

# Streaming de Dados em Tempo Real: Aula 1

*Prof. Felipe Timbó*



# Objetivos

Estudar/investigar princípios, técnicas e ferramentas necessárias para lidar com **streaming de dados**.

Desenvolver soluções em tempo real, isto é, à medida que dados são produzidos.

Resolver problemas relacionados a **streaming de dados em tempo real** com Kafka e Spark

# Ementa

- **Dia 1:**
  - Introdução a Streaming de Dados
  - Introdução ao Apache Kafka
  - Setup do ambiente
- **Dia 2:**
  - Data Ingestion com Apache Kafka
  - Kafka Connect
  - Kafka Web Project

# Ementa

- Dia 3:
  - Introdução ao Apache Spark
- Dia 4:
  - Processamento de dados de Streaming com Apache Spark

# Tecnologias e Ferramentas deste Curso

Máquina Virtual (VirtualBox)

Linux (Ubuntu)

Python

VS Code

Apache Kafka

Apache Spark

# Metodologia

Aulas expositivas com discussões

Práticas remotas

Leituras

Tarefas individuais

Projeto final (até 3 pessoas)

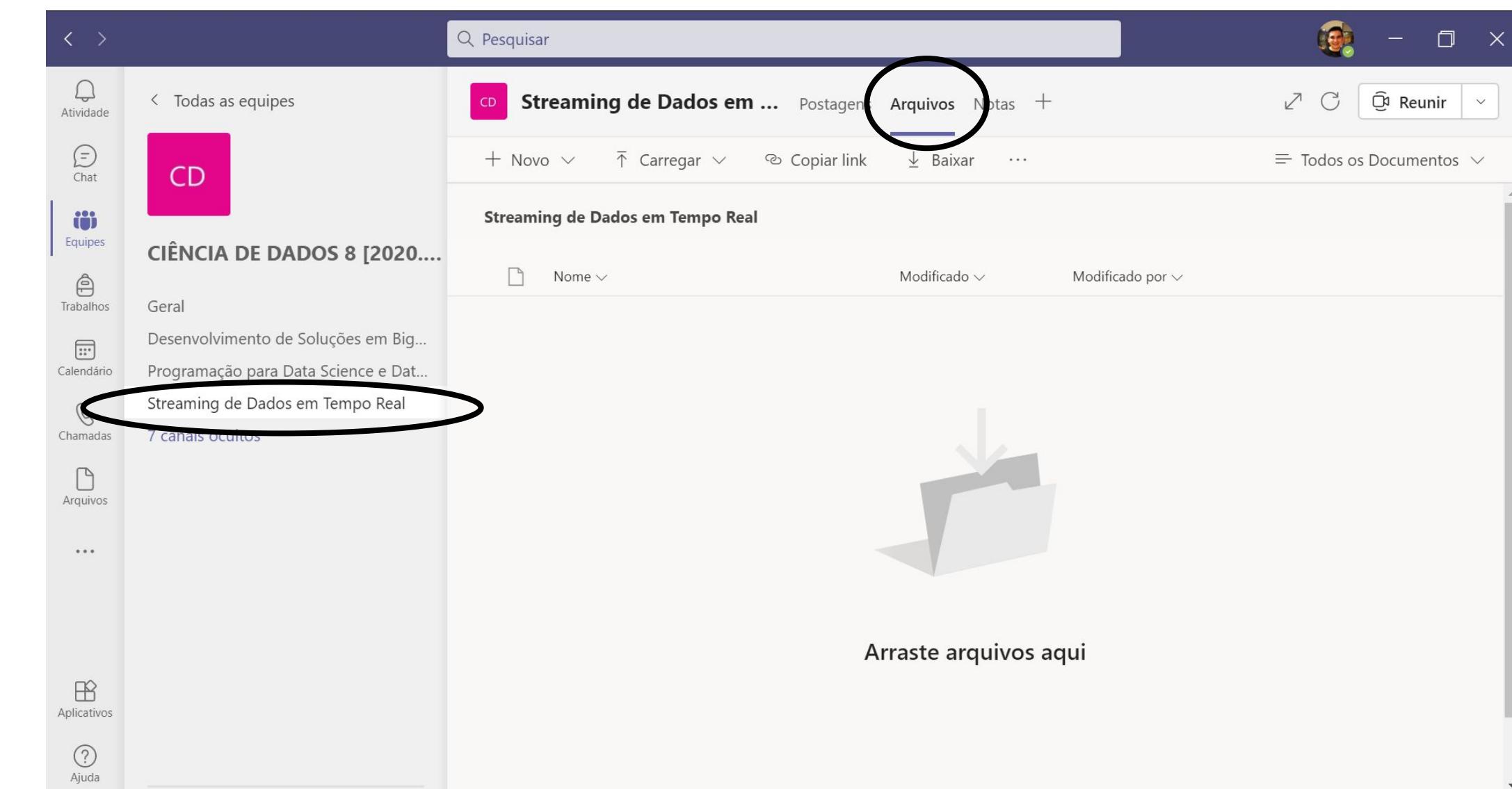
# Recursos

Lista de e-mails:

[uni7-ciencia-de-dados-turma8@googlegroups.com](mailto:uni7-ciencia-de-dados-turma8@googlegroups.com)

Arquivos (slides, livros e artigos)

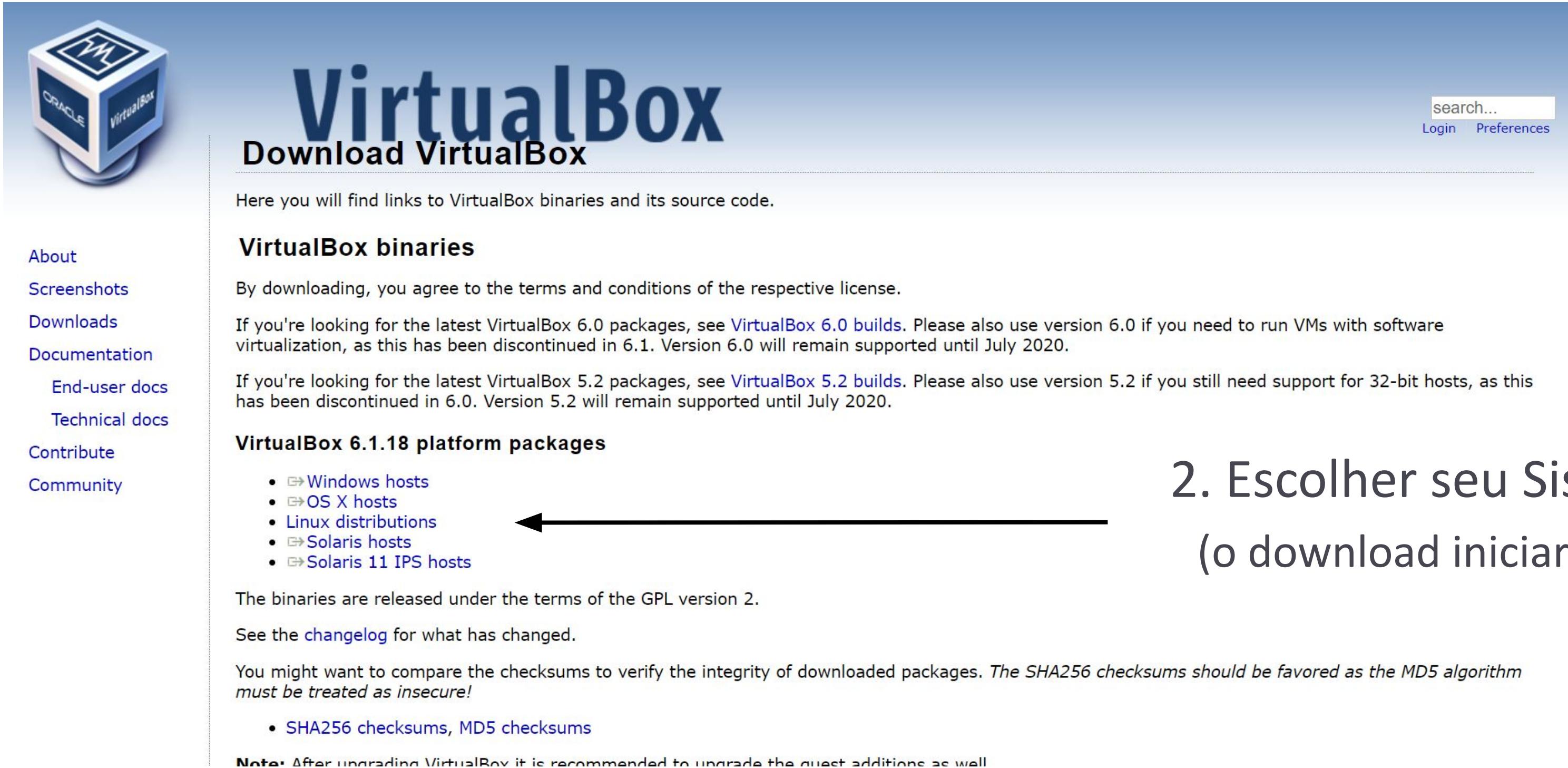
Microsoft Teams



Antes de tudo...

# Download do VirtualBox

1. Acessar <https://www.virtualbox.org/wiki/Downloads>



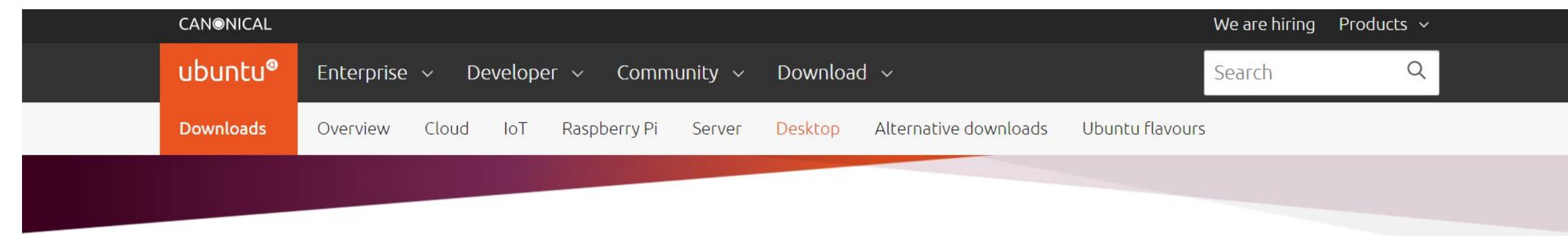
The screenshot shows the "VirtualBox Download" page. At the top left is the Oracle VM VirtualBox logo. The main title "VirtualBox" is in large blue letters, with "Download VirtualBox" below it. To the right are "search...", "Login", and "Preferences" buttons. On the left, a sidebar lists links: "About", "Screenshots", "Downloads", "Documentation", "End-user docs", "Technical docs", "Contribute", and "Community". The main content area starts with a note about finding binaries and source code. It then has two sections: "VirtualBox binaries" and "VirtualBox 6.1.18 platform packages". The "VirtualBox binaries" section contains text about license terms and links to "VirtualBox 6.0 builds" and "VirtualBox 5.2 builds". The "VirtualBox 6.1.18 platform packages" section lists "Windows hosts", "OS X hosts", "Linux distributions", "Solaris hosts", and "Solaris 11 IPS hosts". Below these sections are notes about GPL version 2, a changelog link, and checksum verification tips. A note at the bottom advises upgrading guest additions after an upgrade.

2. Escolher seu Sistema Operacional  
(o download iniciará automaticamente)

# Download do Ubuntu

3. Acessar: <https://ubuntu.com/download/desktop>

4. Realizar o download do  
Ubuntu 20.04.2.0 LTS



Download Ubuntu Desktop

The page shows the title "Ubuntu 20.04.2.0 LTS". Below it, a paragraph explains that this is the latest LTS version, providing long-term support until April 2025. It includes a link to "Ubuntu 20.04 LTS release notes". A large green "Download" button is prominently displayed. At the bottom, there's a link to "Recommended system requirements". To the right, a sidebar provides information about other versions of Ubuntu Desktop, including torrents, network installers, local mirrors, and past releases.

Ubuntu 20.04.2.0 LTS

Download the latest LTS version of Ubuntu, for desktop PCs and laptops. LTS stands for long-term support — which means five years, until April 2025, of free security and maintenance updates, guaranteed.

[Ubuntu 20.04 LTS release notes ↗](#)

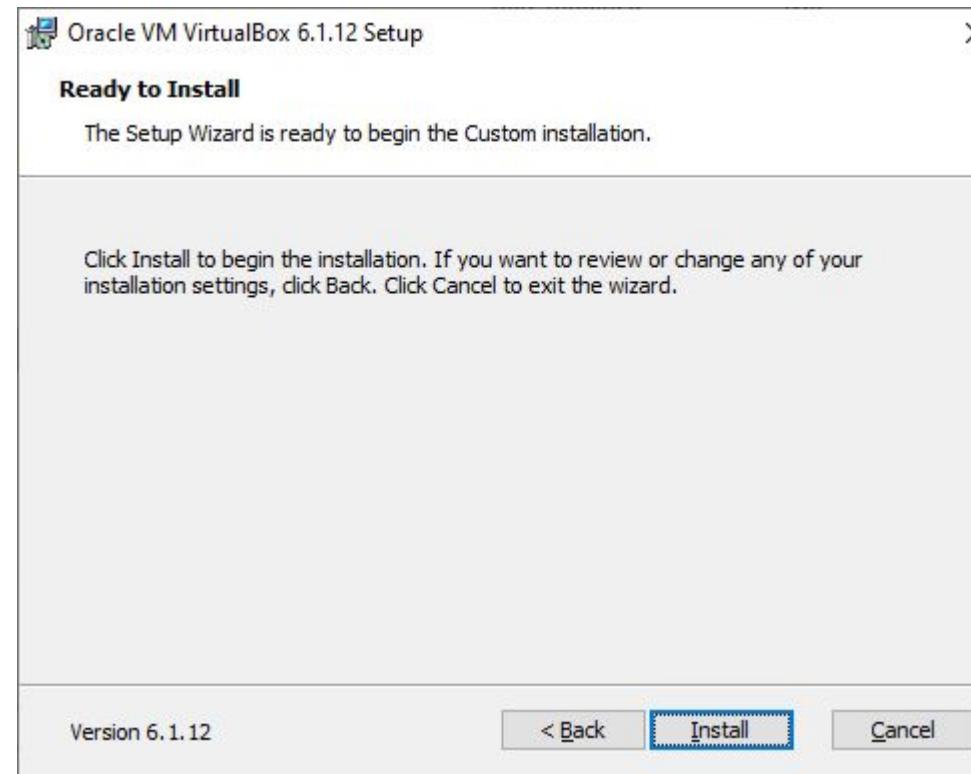
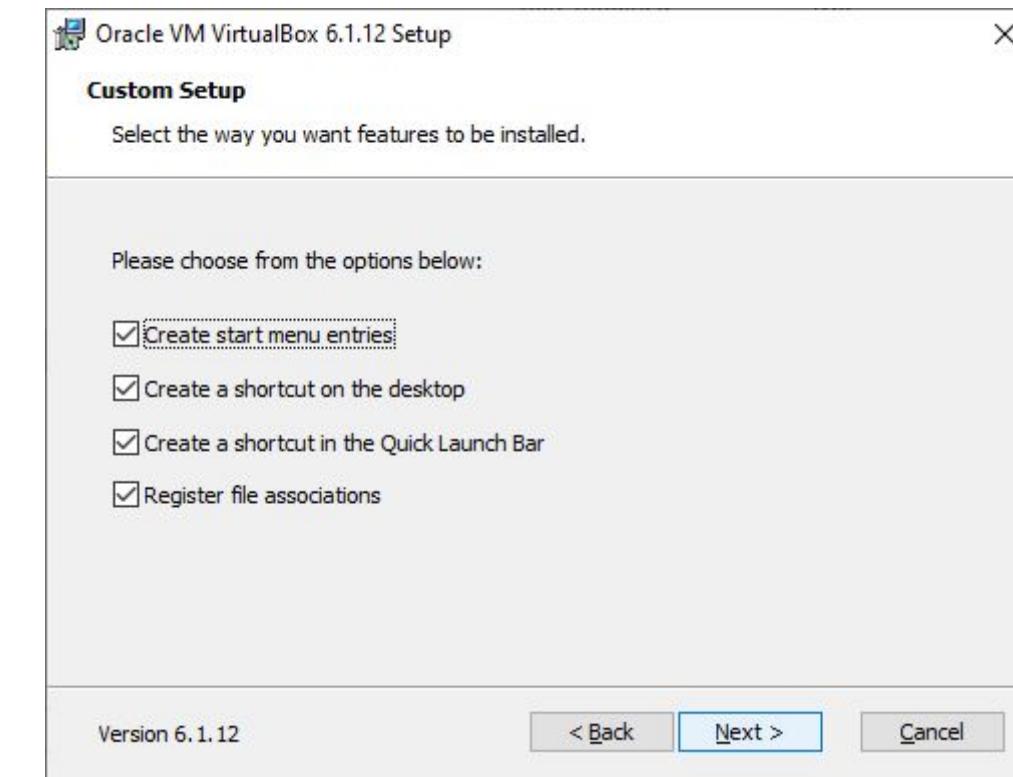
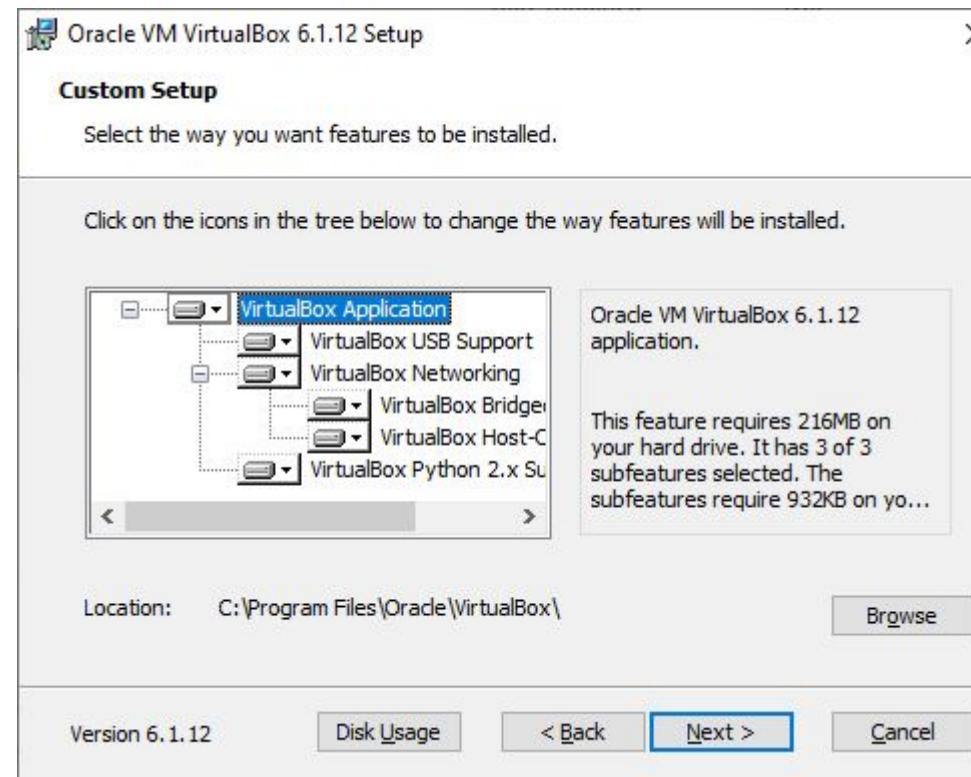
[Recommended system requirements](#)

Download

For other versions of Ubuntu Desktop including torrents, the network installer, a list of local mirrors, and past releases see our [alternative downloads](#).

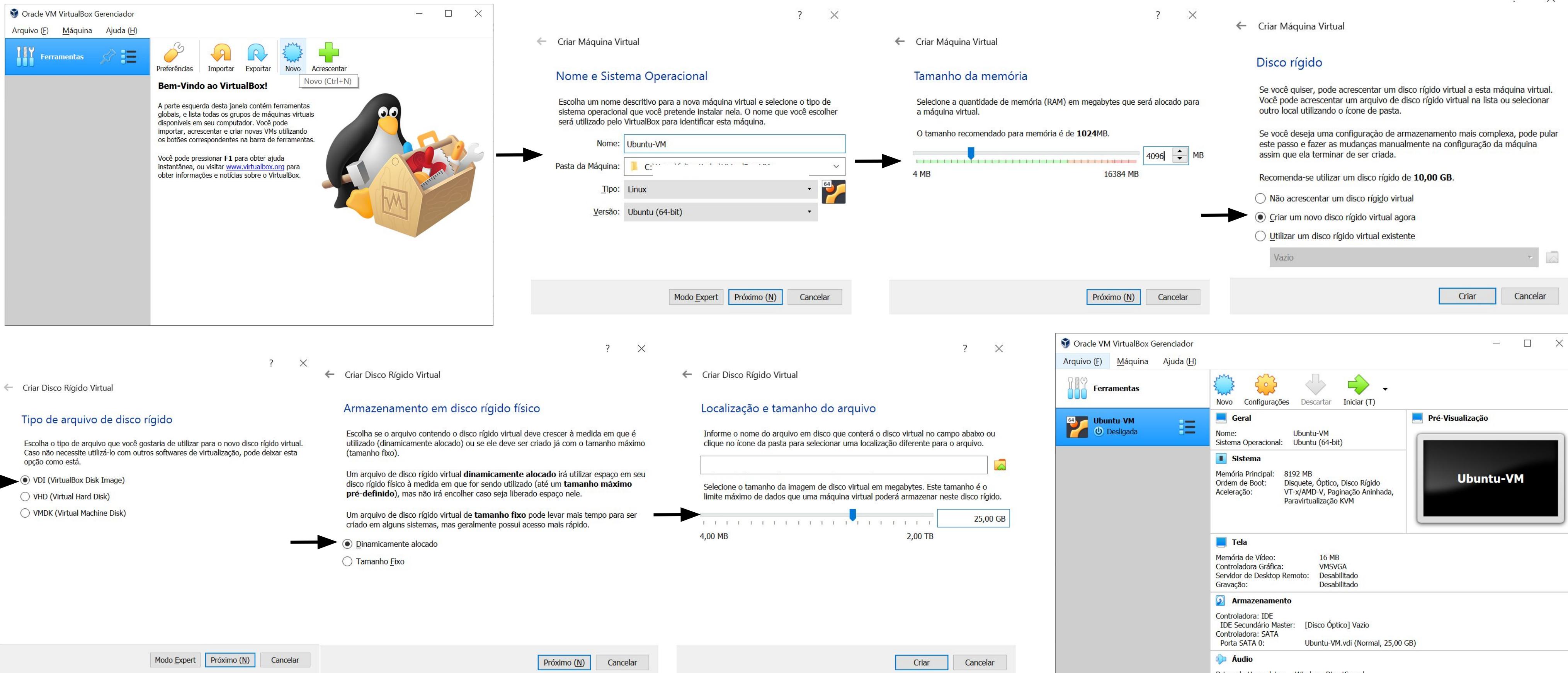
# Instalação do VirtualBox

## 5. Instalar o VirtualBox



# Criação de uma VM (Virtual Machine)

## 6. Criar uma nova Máquina Virtual Ubuntu



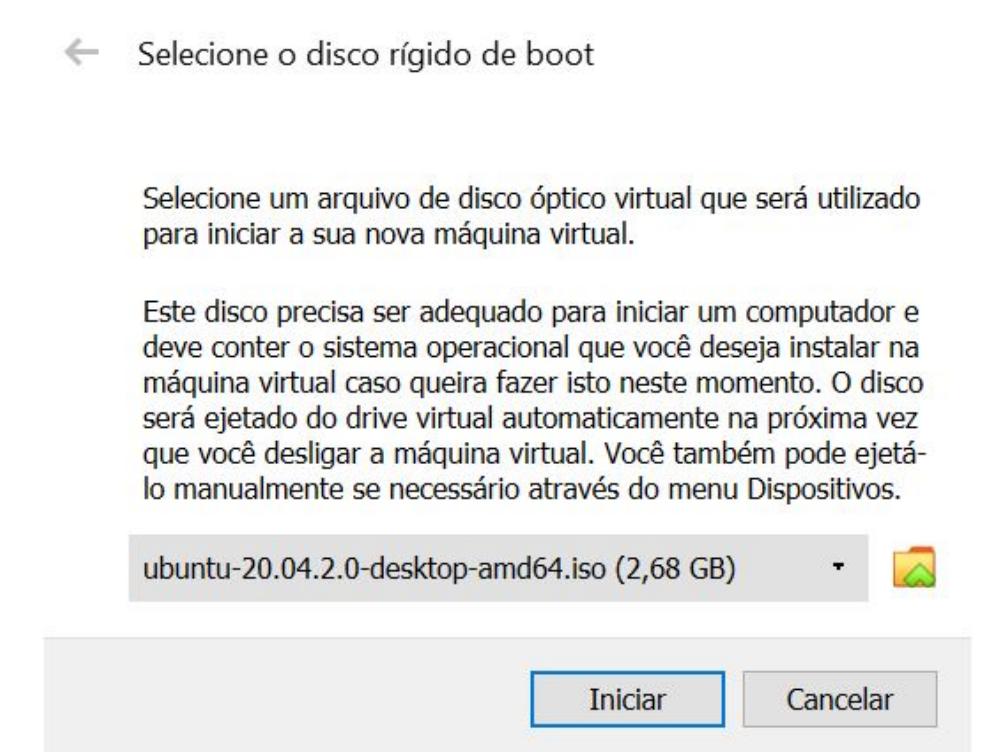
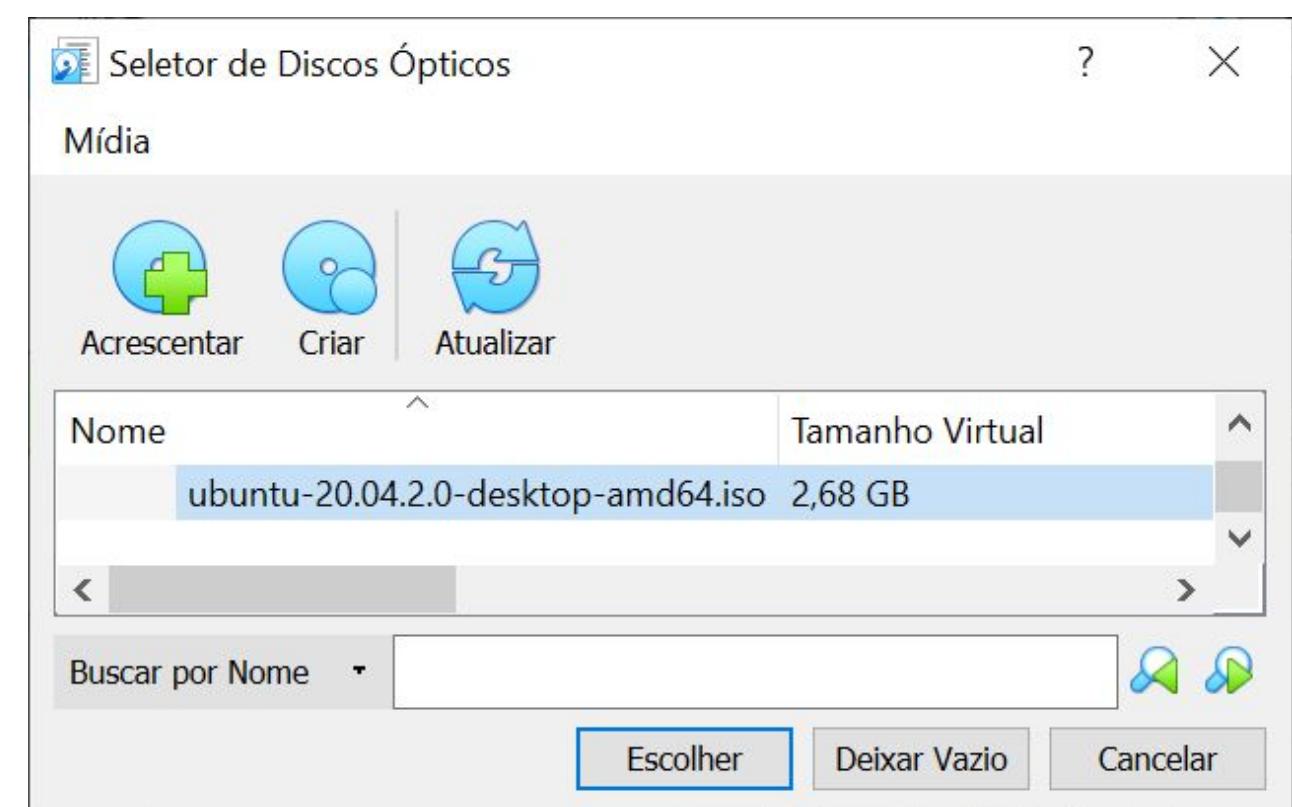
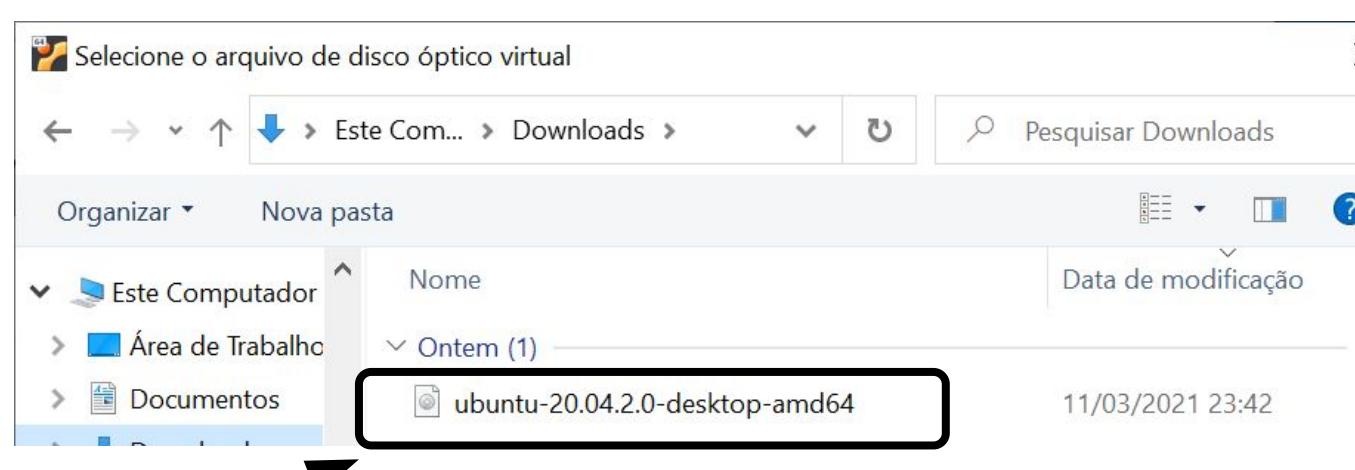
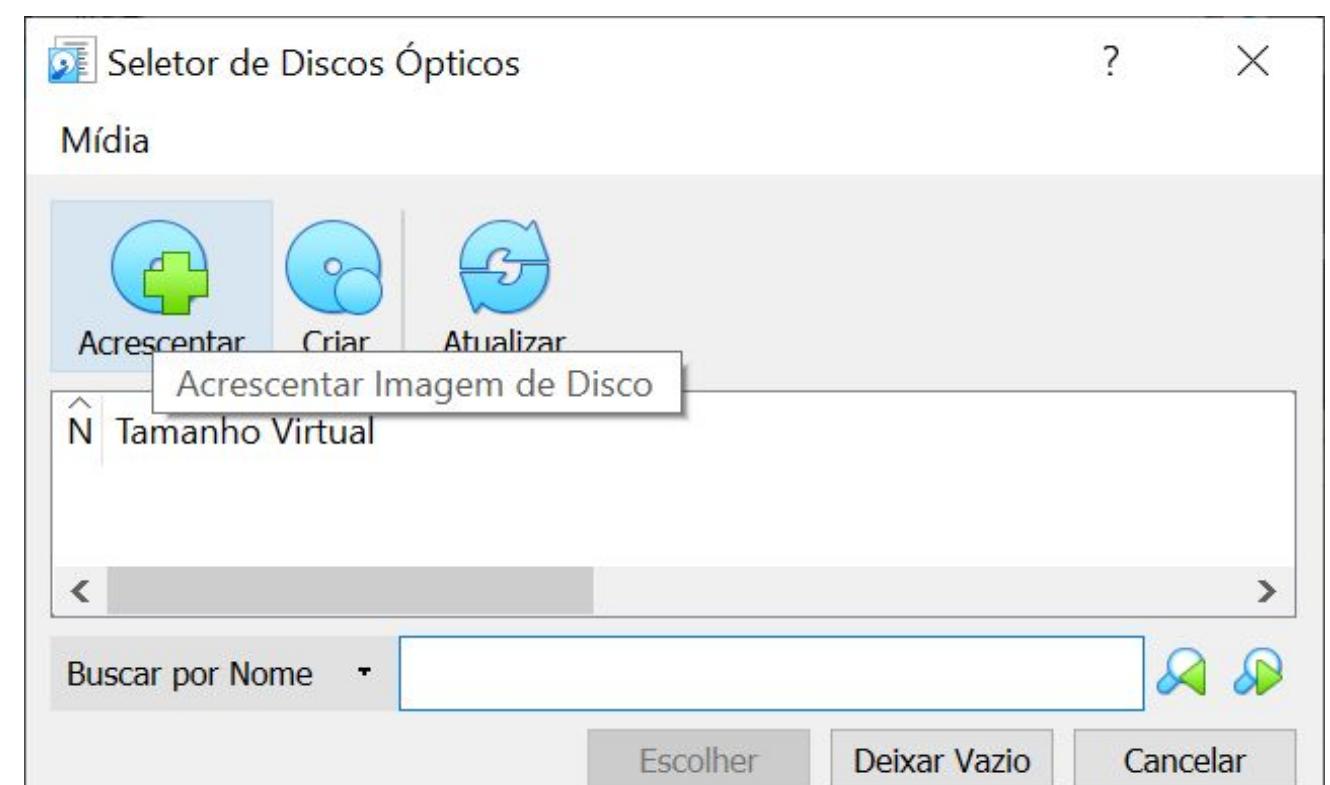
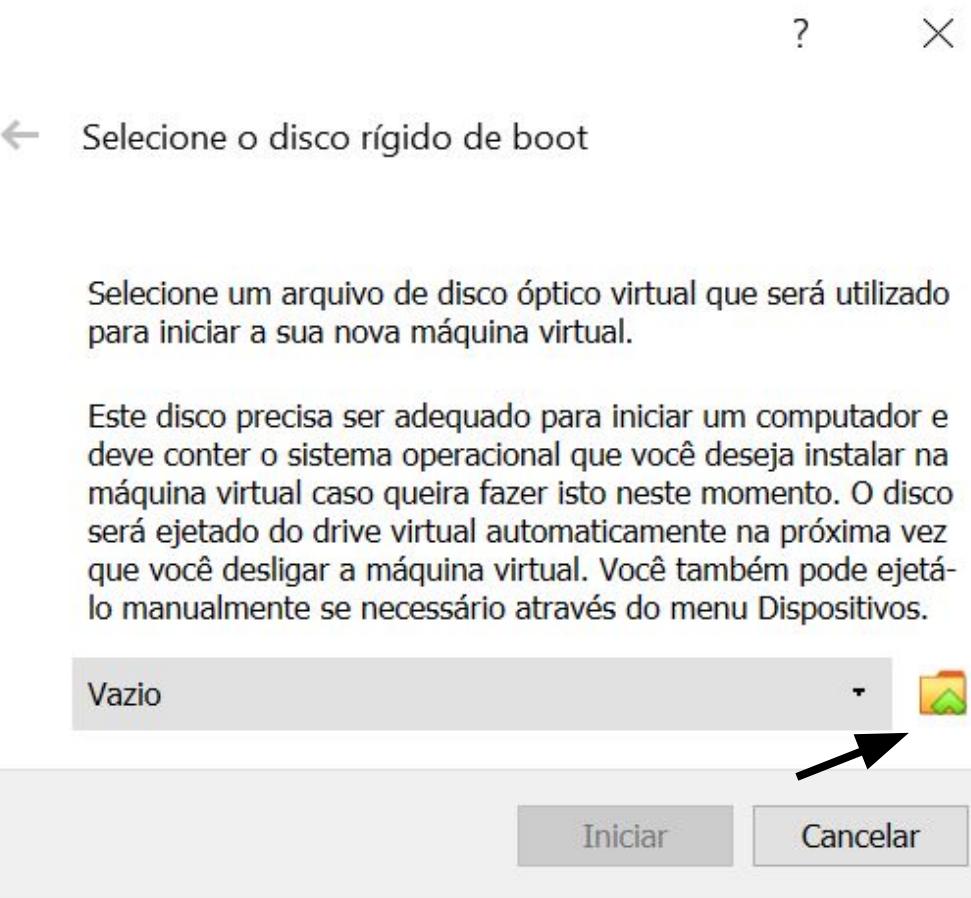
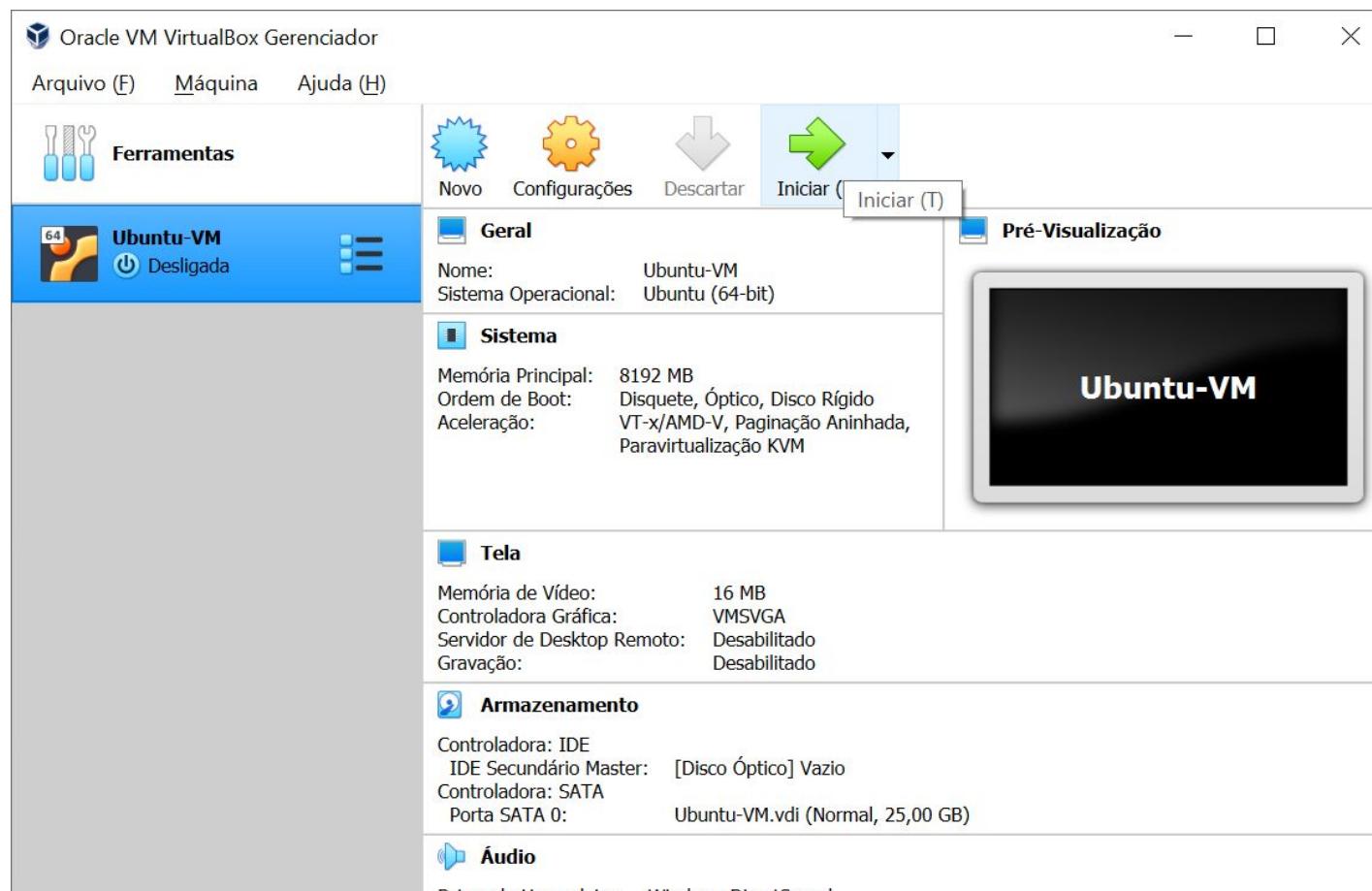
The image shows the process of creating a new Virtual Machine (VM) in Oracle VM VirtualBox. It consists of six windows arranged in a flow:

- Oracle VM VirtualBox Gerenciador**: Shows the main interface with a penguin icon and various tools.
- Criar Máquina Virtual - Nome e Sistema Operacional**: Step 1 of 3. Set Name to "Ubuntu-VM", Host Path to "C:\", Type to "Linux", and Version to "Ubuntu (64-bit)".
- Criar Máquina Virtual - Tamanho da memória**: Step 2 of 3. Set RAM size to 4096 MB (1024 MB recommended).
- Criar Máquina Virtual - Disco rígido**: Step 3 of 3. Set Disk Type to "VHD (Virtual Hard Disk)" and click "Criar".
- Criar Disco Rígido Virtual - Tipo de arquivo de disco rígido**: Step 1 of 2. Set Type to "VDI (VirtualBox Disk Image)" (selected).
- Criar Disco Rígido Virtual - Localização e tamanho do arquivo**: Step 2 of 2. Set Size to 25,00 GB.
- Oracle VM VirtualBox Gerenciador**: Final view showing the newly created VM "Ubuntu-VM" in the list, which is currently off.

Arrows indicate the progression from the main manager to the creation steps and back to the final configuration screen.

# Criação de uma VM (Virtual Machine)

## 7. Iniciar a Máquina Virtual Ubuntu



# Instalação do Ubuntu

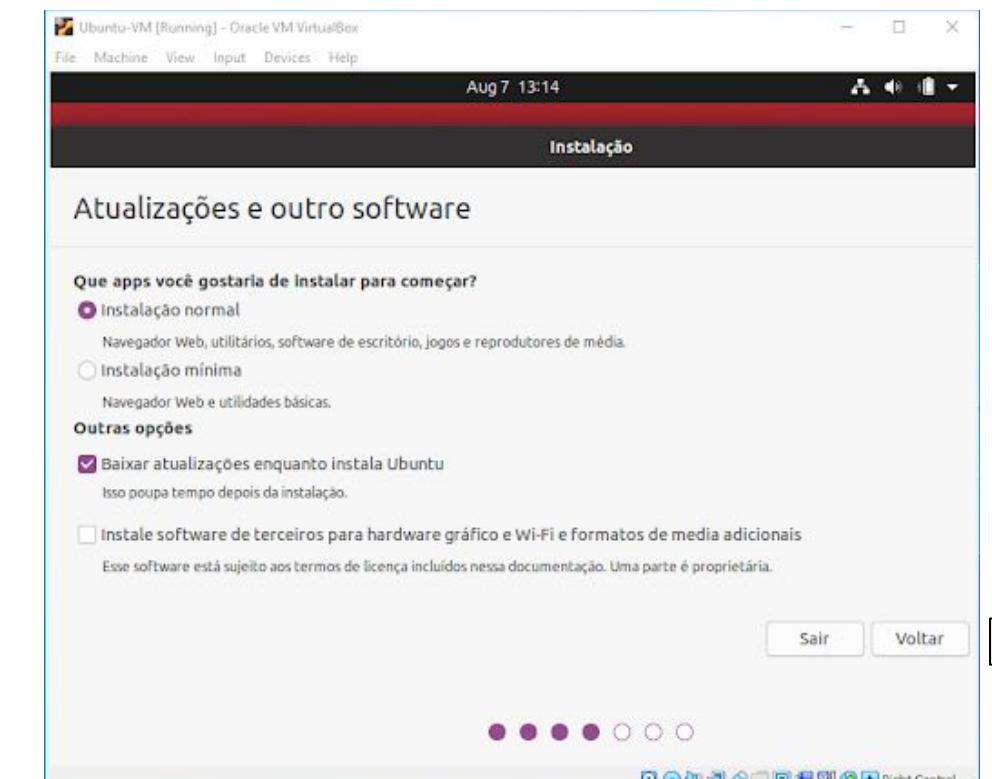
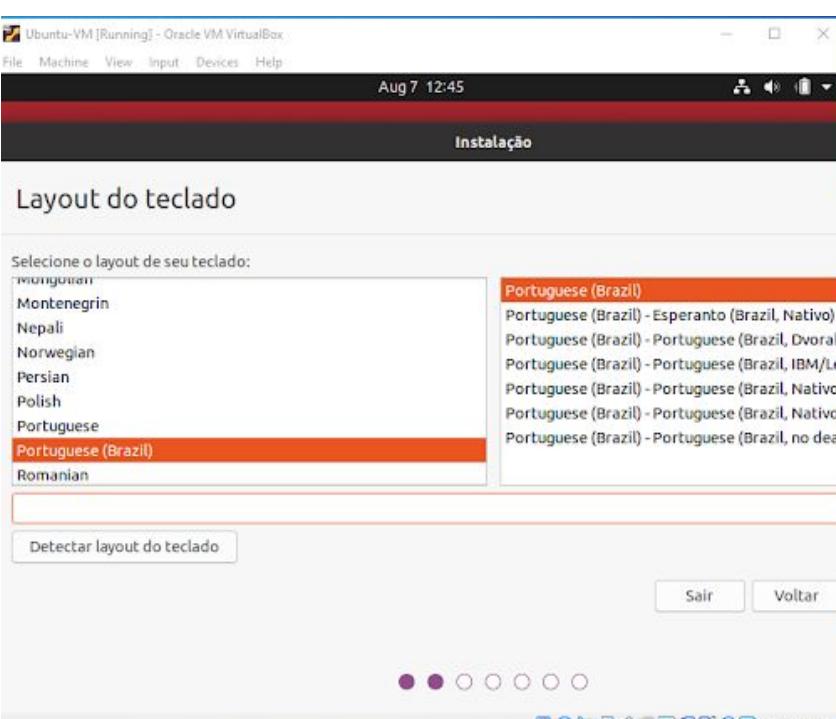
## 8. Instalar o Ubuntu na VM



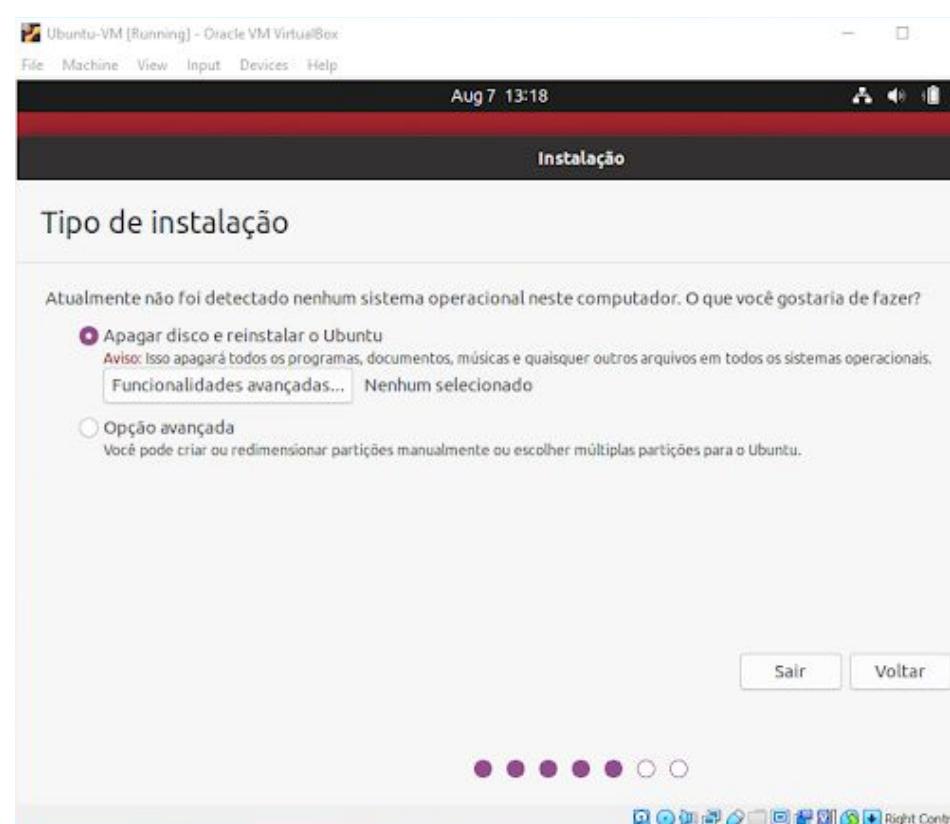
dois cliques



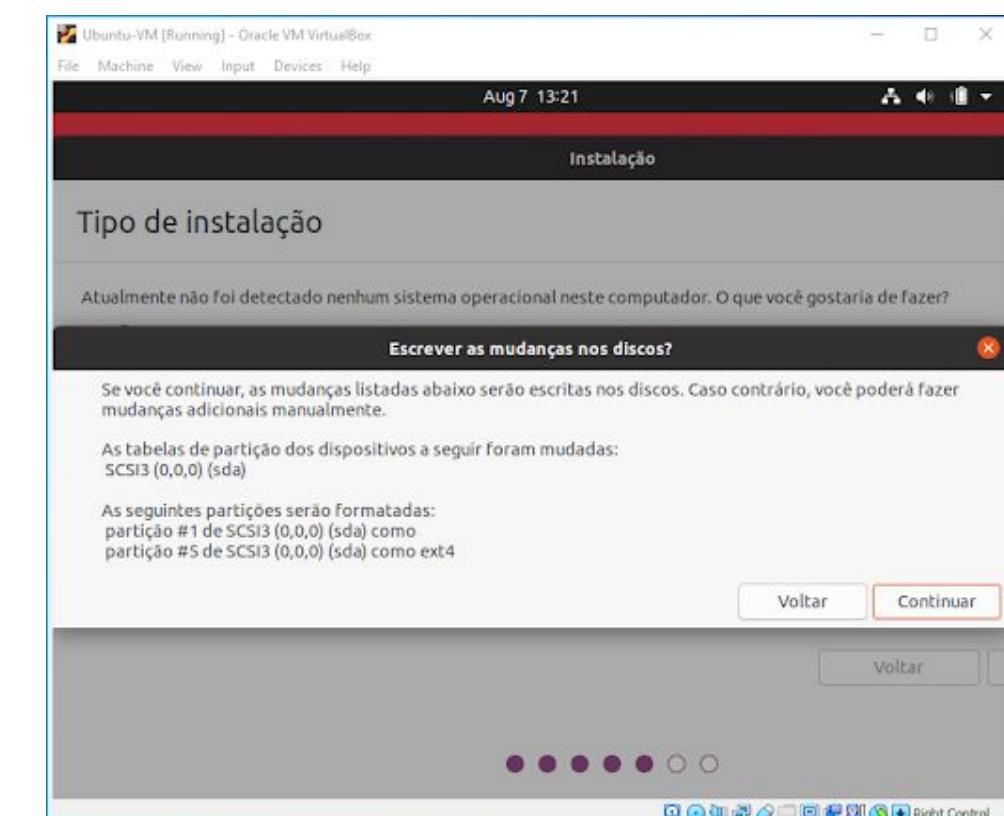
→



o botão 'continuar' pode  
estar escondido aqui.  
Acessá-lo via tecla 'tab'

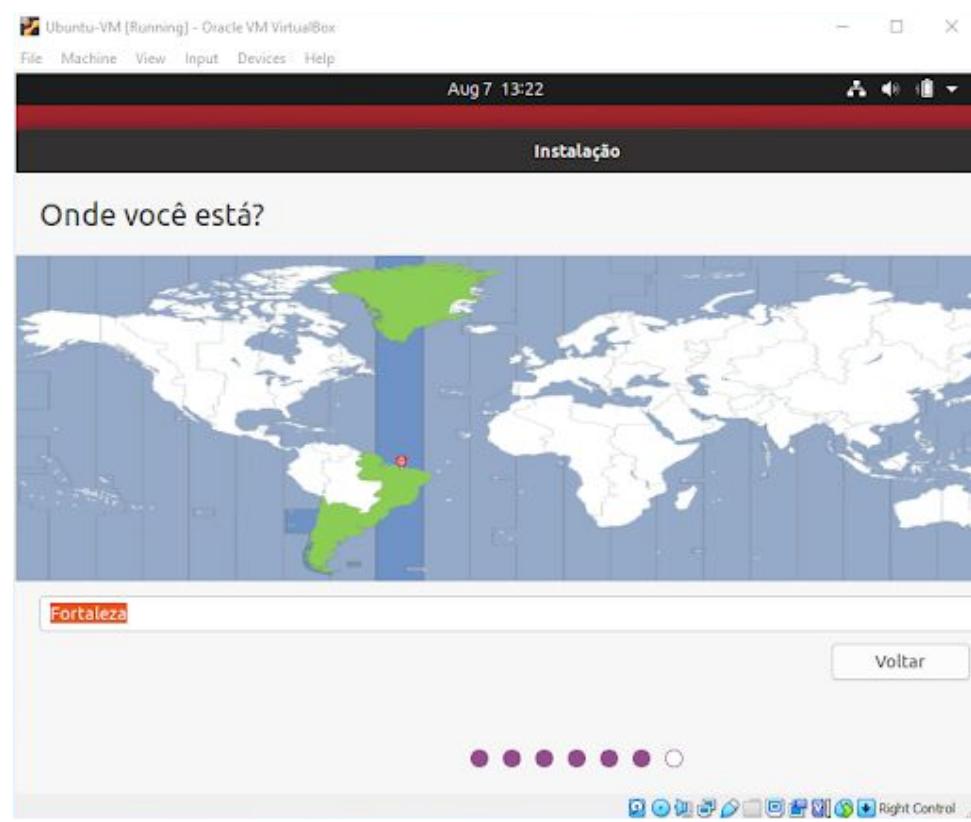


o botão 'continuar' pode  
estar escondido aqui.  
Acessá-lo via tecla 'tab'

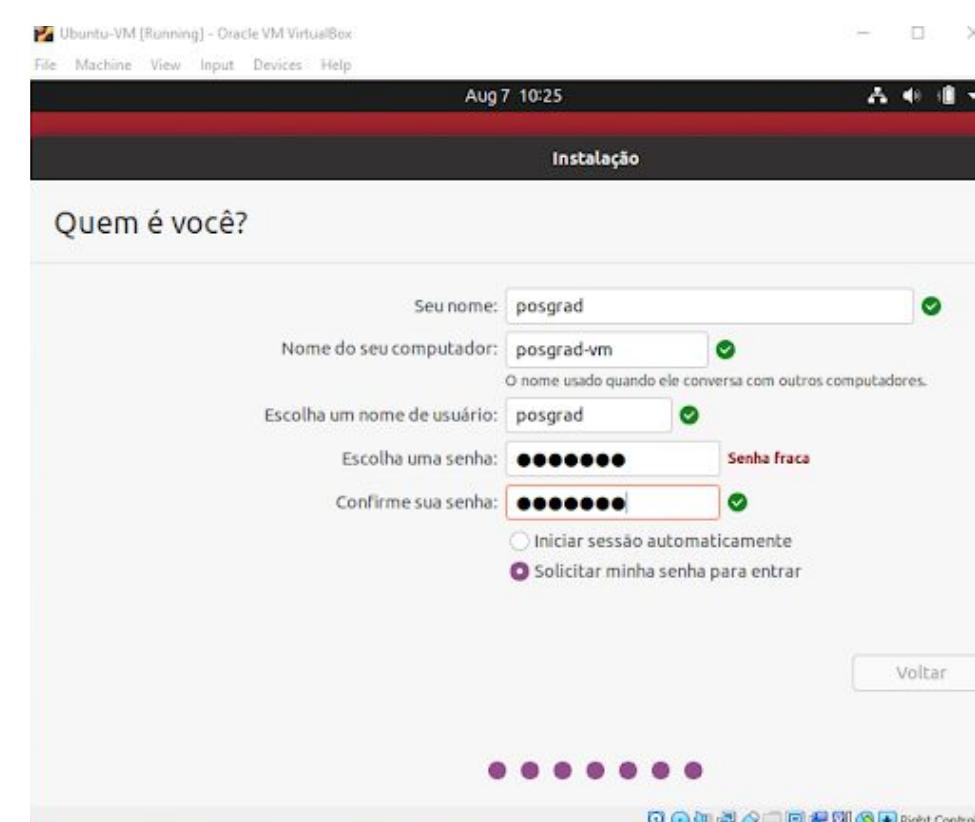


# Instalação do Ubuntu

## 8. Instalar o Ubuntu na VM

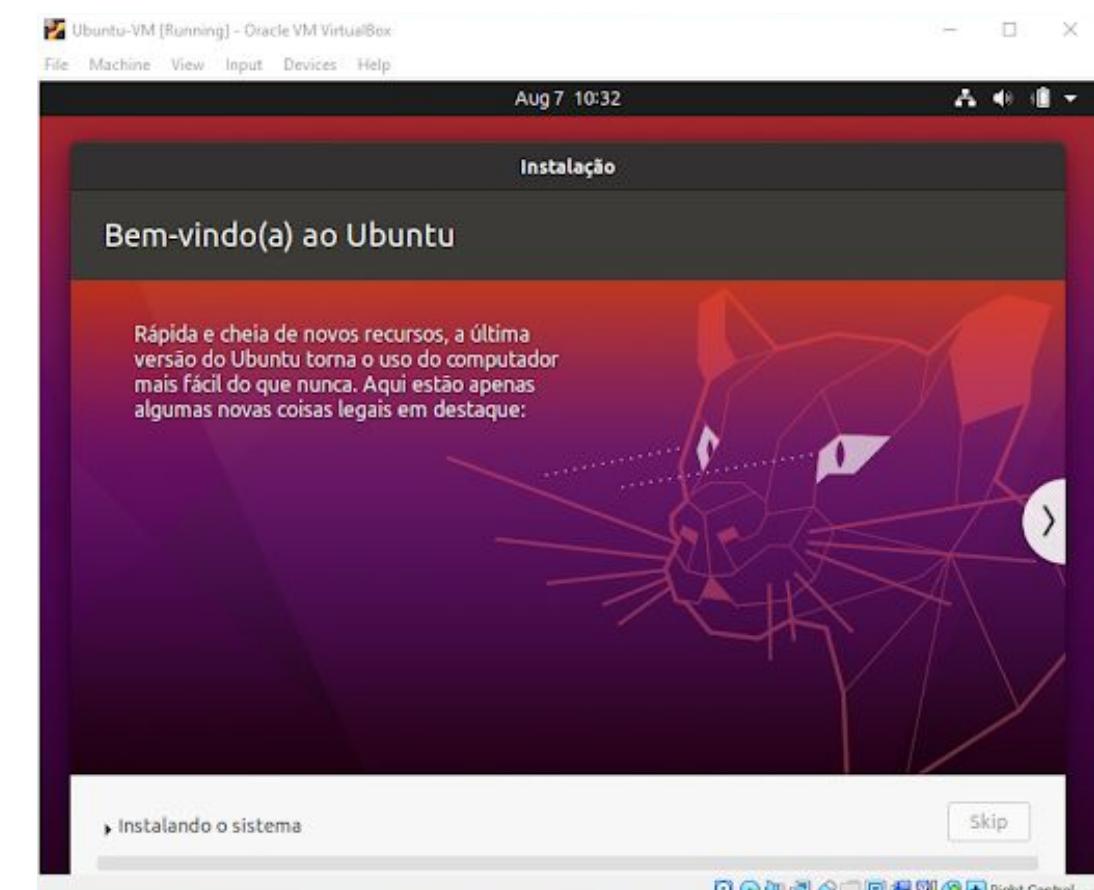


o botão 'continuar' pode  
estar escondido aqui.  
Acessá-lo via tecla 'tab'



o botão 'continuar' pode  
estar escondido aqui.  
Acessá-lo via tecla 'tab'

{ usuário: posgrad  
senha: posgrad



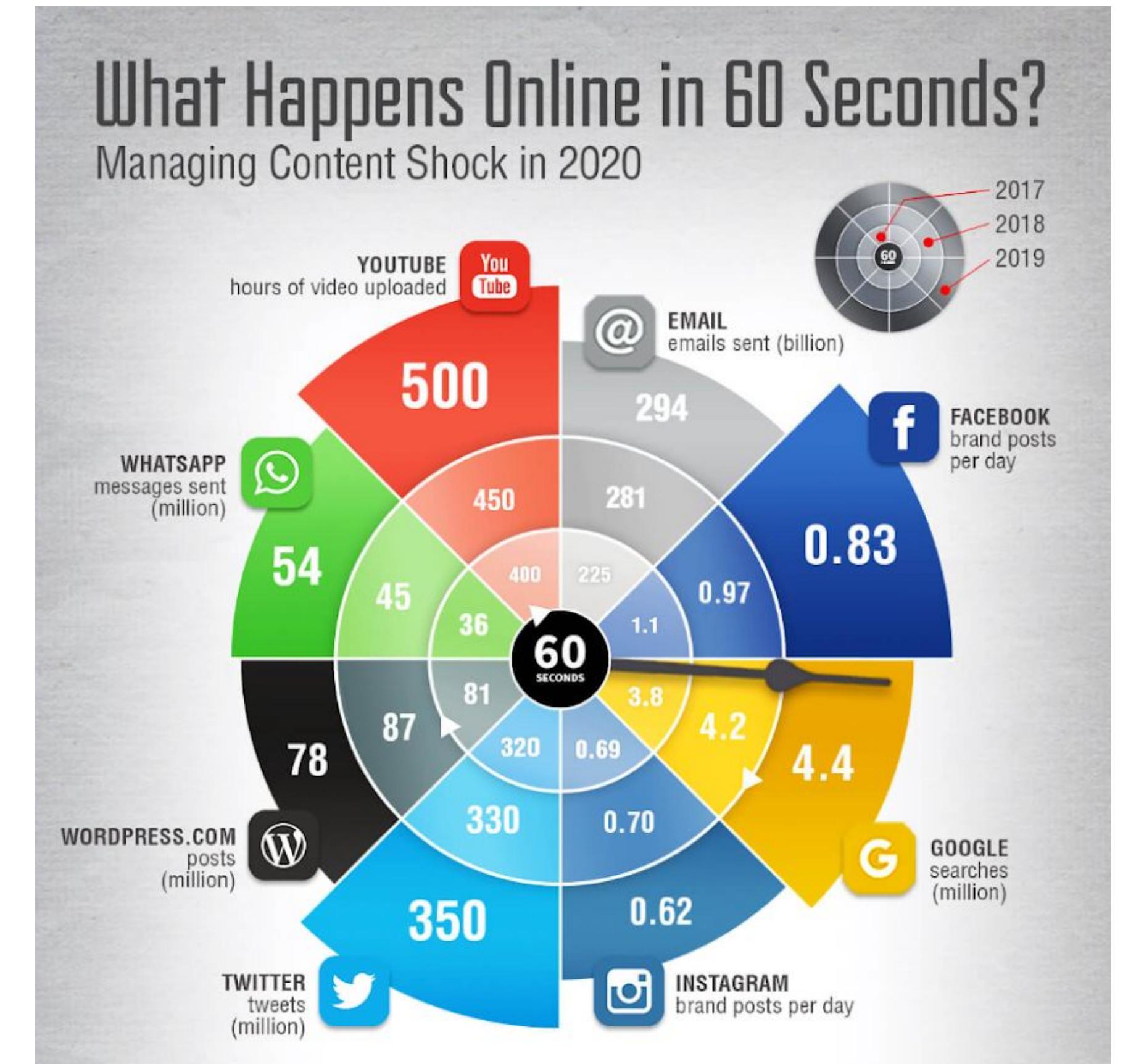
Deixar instalando...

# Introdução

# O que acontece em 60 segundos?

# Upload de 500h de vídeo no YouTube

54 milhões de  
mensagens no Whatsapp



# Big Data

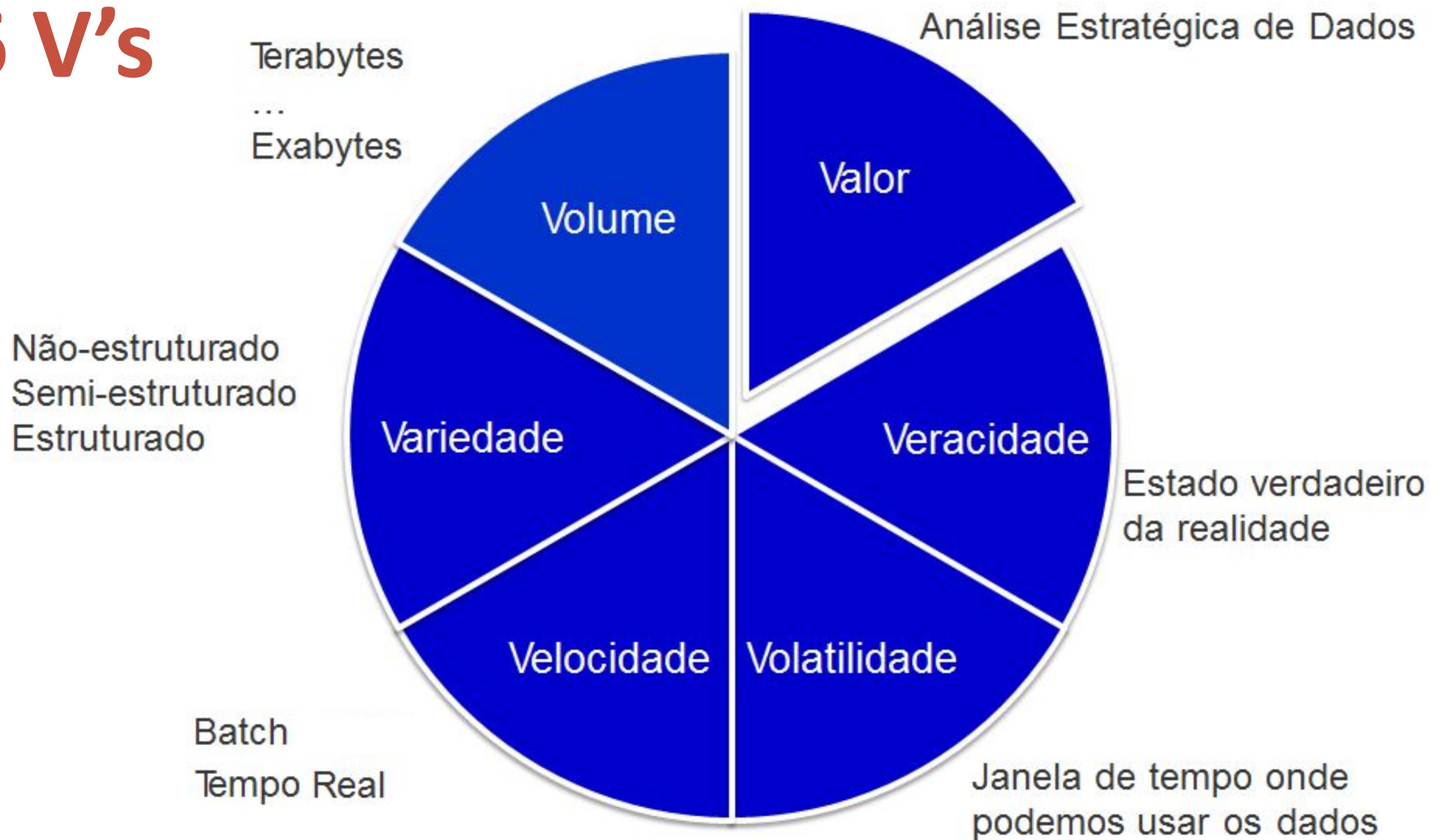
Big Data são dados que excedem o **armazenamento, o processamento e a capacidade dos sistemas convencionais**

Volume de dados muito grande

Dados são gerados rapidamente

Dados não se encaixam nas estruturas de arquiteturas de sistemas atuais

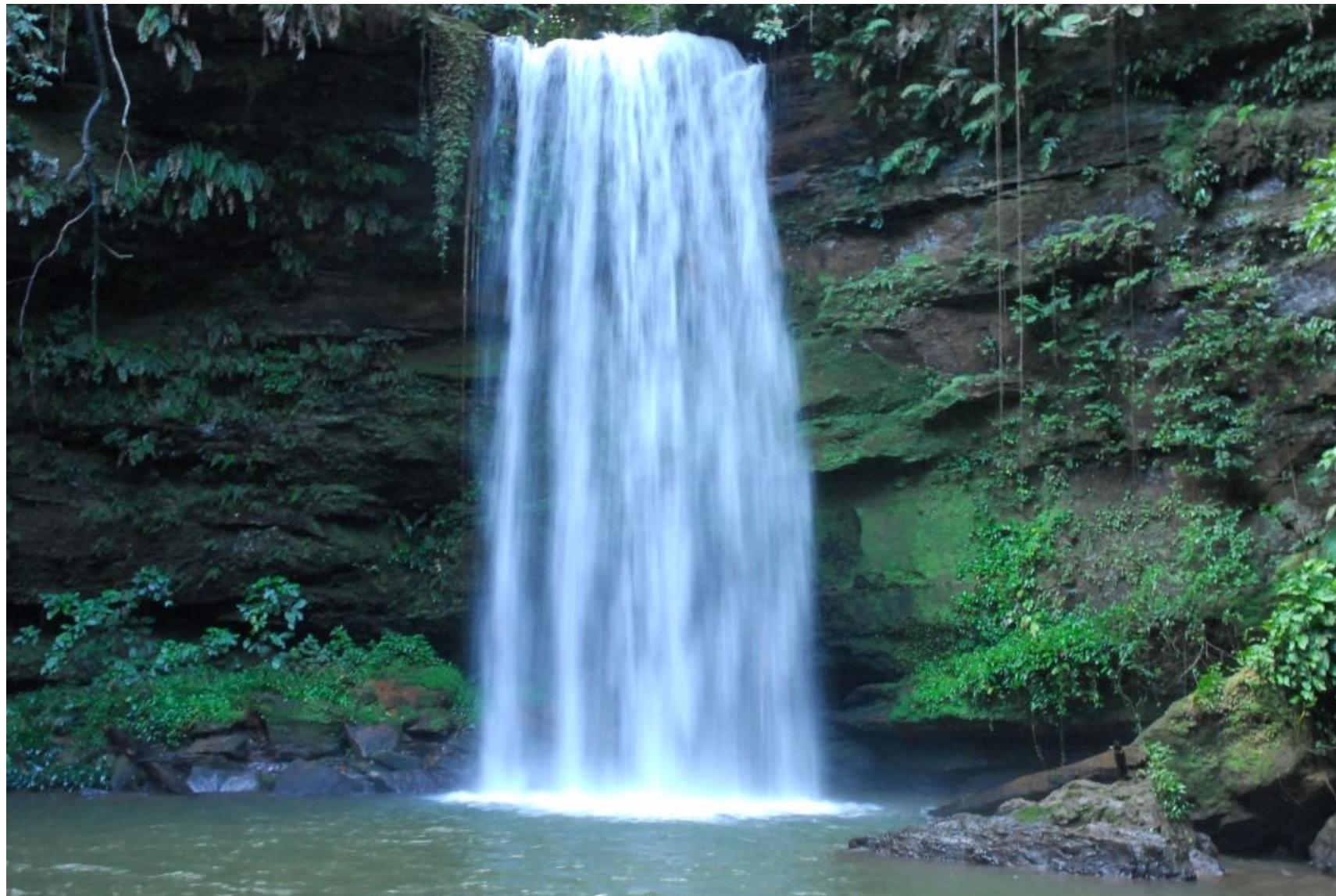
# 6 V's



# Streaming

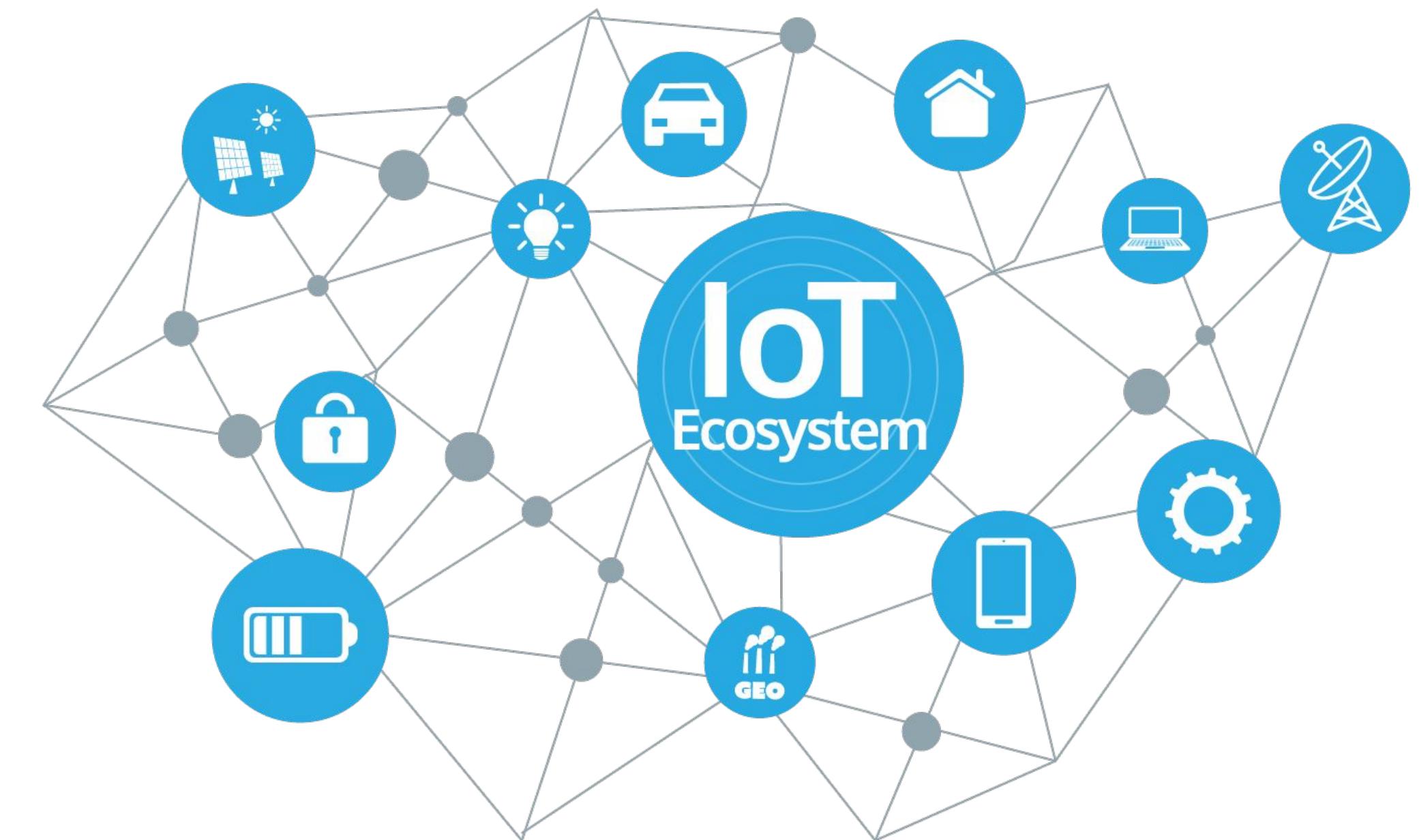
# Streaming

Fluxo contínuo (contínuo  $\neq$  constante).



# Streaming de dados

Fluxo contínuo de dados.



# Streaming de dados: Exemplos

- Sensores (IoT)
- Tráfego de rede
- Registros de call center
- Tendências em redes sociais
- Serviços de áudio e vídeo
- Análise de log
- Estatísticas de sites web



# Tipos de streaming de dados

Dados de texto: web, log

Dados relacionais: tabelas, transações

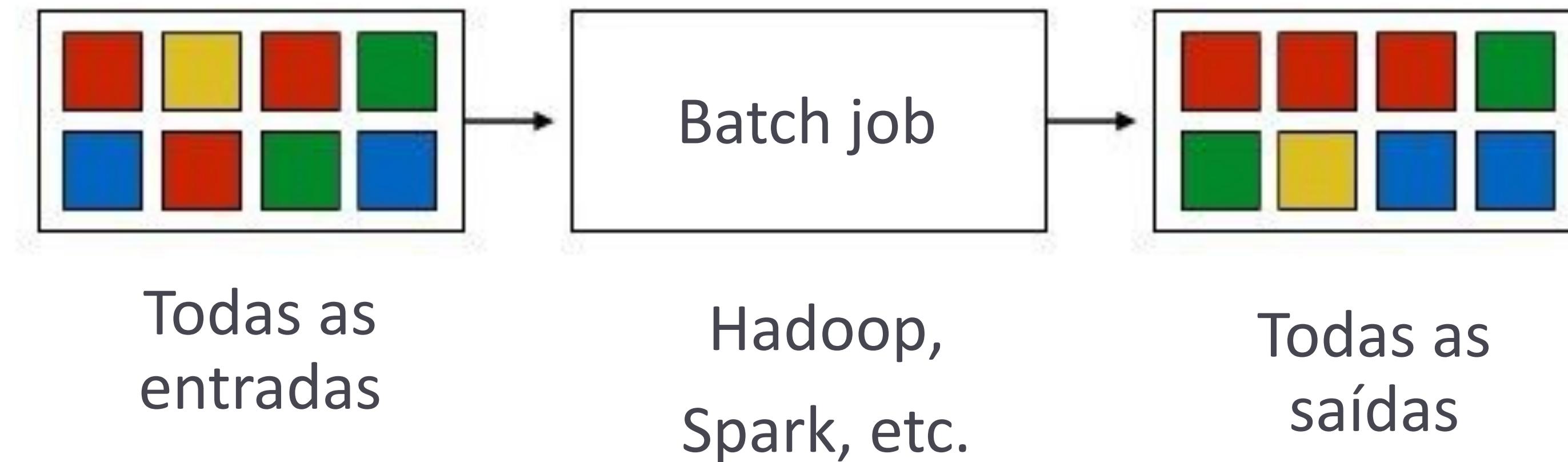
Dados semi-estruturados: XML, json

Dados em grafo: redes sociais

Dados de mobilidade: coordenadas geográficas x tempo

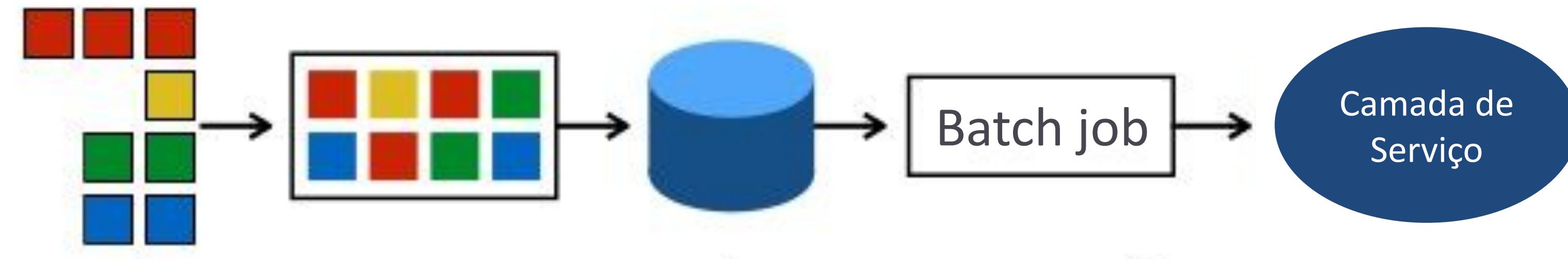
Etc.

# Processamento em Batch



# Processamento de Streaming

Em geral:



continuamente  
produzido

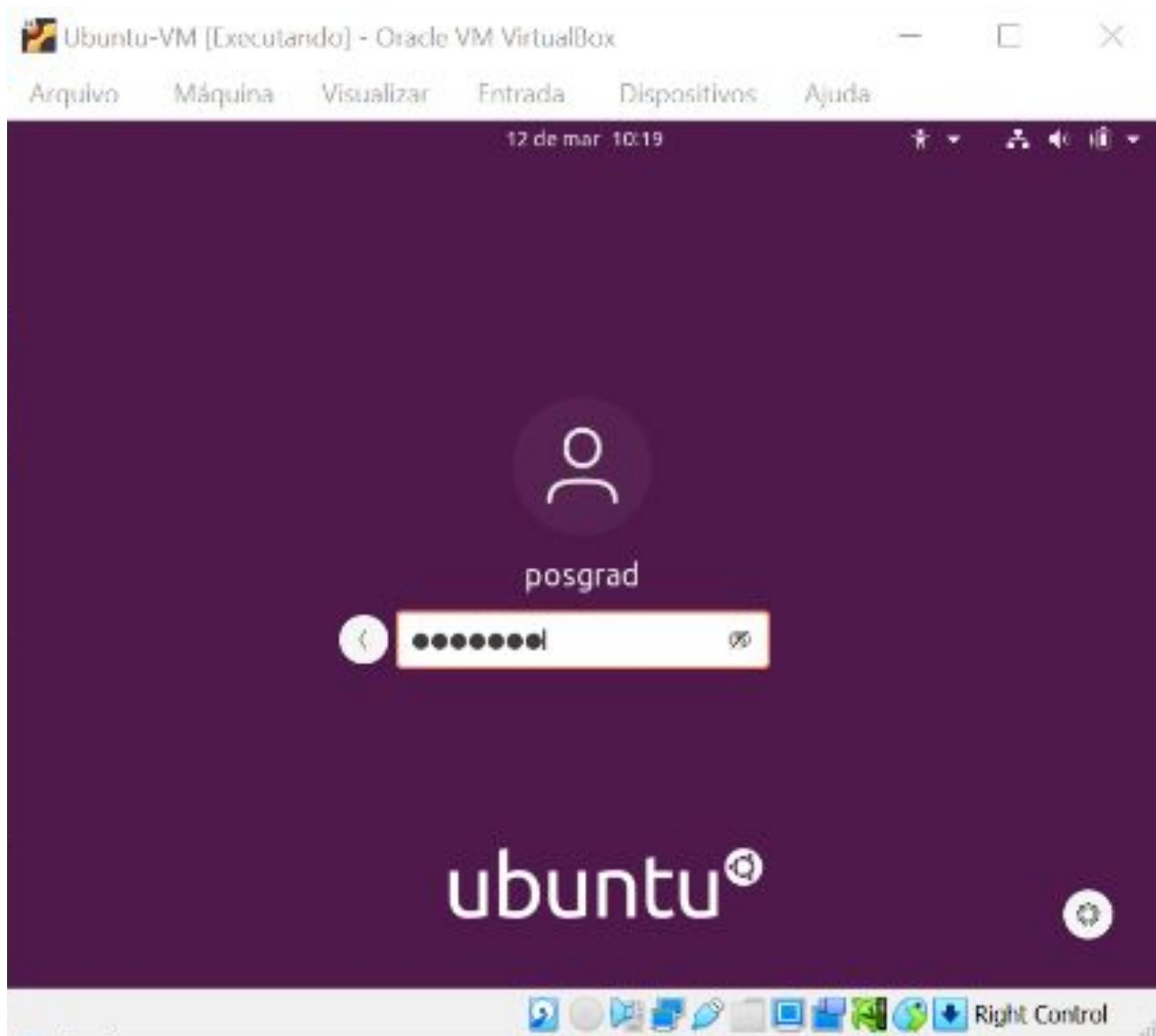
arquivos são  
streams finitos

periodicamente  
executado

Continuando  
nossa Setup...

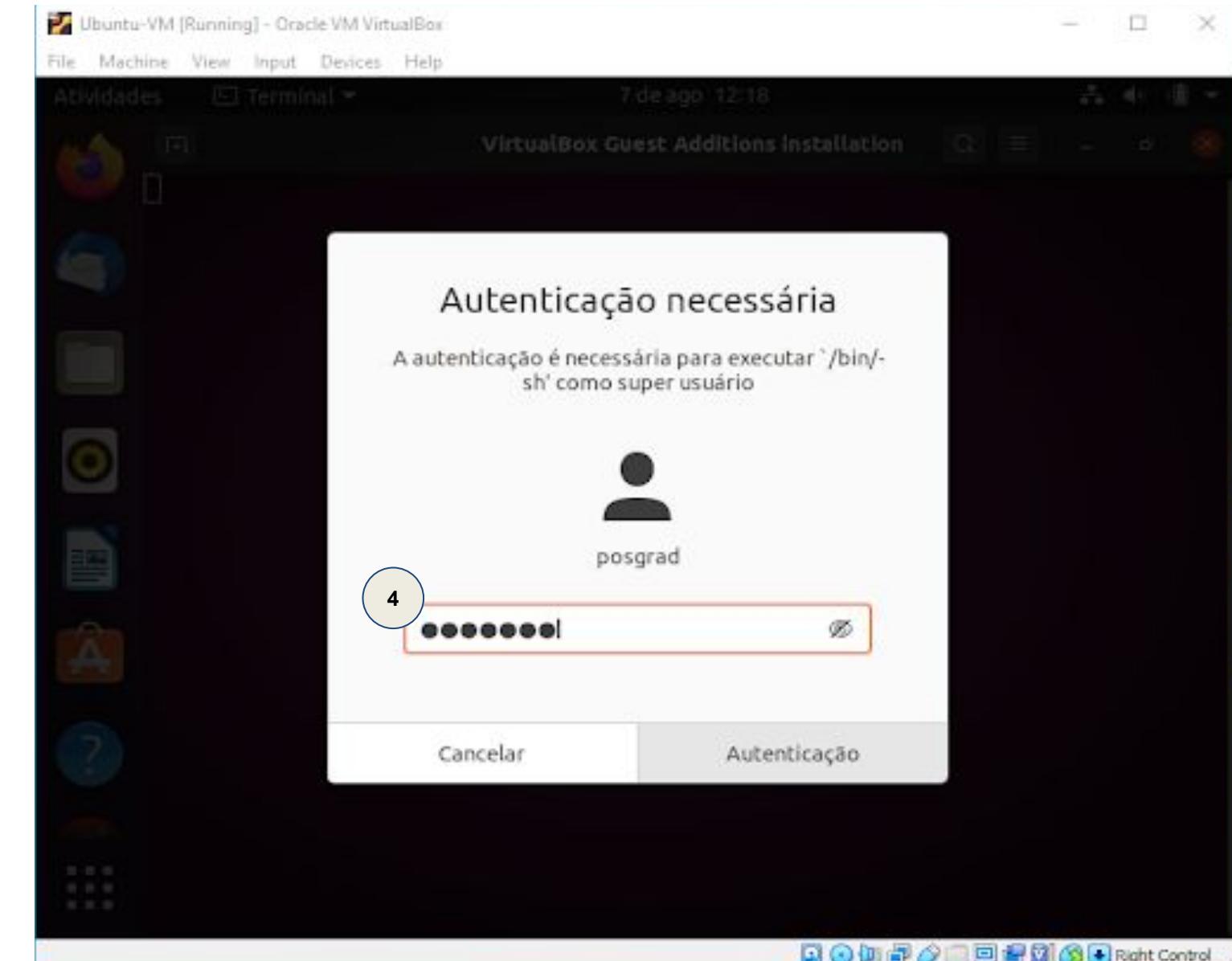
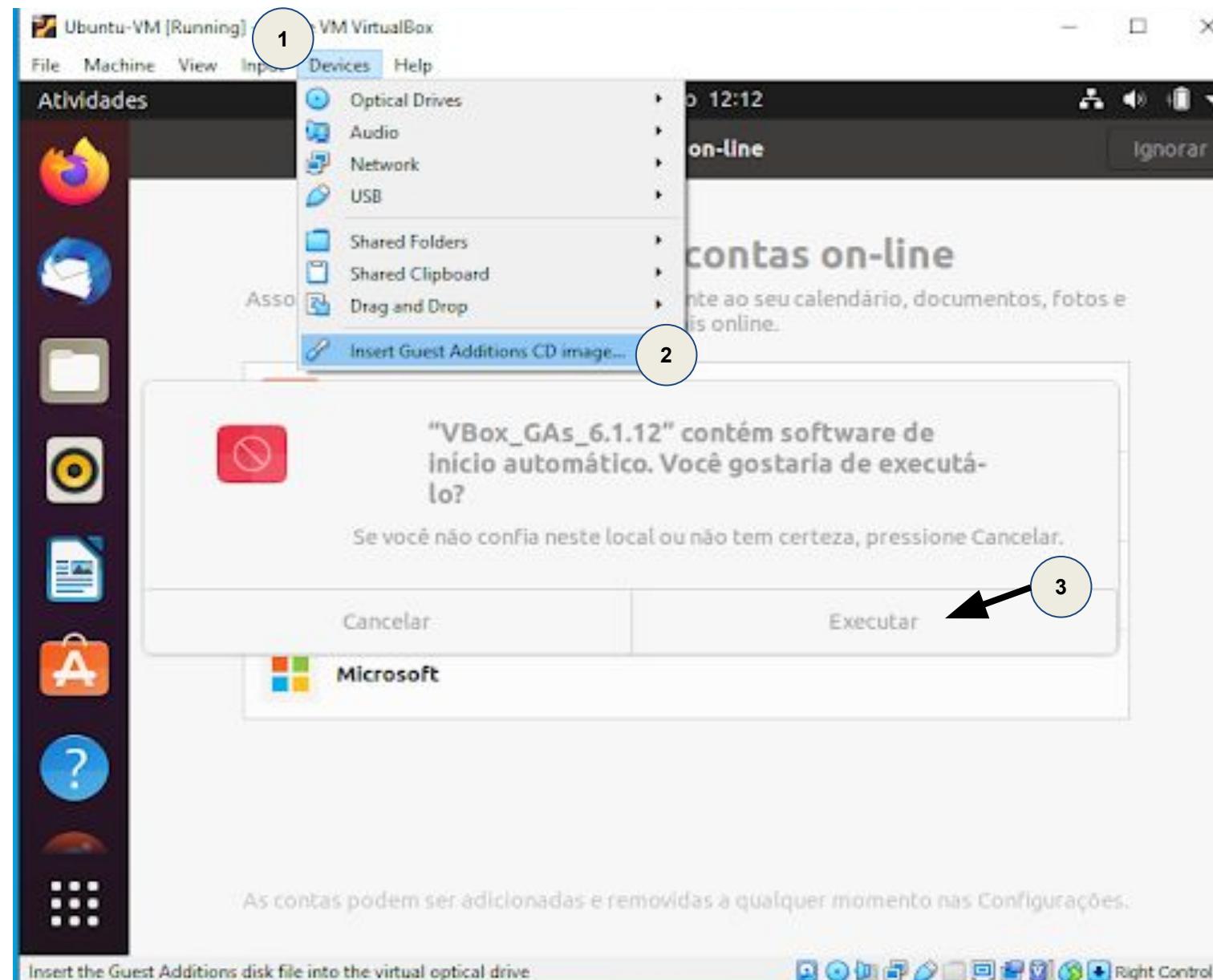
# Configuração da VM Ubuntu

## 9. Logar na VM



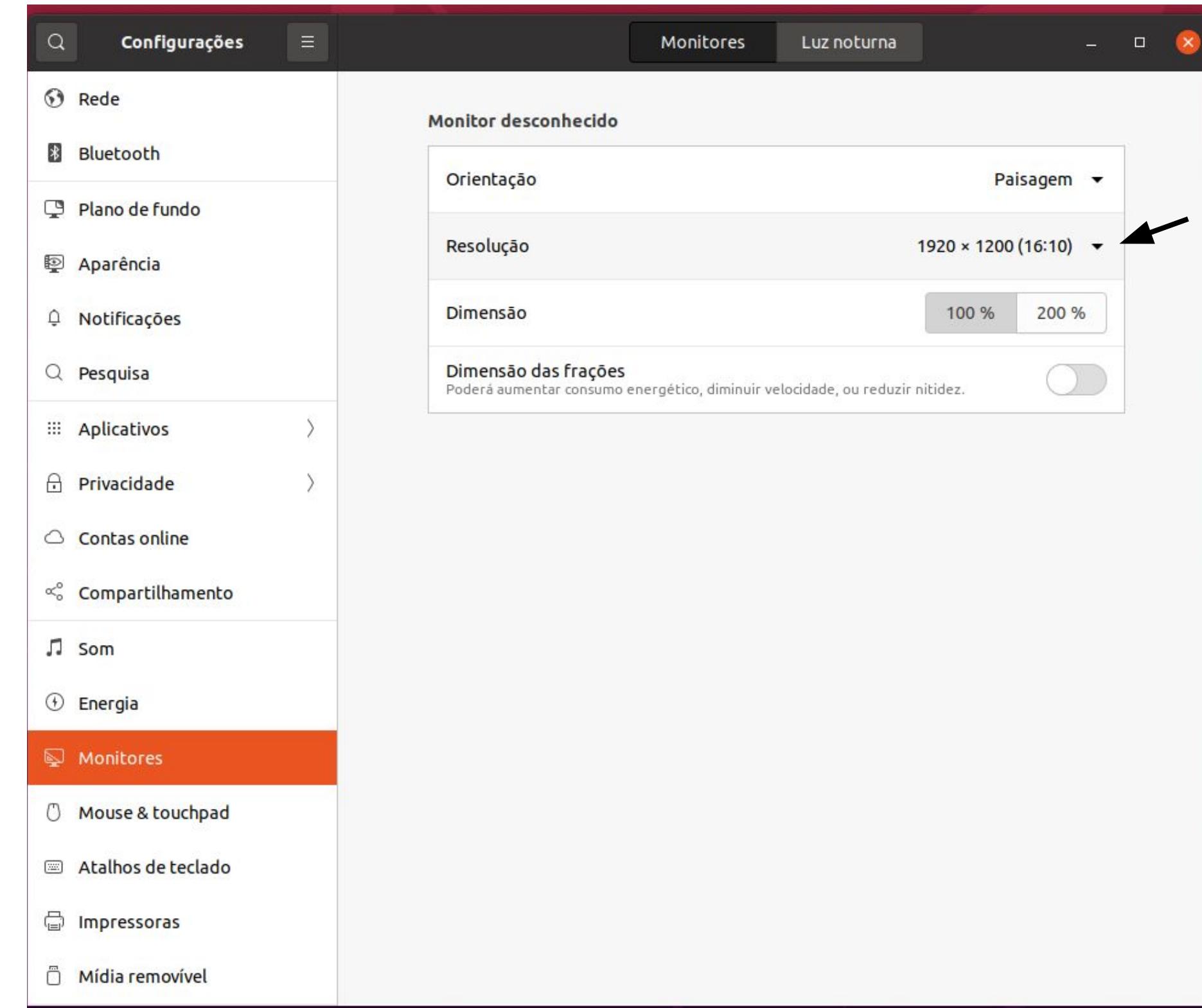
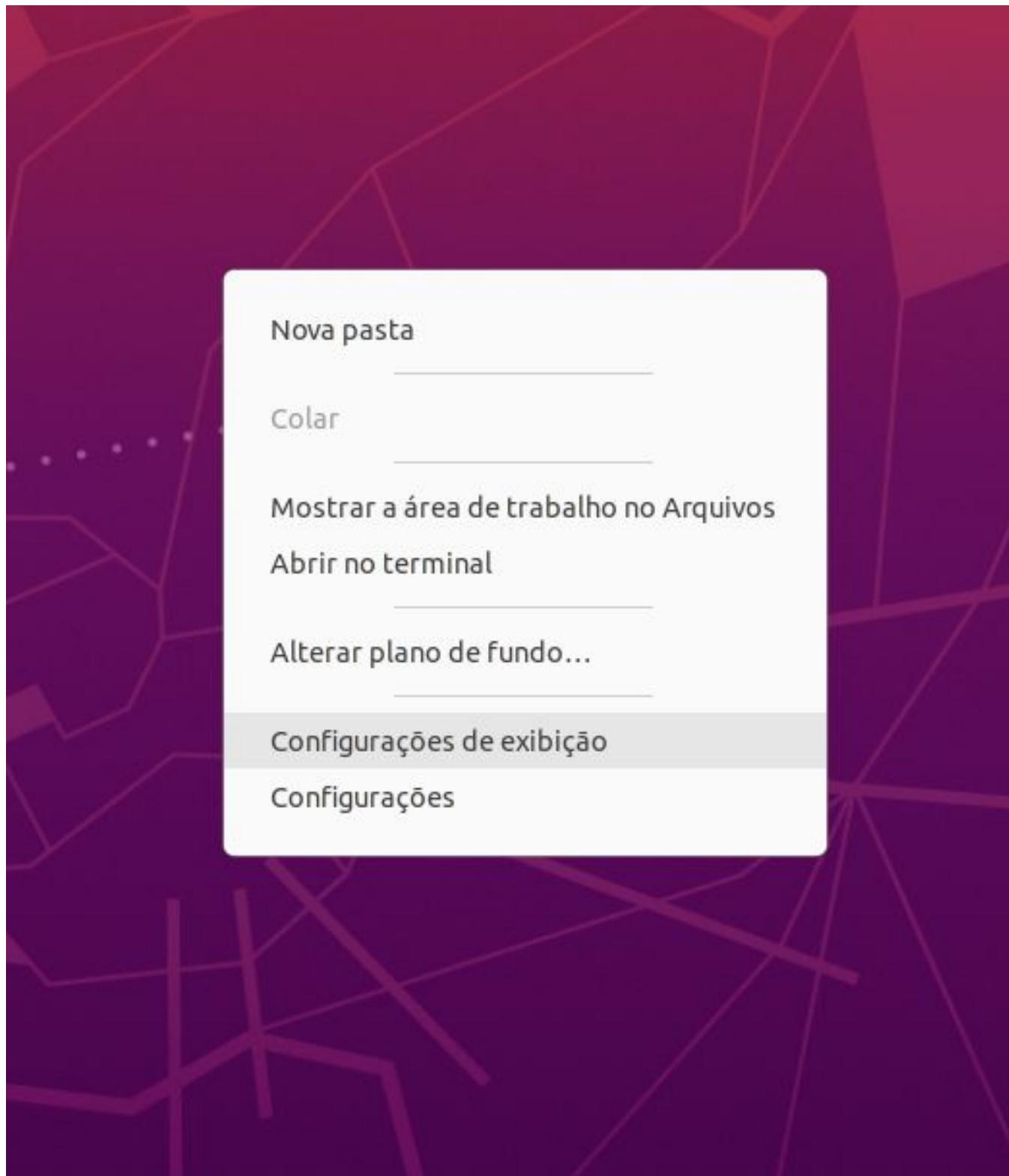
# Configuração da VM Ubuntu

## 10. Instalar add-ons (para melhor experiência com o Ubuntu)



# Configuração da VM Ubuntu

## 11. Ajustar resolução de tela



# Atualizando o Ubuntu

Abrir o terminal (Alt+F2 e digitar gnome-terminal)

Atualizar o Ubuntu

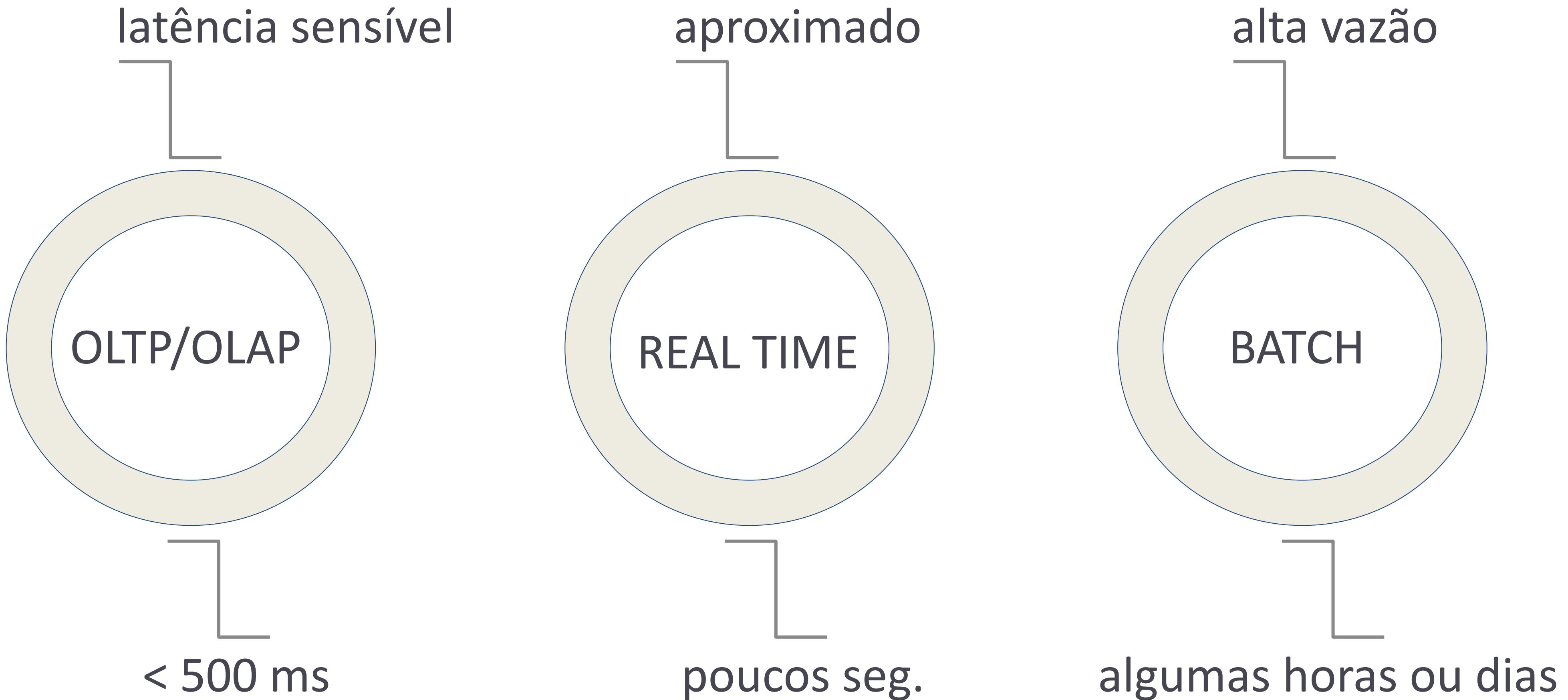
- sudo apt update
- sudo apt upgrade



# O que é tempo real?

Milissegundos, segundos, minutos?

# O que é Tempo Real?



# O que é Tempo Real?

## REAL TIME TRENDS



Emerging break out  
trends in Twitter (in the  
form #hashtags)

## REAL TIME CONVERSATIONS



Real time sports  
conversations related  
with a topic (recent goal  
or touchdown)

## REAL TIME RECOMMENDATIONS



Real time product  
recommendations based  
on your behavior &  
profile

## REAL TIME SEARCH



Real time search of  
tweets with a budget <  
200 ms

Fonte: Real-Time Analytics with Apache Storm - <https://www.udacity.com/course/ud381>

# Problemas em streaming

1. Como obter dados a partir de várias fontes em tempo real?
2. Como processar esses dados?



Finalizando  
nossa Setup...

# Instalando algumas libs

## Instalar o curl

```
➤ sudo apt install curl
```

## Instalar o VSCode

```
➤ sudo snap install --classic code
```

## Instalar o Python

```
➤ sudo apt install python3-pip
```

## Instalar o Java

```
➤ sudo apt install default-jdk
```

Deixar instalando...

# Apache Kafka

# Apache Kafka

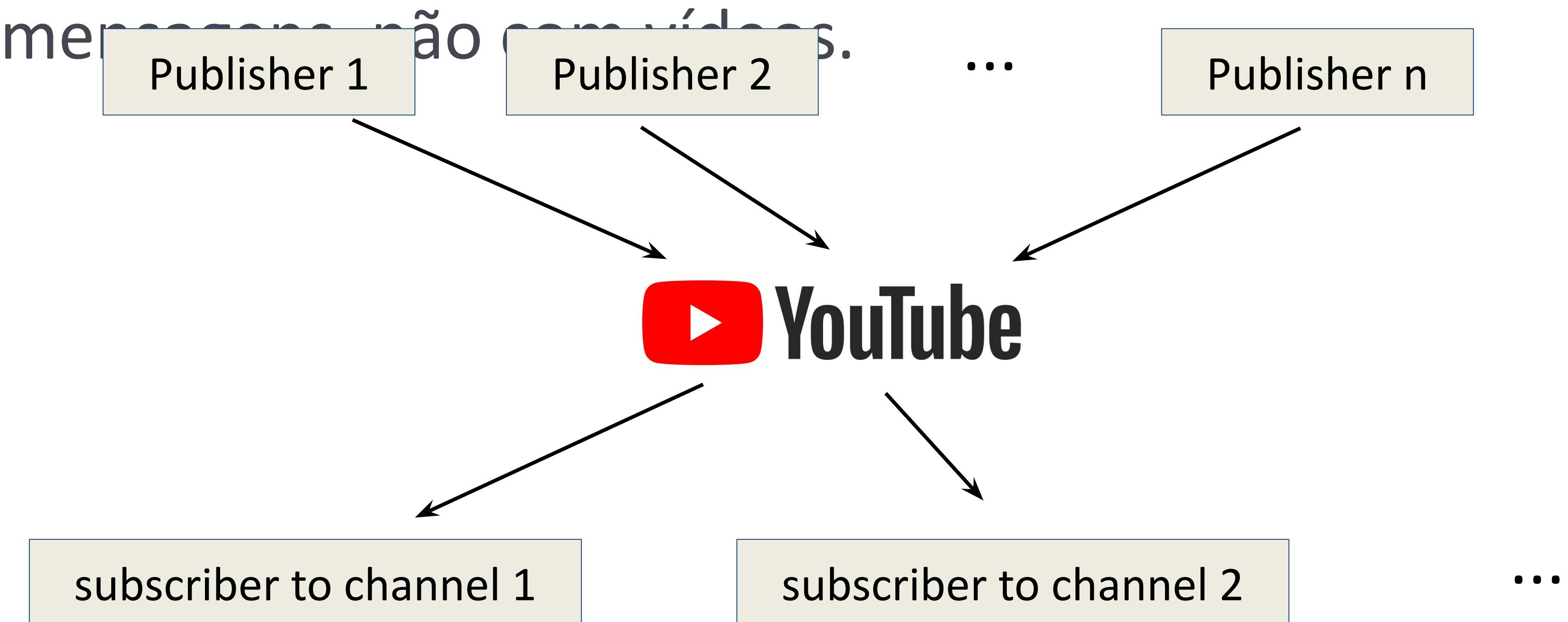
- Sistema de mensagens
  - Distribuído
  - Com alta vazão (*throughput*)
  - De geração (publicação) e leitura (sub-inscrição)
- Principais casos de uso:
  - Agregação de log
  - Mover/transformar conjuntos de dados em tempo real
  - Monitoramento

# Apache Kafka

- Originalmente desenvolvido pelo LinkedIn.
- Implementado em scala/Java.
- *Producers & Consumers.*
- Mensagens são associadas a tópicos, os quais representam um stream específico.
  - Logs web
  - Dados de sensores
- *Consumers* se inscrevem em um ou mais tópicos.

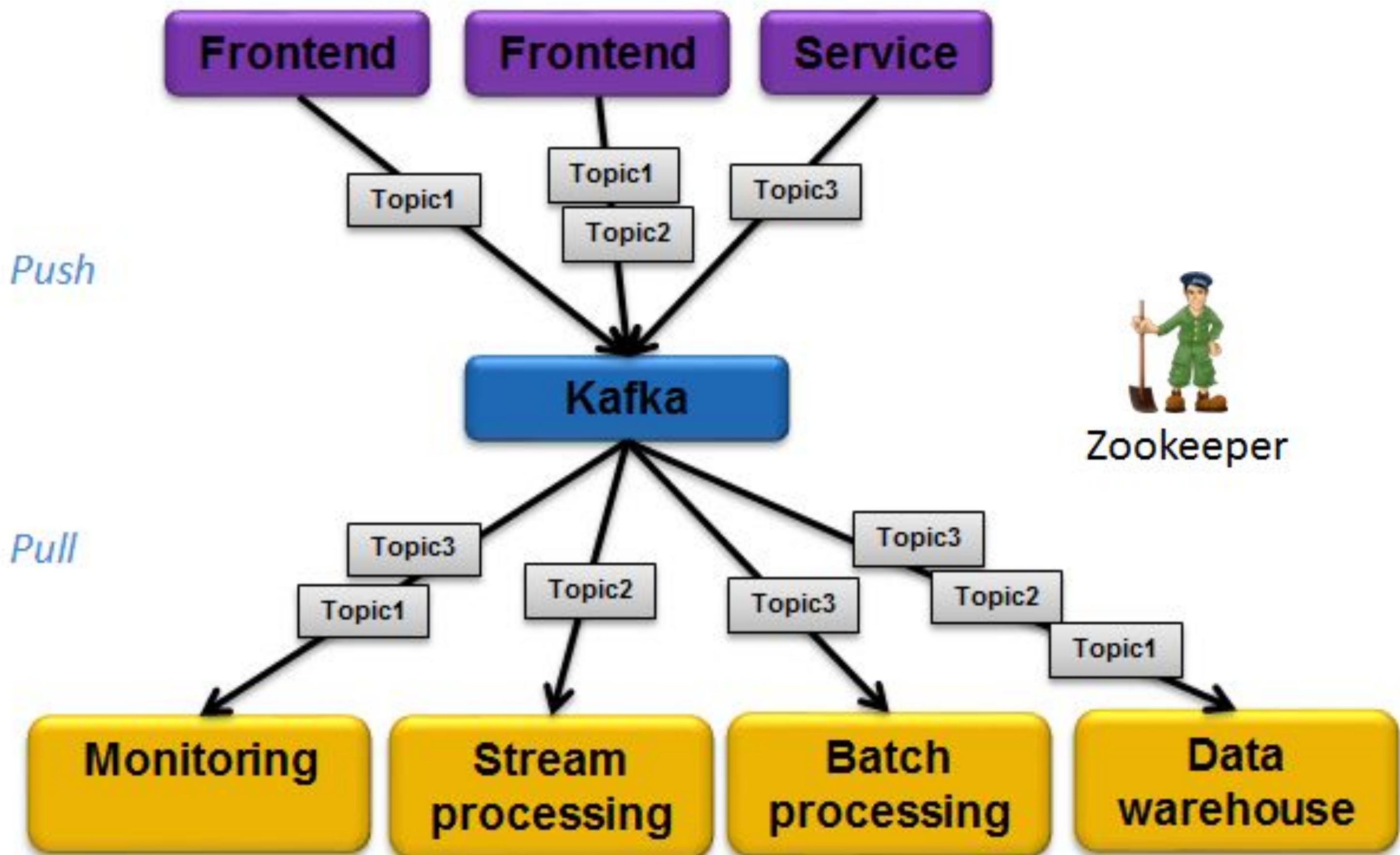
# Publisher-subscriber system

- Kafka pode ser visto como um sistema publisher/subscriber, como o Youtube, mas com mensagens não como vídeos.



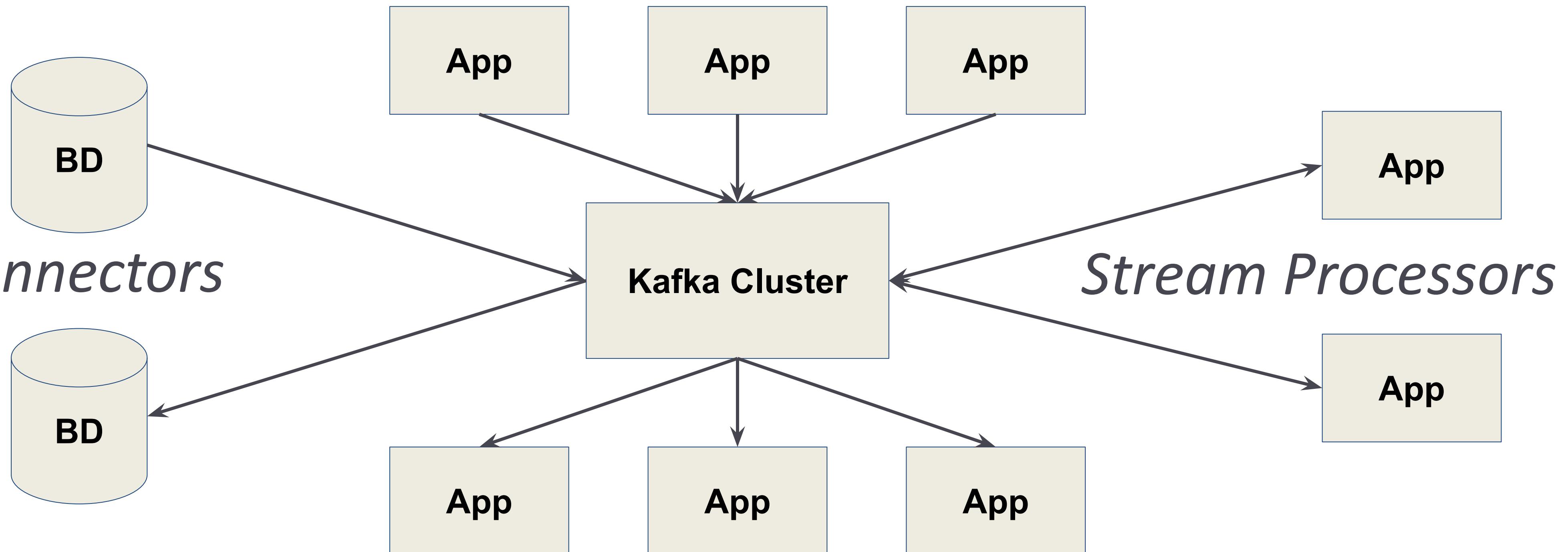
# Kafka: conceitos

Producers



# Kafka: arquitetura

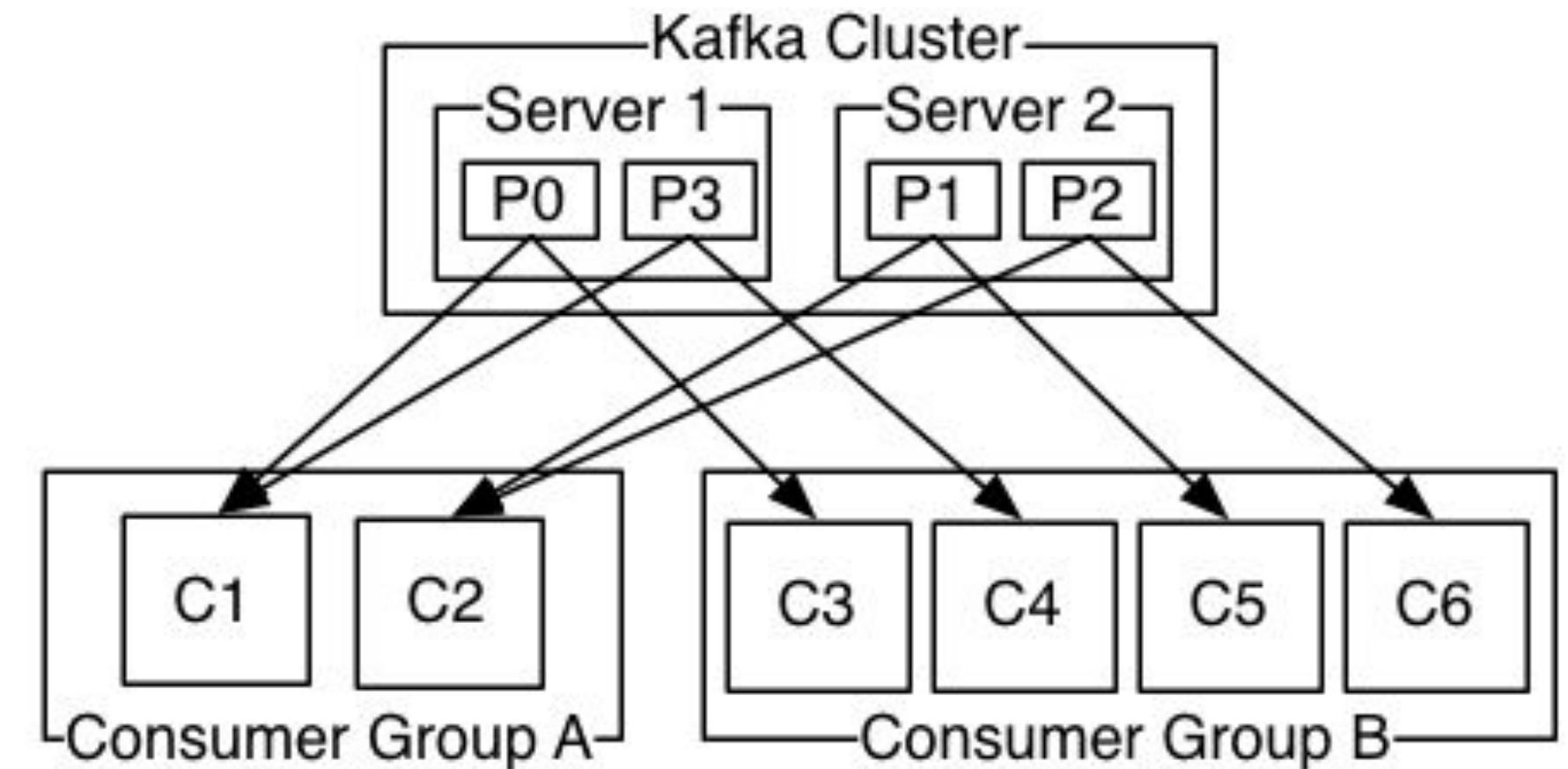
*Producers*



*Consumers*

# Kafka: escalabilidade

- Kafka pode ser distribuído entre muitos processos em vários servidores.
- *Consumers* também podem ser distribuídos.
- Tolerante a falhas.



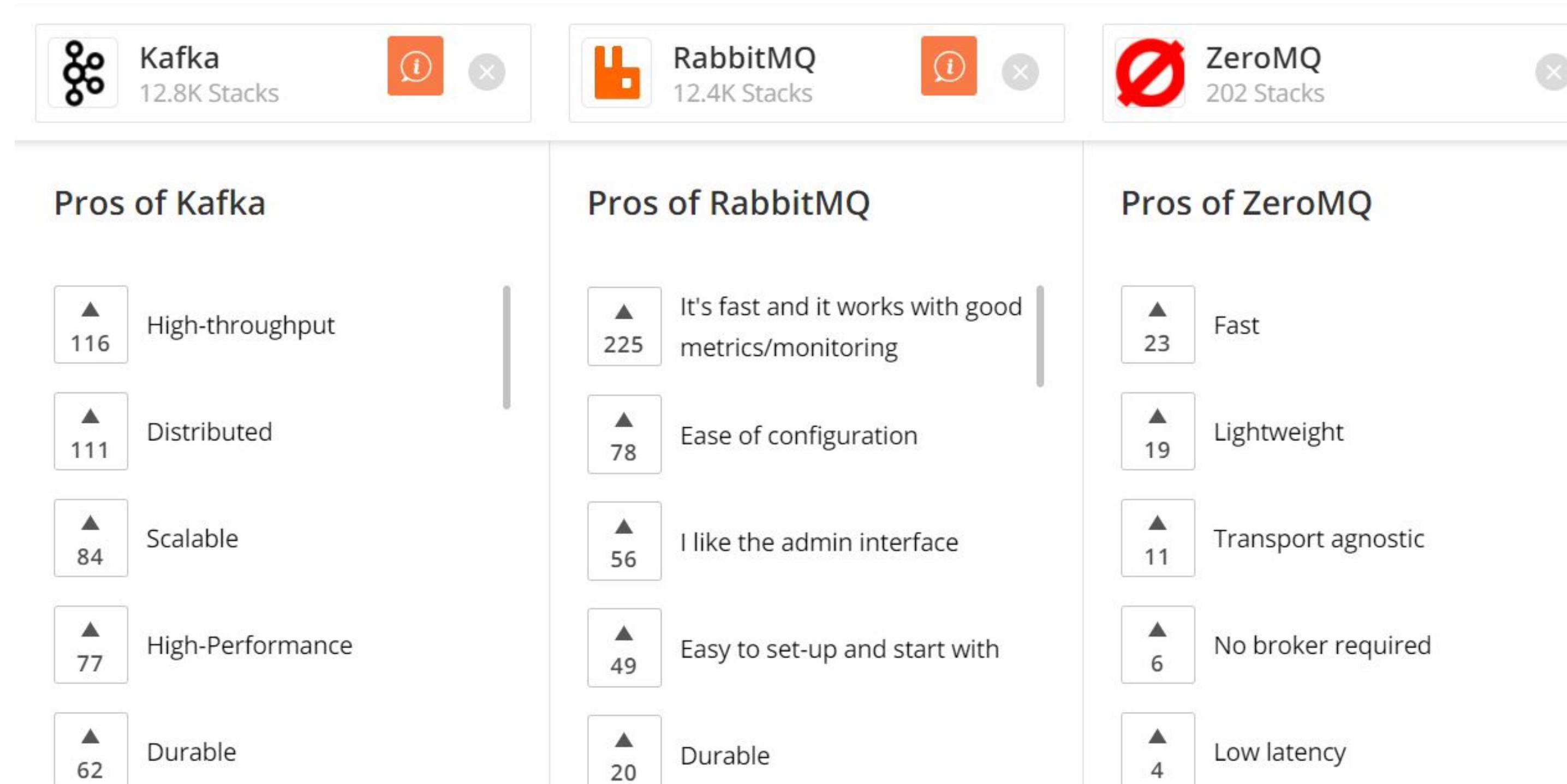
Fonte: <https://kafka.apache.org/intro.html>

# Kafka: pontos a considerar

- Simples sistema de mensagens, não de processamento.
- Não vive sem o **Zookeeper**, o qual pode se tornar um gargalo quando o número de tópicos/partições é muito grande ( $>>10000$ ).

# Kafka: quando comparado a outros concorrentes

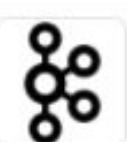
## Prós



Fonte: <https://stackshare.io/stackups/kafka-vs-rabbitmq-vs-zeromq>

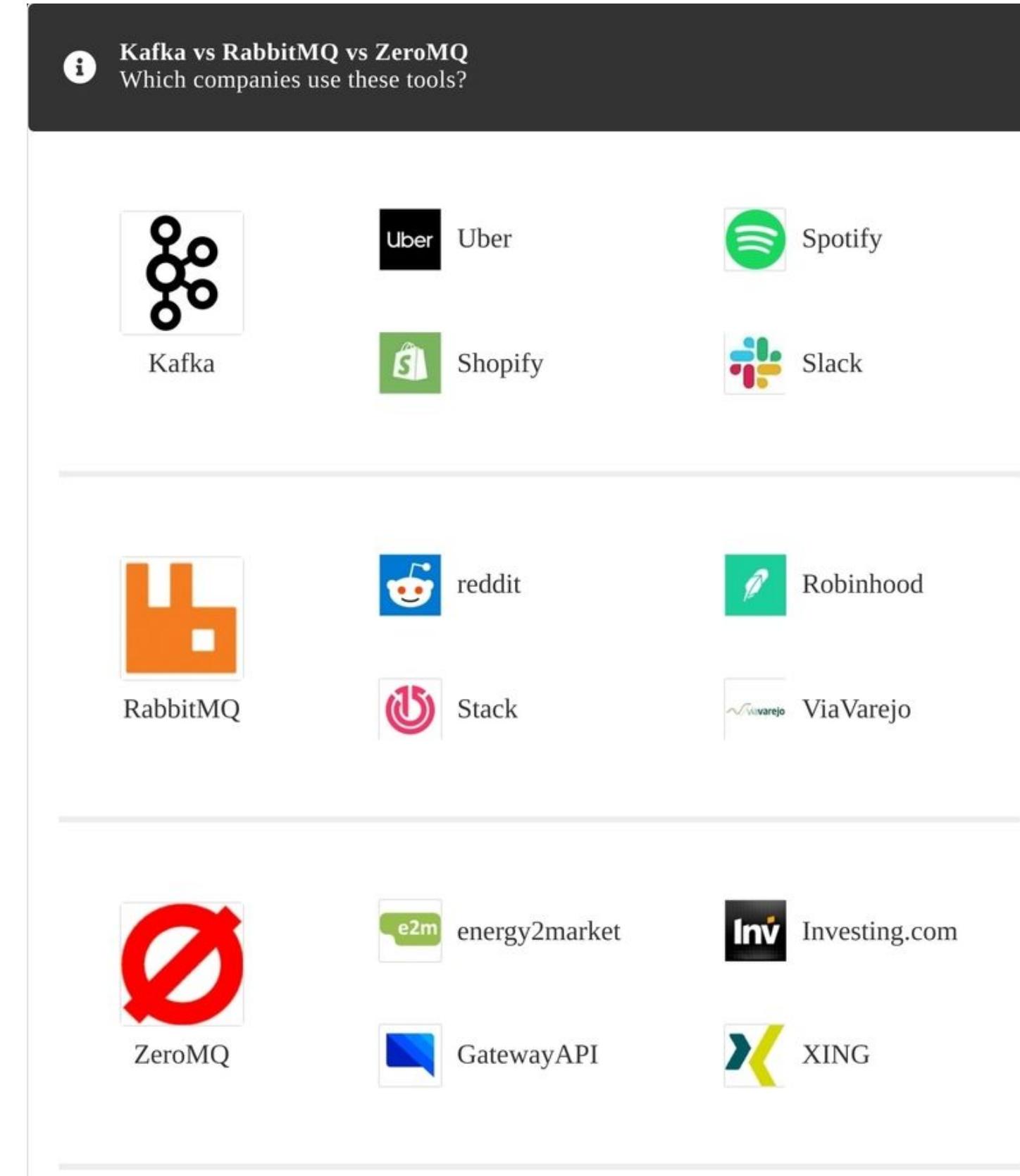
# Kafka: quando comparado a outros concorrentes

## Contras

 Kafka 12.8K Stacks	 RabbitMQ 12.4K Stacks	 ZeroMQ 202 Stacks
<h3>Cons of Kafka</h3> <ul style="list-style-type: none"><li>▲ 26 Non-Java clients are second-class citizens</li><li>▲ 25 Needs Zookeeper</li><li>▲ 7 Operational difficulties</li><li>▲ 1 Terrible Packaging</li></ul>	<h3>Cons of RabbitMQ</h3> <ul style="list-style-type: none"><li>▲ 9 Too complicated cluster/HA config and management</li><li>▲ 6 Needs Erlang runtime. Need ops good with Erlang runtime</li><li>▲ 5 Configuration must be done first, not by your code</li><li>▲ 4 Slow</li></ul>	<h3>Cons of ZeroMQ</h3> <ul style="list-style-type: none"><li>▲ 5 No message durability</li><li>▲ 3 Not a very reliable system - message delivery wise</li><li>▲ 1 M x N problem with M producers and N consumers</li></ul>

Fonte: <https://stackshare.io/stackups/kafka-vs-rabbitmq-vs-zeromq>

# Kafka: quando comparado a outros concorrentes



Fonte: <https://stackshare.io/stackups/kafka-vs-rabbitmq-vs-zeromq>

# Instalando o Apache Kafka

Realizar o download do Apache Kafka:

```
➤ curl  
http://ftp.unicamp.br/pub/apache/kafka/2.7.0/ka  
fka_2.12-2.7.0.tgz -o ~/Downloads/kafka.tgz
```

```
% Total    % Received % Xferd  Average Speed   Time     Time     Time  Current  
          Dload  Upload   Total   Spent   Left  Speed  
100 62.6M  100 62.6M    0      0  528k      0  0:02:01  0:02:01 --:--:-- 345k  
posgrad@posgrad-vm:~$
```

# Instalando o Apache Kafka

Criar um diretório chamado kafka e entrar nele:

- mkdir kafka
- cd kafka

Extrair os arquivos que estão na pasta download para o diretório criado:

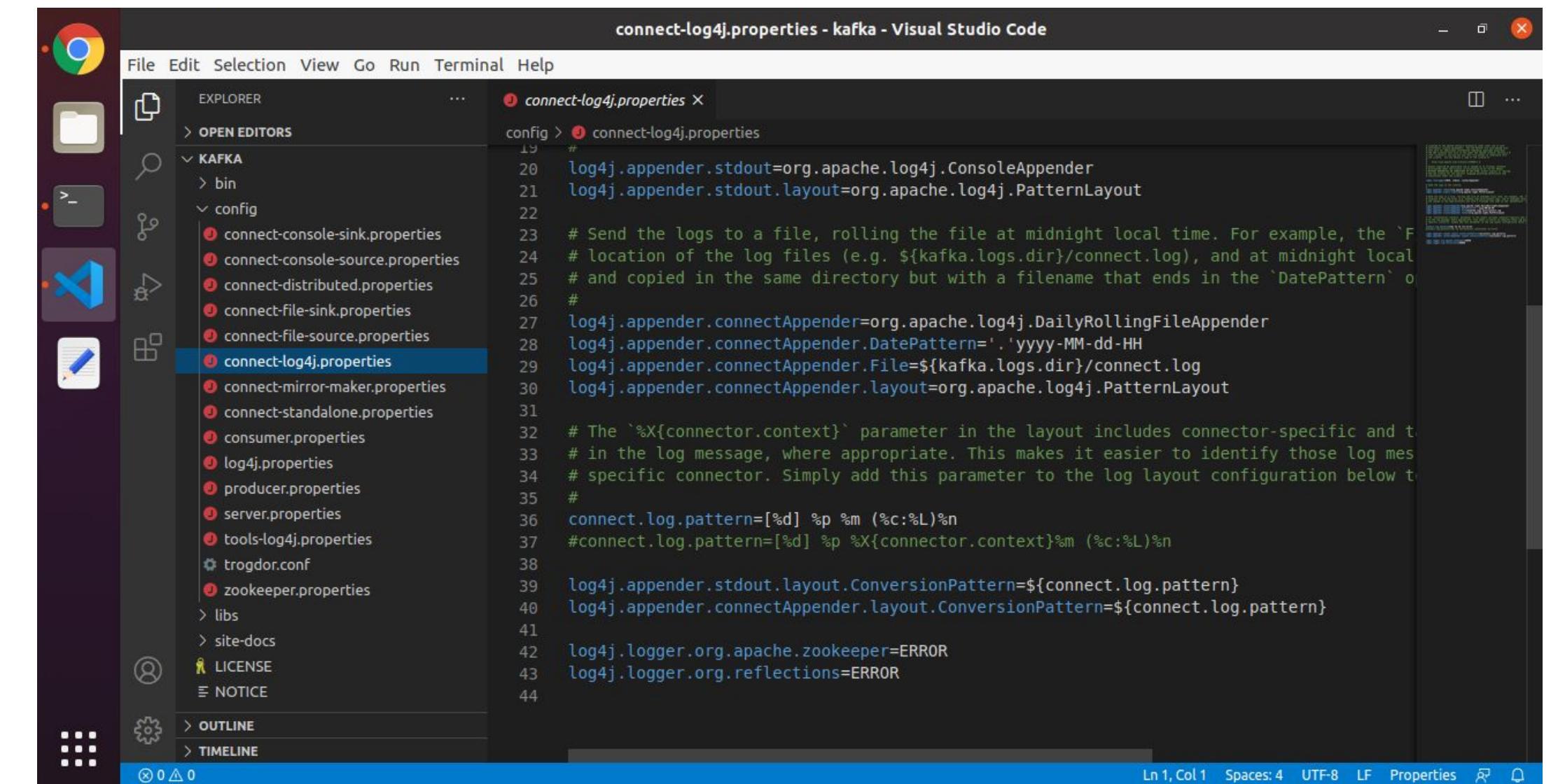
- tar -xvzf ~/Downloads/kafka.tgz --strip 1

# Visualizando os arquivos do Kafka

Abrir o VS Code

Clicar em Open Folder

Selecionar a pasta kafka



The screenshot shows the Visual Studio Code interface with the title bar "connect-log4j.properties - kafka - Visual Studio Code". The left sidebar displays a file tree under the "KAFKA" folder, with "connect-log4j.properties" selected. The main editor area shows the content of the "connect-log4j.properties" file:

```
connect-log4j.properties - kafka - Visual Studio Code

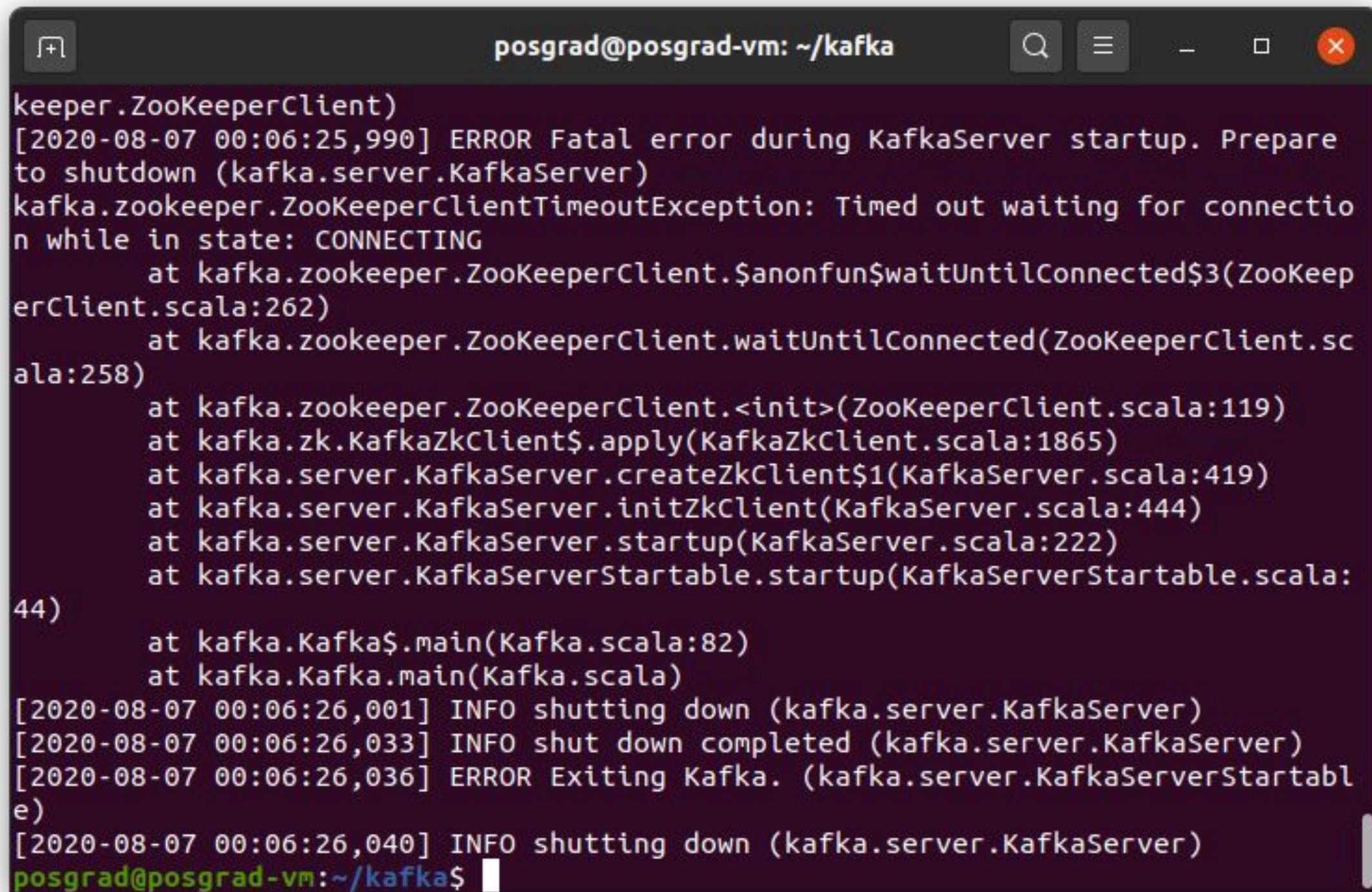
File Edit Selection View Go Run Terminal Help
EXPLORER
OPEN EDITORS
KAFKA
> bin
config
  connect-console-sink.properties
  connect-console-source.properties
  connect-distributed.properties
  connect-file-sink.properties
  connect-file-source.properties
  connect-log4j.properties
  connect-mirror-maker.properties
  connect-standalone.properties
  consumer.properties
  log4j.properties
  producer.properties
  server.properties
  tools-log4j.properties
  trogdr.conf
  zookeeper.properties
> libs
> site-docs
LICENSE
NOTICE
OUTLINE
TIMELINE

connect-log4j.properties ×
config > connect-log4j.properties
19 #
20 log4j.appenders.stdout=org.apache.log4j.ConsoleAppender
21 log4j.appenders.stdout.layout=org.apache.log4j.PatternLayout
22
23 # Send the logs to a file, rolling the file at midnight local time. For example, the `F
24 # location of the log files (e.g. ${kafka.logs.dir}/connect.log), and at midnight local
25 # and copied in the same directory but with a filename that ends in the `DatePattern` o
26 #
27 log4j.appenders.connectAppender=org.apache.log4j.DailyRollingFileAppender
28 log4j.appenders.connectAppender.DatePattern='yyyy-MM-dd-HH
29 log4j.appenders.connectAppender.File=${kafka.logs.dir}/connect.log
30 log4j.appenders.connectAppender.layout=org.apache.log4j.PatternLayout
31
32 # The `'%X{connector.context}'` parameter in the layout includes connector-specific and t
33 # in the log message, where appropriate. This makes it easier to identify those log mes
34 # specific connector. Simply add this parameter to the log layout configuration below t
35 #
36 connect.log.pattern=[%d] %p %m (%c:%L)%n
37 #connect.log.pattern=[%d] %p %X{connector.context}%m (%c:%L)%n
38
39 log4j.appenders.stdout.layout.ConversionPattern=${connect.log.pattern}
40 log4j.appenders.connectAppender.layout.ConversionPattern=${connect.log.pattern}
41
42 log4j.logger.org.apache.zookeeper=ERROR
43 log4j.logger.org.reflections=ERROR
44
```

At the bottom right, status icons include "Ln 1, Col 1", "Spaces: 4", "UTF-8", "LF", "Properties", and a refresh icon.

# Inicializando o Servidor Kafka

➤ bin/kafka-server-start.sh config/server.properties



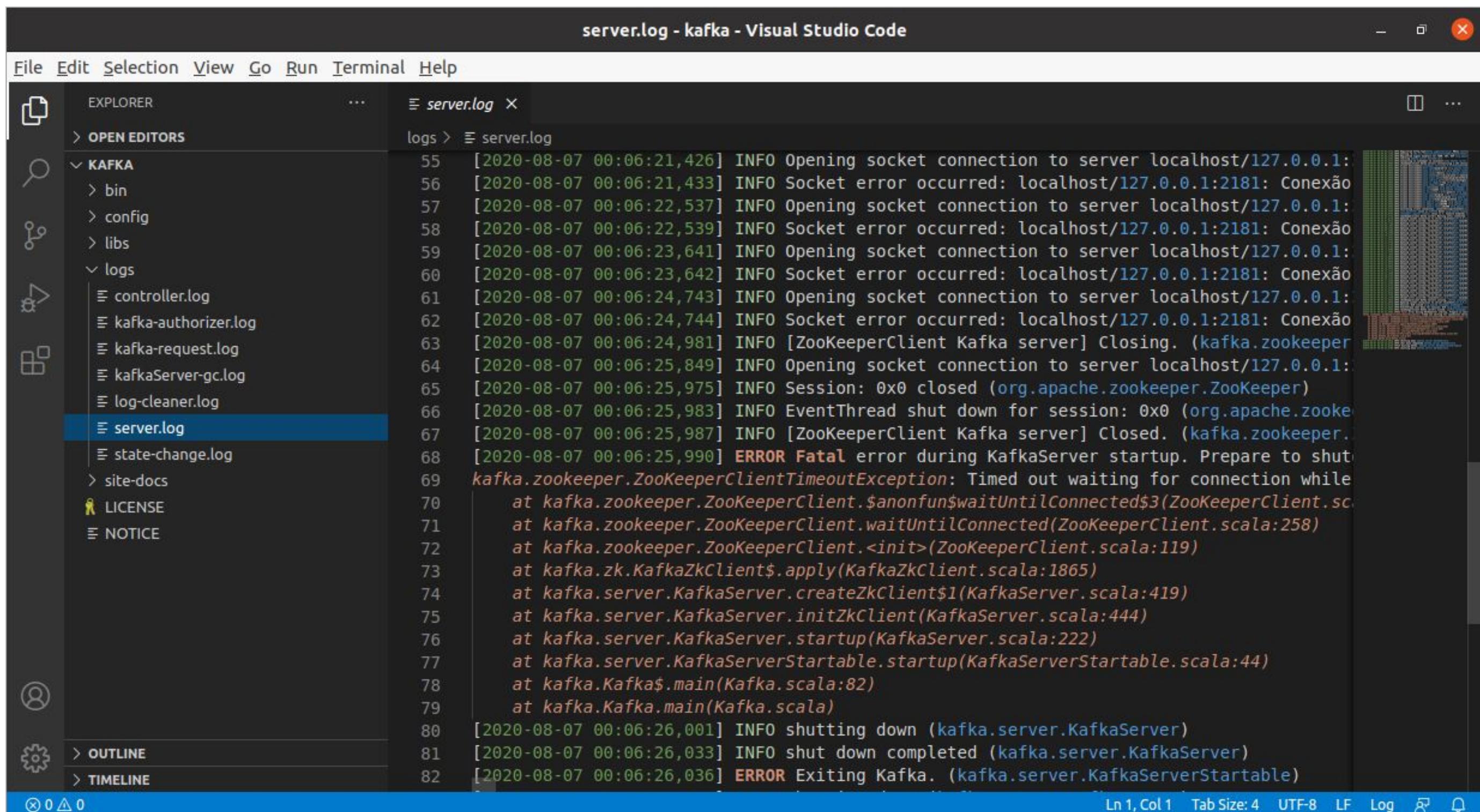
A screenshot of a terminal window titled "posgrad@posgrad-vm: ~/kafka". The window contains a stack trace and log messages. The stack trace is as follows:

```
keeper.ZooKeeperClient)
[2020-08-07 00:06:25,990] ERROR Fatal error during KafkaServer startup. Prepare
to shutdown (kafka.server.KafkaServer)
kafka.zookeeper.ZooKeeperClientTimeoutException: Timed out waiting for connectio
n while in state: CONNECTING
    at kafka.zookeeper.ZooKeeperClient.$anonfun$waitForConnected$3(ZooKeep
erClient.scala:262)
    at kafka.zookeeper.ZooKeeperClient.waitForConnected(ZooKeeperClient.sc
ala:258)
    at kafka.zookeeper.ZooKeeperClient.<init>(ZooKeeperClient.scala:119)
    at kafka.zk.KafkaZkClient$.apply(KafkaZkClient.scala:1865)
    at kafka.server.KafkaServer.createZkClient$(KafkaServer.scala:419)
    at kafka.server.KafkaServer.initZkClient(KafkaServer.scala:444)
    at kafka.server.KafkaServer.startup(KafkaServer.scala:222)
    at kafka.server.KafkaServerStartable.startup(KafkaServerStartable.scala:
44)
    at kafka.Kafka$.main(Kafka.scala:82)
    at kafka.Kafka.main(Kafka.scala)
[2020-08-07 00:06:26,001] INFO shutting down (kafka.server.KafkaServer)
[2020-08-07 00:06:26,033] INFO shut down completed (kafka.server.KafkaServer)
[2020-08-07 00:06:26,036] ERROR Exiting Kafka. (kafka.server.KafkaServerStartabl
e)
[2020-08-07 00:06:26,040] INFO shutting down (kafka.server.KafkaServer)
```

The terminal prompt "posgrad@posgrad-vm:~/kafka\$" is visible at the bottom.

# Observando os logs de erro do kafka

No VS Code, abrir a pasta logs, e o arquivo server.log



The screenshot shows a Visual Studio Code window titled "server.log - kafka - Visual Studio Code". The left sidebar has "OPEN EDITORS" expanded, showing a tree view of a "KAFKA" directory containing files like "bin", "config", "libs", "logs", and several log files: "controller.log", "kafka-authorizer.log", "kafka-request.log", "kafkaServer-gc.log", "log-cleaner.log", "server.log" (which is selected and highlighted in blue), and "state-change.log". The main editor area displays the contents of the "server.log" file, which is a log of Kafka server startup. Lines 55 through 82 are shown, detailing socket connections, errors, and finally the shutdown of the Kafka server. The log ends with line 82: "[2020-08-07 00:06:26,036] **ERROR** Exiting Kafka. (kafka.server.KafkaServerStartable)".

```
55 [2020-08-07 00:06:21,426] INFO Opening socket connection to server localhost/127.0.0.1:  
56 [2020-08-07 00:06:21,433] INFO Socket error occurred: localhost/127.0.0.1:2181: Conexão  
57 [2020-08-07 00:06:22,537] INFO Opening socket connection to server localhost/127.0.0.1:  
58 [2020-08-07 00:06:22,539] INFO Socket error occurred: localhost/127.0.0.1:2181: Conexão  
59 [2020-08-07 00:06:23,641] INFO Opening socket connection to server localhost/127.0.0.1:  
60 [2020-08-07 00:06:23,642] INFO Socket error occurred: localhost/127.0.0.1:2181: Conexão  
61 [2020-08-07 00:06:24,743] INFO Opening socket connection to server localhost/127.0.0.1:  
62 [2020-08-07 00:06:24,744] INFO Socket error occurred: localhost/127.0.0.1:2181: Conexão  
63 [2020-08-07 00:06:24,981] INFO [ZooKeeperClient Kafka server] Closing. (kafka.zookeeper.  
64 [2020-08-07 00:06:25,849] INFO Opening socket connection to server localhost/127.0.0.1:  
65 [2020-08-07 00:06:25,975] INFO Session: 0x0 closed (org.apache.zookeeper.ZooKeeper)  
66 [2020-08-07 00:06:25,983] INFO EventThread shut down for session: 0x0 (org.apache.zooke  
67 [2020-08-07 00:06:25,987] INFO [ZooKeeperClient Kafka server] Closed. (kafka.zookeeper.  
68 [2020-08-07 00:06:25,990] ERROR Fatal error during KafkaServer startup. Prepare to shut  
69 kafka.zookeeper.ZooKeeperClientTimeoutException: Timed out waiting for connection while  
70 at kafka.zookeeper.ZooKeeperClient.$anonfun$waitForConnected$3(ZooKeeperClient.sc  
71 at kafka.zookeeper.ZooKeeperClient.waitForConnected(ZooKeeperClient.scala:258)  
72 at kafka.zookeeper.ZooKeeperClient.<init>(ZooKeeperClient.scala:119)  
73 at kafka.zk.KafkaZkClient$.apply(KafkaZkClient.scala:1865)  
74 at kafka.server.KafkaServer.createZkClient$1(KafkaServer.scala:419)  
75 at kafka.server.KafkaServer.initZkClient(KafkaServer.scala:444)  
76 at kafka.server.KafkaServer.startup(KafkaServer.scala:222)  
77 at kafka.server.KafkaServerStartable.startup(KafkaServerStartable.scala:44)  
78 at kafka.Kafka$.main(Kafka.scala:82)  
79 at kafka.Kafka.main(Kafka.scala)  
80 [2020-08-07 00:06:26,001] INFO shutting down (kafka.server.KafkaServer)  
81 [2020-08-07 00:06:26,033] INFO shut down completed (kafka.server.KafkaServer)  
82 [2020-08-07 00:06:26,036] ERROR Exiting Kafka. (kafka.server.KafkaServerStartable)
```

# Inicializando o Zookeeper

➤ bin/zookeeper-server-start.sh config/zookeeper.properties

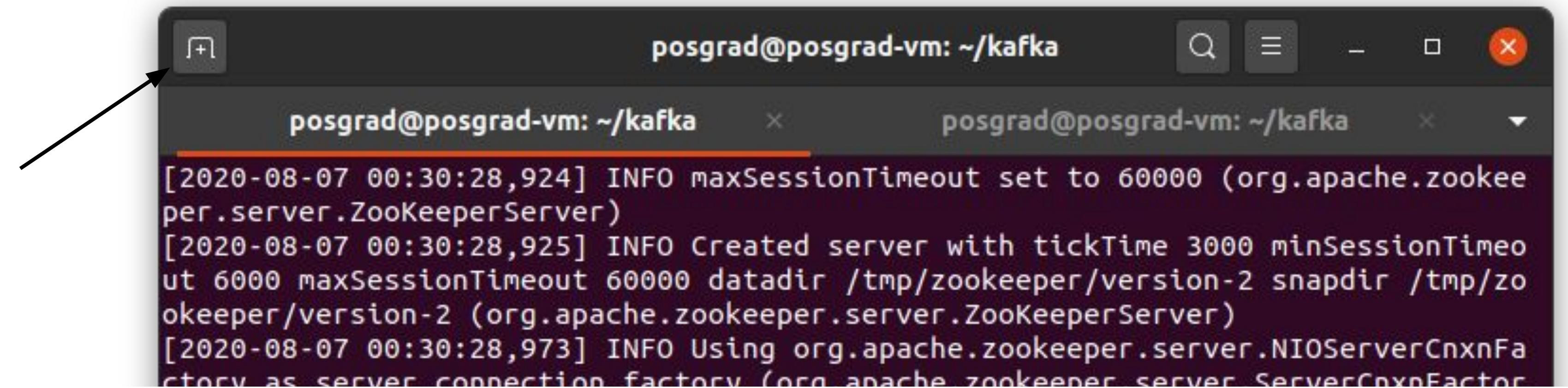
Zookeeper por padrão escuta a porta 2181

Observar arquivo ***zookeeper.properties*** na pasta ***config***

```
connect-log4j.properties          15 # the directory where the snapshot is stored.
connect-mirror-maker.properties    16 dataDir=/tmp/zookeeper
connect-standalone.properties      17 # the port at which the clients will connect
consumer.properties                18 clientPort=2181
log4j.properties                  19 # disable the per-ip limit on the number of connections since this is a non-production
                                20 maxClientCnxns=0
```

# Inicializando o Servidor Kafka (novamente)

Em outro terminal



```
[2020-08-07 00:30:28,924] INFO maxSessionTimeout set to 60000 (org.apache.zookeeper.server.ZooKeeperServer)
[2020-08-07 00:30:28,925] INFO Created server with tickTime 3000 minSessionTimeout 6000 maxSessionTimeout 60000 datadir /tmp/zookeeper/version-2 snapdir /tmp/zookeeper/version-2 (org.apache.zookeeper.server.ZooKeeperServer)
[2020-08-07 00:30:28,973] INFO Using org.apache.zookeeper.server.NIOServerCnxnFactory as server connection factory (org.apache.zookeeper.server.NIOServerCnxnFactory)
```

Inicie o Kafka novamente:

➤ bin/kafka-server-start.sh config/server.properties

# Voilà

Zookeeper **localhost:2181**

Kafka server (broker) **localhost:9092**

# Dúvidas?