



# Déploiement d'une architecture Hadoop pour analyse de flux

[françois-xavier.andreu@renater.fr](mailto:françois-xavier.andreu@renater.fr)

# plan

- Introduction
- Hadoop
  - Présentation
  - Architecture d'un cluster
  - HDFS & MapReduce
- L'architecture déployée
  - Les serveurs
  - Les applications
  - Contraintes
  - Remarques
- Après ?
- Conclusion

# plan

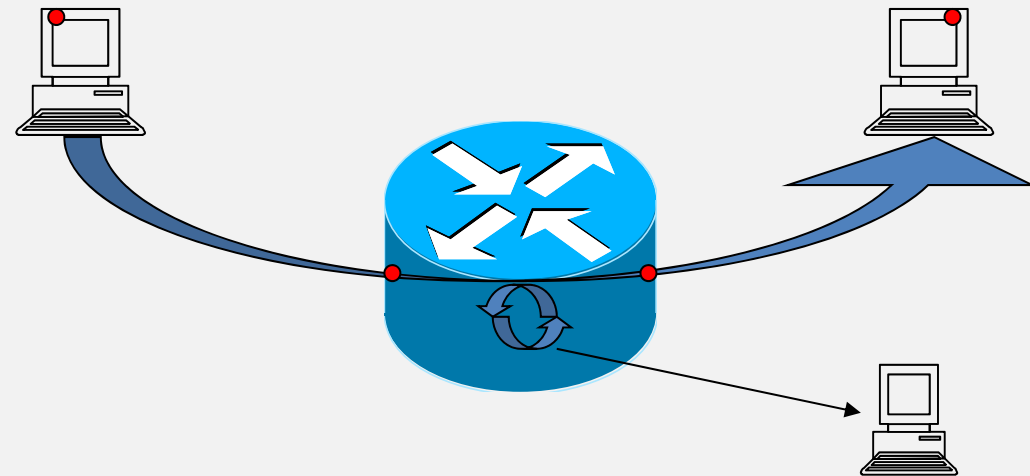
- Introduction
- Hadoop
  - Présentation
  - Architecture d'un cluster
  - HDFS & MapReduce
- L'architecture déployée
  - Les serveurs
  - Les applications
  - Contraintes
  - Remarques
- Après ?
- Conclusion

# Introduction

- Besoin : historique des exports NetFlow
  - Analyse anomalies
  - *Capacity planning*
  - Stockage : > 1To de données par jour
  - Temps de réponse des applications
- Solution : Hadoop ?

# Rappel : NetFlow

- Définition d'un flux:
  - IP source
  - IP destination
  - port source
  - port destination
  - protocole
  - type de service
  - index de l'interface d'entrée



# Rappel : NetFlow

IP Header

0		15	16	31
<b>Version</b>	<b>HLEN</b>	<b>ToS</b>		<b>Total Length</b>
<b>Identification</b>			<b>flags</b>	<b>Fragment Offset</b>
<b>Time to Live</b>		<b>Protocol</b>		<b>Header Checksum</b>
<b>Source IP Address</b>				
<b>Destination IP Address</b>				

TCP Header

<b>Source Port Number</b>			<b>Destination Port Number</b>	
<b>Sequence Number</b>				
<b>Acknowledgement Number</b>				
<b>Header</b>	<b>Reserved</b>	<b>TCP Flags</b>		<b>Window Size</b>
<b>TCP Checksum</b>			<b>Urgent Pointer</b>	

# Rappel : NetFlow

- Exemple : connexion au site [www.jres.org](http://www.jres.org)

Nb Octets	Nb Paquets	Index In	Index Out	IP Source	IP Dest	Prot	Port Sce	Port Dst
625	6	5	13	193.49.160.34	195.220.94.176	6	50694	80
666	5	13	5	195.220.94.176	193.49.160.34	6	80	50694
8791	97	5	13	193.49.160.34	195.220.94.176	6	50695	443
265758	219	13	5	195.220.94.176	193.49.160.34	6	443	50695
5456	37	5	13	193.49.160.34	195.220.94.176	6	50698	443
81686	73	13	5	195.220.94.176	193.49.160.34	6	443	50698
3879	35	5	13	193.49.160.34	195.220.94.176	6	50699	443
69363	63	13	5	195.220.94.176	193.49.160.34	6	443	50699
13268	197	5	13	193.49.160.34	195.220.94.176	6	50700	443
559836	445	13	5	195.220.94.176	193.49.160.34	6	443	50700
4908	37	5	13	193.49.160.34	195.220.94.176	6	50701	443
71157	64	13	5	195.220.94.176	193.49.160.34	6	443	50701
4480	27	5	13	193.49.160.34	195.220.94.176	6	50702	443
51704	49	13	5	195.220.94.176	193.49.160.34	6	443	50702
1636	9	5	13	193.49.160.34	195.220.94.176	6	50703	443
2054	9	13	5	195.220.94.176	193.49.160.34	6	443	50703

# Rappel : NetFlow

- Exemple : connexion au site [www.jres.org](http://www.jres.org)

Nb Octets	Nb Paquets	Index In	Index Out	IP Source	IP Dest	Prot	Port Sce	Port Dst
625	6	5	13	193.49.160.34	195.220.94.176	6	50694	80
666	5	13	5	195.220.94.176	193.49.160.34	6	80	50694
8791	97	5	13	193.49.160.34	195.220.94.176	6	50695	443
265758	219	13	5	195.220.94.176	193.49.160.34	6	443	50695
5456	37	5	13	193.49.160.34	195.220.94.176	6	50698	443
81686	73	13	5	195.220.94.176	193.49.160.34	6	443	50698
3879	35	5	13	193.49.160.34	195.220.94.176	6	50699	443
69363	63	13	5	195.220.94.176	193.49.160.34	6	443	50699
13268	197	5	13	193.49.160.34	195.220.94.176	6	50700	443
559836	445	13	5	195.220.94.176	193.49.160.34	6	443	50700
4908	37	5	13	193.49.160.34	195.220.94.176	6	50701	443
71157	64	13	5	195.220.94.176	193.49.160.34	6	443	50701
4480	27	5	13	193.49.160.34	195.220.94.176	6	50702	443
51704	49	13	5	195.220.94.176	193.49.160.34	6	443	50702
1636	9	5	13	193.49.160.34	195.220.94.176	6	50703	443
2054	9	13	5	195.220.94.176	193.49.160.34	6	443	50703

# Rappel : NetFlow

- Exemple : connexion au site [www.jres.org](http://www.jres.org)

Nb Octets	Nb Paquets	Index In	Index Out	IP Source	IP Dest	Prot	Port Sce	Port Dst
625	6	5	13	193.49.160.34	195.220.94.176	6	50694	80
666	5	13	5	195.220.94.176	193.49.160.34	6	80	50694
8791	97	5	13	193.49.160.34	195.220.94.176	6	50695	443
265758	219	13	5	195.220.94.176	193.49.160.34	6	443	50695
5456	37	5	13	193.49.160.34	195.220.94.176	6	50698	443
81686	73	13	5	195.220.94.176	193.49.160.34	6	443	50698
3879	35	5	13	193.49.160.34	195.220.94.176	6	50699	443
69363	63	13	5	195.220.94.176	193.49.160.34	6	443	50699
13268	197	5	13	193.49.160.34	195.220.94.176	6	50700	443
559836	445	13	5	195.220.94.176	193.49.160.34	6	443	50700
4908	37	5	13	193.49.160.34	195.220.94.176	6	50701	443
71157	64	13	5	195.220.94.176	193.49.160.34	6	443	50701
4480	27	5	13	193.49.160.34	195.220.94.176	6	50702	443
51704	49	13	5	195.220.94.176	193.49.160.34	6	443	50702
1636	9	5	13	193.49.160.34	195.220.94.176	6	50703	443
2054	9	13	5	195.220.94.176	193.49.160.34	6	443	50703

# En chiffres

- ~ 70 routeurs
  - échantillonnage 1:1,1:10,1:20
  - NetFlow export version 9
  - >180 Mbits/s d'export UDP
- > 800 interfaces clientes
- ~ 400 000 flux/s
- ~ 16 milliards flux par jour
- → **1,4 To / jour** (8 milliards de flux après déduplication)

# plan

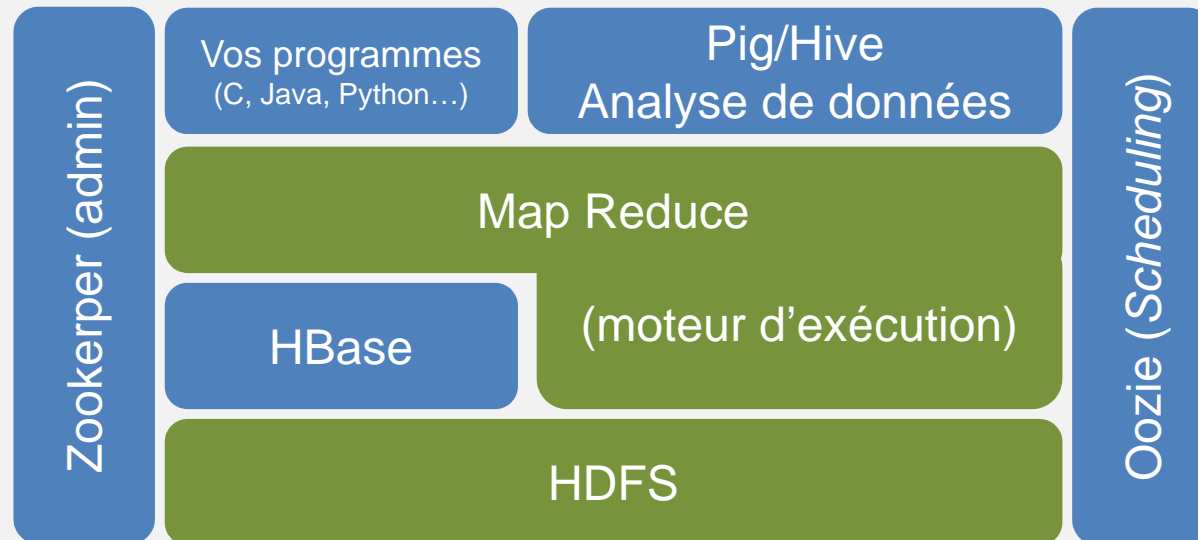
- Introduction
- Hadoop
  - Présentation
  - Architecture d'un cluster
  - HDFS & MapReduce
- L'architecture déployée
  - Les serveurs
  - Les applications
  - Contraintes
  - Remarques
- Après ?
- Conclusion

# Hadoop

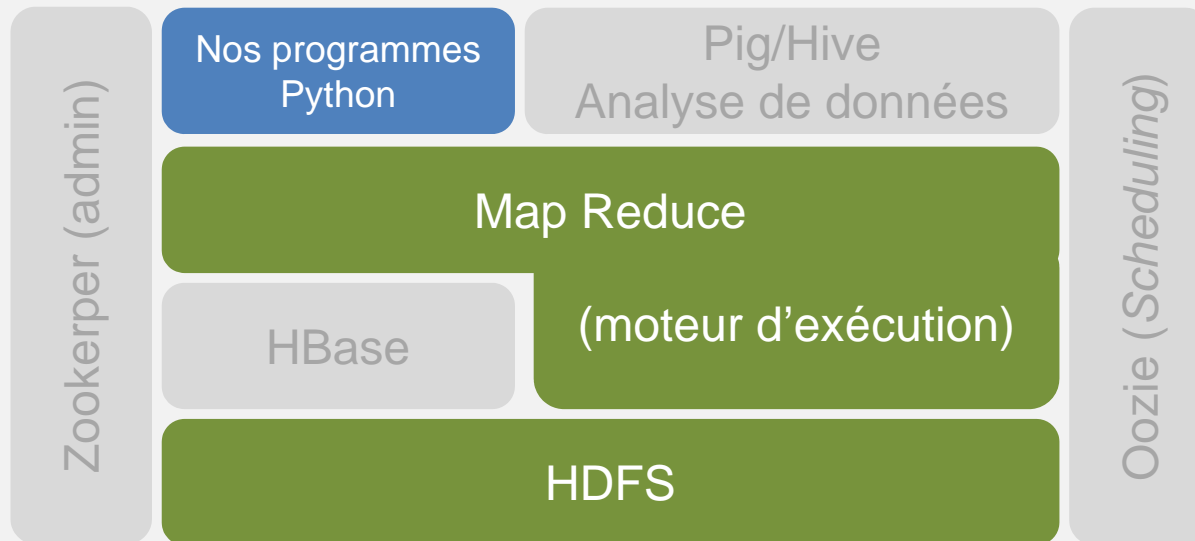
- Framework opensource
- Fondation Apache
- A quoi ça sert ?
  - Stocker
  - Compter
  - Trier
  - Joindre
  - Indexer
  - Agréger
  - Graphes
  - ...



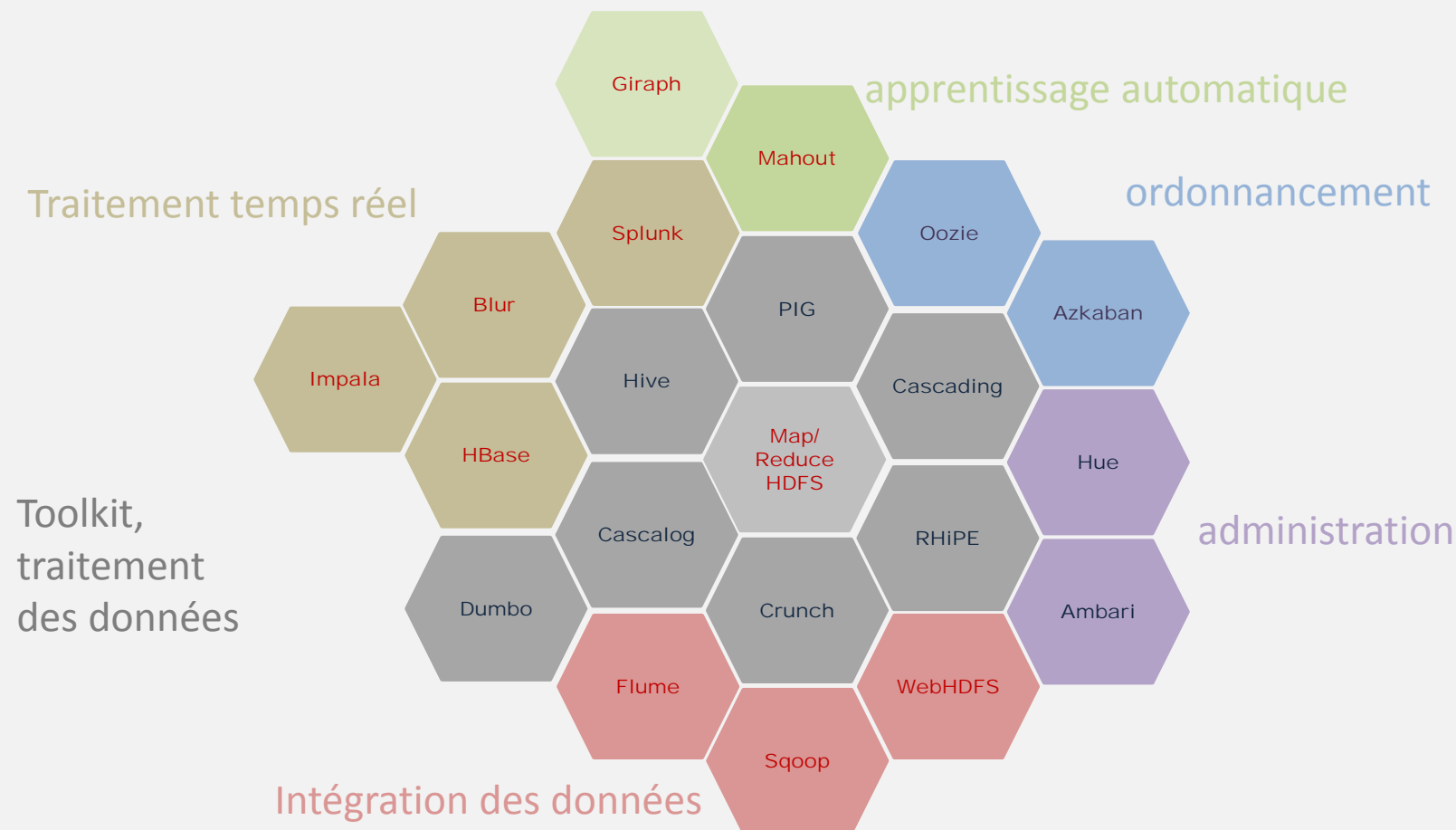
# Hadoop – briques de base



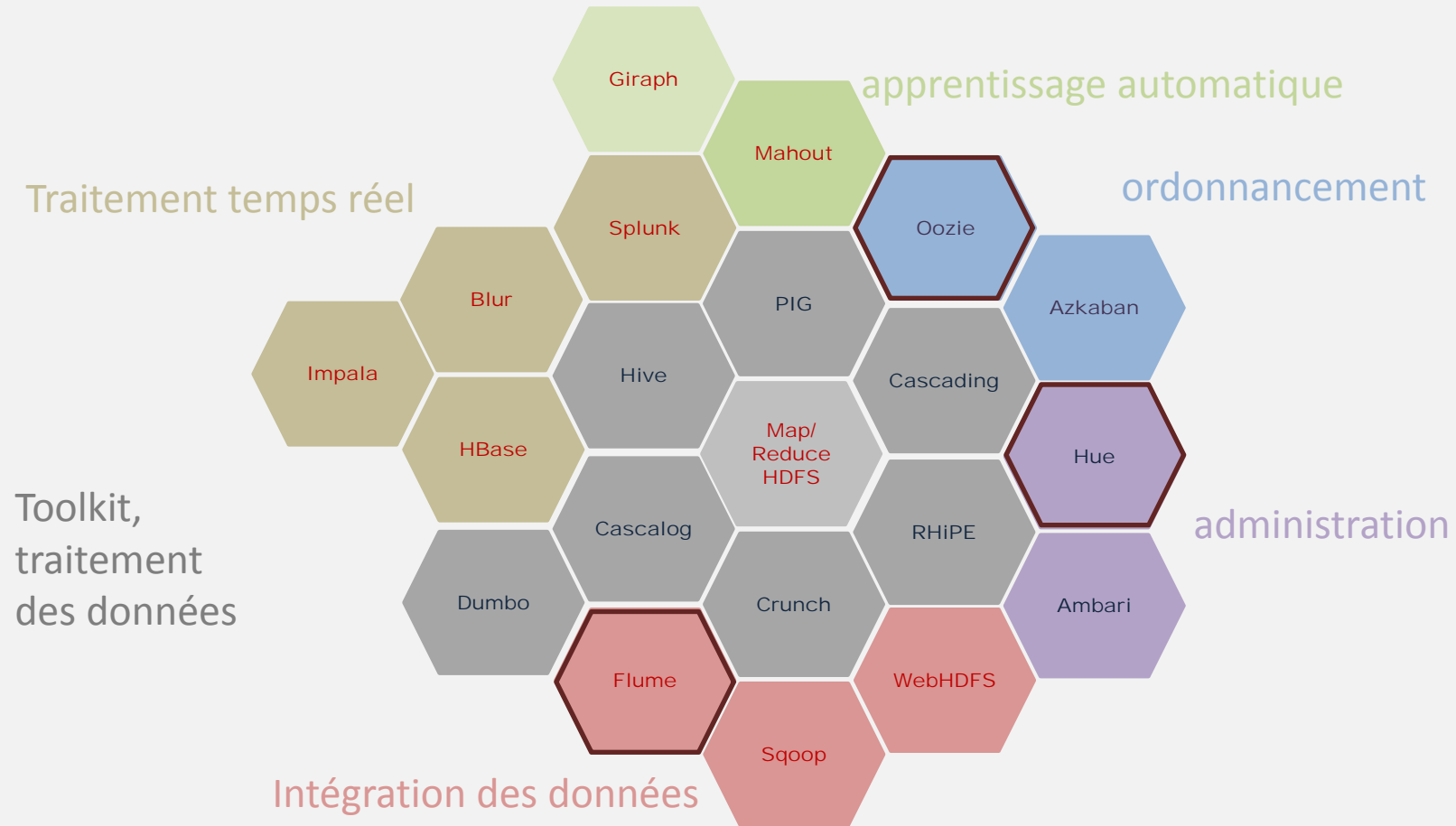
# Hadoop – briques de base



# Hadoop - écosystème



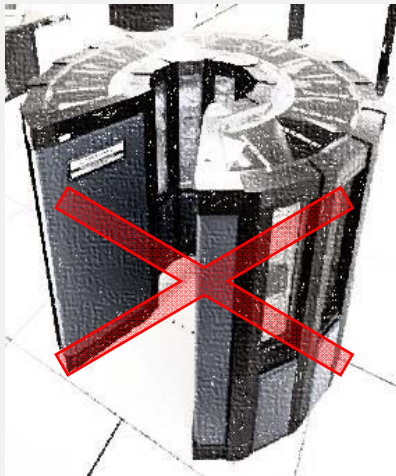
# Hadoop - écosystème



# Hadoop - distributions

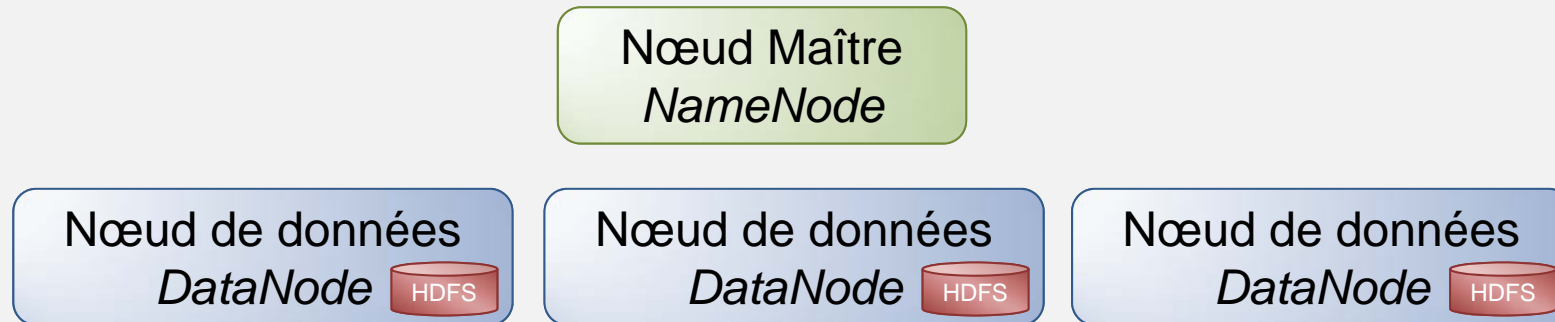
- Apache Bigtop 
- Cloudera distribution for Hadoop 
  - HDFS, MapReduce, HBase, Hive, Mahout, Oozie, Pig, Sqoop, Whirr, Zookeeper, Flume
- MapR Distribution 
  - n'utilise pas HDFS mais leur propre système de fichier basé sur NFS
- Hortonworks Data Plateforme   
**Hortonworks**

# Hadoop - principe

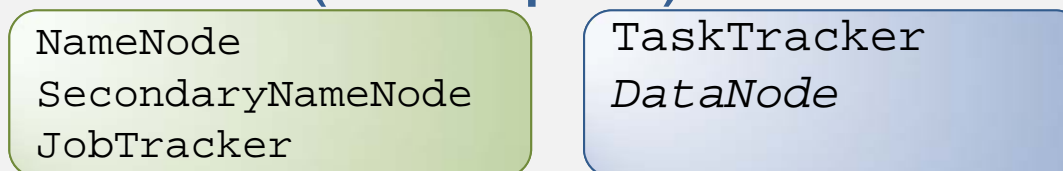


# Hadoop - principe

- Architecture maître/esclave :



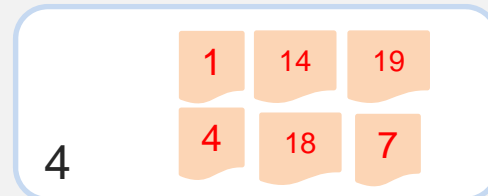
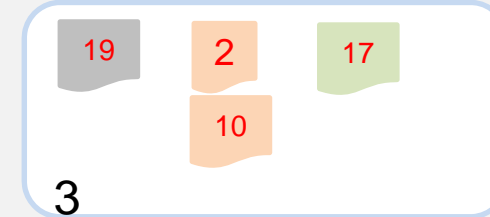
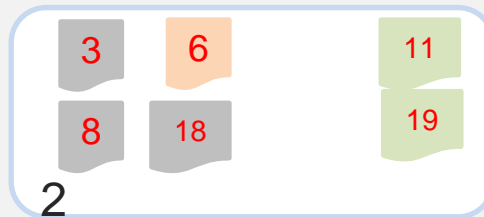
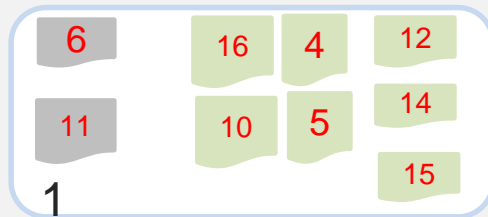
- Processus (au repos):



# Hadoop - HDFS

- HDFS (Java) au dessus de EXT3 (ou EXT4, XFS)
- Pas de RAID ni LVM
- Fichiers en blocs de 64Mo (ou +)
  - Taille qui sera traitée par une tâche Map
- Réplication des données automatique !!

# HDFS - réplication



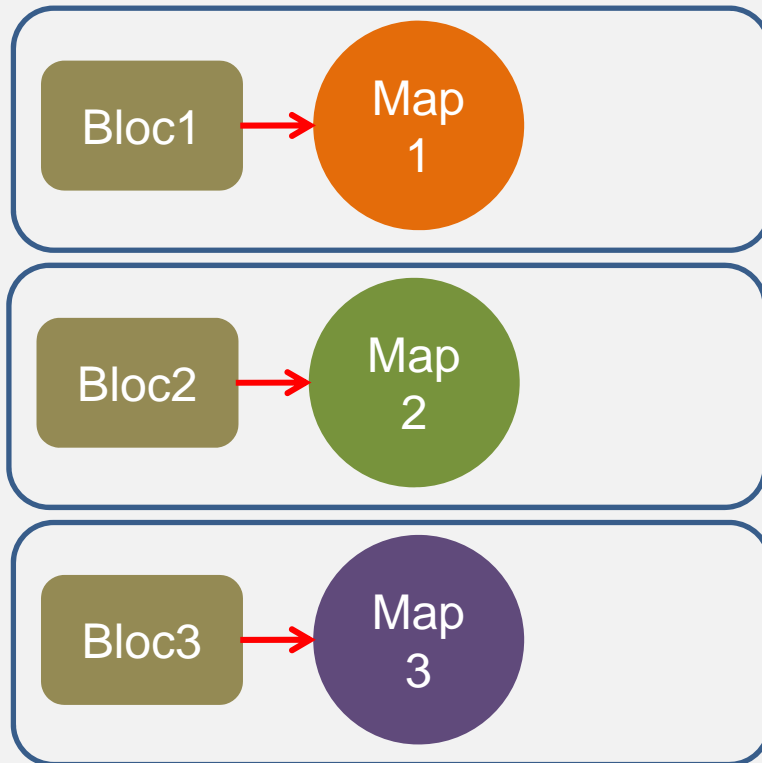
Parties du fichier sans réplication activée

Parties du fichier de la 1ère réplication

Parties du fichier de la seconde réplication

# Hadoop – Map/Reduce - Principe

## Tâches Map

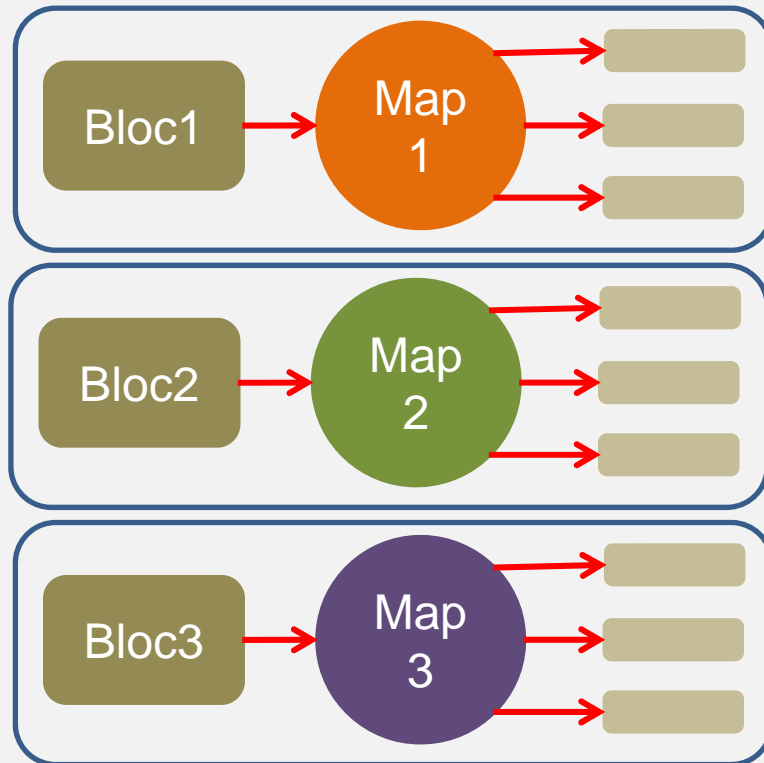


Entrée  
<clé, valeur>

Données intermédiaires  
<clé2, valeur2>

# Hadoop – Map/Reduce - Principe

## Tâches Map



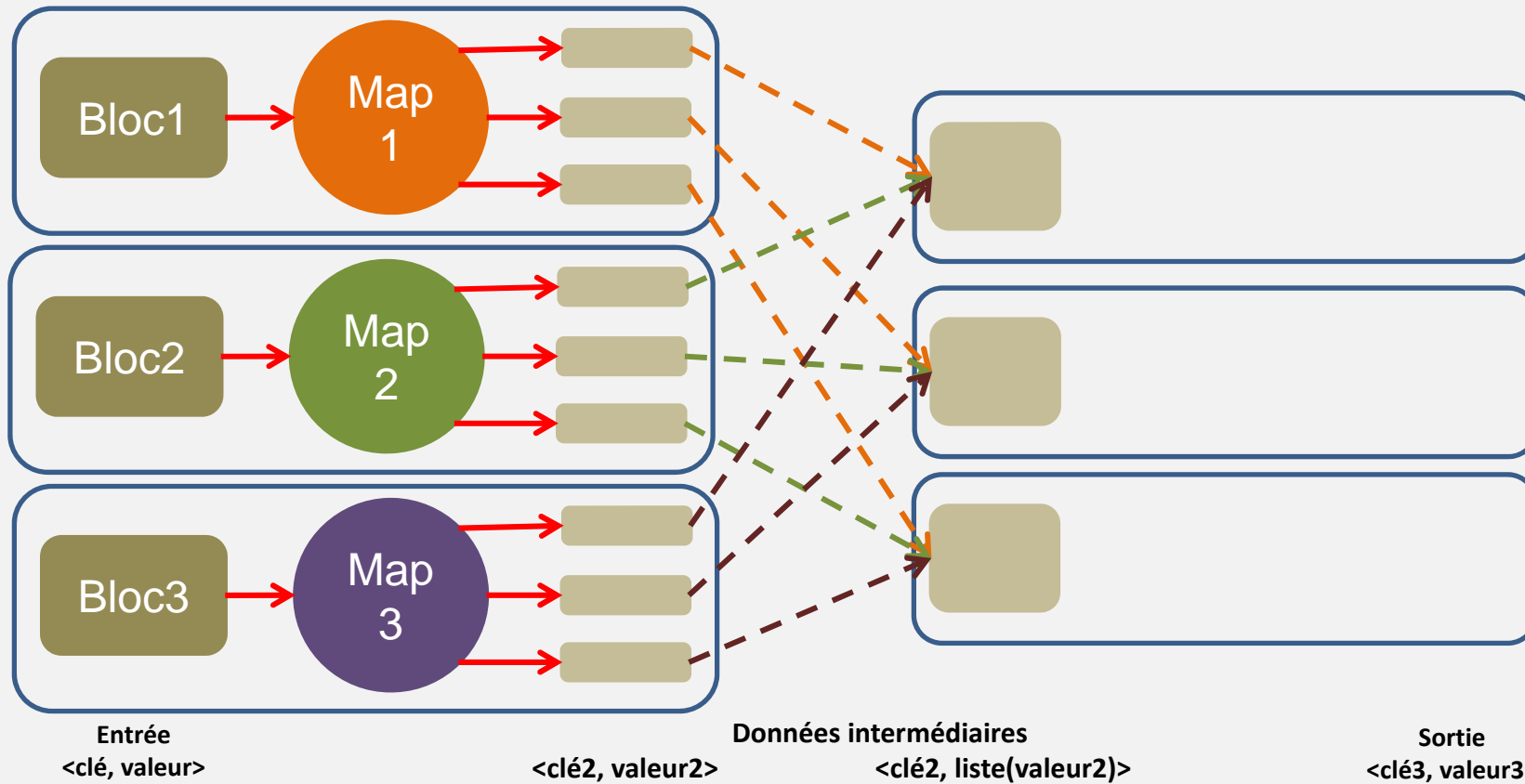
Entrée  
<clé, valeur>

Données intermédiaires

<clé2, valeur2>

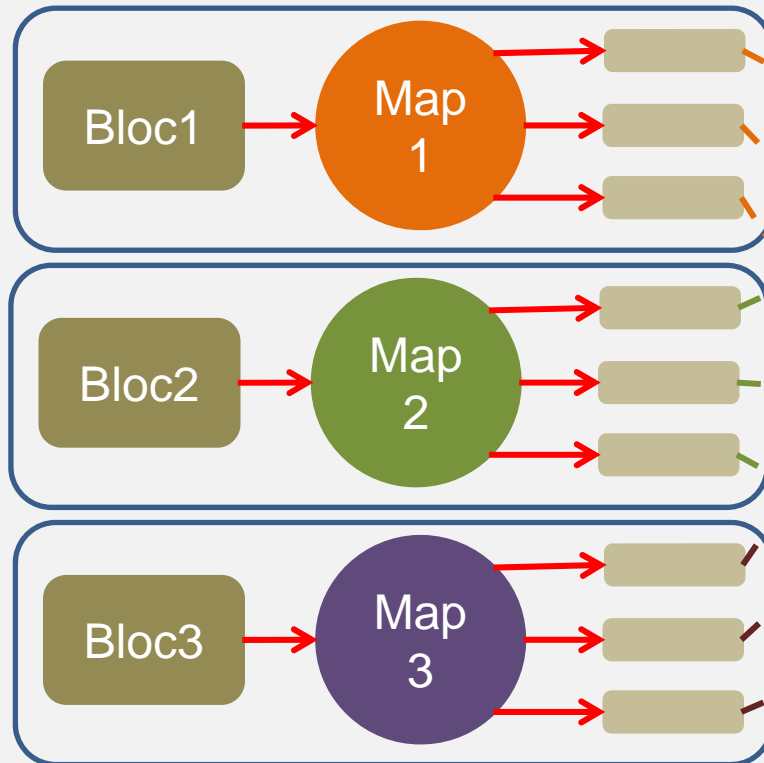
# Hadoop – Map/Reduce - Principe

## Tâches Map

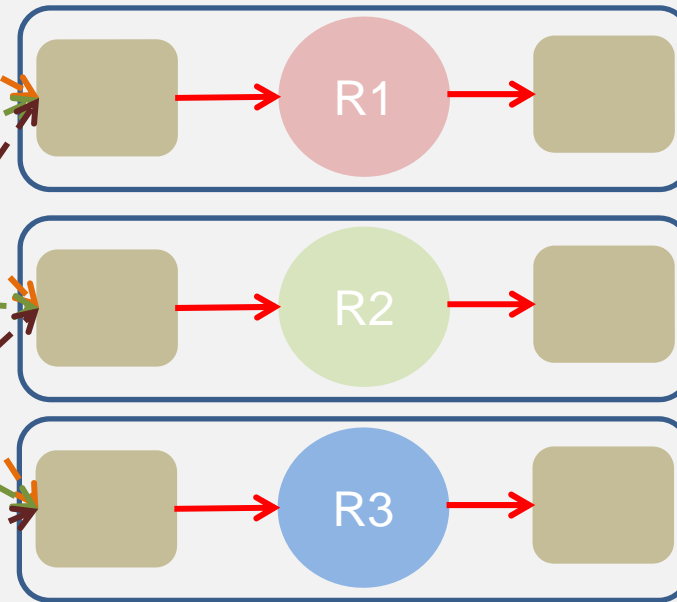


# Hadoop – Map/Reduce - Principe

## Tâches Map



## Tâches Reduce



Entrée  
<clé, valeur>

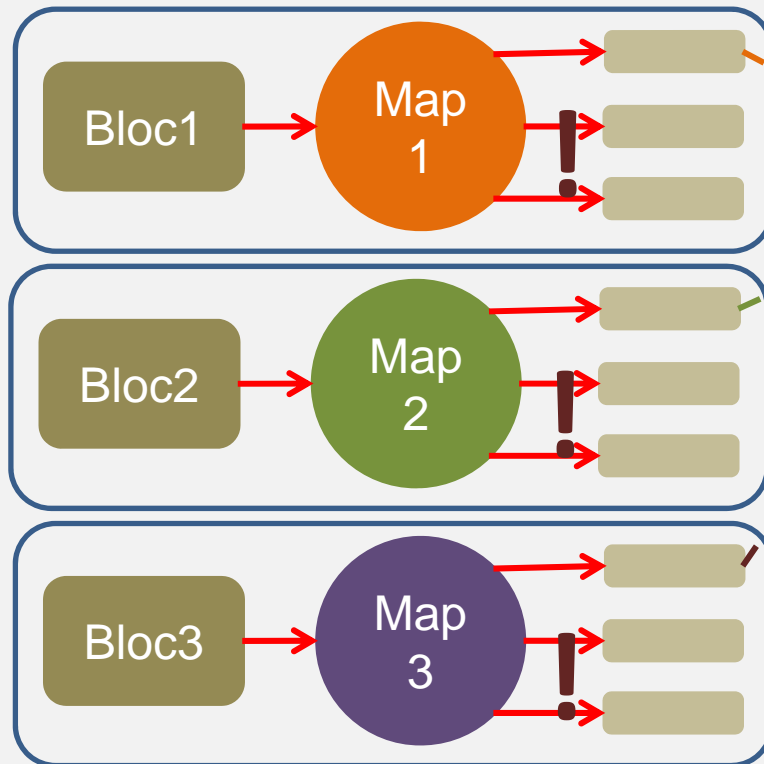
Données intermédiaires  
<clé2, valeur2>

Données intermédiaires  
<clé2, liste(valeur2)>

Sortie  
<clé3, valeur3>

# Hadoop – Map/Reduce - Principe

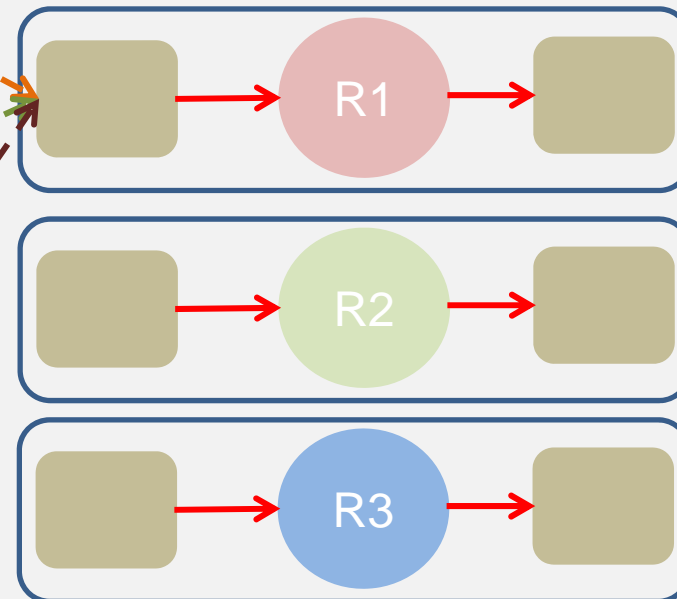
## Tâches Map



Entrée  
<clé, valeur>

<clé2, valeur2>

## Tâches Reduce

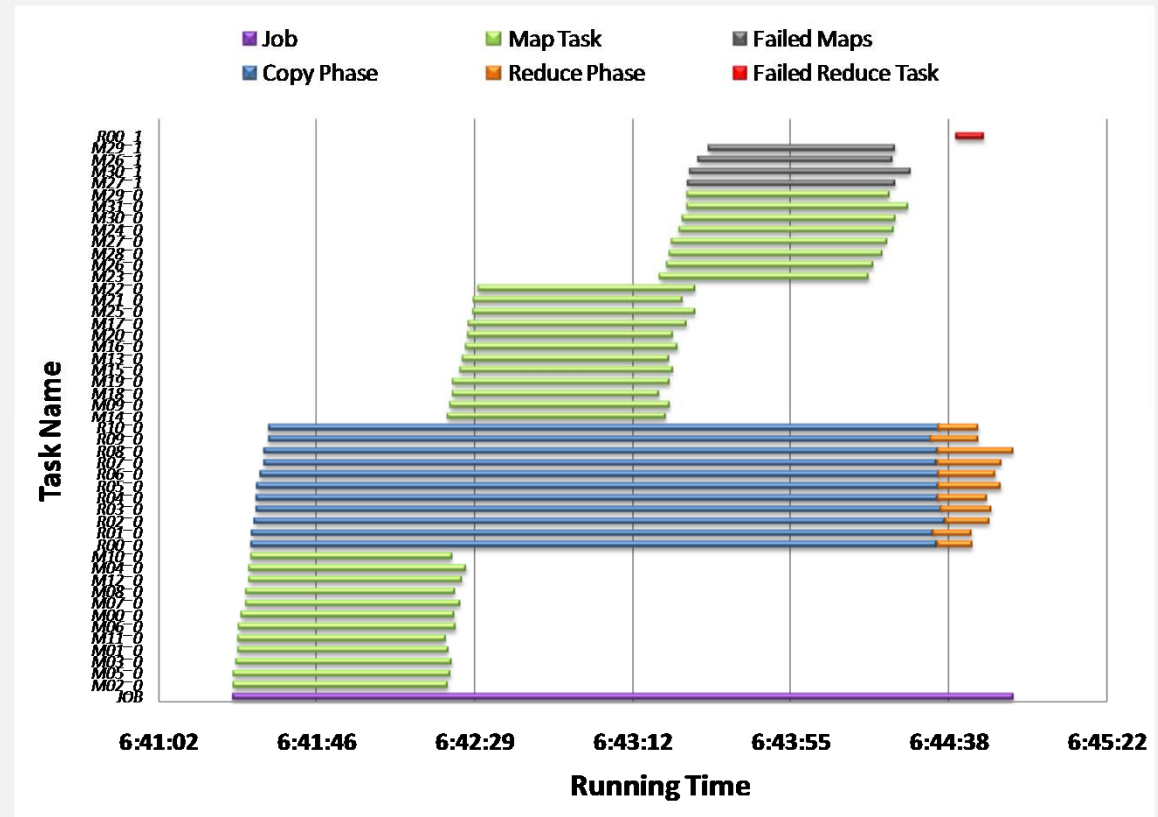


Données intermédiaires  
<clé2, liste(valeur2)>

Sortie  
<clé3, valeur3>

# Hadoop – Map/Reduce execution

- 1 JVM par tâche
- 2 tâches / *thread* ⚠
- Exécution d'un Job :
  - En vert : Exécution des tâches *Maps*
  - En bleu : copie des résultats
  - En orange : tâches *Reduces*



## Hadoop – Map/Reduce – shell vs hadoop

- En shell :  
`cat <mes données> | <mon programme de parcours> | sort | <mon programme d'agrégation> > <mon résultat>`
- Avec Hadoop :  
`hadoop jar contrib/streaming/hadoop-*streaming*.jar \  
-file <mon programme de parcours> \  
-mapper <mon programme de parcours> \  
-file <mon programme d'agrégation> \  
-reducer <mon programme d'agrégation> \  
-input <mes données> \  
-output <mon résultats>`

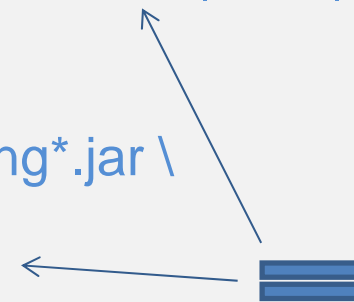
## Hadoop – Map/Reduce – shell vs hadoop

- En shell :

```
cat <mes données> | <mon programme de parcours> | sort | <mon  
programme d'agrégation> > <mon résultat>
```

- Avec Hadoop :

```
hadoop jar contrib/streaming/hadoop-*streaming*.jar \  
-file <mon programme de parcours> \  
-mapper <mon programme de parcours> \  
-file <mon programme d'agrégation> \  
-reducer <mon programme d'agrégation> \  
-input <mes données> \  
-output <mon résultats>
```



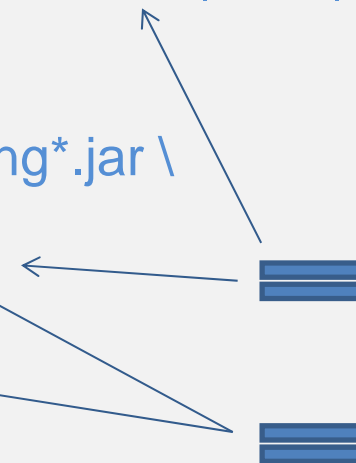
## Hadoop – Map/Reduce – shell vs hadoop

- En shell :

```
cat <mes données> | <mon programme de parcours> | sort | <mon  
programme d'agrégation> > <mon résultat>
```

- Avec Hadoop :

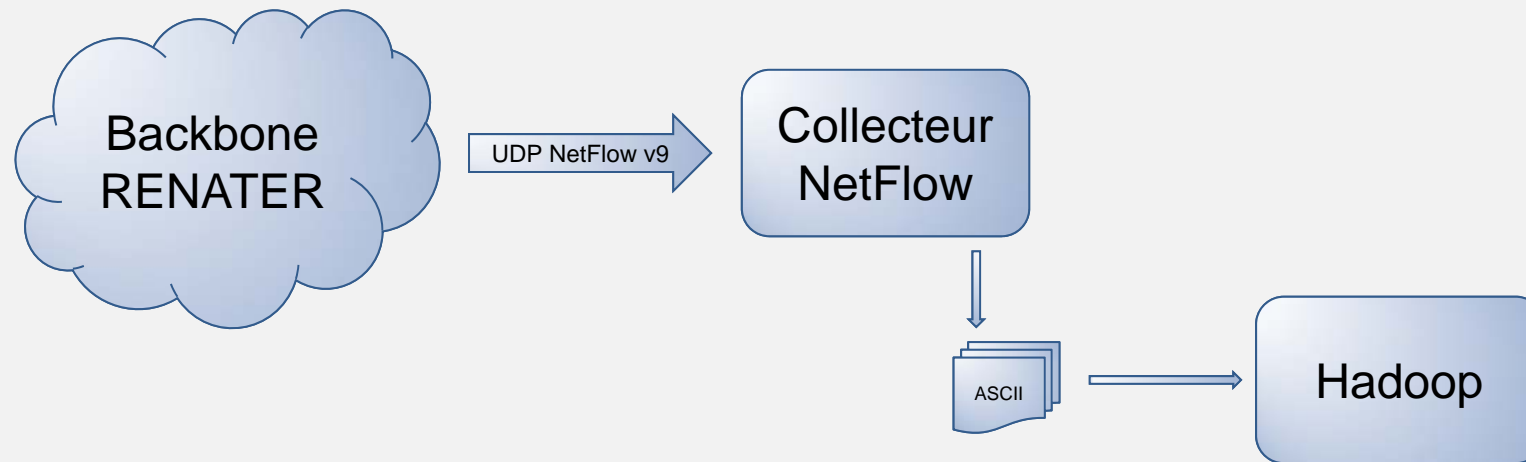
```
hadoop jar contrib/streaming/hadoop-*streaming*.jar \  
-file <mon programme de parcours> \  
-mapper <mon programme de parcours> \  
-file <mon programme d'agrégation> \  
-reducer <mon programme d'agrégation> \  
-input <mes données> \  
-output <mon résultats>
```



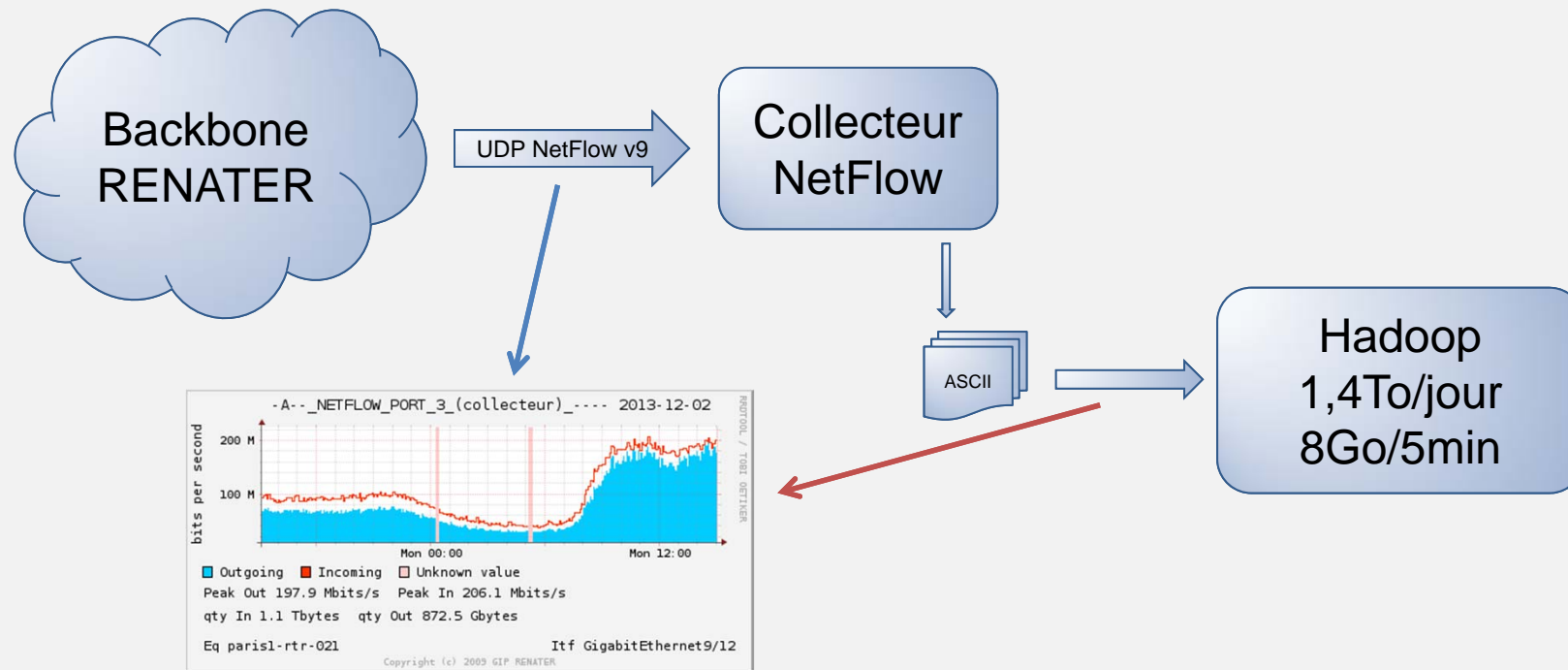
# plan

- Introduction
- Hadoop
  - Présentation
  - Architecture d'un cluster
  - HDFS & MapReduce
- L'architecture déployée
  - Les serveurs
  - Les applications
  - Contraintes
  - Remarques
- Après ?
- Conclusion

# L'architecture déployée



# L'architecture déployée



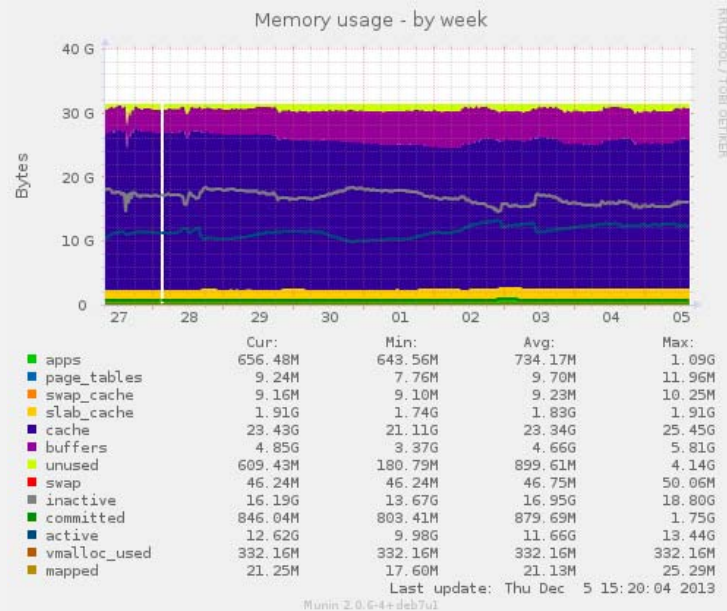
# Les serveurs (1)

- DELL R420 1U - \* 11
  - Bi processeur Xeon E5-2440 (2,4GHz, 6 cœurs – 12 *Threads*)
  - 32 Go RAM
  - 1DD 146 Go
  - 2 DD 15Krpm 300Go

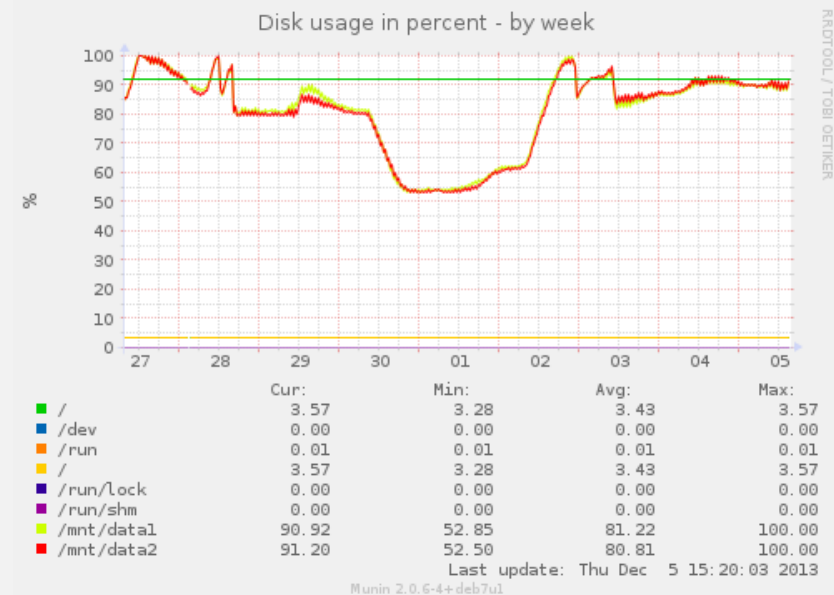


# Les serveurs (2)

- Utilisation mémoire



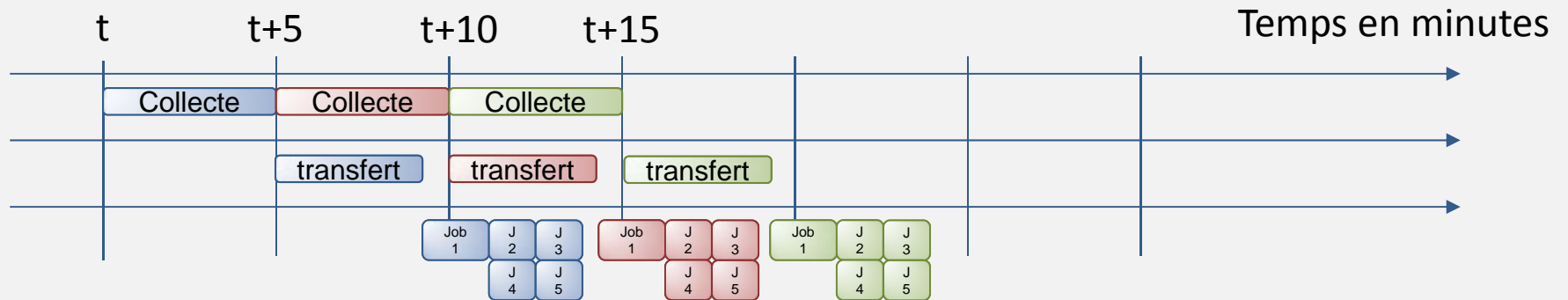
- Utilisation disque



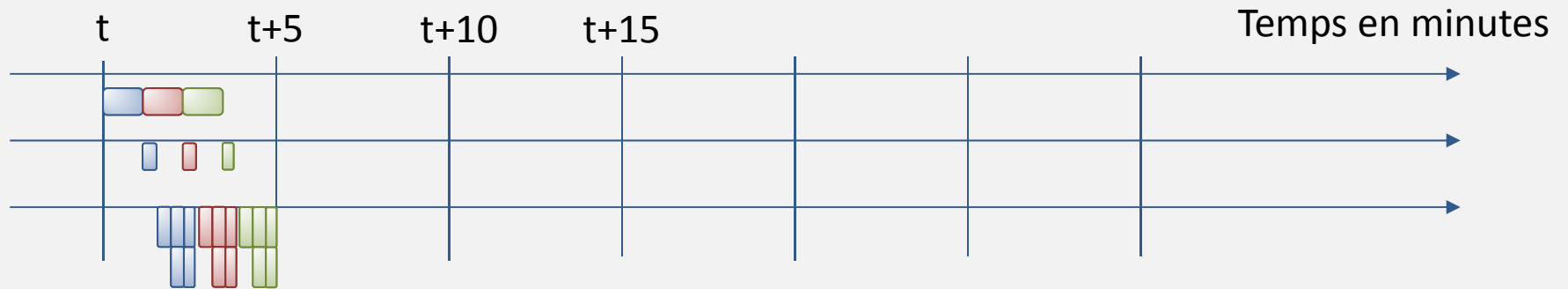
# Les applications

- En Python, utilisation de la classe Java « streaming » de Hadoop
- Recherche de flux (à la demande)
- Agrégation de données (en continu) :
  - TOP n IPs
  - TOP n AS BGP
  - Détection anomalies (DDOS)
  - ...

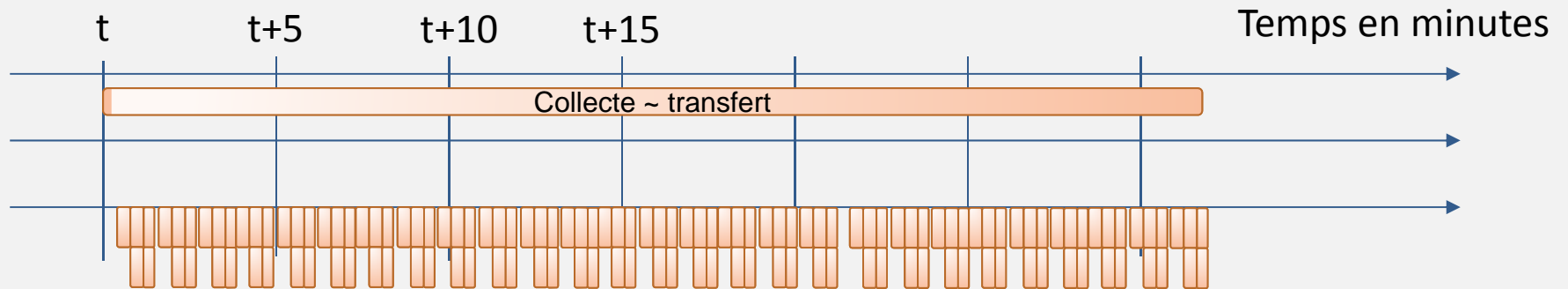
# Contraintes de temps



# Contraintes de temps



# Contraintes de temps



# Exemples de Jobs

[QUICK LINKS](#)

Maps	Reduces	Total Submissions	Nodes	Map Task Capacity	Reduce Task Capacity	Avg. Tasks/Node	Blacklisted Nodes
2	6	10615	<a href="#">10</a>	320	160	48,00	<a href="#">0</a>

## Scheduling Information

Queue Name	Scheduling Information
<a href="#">default</a>	N/A

 Filter (Jobid, Priority, User, Name) 

Example: 'user:smith 3200' will filter by 'smith' only in the user field and '3200' in all fields

## Running Jobs

Jobid	Priority	User	Name	Map % Complete	Map Total	Maps Completed	Reduce % Complete	Reduce Total	Reduces Completed	Job Scheduling Information
<a href="#">job_201310171329_10895</a>	NORMAL	hduser	streamjob3779241140271653301.jar	<input type="text" value="100,00%"/>	8	8	<input type="text" value="0,00%"/>	6	0	NA
<a href="#">job_201310171329_10896</a>	NORMAL	hduser3	streamjob5966715402041840844.jar	<input type="text" value="0,00%"/>	8	0	<input type="text" value="0,00%"/>	6	0	NA
<a href="#">job_201310171329_10897</a>	NORMAL	hduser4	streamjob5652286642674746771.jar	<input type="text" value="50,00%"/>	8	4	<input type="text" value="0,00%"/>	6	0	NA
<a href="#">job_201310171329_10898</a>	NORMAL	hduser2	streamjob5244516507197695747.jar	<input type="text" value="0,00%"/>	8	0	<input type="text" value="0,00%"/>	6	0	NA

# Exemples de Jobs

[QUICK LINKS](#)

Maps	Reduces	Total Submissions	Nodes	Map Task Capacity	Reduce Task Capacity	Avg. Tasks/Node	Blacklisted Nodes
2	6	10615	<a href="#">10</a>	320	160	48,00	<a href="#">0</a>

## Scheduling Information

Queue Name	Scheduling Information
<a href="#">default</a>	N/A

 Filter (Jobid, Priority, User, Name) 

Example: 'user:smith 3200' will filter by 'smith' only in the user field and '3200' in all fields

## Running Jobs

Jobid	Priority	User	Name	Map % Complete	Map Total	Maps Completed	Reduce % Complete	Reduce Total	Reduces Completed	Job Scheduling Information
<a href="#">job_201310171329_10895</a>	NORMAL	hduser	streamjob3779241140271653301.jar	<input type="text" value="100,00%"/>	8	8	<input type="text" value="0,00%"/>	6	0	NA
<a href="#">job_201310171329_10896</a>	NORMAL	hduser3	streamjob5966715402041840844.jar	<input type="text" value="0,00%"/>	8	0	<input type="text" value="0,00%"/>	6	0	NA
<a href="#">job_201310171329_10897</a>	NORMAL	hduser4	streamjob5652286642674746771.jar	<input type="text" value="50,00%"/>	8	4	<input type="text" value="0,00%"/>	6	0	NA
<a href="#">job_201310171329_10898</a>	NORMAL	hduser2	streamjob5244516507197695747.jar	<input type="text" value="0,00%"/>	8	0	<input type="text" value="0,00%"/>	6	0	NA

# Exemples de Jobs

[Quick Links](#)

Maps	Reduces	Total Submissions	Nodes	Map Task Capacity	Reduce Task Capacity	Avg. Tasks/Node	Blacklisted Nodes
2	6	10615	<a href="#">10</a>	320	160	48,00	<a href="#">0</a>

## Scheduling Information

Queue Name	Scheduling Information
<a href="#">default</a>	N/A

 Filter (Jobid, Priority, User, Name) 

Example: 'user:smith 3200' will filter by 'smith' only in the user field and '3200' in all fields

## Running Jobs

Jobid	Priority	User	Name	Map % Complete	Map Total	Maps Completed	Reduce % Complete	Reduce Total	Reduces Completed	Job Scheduling Information
<a href="#">job_201310171329_10895</a>	NORMAL	hduser	streamjob3779241140271653301.jar	<div style="width: 100%;"><div style="background-color: #0070C0; height: 10px;"></div></div> 100,00%	8	8	<div style="width: 0%;"><div style="background-color: #0070C0; height: 10px;"></div></div> 0,00%	6	0	NA
<a href="#">job_201310171329_10896</a>	NORMAL	hduser3	streamjob5966715402041840844.jar	<div style="width: 0%;"><div style="background-color: #0070C0; height: 10px;"></div></div> 0,00%	8	0	<div style="width: 0%;"><div style="background-color: #0070C0; height: 10px;"></div></div> 0,00%	6	0	NA
<a href="#">job_201310171329_10897</a>	NORMAL	hduser4	streamjob5652286642674746771.jar	<div style="width: 50%;"><div style="background-color: #0070C0; height: 10px;"></div></div> 50,00%	8	4	<div style="width: 0%;"><div style="background-color: #0070C0; height: 10px;"></div></div> 0,00%	6	0	NA
<a href="#">job_201310171329_10898</a>	NORMAL	hduser2	streamjob5244516507197695747.jar	<div style="width: 0%;"><div style="background-color: #0070C0; height: 10px;"></div></div> 0,00%	8	0	<div style="width: 0%;"><div style="background-color: #0070C0; height: 10px;"></div></div> 0,00%	6	0	NA

# Exemples de Jobs

[Quick Links](#)

Maps	Reduces	Total Submissions	Nodes	Map Task Capacity	Reduce Task Capacity	Avg. Tasks/Node	Blacklisted Nodes
2	6	10615	<a href="#">10</a>	320	160	48,00	<a href="#">0</a>

## Scheduling Information

Queue Name	Scheduling Information
<a href="#">default</a>	N/A

 Filter (Jobid, Priority, User, Name) 

Example: 'user:smith 3200' will filter by 'smith' only in the user field and '3200' in all fields

## Running Jobs

Jobid	Priority	User	Name	Map % Complete	Map Total	Maps Completed	Reduce % Complete	Reduce Total	Reduces Completed	Job Scheduling Information
<a href="#">job_201310171329_10895</a>	NORMAL	hduser	streamjob3779241140271653301.jar	<div style="width: 100%;"><div>100,00%</div></div>	8	8	<div style="width: 0%;"><div>0,00%</div></div>	6	0	NA
<a href="#">job_201310171329_10896</a>	NORMAL	hduser3	streamjob5966715402041840844.jar	<div style="width: 0%;"><div>0,00%</div></div>	8	0	<div style="width: 0%;"><div>0,00%</div></div>	6	0	NA
<a href="#">job_201310171329_10897</a>	NORMAL	hduser4	streamjob5652286642674746771.jar	<div style="width: 50%;"><div>50,00%</div></div>	8	4	<div style="width: 0%;"><div>0,00%</div></div>	6	0	NA
<a href="#">job_201310171329_10898</a>	NORMAL	hduser2	streamjob5244516507197695747.jar	<div style="width: 0%;"><div>0,00%</div></div>	8	0	<div style="width: 0%;"><div>0,00%</div></div>	6	0	NA

## Temps d'exécution des applications

Données en Go	Période observée	temps	# tâches Map	# blocs traités sur place	# blocs déplacés	# tâches Reduce
6,8	5min	44s	105	91	14	16
13,4	10min	37s	203	190	13	16
13,4	10min	84s	203	196	7	64
80,9	1h	76s	1220	1210	10	16
823	~16h	8m50	12553	12532	21	16

## Temps d'exécution des applications

Données en Go	Période observée	temps	# tâches Map	# blocs traités sur place	# blocs déplacés	# tâches Reduce
6,8	5min	44s	105	91	14	16
13,4	10min	37s	203	190	13	16
13,4	10min	84s	203	196	7	64
80,9	1h	76s	1220	1210	10	16
823	~16h	8m50	12553	12532	21	16

## Temps d'exécution des applications

Données en Go	Période observée	temps	# tâches Map	# blocs traités sur place	# blocs déplacés	# tâches Reduce
6,8	5min	44s	105	91	14	16
13,4	10min	37s	203	190	13	16
13.4	10min	84s	203	196	7	64
80,9	1h	76s	1220	1210	10	16
823	~16h	8m50	12553	12532	21	16

## Temps d'exécution des applications

Données en Go	Période observée	temps	# tâches Map	# blocs traités sur place	# blocs déplacés	# tâches Reduce
6,8	5min	44s	105	91	14	16
13,4	10min	37s	203	190	13	16
13,4	10min	84s	203	196	7	64
80,9	1h	76s	1220	1210	10	16
823	~16h	8m50	12553	12532	21	16



## Temps d'exécution des applications

Données en Go	Période observée	temps	# tâches Map	# blocs traités sur place	# blocs déplacés	# tâches Reduce
6,8	5min	44s	105	91	14	16
13,4	10min	37s	203	190	13	16
13,4	10min	84s	203	196	7	64
80,9	1h	76s	1220	1210	10	16
823	~16h	8m50	12553	12532	21	16

# Quelques remarques

- Modification de la réplication : attention !
- Ajout d'un nœud : sans problème
- stop/start Hadoop : attention au temps de démarrage de HDFS
- Modification des paramètres d'optimisation
  - À faire en fonction de vos applications et non en fonction des benchmarks du web

# Après

- Utilisation d'un ressource manager !!
  - Hadoop 2 (Yarn)
- Update temps réel (Flume)
- Compression ? Pas très *green*
- Spark, *better than Hadoop ?*
- *Hadoop & Cloud*

# Conclusion

- *Big Data analytics ?*
  - *Take it easy with Hadoop*

# Remerciements

- Ker Data Team (INRIA Rennes - Bretagne Atlantique et ENS Cachan - Antenne de Bretagne)
- JRES