

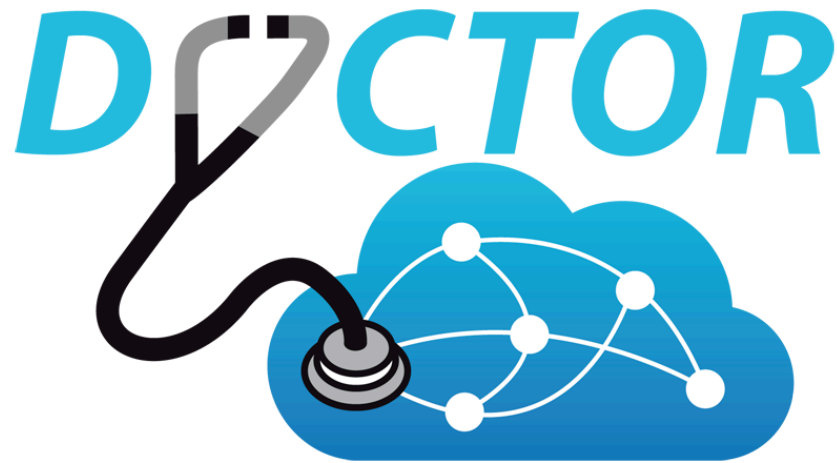
“MICRO-SERVICING” NDN

AND ITS VALUE FOR IMPROVED SECURITY, SCALABILITY AND INTEGRATION

Olivier Festor



mardi 21 avril 2020



<http://www.doctor-project.org>



THALES



Outline

- Background
 - Content and softwarization convergence in and through networks
 - Information Centric Networks - a nice use case and a possible solution
- Softwarizing NDN
 - Functional decomposition NDN
 - Virtualization and orchestration
 - Management and performance
- Integrating NDN in legacy IP/HTTP networks
 - A gateway/islands approach
 - Orchestrated delivery
 - Measurements
- Conclusion

“Content is King”

Bill Gates, 1996

« Content is where I expect much of the real money will be made on the Internet, just as it was in broadcasting »



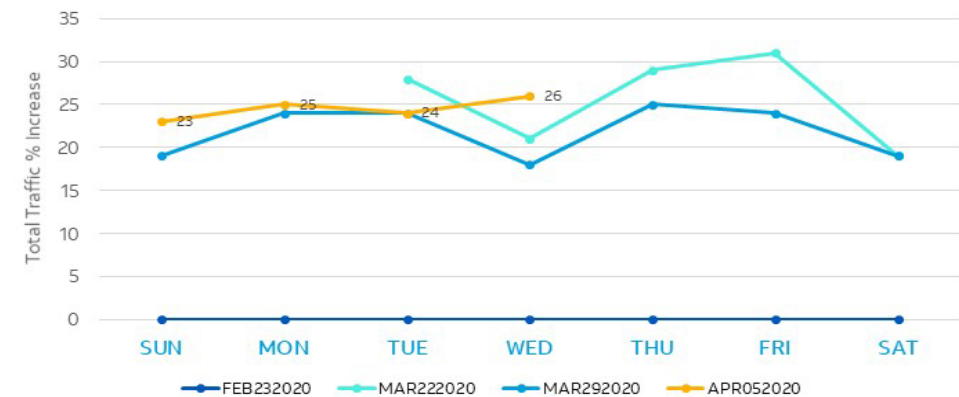
Content & exchange growth

| Data usage type: | Increase compared to typical day | Growth rate change week over week, (peak hour usage) |
|------------------|----------------------------------|--|
| Gaming | 102% | 1.6% |
| VPN | 40% | 4% |
| Video | 33% | 3% |
| Downloads | 27% | -25% |
| Web | 24% | Flat |

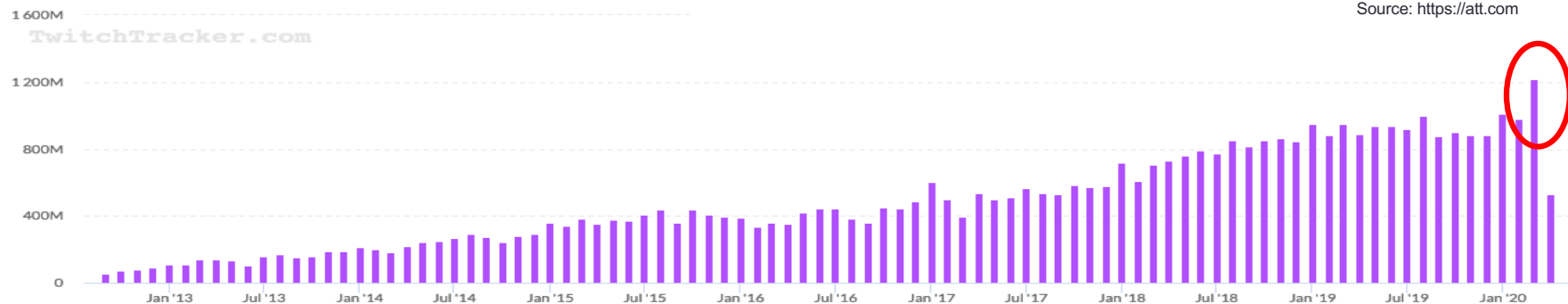
Source: <https://verizon.com> (13/04/2020)

Core Network Traffic

Business, home broadband and wireless usage



Source: <https://att.com>



Source: <https://twitchtracker.com/>



“Software
is eating
the world !”

*Marc Andreessen
(Wall Street Journal,
August 20, 2011)*

Network Softwarization is maturing

- Need for enabling/testing/deploying new protocols/paradigms
- Time to Market is a major driver
- Devops practices become common in the networking world (e.g. GITOps)
- Technology, communities & business maturity increases
 - Programmable switches & equipment
 - Virtualization technologies & solutions
- Standards maturity
 - SDN/NFV initiatives, TOSCA, ETSI MANO, ...

Our interest : enable incremental deployment and interoperability, scalability and fine grained dynamic management of an ICN framework

INFORMATION CENTRIC NETWORKS

One possible solution

Information/Content/Named-Data Networking Principles

- Content is the core element of concern and operation (Information-Centric Networking)
 - Content is named & represents the unit of processing & exchange
 - The network works only with interfaces, names, content units & caches
- Foreseen native advantages
 - Caching,
 - Multipath,
 - Security, ...

Networking Named Content

Van Jacobson Diana K. Smetters James D. Thornton Michael F. Plass
Nicholas H. Briggs Rebecca L. Braynard
Palo Alto Research Center
Palo Alto, CA, USA
{van,smetters,jthornton,plass,briggs,rbraynar}@parc.com

ABSTRACT

Network use has evolved to be dominated by content distribution and retrieval, while networking technology still speaks only of connections between hosts. Accessing content and services requires mapping from the what that users care about to the network's where. We present *Content-Centric Networking* (CCN) which treats content as a primitive – decoupling location from identity, security and access, and retrieving content by name. Using new approaches to routing named content, derived heavily from IP, we can simultaneously achieve scalability, security and performance. We implemented our architecture's basic features and demonstrate resilience and performance with secure file downloads and VoIP calls.

Categories and Subject Descriptors

C.2.1 [Computer Systems Organization]: Network Architecture and Design; C.2.2 [Computer Systems Organization]: Network Protocols

General Terms

Design, Experimentation, Performance, Security

1. INTRODUCTION

The engineering principles and architecture of today's Internet were created in the 1960s and '70s. The problem networking aimed to solve was *resource sharing* – remotely using scarce and expensive devices like card readers or high-speed tape drives or even supercomputers. The communication model that resulted is a conversation between exactly two machines, one wishing to use the resource and one providing access to it. Thus IP packets contain two identifiers (addresses), one for the source and one for the destination host, and almost all the traffic on the Internet consists of (TCP) conversations between pairs of hosts.

In the 50 years since the creation of packet networking, computers and their attachments have become cheap, ubiquitous commodities. The connectivity offered by the Internet and low storage costs enable access to a staggering amount of new content – 500 exabytes

created in 2008 alone [13]. People value the Internet for *what* content it contains, but communication is still in terms of *where*.

We see a number of issues that affect users arising from this incompatibility between models.

- **Availability:** Fast, reliable content access requires awkward, pre-planned, application-specific mechanisms like CDNs and P2P networks, and/or imposes excessive bandwidth costs.

- **Security:** Trust in content is easily misplaced, relying on untrustworthy location and connection information.

- **Location-dependence:** Mapping content to host locations complicates configuration as well as implementation of network services.

The direct, unified way to solve these problems is to replace *where* with *what*. Host-to-host conversations are a networking *abstraction* chosen to fit the problems of the '60s. We argue that *named data* is a better abstraction for today's communication problems than *named hosts*. We introduce *Content-Centric Networking* (CCN), a communications architecture built on named data. CCN has no notion of host at its lowest level – a packet “address” names content, not location. However, we preserve the design decisions that make TCP/IP simple, robust and scalable.

Figure 1 compares the IP and CCN protocol stacks. Most layers of the stack reflect bilateral agreements; e.g., a layer 2 framing protocol is an agreement between the two ends of a physical link and a layer 4 transport protocol is an agreement between some producer and consumer. The only layer that requires universal agreement is layer 3, the network layer. Much of IP's success is due to the simplicity of its network layer (the IP packet – the thin “waist” of the stack) and the weak demands it makes on layer 2, namely: stateless, unreliable, unordered, best-effort delivery. CCN's network layer (Section 3) is similar to IP's and makes fewer demands on layer 2, giving it many of the same attractive properties. Additionally, CCN can be layered over anything, including IP itself.

CCN departs from IP in a number of critical ways. Two of these, *strategy* and *security*, are shown as new layers in its protocol stack. CCN can take maximum advantage of multiple simultaneous connectivities (e.g., ethernet and 3G and bluetooth and 802.11) due to its simpler relationship with layer 2. The *strategy* layer (Section 3.3) makes the fine-grained, dynamic optimization choices needed to best exploit multiple connectivities under changing conditions. CCN secures content itself (Section 5), rather than the connections over which it travels, thereby avoiding many of the host-based vulnerabilities that plague IP networking.

We describe the architecture and operation of CCN in Sections 2 through 5. In Section 6 we evaluate performance using our prototype implementation. Finally, in Sections 7 and 8, we discuss related work and conclude.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CoNEXT'09, December 1–4, 2009, Rome, Italy.
Copyright 2009 ACM 978-1-60558-636-6/09/12 ...\$10.00.

Better data delivery and use of network resources

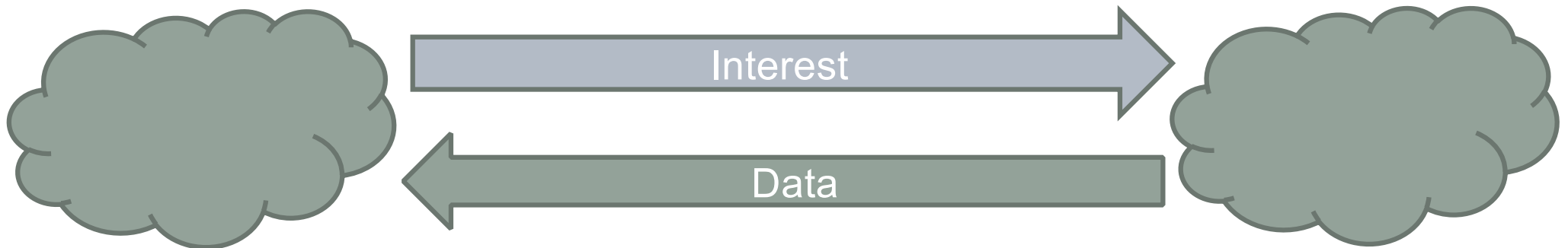
Jacobson, Van, et al. "Networking named content." *Proceedings of the 5th international conference on Emerging networking experiments and technologies*. ACM, 2009. CoNext'2009

Named-Data Networking Principles

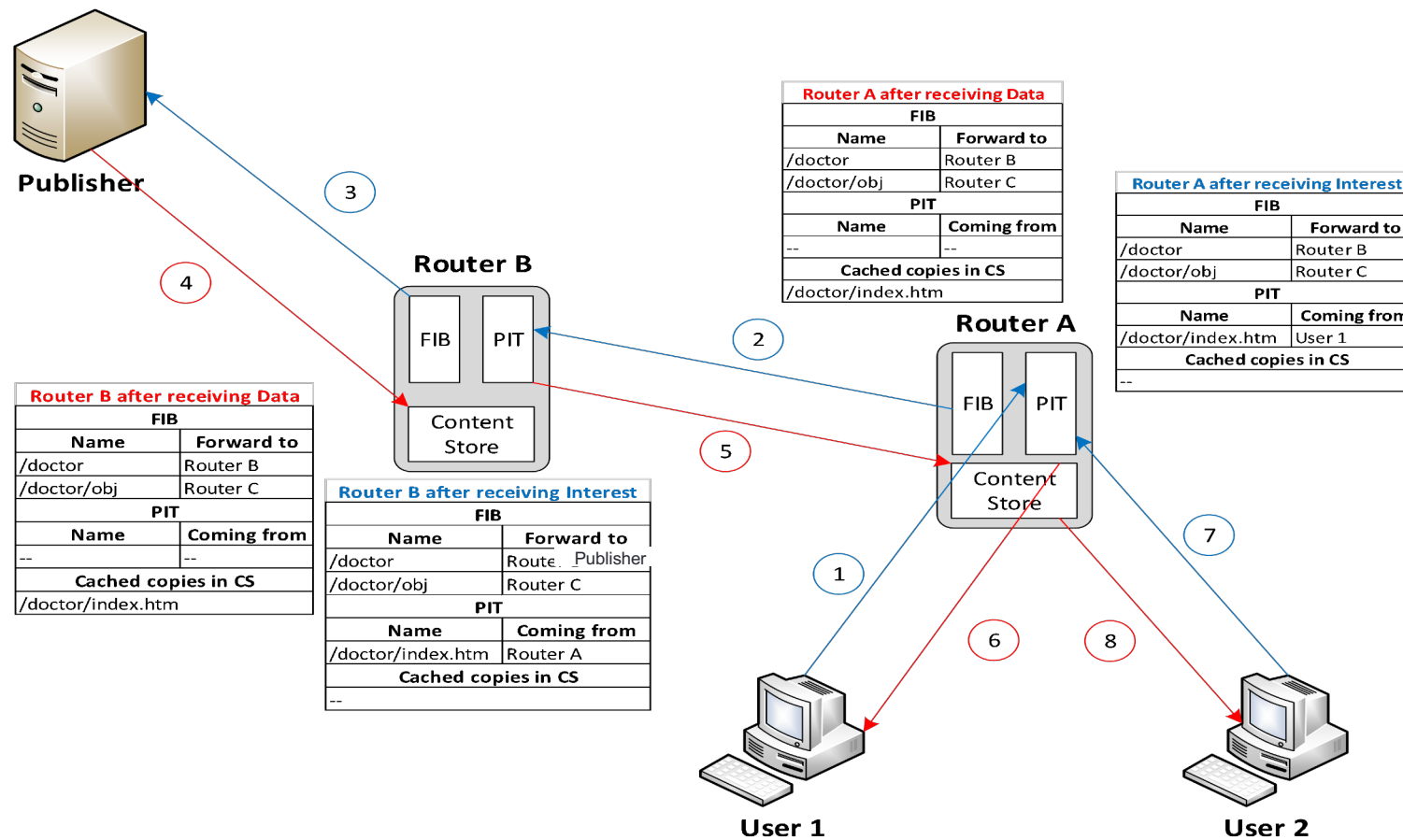
- Hierarchical names for content

[/wikipedia.org/wiki/doc/telecomnancy/video/part_0576](#)

- A Pull-based protocol based on 2 primitives



Named Data Networking Operations

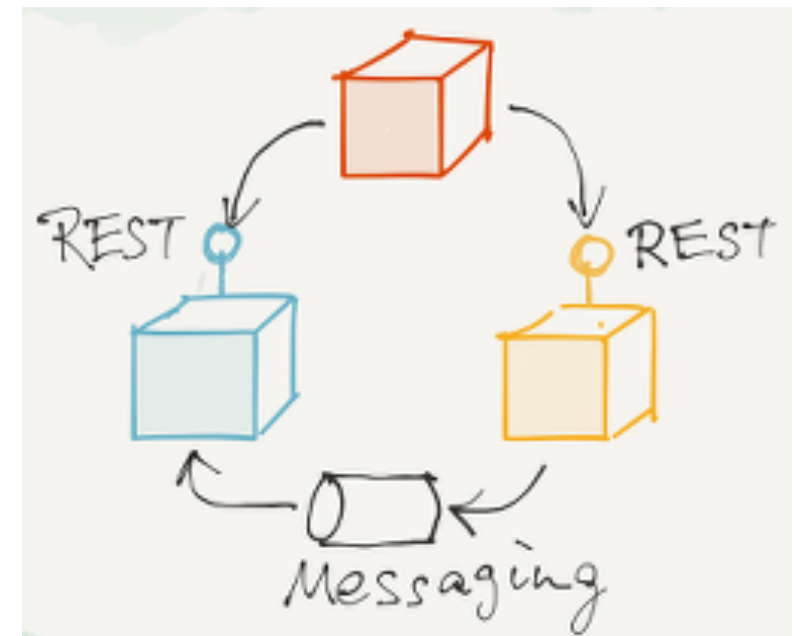


SOFTWAREIZING NDN

NDN NFV Chaining & Orchestration

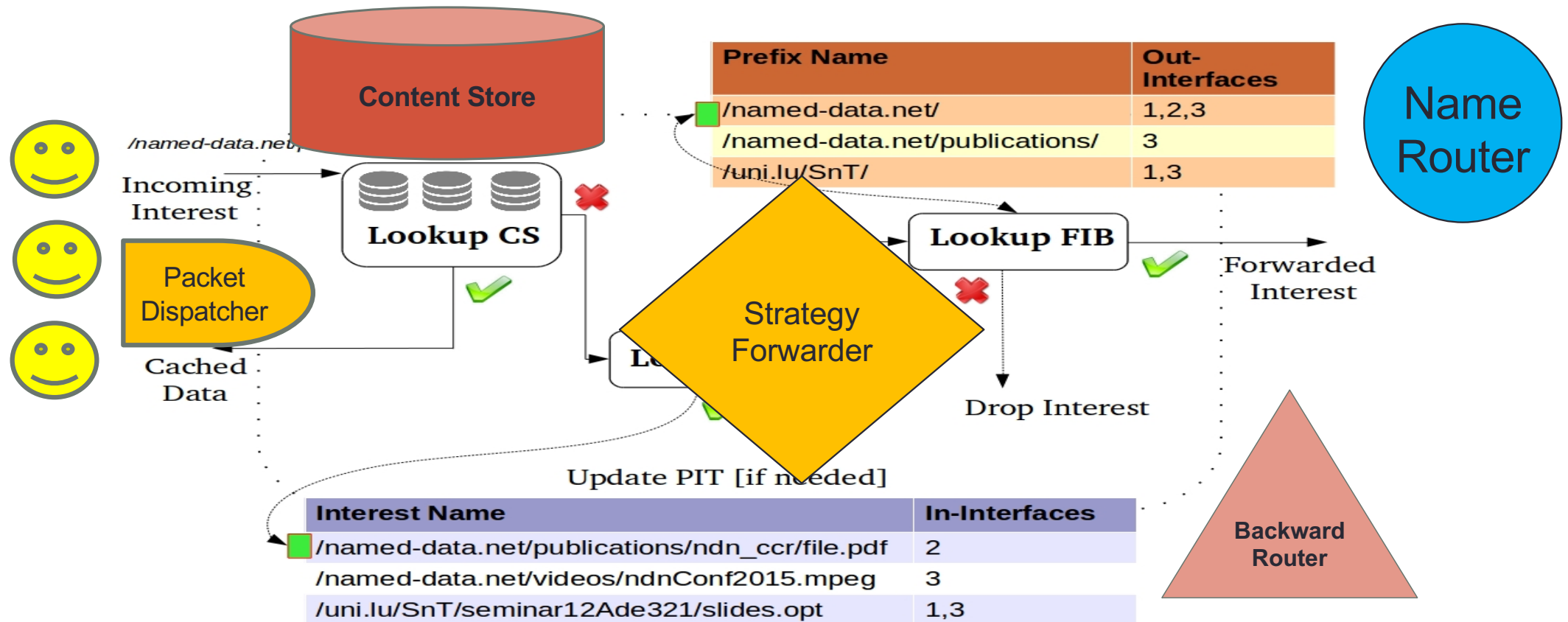
μ NDN A full NDN stack using Micro-services

- Micro-services (2011+)
 - Split a monolithic application in cooperating compact loosely single task functions
 - Lightweight protocols
- Support
 - Network Function Virtualization
 - Service Chaining
 - DevOps
- Expected « gain »
 - More management flexibility
 - Increased horizontal scalability
 - Dynamic on-demand evolution of functions
- Our Challenge
 - Decomposition of a monolithic NDN router
 - Dynamic Linkage and packet processing
 - Advanced Management services



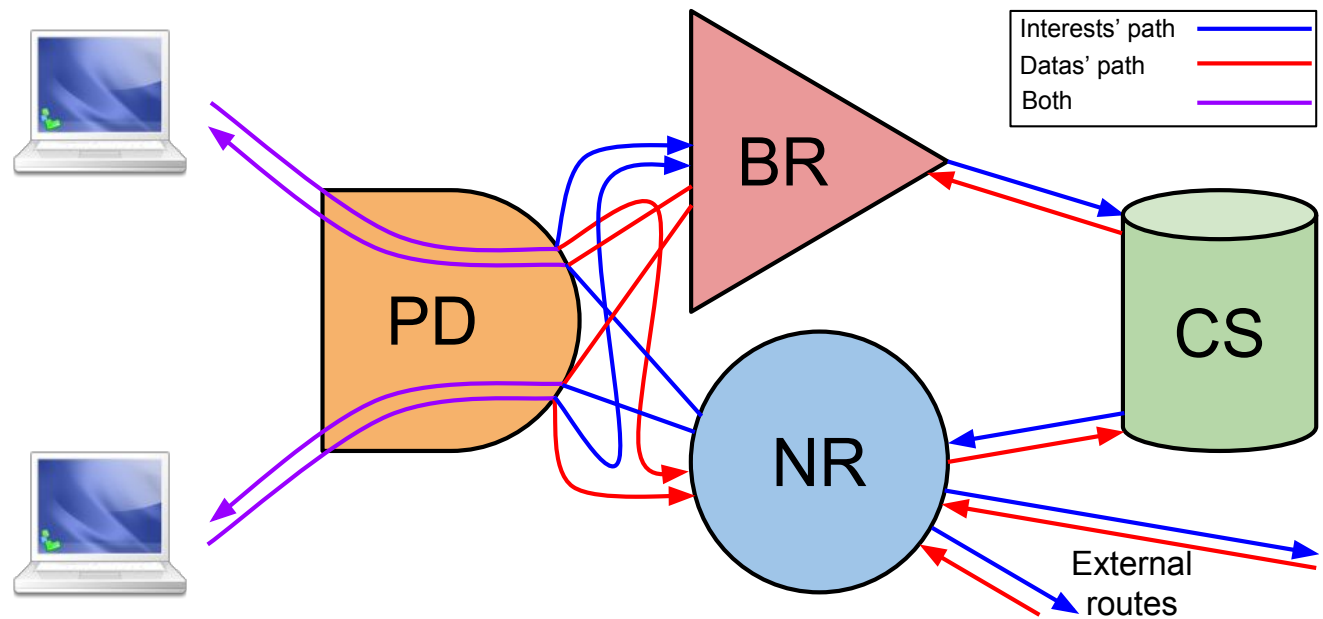
<http://www.klewitz.info/2016/02/26/modern-microservices-architectures/>

μNDN – Service Decomposition



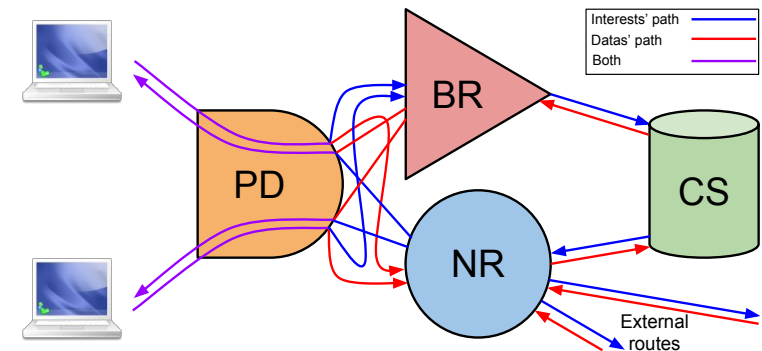
μ NDN – Service Decomposition

- Core routing functions
 - Name Router (NR) (*FIB*)
 - Backward Router (BR) (*PIT*)
 - Content Store (CS)
- Support functions
 - Packet Dispatcher (PD)
 - *Strategy Forwarder (SF)*
- Security functions
 - *Name Verifier (NV)*
 - *Name Filter (NF)*



CPU Usage

- Platform
 - 2 Intel Xeon 8 cores 2.4GHz (E5 2630v3)
 - Docker CE 18.03
 - ndn-cxx v0.6.1
- C++ developed single-threaded Microservices
- TCP/IP communication among the services (carrying NDN packets)
- NDN Data packets always carry 8192 octets
- Docker bridge among containerized micro services
- Host-centric producers and consumers

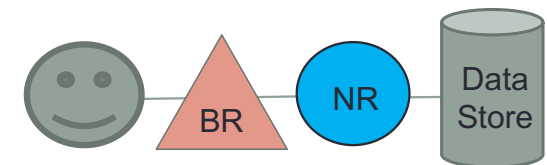


| | Microservices | | | | NFD full |
|----------------------|---------------|----|----|----|----------|
| | PD | CS | BR | NR | |
| %CPU core usage | 100 | 59 | 89 | 64 | 100 |
| Throughput (in Mbps) | 776 | | | | 527 |
| Latency (in ms) | 2,63 | | | | 3,88 |

996 Mbps if PD « freed »

Docker impact evaluation

| Micro-service | Bare Metal Throuput (Mbps) | Container Throughtput | Difference (%) |
|-------------------------------|----------------------------|-----------------------|----------------|
| Name Router | 2144 | 1935 | -10,5% |
| Backward Router | 1480 | 1380 | -6,8% |
| Packet Dispatcher | 2334 | 2081 | -10,8% |
| Content Store (from cache) | 3431 | 2852 | -16,9% |
| Strategy Forwarder | 2281 | 2058 | -9,8 % |
| Signature Verifier (RSA2048) | 665 | 630 | -5,3% |
| Signature Verifier (ECDSA256) | 212 | 218 | -2,5% |
| Name Filter | 2184 | 1971 | -9,8% |



- Signature verification is a heavy task
- **2-15% throughput penalty induced by the « dockerization »**

NDN NFV CHAINING & ORCHESTRATION

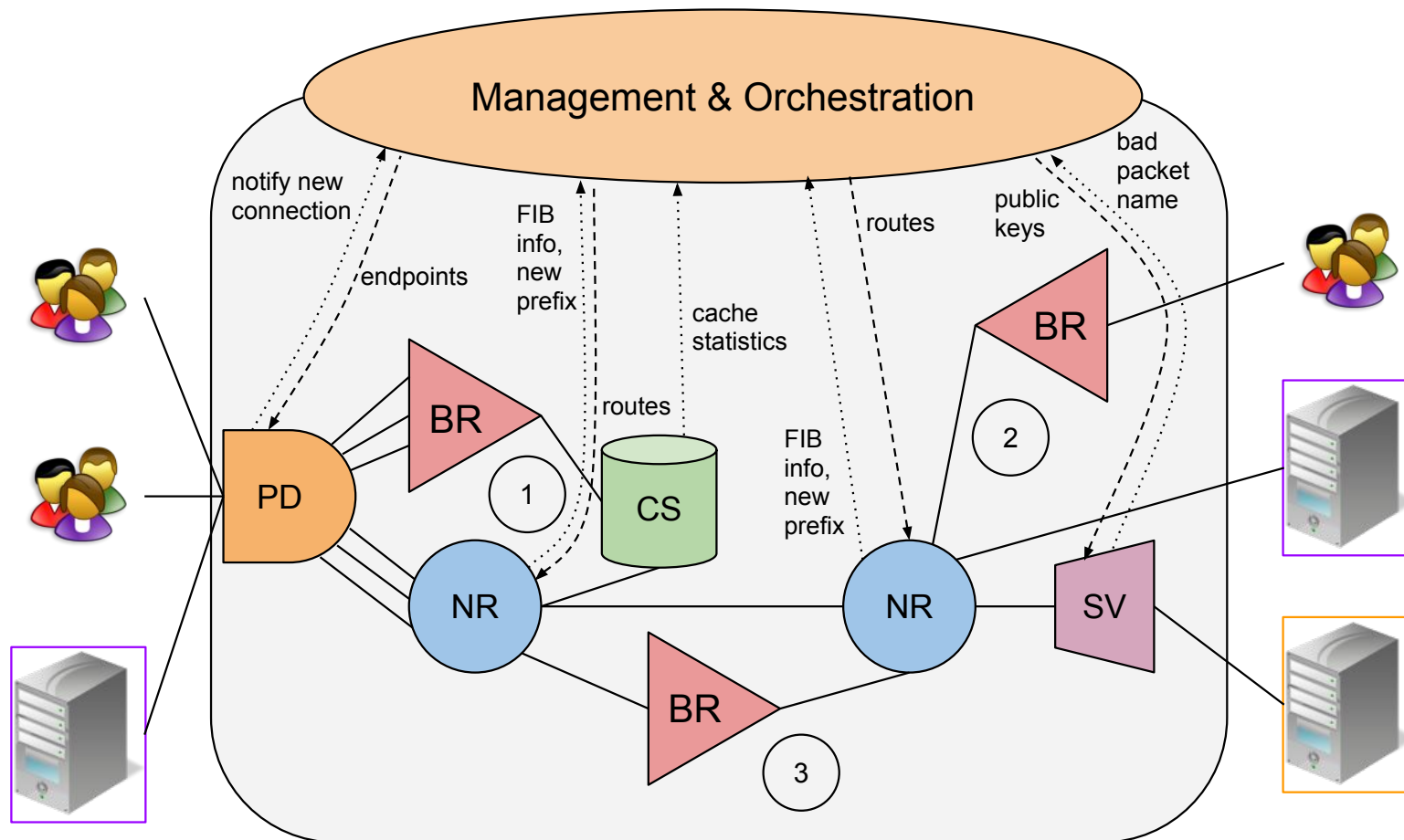
Managing microservices

- Needed for efficient microservice architecture
- Minimal set of required operations :
 - On-demand and automated microservices provisioning
 - Monitoring the activity of services
 - Policy driven network operations
 - *Managing the name-routes*
- Microservices must implement a management interface
 - Get command from manager
 - Send request to the manager
 - Periodically report statistics

| Microservice Functions Metrics | |
|--------------------------------|--|
| Name | Values |
| Name Router | Route statistics |
| Backward Router | Unsolicited Data packets Retransmitted Interest packets |
| Packet Dispatcher | User traffic statistics |
| Content Store | Hit/Miss count |
| Signature Verifier | Name of failed packets |
| Name Filter | Drop count |

| System & Network Metrics | |
|--------------------------|--|
| CPU Usage | |
| Throughput | |

Management & orchestration plane

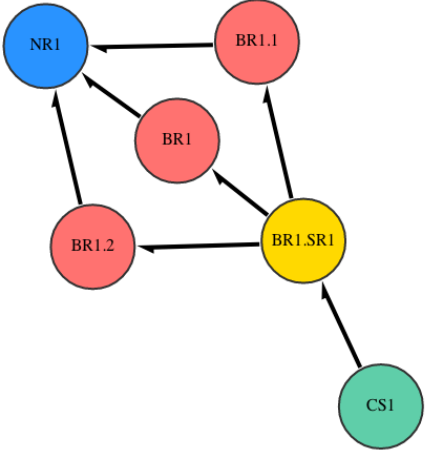


GUI network builder

152.81.47.226:8080/

152.81.47.226:8080

Network Topology



```
graph LR; NR1((NR1)) <--> BR1.1((BR1.1)); NR1 <--> BR1((BR1)); BR1.1 <--> BR1.SR1((BR1.SR1)); BR1.2((BR1.2)) <--> BR1.SR1; BR1.SR1 <--> CS1((CS1));
```

API

New node

Node type:

Result

Get node info

Node name:

Result

Remove node

Node name:

Result

Update node config (you may fill only part of the form)

Node name: Manager address:

Manager port: Report delay:

Strategy type:

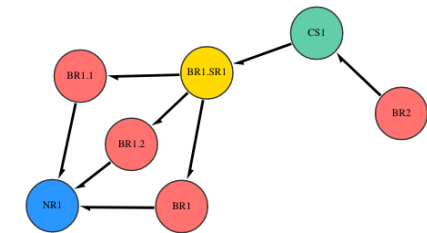
Result

New link

Source node: Target node: As producer: ☐

Result

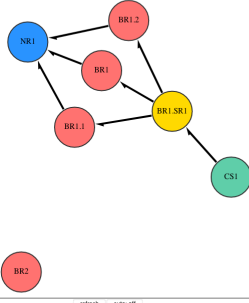
Get link info



152.81.47.226:8080/

152.81.47.226:8080

Network Topology



```
graph LR; NR1((NR1)) <--> BR1.1((BR1.1)); NR1 <--> BR1((BR1)); BR1.1 <--> BR1.SR1((BR1.SR1)); BR1.2((BR1.2)) <--> BR1.SR1; BR1.SR1 <--> CS1((CS1));
```

API

New node

Node type:

Result

Get node info

Node name:

Result

Remove node

Node name:

Result

Update node config (you may fill only part of the form)

Node name: Manager address:

Manager port: Report delay:

Strategy type:

Result

New link

Source node: Target node: As producer: ☐

Result

Get link info

Source node: Target node:

Result

Remove link

Source node: Target node:

Result

Get node info

Node name:

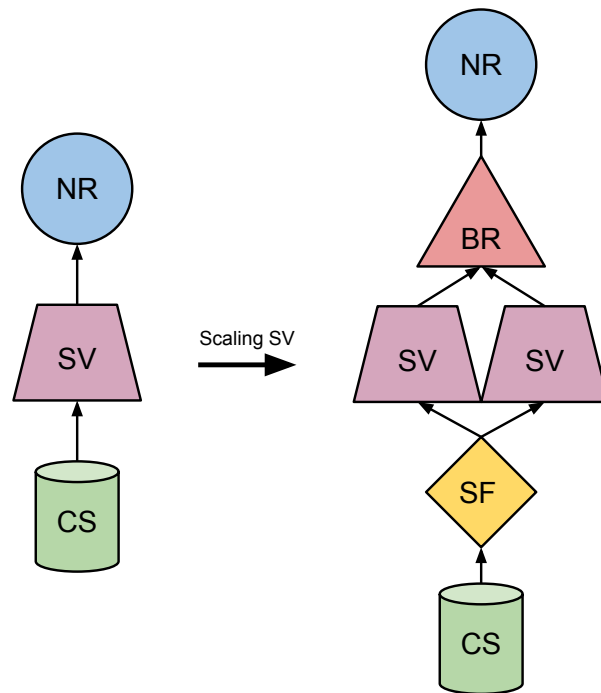
Result

```
{ "editable": true, "scalable": true, "addresses": { "data": "172.18.0.8", "command": "172.19.0.8" }, "cpu_stats": { "cpu_percent": 27.2, "cpu_system": 5134015296000000, "cpu_total": 13421200566, "last_update": 1539259418.3360357 }, "type": "BR", "size": 250, "cpu_quota": 100000 }
```

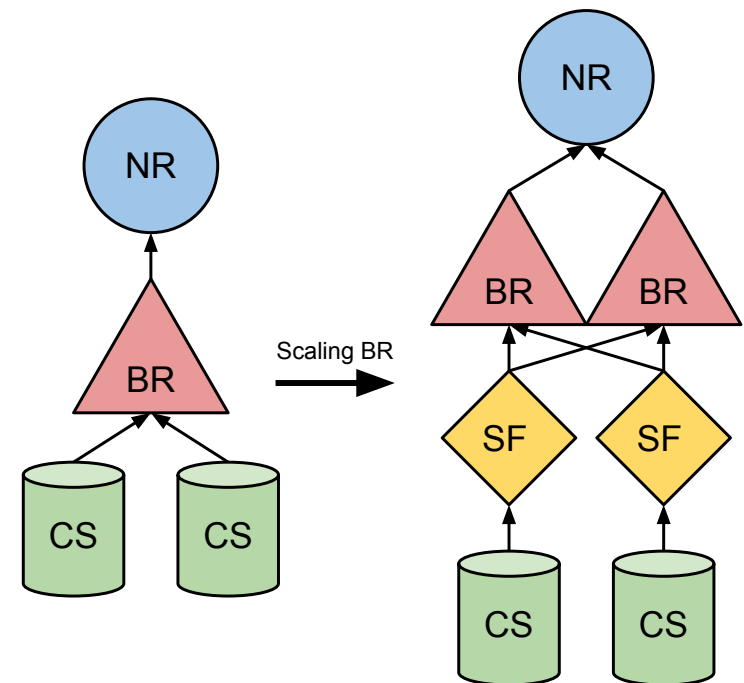
Remove node

A scaling procedure example

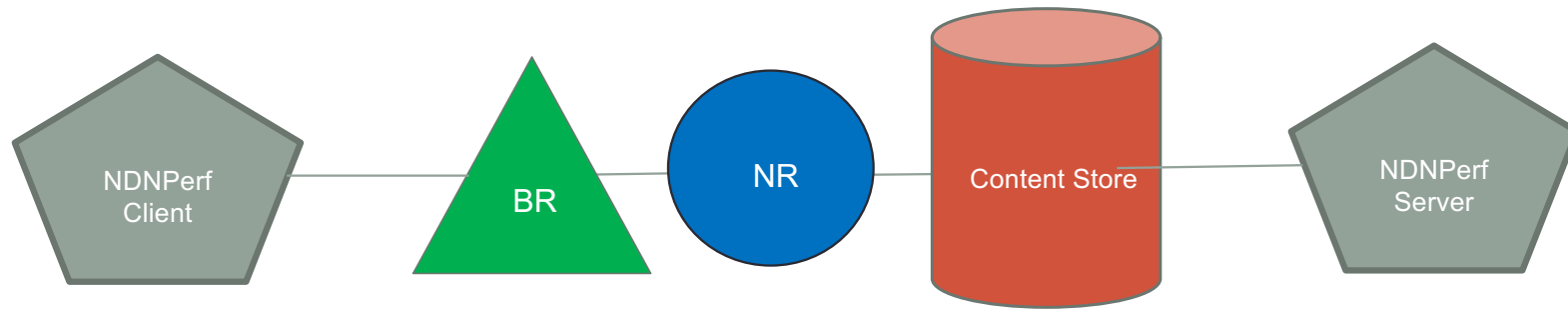
Support Function Scaling



Possible Backward Router Scaling

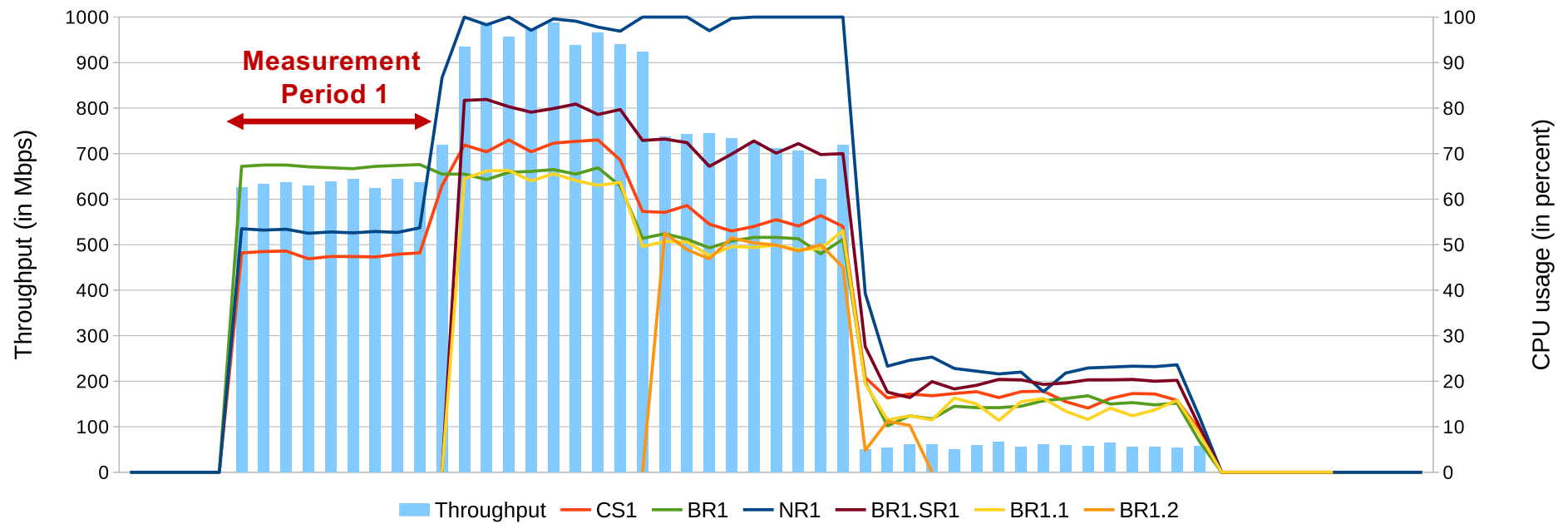


Backward Router Scaling

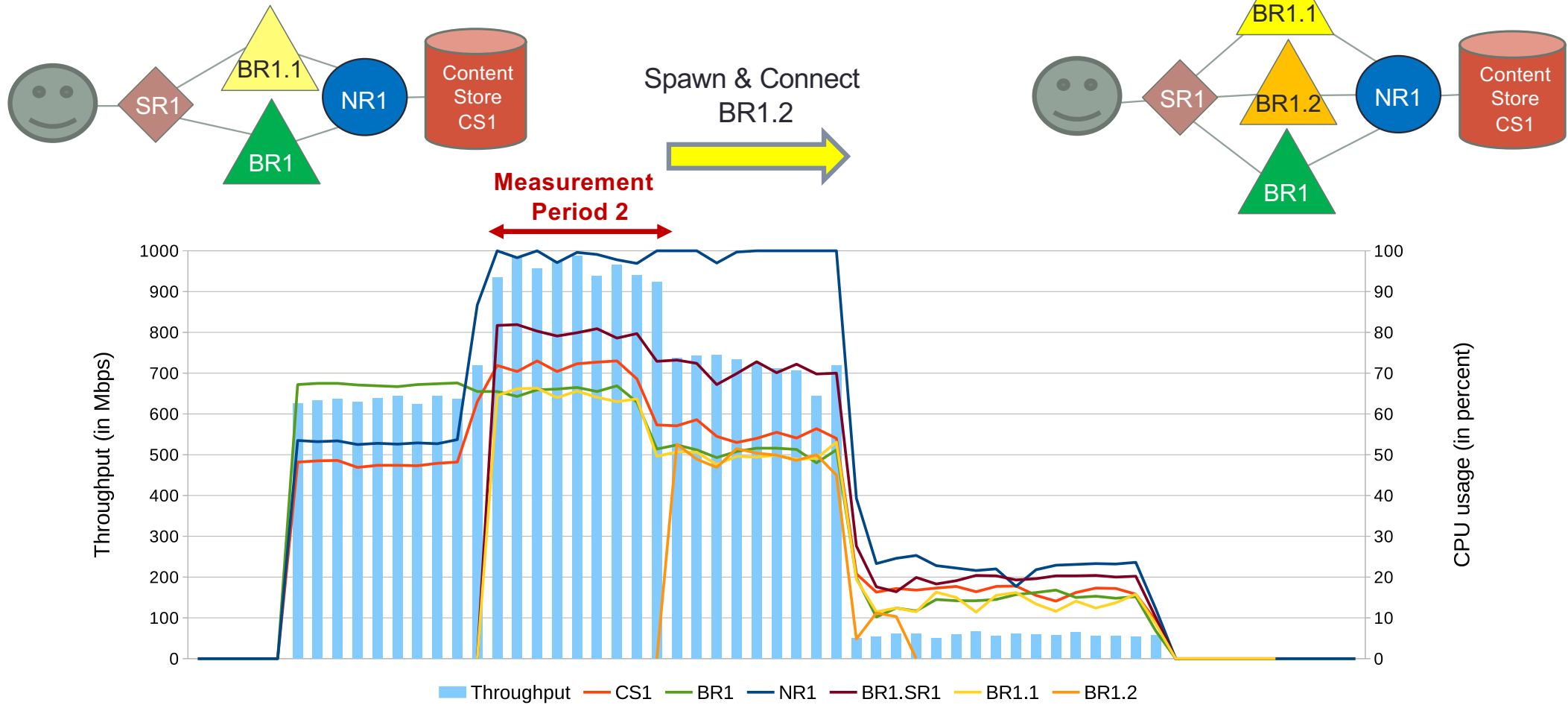


- Policy :
 - Automated scale-up if CPU consumption of a BR growth beyond 80% of allocated ressources
 - Scale down if the CPU consumption of a BR goes below 50%
- Conditions
 - Grow Content demand & delivery (NDNPerf [Marchal 16])
 - Content Store just relays data (no cache management)
 - CPU limitation set to 67% of a core

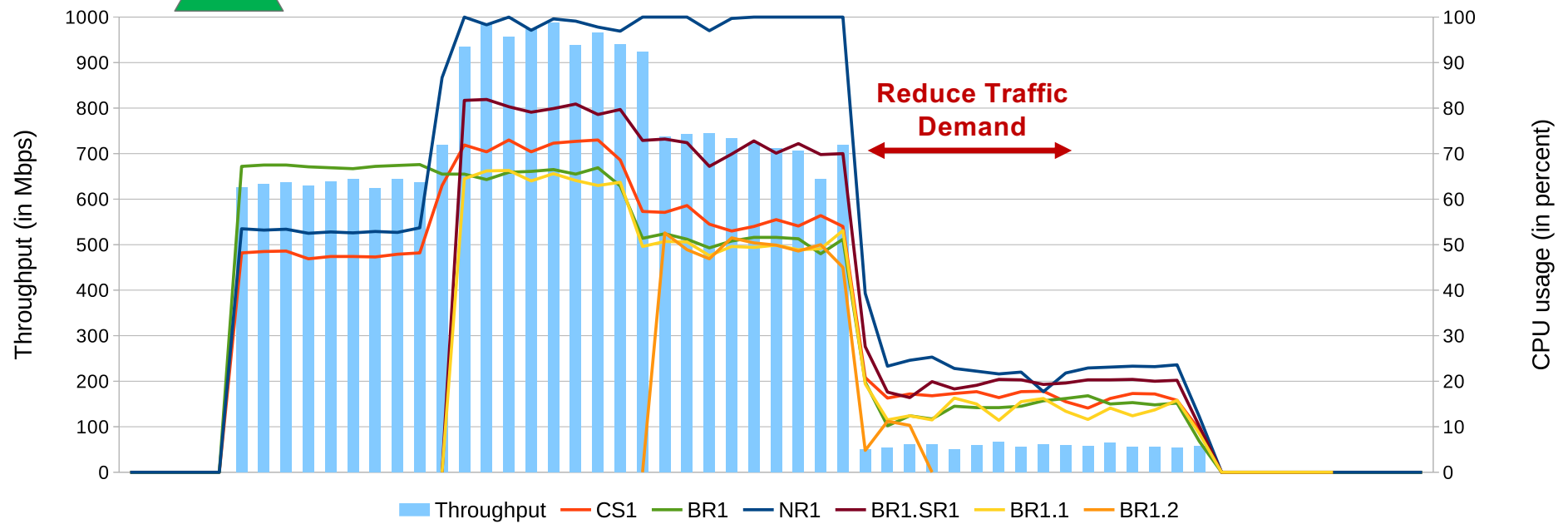
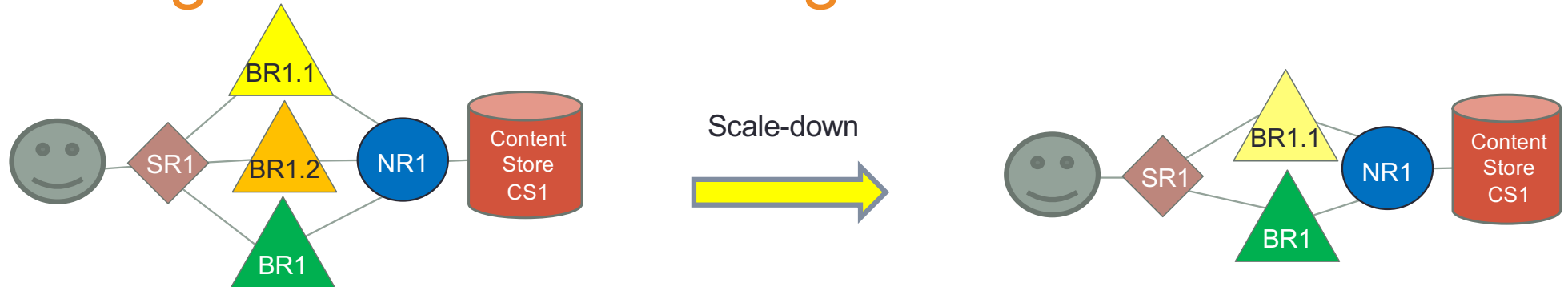
Background Router scaling



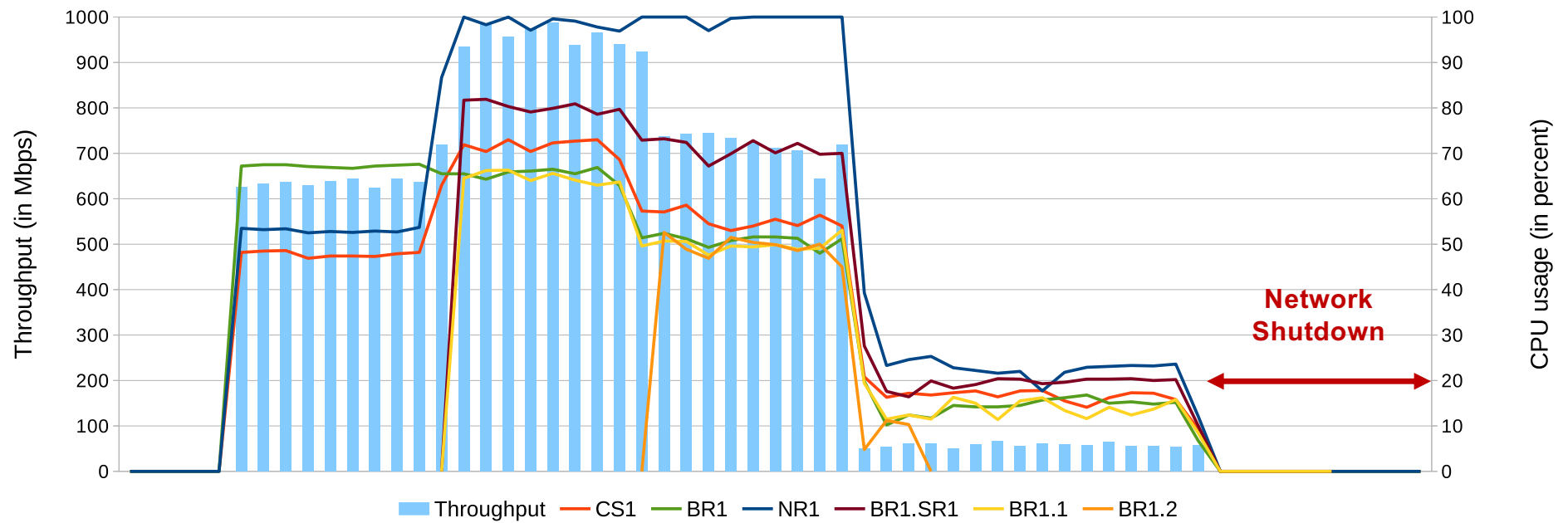
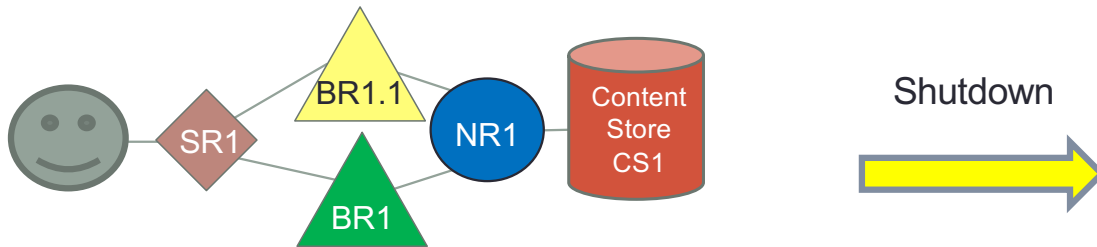
Background Router scaling



Background Router scaling



Background Router scaling



INTEGRATION WITH LEGACY SERVICES

NDN/HTTP Gateway

HPPT/NDN Integration

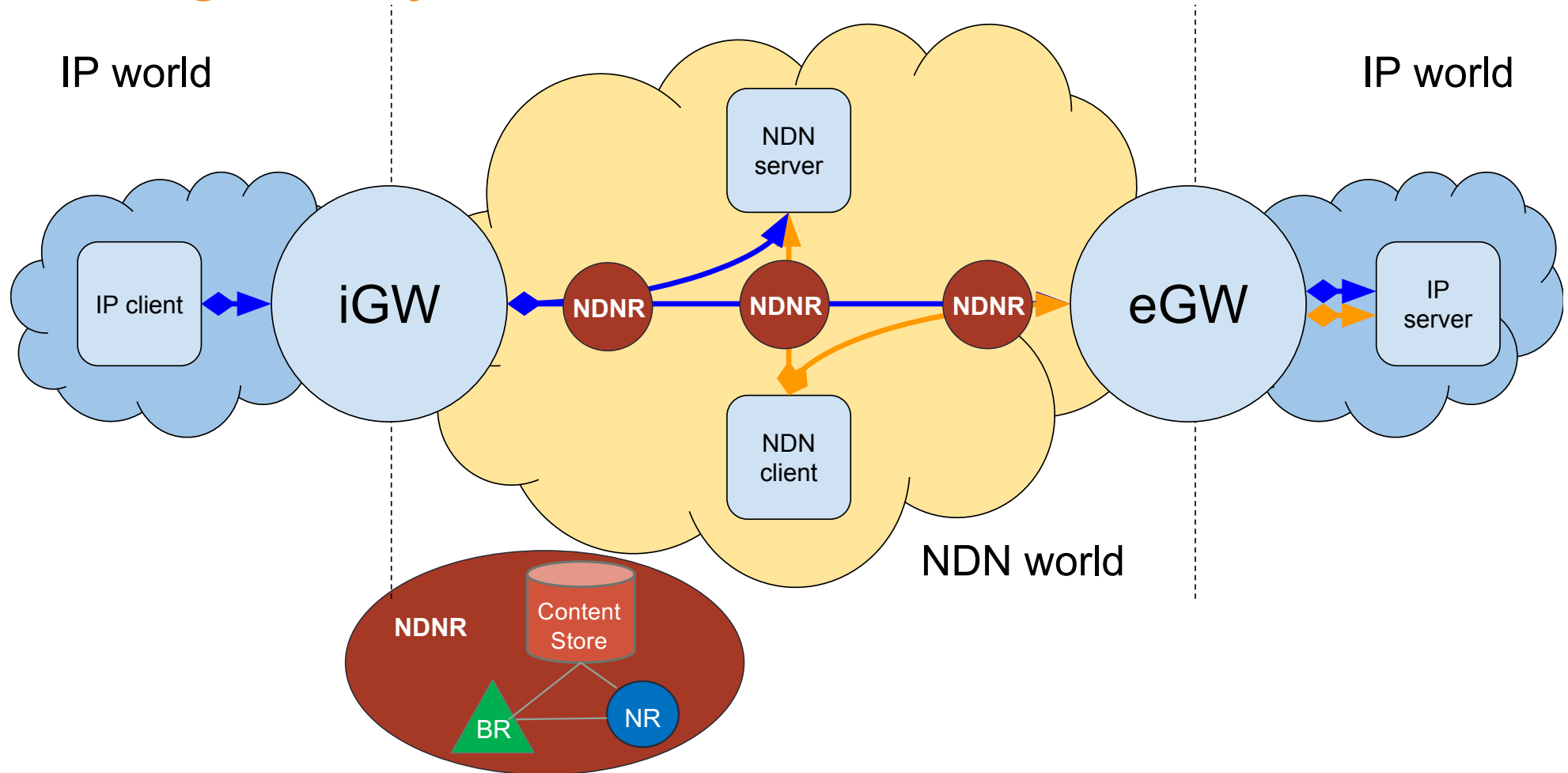
- Motivation

- Protocol migration is a complex, costly and long process
- NDNs without content have no value
- The Web holds a significant part of the world content

- Objectives

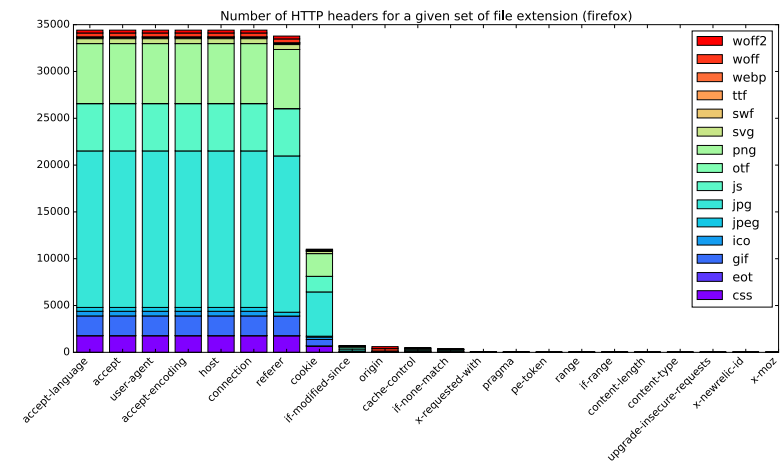
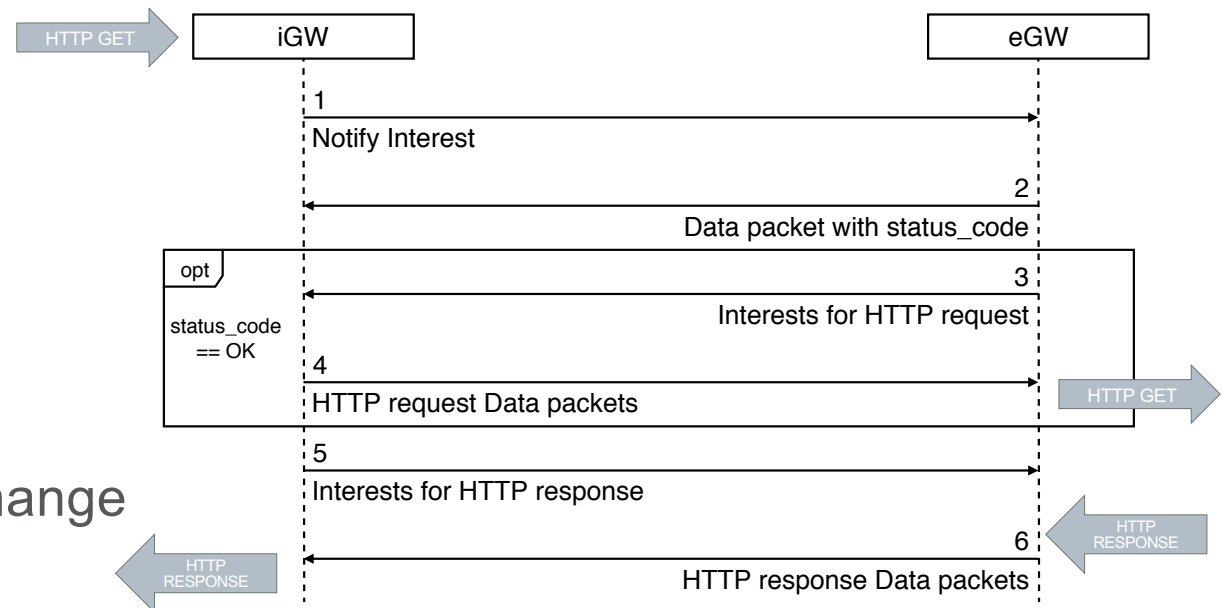
- Insert virtualized and chained NDN functions in the legacy Internet
- Guarantee transparency (standard IP servers/clients) do not notice
- Activate NDN caches
- Improve performance and user experience

iNDN gateway and NDN islands



xGW Operations

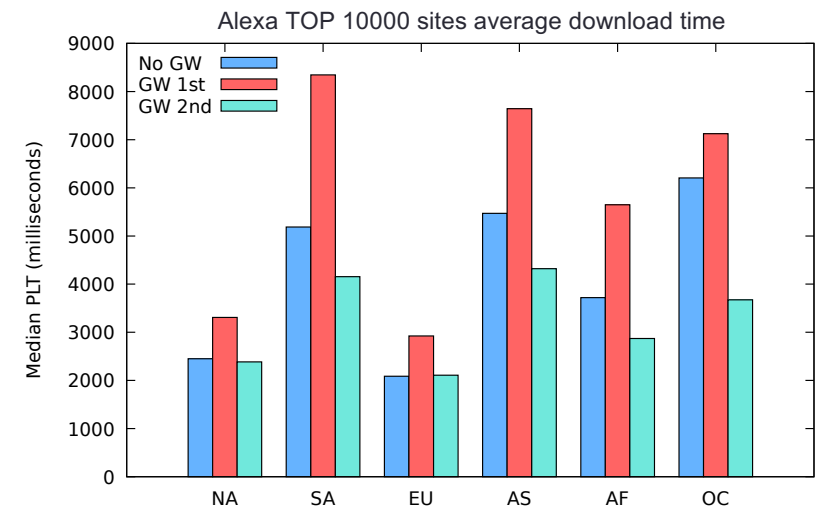
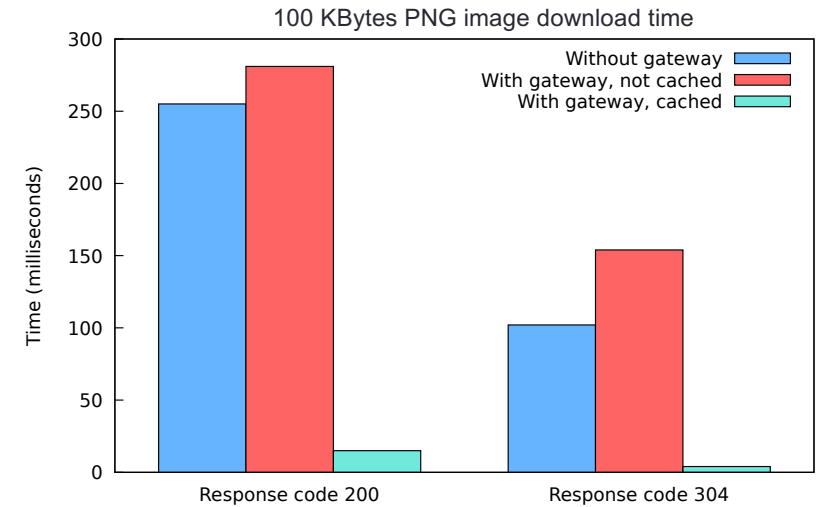
- Inverse name-mapping
- User generated token based exchange
 - Steps 1&2: find an interested eGW
 - Steps 3&4: transfert the request details
 - Steps 5&6: deliver the content chunks
- Requests « unification »
 - Careful selection of fields HTTP fields relevant in a request (user-agent, accept-language, ...)



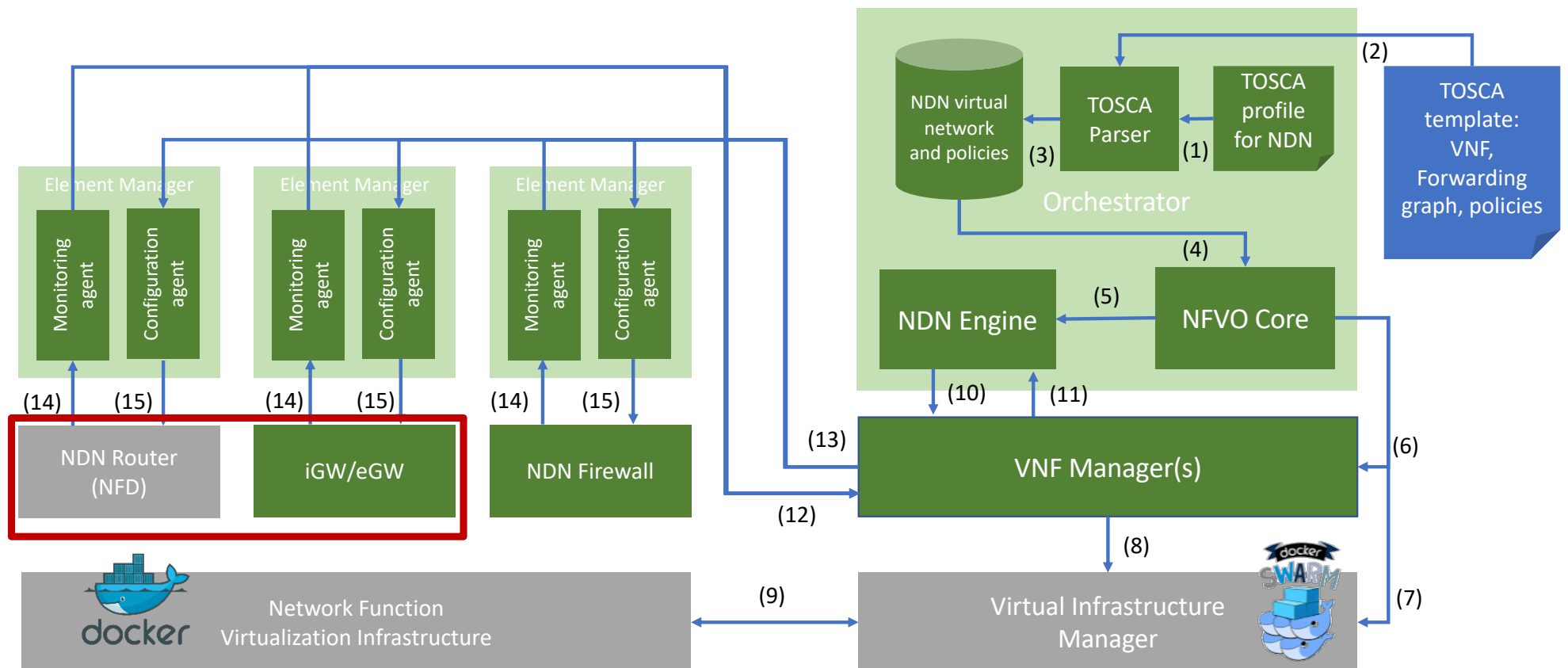
Some GW measurements

- Reference sites : TOP 10000 Alexa sites
- 36,2% of ressources are HTTP ressources

| | NA | SA | EU | AS | AF | OC | Total |
|-------|-------|------|-------|------|------|------|--------|
| HTTP | 16,42 | 0,53 | 10,72 | 8,12 | 0,19 | 0,13 | 36,2 % |
| HTTPS | 35,27 | 0,59 | 20,11 | 7,56 | 0,15 | 0,12 | 63,8 % |



Full integration [COMMAG'19]



CONCLUSION

Conclusion

- NDN has a nice and dynamic research community and potential applications
- Virtualizing, Micro-servicing and orchestrating NDN works !
- NFV & Chaining enables :
 - Incremental deployment of new protocols
 - Dynamic management of functions
 - Optimized functions placement and reorganization of chains
- Network functions virtualization & orchestration
 - Simpler functions, more complex chaining and orchestration ...
 - Alternative implementations : Serverless frameworks, P4 NDN [Signorello 2018], EBPF, ...
 - Need to rethink orchestration in both vertical (stack level) and horizontal levels (scale-up/down)



Softwarized
Management
is King !

THANK YOU!

Olivier Festor



mardi 21 avril 2020