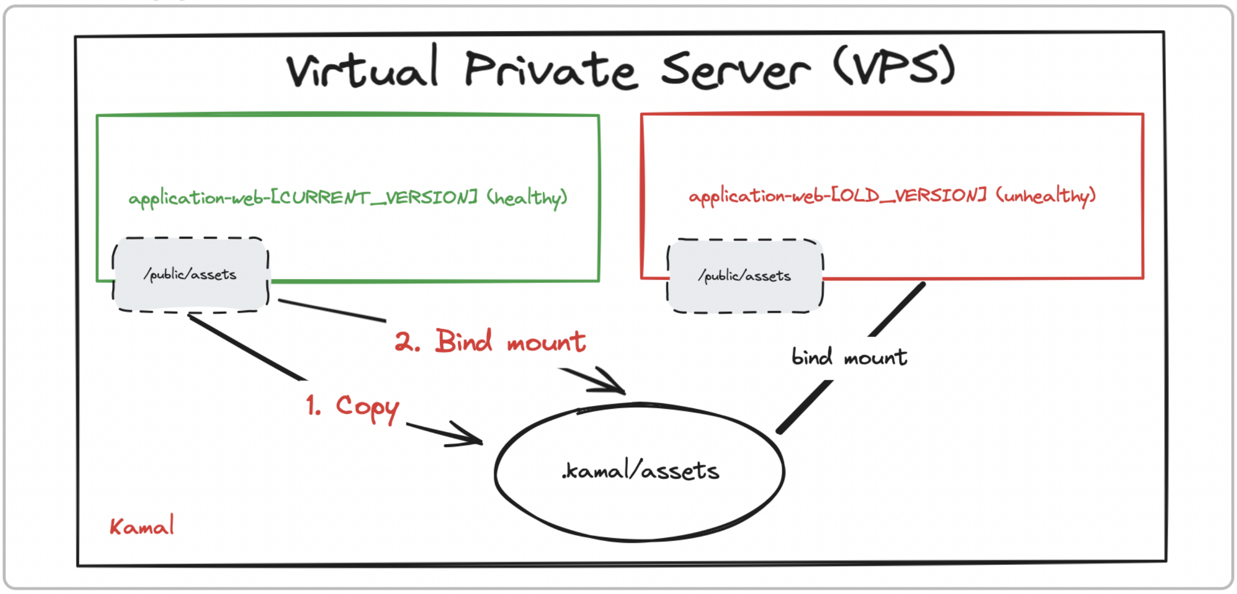
Chapter 11. Assets

Assets such as CSS and JavaScript files are served automatically from your container image as you would expect. However, we have seen that Kamal might run two versions of the application at the same time.

What if a front-end request comes from the newer version but ends up being served by the old container? To prevent this kind of `404` error Kamal supports asset bridging:

Kamal assets bridging


Virtual Private Server (VPS)
application-web-[CURRENT_VERSION] (healthy)
application-web-[OLD_VERSION] (unhealthy)
/public/assets
/public/assets
2. Bind mount
bind mount
1. Copy
.kamal/assets
Kamal

Kamal will replace the assets path in both containers specified by the `asset_path` configuration with a mapped volume containing both sets of files. It will create a `.kamal/assets` directory, extract the files there, and bind mounts a new volume directory from there automatically.

To configure the path in the container image, set the path to the assets:

```yaml
# config/deploy.yml
...
asset_path: /rails/public/assets
```

Asset bridging requires that file names change when the contents change (e.g. by including a hash of the contents in the name).

IMPORTANT	Even if it looks like a Kamal's default, it's not! If you leave this uncommented, Kamal won't bridge assets for you. This is because Kamal is framework-independent and cannot know where your assets are or if you want to serve them at all.