Instalação do Docker

O **Docker** é uma plataforma que permite rodar aplicações dentro de "contêineres" — ambientes isolados e padronizados que funcionam de forma independente do sistema operacional. É através dele que o sistema será executado no servidor.

> **Pré-requisito:** Você deve estar logado como administrador (`sudo su`). Se ainda não fez isso, execute `sudo su` antes de continuar.

> ⚠ **Limite de memória obrigatório.** A imagem do sistema traz a mitigação das falhas de segurança do PHP **CVE-2026-109618** e **CVE-2026-109619**. Por causa dela, o PHP roda sem teto de memória e quem limita o consumo é o contêiner. **Sem limite de memória, o contêiner não sobe.** O script pergunta esse valor no passo 5.

1. Abrindo o Terminal como Administrador

```
sudo su
```

2. Instalar o Docker e o Docker Compose

O comando abaixo instala dois componentes:

```
apt install docker.io -y && \
apt install docker-compose
```

O que cada um faz:

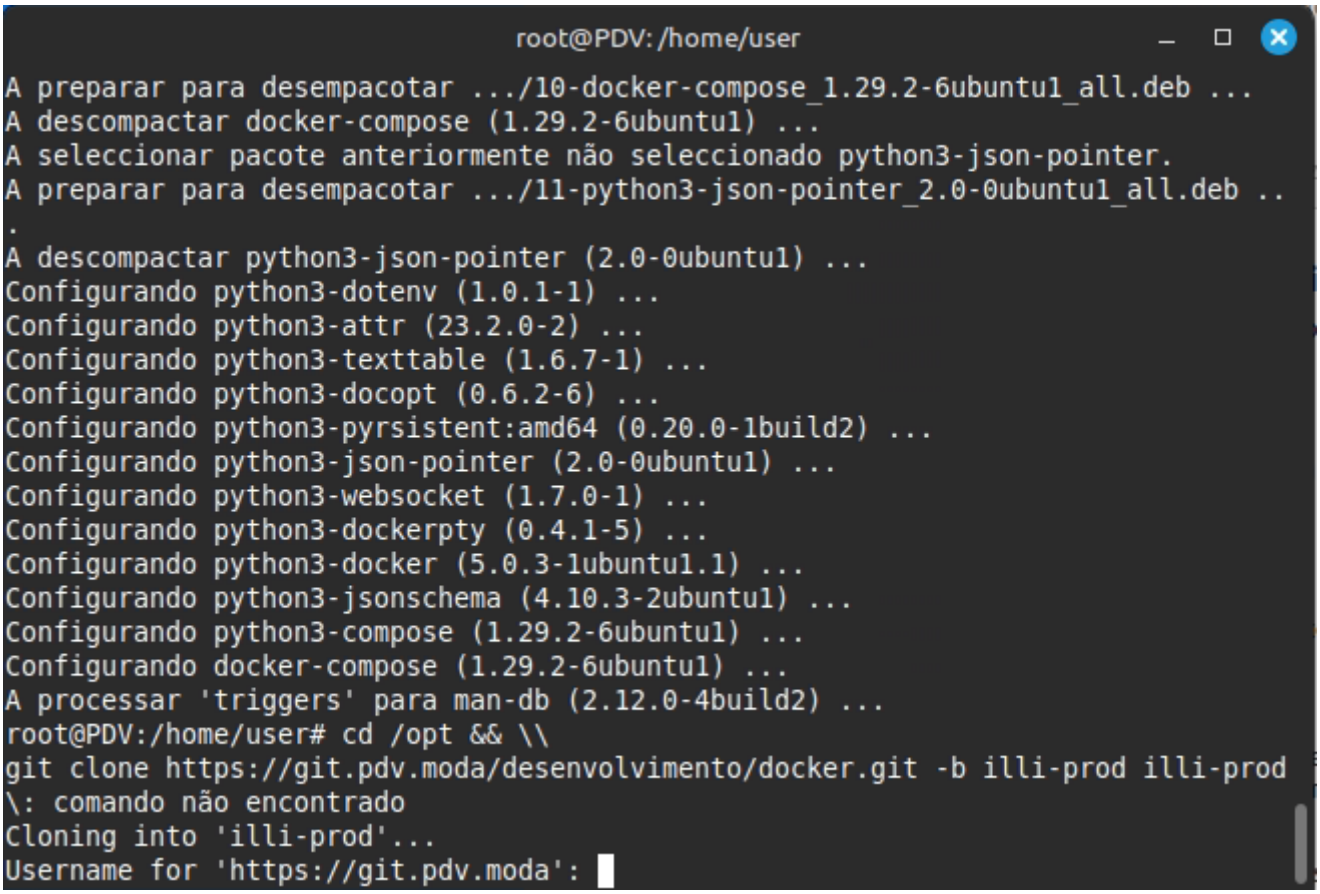
- **docker.io** — o Docker em si, responsável por criar e executar os contêineres
- **docker-compose** — ferramenta que permite iniciar e gerenciar vários contêineres de uma só vez usando um arquivo de configuração

3. Clonar o Repositório de Instalação

Agora vamos baixar os arquivos necessários para configurar o ambiente. Acesse a pasta `/opt` e faça o clone do repositório:

```
cd /opt && \
git clone https://git.pdv.moda/desenvolvimento/docker.git -b illi-prod illi-prod
```

Quando solicitado, digite seu **usuário** e **senha** de acesso ao `git.pdv.moda` e pressione **Enter** após cada um.



> **O que esse comando faz:** Baixa os arquivos de configuração do ambiente Docker a partir do repositório oficial da Illimitar, salvando-os na pasta `illi-prod` dentro de `/opt`.

4. Acessar a Pasta Clonada

Após o clone ser concluído, entre na pasta que foi baixada:

```
cd illi-prod
```

Dentro dessa pasta estará o arquivo `illi.sh`, que é o script responsável por instalar e configurar o ambiente.

5. Executar o Script de Instalação

Execute o script com o comando abaixo:

```
bash illi.sh
```

O script fará algumas perguntas para configurar a instância. Responda conforme indicado:

- **Nome da instância:** `illi`
- **Porta do host [8088]:** pressione **Enter** para manter a porta padrão `8088`
- **Limite de memória do container:** pressione **Enter** para aceitar o valor sugerido pelo script (tabela abaixo)

> **O que é uma instância?** É o nome que identifica esta instalação do sistema no servidor. O nome `illi` é o padrão recomendado.

> **O que é a porta 8088?** É o número pelo qual o sistema será acessado no navegador. Por exemplo: `http://IP-do-servidor:8088`. Mantenha o padrão a menos que essa porta já esteja em uso.

>

> **O que é o limite de memória?** É o máximo de memória RAM que o contêiner do sistema pode usar. É obrigatório: sem ele, o contêiner não sobe. O script mostra a RAM do servidor, sugere o valor e não aceita um limite igual ou maior que a RAM.

Quando o MySQL roda no mesmo servidor, o sistema (web) fica com **até 30% da RAM** e o banco com o restante (**70% ou mais**). O script já sugere o valor certo para o servidor:

RAM do servidor	Limite do contêiner (web)	Fica para o banco
8 GB	`2g`	6 GB
16 GB	`4g`	12 GB
32 GB	`8g`	24 GB
64 GB	`16g`	48 GB

Acima de 64 GB, a regra continua: o limite dobra a cada vez que a RAM dobra (128 GB → `32g`).

> A configuração de memória do MySQL (`innodb_buffer_pool_size` e demais) deve caber na parte do banco. Veja a página **Instalação do Percona MySQL**.

O valor fica gravado no `.env` da instância (`ILLI_MEM_LIMIT`). Para alterar depois, use as opções **16) Configurar** e **13) Reconstruir container** do menu.

6. Confirmar a Inicialização do Contêiner

Após configurar a instância, o script fará o clone do repositório standalone automaticamente. Ao final, ele perguntará se deseja subir o contêiner:

```
  Nome da instância [illi-web]: illi
  Porta do host [8088]: 8088
  Limite de memória do container: obrigatório (mitigação CVE-2026-109618).
  RAM do servidor: 15890 MB. Com MySQL no mesmo servidor: até 30% para o web (4g), o resto
para o banco.
  Limite de memória do container [4g]:
  Clonando repositório standalone em www/illi...
```

```
Cloning into '/opt/illi-prod/illi/www/illi'...
  ...
  ✔  Repositório clonado em illi/www/illi
  ✔  Host: chmod +x *.sh  |  chown www-data:www-data
  ──────────────────────────────────────

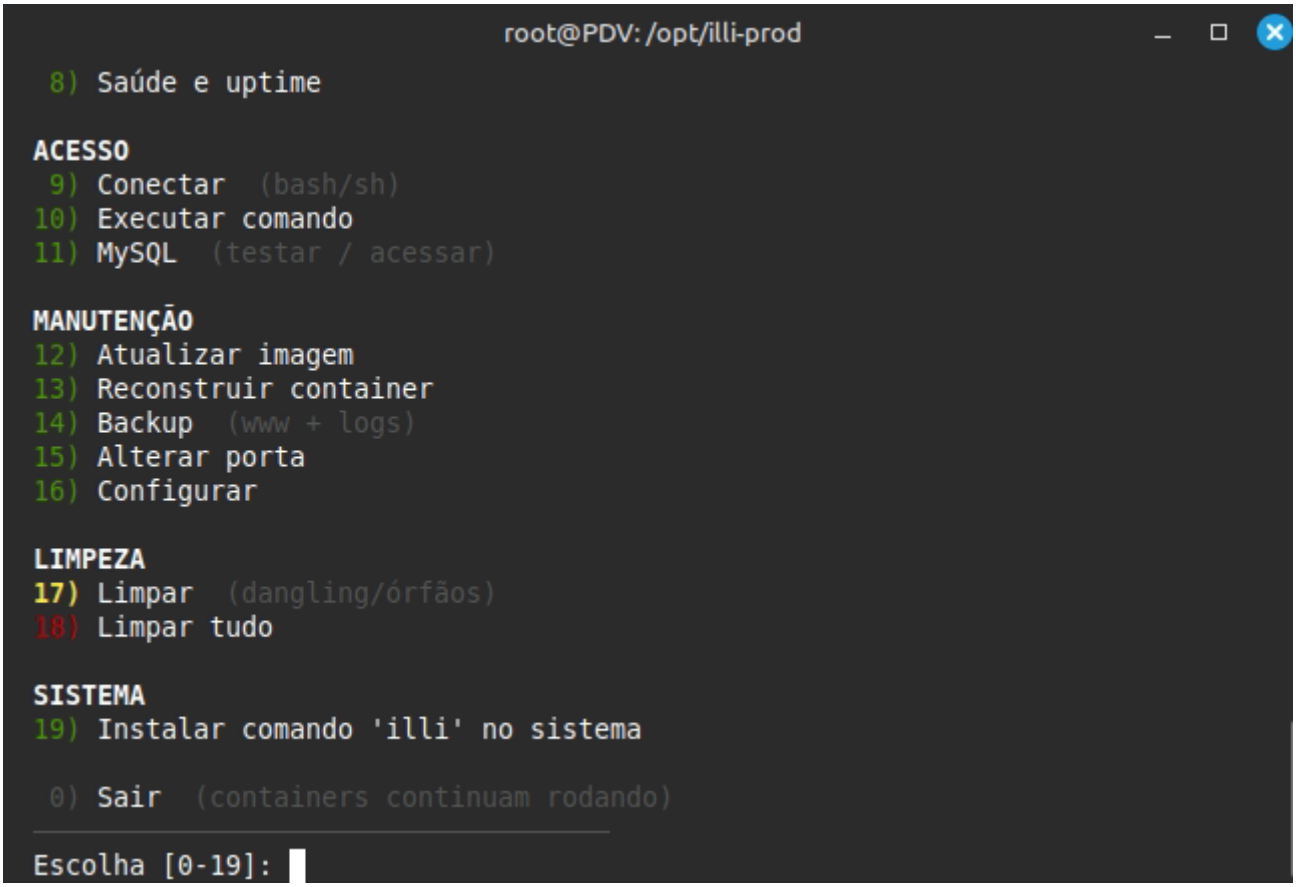
  ✔  Instância criada em: /opt/illi-prod/illi
    Container : illi
    Porta     : 8088
    Memória   : 4g
  ──────────────────────────────────────

  Subir o container agora? [S/n]:
```

Responda **S** e pressione **Enter** para iniciar o contêiner.

O Docker irá baixar as imagens necessárias e iniciar todos os serviços. Esse processo pode levar alguns minutos dependendo da velocidade da conexão.

Digite **0** para sair do utilitário:



7. Conferir o Contêiner

```
docker ps
docker inspect illi --format '{{.HostConfig.Memory}}'
```

- Em `docker ps`, o **STATUS** deve ser `Up`. Se aparecer `Restarting`, veja **Problemas comuns** abaixo.
- O `docker inspect` deve mostrar um número diferente de `0` (o limite em bytes). Para `4g`, aparece `4294967296`.

Instalação Concluída

O Docker está instalado e o contêiner do sistema está em execução. O ambiente estará acessível pelo navegador no endereço `http://IP-do-servidor:8088`.

Problemas comuns

Contêiner reiniciando sem parar (`Restarting`)

Ao rodar `docker logs illi`, aparece:

```
[erro] container sem limite de memória (exigido pela mitigação CVE-2026-109618).
```

Causa: a instância foi criada sem limite de memória, com uma versão do `illi.sh` anterior à mitigação.

Solução:

```
cd /opt/illi-prod && git pull
bash illi.sh
```

1. Escolha a instância.
2. Escolha **13) Reconstruir container** e confirme.
3. Quando o script pedir o limite de memória, pressione **Enter** para aceitar o valor sugerido (tabela do passo 5).

O script atualiza os arquivos da instância, guarda o `docker-compose.yml` antigo como backup (`docker-compose.yml.bak-<data>`) e recria o contêiner com o limite. Depois, confira com o passo 7.

Próximo Passo

Continue para: **Configuração da Interface**

Revision #7
Created 2026-05-15 13:39:12 UTC by Daniel
Updated 2026-10-27 20:20:10 UTC by Heron Santos