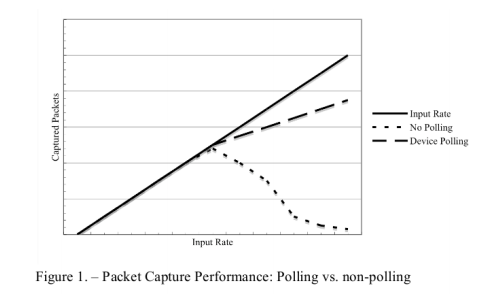
**Techniques du noyau :**

* IRQ vs Polling
  + Chaque paquet qui arrive entraine une IRQ .Si beaucoup de paquets entrants, peut provoquer une tempête d’IRQ : le CPU passe son temps à traiter les IRQ.
    - IRQ balance : repartie les IRQ sur tous les CPU
  + Polling : On attend un certain laps de temps avant de sonder le NIC pour des paquets entrants.
    - NAPI : utilisation du polling à partir d’un certain seuil d’IRQ



* Memory Map APproach (mmap) vs socket approche :
  + Socket : L’appli lit et copie l’info à partir de la socket
  + Mmap : Utilisation d’un ring buffer à la place de la socket, l’appli accède à la data via un pointeur du ring buffer

L’espace utilisateur accède donc directement à une mémoire du noyau.

* Ne pas oublier d’accorder le Kernel avec les techniques du NIC : RSS et offloading.

**Sockets :**

* Raw socket : autorise l’envoie et la réception directe de paquets sans formatage des paquets dans un protocole donné : permet d’envoyer n’importe quel type de trames

Permet donc d’envoyer des messages bas niveau : ICMP, IGMP, et d’implémenter des protocoles dans l’espace utilisateur.

**Méthode d’acquisition des paquets par le noyau :**

* PCAP basé sur libpcap
* PFRING : sélection du type de cluster
  + Ring buffer
  + Mmap
  + PF\_RING DNA vs non DNA
  + PF\_RING zero copy
* AF\_PACKET : sélection du type de cluster

AF\_PACKET + mmap a peu prêt pareil que PF\_RING

* NETMAP : pas de sélection du type de cluster
  + Approche similaire à PF\_RING
* UNIX\_SOCKET

**Linux :**

NIC envoie un signal à la réception d’un paquet.

* 1 ksoftIRQd s’occupe de tourner en boucle sur chaque CPU : permet de soulager l’interrupt handler qui va bloquer les autres IRQ.KsoftIRQ s execute à un haut niveau de PRIO mais pas autant que le handler d’IRQ, et il tourne avec IRQ activés.
* 1 structure softnet\_data crée sur chaque CPU (qui contient entre autre un poll\_list).

Linux utilise NAPI : permet d’éviter les tempêtes IRQ ( linux utilise les IRQ et au-delà d’un certain seuil utilise NAPI : sondage du device Ethernet toutes les X secondes : permet en gros de traiter plusieurs paquets d’un coup plutôt que un par un : technique du polling : mais attention si le buffer est trop plein : pertes de paquets !)