

Comparatif des systèmes de stockage distribués en écriture intensive

10^{èmes} journées informatique de l'IN2P3-CEA/IRFU, 26-29/09/2016

Plan de la présentation

Présentation de l'expérience à l'origine du besoin

Besoins de stockage de l'expérience

Plate-forme de tests

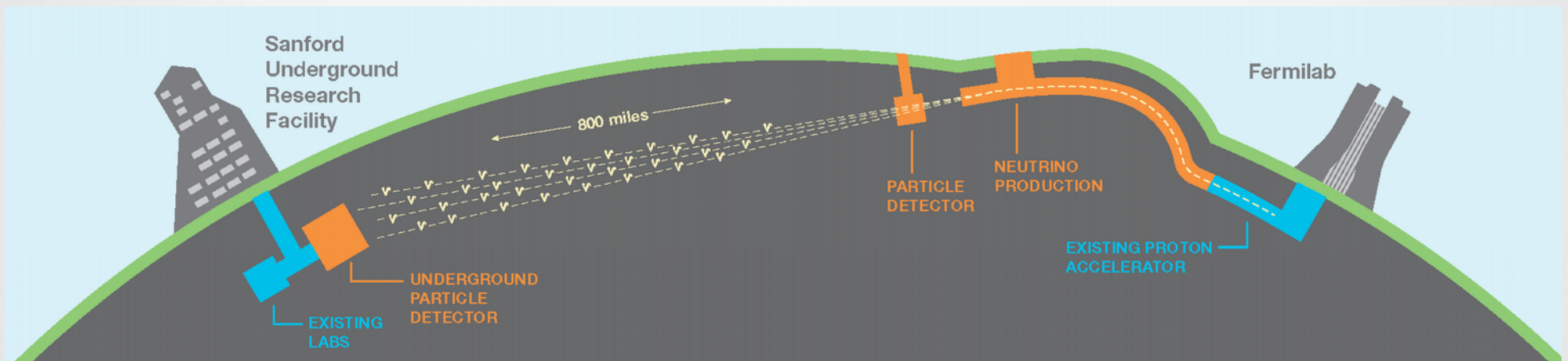
Méthodologie

Résultats des tests

Conclusion

(proto)DUNE <http://www.dunescience.org/>

- DUNE : Deep Underground Neutrino Experiment
- Expérience d'oscillations de neutrinos de nouvelle génération (mesure de la hiérarchie de masses, étude CP)



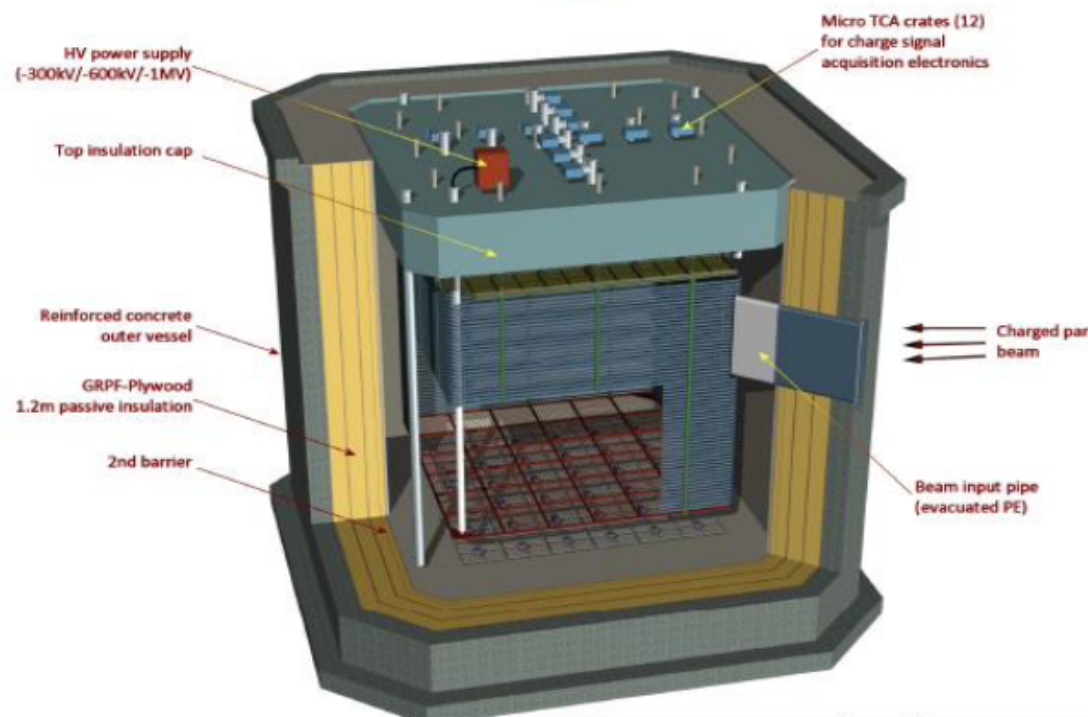
- Faisceau + détecteur proche @ Fermilab
- Détecteur lointain (TPC argon liquide) @ Sanford
- 2 technologies : single (US) / dual (Europe) phase
- 2 expériences protoDUNE @ CERN : NP02/WA105 (dual phase) et NP04 (single phase)

Dual-phase protoDUNE : WA105

WA105

DLAr 6x6x6m³ design

- Membrane GTT® tank with passive insulation
- Top deck with chimneys and insulation
- **6x6m² anode large readout area**, **6m long drift length** (3ms max drift time @ 1kV/cm)
- Charged particle beam window
- 300 ton LAr instrumented: 7680 charge readout channels, 36 PMTs (baseline layout)



Liquid argon density	T/m ³	1.38
Liquid argon volume height	m	7.6
Active liquid argon height	m	5.99
Hydrostatic pressure at the bottom	bar	1.03
Inner vessel size (WxLxH)	m ³	8.3 × 8.3 × 8.1
Inner vessel base surface	m ²	67.6
Total liquid argon volume	m ³	509.6
Total liquid argon mass	t	705
Active LAr area	m ²	36
Charge readout module (0.5 x0.5 m ²)		36
N of signal feedthrough		12
N of readout channels		7680
N of PMT		36

Crédit : J.Martéau (IPNL)

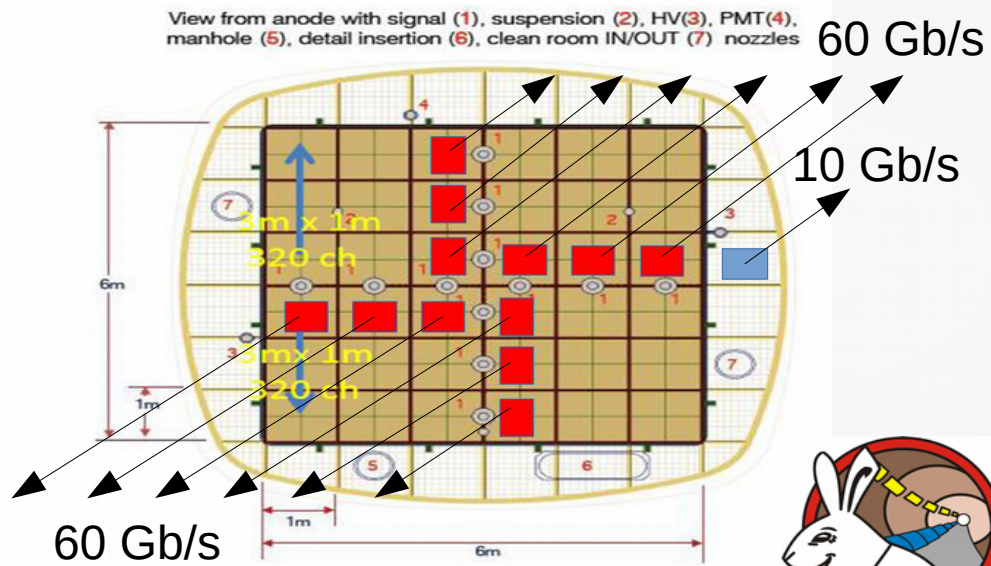
WA105 μ TCA-DAQ architecture

2 x 40 Gb/s

E.B.1



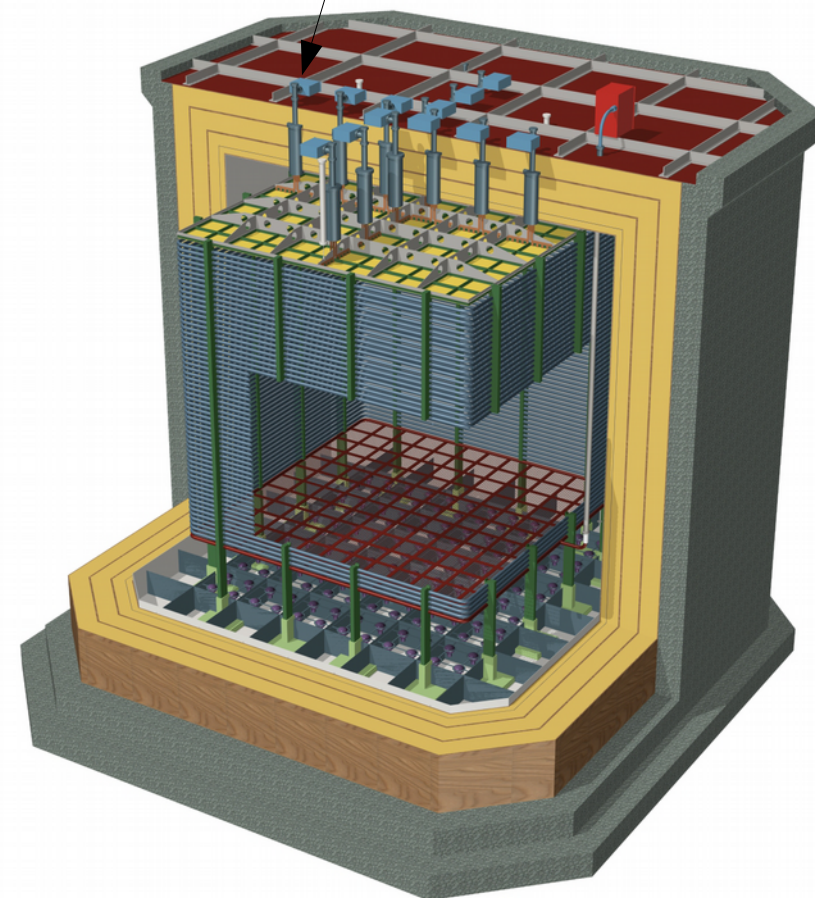
View from anode with signal (1), suspension (2), HV(3), PMT(4), manhole (5), detail insertion (6), clean room IN/OUT (7) nozzles



μ TCA.1 option

60 Gb/s

10 Gb/s



E.B.2

2 x 40 Gb/s

WA105 data network

B/E
(storage/
processing)



Storage :
15 disks servers R730
2 metadata servers R630
1 config. server R430



Processing :
16 lames M630
16x24 = 384 cores

C.C.

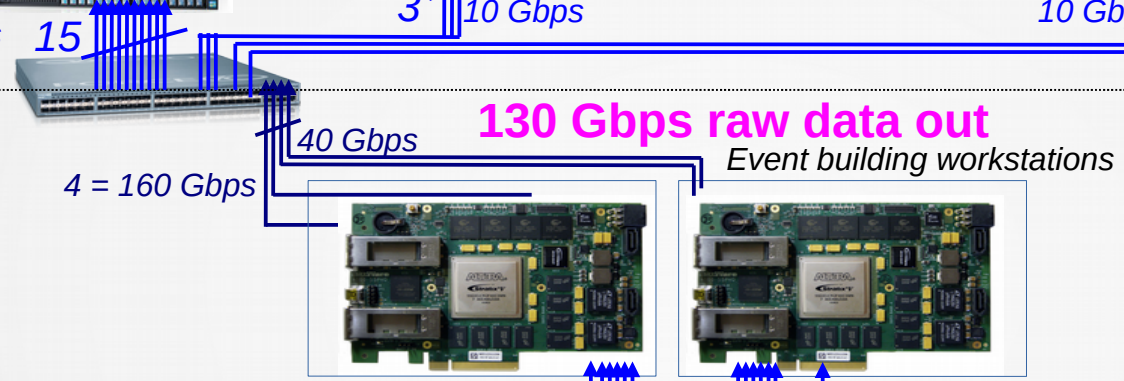
20 Gbps comp. data out

C.R.

CERN C.C.

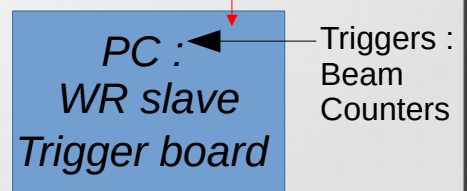
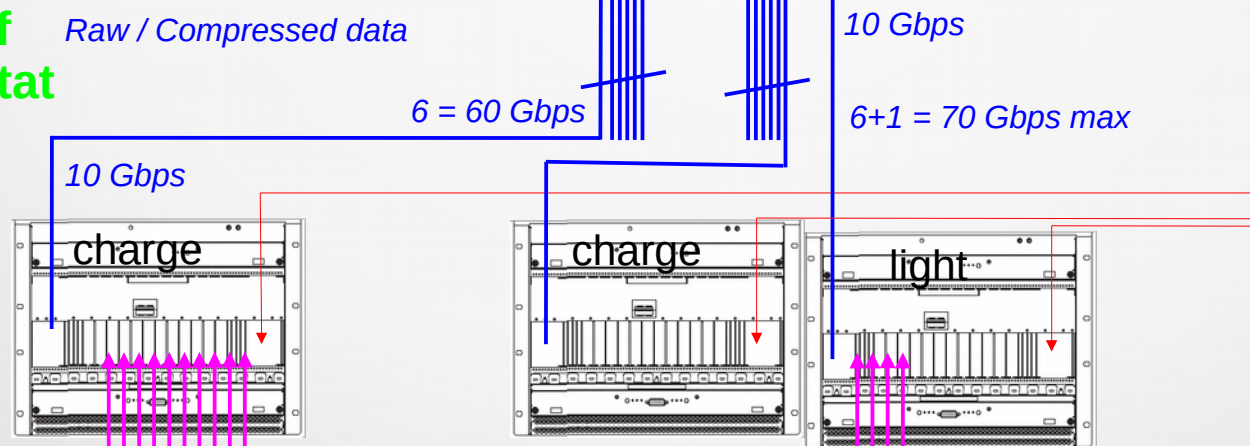
B/E
(sorting/
Filtering/
Clock)

C.R.



F/E-out :
Charge + PMT

Top of
cryostat



C.R.

F/E-in

LAr

10 Raw data : charge

Raw data : light

Crédit : J.Marteau (IPNL)



Besoins de stockage de l'expérience

- Taille 1 Po (buffer local de 1 à 3 jours de prise de données)
- La solution de stockage doit pouvoir :
 - absorber 130Gb/s (60Gb/s + 70Gb/s) en écriture pendant la phase de debug
 - Pendant phase de production : Facteur de compression (Huffman) estimé a ~90 %
 - Permettre la lecture des données pendant l'enregistrement afin de déterminer la qualité des données
 - => Prise de décision : stopper / continuer la prise de données
 - La recopie des données vers un stockage externe (CERN)
- Recherche de candidats de systèmes de stockage :
 - Sachant exploiter efficacement les capacités des serveurs de données
 - **Économies en cycles CPU coté client**

=> objectif de ce 1^{er} test : Caractériser les systèmes de stockage sous l'angle de la performance en écriture

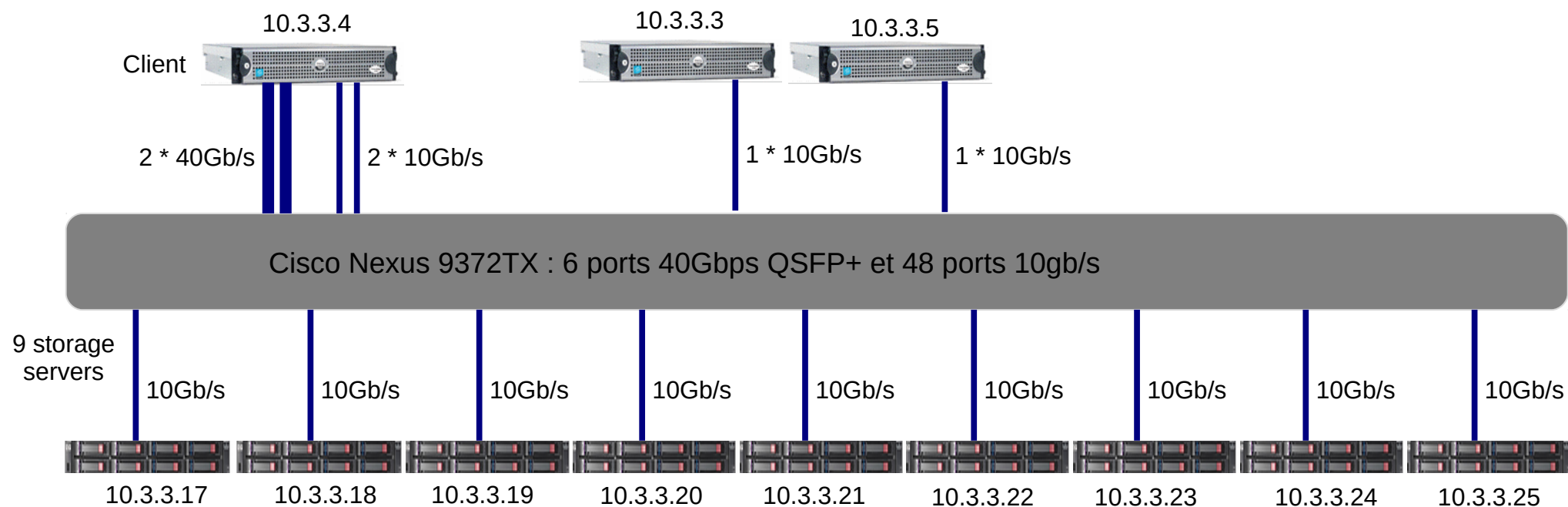
Plateforme ...temporaire... de tests

Client : Dell R630

- 1 CPU E5-2637 @ 3.5Ghz (4c, 8c HT),
- 32Go RAM 2133 Mhz DDR4
- 2 * Mellanox CX313A 40Gb/s
- 2 * 10Gb/s (X540-AT2)
- CentOS 7.0

MDS / Managment : 2 * Dell R630

- 1 CPU E5-2637 @ 3.5Ghz (4c, 8c HT),
- 32Go RAM 2133 Mhz DDR4
- 2 * 10Gb/s (X540-AT2)
- Scientific Linux 6.5 et Centos 7.0



9 Storage Servers : (9 * Dell R510)

- 2 * CPU E5620 @ 2.40GHz (4c, 8c HT), 16Go RAM
- 1 carte PERC H700 (512MB) : 1 Raid 6 12HDD 2TB (10D+2P) = 20TB
- 1 Ethernet intel 10Gb/s (X520/X540)
- Scientific Linux 6.5

Systemes de stockage testés

	Lustre	BeeGFS	GlusterFS	GPFS	MooseFS	XtreemFS	XRootD	EOS
Versions	v2.7.0-3	v2015.03.r10	3.7.8-4	v4.2.0-1	2.0.88-1	1.5.1	4.3.0-1	Citrine 4.0.12
POSIX	Oui	Oui	Oui	Oui	Oui	Oui	via FUSE	via FUSE
Open Source	Oui	Client=Oui, Serveur=EULA	Oui	Non	Oui	Oui	Oui	Oui
MDS nécessaire	Oui	Metadata + Manager	Non	Non	Metadata + Manager		Oui	Oui
Support RDMA / Infiniband	Oui	Oui	Oui	Oui	Non	Non	Non	Non
Striping	Oui	Oui	Oui	Oui	Non	Oui	Non	Non
Failover	M + D (1)	DR (1)	M + D (1)	M + D (1)	M + DR (1)	M + DR (1)	Non	M + D (1)
Quota	Oui	Oui	Oui	Oui	Oui	Non	Non	Oui
Snapshots	Non	Non	Oui	Oui	Oui	Oui	Non	Non
Outils de déplacement des données	Oui	Oui	Oui	Oui	Non	Oui	Non	Oui

(1) : M=Metadata, D=Data, M+D=Metadata+Data, DR=Data Replication

Notes sur les systèmes de stockage testés

Tous sont dans la classe « software defined storage »

– Systèmes de fichiers :

- GPFS, Lustre et BeeGFS sont connus dans le monde HPC (High Performance Computing) : ce sont des systèmes de fichiers parallèles performants en présence de nombreux clients et de nombreux serveurs de données.
- Je souhaitais aussi en profiter pour tester GlusterFS, MooseFS, XtremFS pour connaître leurs caractéristiques

– Systèmes de stockage :

- XRootD est un protocole de transfert de données très populaire en High Energy Physics, il s'intègre bien avec ROOT (le framework et le format de fichier)
- EOS : est un système de stockage massif (135PB @CERN), dont l'accès est multi-protocole (XRootD, http(s), webdav, POSIX-like FUSE)

Tous ces systèmes ont leur avantages et inconvénients respectifs, non discutés ici

Méthodologie adoptée

1 : Tests réseau

+

2 : Tests du client

+

3 : Tests du stockage

+

4 : Tests de la chaîne
complète

Tests principalement protocolaires :

- Protocoles TCP / UDP : iperf, nuttcp...
- Saturation des interfaces : Algorithmes de contrôle de congestion : cubic, reno, bic, htcp...
- UDP : % de perte de paquets
- TCP : nombre de retransmissions
- Drop de paquets
- Puis reste des tests en TCP, flux en écriture

Connaître précisément quel flux peut générer le client :

Test initiaux => optimisations => caractérisation

- Optimisations :
 - Bonding réseau : test des algorithmes LACP, balance-alb, balance-tlb
 - Optimisation buffers réseau : modif /etc/sysctl.conf
 - Utilisation de Jumbo frames (MTU 9216)
 - Charge CPU : répartition des IRQ sur tous les coeurs
 - chkconfig irqbalance off ; service irqbalance stop
 - Mellanox : set_irq_affinity.sh p2p1

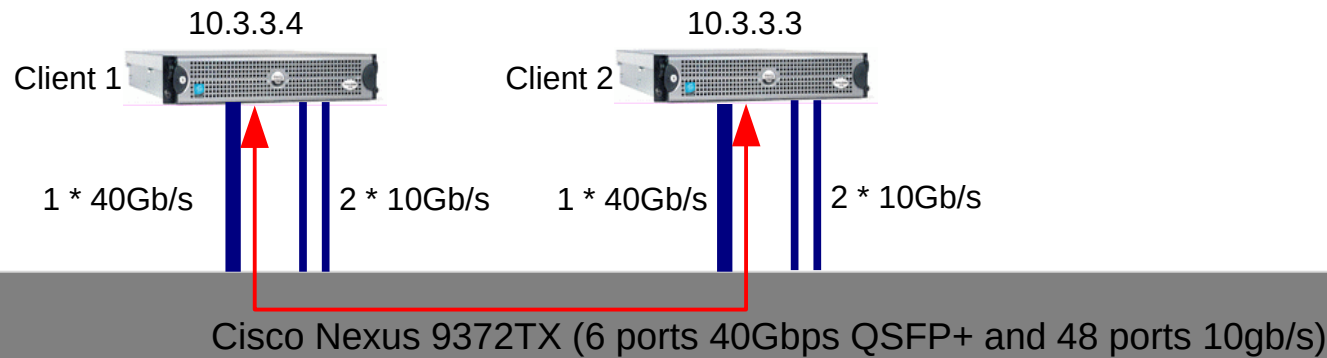
Tests individuels des éléments de stockage :

- lozone / fio / dd : benchmark du système de fichier local

Tests de la chaîne complète :

- Sur le client
 - Stockage : lozone, fio, dd, xrdcp
 - Réseau / système : dstat
- Sur les éléments de stockage : dstat

1-a. Tests réseau entre 2 clients



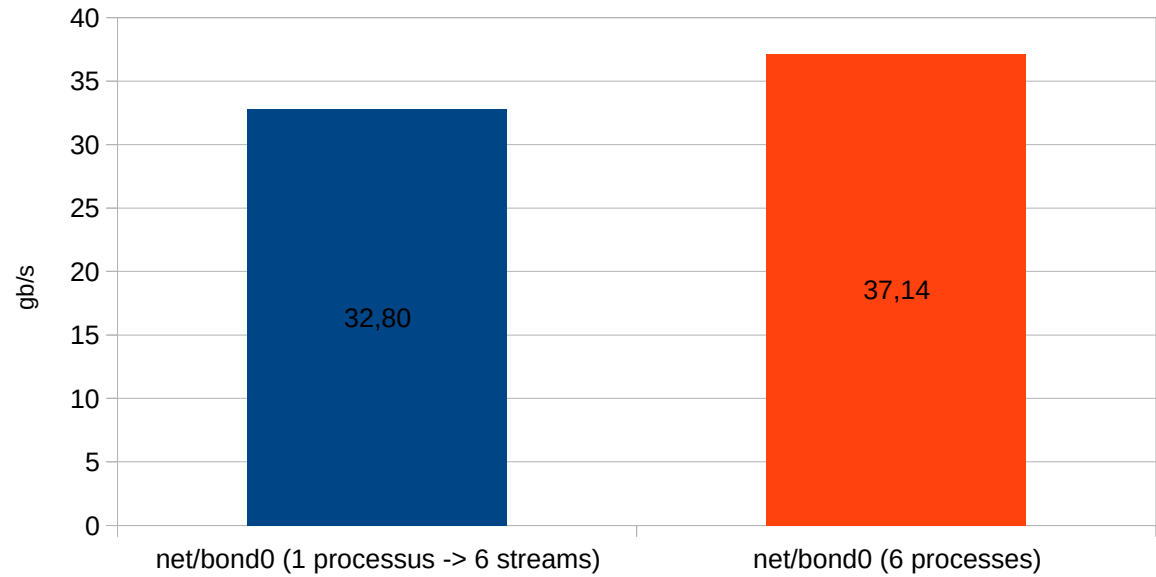
Quel est le comportement des flux réseau entre 2 clients avec pour chacun 1 * 40Gb/s + 2 * 10Gb/s ?

- Tests réalisés avec beaucoup de paramètres réseau :
 - net.ipv4.tcp_congestion_control = cubic, MTU=9216, irq affinity on all CPU cores, tuning mellanox,
 - Bonding, tests plusieurs algorithmes : mode=balance-tlb, balance-alb (xmit_hash_policy=layer2 ou layer3+4), but not LACP (IEEE 802.3ad)
- Network bandwidth tests between **only 2 « clients » with 1*40Gb/s + 2*10Gb/s each** (nuttcp, 5 seconds test)
 - **Only 1 flow** between 10.3.3.3 et 10.3.3.4 : TCP = **34746 Mb/s (0 TCP retrans)**,
UDP = 38561Mb/s (2.74 % UDP packet losses)
 - **1 processus, 6 flows** between 10.3.3.3 et 10.3.3.4 : TCP = **35058 Mb/s (0 TCP retrans)**
 - **6 processus (1 flow / processus)** between 10.3.3.3 et 10.3.3.4 : TCP = **39532 Mb/s (0 TCP retrans)**

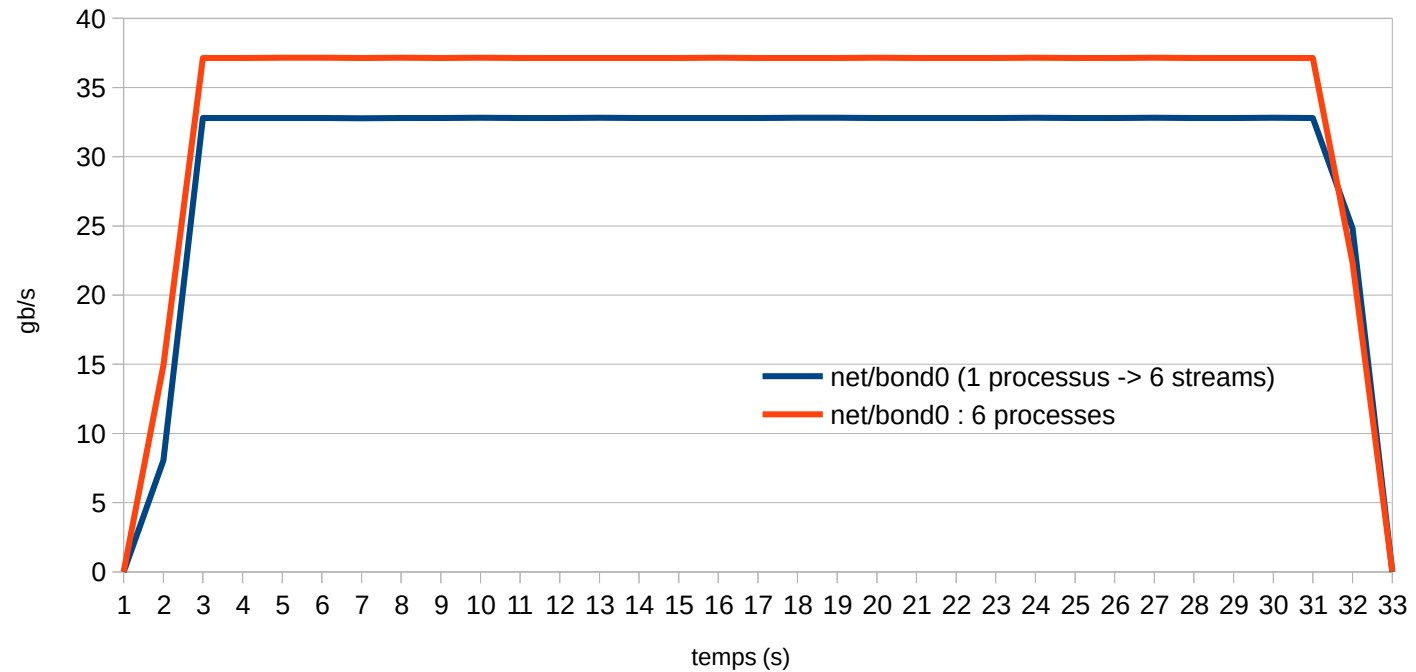
Comparaison : 1 vs 6 processus :

- Comparaison de bande passante de :
 - 1 process qui génère 6 flux
 - 6 processus, 1 flux / processus
- 30 secondes test
- Carte 40Gb/s proche de la saturation
- Le flux ne passe pas par les cartes 2*10Gb/s (all bonding algorithms tested)
- +12.7 % quand les flux sont générés par 6 processus indépendants

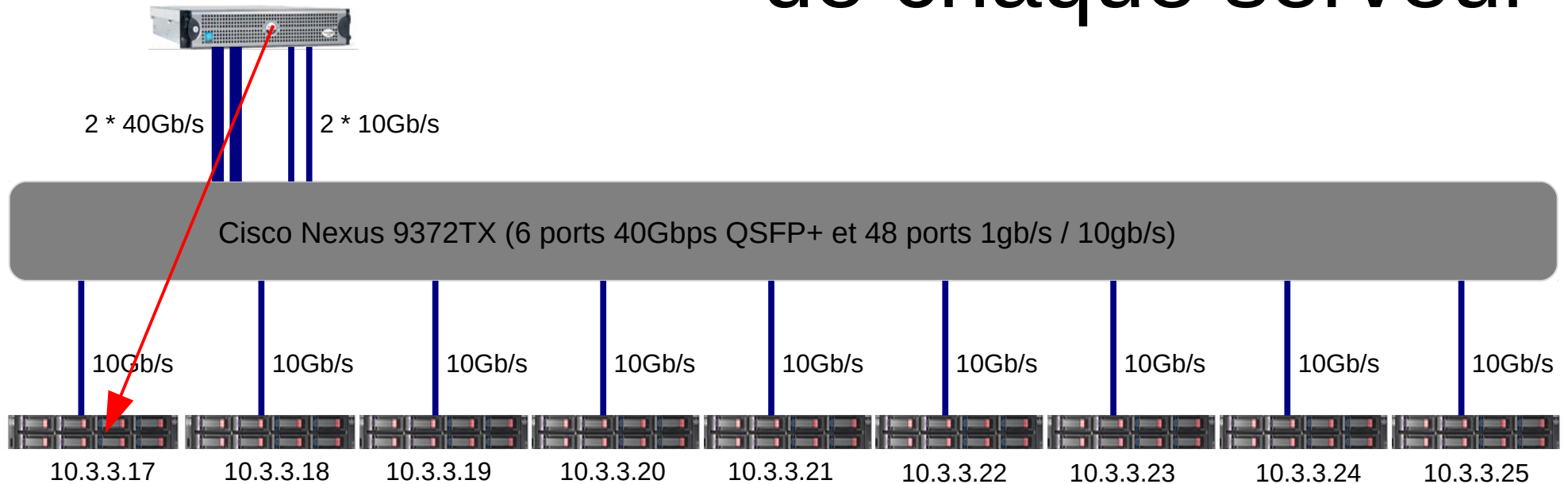
Tests between 2 clients 1*40gb/s + 2 * 10gb/s (TCP)



Tests between 2 clients 1*40gb/s + 2 * 10gb/s (TCP)

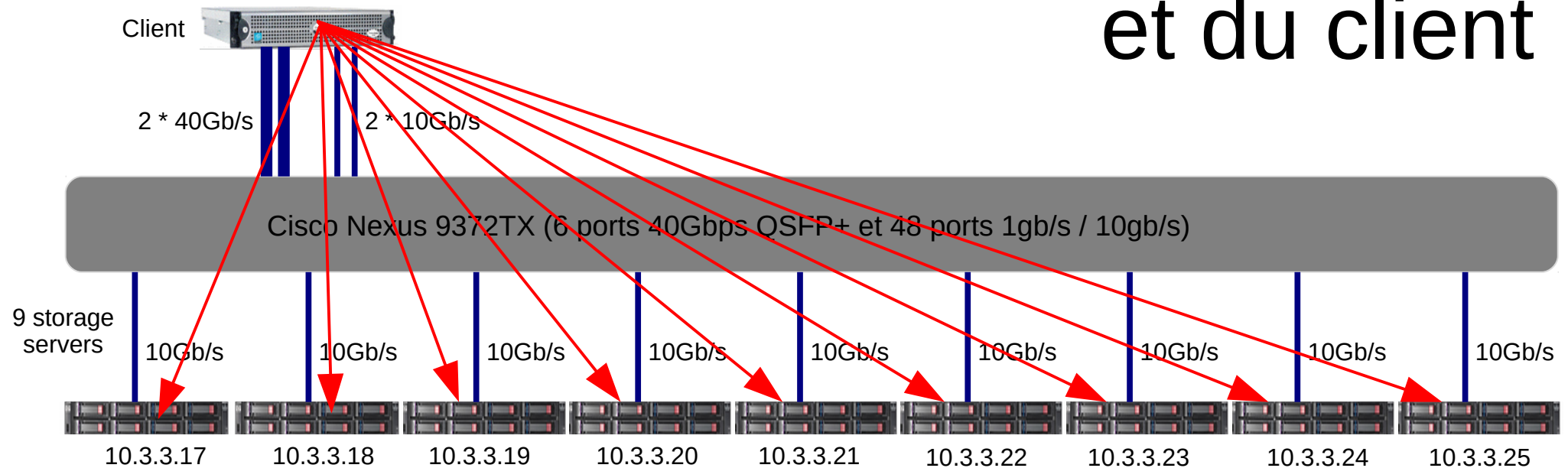


1. Tests individuels du réseau de chaque serveur



- Tests bande passante réseau de chaque serveur de stockage (nuttcp)
 - Individuellement : 1 flux TCP ou UDP vers 1 serveur :
 - TCP client → serveur : somme des 9 = 87561.23 Mb/s
 - TCP serveur → client : somme des 9 = 89190.71 Mb/s
 - UDP client → serveur : somme des 9 = 52761.45 Mb/s
 - UDP serveur → client : somme des 9 = 70709.24 Mb/s

1. et 2. Tests du réseau (tous les serveurs simultanément) et du client

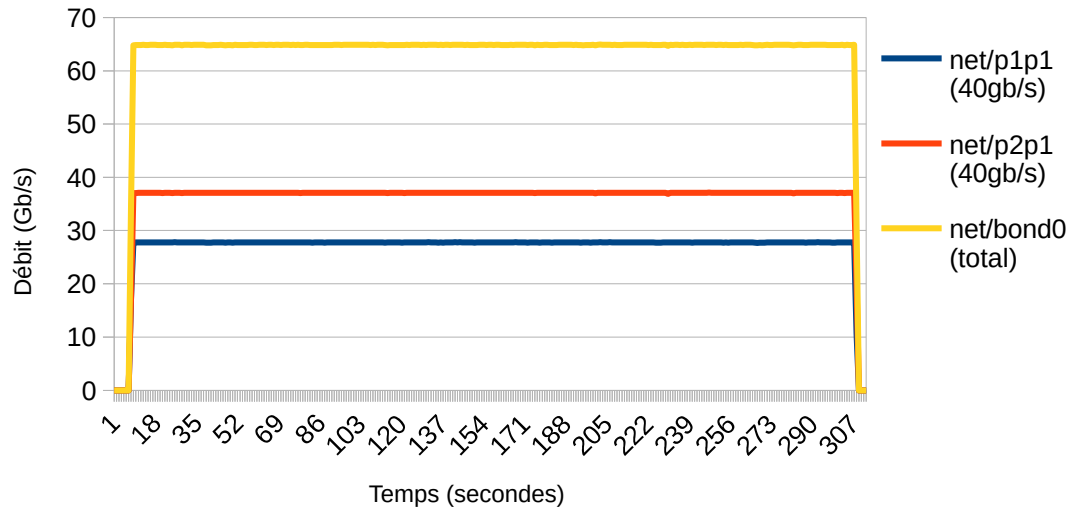


Comment se comportent les différents algorithmes de bonding ?

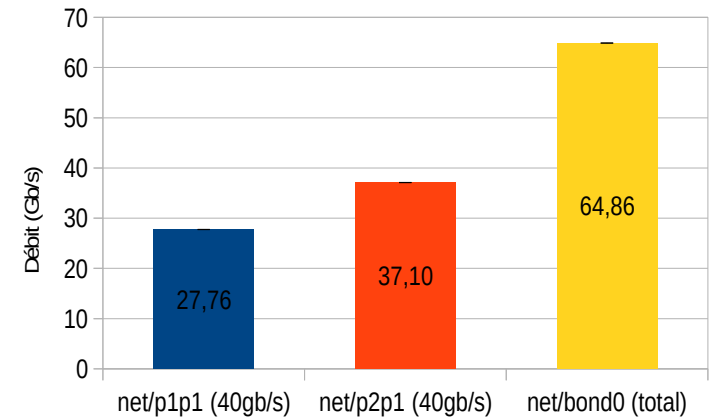
- Envoi de 9 flux TCP simultanés (1 vers chaque serveur) pendant 5 minutes (nuttcp)
 - 1er test : nous testons individuellement chaque carte 40Gb/s → 9 serveurs : saturation de la carte 40Gb/s
 - 2^{eme} test : Client Bonding avec seulement 2 * 40Gb/s → 9 serveurs :
 - Algorithmes de bonding testés : mode=balance-alb, balance-tlb, LACP
 - Grande variation des mesures (sauf LACP), meilleur = LACP (802.3ad xmit_hash_policy=layer2+3)
 - 3^{eme} test : Client bonding avec 2 * 40Gb/s + 2 * 10Gb/s → 9 serveurs :
 - Bonding testé : mode=balance-alb, balance-tlb mais pas LACP (incompatible)
 - Grande variation des mesures, meilleur = balance-alb xmit_hash_policy layer2+3

2nd test : bonding with 2*40gb/s, best = 802.3ad xmit_hash_policy=layer2+3

1 client, 2*40gb/s, bonding 802.ad (LACP), xmit_hash policy=layer2-3

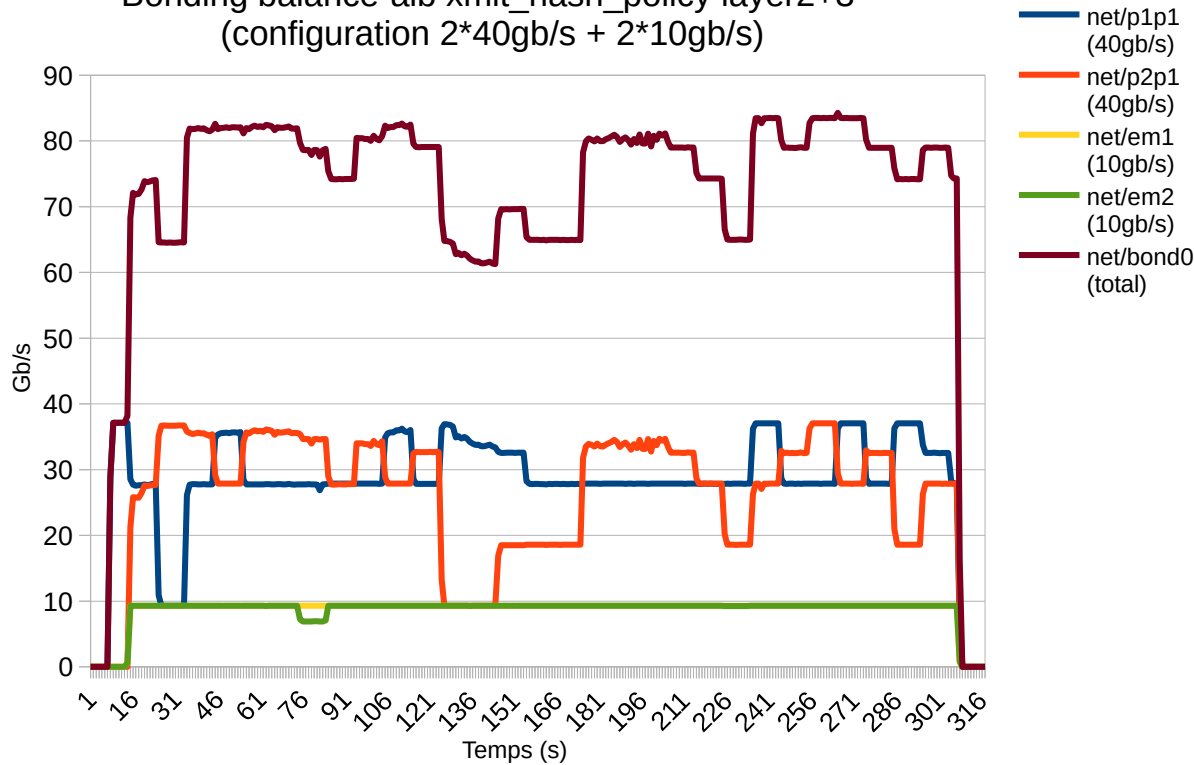


1 client, 2*40gb/s, bonding 802.ad (LACP), xmit_hash policy=layer2-3

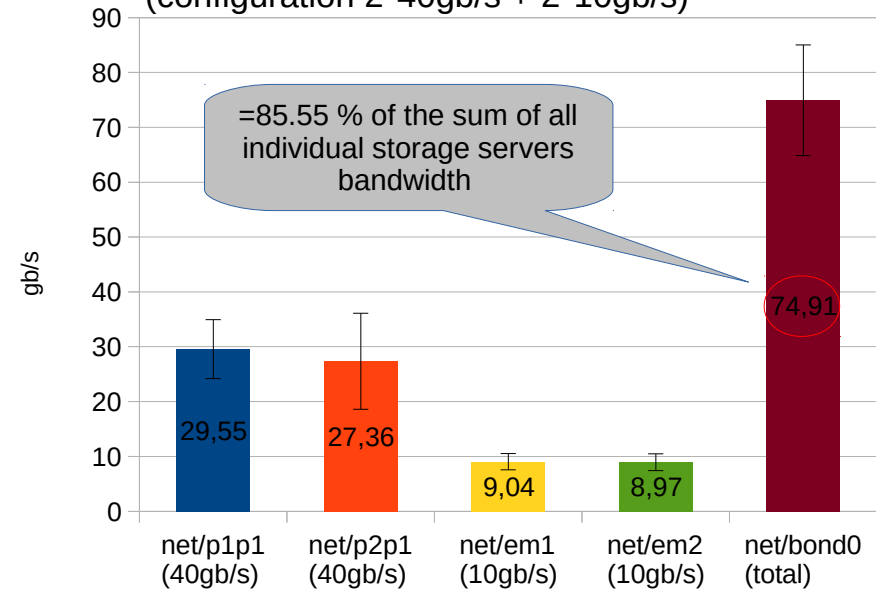


3rd test : bonding with 2*40gb/s + 2*10gb/s, best = balance-alb xmit_hash_policy=layer2+3

Bonding balance-alb xmit_hash_policy layer2+3
(configuration 2*40gb/s + 2*10gb/s)



Bonding balance-alb xmit_hash_policy layer2+3
(configuration 2*40gb/s + 2*10gb/s)



3. Test des éléments de stockage

- Configuration du client : 1 seul client avec 2 * 40Gb/s + 2 * 10Gb/s
- Configuration du stockage : 9 éléments de stockage
 - 1 carte ethernet 10Gb/s Intel X520
 - 1 Raid 6 sur 12 disques de 2TB (10 Data + 2 Parity)
 - ~20 TB disponibles / serveur
 - Stripe size 1M
- Utilisation d'outils standards :
 - fio (read, write, readwrite, randread, randwrite, randrw), avec variation sur la taille et nb processus concurrents
 - iozone (write, read, random-read/write, random_mix) avec variation sur la taille et nb processus concurrents
 - dd (sync, async, direct...)
- Le challenge est sur le débit d'écriture séquentielle sur les éléments de stockage
 - test dd (écriture séquentielle avec et sans buffer d'E/S) :

```
# dd if=/dev/zero of=test10G.dd bs=1M count=10000 oflag=sync  
10485760000 bytes (10 GB) copied, 22,6967 s, 462 MB/s  
  
# dd if=/dev/zero of=test10G.dd bs=1M count=10000 oflag=direct  
10485760000 bytes (10 GB) copied, 9,91637 s, 1,1 GB/s
```

À se rappeler pour la suite : 462 MB/s est le débit en écriture séquentielle qu'un serveur de ce type peut absorber

- Test fio (écriture random) :

```
# fio --name=randwrite --ioengine=libaio --iodepth=1 --rw=randwrite --bs=4k  
--direct=0 --size=512M --numjobs=8 --runtime=240 --group_reporting  
  
bw=508339KB/s
```

Systemes de stockage testés

Rappel :

	Lustre	BeeGFS	GlusterFS	GPFS	MooseFS	XtreemFS	XRootD	EOS
Versions	v2.7.0-3	v2015.03.r10	3.7.8-4	v4.2.0-1	2.0.88-1	1.5.1	4.3.0-1	Citrine 4.0.12
POSIX	Oui	Oui	Oui	Oui	Oui	Oui	via FUSE	via FUSE
Open Source	Oui	Oui	Oui	Oui	Oui	Oui	Oui	Oui
MDS nécessaire	Oui	Oui	Oui	Oui	Oui	Oui	Oui	Oui
Support RDMA / Infiniband	Oui	Oui	Oui	Oui	Non	Non	Non	Non
Striping	Oui	Oui	Oui	Oui	Non	Oui	Non	Non
Failover	M + D (1)	DR (1)	M + D (1)	M + D (1)	M + DR (1)	M + DR (1)	Non	M + D (1)
Quota	Oui	Oui	Oui	Oui	Oui	Non	Non	Oui
Snapshots	Non	Non	Oui	Oui	Oui	Oui	Non	Non
Outils de déplacement des données	Oui	Oui	Oui	Oui	Non	Oui	Non	Oui

Chaque fichier est divisé en « chunks » distribués vers tous les serveurs de stockage.

C'est toujours à la charge de la CPU du client (DAQ back-end)

(1) : M=Metadata, D=Data, M+D=Metadata+Data, DR=Data Replication

Test de la chaine complète

- Ça se complique :
 - Les systèmes de fichiers locaux sur les éléments de stockage ne sont pas toujours les mêmes (XFS, EXT4, formats propriétaire...)
 - Les paramètres de tuning des systèmes de stockage sont nombreux
 - Le fonctionnement n'est pas toujours comparable d'un produit à l'autre :
 - Lustre permet de créer des fichiers stripés sur 1 à n OSTs (parmi 9 serveurs), la taille du stripe est ajustable (ici 1MB)
 - BeeGFS permet de créer des fichiers stripés sur 1 à n OSTs (parmi 9 serveurs), la taille du stripe est ajustable (ici 1MB)
 - GlusterFS permet de créer des volumes stripés sur n targets (parmi 9 bricks), des volumes distribués (/9 bricks), ou des volumes EC (8+1)
 - GPFS découpe les fichiers en stripes de la taille d'un block (1M), par défaut il stocke les stripes sur tous les NSD (ici 9 NSD). Dans les tests suivants, le nombre de « targets » GPFS correspond au nombre de serveurs mis en production pour réaliser le test (contrairement à Lustre, BeeGFS, Gluster et XtremFS où c'est le nombre de « chunks » qui sont paramétrables).
 - MooseFS stripe les fichiers en chunks de 64MB
 - XtremFS permet de créer des fichiers stripés sur 1 à n OSDs (parmi 9 serveurs), la taille du stripe est ajustable (ici 1MB)
 - XrootD ne stripe pas les fichiers
 - EOS ne stripe pas les fichiers
- Choix de paramètres communs :
 - Block size = 1M
 - Pas de réplication (ni donnée, ni meta-donnée), pas de quota, pas de snapshot

Différences dans les tests

- Différences dans les tests entre les systèmes POSIX et Non POSIX : Dans les graphiques suivants :

- Systèmes de fichiers POSIX : iozone

```
iozone -e -+u -r 1m -s ${size} -i 0 -t ${threads}
```

- XrootD et EOS : xrdcp (client XrootD)

- Méthode :

- Création d'un RAMDISK de 21Go (et montage sur /test-ramdisk)
- Création du fichier de la taille souhaitée dans le RAMDISK

- Copie du fichier avec xrdcp :

```
xrdcp /test-ramdisk/fichier root://$server/$repertoire/f_${i}_t_${thread}
```

- Tests xrdcp :

- Copie de 300 fois le même fichier de 100Mo vers le système de stockage
- Copie de 300 fois le même fichier de 1Go vers le système de stockage
- Copie de 30 fois le même fichier de 10Go vers le système de stockage
- Copie de 15 fois le même fichier de 20Go vers le système de stockage

Tests communs à l'ensemble des systèmes

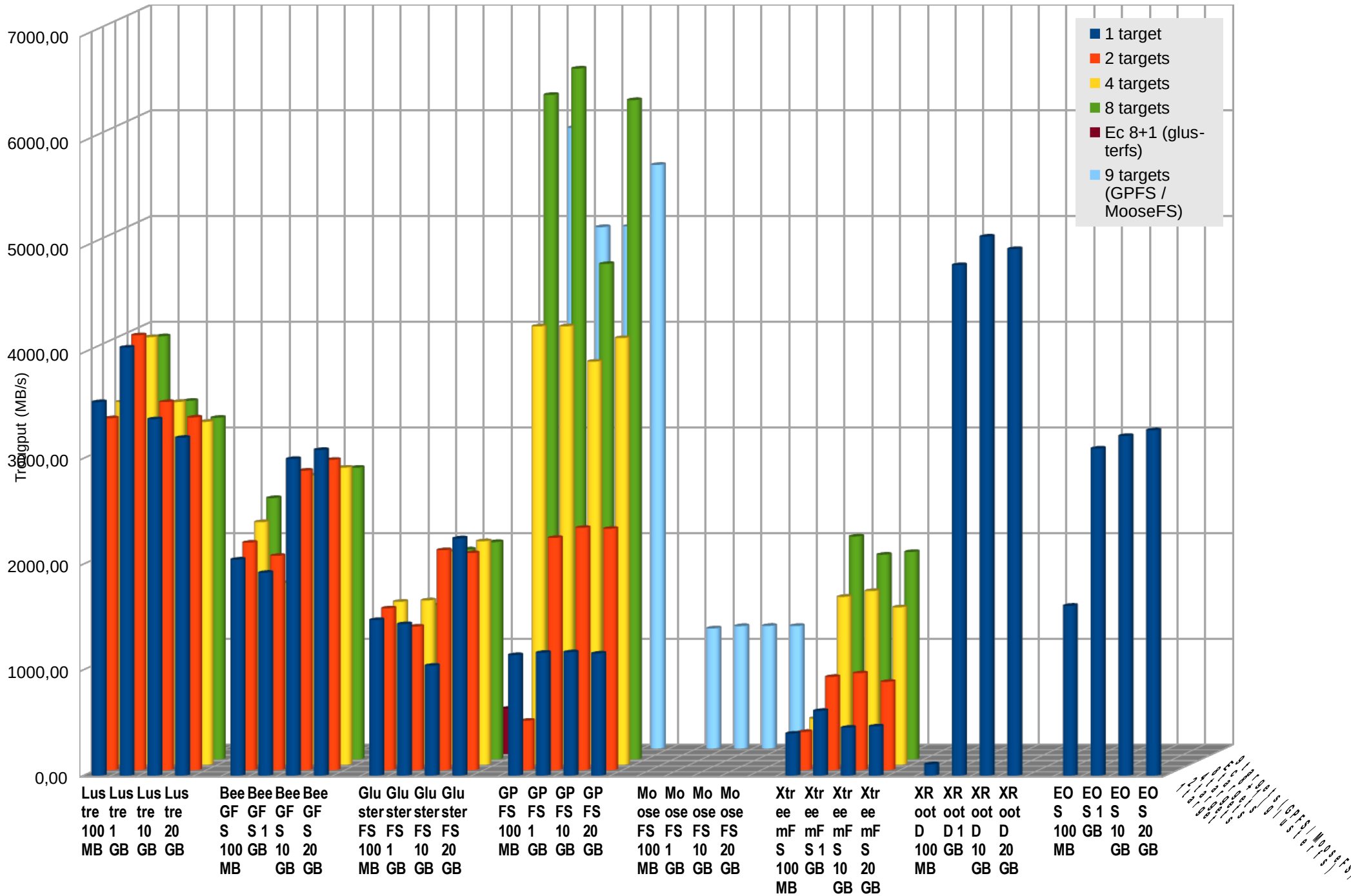
- Paramètres testés :
 - **Taille des fichiers à écrire** (pour déterminer quelle est la taille optimale, détermination du coût du traitement des meta-données) :
choix => **100Mo, 1Go, 10Go** et **20Go**
 - **Nombre de flux** / nombre de threads / nombre de processus en parallèle écrivant les données :
choix => 1, 6, 8
 - 1 = pour connaître le débit d'un flux individuel
 - 6 = nombre de flux reçus par un « Event Builder »
 - 8 = nombre de coeurs hyper-threadés de mon client
 - Nombre de **chunks** (nombre de blocs de données pour chaque fichier écrits sur plusieurs éléments de stockage, pour connaître l'effet de la répartition des données sur plusieurs serveurs de stockage) :
choix => 1, 2, 4, 8
 - **Nombre de targets** : Nombre de serveurs de données utilisés pour l'enregistrement des données

=> $4 \times 3 \times 4 = 48$ combinaisons à tester

=> 48 combinaisons * 8 FS = 384 tests

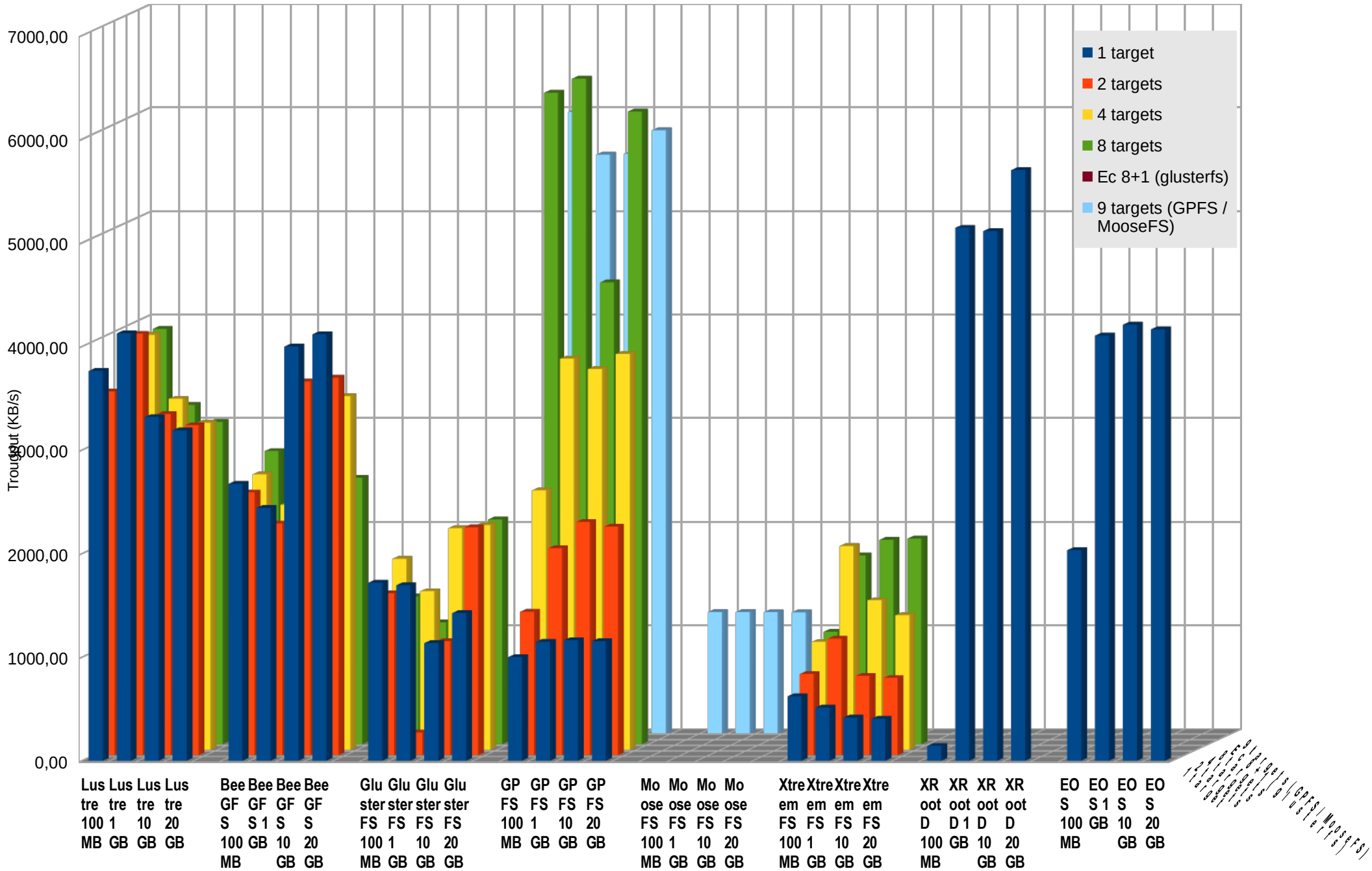
→ Par rapport a seulement 1 thread : Parallelisation des écriture sur des serveurs différents

Distributed file systems (1 client, 6 threads)

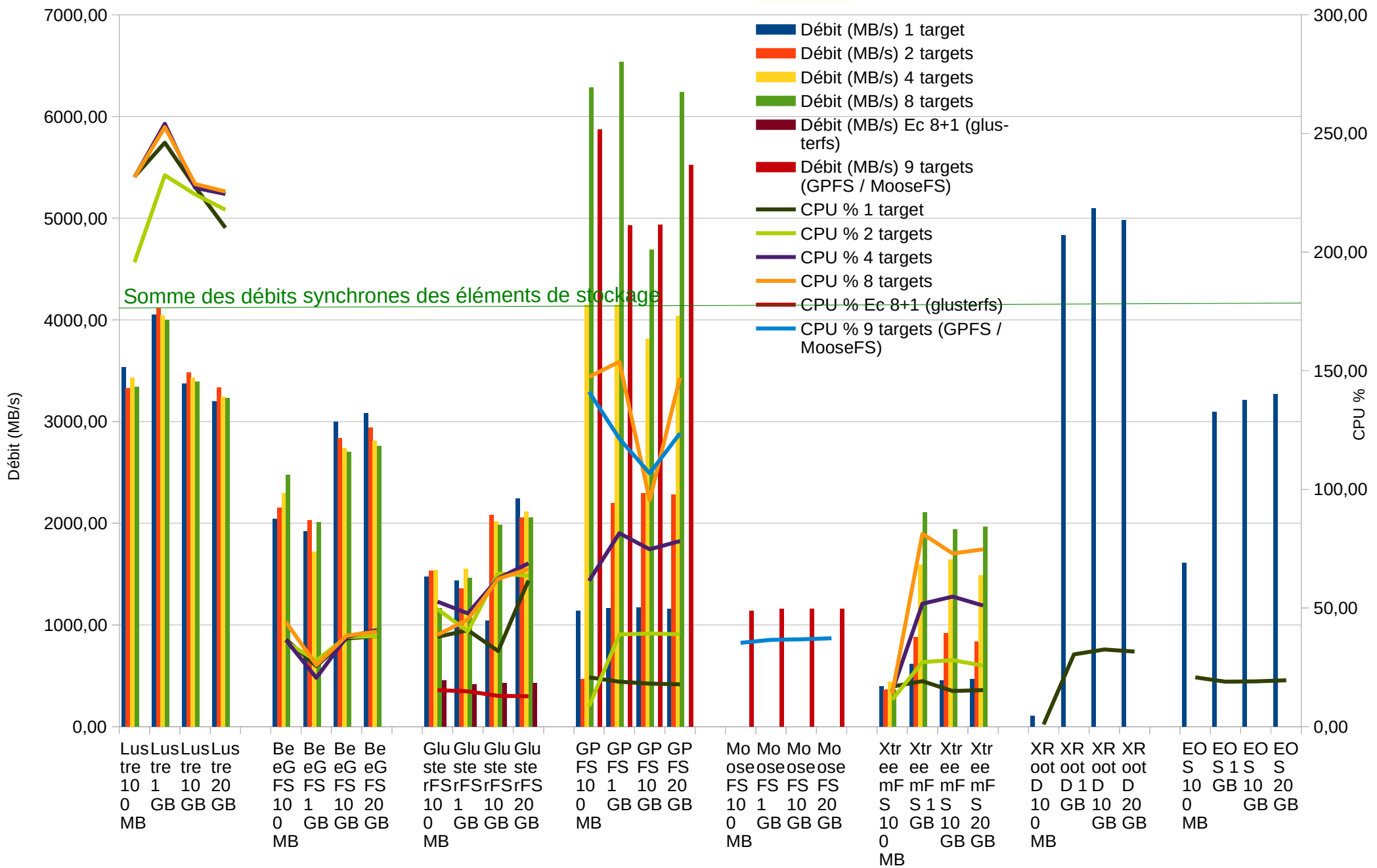


→ tous les coeurs du client sont utilisés

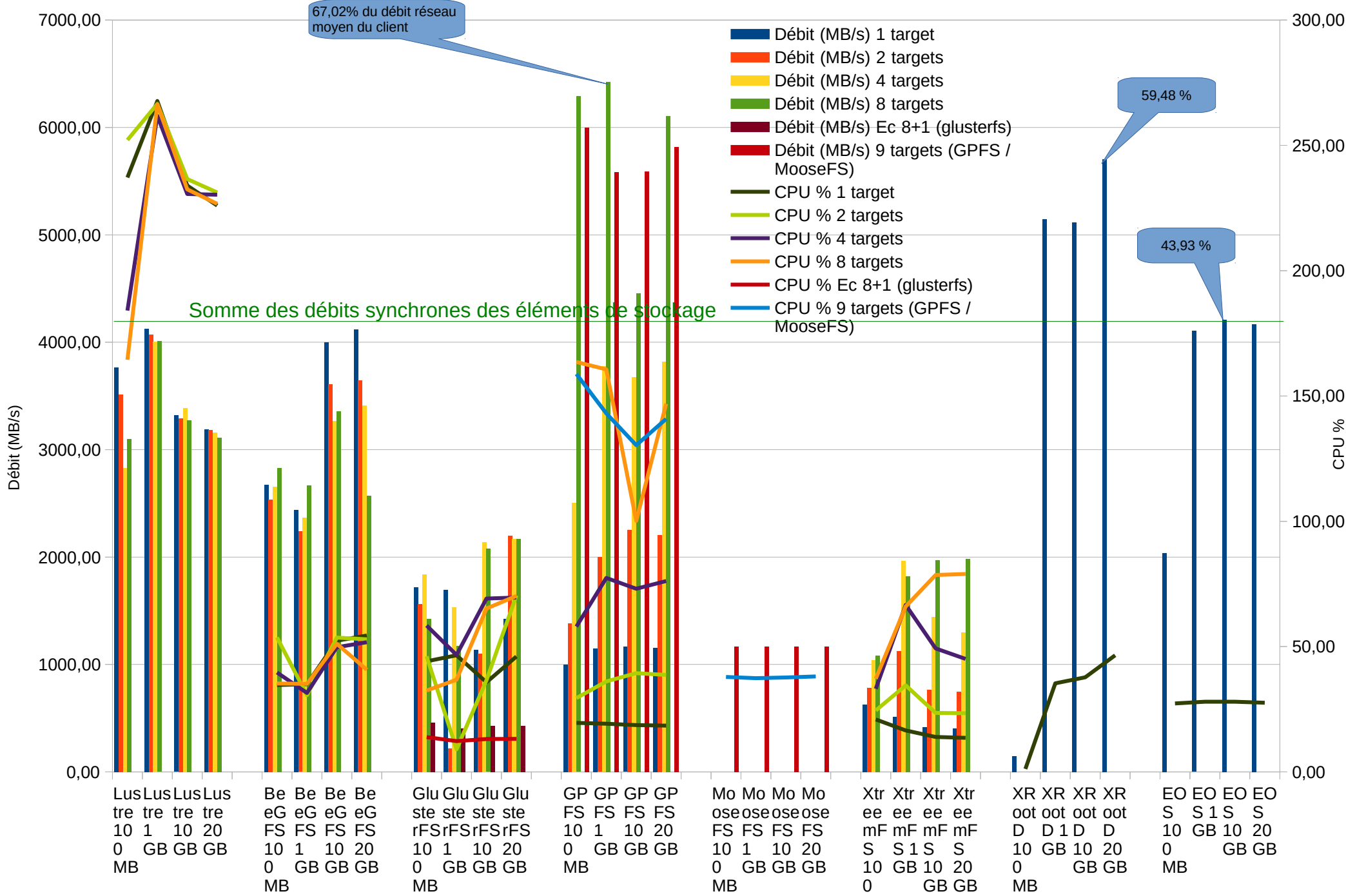
Distributed file systems (1 client, 8 threads)



Distributed filesystems (6 threads)



Distributed filesystems (8 threads)



Conclusion

- Classification :
 - Haute performance : GPFS, Lustre, BeeGFS
 - PRA : XtremFS, MooseFS
 - Pour tous usages : GlusterFS
- Conclusion sur les tests :
 - La parallélisation des I/O (chunks répartis sur les serveurs de stockage) apporte un gain seulement pour un nombre faible de clients
 - Grosses différences dans les performances :
 - Impact négatif de Fuse
 - Pile logicielle ± optimisée
 - La couche POSIX a un coût CPU important coté client
 - Acquisition :
 - Distribuer les flux d'acquisition sur différents processus indépendants (ideal ratio : 1 flux d'acquisition / CPU core)
 - Pour distribuer la charge sur le système de stockage, créer autant de flux réseau indépendants que possible (ideal ratio : 1 flux réseau par serveur de stockage)
 - Network tests :
 - Utiliser de préférence le même débit d'interfaces sur tous les systèmes: 10Gb/s → 10Gb/s, 40gb/s -> 40gb/s...
 - Préférer LACP (IEEE 802.3ad) : il est plus efficace que les autres algorithmes (quand toutes les interfaces du même système ont le même débit)
 - 4 meilleurs candidats montrés dans le test : **GPFS**, Lustre, **XRootD** et EOS.
 - GPFS est très efficace (il utilise toutes les capacités hardware disponibles), le problème est le coût de la licence logicielle (€€€)
 - Dans la version testée, Lustre nécessite beaucoup plus de CPU que les autres
 - XRootD est très efficace (autant que GPFS)
 - EOS a été moins efficace que XRootD natif, mais il a des fonctionnalités bien conçues pour un système de stockage en production
 - Taille des fichiers stockés dans le système de fichiers :
 - Ne pas créer des fichiers trop petits (impact négatif du traitement des métadonnées) : créer des fichiers > 1GB
 - Ne pas créer des fichiers trop gros : Tenir compte des contraintes de traitement des données sur les worker nodes dans les phases d'analyse online ou offline (< 20GB / file ?)

Remerciements

- **Telindus / SFR pour le prêt du switch**
- R. Barbier (IPNL/EBCMOS), B. Carlus (IPNL/WA105) et J. Marteau (IPNL/WA105) pour le prêt des cartes Mellanox 40gb/s
- L'équipe CMS de l'IPNL pour l'utilisation temporaire des 9 Dell R510 avant le début de la prise de données du RUN 2 du LHC
- L-M Dansac (Univ-Lyon 1/CRAL) pour l'utilisation temporaire d'un Dell R630
- Pour l'aide : C. Perra (Univ-lyon 1/FLCHP), Y. Calas (CC-IN2P3), L. Tortay (CC-IN2P3), B. Delaunay (CC-IN2P3), J-M. Barbet (SUBATECH), A-J. Peters (CERN)

Quelques commandes de configuration

FS	Client	Serveurs de données
Lustre	<pre>mount -t lustre 10.3.3.3:/lustre /lustre lfs setstripe --size \$stripe_size --count \$nb_targets /lustre/nt\$nb_targets The default stripe_size is 1 MB</pre>	<pre>mkfs.lustre --reformat --fsname=lustre --mgsnode=10.3.3.3@tcp --ost --index=\$i /dev/sdb</pre>
BeeGFS	<pre>beegfs-ctl --setpattern --numtargets=\$nb_targets --chunksize=1m /mnt/beegfs/test3</pre>	<pre>/etc/beegfs/*.conf : tuneNumWorkers = 8</pre>
GlusterFS	<pre>gluster volume set vol-\$i cluster.stripe-block-size 1MB</pre>	
GPFS	<pre>mmcrfs /dev/d1 "st17;st18;st19;st20;st21;st22;st23;st24;st25" -A yes -B 1M -D posix -j cluster -T /gpfs mmls /dev/d1 /gpfs</pre>	<pre>1 NSD / serveur, bs=1M, pas de quota, pas de réplication Data ni Metadata usage=dataAndMetadata pour chaque NSD</pre>
XtreemFS	<pre>mount.xtreemfs 10.3.3.5/test2 /xtreemfs --enable- async-writes xtfsutil --set-dsp -p RAID0 -w \$nb_targets -s 1024 /xtreemfs/nt4 xtfsutil /xtreemfs --set-drp --replication-policy none # OSD selection policy : roundrobin xtfsutil --set-osp roundrobin /xtreemfs xtfsutil --disable-snapshots /xtreemfs</pre>	
EOS		<pre>mkfs.ext4 -E stride=10,stripe-width=1024 /dev/sdb mount /dev/sdb -o defaults,user_xattr /gridcms1</pre>

Links / bibliography

- Storage systems :
 - GPFS : https://www.ibm.com/support/knowledgecenter/SSFKCN/gpfs_welcome.html
 - Lustre : <http://lustre.org/>
 - BeeGFS :
 - <http://www.beegfs.com/content>
 - http://www.beegfs.com/docs/Introduction_to_BeeGFS_by_ThinkParQ.pdf
 - GlusterFS : <https://www.gluster.org>
 - MooseFS : <https://moosefs.com>
 - XtremFS :
 - <http://www.xtreemfs.org>
 - <http://www.xtreemfs.org/xtfs-guide-1.5.1.pdf>
 - XrootD : <http://xrootd.org>
 - EOS : <http://eos.readthedocs.io/en/latest>
- Bonding : <https://www.kernel.org/doc/Documentation/networking/bonding.txt>
- System, network and Mellanox tuning :
 - http://www.mellanox.com/related-docs/prod_software/MLNX_EN_Linux_README.txt
 - http://supercomputing.caltech.edu/docs/Chep2012_40GEKit_azher.pdf
 - http://www.nas.nasa.gov/assets/pdf/papers/40_Gig_Whitepaper_11-2013.pdf
 - https://access.redhat.com/sites/default/files/attachments/20150325_network_performance_tuning.pdf
 - <https://fasterdata.es.net/host-tuning/40g-tuning/>
- The new CMS DAQ system for LHC operation after 2014 (DAQ2) :
 - <http://iopscience.iop.org/article/10.1088/1742-6596/513/1/012014/pdf>