios Core:

Network Routers (routers)			
Host	Status	Services	Actions
MACROLAN	UP	1 OK	
MACROLAN_VILA	UP	1 OK	

Figura 182

- CAPS DC: Procedemos con la definición de uno de los controladores de dominio en su correspondiente plantilla:

```

define host {
    use                windows-server
    host_name          HLSLRV01.hsl.csdtd.es
    alias              HLSLRV01_Server
    address            HLSLRV01.hsl.csdtd.es
    parents            HSL001
    hostgroups         DC_Primary
}

```

Como se puede apreciar, forma parte del grupo DC_Primary, de este modo establecemos un único servicio para monitorear mediante ICMP un conjunto de servidores:

```

define service {
    use                generic-service
    hostgroup          DC_Primary
    service_description  PING
    check_command       check_ping!200.0,20%!600.0,60%
    check_interval      3
    retry_interval      1
}

```

Y podremos apreciar su monitorización en la interfaz web de Nagios Core:

Host	Service	Status
HLSLRV01.hsl.csdtd.es	PING	OK

Figura 183

Hacemos lo mismo para el resto y visualizamos sus estados:








































DC_Primary (DC_Primary)			
Host	Status	Services	Actions
CBSSRV01.cbs.csdtd.es	UP	1 OK	  
CFSSRV01.cfs.csdtd.es	UP	1 OK	  
CRBSSRV01.crb.csdtd.es	UP	1 OK	  
CSGSRV01.csg.csdtd.es	UP	1 OK	  
CSLSRV01.csl.csdtd.es	UP	1 OK	  
CTNSRV01.ctn.csdtd.es	UP	1 OK	  
ESTSRV01.est.csdtd.es	UP	1 OK	  
HSJSRV01.hsj.csdtd.es	UP	1 OK	  
HLSRV01.hsl.csdtd.es	UP	1 OK	  
MDPSRV01.mdp.csdtd.es	UP	1 OK	  
ROCSR01.hsp.csdtd.es	UP	1 OK	  
RTPSRV02.rtp.csdtd.es	UP	1 OK	  
SRCSR01.src.csdtd.es	UP	1 OK	  

Figura 184

- CAPS Routers: En este caso, monitorearemos 2 IP Routers físicos y otro virtual para garantizar la alta disponibilidad. Para ello, únicamente monitoraremos mediante ICMP por ping. Configuramos los hosts:

```

define host {
    use                generic-routers
    host_name          RouterHSL_Virtual;
    hostgroups         Routers_Primary
    alias              RouterHSL_Virtual
    address            10.10.0.1
    parents            MACROLAN
}

define host {
    use                generic-routers
    host_name          RouterHSL_Fisic_PP;
    hostgroups         Routers_Primary
    alias              RouterHSL_Virtual
    address            10.10.0.2
    parents            RouterHSL_Virtual
}

define host {
    use                generic-routers
    host_name          RouterHSL_Fisic_BCK;
    hostgroups         Routers_Primary
    alias              RouterHSL_Virtual
    address            10.10.0.3
    parents            RouterHSL_Virtual
}

```

Como se puede apreciar, el router virtual es dependiente del router de la macrolan y el resto del virtual.

Como todos los enrutadores forman parte del grupo Routers_Primarya, establemos el servicio a monitorear a la vez para todos los miembros:

```
define service {
    use                generic-service
    hostgroup          Routers_Primarya
    service_description PING
    check_command      check_ping!200.0,20%!600.0,60%
    check_interval    2
    retry_interval    1
}
```

A continuación, visualizamos sus estados en el sitio web de Nagios Core:

RouterHSL_Fisic_BCK	PING	OK	05-11-2024 20:22:35
RouterHSL_Fisic_PP	PING	CRITICAL	05-11-2024 20:22:35
RouterHSL_Virtual	PING	OK	05-11-2024 20:22:35

Figura 185

Y se efectúa el mismo proceso para el resto:

RouterALTHAIA	PING	OK
RouterCBS_Fisic_BCK	PING	OK
RouterCBS_Fisic_PP	PING	OK
RouterCBS_Virtual	PING	OK
RouterCDSM	PING	OK
RouterCFS_Fisic_BCK	PING	OK
RouterCFS_Fisic_PP	PING	OK
RouterCFS_Virtual	PING	OK
RouterCRB_Fisic_BCK	PING	OK
RouterCRB_Fisic_PP	PING	OK
RouterCRB_Virtual	PING	OK

RouterCRB_Virtual	PING	OK
RouterCSG_Fisic_BCK	PING	OK
RouterCSG_Fisic_PP	PING	OK
RouterCSG_Virtual	PING	OK
RouterCSL_Fisic_BCK	PING	OK
RouterCSL_Fisic_PP	PING	OK
RouterCSL_Virtual	PING	OK
RouterCTN_Fisic_BCK	PING	OK
RouterCTN_Fisic_PP	PING	OK
RouterCTN_Virtual	PING	OK
RouterEST_Fisic_PP	PING	OK
RouterEST_Virtual	PING	OK
RouterHSJ_Fisic_BCK	PING	OK
RouterHSJ_Fisic_PP	PING	OK
RouterHSJ_Virtual	PING	OK

Figura 186

Figura 187

RouterMDP_Fisic_BCK	PING	OK
RouterMDP_Fisic_PP	PING	OK
RouterMDP_Virtual	PING	OK
RouterROC_Fisic_BCK	PING	OK
RouterROC_Fisic_PP	PING	OK
RouterROC_Virtual	PING	OK
RouterSALUT+_Fisic_BCK	PING	OK
RouterSALUT+_Fisic_PP	PING	OK
RouterSALUT+_Virtual	PING	OK
RouterSRC_Fisic_BCK	PING	OK
RouterSRC_Fisic_PP	PING	OK
RouterSRC_Virtual	PING	OK

Figura 188

Y podremos apreciar en Summary el monitoreo de todo el conjunto:

Routers_Primary (Routers_Primary)	39 UP	39 OK
	1 DOWN : 1 Unhandled	1 CRITICAL : 1 on Problem Hosts

Figura 189

- CAPS Switch: Para monitorear este tipo de equipos se deben construir los respectivos comandos de monitorización mediante los OID de las interfaces correspondientes.

Primero de todo se debe de definir el host que en la correspondiente plantilla:

```
define host {
    use          generic-switch
    host_name    CFS001
    alias        CFS001
    address      172.60.45.30
    hostgroups   Switch_Primary
    parents      RouterCFS_Virtual
}
```

Como podemos apreciar, hacemos que este objeto sea dependiente del RouterCFS_Virtual. Es el Router principal del centro. Por otro lado, se ha definido el “hostgroup” Switch_Primary para de este modo, monitorear el ICMP por ejemplo de todos los Switch sin tener que definirlo para cada uno de ellos:

```
define hostgroup {
    hostgroup_name    Switch_Primary
    alias              Switch_Primary
}
```

Y definición de monitorización de ICMP para todos los equipos que pertenezcan al grupo Switch_Primary:

```
define service {  
  
    use          generic-service  
    hostgroup    Switch_Primary  
    service_description PING  
    check_command check_ping!200.0,20%!600.0,60%  
}  

```

Una vez definido el equipo, podemos configurar los servicios a monitorear, en este caso comenzaremos por el “uptime” para comprobar el tiempo de encendido del equipo:

```
define service {  
  
    use          generic-service  
    host_name    CFS001  
    service_description Uptime  
    check_command check_snmpv2!-C SCom -o sysUpTime.0  
    check_interval 3  
    retry_interval 1  
  
}  

```

Posteriormente, definiremos los correspondientes puertos que se quieren monitorear indicándole sus respectivos OID:

```
define service{  
    use          generic-service  
    host_name    CFS001  
    service_description Port GE0/0/43 SERVIDOR VEU  
    check_command check_snmpv2!-C SCom -o ifOperStatus.248 -r 1 -m  
RFC1213-MIB  
}  

```

```
define service{  
    use          generic-service  
    host_name    CFS001  
    service_description Port GE0/0/44 SERVIDOR DADES  
    check_command check_snmpv2!-C SCom -o ifOperStatus.249 -r 1 -m  
RFC1213-MIB  
}  

```

```
define service{  
    use          generic-service  
    host_name    CFS001  
    service_description Port GE0/0/45 MACROLAN BCK VEU  
    check_command check_snmpv2!-C SCom -o ifOperStatus.250 -r 1 -m  
RFC1213-MIB  
}  

```

```

define service{
    use          generic-service
    host_name    CFS001
    service_description Port GE0/0/46 MACROLAN BCK DADES
    check_command check_snmpv2!-C SCom -o ifOperStatus.251 -r 1 -m
RFC1213-MIB
}

```

```

define service{
    use          generic-service
    host_name    CFS001
    service_description Port GE0/0/47 MACROLAN PPAL VEU
    check_command check_snmpv2!-C SCom -o ifOperStatus.252 -r 1 -m
RFC1213-MIB
}

```

```

define service{
    use          generic-service
    host_name    CFS001
    service_description Port GE0/0/48 MACROLAN PPAL DADES
    check_command check_snmpv2!-C SCom -o ifOperStatus.253 -r 1 -m
RFC1213-MIB
}

```

Finalmente, podremos ver su monitorización en el sitio web de Nagios Core:

Host	Service	Status	Last Check	Next Check	Output	Performance Data
CFS001	PING	OK	05-11-2024 11:20:34	1d 23h 23m 59s	1/3	PING OK - Packet loss = 0%, RTA = 2.30 ms
	Port GE0/0/43 SERVIDOR VEU	OK	05-11-2024 11:20:35	1d 23h 24m 2s	1/3	SNMP OK - up(1)
	Port GE0/0/44 SERVIDOR DADES	OK	05-11-2024 11:20:35	1d 23h 24m 2s	1/3	SNMP OK - up(1)
	Port GE0/0/45 MACROLAN BCK VEU	OK	05-11-2024 11:20:35	1d 23h 24m 2s	1/3	SNMP OK - up(1)
	Port GE0/0/46 MACROLAN BCK DADES	OK	05-11-2024 11:20:35	1d 23h 23m 14s	1/3	SNMP OK - up(1)
	Port GE0/0/47 MACROLAN PPAL VEU	OK	05-11-2024 11:20:35	1d 23h 24m 2s	1/3	SNMP OK - up(1)
	Port GE0/0/48 MACROLAN PPAL DADES	OK	05-11-2024 11:20:35	1d 23h 23m 31s	1/3	SNMP OK - up(1)
Uptime	OK	05-11-2024 11:22:34	1d 23h 25m 4s	1/3	SNMP OK - Timeticks: (100929116) 11 days, 16:21:31.16	

Figura 190

Una vez explicado como configurar un cliente SNMP de monitorización, se van a presentar capturas de los servicios monitoreados del resto de equipos:

Host	Service	Status	Last Check	Next Check	Output	Performance Data
CBS001	PING	OK	05-11-2024 11:20:34	1d 23h 23m 17s	1/3	PING OK - Packet loss = 0%, RTA = 4.76 ms
	Port Eth-Trunk1 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:20:34	1d 23h 18m 29s	1/3	SNMP OK - up(1)
	Port GE0/0/1 MACROLAN-WIFIVLAN_PRINCIPAL	OK	05-11-2024 11:20:35	1d 23h 23m 59s	1/3	SNMP OK - up(1)
	Port GE0/0/10 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:20:34	1d 23h 22m 33s	1/3	SNMP OK - up(1)
	Port GE0/0/46 FORTIGATE - Port 01	OK	05-11-2024 11:20:34	1d 23h 24m 2s	1/3	SNMP OK - up(1)
	Port GE0/0/47 MACROLAN DADES BACKUP	OK	05-11-2024 11:20:35	1d 23h 24m 2s	1/3	SNMP OK - up(1)
	Port GE0/0/48 MACROLAN VEU BACKUP	OK	05-11-2024 11:20:35	1d 23h 24m 2s	1/3	SNMP OK - up(1)
	Port GE0/0/6 PB-34 CENTRALETA MCP40 - CBS	OK	05-11-2024 11:20:35	1d 23h 22m 27s	1/3	SNMP OK - up(1)
	Port GE8/0/1 MACROLAN-WIFIVLAN_BACKUP	OK	05-11-2024 11:20:35	1d 23h 23m 16s	1/3	SNMP OK - up(1)
	Port GE8/0/10 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:20:34	1d 23h 24m 3s	1/3	SNMP OK - up(1)
	Port GE8/0/42 FORTIGATE - Port 02	OK	05-11-2024 11:20:35	1d 23h 24m 3s	1/3	SNMP OK - up(1)
	Port GE8/0/43 MODUL SUPERVIVENCIA CTTI-VEU	OK	05-11-2024 11:20:34	1d 23h 22m 27s	1/3	SNMP OK - up(1)
	Port GE8/0/45 MACROLAN VEU PRINCIPAL	OK	05-11-2024 11:20:35	1d 23h 23m 59s	1/3	SNMP OK - up(1)
	Port GE8/0/48 MACROLAN DADES PRINCIPAL	OK	05-11-2024 11:20:36	1d 23h 23m 59s	1/3	SNMP OK - up(1)
	Uptime	OK	05-11-2024 11:22:34	1d 23h 25m 5s	1/3	SNMP OK - Timeticks: (1054213702) 122 days, 0:22:17.02

Figura 191

Host	Service	Status	Last Check	Next Check	Output	Performance Data
CDSM001_SRCT	PING	OK	05-11-2024 11:20:35	1d 23h 25m 32s	1/3	PING OK - Packet loss = 0%, RTA = 28.95 ms
	Port GE0/0/47 FTTH VEU-IP CST	OK	05-11-2024 11:20:35	1d 23h 25m 2s	1/3	SNMP OK - up(1)
	Port GE0/0/48 FTTH DADES CST	OK	05-11-2024 11:20:35	1d 23h 24m 2s	1/3	SNMP OK - up(1)
Uptime	OK	05-11-2024 11:25:34	1d 23h 27m 22s	1/3	SNMP OK - Timeticks: (1952829519) 226 days, 0:31:35.19	

Figura 192

CRB001	PING	OK	05-11-2024 11:20:35	1d 23h 25m 29s	1/3	PING OK - Packet loss = 0%, RTA = 4.22 ms
	Port Eth-Trunk1 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:20:35	1d 23h 25m 33s	1/3	SNMP OK - up(1)
	Port GE0/0/20 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:20:34	1d 23h 25m 33s	1/3	SNMP OK - up(1)
	Port GE0/0/24 MODUL SUPERVIVENIA CTTI	OK	05-11-2024 11:20:35	1d 23h 25m 32s	1/3	SNMP OK - up(1)
	Port GE0/0/26 MACROLAN DADES PPAL	OK	05-11-2024 11:20:35	1d 23h 25m 32s	1/3	SNMP OK - up(1)
	Port GE0/0/38 Central Intrusio	OK	05-11-2024 11:20:35	1d 23h 24m 0s	1/3	SNMP OK - up(1)
	Port GE0/0/40 MACROLAN-WIFIVLAN_PRINCIPAL	OK	05-11-2024 11:20:34	1d 23h 25m 32s	1/3	SNMP OK - up(1)
	Port GE8/0/2 MEGAFONIA PLANTA 0	OK	05-11-2024 11:20:35	1d 23h 25m 29s	1/3	SNMP OK - up(1)
	Port GE8/0/3 ENLACE SW TEMPORAL CRB002	OK	05-11-2024 11:20:35	1d 23h 25m 32s	1/3	SNMP OK - up(1)
	Port GE8/0/4 CENTRALETA MCP40 - CAB	OK	05-11-2024 11:20:35	1d 23h 25m 29s	1/3	SNMP OK - up(1)
	Port GE8/0/40 MACROLAN-WIFIVLAN_BACKUP	OK	05-11-2024 11:20:34	1d 23h 24m 59s	1/3	SNMP OK - up(1)
	Port GE8/0/47 MACROLAN DADES BCK	OK	05-11-2024 11:20:34	1d 23h 25m 32s	1/3	SNMP OK - up(1)
	Port GE8/0/9 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:20:35	1d 23h 25m 33s	1/3	SNMP OK - up(1)
	Uptime	OK	05-11-2024 11:25:34	1d 23h 26m 24s	1/3	SNMP OK - Timeticks: (96279765) 11 days, 3:26:37.65

Figura 193

CSG001	PING	OK	05-11-2024 11:20:34	1d 23h 25m 33s	1/3	PING OK - Packet loss = 0%, RTA = 2.42 ms
	Port Eth-Trunk1 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:20:35	1d 23h 25m 29s	1/3	SNMP OK - up(1)
	Port GE0/0/12 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:20:34	1d 23h 25m 33s	1/3	SNMP OK - up(1)
	Port GE0/0/17 MACROLAN-WIFIVLAN_PRINCIPAL	OK	05-11-2024 11:20:35	1d 23h 25m 32s	1/3	SNMP OK - up(1)
	Port GE0/0/2 MACROLAN BACKUP - VEU	OK	05-11-2024 11:20:35	1d 23h 25m 29s	1/3	SNMP OK - up(1)
	Port GE0/0/38 MEGAFONIA GENERAL	OK	05-11-2024 11:20:35	1d 23h 24m 57s	1/3	SNMP OK - up(1)
	Port GE0/0/43 MEGAFONIA ATENCIO CONTINUADA P0	OK	05-11-2024 11:20:35	1d 23h 25m 29s	1/3	SNMP OK - up(1)
	Port GE0/0/44 MEGAFONIA S. ESPERA ADULTS P0	OK	05-11-2024 11:20:35	1d 23h 25m 33s	1/3	SNMP OK - up(1)
	Port GE0/0/48 MACROLAN PRINCIPAL DADES	OK	05-11-2024 11:20:35	1d 23h 25m 33s	1/3	SNMP OK - up(1)
	Port GE8/0/1 CENTRALETA MCP040 - CSG	OK	05-11-2024 11:20:35	1d 23h 25m 29s	1/3	SNMP OK - up(1)
	Port GE8/0/13 MACROLAN-WIFIVLAN_BACKUP	OK	05-11-2024 11:20:34	1d 23h 24m 42s	1/3	SNMP OK - up(1)
	Port GE8/0/2 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:20:35	1d 23h 25m 57s	1/3	SNMP OK - up(1)
	Port GE8/0/38 MEGAFONIA PEDIATRIA P0	OK	05-11-2024 11:20:35	1d 23h 25m 32s	1/3	SNMP OK - up(1)
	Port GE8/0/39 MACROLAN PRINCIPAL - VEU	OK	05-11-2024 11:20:35	1d 23h 25m 33s	1/3	SNMP OK - up(1)
	Port GE8/0/4 MODUL SUPERVIVENCIA VEU CTTI	OK	05-11-2024 11:20:36	1d 23h 25m 29s	1/3	SNMP OK - up(1)
	Port GE8/0/43 MEGAFONIA S. ESPERA ADULTS P1	OK	05-11-2024 11:20:35	1d 23h 25m 29s	1/3	SNMP OK - up(1)
	Port GE8/0/45 MEGAFONIA GINECOLOGIA P1	OK	05-11-2024 11:20:35	1d 23h 25m 32s	1/3	SNMP OK - up(1)
	Port GE8/0/47 MACROLAN BACKUP DADES	OK	05-11-2024 11:20:35	1d 23h 25m 33s	1/3	SNMP OK - up(1)
	Uptime	OK	05-11-2024 11:25:34	1d 23h 26m 22s	1/3	SNMP OK - Timeticks: (1469746649) 170 days, 2:37:46.49

Figura 194

CSL001	PING	OK	05-11-2024 11:20:34	1d 23h 24m 21s	1/3	PING OK - Packet loss = 0%, RTA = 2.84 ms
	Port GE0/0/1 MEGAFONIA PLANTA 3 - S. ESPERA 2	OK	05-11-2024 11:20:36	1d 23h 25m 33s	1/3	SNMP OK - up(1)
	Port GE0/0/11 MEGAFONIA GENERAL	OK	05-11-2024 11:20:36	1d 23h 25m 33s	1/3	SNMP OK - up(1)
	Port GE0/0/12 MEGAFONIA TOTES LES CONSULTES	OK	05-11-2024 11:20:35	1d 23h 25m 32s	1/3	SNMP OK - up(1)
	Port GE0/0/14 MEGAFONIA PLANTA 2 - S. ESPERA 1	OK	05-11-2024 11:20:35	1d 23h 25m 29s	1/3	SNMP OK - up(1)
	Port GE0/0/15 MEGAFONIA PLANTA 2 - S. ESPERA 2	OK	05-11-2024 11:20:34	1d 23h 25m 29s	1/3	SNMP OK - up(1)
	Port GE0/0/16 MEGAFONIA PLANTA 2 - S. ESPERA 3	OK	05-11-2024 11:20:35	1d 23h 25m 29s	1/3	SNMP OK - up(1)
	Port GE0/0/20 CENTRALETA MPC40 - CSL	OK	05-11-2024 11:20:35	1d 23h 25m 29s	1/3	SNMP OK - up(1)
	Port GE1/0/12 MEGAFONIA PLANTA 3 - S. ESPERA 1	OK	05-11-2024 11:20:35	1d 23h 25m 33s	1/3	SNMP OK - up(1)
	Port GE1/0/22 MEGAFONIA PLANTA 4 - S. ESPERA 2	OK	05-11-2024 11:20:35	1d 23h 24m 39s	1/3	SNMP OK - up(1)
	Port GE1/0/35 MEGAFONIA PLANTA 4 - S. ESPERA 3	OK	05-11-2024 11:20:35	1d 23h 25m 29s	1/3	SNMP OK - up(1)
	Port GE1/0/6 MEGAFONIA PLANTA 2 - S. ESPERA 4	OK	05-11-2024 11:20:36	1d 23h 25m 32s	1/3	SNMP OK - up(1)
	Port GE8/0/1 MODUL SUPERVIVENCIA - CTTI	OK	05-11-2024 11:20:35	1d 23h 25m 32s	1/3	SNMP OK - up(1)
	Port GE8/0/2 MEGAFONIA PLANTA 4 - S. ESPERA 4	OK	05-11-2024 11:20:35	1d 23h 25m 32s	1/3	SNMP OK - up(1)
	Port GE8/0/48 SERVIDOR CSLSRV01	OK	05-11-2024 11:20:35	1d 23h 25m 32s	1/3	SNMP OK - up(1)
	Port GE8/0/5 MEGAFONIA PLANTA 4 - S. ESPERA 5	OK	05-11-2024 11:20:35	1d 23h 24m 53s	1/3	SNMP OK - up(1)
	Port GE8/0/7 MEGAFONIA PLANTA 4 - S. ESPERA 6	OK	05-11-2024 11:20:35	1d 23h 25m 32s	1/3	SNMP OK - up(1)
	Port XGE0/0/2 LINK CSL-HSL	OK	05-11-2024 11:20:36	1d 23h 27m 55s	1/3	SNMP OK - up(1)
	Port XGE0/0/4 Enlace CSL001 amb CSL002	OK	05-11-2024 11:20:35	1d 23h 25m 18s	1/3	SNMP OK - up(1)
	Port XGE8/0/2 LINK CSL-HSL	OK	05-11-2024 11:20:35	1d 23h 25m 33s	1/3	SNMP OK - up(1)
	Port XGE8/0/4 Enlace CSL001 amb CSL002	OK	05-11-2024 11:20:35	1d 23h 25m 29s	1/3	SNMP OK - up(1)
	Uptime	OK	05-11-2024 11:25:34	1d 23h 26m 37s	1/3	SNMP OK - Timeticks: (3378164035) 390 days, 23:47:20.35

Figura 195

CSL002	PING	OK	05-11-2024 11:20:36	1d 21h 12m 5s	1/3	PING OK - Packet loss = 0%, RTA = 10.59 ms
	Port Eth-Trunk1 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:20:35	1d 23h 25m 18s	1/3	SNMP OK - up(1)
	Port Eth-Trunk2 Enlace LACP CSL001	OK	05-11-2024 11:20:35	1d 23h 25m 32s	1/3	SNMP OK - up(1)
	Port GE0/0/47 LAG Extreme 4-24	OK	05-11-2024 11:20:35	1d 23h 25m 29s	1/3	SNMP OK - up(1)
	Port GE0/0/48 LAG Extreme 1-24	OK	05-11-2024 11:20:35	1d 23h 25m 33s	1/3	SNMP OK - up(1)
	Port XGE0/0/1 Enlace CSL001 - CSL002	OK	05-11-2024 11:20:36	1d 23h 25m 18s	1/3	SNMP OK - up(1)
	Port XGE0/0/4 Enlace CSL001 - CSL002	OK	05-11-2024 11:20:35	1d 23h 24m 51s	1/3	SNMP OK - up(1)
	Uptime	OK	05-11-2024 11:25:34	1d 23h 28m 53s	1/3	SNMP OK - Timeticks: (312022011) 36 days, 2:43:40.11

Figura 196

CTN001	PING	OK	05-11-2024 11:20:36	1d 23h 24m 38s	1/3	PING OK - Packet loss = 0%, RTA = 1.88 ms
	Port Eth-Trunk1 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:20:35	1d 23h 25m 54s	1/3	SNMP OK - up(1)
	Port GE0/0/1 MACROLAN-DADES-PRINCIPAL	OK	05-11-2024 11:20:34	1d 23h 25m 29s	1/3	SNMP OK - up(1)
	Port GE0/0/17 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:20:35	1d 23h 25m 32s	1/3	SNMP OK - up(1)
	Port GE0/0/20 MODUL SUPERVIVENCIA VEU CTTI	OK	05-11-2024 11:20:35	1d 23h 24m 36s	1/3	SNMP OK - up(1)
	Port GE0/0/22 IF VEU MACROLAN PRINCIPAL	OK	05-11-2024 11:20:35	1d 23h 25m 29s	1/3	SNMP OK - up(1)
	Port GE0/0/22 IF VEU MACROLAN PRINCIPAL	OK	05-11-2024 11:20:35	1d 23h 27m 9s	1/3	SNMP OK - up(1)
	Port GE0/0/35 MACROLAN-WIFIVLAN_PRINCIPAL	OK	05-11-2024 11:20:35	1d 23h 27m 9s	1/3	SNMP OK - up(1)
	Port GE0/0/42 MEGAFONIA GENERAL	OK	05-11-2024 11:20:35	1d 23h 27m 9s	1/3	SNMP OK - up(1)
	Port GE0/0/43 MEGAFONIA ZONA 1	OK	05-11-2024 11:20:35	1d 23h 27m 12s	1/3	SNMP OK - up(1)
	Port GE0/0/44 MEGAFONIA ZONA 2	OK	05-11-2024 11:20:34	1d 23h 27m 12s	1/3	SNMP OK - up(1)
	Port GE0/0/1 MACROLAN-DADES-BACKUP	OK	05-11-2024 11:20:35	1d 23h 26m 15s	1/3	SNMP OK - up(1)
	Port GE0/0/14 MACROLAN-WIFIVLAN_BACKUP	OK	05-11-2024 11:20:34	1d 23h 25m 35s	1/3	SNMP OK - up(1)
	Port GE0/0/21 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:20:35	1d 23h 27m 9s	1/3	SNMP OK - up(1)
	Port GE0/0/22 IF VEU MACROLAN BACKUP	OK	05-11-2024 11:20:35	1d 23h 27m 12s	1/3	SNMP OK - up(1)
	Port GE0/0/41 MEGAFONIA ZONA 3	OK	05-11-2024 11:20:35	1d 23h 27m 9s	1/3	SNMP OK - up(1)
	Port GE0/0/47 MEGAFONIA ZONA 5	OK	05-11-2024 11:20:35	1d 23h 27m 13s	1/3	SNMP OK - up(1)
	Port GE0/0/48 MEGAFONIA ZONA 6	OK	05-11-2024 11:20:35	1d 23h 26m 13s	1/3	SNMP OK - up(1)
	Uptime	OK	05-11-2024 11:25:34	1d 23h 28m 12s	1/3	SNMP OK - Timeticks: (73554991) 8 days, 12:19:09.91

Figura 197

EST001	PING	OK	05-11-2024 11:20:35	1d 0h 59m 21s	1/3	PING OK - Packet loss = 0%, RTA = 7.32 ms
	Port Eth-Trunk1 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:20:35	1d 23h 28m 29s	1/3	SNMP OK - up(1)
	Port GE0/0/11 MACROLAN-WIFIVLAN_PRINCIPAL NADM 9497	OK	05-11-2024 11:20:35	1d 23h 28m 44s	1/3	SNMP OK - up(1)
	Port GE0/0/27 CENTRALETA MCP40 - EST	OK	05-11-2024 11:20:35	1d 23h 28m 40s	1/3	SNMP OK - up(1)
	Port GE0/0/45 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:20:35	1d 23h 28m 43s	1/3	SNMP OK - up(1)
	Port GE0/0/46 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:20:34	1d 23h 28m 29s	1/3	SNMP OK - up(1)
	Port GE0/0/47 MACROLAN-DADES-BACKUP NADM 8353	OK	05-11-2024 11:20:35	1d 23h 28m 0s	1/3	SNMP OK - up(1)
	Port GE0/0/33 MEGAFONIA GENERAL	OK	05-11-2024 11:20:35	1d 17h 41m 45s	1/3	SNMP OK - up(1)
	Port GE0/0/34 MEGAFONIA PLANTA 0	OK	05-11-2024 11:20:35	1d 23h 28m 43s	1/3	SNMP OK - up(1)
	Port GE0/0/35 MEGAFONIA PLANTA 1	OK	05-11-2024 11:20:34	1d 23h 22m 8s	1/3	SNMP OK - up(1)
	Port GE0/0/36 MEGAFONIA PLANTA 2	OK	05-11-2024 11:20:35	1d 23h 28m 40s	1/3	SNMP OK - up(1)
	Port GE0/0/40 MODUL SUPERVIVENCIA VEU CTTI	OK	05-11-2024 11:20:35	1d 23h 28m 40s	1/3	SNMP OK - up(1)
	Port GE0/0/42 FTTH VEU	OK	05-11-2024 11:20:35	1d 23h 31m 2s	1/3	SNMP OK - up(1)
	Port GE0/0/43 SERVIDOR_XARXADADES	OK	05-11-2024 11:20:35	1d 23h 28m 40s	1/3	SNMP OK - up(1)
	Port GE0/0/45 MACROLAN-PRINCIPAL-DADES NADM 9497	OK	05-11-2024 11:20:35	1d 23h 28m 40s	1/3	SNMP OK - up(1)
	Port GE0/0/46 SERVIDOR-XARXAVEU	OK	05-11-2024 11:20:35	1d 23h 28m 44s	1/3	SNMP OK - up(1)
	Port GE0/0/7 MACROLAN-WIFIVLAN_BACKUP NADM 8353	OK	05-11-2024 11:20:35	1d 23h 28m 43s	1/3	SNMP OK - up(1)
	Uptime	OK	05-11-2024 11:28:34	1d 23h 31m 49s	1/3	SNMP OK - Timeticks: (2564888676) 296 days, 20:41:26.76

Figura 198

HSJ001	PING	OK	05-11-2024 11:20:35	1d 23h 22m 59s	1/3	PING OK - Packet loss = 0%, RTA = 15.89 ms
	Port Ethernet1/0/12 MACROLAN PRINCIPAL - VEU	OK	05-11-2024 11:20:35	1d 23h 27m 59s	1/3	SNMP OK - up(1)
	Port Ethernet1/0/22 MACROLAN PRINCIPAL - DADES	OK	05-11-2024 11:20:35	1d 23h 28m 44s	1/3	SNMP OK - up(1)
	Port Ethernet1/0/23 MACROLAN BACKUP - VEU	OK	05-11-2024 11:20:34	1d 23h 28m 44s	1/3	SNMP OK - up(1)
	Port Ethernet1/0/24 MACROLAN BACKUP - DADES	OK	05-11-2024 11:20:35	1d 23h 28m 26s	1/3	SNMP OK - up(1)
	Port Ethernet1/0/7 AP-WIFI	OK	05-11-2024 11:20:35	1d 23h 28m 43s	1/3	SNMP OK - up(1)
	Uptime	OK	05-11-2024 11:28:34	1d 23h 29m 53s	1/3	SNMP OK - Timeticks: (500356122) 57 days, 21:52:41.22

Figura 199

HSL-CSMA001	PING	OK	05-11-2024 11:30:35	1d 23h 30m 13s	1/3	PING OK - Packet loss = 0%, RTA = 4.08 ms
	PORT XGE0/0/1 ENLACE HSL001	OK	05-11-2024 11:30:35	1d 23h 29m 53s	1/3	SNMP OK - up(1)
	Uptime	OK	05-11-2024 11:28:34	1d 23h 30m 44s	1/3	SNMP OK - Timeticks: (440553667) 50 days, 23:45:36.67

Figura 200

HSL001	PING	OK	05-11-2024 11:30:35	1d 23h 30m 14s	1/3	PING OK - Packet loss = 0%, RTA = 12.54 ms
	Port GE0/0/17 MACROLAN PPAL-VEU HSL	OK	05-11-2024 11:30:36	1d 23h 30m 14s	1/3	SNMP OK - up(1)
	Port GE0/0/18 MACROLAN PPAL-DADES CSL	OK	05-11-2024 11:30:35	1d 23h 29m 52s	1/3	SNMP OK - up(1)
	Port GE0/0/19 MACROLAN PPAL-VEU CSL	OK	05-11-2024 11:30:35	1d 23h 29m 27s	1/3	SNMP OK - up(1)
	Port GE0/0/28 CENTRALETA SV9100	OK	05-11-2024 11:30:34	1d 23h 30m 14s	1/3	SNMP OK - up(1)
	Port GE0/0/36 MACROLAN-WIFIVLAN_PRINCIPAL	OK	05-11-2024 11:30:36	1d 23h 30m 14s	1/3	SNMP OK - up(1)
	Port GE1/0/1 Servidor HSL - Pota de Veu	OK	05-11-2024 11:30:35	1d 23h 29m 48s	1/3	SNMP OK - up(1)
	Port GE1/0/11 MACROLAN-WIFIVLAN_BACKUP	OK	05-11-2024 11:30:35	1d 23h 30m 10s	1/3	SNMP OK - up(1)
	Port GE1/0/34 Servidor HSL - Pota LAN	OK	05-11-2024 11:30:35	1d 23h 30m 14s	1/3	SNMP OK - up(1)
	Port GE1/0/5 ENLACE HSL001-SEVAD	OK	05-11-2024 11:30:35	1d 23h 30m 10s	1/3	SNMP OK - up(1)
	Port GE0/0/17 MACROLAN BACKUP-VEU HSL	OK	05-11-2024 11:30:34	1d 23h 29m 48s	1/3	SNMP OK - up(1)
	Port GE0/0/18 MACROLAN BACKUP-DADES CSL	OK	05-11-2024 11:30:35	1d 23h 30m 14s	1/3	SNMP OK - up(1)
	Port GE0/0/19 MACROLAN BACKUP-VEU CSL	OK	05-11-2024 11:30:34	1d 23h 30m 10s	1/3	SNMP OK - up(1)
	Port XGE0/0/2 ENLACE HSL-CSMA	OK	05-11-2024 11:30:35	1d 23h 30m 14s	1/3	SNMP OK - up(1)
	Port XGE0/0/4 LINK HSL-CSL-EUIT	OK	05-11-2024 11:30:35	1d 23h 29m 43s	1/3	SNMP OK - up(1)
	Port XGE0/0/4 LINK HSL-CSL-EUIT	OK	05-11-2024 11:30:35	1d 23h 28m 33s	1/3	SNMP OK - up(1)
	Uptime	OK	05-11-2024 11:28:34	1d 23h 30m 41s	1/3	SNMP OK - Timeticks: (1735630506) 200 days, 21:11:45.06

Figura 201

MDP001	PING	OK	05-11-2024 11:30:36	1d 23h 30m 14s	1/3	PING OK - Packet loss = 0%, RTA = 3.42 ms
	Port Eth-Trunk1 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:30:34	1d 23h 29m 42s	1/3	SNMP OK - up(1)
	Port GE0/0/1 MACROLAN PRINCIPAL DADES	OK	05-11-2024 11:30:35	1d 23h 30m 28s	1/3	SNMP OK - up(1)
	Port GE0/0/2 FORTIGATE - Port 01	OK	05-11-2024 11:30:35	1d 23h 30m 14s	1/3	SNMP OK - up(1)
	Port GE0/0/4 CENTRALETA MCP40 - MDP	OK	05-11-2024 11:30:35	1d 23h 30m 10s	1/3	SNMP OK - up(1)
	Port GE0/0/44 MODUL SUPERVIVENCIA CTTI-VEU	OK	05-11-2024 11:30:35	1d 23h 29m 41s	1/3	SNMP OK - up(1)
	Port GE0/0/45 VEU - MACROLAN PRINCIPAL	OK	05-11-2024 11:30:35	1d 23h 28m 29s	1/3	SNMP OK - up(1)
	Port GE0/0/47 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:30:35	1d 23h 30m 13s	1/3	SNMP OK - up(1)
	Port GE0/0/48 MACROLAN-WIFIVLAN_PRINCIPAL	OK	05-11-2024 11:30:35	1d 23h 29m 12s	1/3	SNMP OK - up(1)
	Port GE8/0/1 MACROLAN BACKUP DADES	OK	05-11-2024 11:30:35	1d 23h 29m 40s	1/3	SNMP OK - up(1)
	Port GE8/0/2 FORTIGATE - Port 02	OK	05-11-2024 11:30:35	1d 23h 24m 24s	1/3	SNMP OK - up(1)
	Port GE8/0/46 VEU MACROLAN BACKUP	OK	05-11-2024 11:30:35	1d 23h 32m 25s	1/3	SNMP OK - up(1)
	Port GE8/0/47 LAG Switchos Extreme - Wifi	OK	05-11-2024 11:30:35	1d 23h 30m 13s	1/3	SNMP OK - up(1)
	Port GE8/0/48 MACROLAN-WIFIVLAN_BACKUP	OK	05-11-2024 11:30:34	1d 23h 29m 37s	1/3	SNMP OK - up(1)
	Uptime	OK	05-11-2024 11:28:34	1d 23h 30m 54s	1/3	SNMP OK - Timeticks: (1709964318) 197 days, 21:54:03.18

Figura 202

ROC001	PING	OK	05-11-2024 11:30:36	1d 23h 29m 23s	1/3	PING OK - Packet loss = 0%, RTA = 3.56 ms
	Port GE0/0/1 MACROLAN-DADES-PRINCIPAL	OK	05-11-2024 11:30:34	1d 23h 30m 13s	1/3	SNMP OK - up(1)
	Port GE0/0/2 MACROLAN-VEU-PRINCIPAL	OK	05-11-2024 11:30:35	1d 23h 29m 35s	1/3	SNMP OK - up(1)
	Port GE1/0/1 IF DADES SERVIDOR CENTRE	OK	05-11-2024 11:30:35	1d 23h 24m 22s	1/3	SNMP OK - up(1)
	Port GE8/0/1 MACROLAN-DADES-BACKUP	OK	05-11-2024 11:30:35	1d 23h 30m 14s	1/3	SNMP OK - up(1)
	Port GE8/0/17 Control Climes - RRF	OK	05-11-2024 11:30:35	1d 23h 30m 14s	1/3	SNMP OK - up(1)
	Port GE8/0/2 MACROLAN-VEU-BACKUP	OK	05-11-2024 11:30:35	1d 23h 29m 33s	1/3	SNMP OK - up(1)
	Port GE8/0/3 MCP-120 REGISTRE JORNADA ROC	OK	05-11-2024 11:30:35	1d 23h 32m 24s	1/3	SNMP OK - up(1)
	Uptime	OK	05-11-2024 11:28:34	1d 23h 31m 2s	1/3	SNMP OK - Timeticks: (86601455) 10 days, 0:33:34.55

Figura 203

Finalmente, si vamos al apartado Summary de Host Groups del sitio web de Nagios podemos apreciar que tenemos 16 equipos activos y 185 servicios en estado OK monitoreados:

Switch_Primary (Switch_Primary)	16 UP	185 OK
---------------------------------	-------	--------

Figura 204

- Switch: En esta plantilla se definen los Switch de la sede principal. Primero de todo definimos el host principal:

```
define host {
    use                generic-switch
    host_name          CORE_CST
    alias              CORE_CST
    address            10.68.100.22
    hostgroups         switches
}
```

Posteriormente, definimos el grupo:

```
define hostgroup {
    hostgroup_name     switches
    alias              Network Switches
}
```

Y finalmente, podremos monitorear el por ICMP el ping por ejemplo:

```
define service {
    use                generic-service
    hostgroup          switches
    service_description PING
    check_command      check_ping!200.0,20%!600.0,60%
    check_interval     1
    retry_interval     1
}
```

}

Y podremos ver su estado en Nagios Core:

Host	Service	Status	Last Check	Duration	Attempt	Status Information
CORE_CST	PING	OK	05-12-2024 09:50:06	0d 0h 6m 12s	1/3	PING OK - Packet loss = 0%, RTA = 1.14 ms

Figura 205




Network Switches (switches)			
Host	Status	Services	Actions
CORE_CST	UP	1 OK	  

Figura 206

5.3 Configurar copias de seguridad de los ficheros de configuración.

Primero de todo, en nuestra cabina de almacenamiento creamos la carpeta donde destinaremos las copias de seguridad:

<input type="checkbox"/>	Nombre
<input type="checkbox"/>	NAGIOS

Figura 207

Una vez creada, creamos el recurso compartido por el protocolo NFS:

Crear una carpeta compartida

Llene los siguientes campos para crear una carpeta compartida

Nombre de Carpeta:	<input type="text" value="NFSNAGIOS"/>
Comentarios (opcional):	<input type="text"/>
Volumen de disco:	<input type="text" value="DataVol1 (Tamaño del espacio libre: 55.32 TB)"/>
Organización automática por niveles de Qtier :	<input type="checkbox"/> Habilitar organización automática por niveles
Ruta:	<input type="radio"/> Especificar la ruta automáticamente <input checked="" type="radio"/> Introducir la ruta manualmente
	<input type="text" value="/BackupsNFS/NAGIOS"/>

Figura 208

Una vez creado el recurso, vamos a sus propiedades, al apartado NFS y le concedemos acceso al usuario invitado del servidor de monitorización Nagios Core:

Nombre de recurso: NFSNAGIOS

compartido de red:

Derecho de acceso

sync

secure

Dirección IP o nombre de dominio permitido

<input type="checkbox"/>	Host / IP / Red	Seguridad	Opción ...	Opción Squash	GID anónimo	UID anónimo
<input type="checkbox"/>	172.31.3.74	sys,	lectura y es	No asignar a nin...	guest	guest

Figura 209

Instalamos NFS:

```
root@monsrv01:/# apt-get install nfs-common
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
 keyutils libnfsidmap1 rpcbind
Paquetes sugeridos:
 watchdog
Se instalarán los siguientes paquetes NUEVOS:
 keyutils libnfsidmap1 nfs-common rpcbind
0 actualizados, 4 nuevos se instalarán, 0 para eliminar y 28 no actualizados.
Se necesita descargar 381 kB de archivos.
Se utilizarán 1.447 kB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] S
```

Figura 210

Creamos una carpeta donde haremos posteriormente el montaje del recurso compartido de la cabina de almacenamiento:

```
root@monsrv01:/# mkdir BCK_NAGIOS
root@monsrv01:/# ls
BCK_NAGIOS boot dev home lib32 libx32 media opt root sbin srv sys usr
bin cdrom etc lib lib64 lost+found mnt proc run snap swap.img tmp var
root@monsrv01:/#
```

Figura 211

Una vez creada a carpeta, hacemos el montaje con el commando: "mount bckcst01.hsp.csd.es:/NFSNAGIOS BCK_NAGIOS"

```
root@monsrv01:/# mount bckcst01.hsp.csd.es:/NFSNAGIOS BCK_NAGIOS
root@monsrv01:/# ls
BCK_NAGIOS boot dev home lib32 libx32 media opt root sbin srv sys usr
bin cdrom etc lib lib64 lost+found mnt proc run snap swap.img tmp var
root@monsrv01:/#
```


Figura 212

Para que el montaje sea permanente, debemos de editar el fichero “etc/fstab” de la siguiente manera:

```
/etc/fstab - root@172.31.3.74 - Editor - WinSCP
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point> <type> <options> <dump> <pass>
# / was on /dev/ubuntu-vg/ubuntu-lv during curtin installation
/dev/disk/by-id/dm-uuid-LVM-TeI0z10CvRZGhXinR1tSSn00dm0xwtyx3To0mQKW1a70TCVlWeDZe3ei98VkG17c / ext4 defaults 0 1
# /boot was on /dev/sda2 during curtin installation
/dev/disk/by-uuid/6a76bbd4-62f3-4a60-9906-8e6eb7493787 /boot ext4 defaults 0 1
/swap.img none swap sw 0 0
#MONTAJE BCK
bckcst01.hsp.csd.t.es:/NFSNAGIOS BCK_NAGIOS nfs4 defaults 0 0
```

Figura 213

Si hacemos un mount -a se monta:

```
root@monsrv01:/# ls
BCK_NAGIOS boot dev home lib32 libx32 media opt root sbin srv sys usr
bin cdrom etc lib lib64 lost+found mnt proc run snap swap.img sys var
root@monsrv01:/# mount -a
root@monsrv01:/# ls
BCK_NAGIOS boot dev home lib32 libx32 media opt root sbin srv sys usr
bin cdrom etc lib lib64 lost+found mnt proc run snap swap.img sys var
root@monsrv01:/#
```

Figura 214

Una vez tenemos el montaje, creamos un directorio “SCRIPT” y le subimos el siguiente Script que se encargará de montar la unidad si no está montada, hacer la copia de seguridad diaria de forma comprimida, asignarle la fecha, tendrá una retención de 1 semana (eliminará comprimidos antiguos), y enviará un mail al buzón de copias de seguridad indicando si se ha efectuado correctamente o no:

```
#!/bin/bash

#Montamos los recursos por si no estuvieran montados:
cd ..
mount -a

# Ruta de origen y destino
ruta_origen="/usr/local/nagios/"
ruta_destino="/BCK_NAGIOS/"

# Nombre del archivo de copia con fecha
fecha=$(date +%Y-%m-%d)
archivo_copia="nagios_backup_`date +%Y-%m-%d`.tar.gz"

# Realizar la copia y comprimir directamente en el destino
tar -czf "$ruta_destino$archivo_copia" -C "$(dirname "$ruta_origen")" "$(basename "$ruta_origen")"

# Eliminar copias antiguas (mayores a una semana)
find "$ruta_destino" -type f -name "nagios_backup_*.tar.gz" -mtime +7 -delete

# Verificar si la copia se realizó correctamente
if [ $? -eq 0 ]; then
    estado="éxito"
else
    estado="fracaso"
fi

# Enviar correo electrónico
correo_destino="backups@cst.cat"
asunto="Copia de seguridad de Nagios ($fecha)"
mensaje="La copia de seguridad de Nagios realizada el $fecha se ha completado con $estado."

echo "$mensaje" | mail -s "$asunto" "$correo_destino"

exit 0
```

Figura 215

Una vez se verifica el correcto funcionamiento del Script, se debe de programar una tarea para que sea ejecutado cada día de forma local, a las 3 de la mañana. Para ello, editamos el fichero “etc/crontab” y añadimos la ultima línea:

```
/etc/crontab - root@172.31.3.74 - Editor - WinSCP

# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
# You can also override PATH, but by default, newer versions inherit it from the environment
#PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# Example of job definition:
# .----- minute (0 - 59)
# | .----- hour (0 - 23)
# | | .----- day of month (1 - 31)
# | | | .----- month (1 - 12) OR jan,feb,mar,apr ...
# | | | | .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# | | | | |
# * * * * * user-name command to be executed
17 * * * * root cd / && run-parts --report /etc/cron.hourly
25 6 * * * root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6 * * 7 root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6 1 * * root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
# Backup NAGIOS Core
0 3 * * * /SCRIPT/Backup_Nagios.sh
```

Figura 216

6. Conclusiones

En general ha sido un proyecto muy satisfactorio. He aprendido mucho durante el transcurso de estos meses. Como lecciones aprendidas podría destacar las siguientes:

- Instalación y configuración de Nagios Core.
- Instalación y uso de Postfix como relay con SMTP local.
- Limitación de acceso al sitio web de Nagios Core.
- Uso de protocolos de monitorización de Nagios Core
- Aprendizaje sobre SSL/TLS así como funcionan los certificados y su uso.
- Aprendizaje y configuración de los diferentes clientes y plugins de monitorización.
- Uso de SNMP así como de MIB Browser y conocer sus diferentes versiones.
- Creación de plantillas para agrupas por categorías de los diferentes equipos.
- Configuración de copias de seguridad en UNIX.

Sobre los objetivos propuestos al inicio tan solo no he podido cumplir con uno ya que no ha sido posible. El objetivo en cuestión en la monitorización de CRC de las interfaces de los Switch pero con SNMP no es posible monitorear dicho parámetro.

Referente a la planificación del proyecto, considero que ha sido bastante buena, ya que para lo que no tenía tales conocimientos le he dedicado y pautado más tiempo y para lo que tenía más asumido menos. En general la planificación ha sido adecuada y se ha adaptado perfectamente al escenario.

Como líneas de trabajo futuro, quedará la implementación de más equipos y servicios para la monitorización. Durante el transcurso de este verano se irán implementando. Adicionalmente, se mirará de optimizar Nagios Core añadiéndole integraciones como con Grafana para visualizar gráficos o Nagvis para poder crear mapas topológicos.

7. Glosario

NCPA: “Nagios Cross-Platform Agent”, Agente Cruzado de Plataforma Nagios.

NSCA: “Nagios Service Check Acceptor”, Aceptor de Verificación de Servicio Nagios.

NRPE: “Nagios Remote Plugin Executor”, Ejecutor de Plugin Remoto Nagios.

NRPD: “Nagios Remote Data Processor,” Procesador de Datos Remotos Nagios.

SNMP: “Simple Network Management Protocol”, Protocolo Simple de Administración de Red.

MIBS: “Management Information Base”, Base de Información de Administración.

DNS: “Domain Name Server”, Servidor de Nombres de Dominio.

HTTP: “HyperText Transfer Protocol”, Protocolo de Transferencia de HiperTexto.

HTTPS: “HyperText Transfer Protocol Secure”, Protocolo de Transferencia de HiperTexto Seguro.

POP: “Post Office Protocol”, Protocolo de Oficina de Correos.

IMAP: “Internet Message Access Protocol”, Protocolo de Acceso a Mensajes de Internet.

SSH: “Secure Shell”, Shell Seguro.

LDAP: “Lightweight Directory Access Protocol”, Protocolo Ligero de Acceso a Directorios.

FTP: “File Transfer Protocol”, Protocolo de Transferencia de Archivos.

SFTP: “Secure File Transfer Protocol”, Protocolo de Transferencia de Archivos Seguro.

TCP: “Transmission Control Protocol”, Protocolo de Control de Transmisión.

UDP: “User Datagram Protocol”, Protocolo de Datagrama de Usuario.

ICMP: “Internet Control Message Protocol”, Protocolo de Mensajes de Control de Internet.

PING: “Packet Internet Groper”, Buscador de Paquetes de Internet.

8. Bibliografía

- [1] Descarga de Ubuntu Server:
<https://ubuntu.com/download>. Visitado en marzo del 2024.
- [2] Información sobre Nagios Core:
<https://www.openitnet.com/index.php/software/inst-software-libre/nagios-core>
Visitado en marzo del 2024.
- <https://www.ionos.es/digitalguide/servidores/herramientas/nagios-todos-los-procesos-de-red-a-tu-alcance/> Visitado en marzo del 2024.
- [3] Información sobre Zabbix:
http://911-ubuntu.weebly.com/zabbix_como_funciona/conoce-la-estructura-de-zabbix-y-como-usarlo Visitado en marzo del 2024.
- [4] Información sobre Zenoss:
<https://serviciosderednoona.wordpress.com/sistema-de-monitoreo-zenoss-4-2/> Visitado en marzo del 2024.
- <https://prezi.com/owsuld8neber/zenoss/> Visitado en marzo del 2024.
- [5] Manual de instalación Nagios Core: <https://support.nagios.com/kb/article/nagios-core-installing-nagios-core-from-source-96.html#Ubuntu> Visitado en marzo del 2024.
- [6] Pagina oficial Nagios Core:
<https://www.nagios.org/downloads/nagios-core/> Visitado en marzo del 2024.
- [7] Manual actualización de Nagios Core:
<https://assets.nagios.com/downloads/nagioscore/docs/nagioscore/4/en/upgrading.html>
Visitado en marzo del 2024.
- [8] Instalación MIBS Ubuntu Server:
<https://www.ibm.com/docs/en/linux-on-z?topic=s-install-mibs> Visitado en abril del 2024.
- [9] SNMPV2-PDU Ubuntu Server:
<https://pastebin.com/raw/p3QyuXzZ> Visitado en abril del 2024.
- [10] Manual instalación NRPE Ubuntu Server:
<https://support.nagios.com/kb/article.php?id=515#Ubuntu> Visitado en abril del 2024.
- [11] Manual instalación NSCA Ubuntu Server:
<https://www.pluralsight.com/cloud-guru/labs/aws/configure-nagios-server-to-accept-passive-check-results-via-nsca> Visitado en abril del 2024.
- [12] Instalación NCPA Ubuntu Server:

<https://assets.nagios.com/downloads/nagioscore/docs/nagioscore/3/en/monitoring-windows.html> Visitado en abril del 2024.

[13] Instalación NRPD Ubuntu Server:

<https://support.nagios.com/kb/article/nrdp-installing-nrdp-from-source-602.html#Ubuntu> Visitado en abril del 2024.

[14] Certificación del sitio web de Nagios Core:

<https://support.nagios.com/kb/article/nagios-core-configuring-ssl-tls-595.html#Ubuntu> Visitado en marzo del 2024.

[15] Generación de CSR con nombre alternativo de sujeto:

<https://netassured.co.uk/openssl-certificate-signing-request-subject-alternative-name/> Visitado en marzo del 2024.

[16] Documentación NRPE:

<https://www.sysadminsdecuba.com/2019/09/monitoreo-de-la-red-con-nagios-parte-iii/#:~:text=7.1.-,%C2%BFQu%C3%A9%20es%20NRPE%3F,los%20programas%20necesarios%20para%20ello> Visitado en abril del 2024.

[17] Documentación NSCA:

<https://support.nagios.com/kb/article/nsca-overview-78.html> Visitado en abril del 2024.

[18] Documentación SNMP:

https://es.wikipedia.org/wiki/Protocolo_simple_de_administraci%C3%B3n_de_red Visitado en abril del 2024.

[19] Documentación Check_nt:

https://shinken.readthedocs.io/en/1.4.2/02_gettingstarted/gettingstarted-monitoring-windows.html Visitado en abril del 2024.

[20] Documentación NRPE:

<https://medium.com/@shan1024/nrpe-780e0fb15bb3#:~:text=NRPE%20allows%20you%20to%20remotely,remote%20Windows%20machines%20as%20well>. Visitado en abril del 2024.

<https://assets.nagios.com/downloads/nagioscore/docs/nrpe/NRPE.pdf> Visitado en abril del 2024.

[21] Documentación NSCA:

<https://support.nagios.com/kb/article/nsca-overview-78.html#:~:text=NSCA%20is%20a%20Nagios%20service,to%20Nagios%20as%20Passive%20checks>. Visitado en abril del 2024.

[22] Agente NRPE:

<https://assets.nagios.com/downloads/nagiosxi/agents/linux-nrpe-agent.tar.gz> Visitado en abril del 2024.

- [23] Agente NSCA:
<https://github.com/NagiosEnterprises/nsca/releases> Visitado en abril del 2024.
- [24] Agente NCPA:
<https://www.nagios.org/ncpa/> Visitado en abril del 2024.
- [25] Documentación plantillas Nagios Core:
<https://www.soluciones-im.com/es/estructura-de-archivos-de-nagios-core> Visitado en abril del 2024.
- [26] Documentación inicial NSClient++:
<https://github.com/mickem/nscp/issues/428> Visitado en abril del 2024.
- [27] Uso NRPE:
<https://www.bujarra.com/nagios-monitorizando-nrpe/> Visitado en abril del 2024.
https://support.nagios.com/kb/article/nrdp-send_nrdp-client-599.html Visitado en abril del 2024.
- [28] uso de NSCA con NSClient++:
<https://nagiosenterprises.my.site.com/support/s/article/Using-NSClient-for-Passive-Checks-68988c19> Visitado en abril del 2024.
<https://www.nagios.org/ncpa/help.php#passive> Visitado en abril del 2024.
- [29] Uso de NCPA clientes Windows:
<https://assets.nagios.com/downloads/nagioscore/docs/nagioscore/3/en/monitoring-windows.html> Visitado en abril del 2024.
- [30] Uso de NSCA entorno Unix:
<https://www.pluralsight.com/cloud-guru/labs/aws/configure-nagios-server-to-accept-passive-check-results-via-nsca> Visitado en abril del 2024.
- [31] Configuración NCPA entorno Unix:
<https://www.nagios.org/ncpa/help/1.8/configuration.html> Visitado en abril del 2024.
- [32] Descarga Mib Browser:
<https://www.ireasoning.com/mibbrowser.shtml>. Visitado en abril del 2024.
- [33] Monitorización con SNMP:
<https://www.psychz.net/client/question/en/configure-nagios-snmp-monitoring.html>
Visitado en abril del 2024.
- [34] ISO Ubuntu Server:
<https://ubuntu.com/download> Visitado en abril del 2024.
- [35] Instalación y configuración Postfix:
<https://shape.host/resources/configurar-un-correo-web-en-ubuntu-22-04-una-guia-paso-a-paso-es> Visitado en abril del 2024.

[36] Descarga NSClient++:

<https://github.com/mickem/nscp/releases> Visitado en abril del 2024.

[37] Check_NT:

https://www.monitoring-plugins.org/doc/man/check_nt.html Visitado en abril del 2024.

[38] Certificación NRPE:

<https://support.nagios.com/kb/article.php?id=519> Visitado en abril del 2024.

[39] NRPD Nagios Core:

<https://support.nagios.com/kb/article/nrdp-installing-nrdp-from-source-602.html>

Visitado en abril del 2024.

9. Anexos

9.1 Instalación de Ubuntu Server

Para poder alcanzar el objetivo planteado, primero de todo se debe de instalar el sistema operativo que será responsable de gestionar los recursos del sistema, proporcionar una interfaz para la interacción entre el usuario y la computadora, y coordinar las actividades de hardware y software. Para ello, primero de todo debemos de descargar la ISO desde la página oficial: <https://ubuntu.com/download> [34].

Una vez hemos descargada la imagen se debe instalar, ya puede ser en un equipo físico o virtual. En este caso, se va a configurar una nueva máquina virtual en vSphere. Para su creación debemos de asignarle un nombre a la nueva máquina virtual para poder identificarla en la granja, en este caso: "MONSRV01", posteriormente, se debe de seleccionar el datastore donde se almacenará, después de ello la distribución a implementar y que recursos son necesarios. En este caso se le asigna 2 CPU, 4 GB de memoria ram y 60 GB de disco:

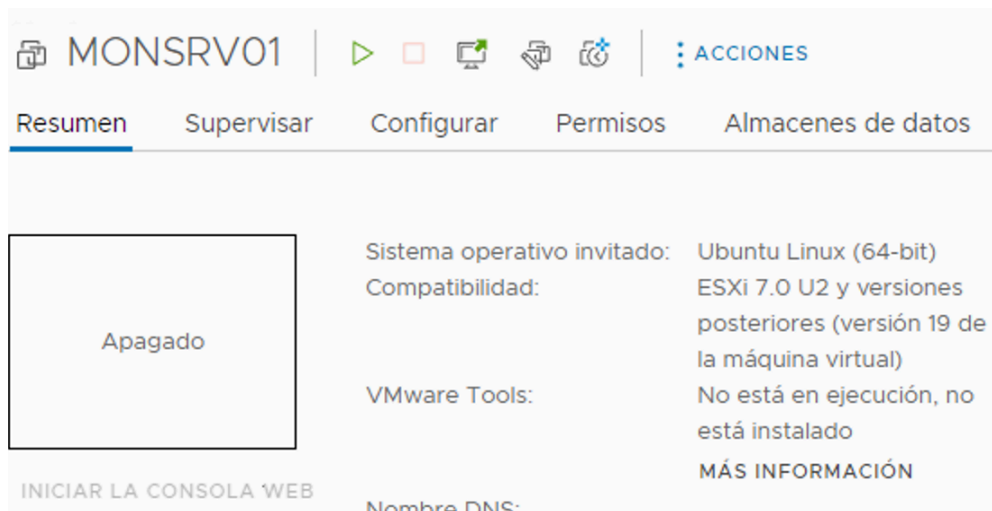


Figura 217

Una vez tenemos la máquina virtual creada, la arrancamos con la imagen ISO descargada y procedemos con su instalación:



Figura 218

Durante el proceso de instalación deberemos especificar el idioma de la distribución, tanto para la interfaz como para la distribución del teclado. Posteriormente, se debe de indicar que distribución se quiere instalar la distribución, en este caso la de Ubuntu Server. Editamos la interfaz de red para asignarle una IP fija, donde aparte de indicarle la dirección, se le debe establecer: subred, puerta de enlace, servidores de resolución de nombres de dominio y los dominios de búsqueda:

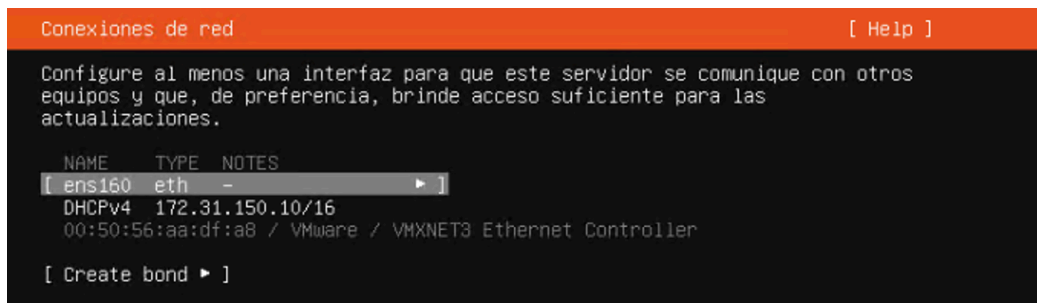


Figura 219

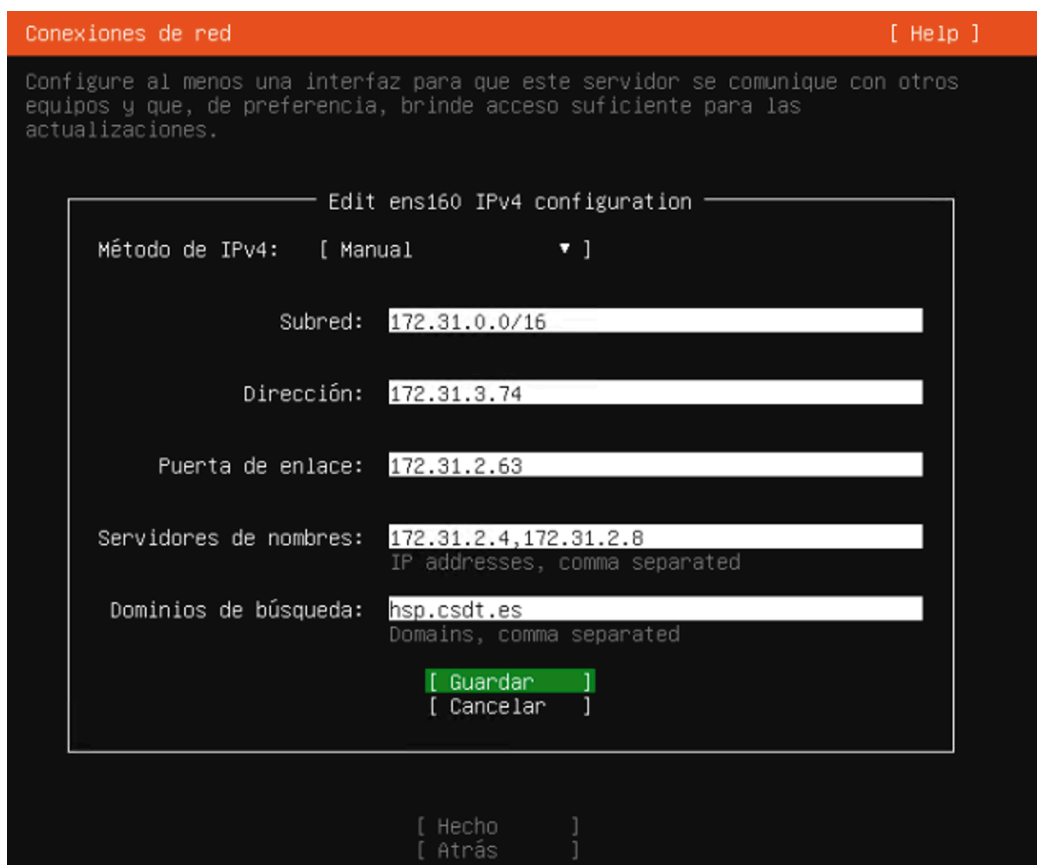


Figura 220

Una vez se tiene operativa la interfaz de red se procede con el disco, donde se selecciona el disco entero para su instalación. Después de esto, se le debe indicar el nombre que tendrá este equipo "hostname", el nombre del usuario, así como su usuario y la credencial:



Figura 221

Habilitaremos la instalación del servidor OpenSSH server que nos permitirá establecer conexiones encriptadas con nuestro host de forma remota:

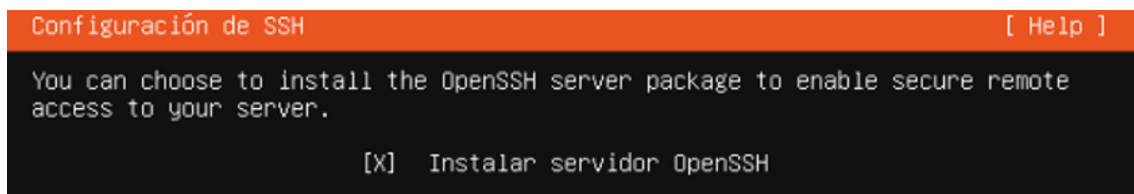


Figura 222

Con todo ello configurado, se procede con la instalación del sistema operativo y tras reiniciarse y desmontar la imagen ISO se tiene operativo para proceder con su configuración.

9.2 Configuración de Ubuntu Server

Se pretende habilitar que el usuario “root” pueda autenticarse para establecer conexiones por el protocolo SSH. Para ello, se debe de editar el fichero ubicado en “/etc/ssh/sshd_config” con un editor de texto como nano. Debemos de cambiar el “PermitRootLogin prohibit-password” por “PasswordRootLogin Yes” y “PasswordAuthentication Yes” descomentando las líneas:

```
# Authentication:

#LoginGraceTime 2m
#PermitRootLogin prohibit-password
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```

Figura 223

```
# To disable tunneled clear text passwords, change to no here!  
#PasswordAuthentication yes  
#PermitEmptyPasswords no
```

Figura 224

Cambio efectuado:

```
# Authentication:  
  
#LoginGraceTime 2m  
PermitRootLogin yes  
#StrictModes yes  
#MaxAuthTries 6  
#MaxSessions 10
```

Figura 225

```
# To disable tunneled clear text passwords, change to no here!  
PasswordAuthentication yes  
#PermitEmptyPasswords no
```

Figura 226

Posteriormente se debe guardar y reiniciar el servidor SSH con el comando “service ssh restart”:

```
root@monsrv01:/etc/ssh# service ssh restart  
root@monsrv01:/etc/ssh#
```

Figura 227

Una vez reiniciado, se podrá acceder como usuario root y establecer una credencial segura. Para la conexión se puede hacer uso de la herramienta Putty:

```
root@monsrv01:/home/moncst# su root  
root@monsrv01:/home/moncst#  
root@monsrv01:/home/moncst#  
root@monsrv01:/home/moncst#  
root@monsrv01:/home/moncst# passwd  
New password:  
Retype new password:  
passwd: password updated successfully  
root@monsrv01:/home/moncst#  
root@monsrv01:/home/moncst#  
root@monsrv01:/home/moncst#
```

Figura 228

Por último, deberemos de establecer la zona horaria, para ello ejecutamos el siguiente comando: `timedatectl set-timezone Europe/Madrid`:

```
root@monsrv01:~# timedatectl set-timezone Europe/Madrid
root@monsrv01:~#
```

Figura 229

9.3 Instalación de un servidor de correo electrónico en Ubuntu Server

Para poder recibir alertas a determinados correos electrónicos, se debe de instalar un servidor de correo [35]. El servidor de correo que se va a proceder a instalar es Postfix. Proporciona un sistema de entrega de correo electrónico eficiente y confiable, soportando protocolos estándar como SMTP (Simple Mail Transfer Protocol). Para ello, primero de todo, vamos a instalar Postfix en nuestro Ubuntu Server, para ello, ejecutaremos el siguiente comando “`sudo apt-get install postfix`” y seleccionamos sitio de internet:

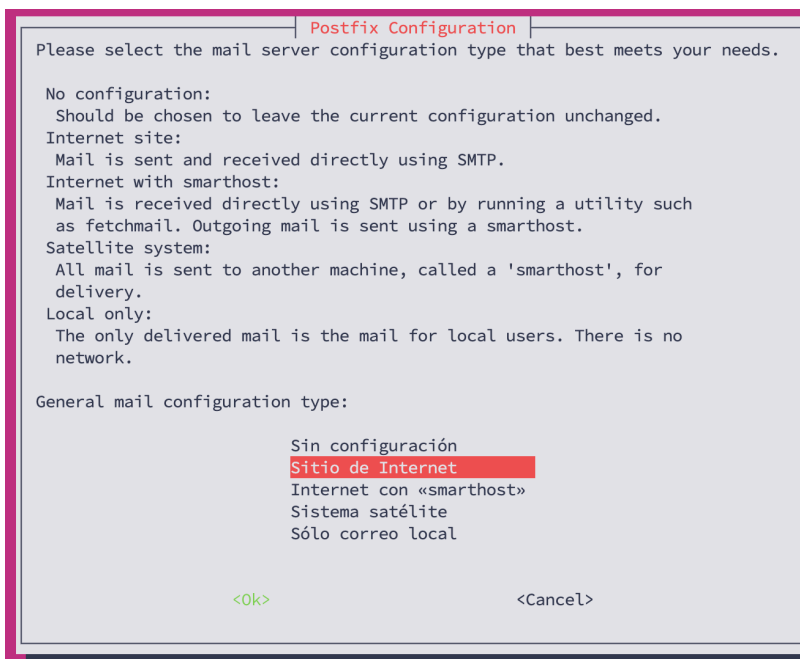


Figura 230

En este caso, se deja el nombre de dominio por defecto desde el cual se enviarán los correos electrónicos:



Figura 231

Posteriormente, se reinicia el servicio del Postfix con el comando: “sudo systemctl restart postfix”:

```
root@monsrv01:~# sudo systemctl restart postfix
root@monsrv01:~#
```

Figura 232

Y se ejecuta el siguiente comando para instalar mailutils que proporciona utilidades como sendmail, mailq etc:

```
root@monsrv01:~# apt-get install mailutils
```

Figura 233

Después de instalar “mailutils”, se verifica si funciona correctamente emitiendo un correo electrónico:

```
root@monsrv01:~# echo "Este es un correo de prueba" | mail -s "Correo de prueba" jmendez@cst.ca
t
```

Figura 234

Se verificamos que se recibe correctamente:



Figura 235

9.4 Instalación de clientes para Nagios Core en entornos Windows.

A continuación, se va a explicar cómo se instalan los diferentes clientes en entornos Windows:

NSCLIENT++

Como se ha explicado anteriormente, este cliente admite diversas formas de configuración. Primero de todo se debe de descargar el agente desde el siguiente enlace: <https://github.com/mickem/nscp/releases> [36]. Una vez descargado, ejecutamos el .msi e indicamos que contraseña se va a utilizar y seleccionamos las siguientes opciones:

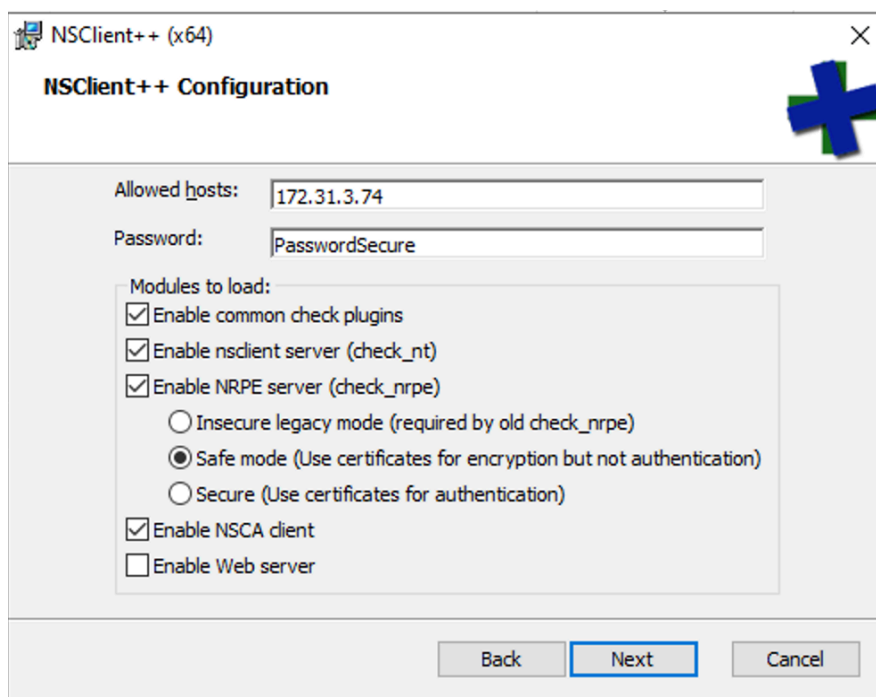


Figura 236

CHECK_NT [37]

Para que se pueda monitorear mediante NSClient++ Check_NT debemos de habilitar el comando, se le debe de establecer la contraseña definida en la instalación para que se proceda al uso de la autenticación. Para ello, se debe de editar el fichero “commands.cfg” ubicado en “usr/local/nagios/etc/objects” y modificar el comando de la siguiente manera:

```
define command {  
    command_name    check_nt  
    command_line    $USER1$/check_nt -H $HOSTADDRESS$ -p 12489 -s PasswordSecure -v $ARG1$ $ARG2$  
}
```

Figura 237

Certificado emitido

Se emitió el certificado que ha solicitado.

Codificado en DER o Codificado en Base64



[Descargar certificado](#)

[Descargar cadena de certificado](#)

Figura 240

Una vez descargado, en este caso le asignamos el nombre de “nrpe_dh_2048.pem” y lo copiamos en el directorio “C:\Program Files\NSClient++\security” de nuestro cliente Windows. Posteriormente, se debe de abrir un terminal CMD y ejecutar los siguientes comandos:

```
cd "\Program Files\NSClient++"  
nscp settings --path /settings/NRPE/server --key dh --set "${certificate-path}/nrpe_dh_2048.pem"
```

```
C:\Users\27537>cd "\Program Files\NSClient++"  
C:\Program Files\NSClient++>nscp settings --path /settings/NRPE/server --key dh --set "${certificate-path}/nrpe_dh_2048.pem"  
C:\Program Files\NSClient++>_
```

Figura 241

Y reiniciar el servicio.

Una se ha configurado el cliente, se debe instalar en el servidor de monitorización el certificado de la entidad certificadora. Para ello se debe acceder a la URL de la entidad y descargar su certificado:

Servicios de certificados de Active Directory de Microsoft -- Consorci Sanitari de Terrassa

Descargar certificado de CA, cadena de certificados o CRL

Para confiar en los certificados emitidos por esta entidad de certificación, [instale este certificado de CA](#).

Para descargar un certificado de CA, una cadena de certificados o una lista de revocación de certificados, seleccione el certificado y método de codificación.

Certificado de CA: _____

Figura 242

Una vez descargado, se debe copiar en la ruta “/usr/local/share/ca-certificates”:

```
root@monsrv01:/usr/local/share/ca-certificates# ls  
entidad.crt
```

Figura 243

Posteriormente se debe de actualizar el almacén de certificados con el comando “sudo update-ca-certificates” y después editar el fichero de configuración “/usr/local/nagios/etc/nrpe.cfg” y para añadir las rutas de los certificados. Se debe de indicar el certificado de la entidad certificadora, el certificado firmado por la entidad (el mismo utilizado en el webservice de Apache2) y la correspondiente clave con la que se generó el certificado del servidor de monitorización:

```
ssl_cacert_file=/usr/local/share/ca-certificates/entidad.crt
ssl_cert_file=/etc/ssl/certs/certnew.crt
ssl_privatekey_file=/etc/ssl/private/keyfile.key
```

Figura 244

Configuramos también la siguiente línea para requerir un certificado por parte del cliente:

```
# SSL USE CLIENT CERTS
# This options determines client certificate usage.
# Values: 0 = Don't ask for or require client certificates (default)
#         1 = Ask for client certificates
#         2 = Require client certificates

ssl_client_certs=2
```

Figura 245

Se debe de guardar los cambios y reiniciar el servicio, se verifica que funciona correctamente:

```
root@monsrv01:/usr/local/share/ca-certificates# sudo systemctl restart nrpe.service
root@monsrv01:/usr/local/share/ca-certificates# sudo systemctl status nrpe.service
● nrpe.service - Nagios Remote Plugin Executor
   Loaded: loaded (/lib/systemd/system/nrpe.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2024-03-28 14:14:54 CET; 18s ago
     Docs: http://www.nagios.org/documentation
   Main PID: 14124 (nrpe)
    Tasks: 1 (limit: 4557)
   Memory: 1.4M
      CPU: 25ms
   CGroup: /system.slice/nrpe.service
           └─14124 /usr/local/nagios/bin/nrpe -c /usr/local/nagios/etc/nrpe.cfg -f

mar 28 14:14:54 monsrv01 systemd[1]: Started Nagios Remote Plugin Executor.
mar 28 14:14:54 monsrv01 nrpe[14124]: Starting up daemon
mar 28 14:14:54 monsrv01 nrpe[14124]: Server listening on 0.0.0.0 port 5666.
mar 28 14:14:54 monsrv01 nrpe[14124]: Server listening on :: port 5666.
mar 28 14:14:54 monsrv01 nrpe[14124]: Listening for connections on port 5666
mar 28 14:14:54 monsrv01 nrpe[14124]: Allowing connections from: 127.0.0.1,::1,172.31.3.74
root@monsrv01:/usr/local/share/ca-certificates# █
```

Figura 246

Posteriormente, se va a crear el comando para hacer uso de la versión 2 del protocolo NRPE. Para ello se debe editar el fichero “/usr/local/nagios/etc/commands.cfg” y definiremos el comando añadiéndole al final del todo el parámetro “-2”:

```
define command {

    command_name check_nrpe_windows
    command_line $USER1$/check_nrpe -H $HOSTADDRESS$ -c $ARG1$ $ARG2$ -2
}
```

Figura 247

NSCA [28]

Primero de todo se debemos de configurar nuestro agente NSCA de NSClient++, para ello, editamos el fichero de configuración para habilitar los siguientes módulos en el archivo “nsclient.ini” del cliente:

```
[/modules]
CheckSystem=enabled
CheckDisk=enabled
CheckExternalScripts=enabled
CheckHelpers=enabled
Scheduler=enabled
NSCAClient=enabled
```

Una vez habilitados los módulos, se deben de configurar las tareas programadas añadiendo lo siguiente en el mismo archivo:

```
[/settings/scheduler/schedules/foo]
command=bar
[/settings/scheduler/schedules/alias]
command=command
```

NCPA [29]

Para instalar el cliente, se debe de descargar desde la siguiente página web: <https://www.nagios.org/ncpa/>. Una vez descargado, se debe de ejecutar e indicar el token que utiliza el servidor para que se pueda establecer la autenticación entre servidor y cliente:

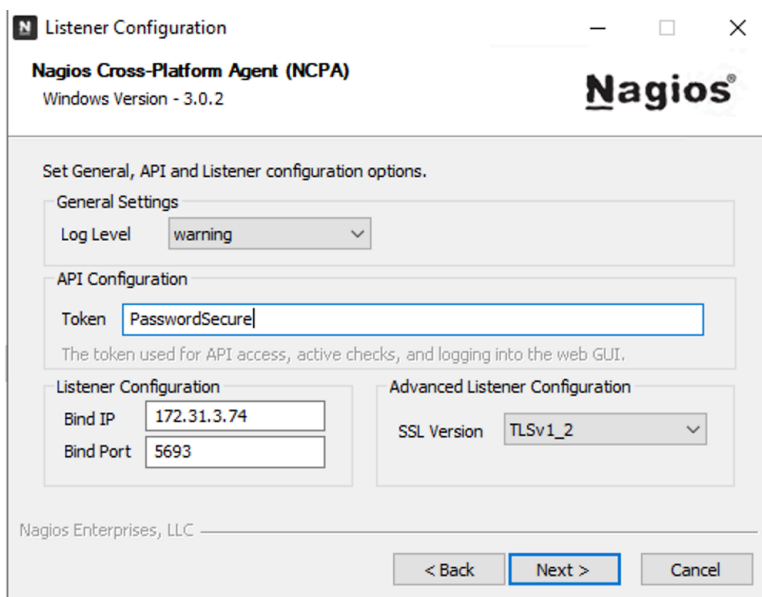


Figura 248

Configuramos también el apartado de checks pasivos NRDP:

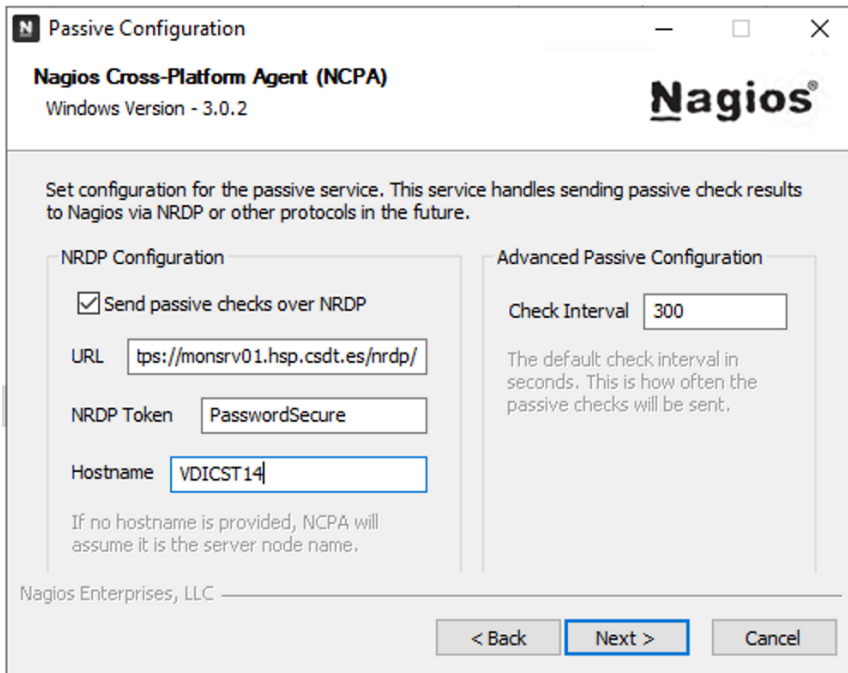


Figura 249

Como podemos apreciar una vez le damos a siguiente, nos indica:

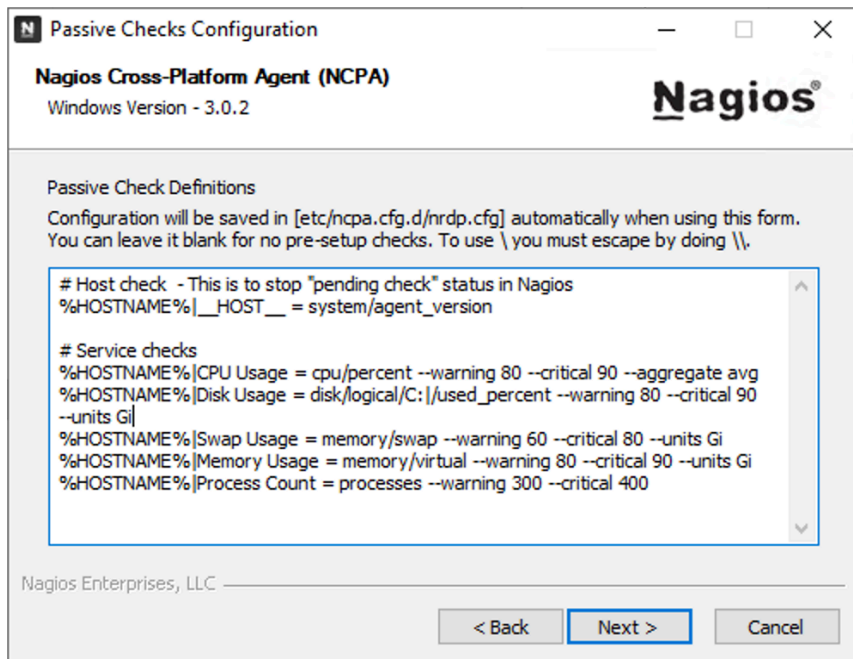


Figura 250

Instalamos para todos los usuarios:

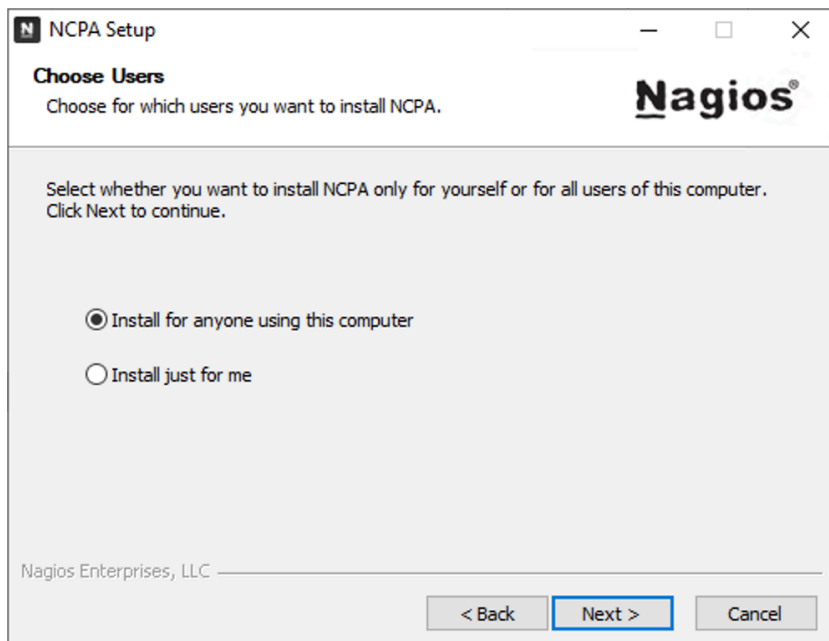


Figura 251

Una vez instalado, definimos el comando en el servidor de monitorización de Nagios Core:

```

/usr/local/nagios/etc/objects/commands.cfg - root@172.31.3.74 - Editor - WinSCP
[Icons] Codificación Color [?]

define command {
    command_name check_ncpa
    command_line $USER1$/check_ncpa.py -H $HOSTADDRESS$ $ARG1$
}

```

Figura 252

Una vez configurado NCPA, procedemos con NRDP para el envío de checks pasivos. Se establecen de forma muy sencilla, primero de todo, en el cliente vemos que envíos pasivos se realizan en la ruta "C:\Program Files\Nagios\NCPA\etc\ncpa.cfg.d\nrpd.cfg":

```

#
# AUTO GENERATED NRDP CONFIG FROM WINDOWS INSTALLER
#
[passive checks]

# Host check - This is to stop "pending check" status in Nagios
%HOSTNAME%|__HOST__ = system/agent_version

# Service checks
%HOSTNAME%|CPU Usage = cpu/percent --warning 80 --critical 90 --aggregate avg
%HOSTNAME%|Disk Usage = disk/logical/C:/used_percent --warning 80 --critical 90 --units Gi
%HOSTNAME%|Swap Usage = memory/swap --warning 60 --critical 80 --units Gi
%HOSTNAME%|Memory Usage = memory/virtual --warning 80 --critical 90 --units Gi
%HOSTNAME%|Process Count = processes --warning 300 --critical 400

```

Figura 253

9.5 Instalación de clientes para Nagios Core en entornos UNIX

A continuación, se va a explicar cómo se instalan los diferentes plugins en entornos Linux.

NRPE [27]

Para instalar el plugin en Linux se debe ejecutar el siguiente comando: “sudo apt install nagios-nrpe-server nagios-plugins”:

```
root@ubuntumanu:/home/josem# sudo apt install nagios-nrpe-server nagios-plugins
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Nota, seleccionando «monitoring-plugins» en lugar de «nagios-plugins»
Se instalarán los siguientes paquetes adicionales:
 libavahi-client3 libavahi-common-data libavahi-common3 libcups2 libdbi1 libldb2 libmysqlclient21
 libnet-snmp-perl libpq5 libradcli4 libsensors-config libsensors5 libsmbclient libsnmp-base
 libsnmp40 libtalloc2 libtdb1 libtevent0 liburiparser1 libwbclient0 monitoring-plugins-basic
 monitoring-plugins-common monitoring-plugins-standard mysql-common python3-gpg python3-ldb
 python3-samba python3-talloc python3-tdb rpcbind samba-common samba-common-bin
 samba-dsdb-modules samba-libs smbclient snmp
Paquetes sugeridos:
 cups-common libcrypt-des-perl libdigest-hmac-perl libio-socket-inet6-perl lm-sensors
 snmp-mibs-downloader icinga2 nagios-plugins-contrib fping postfix | sendmail-bin
 | exim4-daemon-heavy | exim4-daemon-light qstat xinetd | inetd heimdal-clients python3-markdown
 python3-dnspython cifs-utils
Se instalarán los siguientes paquetes NUEVOS:
 libavahi-client3 libavahi-common-data libavahi-common3 libcups2 libdbi1 libldb2 libmysqlclient21
 libnet-snmp-perl libpq5 libradcli4 libsensors-config libsensors5 libsmbclient libsnmp-base
 libsnmp40 libtalloc2 libtdb1 libtevent0 liburiparser1 libwbclient0 monitoring-plugins
 monitoring-plugins-basic monitoring-plugins-common monitoring-plugins-standard mysql-common
 nagios-nrpe-server python3-gpg python3-ldb python3-samba python3-talloc python3-tdb rpcbind
 samba-common samba-common-bin samba-dsdb-modules samba-libs smbclient snmp
0 actualizados, 38 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 16,1 MB de archivos.
Se utilizarán 74,6 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] _
```

Figura 254

Una vez instalado, se debe editar el fichero de configuración “/etc/nagios/nrpe.cfg” para indicarle la IP del servidor de monitorización Nagios Core en la línea “allowed_host”:

```
/etc/nagios/nrpe.cfg - josem@172.31.150.104 - Editor - WinSCP
Codificación Color ?

# ALLOWED HOST ADDRESSES
# This is an optional comma-delimited list of IP address or hostnames
# that are allowed to talk to the NRPE daemon. Network addresses with a bit mask
# (i.e. 192.168.1.0/24) are also supported. Hostname wildcards are not currently
# supported.
#
# Note: The daemon only does rudimentary checking of the client's IP
# address. I would highly recommend adding entries in your /etc/hosts.allow
# file to allow only the specified host to connect to the port
# you are running this daemon on.
#
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd

allowed_hosts=127.0.0.1,172.31.3.74
```

Figura 255

Posteriormente, como queremos habilitar una conexión cifrada mediante SSL/TLSv1.2 tenemos que configurarlo [38]. Para ello, primero de todo,

copiaremos la clave privada, el certificado cliente y el certificado de la entidad certificadora en los directorios correspondientes con sus respectivos permisos:

```
root@ubuntumanu:/etc/ssl# sudo cp clientNagios.crt /etc/ssl/certs
root@ubuntumanu:/etc/ssl# sudo cp keyfileclient.key /etc/ssl/private
root@ubuntumanu:/etc/ssl# sudo chmod go-rwx /etc/ssl/certs/clientNagios.crt
root@ubuntumanu:/etc/ssl# sudo chmod go-rwx /etc/ssl/private/keyfileclient.key
root@ubuntumanu:/etc/ssl# sudo cp entidad.crt /usr/local/share/ca-certificates
root@ubuntumanu:/etc/ssl# sudo chmod go-rwx /usr/local/share/ca-certificates/entidad.crt
root@ubuntumanu:/etc/ssl#
```

Figura 256

Y procedemos a instalar el certificado de la entidad certificadora:

```
root@ubuntumanu:/etc/ssl# sudo update-ca-certificates
Updating certificates in /etc/ssl/certs...
rehash: warning: skipping ca-certificates.crt,it does not contain exactly one certificate or CRL
1 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
root@ubuntumanu:/etc/ssl#
```

Figura 257

Posteriormente, editamos el fichero de configuración de nrpe ubicado en /etc/nagios/nrpe.cfg y añadimos las rutas de los correspondientes ficheros:

```
ssl_cacert_file=/usr/local/share/ca-certificates/entidad.crt
ssl_cert_file=/etc/ssl/certs/clientNagios.crt
ssl_privatekey_file=/etc/ssl/private/keyfileclient.key

# SSL USE CLIENT CERTS
# This options determines client certificate usage.
# Values: 0 = Don't ask for or require client certificates (default)
#         1 = Ask for client certificates
#         2 = Require client certificates

ssl_client_certs=2
```

Figura 258

Se reinicia el servicio NRPE y se verifica que funciona correctamente:

```
root@ubuntumanu:/usr/local/share/ca-certificates# systemctl restart nagios-nrpe-server
root@ubuntumanu:/usr/local/share/ca-certificates# systemctl status nagios-nrpe-server
● nagios-nrpe-server.service - Nagios Remote Plugin Executor
   Loaded: loaded (/lib/systemd/system/nagios-nrpe-server.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2024-03-29 09:13:33 UTC; 2s ago
     Docs: http://www.nagios.org/documentation
  Main PID: 1409 (nrpe)
    Tasks: 1 (limit: 4558)
   Memory: 1.6M
      CPU: 25ms
   CGroup: /system.slice/nagios-nrpe-server.service
           └─1409 /usr/sbin/nrpe -c /etc/nagios/nrpe.cfg -f

mar 29 09:13:33 ubuntumanu systemd[1]: Started Nagios Remote Plugin Executor.
mar 29 09:13:33 ubuntumanu nrpe[1409]: Starting up daemon
mar 29 09:13:33 ubuntumanu nrpe[1409]: Server listening on 0.0.0.0 port 5666.
mar 29 09:13:33 ubuntumanu nrpe[1409]: Server listening on :: port 5666.
mar 29 09:13:33 ubuntumanu nrpe[1409]: Listening for connections on port 5666
mar 29 09:13:33 ubuntumanu nrpe[1409]: Allowing connections from: 127.0.0.1,172.31.3.74
root@ubuntumanu:/usr/local/share/ca-certificates#
root@ubuntumanu:/usr/local/share/ca-certificates#
```

Figura 259

Posteriormente, se debe de crear el correspondiente comando NRPE en la plantilla de comandos de nuestro servidor de monitorización:

```

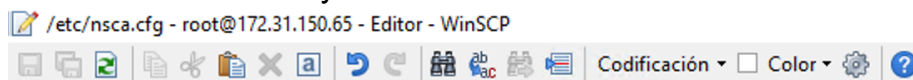
define command {
    command_name check_nrpe
    command_line $USER1$/check_nrpe -H $HOSTADDRESS$ -c $ARG1$ $ARG2$
}

```

Figura 260

NSCA [30]

Para instalar el cliente de NSCA en Linux se debe ejecutar el siguiente comando: “sudo apt-get install nsca”. Posteriormente, editamos el fichero “/etc/nsca.cfg” para añadir la IP del cliente, la contraseña que tienen tanto servidor como cliente y el método de autenticación:



```

# SERVER ADDRESS
# Address that NSCA has to bind to in case there are
# more as one interface and we do not want NSCA to bind
# (thus listen) on all interfaces.

server_address=172.31.150.65

```

Figura 261

```
password=PasswordSecure
```

```
decryption_method=3
```

Figura 262

Arrancamos el servicio y comprobamos su estado:

```

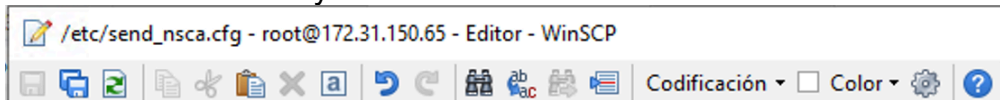
root@ubuntumanu:~# sudo systemctl start nsca
root@ubuntumanu:~# sudo systemctl status nsca
● nsca.service - Nagios Service Check Acceptor
   Loaded: loaded (/lib/systemd/system/nsca.service; disabled; vendor preset: enabled)
   Active: active (running) since Sun 2024-03-31 11:08:22 UTC; 2s ago
     Docs: http://www.nagios.org/documentation
   Main PID: 18662 (nsca)
    Tasks: 1 (limit: 4558)
   Memory: 268.0K
      CPU: 13ms
   CGroup: /system.slice/nsca.service
           └─18662 /usr/sbin/nsca -c /etc/nsca.cfg -f

mar 31 11:08:22 ubuntumanu systemd[1]: Started Nagios Service Check Acceptor.
mar 31 11:08:22 ubuntumanu nsca[18662]: Starting up daemon

```

Figura 263

Posteriormente, editamos el fichero “/etc/send_nsca.cfg” para establecer mismas credenciales y método de autenticación:



```
password=PasswordSecure
```

```
encryption_method=3
```

Figura 264

Una vez hecho esto, debemos de ir al servidor de monitorización y permitir el tráfico de nuestro cliente editando el fichero “/etc/xinetd.d/nsca”:


```

/etc/xinetd.d/nsca - root@172.31.3.74 - Editor - WinSCP
# default: on
# description: NSCA (Nagios Service Check Acceptor)
service nsca
{
    flags          = REUSE
    socket_type    = stream
    wait          = no
    user          = nagios
    group         = nagcmd
    server        = /usr/local/nagios/bin/nsca
    server_args   = -c /usr/local/nagios/etc/nsca.cfg --inetd
    log_on_failure += USERID
    disable       = no
    only_from     = 172.31.3.74 172.31.150.65
}

```

Figura 265

NCPA [31]

Para instalar el cliente de NCPA en Linux, necesitamos instalar el paquete "apt-transport-https", que permite a APT (Advanced Package Tool) utilizar repositorios HTTPS para descargar paquetes con el comando: "apt-get install apt-transport-https", posteriormente, debemos agregar una nueva fuente para descargar recursos con el comando: "echo "deb https://repo.nagios.com/deb/\$(lsb_release -cs) /" > /etc/apt/sources.list.d/nagios.list ", después, descargar la clave publica de Nagios para añadirla al listado de claves confiables de APT con el comando: "wget -qO - https://repo.nagios.com/GPG-KEY-NAGIOS-V3 | apt-key add -" y finalmente, actualizar repositorios e instalar NCPA con los comandos: "apt-get update" y "apt-get install ncpa".

Una vez instalado NCPA, debemos configurar el fichero ncpa.cfg ubicado en "usr/local/ncpa/etc" para indicarle el servidor de monitorización (listener) y el token a utilizar:

```

/usr/local/ncpa/etc/ncpa.cfg - root@172.31.150.65 - Editor - WinSCP
# IP address and port number for the Listener to use for the web GUI and API
#
# :: allows for dual stack (IPv4 and IPv6 on most linux systems) but will only allow
# for IPv6 connections on Windows
# 0.0.0.0 allows for IPv4 connections only on Windows and most linux systems
#
# Default: ip = ::
# Default (Windows): ip = 0.0.0.0
# Default: port = 5693
#
ip = 172.31.3.74
port = 5693

community_string = PasswordSecure

```

Figura 266

Posteriormente, reiniciamos el servicio:

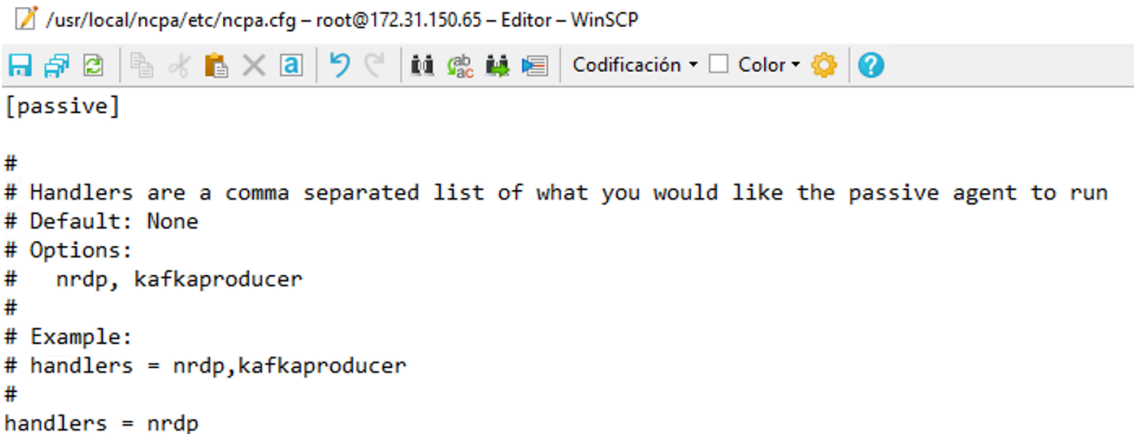
```

root@ubuntumanu:~# systemctl restart ncpa
root@ubuntumanu:~# systemctl stop ncpa
root@ubuntumanu:~# systemctl start ncpa
root@ubuntumanu:~# █

```

Figura 267

Una vez instalado i configurado NCPA, si se quiere habilitar NRDP [39], primero de todo, editamos el fichero “/usr/local/ncpa/etc/ncpa.cfg” para añadir el agente “nrpd” en la linea handlers:



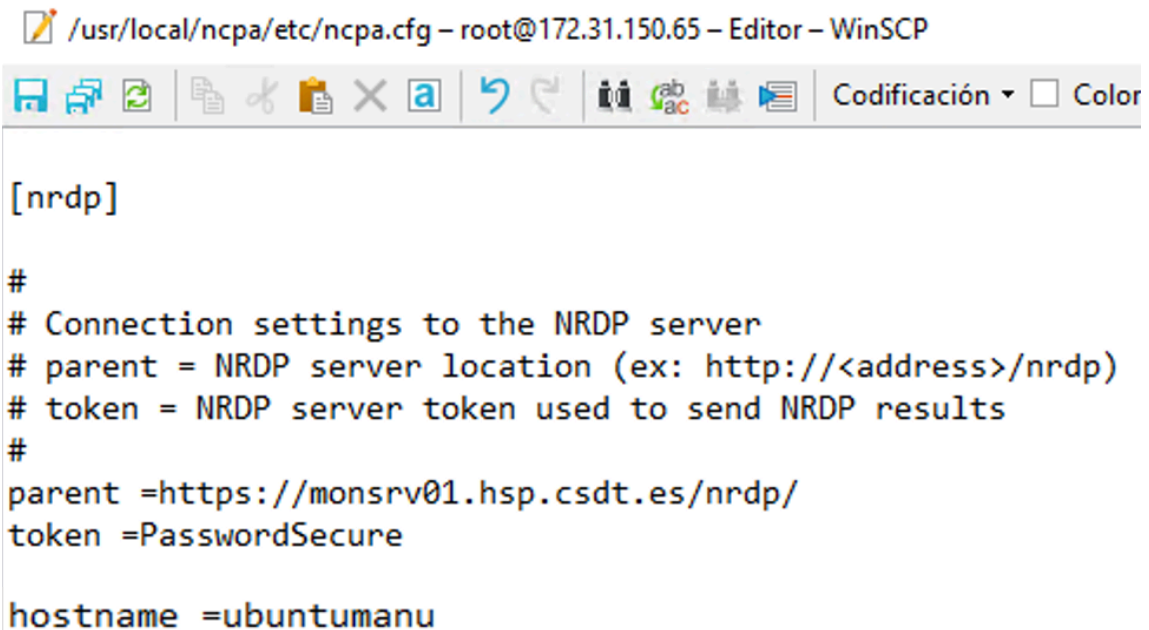
```

/usr/local/ncpa/etc/ncpa.cfg – root@172.31.150.65 – Editor – WinSCP
[passive]
#
# Handlers are a comma separated list of what you would like the passive agent to run
# Default: None
# Options:
# nrpd, kafkaproducer
#
# Example:
# handlers = nrpd,kafkaproducer
#
handlers = nrpd

```

Figura 268

También añadimos el parent, el token utilizado y el hostname:



```

/usr/local/ncpa/etc/ncpa.cfg – root@172.31.150.65 – Editor – WinSCP
[nrdp]
#
# Connection settings to the NRDP server
# parent = NRDP server location (ex: http://<address>/nrdp)
# token = NRDP server token used to send NRDP results
#
parent =https://monsrv01.hsp.csdt.es/nrdp/
token =PasswordSecure

hostname =ubuntumanu

```

Figura 269

Y finalmente, creamos el archivo nrpd.cfg en la ruta que se indica:

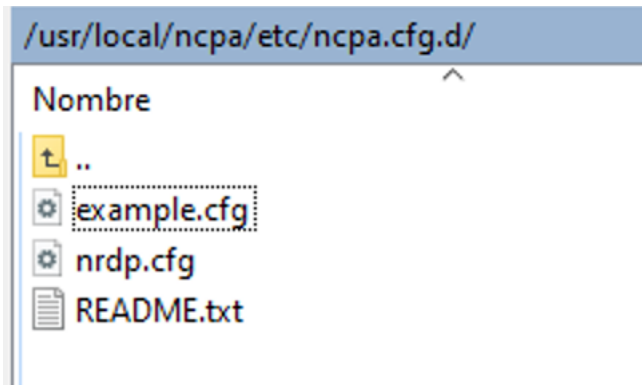


Figura 270