**🏗️ Nagios Architecture & How it Works**

**🔹 1. Nagios Core Components**

1️⃣ **Nagios Server (Core Engine)**

* The brain of Nagios.
* Schedules checks, processes results, and sends alerts.

2️⃣ **Plugins**

* Small executables/scripts (Bash, Python, Perl, etc.).
* Perform actual checks (like check\_ping, check\_http).
* Example: check\_ping google.com → “OK” or “DOWN”.

3️⃣ **Addons**

* Extend Nagios with extra features.
* Example: **Nagiosgraph** (for graphs), **NRPE/NRDP** (for remote monitoring).

4️⃣ **Agents (Optional)**

* Installed on client machines if deeper monitoring is needed.
* Example: NRPE (Nagios Remote Plugin Executor) runs checks on remote Linux/Windows hosts.

5️⃣ **Web Interface (CGI-based UI)**

* Dashboard for admins.
* Shows status of hosts/services (OK, Warning, Critical).
* Provides logs, history, and performance reports.

**🔹 2. Nagios Architecture (Flow Diagram)**

+-------------------+

| Nagios Server |

| (Core Engine) |

+---------+---------+

|

--------------------------

| | |

Plugins Agents Addons

(checks) (remote) (extras)

| | |

Hosts/Services (Servers, Apps, Networks)

**🔹 3. How Nagios Works (Step by Step)**

1. **Configuration Phase**
   * Admin defines monitoring rules in config files:
     + Which **hosts** (servers/devices) to monitor.
     + Which **services** (HTTP, MySQL, CPU, disk) to check.
     + Alert thresholds (e.g., CPU > 80% → Warning, CPU > 95% → Critical).
2. **Monitoring Phase**
   * Nagios runs **plugins** on schedule (e.g., every 5 minutes).
   * Plugins return a status:
     + OK (Service healthy)
     + WARNING (Service under stress)
     + CRITICAL (Service down)
     + UNKNOWN (Check failed)
3. **Processing Phase**
   * Nagios Core collects results.
   * Compares them against thresholds.
4. **Alerting Phase**
   * If issue detected → sends notification (Email, SMS, Slack, Webhook).
   * If system recovers → sends "Recovery" alert.
5. **Visualization Phase**
   * Results shown in the **Web Dashboard**.
   * Admins can check host/service status, uptime, historical trends.

**🔹 4. Example (Web Server Monitoring)**

* You configure Nagios to monitor a web server (check\_http).
* **Step 1:** Nagios schedules the check every 2 minutes.
* **Step 2:** Plugin runs → server responds with HTTP 200 (OK).
* **Step 3:** If next check fails (e.g., HTTP 500), Nagios marks service as **CRITICAL**.
* **Step 4:** Sends an alert →
* CRITICAL: Web Server (myapp.com) is DOWN!
* **Step 5:** Admin fixes server → Next check succeeds → Nagios sends **RECOVERY** message.

**Summary:**

In Nagios, the process starts when an **admin defines configuration files** on the Nagios server → these files specify which **hosts (servers, devices)** and **services (HTTP, MySQL, CPU, memory, disk, etc.)** should be monitored and at what intervals. The **Nagios daemon** reads these configs and schedules checks → if the host is local, it runs **plugins directly** on the server to collect results; if the host is remote, Nagios contacts the **NRPE agent** installed on the client machine → the NRPE agent executes monitoring plugins locally (e.g., check\_load, check\_http, check\_disk) → results are sent back to the **Nagios daemon** on the central server → the daemon evaluates results against thresholds (OK, WARNING, CRITICAL) and updates its internal status database → if an issue is detected, Nagios triggers the **alerting system** (email, SMS, Slack, etc.) → finally, all this data is reflected in the **Nagios Web Dashboard**, giving admins real-time visibility into the health and performance of the entire IT environment.

This way, you see the **end-to-end workflow**:

Config files → Nagios daemon → Plugins/NRPE → Results → Processing → Alerts → Dashboard.