TeamCity Integration with Cloud Solutions

TeamCity integration with cloud (IAAS) solutions allows TeamCity to provision virtual machines running TeamCity agents on-demand based on the build queue state.

This page covers general information about the configuration of integration. For the list of currently supported solutions, refer to Available Integrations.

On this page:

General Description

In a large TeamCity setup with many projects, it can be difficult to predict the load on build agents, and the number of agents we need to be running. With the cloud agent integration configured, TeamCity will leverage clouds elasticity to provision additional build agents on-demand.

For each queued build TeamCity first tries to start it on one of the regular, non-cloud agents. If there are no usual agents available, TeamCity finds a matching cloud image with a compatible agent and starts a new instance for the image. TeamCity ensures that number of running cloud instances limit is not exceeded.

The integration requires:

- a configured virtual machine with an installed TeamCity agent in your cloud pre-configured to start the TeamCity agent on boot,
- a configured cloud profile in TeamCity.

Once a cloud profile is configured in TeamCity with one or several images, TeamCity does a test start of one instance for all the newly added images to learn about the agents configured on them. When the agents are connected, TeamCity stores their parameters to be able to correctly process build configurations-to-agents compatibility. An agent connected from a cloud instance started by TeamCity is automatically authorized, provided there are available agent licenses: the number of cloud agents is limited by the total number of agent licenses you have in TeamCity. After that the agent is processed as a regular agent.

Depending on the profile settings, when TeamCity realizes it needs more agents, it can either

- start an existing virtual machine and stop it (after the build is finished or an idle timeout elapses). The machines that are stopped will be deallocated so the virtual machine fee does not apply when the agent is not active. The storage cost for this type of TeamCity agent will still apply.
- create a new virtual machine from an image. Such machines will be destroyed (after the build is finished or an idle timeout elapses). This ensures that the machines will incur no further running costs.

The disconnected agent will be removed from the authorized agents list and deleted from the system to free up TeamCity build agent licenses.

Available Integrations

Integration with cloud solutions is implemented as plugins. The platform-specific details are covered on the following pages:

- Amazon EC2
- VMWare vSphere

Also available as separate plugins are Windows Azure, Google Cloud Agents and others. New integrations can be implemented using custom TeamCity plugins, see Implementing Cloud support.

TeamCity Setup for Cloud Integration

This section describes general steps required for cloud integration.

Preparing a virtual machine with an installed TeamCity agent

The requirements for a virtual machine/image to be used for TeamCity cloud integration:

- The TeamCity agent must be correctly installed and configured to start automatically on the machine startup.
- the `buildAgent.properties` file can be left "as is". The `serverUrl`, `name`, and `authorizationToken` properties can be left empty or set to any value, they are ignored when TeamCity starts the instance unless otherwise specifically stated in the platform-specific documentation.

Provided these requirements are met, the usual TeamCity agent installation and cloud-provider image bundling procedures are applicable.

If you need the connection between the server and the agent machine to be secure, you will need to set up the agent machine to establish a secure tunnel (e.g. VPN) to the server on boot so that the TeamCity agent receives data via the secure channel. Please keep in mind that communication between TeamCity agent and server is bi-directional and requires an open port on the agent as well as on the server.

Preparing a virtual machine

1. Create and start a virtual machine with desired OS installed.
2. Connect and log in to the virtual machine.
3. Configure the running instance:
    a. Install and configure build agent.
        - Configure the server name and agent name in the `buildAgent.properties` file — this is optional if TeamCity will be configured to launch the image, but it is useful to test the agent is configured correctly.
        - It usually makes sense to specify `tempDir` and `workDir` in `conf/buildAgent.properties` to use a non-system drive (e.g. **`D`** drive under Windows)
    b. Install any additional software necessary for the builds on the machine (e.g. Java, the .Net framework)
    c. Start the agent and wait until it connects to server, ensure it is working OK and is compatible with all necessary build configurations (in the TeamCity Web UI, go to the Agents page, select the build agent and view the Compatible Configurations tab), etc.
    d. Configure the system so that the agent is started on the machine boot (and make sure TeamCity server is accessible on the machine boot).
    e. Check the port on which the build agent will listen for incoming data from TeamCity and open the required firewall ports (usually 9090).
4. Test the setup by rebooting machine and checking that the agent connects normally to the server. Once the agent connects, it will automatically update all the plugins. Please wait until the agent is connected completely so that all plugins are downloaded to the agent machine.

If you want TeamCity to start an existing virtual machine and stop it after the build is finished or an idle timeout elapses, the setup above is all you need. If you want TeamCity to create and start virtual machines from an image and terminate the machine after use, the image should be captured from the virtual machine that was created.

Capturing an image from a virtual machine

Do the following:

1. Complete the steps for creating a virtual machine.
    - Remove any temporary/history information in the system.
    - Stop the agent (under Windows, stop the service but leave it in the Automatic startup type)
    - (optional) Delete the content of the `logs` and `temp` directories in the agent home
    - (optional) Clean up the `<Agent Home>/conf/` directory from platform-specific files
    - (optional) Change the `buildAgent.properties` file to remove the `name`, `serverUrl`, and `authorizationToken` properties unless otherwise specifically stated in the platform-specific documentation.
2. Make a new image from the running instance. Refer the cloud documentation on how to do this.

TeamCity agent auto-upgrades whenever distribution of agent (e.g. after TeamCity upgrade) or agent plugins on the server changes. If you want to reduce the agent startup time, you might want to capture a new virtual machine image after the agent distribution or plugins have been updated.

Configuring a cloud profile in TeamCity

A cloud profile is a collection of settings for TeamCity to start virtual machines with installed TeamCity agents on-demand. Prior to TeamCity 2017.1, profiles are configured on the TeamCity | Administration | Agent Cloud page. In the later versions, cloud profiles are configured at the project level, on the dedicated page of project settings.

Estimating Costs

The cloud provider pricing applies. Please note that the charges can depend on the specific configuration implemented to deploy TeamCity. We advise you to check your configuration and the cloud account data regularly in order to discover and prevent unexpected charges as early as possible.

Please note that traffic volumes and necessary server and agent machines characteristics depend a big deal on the TeamCity setup and nature of the builds run.

Traffic Estimate

Here are some points to help you estimate TeamCity-related traffic:

If the TeamCity server is not located within the same region or affinity group as the agent, the traffic between the server and agent is subject to usual external traffic charges imposed by your provider.
When estimating traffic, please remember that there are many types of traffic related to TeamCity (see the non-complete list below).

External connections originated by server:

- VCS servers
- Email and Jabber servers
- Maven repositories
- NuGet repositories

Internal connections originated by server:

- TeamCity agents (checking status, sending commands, retrieving information like thread dumps, etc.)

External connections originated by agent:

- VCS servers (in case of agent-side checkout)
- Maven repositories
- NuGet repositories
- any connections performed from the build process itself

Internal connections originated by agent:

- TeamCity server (retrieving build sources in case of server-side checkout or personal builds, downloading artifacts, etc.)

Usual connections used by the server

- Web browsers
- IDE plugins

Running Costs

Cloud providers calculate costs based on the virtual machine uptime, so it is recommended to adjust the timeout setting according to your usual builds length. This reduces the amount of time a virtual machine is running.
It is also highly recommended to set an execution timeout for all your builds so that a build hanging does not cause prolonged instance running with no payload.