

UNIVERSIDADE FEDERAL DO PAMPA

Pedro da Silva Santiago

**From Synthetic to Realistic Network
Emulation: Benchmarking Real-world
Topologies on SmartNet**

Alegrete
2025

Pedro da Silva Santiago

From Synthetic to Realistic Network Emulation: Benchmarking Real-world Topologies on SmartNet

Projeto de Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade Federal do Pampa como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Marcelo Caggiani Luizelli

Coorientador: Prof. B.Sc. Victor Hugo Schneider Lopes

Alegrete
2025

Pedro da Silva Santiago

**FROM SYNTHETIC TO REALISTIC NETWORK EMULATION: BENCHMARKING REAL-
WORLD TOPOLOGIES ON SMARTNET**

Trabalho de Conclusão de Curso
apresentado ao Curso de Graduação
em Ciência da Computação da
Universidade Federal do Pampa como
requisito parcial para a obtenção do
título de Bacharel em Ciência da
Computação

Monografia defendida e aprovada em: 12 de dezembro de 2025.

Banca examinadora:

Marcelo Caggiani Luizelli
Orientador
Unipampa

Victor Hugo Schneider Lopes
Coorientador
Unipampa

Francisco Vogt
Unicamp

Ariel Goes de Castro
Unicamp



Assinado eletronicamente por **MARCELO CAGGIANI LUIZELLI, PROFESSOR DO MAGISTERIO SUPERIOR**, em 16/12/2025, às 22:23, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.



Assinado eletronicamente por **VICTOR HUGO SCHNEIDER LOPES, Aluno**, em 16/12/2025, às 23:03, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.



Assinado eletronicamente por **Francisco Germano Vogt, Usuário Externo**, em 17/12/2025, às 04:31, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.



Assinado eletronicamente por **Ariel Góes de Castro, Usuário Externo**, em 17/12/2025, às 09:50, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.



A autenticidade deste documento pode ser conferida no site https://sei.unipampa.edu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **1919523** e o código CRC **D6F018D3**.

RESUMO

A emulação de redes é essencial para validar o desempenho e a confiabilidade das modernas infraestruturas de comunicação. Enquanto ferramentas baseadas em software oferecem flexibilidade e soluções baseadas em switches fornecem velocidade, elas frequentemente falham em entregar um ambiente equilibrado que combine alta fidelidade com modelagem realista de recursos. O SmartNet aborda isso aproveitando Smart Network Interface Cards (SmartNICs) para emular topologias de rede diretamente no hardware. No entanto, sua escalabilidade em cenários realistas e complexos permaneceu amplamente inexplorada. Este estudo valida a eficácia do SmartNet implantando estruturas de rede heterogêneas derivadas do Internet Topology Zoo na plataforma NVIDIA BlueField-2. Através de experimentação rigorosa medindo vazão, latência e jitter, o estudo quantifica os limites de desempenho do framework. Os resultados demonstram que, embora o SmartNet entregue emulação de alta fidelidade com temporização determinística para topologias de pequeno e médio porte, ele enfrenta restrições de escalabilidade impostas pelos recursos de hardware específicos da BlueField-2. Este trabalho visa validar o SmartNet como uma ferramenta viável para experimentação de alta precisão dentro de seus limites operacionais, estabelecendo uma base clara para avanços futuros na emulação de rede acelerada por hardware.

Palavras-chave: Smartnet. Emulação de Redes. BlueField-2. Offloading por Hardware.

ABSTRACT

Network emulation is essential for validating the performance and reliability of modern communication infrastructures. While software-based tools offer flexibility and switch-based solutions provide speed, they often fail to deliver a balanced environment that combines high fidelity with realistic resource modeling. SmartNet addresses this by leveraging Smart Network Interface Cards (SmartNICs) to emulate network topologies directly on hardware. However, its scalability in realistic, complex scenarios remained largely unexplored. This work validates the efficacy of SmartNet by deploying heterogeneous network structures derived from the Internet Topology Zoo on the NVIDIA BlueField-2 platform. Through rigorous experimentation measuring throughput, latency, and jitter, the study quantifies the framework’s performance boundaries. The results demonstrate that while SmartNet delivers high-fidelity emulation with deterministic timing for small to medium-sized topologies, it faces scalability constraints imposed by the specific hardware resources of the BlueField-2, such as ARM core capacity and concurrent application limits. This work aims to validate SmartNet as a viable tool for high-precision experimentation within its operational bounds, establishing a clear baseline for future advancements in hardware-accelerated network emulation.

Key-words: Smartnet. Network Emulation. BlueField-2. Hardware Offloading.

LIST OF FIGURES

Figure 1 – Components and connections in a two-host network created with Mininet.	22
Figure 2 – BNV system overview.	23
Figure 3 – P7 concept and P4 pipeline representation.	24
Figure 4 – Difference between deployment with a traditional Network Interface Card (NIC) (a) and a SmartNIC (b).	25
Figure 5 – Overview of the different implementation options and their trade-offs between flexibility and efficiency.	26
Figure 6 – On-path and off-path SmartNICs.	27
Figure 7 – SmartNet Architecture	29
Figure 8 – SmartNet throughput across different network topologies.	37
Figure 9 – SmartNet packet per second throughput across different network topologies.	38
Figure 10 – SmartNet latency across different network topologies.	38

LIST OF TABLES

Table 1 – Performance Metrics	31
Table 2 – Bluefield-2 Host Configuration	33
Table 3 – TREX Host Configuration	33
Table 4 – BlueField-2 SmartNIC Configuration	33
Table 5 – Topology selection process and remaining number of topologies at each stage.	35
Table 6 – Topology characteristics used in the study	36

LIST OF ACRONYMS

ASIC Application Specific Integrated Circuit.

CPU Central Processing Unit.

DPDK Data Plane Development Kit.

DPU Data Processing Unit.

FPGA Field-Programmable Gate Array.

Gbps Gigabits per second.

ISP Internet Service Provider.

Mpps Million packets per second.

NIC Network Interface Card.

OVS Open vSwitch.

P4 Programming Protocol-independent Packet Processors.

P7 P4 Programmable Patch Panel.

pps packet per second.

RAM Random Access Memory.

SDN Software-Defined Networking.

SF Scalable Function.

SmartNIC Smart Network Interface Card.

SoC System on a Chip.

CONTENTS

1	INTRODUCTION	17
1.1	Context and motivation	17
1.2	Problem	18
1.3	Objectives	18
1.4	Outline	19
2	BACKGROUND	21
2.1	Network emulation	21
2.2	SmartNICs	24
2.2.1	On/Off-Path Designs	26
2.2.2	ASIC-based SmartNICs	27
2.2.3	FPGA-based SmartNICs	28
2.2.4	SoC-based SmartNICs	28
2.3	SmartNet	29
3	METHODOLOGY	31
3.1	Metrics	31
3.2	Topology Selection	32
3.3	Testbeds	32
4	RESULTS AND DISCUSSION	35
4.1	Scalability of Topology Emulation	35
4.2	Throughput Evaluation	36
4.3	Latency Evaluation	37
4.4	Discussion	39
5	FINAL REMARKS	41
5.1	Overview	41
5.2	Challenges and Limitations	42
5.3	Future Work	42
	REFERENCES	45

1 INTRODUCTION

This chapter introduces the context and motivation for this work, outlines the problem being addressed, states the objectives of this work, and presents the outline of the document.

1.1 Context and motivation

Modern communication networks are increasing in complexity and handle growing volumes of data traffic. The widespread adoption of high-speed technologies such as 5G, and the ongoing development of 6G, have continue to push the boundaries of network capacity, latency, and reliability. As these systems evolve to support diverse services and applications, even minor failures in their underlying protocols, architectures, or control mechanisms can cause significant disruptions. Such incidents can compromise the performance and availability of critical services, potentially affecting millions of users worldwide (ULBRICHT et al., 2021).

Network emulation plays a crucial role in this landscape by providing a controllable and repeatable environment in which real protocols, implementations, and applications can be tested under various network conditions. By accurately reproducing impairments such as delay, jitter, loss, and bandwidth constraints, emulation enables developers and operators to stress-test new designs, reveal corner cases, and identify performance bottlenecks before deployment in production networks. This capability is particularly valuable for enhancing the security and robustness of emerging 5G and 6G services, since it allows the safe experimentation of misconfigurations and attack scenarios, supports the evaluation of mitigation strategies, and ultimately reduces the risk that failures in protocol stacks, architectures, or applications will propagate to large user populations (VOGT et al., 2025b).

Although established emulation platforms such as Mininet (LANTZ; HELLER; MCKEOWN, 2010), NSX (KOPONEN et al., 2014) and BNV (KANNAN et al., 2018) are widely adopted, they have very limited because they are software based or hybrid and run on general-purpose Central Processing Units (CPUs), meaning they are predominantly designed for functional verification rather than high-fidelity performance (PETERSEN et al., 2022). Consequently, while these tools are sufficient for early-stage prototyping or software-defined contexts, they often lack the architectural capacity to emulate network conditions at a realistic scale, particularly when subjecting the infrastructure to high-speed throughput requirements (YAN; JIN, 2015). P7 (RODRIGUEZ et al., 2023) has recently proposed a hardware-accelerated network emulator based on Tofino switches that can achieve high performance, but its reliance on specialized hardware limits its accessibility and flexibility for broader research applications.

To address these limitations, SmartNet (VOGT et al., 2025b) has emerged as a hardware-based emulation framework designed to bridge the gap between performance

and realism. Built on NVIDIA BlueField-2 SmartNICs, SmartNet leverages hardware-offloaded Open vSwitch (OVS) and Scalable Functions (SFs) to minimize CPU overhead, theoretically supporting throughputs up to 100 Gigabits per second (Gbps). Although the potential of this architecture has been demonstrated in the context of Network Digital Twins with GhostTwin (VOGT et al., 2025a) and abstract network scenarios, SmartNet performance hasn't been tested in more realistic network topologies, with complex structures and diverse traffic patterns.

1.2 Problem

The validity of network emulation relies heavily on the structural fidelity of the test environment. While synthetic topologies, such as simple linear chains or basic graphs, are sufficient for functional verification, they fail to capture the graph-theoretic complexity of production networks. Real-world infrastructures, such as those cataloged in the Internet Topology Zoo (KNIGHT et al., 2011)¹, exhibit heterogeneous node degree distributions, varied path diversities, and complex bottlenecks that directly influence protocol behavior and traffic dynamics.

In the specific context of SmartNet, a validation gap remains regarding its architectural scalability. Although the framework utilizes hardware-offloaded OVS and SFs to theoretically support throughputs up to 100 Gbps, its performance in complex graph structures remains unverified. SmartNet relies on a finite set of ARM cores to manage resource isolation and host emulation, which inherently constrains processing power (VOGT et al., 2025b). Consequently, as topology complexity increases—requiring more emulated devices and intricate interconnects—there is a non-trivial risk of resource contention that could invalidate the high-speed performance claims observed in simpler scenarios. Therefore, while SmartNet demonstrates potential in abstract setups, it has not yet been extensively validated across a broad spectrum of realistic operational topologies where the interplay between node density and hardware constraints is most critical.

1.3 Objectives

The primary objective of this work is to validate the efficacy and scalability of SmartNet when emulating realistic network infrastructures. Specifically, this work aims to:

- **Validate the functional deployment of realistic topologies:** Assess SmartNet's capability to faithfully instantiate heterogeneous network structures derived from the Internet Topology Zoo. This involves verifying that the emulator can

¹ Available in: <https://topology-zoo.org>

correctly translate complex graph definitions into hardware-offloaded artifacts (SFs and OVS bridges) without compromising structural integrity or connectivity.

- **Quantify emulation fidelity under complex structural loads:** Evaluate the performance of the emulated environment by measuring network metrics, such as throughput, latency and packet loss, across topologies with varying degrees of complexity. The goal is to determine if the hardware-offloaded dataplane maintains precision in link impairment reproduction when subjected to diverse traffic patterns typical of production networks.
- **Characterize hardware resource scalability limits:** Analyze the operational boundaries of the NVIDIA BlueField-2 SmartNIC by identifying the maximum density of emulated nodes and links that can be sustained while maintaining line-rate processing. This includes investigating the correlation between topology size and the consumption of finite on-board resources, such as ARM cores and eSwitch capacity, to define the upper bounds of the current hardware generation.

1.4 Outline

The remainder of this work is organized as follows: Chapter 2 reviews related works in the area of network emulation (Section 2.1) and provides a comprehensive overview of SmartNIC architectures (Section 2.2), concluding with a detailed description of the SmartNet framework (Section 2.3). Chapter 3 presents the methodology used in this work, detailing the performance metrics (Section 3.1), the selection and path-generation process for realistic topologies from the Internet Topology Zoo (Section 3.2), and the experimental testbed configuration (Section 3.3). Chapter 4 discusses the experimental findings regarding the scalability and fidelity of the emulation. Finally, Chapter 5 summarizes the main conclusions, addresses the limitations related to hardware constraints, and outlines directions for future research.

2 BACKGROUND

This chapter presents an overview of the background and related works relevant to the research presented in this work. The subsequent sections will explore network emulation techniques and their architectures (Section 2.1), the architecture SmartNICs (Section 2.2), and finally, the Smartnet tool (Section 2.3).

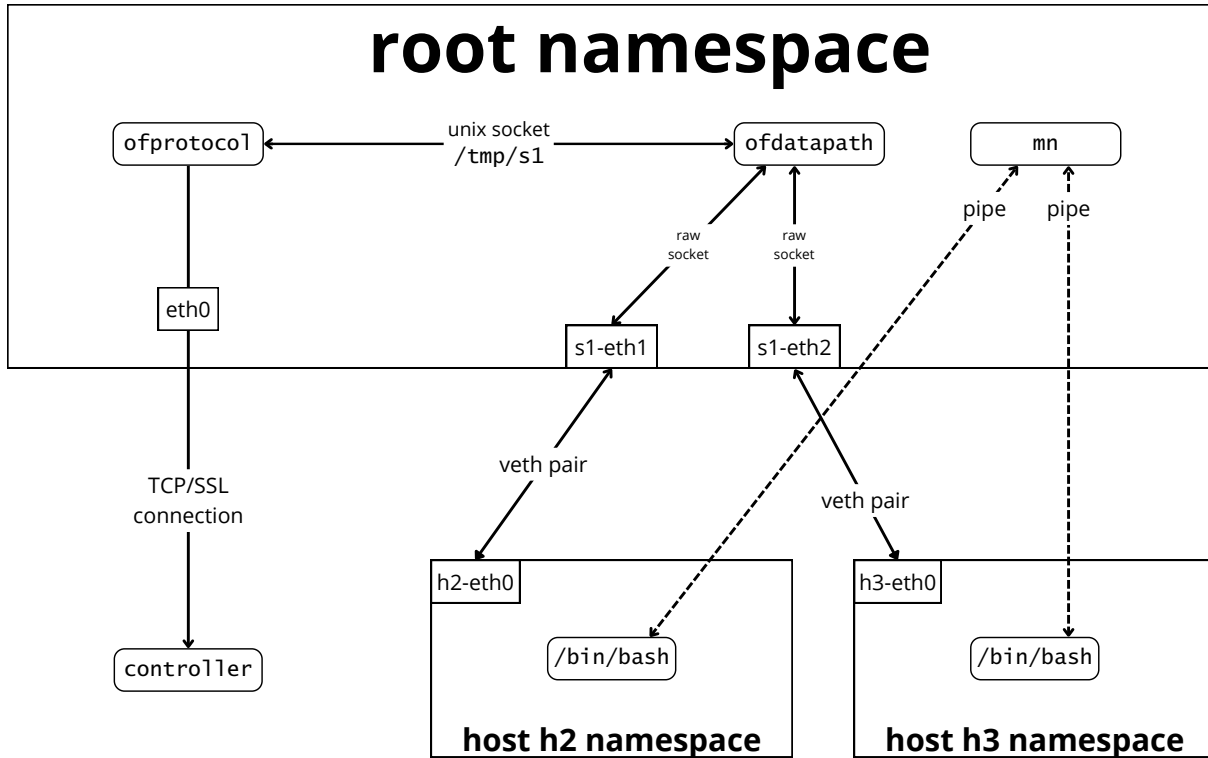
2.1 Network emulation

The complexity of modern network environments is escalating rapidly within both industrial and academic sectors, driven largely by advancements in network programmability. However, validating these programmable architectures requires testbeds that are not only accessible and cost-effective but also capable of high-fidelity, line-rate performance (RODRIGUEZ et al., 2022). Currently, researchers are often constrained by significant resource limitations, preventing the acquisition of high-end networking equipment. Consequently, experimental evaluations are frequently confined to small-scale setups, limited in throughput and topology size, or rely on software-based emulation and simulation tools. This scenario forces a difficult compromise in network experimentation, necessitating trade-offs that undermine realism, flexibility, scalability, and customizability (RODRIGUEZ et al., 2023).

A noteworthy contribution towards network emulation is Mininet (LANTZ; HELLER; MCKEOWN, 2010), widely used in both research and industrial environments. Using Operating System-level virtualization for emulation purposes, Mininet enables users to instantiate a virtual network consisting of hosts, switches, and links on a single host system, enabling them to simulate complex topologies and verify different network applications and protocols. However, Mininet’s abilities to emulate networks are limited by the characteristics of the host machine’s network interface, and its emulation precision can also be impacted by the host computer’s CPU, memory, and the number of emulated switches and host computers, which considerably limits its performance when emulating extensive networks or particular network situations (RODRIGUEZ et al., 2023). Figure 1 illustrates the components and connections in a two-host network created with Mininet. In such a setup, Mininet places host processes in network namespaces and connects them with virtual Ethernet pairs, which then typically link to a userspace OpenFlow switch.

Current network testing and evaluation solutions are largely software-based and, although they are convenient and flexible in their approach, they often struggle to deliver high-fidelity experiments for real-world scenarios due to inherent performance bottlenecks and scalability limitations that restrict their usage in high-performance or high-complexity environments (RODRIGUEZ et al., 2023). To overcome such limitations, new advances have proposed more powerful mechanisms based on hardware-assisted technologies. One such example is BNV (KANNAN et al., 2018), which takes advantage of modern hardware virtualization features, including Intel VT-d and Single-root input/output virtualization,

Figure 1 – Components and connections in a two-host network created with Mininet.

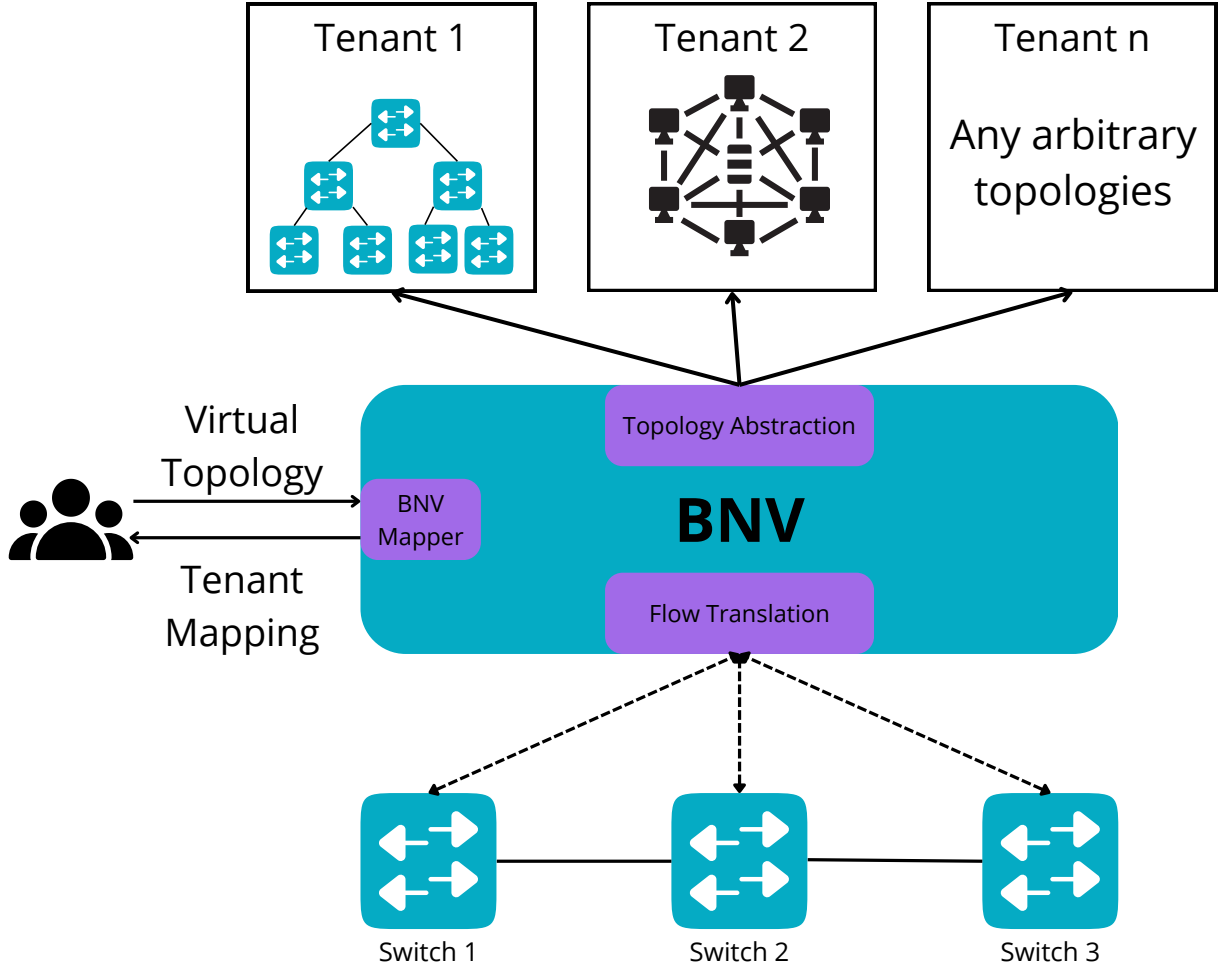


Source: Adapted from (LANTZ; HELLER; MCKEOWN, 2010)

to facilitate the virtualization of network devices. This includes elements such as OVS bridges to provide more realistic and performance-oriented testing environments representative of actual deployment environments. The architecture of BNV is illustrated in Figure 2. BNV directly interfaces with Software-Defined Networking (SDN) switches via OpenFlow, enabling the creation and management of virtual network topologies. The core process begins when users submit their desired virtual topologies to the BNV Mapper. This component is responsible for embedding the requested resources in the physical infrastructure and subsequently provisioning the network for the tenant using the reserved physical resources. A key capability of BNV is its support for multiple tenants and various SDN controllers. This design ensures that each network slice can be controlled independently, preventing interference among tenants co-located on the same physical switch.

NSX (KOPONEN et al., 2014) is VMware’s network virtualization platform that creates software-defined networks on top of physical infrastructure to manage multi-tenant domains in data center environments. It decouples networking from hardware, letting users programmatically create, configure, and manage virtual networks and security policies through software rather than manually configuring physical switches and routers.

Figure 2 – BNV system overview.



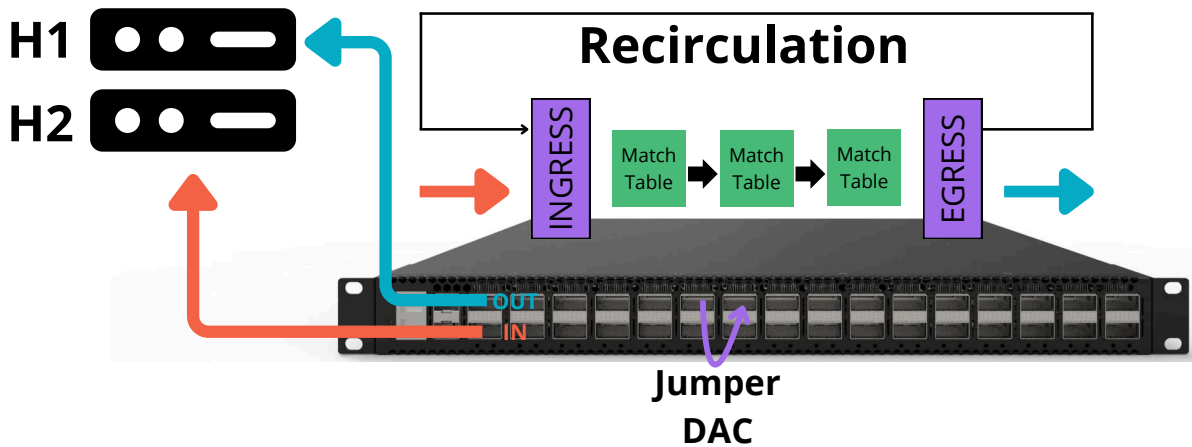
Source: Adapted from (KANNAN et al., 2018)

One of NSX's key features is that it provides a distributed virtualization layer that allows the creation of virtual networks across multiple physical hosts, using Virtual Machines. In addition, NSX provides a network services platform with firewall, load balancing, and Virtual Private Network connectivity services.

The emergence of programmable networks has enabled emulation frameworks to exploit Programming Protocol-independent Packet Processors (P4)-capable switches for high-fidelity network behavior replication. An example of such a framework is P4 Programmable Patch Panel (P7). P7 (RODRIGUEZ et al., 2023) is a network emulation tool that aims to overcome the shortcomings of traditional testbed approaches, such as those mentioned previously. P7 leverages P4-capable hardware to offer realistic network emulation. It achieves this by utilizing programmable hardware pipeline features like packet recirculations, port configurations, and match+action table abstractions, alongside a straightforward path routing solution. P7 also provides crucial flexibility for users and ex-

perimenters: you can connect physical servers to inject custom traffic flows (from traffic or trace generators) into the emulated network scenario, as Figure 3 illustrates. Compared to other tools, P7 can emulate topologies at or near hardware line rate (e.g., 100 Gbit/s per port), offering realistic queuing and timing properties that are difficult to reproduce with software-only emulation. However, this reliance on specific P4-capable switches inherently limits portability and creates a strictly switch-centric design, neglecting other critical components of the programmable network ecosystem, such as SmartNICs.

Figure 3 – P7 concept and P4 pipeline representation.



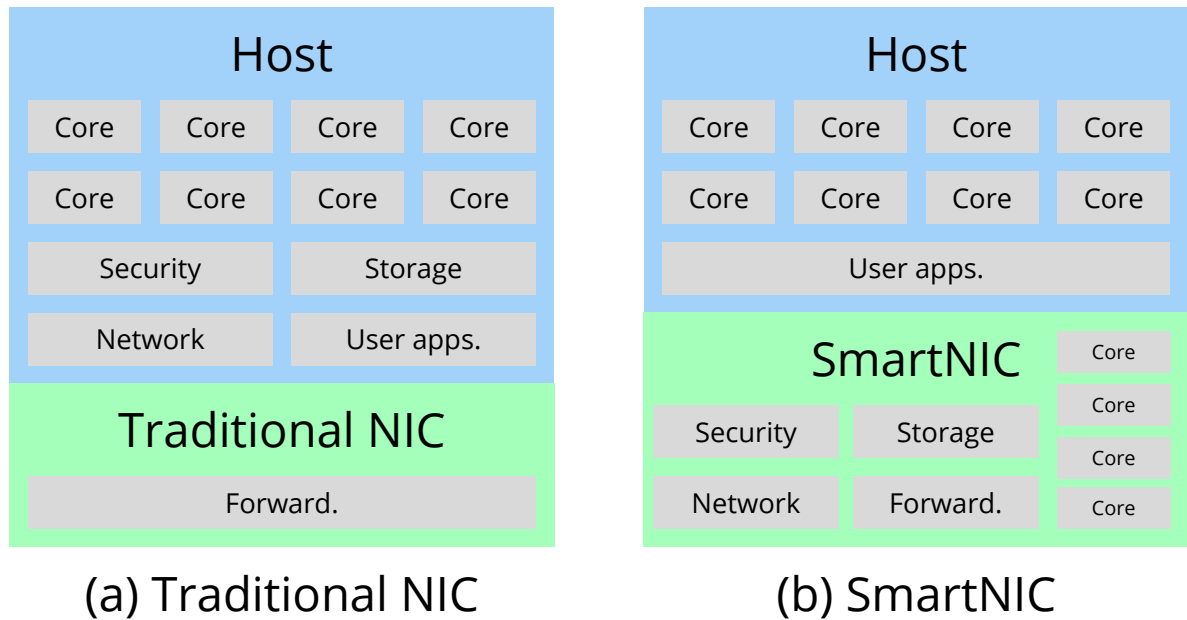
Source: Adapted from (RODRIGUEZ et al., 2022)

2.2 SmartNICs

The concept of SmartNIC was first discussed during the 1990s, when computer NICs were solely responsible for linking computers to the network (PONOMAREV; GHOSE, 1998). They were responsible for simple functions at the time, including sending and receiving packets and offloading simple operations such as checksum. NIC offload has developed immensely since then to accommodate the increasing demands of today's networks, owing to the demand for performance, efficiency, and scalability. During the 2000s, the NICs started to implement Transmission Control Protocol Offload Engines (MACCABE et al., 2002), and by the 2010s, they had further advanced to accommodate features of virtualization such as Virtual Local Area Network tagging and Receive Side Scaling (DONG et al., 2011; LUIZELLI et al., 2024). There are certain NIC devices that are not classified as smart but help the CPU by contributing a wide range of network function acceleration, such as network virtualization. They are hardwired in such functions. They are said to be the stepping stone which gave rise to the SmartNIC (DORING; STUBBE; HOLZINGER, 2021).

Luizelli et al. (LUIZELLI et al., 2024) states that even though NICs have been evolving over time, they were only regarded as SmartNICs when vendors introduce a level of programmability. The programmability level differs: Some SmartNICs support total customization of the hardware logic, whereas others support offload of certain networking operations to separate computing units. To accommodate the flexibility, SmartNICs are developed based on various hardware platforms and employ a variety of programmable languages, including P4, Micro-C, and VHDL/Verilog. Even with all these capabilities, it is a demanding and complicated task to program, debug, and manage SmartNICs. The hardware architecture necessary for achieving high-speed network packet processing requires a high degree of parallelism to achieve performance scalability in NIC programs. To address this, current programmable NICs rely on multiple hardware architectures, including (i) Application Specific Integrated Circuit (ASIC); (ii) System on a Chip (SoC); and (iii) Field-Programmable Gate Array (FPGA). As seen in Figure 4, in a traditional NIC deployment (a), the host CPU cores implement both infrastructure functions as well as user applications. However, with SmartNICs (b), the host CPU cores are reserved for running user applications, with the SmartNIC’s CPU cores and other accelerators performing the infrastructure functions (KFOURY et al., 2024).

Figure 4 – Difference between deployment with a traditional NIC (a) and a SmartNIC (b).



Source: Adapted from (KFOURY et al., 2024)

In the next sections, the different SmartNIC architectures currently available on the market and their characteristics will be explored. Starting with general architecture designs: on-path and off-path. Then a deep dive into FPGA-based, ASIC-based, and SoC-

based SmartNICs. Figure 5 highlights various approaches to accelerate SDN performance and illustrates the trade-off between hardware flexibility and efficiency. Efficiency in this context means the ability to offload and execute SDN operations using less power (watts per operation) while maximizing throughput (number of packets or operations processed per second).

Figure 5 – Overview of the different implementation options and their trade-offs between flexibility and efficiency.



Source: Adapted from (FIRESTONE et al., 2018)

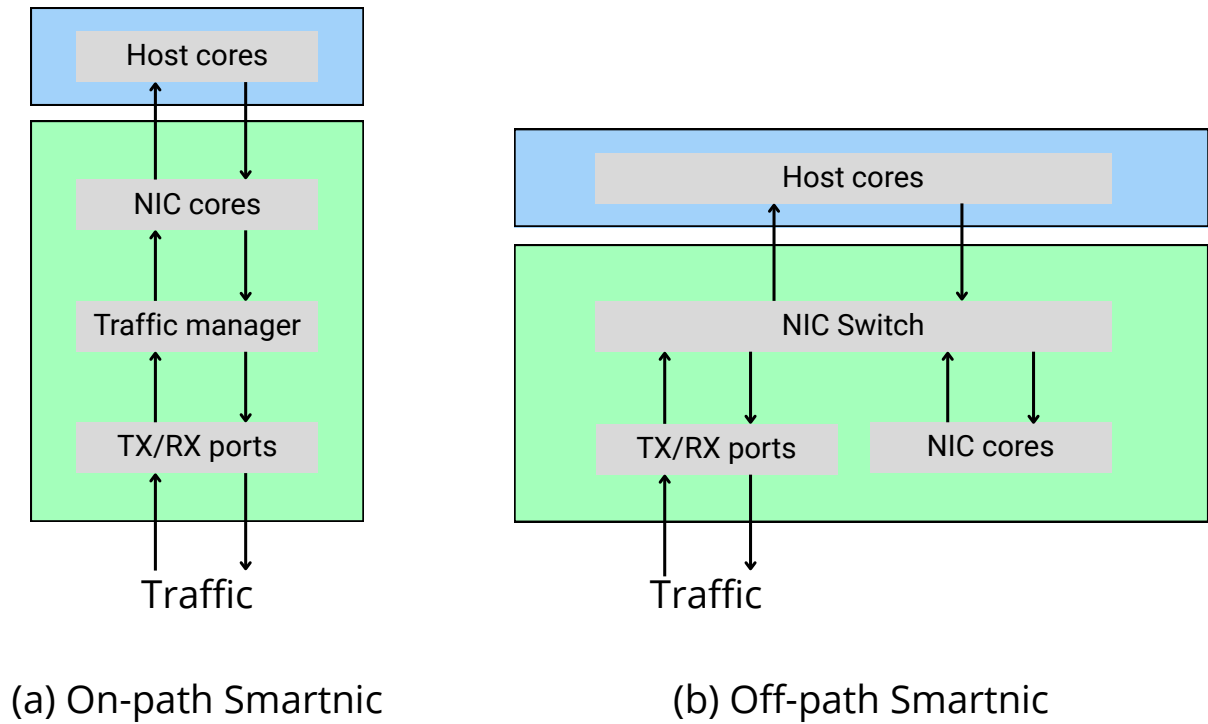
2.2.1 On/Off-Path Designs

SmartNICs can be categorized into two main designs: on-path and off-path (WEI et al., 2023). On-path SmartNICs make the NIC cores visible to the system with low-level programmable interfaces. The offloading code is on the critical path of the packet processing pipeline in this architecture. The benefit is that the offloaded code is closer to the network, making inline requests that interact only with the NIC, such as writing to the on-board memory, extremely efficient (LIU et al., 2019). The limitation is that the offloaded code rivals the NIC resources. Thus, if too much computation is offloaded, the NIC could degrade in performance (LIU et al., 2019). The NIC's cores also have direct access to memory subsystems such as Dynamic Random Access Memory or caching. Figure 6 item (a) shows an overview of the on-path architecture.

Consequently, on off-path architectures, NICs are augmented with additional computation cores and memory within a different SoC, alongside the NIC cores. The offloaded code in this architecture remains off the critical path of the packet processing pipeline. The

offloaded computation here does not impact packet processing performance provided it is not network communication. Wei et al. (WEI et al., 2023) adds that, due to this explicit separation, the SoC can execute a full kernel (e.g., Linux) with a full network stack (i.e., Remote Direct Memory Access), making system development easy and enabling offloading of complicated workloads. Figure 6 item (b) shows an overview of the off-path architecture.

Figure 6 – On-path and off-path SmartNICs.



Source: Adapted from (KFOURY et al., 2024)

2.2.2 ASIC-based SmartNICs

Application-Specific Integrated Circuit (ASIC) is a type of integrated circuit designed for a specific application or function. Unlike general-purpose processors, which can perform a wide range of tasks, ASICs are tailored to execute a particular set of operations efficiently. This specialization allows ASICs to achieve higher performance and lower power consumption compared to general-purpose chips when performing their designated tasks. Higher efficiency and performance are achieved at the cost of the flexibility offered by FPGAs. ASIC development typically requires defining specifications and testing methodologies in advance for all expected functionalities throughout the system's lifetime (DORING; STUBBE; HOLZINGER, 2021). Since basic NICs typically use ASICs, their development cycles are similarly lengthy. In contrast, FPGAs offer much greater

agility for hardware developers. The development of an ASIC can take up to two years, from requirements engineering to finished silicon (FIRESTONE et al., 2018). This long development cycle is a significant drawback for ASIC-based SmartNICs, as it limits their ability to quickly adapt to changing requirements or new technologies. Additionally, ASICs are limited to handling data plane functions. To address this limitation, manufacturers often integrate embedded CPU cores to support new features and manage the control plane. However, these cores are relatively slow and can become bottlenecks as more functionality is offloaded to them. In SmartNICs, a core set of functions known during development benefits from the performance and cost efficiency of ASICs. As a result, ASICs continue to serve as the foundational NIC component but are used in combination with more flexible compute units (DORING; STUBBE; HOLZINGER, 2021).

2.2.3 FPGA-based SmartNICs

An FPGA is a configurable integrated circuit that users can customize after manufacturing, unlike conventional integrated circuits with fixed functionalities. FPGAs achieve this by consisting of programmable logic blocks and interconnects that can be configured to implement various digital circuits simply by loading a configuration file onto the chip, allowing users to specify logic and interconnections. This inherent flexibility makes FPGAs ideal for prototyping, rapid development, and applications that require hardware reconfiguration or customization (DORING; STUBBE; HOLZINGER, 2021). Although FPGAs offer faster performance through parallel processing of data flows compared to the instruction-by-instruction temporal computing of CPUs, their main drawback is the comparatively difficult implementation of new functions. This difficulty stems from the need for expertise in Hardware Description Language to efficiently use the chip, even though manufacturers try to simplify development by offering compatibility with common developer environments like the Data Plane Development Kit (DPDK) (LUIZELLI et al., 2024). Furthermore, due to their flexibility, FPGAs are generally "around 2-3 times larger" than ASICs with the same functionality, as estimated by Microsoft (FIRESTONE et al., 2018), which typically makes them more expensive (DORING; STUBBE; HOLZINGER, 2021).

2.2.4 SoC-based SmartNICs

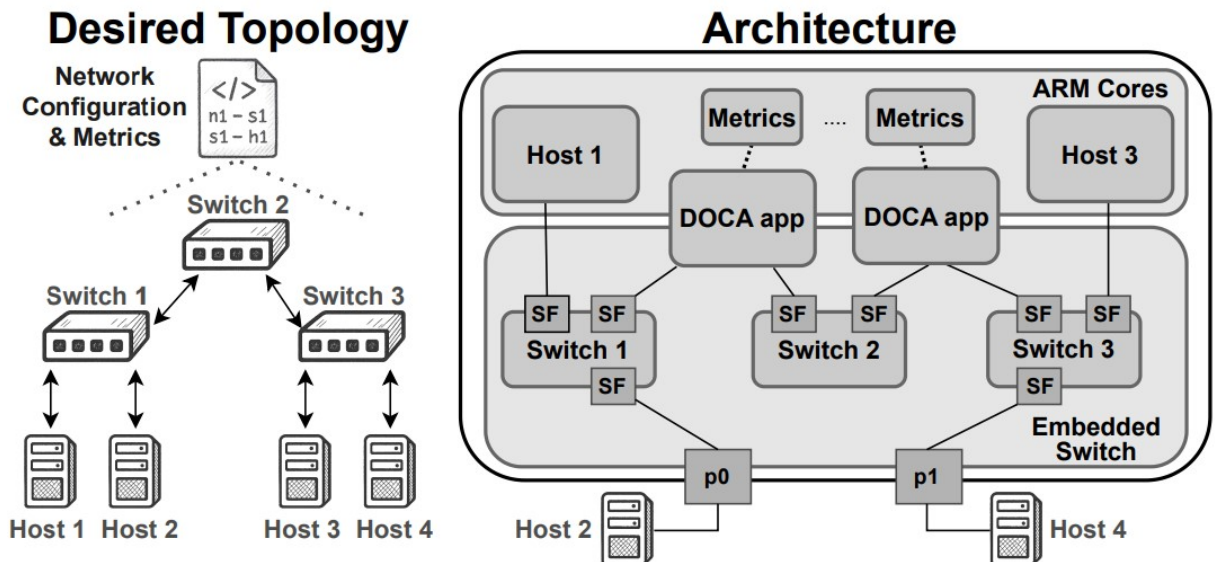
SoC architecture refers to a microchip that integrates most or all components of a computer or electronic system, such as the CPU, memory, Input/Output ports, and peripheral interfaces, such as Universal Serial Bus, High-Definition Multimedia Interface, and Ethernet—into a single integrated circuit. In SmartNICs, SoC architectures enable network programmability within the NIC itself. These networking-focused SoCs typically combine multiple CPUs (e.g., ARM cores), high-speed Random Access Memory (RAM), and fixed-function accelerators such as encryption or regex engines (LUIZELLI et al., 2024). On top of the architecture, an operating system typically runs to manage the

NIC’s resources. Since that is the case, developers can run a wide range of applications on it, from high-performance DPDK-based programs to traditional C-style socket applications. However, to fully optimize performance and resource utilization, vendors typically provide specialized programming frameworks (e.g., the DOCA framework). Unlike programmable switches (e.g., Tofino with its TNA architecture), the performance of SoC-based SmartNICs can vary based on the specific networking application. Since there is no strict upper limit on the number of instructions a packet may undergo, line rate and latency can be affected by the complexity and intensity of packet processing (LUIZELLI et al., 2024).

2.3 SmartNet

SmartNet (VOGT et al., 2025b) is a SmartNIC-based network emulation framework designed to reproduce realistic, high-performance networking environments by directly leveraging the hardware capabilities of modern SmartNICs. Its core idea is to bring SmartNIC awareness into network experimentation by modeling not only switch behavior and link characteristics, but also NIC-level offloading, on-NIC computation, and host interaction. SmartNet enables researchers to define complex topologies that are instantiated directly on SmartNIC hardware for accurate and reproducible evaluation.

Figure 7 – SmartNet Architecture



Source: Adapted from (VOGT et al., 2025b)

SmartNet operates by mapping each element of a user-defined topology, such as a switches, hosts, and links, onto the hardware resources of a BlueField SmartNIC. Switches are emulated using offloaded OVS instances that run on the SmartNIC’s embedded ASIC

cores, allowing forwarding operations to occur with minimal CPU overhead. To interconnect these virtual switches, SmartNet uses SFs, which act as lightweight virtual interfaces on the SmartNIC, enabling the construction of arbitrary link graphs. Emulated hosts run directly on the SmartNIC’s ARM cores, logically partitioning CPU and memory to replicate SmartNIC-equipped end systems, while physical hosts can also connect through SmartNet-managed links. For each link, the framework routes traffic through DOCA/DPDK-based modules that enforce bandwidth limits, fixed or random latency, jitter, and packet loss, ensuring that each virtual link accurately reflects its configured performance characteristics. SmartNet coordinates all of these components through its topology Application Programming Interface, which parses the user’s specification, allocates SmartNIC resources, instantiates virtual devices, and sets up traffic steering between modules. By combining offloaded switching, SF-based connectivity, on-NIC host emulation, and hardware-enforced link dynamics, SmartNet executes entire network topologies within a single SmartNIC while preserving fidelity, performance, and reproducibility. Figure 1 presents the overall architecture of SmartNet, illustrating how switches, hosts, and links are mapped onto the SmartNIC hardware.

SmartNet offers several advantages over existing emulators such as P7 (RODRIGUEZ et al., 2023) and Mininet (LANTZ; HELLER; MCKEOWN, 2010) by introducing the first SmartNIC-aware emulation framework capable of modeling hardware offloading, NIC-host interaction, and the constrained compute environment of modern SmartNICs—capabilities absent from prior systems. Unlike Mininet’s purely software-based operation and P7’s dependence on Tofino ASICs, SmartNet leverages BlueField’s embedded switch and ARM cores to provide hardware-backed packet processing, realistic performance under resource contention, and accurate reproduction of SmartNIC-specific behaviors. Moreover, SmartNet integrates DOCA/DPDK modules to emulate bandwidth, latency, jitter, and packet loss, while supporting heterogeneous combinations of physical and emulated hosts. This results in a more faithful, scalable, and versatile environment for evaluating offloading strategies and NIC-centric network architectures than what P7 or Mininet can offer.

3 METHODOLOGY

This chapter presents the methodology used in this work, including the metrics for performance evaluation, the selection of network topologies, and the description of the testbeds employed for experimentation.

3.1 Metrics

To quantitatively assess the emulation fidelity and scalability of the SmartNet framework, three primary performance metrics were selected, as summarized in Table 1. Throughput serves as the primary indicator of the system’s volumetric capacity, reflecting the maximum forwarding rate the hardware can sustain before saturation. Latency provides a direct measure of the propagation delay introduced by the emulation pipeline, offering insight into the processing overhead of the hardware-offloaded forwarding stages. Finally, Jitter is examined to assess the temporal consistency of packet delivery. This is a critical metric because it quantifies the platform’s stability and determinism, ensuring performance remains predictable even under differing topological loads.

Table 1 – Performance Metrics

Metric	Description
Throughput	Measured in <i>pps</i> , it indicates the rate at which packets are successfully processed by the emulated network.
Latency	Measured in μs , it represents the time taken for a packet to traverse the emulated network from source to destination.
Jitter	Measured in μs , it quantifies the variability in packet latency over time, indicating the stability of the emulated network.

Traffic was generated using TREX¹, with packet sizes of 64, 512 and 1500 bytes, and varying stream counts depending on the number of flows of each topology. These sizes were selected to represent distinct network scenarios: 64 bytes for latency-sensitive applications like Voice over IP, 1500 bytes for bulk data transfers, and 512 bytes as an intermediate average.

Performance metrics were quantified utilizing the intrinsic statistical analysis modules embedded within the TRex traffic generator. For temporal measurements (latency and jitter), the transmission rate was constrained to 1 pps; this low-frequency injection rate was selected because TRex requires software mode for latency tracking (which has performance penalties), and high-rate latency streams would saturate the tracking buffers. Conversely, for throughput and saturation analysis, traffic was generated at the maximum pps where the emulated network could sustain very little packet loss. This dual-mode approach allowed for accurate characterization of both temporal and volumetric performance aspects of the SmartNet emulation framework.

¹ Available at: <https://trex-tgn.cisco.com/>

3.2 Topology Selection

The Internet Topology Zoo was selected as the primary source for network topologies because it offers an extensive collection of real-world operator networks. These datasets capture the structural characteristics of deployed infrastructures more accurately than synthetic graphs. Leveraging these topologies enables evaluation under diverse and realistic conditions, including varying scales and node degrees. Furthermore, The Internet Topology Zoo’s widespread adoption in networking research enhances the comparability and credibility of these results, while its machine-readable formats facilitate straightforward integration into the benchmarking workflow with minimal preprocessing.

To ensure a comprehensive evaluation of SmartNet’s forwarding capabilities, it was necessary to generate traffic patterns that utilized every link within the selected topologies. To achieve this, a pathfinding mechanism based on Dijkstra’s algorithm was employed to compute routing paths between network nodes. The primary objective of this approach was to construct a traffic matrix that forced active flows across every edge in the graph. By guaranteeing that all links were active, the experiments ensured that every hardware-offloaded forwarding rule and Scalable Function within the SmartNet framework was simultaneously exercised, thereby subjecting the system to maximum structural stress.

3.3 Testbeds

The experiments were conducted on a testbed consisting of a server equipped with NVIDIA BlueField-2 Data Processing Unit (DPU) and multiple high-performance network interfaces. The BlueField-2 DPU provides hardware acceleration for packet processing tasks, enabling the evaluation of network functions under realistic traffic loads. Table 2 details the host system configuration used to manage the BlueField-2 DPU, while Table 3 outlines the specifications of the TREX traffic generator host. The BlueField-2 DPU configuration is summarized in Table 4.

Table 2 – Bluefield-2 Host Configuration

Component	Specification
CPU	Intel Xeon Silver 4216 16C/32T CPU @ 2.10GHz
RAM	128 GB
Linux kernel	6.8.0-87-generic
OS	Ubuntu 22.04.4 LTS

Table 3 – TREX Host Configuration

Component	Specification
CPU	AMD Ryzen 7 2700 8C/16T CPU @ 3.2GHz
RAM	24 GB
Linux kernel	3.10.0-1160.95.1.el7.x86_64
OS	CentOS Linux 7
TREX version	3.02

Table 4 – BlueField-2 SmartNIC Configuration

Component	Specification
SoC	MT42822
CPU	A72(D08) 8 Cores
OS	22.04
Linux kernel	5.15.0-1065-bluefield

4 RESULTS AND DISCUSSION

This chapter presents the experimental findings regarding the efficacy and scalability of the SmartNet framework as defined the methodology outlined in Chapter 3.

4.1 Scalability of Topology Emulation

To evaluate SmartNet’s capability to emulate realistic Internet Service Provider (ISP) and carrier-grade networks, the viability of each topology in the Topology Zoo dataset was analyzed for potential deployment on SmartNet. Table 5 shows the selection stages and the number of topologies remaining after each filter. Initially, 172 valid topologies suitable for SmartNet emulation were identified. The first filtering step applied a degree-based constraint derived from the physical limitations of the SmartNIC. Because the BlueField-2 exposes only two physical ports—serving as the entry and exit points of the emulated network—each candidate topology was required to contain no more than two nodes with degree ≤ 1 . Using Dijkstra-based flow construction to ensure that all links could be exercised without cyclic paths, only 45 topologies satisfied this criterion.

Table 5 – Topology selection process and remaining number of topologies at each stage.

Selection Stage	Topologies Remaining
Initial valid Topology Zoo set	172
After degree constraint	45
After node-count constraint	33
After edge constraint	12
Final selected topologies	6

From this filtered set, topologies containing fewer than 10 nodes were excluded, as small-scale emulations had already been explored in prior SmartNet evaluations and would not sufficiently stress the system. Applying this node-count constraint reduced the set to 33 topologies. These remaining topologies were then assessed for feasibility on the BlueField-2 based on SmartNet’s link-emulation mechanism. Because each logical link requires a dedicated DOCA forwarding application connecting two Scalable Functions, empirical testing showed that the BlueField-2 could support no more than 22 concurrent links. Although hugepage availability and RAM capacity contributed to resource pressure, the primary bottleneck was the overhead of launching and maintaining a large number of DOCA applications in parallel. After applying this edge constraint, 12 topologies remained.

From these 12 feasible candidates, 6 topologies were selected for experimentation based on their structural uniqueness and relevance to the scenarios considered in this study. This final subset captures a diverse range of characteristics—including variations

in node degree, graph density, and network diameter, ensuring a broad and meaningful evaluation of SmartNet under realistic and heterogeneous topology structures. The 6 Internet Topology Zoo networks that met the selection criteria and were ultimately chosen for evaluation are summarized in Table 6.

Table 6 – Topology characteristics used in the study

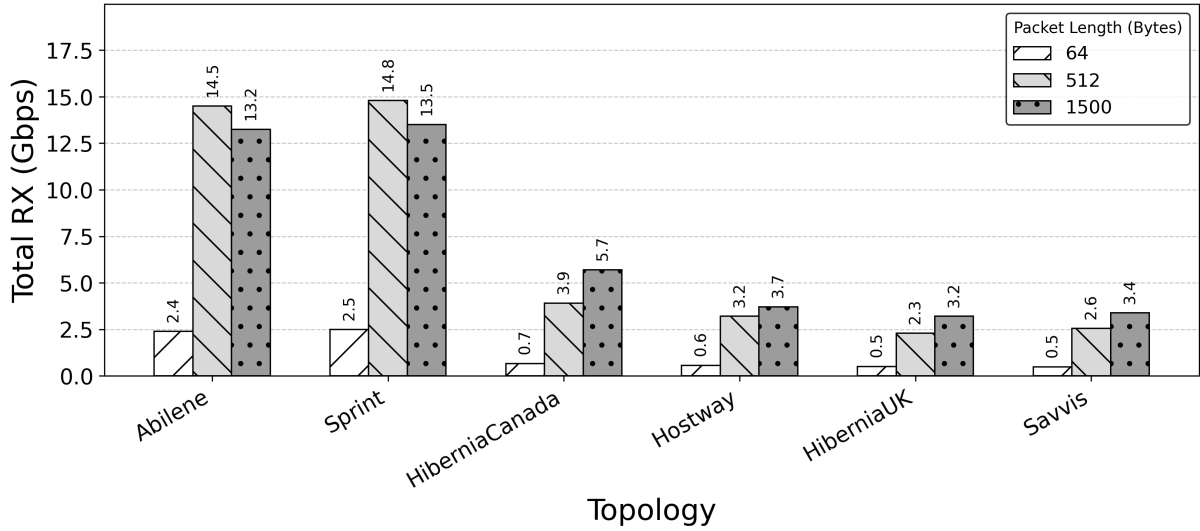
Topology name	No. of nodes	No. of edges	No. of flows
Abilene	11	14	5
HiberniaCanada	13	14	3
HiberniaUK	15	15	6
Hostway	16	22	9
Savvis	19	20	2
Sprint	11	18	3

4.2 Throughput Evaluation

Figure 8 reports the total received throughput (Gbps) observed across the selected Topology Zoo networks for packet sizes of 64, 512, and 1500 bytes. The results show a clear dependence on both packet size and the structural complexity of each topology. For the larger packet sizes (512 and 1500 bytes), the Abilene and Sprint topologies achieved the highest throughput, reaching approximately 14 to 15 Gbps for 512-byte packets and around 13 to 14 Gbps for 1500-byte packets. These topologies are among the smallest in the set, with only 11 nodes each and relatively short end-to-end paths, which reduces the cumulative processing required by SmartNet’s DOCA-based forwarding pipeline. In contrast, topologies with more nodes and edges—such as HiberniaCanada, Hostway, HiberniaUK, and Savvis—exhibited lower throughput, with maximum values ranging from roughly 3 Gbps to 6 Gbps depending on packet size. The additional forwarding stages required in these larger topologies increase the total number of DOCA applications and inter-SF transfers involved in processing each packet, which reduces overall data-plane capacity. The effect is particularly evident at 64-byte packet sizes, where throughput remained below 3 Gbps for all topologies due to the higher per-packet overhead. Taken together, these results indicate that SmartNet sustains high throughput in smaller and moderately connected topologies but experiences expected performance degradation as the number of forwarding hops and emulated links increases.

Figure 9 presents the (pps) throughput achieved across the evaluated Topology Zoo networks for packet sizes of 64, 512, and 1500 bytes. As expected, the smallest topologies—Abilene and Sprint—achieved the highest pps rates, reaching approximately 4.9 to 5.0 (Million packets per seconds (Mpps)) for 64-byte packets and around 3.6 to

Figure 8 – SmartNet throughput across different network topologies.

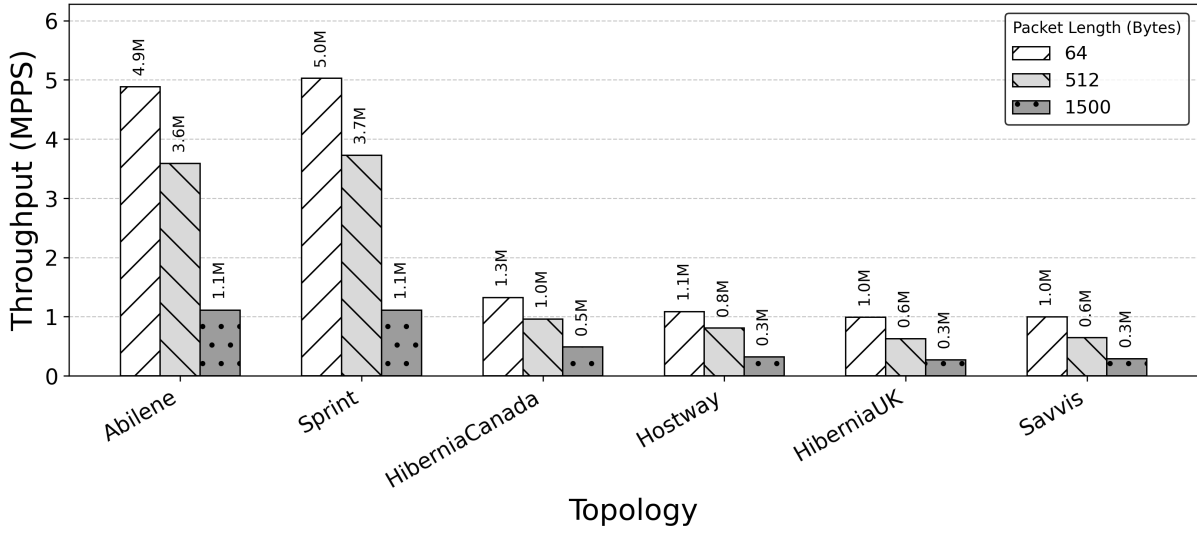


3.7 Mpps for 512-byte packets. In contrast, larger and denser topologies such as HiberniaCanada, Hostway, HiberniaUK, and Savvis delivered significantly lower performance, typically ranging from 1.0 to 1.3 Mpps for 64-byte packets and decreasing sharply for larger packet sizes. These results reflect the cumulative overhead introduced by additional forwarding stages within SmartNet’s DOCA-based pipeline; however, a dominant factor influencing pps throughput is the number of internal packet copies performed by the BlueField-2 eSwitch. Each packet traversing an emulated topology may incur multiple copy operations as it passes between Scalable Functions and DOCA forwarding modules, and these copy operations become increasingly costly as the number of hops grows. This effect is particularly evident with 64-byte packets, where per-packet processing overhead is most pronounced. Consequently, smaller topologies exhibit higher pps performance due to fewer internal copies, while larger topologies experience reduced capacity as packet replication overhead accumulates across multiple emulated links.

4.3 Latency Evaluation

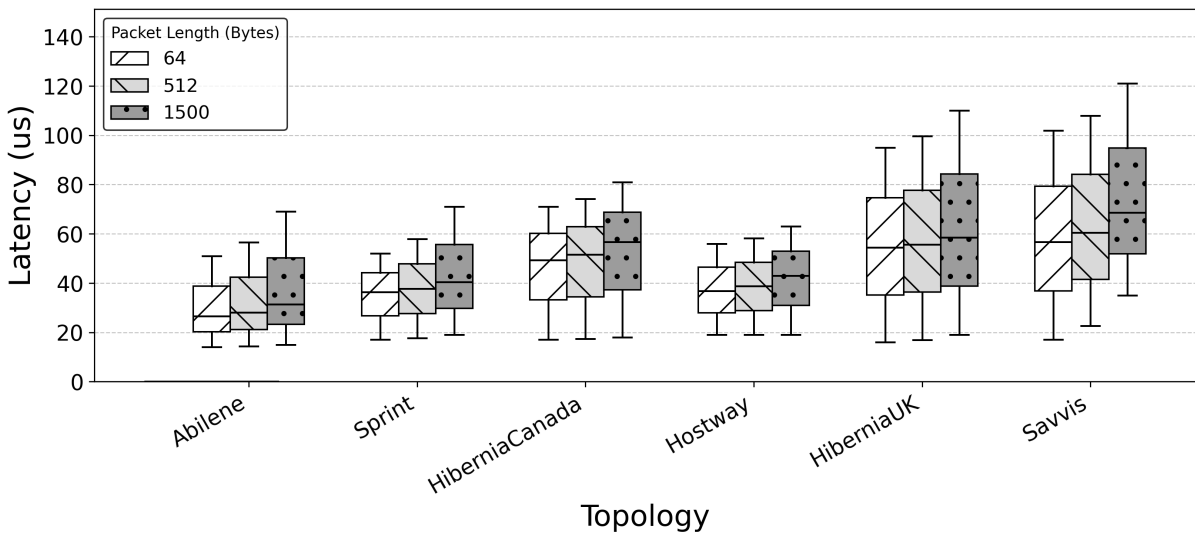
Figure 10 presents the end-to-end latency measured across the selected topologies for packet sizes of 64, 512, and 1500 bytes. Latency was recorded continuously for 1 minute in each experiment, and all values were obtained directly from the T-Rex traffic generator, ensuring accurate and consistent reporting of per-packet timings. The results show a clear progression in mean latency as topology size and forwarding depth increase. Smaller topologies, such as Abilene and Sprint, exhibited mean latency values in the range of 25 to 35 μ s and 30 to 40 μ s, respectively. Medium-scale topologies, HiberniaCanada and Hostway, displayed slightly higher averages, typically around 35 to 55 μ s and 30 to 50

Figure 9 – SmartNet packet per second throughput across different network topologies.



us, reflecting the additional forwarding stages traversed within SmartNet’s DOCA-based pipeline. The largest topologies, HiberniaUK and Savvis, produced the highest mean latencies, approximately 45 to 70 us for HiberniaUK and 55 to 90 us for Savvis. Despite these increases in end-to-end delay, jitter remained consistently low across all experiments, with values between 1.5 and 4 us, indicating that the BlueField-2 and SmartNet forwarding pipeline introduce predictable and stable per-packet timing even when configured with multi-hop, high-complexity logical topologies.

Figure 10 – SmartNet latency across different network topologies.



4.4 Discussion

The experimental results demonstrate that SmartNet is capable of emulating realistic multi-hop network topologies with consistent and predictable behavior, indicating that the framework is viable as a SmartNIC-based network emulation tool. Throughput measurements show that smaller and moderately sized topologies can sustain multi-gigabit performance, and even in more complex scenarios, SmartNet delivers stable forwarding without exhibiting pathological degradation. Latency remains in the tens of microseconds, and jitter remains tightly bounded, confirming that SmartNet provides a highly deterministic execution environment. These characteristics make it suitable for controlled experimentation where reproducibility and timing fidelity are essential. The ability to emulate real-world ISP and carrier topologies from the Topology Zoo further reinforces its practicality for representing realistic network structures.

At the same time, the experiments also reveal clear scalability constraints stemming from the underlying hardware platform. The BlueField-2 SmartNIC can support only a limited number of concurrent DOCA applications, which directly caps the number of emulatable links. Throughput degradation observed in larger topologies is consistent with the internal packet-copy overhead and increasing number of forwarding stages inside the eSwitch. These limitations suggest that some of the observed bottlenecks are not inherent to SmartNet itself but rather reflect the resource constraints of the evaluation platform. A newer generation device, such as the BlueField-3—with significantly higher CPU capacity, improved memory bandwidth, and enhanced packet processing capabilities—would likely reduce many of these overheads. Such hardware could allow SmartNet to emulate larger and denser topologies, achieve higher throughput under load, and support more complex experiments such as multi-pipeline processing or P4-integrated switching.

Overall, the evaluation indicates that SmartNet is a viable and reliable emulation tool for small to moderately sized topologies, providing stable performance and predictable behavior even under multi-hop forwarding scenarios. While enhancements are necessary for SmartNet to scale toward larger and more complex network structures, its current capabilities already offer a practical foundation for SmartNIC-aware experimentation. The fact that the entire emulation runs directly on a SmartNIC is a distinctive and valuable aspect of the framework, demonstrating the feasibility of offloading realistic network emulation tasks to programmable network hardware. Moreover, achieving multi-gigabit performance on a BlueField-2 device highlights promising potential for future deployments: with newer SmartNIC generations offering significantly higher processing capacity, SmartNet can be expected to evolve into an even more capable and scalable platform for high-fidelity network experimentation.

5 FINAL REMARKS

This chapter summarizes the key findings of this work, discusses its limitations, and outlines potential directions for future research.

5.1 Overview

Even amidst a wide range of available network emulation tools—ranging from software-based solutions like Mininet to switch-centric hardware accelerators—SmartNet distinguishes itself through its unique architectural approach. By leveraging the specific capabilities of the NVIDIA BlueField-2 SmartNIC, the framework introduces a novel paradigm that combines the flexibility of general-purpose computing with the performance benefits of hardware offloading. This distinct design allows SmartNet to model network behavior with a level of fidelity and isolation that purely software-based or switch-based alternatives cannot easily replicate, particularly in the context of host-centric emulation.

Prior to this study, preliminary evaluations and demonstrations of SmartNet had successfully validated its fundamental operational mechanics, proving its ability to emulate networks with a small number of nodes with high fidelity. However, these initial tests were largely restricted to abstract or simplistic scenarios. Consequently, a significant validation gap remained regarding the framework’s behavior in highly complex environments. Specifically, it had not yet been subjected to the rigorous structural demands of realistic carrier-grade infrastructures, such as the heterogeneous graph structures cataloged in the Internet Topology Zoo.

Addressing this limitation, the primary focus of this work was to rigorously evaluate SmartNet’s performance and scalability when emulating these complex topologies. By selecting a diverse subset of networks from the Internet Topology Zoo, this study aimed to push the platform beyond its previously established boundaries. The investigation centered on quantifying how the framework handles the non-trivial routing patterns, node densities, and resource contention inherent in production-level topologies, thereby providing the first comprehensive assessment of its viability for large-scale network experimentation.

The experimental outcomes revealed a clear distinction between the framework’s capacity and its precision. Regarding scalability, the results defined a precise operational boundary: the BlueField-2 hardware could successfully emulate topologies containing up to 22 concurrent links. Beyond this threshold, the resource overhead required to maintain independent DOCA forwarding applications saturated the system, preventing the instantiation of more complex graphs. However, within these operational limits, the framework demonstrated exceptional fidelity. Even in the most structurally complex scenarios capable of being deployed, SmartNet maintained stable forwarding behavior characterized by low latency and minimal jitter. This predictable timing behavior is a critical indicator of viability, proving that SmartNet can replicate realistic traffic dynamics

with high precision as long as the topology fits within the hardware’s resource envelope.

5.2 Challenges and Limitations

Several limitations must be acknowledged when interpreting the results of this study. First, the evaluation focused primarily on SmartNet’s ability to emulate network topologies and forwarding behavior, rather than its capability to emulate end hosts. Although SmartNet provides mechanisms for host emulation through ARM-core partitioning on the SmartNIC, these features were not explored in depth in this work. A more comprehensive evaluation including realistic host workloads, traffic generation directly from emulated endpoints, or multi-host scenarios would provide a fuller understanding of SmartNet’s potential and constraints.

A second limitation arises from the characteristics of the Topology Zoo dataset used in this study. To ensure feasibility on a two-port SmartNIC, many topologies with more than two nodes of degree ≤ 1 were excluded. In principle, such topologies could have been adapted by removing low-degree nodes that did not satisfy this constraint; however, doing so would have compromised the structural integrity and realism of the original networks. Because the objective of this study was to evaluate SmartNet using topologies that faithfully represent real-world ISP and carrier infrastructures, no modifications were applied, which consequently reduced the number of topologies eligible for testing.

Finally, the experiments were conducted exclusively on the NVIDIA BlueField-2 SmartNIC. While this platform provides valuable insight into SmartNet’s current behavior, it also imposes hardware constraints—such as limited ARM computing resources, memory availability, and the maximum number of concurrent DOCA applications—that influenced the scalability results. More advanced SmartNICs, such as the BlueField-3, offer increased processing capacity and improved hardware acceleration features that could enable SmartNet to emulate larger and more complex topologies. Testing SmartNet on newer hardware may therefore yield significantly stronger performance and scalability, and represents an important direction for further investigation.

5.3 Future Work

Several avenues for future work emerge from the findings of this study. A natural next step is to deploy SmartNet on newer generations of SmartNIC hardware, particularly the NVIDIA BlueField-3. With substantially higher CPU capacity, increased memory bandwidth, and improved packet-processing capabilities, the BlueField-3 is expected to alleviate many of the scalability bottlenecks encountered on the BlueField-2. Evaluating SmartNet on this platform would provide clearer insights into how much of the current performance ceiling is imposed by the framework itself versus hardware constraints.

Another important direction is to explore SmartNet’s behavior on alternative SmartNIC architectures, especially FPGA-based devices such as the Xilinx SN1000. The reconfigurable nature of FPGAs introduces opportunities for deeper customization of SmartNet’s forwarding pipeline, reductions in per-packet overhead, and offloading of functions that currently rely on software-driven DOCA components. Such architectural diversity would not only broaden the applicability of SmartNet but also reveal how its design can be optimized to take advantage of heterogeneous accelerators.

To address the capacity limits of a single device, future work could investigate partitioning large topologies into smaller, hardware-manageable subgraphs and distributing them across multiple SmartNICs. This form of parallelized emulation would allow SmartNet to scale beyond the resource limits of a single device, enabling the study of significantly larger ISP-grade networks. Coordinating these distributed SmartNet instances—ensuring consistent timing, synchronized forwarding behavior, and unified management—represents a promising path toward high-fidelity, multi-SmartNIC network emulation.

REFERENCES

- DONG, Y. et al. Optimizing network i/o virtualization with efficient interrupt coalescing and virtual receive side scaling. In: **2011 IEEE International Conference on Cluster Computing**. Austin, TX, USA: [s.n.], 2011. p. 26–34. Disponível em: <https://doi.org/10.1109/CLUSTER.2011.12>. 24
- DORING, T.; STUBBE, H.; HOLZINGER, K. SmartNICs: Current trends in research and industry. In: CARLE, G.; GÜNTHER, S.; JAEGER, B. (Ed.). **Proceedings of the Seminar Innovative Internet Technologies and Mobile Communications (IITM), Winter Semester 2020/2021**. Munich, Germany: Chair of Network Architectures and Services, Department of Computer Science, Technical University of Munich, 2021. (Network Architectures and Services (NET), NET-2021-05-1), p. 19–23. Disponível em: https://doi.org/10.2313/NET-2021-05-1_05. 24, 27, 28
- FIRESTONE, D. et al. Azure accelerated networking: SmartNICs in the public cloud. In: **15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)**. Renton, WA: USENIX Association, 2018. p. 51–66. ISBN 978-1-939133-01-4. Disponível em: <https://www.usenix.org/conference/nsdi18/presentation/firestone>. 26, 28
- KANNAN, P. G. et al. BNV: Enabling scalable network experimentation through bare-metal network virtualization. In: **11th USENIX Workshop on Cyber Security Experimentation and Test (CSET 18)**. Baltimore, MD: USENIX Association, 2018. Disponível em: <https://doi.org/10.5555/3292500.3292502>. 17, 21, 23
- KFOURY, E. F. et al. A comprehensive survey on smartnics: Architectures, development models, applications, and research directions. **IEEE Access**, v. 12, p. 107297–107336, 2024. Disponível em: <https://doi.org/10.1109/ACCESS.2024.3437203>. 25, 27
- KNIGHT, S. et al. The internet topology zoo. **IEEE Journal on Selected Areas in Communications**, v. 29, n. 9, p. 1765–1775, 2011. 18
- KOPONEN, T. et al. Network virtualization in multi-tenant datacenters. In: **Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation**. USA: USENIX Association, 2014. (NSDI'14), p. 203–216. ISBN 9781931971096. Disponível em: <https://doi.org/10.5555/2616448.2616460>. 17, 22
- LANTZ, B.; HELLER, B.; MCKEOWN, N. A network in a laptop: rapid prototyping for software-defined networks. In: **Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks**. New York, NY, USA: Association for Computing Machinery, 2010. (Hotnets-IX). ISBN 9781450304092. Disponível em: <https://doi.org/10.1145/1868447.1868466>. 17, 21, 22, 30
- LIU, M. et al. Offloading distributed applications onto smartnics using ipipe. In: **Proceedings of the ACM Special Interest Group on Data Communication**. New York, NY, USA: Association for Computing Machinery, 2019. (SIGCOMM '19), p. 318–333. ISBN 9781450359566. Disponível em: <https://doi.org/10.1145/3341302.3342079>. 26
- LUIZELLI, M. et al. Smartnics: The next leap in networking. In: _____. [s.n.], 2024. p. 40–89. ISBN 9788576696087. Disponível em: <https://doi.org/10.5753/sbc.15408.7.2>. 24, 25, 28, 29

- MACCABE, A. et al. Experience in offloading protocol processing to a programmable nic. In: **Proceedings. IEEE International Conference on Cluster Computing**. Chicago, IL, USA: [s.n.], 2002. p. 67–74. Disponível em: <https://doi.org/10.1109/CLUSTER.2002.1137730>. 24
- PETERSEN, E. et al. Dynamic link network emulation: a model-based design. 2022. Disponível em: <https://arxiv.org/abs/2107.07217>. 17
- PONOMAREV, D.; GHOSE, K. A comparative study of some network subsystem organizations. In: **Proceedings. Fifth International Conference on High Performance Computing (Cat. No. 98EX238)**. Madras, India: [s.n.], 1998. p. 436–443. Disponível em: <https://doi.org/10.1109/HIPC.1998.738019>. 24
- RODRIGUEZ, F. et al. Network emulation with p7: A p4 programmable patch panel on tofino-based hardware. In: **Anais Estendidos do XLI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos**. Porto Alegre, RS, Brasil: SBC, 2023. p. 40–47. ISSN 2177-9384. Disponível em: https://doi.org/10.5753/sbrc_estendido.2023.759. 17, 21, 23, 30
- RODRIGUEZ, F. et al. P4 programmable patch panel (p7): an instant 100g emulated network on your tofino-based pizza box. In: **Proceedings of the SIGCOMM '22 Poster and Demo Sessions**. New York, NY, USA: Association for Computing Machinery, 2022. (SIGCOMM '22), p. 4–6. ISBN 9781450394345. Disponível em: <https://doi.org/10.1145/3546037.3546046>. 21, 24
- ULBRICHT, M. et al. Emulation vs. reality: Hardware/software co-design in emulated and real time-sensitive networks. In: VDE. **European Wireless 2021; 26th European Wireless Conference**. [S.l.], 2021. p. 1–7. 17
- VOGT, F. G. et al. The ghosttwin: Towards live digital twins via smartnic network emulation. In: **Proceedings of the ACM SIGCOMM 2025 Posters and Demos**. New York, NY, USA: Association for Computing Machinery, 2025. (ACM SIGCOMM Posters and Demos '25), p. 187–189. ISBN 9798400720260. Disponível em: <https://doi.org/10.1145/3744969.3748450>. 18
- VOGT, F. G. et al. Smartnet: Bridging performance and realism in network emulation with smartnics. In: **Proceedings of the ACM SIGCOMM 2025 Posters and Demos**. New York, NY, USA: Association for Computing Machinery, 2025. (ACM SIGCOMM Posters and Demos '25), p. 184–186. ISBN 9798400720260. Disponível em: <https://doi.org/10.1145/3744969.3748449>. 17, 18, 29
- WEI, X. et al. Characterizing off-path SmartNIC for accelerating distributed systems. In: **17th USENIX Symposium on Operating Systems Design and Implementation (OSDI 23)**. Boston, MA: USENIX Association, 2023. p. 987–1004. ISBN 978-1-939133-34-2. Disponível em: <https://www.usenix.org/conference/osdi23/presentation/wei-smartnic>. 26, 27
- YAN, J.; JIN, D. A virtual time system for linux-container-based emulation of software-defined networks. In: **Proceedings of the 3rd ACM SIGSIM Conference on Principles of Advanced Discrete Simulation**. New York, NY, USA: Association for Computing Machinery, 2015. (SIGSIM PADS '15), p. 235–246. ISBN 9781450335836. Disponível em: <https://doi.org/10.1145/2769458.2769480>. 17