Run an application with a dedicated systemd service account

A dedicated service account limits which files an application can modify. It also makes ownership and process inspection easier. This guide is for an administrator using systemd on Debian or Ubuntu. It assumes you already have a working application and know its supported startup command. Do not move a database or change its runtime identity without its vendor's migration instructions.

Separate code, configuration and data

List the executable, configuration, logs and writable data directories. Identify any privileged operation such as opening a low-numbered port or accessing a device. A service account cannot solve those requirements by itself. Prefer the application's supported deployment model, and keep the current service working while preparing the change.

Choose an unused account name and create a non-login system account through your distribution's normal account tools. Do not reuse a customer SSH login or give the account sudo access. Record its UID and group when mounted storage or containers depend on numeric ownership. Dynamic users are useful for some applications but are not a drop-in replacement for established persistent ownership.

Grant only required filesystem access

Keep program code and private administrative files outside directories writable by the runtime account. Give the account read access to the configuration it needs and write access only to its

intended state directories. Review all parent-directory traversal permissions. Avoid recursively changing ownership across a shared application tree or mounted filesystem.

Inspect the existing unit and any drop-ins before editing:

```
systemctl cat example.service
systemctl show example.service -p User -p Group -p WorkingDirectory
```

Save the current configuration privately. Unit files should not contain passwords or tokens in command arguments. Follow the application's supported secret-handling method and restrict any environment or credential files.

Define the runtime explicitly

Use a drop-in created with `sudo systemctl edit example.service` where appropriate. The following is a conceptual fragment, not a complete unit to paste without adapting paths:

```
[Service]
User=example
Group=example
WorkingDirectory=/srv/example
```

The existing ExecStart must remain valid for that account and directory. Changing User does not automatically fix every file permission. Review service sandbox settings and the installed `systemd.exec` manual. Additional hardening should be tested one option at a time because it can block legitimate file, network or device access.

Switch during a controlled window

Validate the unit using the local systemd tools, then reload the manager configuration. A running process does not adopt a new identity until the service is restarted. Schedule that restart, retain console access and have the previous drop-in available for recovery.

Afterward, inspect the process owner, service status and fresh logs. Exercise a normal application write, restart persistence and any scheduled job that shares its data. If access fails, restore the previous unit configuration and the specific ownership changes from your record. Do not compensate by granting the account root privileges. The successful result is a functioning application with a documented, limited runtime identity.

Related instructions

Investigate systemd services. Connect using SSH.

Reference: Official documentation. Check the documentation installed on your own operating-system release when an option differs.

Revision #1
Created 2026-11-09 10:42:43 UTC by CloudExa Hosting Admin
Updated 2026-11-09 10:42:43 UTC by CloudExa Hosting Admin