Sisteme de operare avansate

Connection Handoff Policies for TCP Offload Network Interfaces

Hyong-youb Kim and Scott Rixner
Rice University Houston, TX 77005
{hykim, rixner}@rice.edu

Costul procesării TCP

Procesarea TCP e dominată de costul accesării memoriei

- atunci când se accesează date sau structuri de date asociate conexiunii (headere, structuri interne SO)

Accesul la date

- copierea pachetului în user-space; soluţie: zero-copying I/O
- Checksumming; soluţie: hardware offloading

Accesul la structurile asociate conexiunii TCP

Chiar dacă aceste structuri sunt relativ mici (de ordinul KB) la un număr mare de conexiuni vom ieşi din cache

Nu există o soluţie care să remedieze această problemă

Folosirea unui TCP offloader-ul va adresa această problemă prin plasarea structurilor într-o memorie rapidă

TCP offloading

Folosirea plăcii de reţea pentru procesa direct TCP

Poate reduce încărcarea procesorului

anumite pachete un mai sunt procesate de host ci de NIC

Poate reduce încărcarea magistralei

anumite pachete nu mai ajung în host (ACK, FIN, etc.)

Poate creşte throughput-ul

Full TCP offloading

Toată procesare TCP se face de către NIC

Abordarea nu e scalabilă

- puterea de procesare (CPU, memorie) va limita performanţa (i.e. performanţa obţinută nu se "adaugă" la performanţa hostului)
- e nevoie de un protocol de comunicare între host şi NIC pentru a partaja porturile alocate, rutele şi adresele IP

TCP offloading NIC

Numărul de conexiuni pe care le poate suporta un astfel de device e relativ mic

Resursele NIC-ului sunt limitate

- procesor mai slab ca cel al sistemului
- memorie de capacitate mai mica

Viteza memoriei mai bună decât cea a sistemului

Concluzie

Placa de reţea poate fi folosită pentru a degreva procesorul de anumite operaţii TCP, dar acest lucru trebuie făcut cu măsură, în limita resurselor disponibile în cadrul NIC-ului.

Abordare

Connection handoff

- o parte din conexiunile active sunt pasate NIC-ului
- sistemul de operare va trimite cererile de send / recv direct către NIC
- NIC-ul va face procesarea TCP efectivă
- eventual SO poate reclama o conexiune înapoi, spre a fi tratată în stiva software

Abordare [2]

Spre deosebire de alte abordări SO este "în control"

- decide ce conexiuni să fie offloaded
- poate reduce numărul de conexiuni de pe NIC
- SO ia în continuare deciziile de rutare, selectare adresă sursă, alocare de porturi şi filtrare

Ce aduce nou paper-ul

- Studierea problematicii folosirii unui TCP offloading NIC
  - offloader-ul nu trebuie să influenţeze negativ conexiunile procesate de host
  - SO un trebuie să supraîncarce NIC-ul
  - Pentru că NIC-ul nu poate procesa decât un număr limitat de conexiuni SO trebuie să aleagă pentru offload acele conexiuni care pot fi tratate eficient de NIC
- Politici de utilizare a NIC-ului într-un mod eficient

Sunt aceste politici necesare?

Da. Simularea a patru scenarii de încărcare web arată că procesarea TCP făcută de NIC poate degrada performanţa

- cauza: creşte latenta introdusă de NIC de la 10us la 1ms
- în trei din cele patru scenarii performanţa s-a degradat cu 14-30%

Prioritatea pachete host

Dacă NIC-ul dă prioritate pachetelor ce trebuie trimise host-ului, se obţine o creştere de performanţa de 24% când NIC-ul un este suprasolicitat

Dacă NIC-ul este suprasolicitat (prea multe conexiuni), se observă o scădere de performanţă de 44%

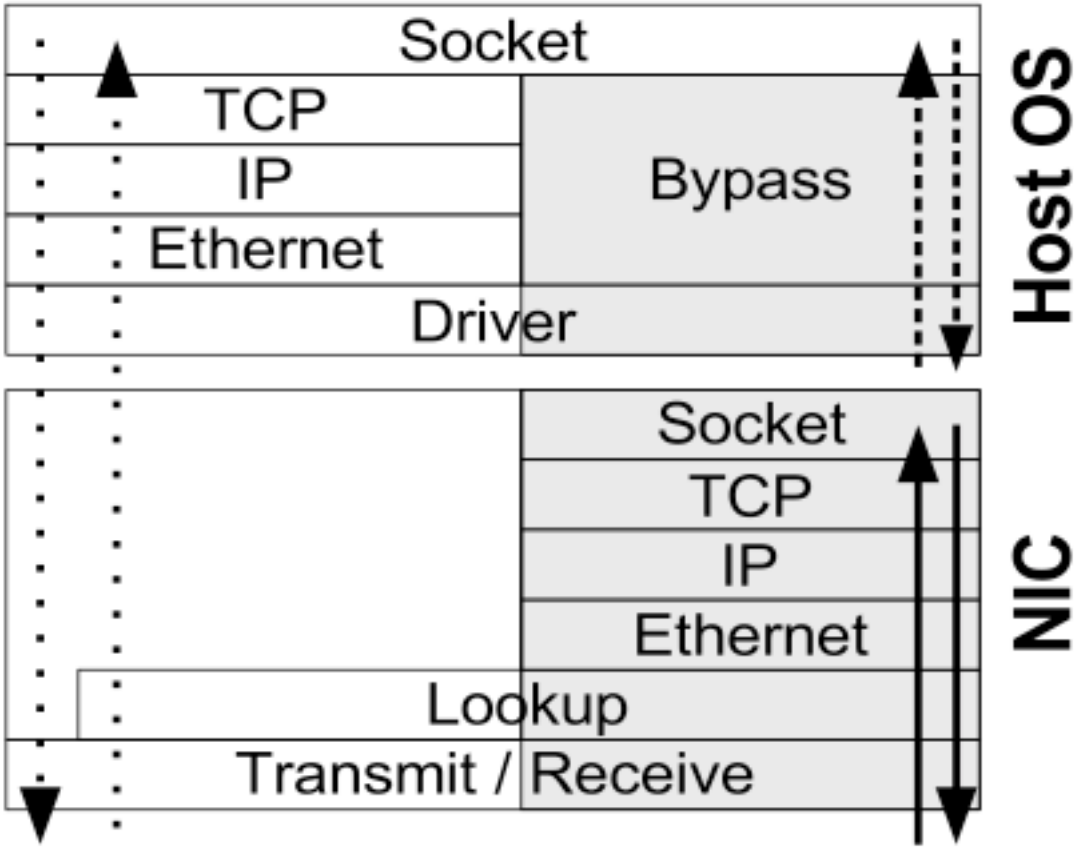
Limitarea conexiunilor

Dacă limităm şi numărul de conexiuni obţinem o creştere a performanţei de până la 31%

Care e politica de hand-off a conexiunilor?

- FCFS: merge dacă numărul de conexiuni concurente suportate de NIC e mare
- Altfel e preferat hand-off-ul pentru conexiunile cu durată mare de viaţă

Connection handoff



Connection handoff

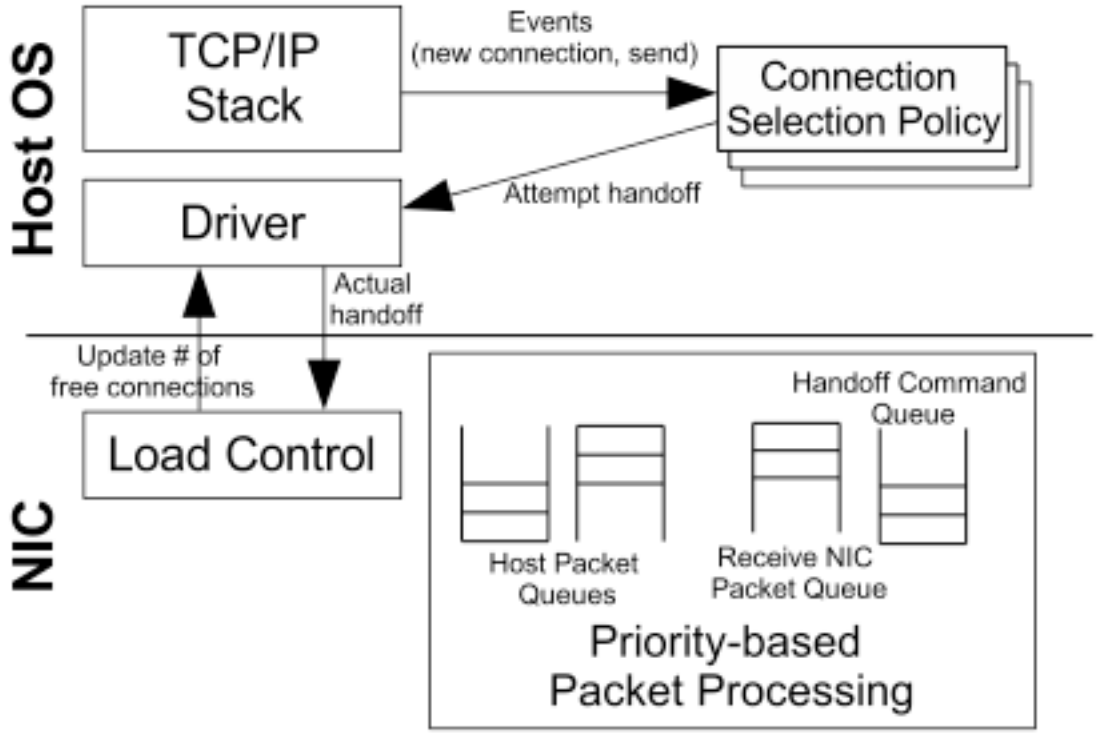
Lookup layer

- decide dacă pachetele trebuie sa ajungă la host sau să rămână în NIC
- hash based

Connection handoff API

- protocolul prin care NIC-ul şi driverul
  - mută o conexiune de la host la NIC şi invers
  - send/receive payload (pentru eficienţă datele sunt ţinute în memoria host-ului)

Arhitectura sistemului



Procesare pachetelor

Relativ la procesarea pachetelor NIC-ul are două roluri:

- procesarea pachetelor pentru conexiunile offloaded
- forwardarea pachetelor către host pentru conexiunile ce nu sunt offloaded

Probleme: procesarea pachetelor în NIC poate întârzia transmiterea pachetelor către host şi astfel afecta performanţa

Exemplu de degradare a performanţelor

Offloaded Connections	Packet Priority	Host			NIC	Requests/s
		Packet Delay (usec)		Idle (%)	Idle (%)	
		Send	Receive			
0	*no handoff*	2	2	0	62	23031
256	FCFS	3	3	0	49	23935
1024	FCFS	679	301	62	3	16121
1024	Host first	10	6	0	5	26663

Exemplu de degradare a performanţelor [2]

FCFS - pachetele sunt servite în ordinea în care vin, indiferent dacă trebuie tratate în NIC sau în host

Host first - prioritate pentru pachetele host

- două cozi: una cu pachetele hostului (send & receive)
- una cu pachetele NIC-ului
- se procesează întâi toate pachetele din coada host

Load control

Controlul dinamic al numărului de conexiuni offloaded

Previne supraîncărcarea NIC-ului

numarul de pachete din coada de recv (pentru pachetele NIC-ului) este folosit pentru a aproxima încărcărea NIC-ului

Parametrii de control

hard limit = limitează numărul maxim de conexiuni

- este setat la iniţializarea firmware-ului
- dependent de câtă memorie avem pe NIC

soft limit = numărul de conexiuni maxime curent

- modificat în funcţie de încărcarea curentă

Qlen = numărul de pachete ale NIC-ului din coada de receive

Parametrii de control [2]

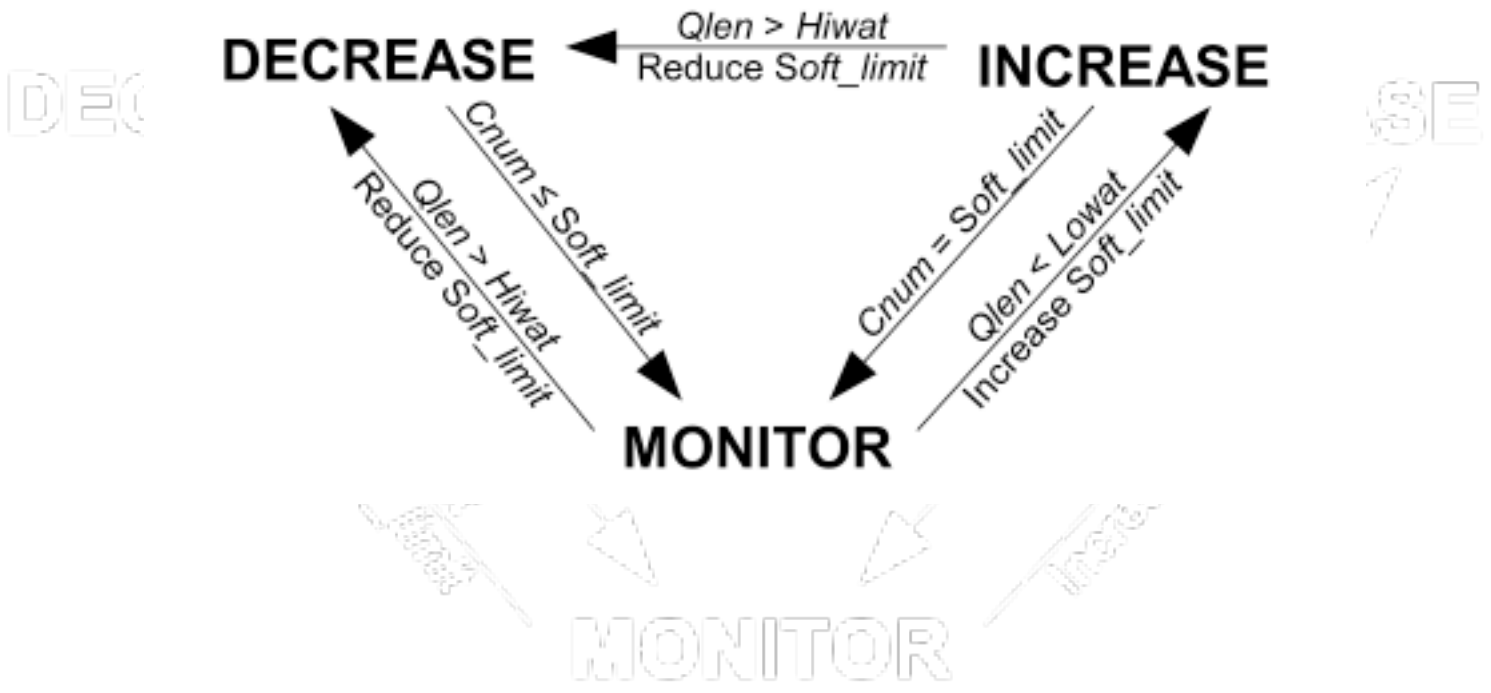
hiwat - high watermark pentru coada de pachete

> când qlen > high watermark trebuie să reducem numărul de conexiuni

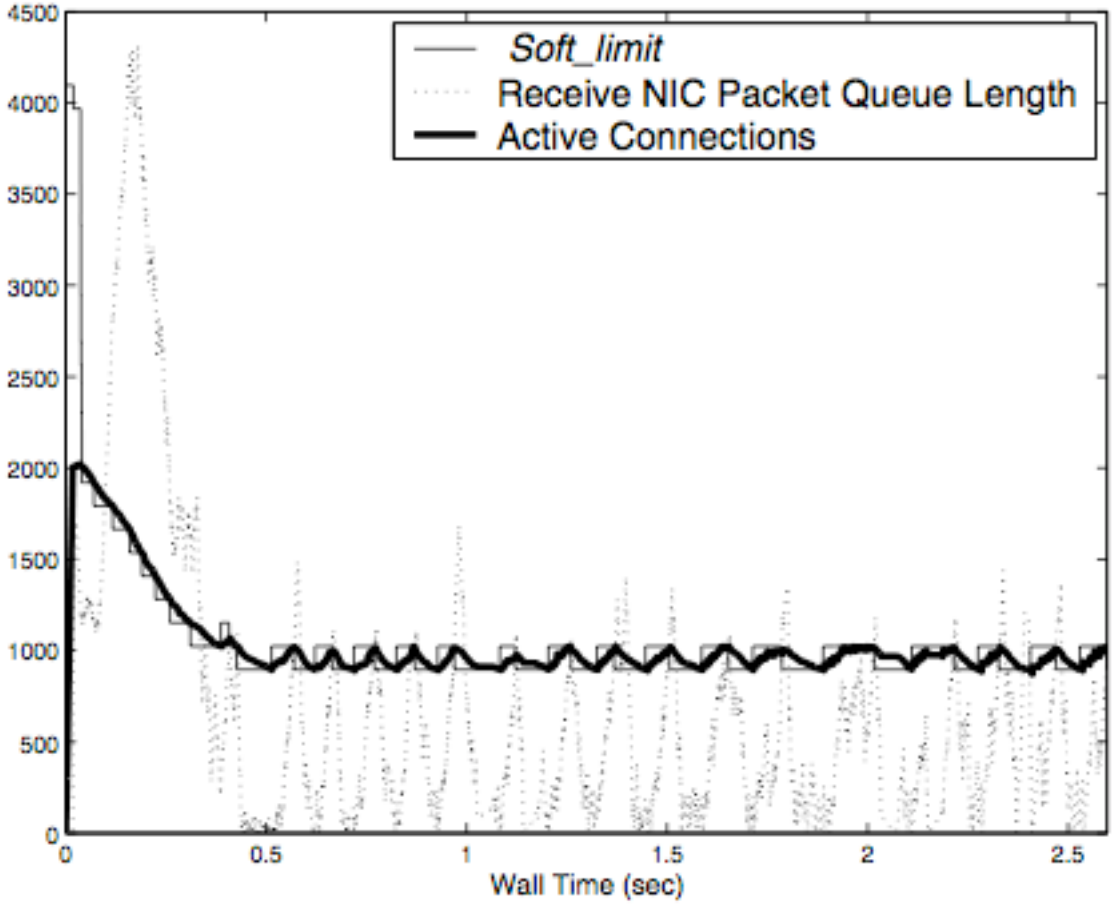
lowat - low watermark pentru coada de pachete

> când qlen <= low watermark putem să creştem numărul de conexiuni

Control load state machine



Exemplu


Soft_limit
Receive NIC Packet Queue Length
Active Connections
4500
4000
3500
3000
2500
2000
1500
1000
500
0
0
0.5
1
1.5
2
2.5
Wall Time (sec)

Selectarea conexiunilor

Decide ce conexiune să fie offloaded atunci când încărcarea este suficient de mică

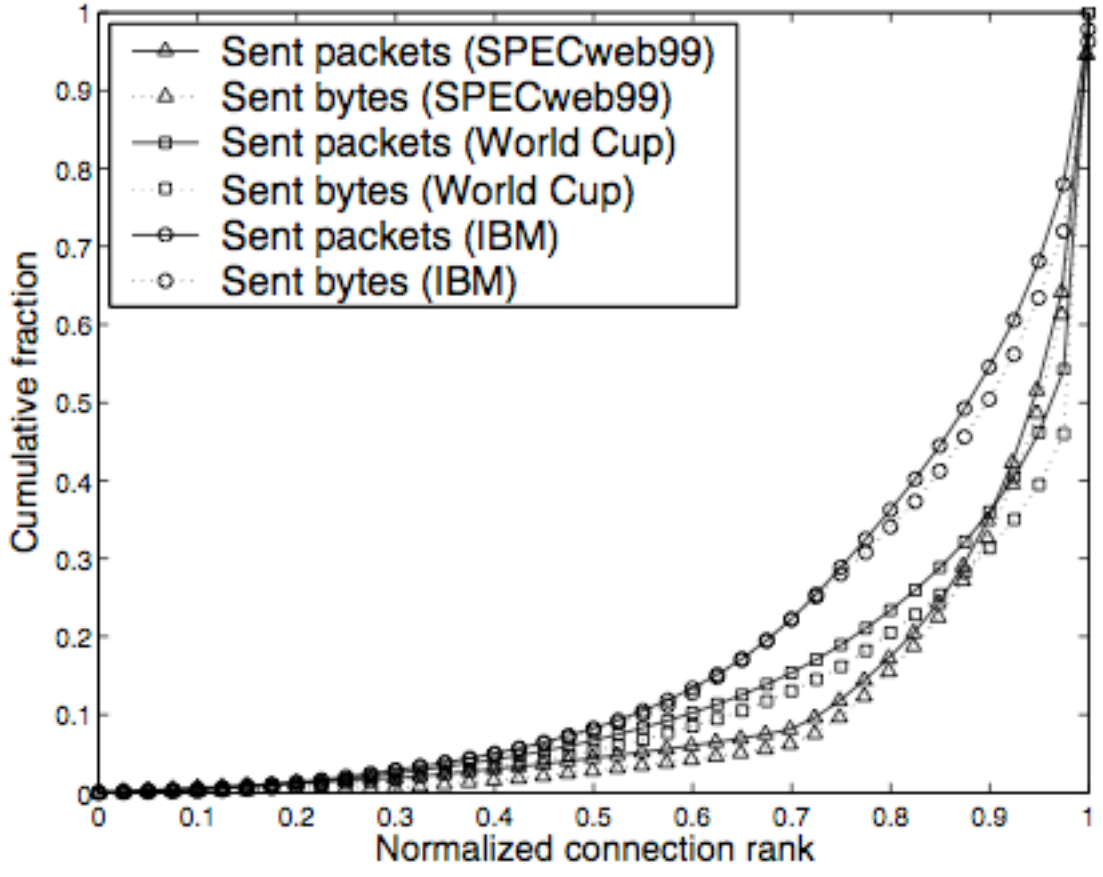
SO încearcă sa facă handoff

- la stabilirea conexiunii sau la send
- dacă handoff-ul eşuează nu se mai încearcă pentru acea conexiune mai târziu

Selectarea conexiunilor [2]

- Costul de hand off trebuie să fie balansat de beneficiile offloading-ului
- Caracteristici ale conexiunilor ce influenţează aceast raport
  - durata de viaţă (măsurată în numărul de pachete trimise per conexiune)
  - rata de pachete
- SO trebuie să estimeze durata de viaţă
  - se foloseşte de o distribuţie ce descrie comportarea traficului web

Distribuţia duratei de viaţă



Discuţie

50% din conexiuni trimit 10% din trafic

celelate 50% trimit 90% din trafic

Ne putem folosi de această observaţie pentru a determina conexiunile cu o durată de viaţă mare pe baza unui valori limită (threshold)

Evaluare

SO: FreeBSD 4.7

Hardware: simulat cu Simics

- Simics = un simulator de sistem complet
- procesor = x86
- a fost extins...
- ... cu un emulator de memorie
  - cache, memory controller, simulator DRAM
- ... şi un NIC

Simulare NIC

Procesor MIPS (400, 600, 800 MIPS)

32MB memorie

Controller PCI

- DMA

Ethernet controller

- MAC, CRC

Timer

Detalii

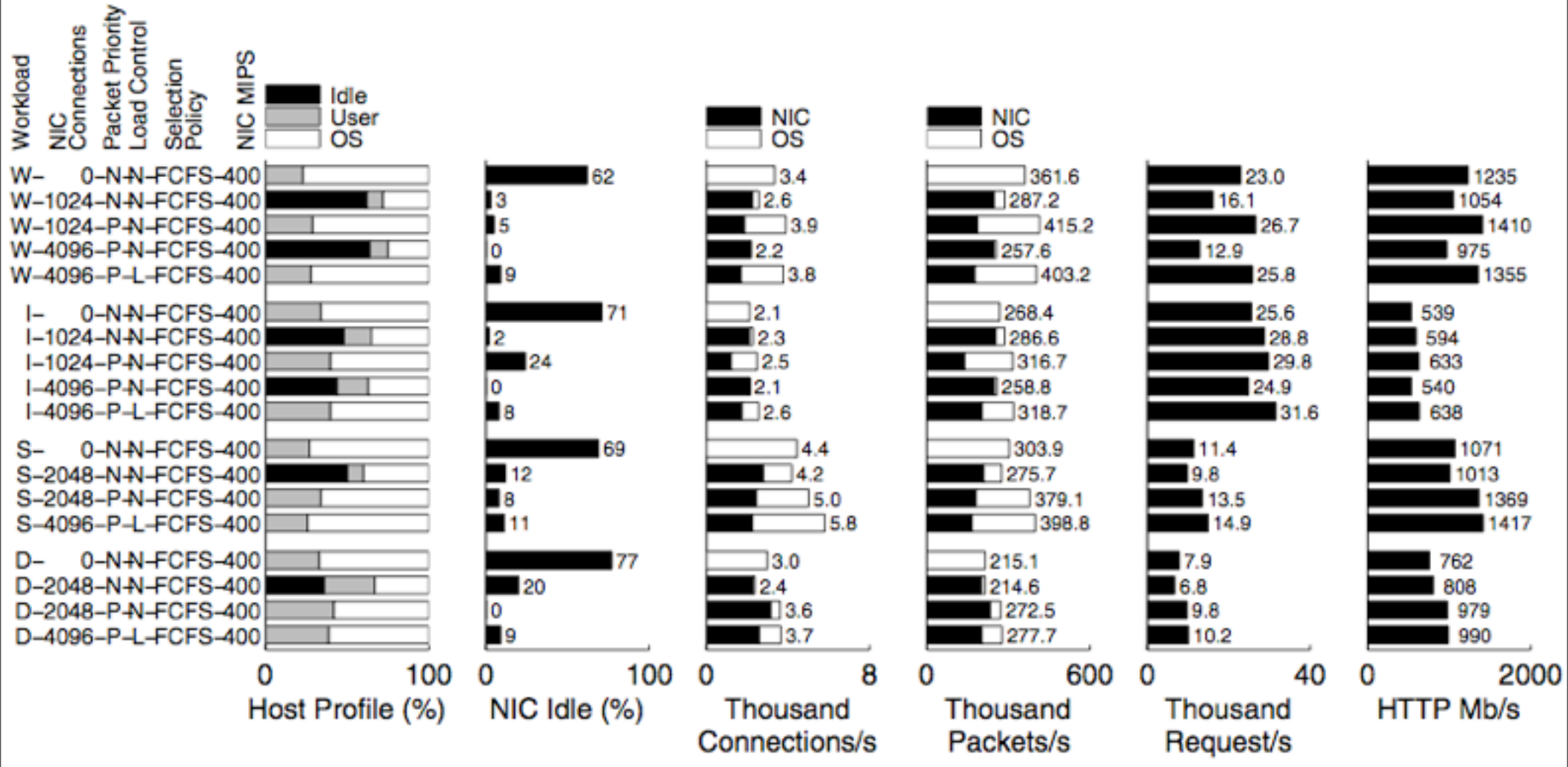
	Configuration
CPU	Functional, single-issue, 2GHz x86 processor Instantaneous instruction fetch
L1 cache	64KB data cache Line size: 64B, associativity: 2-way Hit latency: 1 cycle
L2 cache	1MB data cache Line size: 64B, associativity: 16-way Hit latency: 15 cycles Prefetch: next-line on a miss
DRAM	DDR333 SDRAM of size 2GB Access latency: 195 cycles
NIC	Functional, single-issue processor Varied instruction rates for experiments Varied maximum number of connections 10Gb/s wire

Table 2: Simulator configuration.

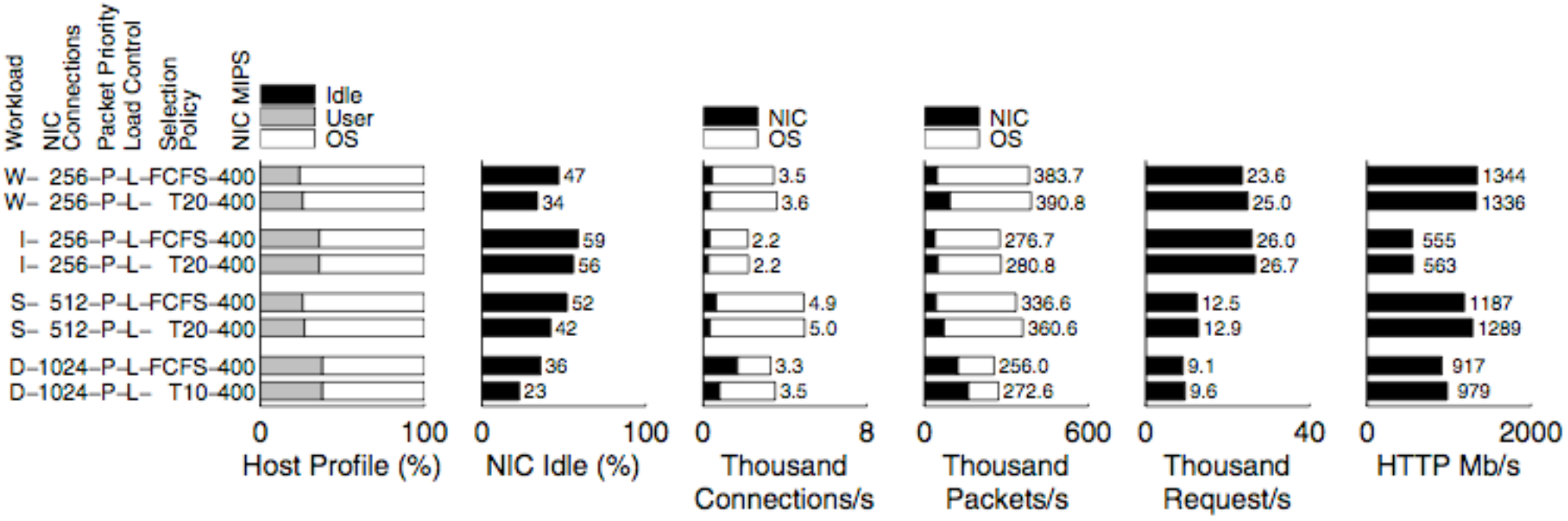
Rezultate [1]

Workload	Web server workload **W**: World Cup, 2048 clients **I**: IBM, 2048 clients **D**: SPECweb99, 2048 clients **S**: SPECweb99-static, 4096 clients
NIC Connections	Maximum number of connections on the NIC **0** means handoff is not used.
Packet Priority	*host first* priority-based packet processing on the NIC **P**: Used **N**: Not used
Load Control	Load control mechanism on the NIC **L**: Used **N**: Not used
Selection Policy	Connection selection policy used by the operating system **FCFS**: First-come, first-served **T***n*: Threshold with value *n*
NIC MIPS	Instruction rate of the NIC in million instructions per second

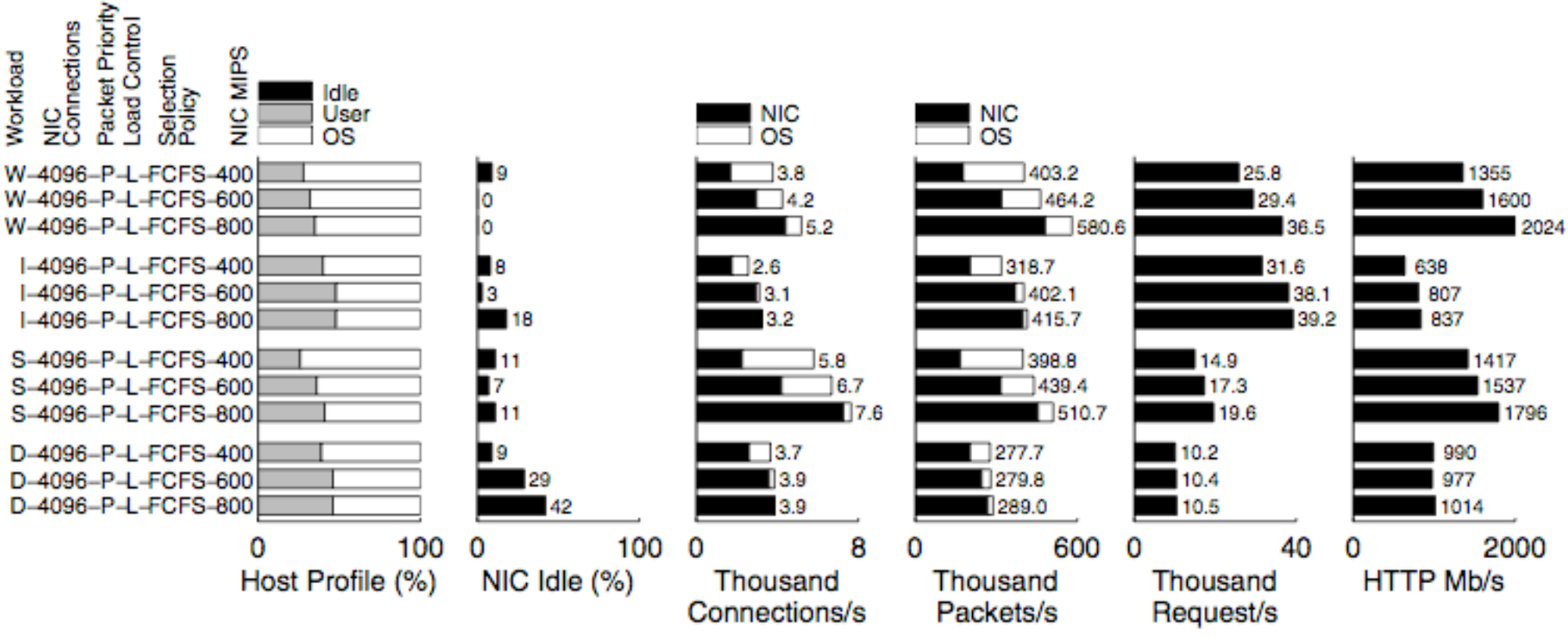
Rezultate [2]


Workload
NIC Connections
Packet Priority
Load Control
Selection Policy
NIC MIPS
Idle
User
OS
W- 0-N-N-FCFS-400
W-1024-N-N-FCFS-400
W-1024-P-N-FCFS-400
W-4096-P-N-FCFS-400
W-4096-P-L-FCFS-400
I- 0-N-N-FCFS-400
I-1024-N-N-FCFS-400
I-1024-P-N-FCFS-400
I-4096-P-N-FCFS-400
I-4096-P-L-FCFS-400
S- 0-N-N-FCFS-400
S-2048-N-N-FCFS-400
S-2048-P-N-FCFS-400
S-4096-P-L-FCFS-400
D- 0-N-N-FCFS-400
D-2048-N-N-FCFS-400
D-2048-P-N-FCFS-400
D-4096-P-L-FCFS-400
0 100
Host Profile (%)
62 3 5 0 9
71 2 24 0 8
69 12 8 11
77 20 0 9
0 100
NIC Idle (%)
NIC
OS
3.4 2.6 3.9 2.2 3.8
2.1 2.3 2.5 2.1 2.6
4.4 4.2 5.0 5.8
3.0 2.4 3.6 3.7
0 8
Thousand Connections/s
NIC
OS
361.6 287.2 415.2 257.6 403.2
268.4 286.6 316.7 258.8 318.7
303.9 275.7 379.1 398.8
215.1 214.6 272.5 277.7
0 600
Thousand Packets/s
23.0 16.1 26.7 12.9 25.8
25.6 28.8 29.8 24.9 31.6
11.4 9.8 13.5 14.9
7.9 6.8 9.8 10.2
0 40
Thousand Request/s
1235 1054 1410 975 1355
539 594 633 540 638
1071 1013 1369 1417
762 808 979 990
0 2000
HTTP Mb/s

Rezultate [3]


Workload
NIC Connections
Packet Priority
Load Control
Selection Policy
NIC MIPS
Idle
User
OS
W– 256–P–L–FCFS–400
W– 256–P–L– T20–400
I– 256–P–L–FCFS–400
I– 256–P–L– T20–400
S– 512–P–L–FCFS–400
S– 512–P–L– T20–400
D–1024–P–L–FCFS–400
D–1024–P–L– T10–400
47
34
59
56
52
42
36
23
NIC
OS
3.5
3.6
2.2
2.2
4.9
5.0
3.3
3.5
383.7
390.8
276.7
280.8
336.6
360.6
256.0
272.6
23.6
25.0
26.0
26.7
12.5
12.9
9.1
9.6
1344
1336
555
563
1187
1289
917
979
0
100
Host Profile (%)
0
100
NIC Idle (%)
0
8
Thousand Connections/s
0
600
Thousand Packets/s
0
40
Thousand Request/s
0
2000
HTTP Mb/s

Rezultate [4]


Workload
NIC Connections
Packet Priority
Load Control
Selection Policy
NIC MIPS
Idle
User
OS
W-4096-P-L-FCFS-400
W-4096-P-L-FCFS-600
W-4096-P-L-FCFS-800
I-4096-P-L-FCFS-400
I-4096-P-L-FCFS-600
I-4096-P-L-FCFS-800
S-4096-P-L-FCFS-400
S-4096-P-L-FCFS-600
S-4096-P-L-FCFS-800
D-4096-P-L-FCFS-400
D-4096-P-L-FCFS-600
D-4096-P-L-FCFS-800
0
100
Host Profile (%)
9
0
0
8
3
18
11
7
11
9
29
42
0
100
NIC Idle (%)
NIC
OS
3.8
4.2
5.2
2.6
3.1
3.2
5.8
6.7
7.6
3.7
3.9
3.9
0
8
Thousand Connections/s
NIC
OS
403.2
464.2
580.6
318.7
402.1
415.7
398.8
439.4
510.7
277.7
279.8
289.0
0
600
Thousand Packets/s
25.8
29.4
36.5
31.6
38.1
39.2
14.9
17.3
19.6
10.2
10.4
10.5
0
40
Thousand Request/s
1355
1600
2024
638
807
837
1417
1537
1796
990
977
1014
0
2000
HTTP Mb/s