Network Monitoring for SDN Virtual Networks

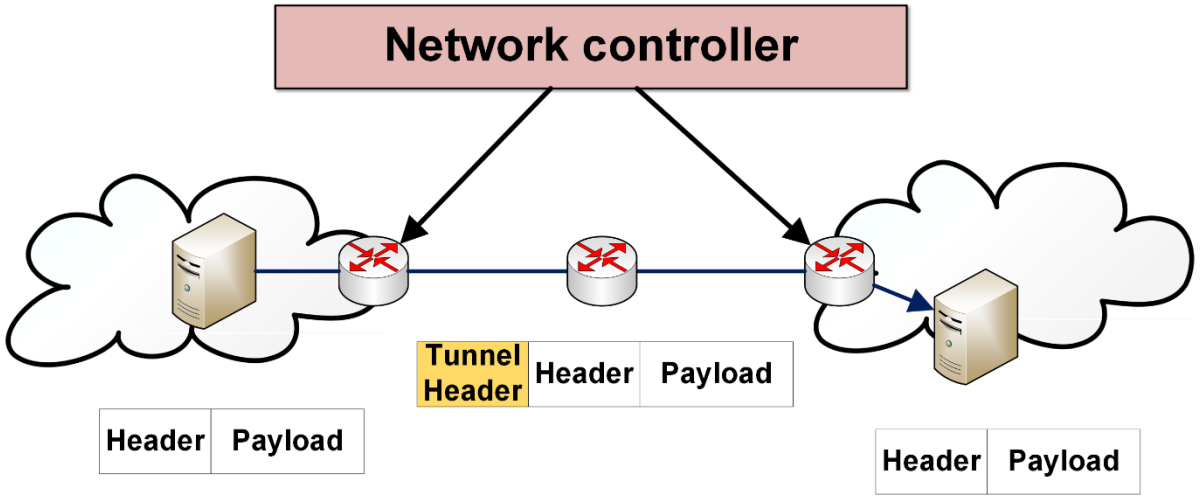
Gyeongsik Yang · Heesang Jin · Minkoo Kang
Gi Jun Moon · Chuck Yoo
ksyang@os.korea.ac.kr

KOREA UNIVERSITY

IEEE INFOCOM
IEEE International Conference on Computer Communications

Network virtualization

- Network infrastructure - virtualized in the form of virtual networks (VNs)
- Current solution: central controller configures tunneling (overlay) policies
  - Tunneling: attaching additional header to differentiate VNs
  - Network controller: install flow rules for tunneling at edge switches



  - Datacenter examples: NVP [1] and VFP [2]

[1] Koponen, Teemu, et al. "Network virtualization in multi-tenant datacenters." 11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14). 2014.
[2] Firestone, Daniel. "VFP: A Virtual Switch Platform for Host SDN in the Public Cloud." 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17). 2017.

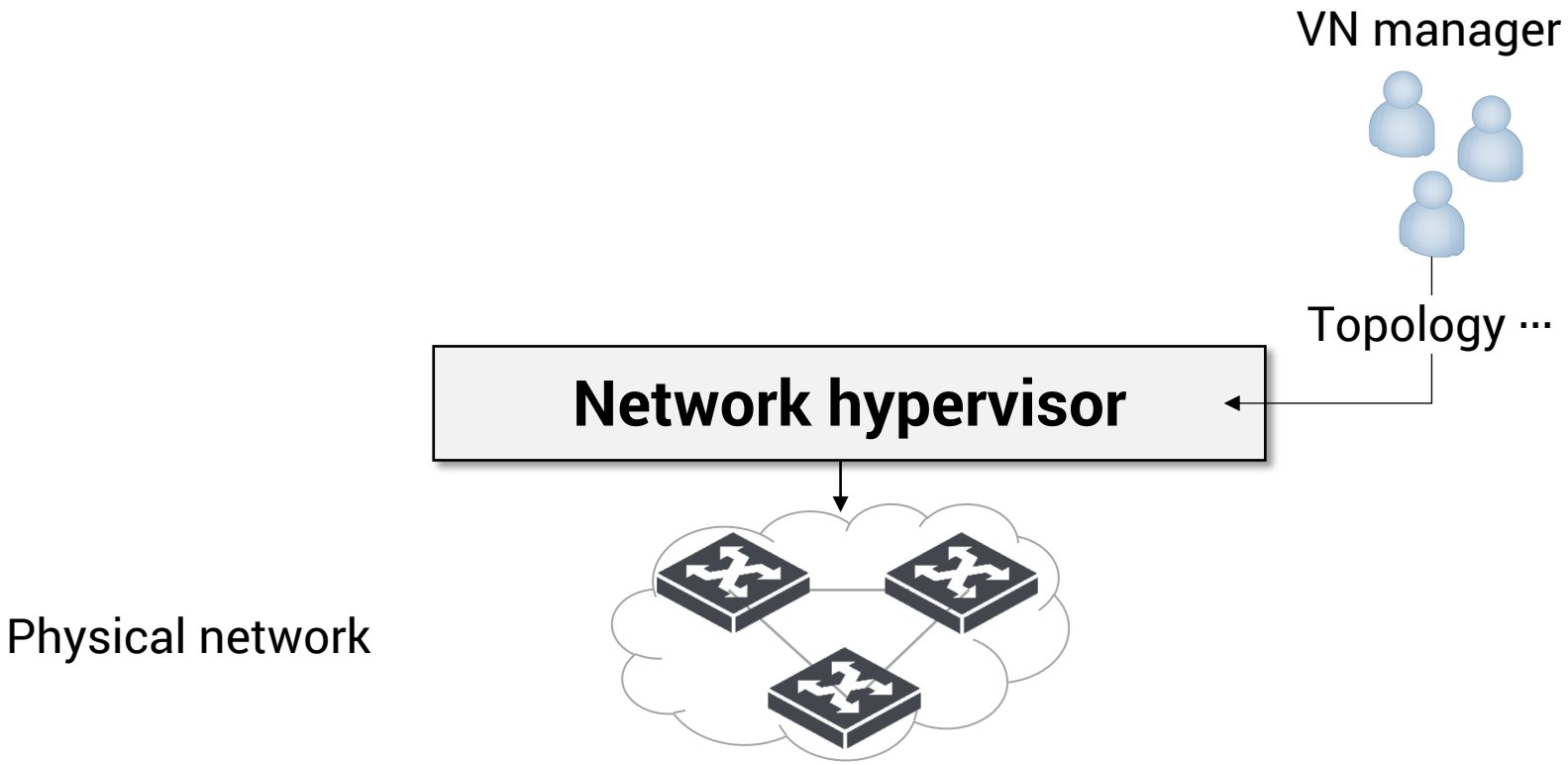
Limitation: VN programmability

- **Network programmability**
  - Topology provisioning, direct control over switches, network monitoring
- **Programmability is not provided for VN users**
  - VN generation and control – done only via the physical network's controller
  - Virtual switches – not disclosed to tenants
- **Time for VN programmability [1-2]**
  - SDN-based in-network optimizations
  - Custom network policies
  - End-to-end performance diagnosis

[1] Yang, Gyeongsik, et al. "Libera for Programmable Network Virtualization." IEEE Communications Magazine 58.4 (2020): 38-44.
[2] Costa, Paolo, et al. "NaaS: Network-as-a-Service in the Cloud." 2nd USENIX Workshop on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services (Hot-ICE 12). 2012.

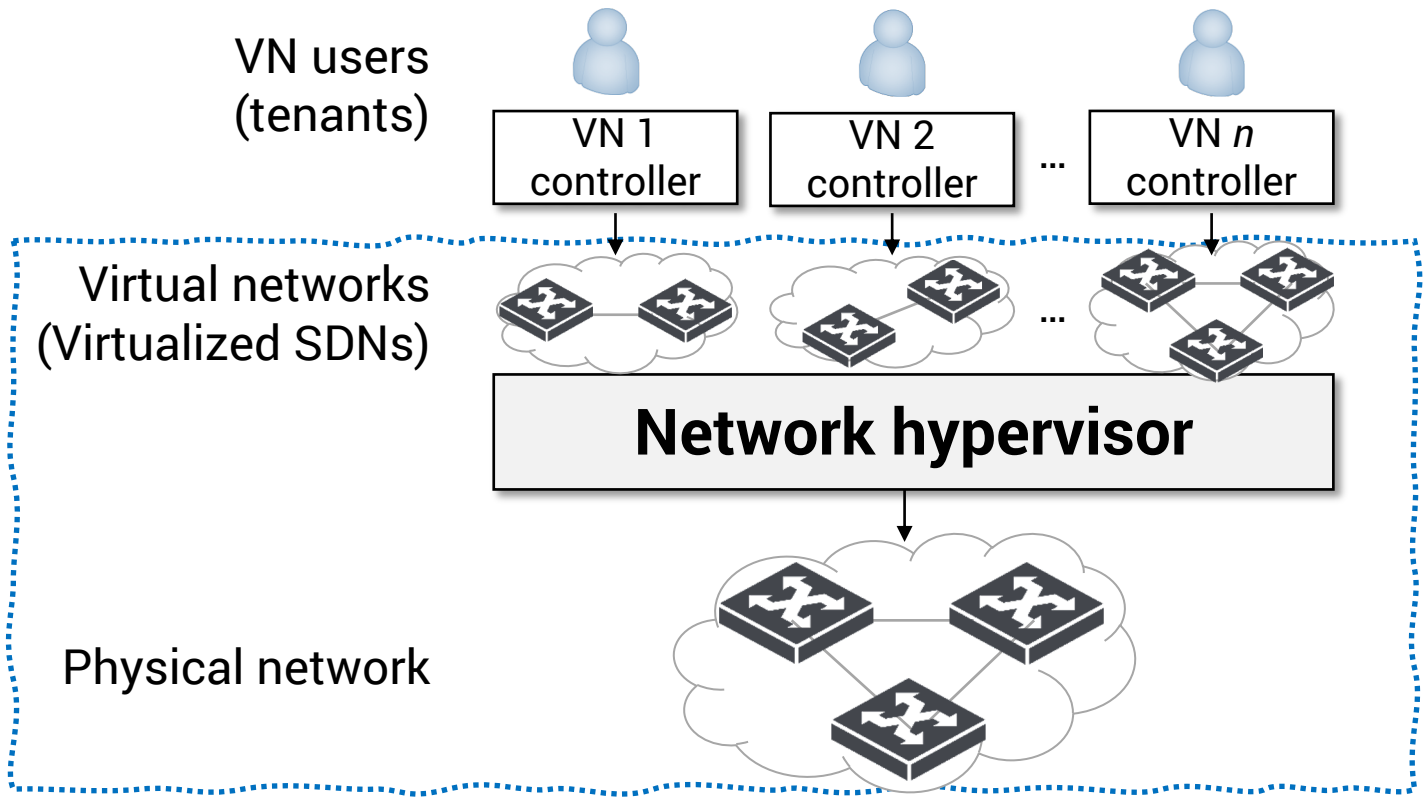
SDN-based network virtualization (SDN-NV)

- Provide programmable virtual networks in datacenters [1-3]



SDN-based network virtualization (SDN-NV)

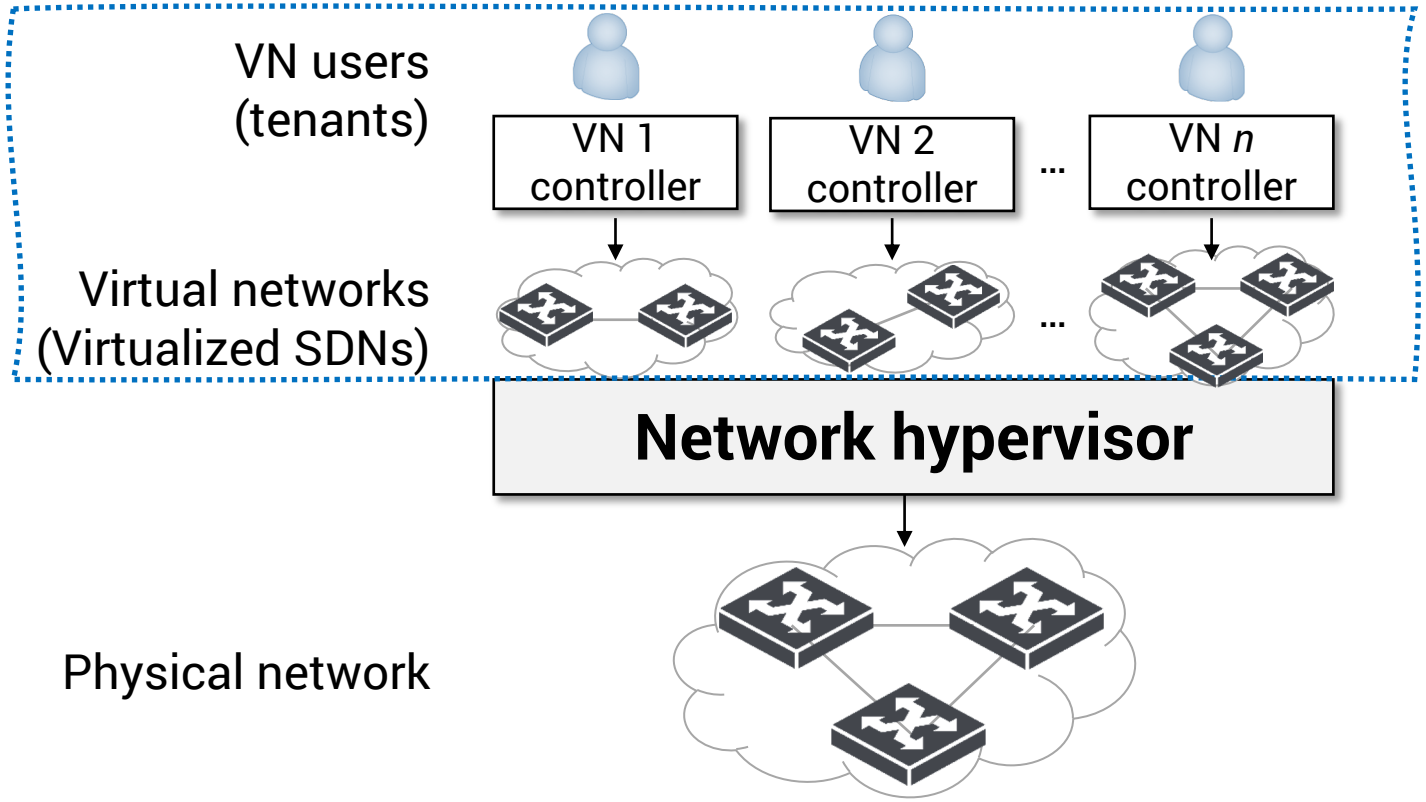
- Network hypervisor (NH): key enabler for SDN-NV



Virtualization of physical network

SDN-based network virtualization (SDN-NV)

- **Network hypervisor (NH): key enabler for SDN-NV**
  - Virtualized SDN (vSDN) for each VN user



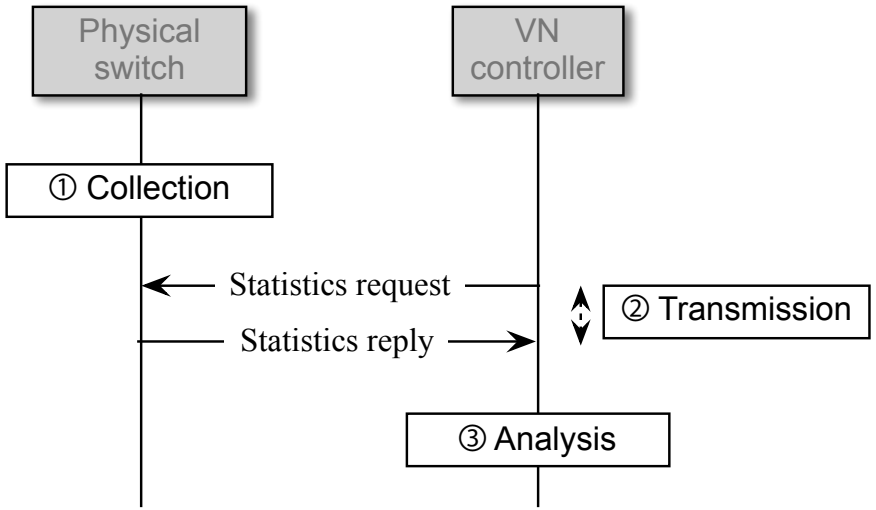
Control of virtualized SDNs

Previous studies

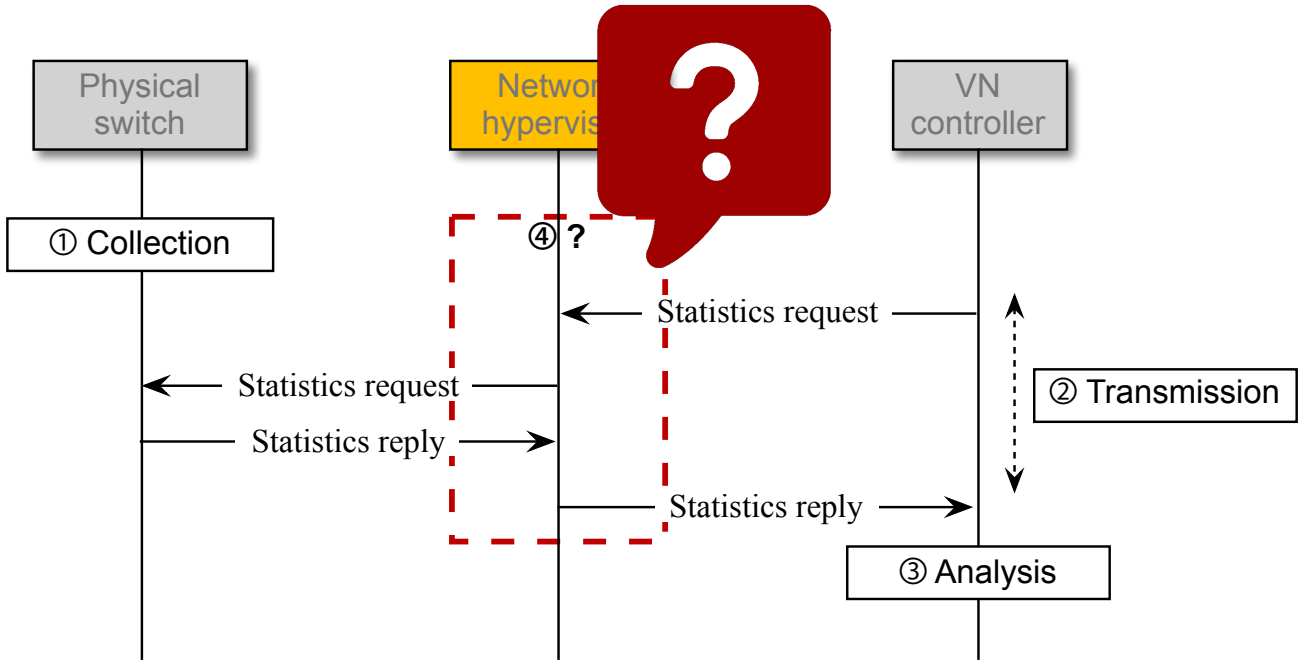
	FlowVisor (2009)	FlowN (2013)	OpenVirteX (2014)	CoVisor (2015)	OnVisor (2018)	Libera (2020)
Multi-tenancy	O	O	O	X	O	O
Architecture	Central	Central	Central	Central	Distributed	Central
Scalability	X	X	X	X	O	O
VM migration support	X	X	X	X	X	O
Traffic engineering	X	X	△	X	X	X
Network monitoring	**X**	**X**	**X**	**X**	**X**	**X**

Network monitoring in virtualized SDN

- **No consideration for network monitoring, a prerequisite for optimizations**
  - Most VN controllers regularly monitor (e.g., ONOS, OpenDayLight) for consistency and optimizations
- **Network monitoring: 1) collection, 2) transmission, and 3) analysis**



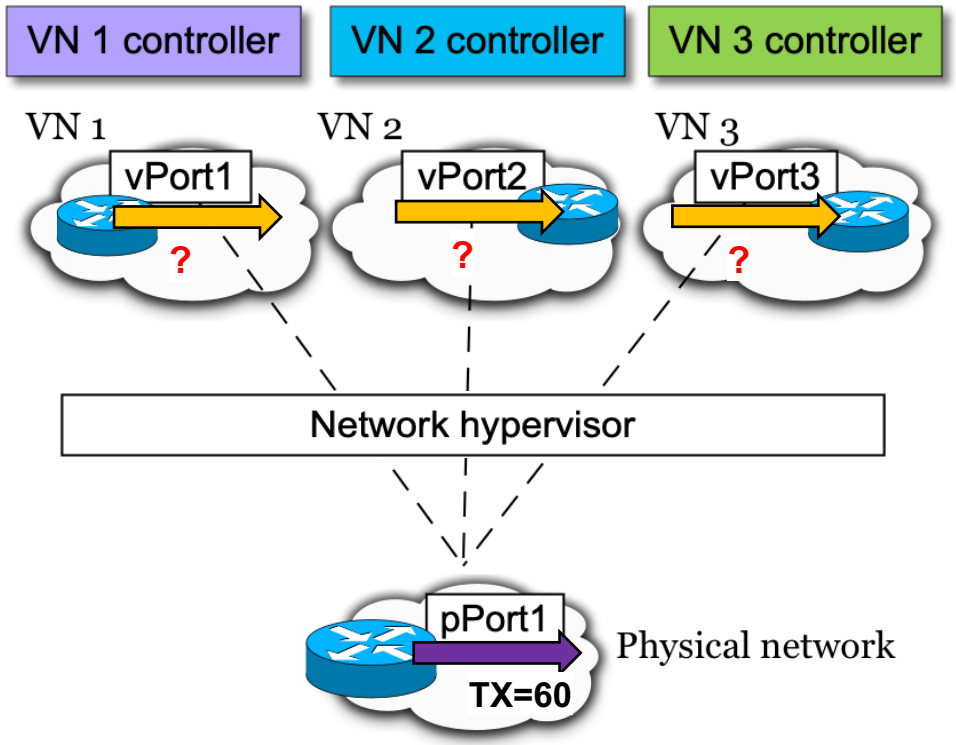
(a) SDN



(b) Virtualized SDN

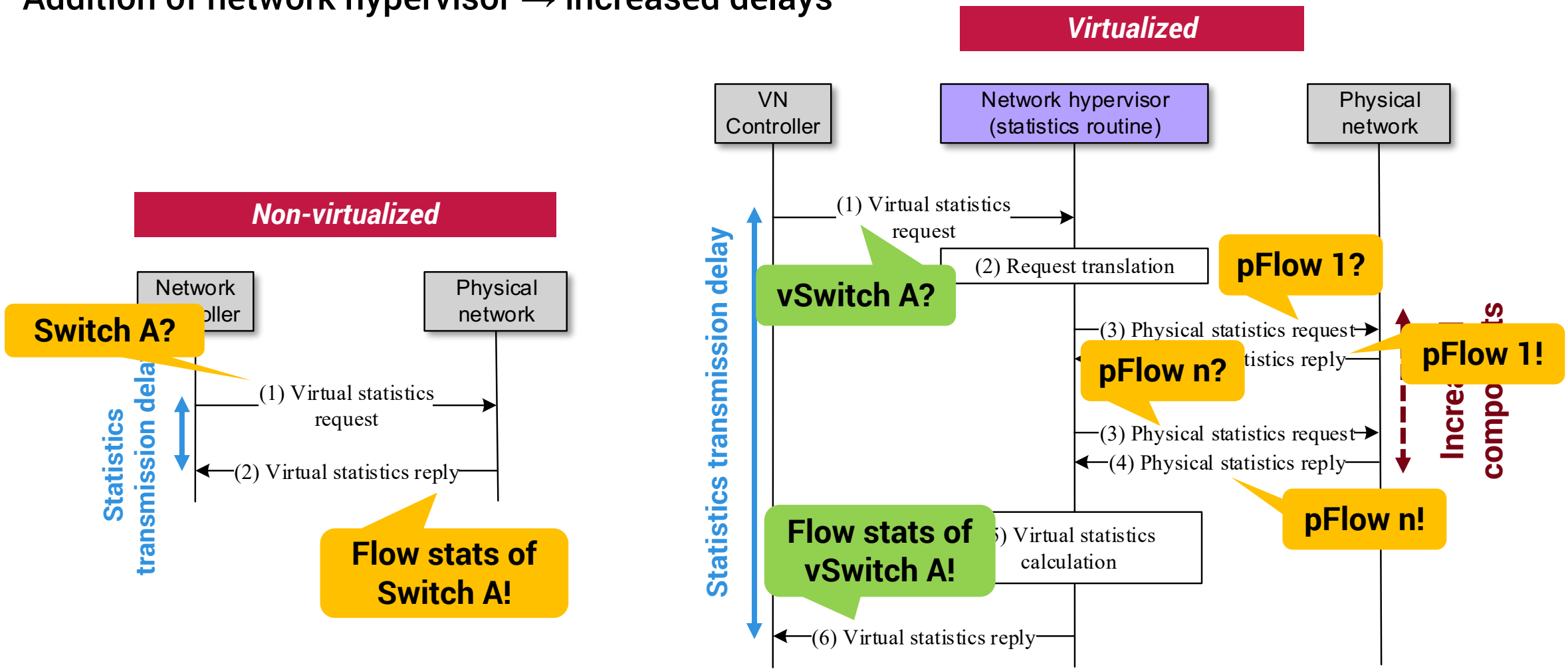
Challenges: (1) non-isolated statistics

- Physical switches / ports: shared among several VN users
  → throughput in physical network - no differentiation between VNs



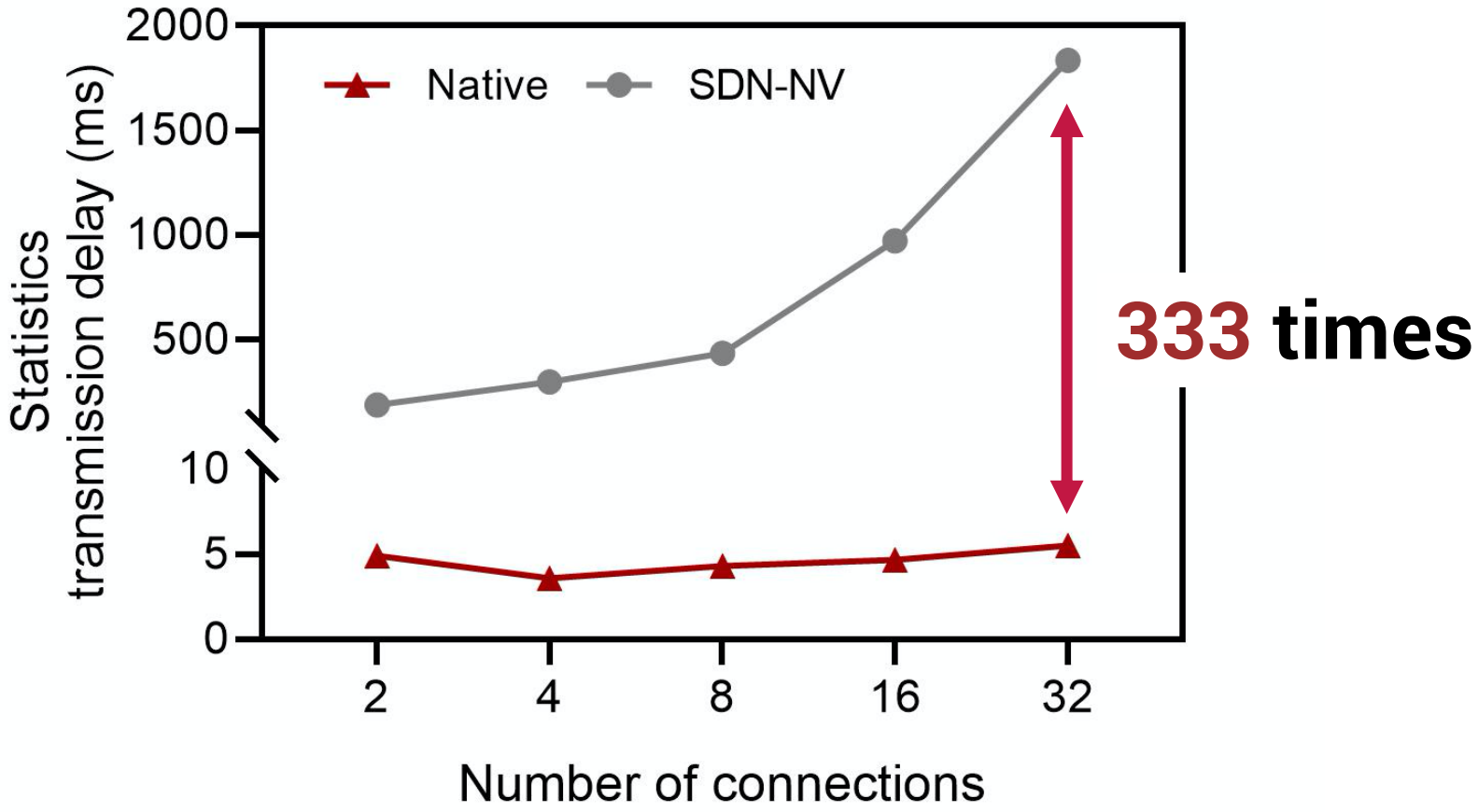
Challenges: (2) high transmission latency

- Addition of network hypervisor → increased delays



Challenges: (2) high transmission latency

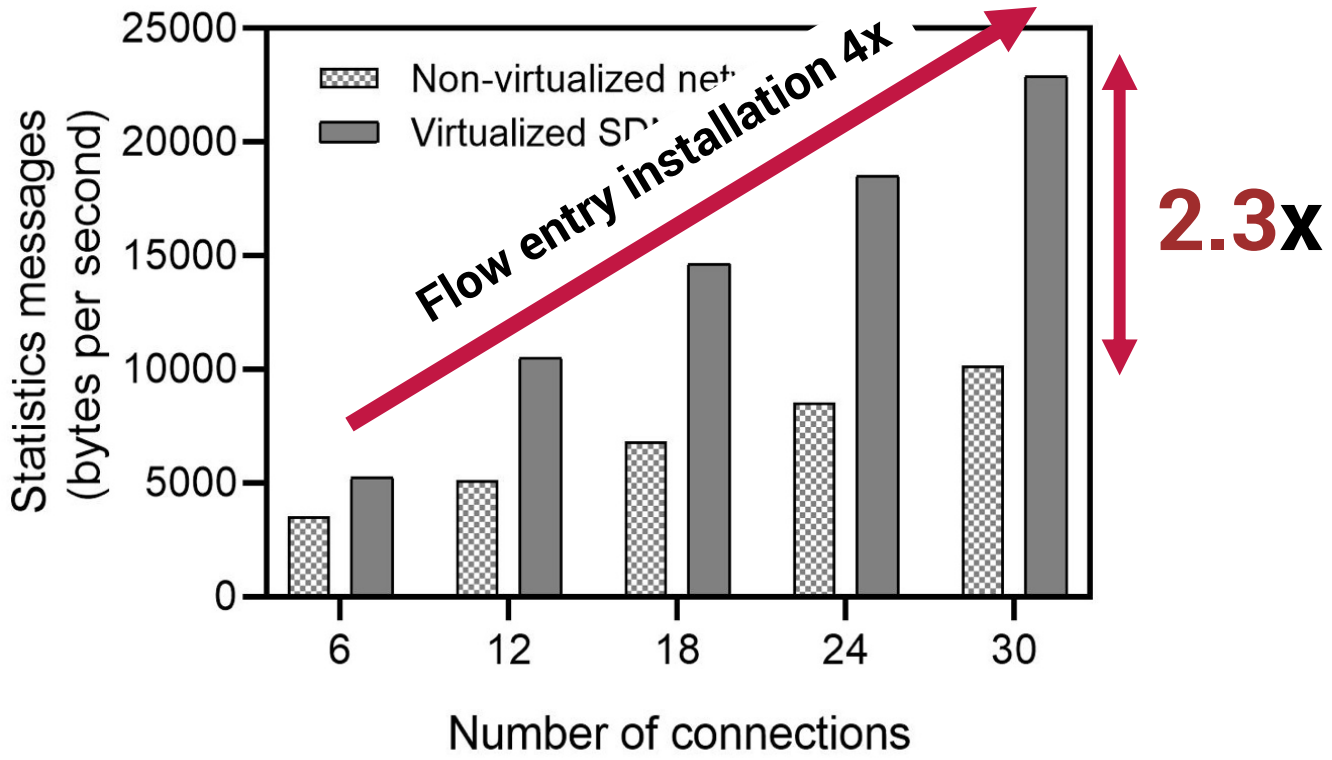
- Addition of network hypervisor → increased delays



4-ary fat-tree topology, one VN user, flow monitoring with ONOS controller

Challenges: (3) excessive control channel consumption

- Excessive control traffic on statistics → delay on other important operations
  - 2.3x on statistics control traffic → 4x flow entry installation



3 VNs , linear topology (5 switches), flow monitoring with ONOS controller

Challenges summary

Challenges

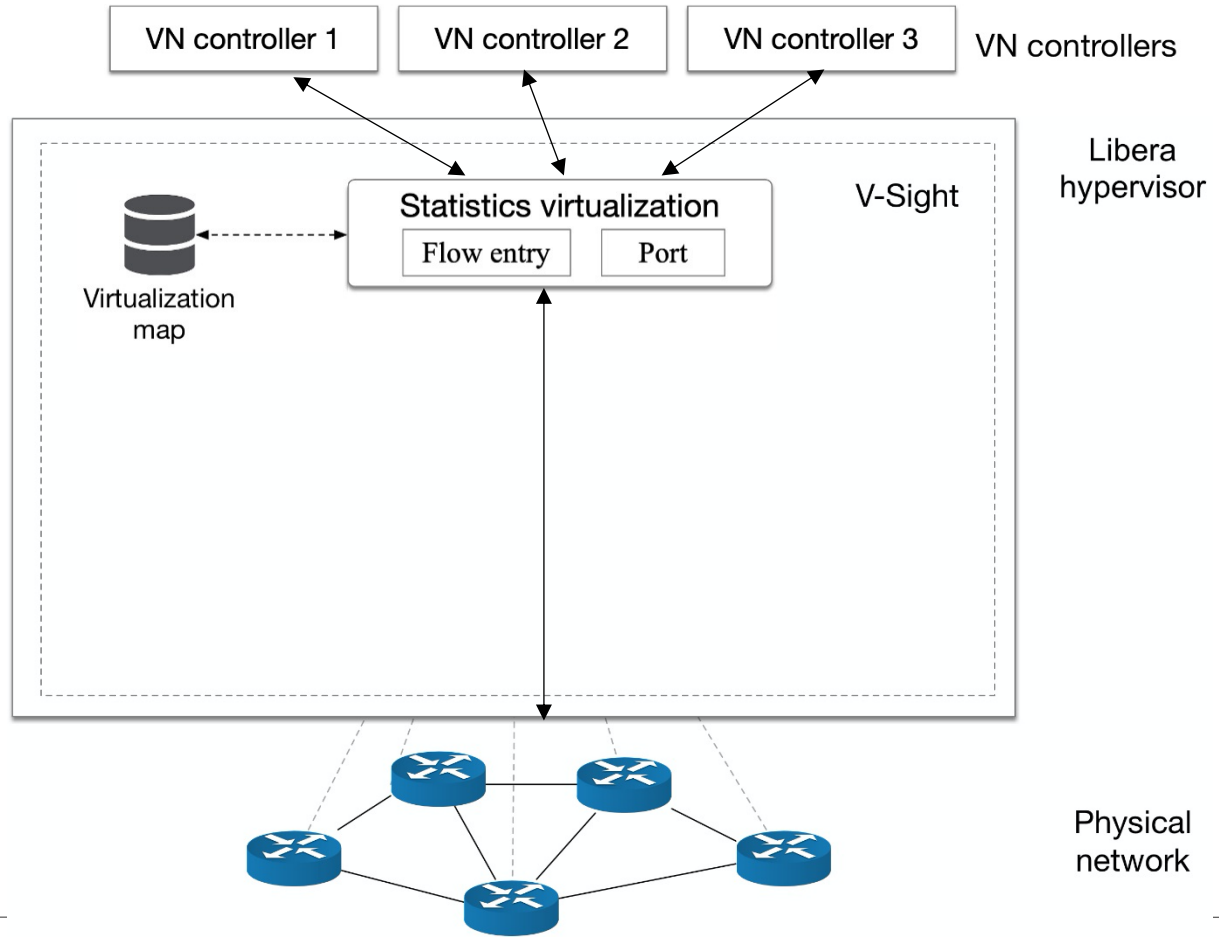
- Non-isolated statistics
- High transmission delay
- Excessive control channel consumption

V-Sight

Challenges	Design
Non-isolated statistics	Statistics virtualization
High transmission delay	Transmission disaggregation
Excessive control channel consumption	pCollector aggregation

Statistics virtualization

- Algorithms for isolated flow entry and port statistics



Statistics virtualization

Algorithm 1: Per-tenant flow entry statistics.

Input: vf: virtual flow entry for which the VN controller requires statistics

Output: $S(vf)$: statistics of the vf

$pf = V'(vf)$
if $|V(pf)| == 1$ **then**
  $S(vf) = S(pf)$
else
  if $|V(pf)| > 1$ **then**
    E_{pf} = Find edge pf of vf
    $S(vf) = S(E_pf)$
Return $S(vf)$

Edge flow entries are isolated

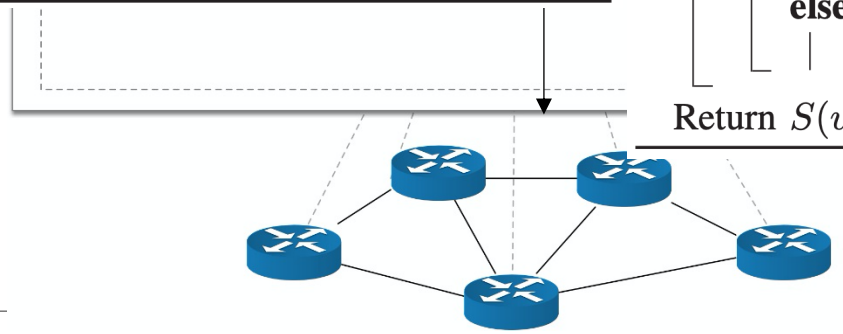
Algorithm 2: Per-tenant port statistics.

Input: vp: virtual port for which the VN controller requires statistics

...hich the vp belongs
... flow entry,
...tput port of the vf
... vp

$pp = V'(vp)$
if $|V(pp)| == 1$ **then**
  $S(vp) = S(pp)$
else
  for vf_i *belongs to* vs **do**
    if $vf_i^{in} == vp$ **then**
      $S(vp).RX+ = S(vf_i)$;
    else if $vf_i^{out} == vp$ **then**
      $S(vp).TX+ = S(vf_i)$
Return $S(vp)$

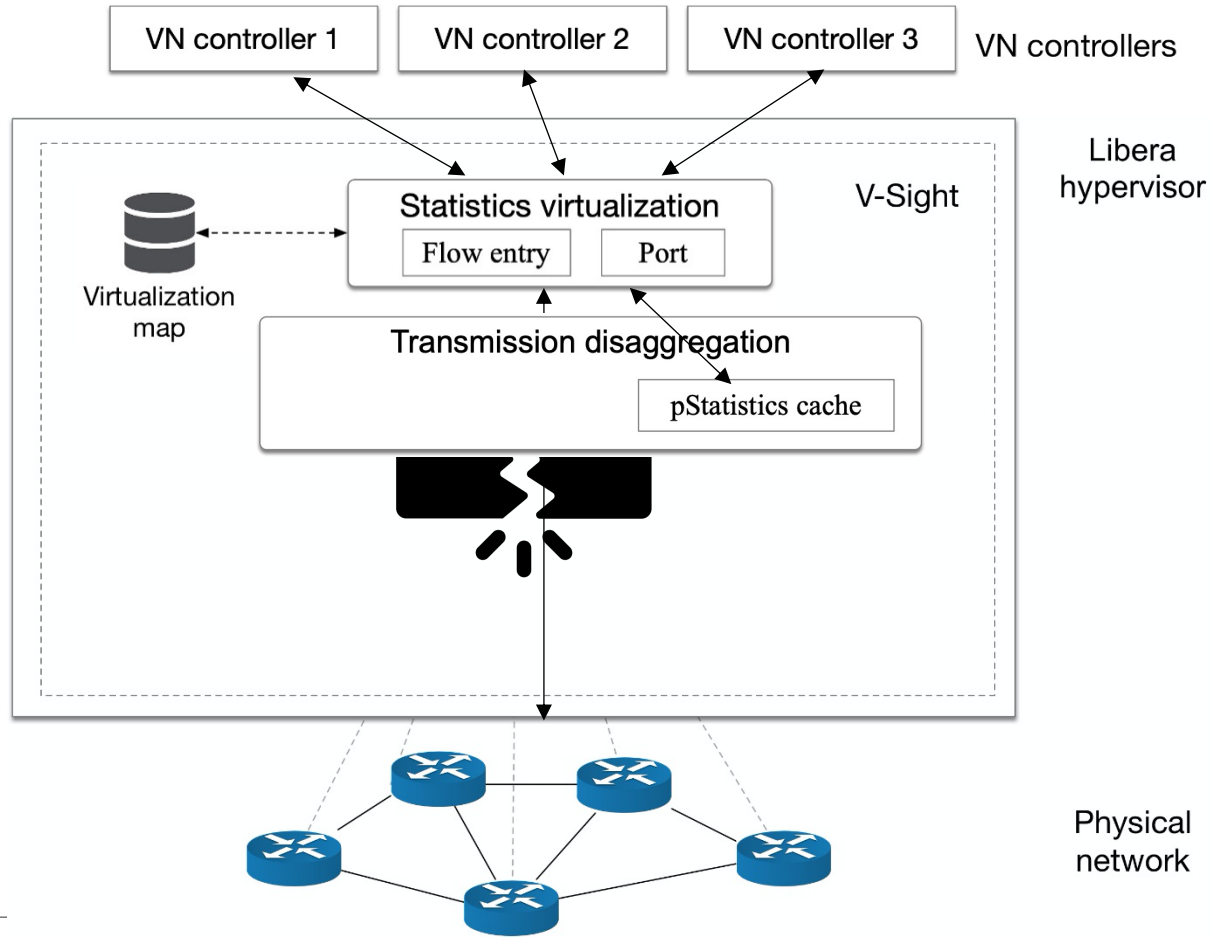
Using mapping relationships



Physical network

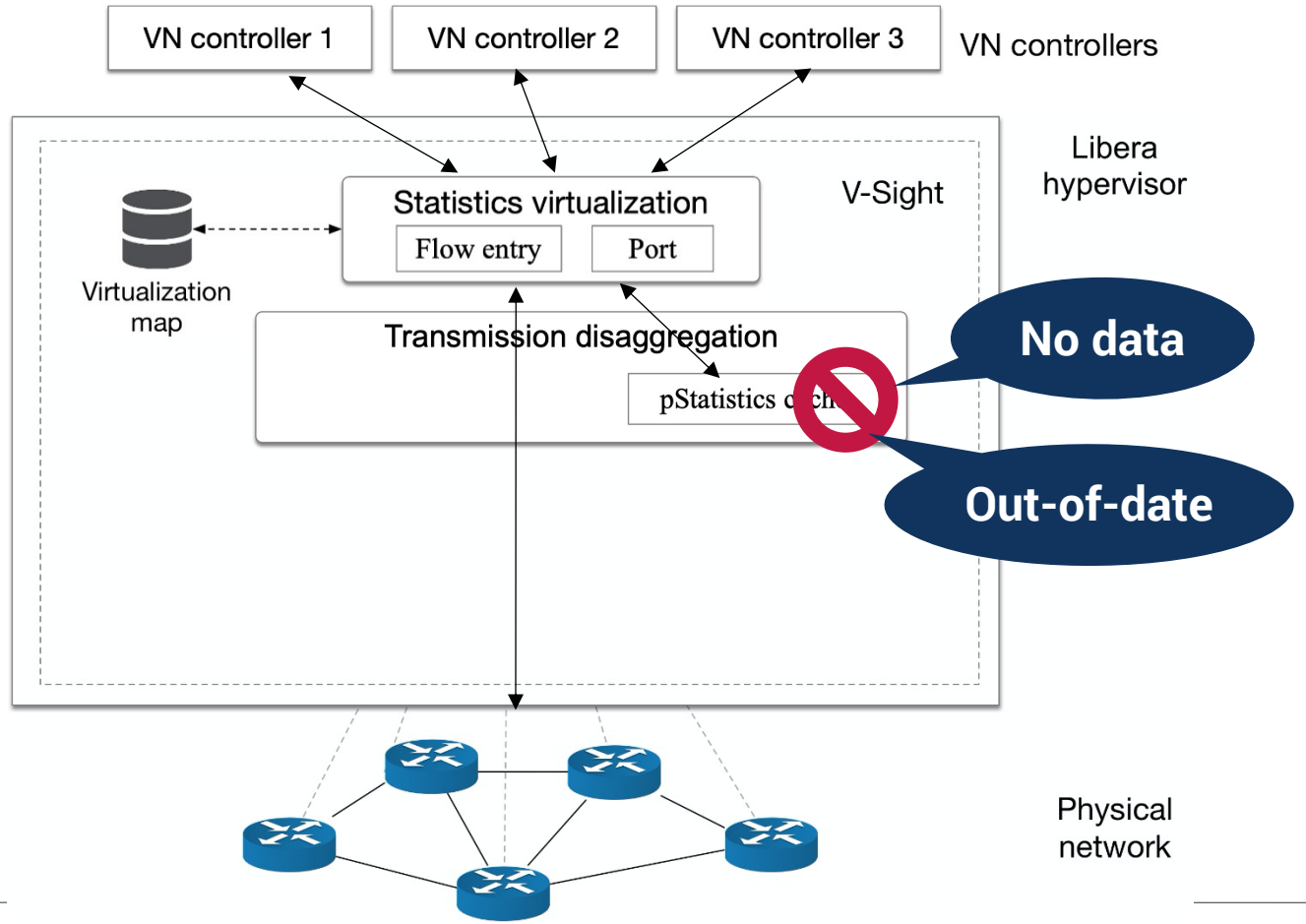
Transmission disaggregation

- Isolate statistics transmission from physical networks upon VN controller requests
- pStatistics cache: a structure that pre-stores the physical statistics



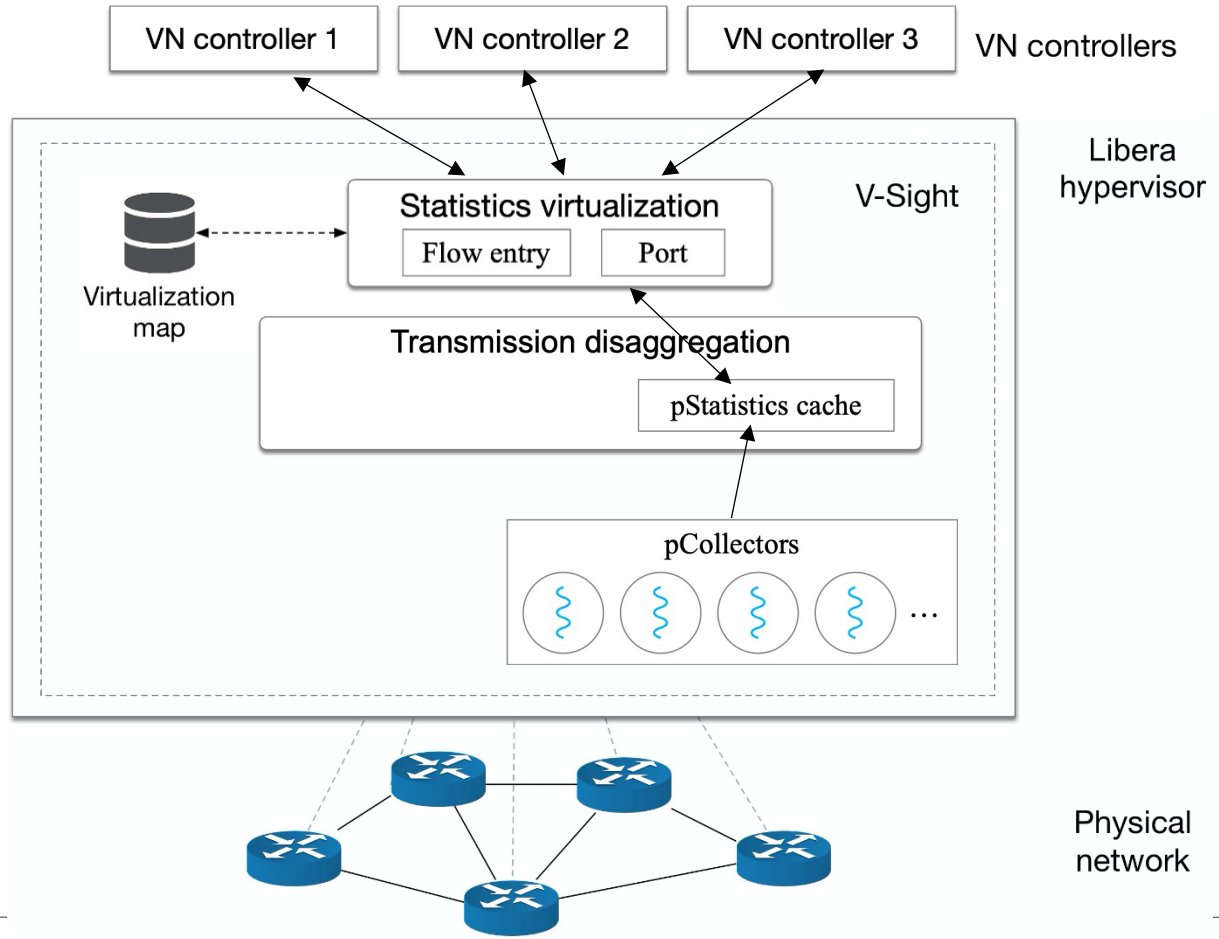
Transmission disaggregation

- pStatistics cache miss – still a high transmission delay
- Timely filling of pStatistics is important!



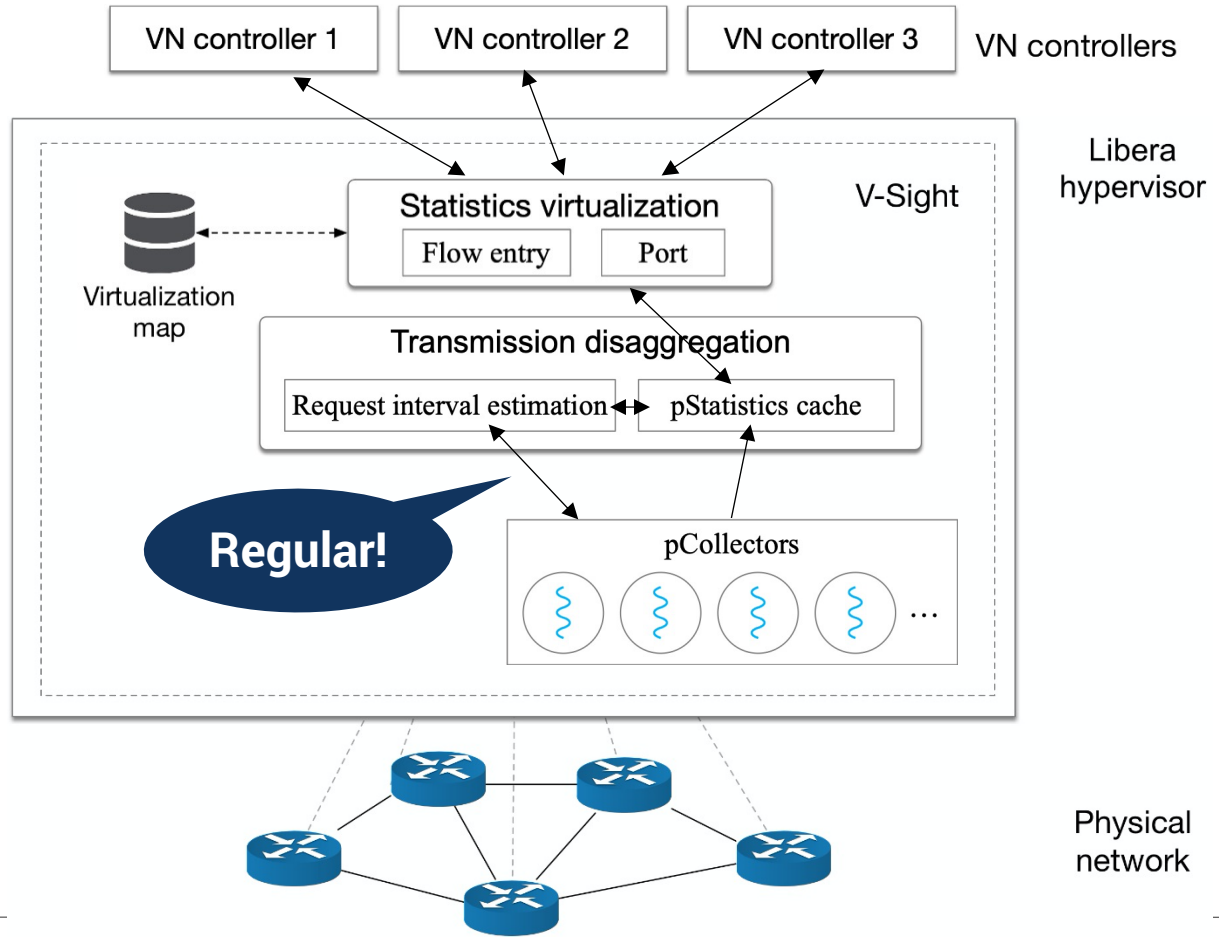
Transmission disaggregation

- pCollectors: threads to separately collect individual physical statistics



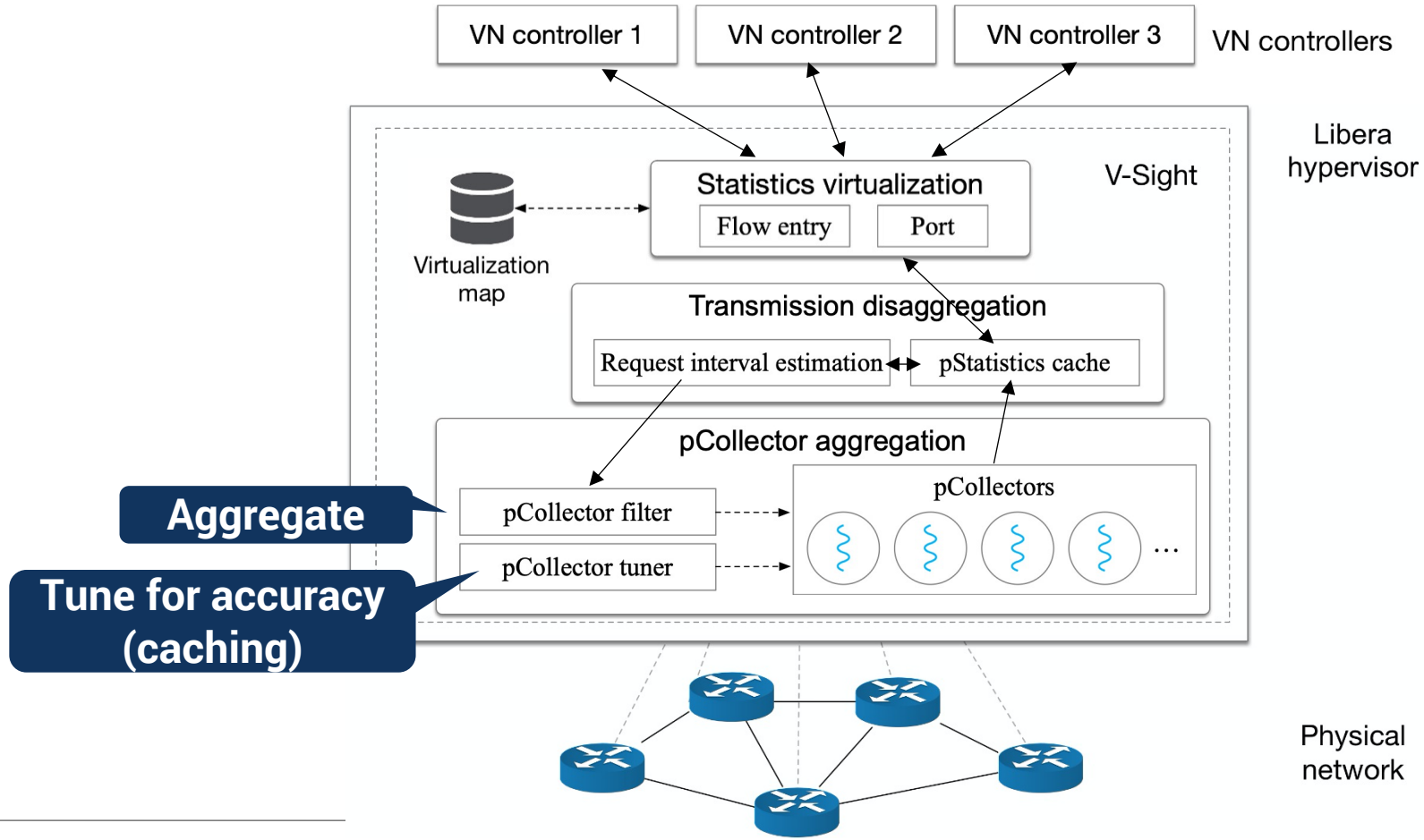
Transmission disaggregation

- Request interval learning
  1) formulate the normal distribution of virtual stats request interval
  2) determine the execution cycle of pCollector



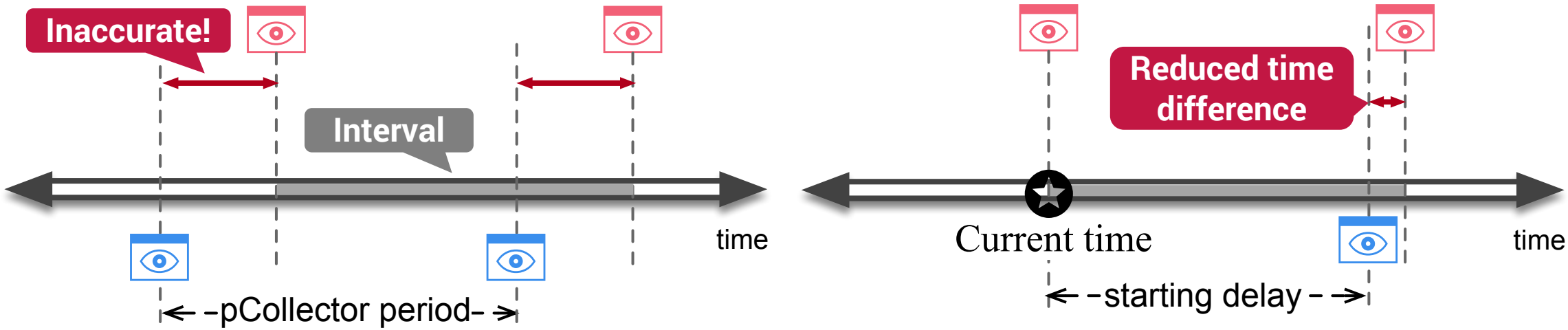
pCollector aggregation

- pCollector filter: aggregate individual pre-monitors into per-switch based pre-monitor
  - Pre-monitor for single flow entry: tiny pCollector
  - for grouped or entire flow entries: aggregated pCollector



pCollector aggregation

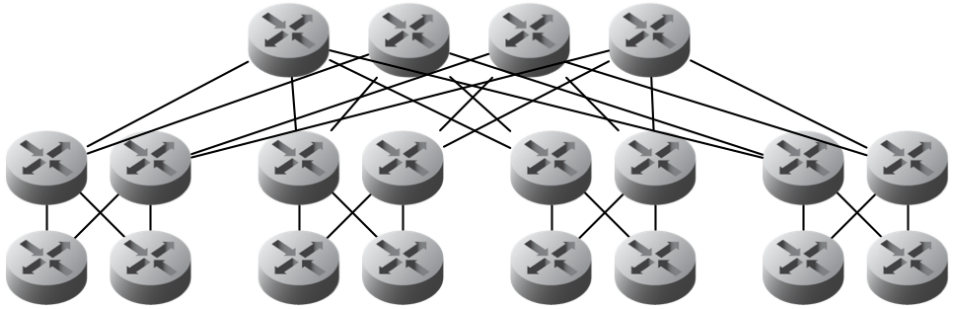
- pCollector tuner – causing additional starting delay to pCollectors



Evaluation settings

- **Network Environment**
  - Topology: Linear, 4-ary fat-tree (Open vSwitch / OpenFlow 1.3 version)
  - TCP traffic by iperf3
  - Statistics requests using default ONOS controller as VN controller*

(a) linear topology (five switches)



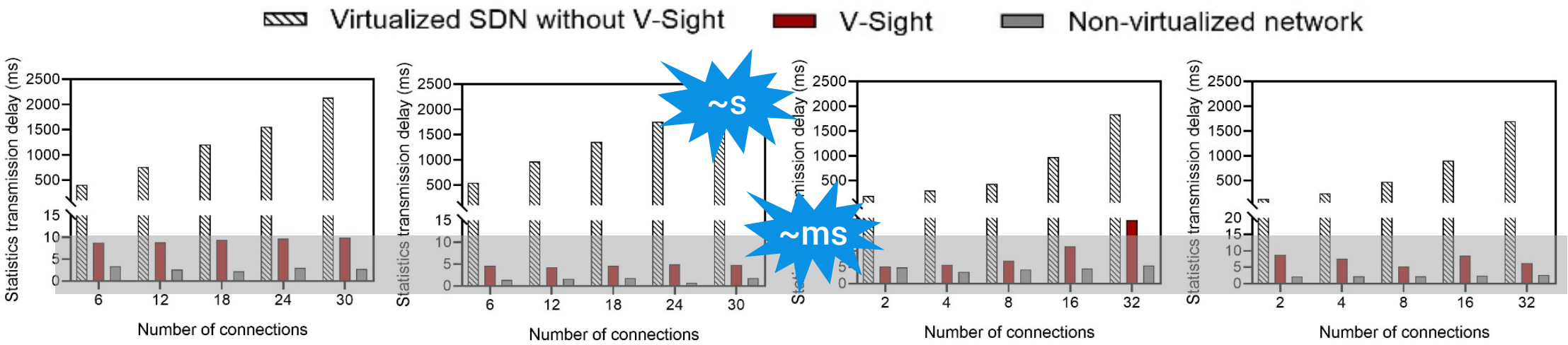
(b) fat-tree topology (4-ary)

- **Comparisons**
  1) Virtualized SDN without V-Sight: only with statistics virtualization
  2) V-Sight
  3) Native

*default settings: flow, port statistics, and 5 s intervals

Statistics transmission delay

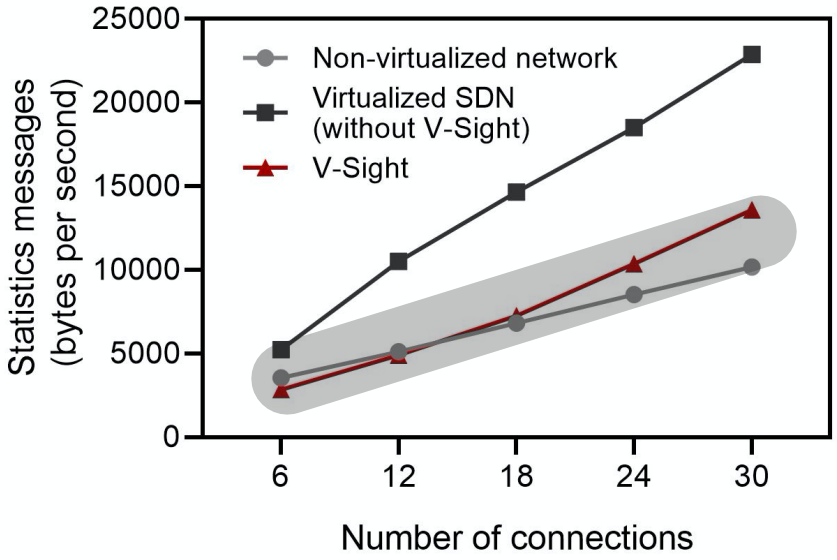
- Transmission delays comparable to those of non-virtualized networks



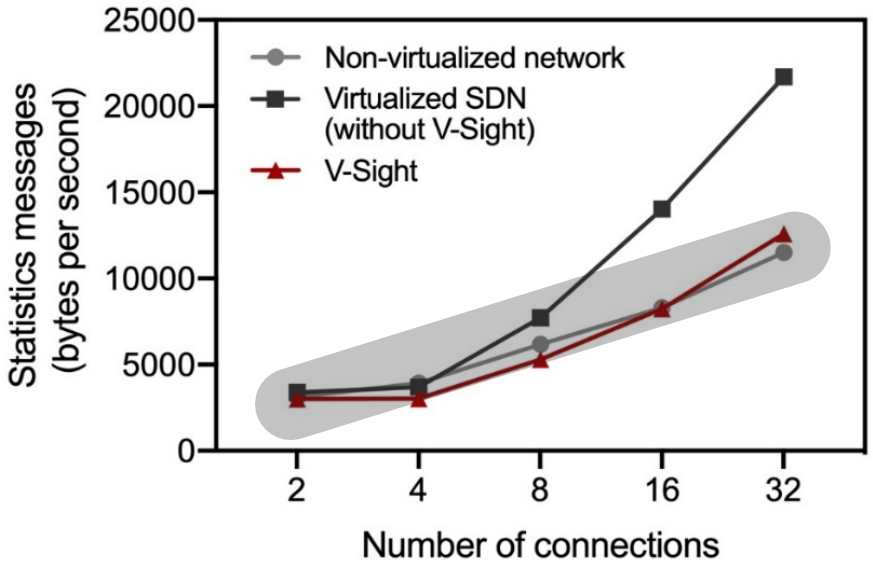
(a) Linear, flow entry (b) Linear, port (c) Fat-tree, flow entry (d) Fat-tree, port

Control channel usage

- Achieving a transmission channel usage like that of a non-virtualized network



(a) Linear topology



(b) Fat-tree topology

Conclusion

- **Critical network monitoring issues in virtualized SDN**
  - 1) Non-isolated statistics, 2) high transmission delay, 3) excessive control channel consumption
- **V-Sight: Network monitoring framework with**
  - Statistics virtualization, transmission disaggregation, pCollector aggregation
- **A level quite comparable to network monitoring for non-virtualized SDN**

THANK YOU!

Gyeongsik Yang

ksyang@os.korea.ac.kr | gsyang33.github.io

KOREA UNIVERSITY