AL2 Tutorials

The following tutorials show you how to perform common tasks using Amazon EC2 instances running AL2. For video tutorials, see AWS Instructional videos and labs.

For AL2023 instructions, see Tutorials in the *AL2023 User Guide*.

Tutorials

Tutorial: Install a LAMP server on AL2

The following procedures help you install an Apache web server with PHP and MariaDB (a community-developed fork of MySQL) support on your AL2 instance (sometimes called a LAMP web server or LAMP stack). You can use this server to host a static website or deploy a dynamic PHP application that reads and writes information to a database.

Important

If you are trying to set up a LAMP web server on a different distribution, such as Ubuntu or Red Hat Enterprise Linux, this tutorial will not work. For AL2023, see Install a LAMP server on AL2023. For Ubuntu, see the following Ubuntu community documentation: ApacheMySQLPHP. For other distributions, see their specific documentation.

Option: Complete this tutorial using automation

To complete this tutorial using AWS Systems Manager Automation instead of the following tasks, run the AWSDocs-InstallALAMPServer-AL2 Automation document.

Tasks

- Troubleshoot
- Related topics

Step 1: Prepare the LAMP server

Prerequisites

- This tutorial assumes that you have already launched a new instance using AL2, with a public DNS name that is reachable from the internet. For more information, see Launch an instance in the *Amazon EC2 User Guide*. You must also have configured your security group to allow SSH (port 22), HTTP (port 80), and HTTPS (port 443) connections. For more information about these prerequisites, see Security group rules in the *Amazon EC2 User Guide*.
- The following procedure installs the latest PHP version available on AL2, currently `php8.2`. If you plan to use PHP applications other than those described in this tutorial, you should check their compatibility with `php8.2`.

To prepare the LAMP server

1. Connect to your instance.
2. To ensure that all of your software packages are up to date, perform a quick software update on your instance. This process may take a few minutes, but it is important to make sure that you have the latest security updates and bug fixes.

   The `-y` option installs the updates without asking for confirmation. If you would like to examine the updates before installing, you can omit this option.

   ```
   [ec2-user ~]$ sudo yum update -y
   ```
3. Install the `mariadb10.5` Amazon Linux Extras repositories to get the latest version of the MariaDB package.

   ```
   [ec2-user ~]$ sudo amazon-linux-extras install mariadb10.5
   ```

   If you receive an error stating `sudo: amazon-linux-extras: command not found`, then your instance was not launched with an Amazon Linux 2 AMI (perhaps you are using the Amazon Linux AMI instead). You can view your version of Amazon Linux using the following command.

```
cat /etc/system-release
```

4. Install the php8.2 Amazon Linux Extras repositories to get the latest version of the PHP package for AL2.

   ```
   [ec2-user ~]$ sudo amazon-linux-extras install php8.2
   ```

5. Now that your instance is current, you can install the Apache web server, MariaDB, and PHP software packages. Use the yum install command to install multiple software packages and all related dependencies at the same time

   ```
   [ec2-user ~]$ sudo yum install -y httpd
   ```

   You can view the current versions of these packages using the following command:

   ```
   yum info package_name
   ```

6. Start the Apache web server.

   ```
   [ec2-user ~]$ sudo systemctl start httpd
   ```

7. Use the **systemctl** command to configure the Apache web server to start at each system boot.

   ```
   [ec2-user ~]$ sudo systemctl enable httpd
   ```

   You can verify that **httpd** is on by running the following command:

   ```
   [ec2-user ~]$ sudo systemctl is-enabled httpd
   ```

8. Add a security rule to allow inbound HTTP (port 80) connections to your instance if you have not already done so. By default, a **launch-wizard-*N*** security group was set up for your instance during initialization. This group contains a single rule to allow SSH connections.

   a. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.

   b. Choose **Instances** and select your instance.

   c. On the **Security** tab, view the inbound rules. You should see the following rule:

      ```
      Port range   Protocol     Source
      ```

```
22           tcp          0.0.0.0/0
```

> **Warning**
>
> Using `0.0.0.0/0` allows all IPv4 addresses to access your instance using SSH. This is acceptable for a short time in a test environment, but it's unsafe for production environments. In production, you authorize only a specific IP address or range of addresses to access your instance.

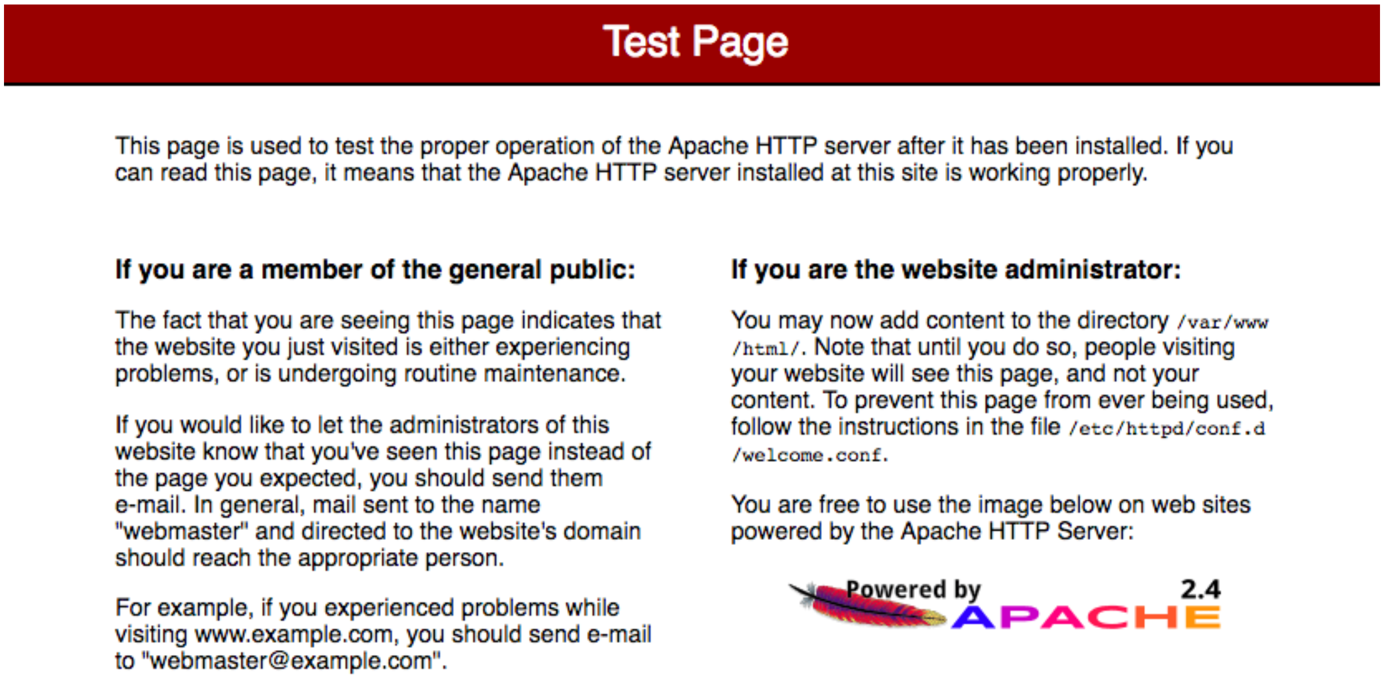
d. Choose the link for the security group. Using the procedures in Add rules to a security group, add a new inbound security rule with the following values:

- **Type**: HTTP
- **Protocol**: TCP
- **Port Range**: 80
- **Source**: Custom

9. Test your web server. In a web browser, type the public DNS address (or the public IP address) of your instance. If there is no content in `/var/www/html`, you should see the Apache test page. You can get the public DNS for your instance using the Amazon EC2 console (check the **Public DNS** column; if this column is hidden, choose **Show/Hide Columns** (the gear-shaped icon) and choose **Public DNS**).

   Verify that the security group for the instance contains a rule to allow HTTP traffic on port 80. For more information, see Add rules to security group.

> **Important**
>
> If you are not using Amazon Linux, you may also need to configure the firewall on your instance to allow these connections. For more information about how to configure the firewall, see the documentation for your specific distribution.



Test Page

This page is used to test the proper operation of the Apache HTTP server after it has been installed. If you can read this page, it means that the Apache HTTP server installed at this site is working properly.

If you are a member of the general public:

The fact that you are seeing this page indicates that the website you just visited is either experiencing problems, or is undergoing routine maintenance.

If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.

For example, if you experienced problems while visiting www.example.com, you should send e-mail to "webmaster@example.com".

If you are the website administrator:

You may now add content to the directory `/var/www/html/`. Note that until you do so, people visiting your website will see this page, and not your content. To prevent this page from ever being used, follow the instructions in the file `/etc/httpd/conf.d/welcome.conf`.

You are free to use the image below on web sites powered by the Apache HTTP Server:

Apache **httpd** serves files that are kept in a directory called the Apache document root. The Amazon Linux Apache document root is `/var/www/html`, which by default is owned by root.

To allow the `ec2-user` account to manipulate files in this directory, you must modify the ownership and permissions of the directory. There are many ways to accomplish this task. In this tutorial, you add `ec2-user` to the `apache` group, to give the `apache` group ownership of the `/var/www` directory and assign write permissions to the group.

To set file permissions

1. Add your user (in this case, `ec2-user`) to the `apache` group.

```
[ec2-user ~]$ sudo usermod -a -G apache ec2-user
```

2. Log out and then log back in again to pick up the new group, and then verify your membership.

   a. Log out (use the **exit** command or close the terminal window):

   ```
   [ec2-user ~]$ exit
   ```

   b. To verify your membership in the apache group, reconnect to your instance, and then run the following command:

      ```
      [ec2-user ~]$ groups
      ec2-user adm wheel apache systemd-journal
      ```

3. Change the group ownership of /var/www and its contents to the apache group.

   ```
   [ec2-user ~]$ sudo chown -R ec2-user:apache /var/www
   ```

4. To add group write permissions and to set the group ID on future subdirectories, change the directory permissions of /var/www and its subdirectories.

   ```
   [ec2-user ~]$ sudo chmod 2775 /var/www && find /var/www -type d -exec sudo chmod
    2775 {} \;
   ```

5. To add group write permissions, recursively change the file permissions of /var/www and its subdirectories:

   ```
   [ec2-user ~]$ find /var/www -type f -exec sudo chmod 0664 {} \;
   ```

Now, ec2-user (and any future members of the apache group) can add, delete, and edit files in the Apache document root, enabling you to add content, such as a static website or a PHP application.

To secure your web server (Optional)

A web server running the HTTP protocol provides no transport security for the data that it sends or receives. When you connect to an HTTP server using a web browser, the URLs that you visit, the content of webpages that you receive, and the contents (including passwords) of any HTML forms that you submit are all visible to eavesdroppers anywhere along the network pathway. The best practice for securing your web server is to install support for HTTPS (HTTP Secure), which protects your data with SSL/TLS encryption.

For information about enabling HTTPS on your server, see Tutorial: Configure SSL/TLS on AL2.

Step 2: Test your LAMP server

If your server is installed and running, and your file permissions are set correctly, your `ec2-user` account should be able to create a PHP file in the `/var/www/html` directory that is available from the internet.

To test your LAMP server

1. Create a PHP file in the Apache document root.

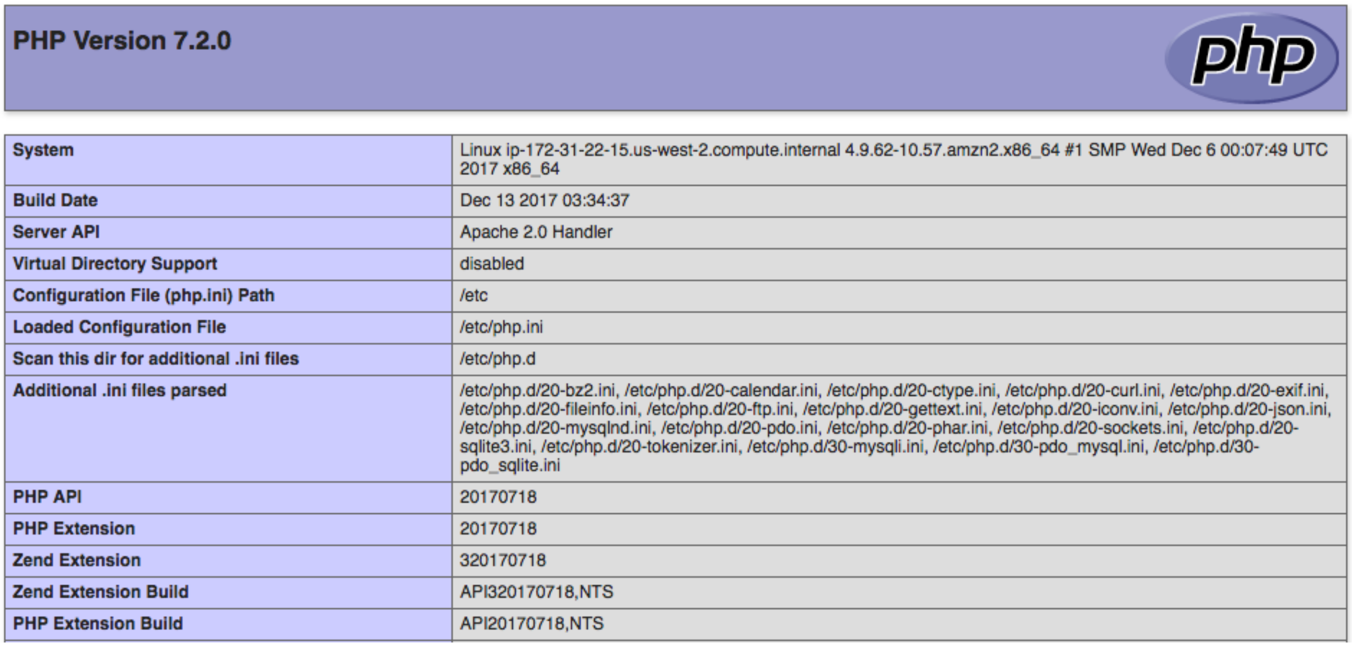
   ```
   [ec2-user ~]$ echo "<?php phpinfo(); ?>" > /var/www/html/phpinfo.php
   ```

   If you get a "Permission denied" error when trying to run this command, try logging out and logging back in again to pick up the proper group permissions that you configured in To set file permissions.

2. In a web browser, type the URL of the file that you just created. This URL is the public DNS address of your instance followed by a forward slash and the file name. For example:

   ```
   http://my.public.dns.amazonaws.com/phpinfo.php
   ```

   You should see the PHP information page:



PHP Version 7.2.0

System	Linux ip-172-31-22-15.us-west-2.compute.internal 4.9.62-10.57.amzn2.x86_64 #1 SMP Wed Dec 6 00:07:49 UTC 2017 x86_64
Build Date	Dec 13 2017 03:34:37
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc
Loaded Configuration File	/etc/php.ini
Scan this dir for additional .ini files	/etc/php.d
Additional .ini files parsed	/etc/php.d/20-bz2.ini, /etc/php.d/20-calendar.ini, /etc/php.d/20-ctype.ini, /etc/php.d/20-curl.ini, /etc/php.d/20-exif.ini, /etc/php.d/20-fileinfo.ini, /etc/php.d/20-ftp.ini, /etc/php.d/20-gettext.ini, /etc/php.d/20-iconv.ini, /etc/php.d/20-json.ini, /etc/php.d/20-mysqlnd.ini, /etc/php.d/20-pdo.ini, /etc/php.d/20-phar.ini, /etc/php.d/20-sockets.ini, /etc/php.d/20-sqlite3.ini, /etc/php.d/20-tokenizer.ini, /etc/php.d/30-mysqli.ini, /etc/php.d/30-pdo_mysql.ini, /etc/php.d/30-pdo_sqlite.ini
PHP API	20170718
PHP Extension	20170718
Zend Extension	320170718
Zend Extension Build	API320170718,NTS
PHP Extension Build	API20170718,NTS

If you do not see this page, verify that the `/var/www/html/phpinfo.php` file was created properly in the previous step. You can also verify that all of the required packages were installed with the following command.

```
[ec2-user ~]$ sudo yum list installed httpd mariadb-server php-mysqlnd
```

If any of the required packages are not listed in your output, install them with the **sudo yum install *package*** command. Also verify that the `php7.2` and `lamp-mariadb10.2-php7.2` extras are enabled in the output of the **amazon-linux-extras** command.

3. Delete the `phpinfo.php` file. Although this can be useful information, it should not be broadcast to the internet for security reasons.

```
[ec2-user ~]$ rm /var/www/html/phpinfo.php
```

You should now have a fully functional LAMP web server. If you add content to the Apache document root at `/var/www/html`, you should be able to view that content at the public DNS address for your instance.

Step 3: Secure the database server

The default installation of the MariaDB server has several features that are great for testing and development, but they should be disabled or removed for production servers. The **mysql_secure_installation** command walks you through the process of setting a root password and removing the insecure features from your installation. Even if you are not planning on using the MariaDB server, we recommend performing this procedure.

To secure the MariaDB server

1. Start the MariaDB server.

```
[ec2-user ~]$ sudo systemctl start mariadb
```

2. Run **mysql_secure_installation**.

```
[ec2-user ~]$ sudo mysql_secure_installation
```

   a. When prompted, type a password for the root account.

   i. Type the current root password. By default, the root account does not have a password set. Press Enter.

   ii. Type **Y** to set a password, and type a secure password twice. For more information about creating a secure password, see https://identitysafe.norton.com/password-generator/. Make sure to store this password in a safe place.

   Setting a root password for MariaDB is only the most basic measure for securing your database. When you build or install a database-driven application, you typically create a database service user for that application and avoid using the root account for anything but database administration.

b. Type **Y** to remove the anonymous user accounts.

c. Type **Y** to disable the remote root login.

d. Type **Y** to remove the test database.

e. Type **Y** to reload the privilege tables and save your changes.

3. (Optional) If you do not plan to use the MariaDB server right away, stop it. You can restart it when you need it again.

```
[ec2-user ~]$ sudo systemctl stop mariadb
```

4. (Optional) If you want the MariaDB server to start at every boot, type the following command.

```
[ec2-user ~]$ sudo systemctl enable mariadb
```

Step 4: (Optional) Install phpMyAdmin

phpMyAdmin is a web-based database management tool that you can use to view and edit the MySQL databases on your EC2 instance. Follow the steps below to install and configure `phpMyAdmin` on your Amazon Linux instance.

Important

We do not recommend using `phpMyAdmin` to access a LAMP server unless you have enabled SSL/TLS in Apache; otherwise, your database administrator password and other data are transmitted insecurely across the internet. For security recommendations from the developers, see Securing your phpMyAdmin installation. For general information about securing a web server on an EC2 instance, see Tutorial: Configure SSL/TLS on AL2.

To install phpMyAdmin

1. Install the required dependencies.

```
[ec2-user ~]$ sudo yum install php-mbstring php-xml -y
```

2. Restart Apache.

```
[ec2-user ~]$ sudo systemctl restart httpd
```

3. Restart `php-fpm`.

```
[ec2-user ~]$ sudo systemctl restart php-fpm
```

4. Navigate to the Apache document root at `/var/www/html`.

```
[ec2-user ~]$ cd /var/www/html
```

5. Select a source package for the latest phpMyAdmin release from https://www.phpmyadmin.net/downloads. To download the file directly to your instance, copy the link and paste it into a **wget** command, as in this example:

```
[ec2-user html]$ wget https://www.phpmyadmin.net/downloads/phpMyAdmin-latest-all-
languages.tar.gz
```

6. Create a `phpMyAdmin` folder and extract the package into it with the following command.

```
[ec2-user html]$ mkdir phpMyAdmin && tar -xvzf phpMyAdmin-latest-all-
languages.tar.gz -C phpMyAdmin --strip-components 1
```

7. Delete the *`phpMyAdmin-latest-all-languages.tar.gz`* tarball.

```
[ec2-user html]$ rm phpMyAdmin-latest-all-languages.tar.gz
```

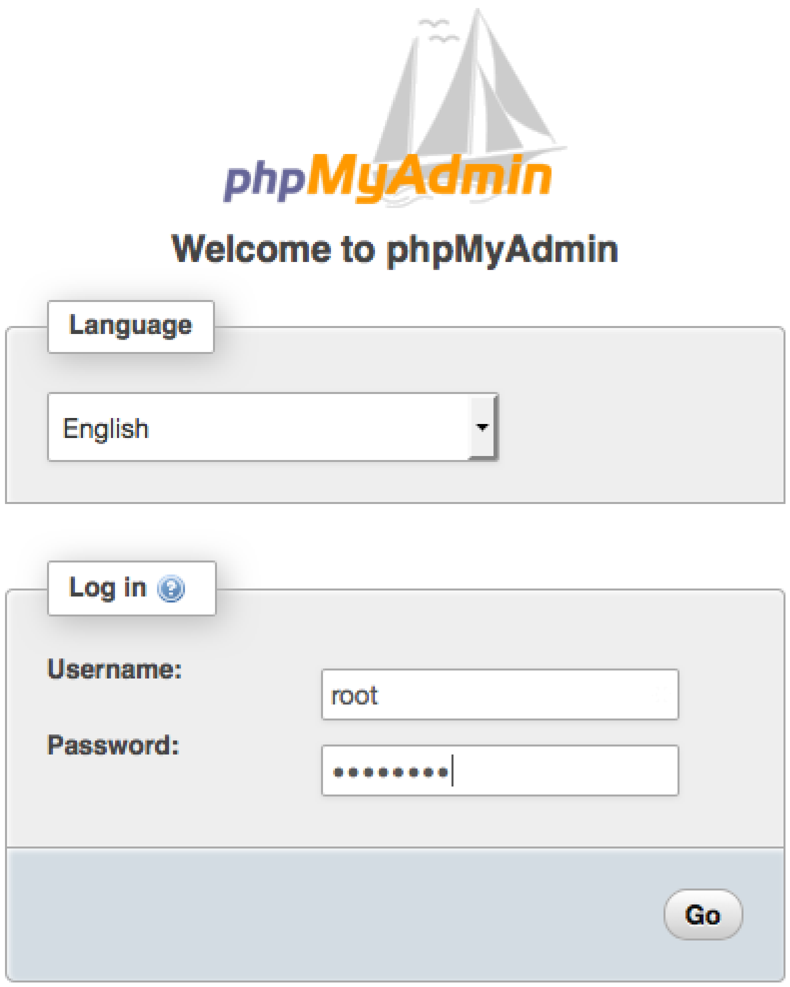
8. (Optional) If the MySQL server is not running, start it now.

```
[ec2-user ~]$ sudo systemctl start mariadb
```

9. In a web browser, type the URL of your phpMyAdmin installation. This URL is the public DNS address (or the public IP address) of your instance followed by a forward slash and the name of your installation directory. For example:

```
http://my.public.dns.amazonaws.com/phpMyAdmin
```

You should see the phpMyAdmin login page:



Language: English

Username: root

Password: •••••••••

10. Log in to your phpMyAdmin installation with the `root` user name and the MySQL root password you created earlier.

Your installation must still be configured before you put it into service. We suggest that you begin by manually creating the configuration file, as follows:

a. To start with a minimal configuration file, use your favorite text editor to create a new file, and then copy the contents of `config.sample.inc.php` into it.

b. Save the file as `config.inc.php` in the phpMyAdmin directory that contains `index.php`.

c. Refer to post-file creation instructions in the Using the Setup script section of the phpMyAdmin installation instructions for any additional setup.

For information about using phpMyAdmin, see the phpMyAdmin User Guide.

Troubleshoot

This section offers suggestions for resolving common problems you may encounter while setting up a new LAMP server.

I can't connect to my server using a web browser

Perform the following checks to see if your Apache web server is running and accessible.

- **Is the web server running?**

  You can verify that **httpd** is on by running the following command:

  ```
  [ec2-user ~]$ sudo systemctl is-enabled httpd
  ```

  If the **httpd** process is not running, repeat the steps described in To prepare the LAMP server.

- **Is the firewall correctly configured?**

  Verify that the security group for the instance contains a rule to allow HTTP traffic on port 80. For more information, see Add rules to security group.

I can't connect to my server using HTTPS

Perform the following checks to see if your Apache web server is configured to support HTTPS.

- **Is the web server correctly configured?**

After you install Apache, the server is configured for HTTP traffic. To support HTTPS, enable TLS on the server and install an SSL certificate. For information, see Tutorial: Configure SSL/TLS on AL2.

- **Is the firewall correctly configured?**

  Verify that the security group for the instance contains a rule to allow HTTPS traffic on port 443. For more information, see Add rules to a security group.

Related topics

For more information about transferring files to your instance or installing a WordPress blog on your web server, see the following documentation:

- Transfer files to your Linux instance using WinSCP.
- Transfer files to Linux instances using an SCP client.
- Tutorial: Host a WordPress blog on AL2

For more information about the commands and software used in this tutorial, see the following webpages:

- Apache web server: http://httpd.apache.org/
- MariaDB database server: https://mariadb.org/
- PHP programming language: http://php.net/
- The `chmod` command: https://en.wikipedia.org/wiki/Chmod
- The `chown` command: https://en.wikipedia.org/wiki/Chown

For more information about registering a domain name for your web server, or transferring an existing domain name to this host, see Creating and Migrating Domains and Subdomains to Amazon Route 53 in the *Amazon Route 53 Developer Guide*.

Tutorial: Configure SSL/TLS on AL2

Secure Sockets Layer/Transport Layer Security (SSL/TLS) creates an encrypted channel between a web server and web client that protects data in transit from being eavesdropped on. This tutorial explains how to add support manually for SSL/TLS on an EC2 instance with AL2 and Apache web