cherenkov telescope array

Système DAQ pour les caméras de l'expérience CTA

Cherenkov Telescope Array

Dirk HOFFMANN, Julien HOULES — CPPM, Marseille

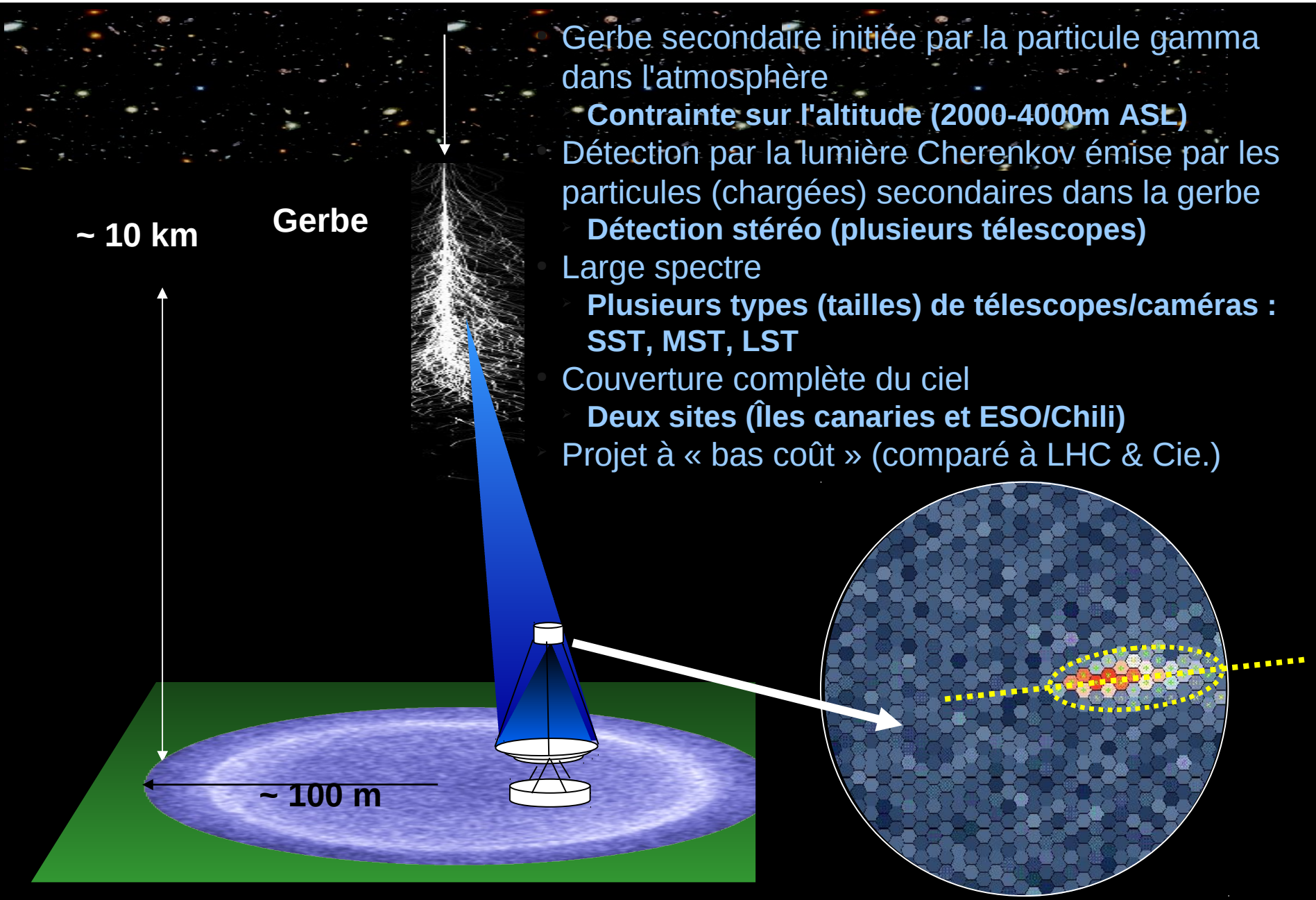
avec l'aide et le matériel de la collaboration NectarCAM

cherenkov telescope array

Sommaire

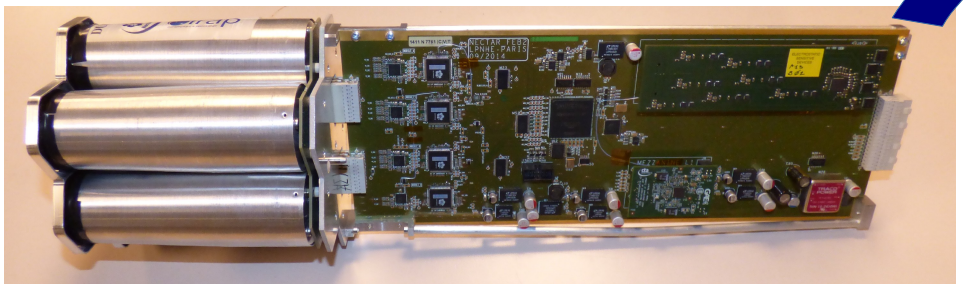
- CTA – concept et sites
- NectarCAM – une caméra de 1855 pixels à 60×1ns, 9kHz
- Implémentation et optimisation d'une DAQ à 40Gbps
- Validation sans le vrai matériel
- Résultats
- Perspectives

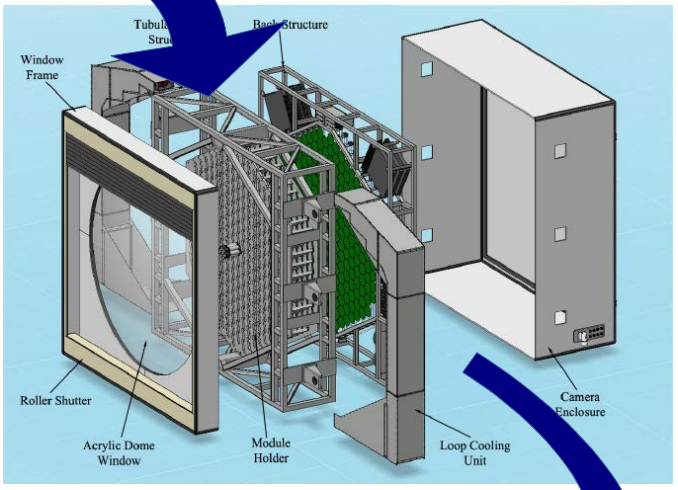
Détection de particules gamma de « très haute » énergie



- Gerbe secondaire initiée par la particule gamma dans l'atmosphère
  - **Contrainte sur l'altitude (2000-4000m ASL)**
- Détection par la lumière Cherenkov émise par les particules (chargées) secondaires dans la gerbe
  - **Détection stéréo (plusieurs télescopes)**
- Large spectre
  - **Plusieurs types (tailles) de télescopes/caméras : SST, MST, LST**
- Couverture complète du ciel
  - **Deux sites (Îles canaries et ESO/Chili)**
- Projet à « bas coût » (comparé à LHC & Cie.)

La caméra « française » : NectarCAM





1855 pixels dans 265 modules

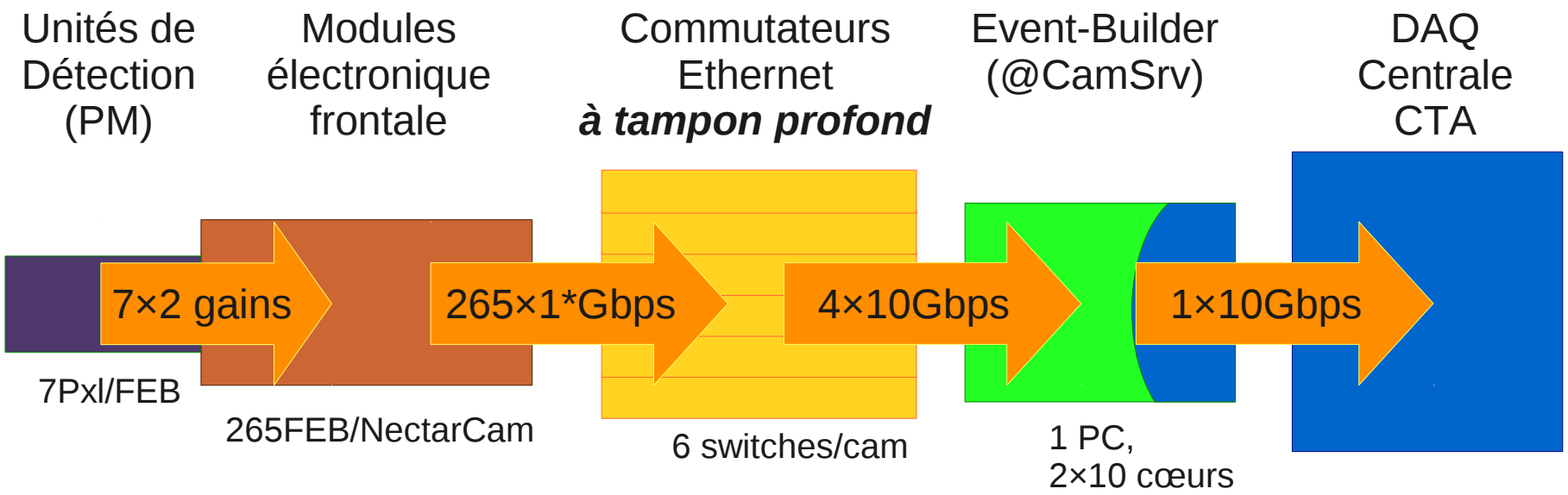
265 connexions Ethernet (UDP) en 1000baseT

Sur les MST, taux de trigger 9kHz (aléatoire)

Echantillonage 60×1ns, résolution < 1ns

L'acquisition des données NectarCAM

Niveaux d'acquisition



1855 Pxls 500kB/evt 9kHz 38Gbps max.
60 échantillons (ns)
2 gains (12 bits)

95% de la vitesse nominale sur 4 liaisons SFP+ (10Gbps)

Codes couleurs pour les responsabilités

*) limité à 300Mbps en réalité

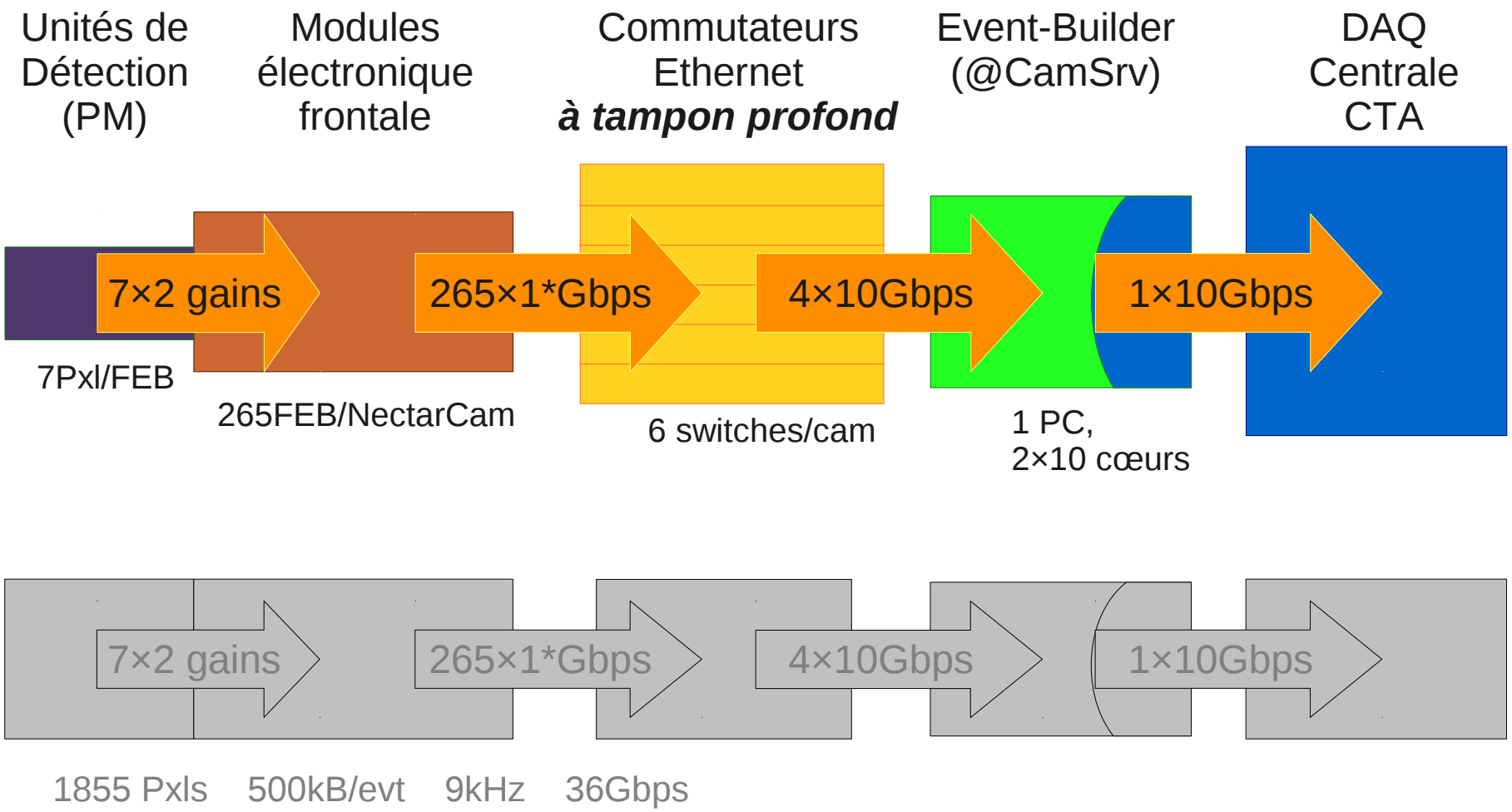
cherenkov telescope array

De la conception à la validation
Techniques mises en œuvre

- En attendant l'électronique frontale ...
  Conception et construction d'un stimulateur
- Architecture réseau ...
  40Gbps sans perte en UDP
- Optimisation de la bande passante ...
  De petits paquets à vitesse maximale

L'acquisition des données NectarCAM

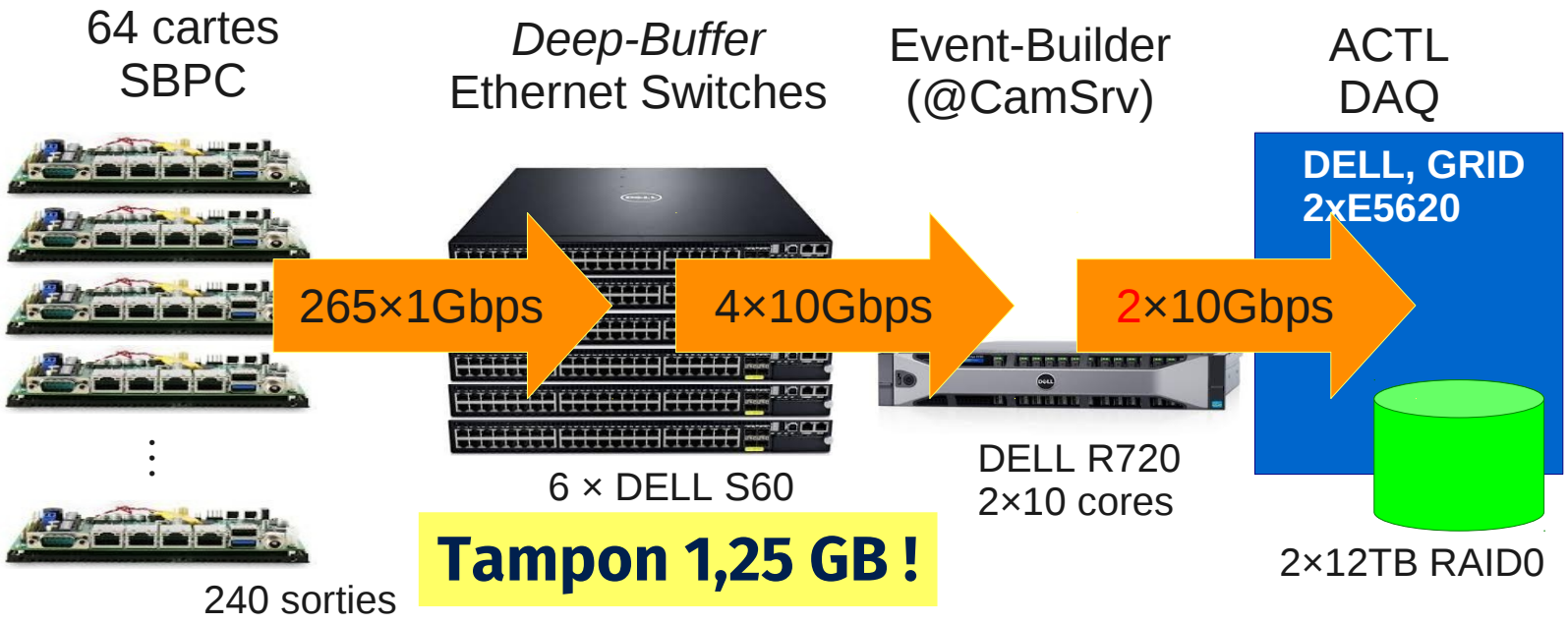
Niveaux d'acquisition

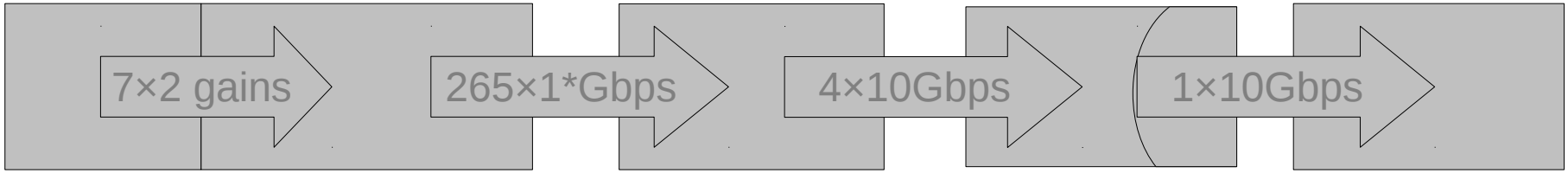


*) limité à 300Mbps en réalité

L'acquisition des données NectarCAM

Le système réel 1 : Stimulateur de DAQ (2014/15)





1855 Pxls 500kB/evt 5…6kHz → 19…36Gbps

Système réel 1 : Stimulateur de DAQ

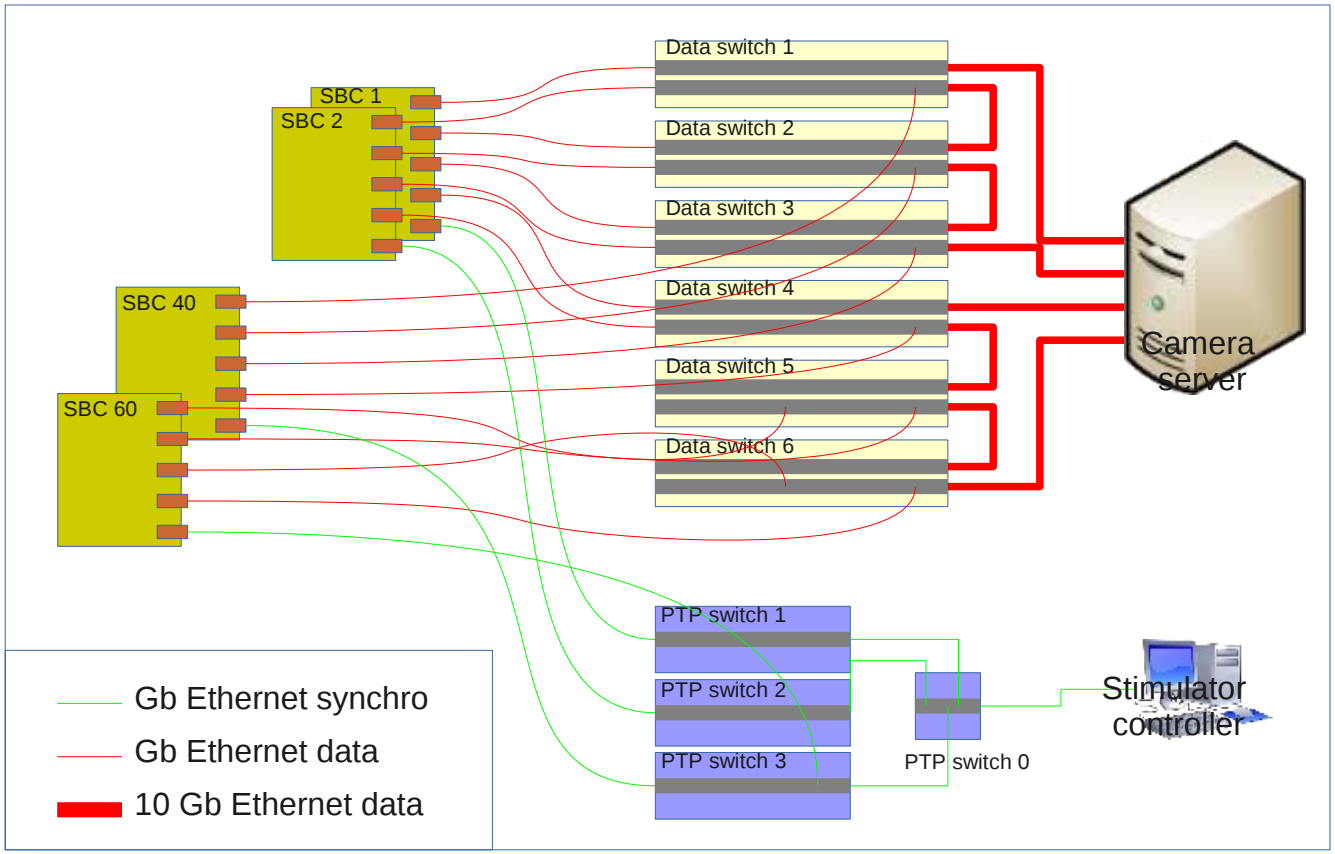
64 cartes SBPC en BOOTP

Serveur pour Event-Builder

Points à valider

- **Synchronisation absolue (1ns)** des données de la caméra
- Petits paquets (1kB) ⇒ **grand nombre** (40 000 000/s) Dépassent les capacités d'un pilote Linux standard.
- **Perte de paquets** (UDP) pendant le routage dans les commutateurs ?

Synchronisation du stimulateur



Sychronisation des horloges

- **Protocole LinuxPTP modifié**
- Envoi d'un signal PTP-SYNC sur 64 chemins identiques
- Synchronisation de l'horloge temps réel de chaque SBPC par rapport à l'horloge de l'interface Gbps

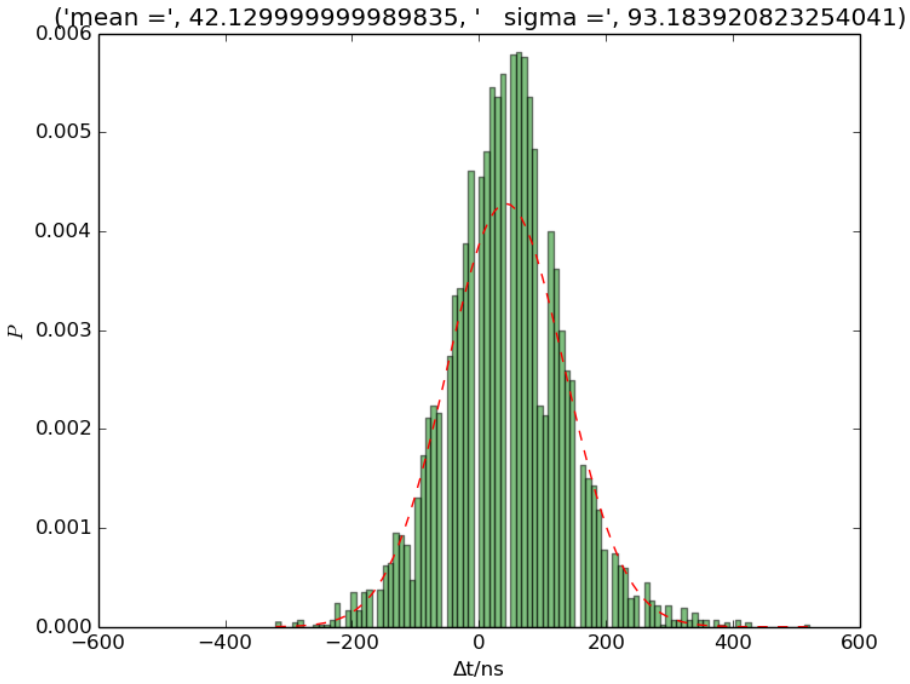
Sychronisation des envois des données

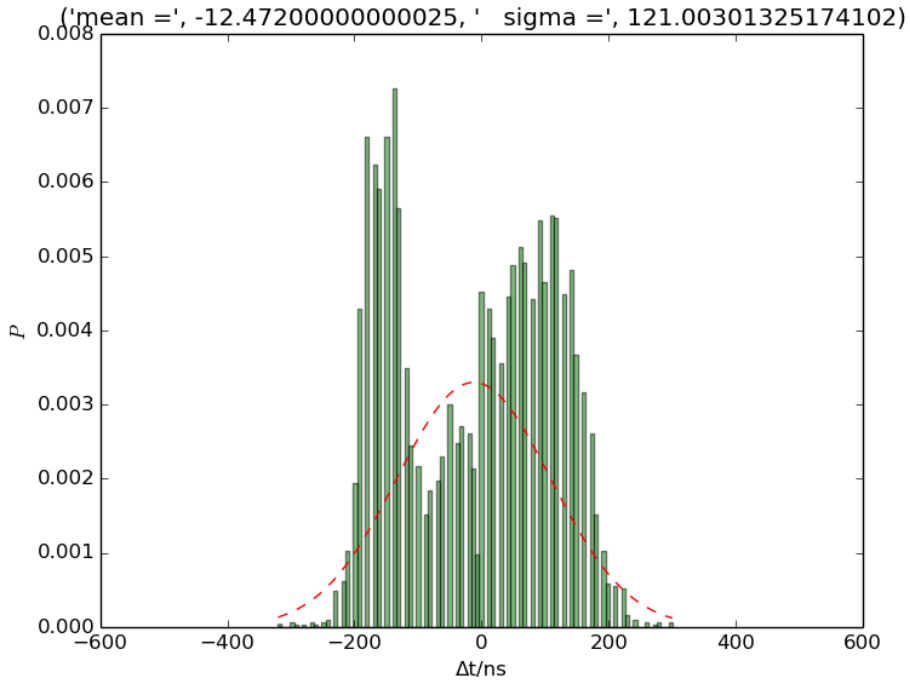
- **Modification de la pile Ethernet Linux**
- Pré-remplissage des zones DMA avec des données
- Boucle de la commande d'envoi aux interfaces sur *n* ports **dans** l'espace noyau

Synchronisation du stimulateur

Le résultat convient.

- Synchronisation <100ns entre ports de même cardinalité
- Synchronisation <200ns entre tous les ports





Système réel 1 : Stimulateur de DAQ

64 cartes SBPC en BOOTP

Serveur pour Event-Builder

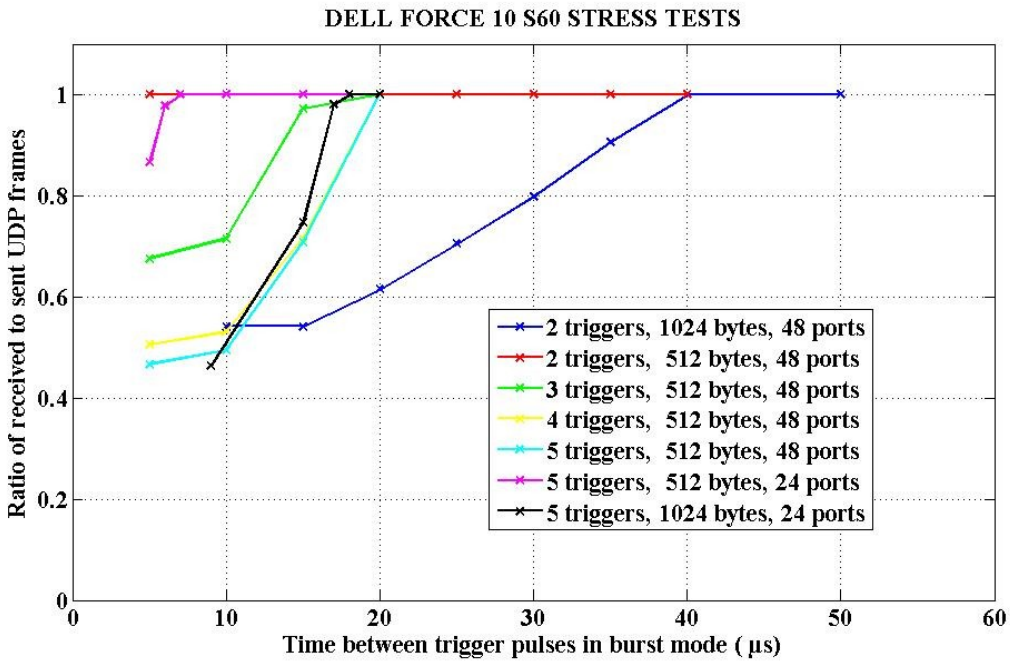
Points à valider

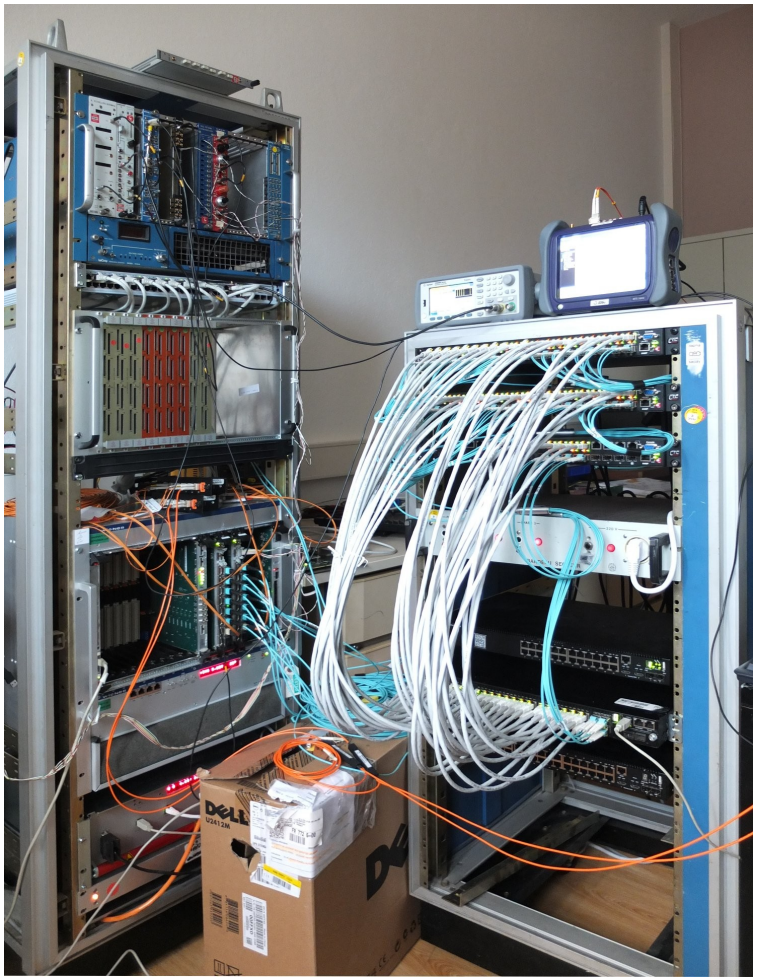
- ✓ **Synchronisation absolue (ns)** des données de la caméra
- **Petits paquets (1kB) ⇒ grand nombre (40 000 000/s) Dépassent les capacités d'un pilote Linux standard.**
- **Perte de paquets (UDP) pendant le routage dans les commutateurs ?**

Synchronisation : Approche complémentaire

Génération de paquets UDP par FPGA

Synchronicité O(1ns)
Envoi sur un commutateur Ethernet (S60)

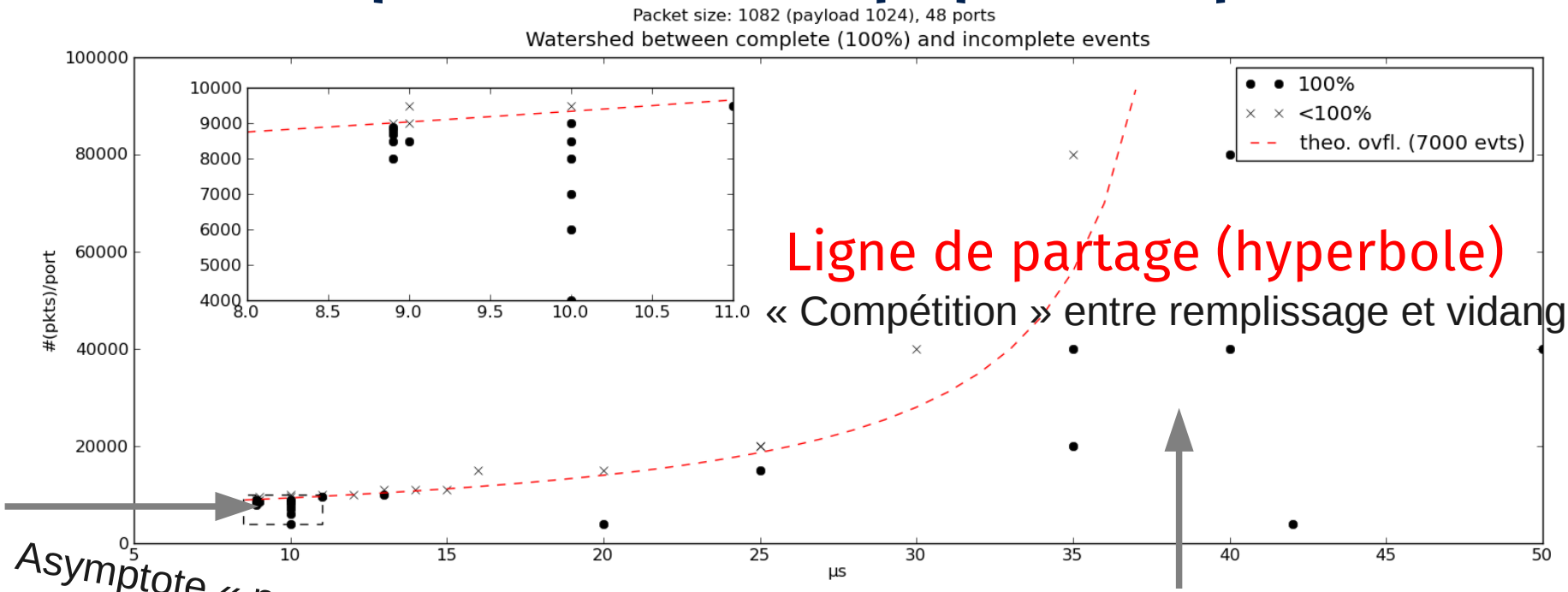




Cette solution étant limitée par son prix (facteur 10), elle a été appliquée à un seul commutateur.

Modèle du mécanisme de tempon dans les commutateurs

Plan « fréquence / nombre de paquets (complets) »



Ligne de partage (hyperbole)

« Compétition » entre remplissage et vidange

Asymptote « nombre de buffers »

Asymptote 37µs = 1/**27kHz**
correspondant au temps de transit 40×1kB/10Gbps

Très bonne correspondance au modèle, permet de dériver le paramètre caractéristique : 8900 paquets

(34 % des 1,25GB disponibles)

Système réel 1 : Stimulateur de DAQ

64 cartes SBPC en BOOTP

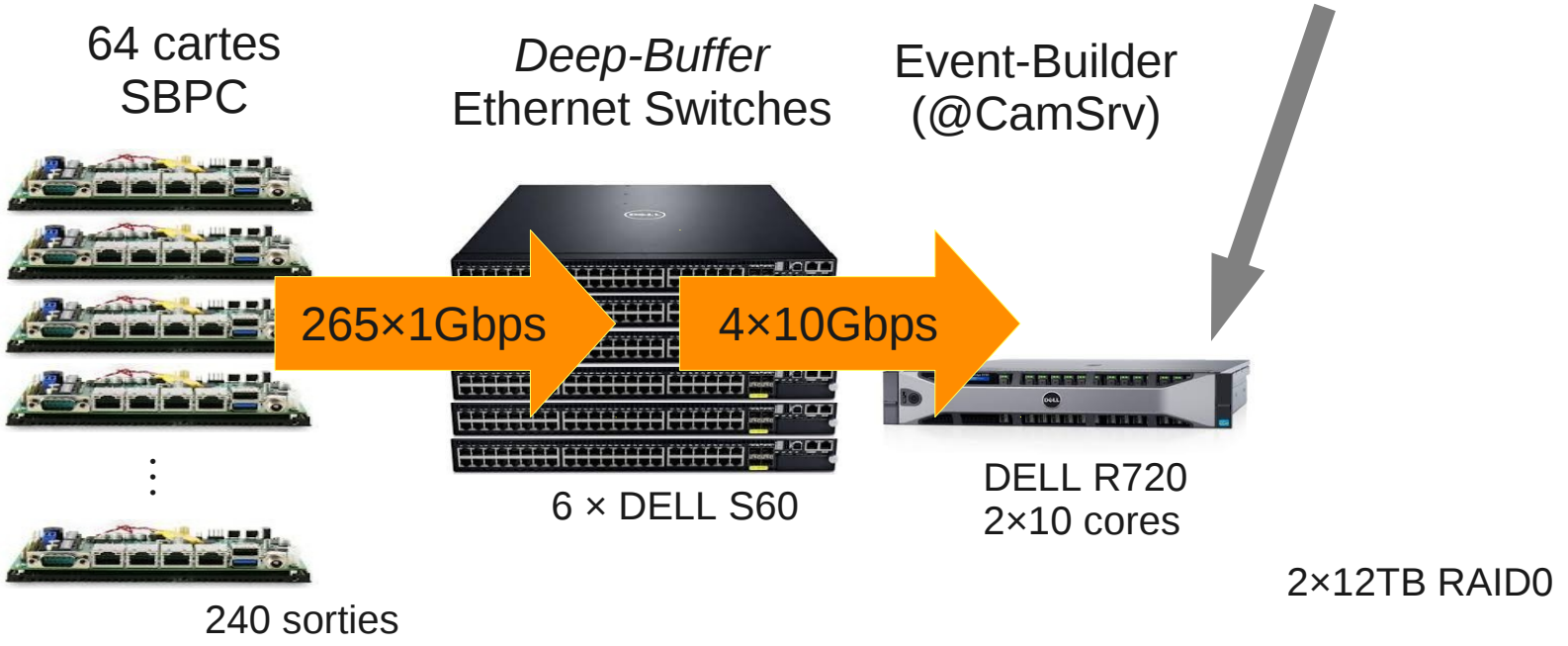
Serveur pour Event-Builder

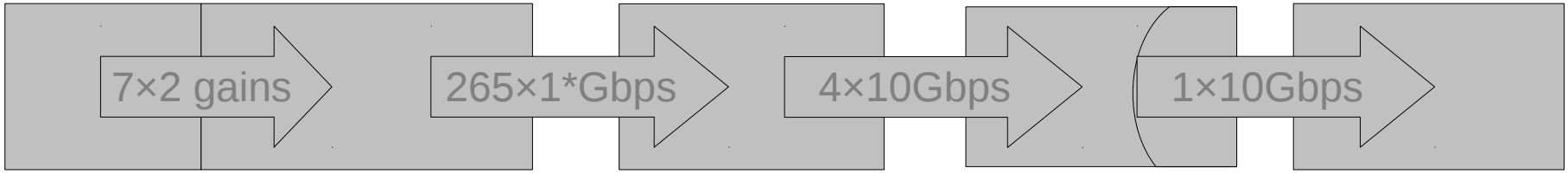
Points à valider

- **Synchronisation absolue (ns)** des données de la caméra
- Petits paquets (1kB) ⇒ **grand nombre** (40 000 000/s) Dépassent les capacités d'un pilote Linux standard.
- **Perte de paquets** (UDP) pendant le routage dans les commutateurs ?

Réception de petits paquets

Ici : entrée de paquets 1kB sur Event-Builder



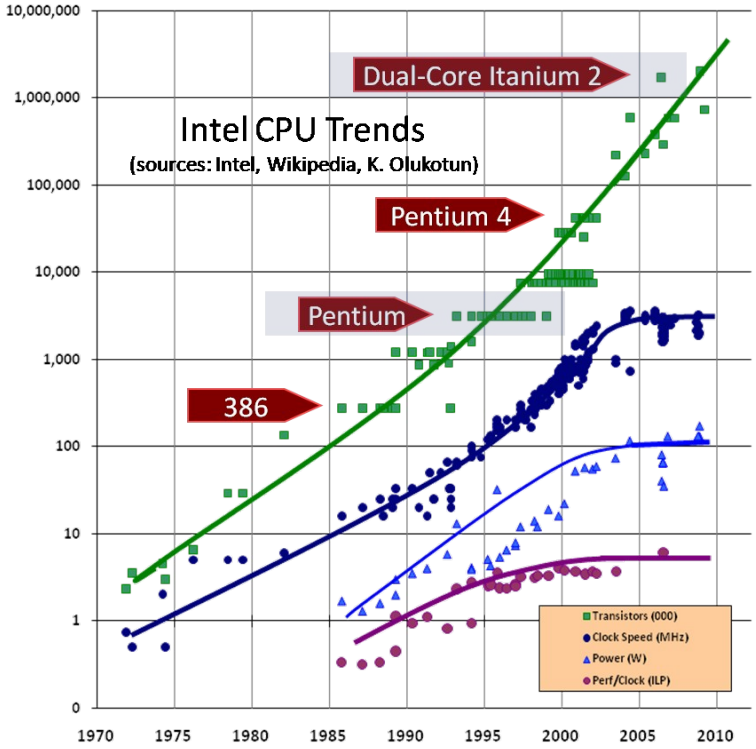


1855 Pxls 500kB/evt 5…6kHz → 19…36Gbps

Réception de petits paquets

Problème bien connu des architectes réseau : "Free lunch is over"

- Conception des piles Ethernet dans Linux/Unix bien adaptée entre les vitesses réseau et les vitesses CPU … **en 1970 !**
- Aujourd'hui, 50 % de bande passante pour MTU 1500 sur 10Gbps.
- Compensation par MTU 9600 (Jumbo Frames)



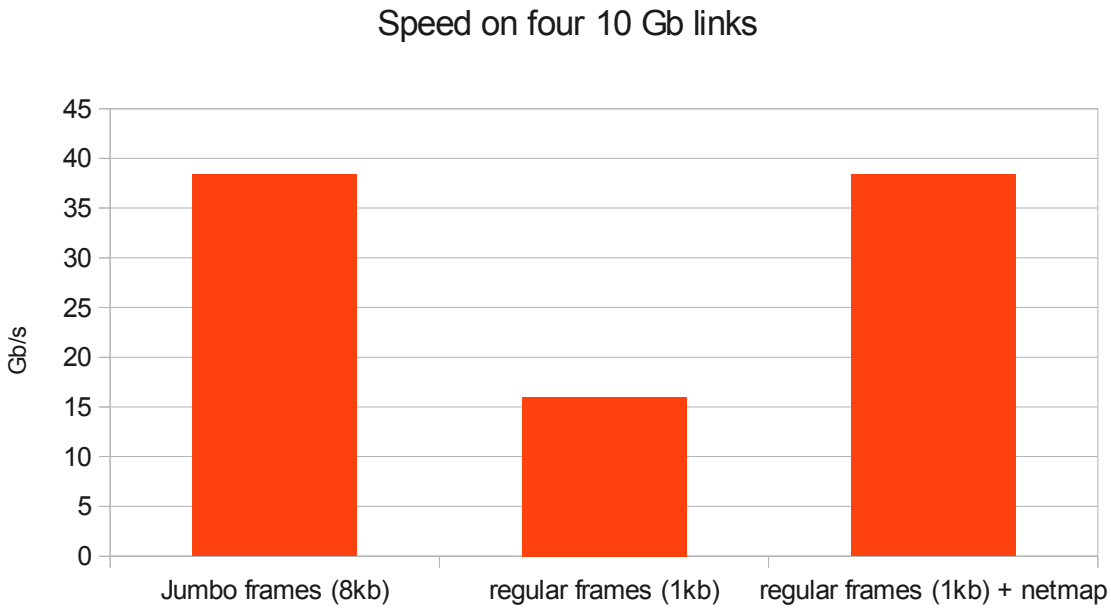
Jumbo Frames pas disponibles dans modules Nectar

- Paquet **netmap** développé à l'Université de Pise
  - Pré-allocation des zones tampon à l'initialisation
  - Accès direct (une copie en moins) à l'espace kernel via mmap()

Réception de petits paquets

Résultat sans équivoque.

Réception, puis envoi des données à pleine vitesse sur API (0MQ).



Avec quelques ajustements supplémentaires pour le traitement :

- Association des interfaces réseau en fonction de la NUMA
- Affinité des tâches à un cœur spécifique (isolcpu et taskset)
- Event pre-fetching
- Busy waiting synchronisation

Système réel 1 : Stimulateur de DAQ

64 cartes SBPC en BOOTP

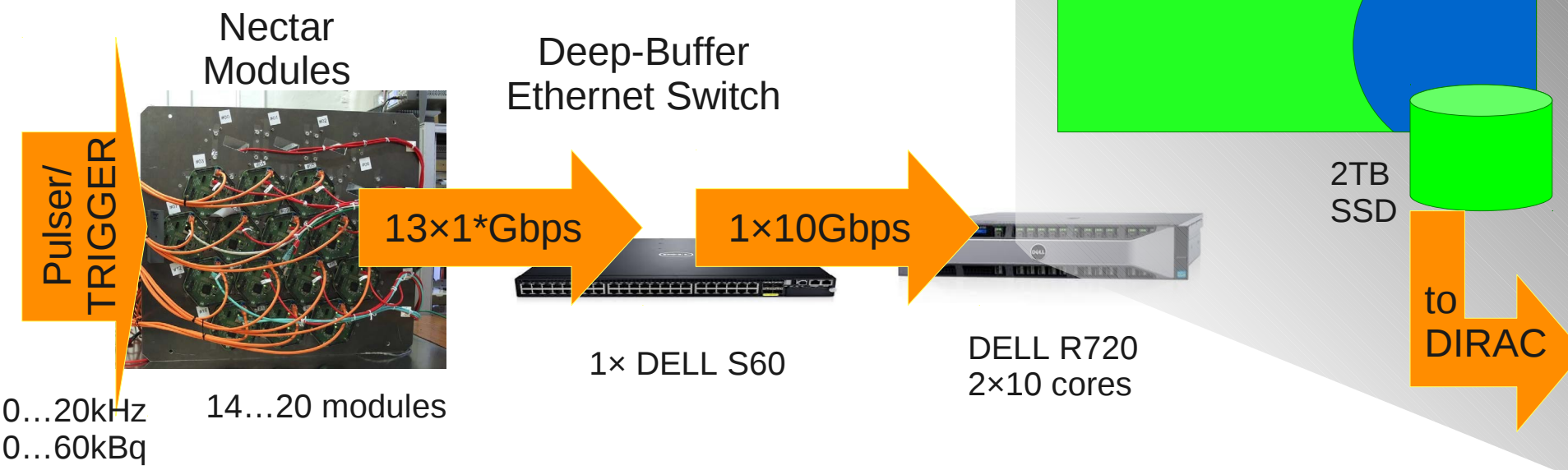
Serveur pour Event-Builder

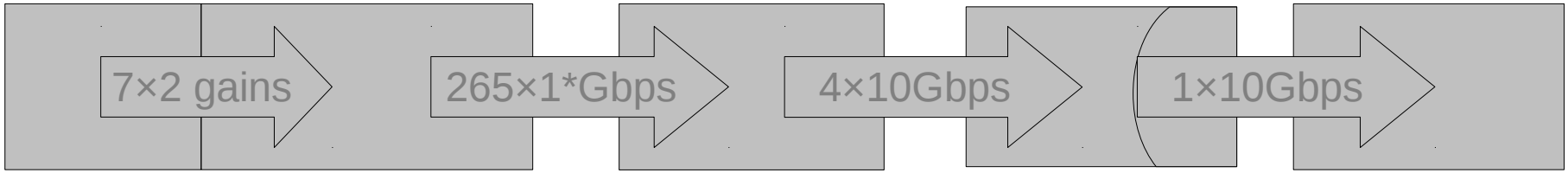
Points à valider

- **Synchronisation absolue (ns)** des données de la caméra
- Petits paquets (1kB) ⇒ **grand nombre** (40 000 000/s) Dépassent les capacités d'un pilote Linux standard.
- **Perte de paquets** (UDP) pendant le routage dans les commutateurs ?

N'oublions pas l'IRFU !

Système réel 2 : NectarCAM Demonstrator (2015/16)





*) actually throttled to 300Mbps

Système réel 2 : NectarCAM Demonstrator (2015/16)

Livraison en décembre 2015

- Démarrage quasiment au quart de tour (souvent à distance)
- Charactéristiques confirmées avec une vingtaine de modules
- Long run OK @10kHz, testing full chain
  - FEB, EVB, ZfitsWriter (l*ast debugs in ZfitsWriter.)*
  - First time in CTA!
  - With official tools/software
  - DATA format proposal (“L0”) implemented in ProtoBuf

Extensions à venir

- Implémentation de la lecture des systèmes trigger + estampillage
- Stockage et gestion des données avec DIRAC.

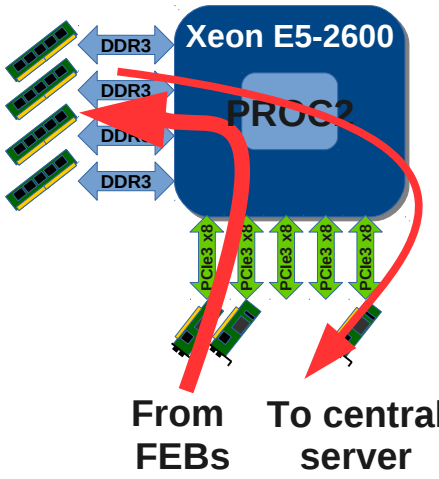
cherenkov telescope array

Perspectives
Problèmes connus et à venir

- Bilan de l'occupation CPU
  et ajout de traitements en ligne
- Remplacement d'un modèle obsolète de commutateurs

Toujours du potentiel : Ajout du traitement en ligne ?

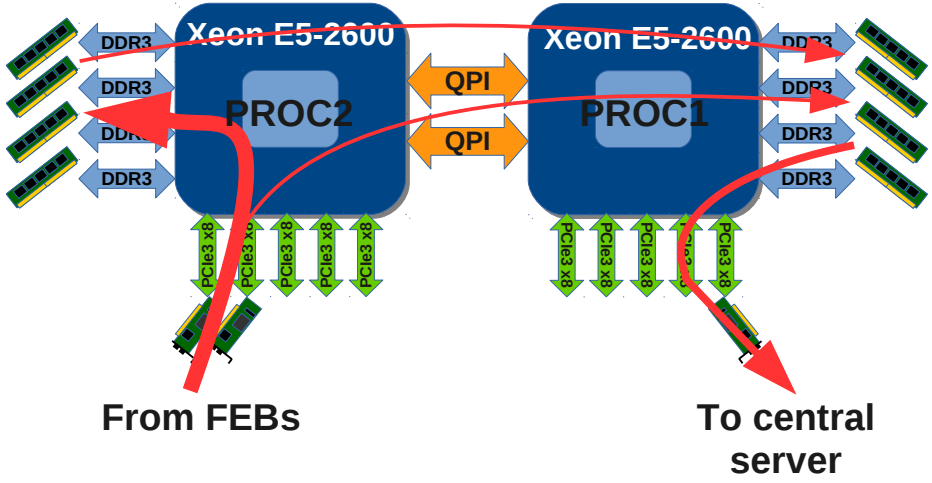
Besoins actuels



Un seul processeur :

- 4 cœurs réception
- 2 cœurs traitement
- 2 cœurs envoi

Pour besoins futurs



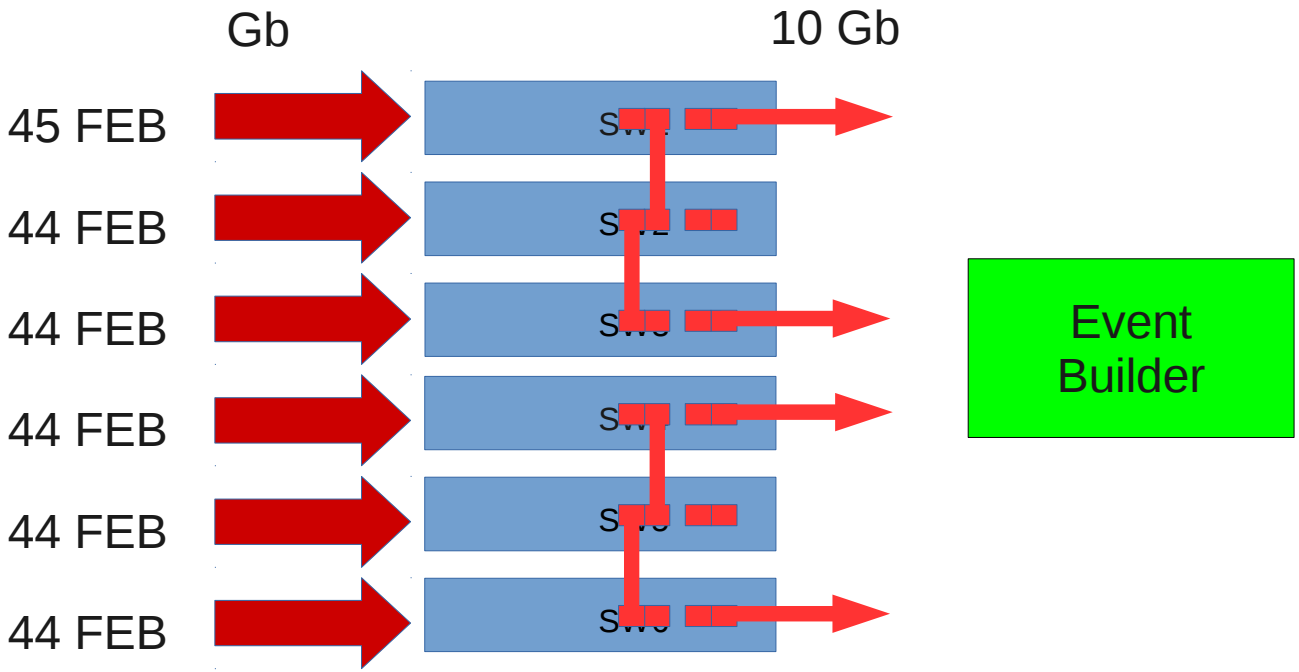
Utilisation du 2e processœur, à condition de minimiser la communication inter-CPU

- Réception et traitement initial sur CPU2
- OS, traitement final, envoi et contrôle sur CPU1
- 6 cœurs disponibles pour traitements initiaux et finals respectivement

Obsolescence des commutateurs

Problème (un peu inattendu) de la pérennité des solutions

- Obsolescence des S60 depuis cette année



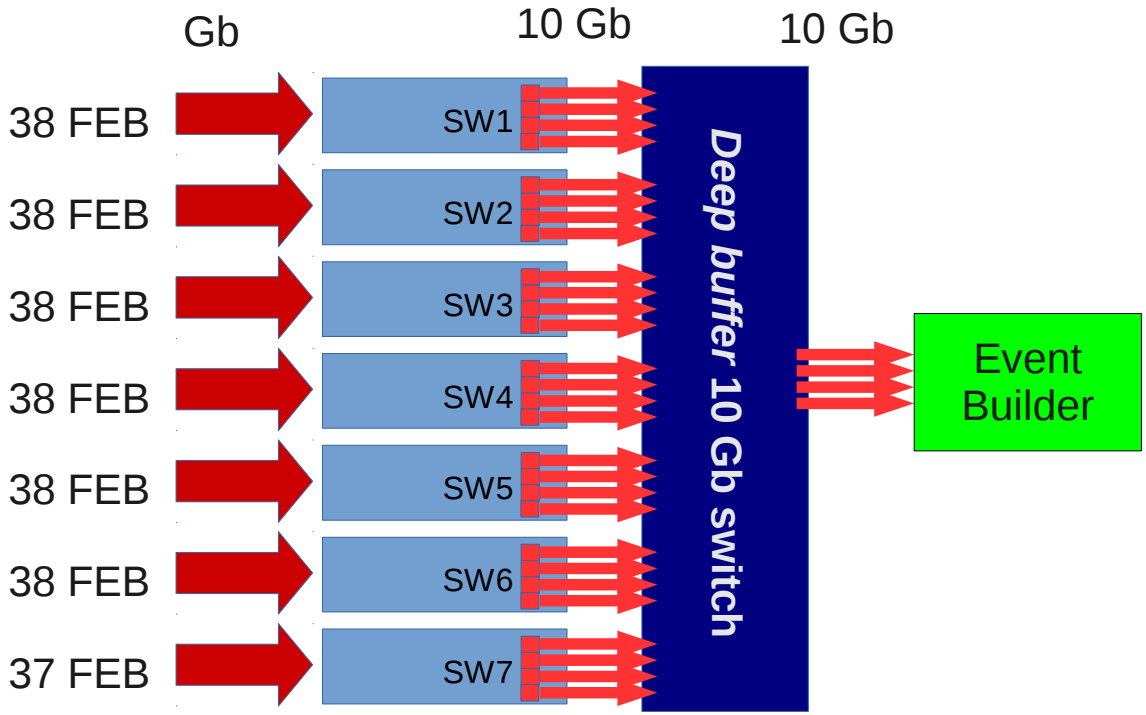
- Obsolescence des commutateurs *deep-buffer* dans la gamme « 1Gbps » tout court !

Obsolescence des commutateurs

Novelle architecture, supposée plus pérenne

- Deux niveaux de commutateurs
  - Concentration 38×1Gbps : 4×10Gbps
  - Temporisation 28×10Gbps : 4…6×10Gbps

- Bridage éventuel au niveau des modules FEB
- Utilisation de la *class of service* pour optimiser l'utilisation du tampon
- 10Gbps plus durable (jusqu'a …)



Résumé

- **Chaîne de lecture complète d'une caméra *de l'électronique frontale jusqu'au stockage* implémentée et testée pour la première fois dans CTA**
- **En mettant en œuvre des technologies à la fois modernes et suffisamment standard :**
  - **netmap** pour la réception rapide
  - PTP modifié pour la synchronisation
- **Un succès confirmé par la demande d'une version LST** (utilisant TCP/IP)