Universidade Federal de São Carlos
Campus Sorocaba

Fundamental concepts and history

Dataplane programmability and P4

Prof. Fábio Luciano Verdi (verdi@ufscar.br)

Relevant facts

- In 1974, VINTON G. CERF AND ROBERT E. KAHN, publish the paper **“A Protocol for Packet Network Intercommunication”** at IEEE Trans on Comms, Vol Com-22, No 5 May 1974.
- Introduction: “IN THE LAST few years considerable effort has been expended on the design and implementation of packet switching networks”.

Relevant facts

- Several protocols have already been developed for this purpose [8]-[12],[16]. However, these protocols have addressed only the problem of communication on the same network. In this paper we present a protocol design and philosophy that supports the sharing of resources that exist in different packet switching networks.
- Gateway:

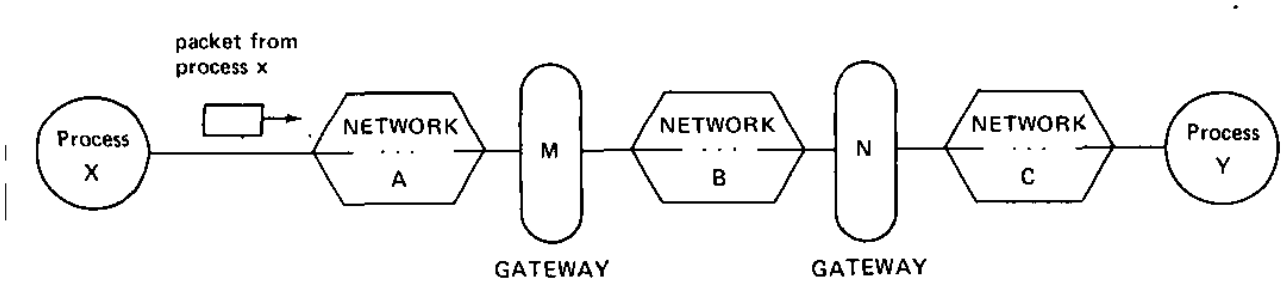


Fig. 2. Three networks interconnected by two GATEWAYS.

In practice, a GATEWAY between two networks may be composed of two halves, each associated with its own network. It is possible to implement each half of a GATEWAY so it need only embed internetwork packets in local packet format or extract them.

Since the GATEWAY must understand the address of the source and destination HOSTS, this information must be available in a standard format in every packet which arrives at the GATEWAY. This information is contained in an *internetwork header* prefixed to the packet by the source HOST.

Relevant facts

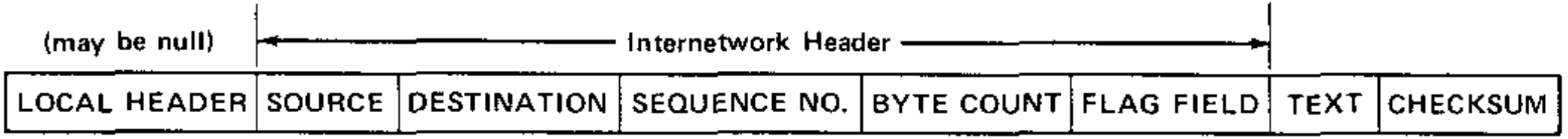


Fig. 3. Internetwork packet format (fields not shown to scale).

The packet format, including the internetwork header, is illustrated in Fig. 3. The source and destination entries uniformly and uniquely identify the address of every HOST in the composite network. Addressing is a subject of considerable complexity which is discussed in greater detail in the next section. The next two entries in the header provide a sequence number and a byte count that may be used to properly sequence the packets upon delivery to the destination and may also enable the GATEWAYS to detect fault conditions affecting the packet. The flag field is used to convey specific control information and is discussed in the section on retransmission and duplicate detection later. The remainder of the packet consists of text for delivery to the destination and a trailing check sum used for end-to-end software verification.

Unless all transmitted packets are legislatively restricted to be small enough to be accepted by every individual network, the GATEWAY may be forced to split a packet into two or more smaller packets. This action is called fragmentation and must be done in such a way that the destination is able to piece together the fragmented packet.

Relevant facts

- TCP ADDRESSING
  - The choice for network identification (8 bits) allows up to 256 distinct networks. This size seems sufficient for the foreseeable future. Similarly, the TCP identifier field permits up to 65 536 distinct TCP's to be addressed, which seems more than sufficient for any given network.

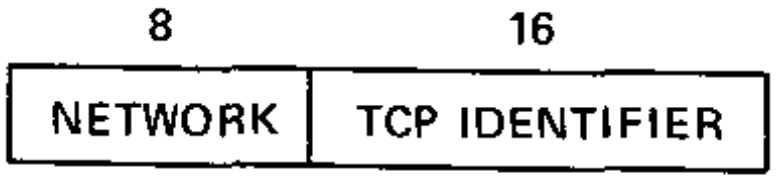


Fig. 4. TCP address.

Relevant facts

- In the Cerf’s paper, nothing is said about congestion control
  - In 1988, Van Jacobson publish the paper Congestion Avoidance and Control, ACM Comp. Communication Review.
  - In October of '86, the Internet had the first of what became a series of 'congestion collapses'. During this period, the data throughput from LBL to UC Berkeley (sites separated by 400 yards and three hops) dropped from 32 Kbps to 40 bps. Mike Karels and I were fascinated by this sudden factor-of-thousand drop in bandwidth and embarked on an investigation of why things had gotten so bad. We wondered, in particular, if the 4.3BSD (Berkeley UNIX) TCP was mis-behaving or if it could be tuned to work better under abysmal network conditions. The answer to both of these questions was “yes”. Since that time, we have put seven new algorithms into the 4BSD TCP:
    - round-trip-time variance estimation
    - exponential retransmit timer backoff
    - slow-start
    - more aggressive receiver ack policy
    - dynamic window sizing on congestion
    - Karn's clamped retransmit backoff
    - fast retransmit (3 duplicated acks)

David D. Clark
Chief Protocol Architect, Internet
1981 - 1989

The Design Philosophy of the DARPA Internet Protocols

David D. Clark*
Massachusetts Institute of Technology
Laboratory for Computer Science
Cambridge, MA. 02139

(Originally published in Proc. SIGCOMM '88 Computer Communication Review Vol. 18, No. 4, August 1988, pp. 106–114)

Abstract

The Internet protocol suite, TCP/IP, was first proposed fifteen years ago. It was developed by the Defense Advanced Research Projects Agency (DARPA), and has been used widely in military and commercial systems. While there have been papers and specifications that describe how the protocols work, it is sometimes difficult to deduce from

architecture into the IP and TCP layers. This seems basic to the design, but was also not a part of the original proposal. These changes in the Internet design arose through the repeated pattern of implementation and testing that occurred before the standards were set.

The Internet architecture is still evol

Relevant facts

- The Cerf´s paper did not mention the term Internet Protocol (IP)
  - INTERNET PROTOCOL SPECIFICATION September 1981: RFC 791, Jon Postel
  - TRANSMISSION CONTROL PROTOCOL September 1981: RFC 793, Jon Postel
- How about UDP?
  - David Clark in his paper THE DESIGN PHILOSOPHY OF THE DARPA INTERNET PROTOCOLS, ACM Sigcomm 1988 writes:
    - The initial concept of TCP was that it could be general enough to support any needed type of service
    - The first example of a service outside the range of TCP was support for XNET
    - Another service which did not fit TCP was real time delivery of digitized speech, which was needed to support the teleconferencing aspect of command and control applications
    - It was thus decided, fairly early in the development of the Internet architecture, that more than one transport service would be required, and the architecture must be prepared to tolerate simultaneously transports which wish to constrain reliability, delay, or bandwidth at a minimum. This goal caused TCP and IP, which originally had been a single protocol in the architecture, to be separated into two layers.
  - RFC 768 defines UDP: Jon Postel in August 1980.

Internet architecture was designed to be….

- Simple
- Dumb
- Distributed

But…

- Hard to manage
- Became big
- Full of failures
- Thousands of standardizations

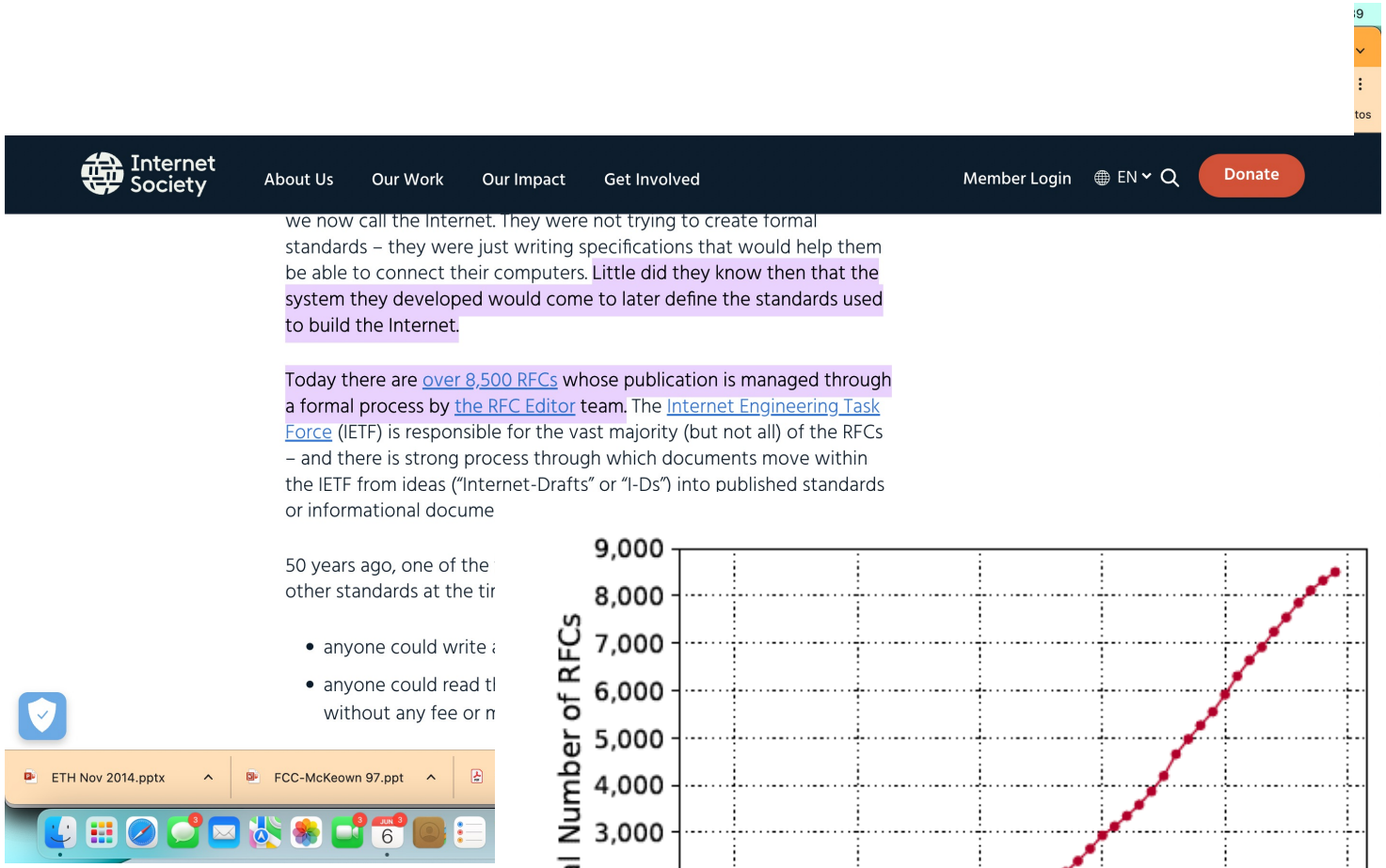
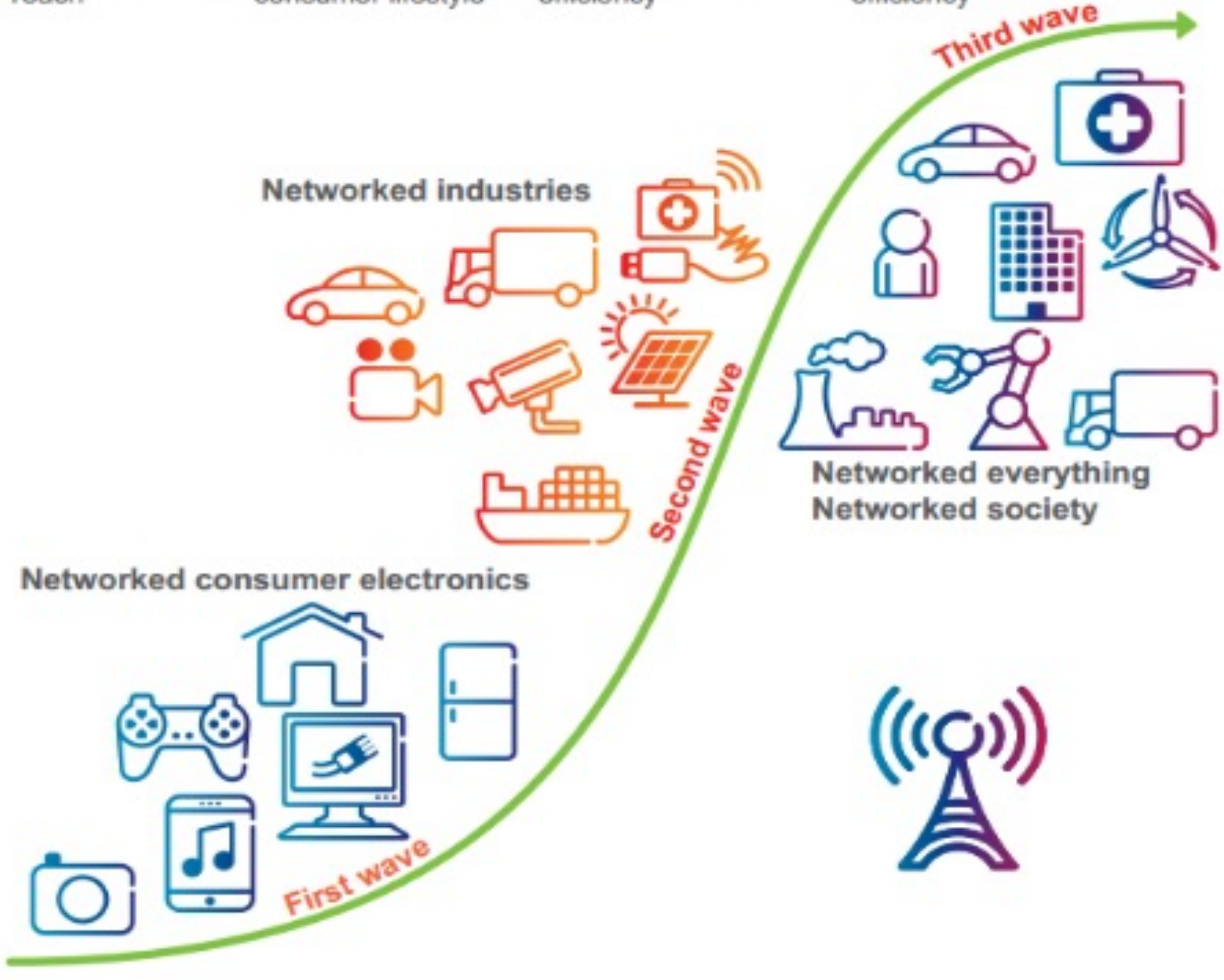
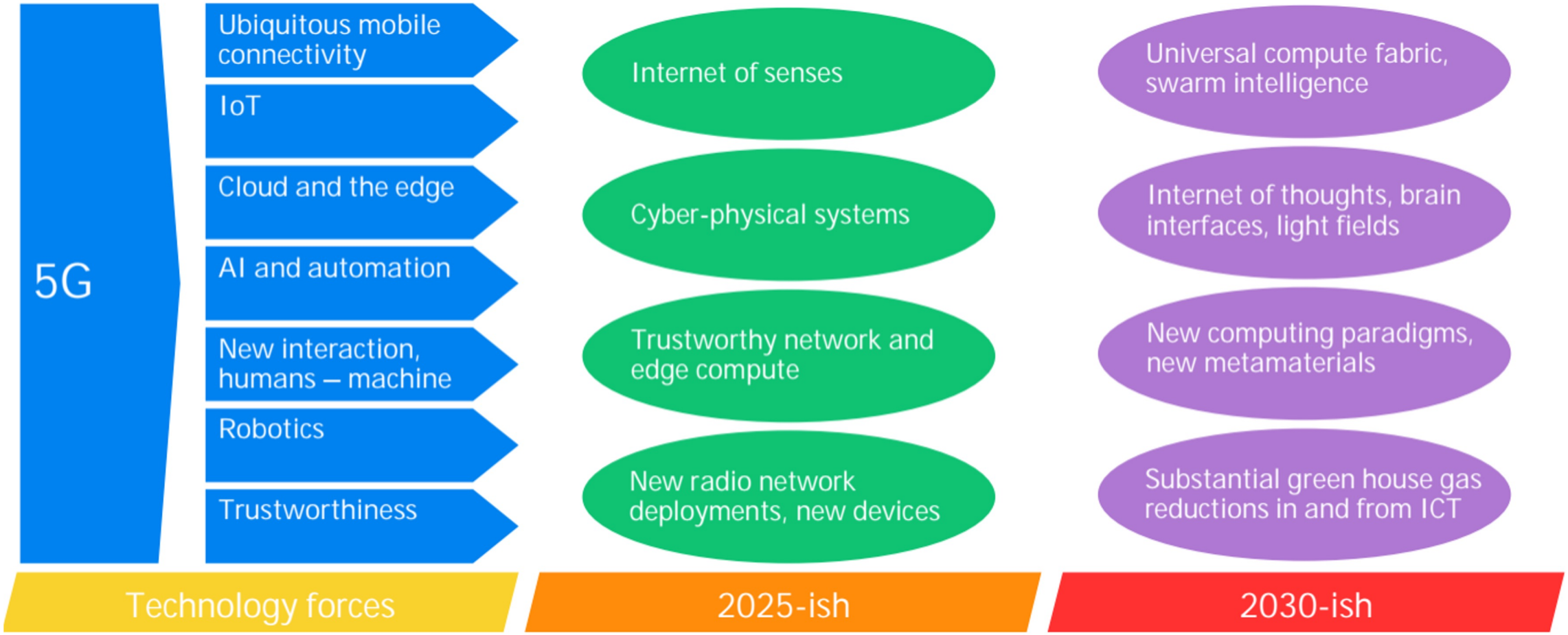


FIGURE 1: Cumulative number of RFCs.


Third wave
Networked industries
Second wave
Networked everything
Networked society
Networked consumer electronics
First wave

From 5G to 2030



Source: Towards a connected intelligent future. Dr. Magnus Frodigh. Head of Ericsson Research, VP. Ericsson Research. 2019-03-26. Ericsson. http://www.6gsummit.com/wp-content/uploads/2019/04/Day3_Session2_Frodigh_Ericsson.pdf

One size won't fit all!

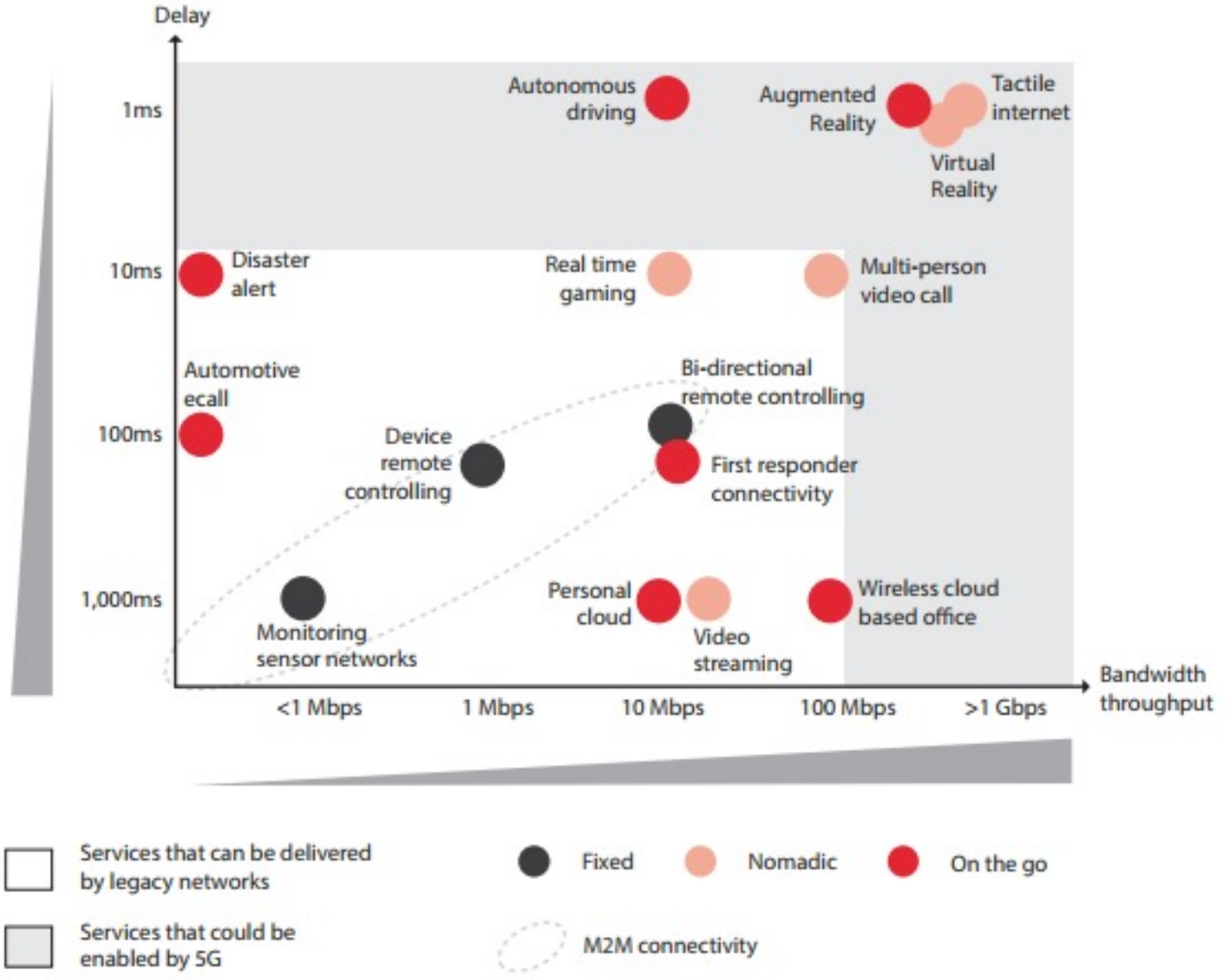
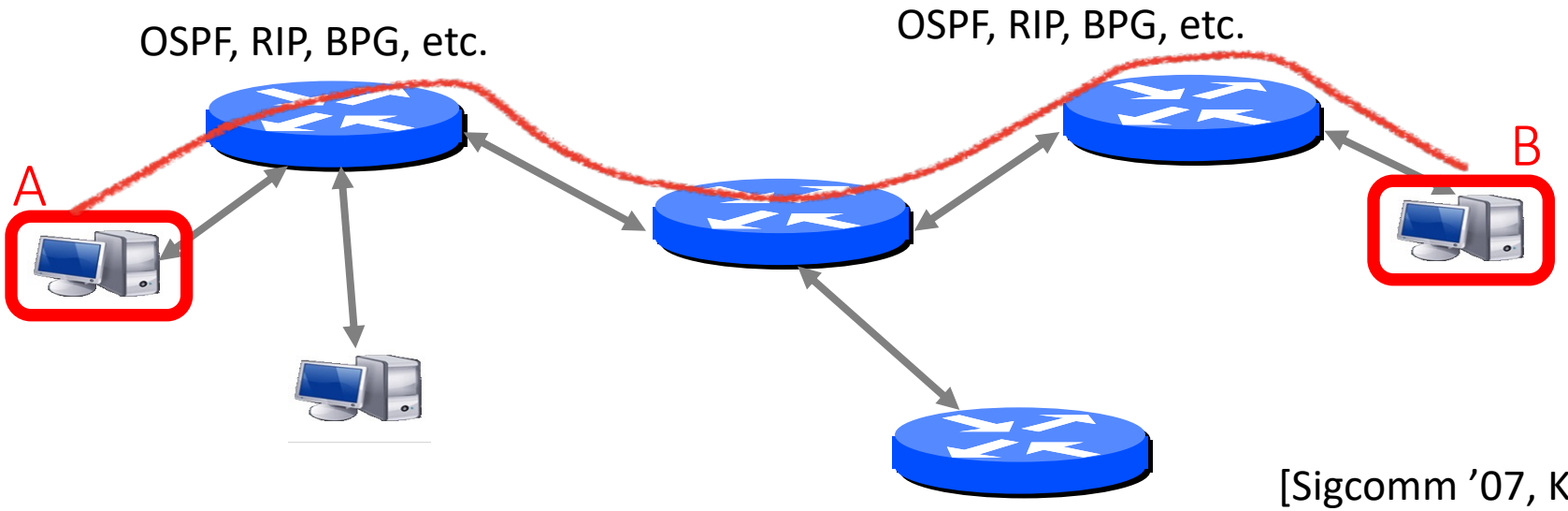


Figure 1: Bandwidth and latency requirements of potential 5G use cases

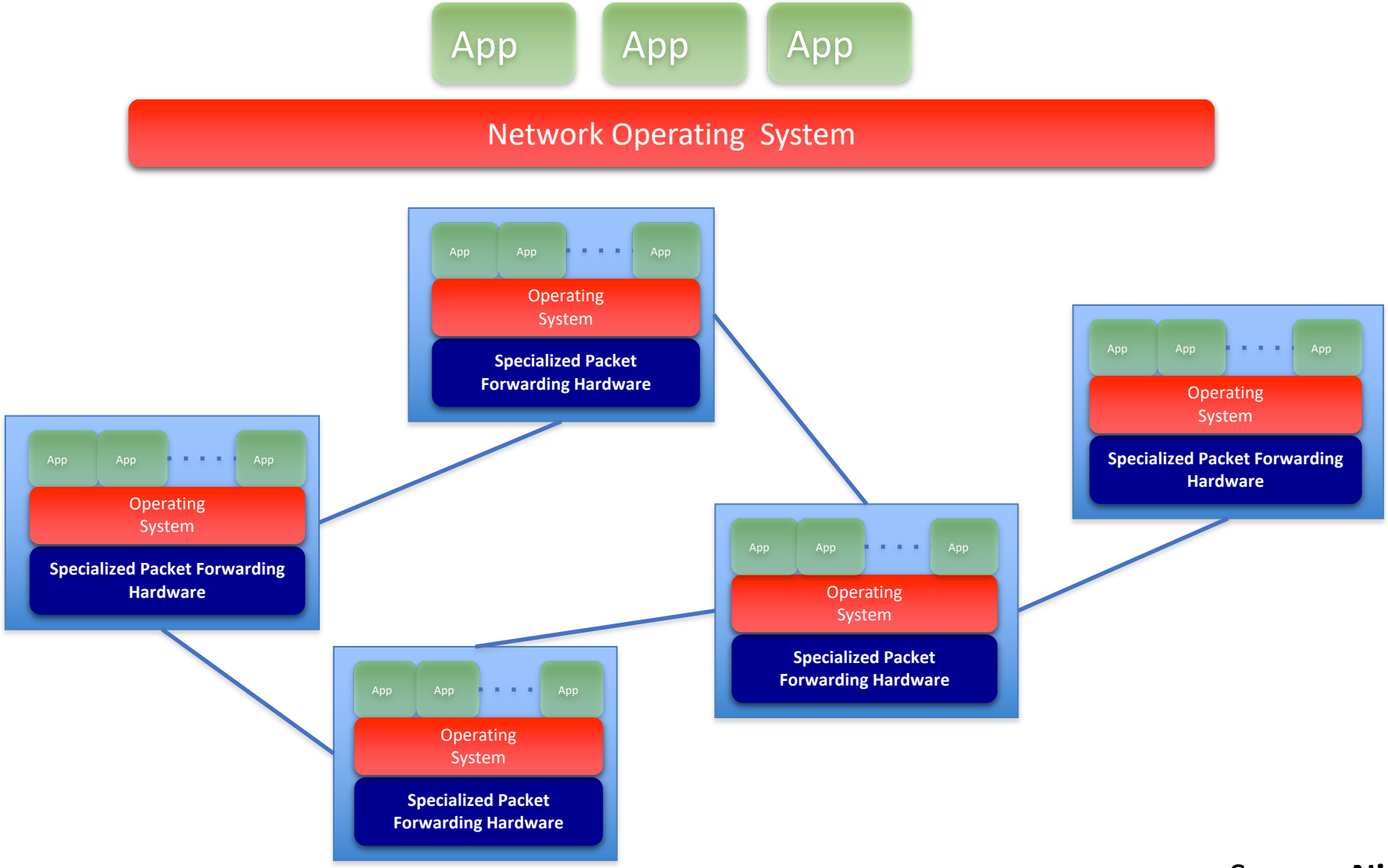
Traditionally…

- Lack of competition
- Closed architectures, proprietary, lock in



[Sigcomm ’07, Kyoto]

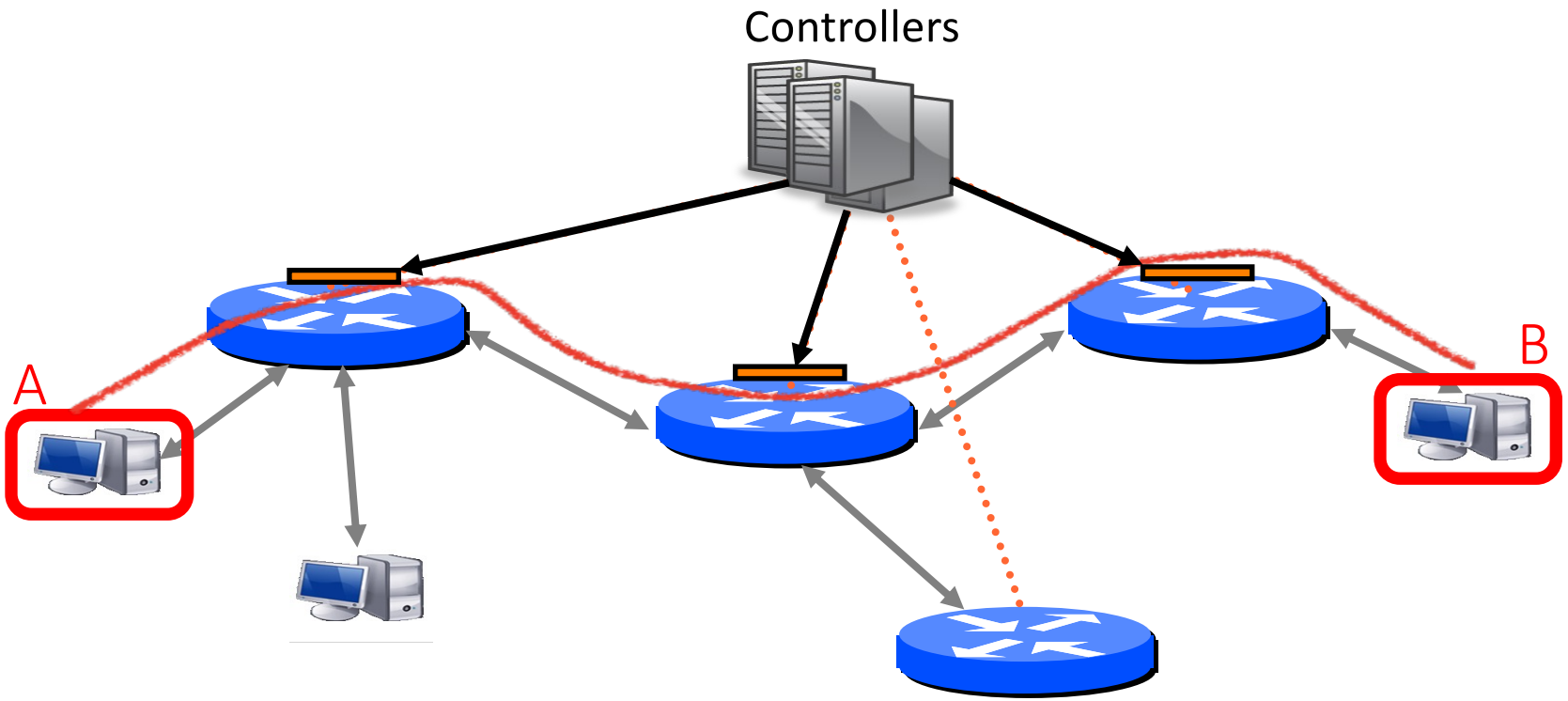
Traditionally➔Software Defined Networking



Source: **Nick McKeown**

Software Defined Networking (SDN)

Separating Control and Data Planes

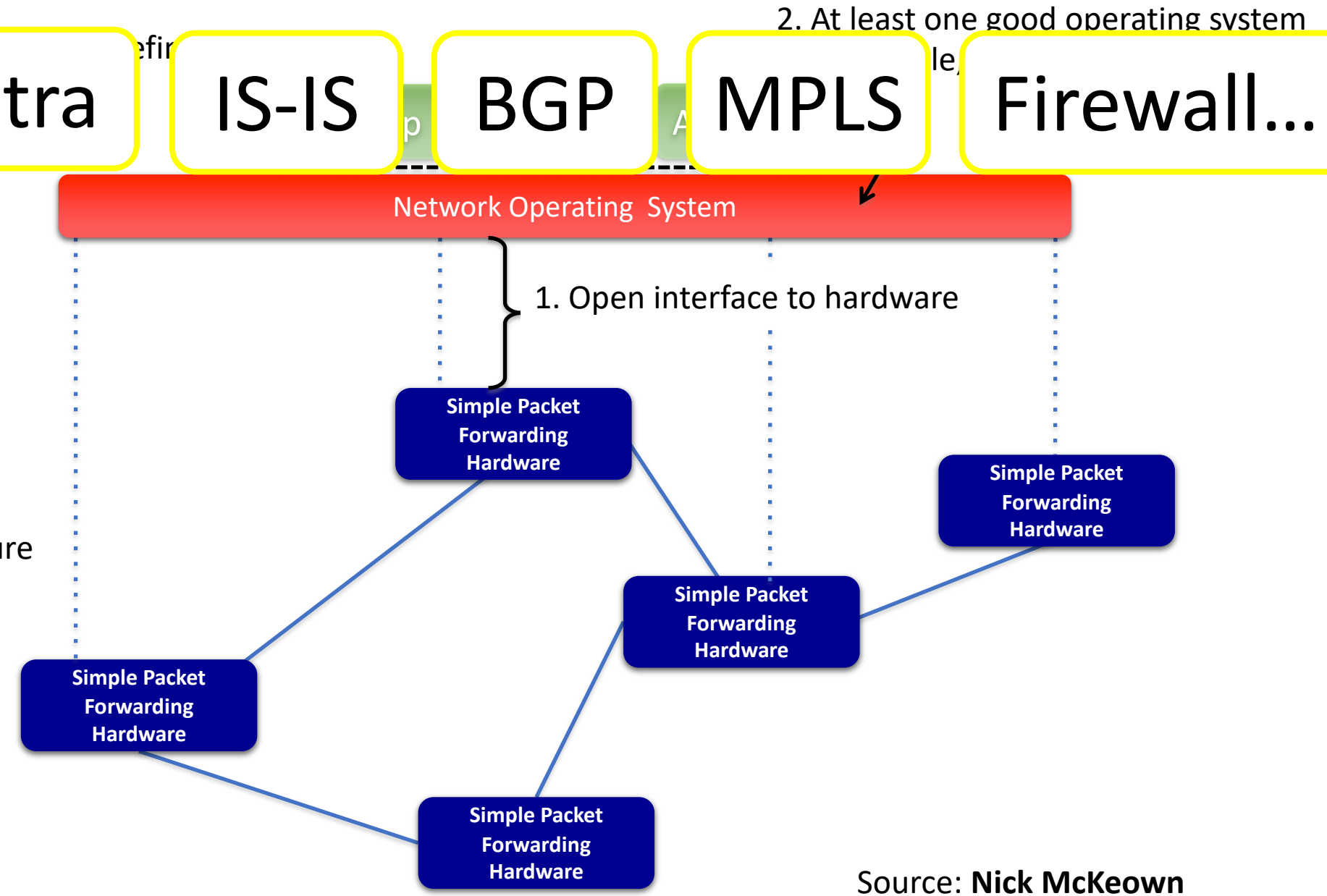


[Sigcomm ’07, Kyoto]

A network in which the control plane is physically separate from the forwarding plane, and a single control plane controls several forwarding devices *(Nick McKeown’s 2013 presentation entitled Software Defined Networking)*

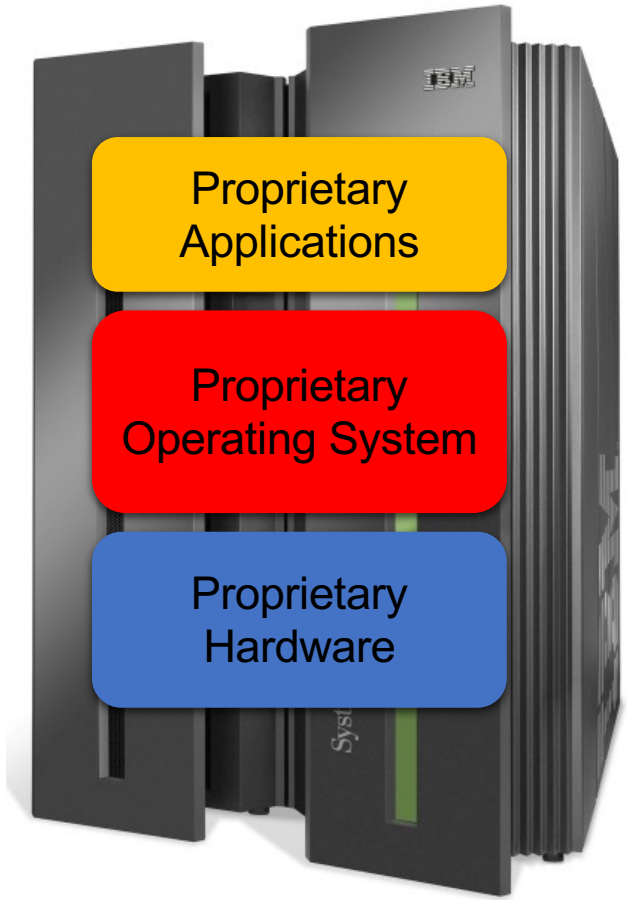
Software Defined Networking (SDN)

- Raise of Linux
- Rise of whitebox servers
- Open source
- Open interfaces
- Novel applications
- Space for new players
- Less cost?
- Towards a 100% open architecture



Source: **Nick McKeown**

Computer Industry



Disaggregation

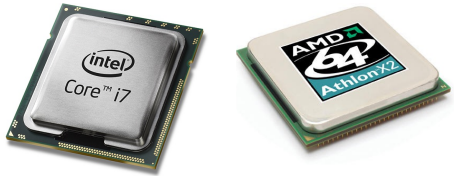
App App App App App App App App App App

Open Interface

Windows (OS) or Linux or Mac OS

Open Interface

Microprocessor



Source: **Nick McKeown**

Servers

Applications
OS
CPU

Legacy

Applications
Linux
x86

Whitebox

Network switches

Applications
OS
CPU + ASIC

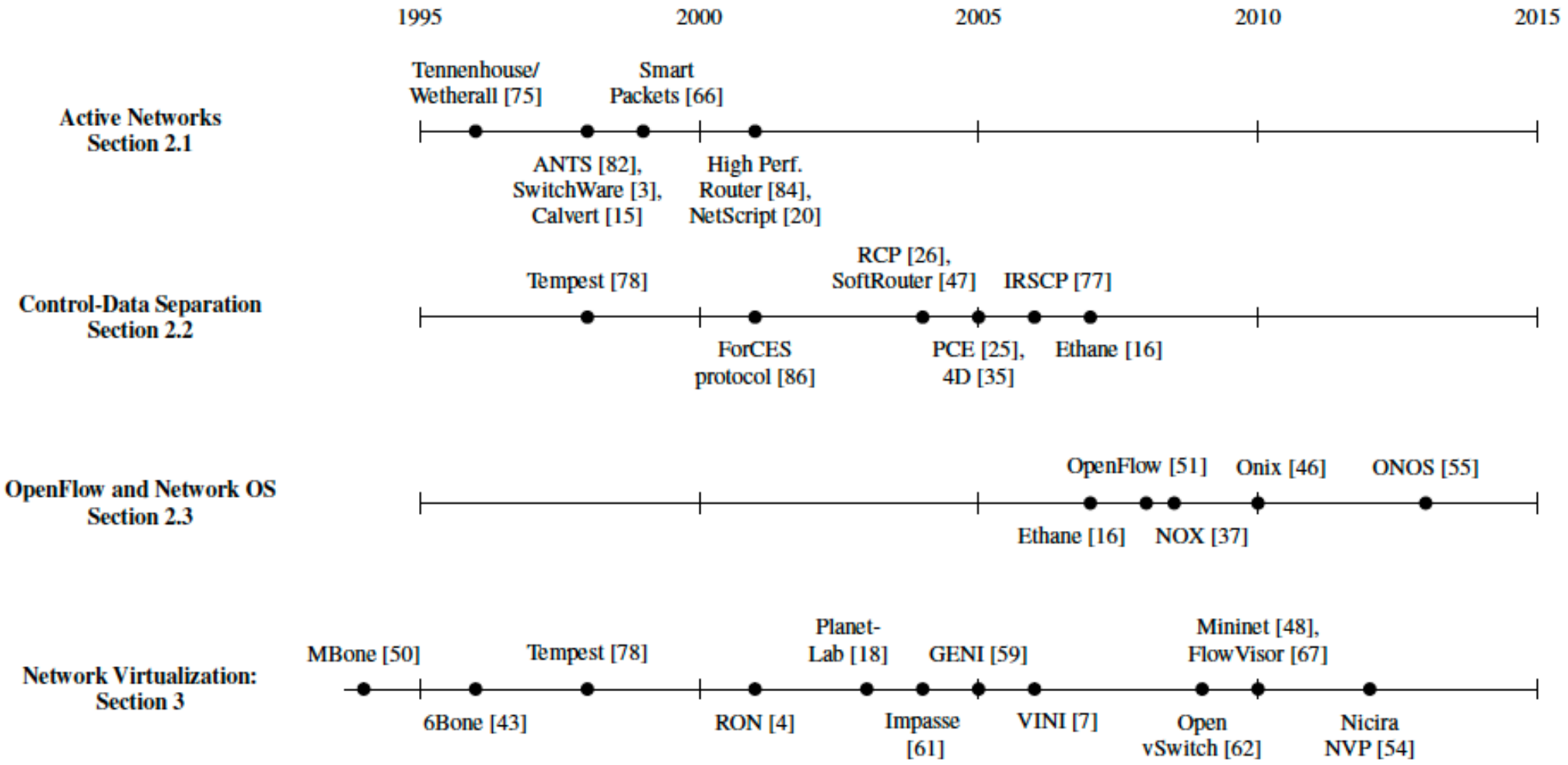
Legacy

Control Programs
Linux
x86 + ASIC

Whitebox

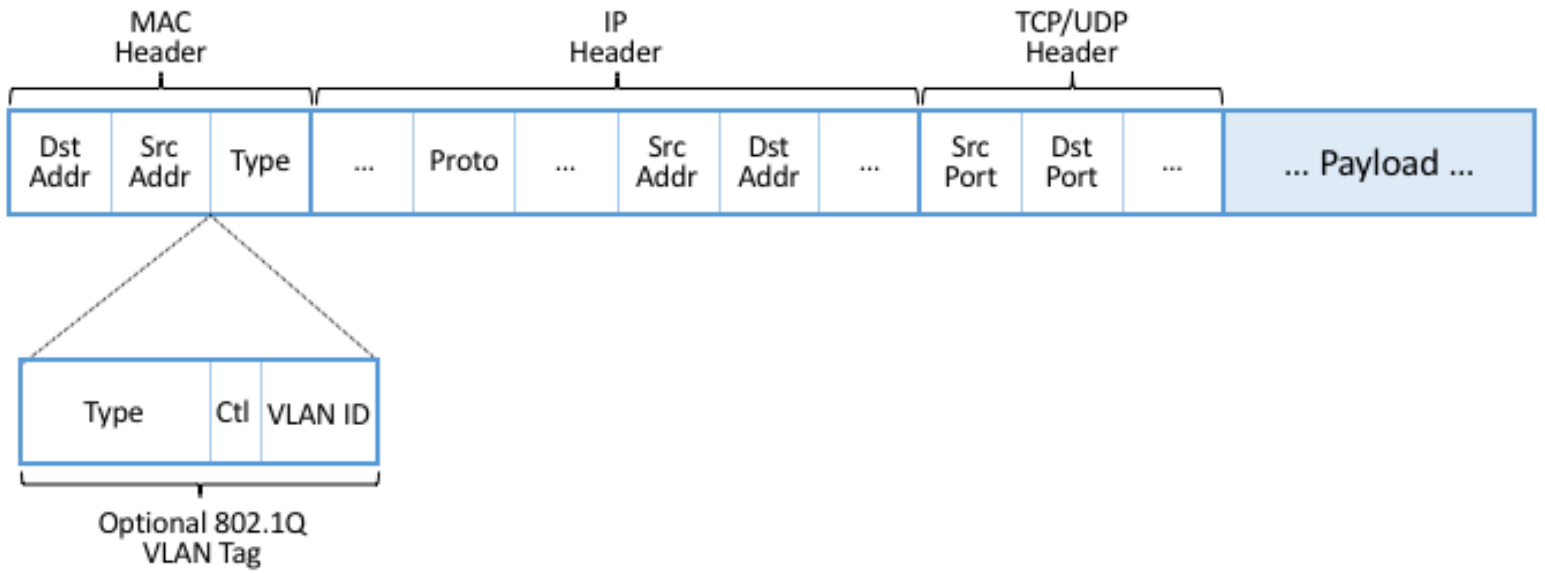
Source: **Nick McKeown**

A bit of history



OpenFlow evolution

OF Version	Release Date	Match fields
1.0	Dec 2009	12
1.1	Feb 2011	15
1.2	Dec 2011	36
1.3	Jun 2012	40
1.4	Oct 2013	41
1.5	Dec 2014	44



OpenFlow

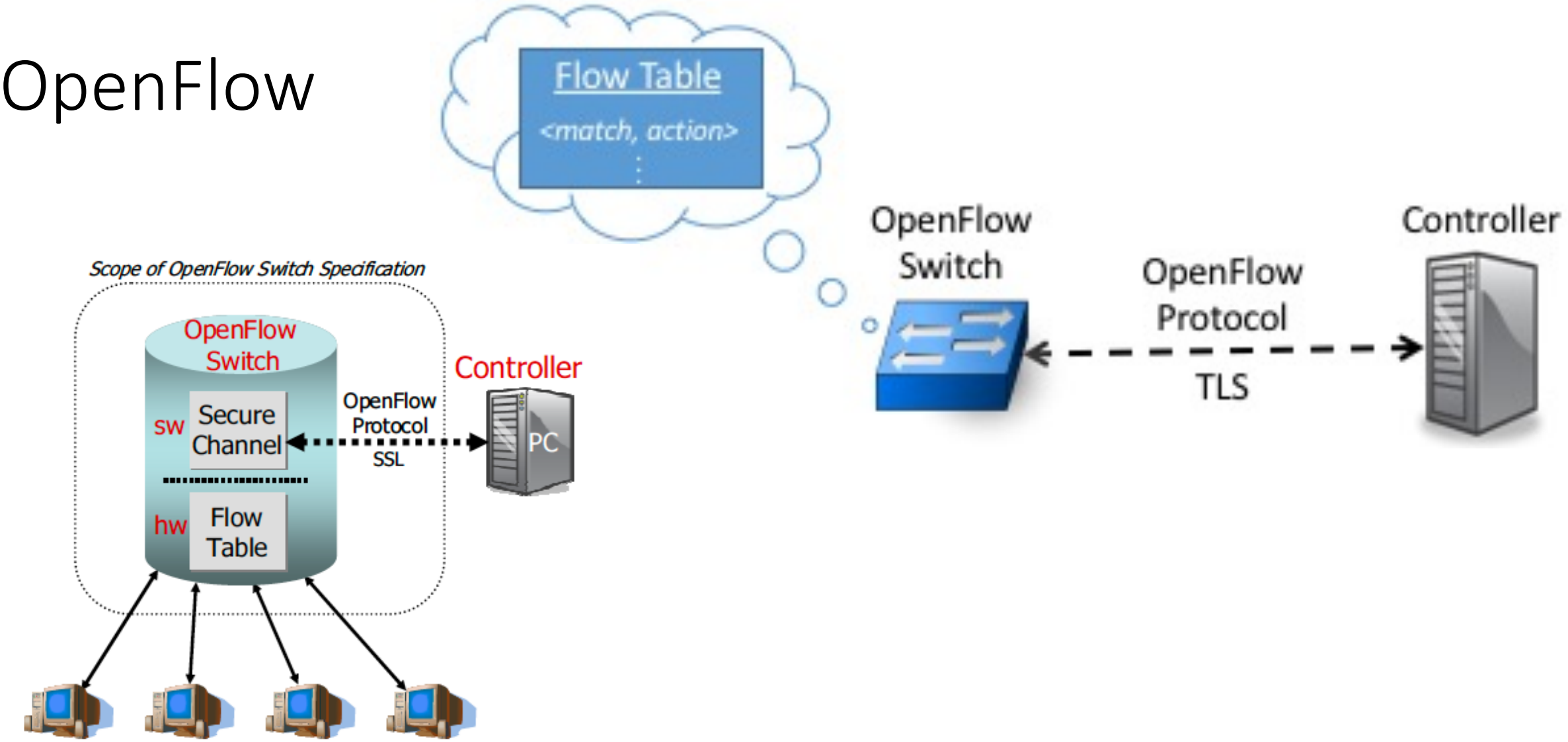


Figure 1: Idealized OpenFlow Switch. The Flow Table is controlled by a remote controller via the Secure Channel.

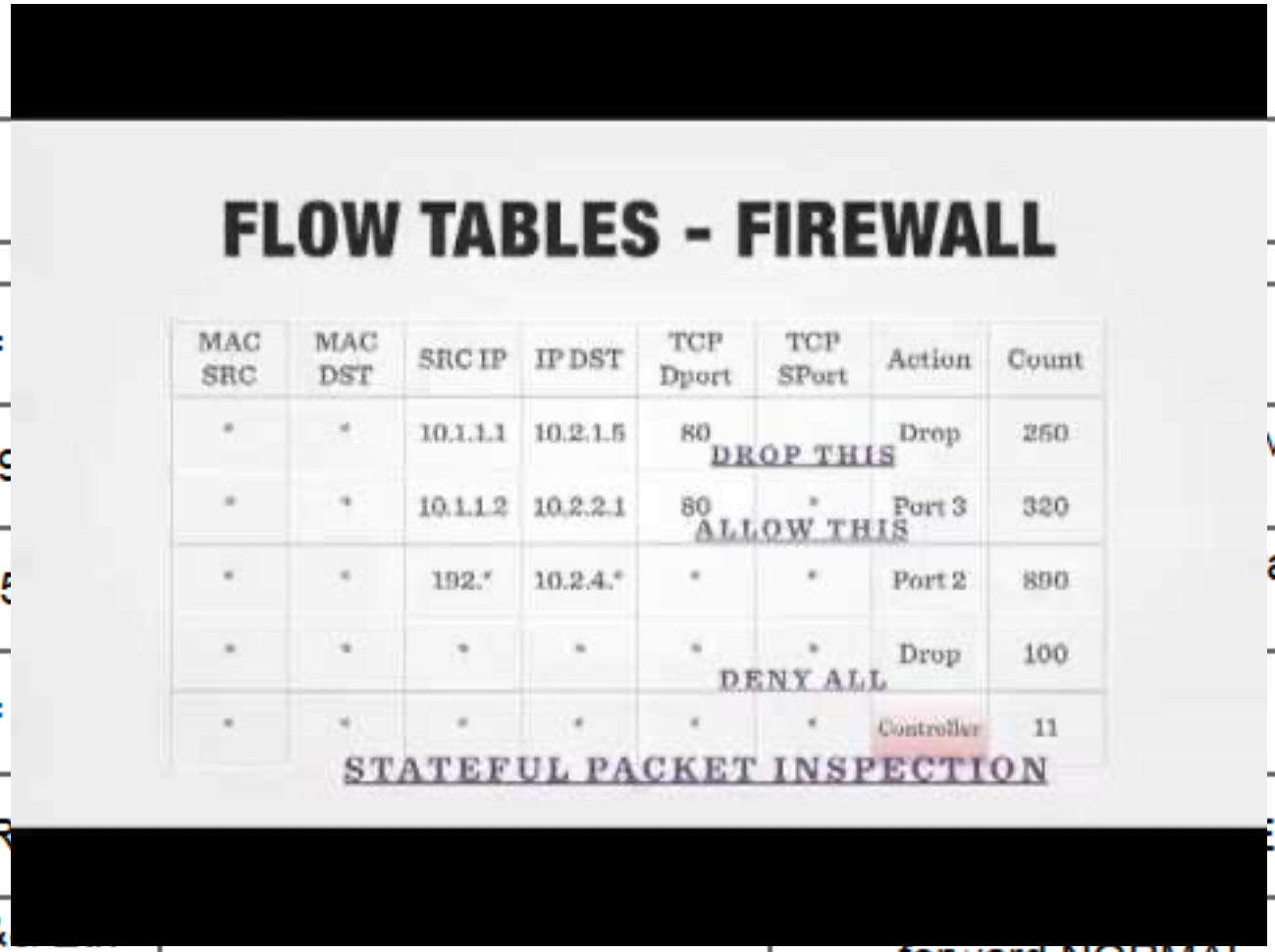
Source: N. McKeown, et. al. OpenFlow: Enabling Innovation in Campus Networks. SIGCOMM CCR, March 2008.

Flow Table

Header Fields	Counters	Actions	Priority
Ingress Port Ethernet Source Addr Ethernet Dest Addr Ethernet Type VLAN id VLAN Priority IP Source Addr IP Dest Addr IP Protocol IP ToS ICMP type ICMP code	**Per Flow Counters** Received Packets Received Bytes Duration seconds Duration nanosecconds	Forward (All, Controller, Local, Table, IN_port, Port# Normal, Flood) Enqueue Drop Modify-Field	

Flow Table

Header Fields			Priority
If ingress port ==			32768
if IP_addr == 129.79			32768
if Eth Addr == 00:45			32768
if ingress port ==			32768
if Eth Type == AR			32768
If ingress port == 2 & Type == ARP		forward NORMAL	40000

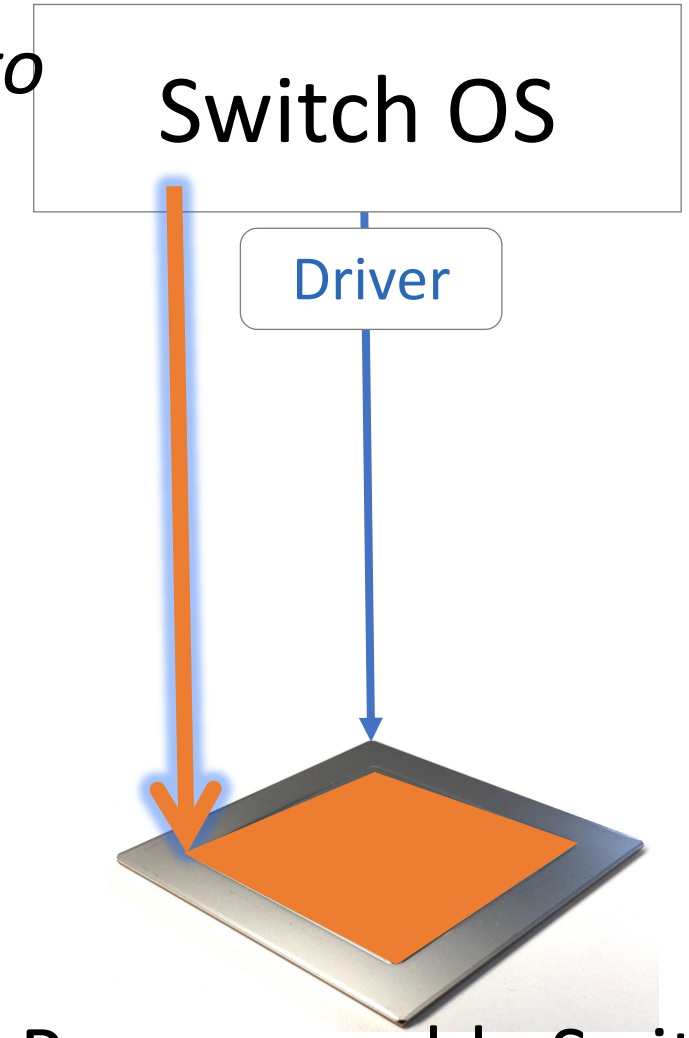


Network systems are starting to be programmed “top-down”

“Tell/program the network how to process packets”

```
table int_table {
  reads {
    ip.protocol;
  }
  actions {
    export_queue_latency;
  }
}
```

```
action export_queue_latency (sw_id) {
  add_header(int_header);
  modify_field(int_header.kind, TCP_OPTION_INT);
  modify_field(int_header.len, TCP_OPTION_INT_LEN);
  modify_field(int_header.sw_id, sw_id);
  modify_field(int_header.q_latency,
               intrinsic_metadata.deq_timedelta);
  add_to_field(tcp.dataOffset, 2);
  add_to_field(ipv4.totalLen, 8);
  subtract_from_field(ingress_metadata.tcpLength,
                      12);
}
```



Programmable Switch

Source: Adapted from Nick Mckeown.

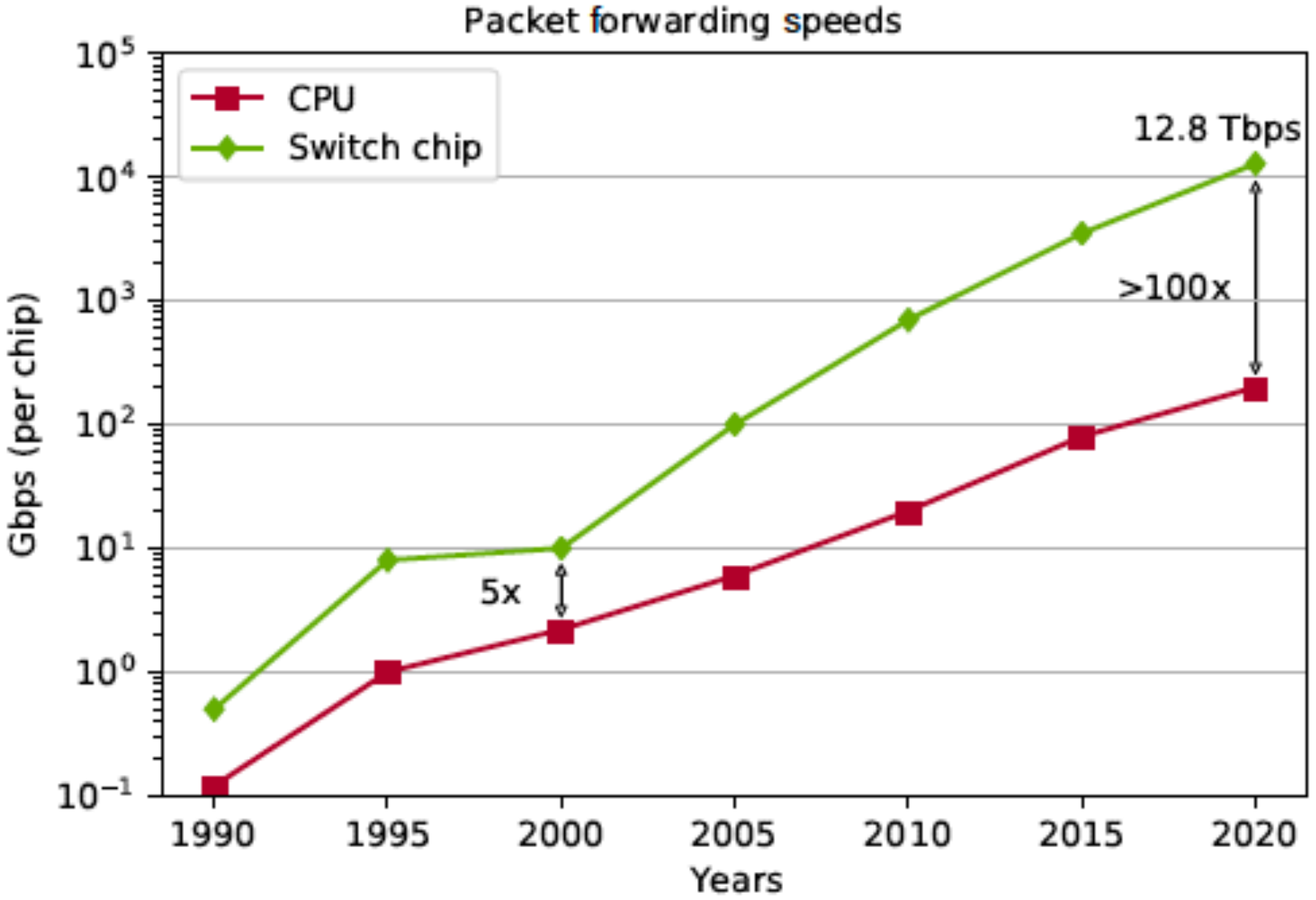


FIGURE 4: Evolution of the packet forwarding speeds of the general-purpose CPU and the switch chip (reproduced from [53]).

Source: An Exhaustive Survey on P4 Programmable Data Plane Switches: Taxonomy, Applications, Challenges, and Future Trends

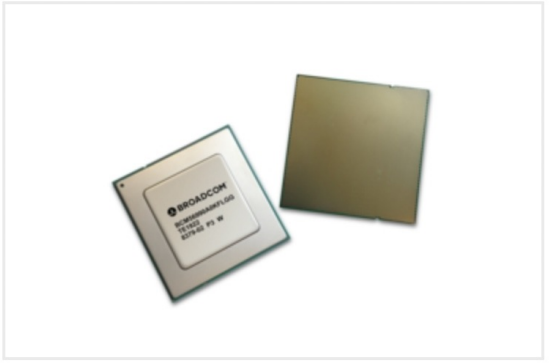
Tomahawk4 / BCM56990 Series

Contact Sales Request Info

25.6 Tb/s and 12.8Tb/s StrataXGS® Tomahawk® 4 Ethernet Switch Series

Overview | Specifications | Documentation

The Broadcom® BCM56990 family is a class of high-radix, high-bandwidth network switching devices supporting up to 64 × 400GbE, 128 × 200GbE, 256 × 100GbE, 256 × 40GbE, 256 × 25GbE, or 256 × 10GbE ports. The device family features a maximum of 64 integrated Blackhawk7 (50G-PAM4) or 32 Osprey (100G-PAM4) SerDes cores, and associated PCS for native support of numerous physical connectivity options, enabling a broad range of media, speed, and reach. The BCM56990 delivers high-bandwidth, glueless network connectivity up to 25.6 Tb/s on a single chip.

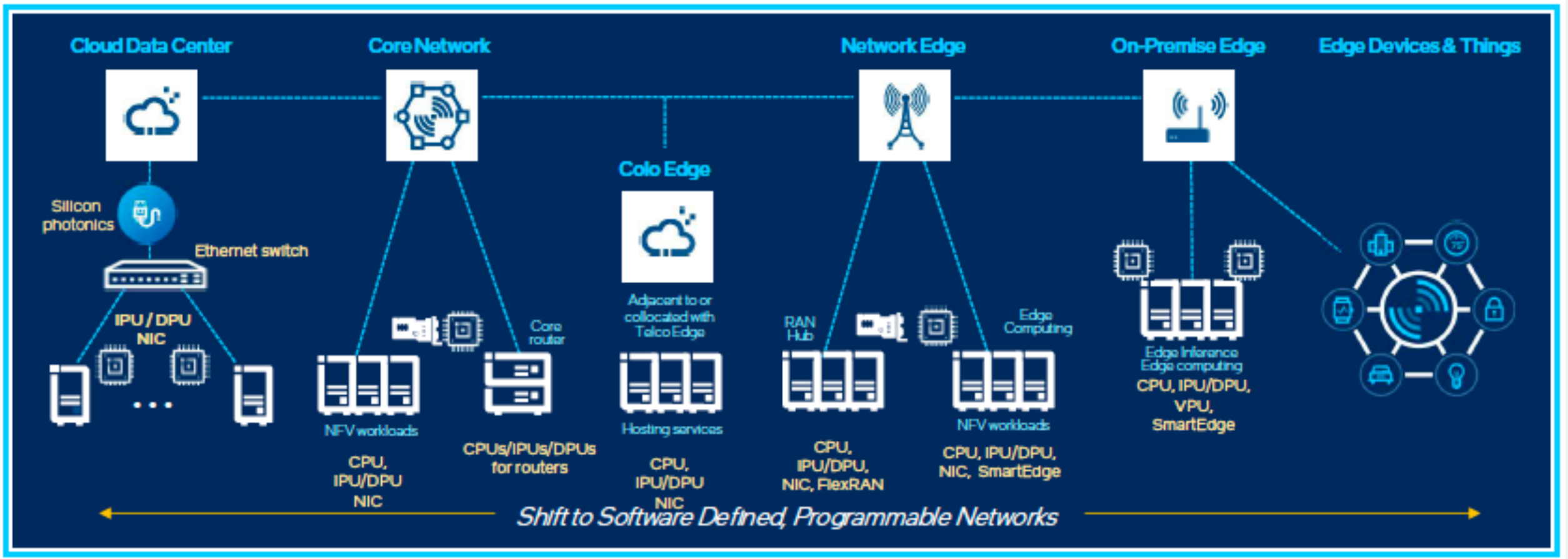


The BCM56990 is a high-performance and high-capacity device designed to meet the requirements for next-generation data center and cloud computing environments. The BCM56990 architecture delivers complete L2 and L3 switching, routing and tunneling capabilities at line rate and maximum port density, with low power and latency. Software compatibility is maintained

The Intel® Tofino™ 3 Intelligent Fabric Processor (IFP) provides P4-programmability, accelerates AI packet processing, provides hyper-scaler program-optimized power consumption, and delivers real-time in-band network telemetry (INT) for workloads spanning the entire Edge-to-Cloud spectrum.

Download do PDF

Cloud to Edge Programmability



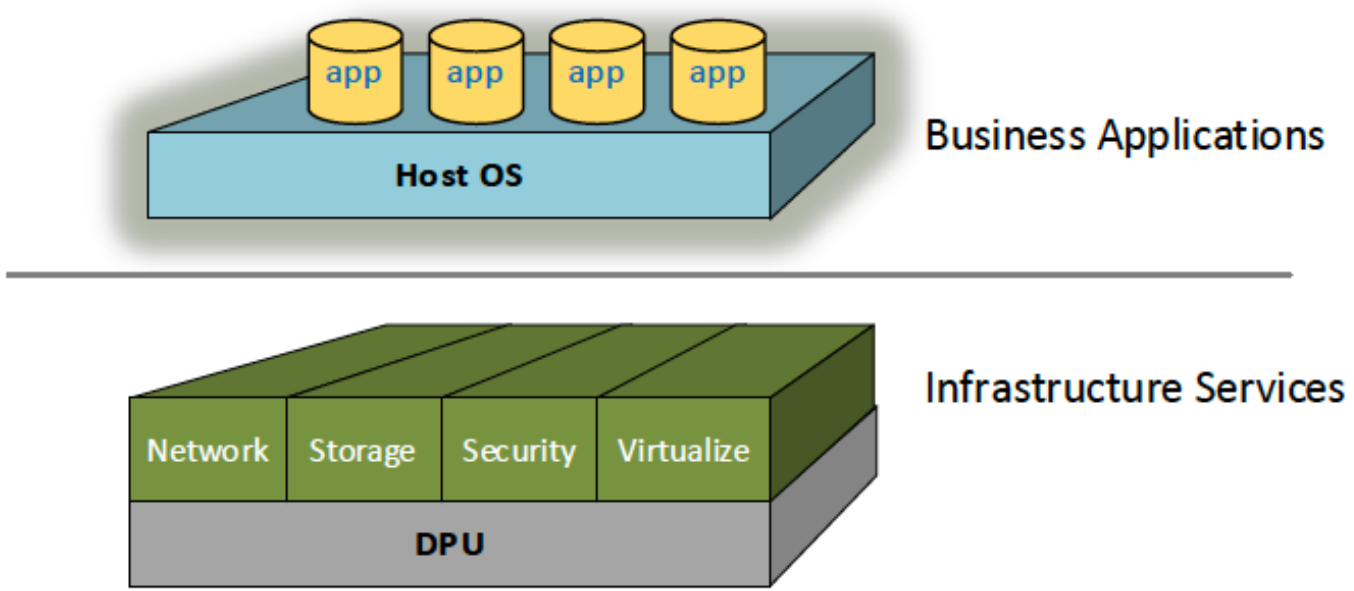
Separating Business Apps from Infrastructure

- **Business Apps** run on the Node
- **Infrastructure Apps** are Services running on the DPU
  - ❑ Network
  - ❑ Storage
  - ❑ Security
  - ❑ Virtualization
- Why move Infrastructure off the node?

  - "30% of CPU cores are being used for datacenter infrastructure needs"

  -"It would take 125 cores to run all the Security, Network, and Storage offloads at 125Gbps"

  Jensen Huang, NVIDIA CEO, @ 2020 GTC Keynote

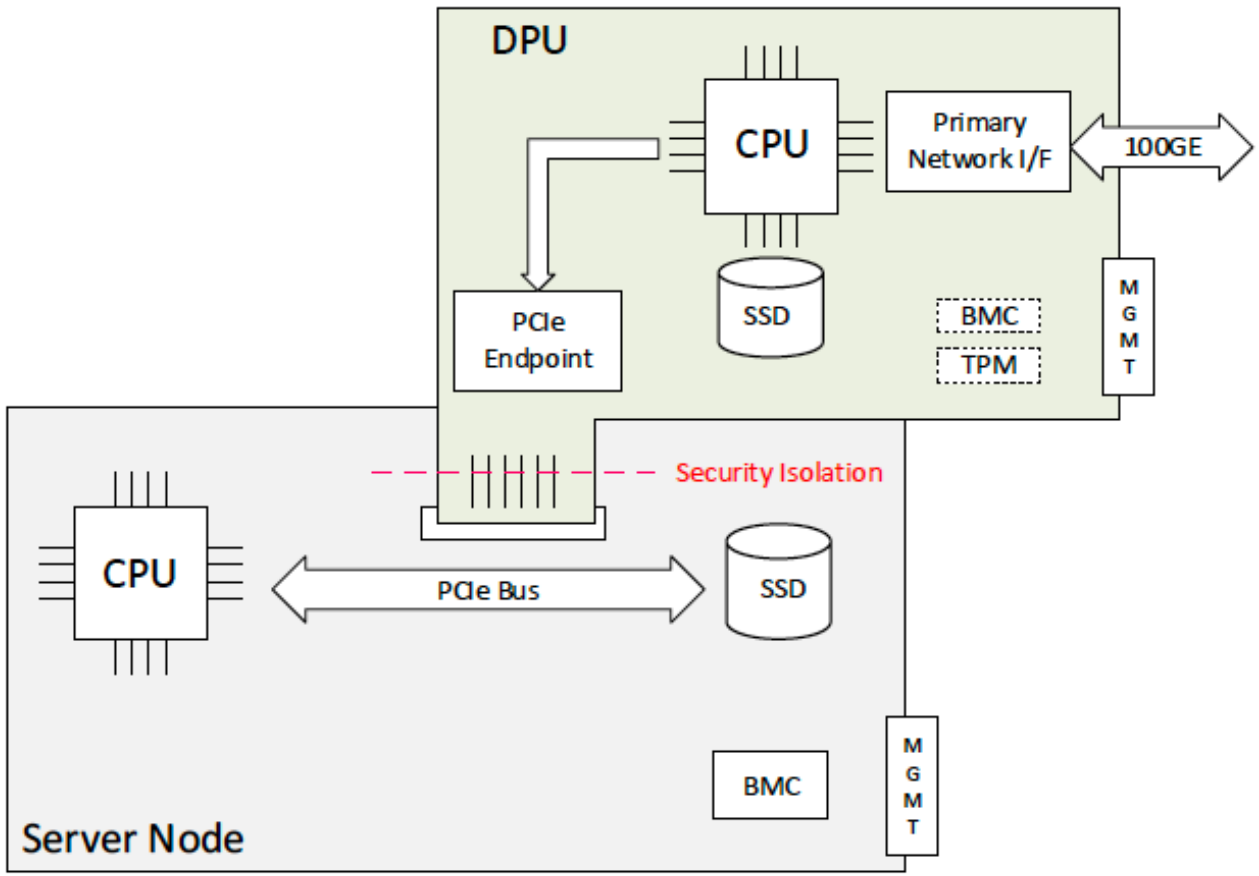


Source: **F5**

The DPU as a “Server” plugged into the Server

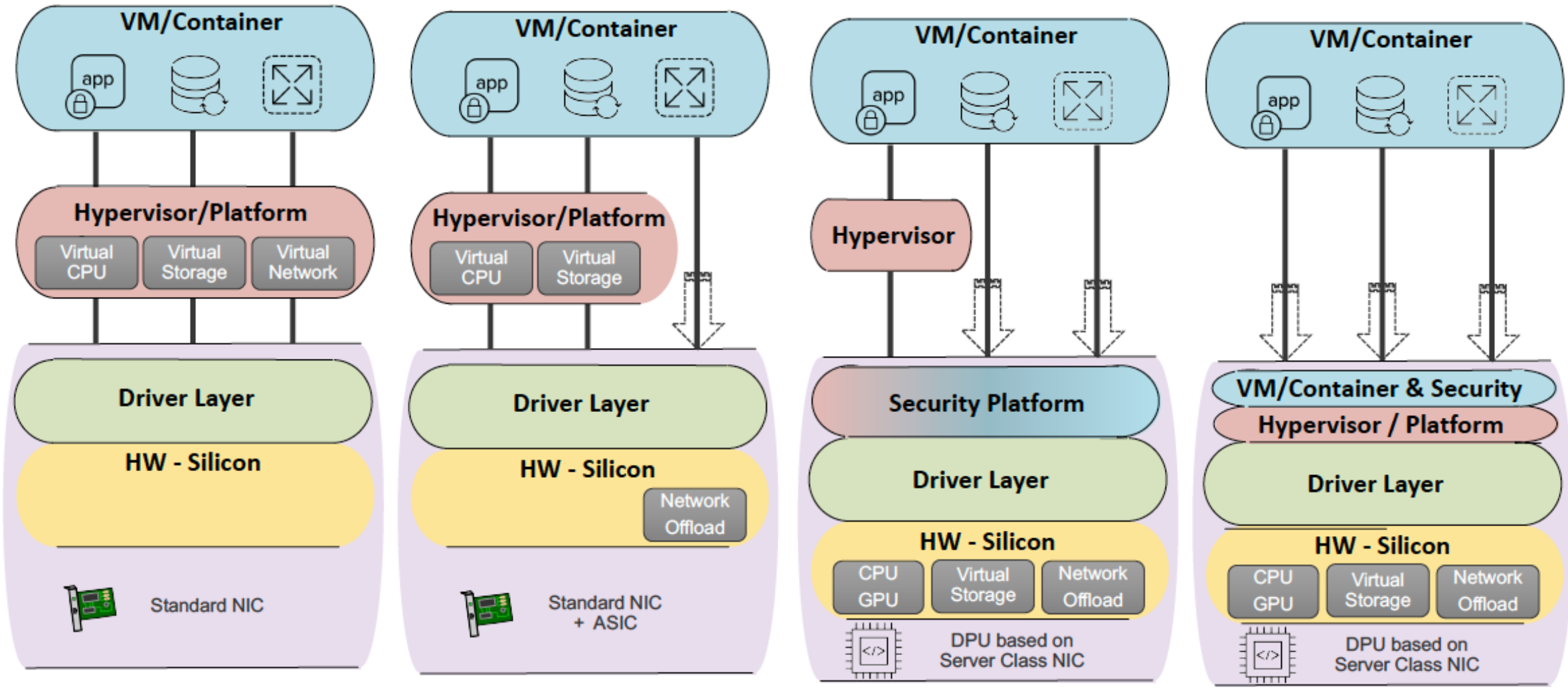
DPU DEFINING TRAITS

- General purpose CPU with significant compute capabilities
- Boots to a general-purpose OS like Linux
- Mix of domain specific HW accelerators
- Strict security isolation from hosting system
- High performance network interface
- Unique Identity on primary network interface
- Independent out-of-band management capability
- Capable of hosting complete infrastructure services



Source: **F5**

Evolution of the Infrastructure Stack leads to DPUs



Source: **F5**

Intel® Infrastructure Processing Unit (Intel® IPU) SoC (Codename: Mount Evans)

Overview

Key features on Mount Evans

- **Hyperscale ready** Co-designed with a top cloud provider with integrated learnings from multiple generations of Intel® FPGA SmartNICs to deliver high performance under real workload with security and isolation from the ground up.
- **Technology Innovation** Highly programmable packet processing engine, NVM Express* storage interface scaled up from Intel® Optane™ Technology, next generation reliable transport, advanced crypto, and compression acceleration.
- **Software** Software/Hardware/Accel co-

Further Reading/watching

- N. Feamster, J. Rexford, and E. Zegura. The Road to SDN: An Intellectual History of Programmable Networks. SIGCOMM CCR, April 2014.
- S. Shenker. The Future of Networking and the Past of Protocols. Open Networking Summit, October 2011.
- N. McKeown, et. al. OpenFlow: Enabling Innovation in Campus Networks. SIGCOMM CCR, March 2008.
- DPU Disruption of Today's Infrastructure Paradigm. https://youtu.be/HwvP3Doyxdc