Kernel a síťové karty

Michal Schmidt
mschmidt@redhat.com

Počátky

- Ethernet vznikal v 70. letech v Xerox PARC.
- DEC/Intel/Xerox první společná specifikace v r. 1980

- Linux v1.0 (1994)
  - "NET2Debugged" síťový stack
  - okolo 20 síťových ovladačů pro tehdy obvyklé 10 Mb/s karty

První síťové ovladače v Linuxu

```
/*
    Written 1993 by Donald Becker.
    Copyright 1993 United States Government
    as represented by the Director,
    National Security Agency.
    [...]
*/
```

Dnes

- 1 Gb/s má v PC kdekdo.
- 10 Gb/s běžně v serverech.
- 40 Gb/s, 100 Gb/s již existují.

- Dnešní Ethernet vypadá jinak než ten původní (fyzické médium, topologie),
- ale mnohé mu zůstalo (formát rámce linkové vrstvy).

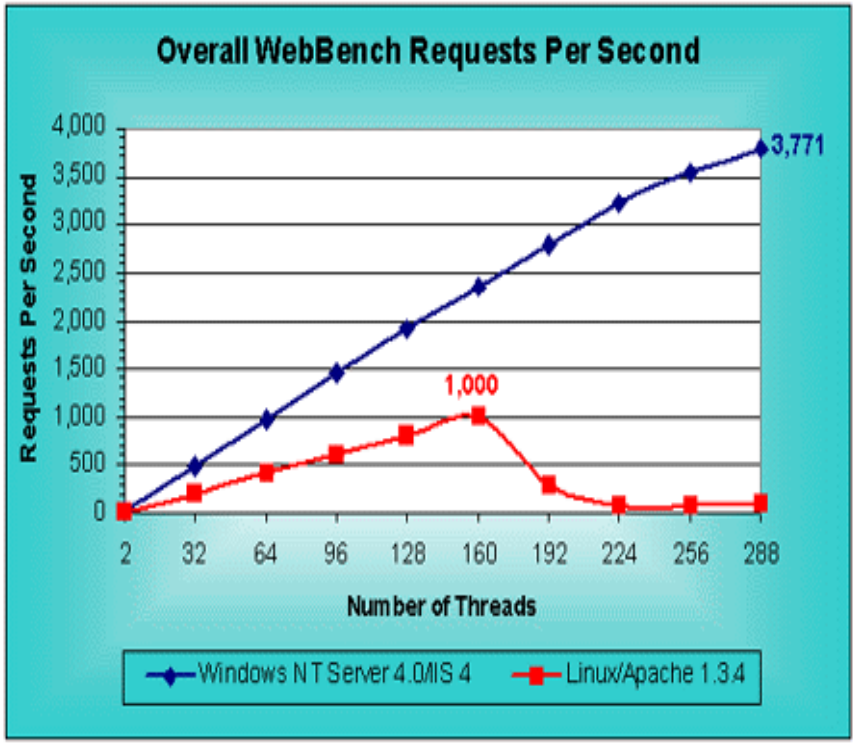
Náznak problému

- Maximum Transmission Unit (MTU) stále 1500 B.
- Jumbo rámce - MTU až do 9 KB - ale ne v Internetu
- 10 Mb/s ~ 1.2 MB/s, 1500 B/pkt => 1.2M / 1500 = 800 pkt/s
- 10 Gb/s ~ 1.2 GB/s, 1500 B/pkt => **800 000 pkt/s**
- Ostatní hardware také zrychlil, ale ne o dost.
- Nutno minimalizovat čas CPU potřebný ke zpracování každého paketu. Případně rozložit zátěž na více CPU.

Události roku 1999

- Bill Clinton ustál impeachment.
- Majetek Billa Gatese vzrostl na 100 mld. USD.
- Microsoft si objednal studii u spol. Mindcraft.

"Microsoft Windows NT Server is 2.5 times faster than Linux as a File Server and 3.7 times faster as a Web Server"



Příčina problému

- V testu server se 4 CPU a 4 síťovými kartami (100Mb/s).
- Linux v2.2 umí více CPU, ale síťový stack stále neumožňuje paralelní zpracování.

CPU1	CPU2	CPU3	CPU4
zpracovává	čeká	čeká	čeká
čeká	čeká	**zpracovává**	čeká
čeká	čeká	čeká	**zpracovává**
čeká	**zpracovává**	čeká	čeká
čeká	čeká	**zpracovává**	čeká

čas

Linux v2.4

- "softnet"
- přijímací fronta paketů pro každé CPU zvlášť
- Zpracování může běžet současně na více CPU. (bottom-halves nahrazeny softirq)

Zahlcení pod zátěží

- Při směrování nad 60 Kpkt/s pakety přicházely, ale žádné neodcházely ven.
- Bez ztrát se dařilo přenášet jen 27 Kpkt/s.
- Jediné vysoce zatížené rozhraní dokázalo vyhladovět všechna ostatní.
- "Interrupt livelock" – systém pouze zpracovává přerušení, na nic jiného nemá čas.

Řešení: NAPI

1. Přijde přerušení od síťové karty.
2. Ovladač zakáže přerušení od karty.
3. Ovladač oznámí kernelu, že karta má připravena data k příjmu.
4. Kernel se dotazuje všech takto oznámených karet (střídavě, po dávkách). Dostává data.
5. Pokud některá karta už další data nemá, je vyjmuta ze seznamu a opětovně povolí přerušení.

Některé výhody NAPI

- Automaticky podle zátěže přechází mezi režimy přerušování a dotazování:
  - Malá zátěž: každý paket vyvolá přerušení a je zpracován s minimálním zpožděním.
  - Velká zátěž: příjem v režimu dotazování, po velkých dávkách, dobré pro CPU cache. Odpadá režie přerušení. Férová obsluha více karet. Při přehlcení jsou pakety zahazovány už kartou na vstupu, bez obtěžování systému.

Ať se karta taky snaží

- "offload"- síťová karta dělá práci za CPU.
- TCP Offload Engine (TOE)
  - Inteligentní karta, sama rozumí TCP. Existovaly už v době 10 Mb/s karet.
  - Nevýhody: Neprůhledné, obchází vrstvy OS, riziko bezpečnostních chyb firmwaru, časem byly pomalejší než OS na rychlejším CPU.
  - Proto v Linuxu nepodporované.
- Bezstavový offload: karta pomáhá nekontroverzně

Rx/Tx checksum

- Kontrolní součty v hlavičkách IP, TCP, UDP
- Rx (receive, příjem) – karta spočítá checksum a buď:
  - jen ohlásí systému, zda byl výsledek v pořádku,
  - nebo ohlásí hodnotu spočtenou pro celý rámec a OS si už dopočítá, zda to bylo správně.
- Tx (transmit, vysílání) – karta před odesláním spočítá checksum a uloží ho na správné místo v rámci.
  - Některé karty to umí pouze pro TCP/IPv4, jiné umí spočítat checksum libovolného úseku v rámci.

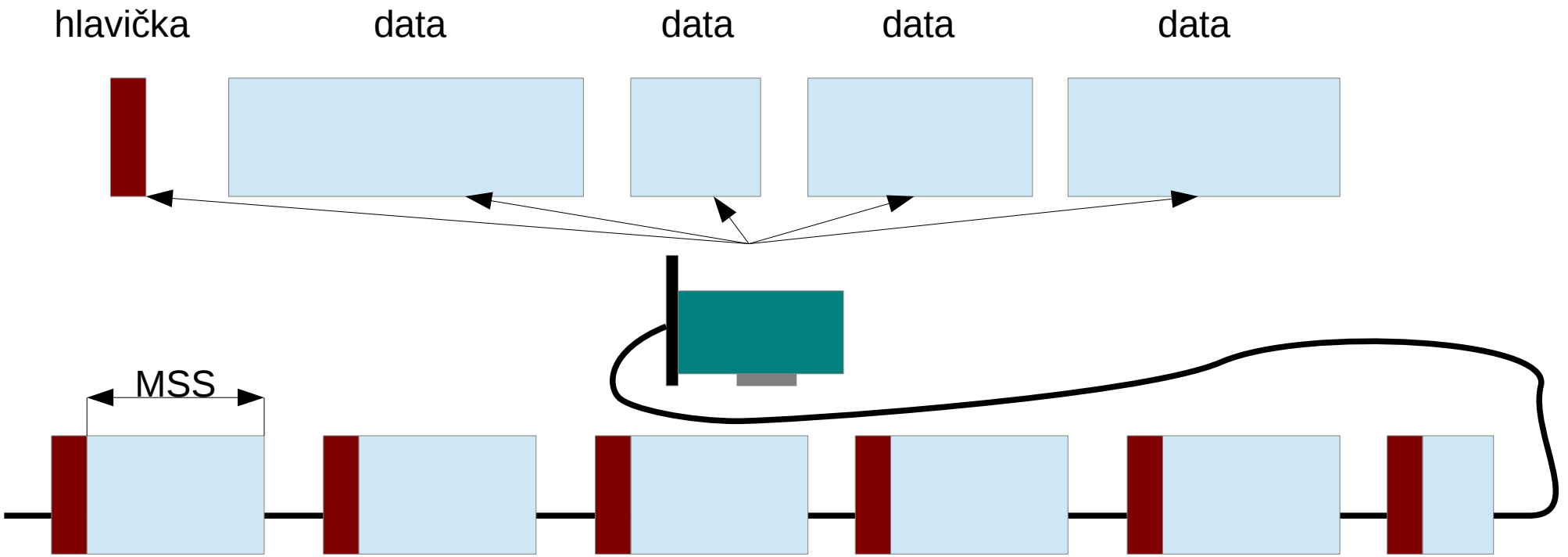
Rx/Tx checksum

- ethtool -k $DEV
- ethtool -K $DEV {tx,rx} {on|off}

- Wireshark může vidět chybné checksumy.

TSO

- TCP Segmentation Offload
- Nasekání velkých dat do TCP segmentů



TSO

- Vyžaduje zapnutou schopnost scatter-gather a Tx checksum
- ethtool -K $DEV sg on tx on tso on

- Wireshark může vidět obří vysílané rámce.

UFO

- UDP Fragmentation Offload
- Nasekání velkých UDP datagramů do IP fragmentů
- Méně obvyklé než TSO
  - Implementuje ho aktuálně jen s2io a některá virtuální zařízení.
- Aspoň se to srandovně jmenuje.

GSO

- Generic Segmentation Offload
- Zobecnění TSO, řešeno v software
- Síťovým stackem (protokoly, směrování, netfilter,...) prochází obří paket.
- Až těsně před předáním ovladači k vysílání je nasekán na patřičné kousky.
- Ušetří se tak zpracování, které by se jinak opakovalo pro každý kousek zvlášť (podobně jako u TSO).
- Lze použít i na protokoly jiné než TCP.

LRO

- Large Receive Offload
- Opak TSO, na přijímací straně
- Karta+driver z přijatých paketů skládají větší kusy, které pak předají do stacku najednou.
- Snižuje zátěž CPU při příjmu dat.
- Není možné zrekonstruovat původní pakety.
- Nelze použít při forwardování ani v bridge. Automaticky se vypíná.
- Skládání může zanést zpoždění.

GRO

- Generic Receive Offload
- Zobecnění LRO.
- Má přísnější kritéria pro skládání paketů.
  - Tak, aby šlo velký paket zpětně nasekat na původní kousky (pomocí GSO/TSO).
- Funguje forwarding i bridge.
- Není omezeno jen na TCP/IPv4.
- Řešeno v software při příjmu dávek v NAPI.

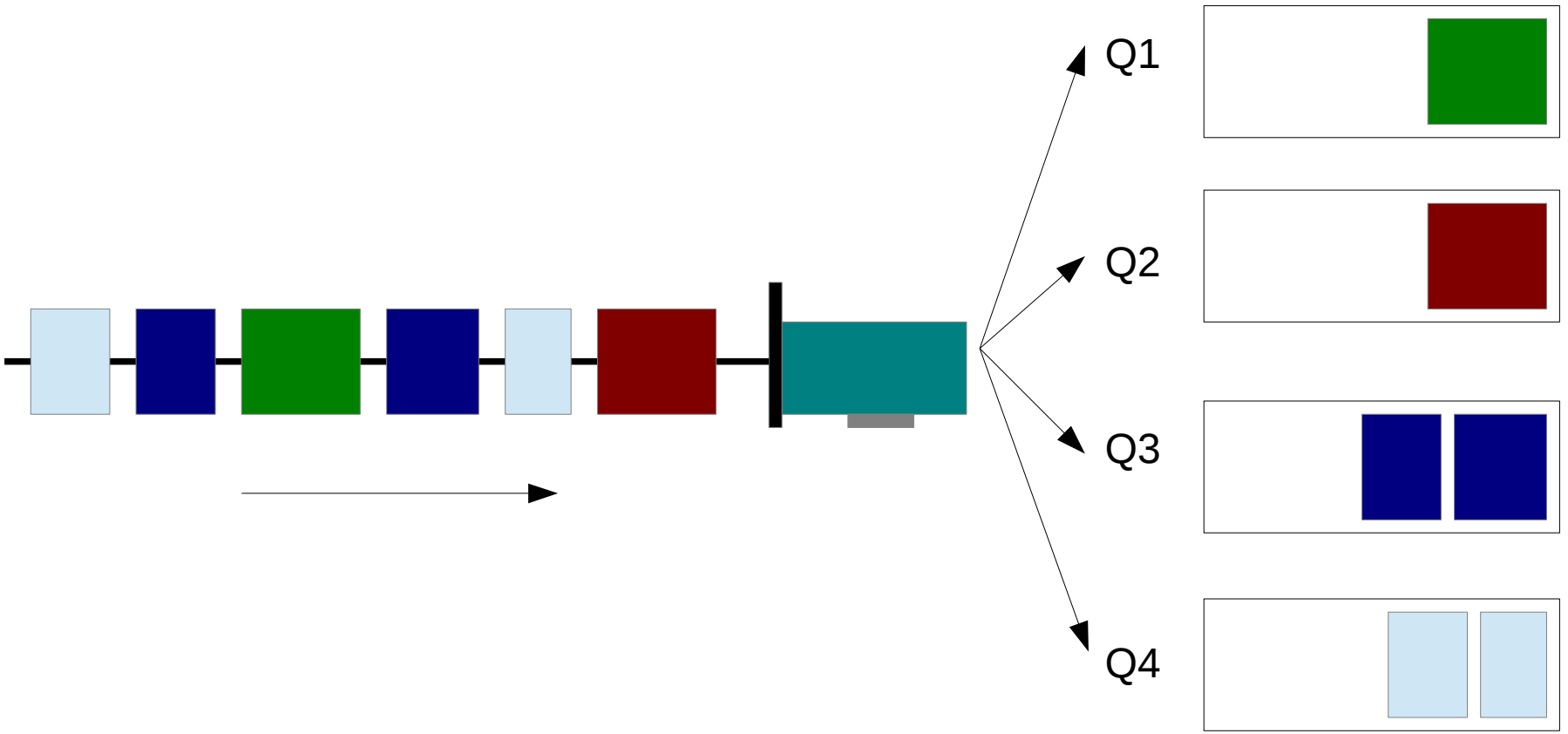
Škálovatelnost

- Umožnit paralelní zpracování
- Zvýšit výkon při větším počtu CPU
- Klíčem je:
  - minimalizovat čekání jednoho CPU na jiné,
  - využívat cache CPU

RSS

- Receive-Side Scaling
- Síťové karty mívají více Tx/Rx front (multi-queue).
- Karta třídí přijaté pakety do Rx front.
  - Snaží se přitom řadit pakety patřící do jednoho toku stále do stejné fronty.
    - např. na základě hashe z IP adres a čísel TCP portů
- Každá fronta má vlastní MSI-X přerušení.
  - Lze jej směrovat na vybrané CPU.

RSS ilustrace



RSS

- Tabulku, do které fronty která hodnota Rx hashe paket pošle, lze předepsat
  - ethtool --show-rxfh-indir ...
  - ethtool --set-rxfh-indir ...
- Některé karty umí nastavit konkrétní filtry pro rozdělování paketů do Rx front.
  - např. TCP port 80 do specifické fronty.
  - ethtool --show-nfc ...
  - ethtool --config-nfc ...

RPS

- Receive Packet Steering
- softwarová implementace RSS
  1. CPU1 přijme paket a spočítá pro něj hash (nebo ho dostane spočítaný už od karty)
  2. Na základě hashe vybere jiné CPU
  3. CPU1 vloží paket do přijímací fronty pro CPU2 a pošle CPU2 Inter-Processor Interrupt.
  4. CPU2 zpracuje paket.
- Defaultně vypnuto. Nutno nastavit bitmapu povolených přijímacích CPU:

  /sys/class/net/<dev>/queues/rx-<n>/rps_cpus

Výhody RPS

- Díky implementaci v software lze snadno přidat filtry pro nové protokoly.
- Lze použít i pro karty s jedinou Rx frontou.
- I pro multi-queue karty lze rozložit zátěž na více CPU než má karta front.
  - Nepřehánět. Nepřeposílat pakety na CPU s jinou pamětí (NUMA).

RFS

- Receive Flow Steering
- rozšíření RPS
- Bere navíc v úvahu, na kterém CPU běží aplikace, která data přijímá.
  - pro ještě lepší využití CPU cache
  - Když aplikace volá recvmsg(), sendmsg(), kernel si uloží číslo aktuálního CPU do tabulky pro vyhledávání datových toků. Tabulkou se pak řídí při příjmu paketů.
- Defaultně vypnuto.
  - /proc/sys/net/core/rps_sock_flow_entries
  - /sys/class/net/<dev>/queues/rx-<n>/rps_flow_cnt

Accelerated RFS

- A-RFS se má k RFS stejně jako RSS k RPS.
- rychlejší než prosté RFS
  - Odpadá mezikrok s prvotním příjmem paketu na CPU, které není finálním konzumentem.
  - Pakety chodí rovnou na CPU, na kterém běží přijímací aplikace.
  - Kernel kartu podle potřeby informuje o nové tabulce mapující toky na čísla Rx front. Rx fronty už jsou asociovány na konkrétní CPU.
- Zatím to umí pouze mlx4, sfc.

XPS

- Transmit Packet Steering
- Snaží se rozumně vybírat Tx frontu vhodnou pro CPU odesílající data.
  - vyhradit frontu výlučně pro takovou podmnožinu všech CPU, aby zpracování dokončení Tx operací probíhalo pouze na CPU z této podmnožiny.
    - Méně CPU soupeří o zamykání fronty. Ideálně má každé CPU svou Tx frontu, o kterou se nemusí dělit. Když už se musí dělit, tak aspoň jenom s CPU, se kterými má společnou cache.
- Defaultně vypnuto. Bitmapa CPU, které smí používat frontu:
  - /sys/class/net/<dev>/queues/tx-<n>/xps_cpus

Odkazy 1/2

Mindcraft studie:
http://www.mindcraft.com/whitepapers/first-nts4rhlinux.html

NAPI:
https://www.usenix.org/legacy/publications/library/proceedings/
als01/jamal.html
http://lwn.net/2002/0321/a/napi-howto.php3
https://lwn.net/Articles/30107/
https://lwn.net/Articles/244640/

TSO:
https://lwn.net/Articles/9123/
http://en.wikipedia.org/wiki/Large_segment_offload

GSO:
https://lwn.net/Articles/188489/

LRO:
https://lwn.net/Articles/243949/
http://www.linuxinsight.com/files/ols2005/grossman-reprint.pdf

Odkazy 2/2

GRO:
`https://lwn.net/Articles/358910/`

Linux a TOE:
`https://lwn.net/Articles/148697/`

Linux Device Drivers:
`https://lwn.net/Kernel/LDD3/`

Škálovatelnost:
`Documentation/networking/scaling.txt` ve zdrojácích Linux kernelu

RPS:
`https://lwn.net/Articles/362339/`

RFS:
`https://lwn.net/Articles/382428/`

Accelerated RFS:
`https://lwn.net/Articles/416332/`

XPS:
`https://lwn.net/Articles/412062/`