The Case for Validating SDN Inputs

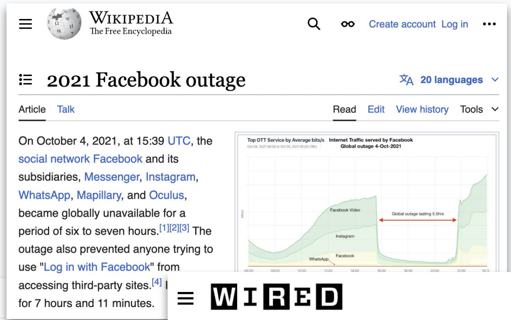
Alexander Krentsel[12], Rishabh Iyer[1], Sylvia Ratnasamy[12], Isaac Keslassy[3], Rob Shakir[2], Anees Shaikh[2]

[1]UC Berkeley, [2]Google, [3]Technion

No end in sight for WAN outages

Outages continue despite decades of experience and vast literature of best practices:

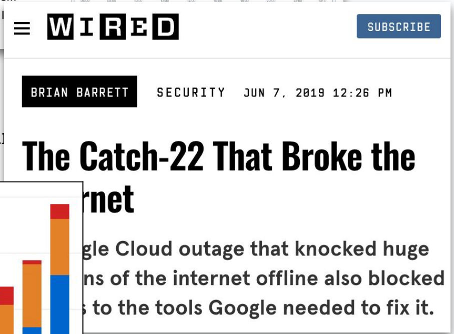
- Simulation/Emulation
  - SimBricks [SIGCOMM ’22]
  - CrystalNet [SOSP '17]
  - Mininet [HotNets ‘10]
- Testing:
  - NetCastle [NSDI ‘24]
  - Ixia (Keysight)
- Verification:
  - Batfish [NSDI ‘15]
  - Header Space Analysis [NSDI ‘12]
  - …



2021 Facebook outage

On October 4, 2021, at 15:39 UTC, the social network Facebook and its subsidiaries, Messenger, Instagram, WhatsApp, Mapillary, and Oculus, became globally unavailable for a period of six to seven hours.[1][2][3] The outage also prevented anyone trying to use "Log in with Facebook" from accessing third-party sites.[4] [illegible] for 7 hours and 11 minutes.

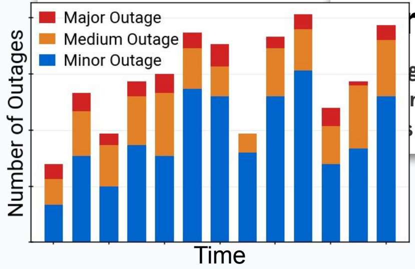
Azure outage disconnects thousands from Outlook a[illegible] Teams around the world



BRIAN BARRETT SECURITY JUN 7, 2019 12:26 PM

The Catch-22 That Broke the [illegible]rnet

[illegible]gle Cloud outage that knocked huge [illegible]ns of the internet offline also blocked [illegible] to the tools Google needed to fix it.

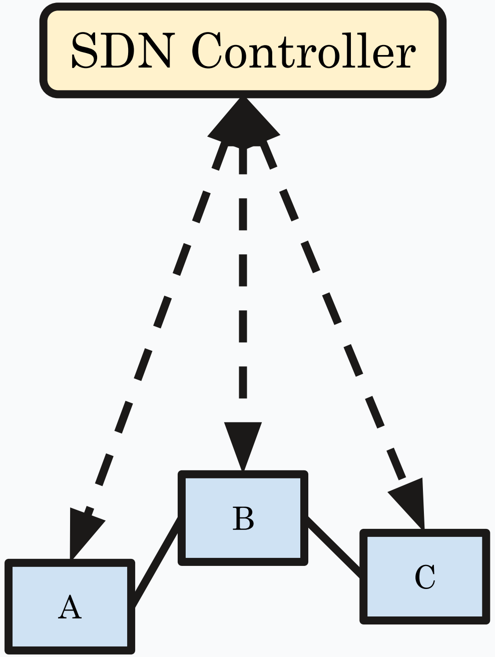


From Decentralized SDN, SIGCOMM ‘24

Why do they continue?

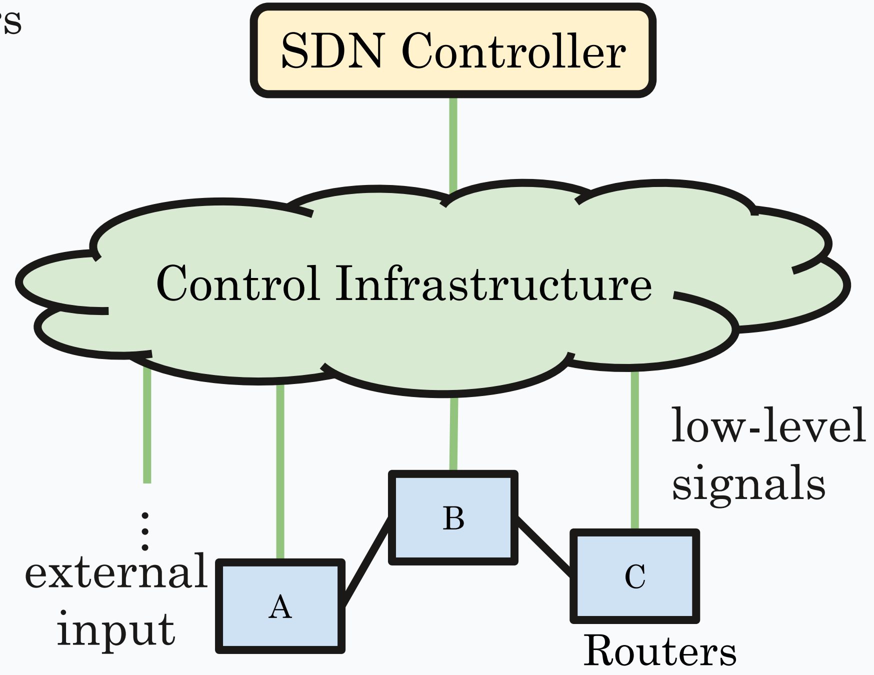
Rev. 11.18.2024

Background

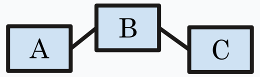


Background

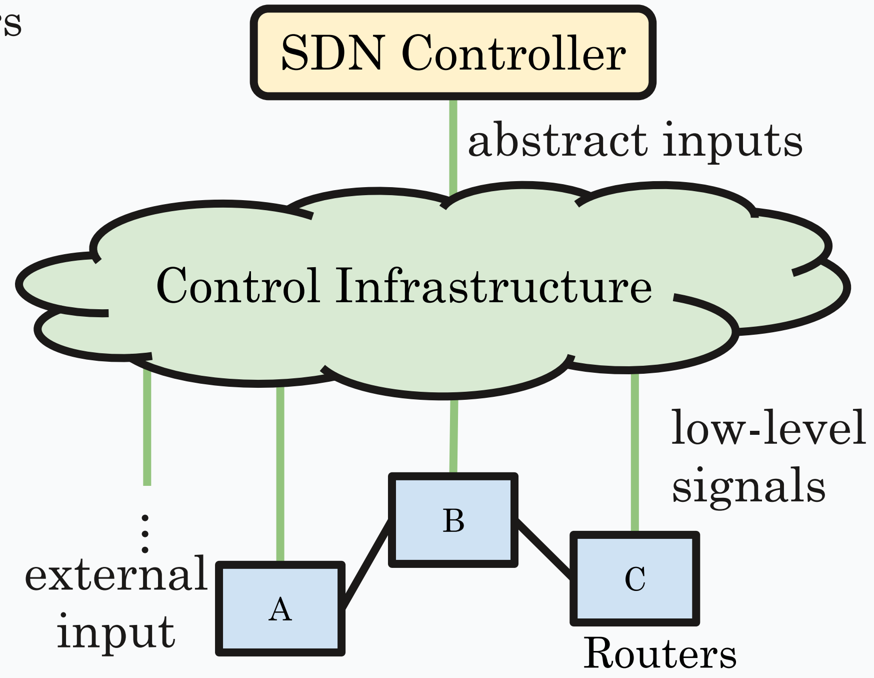
- Low-level signals collected from routers and external input
  - interface up/down status
  - host per-destination sending rates
  - …



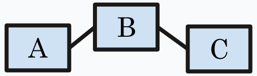
Background

- Low-level signals collected from routers and external input
  - interface up/down status
  - host per-destination sending rates
  - …
- Aggregated into abstract inputs
  - topology: 

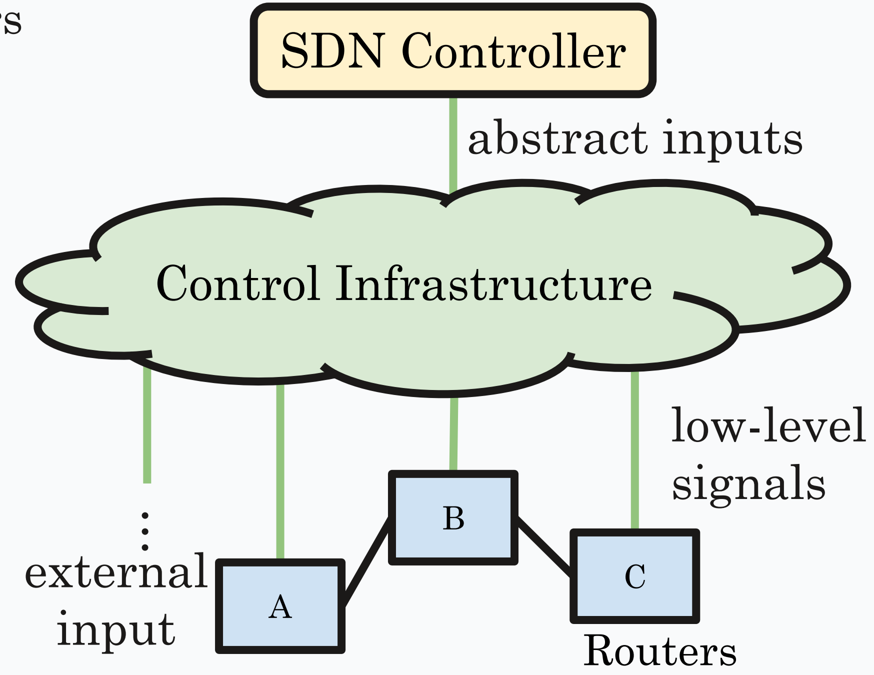
  - demand: $\begin{bmatrix} 0 & D_{ab} & D_{ac} \\ D_{ba} & 0 & D_{bc} \\ D_{ca} & D_{cb} & 0 \end{bmatrix}$



Background

- Low-level signals collected from routers and external input
  - interface up/down status
  - host per-destination sending rates
  - …
- Aggregated into abstract inputs
  - topology: 

  - demand: $\begin{bmatrix} 0 & D_{ab} & D_{ac} \\ D_{ba} & 0 & D_{bc} \\ D_{ca} & D_{cb} & 0 \end{bmatrix}$
- SDN controller computes new routes
- Routes programmed back into routers

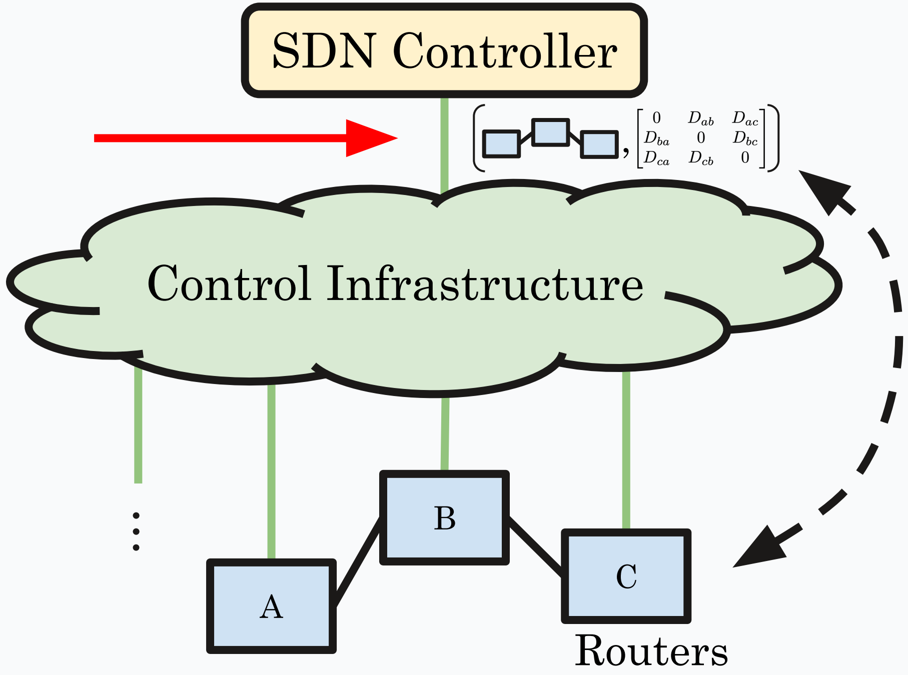


Key cause: *incorrect inputs*

Conducted analysis of high-impact SDN WAN outages over past 5 years…

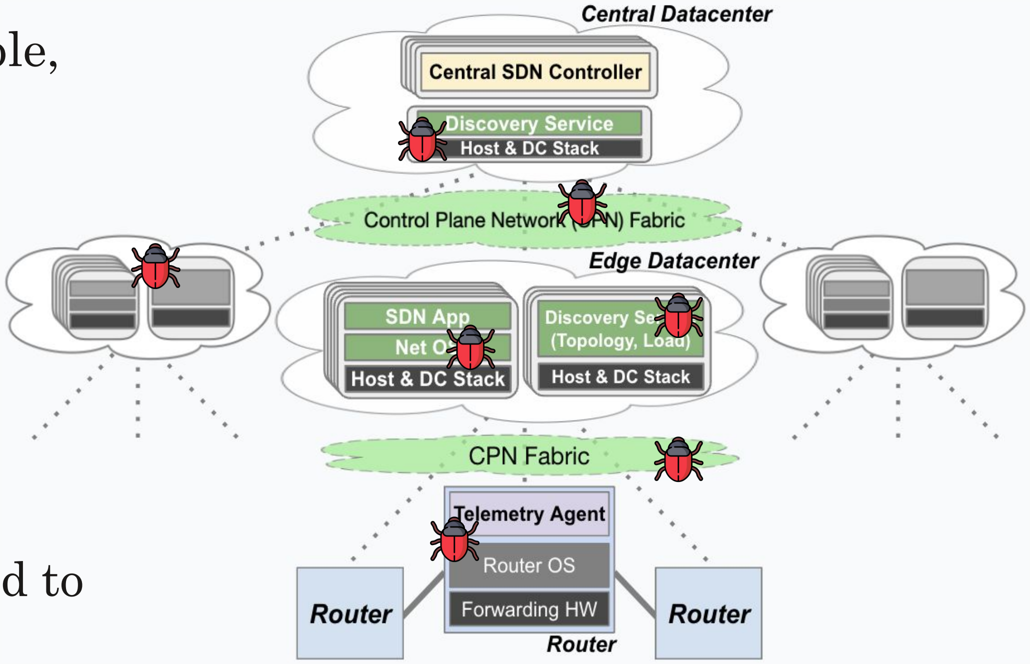
⇒ Over 1/3rd root-caused to <u>incorrect inputs</u>.

<u>incorrect</u>: do not reflect reality


SDN Controller
$\left(\ ,\begin{bmatrix} 0 & D_{ab} & D_{ac} \\ D_{ba} & 0 & D_{bc} \\ D_{ca} & D_{cb} & 0 \end{bmatrix}\right)$
Control Infrastructure
A
B
C
Routers

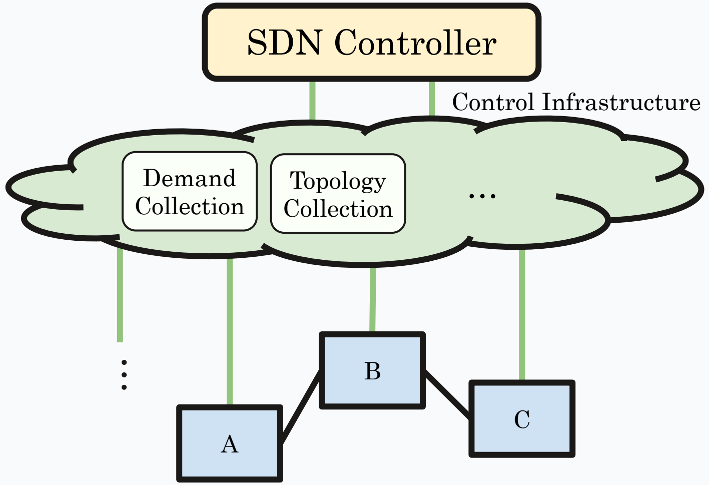
How can inputs be wrong?

Control system is conceptually simple, but practically *complex*…



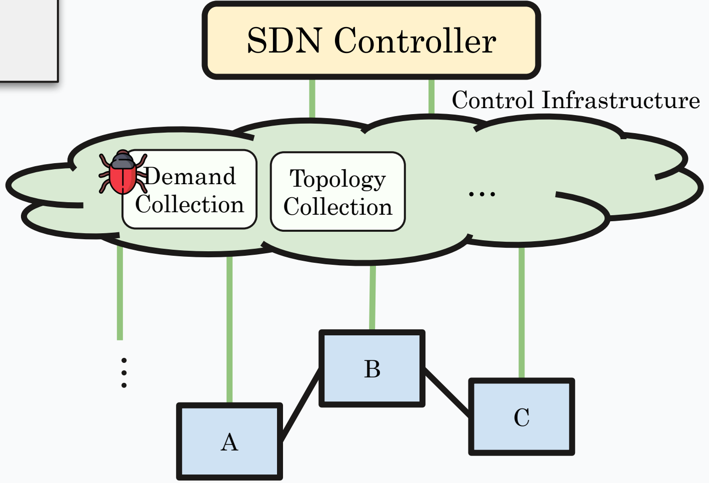
Bugs can happen *anywhere* that lead to incorrect input produced.

Real world examples…



Real world examples…

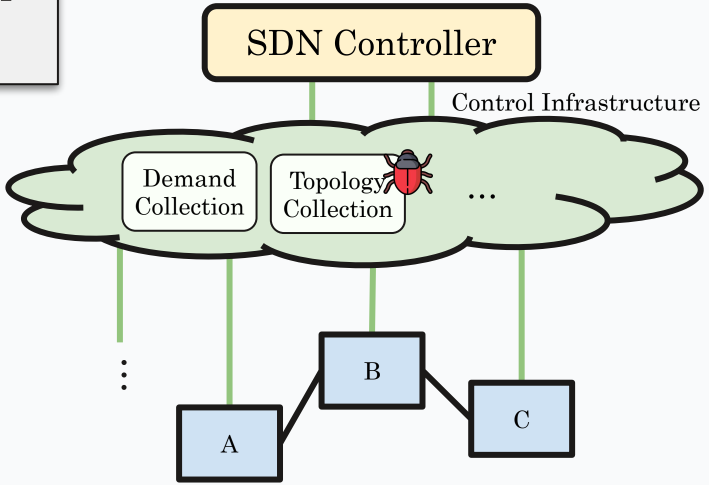
Outage 1. Bug in demand instrumentation service led to demand collected incorrectly, resulting in partial demand fed to SDN controller. Severe congestion.



Real world examples…

Outage 1. Bug in demand instrumentation service led to demand collected incorrectly, resulting in partial demand fed to SDN controller. Severe congestion.

Outage 2. Bug causes topology instrumentation service to aggregate topology before all routers report, feeding partial topology to controller.

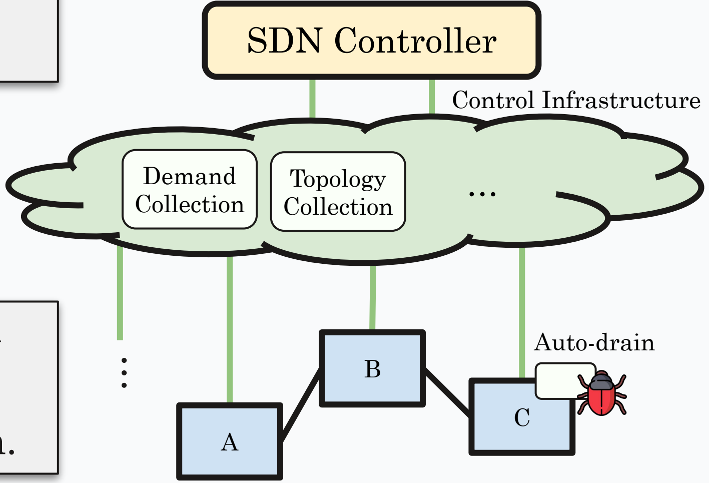


Real world examples…

Outage 1. Bug in demand instrumentation service led to demand collected incorrectly, resulting in partial demand fed to SDN controller. Severe congestion.

Outage 2. Bug causes topology instrumentation service to aggregate topology before all routers report, feeding partial topology to controller.

Outage 3. Overly-sensitive automatic drain signal gets triggered, incorrectly draining a significant number of perfectly-fine routers. Severe congestion.



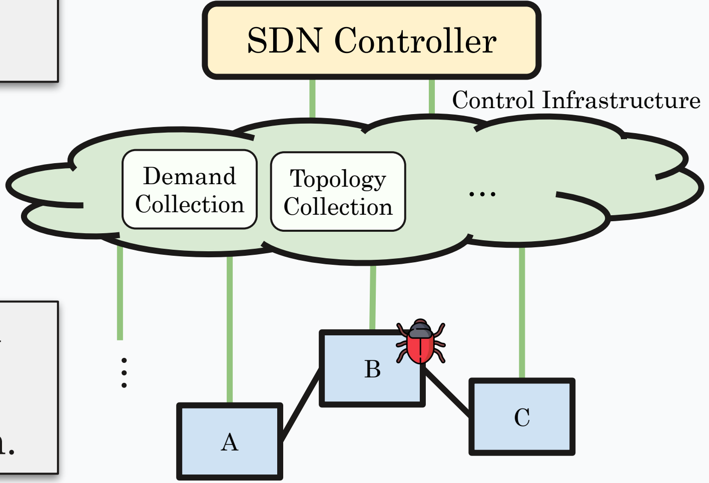
Real world examples…

Outage 1. Bug in demand instrumentation service led to demand collected incorrectly, resulting in partial demand fed to SDN controller. Severe congestion.

Outage 2. Bug causes topology instrumentation service to aggregate topology before all routers report, feeding partial topology to controller.

Outage 3. Overly-sensitive automatic drain signal gets triggered, incorrectly draining a significant number of perfectly-fine routers. Severe congestion.

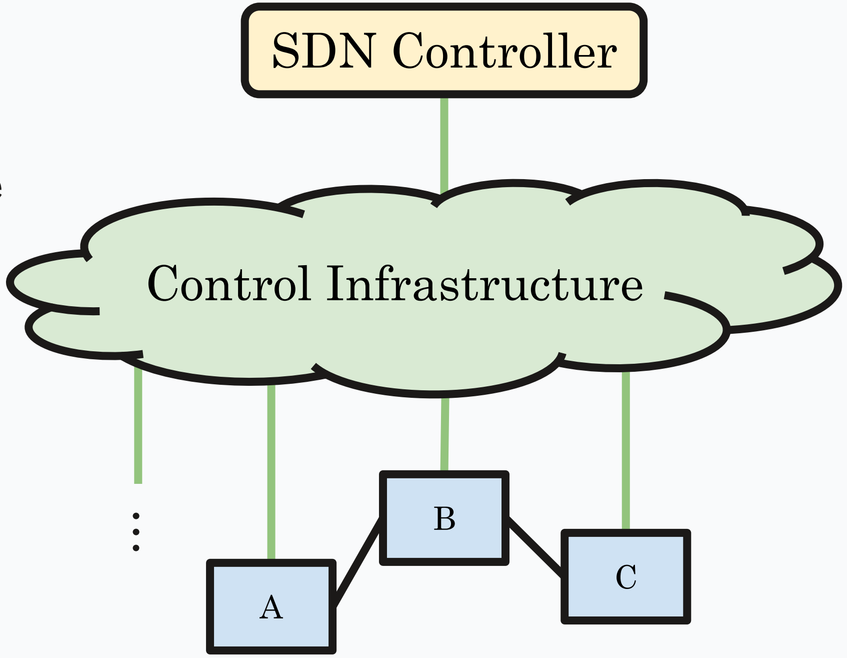
(small) Outage 4. New router OS rollout for vendor X causes telemetry messages to oscillate between 0 and actual. Triggers flap protection drains.



Why is it not caught today?

Eng. teams check for outliers/anomalies, but this misses the underlying issue:

...incorrect inputs are often possible values but *not current values*.



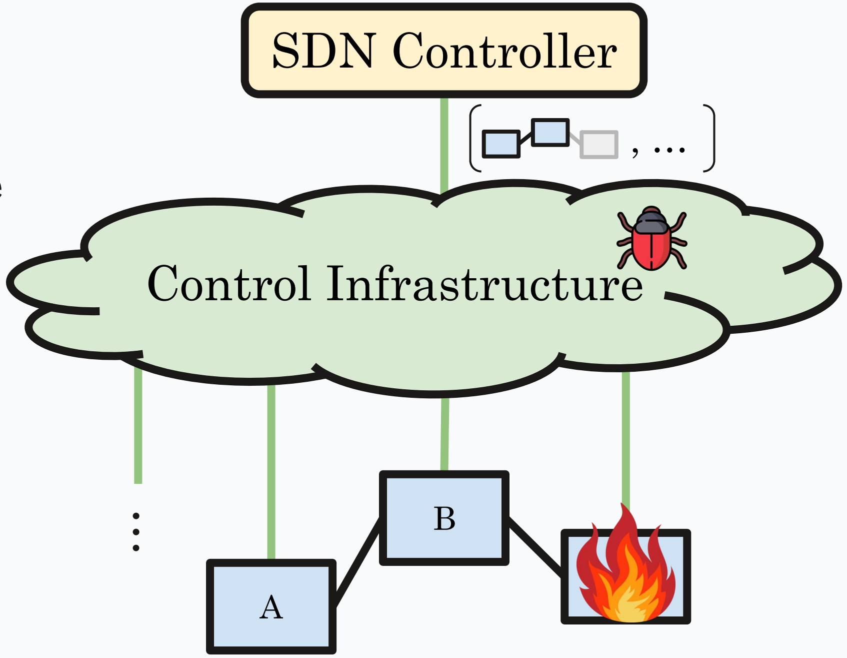
Why is it not caught today?

Eng. teams check for outliers/anomalies, but this misses the underlying issue:

…incorrect inputs are often possible values but *not current values*.

No way to distinguish from input alone.

Requires an alternative approach…

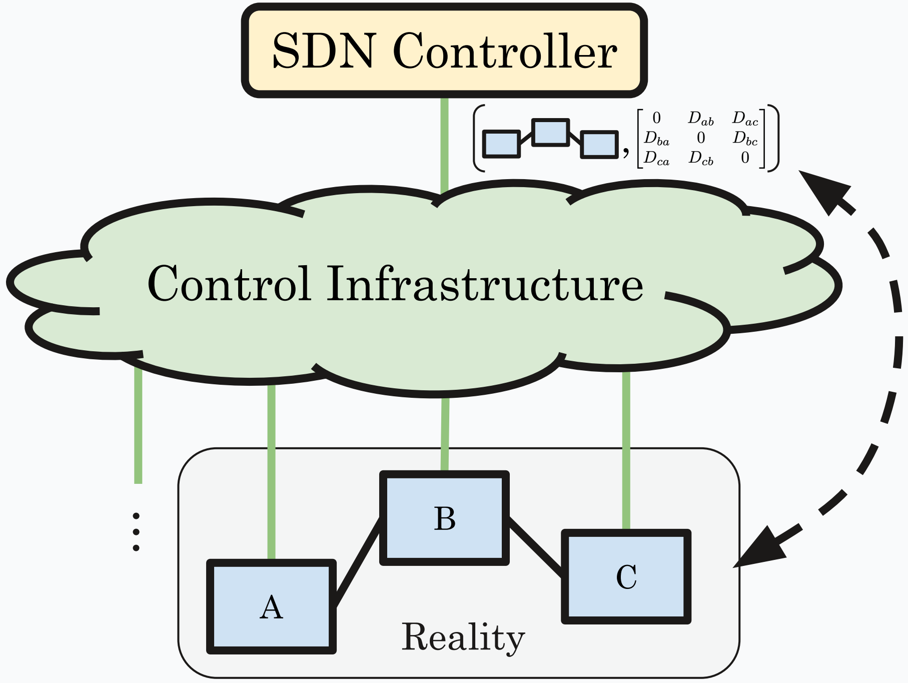


Validating controller inputs

Goal: validate that the abstract inputs agree with reality.

What is *reality*?

- Whatever we know is happening at the routers, "ground truth"

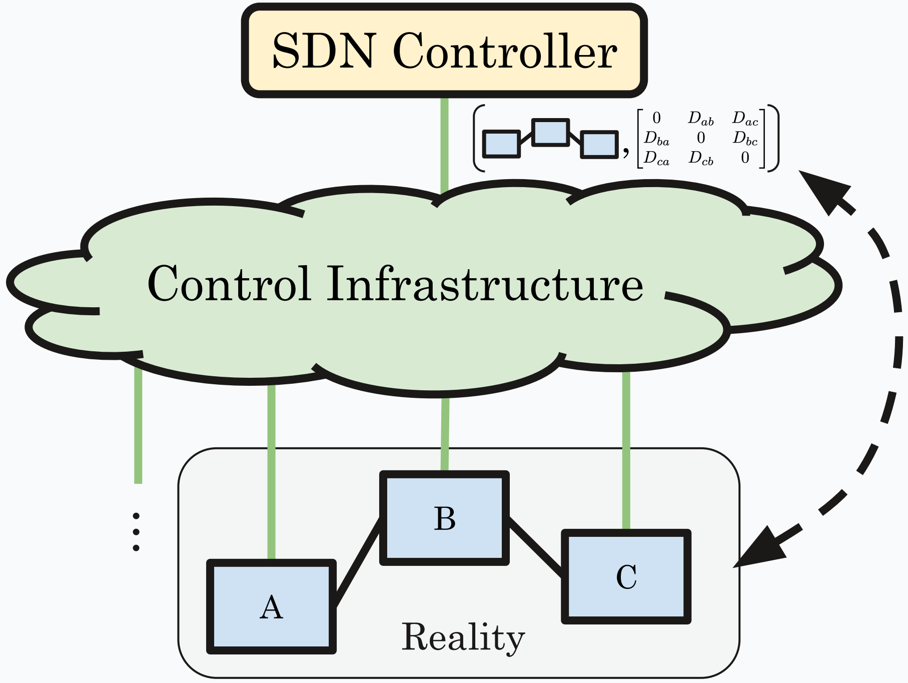

SDN Controller
$\left(\ , \begin{bmatrix} 0 & D_{ab} & D_{ac} \\ D_{ba} & 0 & D_{bc} \\ D_{ca} & D_{cb} & 0 \end{bmatrix}\right)$
Control Infrastructure
B
A
C
Reality

Rich view of reality

Many low-level signals available:

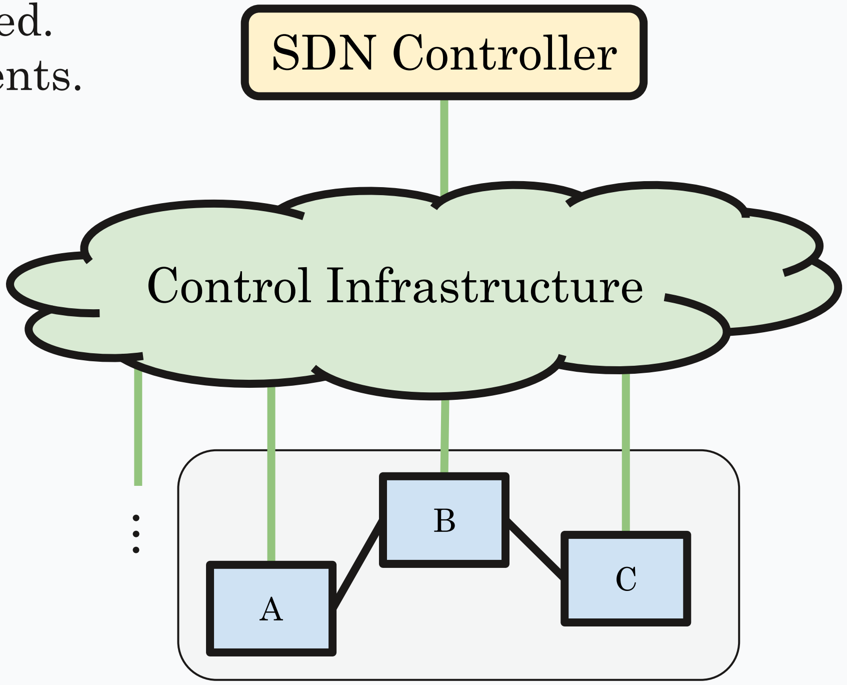
- Interface byte counters
- Packet drop counters
- Forwarding entries
- Bidirectional Forwarding Detection (BFD) link monitoring updates
- Probes
- …

Recursive problem: all sources complex, how can we trust the signals?


SDN Controller
$\left(\cdot, \begin{bmatrix} 0 & D_{ab} & D_{ac} \\ D_{ba} & 0 & D_{bc} \\ D_{ca} & D_{cb} & 0 \end{bmatrix} \right)$
Control Infrastructure
A
B
C
Reality

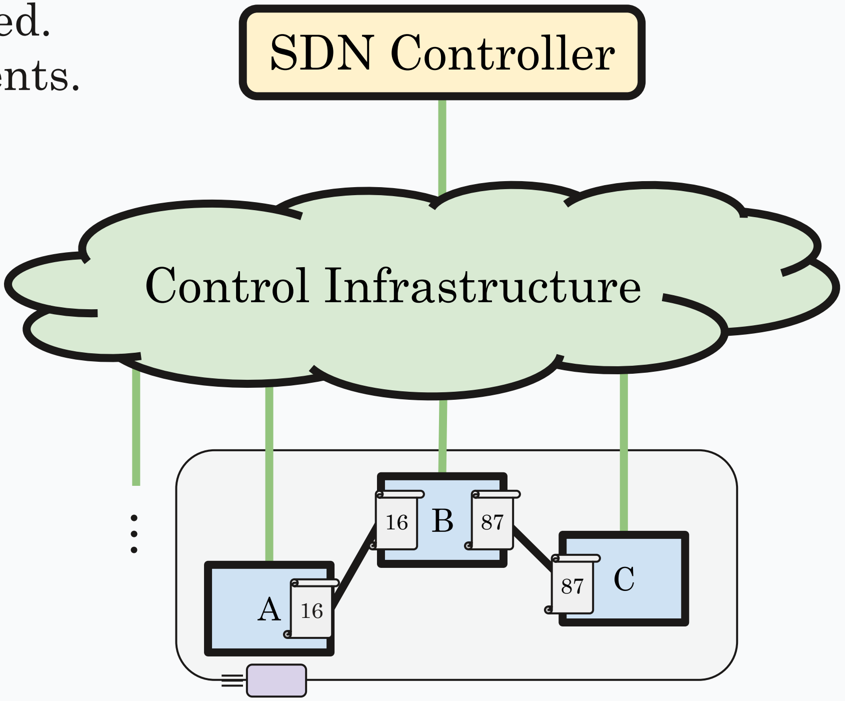
Making low-level signals trustworthy

Network signals are naturally interconnected.
→ Actions reflected in multiple measurements.



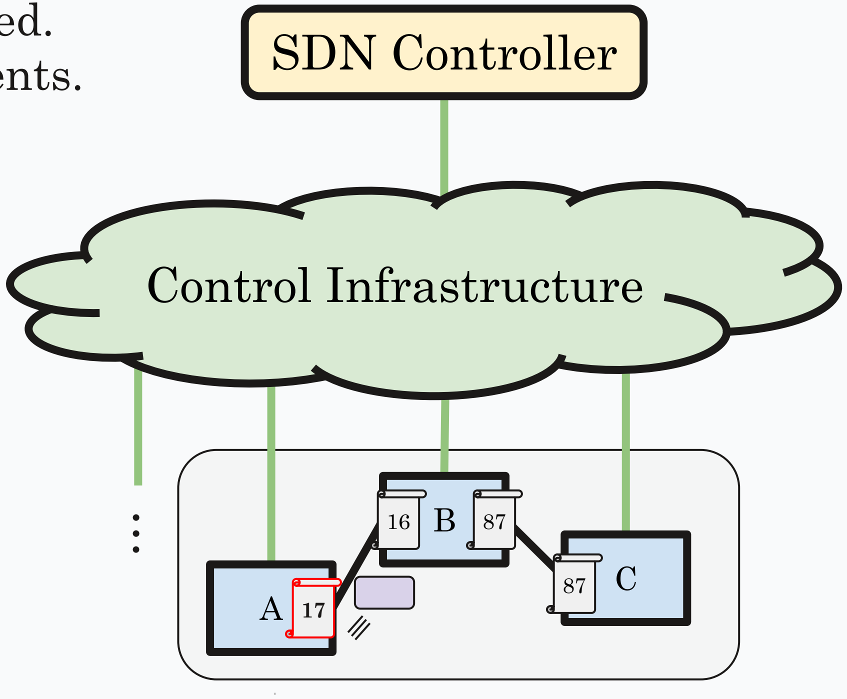
Making low-level signals trustworthy

Network signals are naturally interconnected.
→ Actions reflected in multiple measurements.



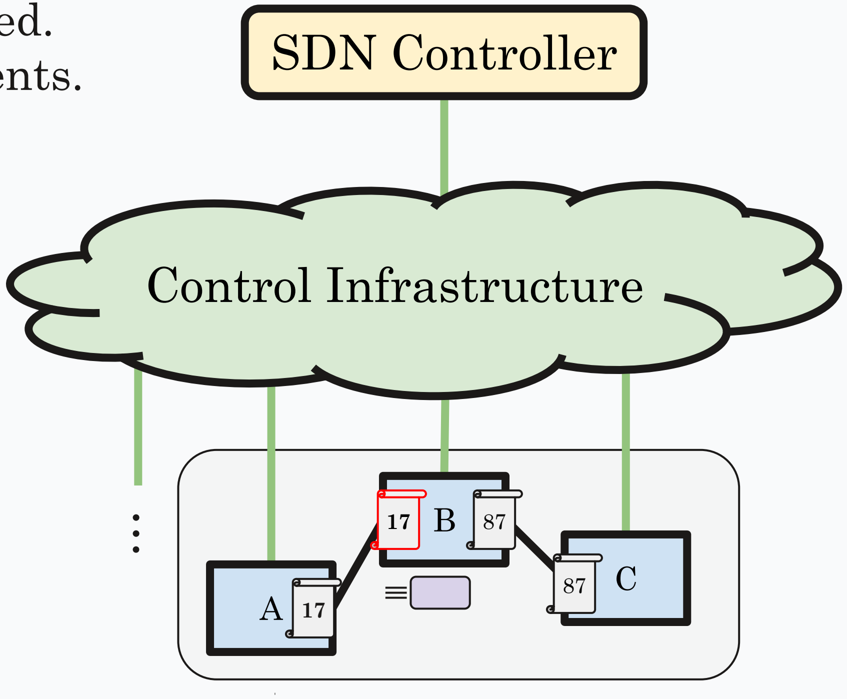
Making low-level signals trustworthy

Network signals are naturally interconnected.
→ Actions reflected in multiple measurements.



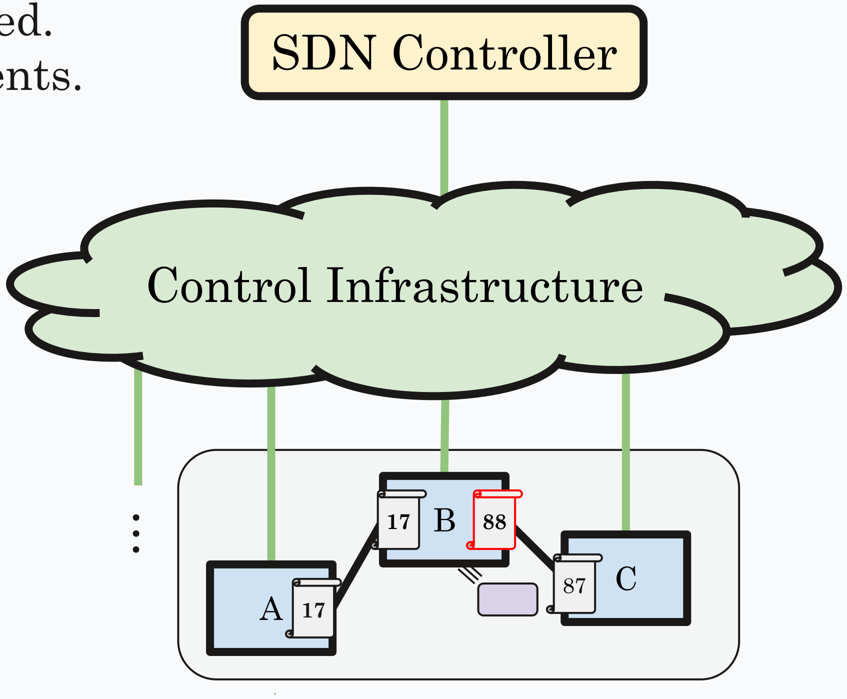
Making low-level signals trustworthy

Network signals are naturally interconnected.
→ Actions reflected in multiple measurements.



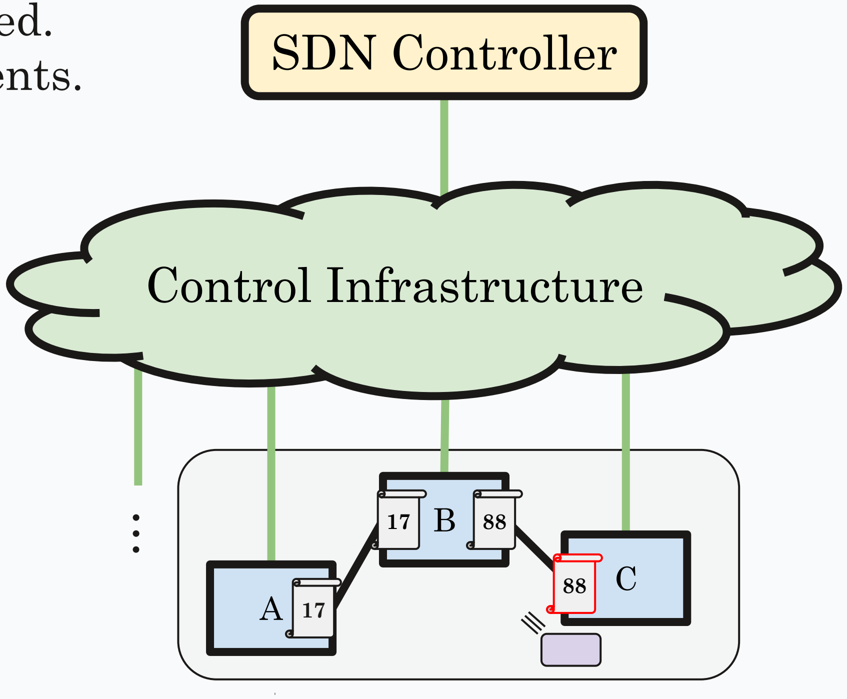
Making low-level signals trustworthy

Network signals are naturally interconnected.
$\rightarrow$ Actions reflected in multiple measurements.



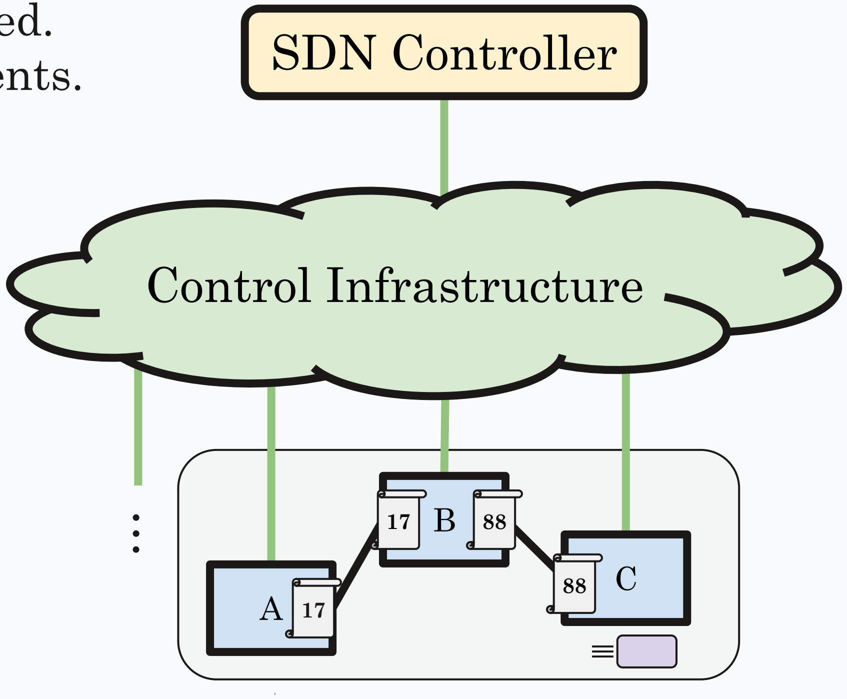
Making low-level signals trustworthy

Network signals are naturally interconnected.
→ Actions reflected in multiple measurements.



Making low-level signals trustworthy

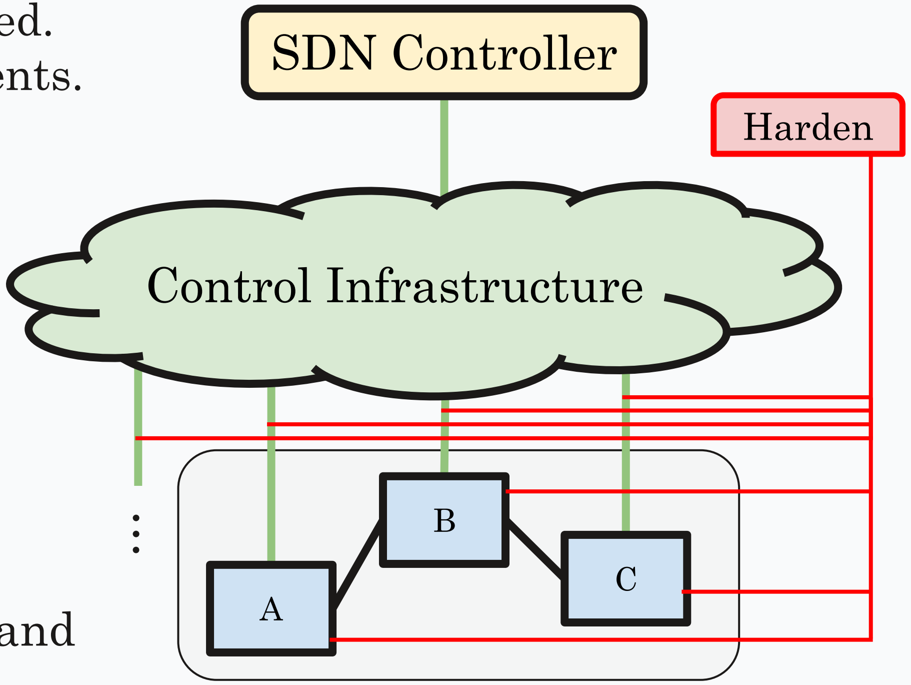
Network signals are naturally interconnected.
→ Actions reflected in multiple measurements.



Making low-level signals trustworthy

Network signals are naturally interconnected.
$\rightarrow$ Actions reflected in multiple measurements.

Use redundancy to check that reported network state agrees *with itself*, then even self-repair.

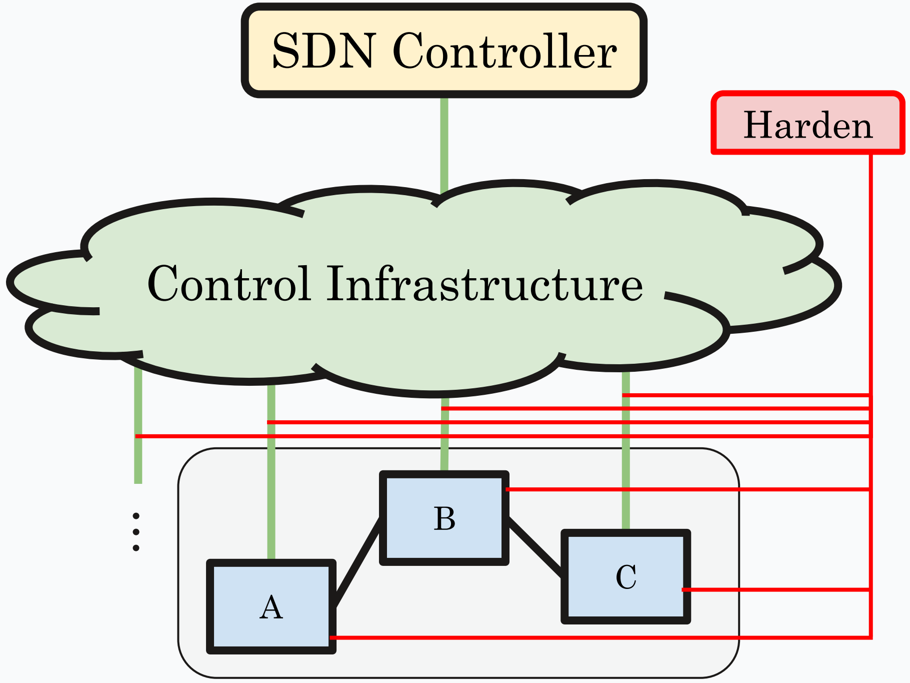


Practically: can collect all known signals, and catch & repair inconsistencies: “harden”

Input validation: agreement with high-level inputs

Once we have hardened signals, can use them to do input validation.

- Check that expected relations hold between abstract inputs and low-level signals.

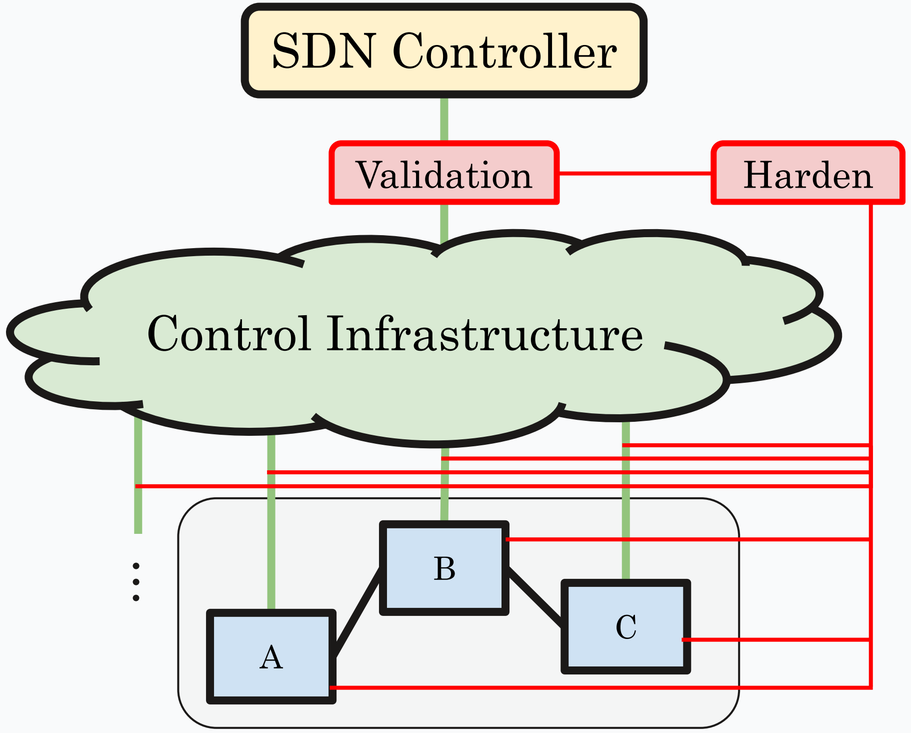


Input validation: agreement with high-level inputs

Once we have hardened signals, can use them to do input validation.

- Check that expected relations hold between abstract inputs and low-level signals.

We present our system, **Hodor**, that performs continuous validation.



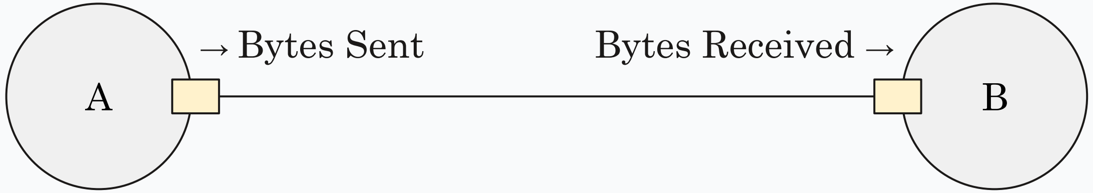
Hodor: (1) Hardening low-level signals

Many examples of redundancy in network data…

Hodor: (1) Hardening low-level signals

Many examples of redundancy in network data…

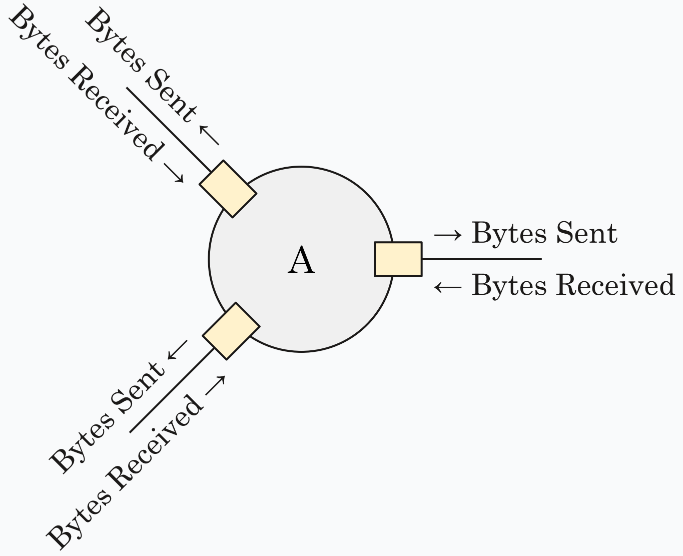
- **R1. Symmetry** across two ends of link: bytes in = bytes out.



Hodor: (1) Hardening low-level signals

Many examples of redundancy in network data…

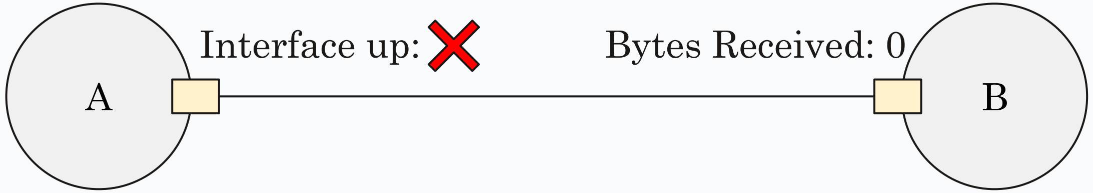
- **R1. Symmetry** across two ends of link: bytes in = bytes out.
- **R2. Flow conservation** across ports: sum in = sum out at a router.



Hodor: (1) Hardening low-level signals

Many examples of redundancy in network data…

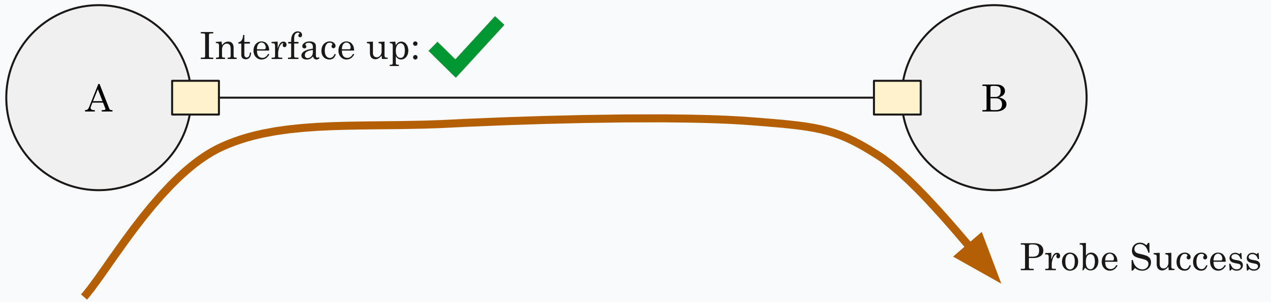
- **R1. Symmetry** across two ends of link: bytes in = bytes out.
- **R2. Flow conservation** across ports: sum in = sum out at a router.
- **R3. Alternative signal** substitutes: link down ⇒ intf count = 0.



Hodor: (1) Hardening low-level signals

Many examples of redundancy in network data…

- **R1. Symmetry** across two ends of link: bytes in = bytes out.
- **R2. Flow conservation** across ports: sum in = sum out at a router.
- **R3. Alternative signal** substitutes: link down ⇒ intf count = 0.
- **R4. Manufactured signals**: probe confirms link up/down.

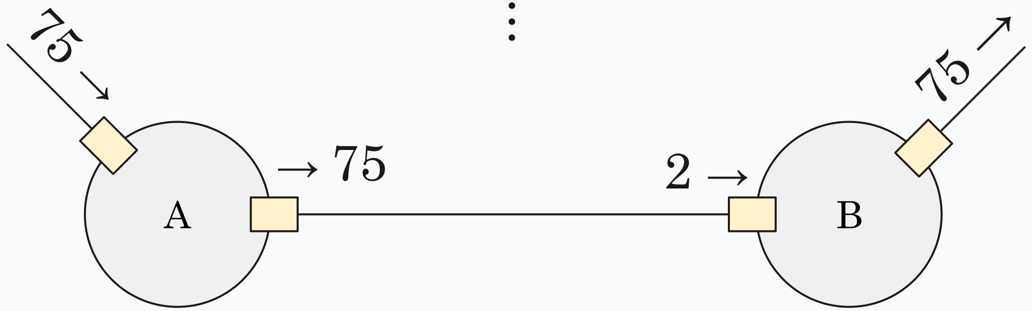


Hodor: (1) Hardening low-level signals

Many examples of redundancy in network data…

- **R1. Symmetry** across two ends of link: bytes in = bytes out.
- **R2. Flow conservation** across ports: sum in = sum out at a router.
- **R3. Alternative signal** substitutes: link down ⇒ intf count = 0.
- **R4. Manufactured signals**: probe confirms link up/down.

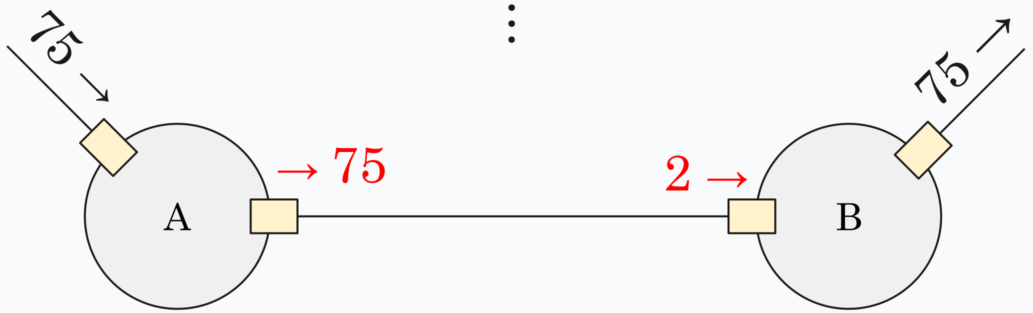
⋮



Hodor: (1) Hardening low-level signals

Many examples of redundancy in network data...

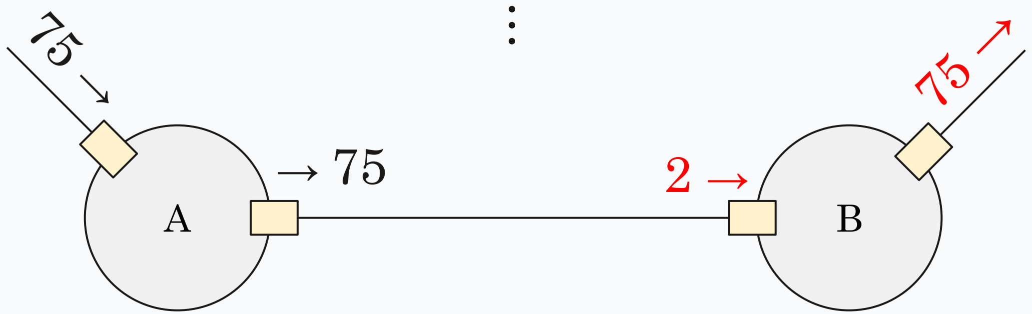
- **R1. Symmetry** across two ends of link: bytes in = bytes out.
- **R2. Flow conservation** across ports: sum in = sum out at a router.
- **R3. Alternative signal** substitutes: link down ⇒ intf count = 0.
- **R4. Manufactured signals**: probe confirms link up/down.



Hodor: (1) Hardening low-level signals

Many examples of redundancy in network data…

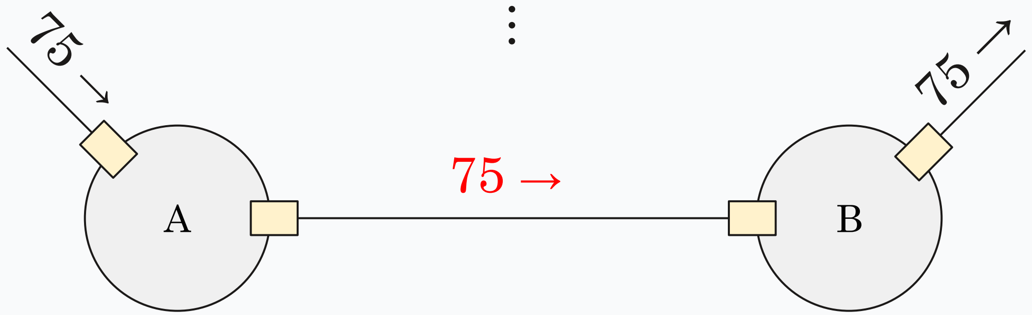
- **R1. Symmetry** across two ends of link: bytes in = bytes out.
- **R2. Flow conservation** across ports: sum in = sum out at a router.
- **R3. Alternative signal** substitutes: link down ⇒ intf count = 0.
- **R4. Manufactured signals**: probe confirms link up/down.



Hodor: (1) Hardening low-level signals

Many examples of redundancy in network data...

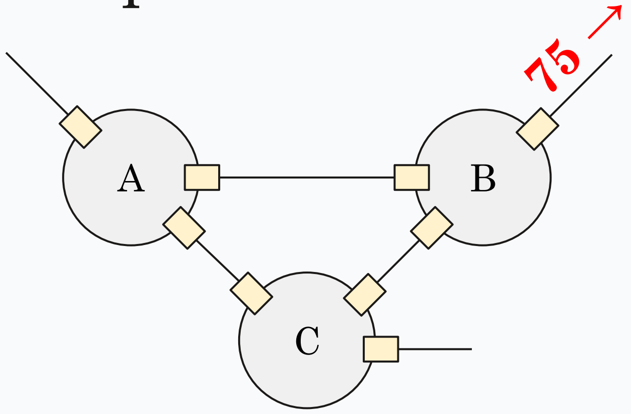
- **R1. Symmetry** across two ends of link: bytes in = bytes out.
- **R2. Flow conservation** across ports: sum in = sum out at a router.
- **R3. Alternative signal** substitutes: link down ⇒ intf count = 0.
- **R4. Manufactured signals**: probe confirms link up/down.



Hodor: (2) Agreement with high-level inputs

System expert can define relationships between low-level observations and high-level abstract inputs.

- Ex: for validating demand, sum of demand to a node == egress interface count



$$D = \begin{array}{c|ccc} & A & B & C \\ \hline A & 0 & \boxed{23} & 55 \\ B & 18 & \boxed{0} & 41 \\ C & 30 & \boxed{52} & 0 \end{array}$$

Hodor verifies that such relations hold for the given inputs and trusted snapshots.

Active questions

- How do we build this?
  - Inconsistent data due to snapshot time skew
- Alternative unsupervised learning approaches? Other strategies?
  - Masked autoencoders, symbolic regression
- Right space of response actions?
- How prevalent are incorrect inputs in other control systems?
  - Anecdotally, heard of similar problems for cloud tenant networks, datacenter networks

Alexander Krentsel – akrentsel@berkeley.edu

Paper Link