Jenkins Master–Slave Architecture

1. Overview

Jenkins uses a Master–Slave (Controller–Agent) architecture to distribute and manage build job execution across multiple machines. The Master manages the job scheduling and configuration, while Slaves execute the jobs.

2. Master Responsibilities

- Provides the Jenkins web interface
- Accepts and schedules build jobs
- Assigns jobs to available agents
- Monitors agent status
- Stores build history and configurations

3. Slave Responsibilities

- Runs on a separate machine (physical or virtual)
- Listens for instructions from the master
- Executes assigned jobs and returns results
- Can have specific software or tools installed for specialized builds

4. Workflow

1. User triggers a build from Jenkins UI or an automated process.
2. Master checks job configuration and decides where to run it.
3. If running on a slave: instructions are sent to the slave, which executes the job and returns results to the master.
4. Master updates job status on the UI.

5. Communication

Communication between master and slave usually happens over Java Web Start (JNLP) or SSH. The master must be able to connect to the slave, or vice versa, depending on configuration.

6. Benefits

- Scalability through parallel job execution
- Platform diversity with different OS agents
- Load distribution to reduce master load
- Specialization with custom environments for certain builds

7. Diagram (Simplified Text Representation)

```
+-------------------+ | Jenkins Master | |-------------------| | UI / API
| | Job Scheduler | | Build Configs | +---------+---------+ |
----------------------- | | | +--------+--+ +---+--------+ +---+--------+
| Slave 1 | | Slave 2 | | Slave 3 | | (Linux) | | (Windows) | | (Mac) | |
Builds A | | Builds B | | Builds C | +-----------+ +------------+
+------------+
```