Packets are Not Pages: Flow-Based Addressing Conserves Memory Bandwidth

Agur Adams, Hui Sub Shim, **Colin Drewes**,
Ben O'Keefe, Philip Levis and Zakir Durumeric

Stanford University

June 22, 2026

The SmartNIC Memory Bandwidth Bottleneck

	DRAM BW Per-Core	**NIC BW Per-Core**	**Ratio**
Google Cloud C3 2x Sapphire Rapids	3.49 GB/s	0.14 GB/s	**24.93**
BlueField-3 SmartNIC DDR5	5.60 GB/s	3.13 GB/s	**1.79**

Networking overheads are expensive to manage on-host[1]

SmartNICs offload but are limited in relative memory bandwidth

[1] Snap: A microkernel approach to host networking

Worsened by a Mismatch of Abstractions

Page-table Abstraction	**Reality**[2]
Pages are fixed size	
Benefits from spatial and temporal locality	
Applications operate on pages	

[2] Fast & Safe IO Memory Protection 2024 and Ensō 2023

Worsened by a Mismatch of Abstractions

Page-table Abstraction	**Reality**[2]
Pages are fixed size	Packets range in size
Benefits from spatial and temporal locality	
Applications operate on pages	

[2] Fast & Safe IO Memory Protection 2024 and Ensō 2023

Worsened by a Mismatch of Abstractions

Page-table Abstraction	**Reality**[2]
Pages are fixed size	Packets range in size
Benefits from spatial and temporal locality	Packets have low locality
Applications operate on pages	

[2] Fast & Safe IO Memory Protection 2024 and Ensō 2023

Worsened by a Mismatch of Abstractions

Page-table Abstraction	**Reality**[2]
Pages are fixed size	Packets range in size
Benefits from spatial and temporal locality	Packets have low locality
Applications operate on pages	Applications operate on flows

[2] Fast & Safe IO Memory Protection 2024 and Ensō 2023

Changing the Abstraction with **Flow-Based Addressing**

Flow-Based Abstraction	**Reality**
	Packets range in size
	Packets have low locality
	Applications operate on flows

Changing the Abstraction with **Flow-Based Addressing**

Flow-Based Abstraction	**Reality**
Interacts directly with the flow, not the packets	Packets range in size
	Packets have low locality
	Applications operate on flows

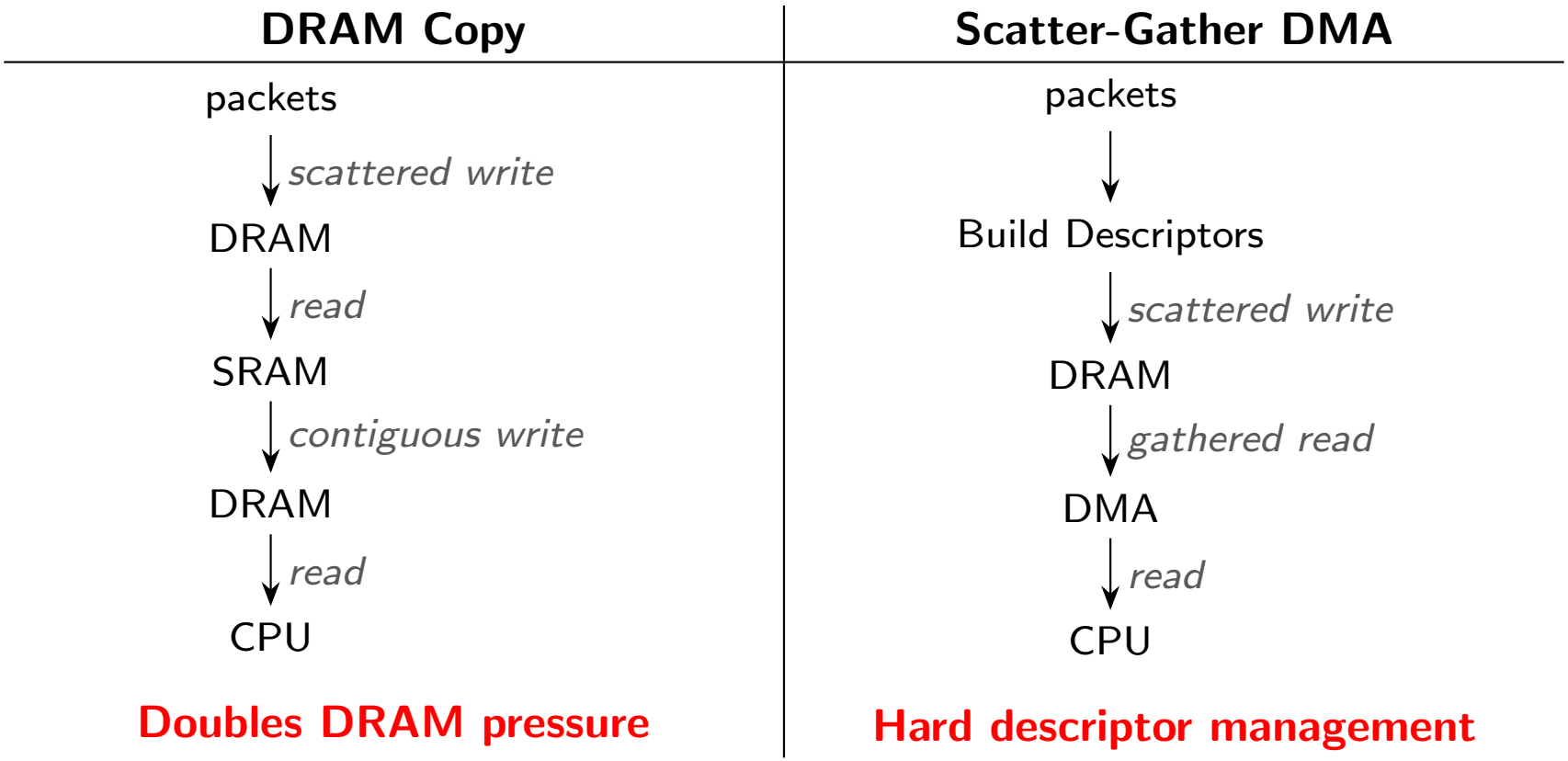
Changing the Abstraction with **Flow-Based Addressing**

Flow-Based Abstraction	**Reality**
Interacts directly with the flow, not the packets	Packets range in size
Flows appears as a contiguous memory regions	Packets have low locality
	Applications operate on flows

Changing the Abstraction with **Flow-Based Addressing**

Flow-Based Abstraction	**Reality**
Interacts directly with the flow, not the packets	Packets range in size
Flows appears as a contiguous memory regions	Packets have low locality
No impedance mismatch between application and memory interface	Applications operate on flows

Existing Approaches Miss the Mark



Design of Flow-Based Addressing

Goal: Make discontiguous packets in memory appear as contiguous

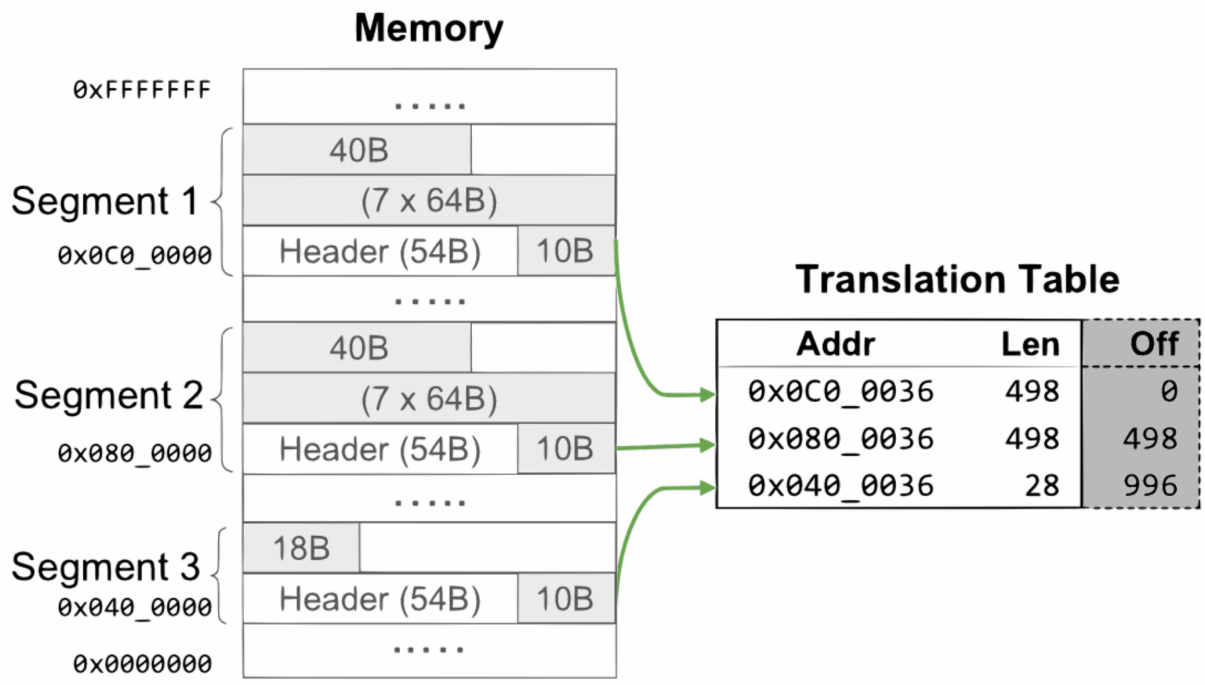


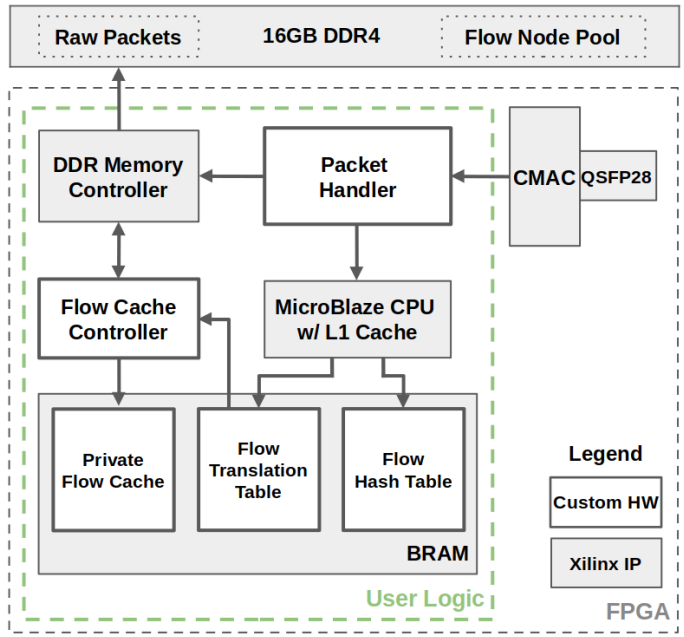
Table stores physical address, length, and offset into bytestream

Remaps CPU reads addresses into physical memory descriptors

Prototype Alveo U250 FPGA Smart-NIC: Packet Reception

Hardware Modules

Software Modules

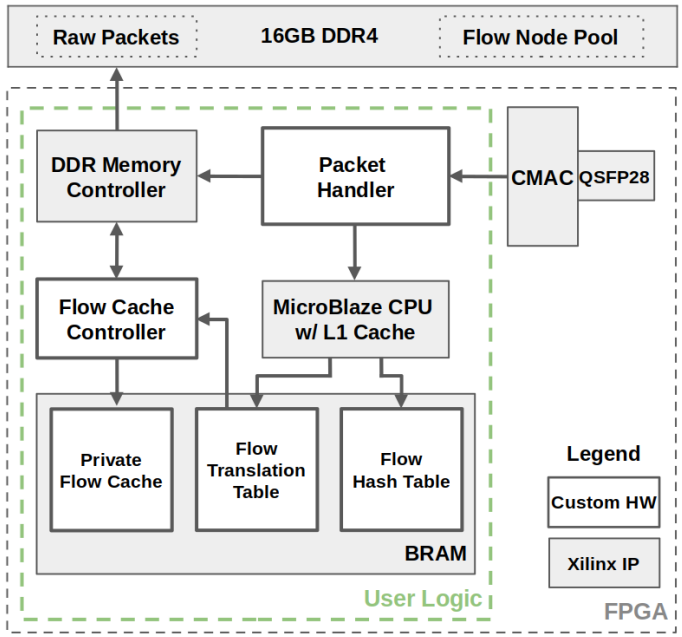


Implemented within AMD OpenNIC shell

Prototype Alveo U250 FPGA Smart-NIC: Packet Reception

Hardware Modules	
1) CMAC QSFP28	Packet reception

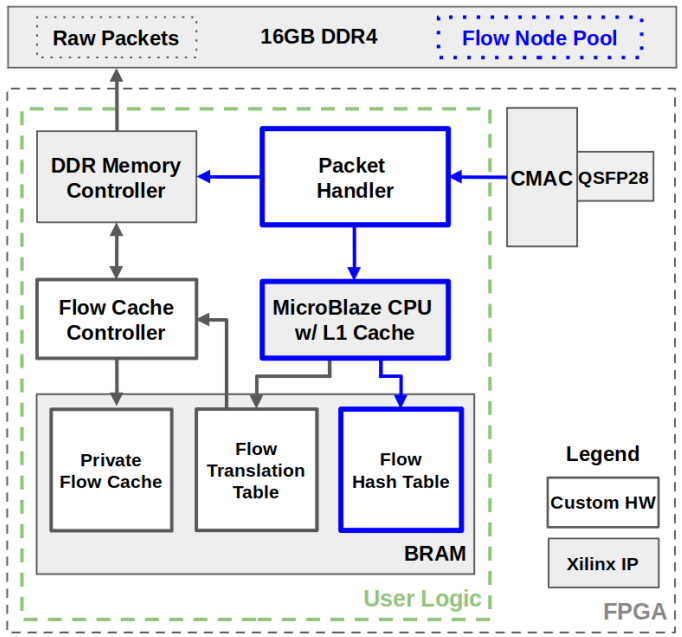
Software Modules	



Implemented within AMD OpenNIC shell

Prototype Alveo U250 FPGA Smart-NIC: Packet Reception

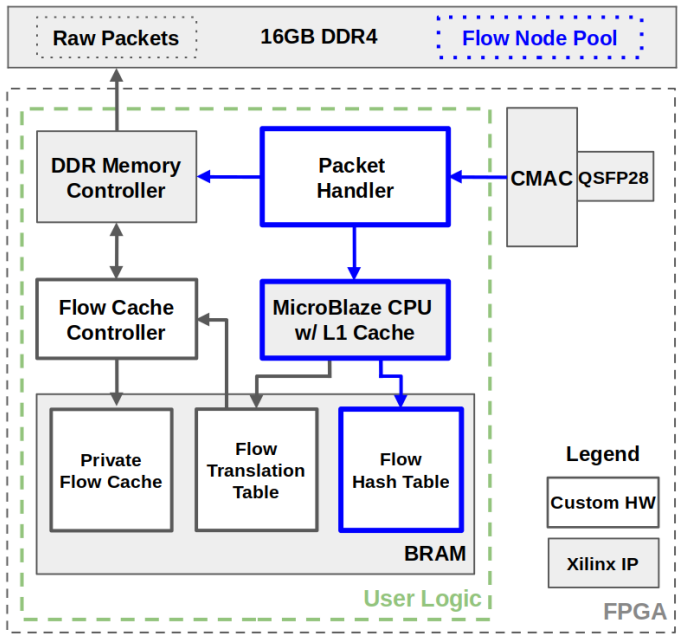
Hardware Modules	
1) CMAC QSFP28	Packet reception
2) Packet Handler	Metadata extraction
Software Modules	



Implemented within AMD OpenNIC shell

Prototype Alveo U250 FPGA Smart-NIC: Packet Reception

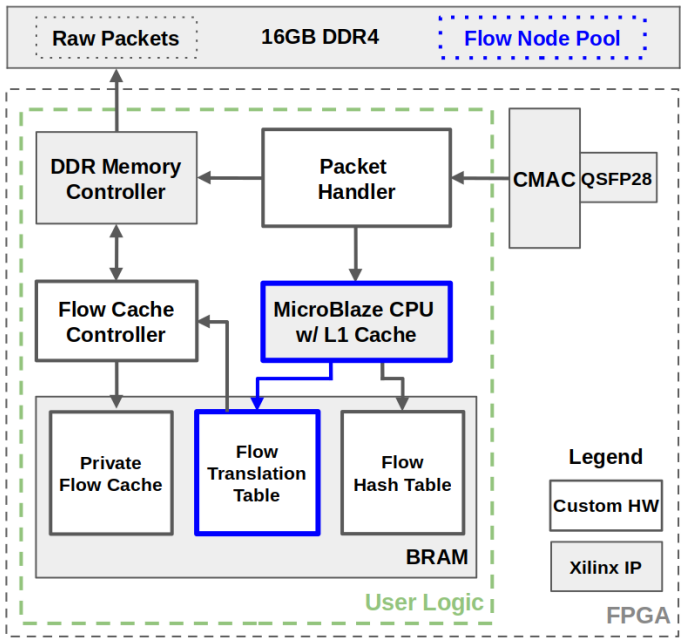
Hardware Modules	
1) CMAC QSFP28	Packet reception
2) Packet Handler	Metadata extraction
Software Modules	
3) Flow Hash Table	Packet mapped to flow



Implemented within AMD OpenNIC shell

Prototype Alveo U250 FPGA Smart-NIC: Packet Reception

Hardware Modules	
1) CMAC QSFP28	Packet reception
2) Packet Handler	Metadata extraction
Software Modules	
3) Flow Hash Table	Packet mapped to flow
4) Translation Table	Linked list insertion

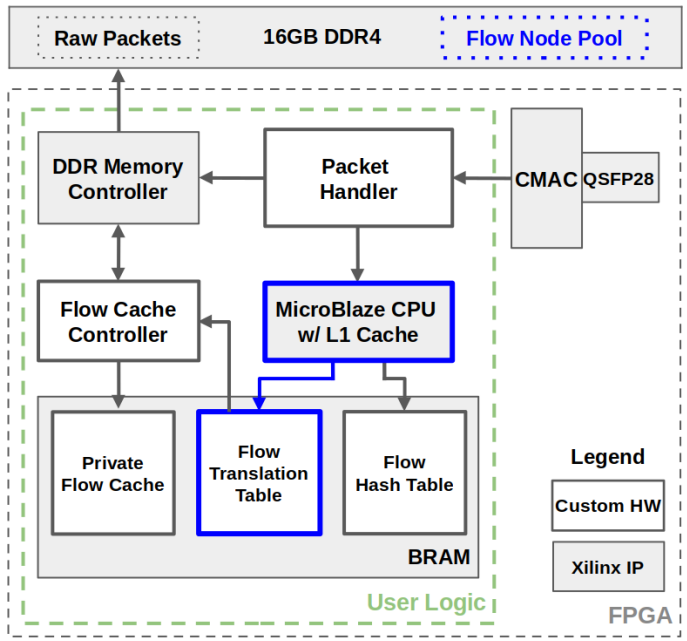


Implemented within AMD OpenNIC shell

Prototype Alveo U250 FPGA Smart-NIC: Processing

Hardware Modules

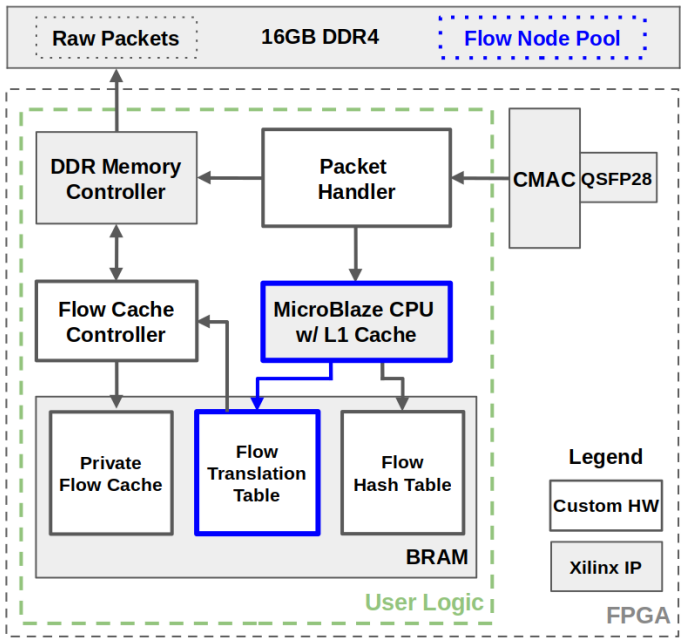
Software Modules



Implemented within AMD OpenNIC shell

Prototype Alveo U250 FPGA Smart-NIC: Processing

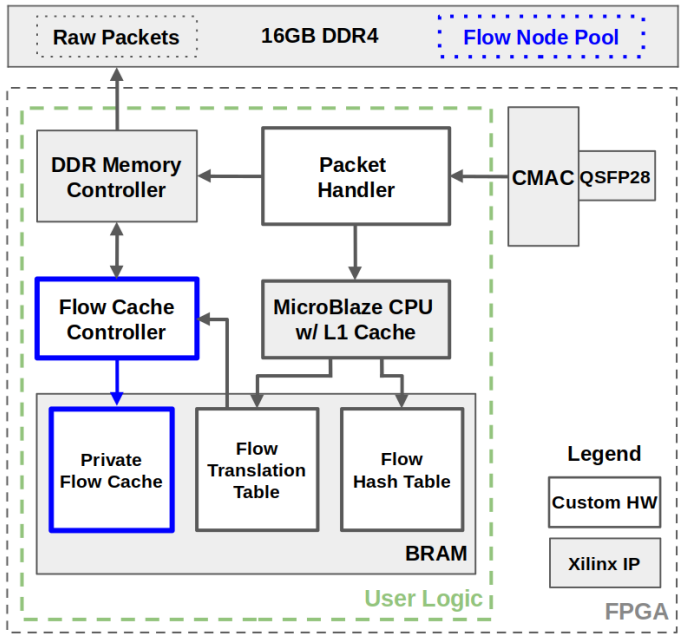
Hardware Modules	
Software Modules	
1) Translation Table	Linked list lookup



Implemented within AMD OpenNIC shell

Prototype Alveo U250 FPGA Smart-NIC: Processing

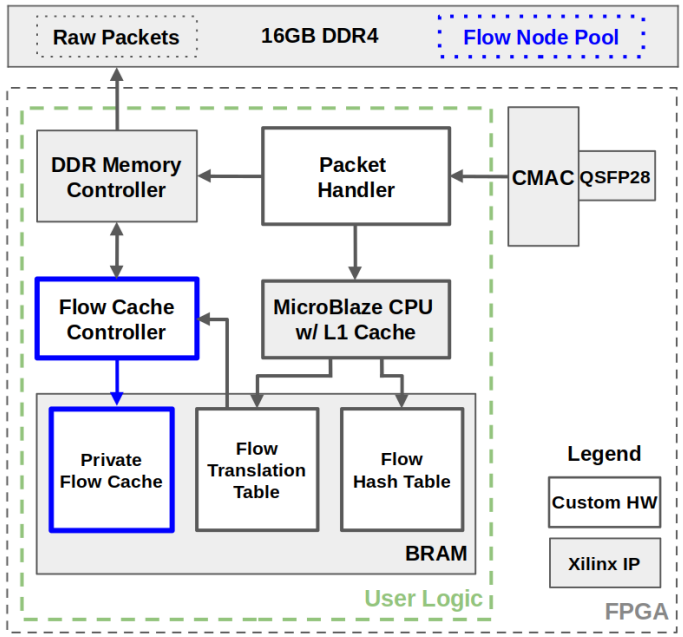
Hardware Modules	
2) Cache Controller	Fetches & Aligns
Software Modules	
1) Translation Table	Linked list lookup



Implemented within AMD OpenNIC shell

Prototype Alveo U250 FPGA Smart-NIC: Processing

Hardware Modules	
2) Cache Controller	Fetches & Aligns
Software Modules	
1) Translation Table	Linked list lookup
3) Application Code	Operate on flow-cache



Implemented within AMD OpenNIC shell

Motivating Application: On-NIC Intrusion Detection

Snort intrusion detection ruleset[3]

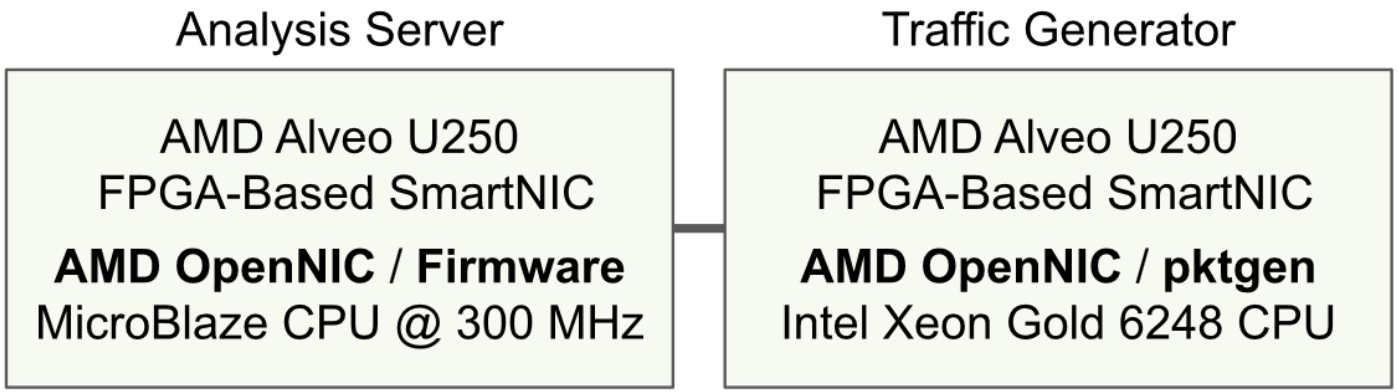
Byte	Occurrences	Symbol
00	4,167 (43.63%)	0x00 (null)
22	1,620 (16.96%)	" (double quote)
3B	714 (7.48%)	; (semicolon)
5C	707 (7.40%)	\ (backslash)
3A	367 (3.84%)	: (colon)

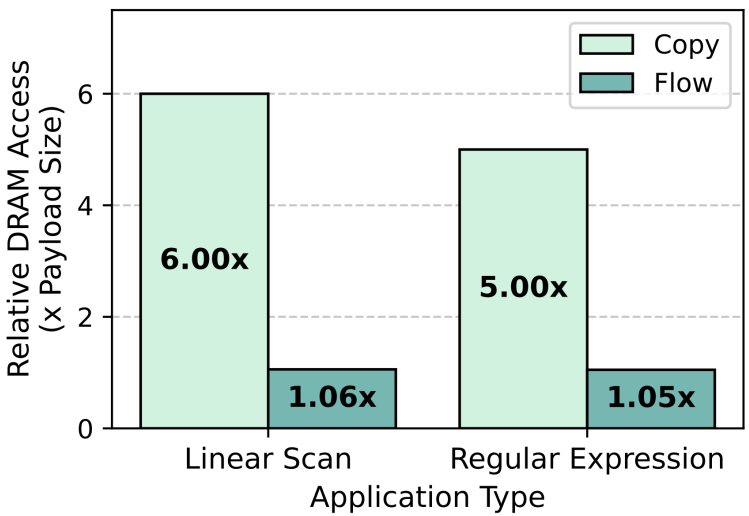
Linear scan searches for the NULL byte

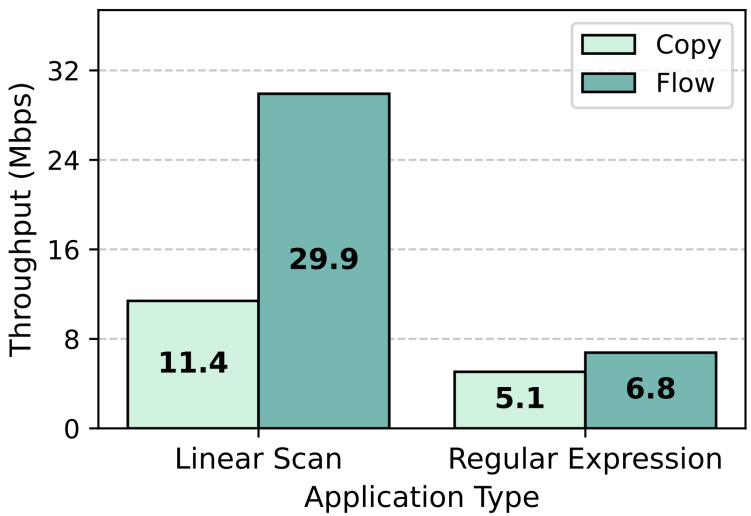
Regular expression match `[a-z][0-9]` pattern

[3] https://www.snort.org/

Flow-Based Addressing Under Synthetic Traces



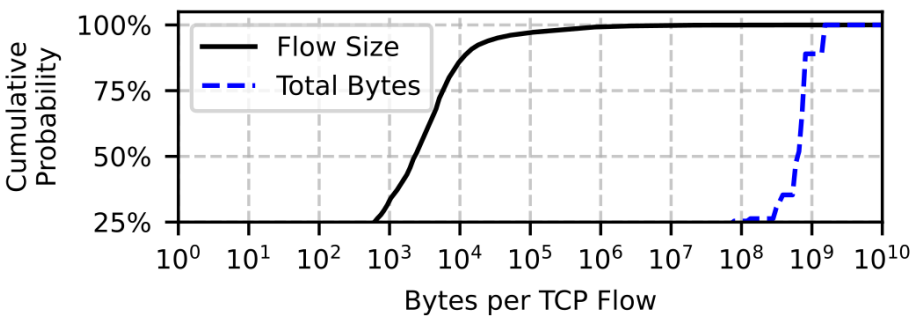




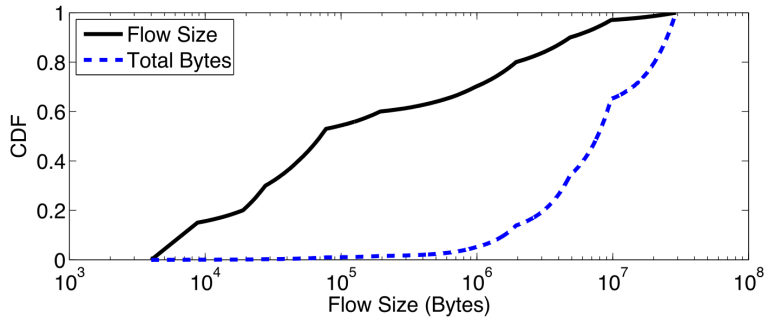
*Ten fixed-size 1,510-byte packets from a single flow

Real World Datasets

Stanford Campus Traffic



Datacenter Web Search



Metric	Stanford Campus	Datacenter
Packets written	14,638,353	18,607
Unique flows	100,211	10,000
Avg. packets per flow	146.1	1.9
Total PCAP size	15.73 GB	28.5 MB

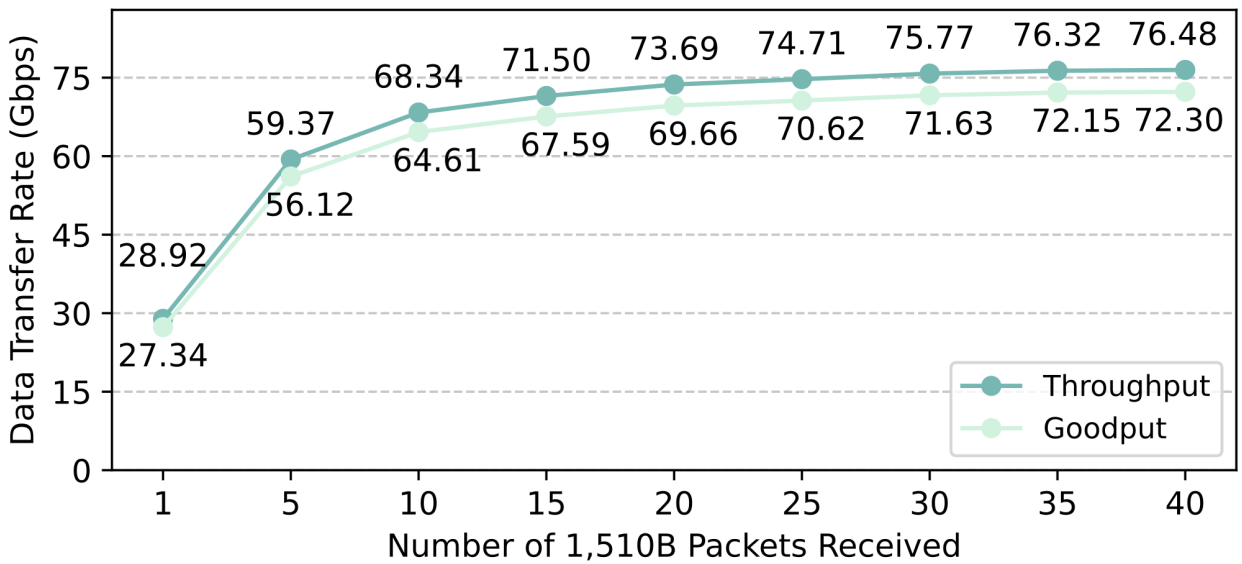
Addressing Under Real World-Traces

Workload	Execution (Reduction)	Throughput (Increase)	DRAM (Reduction)
Linear Scan			
Synthetic	62 %	2.62×	83 %
Campus	62 %	2.66×	82 %
Datacenter	62 %	2.63×	81 %
Regular Expression Search			
Synthetic	25 %	1.33×	79 %
Campus	26 %	1.34×	78 %
Datacenter	25 %	1.34×	77 %

*Versus copy-based approach

Scaling Flow-Based Addressing: Microbenchmark

BlueField-3 can sustain 32 Gbps per-core compute throughput



*Measured from CPU request to fully assembled bytestream

Scaling Flow-Based Addressing: Silicon Footprint

Component	Utilization	% of FPGA	% of MicroBlaze
MicroBlaze CPU and L1 Cache			
LUTs	24,935	1.44%	100%
Registers	45,669	1.32%	100%
Block RAM Tiles	44	1.64%	100%
Flow Cache and Controller			
LUTs	31,101	1.80%	125%
Registers	43,569	1.26%	95%
Block RAM Tiles	113	4.20%	257%

Conclusions

Page-tables are a poor fit for packet based workloads

- Mismatched in size, locality, and memory abstraction
- Flow-based addressing addresses these contentions

Flow-based addressing removes memory inefficiencies

- 80% reduction in DRAM utilization
- 1.3-2.5× compute throughput improvement

Designing around common cases can be vastly simpler

- Flows of single byte payloads are possible
- Most flows are short, instructing cache design

Call to re-examine memory abstractions in network devices

- Rather than accelerate an operation through a new offload, we removed an operation entirely