

GUIA NODE RITUAL

Neste guia, mostraremos como rodar um nó (ou node) do projeto [Ritual](#). A documentação para isso pode ser encontrada [aqui](#).

Os requerimentos para rodar em seu próprio computador são esses abaixo:

	Minimum	Recommended	GPU-heavy workloads
CPU	Single-core vCPU	4 modern vCPU cores	4 modern vCPU cores
RAM	128MB	16GB	64GB
DISK	512MB HDD	500GB IOPS-optimized SSD	500GB NVME
GPU			CUDA-enabled GPU

Se for rodar em seu computador, lembre que ele tem que ficar 100% do tempo online.

O que faremos aqui será alugar um [VPS](#), e rodar o node nesse servidor. Assim garantimos os requerimentos, e que o servidor vai estar online.

O custo é por volta de U\$ 15 por mês.

Também é necessário um pouco de ETH na rede Base, para pagar taxas. Por volta de 0.005 ETH é mais do que suficiente.

Observação útil: na hora de digitar os comandos no terminal, o "CTRL + V" não funciona. Mas clicar com o botão direito do mouse e colar funciona.

1.A) Configurando seu Servidor

Iremos usar um VPS da [Contabo](#). O CLOUD VPS 2 atende nossas necessidades:

Cloud VPS 1	Cloud VPS 2	Cloud VPS 3	Cloud VPS 4	Cloud VPS 5	Cloud VPS 6
\$5⁵⁰ / month	\$12⁵⁰ / month	\$17⁵⁰ / month	\$32⁵⁰ / month	\$42⁰⁰ / month	\$77⁰⁰ / month
4 vCPU Cores	6 vCPU Cores	8 vCPU Cores	12 vCPU Cores	16 vCPU Cores	24 vCPU Cores
6 GB RAM	16 GB RAM	24 GB RAM	48 GB RAM	64 GB RAM	120 GB RAM
100 GB NVMe or 400 GB SSD	200 GB NVMe or 400 GB SSD	300 GB NVMe or 1.2 TB SSD	400 GB NVMe or 1.6 TB SSD	500 GB NVMe or 2 TB SSD	600 GB NVMe or 2.4 TB SSD
1 Snapshot	2 Snapshots	2 Snapshots	3 Snapshots	3 Snapshots	3 Snapshots
32 TB Traffic* Unlimited Incoming	32 TB Traffic* Unlimited Incoming	32 TB Traffic* Unlimited Incoming	32 TB Traffic* Unlimited Incoming	32 TB Traffic* Unlimited Incoming	32 TB Traffic* Unlimited Incoming
Select	Select	Select	Select	Select	Select

Our Recommendation

Selecione o tempo que quer alugar. Pode colocar 1 mês e renovar se necessário, ou já alugar por um período maior como 3 meses.

Cloud VPS 2

Monthly Base Price \$12.50

CPU	RAM	STORAGE	SNAPSHOT
6 vCPU Cores	16 GB RAM	200 GB NVMe or 400 GB SSD	2 Snapshots

1. Select your term length

1 Month

Deixe a região já pré-selecionada.

2. Region

☒ European Union (Germany)	 Good Latency	
☐ United Kingdom	 Good Latency	\$2.65
☐ United States (Central)	 Good Latency 144 ms	\$2.65
☐ United States (East)	 Good Latency	\$3.60
☐ United States (West)	 Good Latency 120 ms	\$3.15
☐ Asia (Singapore)	 Good Latency	\$6.00
☐ Asia (Japan)	 Good Latency	\$6.00
☐ Australia (Sydney)	 Good Latency	\$5.15

Escolha a opção de 600 GB SSD, conforme as recomendações da Ritual.

3. Storage Type

You can now choose either NVMe or SSD for your VPS Instance.

☐ 400 GB SSD	Free
☒ 600 GB SSD	\$2.80
☐ 200 GB NVMe	Free
☐ 300 GB NVMe	\$3.50

Em “Image”, selecione “Apps & Panels”, “Docker” e “Ubuntu 22.04”.

4. Image

Popular OS **Apps & Panels** Blockchain



cPanel

cPanel/WHM (5 accounts) ▼

Ubuntu 20.04

\$22.99



Plesk + Windows

Plesk Obsidian Web Admin Edition 1 month free ▼

Windows Server 2019 Datacenter ▼

\$31.99



Plesk + Linux

Plesk Obsidian Web Admin Edition 1 month free ▼

Ubuntu 20.04 ▼

\$13.99



Webmin + LAMP

Ubuntu 20.04 ▼

Included



Docker

Ubuntu 22.04 ▼

Included



LAMP

Ubuntu 20.04 ▼

Included



Webmin

Ubuntu 20.04 ▼

Included

Escolha uma senha.

5. Login & password for your server

Username root

[Generate new password](#)

Password

xrYMcN7s7baNmgCq0EEGT7uXI0Snf 

In order to use SSH Keys you can add them in the Customer Control Panel later.

O resto das configurações podem ser mantidas como estão.

6. Object Storage [?]

S3 Compatible Object Storage

None

7. Networking

Private
Networking

[?]

No Private Networking

Bandwidth

[?]

32 TB Out + Unlimited In
400 Mbit/s Connection

IPv4

1 IP Address

Free

8. Add-Ons

Backup
Space

[?]

None

Server
Management

[?]

Unmanaged

Free

Monitoring

[?]

None

SSL

[?]

None

Server Quantity

1

Next

Clique em "Next", crie sua conta, e faça o pagamento.

Você vai receber um email inicial, e depois de um tempo um outro email com as informações que você precisa para acessar seu servidor (como o IP).

O próximo passo é baixar o PuTTY, um software gratuito para emular seu terminal. Faça o download [aqui](#).

Package files

You probably want one of these. They include versions of all the PuTTY utilities (except the new and slightly experimental Windows pterm).

(Not sure whether you want the 32-bit or the 64-bit version? Read the [FAQ entry](#).)

We also publish the latest PuTTY installers for all Windows architectures as a free-of-charge download at the [Microsoft Store](#); they usually take a few days to appear there after we release them.

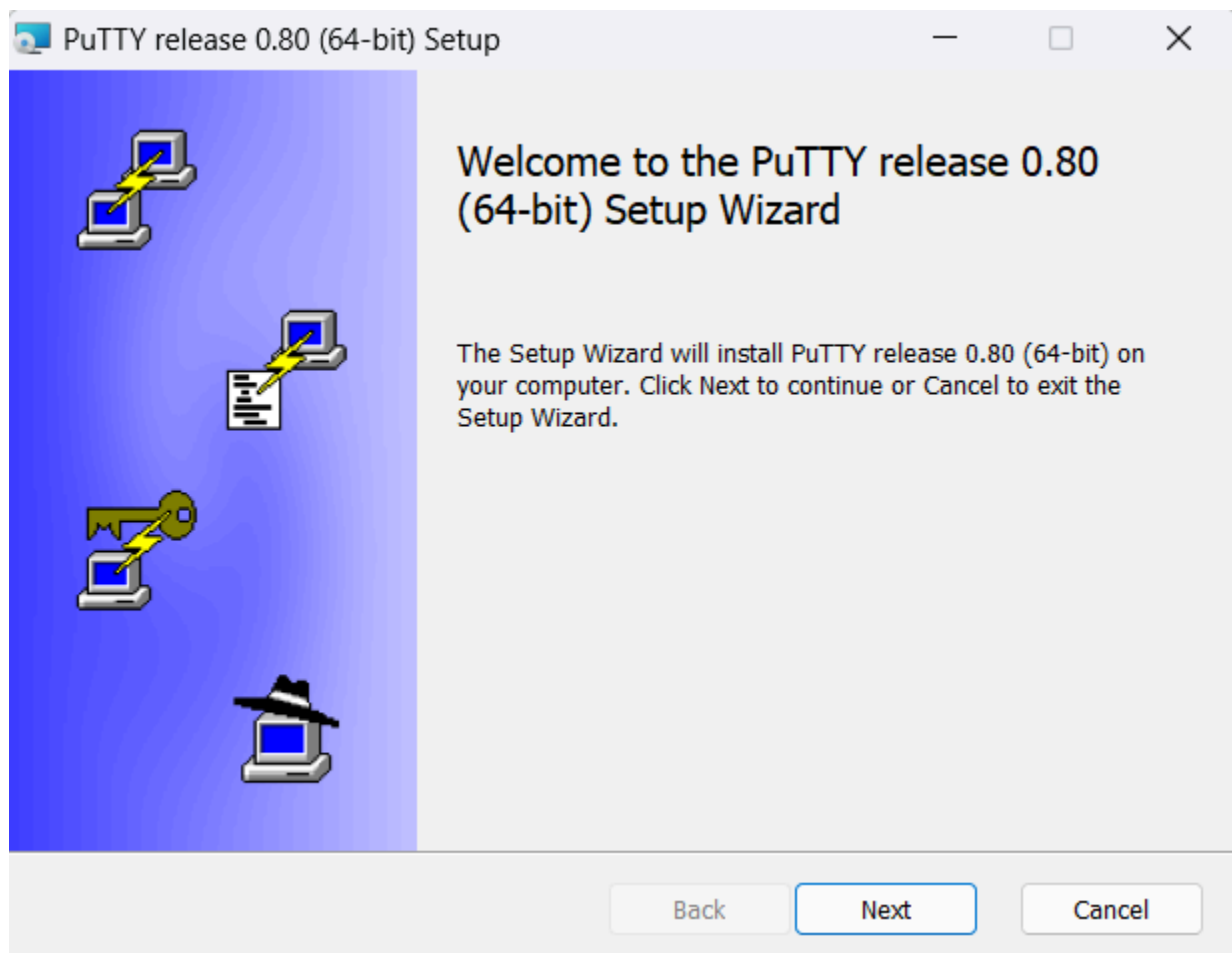
MSI ("Windows Installer")

64-bit x86:	putty-64bit-0.80-installer.msi	(signature)
64-bit Arm:	putty-arm64-0.80-installer.msi	(signature)
32-bit x86:	putty-0.80-installer.msi	(signature)

Unix source archive

.tar.gz:	putty-0.80.tar.gz	(signature)
----------	-----------------------------------	-----------------------------

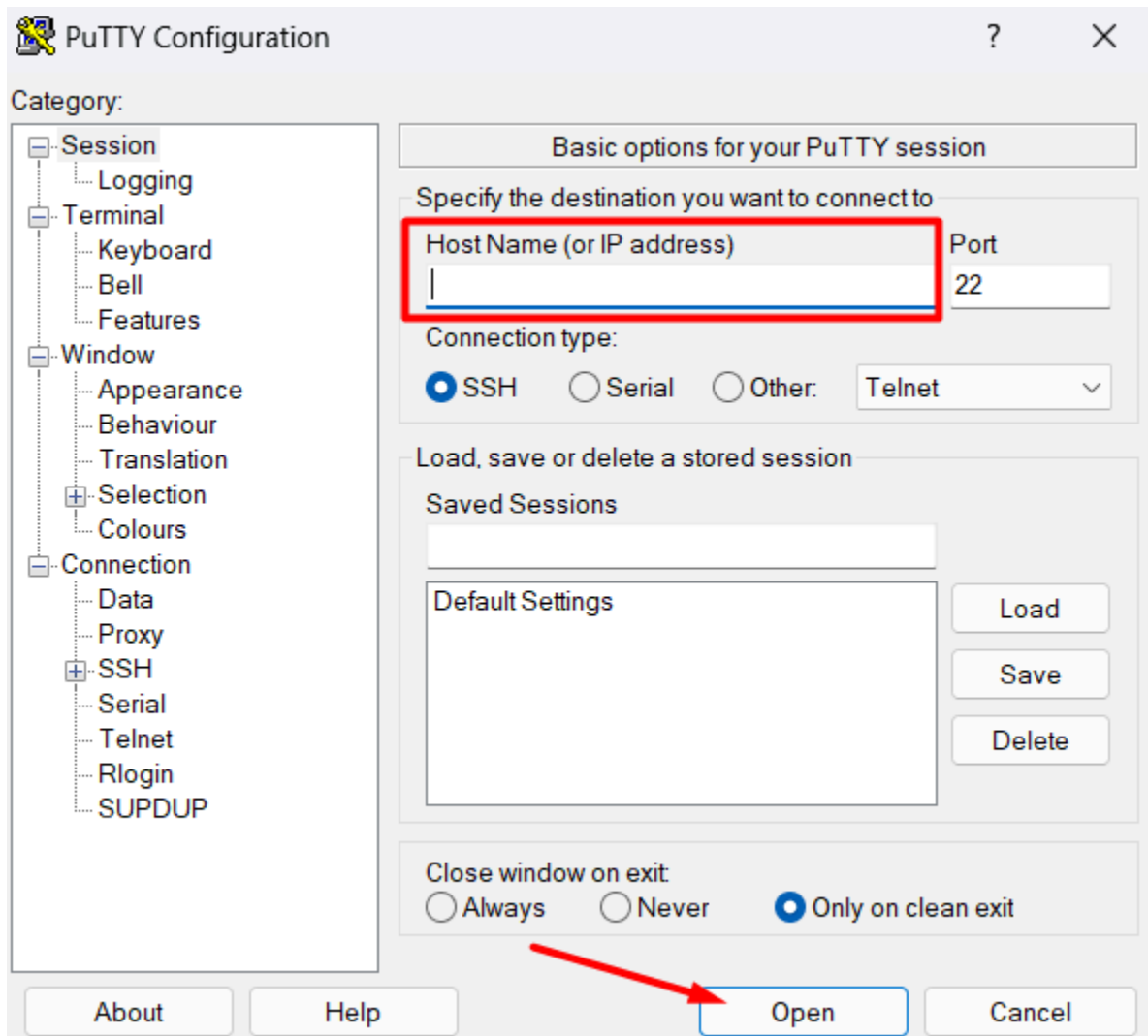
Instalar o PuTTY, seguindo o processo como indicado por eles



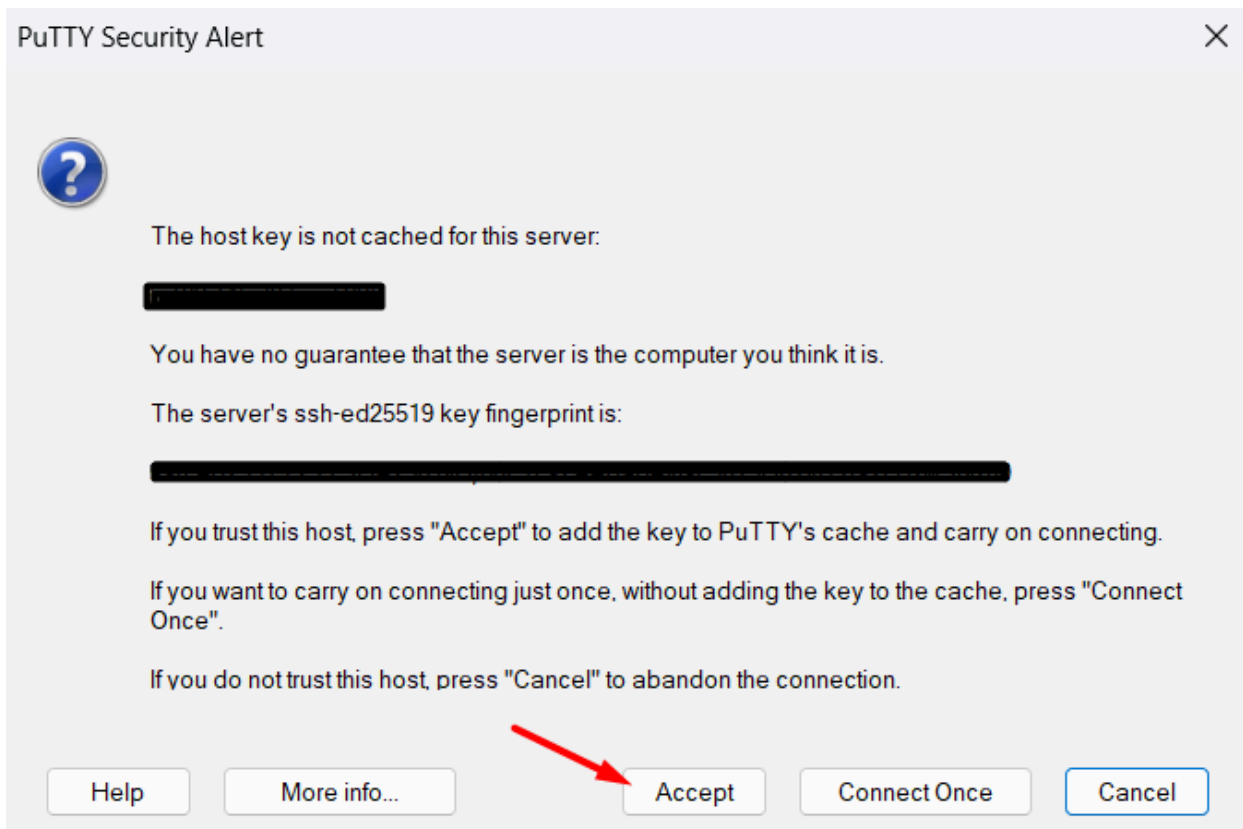
1.B) Acessando seu Servidor

Uma vez instalado, abra o programa.

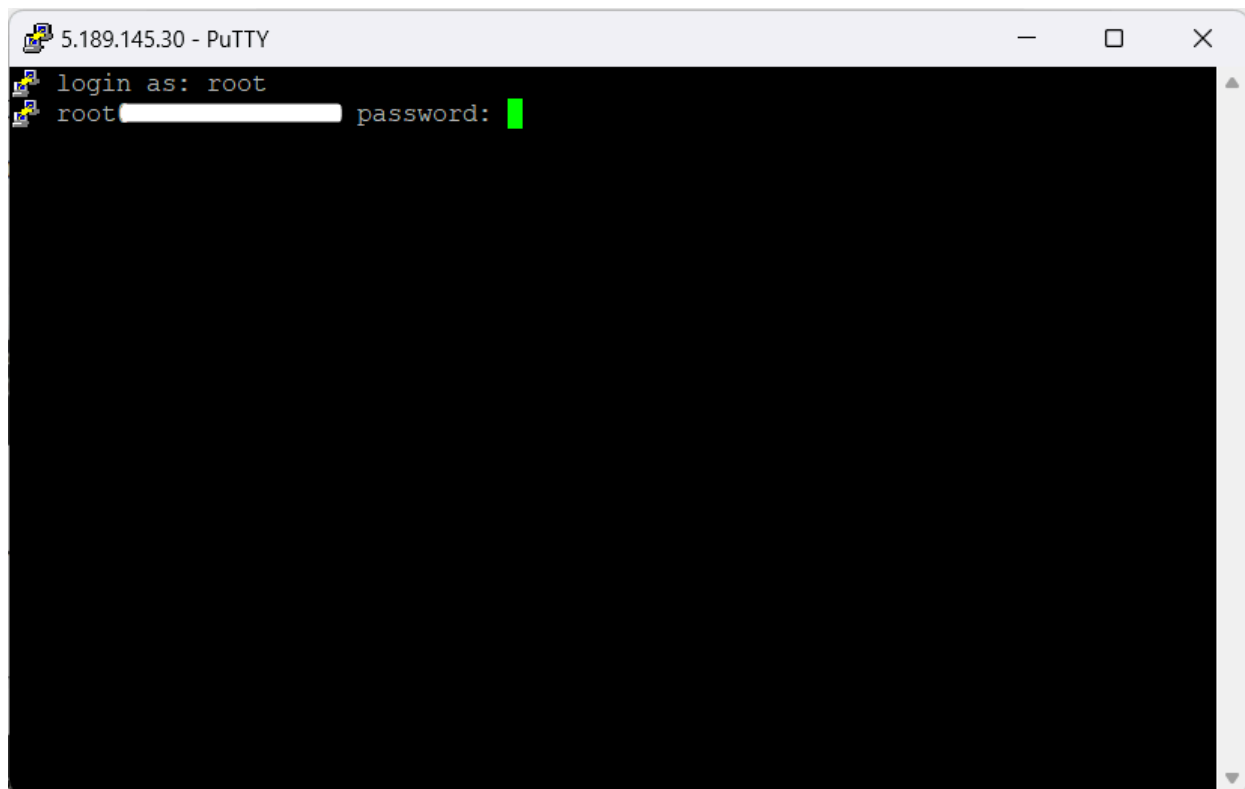
A Contabo vai ter te enviado um email com o IP do seu VPS. Abre o PuTTY, e coloca seu IP em "Host Name"



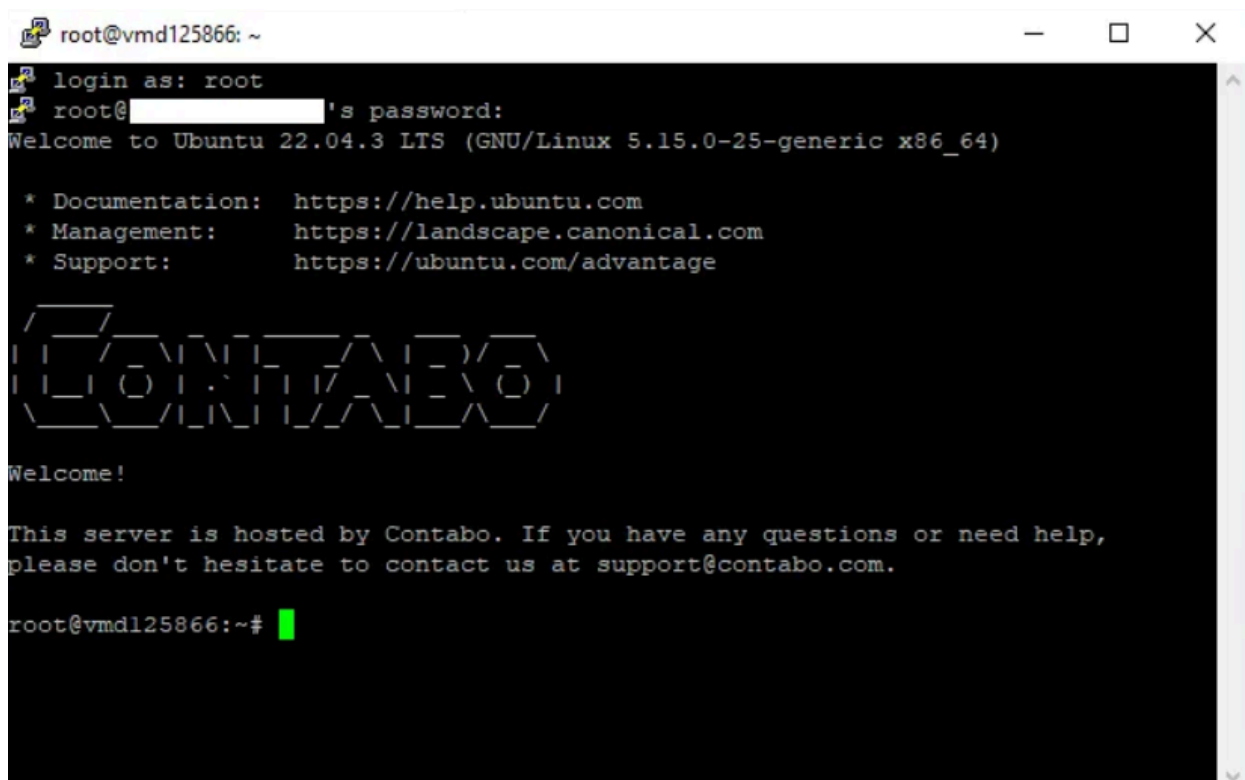
Se aparecer o erro abaixo, pode clicar em "Accept"



Feito isso, vai abrir a sua janela de comando. Colocar "root" em login, dar enter, colocar a senha que você definiu na Contabo e enter novamente.

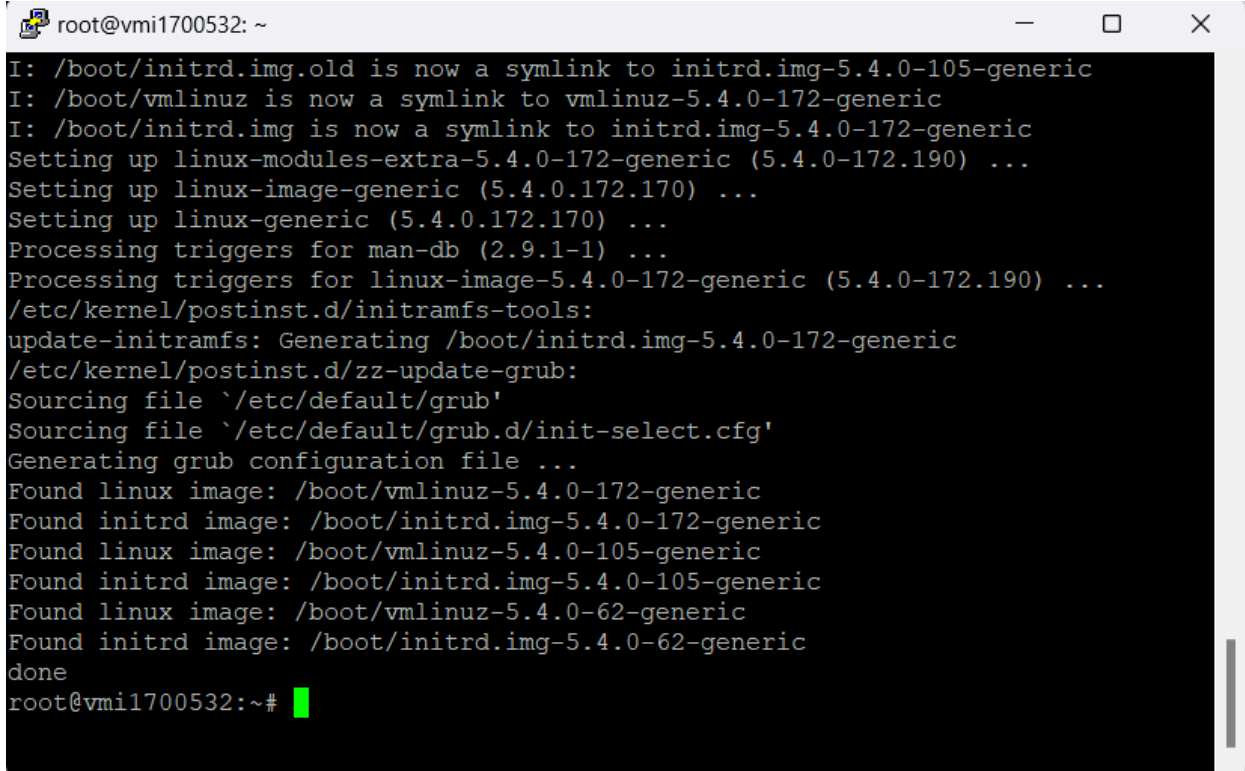


Você vai ter logado no seu servidor remoto!



2) Preparações

Vamos atualizar uns pacotes. Digite “**sudo apt update && sudo apt upgrade -y**” (sem as aspas) no terminal, e aperte “enter”. Sua tela vai ficar similar a abaixo.

A terminal window titled 'root@vml1700532: ~' with standard window controls. The terminal output shows the process of updating and upgrading packages. It lists several packages being updated, including linux-modules-extra-5.4.0-172-generic, linux-image-generic, and linux-generic. It also shows the processing of triggers for man-db and linux-image-5.4.0-172-generic. The output concludes with the generation of the grub configuration file and the discovery of various kernel images in the boot directory. The prompt 'root@vml1700532:~#' is followed by a green cursor.

```
root@vml1700532: ~
I: /boot/initrd.img.old is now a symlink to initrd.img-5.4.0-105-generic
I: /boot/vmlinuz is now a symlink to vmlinuz-5.4.0-172-generic
I: /boot/initrd.img is now a symlink to initrd.img-5.4.0-172-generic
Setting up linux-modules-extra-5.4.0-172-generic (5.4.0-172.190) ...
Setting up linux-image-generic (5.4.0.172.170) ...
Setting up linux-generic (5.4.0.172.170) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for linux-image-5.4.0-172-generic (5.4.0-172.190) ...
/etc/kernel/postinst.d/initramfs-tools:
update-initramfs: Generating /boot/initrd.img-5.4.0-172-generic
/etc/kernel/postinst.d/zz-update-grub:
Sourcing file `/etc/default/grub'
Sourcing file `/etc/default/grub.d/init-select.cfg'
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-5.4.0-172-generic
Found initrd image: /boot/initrd.img-5.4.0-172-generic
Found linux image: /boot/vmlinuz-5.4.0-105-generic
Found initrd image: /boot/initrd.img-5.4.0-105-generic
Found linux image: /boot/vmlinuz-5.4.0-62-generic
Found initrd image: /boot/initrd.img-5.4.0-62-generic
done
root@vml1700532:~#
```

Digite agora “**sudo apt -qy install curl git jq lz4 build-essential screen**” (sem as aspas, como todos os comandos neste guia) e aperte enter.

```
root@vmi1700532: ~  
Setting up libasan5:amd64 (9.4.0-1ubuntu1~20.04.2) ...  
Setting up jq (1.6-1ubuntu0.20.04.1) ...  
Setting up cpp-9 (9.4.0-1ubuntu1~20.04.2) ...  
Setting up libc6-dev:amd64 (2.31-0ubuntu9.14) ...  
Setting up binutils-x86-64-linux-gnu (2.34-6ubuntu1.9) ...  
Setting up binutils (2.34-6ubuntu1.9) ...  
Setting up dpkg-dev (1.19.7ubuntu3.2) ...  
Setting up libgcc-9-dev:amd64 (9.4.0-1ubuntu1~20.04.2) ...  
Setting up cpp (4:9.3.0-1ubuntu2) ...  
Setting up gcc-9 (9.4.0-1ubuntu1~20.04.2) ...  
Setting up libstdc++-9-dev:amd64 (9.4.0-1ubuntu1~20.04.2) ...  
Setting up gcc (4:9.3.0-1ubuntu2) ...  
Setting up g++-9 (9.4.0-1ubuntu1~20.04.2) ...  
Setting up g++ (4:9.3.0-1ubuntu2) ...  
update-alternatives: using /usr/bin/g++ to provide /usr/bin/c++ (c++) in auto mode  
Setting up build-essential (12.8ubuntu1.1) ...  
Processing triggers for libc-bin (2.31-0ubuntu9.14) ...  
Processing triggers for systemd (245.4-4ubuntu3.23) ...  
Processing triggers for man-db (2.9.1-1) ...  
Processing triggers for install-info (6.7.0.dfsg.2-5) ...  
root@vmi1700532:~#
```

3) Instalando Docker e Docker Compose

Agora vamos instalar o Docker, isso pode demorar alguns minutos.

Digite “**sudo apt install docker.io**” e aperte enter.

Se der uma mensagem de erro, provavelmente é que seu VPS já está com o Docker instalado, pode passar pra próxima etapa.

Uma vez instalado, cheque a versão usando “**docker --version**”. A versão deve ser, no mínimo, 24.0.5.

```
root@vmi1700532:~# docker --version  
Docker version 25.0.3, build 4debf41  
root@vmi1700532:~#
```

Vamos instalar agora o Docker Compose usando “**sudo curl -L "https://github.com/docker/compose/releases/download/v2.24.5/docker-compose-\$(uname -s)-\$(uname -m)" -o /usr/local/bin/docker-compose**”. (as aspas que estão no meio desse comando devem ser digitadas, se atentar também aos espaços)

```

For more help on how to use Docker, head to https://docs.docker.com/go/guides/
root@vmil700532:~# docker --version
Docker version 25.0.3, build 4debf41
root@vmil700532:~# sudo curl -L "https://github.com/docker/compose/releases/download/v2.24.5/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left   Speed
  0     0    0     0    0     0      0     0  --:--:-- --:--:-- --:--:--    0
100 58.5M 100 58.5M    0     0 30.0M      0  0:00:01  0:00:01 --:--:-- 46.5M
root@vmil700532:~#

```

Use agora “**sudo chmod +x /usr/local/bin/docker-compose**”, aperte enter. Depois verifique a versão do docker com “**docker-compose --version**”. Deve ser pelo menos v2.24.5

```

load/v2.24.5/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-co
mpose
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left   Speed
  0     0    0     0    0     0      0     0  --:--:-- --:--:-- --:--:--    0
100 58.5M 100 58.5M    0     0 30.0M      0  0:00:01  0:00:01 --:--:-- 46.5M
root@vmil700532:~# sudo chmod +x /usr/local/bin/docker-compose
root@vmil700532:~# docker-compose --version
Docker Compose version v2.24.5
root@vmil700532:~#

```

4) Clonando o Repositório

Agora vamos clonar o repositório starter, digitando “**git clone** <https://github.com/ritual-net/infernet-container-starter>”. Depois vamos navegar até ele, com “**cd infernet-container-starter**”.

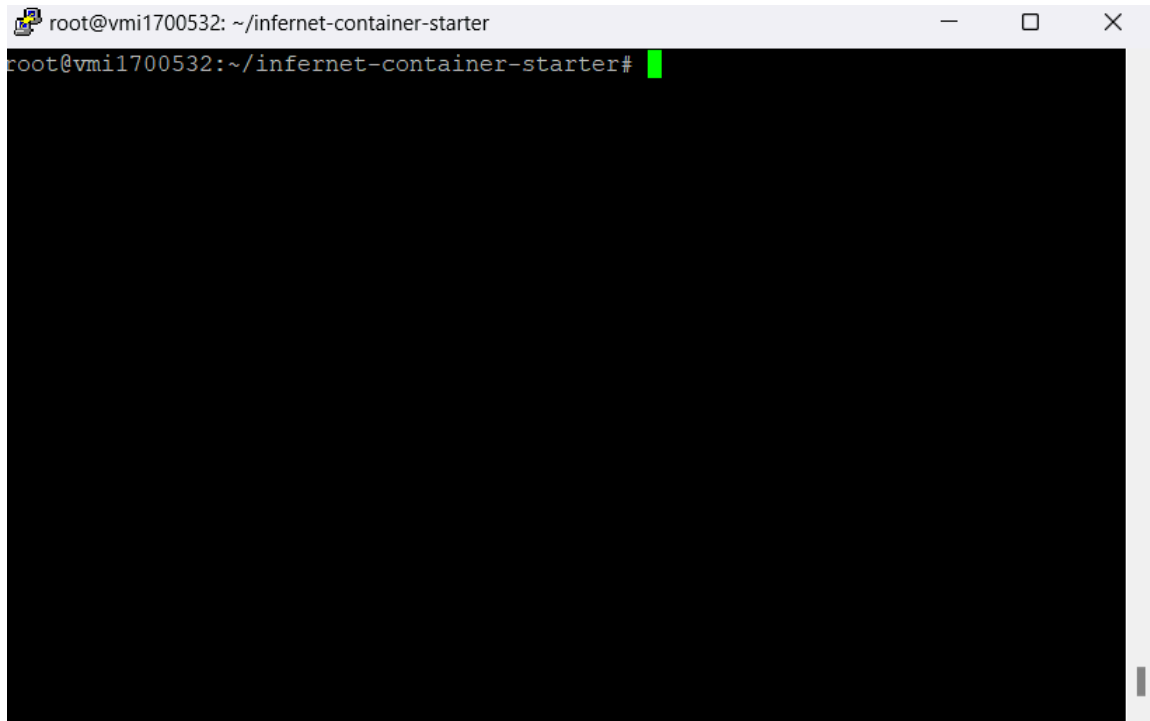
```

root@vmil700532:~# git clone https://github.com/ritual-net/infernet-container-starter
Cloning into 'infernet-container-starter'...
remote: Enumerating objects: 199, done.
remote: Counting objects: 100% (199/199), done.
remote: Compressing objects: 100% (125/125), done.
remote: Total 199 (delta 76), reused 175 (delta 52), pack-reused 0
Receiving objects: 100% (199/199), 41.95 KiB | 10.49 MiB/s, done.
Resolving deltas: 100% (76/76), done.
root@vmil700532:~# cd infernet-container-starter
root@vmil700532:~/infernet-container-starter#

```

5) Rodar o “Hello World”

Vamos iniciar uma nova tela, chamada ritual (ou qualquer outro nome), com o seguinte comando: “**screen -S ritual**”.

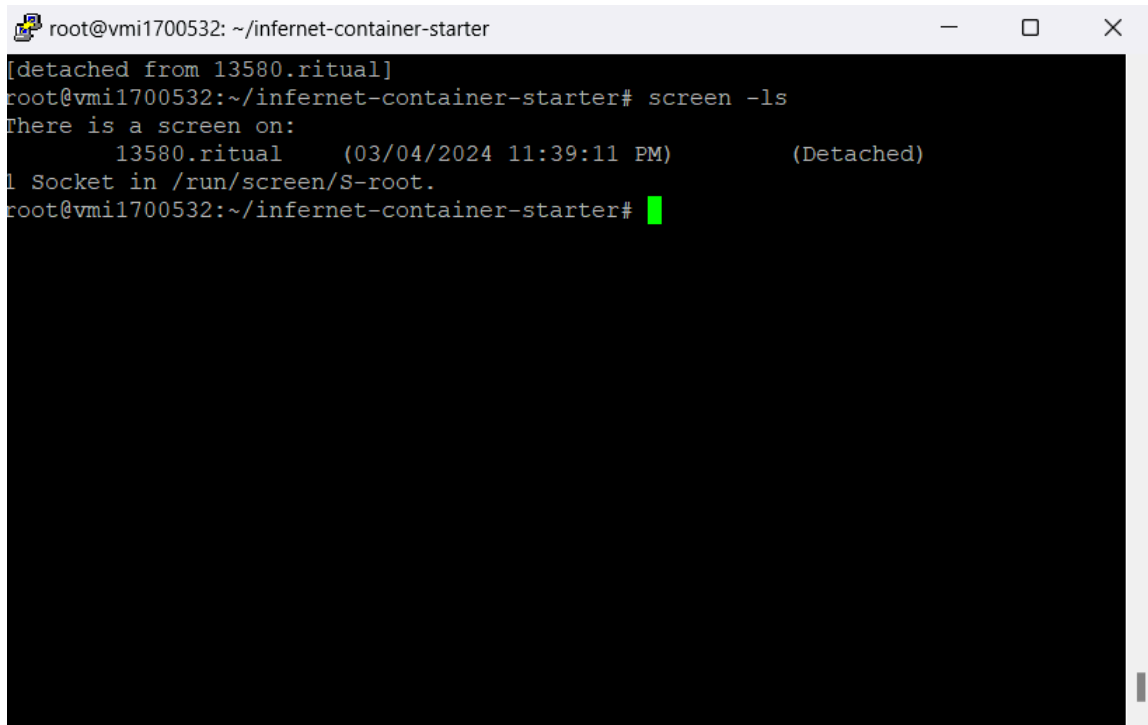
A screenshot of a terminal window. The title bar at the top reads "root@vml1700532: ~/infernct-container-starter" and includes standard window control icons (minimize, maximize, close). The terminal content shows the same prompt "root@vml1700532:~/infernct-container-starter#" followed by a green cursor. The rest of the terminal area is black and empty.

Vamos iniciar nosso projeto com o comando “**project=hello-world make deploy-container**”.

```
root@vmi1700532: ~/infernnet-container-starter
[chain.listener] head=0 behind=0
node-1 | 2024-03-04 22:41:37 [info] ] Collected highest subscription id
[chain.listener] id=1
node-1 | 2024-03-04 22:41:37 [info] ] Ignored subscription creation
[chain.listener] id=1 err=Container not supported
node-1 | 2024-03-04 22:41:37 [info] ] Finished snapshot sync
[chain.listener] new_head=0
node-1 | 2024-03-04 22:41:37 [info] ] Running node lifecycle
[ __main__ ]
node-1 | 2024-03-04 22:41:37 [info] ] Started ChainListener lifecycle
[chain.listener] last_synced=0
node-1 | 2024-03-04 22:41:37 [info] ] Serving REST webserver
[server.rest] addr=0.0.0.0 port=4000
fluentbit-1 | [0] stats.node: [[1709592097.000000000, {}], {"uid"=>"ac3a8ca0fb5
456b2fe63fcc5cf08d20550a87250c138f9b4a839ce66d8433ccc", "address"=>"0x70997970C5
1812dc3A010C7d01b50e0d17dc79C8", "containers"=>{"hello-world"=>{"allowed_ips"=>[
], "allowed_addresses"=>[], "allowed_delegate_addresses"=>[], "external"=>true},
"anvil-node"=>{"allowed_ips"=>[], "allowed_addresses"=>[], "allowed_delegate_ad
dresses"=>[], "external"=>true}}, "jobs_completed"=>{"offchain"=>{}, "onchain"=>
{}}, "container"=>{}}, "ip"=>"5.189.145.30", "resources"=>{"cpu"=>"x86_64 6 Intel
(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz", "disk"=>"588G", "gpu"=>nil, "kernel"=>"Li
nux 5.4.0-105-generic x86_64", "memory"=>"15Gi"}, "uptime"=>"up 47 minutes", "ve
rsion"=>"0.0.1"]]
```

Feito isso, vamos sair dessa sessão usando CTRL + A + D. Desse modo, saímos da tela sem interromper. Lembre desse comando.

Para listar todas as telas, você pode usar o comando `screen -ls`. Resultado é algo como mostrado abaixo, onde 13580 é a ID da tela.

A terminal window titled 'root@vml1700532: ~/infernnet-container-starter' with standard window controls. The terminal output shows a detached session '[detached from 13580.ritual]', followed by the command 'screen -ls'. The output lists a screen session: '13580.ritual (03/04/2024 11:39:11 PM) (Detached)' and a socket path. The prompt returns to 'root@vml1700532:~/infernnet-container-starter#' with a green cursor.

```
root@vml1700532: ~/infernnet-container-starter
[detached from 13580.ritual]
root@vml1700532:~/infernnet-container-starter# screen -ls
There is a screen on:
      13580.ritual      (03/04/2024 11:39:11 PM)      (Detached)
1 Socket in /run/screen/S-root.
root@vml1700532:~/infernnet-container-starter#
```

Para se conectar de volta na tela, o comando usado é “**screen -x [screen id]**”, onde você substitui o [screen id] pelo número do ID da tela - por exemplo, screen -x 13580.

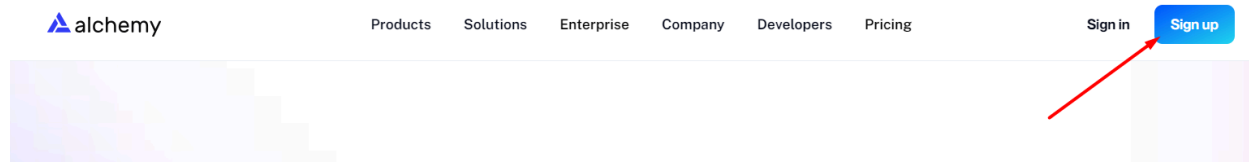
Após ter saído da tela (usando CTRL + A + D), vamos abrir uma segunda tela no terminal, do mesmo jeito que fizemos em 1.B.

Nessa segunda tela, usar o comando “**docker container ls**”, para checar se as implementações anteriores foram bem sucedidas. Deve aparecer algo similar a alguma das duas imagens abaixo em sua tela.

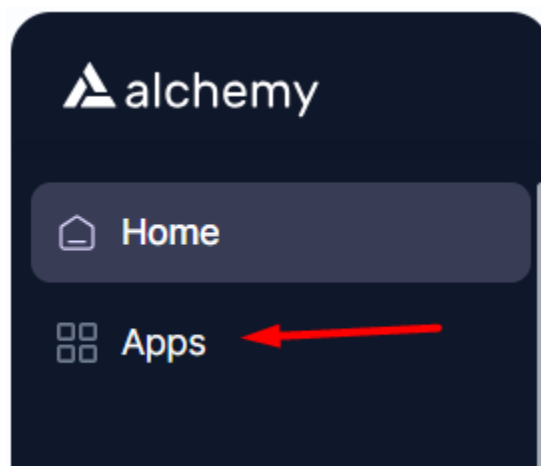
```
root@vmi1700532: ~  
This server is hosted by Contabo. If you have any questions or need help,  
please don't hesitate to contact us at support@contabo.com.  
  
Last login: Mon Mar  4 23:10:10 2024 from .....  
root@vmi1700532:~# docker container ls  
CONTAINER ID   IMAGE                                COMMAND                                  NAMES  
dd66746cff0    ritualnetwork/infernet-anvil:0.0.0  "anvil --host 0.0.0...              anvil-no  
5 minutes ago  Up 5 minutes      0.0.0.0:8545->3000/tcp  
ba76f6c0357    ritualnetwork/hello-world-infernet:latest  "gunicorn app:create...            hello-wo  
5 minutes ago  Up 5 minutes      0.0.0.0:3000->3000/tcp  
e5dd3538a89    ritualnetwork/infernet-node:latest  "/app/entrypoint.sh"              deploy-n  
5 minutes ago  Up 5 minutes      0.0.0.0:4000->4000/tcp  
a27b1115370    fluent/fluent-bit:latest            "/fluent-bit/bin/flu...            deploy-f  
5 minutes ago  Up 5 minutes      2020/tcp, 0.0.0.0:24224->24224/tcp  
1195a8a34d9    redis:latest                        "docker-entrypoint.s...            deploy-r  
5 minutes ago  Up 5 minutes      0.0.0.0:6379->6379/tcp  
root@vmi1700532:~#  
  
root@vmd130190:~# docker container ls  
CONTAINER ID   IMAGE                                COMMAND                                  CREATED        STATUS        PORTS                                NAMES  
1eab6b0e5db    ritualnetwork/infernet-anvil:0.0.0  "anvil --host 0.0.0..."  3 minutes ago  Up 3 minutes  0.0.0.0:8545->3000/tcp              anvil-node  
5deb618e046c    ritualnetwork/hello-world-infernet:latest  "gunicorn app:create..."  3 minutes ago  Up 3 minutes  0.0.0.0:3000->3000/tcp              hello-world  
f7efc13d0b68    ritualnetwork/infernet-node:latest  "/app/entrypoint.sh"      4 minutes ago  Up 4 minutes  0.0.0.0:4000->4000/tcp              deploy-node-1  
cd593138de1d    fluent/fluent-bit:latest            "/fluent-bit/bin/flu..."  4 minutes ago  Up 4 minutes  2020/tcp, 0.0.0.0:24224->24224/tcp  deploy-fluentbit-1  
130e3ef335fa    redis:latest                        "docker-entrypoint.s..."  4 minutes ago  Up 4 minutes  0.0.0.0:6379->6379/tcp              deploy-redis-1
```

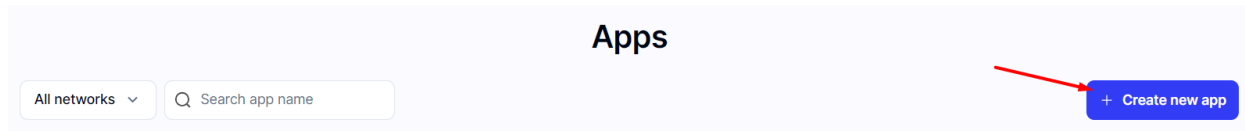
6) Conectando a Alchemy

Faça uma conta na [Alchemy](#).



Feito a conta, vá em “Apps” e em “Create new app”.





Crie um app na Chain Base, Network Base. Escolha um nome e uma descrição.

Create new app



Chain

 Base ▼

Network

 Base Mainnet ▼

Name ?

Ritual

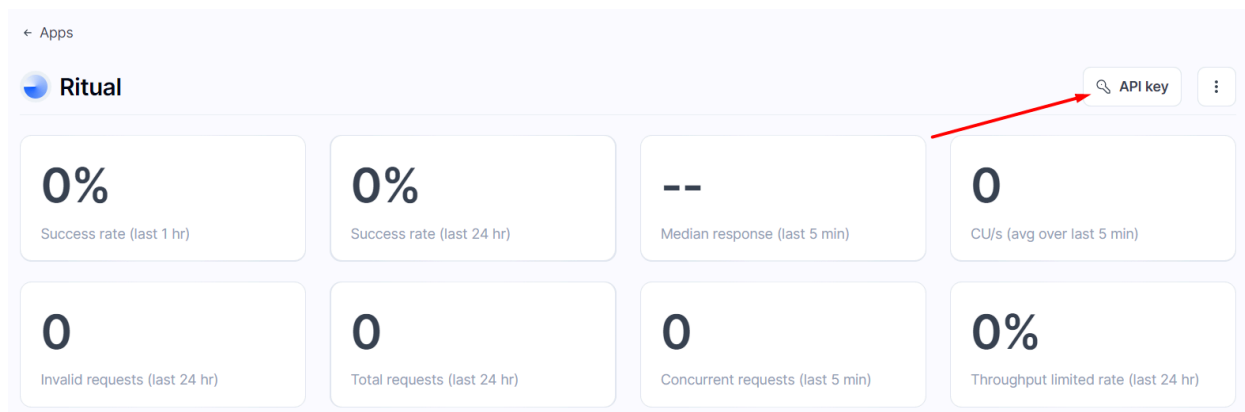
Description ?

Rodando um Node

Cancel

Create app

Uma vez criado seu app, vá em “API key”.



Copie o HTTPS, vamos precisar dele.

Connect to Alchemy



API Key

[Redacted API Key]

Copy

HTTPS

https://base-mainnet.g.alchemy.com/[Redacted API Key]

Copy

WebSocket

wss://base-mainnet.g.alchemy.com/v2/[Redacted API Key]

Copy

Code examples



Volte agora para a sua segunda tela de comando. Digite “[nano ~/infernnet-container-starter/deploy/config.json](#)”. Vai aparecer uma tela com dados como na imagem abaixo

Em `rpc_url`, coloque o HTTPS que você copiou na Alchemy

Em `coordinator_address`, coloque o seguinte endereço:
`0x8D871Ef2826ac9001fB2e33fDD6379b6aaBF449c`

Em `private_key`, você vai colocar as chaves privadas de sua carteira. Recomendamos usar uma carteira “burner”, criada apenas para isso, e com poucos fundos. Você pode ver como exportar sua chave [nesse guia](#). Se sua chave não começar com “0x”, adicione “0x” no início, esse prefixo é importante.

```
{
  "log_path": "infernnet_node.log",
  "server": {
    "port": 4000
  },
  "chain": {
    "enabled": true,
    "trail_head_blocks": 5,
    "rpc_url": "https://base-mainnet.g.alchemy.com/v2/[Redacted API Key]",
    "coordinator_address": "0x8D871Ef2826ac9001fB2e33fDD6379b6aaBF449c",
    "wallet": {
      "max_gas_limit": 5000000,
      "private_key": "[Redacted Private Key]"
    }
  }
}
```

Uma vez que tenha editado os parâmetros, vamos salvar e sair. Aperte CTRL + X, depois Y para salvar as mudanças, e aperte enter para sair.

7) Iniciar na Nova Configuração

Já estamos com sua carteira vinculada. Vamos aplicar essa nova configuração.

Vá para sua segunda tela do terminal, e digite os comandos abaixo, um por um, apertando enter após cada um.

`“docker restart anvil-node”`

`“docker restart hello-world”`

`“docker restart deploy-node-1”`

`“docker restart deploy-fluentbit-1”`

`“docker restart deploy-redis-1”`

Use agora o comando `“docker ps”` para listar seus containers. Deve aparecer uma tela parecida com alguma dessas abaixo:

```
root@vmd130190:~/infern-container-starter# docker ps
CONTAINER ID   IMAGE                                COMMAND                                  CREATED      STATUS      PORTS                                NAMES
1eab6bb9e5db   ritualnetwork/infernet-anvil:0.0.0  "anvil --host 0.0.0..."              3 hours ago  Up 5 minutes  0.0.0.0:8545->3000/tcp               anvil-node
5deb618e046c   ritualnetwork/hello-world-infernet:latest  "unicorn app:create..."              3 hours ago  Up 5 minutes  0.0.0.0:3000->3000/tcp               hello-world
f7efc13d0b68   ritualnetwork/infernet-node:latest  "/app/entrypoint.sh"                  3 hours ago  Up 11 seconds  0.0.0.0:4000->4000/tcp               deploy-node-1
cd593138de1d   fluent/fluent-bit:latest             "/fluent-bit/bin/flu..."             3 hours ago  Up 22 minutes  2020/tcp, 0.0.0.0:24224->24224/tcp    deploy-fluentbit-1
130a3af335fa   redis:latest                         "docker-entrypoint.s..."             3 hours ago  Up 22 minutes  0.0.0.0:6379->6379/tcp               deploy-redis-1
```



```
CONTAINER ID   IMAGE                                COMMAND                                  CREATED      STATUS      PORTS                                NAMES
2dd66746cff0   ritualnetwork/infernet-anvil:0.0.0  "anvil --host 0.0.0..."              40 minutes ago  Up 33 seconds  0.0.0.0:8545->3000/tcp               anvil-node
dba76f6c0357   ritualnetwork/hello-world-infernet:latest  "unicorn app:create..."              40 minutes ago  Up 33 seconds  0.0.0.0:3000->3000/tcp               hello-world
ae5dd3538a89   ritualnetwork/infernet-node:latest  "/app/entrypoint.sh"                  40 minutes ago  Up 2 seconds  0.0.0.0:4000->4000/tcp               deploy-node-1
5a27b1115370   fluent/fluent-bit:latest             "/fluent-bit/bin/flu..."             40 minutes ago  Up 23 seconds  2020/tcp, 0.0.0.0:24224->24224/tcp    deploy-fluentbit-1
41195a8a34d9   redis:latest                         "docker-entrypoint.s..."             40 minutes ago  Up 10 seconds  0.0.0.0:6379->6379/tcp               deploy-redis-1
root@vml1700532:~#
```

Você pode checar se cada um de seus containers está atualizado, usando o comando `“docker logs <CONTAINER ID>”`.

Lembre de substituir <CONTAINER ID> pelo ID do container, a sequência de letras e números destacado nas imagens acima.

Quando você fizer isso pro container “deploy-node-1”, deve aparecer algo similar a tela abaixo, com seu HTTPS da Alchemy no Initialized RPC, e o endereço de sua carteira no “Initialized Wallet”.

```

2024-02-28 22:17:45 [info] Running startup [chain_enabled=True]
2024-02-28 22:17:45 [info] Initialized Container Manager [orchestration.docker] port_mappings={'hello-world': '3000', 'anvil-node': '8545'}
2024-02-28 22:17:45 [info] Initialized Redis client [orchestration.store] host=redis port=6379
2024-02-28 22:17:45 [info] Initialized Guardian [orchestration.guardian]
2024-02-28 22:17:45 [info] Initialized RPC [chain.rpc] url=https://base-mainnet.g.alchemy.com/v2/PzC...
2024-02-28 22:17:46 [info] Initialized Coordinator [chain.coordinator] address=0x5E50B2315678a1ecb3671032d03F6d4210aa3
2024-02-28 22:17:46 [info] Initialized Wallet [chain.wallet] address=0x3585...
2024-02-28 22:17:46 [info] Initialized ChainProcessor [chain.processor]
2024-02-28 22:17:46 [info] Initialized ChainListener [chain.listener]
2024-02-28 22:17:46 [info] Initialized RESTServer [server.rest] port=4000
2024-02-28 22:17:46 [info] Running node lifecycle setup [main]

```

8) Instalar Foundry e Dependências

Dentro da segunda tela do terminal, onde estamos, digite os seguintes comandos, separadamente, e apertando enter após cada um. Estamos instalando o Foundry, um kit para desenvolvimento na Ethereum.

```

"cd"
"mkdir foundry"
"cd foundry"
"curl -L https://foundry.paradigm.xyz | bash"
"source ~/.bashrc"
"foundryup"

```

Vamos agora instalar dependências, digitando os comandos abaixo, separadamente, e apertando enter após cada um.

```

"cd ~/inferneter-container-starter/projects/hello-world/contracts"
"forge install --no-commit foundry-rs/forge-std"
"forge install --no-commit ritual-net/inferneter-sdk"
"cd ../../.."

```

9) Implementar um Contrato

Abra uma terceira tela no seu terminal (do mesmo jeito que fizemos em 1B). Essa vai ser uma tela de monitoramento. Digite nela o comando **"docker logs -f anvil-node"**.

Vamos agora implementar o contrato "SaysGM", um exemplo fornecido pelo time da Ritual. Volte para sua segunda tela, e execute os comandos abaixo, separadamente, e apertando enter após cada um.

```

"cd ~/inferneter-container-starter"
"project=hello-world make deploy-contracts"

```

Volte para sua terceira tela, a de monitoramento. Você deve ver algo similar a imagem abaixo, que é um sinal que o contrato foi implementado.

```
eth_sendRawTransaction
eth_getTransactionReceipt

Transaction: 0x23ca6b1d1823ad5af175c207c2505112f60038fc000e1e22509816fa29a3afd6
Contract created: 0x663f3ad617193148711d28f5334ee4ed07016602
Gas used: 476669
```

```
Block Number: 1
Block Hash: 0x6b026b70fbe97b4a733d4812ccd6e8e25899a1f6c622430c3fb07a2e5c5c96b7
Block Time: "Wed, 17 Jan 2024 22:17:31 +0000"
```

```
eth_getTransactionByHash
eth_getTransactionReceipt
eth_blockNumber
```

10) “Chamar” o Contrato

Na sua segunda tela, digite o seguinte comando “**project=hello-world make call-contract**”.

Isso deve gerar uma transação no seu contrato. Volte para a terceira tela, a de monitoramento, e você deve ver algo similar a imagem abaixo.

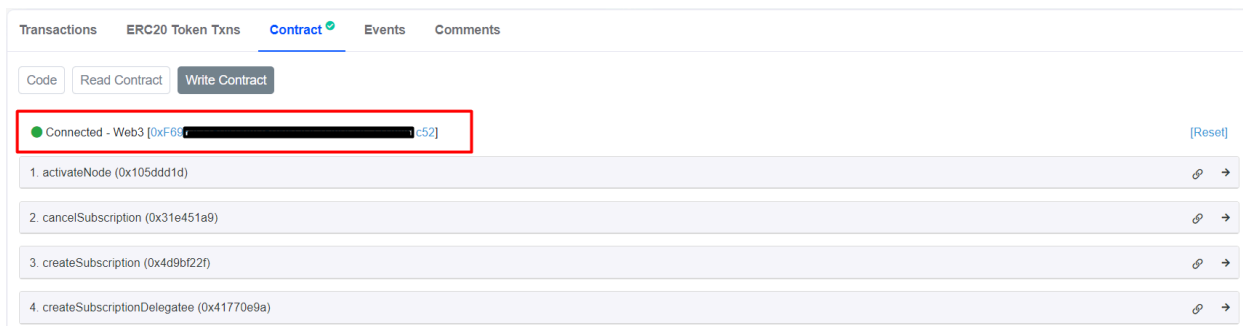
```
eth_getTransactionReceipt

Transaction: 0xe56b5b6ac713a978a1631a44d6a0c9eb6941dce929e1b66b4a2f7a61b0349d65
Gas used: 123323

Block Number: 2
Block Hash: 0x3d6678424adcdecfa0a8edd51e014290e5f54ee4707d4779e710a2a4d9867c08
Block Time: "Wed, 17 Jan 2024 22:18:39 +0000"
eth_getTransactionByHash
```

Um pouco depois, você deve ver outra transação.

Edit: essa terceira transação (com o escrito RITUAL grande) não está aparecendo para a maioria das pessoas. Se não aparecer para você, pode seguir adiante sem problemas.



Uma vez conectado, vá no fim da página, em “registerNode”. Clique, e cole o endereço de sua carteira na barra. Depois, clique em “Write”. Vai aparecer uma transação em sua Metamask, aceite ela.



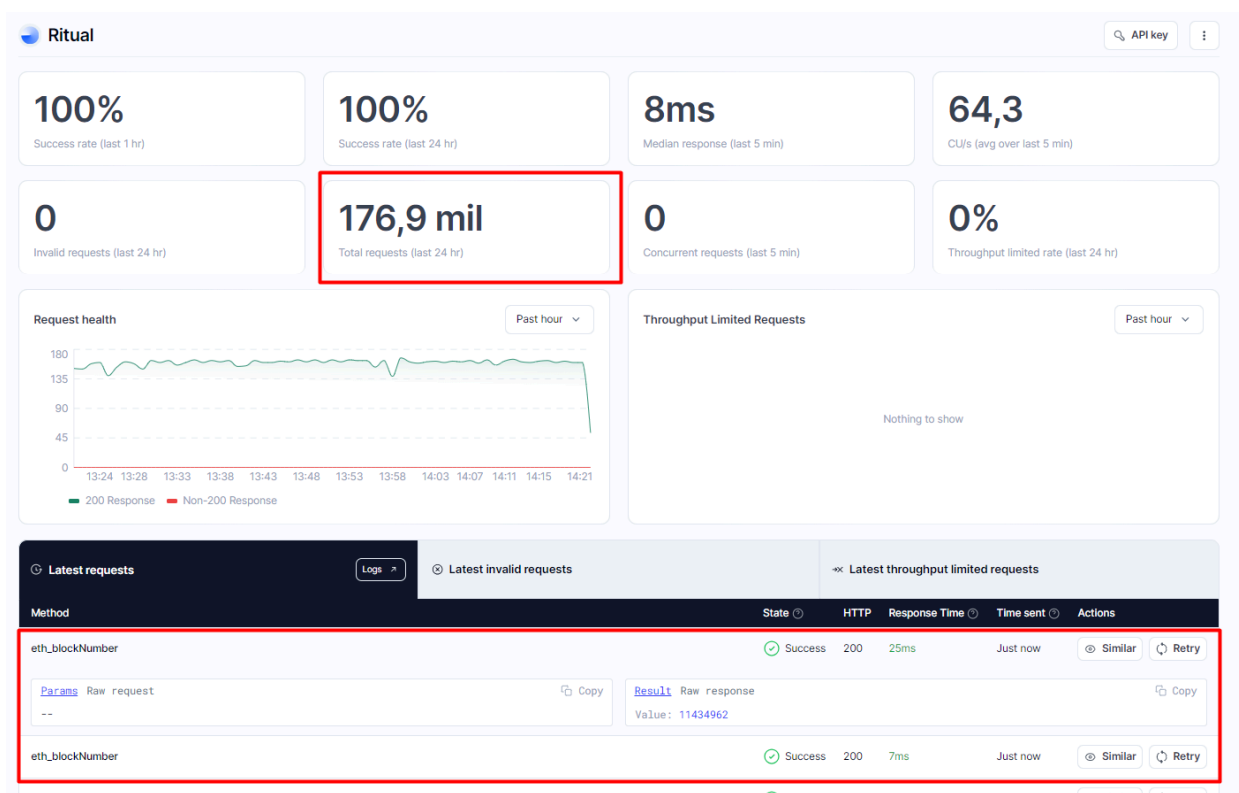
Espere 1 hora. Depois, nessa mesma página, vá em “activateNode”. Clique em “Write”, e confirme a transação em sua Metamask.



Parabéns! Seu Node deve estar funcionando.

12) Conferir se Está Tudo Certo

Vá na página do seu app na Alchemy. Confira se está aparecendo os “requests”.



Vá no [explorador de blocos da Base](#), procure pelo seu endereço. Confira se aparece as transações de registrar e ativar o Node.

Txn Hash	Method ⓘ	Block	Age
 0xb63eda19839c44c0a6...	Activate Node	11405677	27 secs ago
 0x973220eb9942b6c2a8...	Register Node	11403794	1 hr 3 mins ago

Volte no seu terminal de comando. Cheque se todos seus containers estão sem erros, usando o comando “**docker logs <CONTAINER ID>**”. Lembre de substituir <CONTAINER ID> pelo ID de cada container.

Se essas 3 checagens (Alchemy, explorador de blocos, e no terminal de comando) estiverem sem erros, seu Node deve estar funcionando sem problemas.

13) Possíveis Erros

- a) Se o seu node já estava funcionando, mas parou de fazer “requests” no dashboard da Alchemy, você pode ter que reiniciar os dockers. Digite os comandos abaixo, um a um, apertando enter após cada um.

“docker restart anvil-node”

“docker restart hello-world”

“docker restart deploy-node-1”

“docker restart deploy-fluentbit-1”

“docker restart deploy-redis-1”

Se esse problema estiver sendo constante, você pode automatizar a reiniciação dos dockers, com [esse mini-guia em nosso discord](#).

- b) Outro erro que está acontecendo com algumas pessoas é a carteira ficar mandando transações falhas, como na imagem abaixo.

Enquanto isso não interfere no funcionamento do node, essas transações gastam ETH e podem drenar o dinheiro que está em sua carteira.

A melhor solução, por enquanto, é simplesmente deixar pouco ETH na carteira (5 dólares ou menos), e assim elas irão parar.

Para fazer isso, você pode simplesmente fazer um swap de seu ETH para wETH.

	0x0e0ec1551919cce0304...	Deliver Compute	11524240	4 days 16 mins ago	0x4a3cf284c3bd62fa904...		0x8d871ef2826ac9001fb...	0 ETH	0.000266007184
	0x783dadff6e15f7d7c4a...	Deliver Compute	11522703	4 days 1 hr ago	0x4a3cf284c3bd62fa904...		0x8d871ef2826ac9001fb...	0 ETH	0.000376275279
	0x6869a334ced6c3495d...	Deliver Compute	11522697	4 days 1 hr ago	0x4a3cf284c3bd62fa904...		0x8d871ef2826ac9001fb...	0 ETH	0.000377183342
	0x3d51204c651b694684...	Deliver Compute	11520572	4 days 2 hrs ago	0x4a3cf284c3bd62fa904...		0x8d871ef2826ac9001fb...	0 ETH	0.000283930896
	0xb90045eefe9ff1233aa...	Deliver Compute	11516080	4 days 4 hrs ago	0x4a3cf284c3bd62fa904...		0x8d871ef2826ac9001fb...	0 ETH	0.000284370451
	0xafa46579fe98fb0790c...	Deliver Compute	11515849	4 days 4 hrs ago	0x4a3cf284c3bd62fa904...		0x8d871ef2826ac9001fb...	0 ETH	0.000312614475
	0xb873889e506ab5b5b9...	Deliver Compute	11514524	4 days 5 hrs ago	0x4a3cf284c3bd62fa904...		0x8d871ef2826ac9001fb...	0 ETH	0.000219025078

Esse guia é uma adaptação deste [artigo](#).

Atualizando seu Node para a Versão 0.2.0

Edit: um membro da Paradigma fez um tutorial com as atualizações novas do Node. Confira [aqui](#). (obrigado, Bruno)