

Nagios NRPE Monitoring Configuration – Overview

This work documents the configuration and validation of Nagios Remote Plugin Executor (NRPE) monitoring for a Linux staging web server environment. The objective of this effort was to securely integrate a new host into the existing Nagios monitoring infrastructure, ensuring reliable service checks, controlled access, and continuous availability monitoring.

The process begins with configuring the NRPE daemon to run under a dedicated service account and restricting access to explicitly authorized monitoring hosts. Network-level controls and daemon settings are validated to ensure NRPE listens on the correct port and only accepts approved connections. Service management is handled through `systemctl`, confirming that NRPE is enabled at boot and actively running.

On the Nagios server side, a structured host definition is created using standardized templates to maintain consistency across environments. Monitoring parameters such as check intervals, retry limits, and notification periods are defined to support 24x7 monitoring and timely alerting. Host object files are organized and included modularly within the Nagios configuration to support scalability and maintainability across development and staging systems.

The configuration concludes with verification steps to confirm successful service startup, host registration, and communication between the Nagios server and the monitored host. Together, these steps demonstrate a complete, production-style monitoring setup aligned with best practices for security, clarity, and operational reliability.

The terminal output shows the installation of NRPE (Nagios Remote Plugin Executor) on an Enterprise Linux 9 (x86_64) system using yum.

The command executed is `sudo yum install nrpe -y`.

A warning appears indicating that `/etc/yum.repos.d/MariaDB.repo` could not be loaded, but the operation continues and is not blocked.

Package metadata is retrieved from the EPEL 9 repositories, including EPEL, EPEL Next, and EPEL OpenH264.

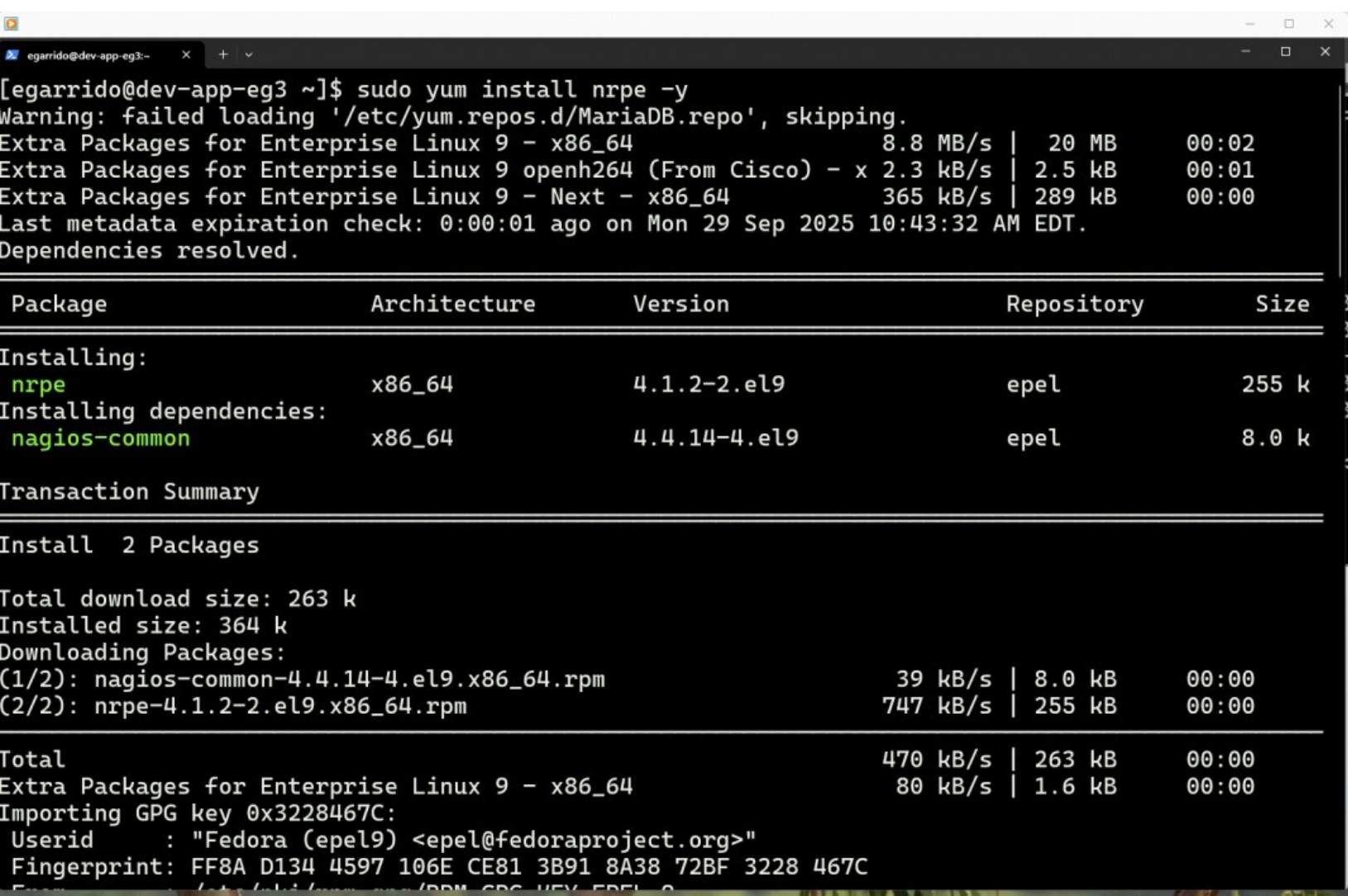
Dependencies are successfully resolved, and two packages are selected for installation:

- `nrpe` version `4.1.2-2.el9`
- `nagios-common` version `4.4.14-4.el9`

Both packages are downloaded and installed successfully with a total download size of 263 KB.

The system imports and trusts the EPEL GPG key, confirming package authenticity.

Overall, the screenshot confirms a successful NRPE installation from the EPEL repository, with only a non-fatal repository warning unrelated to NRPE.



```
[egarrido@dev-app-eg3 ~]$ sudo yum install nrpe -y
Warning: failed loading '/etc/yum.repos.d/MariaDB.repo', skipping.
Extra Packages for Enterprise Linux 9 - x86_64                8.8 MB/s | 20 MB      00:02
Extra Packages for Enterprise Linux 9 openh264 (From Cisco) - x 2.3 kB/s | 2.5 kB      00:01
Extra Packages for Enterprise Linux 9 - Next - x86_64        365 kB/s | 289 kB     00:00
Last metadata expiration check: 0:00:01 ago on Mon 29 Sep 2025 10:43:32 AM EDT.
Dependencies resolved.

Package Architecture Version Repository Size
Installing:
nrpe x86_64 4.1.2-2.el9 epel 255 k
Installing dependencies:
nagios-common x86_64 4.4.14-4.el9 epel 8.0 k

Transaction Summary
Install 2 Packages

Total download size: 263 k
Installed size: 364 k
Downloading Packages:
(1/2): nagios-common-4.4.14-4.el9.x86_64.rpm 39 kB/s | 8.0 kB 00:00
(2/2): nrpe-4.1.2-2.el9.x86_64.rpm 747 kB/s | 255 kB 00:00
Total 470 kB/s | 263 kB 00:00
Extra Packages for Enterprise Linux 9 - x86_64 80 kB/s | 1.6 kB 00:00
Importing GPG key 0x3228467C:
Userid : "Fedora (epel9) <epel@fedoraproject.org>"
Fingerprint: FF8A D134 4597 106E CE81 3B91 8A38 72BF 3228 467C
From: https://mirrors.fedoraproject.org/mirrors/epel/9/primary/repodata/repomd.xml
```

This screenshot displays a portion of the NRPE configuration file located at `/etc/nagios/nrpe.cfg` opened in a text editor on an Enterprise Linux 9 system.

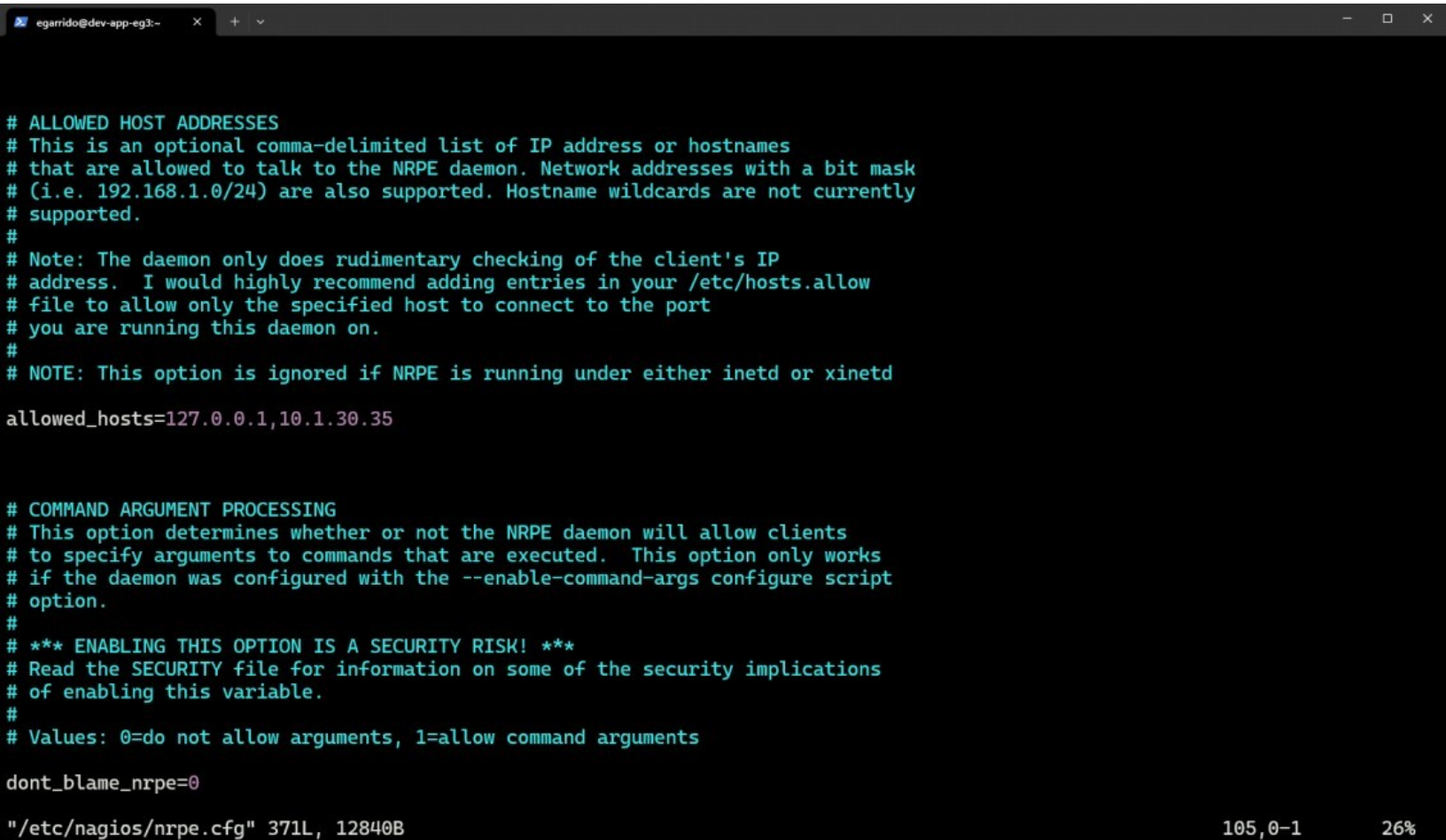
The `allowed_hosts` directive is configured to restrict which systems are permitted to communicate with the NRPE daemon.

Only the local host (127.0.0.1) and a specific monitoring server IP (10.1.30.35) are allowed to connect, enforcing basic network-level access control.

Inline comments explain that NRPE performs only minimal client IP validation and recommends additional protection via `/etc/hosts.allow`.

The configuration notes that this setting is ignored if NRPE is run under `inetd` or `xinetd`.

The command argument processing section shows `dont_blame_nrpe=0`, meaning NRPE is configured not to accept command arguments from remote clients, which helps reduce security risk.

A screenshot of a terminal window with a dark background. The window title bar shows 'egarrido@dev-app-eg3:~' and standard window controls. The terminal displays the contents of the NRPE configuration file, showing comments and configuration directives. The configuration includes a list of allowed hosts (127.0.0.1 and 10.1.30.35) and a security setting (dont_blame_nrpe=0). The bottom status bar of the terminal shows the file path, line count, and percentage of the file displayed.

```
# 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=127.0.0.1,10.1.30.35

# COMMAND ARGUMENT PROCESSING
# This option determines whether or not the NRPE daemon will allow clients
# to specify arguments to commands that are executed. This option only works
# if the daemon was configured with the --enable-command-args configure script
# option.
#
# *** ENABLING THIS OPTION IS A SECURITY RISK! ***
# Read the SECURITY file for information on some of the security implications
# of enabling this variable.
#
# Values: 0=do not allow arguments, 1=allow command arguments

dont_blame_nrpe=0

"/etc/nagios/nrpe.cfg" 371L, 12840B 105,0-1 26%
```

The NRPE service has been successfully started, enabled, and verified on an Enterprise Linux 9 system using systemd. The service is configured to launch automatically at boot and is currently running without errors.

Systemd confirms that nrpe.service is loaded from /usr/lib/systemd/system/nrpe.service and is in an active (running) state. The NRPE daemon is operating under its own process and is using the configuration file located at /etc/nagios/nrpe.cfg.

Runtime logs show that NRPE is listening on the default monitoring port 5666 for both IPv4 and IPv6 connections. Incoming connections are restricted to the local host and the authorized Nagios monitoring server, as defined in the allowed_hosts configuration.

This verification confirms that NRPE is properly configured, securely restricted, and ready to accept remote health checks from approved monitoring system

```
egarrido@dev-app-eg3:~$ sudo systemctl start nrpe.service
egarrido@dev-app-eg3:~$ sudo systemctl enable nrpe.service
egarrido@dev-app-eg3:~$ sudo systemctl status nrpe.service
● nrpe.service - Nagios Remote Plugin Executor
   Loaded: loaded (/usr/lib/systemd/system/nrpe.service; enabled; preset: disabled)
   Active: active (running) since Tue 2025-09-30 11:06:03 EDT; 1h 24min ago
     Docs: http://www.nagios.org/documentation
   Main PID: 1144 (nrpe)
    Tasks: 1 (limit: 4605)
   Memory: 168.0K
      CPU: 39ms
   CGroup: /system.slice/nrpe.service
           └─1144 /usr/sbin/nrpe -c /etc/nagios/nrpe.cfg -f

Sep 30 11:06:03 dev-app-eg3.procore.prod1 systemd[1]: Started Nagios Remote Plugin Executor.
Sep 30 11:06:04 dev-app-eg3.procore.prod1 nrpe[1144]: Starting up daemon
Sep 30 11:06:04 dev-app-eg3.procore.prod1 nrpe[1144]: Server listening on 0.0.0.0 port 5666.
Sep 30 11:06:04 dev-app-eg3.procore.prod1 nrpe[1144]: Server listening on :: port 5666.
Sep 30 11:06:04 dev-app-eg3.procore.prod1 nrpe[1144]: Listening for connections on port 5666
Sep 30 11:06:04 dev-app-eg3.procore.prod1 nrpe[1144]: Allowing connections from: 127.0.0.1,10.1.30.35
egarrido@dev-app-eg3:~$
```


Access to the Nagios configuration directory on the stage monitoring server is shown, where host and service definitions are centrally managed. The administrator navigates to `/usr/local/nagios/etc/objects/` and edits a host-specific configuration file for the `dev-app-eg3` system within the production environment.

A directory listing confirms the presence of multiple Nagios object configuration files, including host definitions, service checks, contact definitions, and command configurations. Files follow a consistent naming convention that reflects environment, application role, and domain structure, supporting clarity and maintainability at scale.

The permissions and ownership displayed indicate that Nagios configuration files are properly secured while remaining editable by authorized users. This step demonstrates how new or updated application hosts are integrated into the Nagios monitoring framework by creating or modifying dedicated object configuration files.

Overall, this validates the host onboarding process within Nagios, ensuring monitored systems are accurately defined and aligned with existing monitoring standards and conventions.

```
egarrido@stage-nagios:~$ clear
[egarrido@stage-nagios ~]$ sudo vi /usr/local/nagios/etc/objects/dev-app-eg3.procore.prod1.cfg
[sudo] password for egarrido:
[egarrido@stage-nagios ~]$ sudo vi /usr/local/nagios/etc/objects/dev-app-eg3.procore.prod1.cfg
[egarrido@stage-nagios ~]$ ll /usr/local/nagios/etc/objects/
total 1032
-rw-r--r--. 1 root root 639 Jul 19 2024 !!
-rw-r--r--. 1 root root 405 Jul 11 19:53 10.1.31.98.cfg
-rw-r--r--. 1 1000 nagios 478 Oct 23 2024 app-thony.cfg
-rw-r--r--. 1 1000 nagios 834 May 20 2023 boyeyipo.cfg
-rw-r--r--. 1 1000 nagios 5643 Aug 20 2024 canuforoh.cfg
-rw-r--r--. 1 1000 nagios 7542 Jul 24 2024 commands.cfg
-rwxr--r--. 1 1000 nagios 6765 Apr 8 2022 commands.cfg~
-rw-r--r--. 1 1000 nagios 2175 Sep 20 10:30 contacts.cfg
-rwxr--r--. 1 1000 nagios 1797 Apr 8 2022 contacts.cfg~
-rw-r--r--. 1 root root 408 Aug 30 17:28 dev-app-ac3.procore.prod1.cfg
-rw-r--r--. 1 1000 nagios 406 Apr 19 11:31 dev-app-ad2.procore.prod1.cfg
-rw-r--r--. 1 1000 nagios 1240 Oct 9 2024 dev-app-ad.procore.prod1.cfg
-rwxr--r--. 1 1000 nagios 1248 Feb 4 2025 dev-app-ae.procore
-rw-r--r--. 1 1000 nagios 1240 Feb 5 2025 dev-app-ae.procore.prod1.cfg
-rwxr--r--. 1 root root 1242 Jun 22 14:44 dev-app-ah5.cfg
-rw-r--r--. 1 root root 407 Jun 22 14:16 dev-app-ah5.procore.prod1.cfg
-rw-r--r--. 1 1000 nagios 319 May 3 21:55 dev-app-aj1.procore.prod1.cfg
-rwxr--r--. 1 root root 1242 Jun 1 16:00 dev-app-ak1.procore.prod1.cfg
-rw-r--r--. 1 root root 1149 Jun 12 14:05 dev-app-al.procore.prod1.cfg
-rw-r--r--. 1 root root 406 Jul 30 15:54 dev-app-aml.procore.prod1.cfg
-rwxr--r--. 1 root root 1240 Jul 11 11:07 dev-app-am.procore.prod1.cfg
-rw-r--r--. 1 1000 nagios 1242 Jul 16 2024 dev-app-ao1.procore.prod1.cfg
-rw-r--r--. 1 1000 nagios 2014 Oct 4 2024 dev-app-ar.procore.prod1-202.cfg
-rw-r--r--. 1 1000 nagios 404 Jan 2 2025 dev-app-as.procore.prod1.cfg
-rw-r--r--. 1 1000 nagios 1242 May 30 21:09 dev-app-at1.procore.prod1.cfg
-rwxr--r--. 1 root root 1236 Jul 9 20:45 dev-app-bl1.cfg
-rw-r--r--. 1 1000 nagios 405 Nov 16 2024 dev-app-bl.procore.prod1
-rwxr--r--. 1 1000 nagios 1238 Jul 6 2024 dev-app-bp.procore.prod1
-rw-r--r--. 1 1000 nagios 734 Oct 8 2024 dev-app-bt.procore.prod1.cfg
-rw-r--r--. 1 1000 nagios 407 Apr 19 11:30 dev-app-cab.procore.prod1.cfg
-rw-r--r--. 1 1000 nagios 1239 Aug 20 2024 dev-app-ca.procore.prod1.cfg
-rw-r--r--. 1 root root 285 Jul 31 19:12 dev-app-cb2.procore.prod1.cfg
-rw-r--r--. 1 root root 1239 Aug 16 13:34 dev-app-cd.procore.prod1.cfg
-rw-r--r--. 1 1000 nagios 1239 Jun 28 2024 dev-app-cf-procore.prod1.cfg
-rw-r--r--. 1 1000 nagios 824 Jan 28 2025 dev-app-ch.procore.prod1.cfg
```

A new Nagios host object definition is being created to onboard an application server into centralized monitoring. The configuration file defines a Linux-based host using the standard linux-server template, allowing the host to inherit common monitoring settings and thresholds.

The host is identified by the name dev-app-eg3.procore.prod1, with an alias for readability and an explicitly defined IP address. Monitoring parameters such as maximum check attempts, continuous check coverage (24x7), and notification intervals are configured to ensure consistent availability tracking and alerting behavior.

By using a dedicated host object file and inheriting from an existing template, this approach maintains consistency across environments while simplifying ongoing monitoring management. This step completes the host definition phase required before attaching service checks and NRPE-based monitoring to the system

```
egarrido@stage-nagios:~  
#####  
# DEV-APP-KR.CFG - SAMPLE OBJECT CONFIG FILE FOR MONITORING THIS MACHINE  
#  
#  
# NOTE: This config file is intended to serve as an *extremely* simple  
#       example of how you can create configuration entries to monitor  
#       the local (Linux) machine.  
#  
#####  
  
#####  
#  
# HOST DEFINITION  
#  
#####  
  
# Define a host for the local machine  
  
define host {  
  
    use                linux-server                ; Name of host template to use  
                                                ; This host definition will inherit all variables that are defined  
                                                ; in (or inherited by) the linux-server host template definition.  
  
    host_name          {dev-app-eg3.procore.prod1}  
    alias               {dev-app-eg3}  
    address             {10.1.31.124}  
    max_check_attempts  5  
    check_period        24x7  
    notification_interval 30  
    notification_period 24x7  
  
}  
  
~  
~  
~  
~  
~  
~  
-- INSERT --
```


The Nagios monitoring service is validated and brought into a running state using `systemd` on the stage monitoring server. An initial attempt to reload the service indicates that the Nagios unit does not support a reload operation, requiring a full service enablement and start sequence instead.

Nagios is successfully enabled to start at boot and confirmed to be active (running). The service is loaded from `/etc/systemd/system/nagios.service` and is running version Nagios Core 4.4.14 under the main Nagios process, with multiple worker processes handling active checks concurrently.

The process tree shows active plugin executions, including ICMP reachability checks using `check_ping`, confirming that Nagios is actively monitoring configured hosts. Log output at the bottom reflects real-time worker activity, including completed and timed-out checks, which demonstrates that the monitoring engine is actively processing host and service checks.

This step confirms that Nagios is operational, enabled, and actively executing monitoring jobs, validating the successful integration of newly added host definitions into the monitoring system.

```
egarrido@stage-nagios ~]$ sudo systemctl status nagios | less
egarrido@stage-nagios ~]$ sudo systemctl reload nagios.service
Failed to reload nagios.service: Job type reload is not applicable for unit nagios.service.
See system logs and 'systemctl status nagios.service' for details.
egarrido@stage-nagios ~]$ sudo systemctl enable nagios
Unknown operation 'enable'.
egarrido@stage-nagios ~]$ sudo systemctl enable nagios
egarrido@stage-nagios ~]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.4.14
   Loaded: loaded (/etc/systemd/system/nagios.service; enabled; vendor preset: disabled)
   Active: active (running) since Mon 2025-09-29 20:00:20 EDT; 1min 55s ago
     Main PID: 7417 (nagios)
    CGroup: /system.slice/nagios.service
            └─7417 /usr/local/nagios/bin/nagios /usr/local/nagios/etc/nagios.cfg
              └─7419 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                └─7420 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                  └─7421 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                    └─7422 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                      └─7423 /usr/local/nagios/bin/nagios /usr/local/nagios/etc/nagios.cfg
                        └─7732 /usr/local/nagios/libexec/check_ping -H 10.1.31.151 -w 3000.0,80% -c 5000.0,100% -p 5
                          └─7733 /usr/bin/ping -n -U -W 30 -c 5 10.1.31.151
                            └─7796 /usr/local/nagios/libexec/check_ping -H 10.1.31.199 -w 3000.0,80% -c 5000.0,100% -p 5
                              └─7797 /usr/bin/ping -n -U -W 30 -c 5 10.1.31.199
                                └─7800 /usr/local/nagios/libexec/check_ping -H 10.1.31.110 -w 3000.0,80% -c 5000.0,100% -p 5
                                  └─7801 /usr/bin/ping -n -U -W 30 -c 5 10.1.31.110
                                    └─7804 /usr/local/nagios/libexec/check_ping -H 10.1.31.160 -w 3000.0,80% -c 5000.0,100% -p 5
                                      └─7805 /usr/bin/ping -n -U -W 30 -c 5 10.1.31.160
                                        └─7807 /usr/local/nagios/libexec/check_ping -H 10.1.31.250 -w 3000.0,80% -c 5000.0,100% -p 5
                                          └─7808 /usr/bin/ping -n -U -W 30 -c 5 10.1.31.250

Sep 29 20:02:13 stage-nagios.procore.prod1 nagios[7417]: wproc: CHECK job 28 from worker Core Worker 7419 timed out after 30.01s
Sep 29 20:02:13 stage-nagios.procore.prod1 nagios[7417]: wproc: host=dev-app-km-procore.prod1; service=(null);
Sep 29 20:02:13 stage-nagios.procore.prod1 nagios[7417]: wproc: early_timeout=1; exited_ok=0; wait_status=0; error_code=62;
Sep 29 20:02:13 stage-nagios.procore.prod1 nagios[7417]: Warning: Check of host 'dev-app-km-procore.prod1' timed out after 3...conds
Sep 29 20:02:13 stage-nagios.procore.prod1 nagios[7417]: wproc: CHECK job 28 from worker Core Worker 7419 timed out after 30.01s
Sep 29 20:02:13 stage-nagios.procore.prod1 nagios[7417]: wproc: host=dev-app-km-procore.prod1; service=(null);
Sep 29 20:02:13 stage-nagios.procore.prod1 nagios[7417]: wproc: early_timeout=1; exited_ok=0; wait_status=0; error_code=62;
Sep 29 20:02:13 stage-nagios.procore.prod1 nagios[7417]: Warning: Check of host 'dev-app-km-procore.prod1' timed out after 3...conds
Sep 29 20:02:14 stage-nagios.procore.prod1 nagios[7417]: wproc: Core Worker 7419: job 28 (pid=7725): Dormant child reaped
Sep 29 20:02:14 stage-nagios.procore.prod1 nagios[7417]: wproc: Core Worker 7419: job 28 (pid=7725): Dormant child reaped
hint: Some lines were ellipsized, use -l to show in full.
egarrido@stage-nagios ~]$
```

Nagios is configured to explicitly load host and service object definitions by referencing individual configuration files using the `cfg_file` directive. This approach provides precise control over which monitoring objects are included in the active configuration.

The configuration is logically organized by administrator or responsibility group, with comments separating related host definitions. New application and web hosts are added by appending their corresponding object files to the main Nagios configuration, ensuring they are recognized during startup or service restarts.

Each referenced file points to a dedicated object definition under `/usr/local/nagios/etc/objects/`, allowing hosts to be managed independently while maintaining a consistent naming convention across environments.

This step finalizes the Nagios onboarding process, ensuring newly defined hosts are included in the monitoring engine and actively evaluated once the service is running.

```
egarrido@stage-nagios:~  
cfg_file=/usr/local/nagios/etc/objects/dev-app-lw.procore.prod1.cfg  
cfg_file=/usr/local/nagios/etc/objects/stage-web-lw.procore.prod1.cfg  
#  
  
#egiron  
  
cfg_file=/usr/local/nagios/etc/objects/dev-app-eg2.procore.prod1.cfg  
cfg_file=/usr/local/nagios/etc/objects/stage-web-eg2.procore.prod1.cfg  
#  
  
#egarrido  
  
cfg_file=/usr/local/nagios/etc/objects/dev-app-eg3.procore.prod1.cfg  
cfg_file=/usr/local/nagios/etc/objects/stage-web-eg3.procore.prod1.cfg  
#  
  
  
# You can also tell Nagios to process all config files (with a .cfg  
# extension) in a particular directory by using the cfg_dir  
# directive as shown below:  
  
#cfg_dir=/usr/local/nagios/etc/servers  
#cfg_dir=/usr/local/nagios/etc/printers  
#cfg_dir=/usr/local/nagios/etc/switches  
#cfg_dir=/usr/local/nagios/etc/routers
```

```
# OBJECT CACHE FILE  
# This option determines where object definitions are cached when
```


After updating the Nagios configuration, the monitoring service is restarted to apply the new object definitions. An attempted reload confirms that the Nagios systemd unit does not support reload operations, requiring a full service restart instead.

Nagios is successfully restarted and enabled to start automatically at boot. The service status output confirms Nagios Core 4.4.14 is active and running, using the primary configuration file located at /usr/local/nagios/etc/nagios.cfg.

The process tree shows multiple active worker processes and live plugin executions, including ICMP reachability checks via check_ping, indicating that host checks are being executed in real time across multiple monitored systems.

This validation confirms that the updated configuration has been successfully applied and that Nagios is fully operational, actively monitoring hosts, and processing checks as expected.

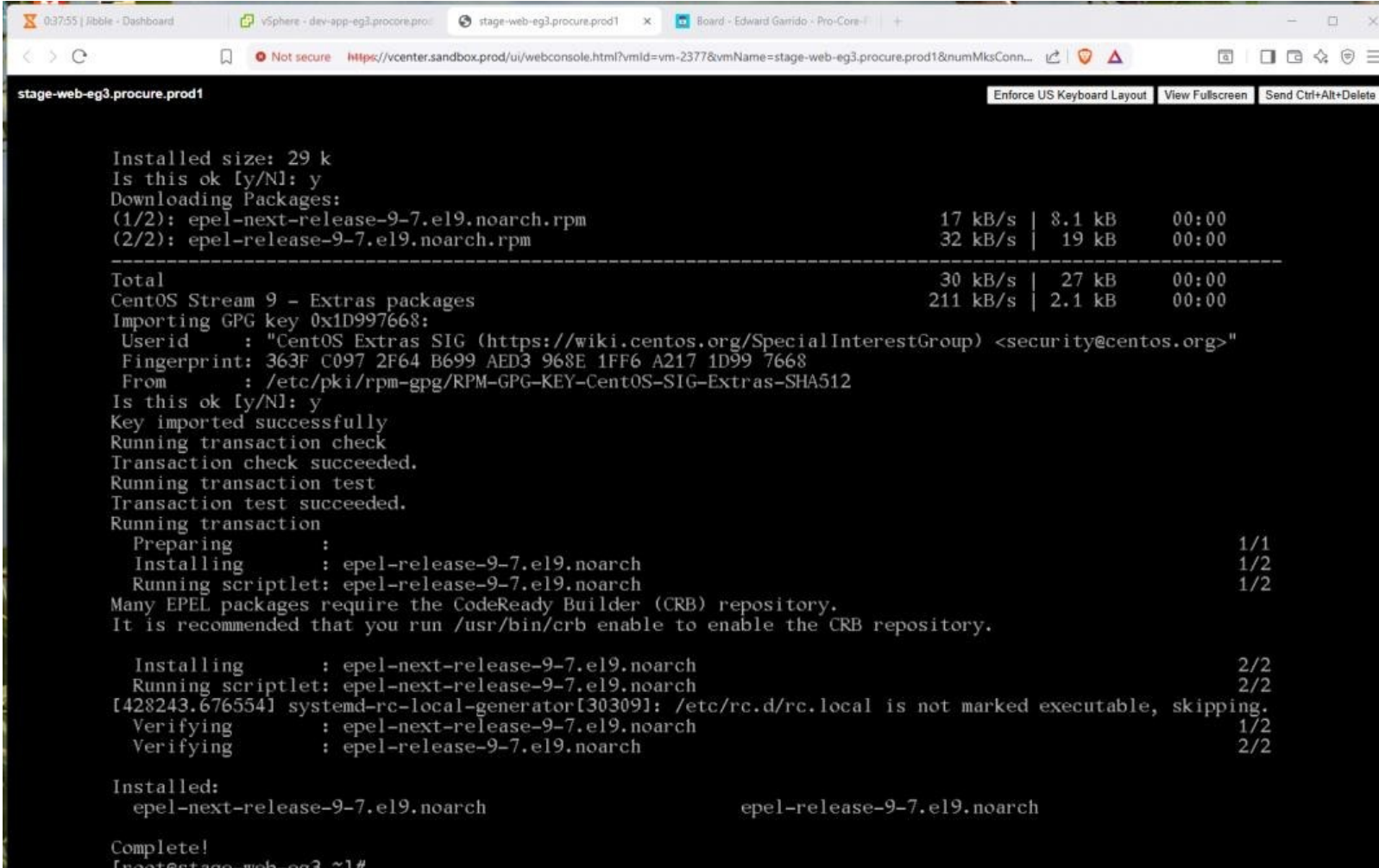
```
egarrido@stage-nagios:/usr/1 x + v
drwxr-xr-x. 2 1000 nagios 4096 Sep 24 10:05 servers
-rw-r--r--. 1 root root 198 Aug 30 01:43 stage-web-sm.procore.prod1
[egarrido@stage-nagios etc]$ sudo vim nagios.cfg
[sudo] password for egarrido:
[egarrido@stage-nagios etc]$ sudo systemctl reload nagios.service
Failed to reload nagios.service: Job type reload is not applicable for unit nagios.service.
See system logs and 'systemctl status nagios.service' for details.
[egarrido@stage-nagios etc]$ sudo systemctl restart nagios
[egarrido@stage-nagios etc]$ sudo systemctl enable nagios
[egarrido@stage-nagios etc]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.4.14
   Loaded: loaded (/etc/systemd/system/nagios.service; enabled; vendor preset: disabled)
   Active: active (running) since Tue 2025-09-30 12:48:04 EDT; 14s ago
 Main PID: 5313 (nagios)
  CGroup: /system.slice/nagios.service
          └─5313 /usr/local/nagios/bin/nagios /usr/local/nagios/etc/nagios.cfg
             └─5315 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                └─5316 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                   └─5317 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                      └─5318 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                         └─5319 /usr/local/nagios/bin/nagios /usr/local/nagios/etc/nagios.cfg
                            └─5320 /usr/local/nagios/libexec/check_ping -H 10.1.31.67 -w 3000.0,80% -c 5000.0,100% -p 5
                               └─5321 /usr/bin/ping -n -U -W 30 -c 5 10.1.31.67
                                  └─5322 /usr/local/nagios/libexec/check_ping -H 10.1.31.213 -w 3000.0,80% -c 5000.0,100% -p 5
                                     └─5324 /usr/bin/ping -n -U -W 30 -c 5 10.1.31.213
                                        └─5326 /usr/local/nagios/libexec/check_ping -H 10.1.31.35 -w 3000.0,80% -c 5000.0,100% -p 5
                                           └─5328 /usr/bin/ping -n -U -W 30 -c 5 10.1.31.35
                                              └─5330 /usr/local/nagios/libexec/check_ping -H 10.1.31.159 -w 3000.0,80% -c 5000.0,100% -p 5
                                                 └─5331 /usr/local/nagios/libexec/check_ping -H 10.1.31.249 -w 3000.0,80% -c 5000.0,100% -p 5
                                                    └─5332 /usr/bin/ping -n -U -W 30 -c 5 10.1.31.159
                                                       └─5333 /usr/bin/ping -n -U -W 30 -c 5 10.1.31.249
                                                          └─5360 /usr/local/nagios/libexec/check_ping -H 10.1.31.231 -w 3000.0,80% -c 5000.0,100% -p 5
                                                             └─5361 /usr/bin/ping -n -U -W 30 -c 5 10.1.31.231
                                                                └─5362 /usr/local/nagios/libexec/check_ping -H 10.1.31.38 -w 3000.0,80% -c 5000.0,100% -p 5
                                                                   └─5363 /usr/bin/ping -n -U -W 30 -c 5 10.1.31.38
```

The EPEL (Extra Packages for Enterprise Linux) repositories are installed on a CentOS Stream 9 system to enable access to additional community-supported packages required for monitoring and tooling.

The system downloads and installs both epel-release and epel-next-release packages, importing the CentOS Extras SIG GPG key to ensure package authenticity. Transaction checks and tests complete successfully, confirming repository integrity before installation proceeds.

During installation, the system notes that some EPEL packages may depend on the CRB (CodeReady Builder) repository and recommends enabling it if required by future packages. The installation completes without errors, and the repositories are verified as installed.

This step prepares the system for installing NRPE, Nagios plugins, and other EPEL-hosted dependencies, ensuring compatibility with enterprise monitoring workflows.



```
stage-web-eg3.procure.prod1
Installed size: 29 k
Is this ok [y/N]: y
Downloading Packages:
(1/2): epel-next-release-9-7.el9.noarch.rpm      17 kB/s | 8.1 kB      00:00
(2/2): epel-release-9-7.el9.noarch.rpm          32 kB/s | 19 kB      00:00
-----
Total                                           30 kB/s | 27 kB      00:00
CentOS Stream 9 - Extras packages              211 kB/s | 2.1 kB     00:00
Importing GPG key 0x1D997668:
  Userid   : "CentOS Extras SIG (https://wiki.centos.org/SpecialInterestGroup) <security@centos.org>"
  Fingerprint: 363F C097 2F64 B699 AED3 968E 1FF6 A217 1D99 7668
  From      : /etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-SIG-Extras-SHA512
Is this ok [y/N]: y
Key imported successfully
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing      :                                1/1
  Installing     : epel-release-9-7.el9.noarch   1/2
  Running scriptlet: epel-release-9-7.el9.noarch 1/2
Many EPEL packages require the CodeReady Builder (CRB) repository.
It is recommended that you run /usr/bin/crb enable to enable the CRB repository.

  Installing     : epel-next-release-9-7.el9.noarch 2/2
  Running scriptlet: epel-next-release-9-7.el9.noarch 2/2
[428243.676554] systemd-rc-local-generator[303091]: /etc/rc.d/rc.local is not marked executable, skipping.
  Verifying      : epel-next-release-9-7.el9.noarch 1/2
  Verifying      : epel-release-9-7.el9.noarch     2/2

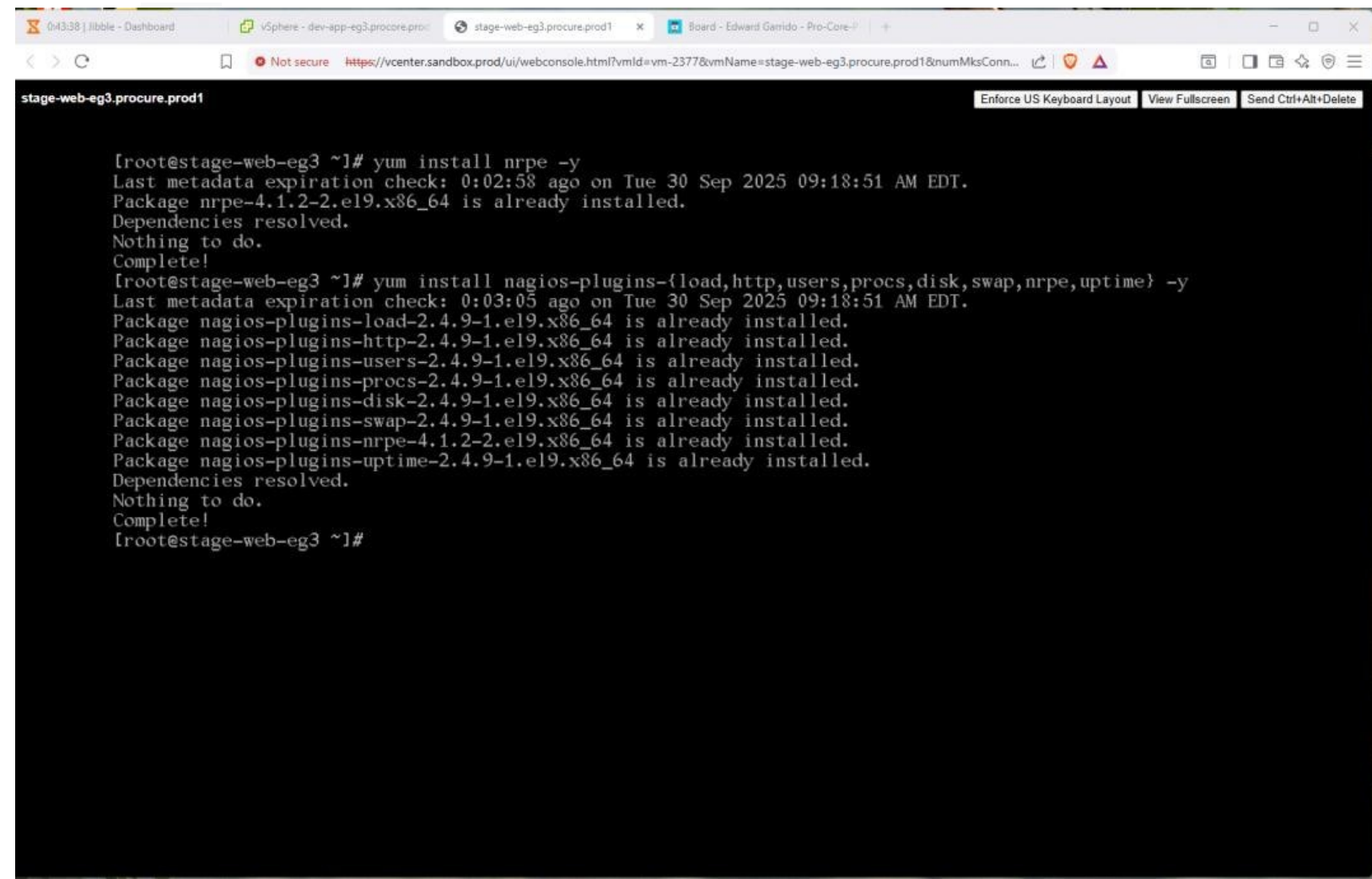
Installed:
  epel-next-release-9-7.el9.noarch                epel-release-9-7.el9.noarch

Complete!
[root@stage-web-eg3 ~]#
```

NRPE and the required Nagios monitoring plugins are verified on the application host to ensure readiness for remote monitoring. The system confirms that the NRPE package is already installed and up to date, requiring no further action.

A targeted set of Nagios plugins is then checked for availability, including load, HTTP, user sessions, process counts, disk usage, swap usage, NRPE checks, and system uptime. Each plugin is reported as already installed, confirming that the host has all necessary monitoring components in place.

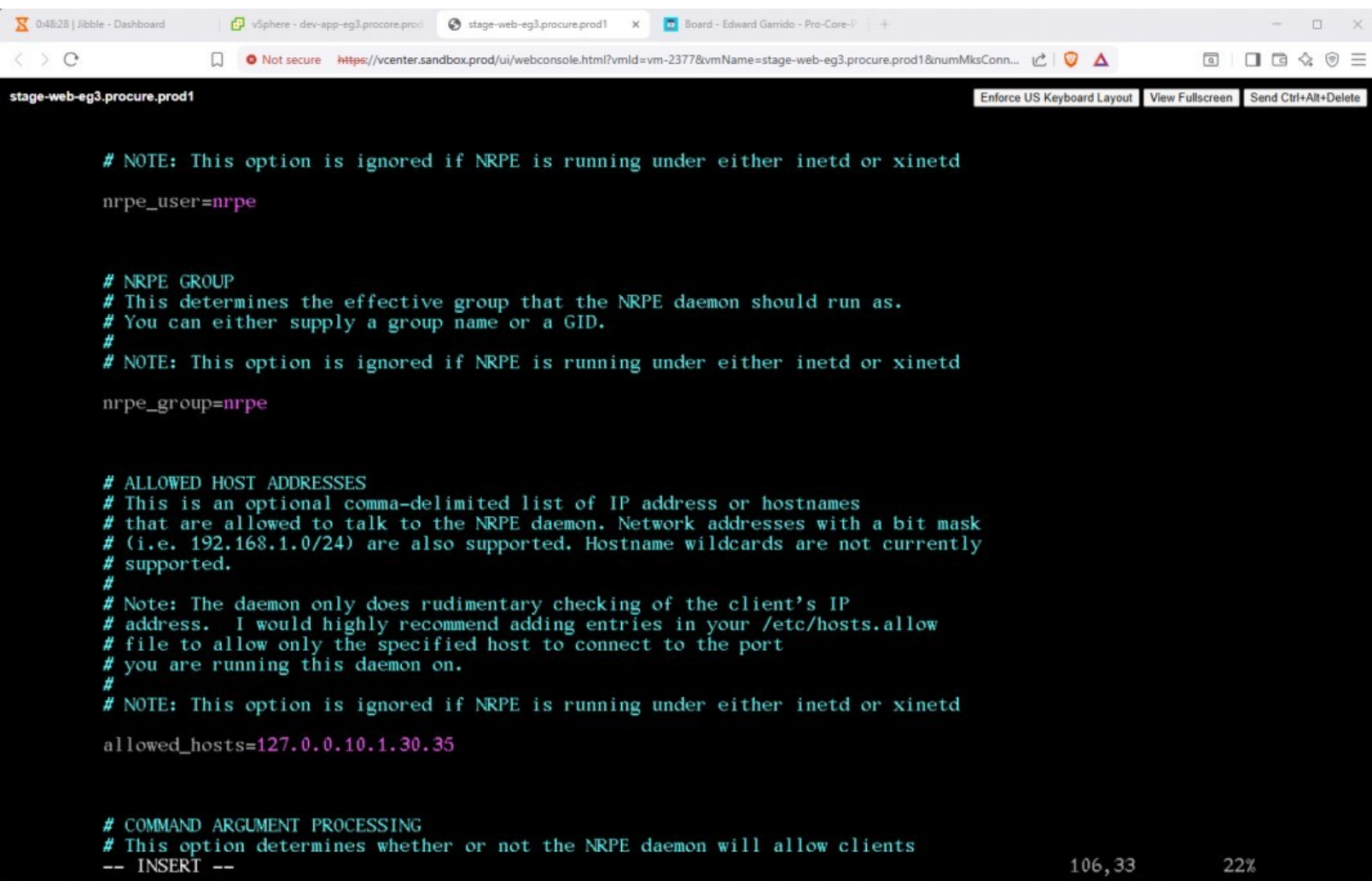
This verification step ensures the server is fully prepared to respond to NRPE-based health checks from the Nagios monitoring server, completing the agent-side monitoring setup without introducing redundant package changes.

A screenshot of a vSphere web console interface. The browser address bar shows a URL starting with 'https://vcenter.sandbox.prod/ui/webconsole.html?vmId=vm-2377&vmName=stage-web-eg3.procore.prod1&numMksConn...'. The console title is 'stage-web-eg3.procore.prod1'. The terminal window shows the following commands and output:

```
[root@stage-web-eg3 ~]# yum install nrpe -y
Last metadata expiration check: 0:02:58 ago on Tue 30 Sep 2025 09:18:51 AM EDT.
Package nrpe-4.1.2-2.el9.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!

[root@stage-web-eg3 ~]# yum install nagios-plugins-{load,http,users,procs,disk,swap,nrpe,uptime} -y
Last metadata expiration check: 0:03:05 ago on Tue 30 Sep 2025 09:18:51 AM EDT.
Package nagios-plugins-load-2.4.9-1.el9.x86_64 is already installed.
Package nagios-plugins-http-2.4.9-1.el9.x86_64 is already installed.
Package nagios-plugins-users-2.4.9-1.el9.x86_64 is already installed.
Package nagios-plugins-procs-2.4.9-1.el9.x86_64 is already installed.
Package nagios-plugins-disk-2.4.9-1.el9.x86_64 is already installed.
Package nagios-plugins-swap-2.4.9-1.el9.x86_64 is already installed.
Package nagios-plugins-nrpe-4.1.2-2.el9.x86_64 is already installed.
Package nagios-plugins-uptime-2.4.9-1.el9.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[root@stage-web-eg3 ~]#
```


This screenshot shows the NRPE configuration file being edited on the host stage-web-eg3.procore.prod1. The configuration defines the NRPE runtime user and group (nrpe) and specifies the allowed host addresses permitted to communicate with the NRPE daemon. The allowed_hosts directive includes localhost and a designated monitoring server IP, ensuring that only authorized systems can execute remote Nagios checks. This step confirms proper access control and secure NRPE communication as part of the monitoring setup.



```
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd
nrpe_user=nrpe

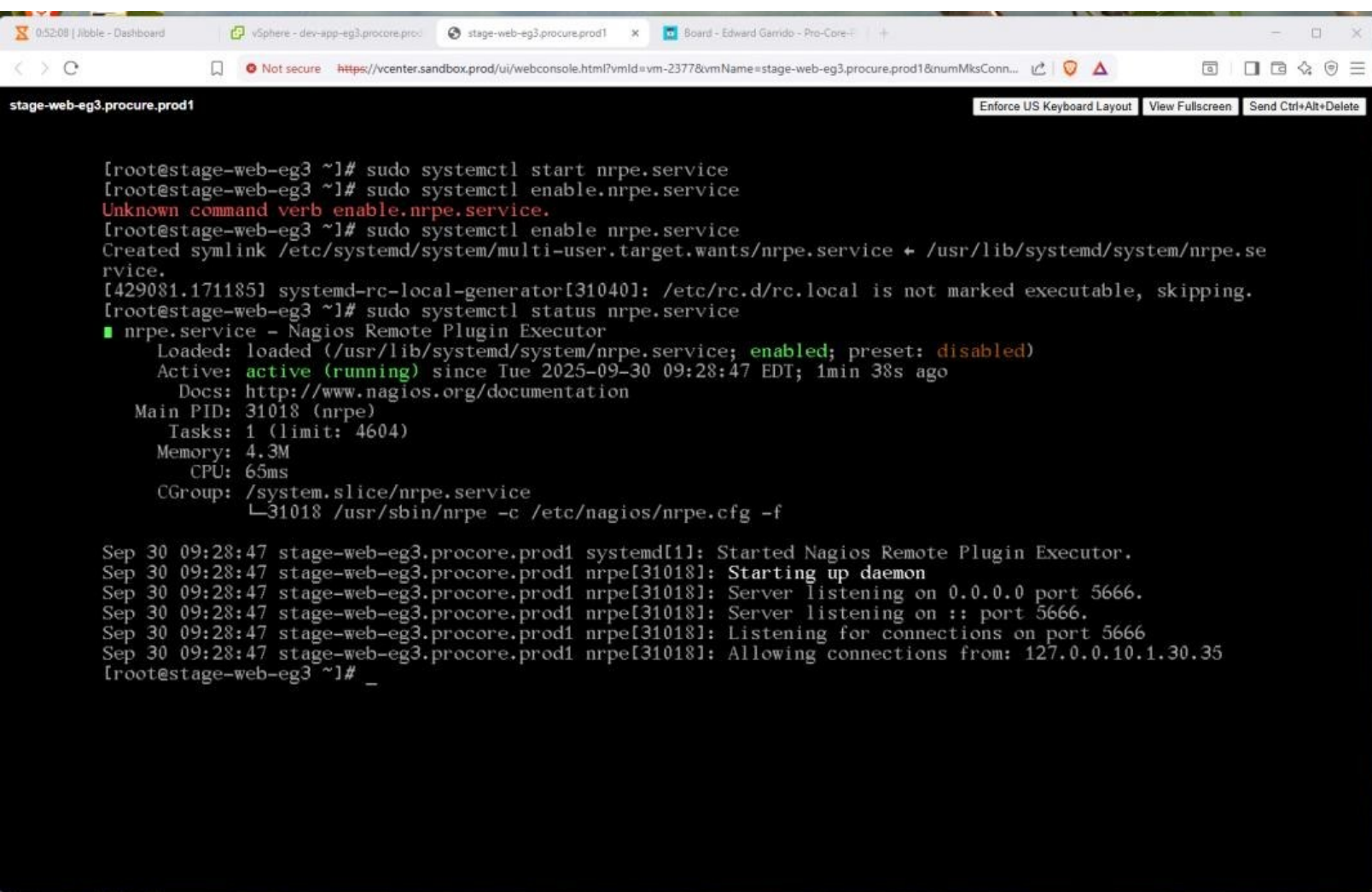
# NRPE GROUP
# This determines the effective group that the NRPE daemon should run as.
# You can either supply a group name or a GID.
#
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd
nrpe_group=nrpe

# 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=127.0.0.10.1.30.35

# COMMAND ARGUMENT PROCESSING
# This option determines whether or not the NRPE daemon will allow clients
-- INSERT --
```

106,33 22%

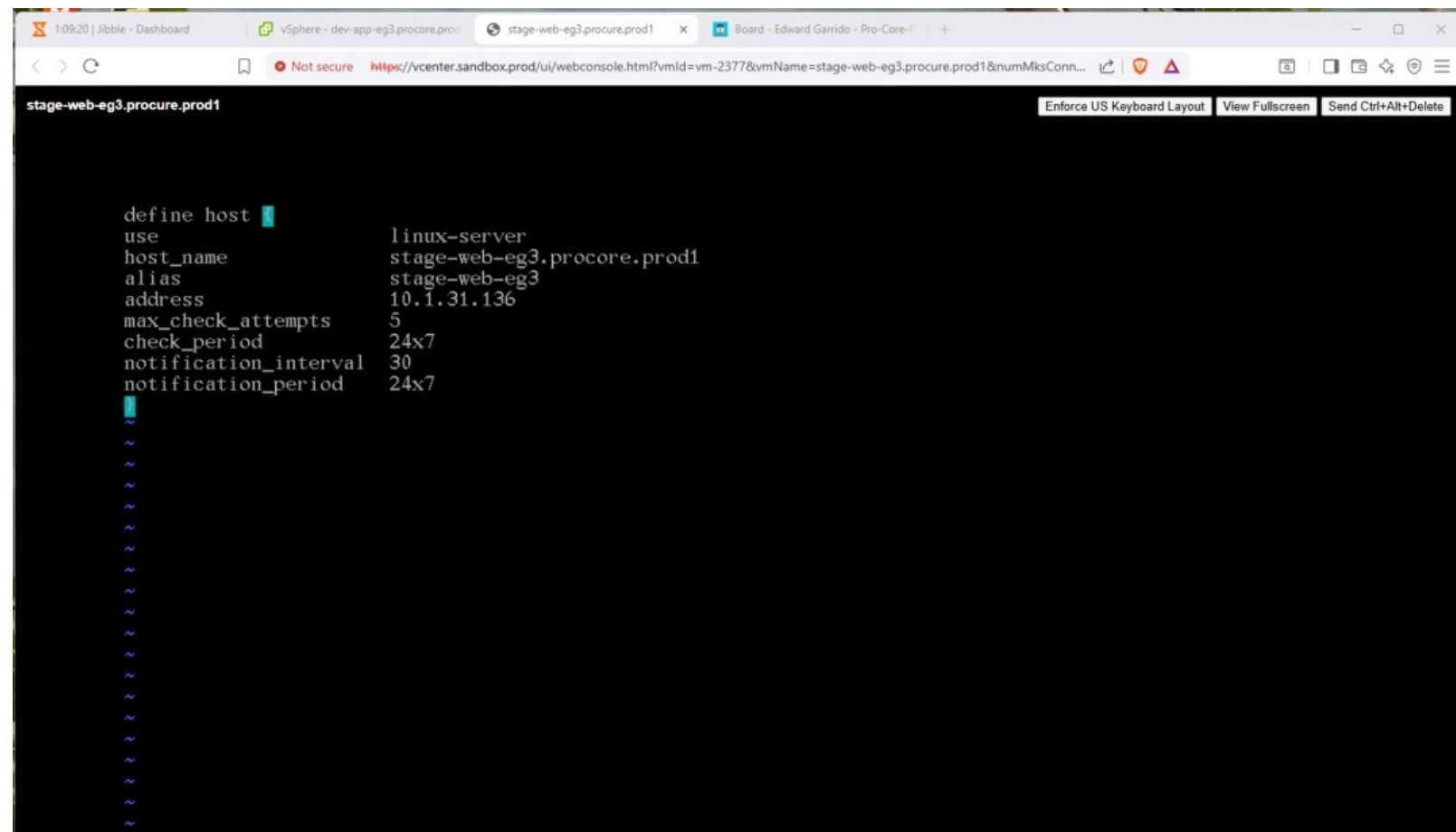
The NRPE service is started and enabled using systemctl, ensuring it runs automatically at boot. Service status output confirms that nrpe.service is active and running on stage-web-eg3.procore.prod1. The daemon is listening on the default NRPE port 5666 and explicitly accepting connections from the authorized monitoring hosts defined in the configuration file. This confirms a successful NRPE startup and validates readiness for remote Nagios health checks.



```
[root@stage-web-eg3 ~]# sudo systemctl start nrpe.service
[root@stage-web-eg3 ~]# sudo systemctl enable nrpe.service
Unknown command verb enable.nrpe.service.
[root@stage-web-eg3 ~]# sudo systemctl enable nrpe.service
Created symlink /etc/systemd/system/multi-user.target.wants/nrpe.service → /usr/lib/systemd/system/nrpe.service.
[429081.171185] systemd-rc-local-generator[31040]: /etc/rc.d/rc.local is not marked executable, skipping.
[root@stage-web-eg3 ~]# sudo systemctl status nrpe.service
■ nrpe.service - Nagios Remote Plugin Executor
   Loaded: loaded (/usr/lib/systemd/system/nrpe.service; enabled; preset: disabled)
   Active: active (running) since Tue 2025-09-30 09:28:47 EDT; 1min 38s ago
     Docs: http://www.nagios.org/documentation
  Main PID: 31018 (nrpe)
    Tasks: 1 (limit: 4604)
   Memory: 4.3M
      CPU: 65ms
   CGroup: /system.slice/nrpe.service
           └─31018 /usr/sbin/nrpe -c /etc/nagios/nrpe.cfg -f

Sep 30 09:28:47 stage-web-eg3.procore.prod1 systemd[1]: Started Nagios Remote Plugin Executor.
Sep 30 09:28:47 stage-web-eg3.procore.prod1 nrpe[31018]: Starting up daemon
Sep 30 09:28:47 stage-web-eg3.procore.prod1 nrpe[31018]: Server listening on 0.0.0.0 port 5666.
Sep 30 09:28:47 stage-web-eg3.procore.prod1 nrpe[31018]: Server listening on :: port 5666.
Sep 30 09:28:47 stage-web-eg3.procore.prod1 nrpe[31018]: Listening for connections on port 5666
Sep 30 09:28:47 stage-web-eg3.procore.prod1 nrpe[31018]: Allowing connections from: 127.0.0.10.1.30.35
[root@stage-web-eg3 ~]# _
```

A new Nagios host definition is created for stage-web-eg3.procore.prod1, associating the system with the linux-server template. The configuration specifies the hostname, alias, and IP address, along with monitoring parameters such as check frequency, maximum retry attempts, and notification intervals. This ensures the host is properly registered in Nagios and continuously monitored with 24x7 availability and alerting enabled.



```
define host {  
    use linux-server  
    host_name stage-web-eg3.procore.prod1  
    alias stage-web-eg3  
    address 10.1.31.136  
    max_check_attempts 5  
    check_period 24x7  
    notification_interval 30  
    notification_period 24x7  
}
```


Nagios is configured to load multiple host object definition files by explicitly including them in the main configuration. Entries are grouped and commented for clarity, with separate object files defined for development and staging systems. The configuration includes the newly added stage-web-eg3.procore.prod1 host file, ensuring that the monitoring server recognizes the host and applies the appropriate checks and alerting policies after a configuration reload.

```
#Mbeke Ekanem
cfg_file=/usr/local/nagios/etc/objects/dev-app-ME.cfg
cfg_file=/usr/local/nagios/etc/objects/stage-web-ME.cfg

#=====kgates
cfg_file=/usr/local/nagios/etc/objects/dev-app-kg2.cfg
cfg_file=/usr/local/nagios/etc/objects/stage-web-kg2.cfg

#-----
#LWaugh
cfg_file=/usr/local/nagios/etc/objects/dev-app-lw.procore.prod1.cfg
cfg_file=/usr/local/nagios/etc/objects/stage-web-lw.procore.prod1.cfg
#-----

#egiron
cfg_file=/usr/local/nagios/etc/objects/dev-app-eg2.procore.prod1.cfg
cfg_file=/usr/local/nagios/etc/objects/stage-web-eg2.procore.prod1.cfg
#-----

#egarrido
cfg_file=/usr/local/nagios/etc/objects/dev-app-eg3.procore.prod1.cfg
cfg_file=/usr/local/nagios/etc/objects/stage-web-eg3.procore.prod1.cfg
#-----
```

Nagios is configured to load multiple host object definition files by explicitly including them in the main configuration. Entries are grouped and commented for clarity, with separate object files defined for development and staging systems. The configuration includes the newly added stage-web-eg3.procore.prod1 host file, ensuring that the monitoring server recognizes the host and applies the appropriate checks and alerting policies after a configuration reload.

Board - Edward Garrido - Pro-Core-1	Nagios screenshot instructions	Nagios: 10.1.10.37		
10.1.10.37/nagios/				
stage-web-nr-procore.prod1		UP		10-03-2025 13:18:49
stage-web-nrivera.procore.prod1		DOWN		10-03-2025 13:18:50
stage-web-rc.procore.prod1.cfg		DOWN		10-03-2025 13:18:52
stage-web-rf.procore.prod1		DOWN		10-03-2025 13:18:53
stage-web-rf1		DOWN		10-03-2025 13:18:56
stage-web-rt.procore.prod1		DOWN		10-03-2025 13:18:58
stage-web-sd.procore.prod1-228		DOWN		10-03-2025 13:18:59
stage-web-sj1.procore.prod1		UP		10-03-2025 13:19:02
stage-web-sl.procore.prod1		UP		10-03-2025 13:23:39
stage-web-sm.procore.prod1		UP		10-03-2025 13:19:06
stage-web-ta.procore.prod1		UP		10-03-2025 13:23:16
stage-web-td.procore.prod1-153		DOWN		10-03-2025 13:19:08
stage-web-td3		UP		10-03-2025 13:19:12
stage-web-ts2.procore.prod1		DOWN		10-03-2025 13:19:13
stage-web-tu.procore.prod1-174		DOWN		10-03-2025 13:19:16
stage-web-ul-procore.prod1-216		UP		10-03-2025 13:19:16
stage-web-wr.procore.prod1-171		DOWN		10-03-2025 13:19:20
stage-web-yo.procore.prod1		UP		10-03-2025 13:19:20
{dev-app-eg3.procore.prod1}		DOWN		10-03-2025 13:23:22
{dev-app-nl@procore.prod1}		DOWN		10-03-2025 13:19:24
{stage-web-eg3.procore.prod1}		DOWN		10-03-2025 13:23:26
{stage-web-lg.procore.prod1-40}		DOWN		10-03-2025 13:19:30
{stage-web-nl.procore.prod1}		DOWN		10-03-2025 13:20:04

12

Results: 100 / 154 of 154 Matching Hosts

Summar

This work documents the successful integration of a staging web server into an existing Nagios monitoring environment using NRPE. The configuration includes installing required monitoring components, securing NRPE communication by restricting allowed hosts, enabling and validating the NRPE service, and defining Nagios host objects with appropriate monitoring and notification parameters. Final verification confirms that the monitoring server recognizes the host, executes remote checks successfully, and applies configuration changes as expected, demonstrating a complete and secure monitoring setup suitable for enterprise environments.