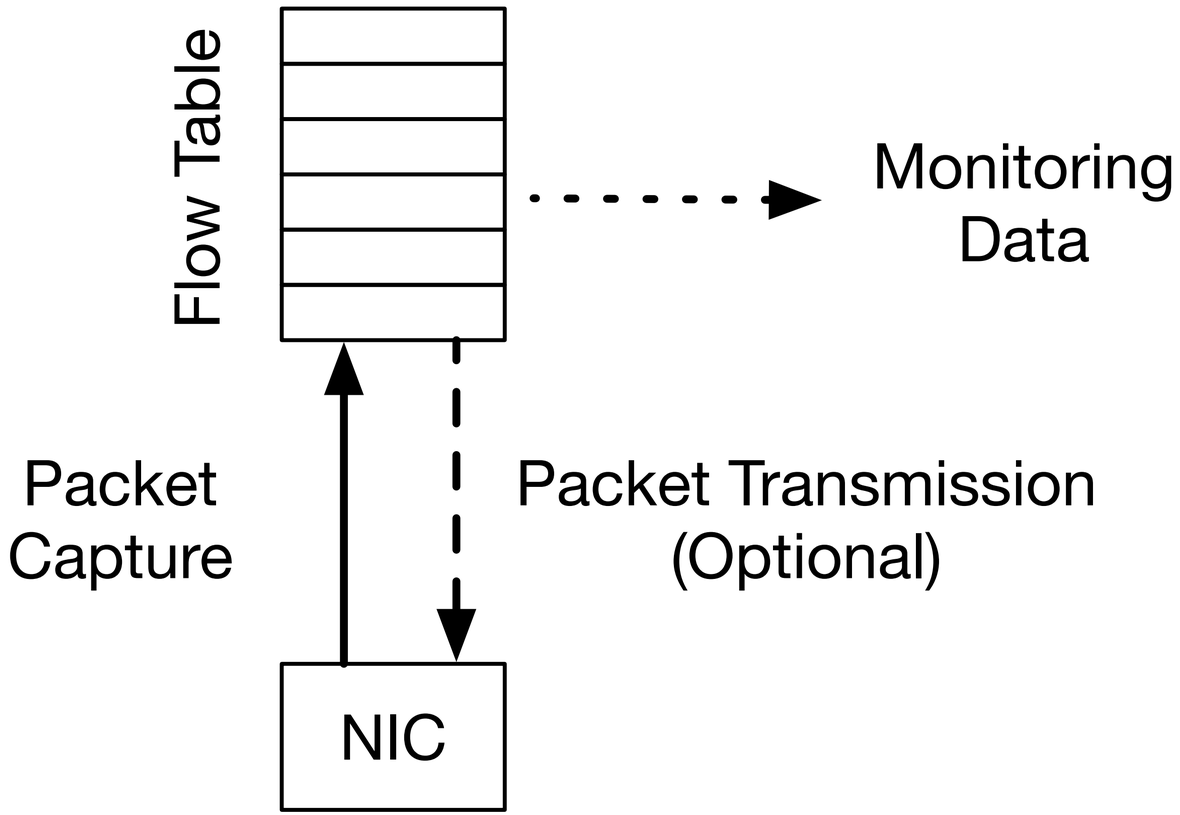
Advancements in Traffic Processing Using Programmable Hardware Flow Offload

Luca Deri, Alfredo Cardigliano <{deri,cardigliano}@ntop.org>
Francesco Fusco <ffu@zurich.ibm.com>

Introduction

- Network traffic processing applications play a critical role in monitoring and safeguarding modern networks.
- The complexity of the monitored traffic made software-based DPI-enabled probes necessary (packet header inspection is not enough).
- Enabling the implementation of software-based probes capable of DPI-inspecting traffic at wire-rate on commodity hardware has been a journey requiring significant research and development efforts.

Flow Classification



Motivation [1/2]

- Feeding the application on high speed links has been made possible with packet capture acceleration technologies:
  - By enabling kernel-bypass (application-level DMA).
  - Exploiting modern multi-cores CPU by load-balancing traffic in hardware with RSS-like technologies.
- Monitoring applications, both passive (e.g. NetFlow or IDS systems) or inline (e.g. IPS systems) typically have to analyse and maintain the state of each network flow.

Motivation [2/2]

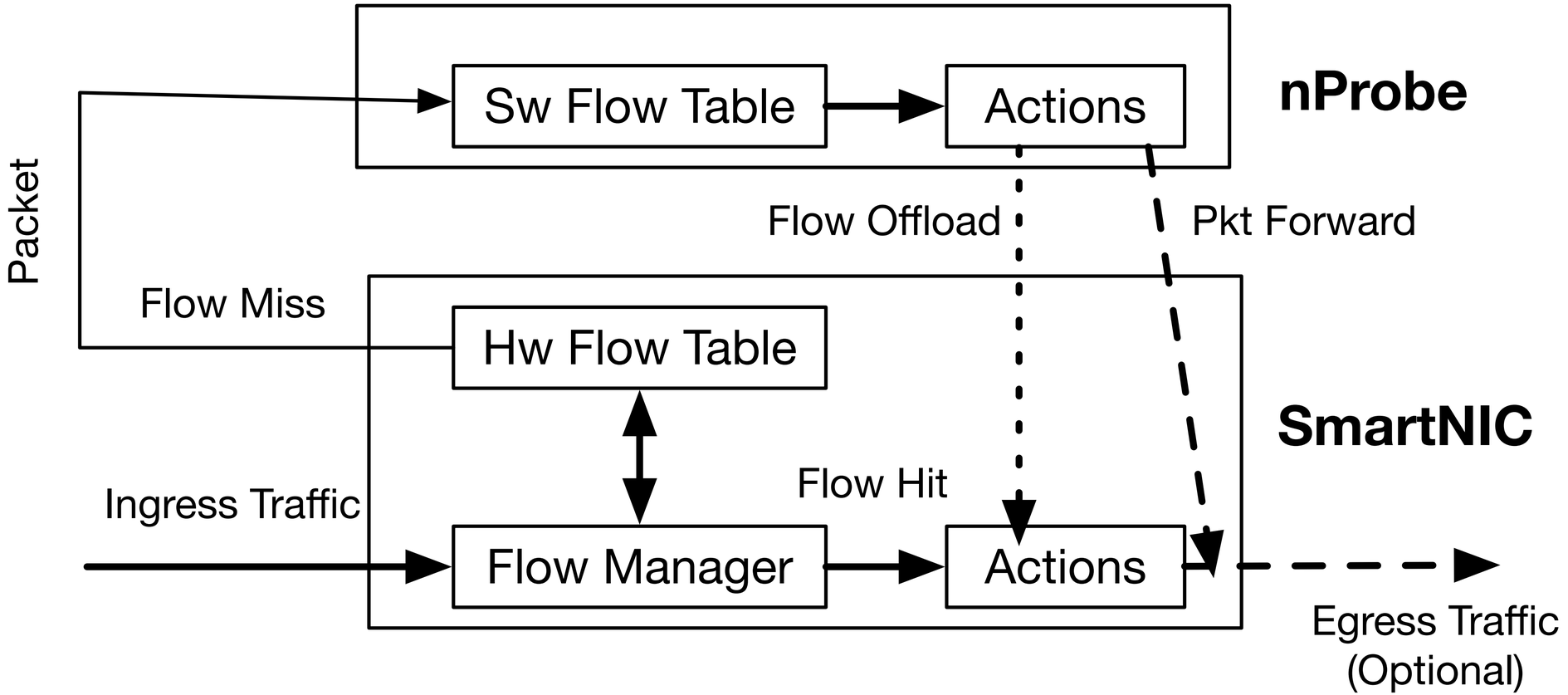
- A flow table is an in-memory data structure where the network probe stores information (typically the flow key and metadata).
- On DPI (Deep Packet Inspection) enabled probes the metadata may include:
  - Flow detection stage metadata.
  - Information extracted from application layer protocols (e.g. the HTTP URL or the VoIP caller). It is worth noting that the probe rarely needs more than just a few packets at the beginning of the communication to extract those information.
  - Simple statistics about the packet stream belonging to the flow (number of bytes and packets transmitted).

Flow Table Offload

- Modern SmartNICs have introduced hardware-accelerated flow offload mechanisms which can be seen as the next generational step of acceleration technologies.

- Target: monitoring and cybersecurity probes implemented in software and deployed on general-purpose operating systems.

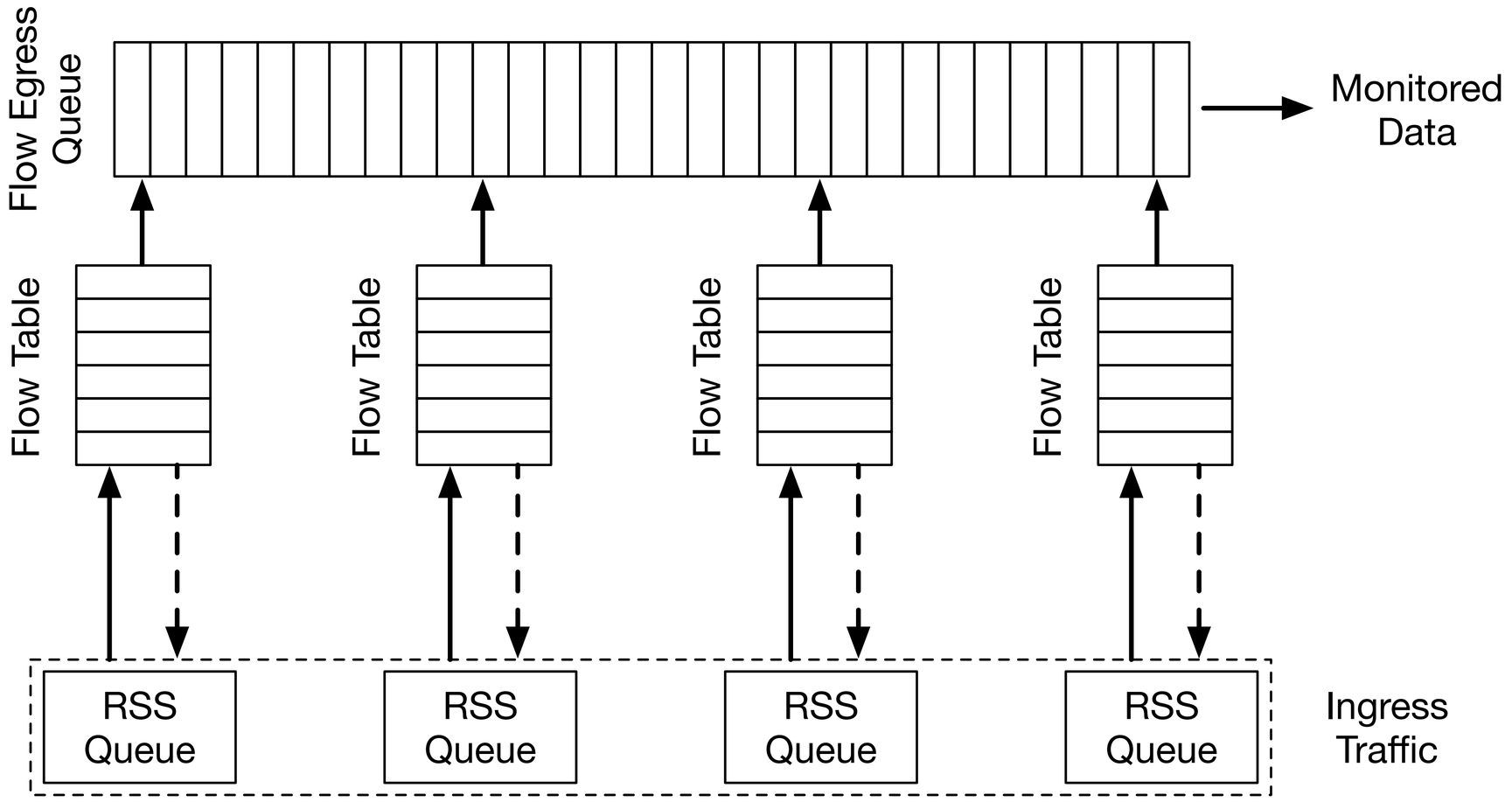
Packet Lifecycle with Flow Offload



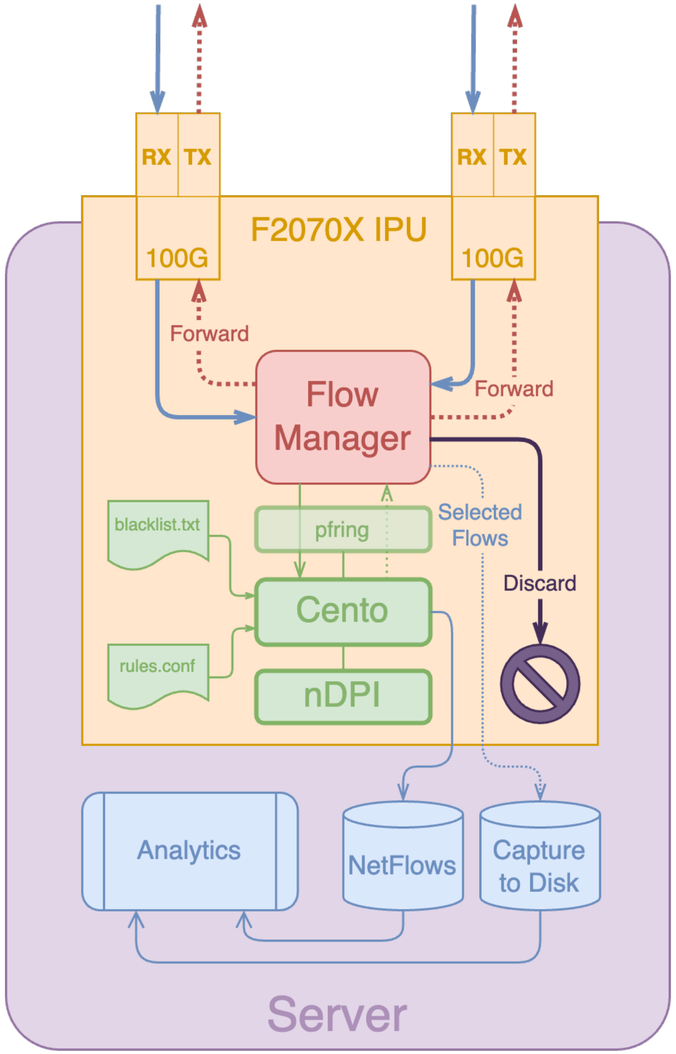
Validation

- Goal: evaluate the performance gain in terms of CPU utilisation and packet loss with flow table offload enabled with respect to a pure software-based solution.
- Software used for the validation: nProbe Cento, a 100 Gbit capable NetFlow probe.
- Adapter used for the validation: Napatech SmartNIC NT200A02 (PCIe Gen3, supporting up to 100 Gbps in/out).

nProbe Cento Architecture



Napatech Flow Manager



Packet Markers

- Received packets (pfring_recv_pkt API) are marked:
  - Known flow packet (hit: flow match in the adapter)
  - Unknown flow packet (miss: no flow match)
  - Unhandled packet (no hardware pipeline resources, unable to match the flow)

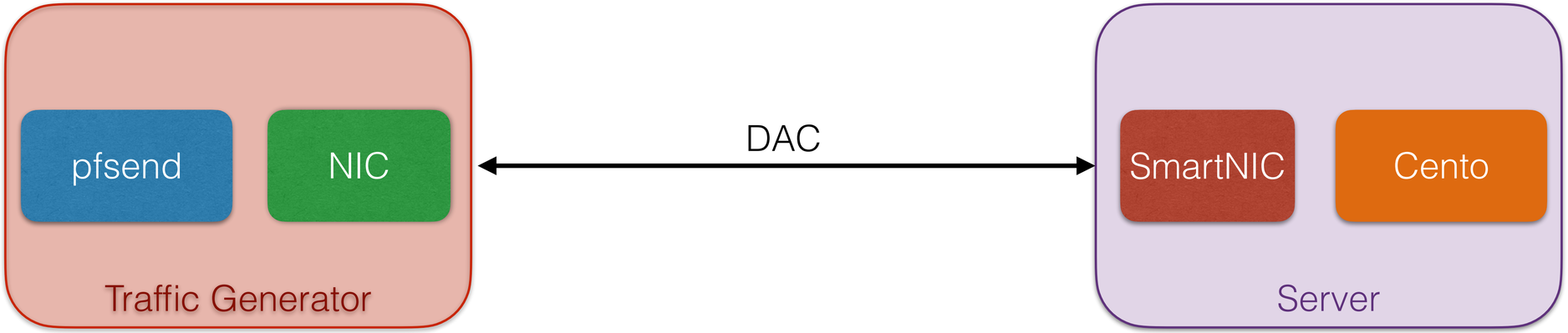
Learning Flows

- Flows are programmed by the software upon packet processing (i.e. when DPI completes)
- Flows are configured by 5-tuple (pfring_add_hw_rule API) with:
  - Metadata: Flow ID generated by the application.
  - Options for the Flow Manager (e.g. TCP auto unlearn).
  - Action: discard, forward (to the software or a twin interface).

Hardware Flow Events

- Flow events delivered by the adapter (pfring_recv_flow API) include:
  - Flow unlearned due to (lifetime, idle) timeout.
  - Flow unlearned due to TCP (flags) termination.
  - Flow unlearned due to application unlearn command.

Testbed



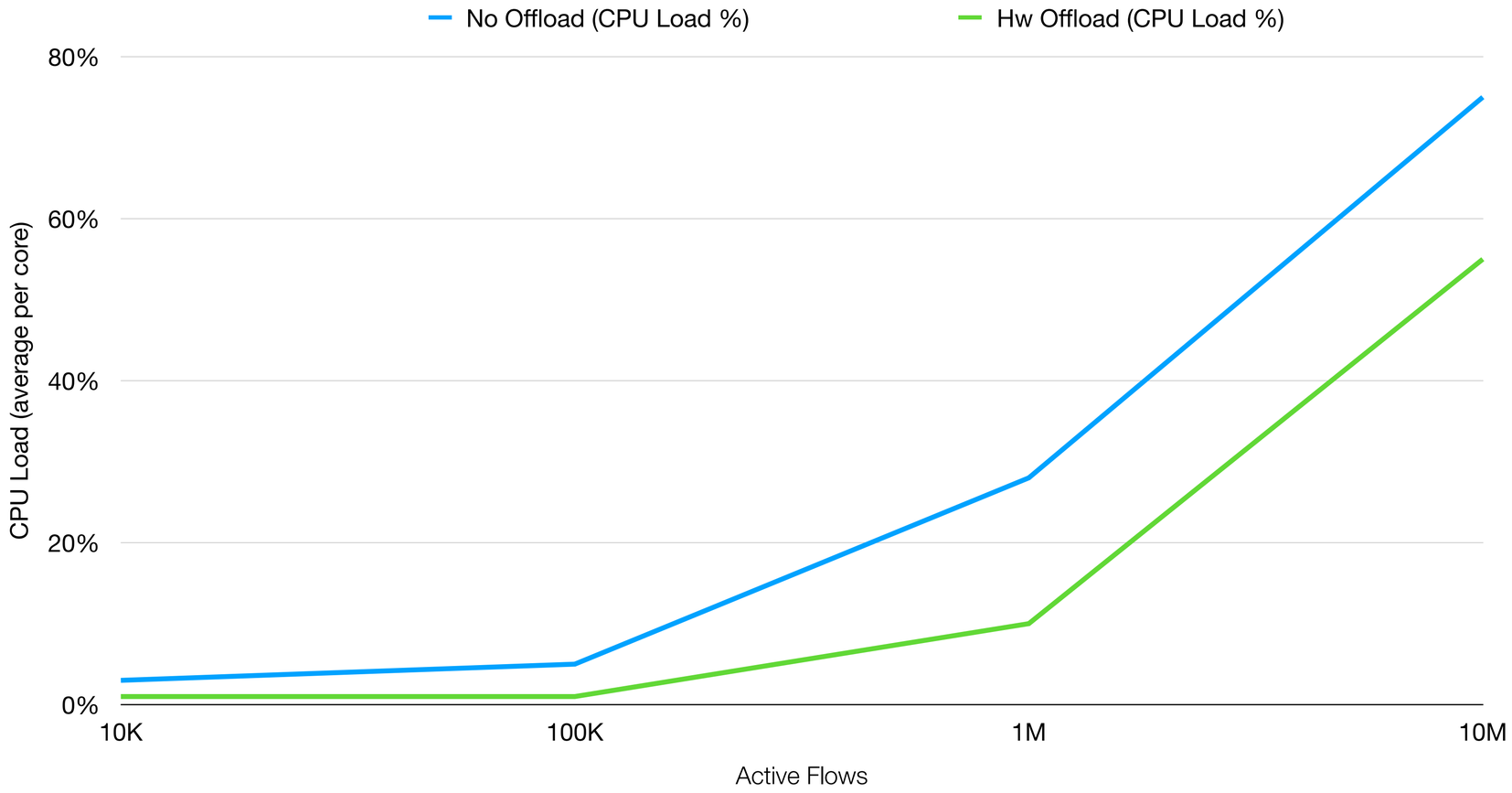
- Traffic Generator specs:
  - CPU: Intel Xeon E-2136
  - RAM: 2x 16 GB DDR4
  - NIC: Napatech NT100E3
- Device Under Test specs:
  - CPU: Intel Xeon Gold 6526Y
  - RAM: 8 x 16 GB DDR5
  - NIC: Napatech NT200A02

Test Results

- The traffic generator was able to generate 80 Gbps with 970-byte packets (10 Mpps), or 60 Gbps with 60-byte packets (89 Mpps).
- The number of active flows and new flows per second have been tuned to test the solution at different traffic conditions:
  - 10K active flows, 1K new flows/sec (flow birth rate).
  - 100K active flows, 10K new flows/sec.
  - 1M active flows, 100K new flows/sec.
  - 10M active flows, 1M new flows/sec.
- CPU load (average per core) and packet loss have been measured.

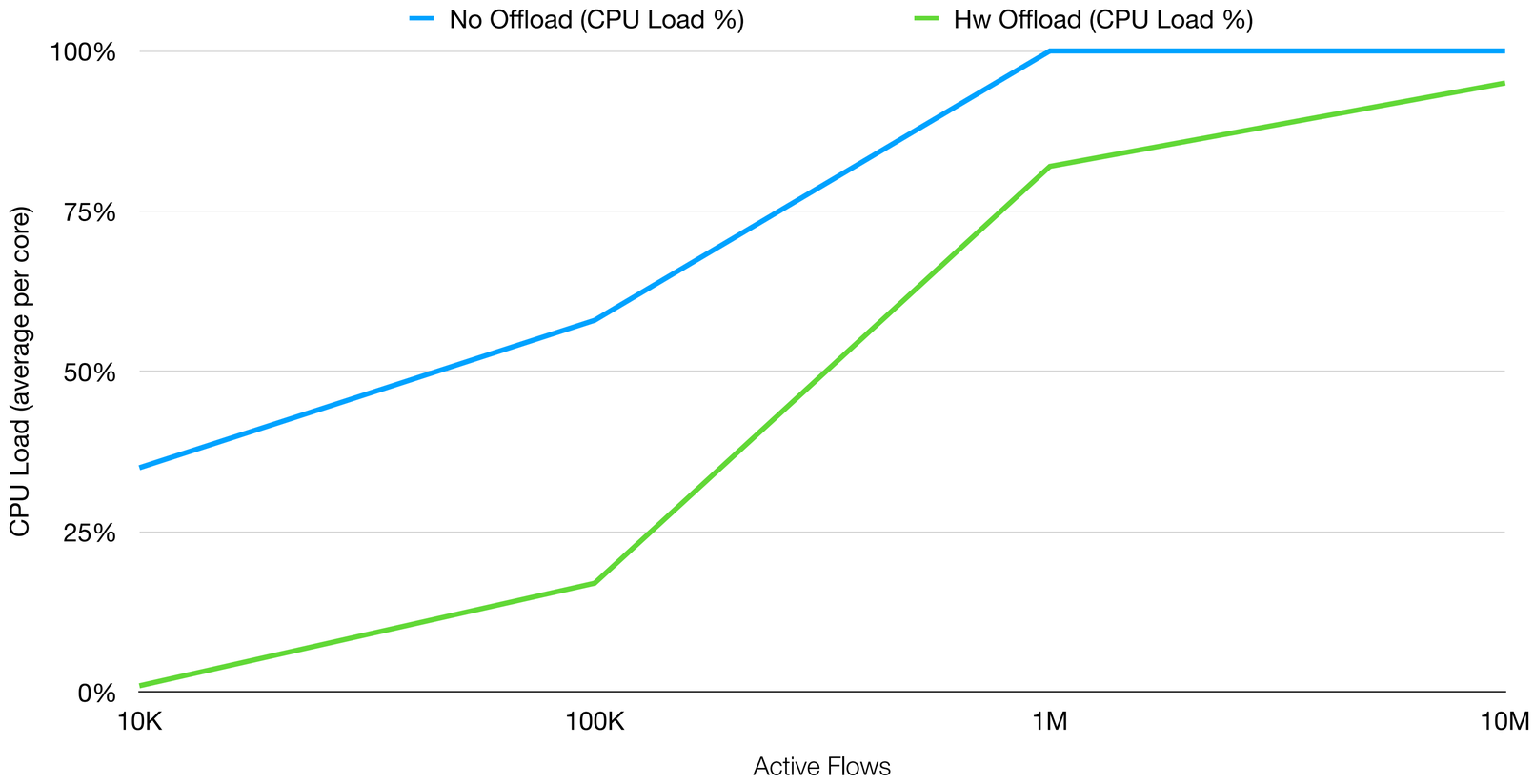
Passive Mode: CPU Load

- @ 10 Mpps 80 Gbps with DPI on 16 cores (RSS)



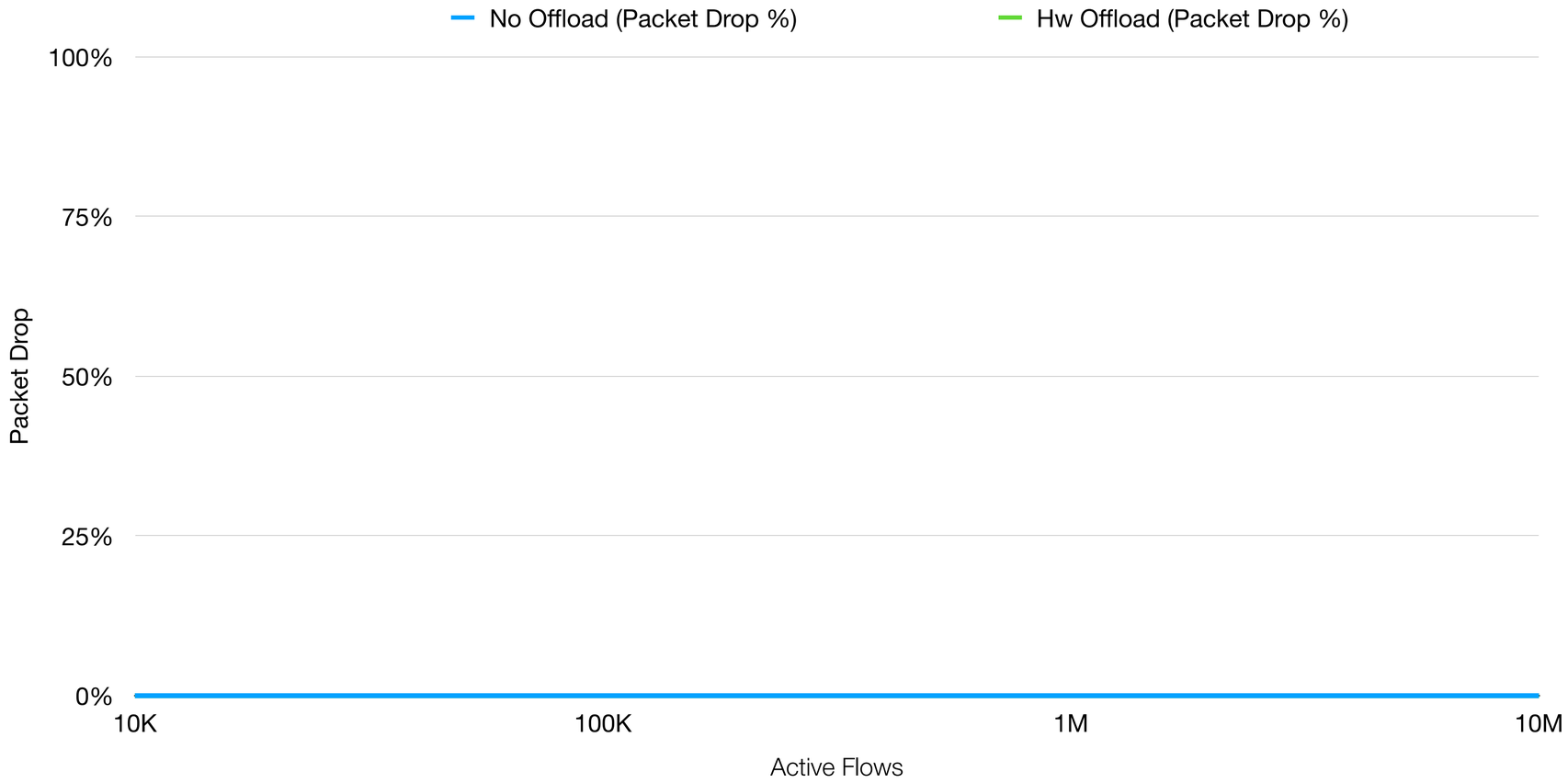
Passive Mode: CPU Load

- @ 89 Mpps 60 Gbps with DPI on 16 cores (RSS)



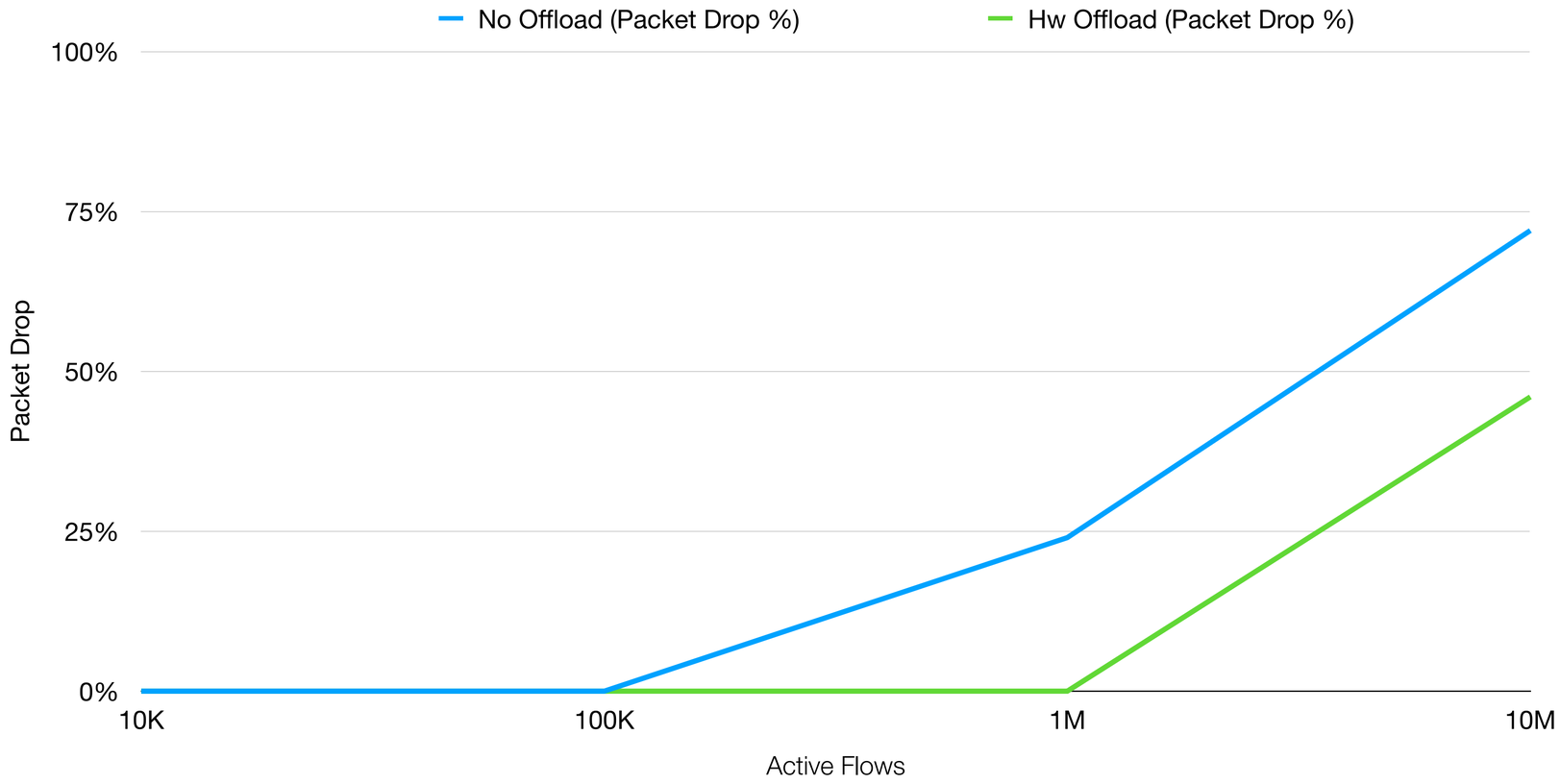
Passive Mode: Packet Drops

- @ 10 Mpps 80 Gbps with DPI on 16 cores (RSS)



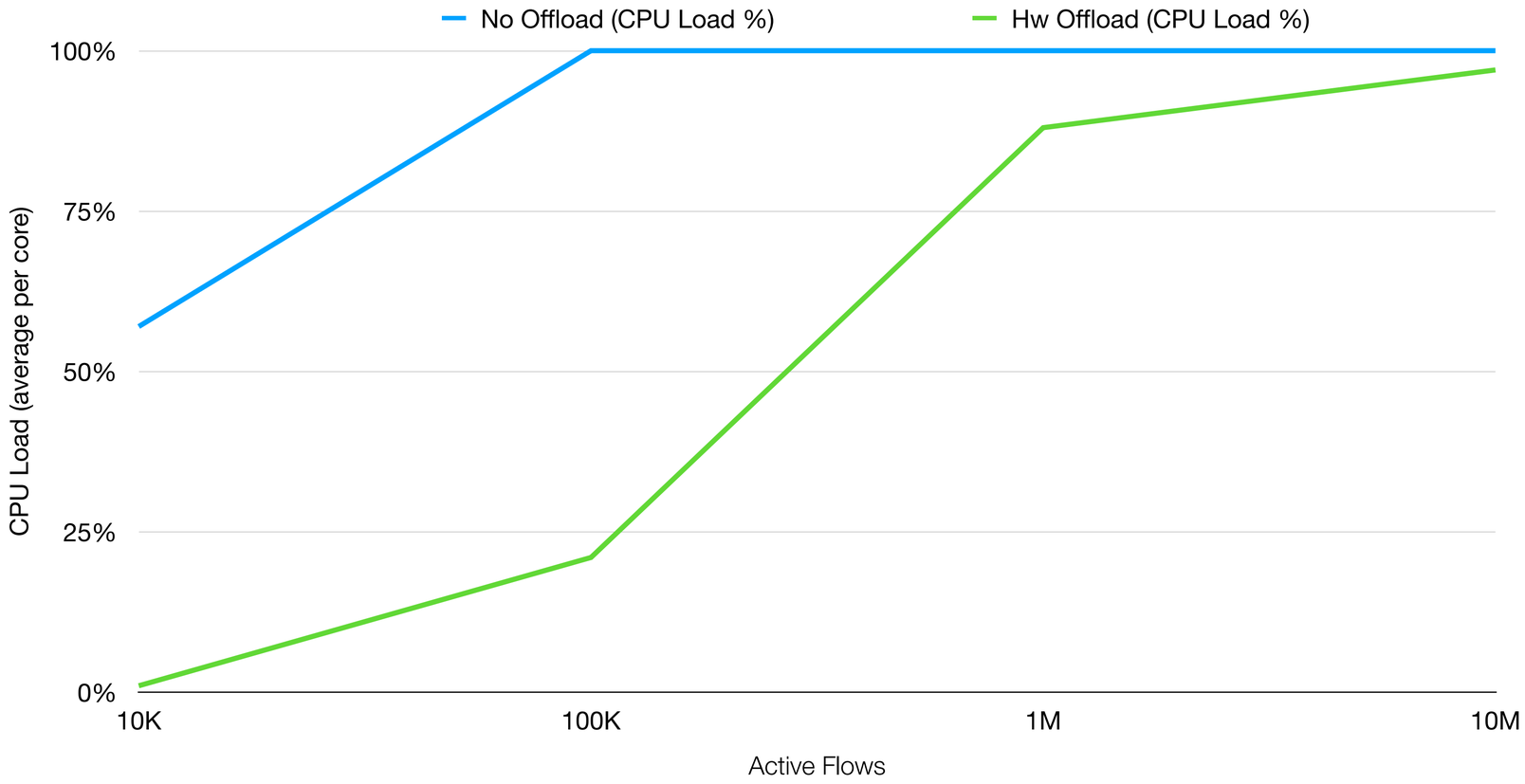
Passive Mode: Packet Drops

- @ 89 Mpps 60 Gbps with DPI on 16 cores (RSS)



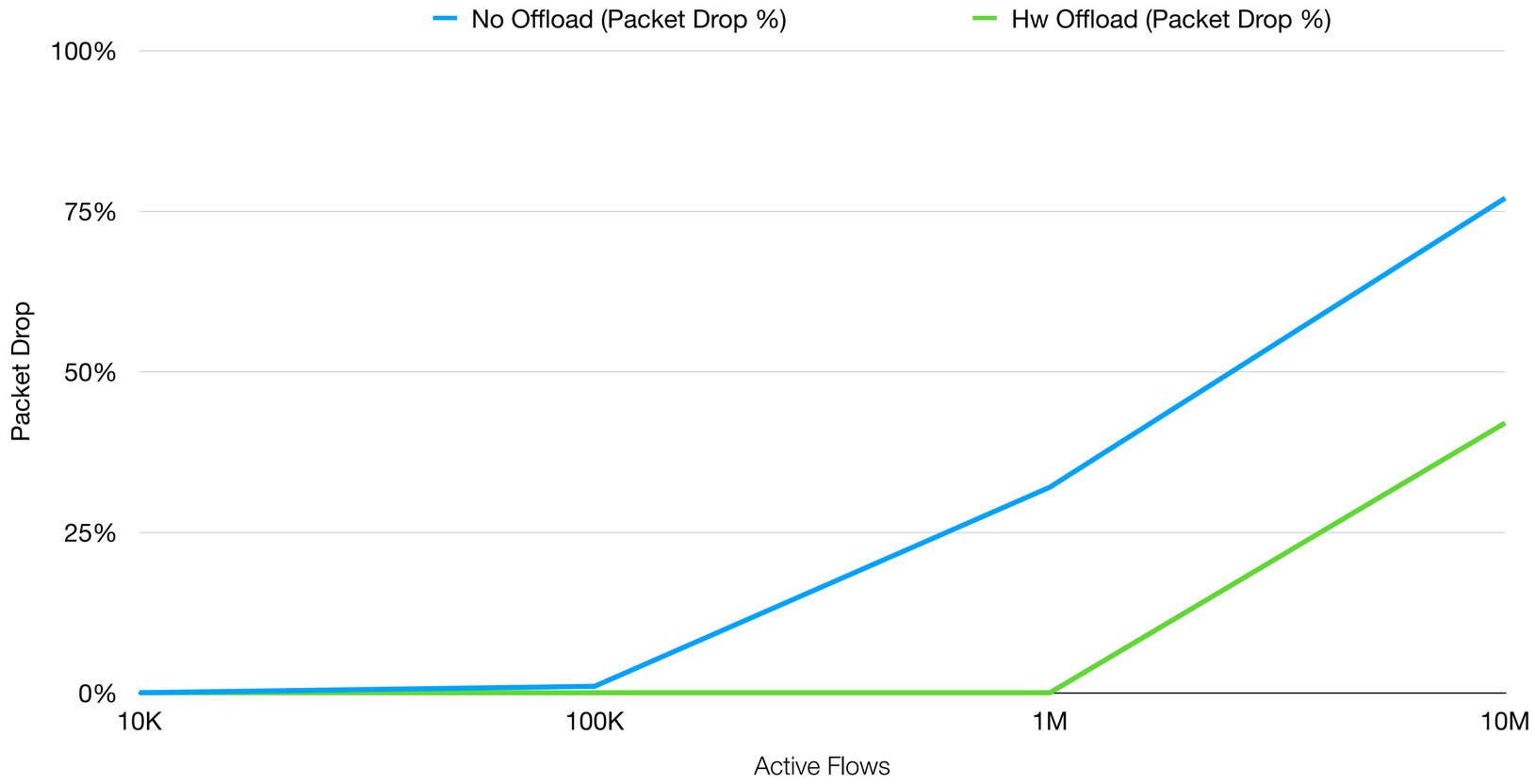
Inline Mode: CPU Load

- @ 89 Mpps 60 Gbps with DPI on 16 cores (RSS)



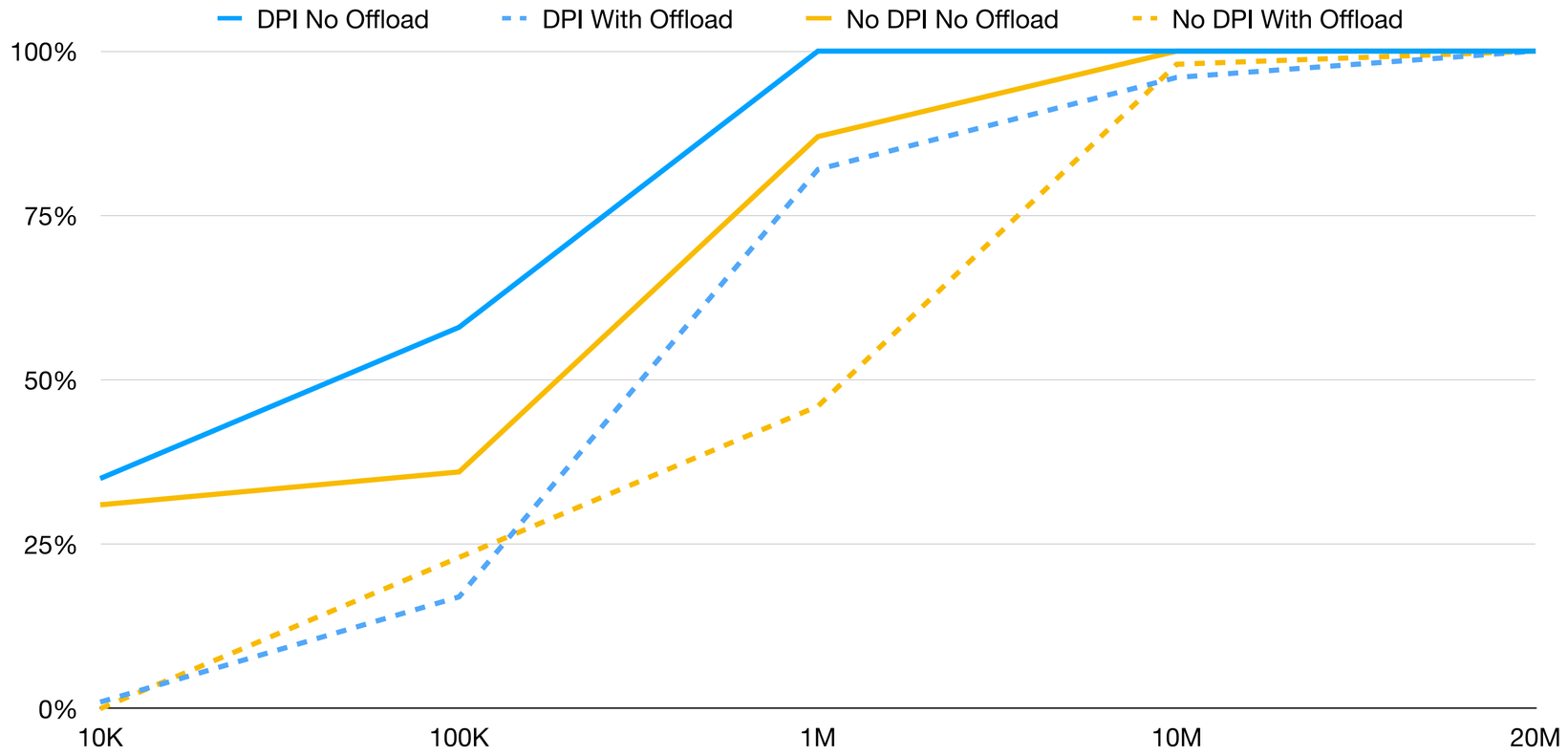
Inline Mode: Packet Drops

- @ 89 Mpps 60 Gbps with DPI on 16 cores (RSS)



nDPI: CPU Load Overhead

- Inline mode @ 10 Mpps 80 Gbps on 16 cores (RSS)



Conclusions

- Software-based monitoring systems are required to inspect and analyse complex traffic, especially when DPI is required.
- While software flexibility is undeniable, it should be combined with hardware offloads able to scale to hundreds of Gbits.
- Hardware flow tables implemented by modern SmartNICs offer the possibility to keep traffic statistics directly in hardware and also perform forwarding actions saving CPU cycles and bus bandwidth.
- Exploiting flow offloads, on well-designed software, requires limited changes while offering substantial performance benefits.

Future Work

- Improve the implementation of the host hash table that is a performance bottleneck as shown by our experiments: a pure hw flow table guarantees line rate but it is unable to feature DPI.

- Extend our work to other SmartNICs models in order to compare the features provided by different vendors.

- Take advantage of hardware flow tables to accelerate other applications (e.g. session validation in DDoS Mitigation solutions).

Artefacts and instructions for reproducing results:
https://github.com/ntop/flow-hw-offload-paper