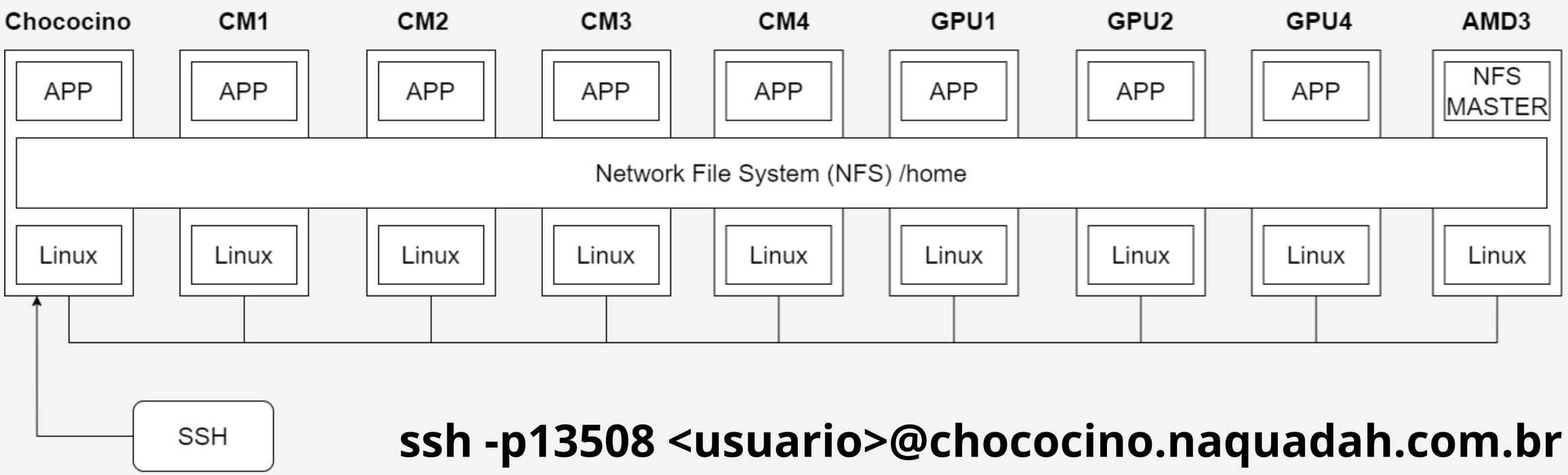
HADOOP & SPARK

Chococino

Chococino

Nome	CPU	Memória	GPU
Chococino	Ryzen 7 2700	4GB	-
CM1 CM2 CM3 CM4	i7-8700	16GB	-
GPU1 GPU2 GPU3	Ryzen 7 2700	16GB	GTX1650

Chococino



ssh -p13508 <usuario>@chococino.naquadah.com.br

Serviços

#	Nome	Endereço
Hadoop	Namenode	hdfs://chococino:9000
	Namenode WebUI	http://chococino:9870
Spark	Master	spark://cm1:7077
	Master WebUI	http://cm1:9090

Inicialização do Ambiente

Sessão

source /home/prof/hadoop/bin/chococino_env

Definitivo

echo "source /home/prof/hadoop/bin/chococino_env" >> $HOME/.bashrc

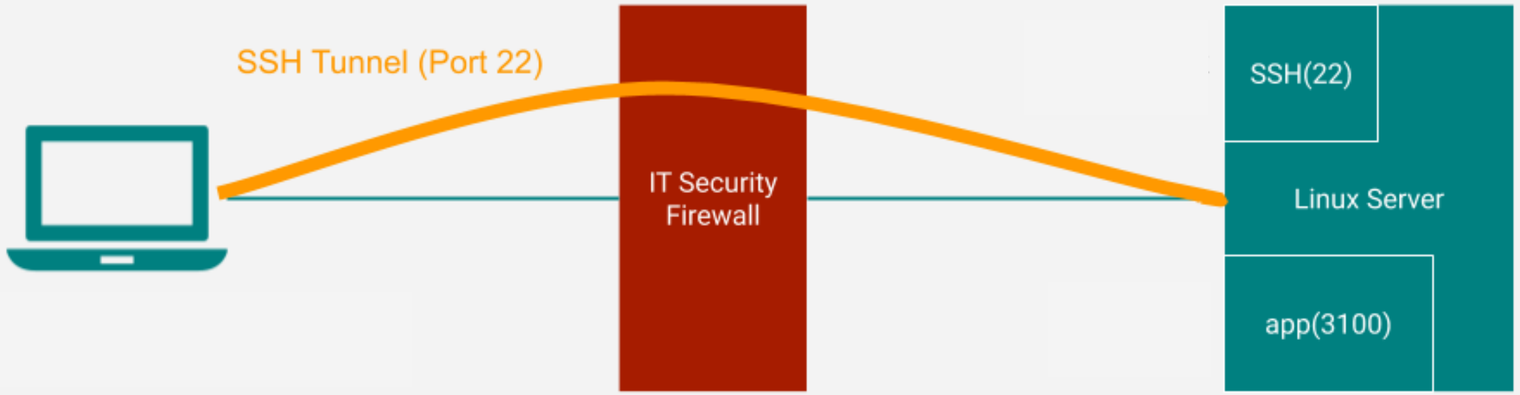
Estrutura do Hadoop

- Um único namenode (chococino)
- Vários datanodes (cmX e gpuX)

Comandos do Hadoop

- Para ver o help: hdfs dfs -h
- Comandos padrões do linux (cat, ls, mkdir, touch, put)

SSH - Port forwarding



ssh -L <porta_local>:<ip_remoto>:<porta_remota>

ssh -L 8000:chococino:9870 -p13508 <usuario>@chococino.naquadah.com.br

ssh -L 9000:cm1:9090 -p13508 <usuario>@chococino.naquadah.com.br

Estrutura do Spark

- Um único master (cm1)
- Vários workers (cm2-cm4 e gpuX)

Comandos do Spark

- Notebook e pyspark
- Querys SQL Like

Executando o notebook

Dentro das máquinas cmX ou gpuX

python3 -m notebook

Flags do notebook

--ip: configura o ip do notebook

--port: configura a porta do notebook

Comandos importantes Pyspark

INICIALIZAÇÃO

```
from pyspark.sql import SparkSession
spark = SparkSession.builder.master("spark://cm1:7077").appName("Nome do app").getOrCreate()
```

LEITURA DO HDFS

```
spark.read.text('hdfs://chococino:9000/user/<usuario>/<arquivo>')
```

FINALIZAÇÃO

```
spark.stop()
```

MANIPULAÇÃO DE ARQUIVOS

- count() # número de linhas
- show() # printa o conteúdo
- select() # seleção das colunas
- alias() # renomear coluna
- take() # coleta n dados
- first() # retorna o primeiro
- collect() # retorna todos
- map() # transforma a linha
- filter() # filtra as linhas
- reduce() # reducao dos dados
- orderBy() # ordena os dados
- sum() # soma as informações
- avg() # calcula a média
- join() # realiza a junção de dataframes
- groupBy() # realiza o agrupamento
- foreach() # itera sobre cada elemento
- split()

[PYSPARK API REFERENCE]