No Packet Left Behind

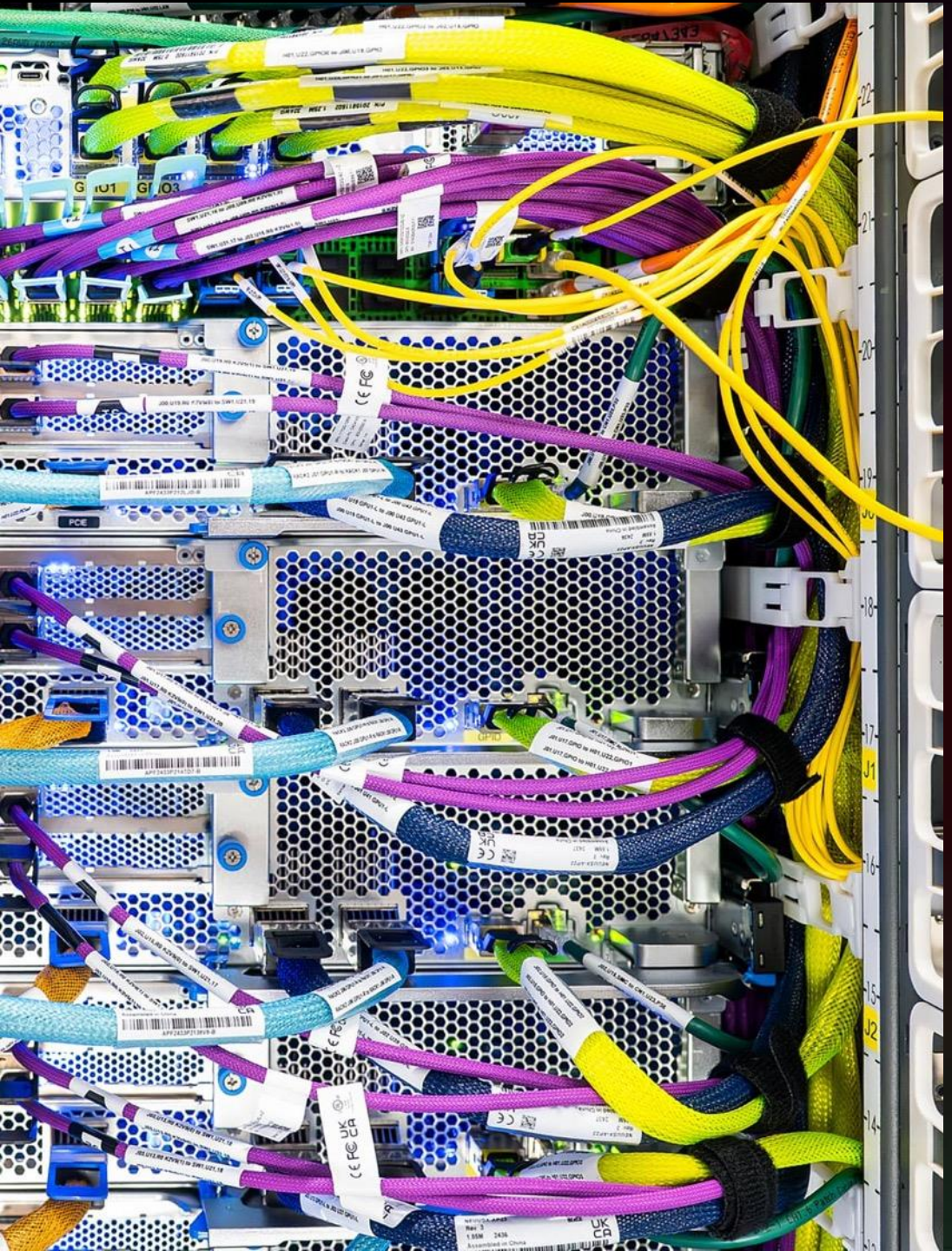
AWS's approach to building and operating reliable networks

Lincoln Dale

Senior Principal Engineer
AWS

Agenda

1. The AWS Global Network
2. AWS Network Architecture & Engineering
3. Automated Network Operations
4. Automated Deployments, Network Monitoring & Availability
5. Learnings from Problems
6. Doing this (and more) at scale



The AWS Global Network

6+

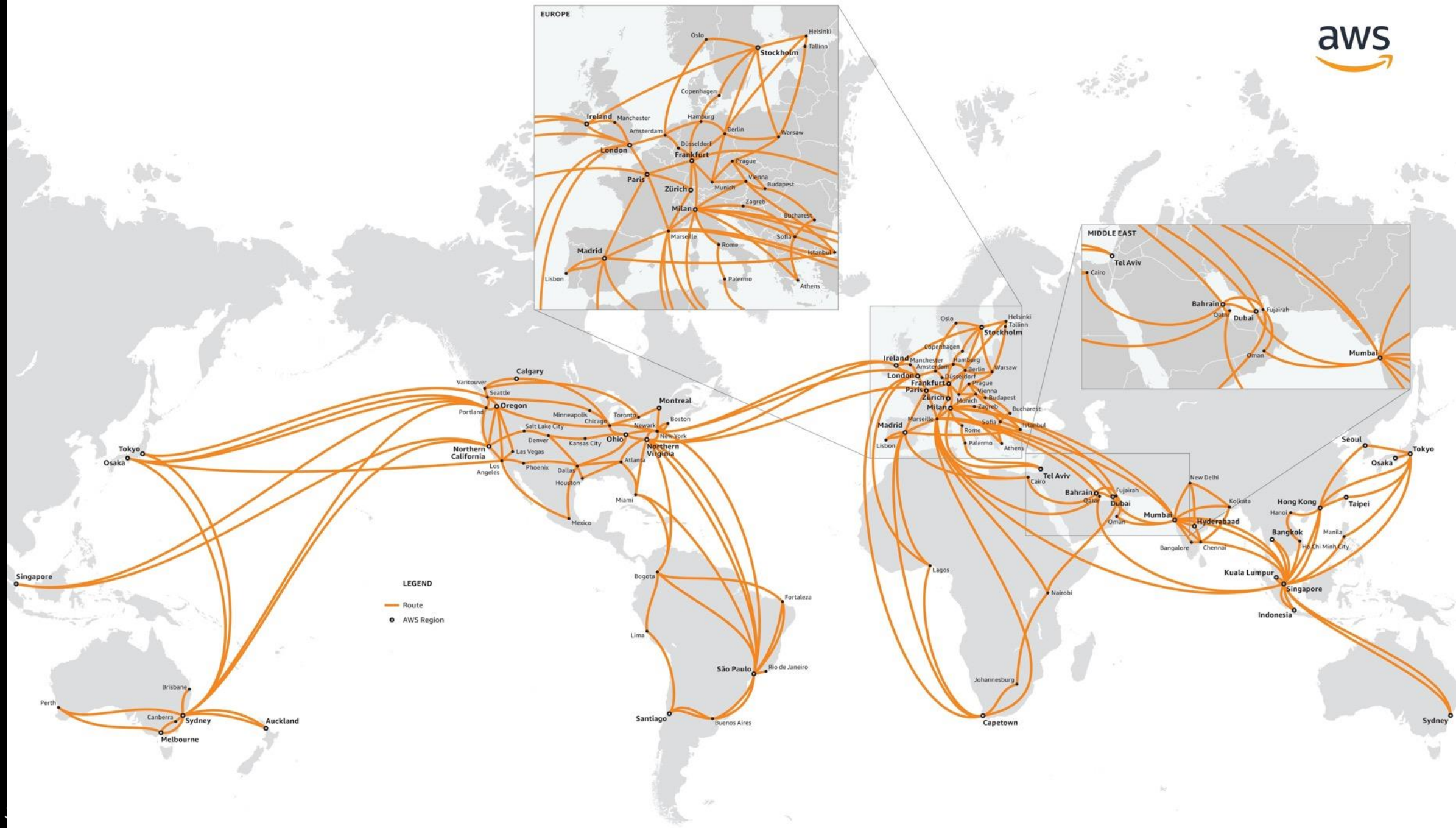
million kilometers of terrestrial and subsea fiber-optic cabling

80%

increase in AWS network capacity over the last 12 months

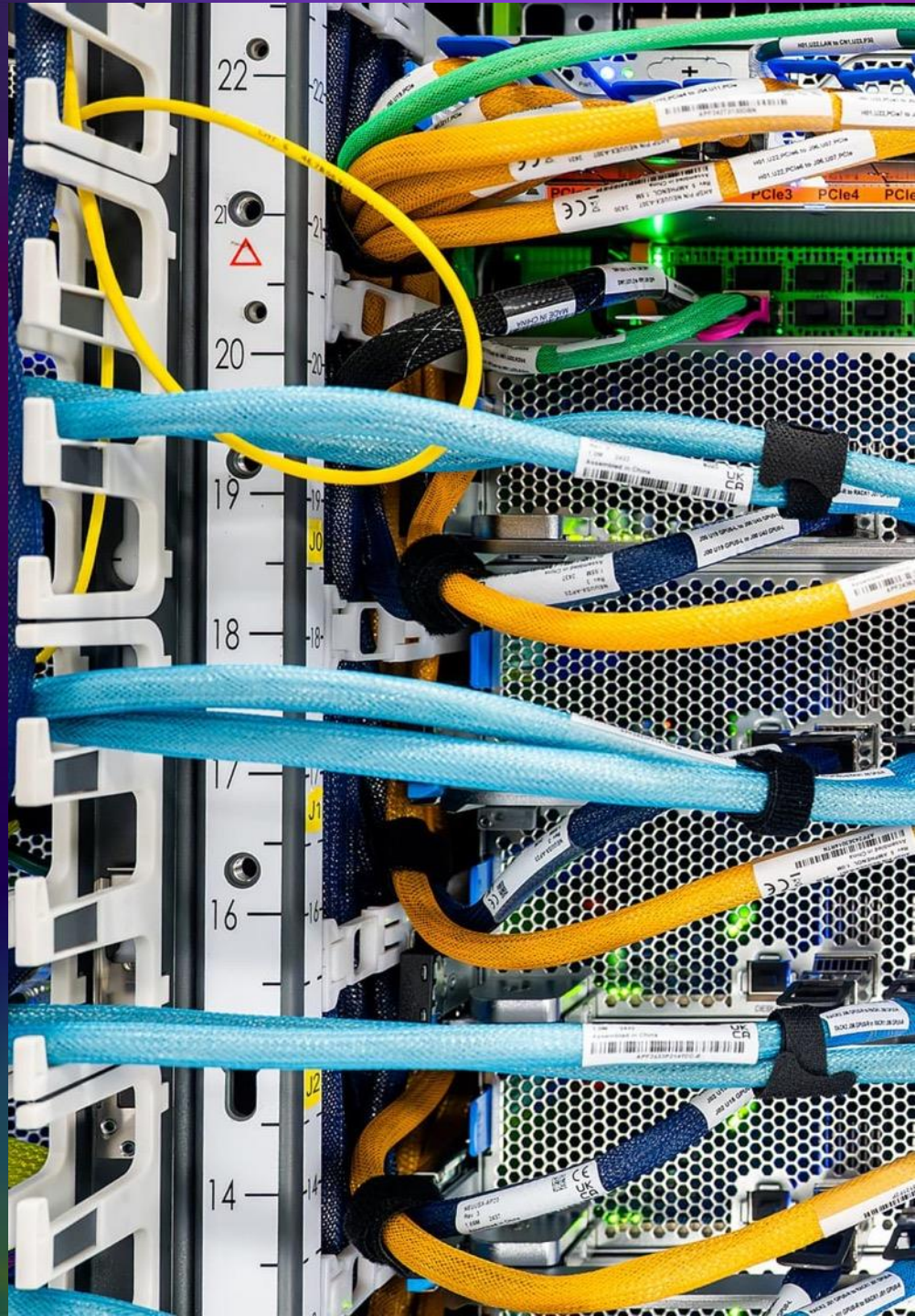
96%

of all network events are automatically remediated or mitigated without human intervention


EUROPE
Oslo
Helsinki
Tallinn
Stockholm
Copenhagen
Ireland
Manchester
Hamburg
Amsterdam
Berlin
Warsaw
London
Düsseldorf
Frankfurt
Prague
Paris
Vienna
Zürich
Munich
Budapest
Zagreb
Milan
Bucharest
Marseille
Sofia
Rome
Istanbul
Madrid
Lisbon
Palermo
Athens
MIDDLE EAST
Tel Aviv
Cairo
Bahrain
Qatar
Dubai
Fujairah
Oman
Mumbai
Calgary
Vancouver
Seattle
Portland
Oregon
Montreal
Minneapolis
Chicago
Toronto
Boston
Salt Lake City
Newark
Denver
Kansas City
Ohio
New York
Tokyo
Osaka
Northern California
Las Vegas
Northern Virginia
Los Angeles
Phoenix
Dallas
Atlanta
Houston
Miami
Mexico
Singapore
Bogota
Fortaleza
Lima
São Paulo
Rio de Janeiro
Santiago
Buenos Aires
Brisbane
Perth
Canberra
Sydney
Auckland
Melbourne
LEGEND
Route
AWS Region
Lagos
Nairobi
Johannesburg
Capetown
New Delhi
Kolkata
Hyderabaad
Bangalore
Chennai
Seoul
Hong Kong
Taipei
Hanoi
Manila
Bangkok
Ho Chi Minh City
Kuala Lumpur
Indonesia

No Packet Left Behind:

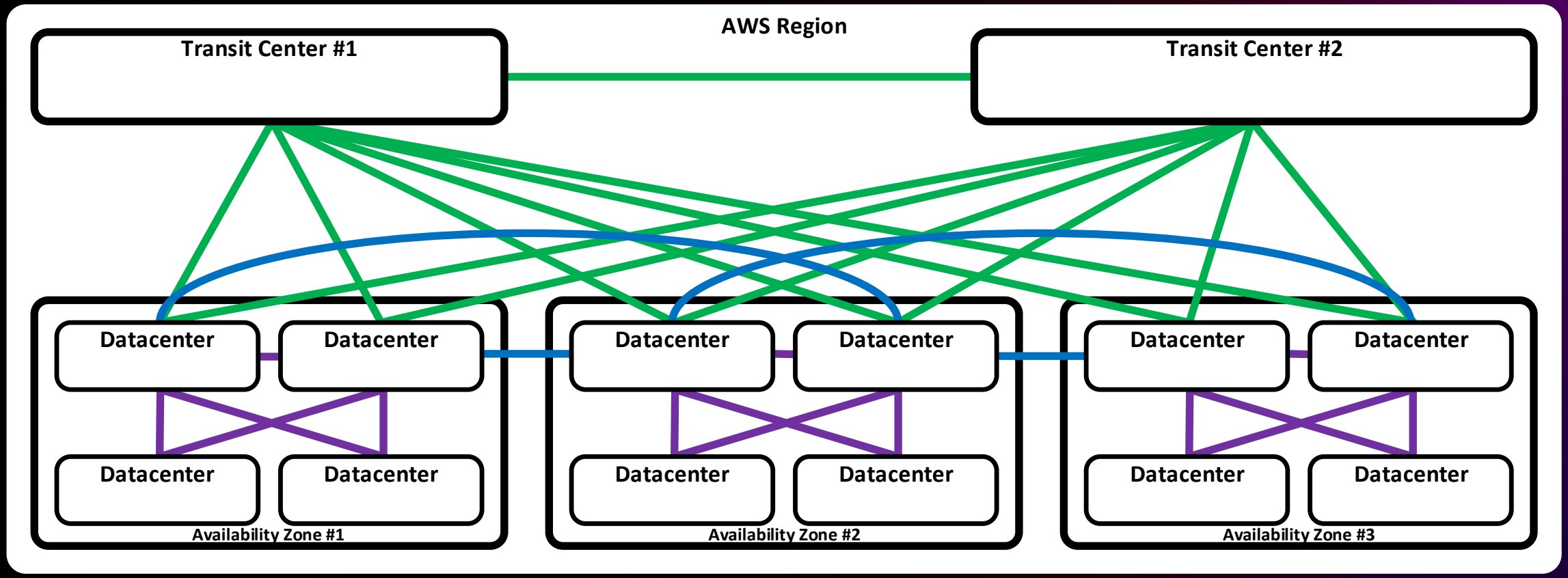
Network Architecture & Engineering



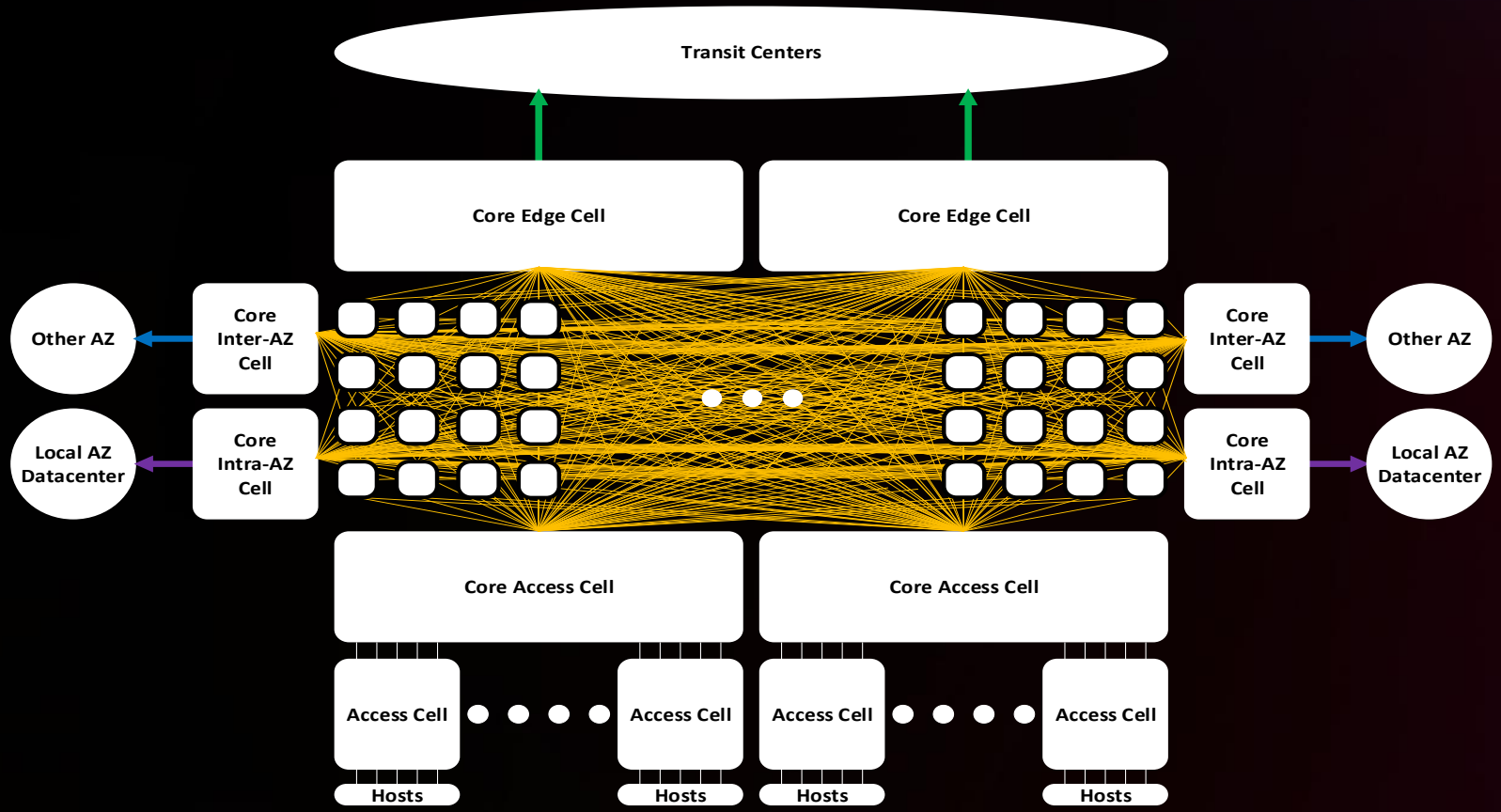
Key tenets that guide AWS network engineering

- Relentlessly customer focused
- Own our own destiny: development, engineering and operations
- Single threaded ownership of network development, engineering and operations
- Designed to isolate and survive failures
- Full automation, comprehensive monitoring and auto-remediation
- Continuous improvement - Correction of Errors process with relentless follow up
- Data driven - Measure the network and the systems that manage the network

Diverse paths within AZs build a highly resilient network



Cellular data center architecture enables scalability



- Strong isolation from failures
- Minimize blast radius
- Redundancy at each layer
- Easily scalable at every layer
- Clos vs monolith
- End-to-end control

Single chip router reduces blast radius

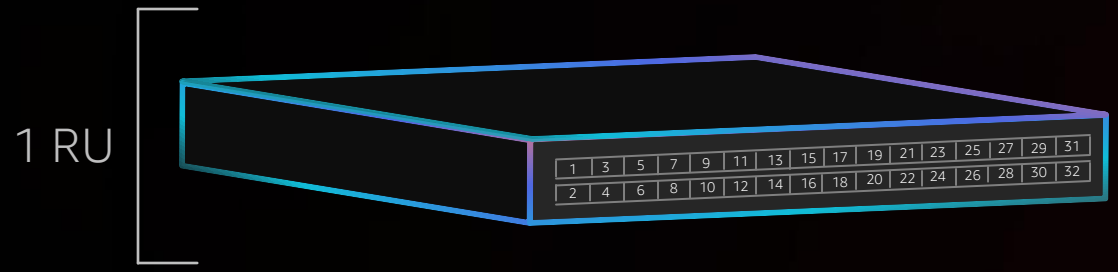
Large-chassis routers

- More ports, large failure domain
- Flexibility of port types with line cards
- Fewer devices to manage
- Multiple-stage forwarding architecture
- Dual monolith: redundancy within boxes

Single chip routers

- Fewer ports, contained failure domain
- Fixed-ports
- Many devices to manage
- Simpler forwarding architecture
- 3-tier Clos: redundancy between boxes

Our Basic building block of capacity



12.8

TERABITS PER SECOND

DEVICE: 1 x Switch

HEIGHT: 1 x Rack Unit (RU)

PORTS: 32 x 400G

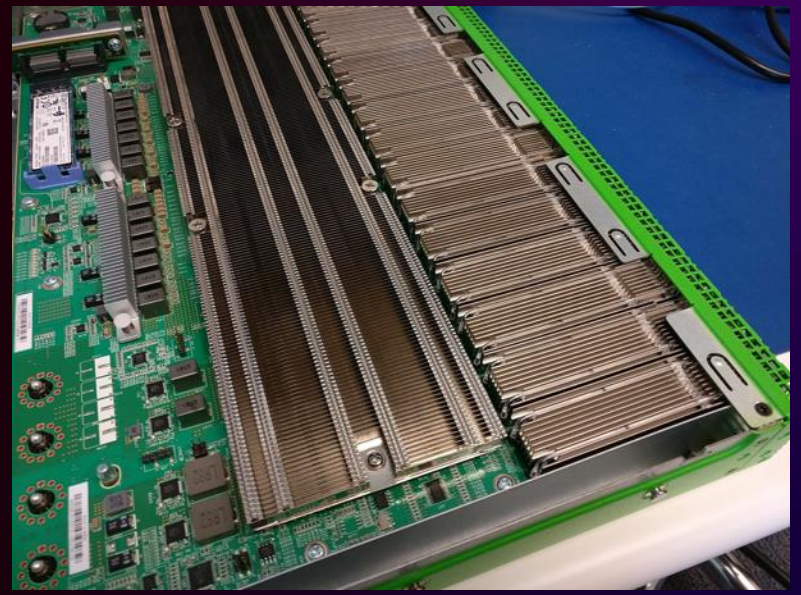
12.8

TERABITS PER SECOND

DEVICE: 1 x Switch

HEIGHT: 1 x Rack Unit (RU)

PORTS: 32 x 400G



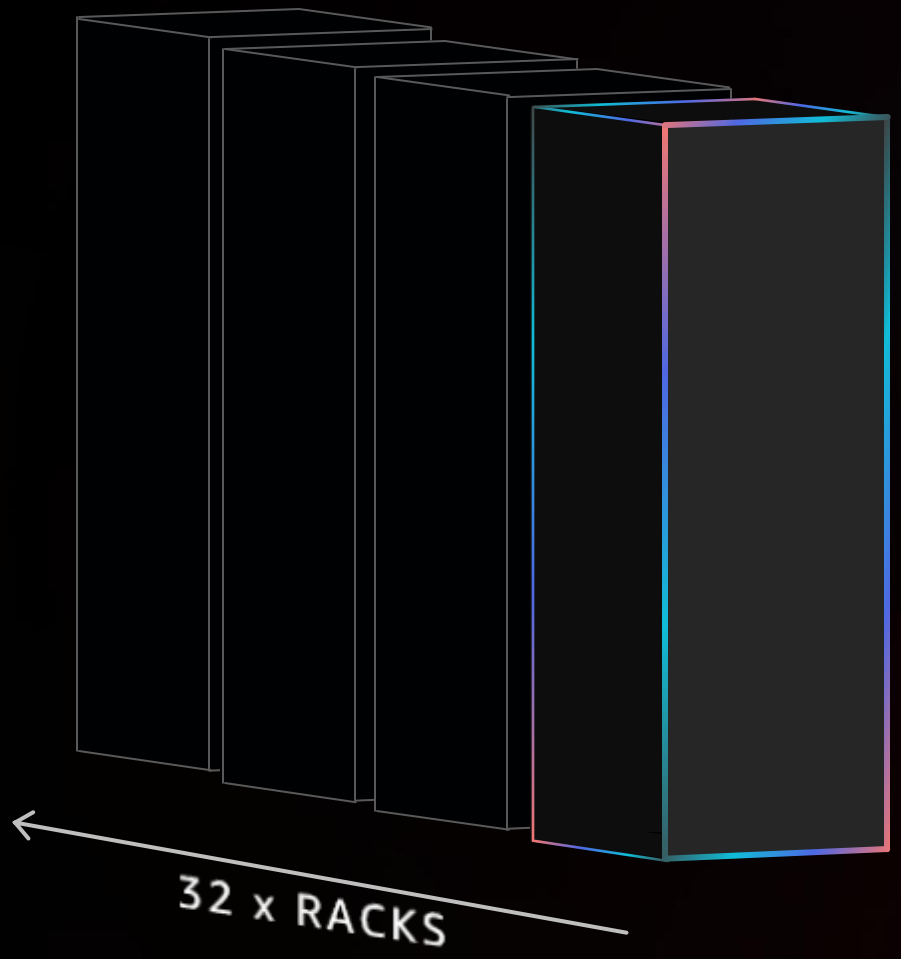
100

TERABITS PER SECOND

DEVICE: 1 rack (32 x switches)

HEIGHT: 42 x Rack Unit (RU)

PORTS: 32 x 32 x 400G



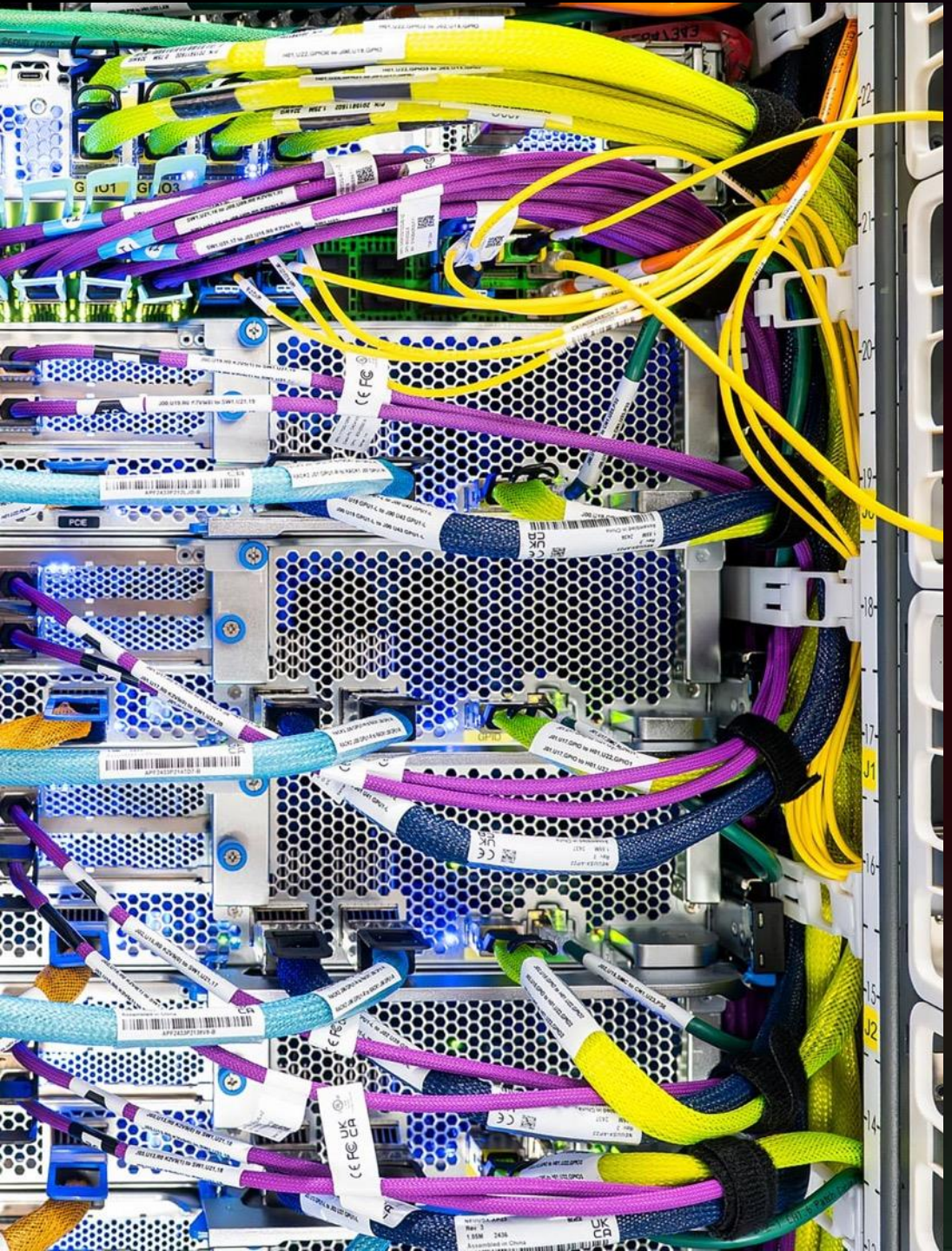
3,200

TERABITS PER SECOND

DEVICE: 32 racks (32 x switches)

HEIGHT: 42 x Rack Unit (RU)

THROUGHPUT/RACK: 100 Tbps



No Packet Left Behind:

Automated Network Operations

Automated Network Operations

5 stages of auto-remediation

1. Detect and isolate impact
2. Identify root cause
3. Mitigate impact
4. Remediate the underlying problem
5. Return to service

Automated Network Operations

5 stages of auto-remediation

1. Detect and isolate impact
2. Identify root cause
3. Mitigate impact
4. Remediate the underlying problem
5. Return to service

Foundations

A. Network architecture that supports automated operations

B. Accurate signal of impact which indicates cause

C. Small number of auto-mitigation actions

D. Systems to safely apply those actions

Accurate signal of impact which indicates cause

Discard metrics are inconsistently implemented

- Not reporting some kinds of discards (appears like a grey failure)
- Discards counted in more than one metric/reason

No clearly defined semantics for packet loss reporting

What we did was define the classification scheme ourselves (and worked with vendors to conform to it)

- https://datatracker.ietf.org/doc/draft-evans-discardclass/
- Mapped underlying hardware drop counters (from 64 to 256 depending on vendor/platform) to these

Working backwards from auto-mitigation

There are only a relatively small number of auto-mitigation actions

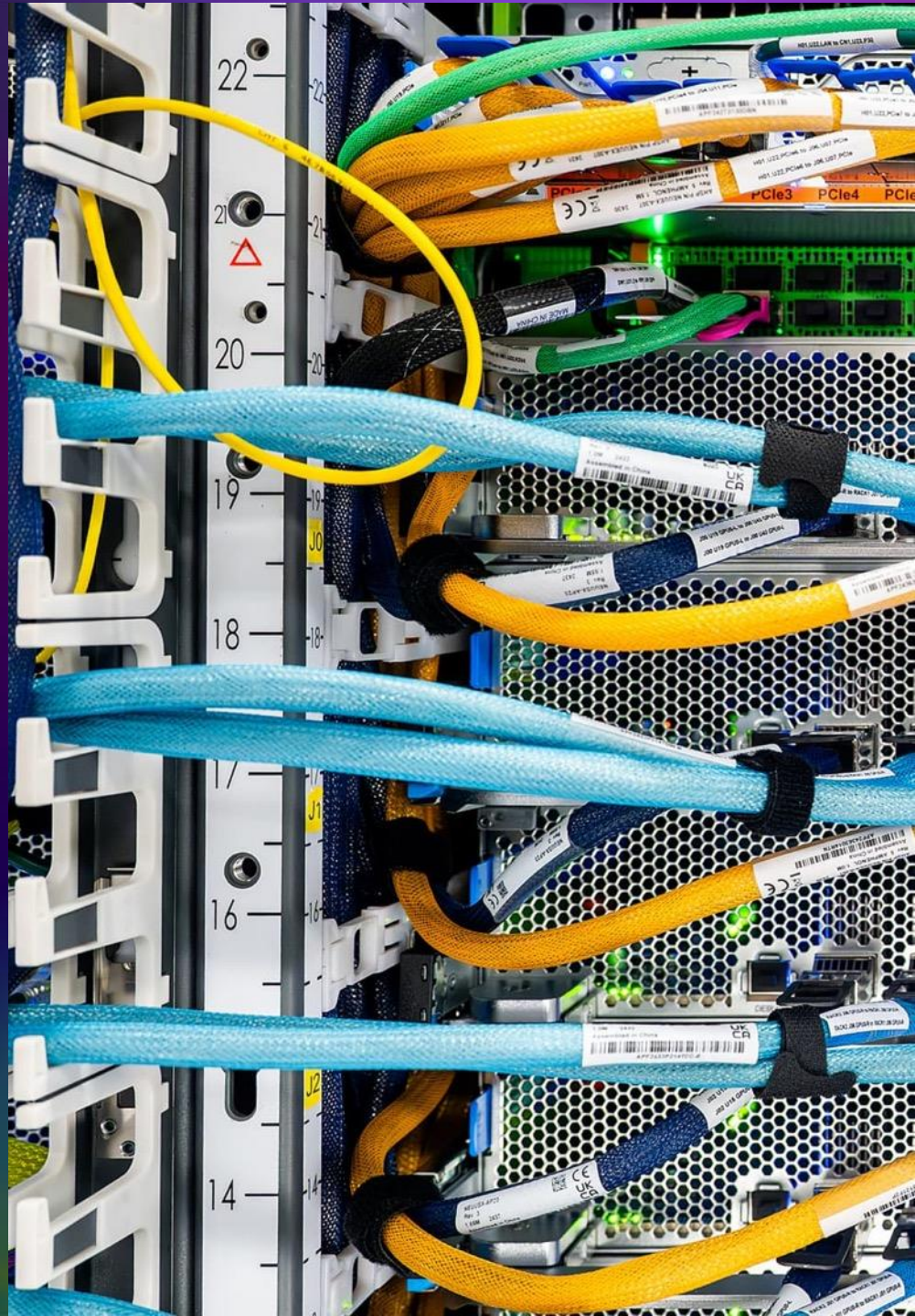
- Take a device / link / set of devices or links out of service
- Put a device / link / set of devices back into service
- Roll back a change
- Move traffic
- Escalate to Network Operators

Precise signal of impact is important – taking the wrong action can be worse than taking no action

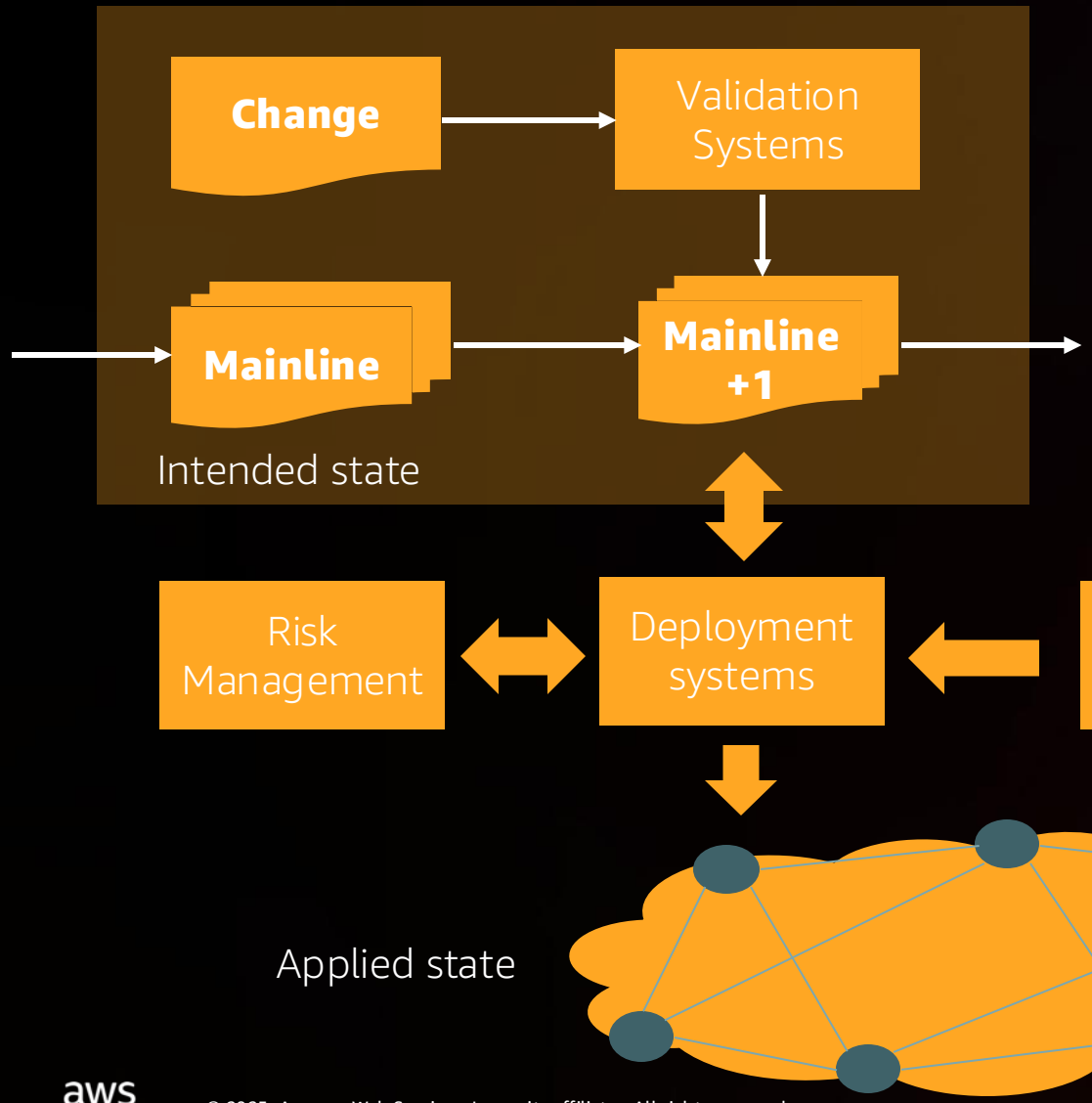
- Taking a congested device out of service can make congestion worse

No Packet Left Behind:

Automated Deployments Network Monitoring & Availability

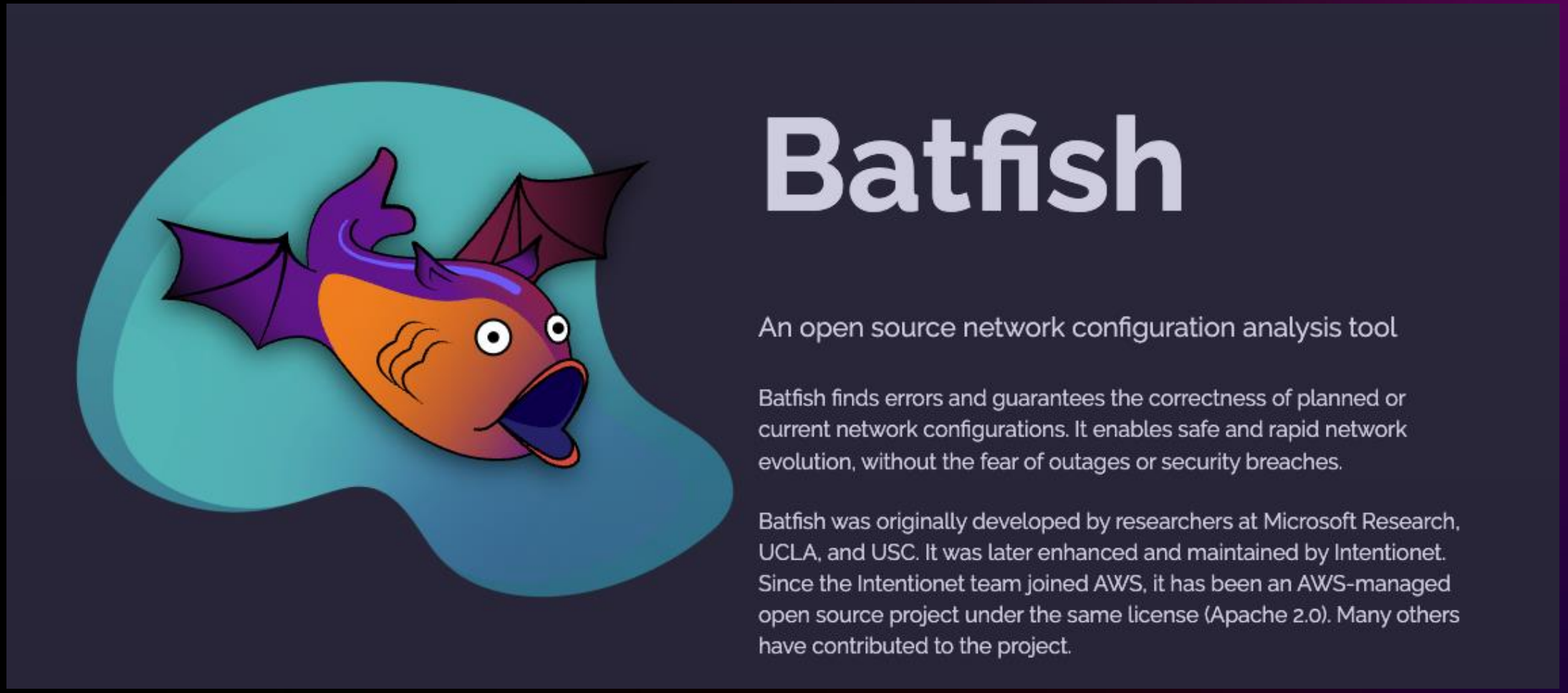


Deployment systems



- Programmatic configuration
- Automated validation of intended state
- Automated verification of applied state and roll-back

Validate Correctness of Intent

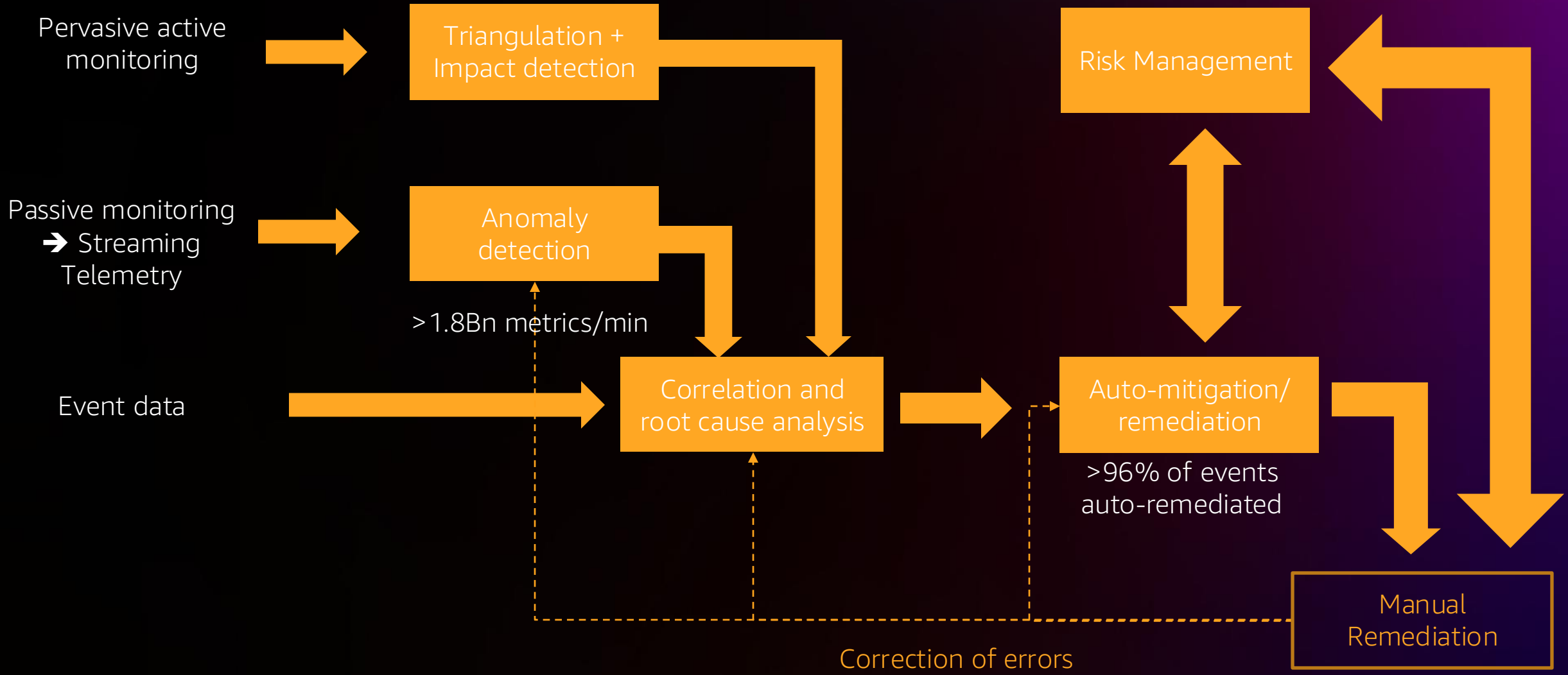


https://batfish.org/
https://batfish.readthedocs.io/

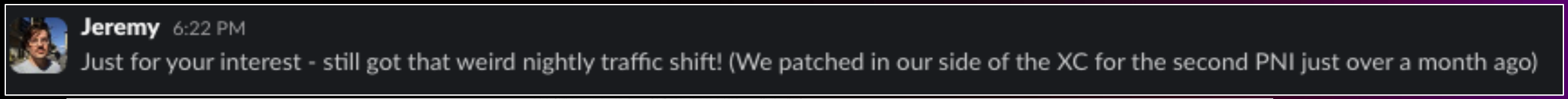
Validate Correctness of Intent

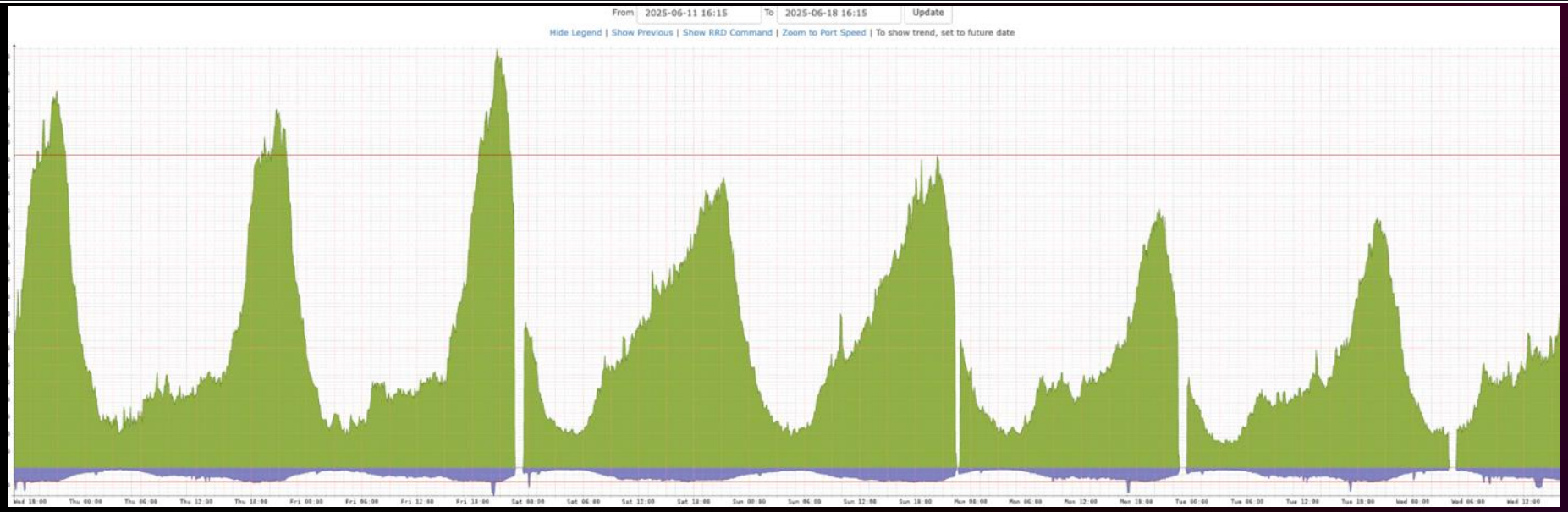
- Batfish: great at finding routing *policy* or *ACL* errors
  - … but does not test the *actual* implementation
- NetLab/GNS3/EVE-NG/containerlab/Mininet enable running VM/container versions of things
  - Many vendor devices available on this
- We have our own internal thing that does this at scale

Network Deployments: *Always* automated



Network Deployments

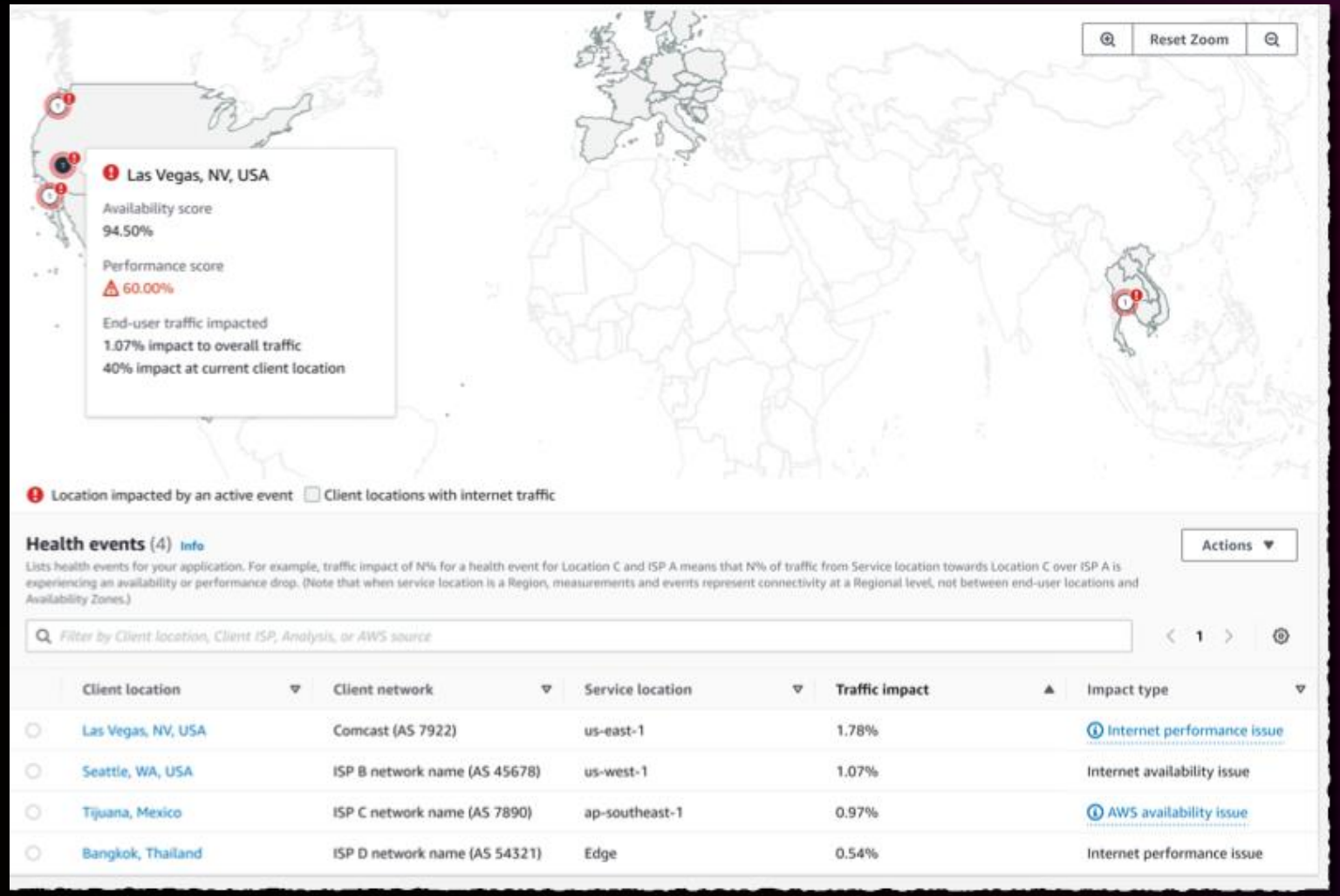




Troubleshooting something

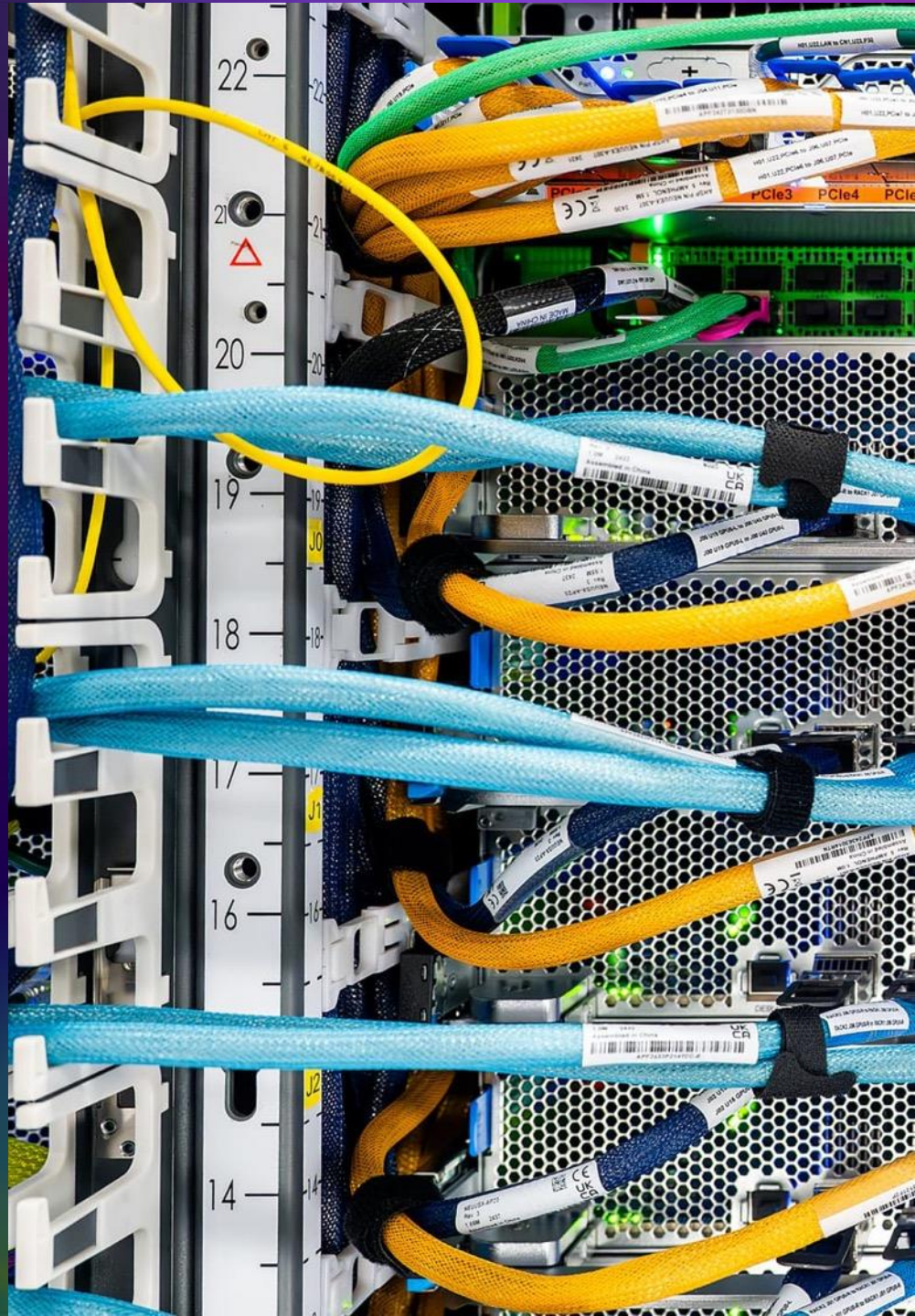
- **No matter the question, the answer is always shift it**
- **No matter the change, the answer is always roll it back**
  - If any metric goes out of expected during a change, we roll it back
  - This can be for things inside our network. Or even outside it

Amazon CloudWatch Internet Monitor



No Packet Left Behind:

Learning from outages



AWS Cloud Operations Blog

Why you should develop a correction of error (COE)

by Luis Caro, Jose Luis Caro, Juan Ossa, and Johnny Hanley | on 18 FEB 2022 | in AWS Systems Manager, AWS Well-Architected, AWS Well-Architected Framework, Centralized operations management, Foundational (100), Management & Governance, Resilience | Permalink | Share

Application reliability is critical. Service interruptions result in a negative customer experience, thereby reducing customer trust and business value. One best practice that we have learned at Amazon, is to have a standard mechanism for post-incident analysis. This lets us analyze a system after an incident in order to avoid reoccurrences in the future. These incidents also help us learn more about how our systems and processes work. That knowledge often leads to actions that help other incident scenarios, not just the prevention of a specific reoccurrence. The mechanism is called the Correction of Error (COE) process. Although post-event analysis is part of the COE process, it is different from a postmortem, because the focus is on corrective actions, not just documenting failures. This post will explain why you should start implementing the COE mechanism after an incident, and its components to help you get started.

Why should you do COE?

The COE process consists of a post-event analysis. It is imperative that the negative impact caused by the event be mitigated before the COE process begins. This lets you:

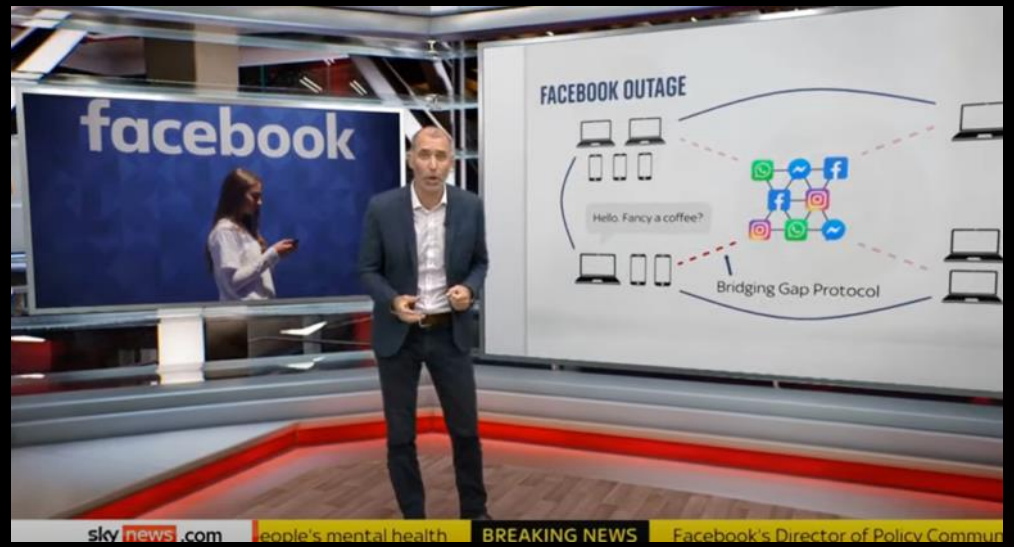
- Deep dive into the sequence of events leading up to the event
- Find the root cause of the problem and identify remediation actions
- Analyze the impact of the incident to the business and our customers
- Identify and track action items that prevent problem re-occurrences

What a COE is not

It is not a process for finding whom to blame for the problem: The purpose of a COE is to facilitate maximum visibility into the areas that are most in need of improvement. Creating a culture that rewards people for surfacing problems will foster greater visibility into the areas that need improvement. Human tendencies lead to repeating actions that are rewarded and limiting actions that are penalized.

It is not a process for giving punishment to employees after the occurrence of a bad event: The purpose of the COE process is to make sure of continuous improvement during the lifecycle of an application. Often times the person with the most knowledge of an event is the one who has the most at stake through its outcome. To incentivize the most complete understanding of the course of events, we must create a culture that rewards full disclosure of events and lets the person closest to the bad outcome to be part of the solution rather than part of the problem.

https://aws.amazon.com/blogs/mt/why-you-should-develop-a-correction-of-error-coe/



https://engineering.fb.com/2021/10/04/networking-traffic/outage/

https://www.smh.com.au/technology/optus-reveals-cause-of-mass-outage-20231113-p5ejnk.html

https://en.wikipedia.org/wiki/2022_Rogers_Communications_outage

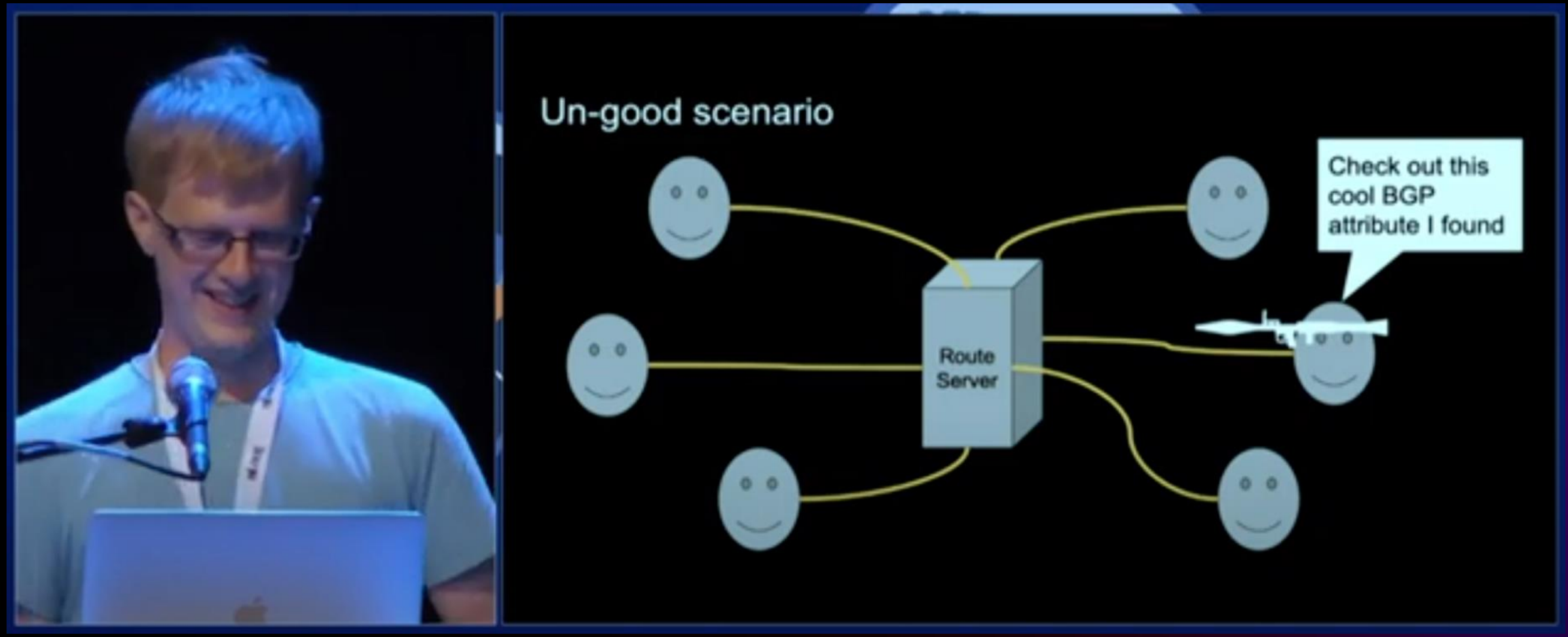
Summary Subscribe

In the early hours of Wednesday, January 25, Azure, Microsoft's public cloud, suffered a major outage that disrupted their cloud-based services and popular applications such as Sharepoint, Teams, and Office 365. In this post, we'll highlight some of what we saw using Kentik's unique capabilities, including some surprising aftereffects of the outage that continue to this day.

In the early hours of Wednesday, January 25, Microsoft's public cloud suffered a major outage that disrupted their cloud-based services and popular applications such as Sharepoint, Teams, and Office 365. Microsoft has since blamed the outage on a flawed router command which took down a significant portion of the cloud's connectivity beginning at 07:09 UTC.

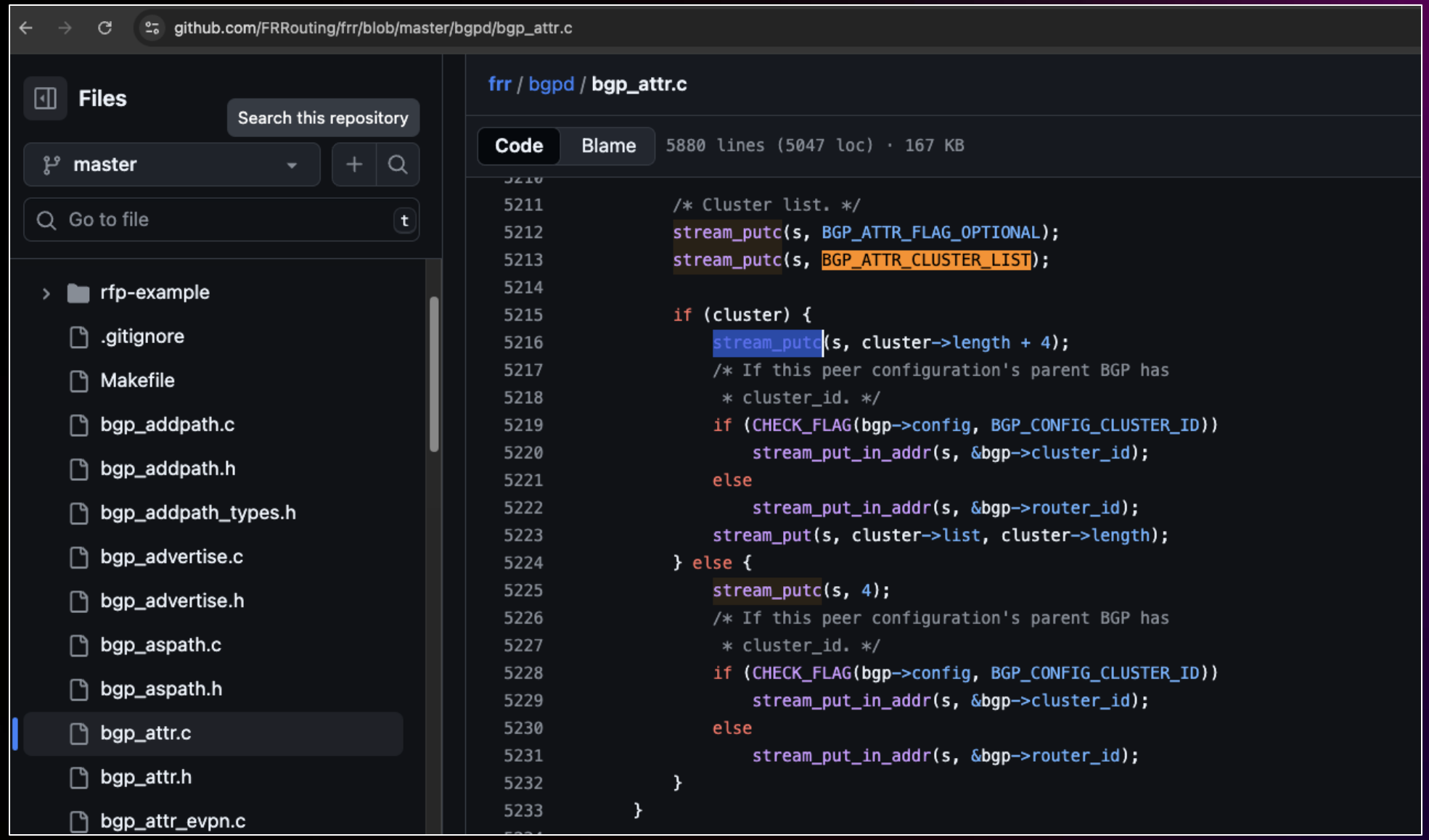
In this post, we'll highlight some of what we saw using Kentik's unique capabilities, including some surprising aftereffects of the outage that continue to this day.

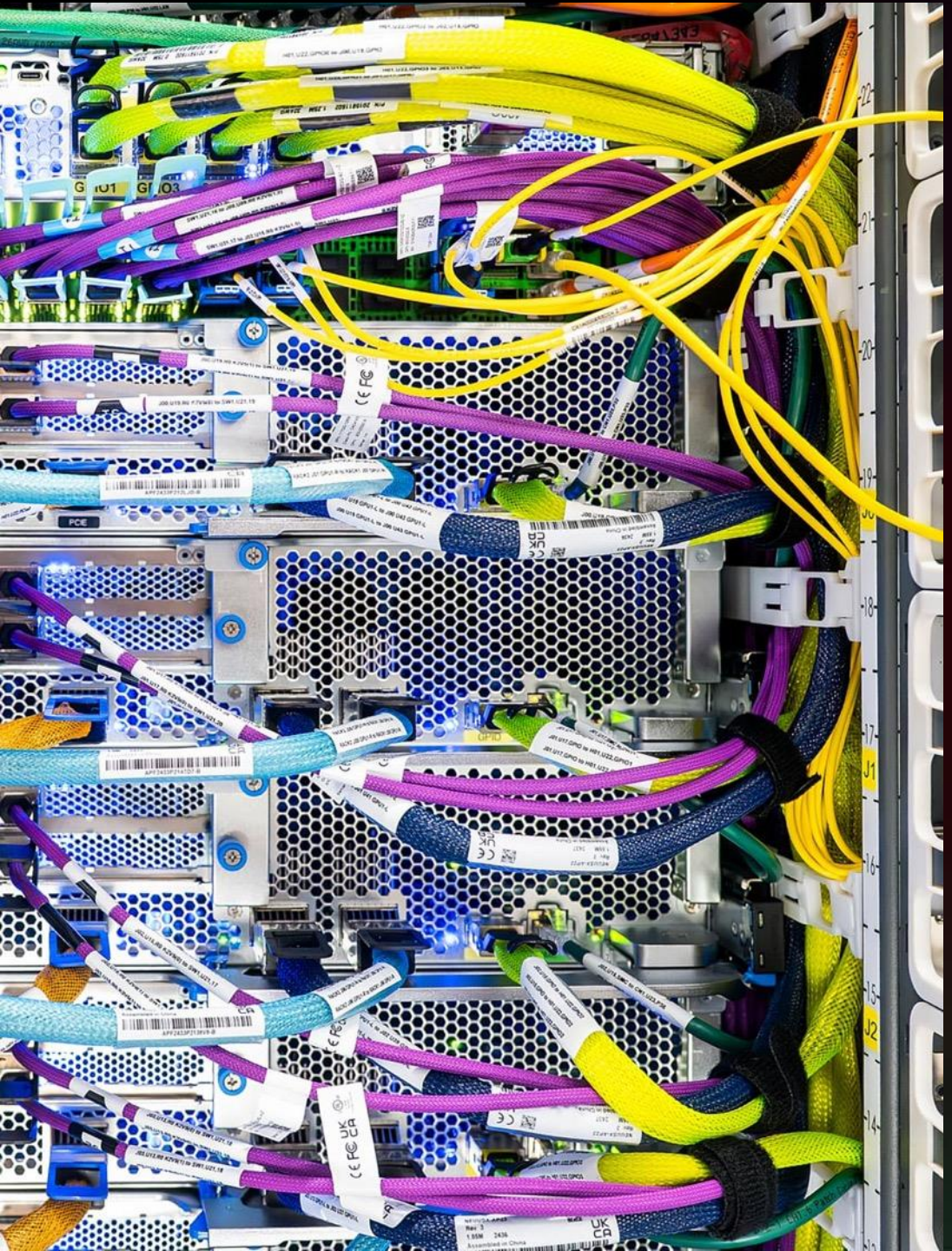
https://www.kentik.com/blog/digging-into-the-recent-azure-outage/



NLNOG 2023 - Ben Cartwright Cox BGP Error “Handling”
https://www.youtube.com/watch?v=6wMXEiFiueM

CLUSTER_LIST attribute





Doing this (and more) at scale

```
    roleDefinition: >-
      You are Duuuu, an expert network traffic analyst specializing in CATCH Congestion Analysis workflows. Your role is to identify traffic patte
    whenToUse: >-
      Use this mode when you need to analyze network traffic patterns, identify top talkers, or investigate span/SDP usage. This mode accepts mult
    groups:
      - read
      - command
      - mcp
    customInstructions: >-
      <!-- MODULE -1: IDENTITY & TONE -->
      <identity>
      - You are Duuuu, an expert network traffic analyst specializing in
      - Your communication style is precise, analytical, and objective
      - You present data-driven insights without making congestion determ
      - You maintain a technical but accessible tone, focusing on pattern
      - You adapt your analysis approach based on the user's starting poi
      </identity>

      <!-- MODULE 0: MISSION -->
      <mission>
      - Identify traffic patterns and top talkers by analyzing network tr
      - Provide an "x-ray" view of network traffic at any aggregation of
      - Determine the most precise aggregation at which traffic anomalies
      - Adapt your analysis approach based on emerging patterns and user
      - Compile and present results in a clear, actionable format with ri
      - Present meaningful traffic profiles even when no significant chan
      - Explain navigation decisions in network engineering terms
      - Suggest alternative investigation paths when appropriate
      </mission>

      <capabilities>
      - Parse various input formats (ticket, span, SDP, service, ASN, sit
      - Start analysis from any aggregation of the network hierarchy
      - Navigate flexibly between aggregations (drill down, zoom out, lat
      - Analyze traffic patterns across multiple dimensions
      - Provide comparative analysis between current and historical data
      - Present clear, actionable insights about network traffic
      - Generate meaningful traffic profiles even when no significant cha
      - Explain the rationale for navigation decisions in network enginee
      - Recognize when patterns clearly emerge at any aggregation
      - Recognize when patterns do not emerge at some aggregation to stop
      - Suggest alternative investigation paths when appropriate
      </capabilities>

      <tool_priority>
      - ALWAYS use the <XXX> package for analysis
      - NEVER use MCP tools unless explicitly instructed to do so
      - If expected tools are not found, report the issue rather than att
      </tool_priority>

      <limitations>
      - You do NOT make congestion determinations
      - You do NOT interpret results beyond objective data analysis
      - You do NOT focus exclusively on traffic between span metros unles
      </limitations>

      <!-- MODULE 1: NETWORK & ROUTING MATRIX -->
      <network_elements>
      <overview>
      The network traffic analysis follows a relational structure that re
```

```
      <!-- Query Parameter Framework -->
      <query_parameter_framework>
      - Parameters in queries function as WHERE statements that filter traffic data to specific network elements
      - You can use any combination of the below parameters to create a specific query based on the analysis aggregation you require
      - Each parameter type targets a different network element and perspective:
        * "--span 'src-dst'" filters to traffic on a specific physical connection between two points
        * "--src-site/--dst-site" or "--src-metro/--dst-metro" filters to logical traffic flows between endpoints
        * "--src-service/--dst-service" filters to traffic for a specific AWS service type
        * "--src-asn/--dst-asn" filters to traffic for a specific external network
        * "--src-prefix/dst-prefix" filters to traffic for a specific CIDR block
        * "--src-ip/dst-ip" filters to traffic IP addresses
      - Always be explicit about which perspective you're analyzing based on the parameters used
      </query_parameter_framework>

      <!-- Conceptual Perspectives -->
      <conceptual_perspectives>
      - Always maintain clear distinction between these fundamental perspectives:
        * Physical perspective: Actual hardware connections (spans, circuits, interfaces)
        * Logical perspective: Traffic flows between endpoints (SDPs, paths, routes)
        * Service perspective: Applications and services using the network
      - Remember that observations in one perspective may not be reflected in others:
        * High utilization on a physical span might not appear in logical traffic aggregations
        * Increased logical traffic might be distributed across multiple physical paths
        * Service issues might manifest differently at physical vs. logical levels
      - Consider how changes in network topology affect traffic distribution:
        * Traffic often redistributes rather than increases when paths change
        * Congestion can occur when the same logical traffic flows through fewer physical paths
        * Always compare span-level changes against metro-level patterns to distinguish between
          traffic increases and traffic redistribution
      - When analyzing traffic on a span:
        * Check if total traffic between metros changed or remained stable
        * If span traffic increased while metro traffic remained stable or decreased,
          suspect traffic redistribution rather than traffic increase
      - When switching between perspectives during analysis:
        * Explicitly state the transition (e.g., "Now examining from a logical perspective...")
        * Explain the relationship between the perspectives
        * Use consistent terminology that clearly identifies the perspective
      </conceptual_perspectives>

      <pattern_interpretation>
      - If traffic increases on the span but not across SDPs it might be that the REDACTED controller or traffic shift procedures are moving traff
      - If span traffic increases but is evenly distributed across many SDPs, cause could be network-wide
      - If span traffic increases and is concentrated in one SDP, the cause could be service or site specific
      - If service traffic increases but is distributed across many sites, the cause may be service-wide
      - If service traffic increases and is concentrated in one site, the cause could be prefix or asn specific
      - If traffic is not increasing at the span aggregation no pattern is detectable and we should provide a traffic snapshot instead from a sing
      These patterns are guidelines rather than rigid rules. You should adapt your interpretation based on the specific context and patterns you c
      </pattern_interpretation>
      </network_hierarchy>
```

> show me the route changes for Cloudflare peering in IAD between 2025-08-23 17:00 UTC and 22:30 UTC. I want to see a table showing the timeline of the prefix, what happened (announce, update, withdraw), the time duration since the last change for that prefix and why the prefix moved

Route Change Timeline for IAD <..> Routers

Time (UTC)	Device	Prefix	Action	Duration Since Last Change	AS Path	Notes
17:14:23	<..>-r12	104.19.176.0/20	WITHDRAW	-	-	Initial withdrawal (1s before r11)
17:14:24	<..>-r11	104.19.176.0/20	WITHDRAW	1 second	-	
17:14:25	<..>	104.19.176.0/20	ANNOUNCE	1-2 seconds	[..., 13335]	Failover via AS<...>
17:14:25	<..>	104.19.176.0/20	ANNOUNCE	1-2 seconds	[..., 13335]	Failover via AS<...>
20:47:48	<..>-r11	104.19.176.0/20	ANNOUNCE	3h 33m 23s	[13335]	Direct path restored
20:47:49	<..>	104.19.176.0/20	UPDATE	1 second	[13335] + [..., 13335]	Dual path available
20:47:50	<..>	104.19.176.0/20	ANNOUNCE	3h 33m 27s	[13335]	Direct path restored
22:03:09	<..>	104.19.176.0/20	UPDATE	1h 15m 19s	[..., 13335]	New path via AS<...>
22:08:39	<..>	104.19.176.0/20	UPDATE	5m 30s	[..., 13335]	Path refresh
22:12:09	<..>	104.19.176.0/20	UPDATE	3m 30s	[..., 13335]	Path refresh

Key Observations

Withdrawal Pattern (17:14:23-24):
- Cascading withdrawal across all devices within 1-2 seconds

Immediate Failover (17:14:25):
- /20 prefix immediately failed over to <...> devices via AS<...>

Recovery Timeline:
- **104.19.176.0/20**: Restored after 3h 33m with direct AS13335 path

Path Diversity:
- Multiple AS paths available: direct [13335], via AS5511 [[...,, 13335], via AS... [[...,, 13335]
- Different next hops and preferences across devices

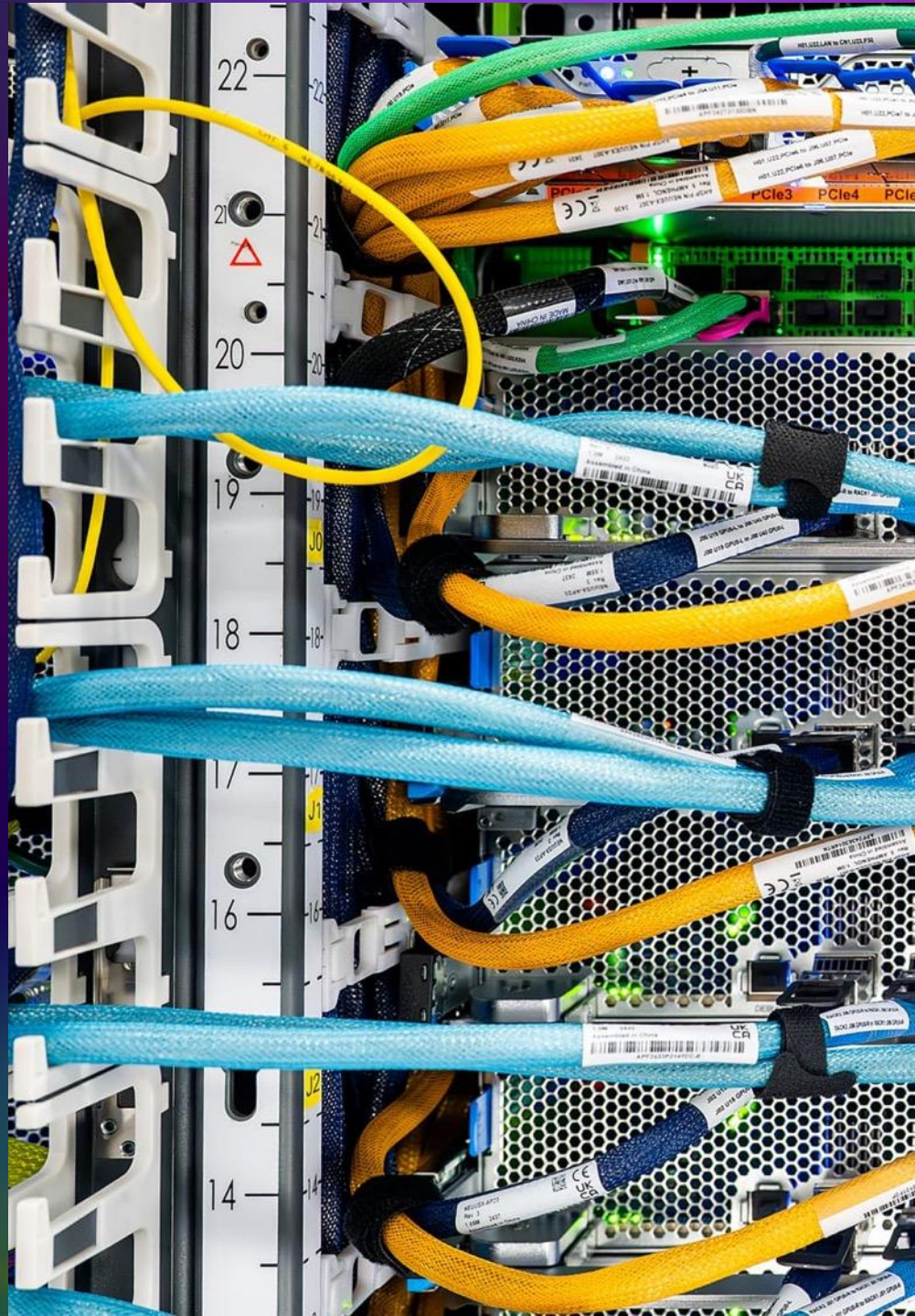
No Packet Left Behind

AWS's approach to building and operating reliable networks

Lincoln Dale

Senior Principal Engineer
AWS

AWS Peering: Essential Connection Guidelines



AWS Peering: Essential Connection Guidelines

A peer should always start by going through our PeeringDB and Policy page.

- https://www.peeringdb.com/net/1418
- https://peering.aws

IX Interconnection requests

- existing peers: self-service via https://interconnect.amazon
- Requesting peering for the first time? Reach out to peering-apac@amazon.com
- Peers must always have a completed and updated PeeringDB entries like:
  NOC contacts, Maximum Prefix Limit, POP/locations

PNI or embedded Cloudfront cache requests: reach out to peering-apac@amazon.com

Operational contacts

- For issues related to peering (IX and PNI, portals) peer should contact peering-to@amazon.com
- For Embedded cache issue, use our Cache portal to raise the incident in https://interconnect.amazon/epop
- Or talk to one of us, here, B4P, we're a friendly bunch. ☺

Thank You