

Data Center Network Architecture: Evolution, Current State, and Future*

Jan POKRYWCZYNSKI¹ and Piotr ZWIERZYKOWSKI²

¹NetWizards Student Research Group, Faculty of Computing and Telecommunications,
Poznan University of Technology, Poznań, Poland

²Institute of Communication and Computer Networks, Faculty of Computing and Telecommunications,
Poznan University of Technology, Poznań, Poland

Correspondence should be addressed to: Jan POKRYWCZYNSKI, pokrywczynski.jan@gmail.com

*Presented at the 47th IBIMA International Conference, 29-30 June 2026, Madrid, Spain

Abstract

The dynamic development of services in ICT networks is inherently associated with the need to adapt network infrastructure to new and increasing demands. Data centers are a crucial element of the networks used to deliver these services. This article presents the evolution of data center structures, with a clear indication of the factors driving changes in their architecture as a response to the requirements of new network services. The paper analyzes the evolution of data center network architectures in the context of horizontal scaling, east-west traffic handling, and the requirements of AI clusters. It outlines the transition from traditional three-tier architectures to Clos topologies and lossless backend networks for GPU clusters. The scope of the study includes scalability, oversubscription, network virtualization, and the specific characteristics of rail-based topologies. The research methodology includes literature analysis, a review of engineering standards, and an evaluation of technical documentation for industry solutions. It is demonstrated that three-tier architectures have become insufficient due to limited scaling and inefficient handling of intra-data center traffic. Furthermore, it is concluded that AI workloads necessitate rail architectures, which shorten transmission paths and reduce congestion by co-locating the same types of GPUs under common switches. Finally, it is confirmed that the evolution of data center architectures is primarily driven by traffic characteristics, latency requirements, and the scale of parallel processing.

Keywords: Data Center, Spine-Leaf Topology, EVPN-VXLAN, GPU Cluster

Introduction

Impact of Service Development on ICT Infrastructure

The modern digital economy and the everyday functioning of society are inextricably linked to data processing. Data centers currently constitute a critical element of ICT infrastructure (Ciałkowski et al, 2022; Skrzypek, 2025) serving not only simple websites but, above all, complex cloud systems, microservice-based applications, and artificial intelligence (AI) solutions. They are essential for the operation of digital services – from online banking, through healthcare and public administration systems, to streaming platforms, social media, and the Internet of Things – enabling the storage, processing, and delivery of vast amounts of data in a confidential, integral, and available manner (David, 2023; NWI, n.d.).

With the dynamic growth in the number of services and the volume of transmitted data, the requirements placed on computer networks have changed radically, forcing the evolution of their architecture. The way in which applications and servers exchange information dictates the shape and construction of both the physical and logical networks within a data center. Traditional hierarchical network design models optimized for classical client-to-server traffic proved inefficient in the face of virtualized and distributed environments, where communication occurs primarily within the data center itself.

The limitations of historical design approaches compelled the industry to seek new, horizontally scalable topologies. To meet the demands of multi-tenancy, performance, and mobility, the hardware infrastructure was integrated with modern network virtualization technologies. The latest impetus driving yet another change in network architecture has been the proliferation of AI workloads, which generate synchronized, massive traffic between graphics processing units (GPUs) and impose qualitatively different requirements on the network than those encountered even in the most demanding cloud environments (Volico, 2025). This article traces that evolution, analyzing the path from legacy networking concepts, through contemporary horizontal scaling topologies, to specialized backend network architectures for GPU clusters.

The Shift in Network Traffic Paradigm

Initially, data center environments were dominated by monolithic applications, in which all key processes and data resided on a single server or only a few servers. In this model, the predominantly generated traffic was client-server traffic, also referred to as north-south (N-S) traffic (Chatterjee, 2024; George, 2025) – the exchange of data between end users located outside the data center and servers within it.

With the popularization of virtualized environments and microservice-based applications, the nature of traffic within data centers underwent a fundamental change. Applications became distributed systems composed of dozens of small, cooperating components (Chatterjee, 2024; George, 2025), and server-to-server traffic – referred to as east-west (E-W) traffic – began to outweigh client-to-server traffic. Even a small client request can generate significant internal traffic as it is processed by multiple cooperating components. The focus shifted from vertical to horizontal scaling, necessitating a substantial redesign of the network architecture.

The next paradigm shift came with the proliferation of distributed AI model training. Unlike the E-W traffic generated by microservices, traffic in GPU clusters is synchronized – hundreds or thousands of GPUs begin transmission simultaneously as part of collective operations, generating massive, short-lived traffic bursts that require lossless transport and minimal latency (Volico, 2025; George, 2025; Gangidi et al, 2024). Traditional network architectures, designed with statistically distributed traffic in mind, are unable to efficiently handle such communication patterns, necessitating the development of specialized network topologies.

Research Methodology

This paper presents a narrative review of the evolution of data center network architectures, an approach appropriate where a strict systematic search across multiple subfields would be neither feasible nor productive. Sources were prioritized by authority and relevance: foundational publications, IETF specifications, and established technical books anchor the analysis of well-documented architectural patterns; deployment scale insight is drawn from technical documentation issued by vendors operating at scale, including Cisco, Juniper, Arista, and NVIDIA; and recent developments, not yet treated in academic literature – particularly in GPU cluster network design – draw additionally on industry analysis and practitioner publications. Recency was a further selection criterion for fast-evolving subfields. Where multiple sources supported the same claim, the more authoritative was cited as primary; where individual claims rest on industry sources alone, this reflects either the uncontroversial nature of the claim or the absence of comparable primary literature.

Classical Three-Tier Architecture and Its Limitations

Hierarchical Network Model

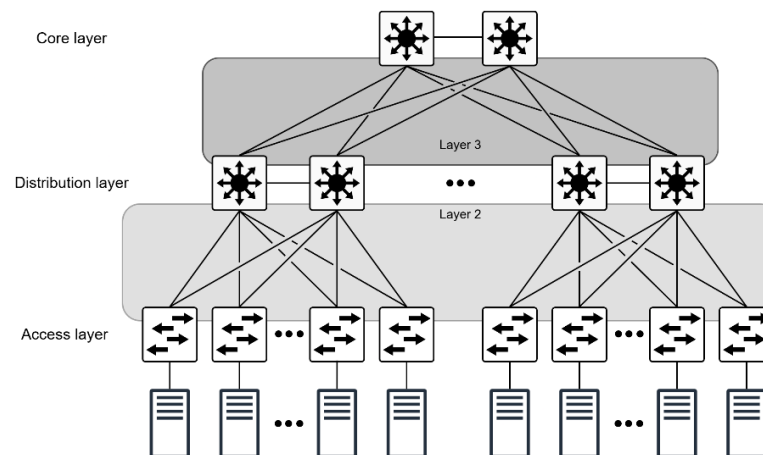
For many years, the standard approach to designing data center networks was the hierarchical three-tier architecture, comprising three layers: access, distribution, and core. This layout served as the foundation for building network infrastructure in enterprises and data centers. Its design naturally fit the prevailing conditions of the time and performed well in supporting monolithic applications and the then-dominant N-S traffic.

This concept was first described in the late 1990s by Cisco Systems, which defined the principles for designing scalable campus networks (Haviland, 1998). In such an architecture, servers were connected directly to the core layer of the campus network, ensuring their high availability. With the dynamic growth in the number of servers, this solution proved insufficient for enterprises, prompting the creation of a separate, dedicated three-tier architecture designed specifically for servers (Santana, 2014).

The three-tier architecture consists of the following layers (Chatterjee, 2024; Cisco, n.d.):

- Access layer – responsible for the physical connectivity of servers and end devices, as well as their logical grouping using Virtual Local Area Networks (VLANs).
- Distribution layer – responsible for connecting the access layer to the core layer, aggregating and terminating Layer 2 broadcast domains, and serving as the default gateways for VLANs.
- Core layer – provides fast and efficient packet transport to and from the data center, serving as the reliable backbone of the network.

Figure 1 illustrates a sample three-tier architecture. Layer 2 operates between the access and distribution layers. Connections between switches are implemented using trunk links, and the Spanning Tree Protocol (STP) or its variants is used to prevent network loops.



**Fig 1. Sample three-tier architecture to which servers are connected.
Inspired by references (Chatterjee, 2024; Santana, 2014; Dutt, 2019)**

Drawbacks of Three-Tier Architecture

The three-tier architecture has significant limitations. The STP protocol, while effectively preventing network loops, causes redundant links between switches to be blocked. This led to the underutilization of available resources and a reduction in overall network efficiency. However, this can be partially mitigated by creating separate spanning trees for specific VLANs or by using Multiple STP (MSTP), although such solutions may still result in suboptimal paths between switches (Chatterjee, 2024; Dutt, 2019).

The consequence of STP operation is that the physically available infrastructure redundancy does not translate into increased available bandwidth – the redundant links are not utilized. In the event of a failure of one of the primary devices, the bandwidth loss can reach as much as 50% (NAKIVO, 2023).

Due to the 12-bit VLAN identifier (IEEE 802.1Q), the number of possible VLANs is limited to 4,094. In large data centers with a high number of isolated tenants, this represents a serious scalability constraint.

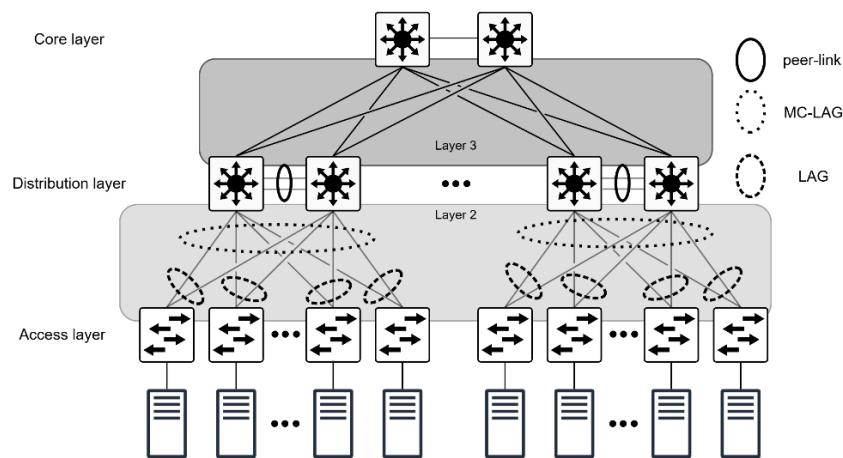
The classical three-tier architecture was characterized by very high oversubscription ratios, reaching 20:1 for access layers, and 4:1 for distribution layer. To increase the number of supported tenants, vertical scaling of server

resources was employed. This allowed a temporary increase in device performance but simultaneously raised the demands on the network infrastructure, meaning that the distribution and core layers also had to be expanded (Chatterjee, 2024).

Link Aggregation and MC-LAG

With the growing demands for reliability and scalability of data center networks, alternative solutions were sought to eliminate the limitations of the classical three-tier architecture. MC-LAG (Multi-Chassis Link Aggregation Group), an extension of Link Aggregation Group (LAG), addressed the most acute of these problems – the blocking of redundant links by STP – by logically combining two physical switches into a single virtual system, enabling all available links to be utilized while providing resilience against device failure (Chatterjee, 2024).

Figure 2 shows a simple hierarchical topology using MC-LAG with visible link aggregation groups. Between the distribution-layer switches, peer-link connections are visible, used for state synchronization, along with additional redundant connections for data transfer (Chatterjee, 2024; Santana, 2014). The distribution-layer switches form MC-LAG groups with the access-layer switches. The access-layer devices, in turn, form LAG groups that connect to the virtually paired distribution-layer switches, perceiving them as a single logical entity.



**Fig 2. Use of link aggregation to utilize all available links.
Inspired by references (Chatterjee, 2024; Santana, 2014; Dutt, 2019)**

However, MC-LAG entails significant challenges: its synchronization protocol is proprietary, it is limited to pairing only two devices, and it consumes physical interfaces that could otherwise serve other purposes (Chatterjee, 2024). These limitations, together with the broader drawbacks of the three-tier architecture, became the impetus for seeking a fundamentally different approach to data center network design – one that would efficiently handle the dominant E-W traffic and be easily scalable horizontally.

Current State: The Era of Clos Architecture and Horizontal Scaling

Three-Stage Clos Topology (Spine-Leaf)

The answer to these challenges – and in particular the need for efficient E-W traffic handling and easy horizontal scaling – was the topology known as Clos architecture.

This concept, although fundamental to modern data centers, is not new. It was described by Charles Clos in a paper published in 1953 in the context of building large, non-blocking switching fabrics used in telephone exchanges (Clos, 1953; Singh, 2018). The original goal was to create a scalable switching fabric composed of smaller elements called switches. Such an architecture made it possible to avoid building a single, large switch with quadratic complexity, while simultaneously ensuring non-blocking connectivity between any input and output (Clos, 1953; Singh, 2018). Figure 3 illustrates a sample Clos architecture consisting of three stages (ingress, middle, and egress), as used in telephony systems, which is strictly non-blocking – meaning that any available input can be connected to any available output without affecting existing connections.

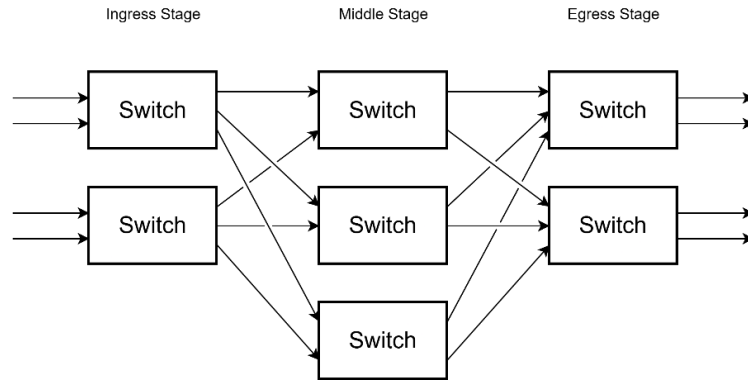


Fig 3. Three-stage Clos architecture. Inspired by references (Chatterjee, 2024; Singh, 2018)

The same scaling property that made the design useful for telephone exchanges is what made it relevant (decades later) for data center networks. Although the application is different – packet forwarding between servers rather than connecting telephone calls – the underlying problem is the same: building a large interconnect from many smaller switching elements while keeping the cost manageable (Sheldon et al, 2023; Singh, 2018).

In modern data centers, this concept has been adapted for packet-switched networks using a so-called folded architecture. In this model, there is no distinction between ingress and egress devices since every device can communicate bidirectionally. Instead, the topology is folded at the middle stage: the former ingress and egress stages are merged into a single layer, and switches are replaced by network switches. Such networks are also called non-contending rather than non-blocking. The change in terminology reflects a structural difference between circuit and packet switching. A circuit-switched fabric can perform admission control at the edge and refuse a connection when bandwidth is insufficient (Sheldon et al, 2023). In packet-switched fabrics a packet enters the moment it is generated, and contention on internal links is resolved by buffering rather than refusing the flow. As a result, even at 1:1 oversubscription ratio, multiple flows may hash onto the same uplink (Sheldon et al, 2023).

This topology, known as the Spine-Leaf architecture, is the folded form of the three-stage Clos architecture, also referred to as a two-layer Clos architecture owing to its two layers of network devices: Spine and Leaf (Chatterjee, 2024). Figure 4 illustrates the folded form of the Clos architecture – the Spine-Leaf topology. Compared to Figure 3, every switch can communicate bidirectionally, and the connections form a full-mesh between layers – each Leaf is connected to every Spine, and each Spine is connected to every Leaf.

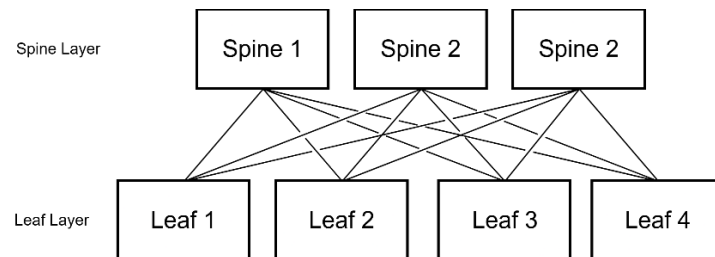


Fig 4. Spine-Leaf architecture. Inspired by references (Chatterjee, 2024; Dutt, 2019)

By employing the Clos architecture, it became possible to reduce the oversubscription ratio at the Leaf device level, which typically stands at 3:1 or lower (Dutt, 2019; Juniper Networks, 2020; Noction, 2021; Arista Networks, 2013). In critical environments, a 1:1 ratio may be required to ensure full bandwidth.

In networks built upon the Clos architecture, Layer 3 is implemented between the Leaf and Spine devices. As a result, STP does not operate between the devices, and consequently, there are no blocked redundant links. This allows the full potential of the devices to be realized by utilizing all available interfaces. The use of routing also enabled the deployment of Equal-Cost MultiPath (ECMP) routing. This mechanism balances traffic across devices but does not ensure uniform bandwidth utilization, as varying flow sizes and hashing can overload some links (Chatterjee, 2024; Dutt, 2019).

Figure 5 illustrates the operation of the ECMP mechanism in a Spine-Leaf topology. Leaf 1 can direct traffic to Leaf 4 through any Spine device – selecting from among equal-cost paths based on a hashing algorithm. Return traffic behaves analogously: Leaf 4 can select any Spine device, not necessarily the same one that was used previously.

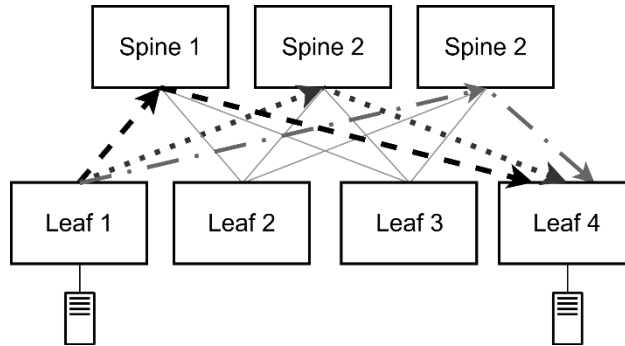


Fig 5. ECMP example. Inspired by references (Chatterjee, 2024)

Increasing the number of Spine devices linearly increases available paths, improving reliability and load balancing while fitting the horizontal scaling model. In the event of the failure of a single Spine device, the bandwidth degradation is only $\frac{1}{N_S}$, where N_S is the number of Spine switches. With a large number of devices in this layer, this is an insignificant value compared to the hierarchical three-tier architecture, where the loss can reach 50% (NAKIVO, 2023).

The Spine-Leaf architecture has been widely adopted as the standard and foundation for building data center networks (Zhao et al, 2019). Increasing the number of Leaf switches allows a higher total workload to be supported, while adding more Spine switches increases the total inter-Leaf bandwidth needed for E-W communication (Cisco, 2025).

Massive Scalability and the Five-Stage Clos Topology

The three-stage Clos architecture has its limitations, primarily related to the number of available interfaces on the device needed to interconnect all switches. If a data center, at the planning stage, is intended to support significant scalability, the two-layer Clos architecture may not be sufficient. In such cases, the network is divided into Points of Delivery (PODs), which represent modular, self-contained sets of compute and networking resources that can be replicated to meet expansion requirements (Chandra, 2017). Each POD consists of a three-stage Clos architecture, and all PODs are connected to a new layer of devices called Super-Spine switches. This architecture is referred to as a five-stage Clos architecture or three-layer Clos architecture.

Figure 6 presents a sample five-stage Clos architecture composed of two PODs. The maximum number of PODs that can be connected via the Super-Spine devices depends on the port density of the switches used.

Traffic originating from a server destined for another server within the same POD does not leave that POD. When devices in different PODs communicate, traffic passes through the Super-Spine layer. Again, as in the Spine-Leaf topology, ECMP ensures that traffic between PODs traverses a fixed number of hops – in this case five hops, hence the name of this topology.

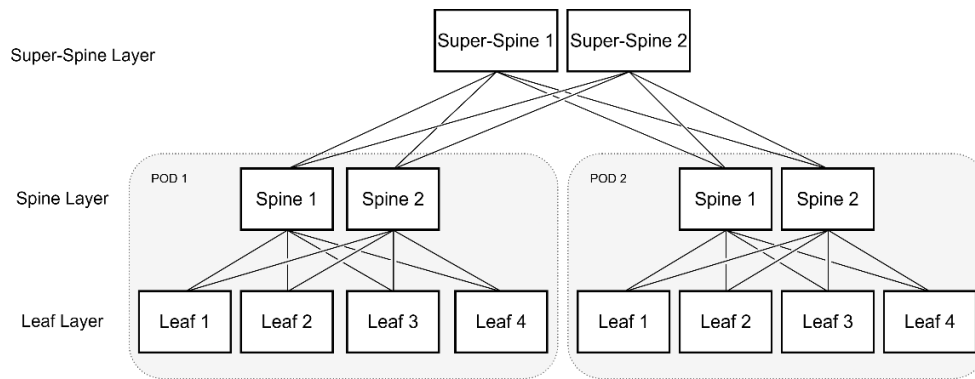


Fig 6. Sample five-stage Clos architecture with the new Super-Spine device layer.
Inspired by references (Chatterjee, 2024; Dutt, 2019)

Use of Super-Spine Planes

The five-stage Clos architecture alone does not resolve all scalability challenges in an equally optimal manner. The key design question remains how to organize the Super-Spine layer, as both scalability and the fault resilience of the entire data center depend on it. The basic implementation of this layer, based on modular switches with high port density, is characterized by a large failure domain – a single Super-Spine device aggregates a significant portion of the data center's total traffic, and its failure leads to the simultaneous loss of a substantial portion of bandwidth. Additionally, the use of modular devices is associated with higher packet processing latency and the risk of vendor lock-in (Cisco, 2024).

The response to these limitations is the application of a plane-based design. To bypass the interface limit when connecting many PODs without using modular high-port-density devices, each Spine connects to only one dedicated Super-Spine plane, thereby creating independent planes. Each plane constitutes a separate failure domain, allowing the entire Super-Spine layer to be built exclusively from smaller, fixed-configuration switches (Cisco, 2024).

Figure 7 presents a sample network topology built using two Super-Spine planes. In each POD, there are two Spine devices, each independently connecting to all Super-Spine devices within its respective plane.

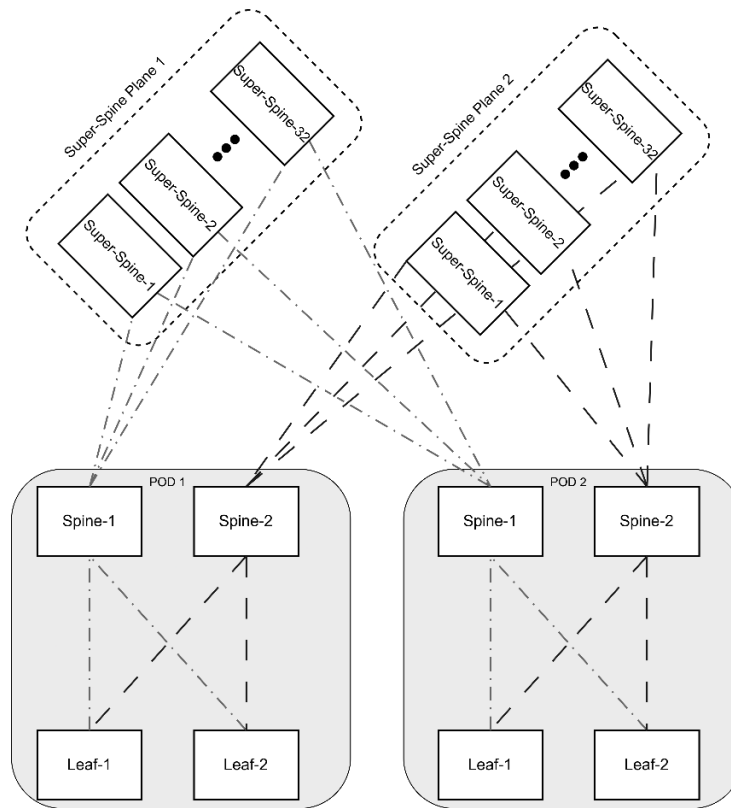


Fig 7. Sample topology built on five-stage Clos architecture using two Super-Spine planes.
Inspired by reference (Cisco, 2024)

Each plane provides an independent data path, with traffic distributed between planes via ECMP. The number of planes equals the number of Spine devices per POD, and the port configuration of Super-Spine devices determines the maximum number of connectable PODs (Cisco, 2024).

Network Virtualization: EVPN-VXLAN as the Communication Standard

With the widespread adoption of the Clos architecture and a fully routed network structure, the limitations of STP were eliminated through the use of ECMP. However, the absence of Layer 2 became a new problem – in certain cases, there was still a need to provide Layer 2 connectivity between hosts. Such requirements were primarily driven by legacy applications such as VMware's VMotion or storage clusters that needed a heartbeat link as a control connection (Chatterjee, 2024; Dutt, 2019).

Network virtualization enables the logical partitioning of a single physical network infrastructure into multiple distinct, isolated virtual networks. The initial inline model, based on VLANs and VRFs (Virtual Routing and Forwarding), requires every device along the packet's path to maintain awareness of all virtual networks, rendering it non-scalable (Dutt, 2019).

The alternative is the overlay model, in which only the edge devices are aware of the virtual networks. A tunnel header is added to the packet, concealing the original headers from intermediate devices. Upon reaching the destination, the header is removed, revealing the original message – intermediate devices (Spine switches) do not need to know anything about the logical network partitioning. It is this model, combined with the fully routed Clos network, that has become the foundation of modern data centers (Dutt, 2019).

The most widely used solution today is VXLAN (Virtual eXtensible Local Area Network) (Dutt, 2019), defined in RFC 7348. It represents the overlay model, allowing the creation of stateless tunnels between devices and thereby enabling the transport of both Layer 2 and Layer 3 traffic. The primary identification unit is VNI (VXLAN Network Identifier) – a 24-bit field enabling the creation of over 16 million unique network segments, representing an enormous increase in scale compared to the approximately 4,000 VLANs available under the IEEE 802.1Q standard (Chatterjee, 2024; Dutt, 2019; Mahalingam et al, 2014).

However, the encapsulation technology alone is only half the solution. Early VXLAN implementations relied on data-plane address learning, which generated excessive broadcast traffic and limited scalability. The answer to this was EVPN (Ethernet Virtual Private Network), an extension of the MP-BGP (Multiprotocol Border Gateway Protocol) protocol. EVPN performs MAC/IP address learning in the control plane, enabling efficient advertisement of host reachability information while minimizing broadcast traffic. It also provides native, standards-based multihoming support (Dutt, 2018; Drake et al, 2015).

The combination of EVPN and VXLAN currently represents the standard model for building modern, scalable data center networks (Dutt, 2019; Arista Networks, 2023).

Complementing this model is the Integrated Routing and Bridging (IRB) mechanism (Rabadan et al, 2021; Sajassi et al, 2021), which moves the default gateway function directly onto the Leaf devices. The use of the anycast distributed gateway mechanism – configuring identical gateway IP addresses on all Leaf devices serving a given virtual network – ensures that hosts see the same gateway regardless of their physical location in the data center, thereby eliminating the inefficient traffic tromboning phenomenon, where traffic had to be directed to centralized default gateways.

Network Operating Systems and Automation

The topological evolution discussed above is accompanied – and made operationally feasible – by two parallel transformations: in the network operating system (NOS) and in the methods used to configure and manage it. Historically, network devices were sold as vertically integrated appliances in which the control plane, the data plane, the management interfaces and the underlying silicon were bundled together within a closed firmware image (Kreutz et al, 2015). The movement toward open programmable networks that began with software-defined networking and the OpenFlow gradually decoupled these layers (Nunes et al, 2014; Anerousis et al, 2021). The role of the NOS has shifted from a closed, vendor-bound firmware image to a cloud-native software platform: a Linux foundation hosting modular, container-based services for routing, management and telemetry, exposed through open programmatic interfaces such as gRPC and REST (Robohn, 2025).

This realignment brings to network devices the software-engineering practices established in cloud platforms, and reframes each switch as a programmable element of a larger distributed system rather than an individually managed appliance.

Network automation methods have undergone a corresponding transformation. As the scale and dynamics of contemporary networks have rendered manual management unsustainable (Arzo et al, 2021), configuration has been recast as version-controlled code, deployed through automated pipelines using tools such as Ansible, Terraform and Python, and exchanged with devices through structured interfaces including NETCONF and gNMI. The concept of intent-based networking, formally defined in RFC 9315 (Clemm et al, 2022) and surveyed in the academic literature (Leivadeas and Falkner, 2023), extends this approach further: operators express high-level outcomes, and the network itself translates them into device-level configurations and continuously verifies compliance.

Together, these two transformations form an essential, software-side counterpart to the physical and logical architectural evolution that is the primary subject of this article. The scale and reconfiguration cadence demanded by modern Clos fabrics, EVPN-VXLAN overlays and AI cluster backend networks are not achievable through topological design alone – they depend on a programmable operating system and a disciplined, automated operations model.

Future Directions: AI Requirements and the Rail Topology

Characteristics of Network Traffic in AI Clusters

Training large language models (LLMs) and other deep learning models requires the cooperation of hundreds or thousands of GPUs, which must exchange data intensively and synchronously (Gangidi et al, 2024; Juniper Networks, 2025). A single GPU server typically contains eight GPUs that are internally interconnected using high-speed internal GPU interconnects (e.g., NVIDIA NVLink) (Wang et al, 2023; Chatterjee and V, 2023), forming a high-bandwidth internal connectivity domain. Communication beyond the boundaries of a single server takes place over the external network, using RDMA (Remote Direct Memory Access) network interface cards (NICs), one per GPU (Juniper Networks, 2025; Chatterjee and V, 2023; Kaye, 2025).

The nature of traffic in AI clusters fundamentally differs from that in traditional data centers. In conventional environments, E-W traffic is asynchronous, statistically distributed, and latency tolerance is relatively high. In GPU clusters, however, traffic is synchronized and massive, generated simultaneously by many GPUs as part of collective operations such as AllReduce, AllGather or All-to-All (Fewell, 2025; Hassan, 2025). These operations serve to synchronize computation results across GPUs – for example, to aggregate gradients computed over different portions of training data – and it is these operations that determine the traffic patterns in the cluster's backend network (Fewell, 2025; Hassan, 2025; Shoeybi et al, 2020).

This traffic is distinguished by three characteristics:

- It is synchronous – all GPUs begin work simultaneously, generating microbursts of traffic (Fewell, 2025; Intelligent Visibility, 2026).
- It is massive – a single server with eight GPUs collectively generates traffic on the order of terabits per second (Nathan, 2024).
- It is sensitive to latency imbalance – the slowest GPU determines the pace of the entire task, because all GPUs must synchronize before the next computational step (Lin et al, 2025). Consequently, the AI cluster's backend network requires a non-blocking topology with a 1:1 oversubscription ratio, lossless transport, and predictable, low latency (Juniper Networks, 2025; Kaye, 2025; Haim, 2025; Route XP, 2026).

The network infrastructure of an AI cluster consists of several independent networks serving distinct functions (Gangidi et al, 2024; Chatterjee and V, 2023). The backend network – a dedicated infrastructure connecting exclusively the RDMA NICs of the GPUs for collective operation traffic – is the subject of further analysis. The frontend network handles management traffic, storage communication, and external connectivity. Their physical separation enables independent optimization and eliminates mutual interference.

Lossless Networks

Traditional Ethernet networks operate on a best-effort model, dropping packets when buffers are congested. In AI clusters, this approach is unacceptable – RDMA technology, which is used for direct communication between GPU memories while bypassing the TCP/IP stack, requires lossless network transport (Gangidi et al, 2024; Chatterjee and V, 2023; Haim, 2025). Packet loss causes costly retransmissions, which can significantly degrade training performance.

Communication between GPUs in AI clusters can utilize two primary networking technologies: InfiniBand or Ethernet with RoCEv2 (RDMA over Converged Ethernet version 2) protocol. Infiniband, originating from high-performance computing (HPC) environments, provides natively lossless transport and low latency (Route XP, 2026).

The RoCEv2 protocol encapsulates RDMA communication in UDP/IP packets, enabling Layer 3 routing and the construction of scalable network structures based on the Clos architecture, using widely available Ethernet switches (Gangidi et al, 2024). In recent years, RoCEv2 has been gaining popularity due to lower costs, a multi-vendor ecosystem, and the ability to build a unified network supporting both training traffic and the remaining data center traffic (Gangidi et al, 2024; Route XP, 2026).

Rail-Optimized Topology

The previously described Spine-Leaf architecture works well in traditional data centers but proves suboptimal in GPU clusters (Kaye, 2025; Haim, 2025; Route XP, 2026). A Leaf switch aggregates traffic from all GPUs in a rack, and communication between GPUs in different racks requires traversing at least two switches via a Spine device. With synchronized collective operations, this leads to traffic concentration on Spine uplinks and the formation of congestion, further exacerbated by the low effectiveness of ECMP hashing algorithms in the face of low-entropy traffic with similarly sized flows (Gangidi et al, 2024; Kaye, 2025).

The response to these limitations is the rail-optimized topology, widely adopted by operators of large AI clusters (Juniper Networks, 2025; Chatterjee and V, 2023; Kaye, 2025). This concept is based on a key observation regarding the architecture of modern GPU servers: each server contains K GPUs (typically $K = 8$), each equipped with a dedicated RDMA NIC (Juniper Networks, 2025; Chatterjee and V, 2023; Kaye, 2025). In the rail-optimized topology, GPUs with the same index (position) across all servers – for example, all GPUs with index 1 – are

connected to a shared Leaf switch, forming a so-called rail. This minimizes the number of hops, latency, and risk of congestion (Juniper Networks, 2025; Chatterjee and V, 2023; Juniper Networks, 2026).

The structural building block is a stripe – a modular unit consisting of as many Leaf switches as there are GPUs per server, connected to shared Spine devices. Additional stripes can be added to scale the cluster (Juniper Networks, 2025; Chatterjee and V, 2023; Juniper Networks, 2026).

The rail-optimized topology retains the Clos architecture as the foundation for routing and the ECMP mechanism but modifies the way GPUs are assigned to switches, aligning the physical network infrastructure with communication patterns generated by collective operations (Juniper Networks, 2025; Kaye, 2025).

Figure 8 presents a sample rail-optimized topology with K rails forming a single stripe (Stripe 1) connected to a shared Spine layer, providing inter-rail connectivity and enabling the attachment of additional stripes.

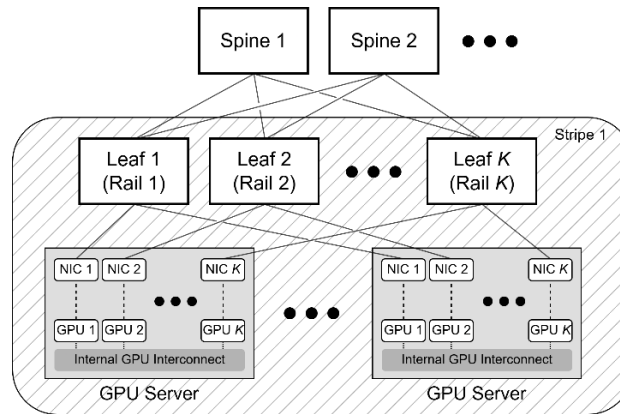


Fig 8. Sample rail-optimized topology. Inspired by references (Juniper Networks, 2025; Wang et al, 2023; Chatterjee and V, 2023; Kaye, 2025; Juniper Networks, 2026; Mandakolathur and Jeaugey, 2022)

Communication within a single rail – for example, between all GPUs with index 1 within a stripe – passes through only one Leaf switch, providing single-hop communication with minimal latency (Kaye, 2025). Communication between GPUs belonging to different rails – for example, between GPU index 1 and GPU index 2, located on different servers – is in modern implementations realized using the server's high-speed internal interconnect: data is transferred within the source server to the local GPU belonging to the target rail, and then sent via that GPU's NIC and the Leaf switch of the target rail to the destination GPU on another server (Mandakolathur and Jeaugey, 2022). As a result, traffic that from the application's perspective constitutes inter-rail communication, from the network's perspective remains intra-rail traffic and does not load the Spine layer (Mandakolathur and Jeaugey, 2022). The Spine layer provides full any-to-any connectivity between all rails and stripes and is utilized in scenarios where optimization via the internal interconnect is insufficient – for example, in large-scale All-to-All operations where each GPU must exchange unique data with many GPUs on different rails across different servers spanning multiple stripes (Juniper Networks, 2025; Mandakolathur and Jeaugey, 2022).

Figure 9 shows the communication paths in the rail-optimized topology. Intra-rail traffic passes through only one Leaf switch, while inter-rail traffic is routed through the server's internal interconnect. The Spine layer serves as an alternative communication path.

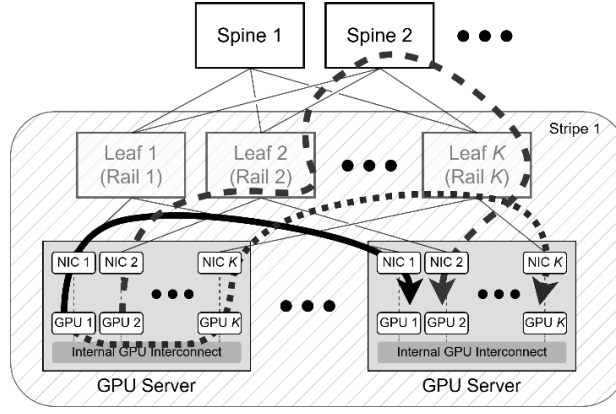


Fig 9. Communication example in the rail-optimized topology. A solid arrow indicates intra-rail traffic, a dashed arrow indicates the alternative communication path via the Spine layer, and a dotted arrow represents communication via the internal GPU interconnect during communication between GPUs of different indices. Inspired by references (Juniper Networks, 2025; Chatterjee and V, 2023; Kaye, 2025; Haim, 2025; Mandakolathur and Jeaugey, 2022)

This topology is particularly effective when intra-rail traffic dominates, which occurs during AllReduce operations in data parallelism.

Rail-Only Topology

A further development of rail-optimized concept is the rail-only architecture, proposed by researchers at MIT in the context of optimizing networks for the traffic patterns generated by LLMs (Wang et al, 2023). The starting point of this concept is the observation that in the rail-optimized topology, the Spine layer connecting rails to each other is in practice utilized to a limited extent – the dominant traffic occurs within rails, and inter-rail communication is primarily realized through the servers' internal interconnects (Wang et al, 2023).

The rail-only architecture draws the logical conclusion from this observation: since the Spine layer is lightly loaded, it can be entirely eliminated (Wang et al, 2023). Instead of a single shared Clos network connecting all GPUs through a common Spine layer, each rail receives its own independent Clos network, providing full-bisection bandwidth exclusively within the given rail (Wang et al, 2023). This means that rail 1 (all GPUs with index 1 from all servers) has its own separate structure of Leaf and Spine switches, rail 2 has its own, and so on for each of the K rails. These networks are not interconnected at the network infrastructure level in any way (Wang et al, 2023).

The switch interfaces that in the rail-optimized topology served to connect the shared Spine layer linking the rails are in the rail-only architecture repurposed as additional ports for connecting GPUs. As a result, with the same switches, the rail-only architecture allows either more GPUs to be connected or a cluster of the same size to be built using fewer network devices (Wang et al, 2023).

Figure 10 presents a sample rail-only topology, in which each rail has its own independent Clos network with its own Leaf and Spine switches, serving exclusively GPUs of the same index.

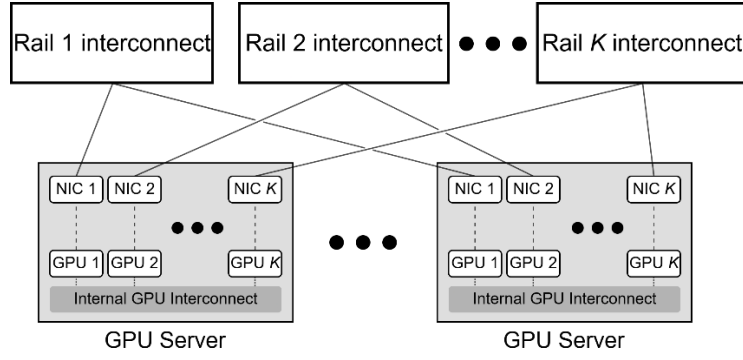


Fig 10. Sample rail-only topology. Each rail interconnect represents a simplified, independent Clos network with no network connections between rails. Inspired by references (Wang et al, 2023; Juniper Networks, 2025; Haim, 2025)

Inter-rail communication occurs in the same way as in the rail-optimized topology – through the server's internal interconnect to a GPU on the target rail, and then through the Clos network of that rail (Wang et al, 2023). The key difference is that in the rail-only architecture, this is the only available method, since there is no Spine layer providing an alternative network path between rails (Wang et al, 2023). Cluster scaling is accomplished by expanding each rail's Clos network independently – each new server's GPUs are connected to their respective rail networks (Wang et al, 2023).

Analysis of traffic patterns in models using hybrid parallelism has shown that the dominant traffic occurs within rails, while inter-rail traffic constitutes a small fraction of the total (Wang et al, 2023). The elimination of the shared Spine layer reduces the number of switches and optical transceivers, lowering construction costs and energy consumption, while maintaining comparable fault tolerance properties – a switch failure in a given rail leads to the unavailability of the GPUs connected to it in both topologies, but the lower total number of devices naturally reduces the number of potential failure points (Wang et al, 2023).

Comparison of Approaches and Selection Criteria

Both rail topologies stem from the same observation: that aligning the physical network infrastructure with GPU communication patterns can significantly reduce the number of hops and limit congestion (Chatterjee and V, 2023; Juniper Networks, 2026; Kaye, 2025). Intra-rail communication proceeds identically in both cases – through a single Leaf switch. Inter-rail communication also uses the same mechanism – transferring data through the server's internal interconnect to a GPU on the target rail. The fundamental difference concerns the presence of a Spine layer connecting the rails to each other.

The rail-optimized topology retains the shared Spine layer, providing any-to-any network connectivity between all GPUs in the cluster (Juniper Networks, 2025). The Spine serves as a supplementary network path – in typical scenarios, inter-rail traffic is directed to the servers' internal interconnects, but the Spine layer remains available for workloads requiring intensive large-scale inter-rail communication (Juniper Networks, 2025). Cluster scaling is achieved by adding additional stripes connected to the shared Spine layer.

The rail-only topology eliminates the shared Spine layer, replacing it with K independent Clos networks – one per rail (Wang et al, 2023). As a result, inter-rail communication relies exclusively on the servers' internal interconnects, with no alternative network path (Wang et al, 2023). The key benefit of this approach is the reduction in the number of required network devices. In the rail-optimized topology, all Leaf switches from all rails share a single common Spine layer – in large clusters, the number of Leaf switches exceeds the port capacity of the Spine switches, necessitating the construction of a multi-tier structure with a Super-Spine layer. In the rail-only topology, each rail builds its own independent Clos network serving eight times fewer GPUs, enabling the use of fewer switch tiers within each rail (Wang et al, 2023). The result is a significant reduction in the total number of switches and optical transceivers for the same cluster size (Wang et al, 2023).

However, the rail-only architecture is optimal only in scenarios where inter-rail traffic is marginal – when intra-rail collective operations dominate (Wang et al, 2023). For workloads generating intensive inter-rail traffic, the absence of a shared Spine layer becomes a limitation, as all inter-rail communication must pass through the internal GPU interconnect (Wang et al, 2023). The rail-optimized topology, by retaining the Spine layer, remains more versatile, at the cost of a greater number of network devices (Juniper Networks, 2025; Wang et al, 2023).

In both network topologies, the AI cluster's backend network constitutes a separate, dedicated infrastructure isolated from the frontend network (Gangidi et al, 2024; Chatterjee and V, 2023), and the oversubscription ratio should be 1:1 (Chatterjee and V, 2023; Route XP, 2026; Juniper Networks, 2026).

Summary

Data center network architecture has undergone a clear evolution, driven at each stage by the changing nature of network traffic and the increasing demands of applications. The hierarchical three-tier architecture, used during the dominance of N-S traffic and monolithic applications, became insufficient when virtualized and microservice-based environments shifted the focus to E-W traffic within the data center (Chatterjee, 2024; George, 2025). The limitations of the Spanning Tree Protocol, the small pool of VLAN identifiers, and high oversubscription ratios necessitated the search for fundamentally different solutions.

The Clos architecture in the form of the Spine-Leaf topology solved these problems by introducing a fully routed structure based on Layer 3 of the OSI model with the ECMP mechanism, eliminating the blocking of redundant links and enabling horizontal scaling (Chatterjee, 2024; Dutt, 2019). The extension to five-stage Clos architecture with the Super-Spine layer and the division into PODs and planes enabled the construction of networks spanning tens of thousands of servers (Chatterjee, 2024; Dutt, 2019; Chandra, 2017). Supplementing this architecture with EVPN-VXLAN technology enabled network virtualization in the overlay model, overcoming the VLAN count limitation and providing logical Layer 2 connectivity without compromising scalability (Dutt, 2019; Mahalingam et al, 2014; Drake et al, 2015; Dutt, 2018).

The emergence of AI workloads introduces the next stage of this evolution. The synchronized, massive nature of traffic generated by collective operations in GPU clusters, combined with the requirement for losslessness and minimal latency, forces further specialization of network topology (Volico, 2025; Gangidi et al, 2024; Fewell, 2025). The standard Spine-Leaf architecture, while remaining the foundation of the physical layer, proves suboptimal from the perspective of GPU communication patterns (Chatterjee and V, 2023; Haim, 2025; Route XP, 2026). The rail-optimized topology aligns the physical network structure with these patterns, grouping GPUs of the same index under shared switches and minimizing the number of hops for the dominant intra-rail operations, while retaining the shared Spine layer as a supplementary path (Juniper Networks, 2025; Chatterjee and V, 2023; Kaye, 2025). The rail-only architecture goes a step further, replacing the single shared network with independent Clos networks for each rail, enabling a reduction in switch tiers and a decrease in the total number of network devices, at the cost of limited flexibility for workloads requiring intensive inter-rail communication (Wang et al, 2023).

The analysis confirms that three factors consistently determine data center network architecture development: the nature of dominant traffic, application latency requirements, and the scale of parallel processing. Each architectural transition – from three-tier to Spine-Leaf, and from Spine-Leaf to rail topologies – was driven by a shift in these factors. The continued growth of AI models will in all likelihood continue this trend, driving further specialization of network topologies.

Acknowledgment

This research was funded in part by the Polish Ministry of Science and Higher Education (grant SBAD-MK 0313/SBAD/1314).

References

- Anerousis, N., Chemouil, P., Lazar, A.A., Mihai, N., and Weinstein, S.B. (2021) 'The Origin and Evolution of Open Programmable Networks and SDN,' *IEEE Communications Surveys & Tutorials*, 23 (3), 1956–1971.
- Arista Networks (2013) *Cloud Networking: Scaling Out Data Center Networks*, White Paper, Arista Networks.
- Arista Networks (2023) *The EVPN Data Center - EVPN Gateway for Hierarchical Multi-Domain EVPN and DCI*, White Paper, Arista Networks.
- Arzo, S., Serugunda, C., Granelli, F., Bassoli, R., Devetsikiotis, M., and Fitzek, F. (2021) 'A theoretical discussion and survey of network automation for IoT: Challenges and opportunity,' *IEEE Internet of Things Journal*, 8 (15), 12021–12045.
- Chandra, D. (2017) *This Week: Data Center Deployment with EVPN/VXLAN*, Juniper Networks Books.

- Chatterjee, A. (2024) *Deploying Juniper data centers with EVPN VXLAN*, Addison-Wesley, Hoboken, New Jersey.
- Chatterjee, A. and V, V. (2023), Designing Data Centers for AI Clusters. [Online].Juniper Networks. <https://www.juniper.net/documentation/us/en/software/nce/ai-clusters-data-center-design/ai-clusters-data-center-design.pdf>.
- Ciałkowski, M., Łodygowski, T., and Żyluk, A. (eds) (2022) *Nauka dla Obronności. Bezpieczeństwo infrastruktury krytycznej. Tom 1*, Instytut Techniczny Wojsk Lotniczych.
- Cisco (2024), Cisco Massively Scalable Data Center Network Fabric Design and Operation White Paper. [Online].Cisco. [Retrieved April 7, 2026], <https://www.cisco.com/c/en/us/products/collateral/switches/nexus-9000-series-switches/white-paper-cl11-743245.html>.
- Cisco (2025), Cisco Nexus 9000 VXLAN BGP EVPN Data Center Fabrics Fundamental Design and Implementation Guide. [Online].Cisco. [Retrieved April 7, 2026], <https://www.cisco.com/c/en/us/td/docs/dcn/whitepapers/cisco-vxlan-bgp-evpn-design-and-implementation-guide.html>.
- Cisco (n.d.) *Small Enterprise Design Profile (SEDP)—Network Foundation Design*, Cisco Validated Design, Cisco Systems, Inc.
- Clemm, A., Ciavaglia, L., Granville, L.Z., and Tantsura, J. (2022), Intent-based networking - concepts and definitions. [Online].RFC Editor. <https://www.rfc-editor.org/info/rfc9315>.
- Clos, C. (1953) 'A Study of Non-Blocking Switching Networks,' *Bell System Technical Journal*, 32 (2), 406–424.
- David, A.-S. (2023), Why are data centers important? No data centers, no digital world. [Online].DATA4 - Smart Data Centers at Scale. [Retrieved April 7, 2026], <https://www.data4group.com/en/news-data4/without-data-center-there-d-be-no-digital-world/>.
- Drake, J., Henderickx, W., Sajassi, A., Aggarwal, R., Bitar, N.N., Isaac, A., and Uttaro, J. (2015) *BGP MPLS-Based Ethernet VPN*, Request for Comments RFC 7432, Internet Engineering Task Force.
- Dutt, D.G. (2018) *EVPN in the Data Center*, O'Reilly Media, Inc., Sebastopol, CA.
- Dutt, D.G. (2019) *Cloud Native Data Center Networking: architecture, protocols, and tools*, O'Reilly Media, Beijing [China] ; Sebastopol, CA.
- Fewell, A. (2025), Collective Communications Explained: The Hidden Coordination Behind Distributed AI Training. [Online].Hedgehog. [Retrieved April 8, 2026], <https://hedgehog.cloud/blog/collective-communications-explained-the-hidden-coordination-behind-distributed-ai-training>.
- Gangidi, A., Miao, R., Zheng, S., Bondu, S.J., Goes, G., Morsy, H., Puri, R., Riftadi, M., Shetty, A.J., Yang, J., Zhang, S., Fernandez, M.J., Gandham, S., and Zeng, H. (2024) 'RDMA over Ethernet for Distributed Training at Meta Scale' *Proceedings of the ACM SIGCOMM 2024 Conference*, ISBN: 979-8-4007-0614-1, 4 August 2024, New York, NY, USA, 57–70.
- George, Dr.A.S. (2025) 'The Evolution of Data Center Networks: Strategies for Modern Infrastructure Design,' *Partners Universal Multidisciplinary Research Journal*, 2 (3), 141–159.
- Haim, S. (2025), Rail vs ToR Architectures: Which Is Best for AI Clusters? [Online].DriveNets. [Retrieved April 8, 2026], <https://drivenets.com/blog/rail-vs-tor-architectures-which-is-best-for-ai-clusters/>.
- Hassan, S. (2025), Understanding LLM GPU Clusters Fabrics Traffic For Networkers. [Online].Dell. [Retrieved April 8, 2026], <https://www.dell.com/en-us/blog/understanding-llm-gpus-clusters-fabrics-traffic-for-networkers/>.
- Haviland, G. (1998) *Designing High-Performance Campus Intranets with Multilayer Switching*, White Paper, Cisco Systems, Inc.
- Intelligent Visibility (2026), Lossless Ethernet Design Guide for AI Fabrics (RoCE v2). [Online].[Retrieved April 8, 2026], <https://intelligentvisibility.com/ai-networking-solutions/lossless-networking-ai>.
- Juniper Networks (2020) *Design Considerations for Spine-and-Leaf IP Fabrics*, White Paper, Juniper Networks, Inc.
- Juniper Networks (2025), Backend GPU Rail Optimized Stripe Architecture. [Online].[Retrieved April 8, 2026], https://www.juniper.net/documentation/us/en/software/jvd/jvd-ai-dc-apstra-nvidia-weka/backend_gpu_rail_optimized_stripe_architecture.html.
- Juniper Networks (2026), Solution Architecture. [Online].[Retrieved April 8, 2026], https://www.juniper.net/documentation/us/en/software/jvd/jvd-ai-dc-apstra-amd/solution_architecture.html#Toc171952249__RAIL_OPTIMIZED_STRIPE_ARCHITECTURE.

- Kaye, A. (2025), Rail-Optimised Networking: How NVIDIA is Rethinking AI Network Design in the Data Centre. [Online].Vespertec. [Retrieved April 8, 2026], <https://vespertec.com/news/rail-optimised-networking-how-nvidia-is-rethinking-ai-network-design-data-centre/>.
- Kreutz, D., Ramos, F.M.V., Verissimo, P.E., Rothenberg, C.E., Azodolmolky, S., and Uhlig, S. (2015) 'Software-Defined Networking: A Comprehensive Survey,' *Proceedings of the IEEE*, 103 (1), 14–76.
- Leivadeas, A. and Falkner, M. (2023) 'A Survey on Intent-Based Networking,' *IEEE Communications Surveys & Tutorials*, 25 (1), 625–655.
- Lin, J., Jiang, Z., Song, Z., Zhao, S., Yu, M., Wang, Z., Wang, C., Shi, Z., Shi, X., Jia, W., Liu, Z., Wang, S., Lin, H., Liu, X., Panda, A., and Li, J. (2025) *Understanding Stragglers in Large Model Training Using What-if Analysis*, arXiv, <http://arxiv.org/abs/2505.05713>.
- Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and Wright, C. (2014) *Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks*, Request for Comments RFC 7348, Internet Engineering Task Force.
- Mandakolathur, K. and Jeaugey, S. (2022), Doubling all2all Performance with NVIDIA Collective Communication Library 2.12. [Online].NVIDIA Technical Blog. [Retrieved April 8, 2026], <https://developer.nvidia.com/blog/doubling-all2all-performance-with-nvidia-collective-communication-library-2-12/>.
- NAKIVO (2023), MSP Network Topology for Beginners. [Online].NAKIVO Team. [Retrieved March 10, 2026], <https://www.nakivo.com/blog/msp-network-topology-for-beginners/>.
- Nathan (2024), Understanding GPU Server Network Card Configuration & Optical Communication Solutions. [Online].NADDOD. [Retrieved April 8, 2026], <https://www.naddod.com/blog/quick-understanding-gpu-server-network-card-configuration-in-ai-era>.
- Noction (2021), Oversubscription in Networking. [Online].Noction. [Retrieved April 7, 2026], <https://www.noction.com/blog/oversubscription-in-networking>.
- Nunes, B.A.A., Mendonca, M., Nguyen, X.-N., Obraczka, K., and Turletti, T. (2014) 'A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks,' *IEEE Communications Surveys & Tutorials*, 16 (3), 1617–1634.
- NWI (n.d.), Why are data centers important? [Online].[Retrieved April 7, 2026], <https://nwidatafacts.nwforum.org/faq/why-are-data-centers-important/>.
- Rabadan, J., Henderickx, W., Drake, J., Lin, W., and Sajassi, A. (2021) *IP Prefix Advertisement in Ethernet VPN (EVPN)*, Request for Comments RFC 9136, Internet Engineering Task Force.
- Robohn, S. (2025), Architectural evolution in data center networks: Modern NOS, open protocols, and high-speed silicon. [Online].[Retrieved May 7, 2026], <https://techstrong.it/sponsored/modern-dcn-checklist-blog-series/architectural-evolution-in-data-center-networks-modern-nos-open-protocols-and-high-speed-silicon/>.
- Route XP (2026), AI Data Center Networking: How GPU Clusters Are Changing Network Design. [Online].The Network DNA: Networking, Cloud, and Security Technology Blog. [Retrieved April 8, 2026], <https://www.thenetworkdna.com/2026/03/ai-data-center-networking-how-gpu.html>.
- Sajassi, A., Salam, S., Thoria, S., Drake, J., and Rabadan, J. (2021) *Integrated Routing and Bridging in Ethernet VPN (EVPN)*, Request for Comments RFC 9135, Internet Engineering Task Force.
- Santana, G.A.A. (2014) *Data center virtualization fundamentals*, Cisco Press, Indianapolis, IN.
- Sheldon, R., Scarpati, J., and Nolle, T. (2023), What is a Clos network? [Online].Search Networking. [Retrieved May 4, 2026], <https://www.techtarget.com/searchnetworking/definition/Clos-network>.
- Shoenybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., and Catanzaro, B. (2020) *Megatron-LM: Training Multi-Billion Parameter Language Models Using Model Parallelism*, arXiv, <http://arxiv.org/abs/1909.08053>.
- Singh, D. (2018), Demystifying DCN Topologies: Clos/Fat Trees – Part1. [Online].Packet Pushers. [Retrieved March 10, 2026], <https://packetpushers.net/blog/demystifying-dcn-topologies-clos-fat-trees-part1/>.
- Skrzypek, R. (2025), Polska liderem transformacji cyfrowej w Europie. Centra danych i nowe technologie napędzają gospodarkę. [Online].ITReseller. [Retrieved April 7, 2026], <https://itreseller.pl/polska-liderem-cyfrowej-transformacji-w-europie-centra-danych-i-nowe-technologie-napedzaja-gospodarke/>.
- Volico (2025), How AI is Transforming Data Center Networking. [Online].[Retrieved April 8, 2026], <https://www.volico.com/how-ai-is-transforming-data-center-networking/>.
- Wang, W., Ghobadi, M., Shakeri, K., Zhang, Y., and Hasani, N. (2023) *Rail-only: A Low-Cost High-Performance Network for Training LLMs with Trillion Parameters*, arXiv, <https://arxiv.org/abs/2307.12169>.

- Zhao, S., Wang, R., Zhou, J., Ong, J., Mogul, J.C., and Vahdat, A. (2019) 'Minimal rewiring: efficient live expansion for clos data center networks' *Proceedings of the 16th USENIX Conference on Networked Systems Design and Implementation*, ISBN: 978-1-931971-49-2, 26 February 2019, USA, 221–234.