

To run a dedicated Minecraft server, you need to have a dedicated server machine, plenty of RAM, and plenty of bandwidth. Why not let Google handle these requirements for you? In this tutorial, you'll learn how to install, configure, and run a standard Java Minecraft server on Compute Engine. This server is compatible with standard Java-based desktop Minecraft clients.

This tutorial is not approved by or associated with Mojang or Minecraft.

Your Minecraft server software will run on a [Compute Engine instance](/compute/docs/instances) (/compute/docs/instances), which is a [virtual machine](https://wikipedia.org/wiki/Virtual_machine) (https://wikipedia.org/wiki/Virtual_machine) that runs on Google's infrastructure. This tutorial uses the default [machine type](/compute/docs/machine-types) (/compute/docs/machine-types) for Compute Engine instances, `n1-standard-1`. The `n1-standard-1` machine type includes a 10 GB boot disk, 1 virtual CPU (vCPU), and 3.75 GB of RAM. This machine type runs [Debian Linux](https://www.debian.org/) (https://www.debian.org/) by default.

To make sure there's plenty of room for your Minecraft server's world data, you'll also attach a high performance 50 GB persistent solid-state drive (SSD) to your instance. With the addition of this persistent SSD, your instance will satisfy the system requirements for a [dedicated Minecraft server](http://minecraft.gamepedia.com/Server/Requirements/Dedicated#Console) (http://minecraft.gamepedia.com/Server/Requirements/Dedicated#Console), supporting up to 50 players comfortably.

Note: The above machine and disk specifications are not meant to be a general recommendation for all Minecraft servers running on Compute Engine. When setting up your own Minecraft server, choose a machine type, disk type, and disk size that reflects your own I/O and resource needs.

- Create a Compute Engine virtual machine instance
- Install and configure the Minecraft server
- Set up automatic backups of your Minecraft world data

This tutorial uses the following billable components of Google Cloud:

- Compute Engine virtual machines
- Compute Engine persistent disks
- Cloud Storage

To generate a cost estimate based on your projected usage, use the [pricing calculator](/products/calculator) (/products/calculator). New Google Cloud users might be eligible for a [free trial](/free-trial) (/free-trial).

1. [Sign in](https://accounts.google.com/Login) (https://accounts.google.com/Login) to your Google Account.

If you don't already have one, [sign up for a new account](https://accounts.google.com/SignUp) (https://accounts.google.com/SignUp).

2. In the Cloud Console, on the project selector page, select or create a Cloud project.

★ **Note:** If you don't plan to keep the resources that you create in this procedure, create a project instead of selecting an existing project. After you finish these steps, you can delete the project, removing all resources associated with the project.

[Go to the project selector page](https://console.cloud.google.com/projectselector2/home/dashboard) (https://console.cloud.google.com/projectselector2/home/dashboard)

3. Make sure that billing is enabled for your Google Cloud project. [Learn how to confirm billing is enabled for your project](/billing/docs/how-to/modify-project) (/billing/docs/how-to/modify-project).

Start by creating and configuring a new Compute Engine instance.

1. In the Cloud Console, go to the **VM instances** page:

[Go to the VM instances page](https://console.cloud.google.com/compute/instances) (https://console.cloud.google.com/compute/instances)

You'll arrive at a dialog prompting you to create a new Compute Engine instance.

2. Click **Create** to get started.
3. On the **Create an instance** page, configure your instance as follows:
 - Name your instance. This tutorial uses the instance name `mc-server` throughout.
 - Select the region and zone (`/compute/docs/zones#available`) in which you want your instance to be hosted. This tutorial uses the region `us-centra11` (Iowa) and the zone `us-centra11-f` throughout.
 - In the **Boot disk** section, click **Change**. The **Boot disk** dialog will pop up.
 - Change the disk type to **SSD Persistent Disk**.
4. Click **Select** to commit the change and close the dialog.

That's it for basic configuration! But don't create your new instance just yet. To meet the requirements for a dedicated Minecraft server, you'll need to configure some advanced settings as well.

Later in this tutorial, you'll learn how to back up your world data to Cloud Storage (`#schedule_backups`), which requires your instance to have read and write access to Cloud Storage. To enable access:

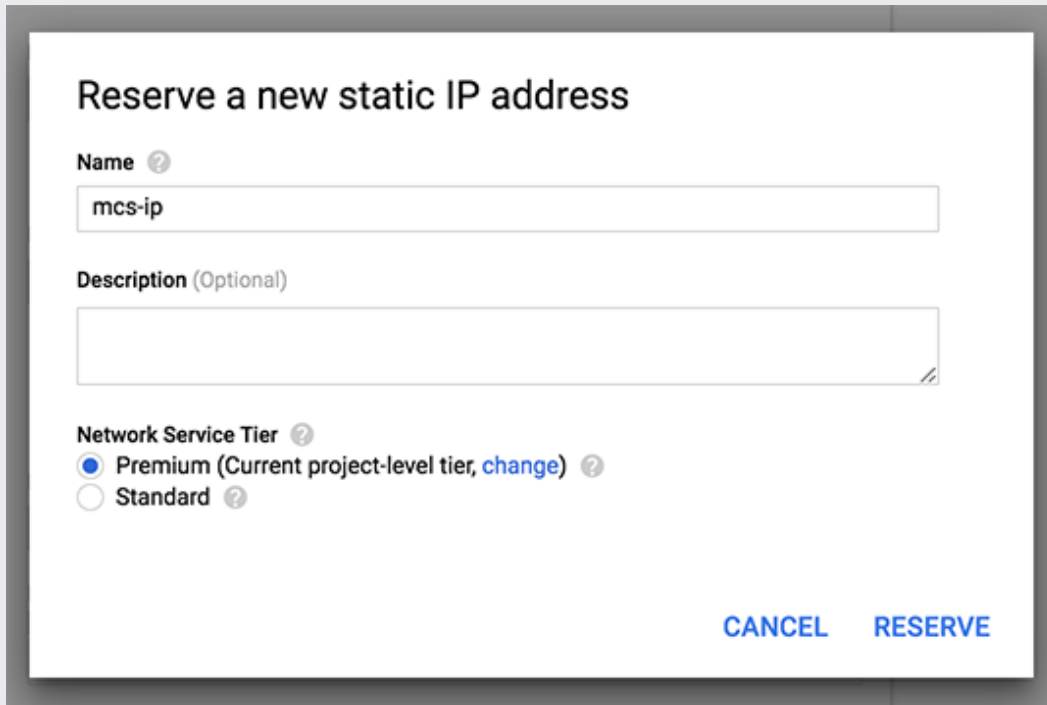
1. Under **Identity and API access**, click the **Service account** dropdown and select **Compute Engine default service account**.
2. Set **Access scopes** to **Set access for each API**.
3. In the **Storage** dropdown, select **Read Write**.

Next, tag your instance. Later in the tutorial, you'll use this tag to create a firewall rule that will allow external Minecraft clients to access your server.

1. Click **Management, security, disks, networking, sole tenancy** to reveal a set of tabs for advanced settings.
2. In the **Networking** tab, add the tag `minecraft-server` to the **Network tags** field.

To be able to forward incoming requests to your instance dependably, your instance will need to have a static IP address. To add a static IP address to your instance:

1. In the **Networking** tab, in the **Network interfaces** section, click **Default**. The **Network interface** configuration menu appears.
2. Click the **External IP** dropdown and select **Create IP address**. A dialog pops up:



Reserve a new static IP address

Name [?]
mcs-ip

Description (Optional)

Network Service Tier [?]
 Premium (Current project-level tier, [change](#)) [?]
 Standard [?]

CANCEL RESERVE



3. Name your IP address **mcs-ip**.
4. Click **Reserve** to create the address.
5. Click **Done** to commit your changes and close the **Network interface** configuration menu.


Next, you'll attach a persistent disk to your instance. Unlike boot disks, persistent disks are not tied to the life of your Compute Engine instance. For example, if your hosting needs change over time, you can move the disk to a more suitable machine type later.


The specific type of persistent disk you will use in this tutorial is a persistent SSD. This type of persistent disk supports very fast I/O operations, which can help reduce server lag.


To add a persistent disk to your instance:

1. In the **Disks** tab, in the **Additional disks** section, click **Add new disk**. A disk creation dialog appears appears:

disk-1 (Image)  

Name (Optional) 

Type 

Standard persistent disk 



Source type 

Image Blank disk


Source image 


Mode

Read/write
 Read only

Deletion rule
When deleting instance

Keep disk
 Delete disk

Size (GB)  (Optional)

Estimated performance 

Operation type	Read	Write
Sustained random IOPS limit		
Sustained throughput limit (MB/s)		

Encryption
Data is encrypted automatically. Select an encryption key management solution.

Google-managed key
No configuration required

Customer-managed key
Manage via Google Cloud Key Management Service

Customer-supplied key
Manage outside of Google Cloud

This new disk will be added once you create the new instance

2. In the dialog, fill out the form as follows:

- *Name:* `minecraft-disk`
- *Disk type:* SSD Persistent Disk
- *Source type:* Blank disk
- *Size (GB):* `50`

3. Click **Done**. When you create the instance, the disk will be created and attached automatically.

That's it for instance configuration! Click the **Create** button at the bottom of the page to create your new instance. This action will also take you back to the **VM instances** page.

Note: It can take up to 20 seconds for your instance to be created.

At this point, your disk is attached to your instance, but it hasn't yet been mounted to the instance. That's okay—if you mounted the disk now, you wouldn't be able to do much with it. That's because, as with any disk, your persistent disk first needs to be formatted with a [filesystem](https://wikipedia.org/wiki/File_system) (https://wikipedia.org/wiki/File_system) that your operating system—in this case, Debian Linux—can understand.

Begin by establishing an SSH connection with your instance. In the row for **mc-server** on your **VM instances** page, click **SSH** to open a browser-based SSH terminal:



```
user@mc-server: ~ - Google Chrome
https://cloudssh.developers.google.com/projects/minecraft-on-gce-micro/zones/us-centr
Connected, host fingerprint: ssh-rsa 2048 61:EA:E2:1D:CE:B5:07:15:0E:5A:12:E0:6A:C...S:
Linux mc-server 3.16.0-0.bpo.4-amd64 #1 SMP Debian 3.16.7-ckt4-3~bpo70+1 (2015-02-12) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
user@mc-server:~$
```

After your SSH terminal opens, create a new directory named `minecraft` in your instance's home directory:

You'll use this directory as a mount point for your persistent disk.

Next, format your disk:

Warning: The following command destroys all of the data on your disk. As such, it should only be used if you are mounting a new persistent disk. If you need to mount your disk without formatting, run the `mount` command instead:

Finally, mount your disk:

Your persistent disk is officially mounted. Now it's time to do what you came here to do in the first place: install and run your Minecraft server.

The Minecraft server runs on top of the Java Virtual Machine (JVM)

(https://wikipedia.org/wiki/Java_virtual_machine), so it requires the Java Runtime Environment (JRE) to run. Because the server doesn't need a graphical user interface, this tutorial uses the headless version of the JRE. This approach reduces the JRE's resource usage on your machine, helping ensure that the Minecraft server has plenty of room to expand its own resource usage if needed.

You'll need to update the Debian repositories on your Debian installation before you can download and install the headless version of the JRE. To do so, run the following command in your SSH terminal:

After your repositories are updated, you can install the headless JRE:

Now that you've set up the JRE, it's time to download and install the Minecraft server. Begin by navigating to your `minecraft` directory:

Because the `minecraft` directory contains your mounted persistent disk, you're going to need a special level of access known as root user access (<https://wikipedia.org/wiki/Superuser>) to run commands on it. Run the following to become the root user:

Next, download the current Minecraft server's Java archive file (JAR) (https://wikipedia.org/wiki/JAR_%28file_format%29) to your instance. Visit the Minecraft download page (<https://www.minecraft.net/en-us/download/server/>), copy the file's URL from the download link, and then replace the URL in the following command with that URL.

Start the server for the first time:

The first run is a little anticlimactic: the server simply starts, reports some issues, and stops. If you run the following command, however, you'll notice that some new files have been created in the `minecraft` directory:

Among these new files, you'll find a file called `eula.txt`. Open this file for editing:

This file contains a single boolean variable, `eula`. To use the Minecraft server, you must accept the terms of the Minecraft End User License Agreement (EULA) (https://account.mojang.com/documents/minecraft_eula). If you accept the terms of the EULA, set the value of `eula` from `false` to `true`, then save and exit.

If you start the Minecraft server again at this point, it will be tied to the life of your SSH session—that is, if you close your SSH terminal, the server will be terminated as well. To get around this issue, you can use `screen`, an application that allows you to create a virtual terminal that can be "detached," becoming a background process, or "reattached," becoming a foreground process. When a virtual terminal is detached to the background, it will run whether you are logged in or not.

In your SSH terminal, run the following to install `screen`:

Next, start your Minecraft server in a `screen` virtual terminal. Use the `-S` flag to name your terminal `mcs`:

Detach the `screen` terminal by pressing **Ctrl + a**, then typing **d**. The terminal will continue to run in the background. To reattach the terminal, run `screen -r <terminal_name>` as follows:

Finally, detach your `screen` terminal again if needed. Type `exit` once to leave root user mode, then type `exit` again to close your SSH connection.

Congratulations! You now have a running Minecraft server. However, it's not quite ready to be shared just yet. Before you can share your server, you need to set up a firewall rule that will allow people to access it.

To be able to forward incoming requests to your instance dependably, you'll need to create a firewall rule. To do so:

1. In the Cloud Console, go to the **Firewall rules** page.

[Go to the Firewall rules page](https://console.cloud.google.com/networking/firewalls/list) (https://console.cloud.google.com/networking/firewalls/list)

2. Click **Create firewall rule**.
3. On the **Create a firewall rule** page, fill out the form as follows:
 - *Name:* `minecraft-rule`
 - *Target tags:* `minecraft-server`
 - *Source filter:* IP ranges
 - *Source IP ranges:* `0.0.0.0/0`
 - *Protocols or ports:* Select **tcp**, and then enter port **25565** into the field provided.

★ **Note:** By default, the Minecraft server uses 25565 as its default listening port. You can change this port number by editing `server.properties`, the configuration file for your Minecraft server, and restarting your server. Make sure to change the number in the firewall rule as well.

4. Click **Create** to create your new firewall rule. Users can now access your server from their local Minecraft clients.

You might want to edit the server's default properties. To do so:

1. Reattach the server's `screen` terminal.
2. Enter `\stop` to stop the Minecraft server.
3. Edit the file `server.properties`. You can find information on each property type and its possible values on the [server.properties](http://minecraft.gamepedia.com/Server.properties) (http://minecraft.gamepedia.com/Server.properties) page of the [Minecraft Wiki](http://minecraft.gamepedia.com/) (http://minecraft.gamepedia.com/).

4. Restart the server.

5. Detach the `screen` terminal.

Whether you're running a local Minecraft client or running a Minecraft server, it's a good idea to back up your Minecraft world data on a regular basis. This section demonstrates how to set up regular backups of your world data using Cloud Storage.

Cloud Storage offers several different [storage classes](/storage/docs/storage-classes) that are optimized for different use cases. This tutorial uses Standard Storage, which provides up to 5GB/month [free of charge](/storage/pricing#cloud-storage-always-free).

Begin by establishing an SSH connection with your instance from the [VM instances page](https://console.cloud.google.com/compute/instances) (<https://console.cloud.google.com/compute/instances>) in your Cloud Console. After the terminal opens, become the root user.

Create a new Cloud Storage Standard Storage bucket, replacing `us-central1` with the [Cloud Storage region](/storage/docs/locations#location-r) closest to you and `[PROJECT_ID]` with your project ID. You'll use this bucket to store your backups.

Note: Your bucket name must be unique across Cloud Storage.

Next, create a new shell script file, `backup.sh`, in your `minecraft` folder, and open it for editing.

Paste the following script into the file. Replace `[BUCKET_NAME]` with your Cloud Storage bucket name:

This script begins by saving the current state of your world data and pausing your server's auto-save functionality. Next, the script backs up your server's world data directory (`world`), placing its contents in a timestamped directory (`[TIMESTAMP]-world`) in your Cloud Storage bucket. After the script finishes backing up the data, it resumes auto-saving on the Minecraft server.

Save and exit, then run the following to make your script executable:

Test the script:

After the script finishes, visit the [Storage browser](https://console.cloud.google.com/storage/browser) (<https://console.cloud.google.com/storage/browser>) in the Cloud Console and click on your bucket. You should see a timestamped backup of your `world` directory.

Unless you prefer to manually initialize each backup, you'll probably want to have your script run automatically at predictable intervals. To accomplish this task, you need to schedule a new [cron job](https://wikipedia.org/wiki/Cron) (<https://wikipedia.org/wiki/Cron>).

To schedule a cron job, begin by opening the cron table for editing:

Scroll to the bottom of the file and paste the following line, which specifies that `backup.sh` will run every 4 hours:

Save and exit.

That's it! Your Compute Engine instance will now automatically back up your world data to a Cloud Storage bucket every 4 hours.

If you back up your world data every 4 hours, that means you're backing it up 6 times a day, 72 times a week, and roughly 300 times a month. You can remove old backups automatically by using a feature of Cloud Storage called [Object Lifecycle Management](/storage/docs/lifecycle) (`/storage/docs/lifecycle`). This feature allows you to configure your Cloud Storage bucket to automatically archive or delete old backups after a certain time or if there are newer backups available.

To set up your Cloud Storage bucket to remove backups automatically:

1. Open the Cloud Storage browser in the Cloud Console:

[Go to the Cloud Storage browser](https://console.cloud.google.com/storage/browser) (<https://console.cloud.google.com/storage/browser>)

2. In the bucket list, find your Minecraft backups bucket.
3. Click **None** in the bucket's **Lifecycle** column. The **View object lifecycle rules** page appears.
4. Click **Add rule**.
5. In the **Select object conditions** section, select **Age**. Set the age to 7 days and click **Continue**.
6. In the **Select action** section, select **Delete** and click **Continue**.
7. Click **Save** to save your settings. You're taken back to the **View object lifecycle rules** page.

Each backup will now be deleted one week after your backup script sends it to Cloud Storage.

If you don't need to run your Minecraft server, you should shut it down to avoid incurring unnecessary expenses.

Begin by establishing an SSH connection with your instance from the [VM instances page](https://console.cloud.google.com/compute/instances) (<https://console.cloud.google.com/compute/instances>) in your Cloud Console. After the terminal opens, stop the Minecraft server by passing a `\stop` command to the `screen` terminal in which it is running.

Now that you've stopped your Minecraft server, you can safely shut down your instance. On the **VM instances** page, click on your instance's name and then click the **Stop** button at the top of the page. You will be logged out of your SSH session.

To start up your instance again, visit your instance page and then click the **Start** button at the top of the page. To start the Minecraft server again, you can establish an SSH connection with the instance, remount your persistent disk, and start your Minecraft server in a new `screen` terminal, as described in the [Run the Minecraft server](#) (`#run_the_minecraft_server`) section.

If you plan to shut down your server regularly, consider adding startup and shutdown scripts to your instance to automate common startup and shutdown procedures. You can automate the startup procedure as follows:

1. Go to the VM instances page in the Cloud Console:

[Go to the VM instances page](https://console.cloud.google.com/compute/instances) (<https://console.cloud.google.com/compute/instances>)

2. Click your instance name.
3. Click **Edit**.

4. In the **Custom metadata** section, add a new key called **startup-script** and copy the following script into its **Value** field:

When you restart your instance, this script will automatically mount your Minecraft disk to the appropriate directory, reinstall your cron job if needed, start your Minecraft server in a **screen** session, and detach the session.

To automate your shutdown procedure, add another key called **shutdown-script** and copy the following into its **Value** field:

At the bottom of the page, click **Save** to commit your changes. When you stop your instance, this script will create a backup of the latest game data and shut down your Minecraft server before the instance shuts down.

The [Minecraft Wiki](http://minecraft.gamepedia.com/) (<http://minecraft.gamepedia.com/>) provides tons of useful resources for Minecraft players and Minecraft server administrators alike.

The vanilla Minecraft server you installed in this tutorial is just one of the many Minecraft servers available. Visit [Custom servers](http://minecraft.gamepedia.com/Custom_servers) (http://minecraft.gamepedia.com/Custom_servers) in the [Minecraft](#)

Wiki (<http://minecraft.gamepedia.com/>) to see a list of alternative servers that have been optimized for game types, machine requirements, easy modifications, and more.

Try out other Google Cloud features for yourself. Have a look at our tutorials (/docs/tutorials).