CONTENTS

// TUTORIAL //

How To Install Apache Tomcat 9 on Debian 10

Published on October 24, 2019

Debian 10 Debian Java

Justin Ellingwood and Hanif Jetha

English

Not using Debian 10?
Choose a different version or distribution.

Debian 10

Introduction

Apache Tomcat is a web server and servlet container that is used to serve Java applications. Tomcat is an open source implementation of the Java Servlet and Java Server Pages technologies, released by the Apache Software Foundation. This tutorial covers the basic installation and some configuration of the latest release of Tomcat 9 on your Debian 10 server.

Prerequisites

Before you begin with this guide, you should have a non-root user with `sudo` privileges set up on your server. You can learn how to do this by completing our Debian 10 initial server setup guide.

Step 1 – Install Java

Tomcat requires Java to be installed on the server so that any Java web application code can be executed. We can satisfy that requirement by installing OpenJDK with apt.

First, update your apt package index:

```
$ sudo apt update
```

Copy

Then install the Java Development Kit package with apt:

```
$ sudo apt install default-jdk
```

Copy

Now that Java is installed, we can create a `tomcat` user, which will be used to run the Tomcat service.

Step 2 – Create Tomcat User

For security purposes, Tomcat should be run as an unprivileged user (i.e. not **root**). We will create a new user and group that will run the Tomcat service.

First, create a new **tomcat** group:

```
$ sudo groupadd tomcat
```

Copy

Next, create a new **tomcat** user. We'll make this user a member of the **tomcat** group, with a home directory of `/opt/tomcat` (where we will install Tomcat), and with a shell of `/bin/false` (so nobody can log into the account):

```
$ sudo useradd -s /bin/false -g tomcat -d /opt/tomcat tomcat
```

Copy

Now th ur **tomcat** user is set up, let's download and install Tomcat.

Step 3 – Install Tomcat

The best way to install Tomcat 9 is to download the latest binary release then configure it manually.

Find the latest version of Tomcat 9 at the Tomcat 9 Downloads page. At the time of writing, the latest version is **9.0.27**, but you should use a later stable version if it is available. Under the **Binary Distributions** section, then under the **Core** list, copy the link to the "tar.gz".

Next, return to your SSH session and move to the `/tmp` directory on your server. This is a good directory to download ephemeral items, like the Tomcat tarball, which we won't need after extracting the Tomcat contents:

```
$ cd /tmp
```
Copy

We'll use the `curl` command-line tool to download the tarball. Install `curl`:

```
$ sudo apt install curl
```
Copy

Now, use `curl` to download the link that you copied from the Tomcat website:

```
$ curl -O http://www-eu.apache.org/dist/tomcat/tomcat-9/v9.0.11/bin/apache
```
Copy :-9

We will install Tomcat to the `/opt/tomcat` directory. Create the directory, then extract the archive to it with these commands:

```
$ sudo mkdir /opt/tomcat
$ sudo tar xzvf apache-tomcat-9*tar.gz -C /opt/tomcat --strip-components=1
```
Copy

Next, we will set up the proper user permissions for our installation.

Step 4 – Update Permissions

The **tomcat** user that we created needs to have access to the Tomcat installation. We'll set that up now.

Change to the directory where we unpacked the Tomcat installation:

Tomcat needs to know where Java is installed. This path is commonly referred to as `JAVA_HOME`. The easiest way to look up that location is by running this command:

```
$ sudo update-java-alternatives -l
```

Copy

```
Output
java-1.11.0-openjdk-amd64      1111      /usr/lib/jvm/java-1.11.0-openjdk-amd64
```

Your `JAVA_HOME` is the output from the last column (highlighted above). Given the example above, the correct `JAVA_HOME` for this server would be:

```
JA
/usr/lib/jvm/java-1.11.0-openjdk-amd64
```

Your `JAVA_HOME` may be different.

With this piece of information, we can create the systemd service file. Open a file called `tomcat.service` in the `/etc/systemd/system` directory by typing:

```
$ sudo nano /etc/systemd/system/tomcat.service
```
Copy

Paste the following contents into your service file. Modify the value of `JAVA_HOME` if necessary to match the value you found on your system. You may also want to modify the memory allocation settings that are specified in `CATALINA_OPTS`:

/etc/systemd/system/tomcat.service

```
[Unit]
Description=Apache Tomcat Web Application Container
After=network.target

[Service]
Type=forking

Environment=JAVA_HOME= /usr/lib/jvm/java-1.11.0-openjdk-amd64
Environment=CATALINA_PID=/opt/tomcat/temp/tomcat.pid
Environment=CATALINA_HOME=/opt/tomcat
Environment=CATALINA_BASE=/opt/tomcat
Environment='CATALINA_OPTS=-Xms512M -Xmx1024M -server -XX:+UseParallelGC'
Environment='JAVA_OPTS=-Djava.awt.headless=true -Djava.security.egd=file:/dev/./urand

ExecStart=/opt/tomcat/bin/startup.sh
ExecStop=/opt/tomcat/bin/shutdown.sh

User=tomcat
Group=tomcat
UMask=0007
RestartSec=10
Restart=always

[Install]
WantedBy=multi-user.target
```

When you are finished, save and close the file.

Next, reload the systemd daemon so that it knows about our service file:

```
$ sudo systemctl daemon-reload
```
Copy

Start the Tomcat service by typing:

```
$ sudo systemctl start tomcat
```

Copy

Double check that it started without errors by typing:

```
$ sudo systemctl status tomcat
```

Copy

You should see output similar to the following:

```
Output
● tomcat.service - Apache Tomcat Web Application Container
   Loaded: loaded (/etc/systemd/system/tomcat.service; disabled; vendor preset: enabl
   Active: active (running) since Thu 2019-10-24 17:18:11 UTC; 4s ago
  Process: 5962 ExecStart=/opt/tomcat/bin/startup.sh (code=exited, status=0/SUCCESS)
 Main PID: 5970 (java)
    Tasks: 44 (limit: 2377)
   Memory: 184.2M
   CGroup: /system.slice/tomcat.service
           └─5970 /usr/lib/jvm/java-1.11.0-openjdk-amd64/bin/java -Djava.util.logging

Oct 24 17:18:10 tomcat systemd[1]: Starting Apache Tomcat Web Application Container..
Oct 24 17:18:11 tomcat startup.sh[5962]: Tomcat started.
Oct 24 17:18:11 tomcat systemd[1]: Started Apache Tomcat Web Application Container.
```

This confirms that Tomcat is up and running on your server.

Step 6 – Adjust the Firewall and Test the Tomcat Server

Now that the Tomcat service is started, we can test to make sure the default page is available.

Before we do that, we need to adjust the firewall to allow our requests to get to the service. If you followed the prerequisites, you will have a `ufw` firewall enabled currently.

Tomcat uses port `8080` to accept requests. Allow traffic to that port by typing:

```
$ sudo ufw allow 8080
```

Copy

With the firewall modified, you can access the default splash page by going to your domain or IP address followed by `:8080` in a web browser:

```
Open in web browser
http:// server_domain_or_IP :8080
```

You will see the default Tomcat splash page, in addition to other information. However, if you click the links for the Manager App, for instance, you will be denied access. We can configure that access next.

If you were able to successfully access Tomcat, now is a good time to enable the service file so that Tomcat automatically starts at boot:

```
$ sudo systemctl enable tomcat
```
Copy

Step 7 – Configure Tomcat Web Management Interface

In order to use the manager web app that comes with Tomcat, we must add a login to our Tomcat server. We will do this by editing the `tomcat-users.xml` file:

```
$ sudo nano /opt/tomcat/conf/tomcat-users.xml
```
Copy

You will want to add a user who can access the `manager-gui` and `admin-gui` (web apps that come with Tomcat). You can do so by defining a user, similar to the example below, between the `tomcat-users` tags. Be sure to change the username and password to something secure:

tomcat-users.xml

```
<tomcat-users>
. . .
    <user username=" admin " password=" password " roles="manager-gui,admin-gui"/>
</tomcat-users>
```

Save and close the file when you are finished.

By de[illegible]newer versions of Tomcat restrict access to the Manager and Host Manager apps to connections coming from the server itself. Since we are installing on a remote

machine, you will probably want to remove or alter this restriction. To change the IP address restrictions on these, open the appropriate `context.xml` files.

For the Manager app, type:

```
$ sudo nano /opt/tomcat/webapps/manager/META-INF/context.xml
```
Copy

For the Host Manager app, type:

```
$ sudo nano /opt/tomcat/webapps/host-manager/META-INF/context.xml
```
Copy

Inside, comment out the IP address restriction to allow connections from anywhere. Alternatively, if you would like to allow access only to connections coming from your own IP address, you can add your public IP address to the list:

context.xml files for Tomcat webapps

```
<Context antiResourceLocking="false" privileged="true" >
  <!-- <Valve className="org.apache.catalina.valves.RemoteAddrValve"
         allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" /> -->
</Context>
```

Save and close the files when you are finished.

To put our changes into effect, restart the Tomcat service:

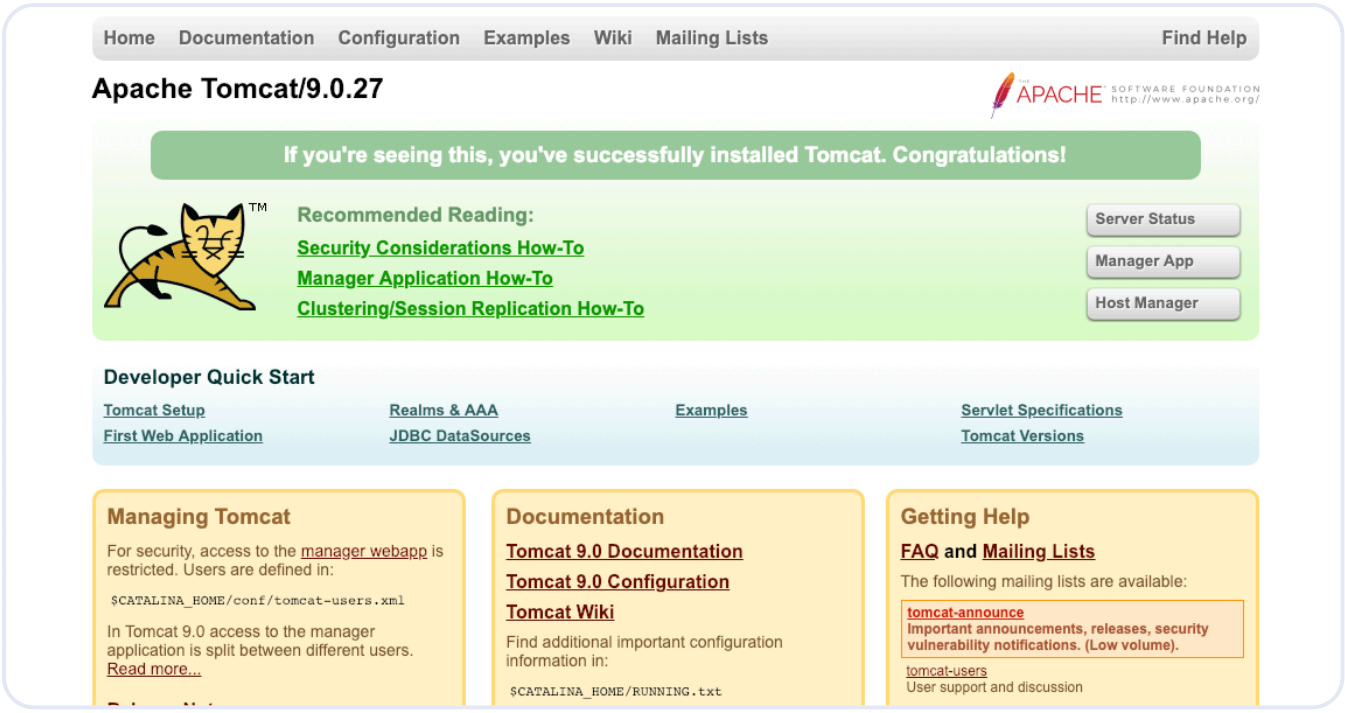
```
$ sudo systemctl restart tomcat
```
Copy

Step 8 – Access the Web Interface

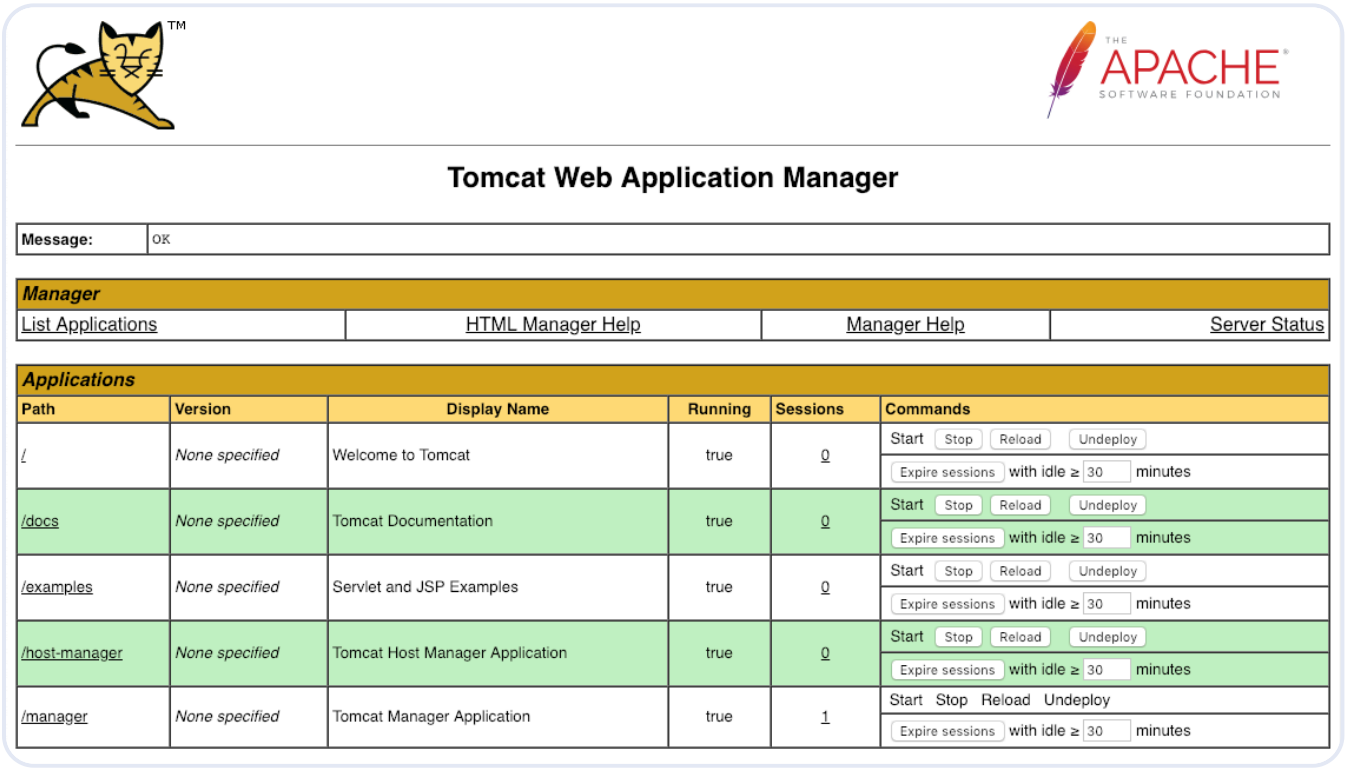
Now that we have create a user, we can access the web management interface again in a web browser. Once again, you can get to the correct interface by entering your server's domain name or IP address followed on port 8080 in your browser:

```
Open in web browser
http:// server_domain_or_IP :8080
```

The page you see should be the same one you were given when you tested earlier:

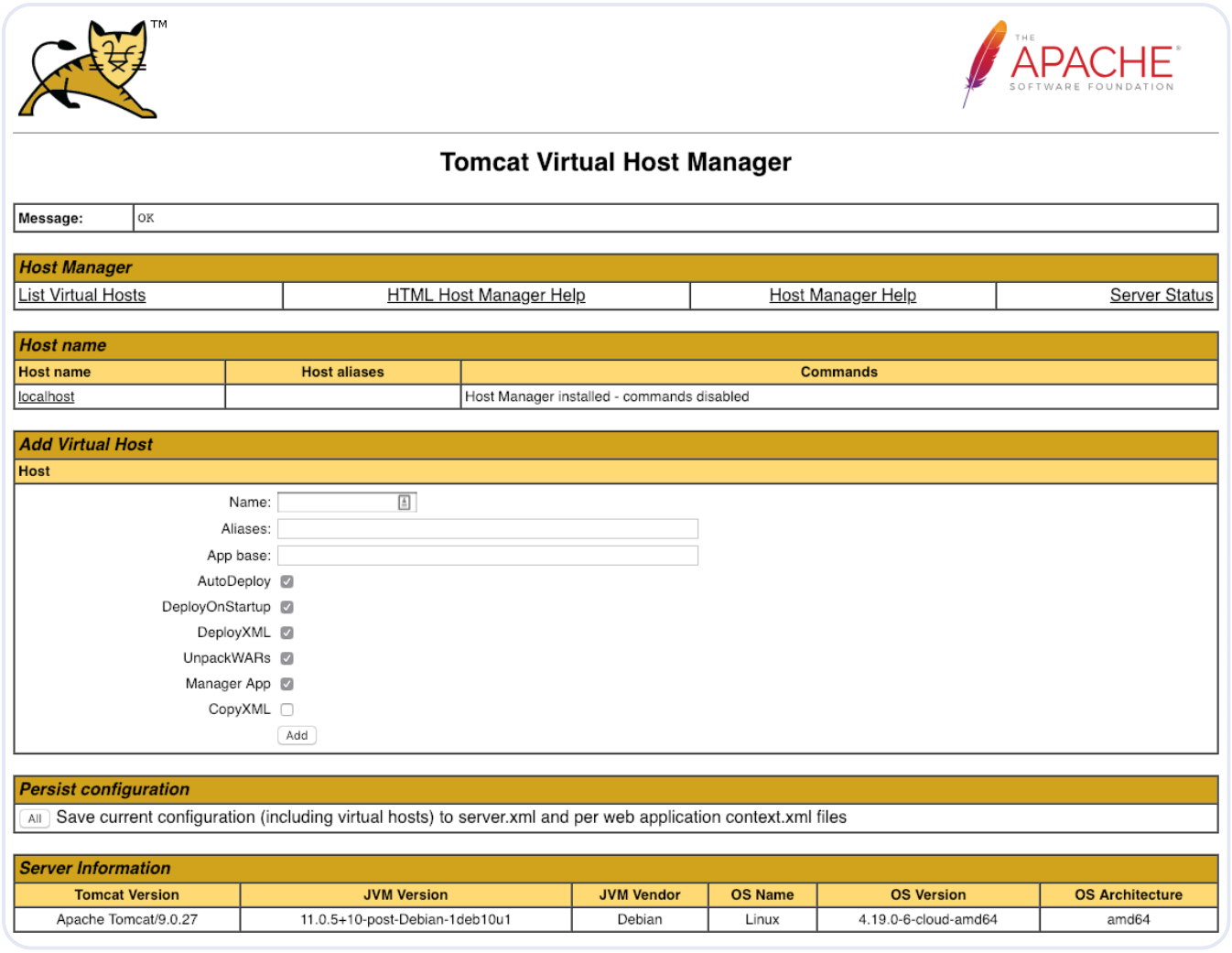


Let's take a look at the Manager App, accessible via the link or `http:// server_domain_or_IP :8080/manager/html`. You will need to enter the account credentials that you added to the `tomcat-users.xml` file. Afterwards, you should see a page that looks like this:



The Web Application Manager is used to manage your Java applications. You can Start, Stop, Reload, Deploy, and Undeploy here. You can also run some diagnostics on your apps (i.e. find memory leaks). Lastly, information about your server is available at the very bottom of this page.

Now let's take a look at the Host Manager, accessible via the link or `http:// server_domain_or_IP :8080/host-manager/html/`:



Tomcat Virtual Host Manager

Message:	OK

Host Manager

List Virtual Hosts	HTML Host Manager Help	Host Manager Help	Server Status

Host name

Host name	Host aliases	Commands
localhost		Host Manager installed - commands disabled

Add Virtual Host

Host

Name:
Aliases:
App base:
AutoDeploy
DeployOnStartup
DeployXML
UnpackWARs
Manager App
CopyXML
Add

Persist configuration

All Save current configuration (including virtual hosts) to server.xml and per web application context.xml files

Server Information

Tomcat Version	JVM Version	JVM Vendor	OS Name	OS Version	OS Architecture
Apache Tomcat/9.0.27	11.0.5+10-post-Debian-1deb10u1	Debian	Linux	4.19.0-6-cloud-amd64	amd64

From the Virtual Host Manager page, you can add virtual hosts to serve your applications from.

Conclusion

Your installation of Tomcat is complete! You are now free to deploy your own Java web applications.

Currently, your Tomcat installation is functional, but entirely unencrypted. This means that all data, including sensitive items like passwords, are sent in plain text that can be intercepted and read by other parties on the internet. In order to prevent this from happening, it is strongly recommended that you encrypt your connections with SSL. You can find out how to encrypt your connections to Tomcat by following this guide (*note: this guide covers Tomcat 8 encryption on Ubuntu 16.04*).

About the authors

Justin Ellingwood Author

Hanif Jetha Author

Still looking for an answer? Ask a question

Search for more help

Was this helpful? Yes No

Comments

7 Comments

Le comment ...

This textbox defaults to using **Markdown** to format your answer.

You can type **!ref** in this text area to quickly search our full set of tutorials, documentation & marketplace offerings and insert the link!

Sign In or Sign Up to Comment

Zezombye • April 3, 2020

An easier way is just `apt install tomcat9`.

Reply

timlg07 • April 24, 2024

The URL now returns a 301, the new URL would be `http://any23.apache.orgdist/tomcat/tomcat-9/v9.0.11/bin/apache-tomcat-9.0.11.tar.gz`

Reply

ZajaczkowskiMathias • September 26, 2023

Fantastic tutorial. Works perfectly to install Tomcat 9 on Debian 11 as well. Thanks a lot.

Reply

Sergey • June 1, 2023

After installing Tomcat 9 on Debian 11 there is no manager in /var/lib/tomcat9/webapps, where do I get it?

Reply

ficherconnor66 • September 23, 2021

works also with jdk16 Oracle on Tomcat10. I installed it on Debian11. Excellent tutorial.

Reply

Evert Wagenaar • January 10, 2020

It works. Unfortunately it doesn't start automatically at system (re)boot, something I would have expected from a service.

Reply

Takipci • November 20, 2019

That's what i need. Step by step installation. I am very happy, now.

Reply

Popular Topics

Ubu

Linux Basics

JavaScript

Python

MySQL

Docker

Kubernetes

All tutorials →

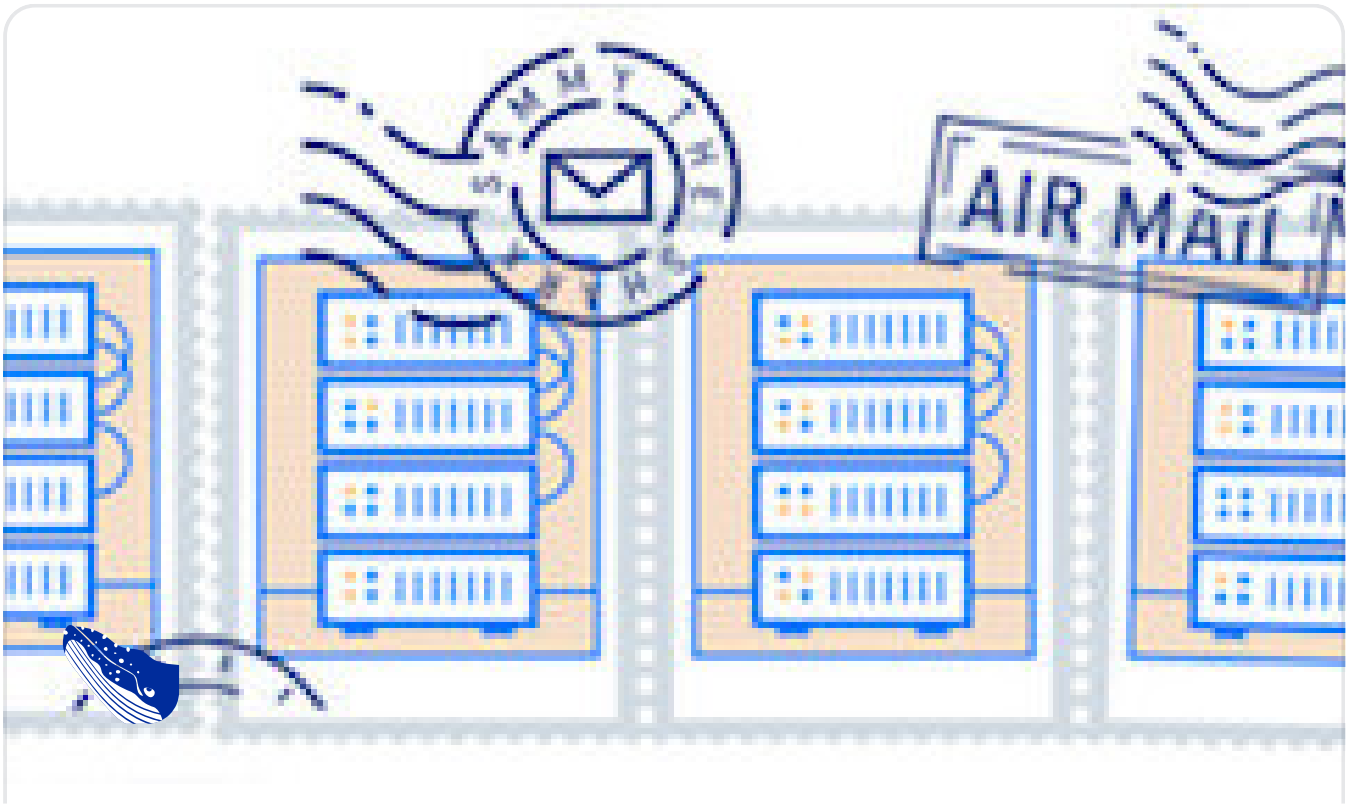
Talk to an expert →

Congratulations on unlocking the whale ambience easter egg! Click the whale button in the bottom left of your screen to toggle some ambient whale noises while you read.

Thank you to the Glacier Bay National Park & Preserve and Merrick079 for the sounds behind this easter egg.

Interested in whales, protecting them, and their connection to helping prevent climate change? We recommend checking out the Whale and Dolphin Conservation.

Reset easter egg to be discovered again / Permanently dismiss and hide easter egg



Get our biweekly newsletter

Sign up for Infrastructure as a Newsletter.

Sign up →

Hollie's Hub for Good

Working on improving health and education, reducing inequality, and spurring economic growth? We'd like to help.

Learn more →

Become a contributor

Get paid to write technical tutorials and select a tech-focused charity to receive a matching donation.

Learn more →

Featured Tutorials

Kubernetes Course Learn Python 3 Machine Learning in Python

Getting started with Go Intro to Kubernetes

DigitalOcean Products

Cloudways Virtual Machines Managed Databases Managed Kubernetes

Block Storage Object Storage Marketplace VPC Load Balancers