AWS re:Invent

DOP208-R

Amazon's approach to failing successfully

Becky Weiss
Senior Principal Engineer
Amazon Web Services

AWS re:Invent

aws

This is a talk about failing … successfully

Agenda

Never waste a failure: The AWS approach to post-mortems

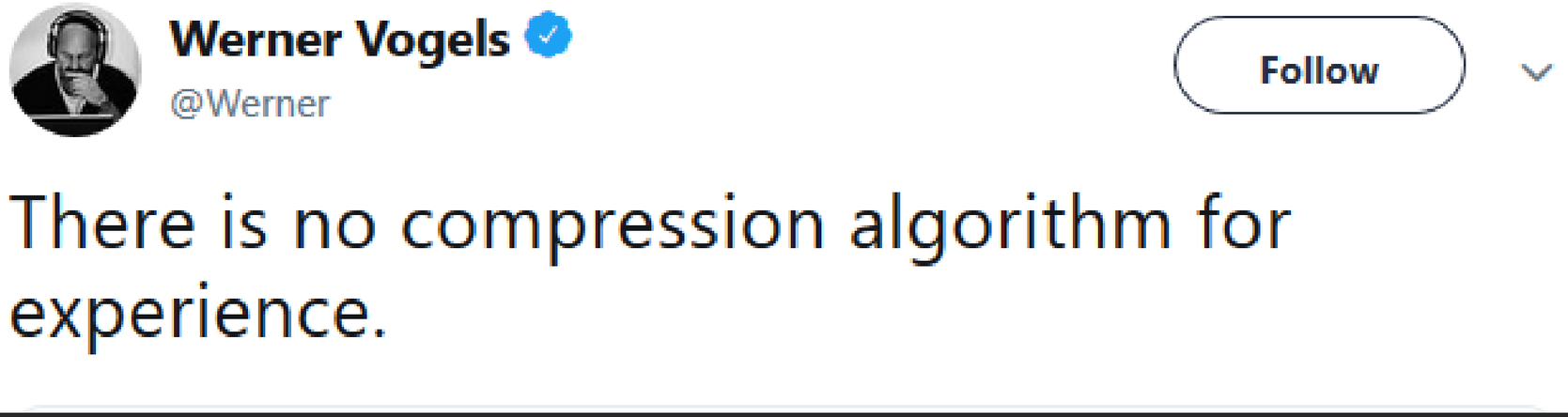
Seeing your failures before your customers do

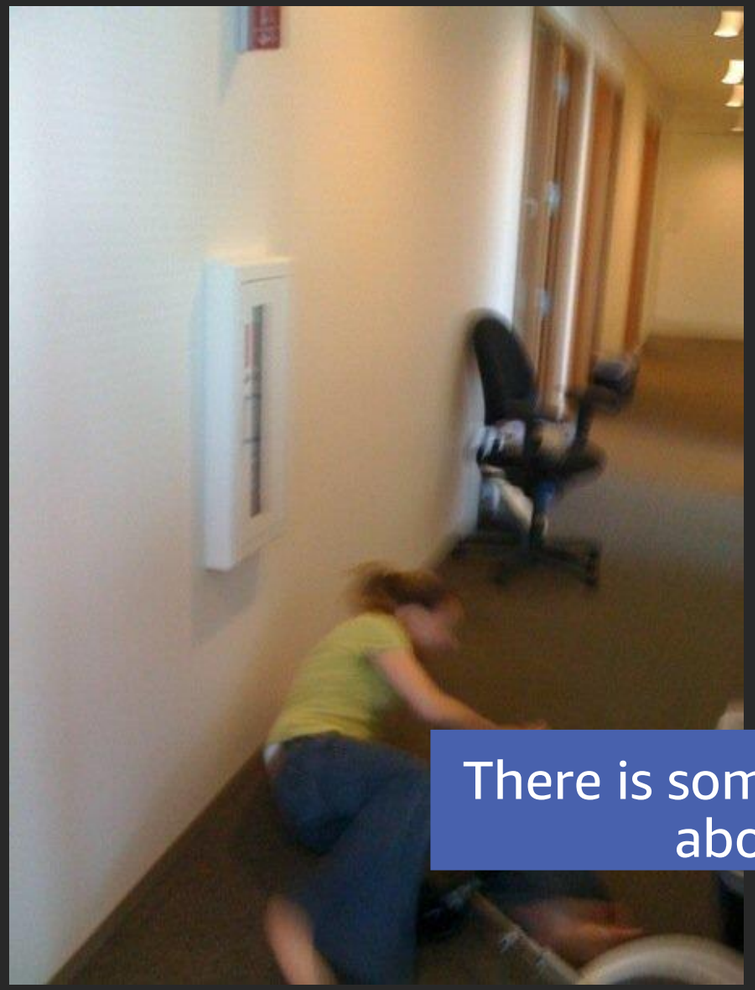
How AWS can help you fail successfully

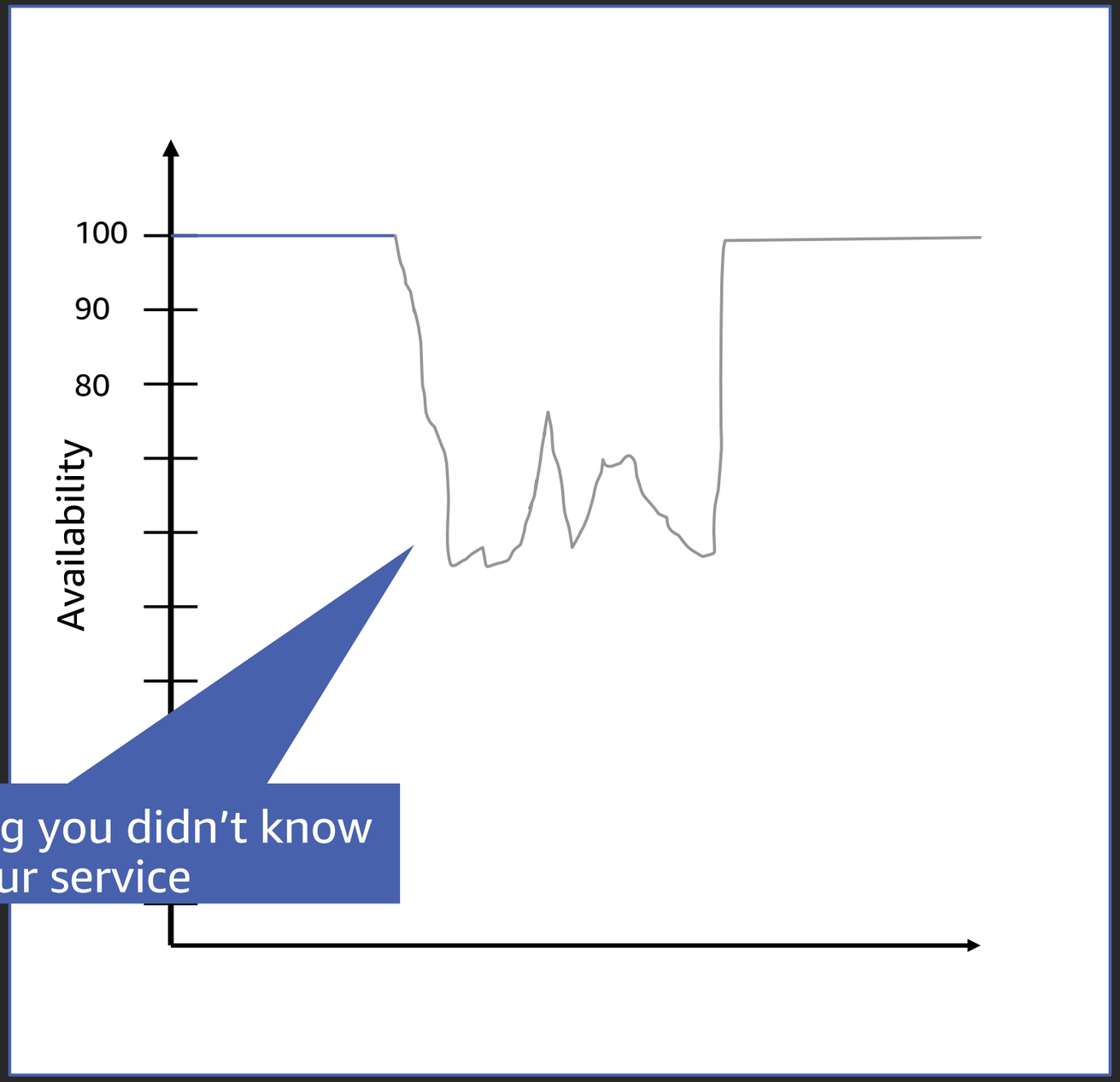
Never waste a failure: Post-mortems at AWS

AWS re:Invent

aws


Werner Vogels
@Werner
Follow
There is no compression algorithm for experience.




100
90
80
Availability
There is something you didn’t know about your service

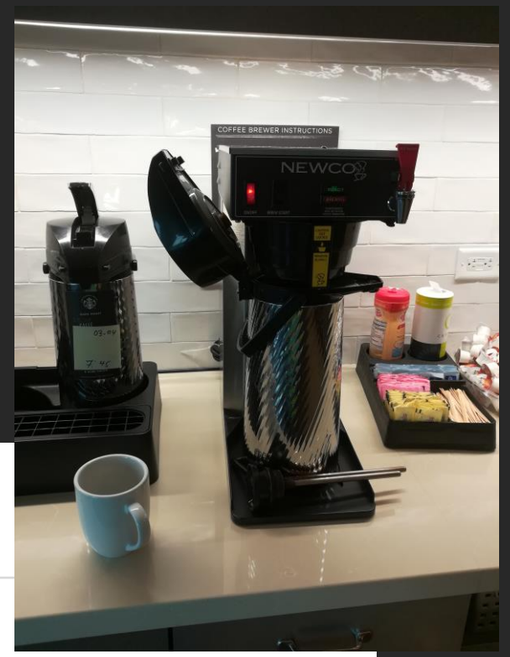
COE: Correction of error

Structured analysis of customer-impacting events

Reflection of Amazon’s peculiar culture

Goes well beyond “How do we prevent this from happening again”

We take these very seriously



[89458] Decaf coffee brewed in a non-decaf pot

Summary

At 8:45 am Pacific on 4/24/2018 on the 12th floor of Alexandria, an operator inadvertently brewed decaf coffee into the "medium roast, non-decaf" coffee pot. The operator was attempting to brew decaf coffee, but chose the wrong pot to brew it into.

Metrics / Graphs

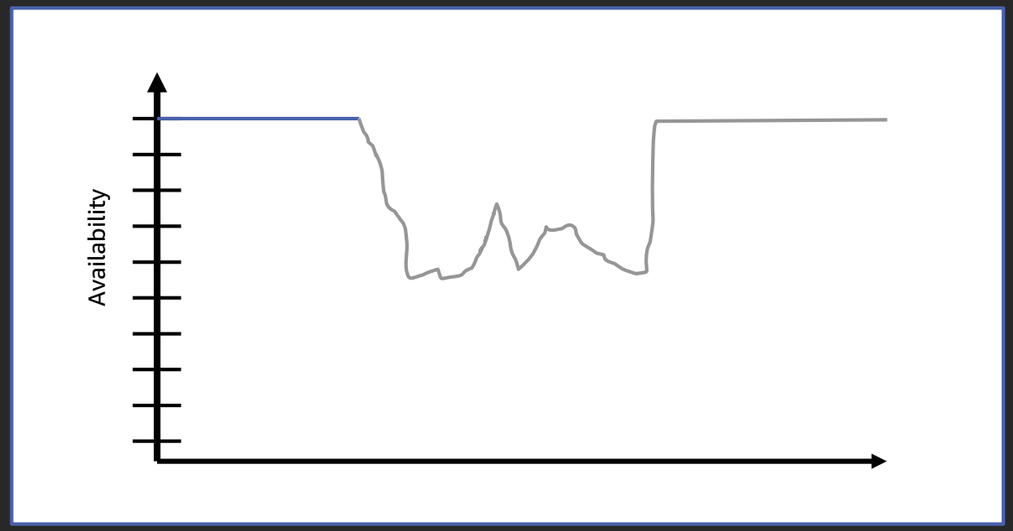
- Number of pots of coffee brewed incorrectly during incident: 1
- Number of customers who unknowingly took a cup of decaf coffee: 2 (estimated)

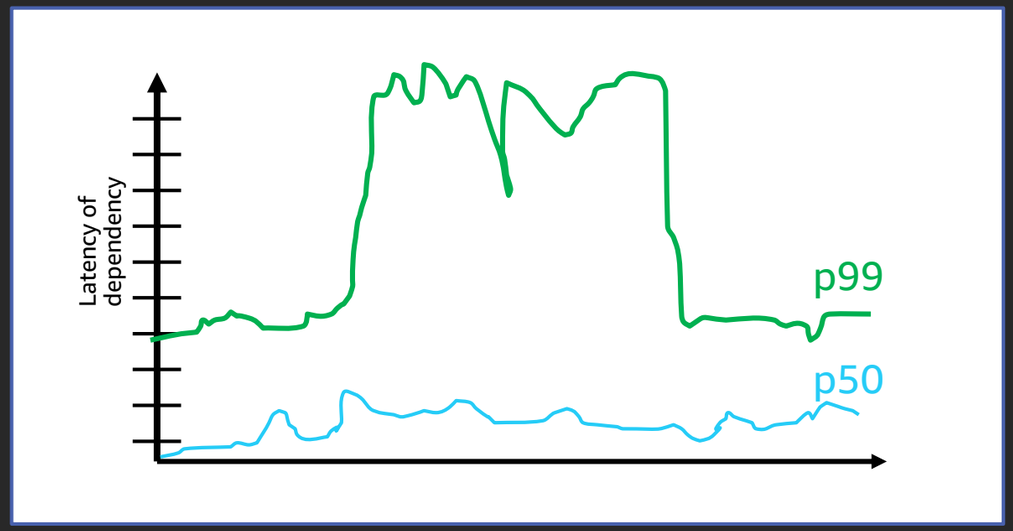
Customer Impact

While there are no metrics available to prove this, we know that there were at least two people who took coffee from the pot. One engineer had a half cup of coffee at his desk. That engineer did not move the pot from the brewing station into its holding spot, so that suggests that at least one other person had coffee. The one confirmed engineer chose to drink the cup of coffee anyway.

COEs start with the customer and work backward

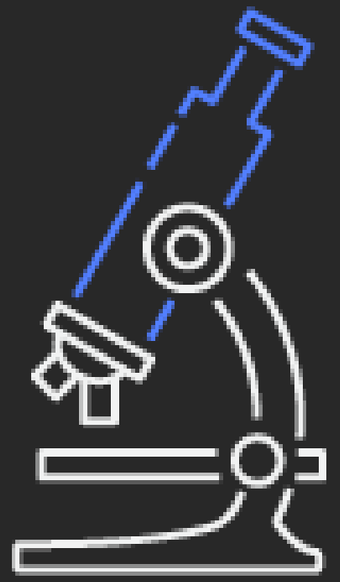
- Summary
  - Narrative description of what happened
- Metrics and graphs
  - Primary impact and supporting graphs
  - If they don’t exist, that’s something to fix
- Customer impact
  - How many customers affected
  - What was the impaired experience





Areas of focus

- Root cause: Why? (x5)
- Blast radius: How widespread was the impact?
- Duration: For how long?
- What can others learn?



Toyota's Five-Whys approach to root cause

The vehicle will not start. ← the problem

Why? - The battery is dead. (First why)

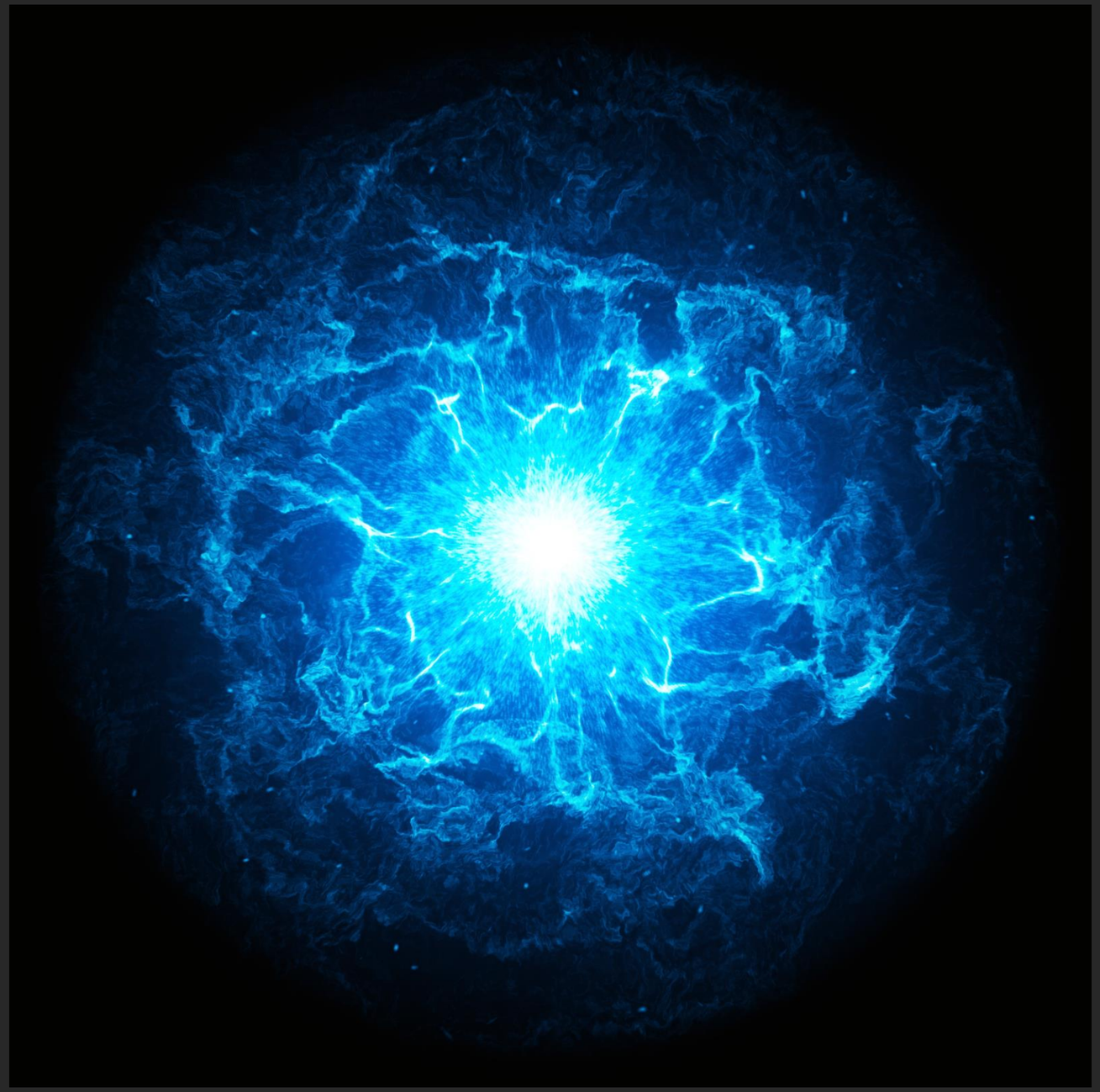
Why? - The alternator is not functioning. (Second why)

Why? - The alternator belt has broken. (Third why)

Why? - The alternator belt was well beyond its useful service life and not replaced. (Fourth why)

Why? - The vehicle was not maintained according to the recommended service schedule. (Fifth why, a root cause)

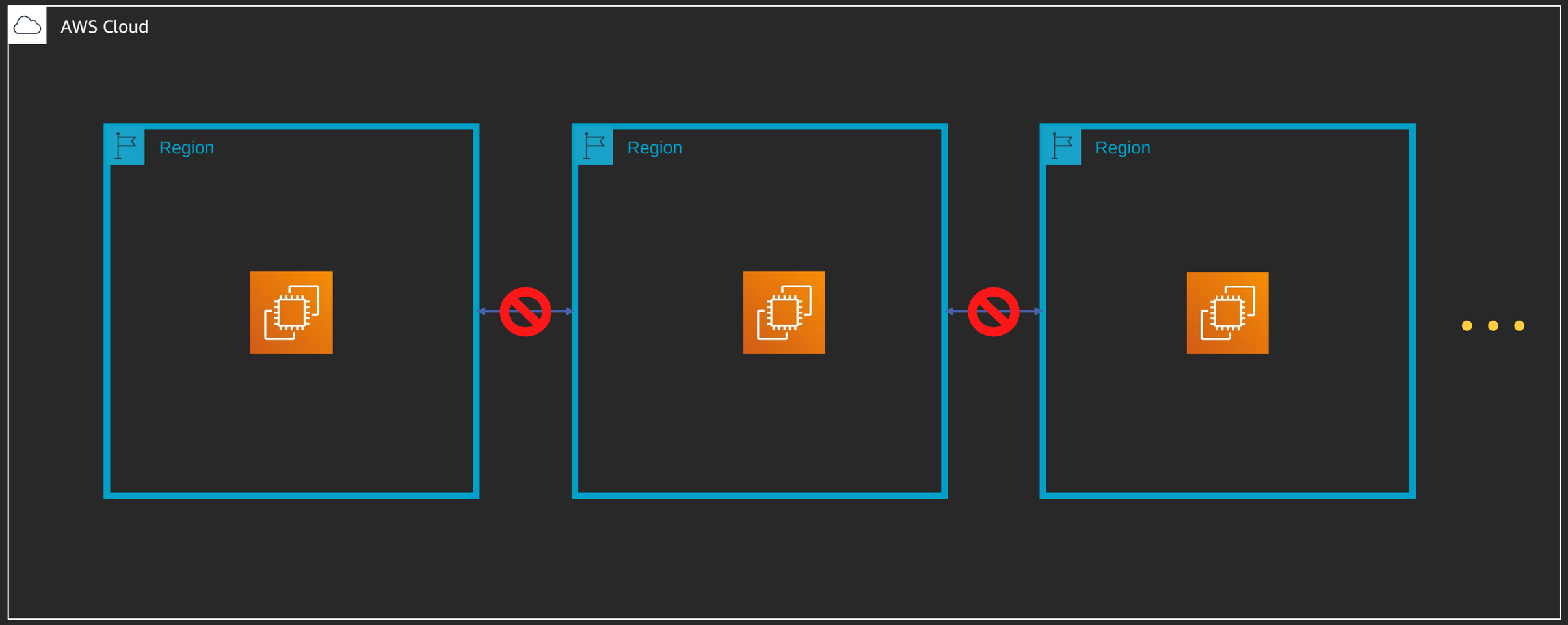
Blast radius



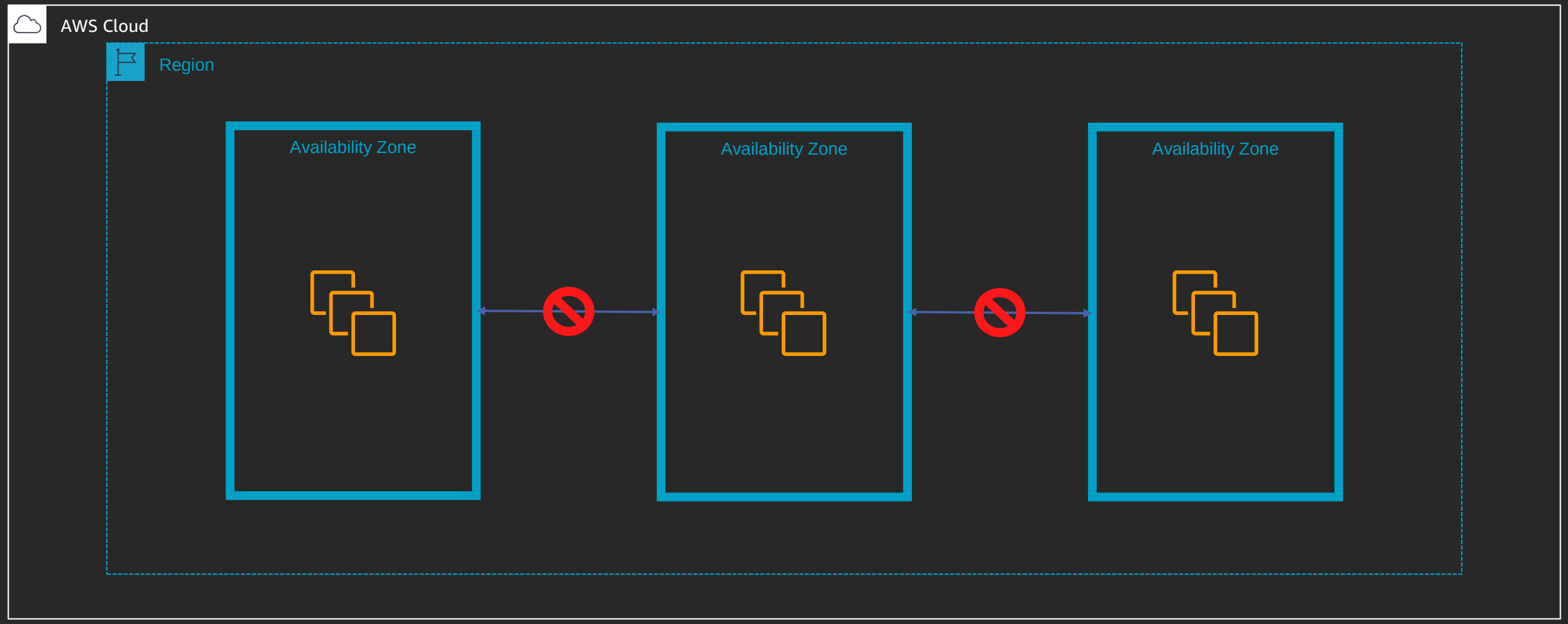
Blast radius

“Consider how blast radius could be reduced. As a thought exercise, how could you cut the blast radius for a similar event in half?”

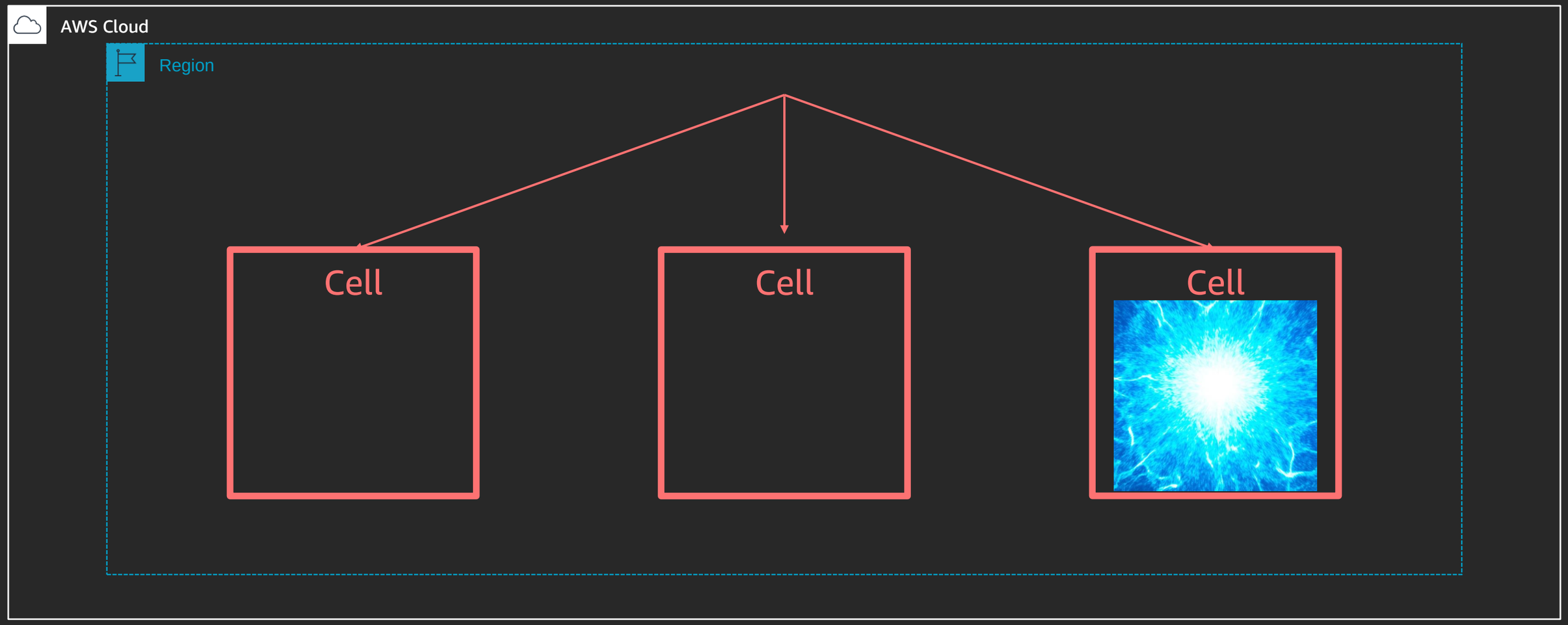
Blast radius containment as a core design tenet



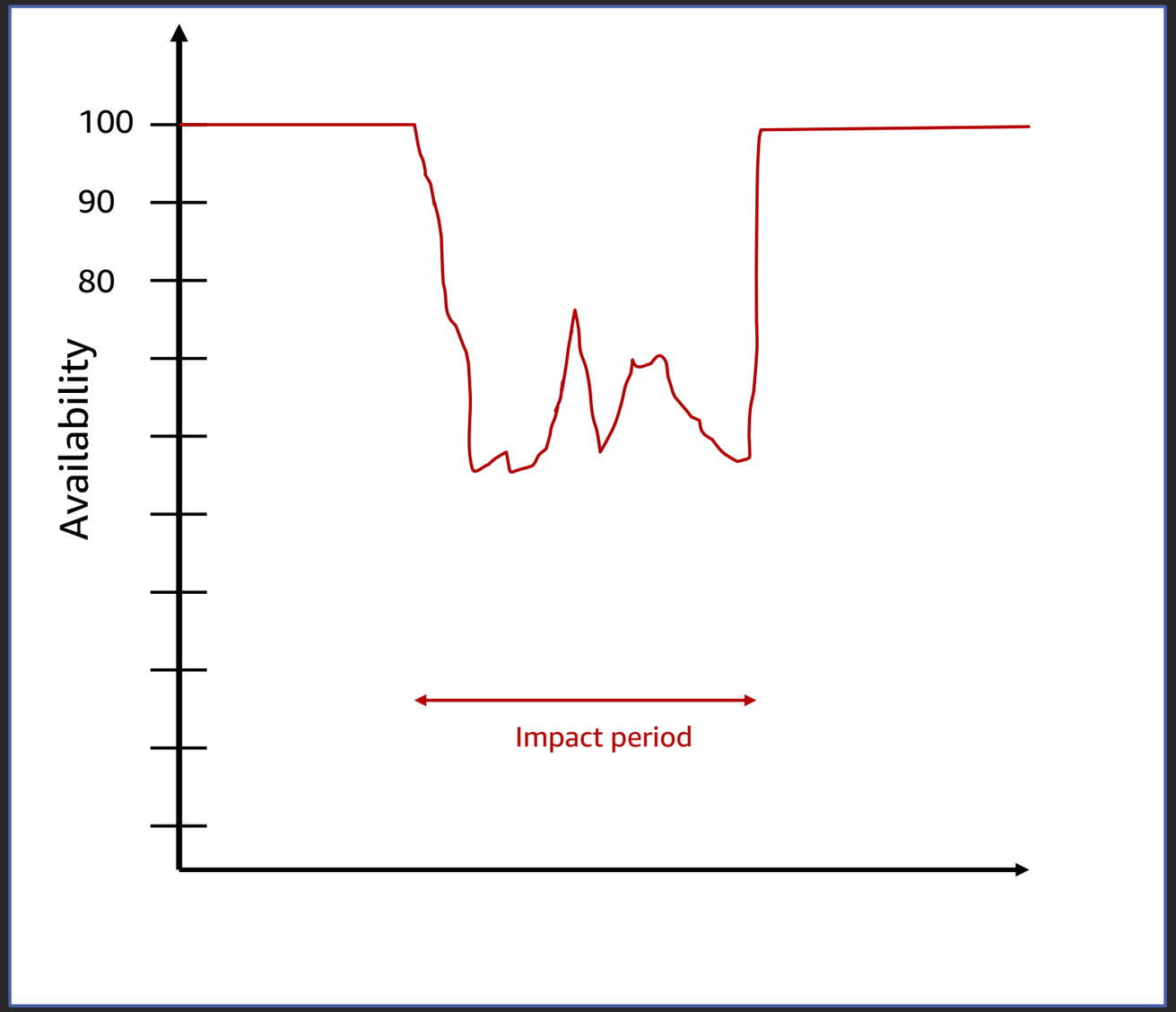
Blast radius containment as a core design tenet



Blast radius containment as a core design tenet

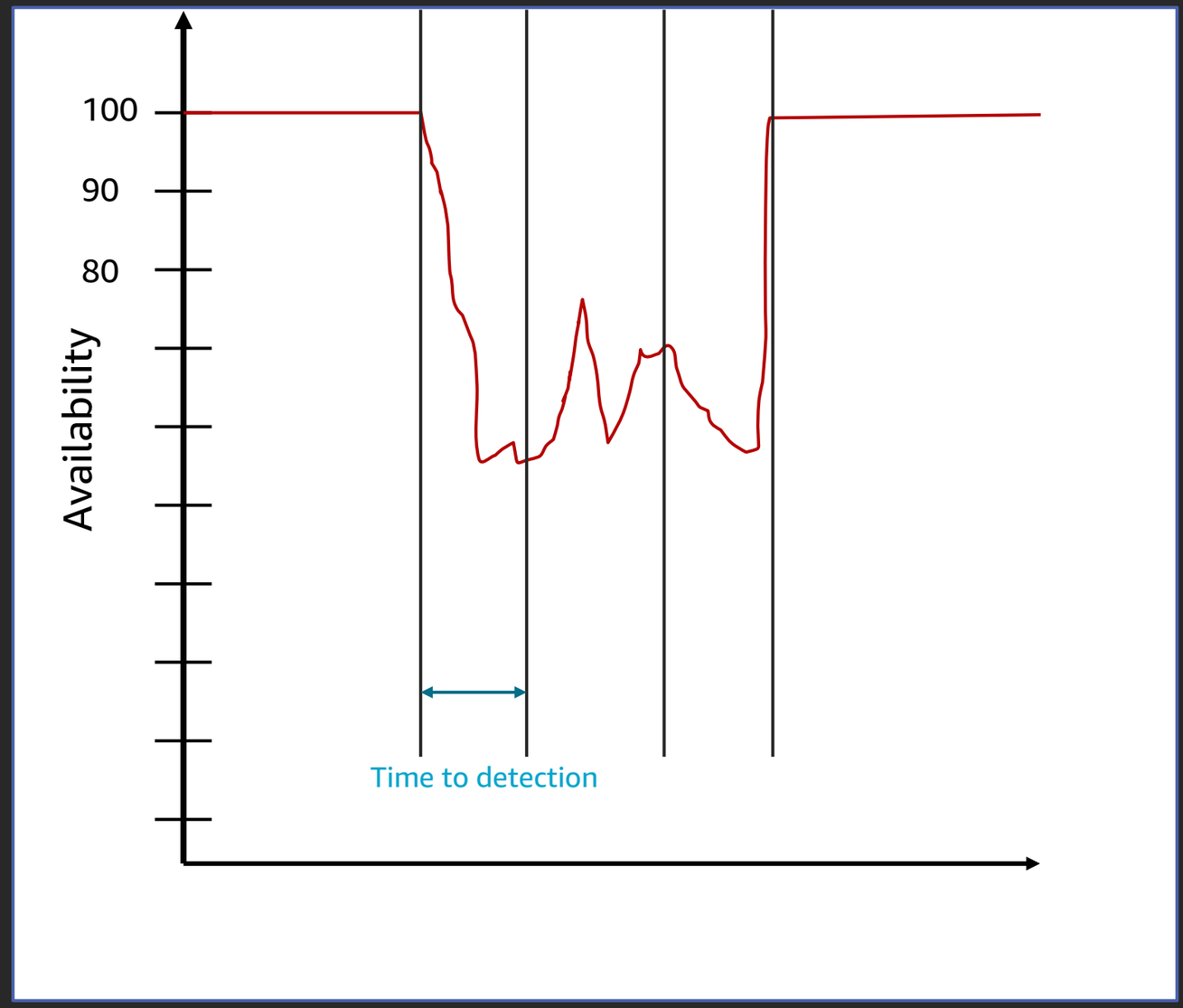


Controlling event duration



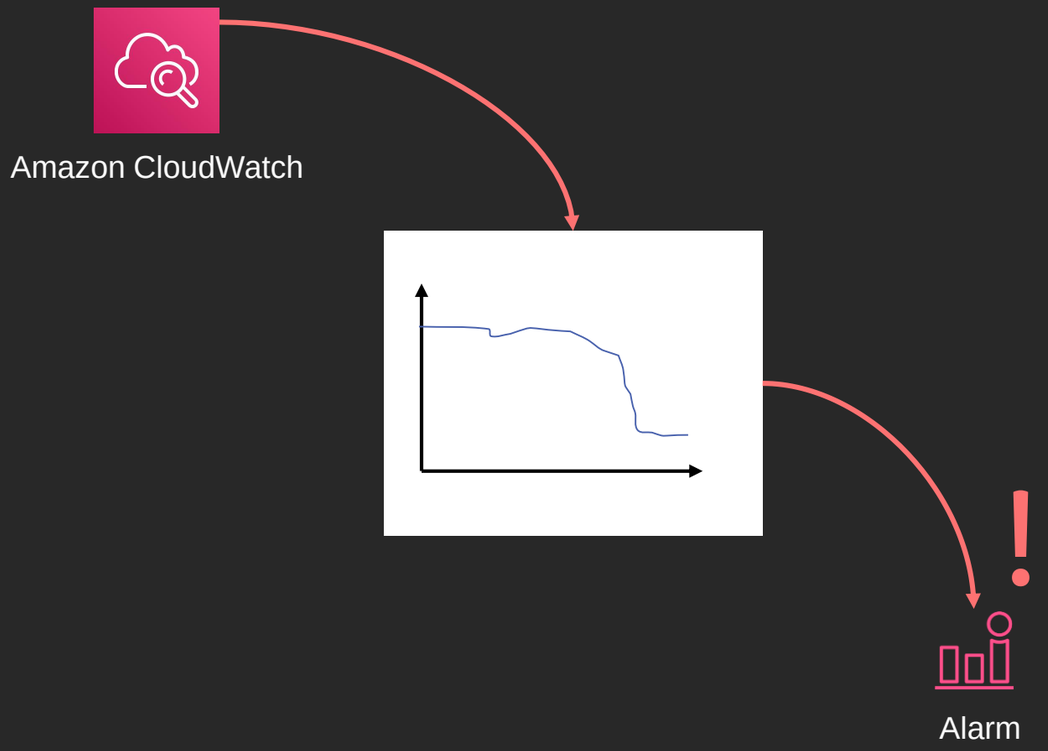
Controlling duration: Improving incident response

- “How was the event detected?”
- “How could time to detection be improved? As a thought experiment, how would you have cut the time in half?”



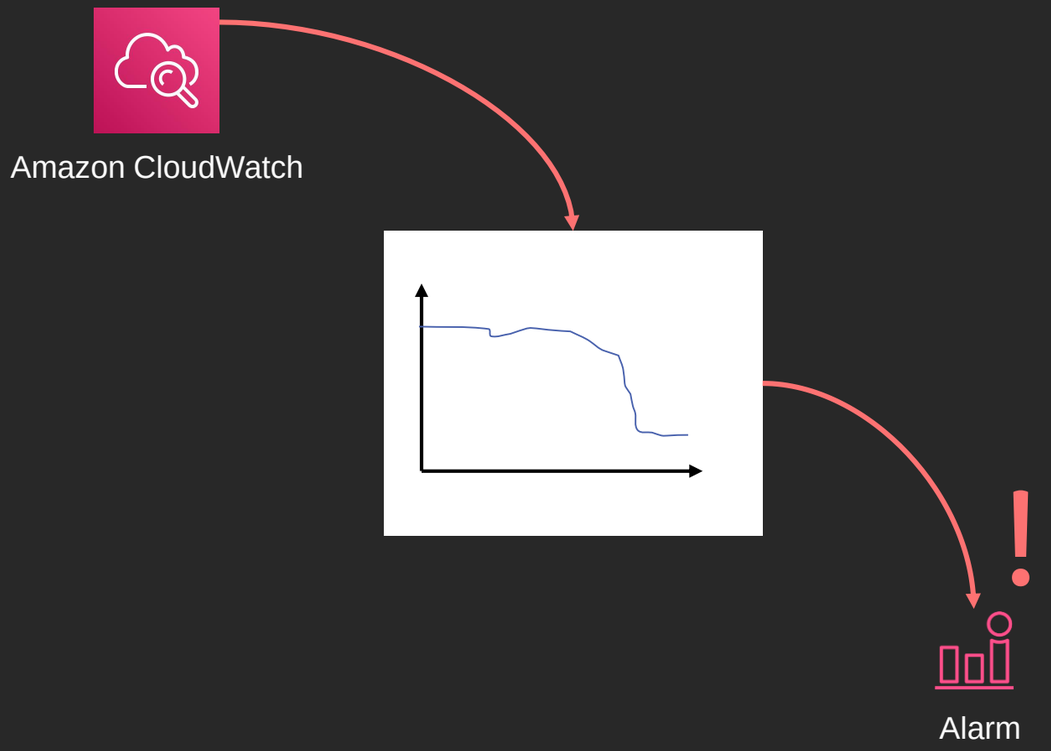
Controlling duration: Improving incident response

Good

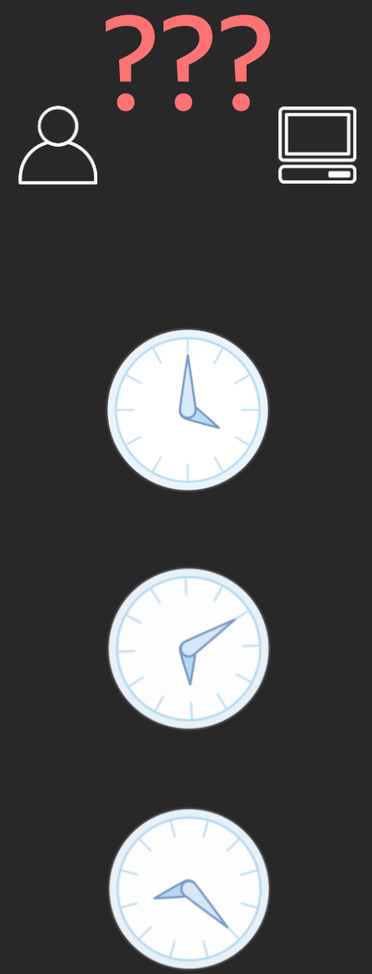


Controlling duration: Improving incident response

Good

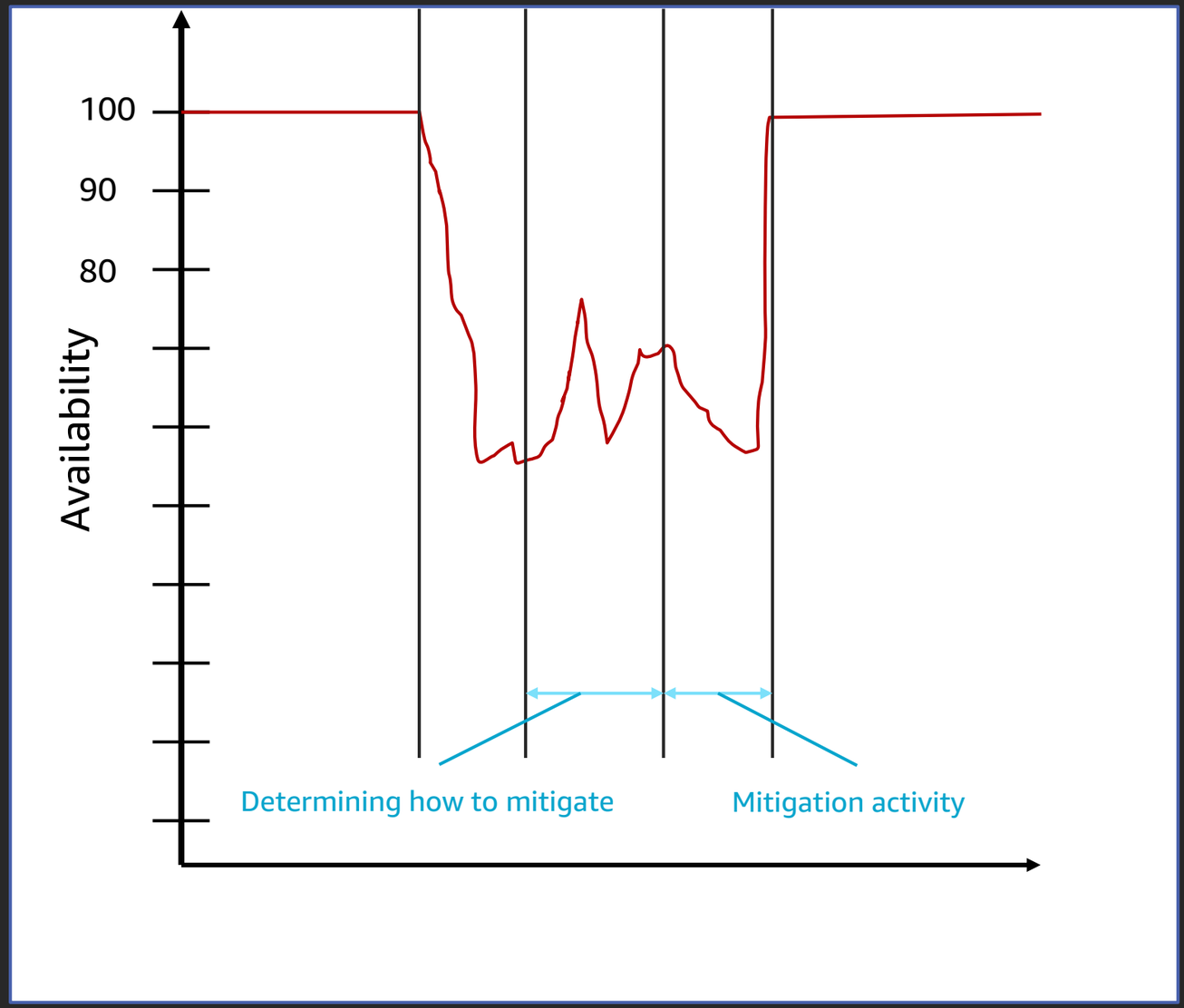


Bad



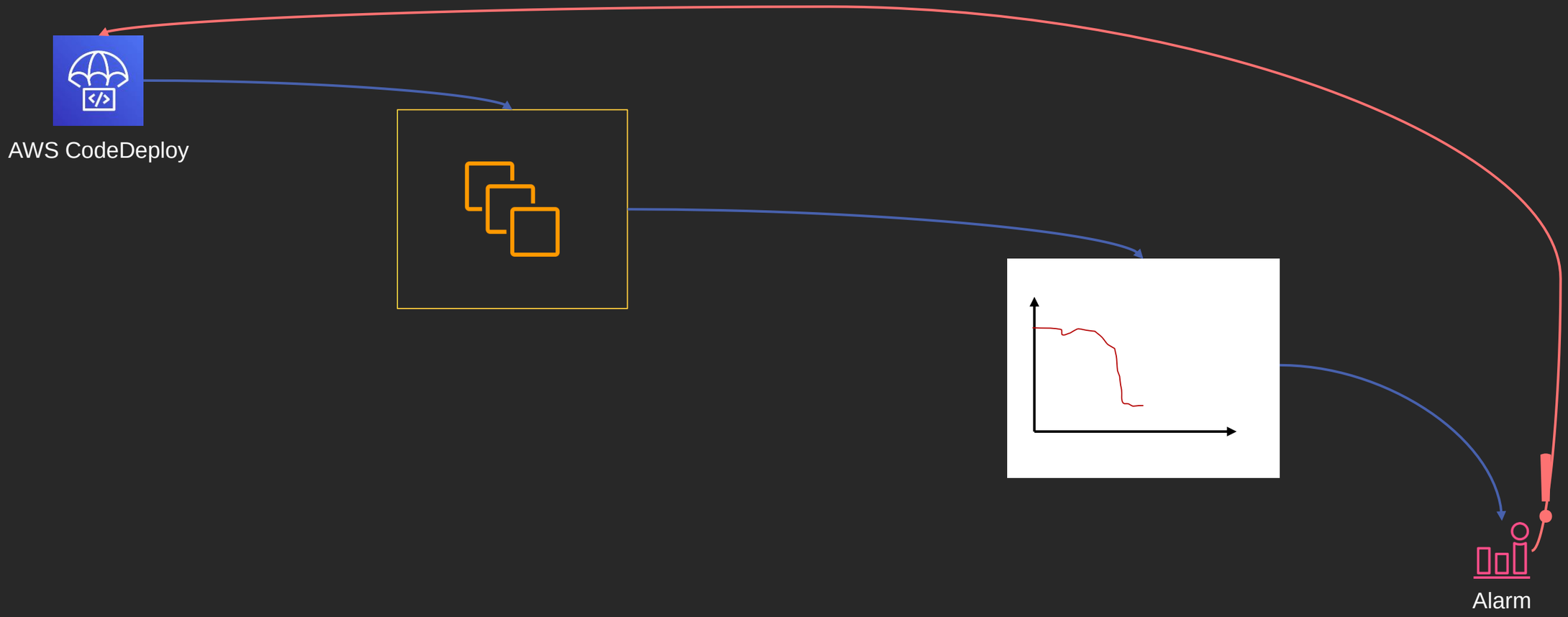
Controlling duration: Improving time to mitigation

- “How did you reach the point where you knew how to mitigate the impact?”
- “How could time to mitigation be improved? As a thought experiment, how would you have cut the time in half?”



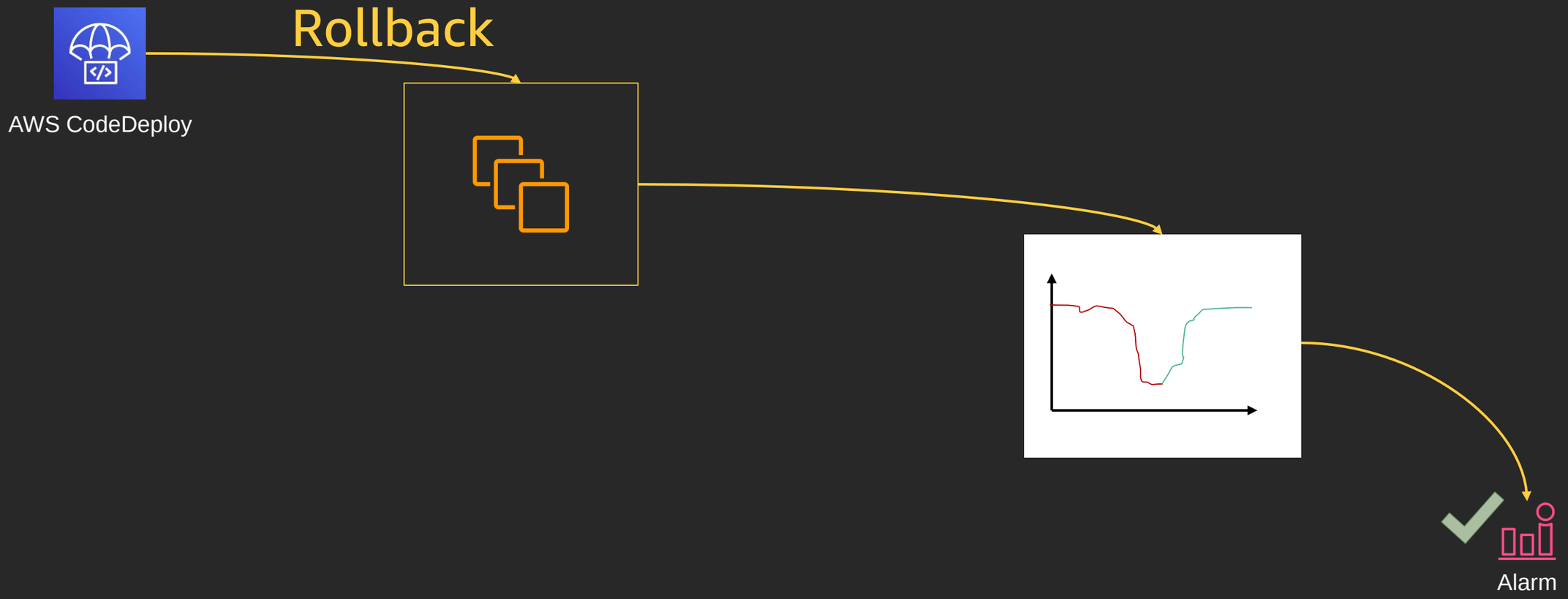
Controlling duration: Improving time to mitigation

Example: Alarm-based automatic rollback

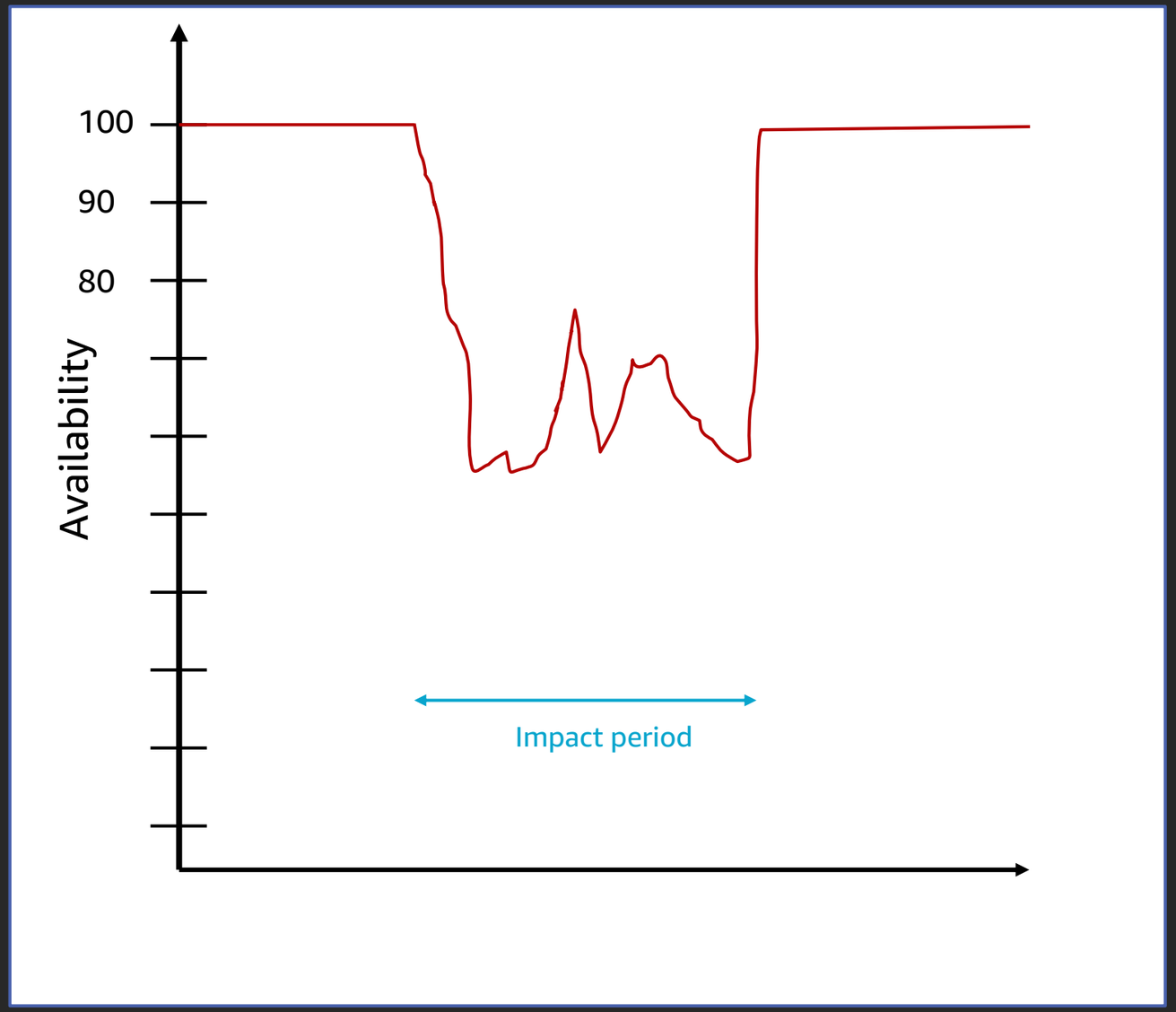


Controlling duration: Improving time to mitigation

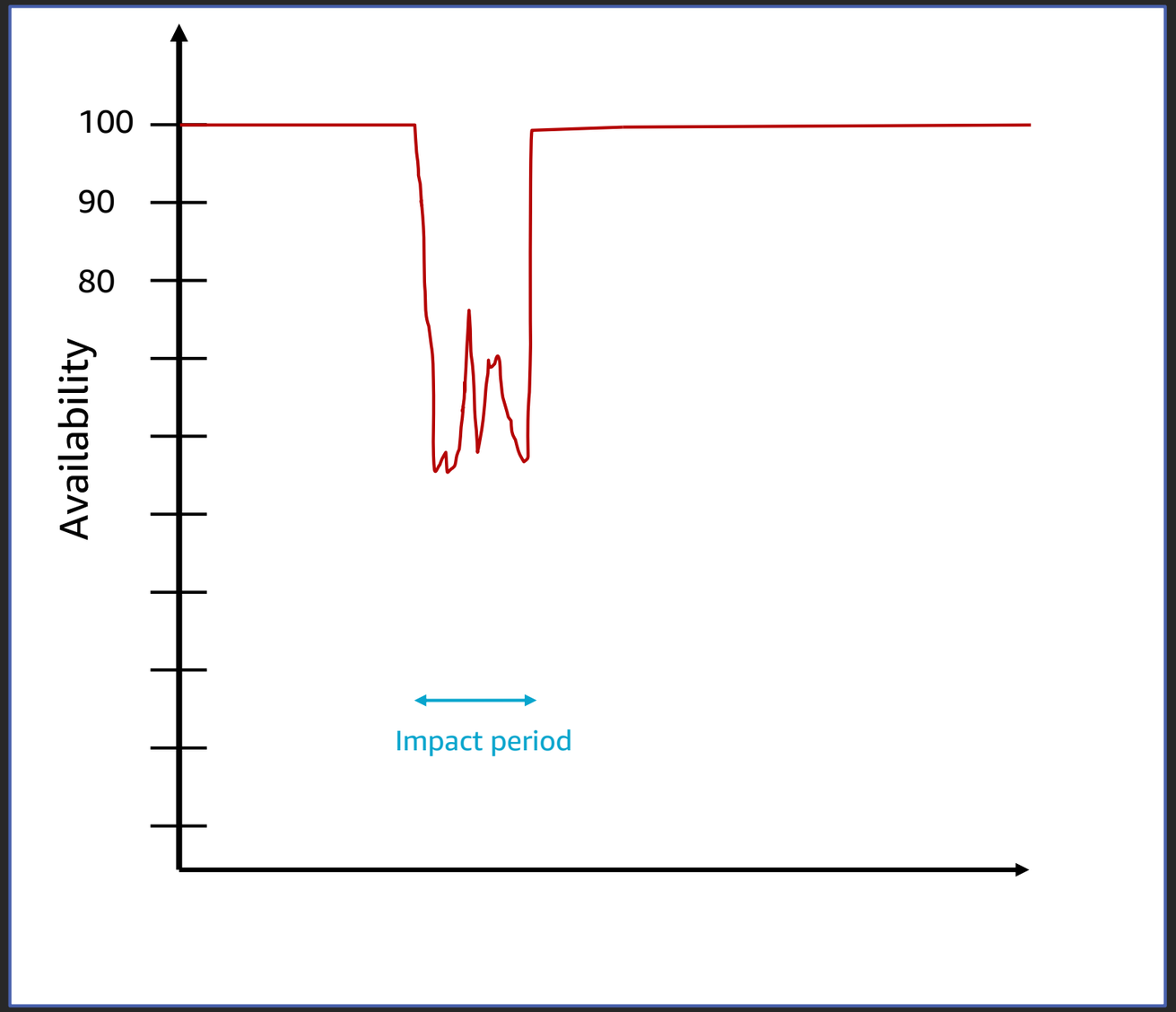
Example: Alarm-based automatic rollback



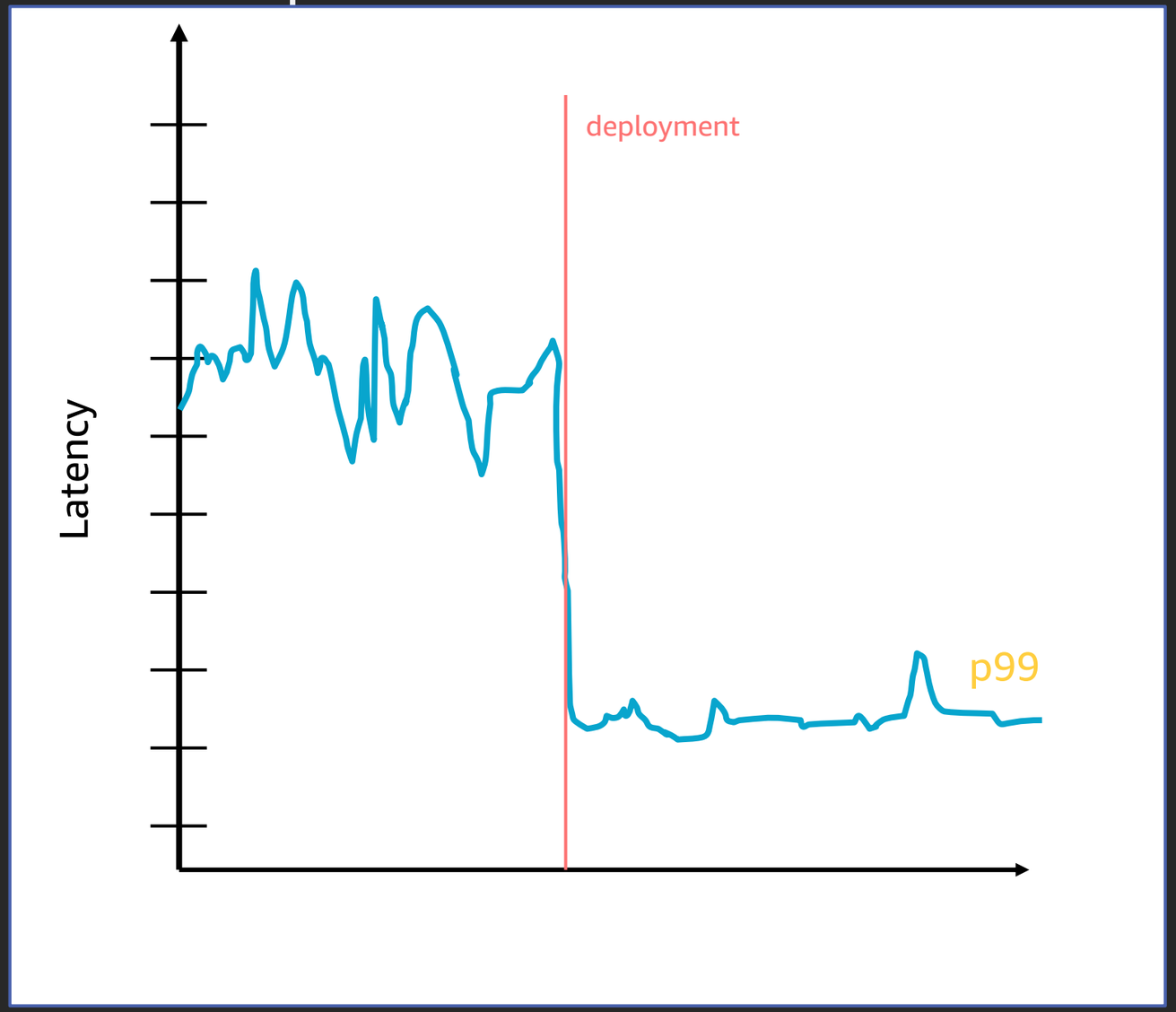
Controlling event duration



Controlling event duration

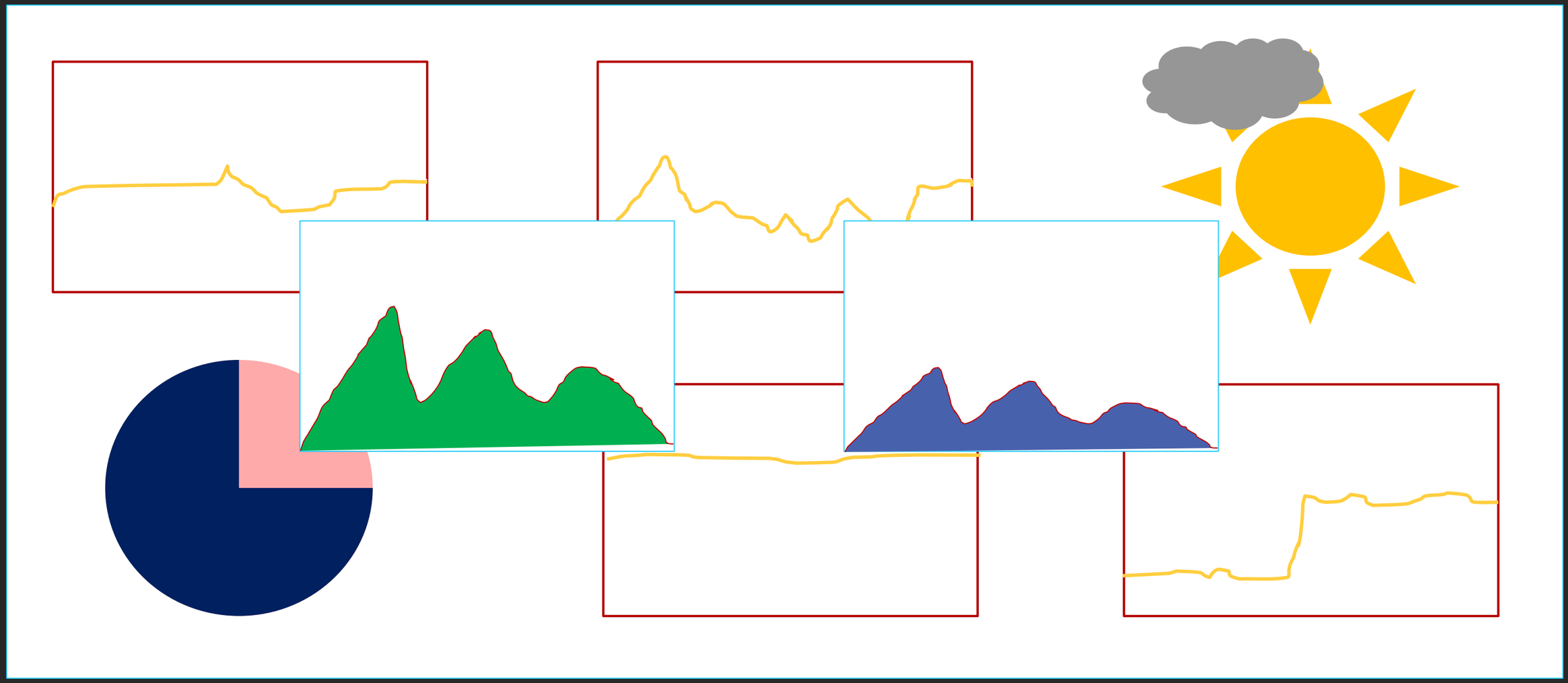


Wins are just as important as failures



Watching yourself fail (before your customers do)

Metrics are very interesting, and that can be a problem



Health metrics and diagnostic metrics

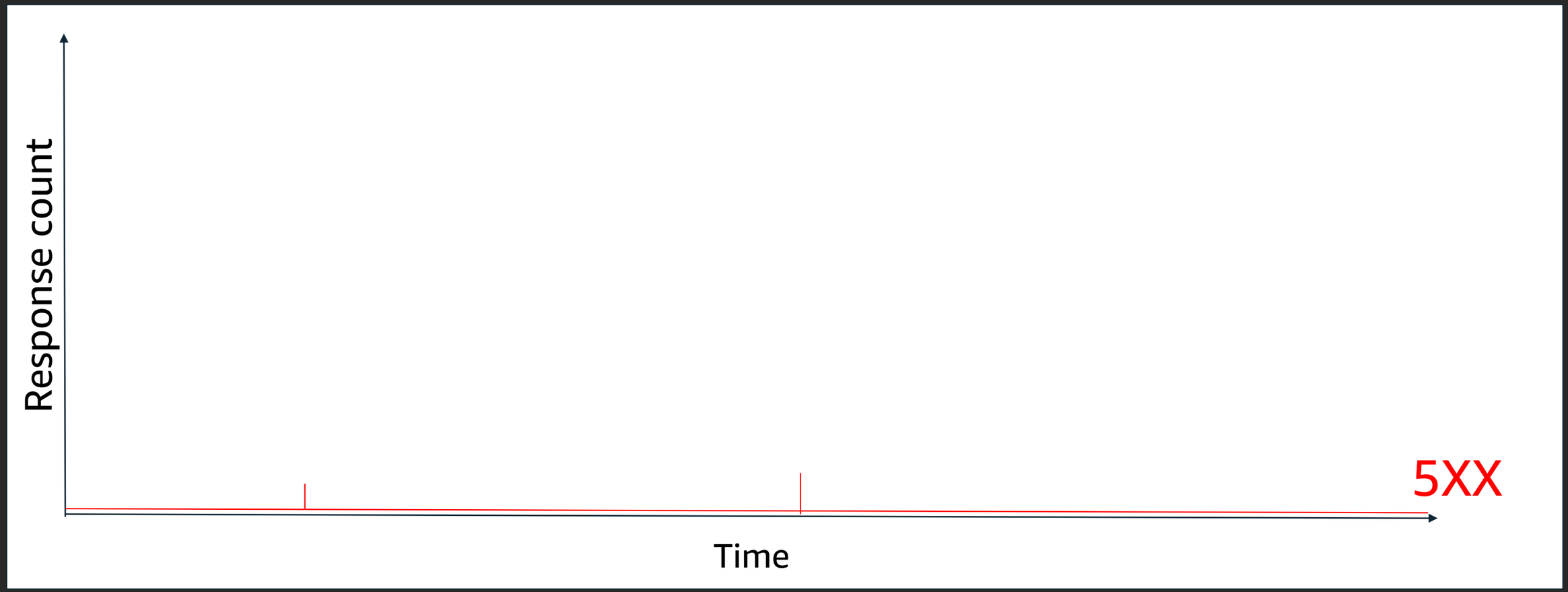
Health metrics

- *Answers* the question: Am I failing?
- *Does not answer* the question: Why am I failing?
- *Always* set alarms on these
- Be conservative in defining

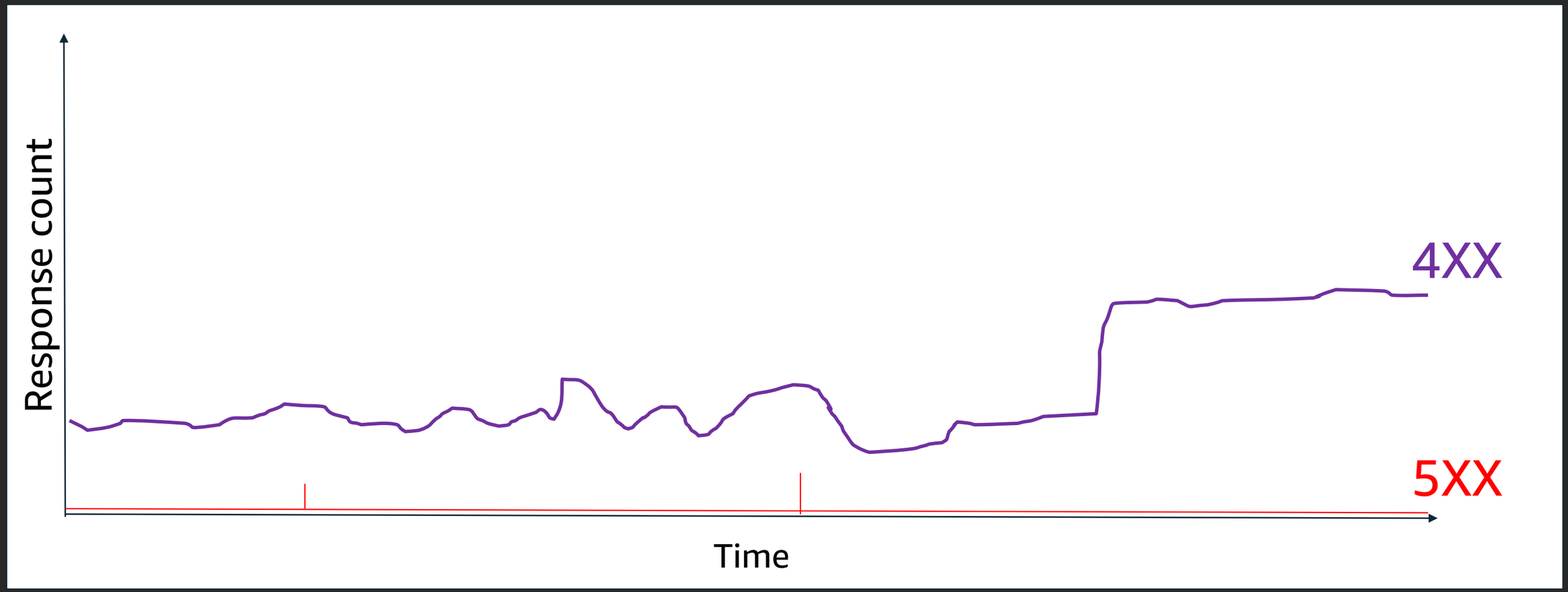
Diagnostic metrics

- *Answers* the question: What is the value of this thing I measured?
- *Might answer* the question: Why isn't my system working?
- *Sometimes* set alarms on these
- Be liberal in defining

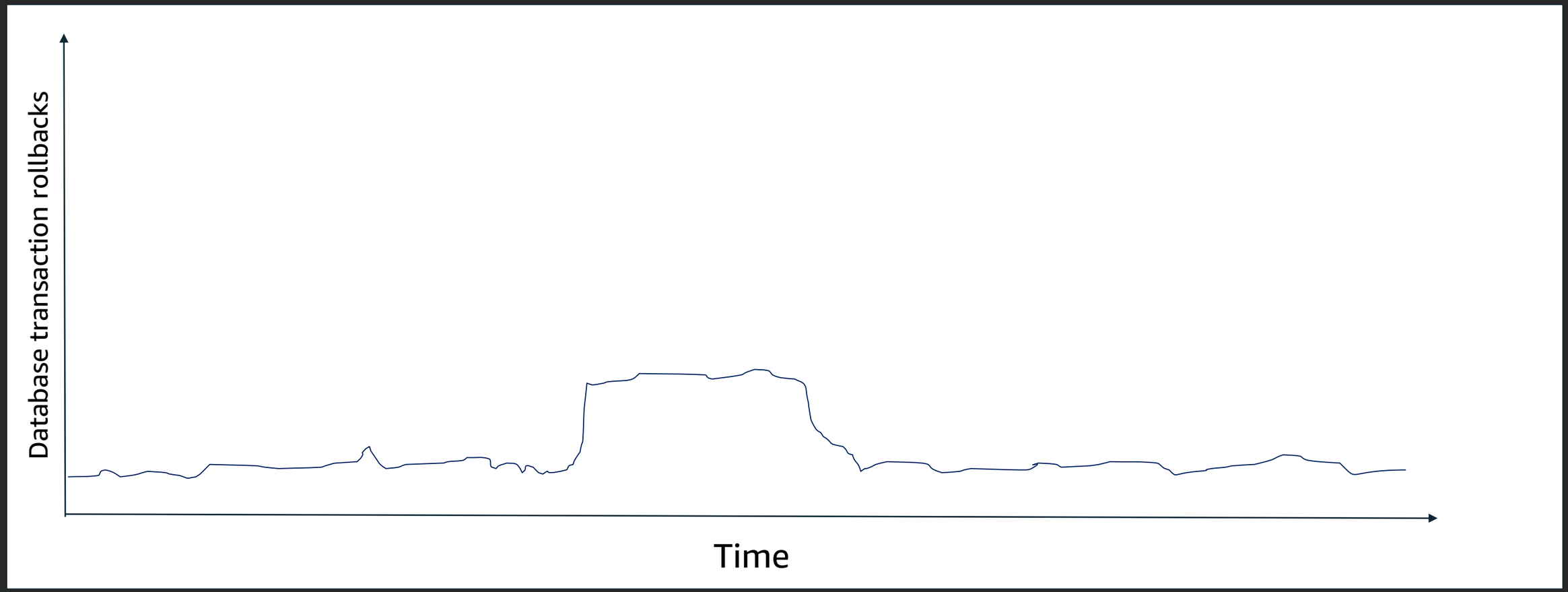
Health or diagnostic?



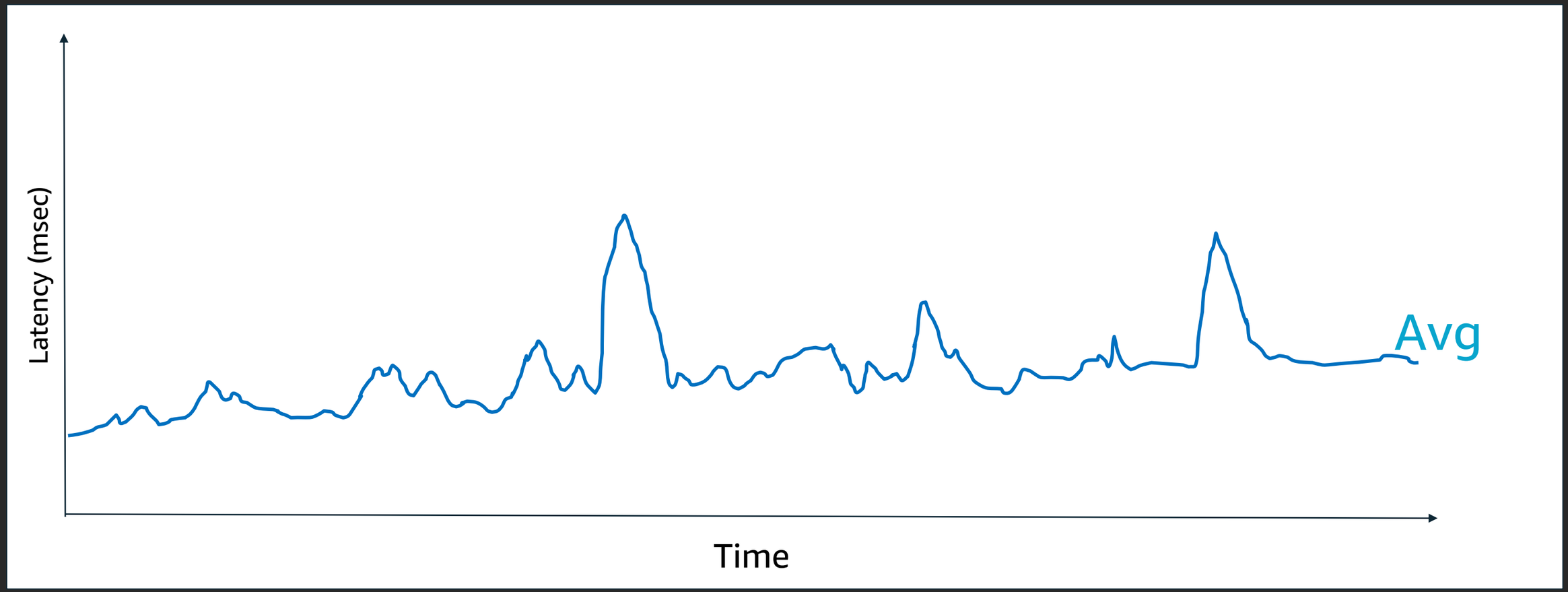
Health or diagnostic?



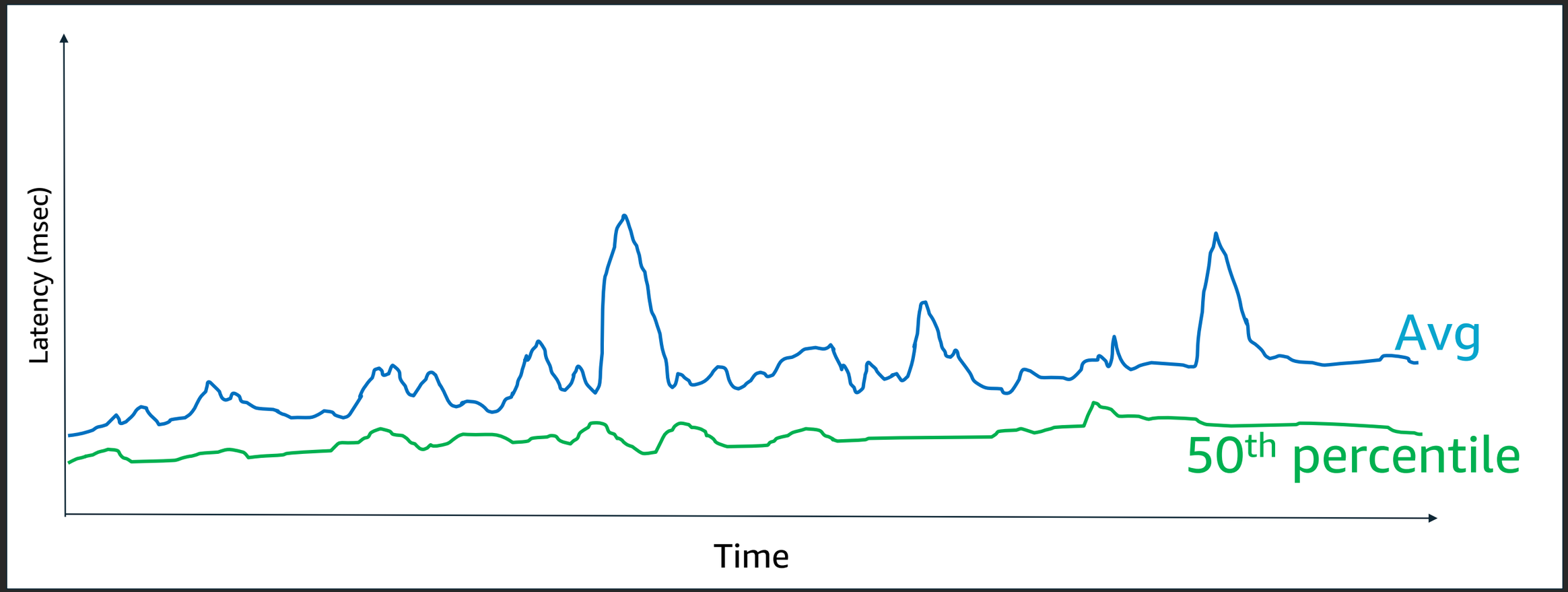
Health or diagnostic?



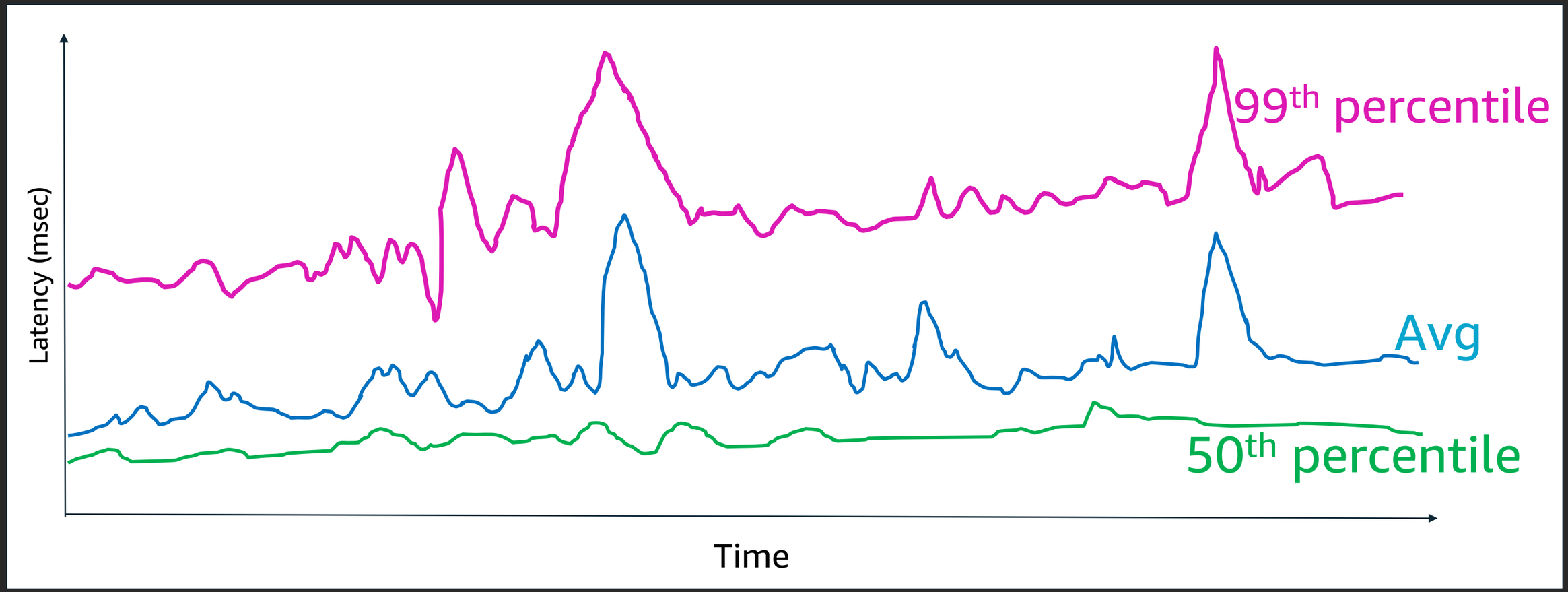
Health or diagnostic?



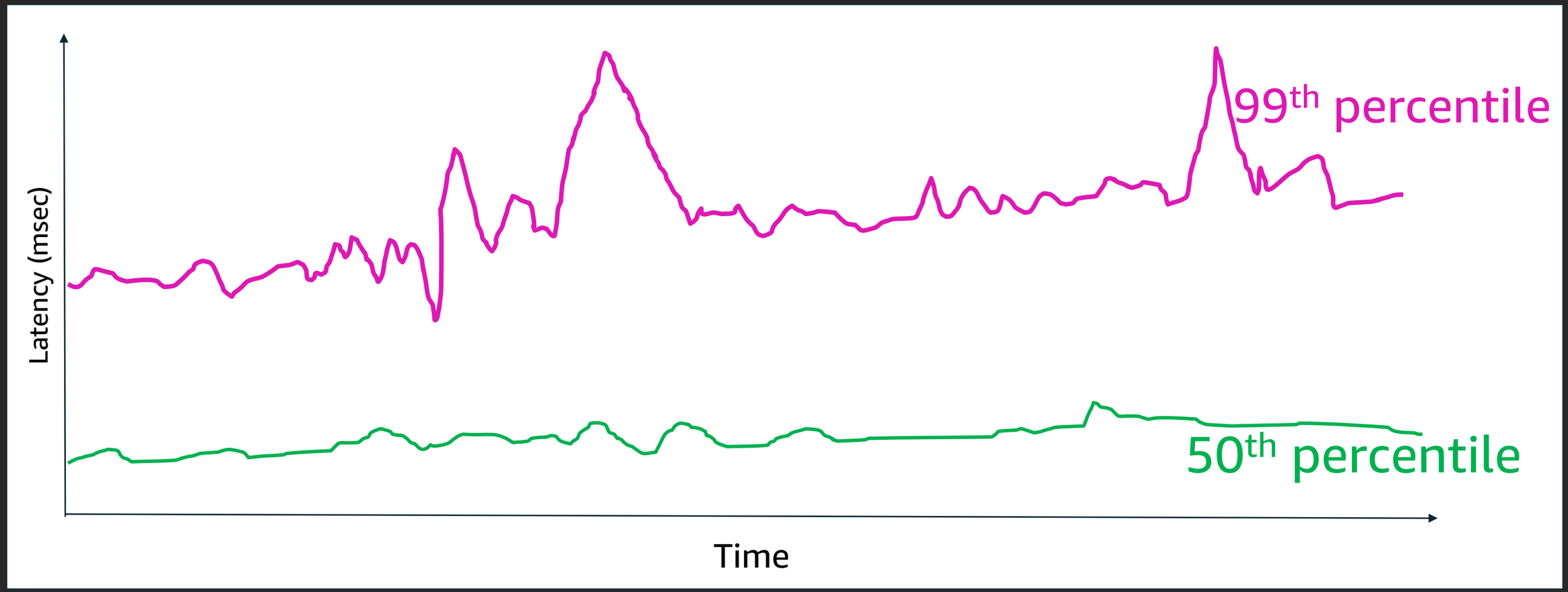
Health or diagnostic?



Health or diagnostic?



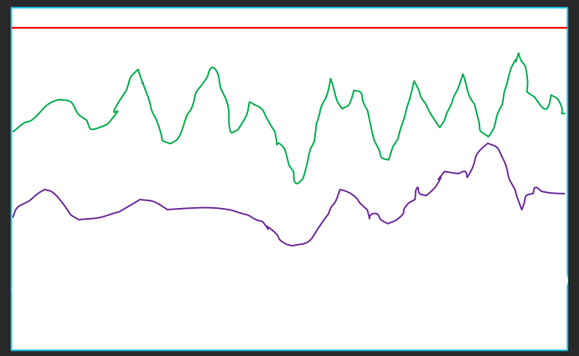
Percentiles >> Avg



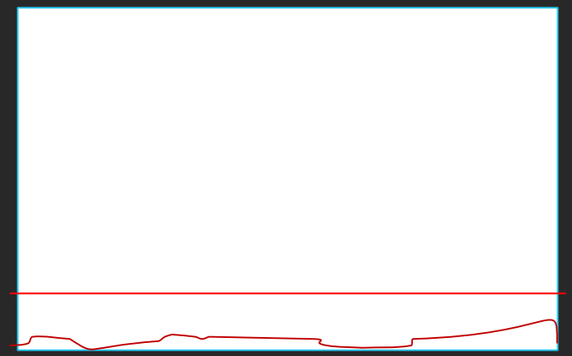
Layout of a great dashboard

Health metrics at the top

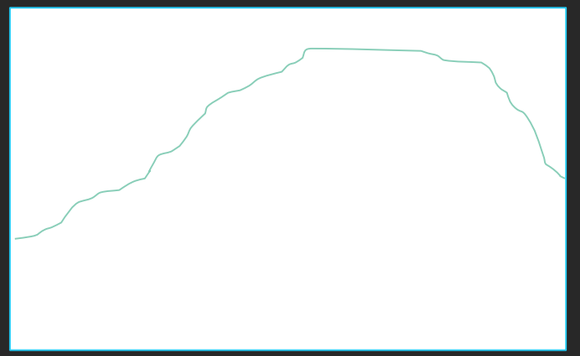
Latency percentiles



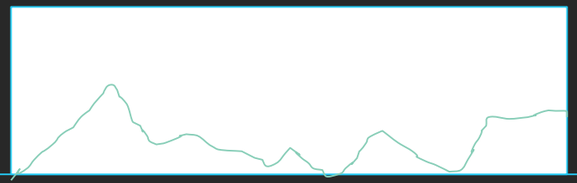
Faults



Volume

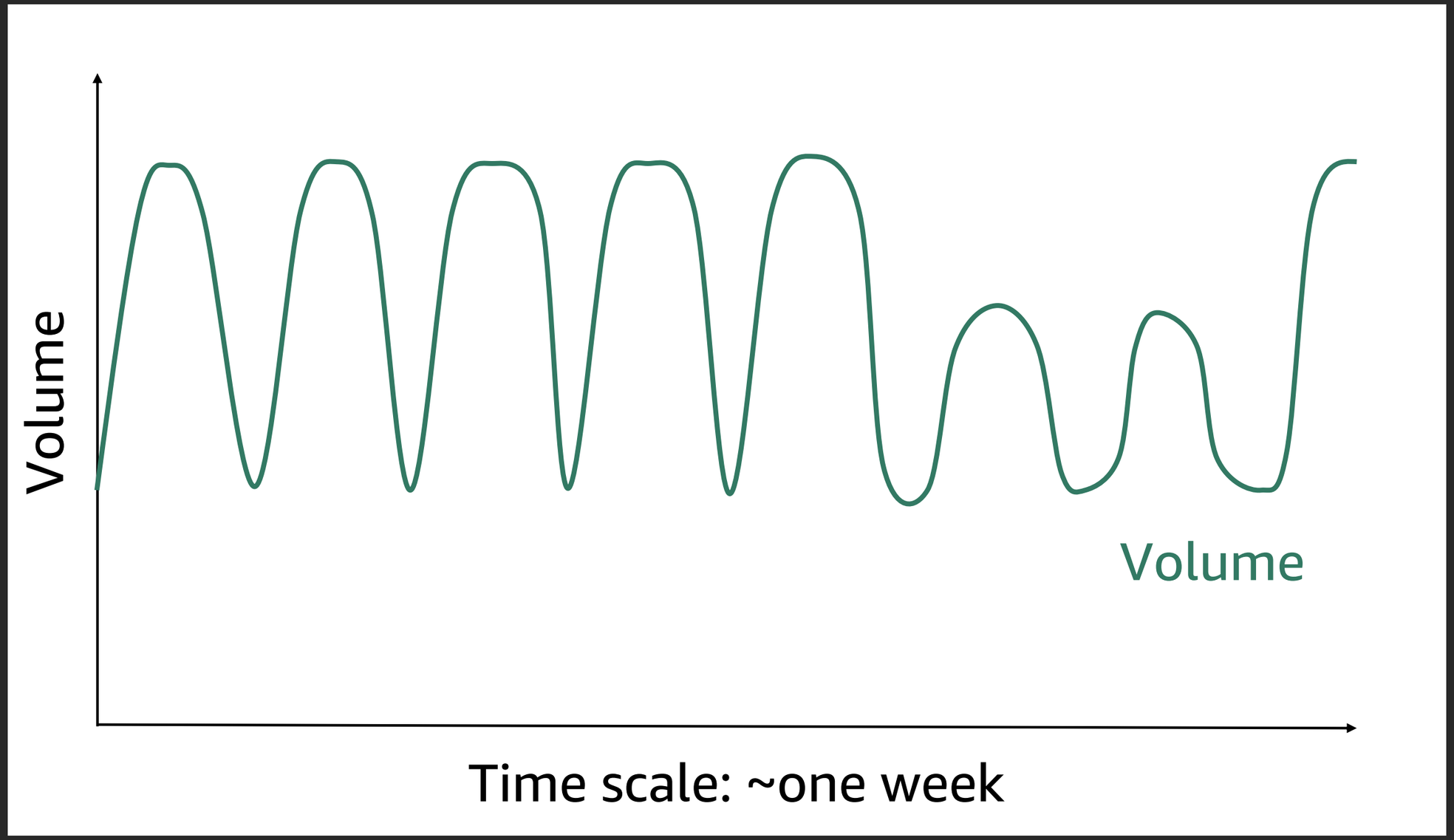


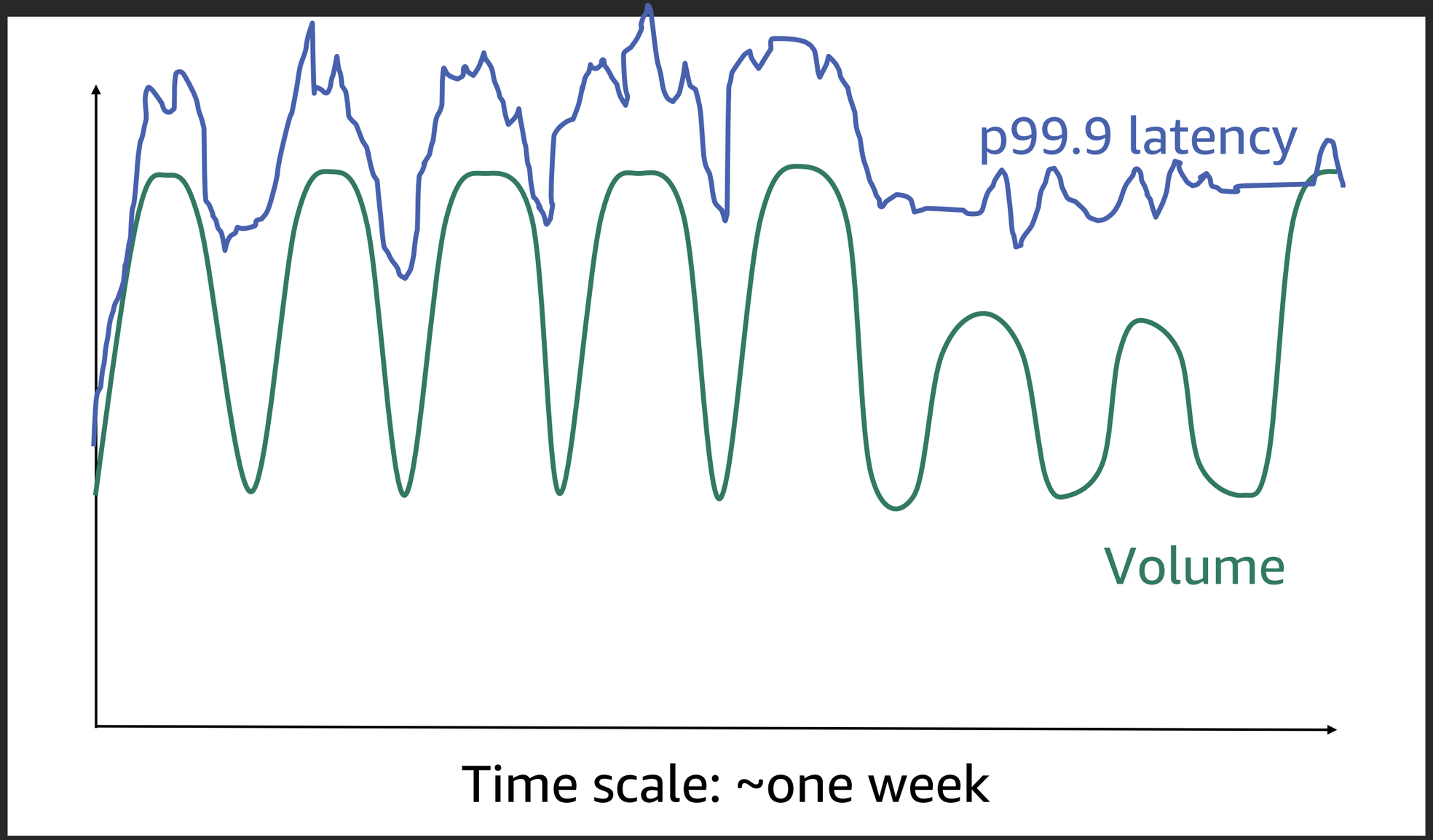
Key diagnostic metrics below the fold

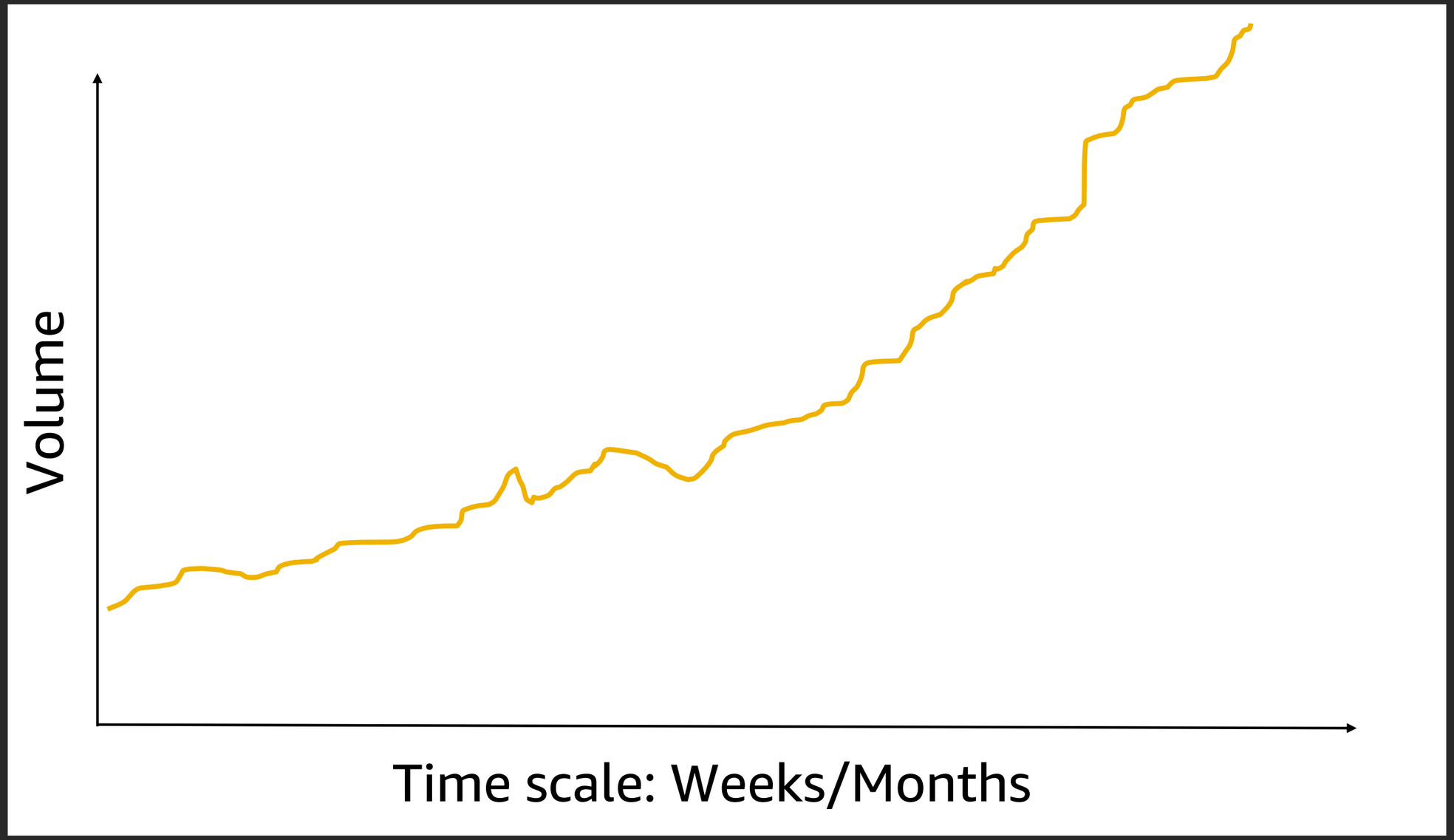


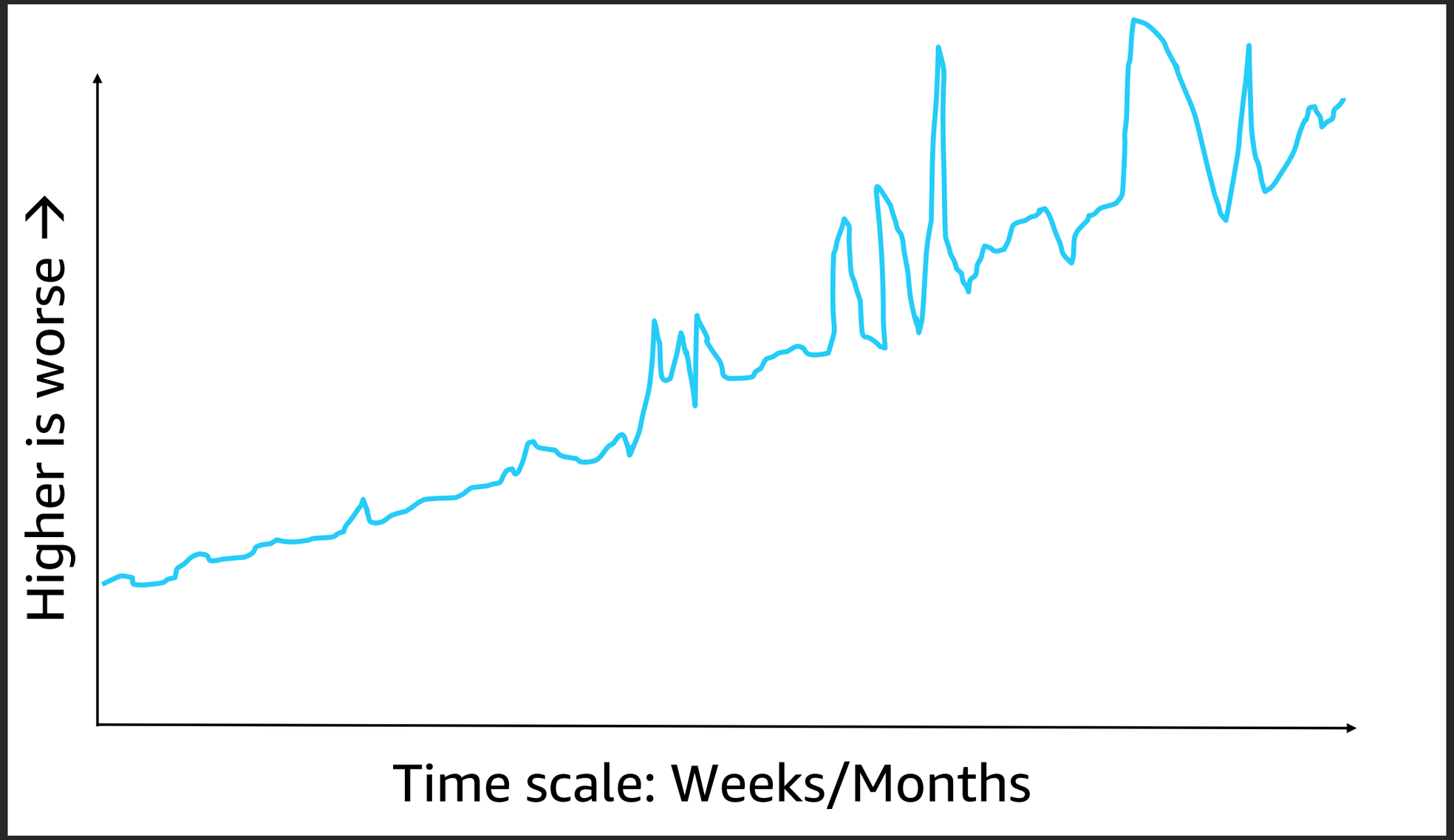
The AWS Ops Wheel

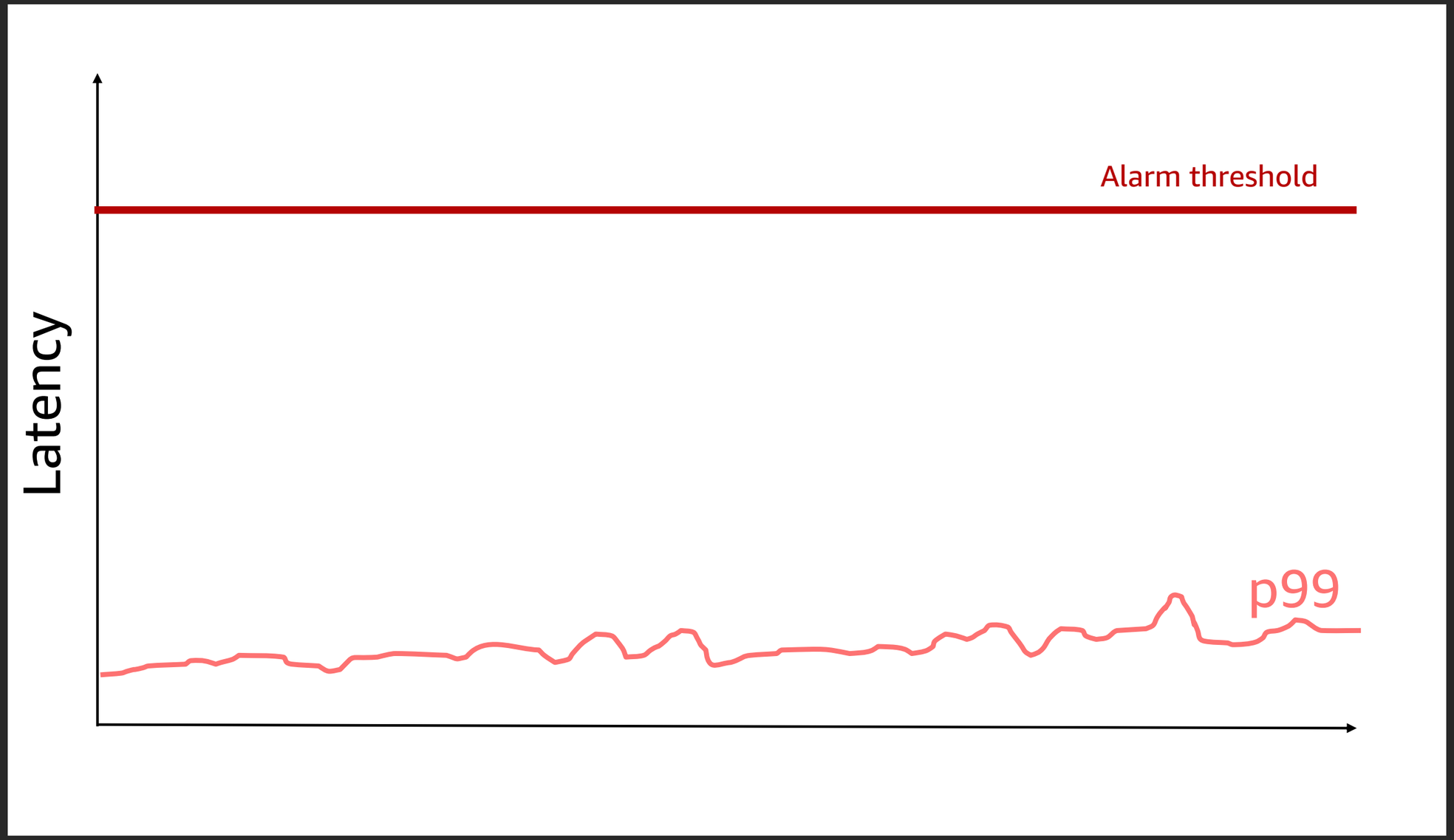
Looking for failure in your metrics

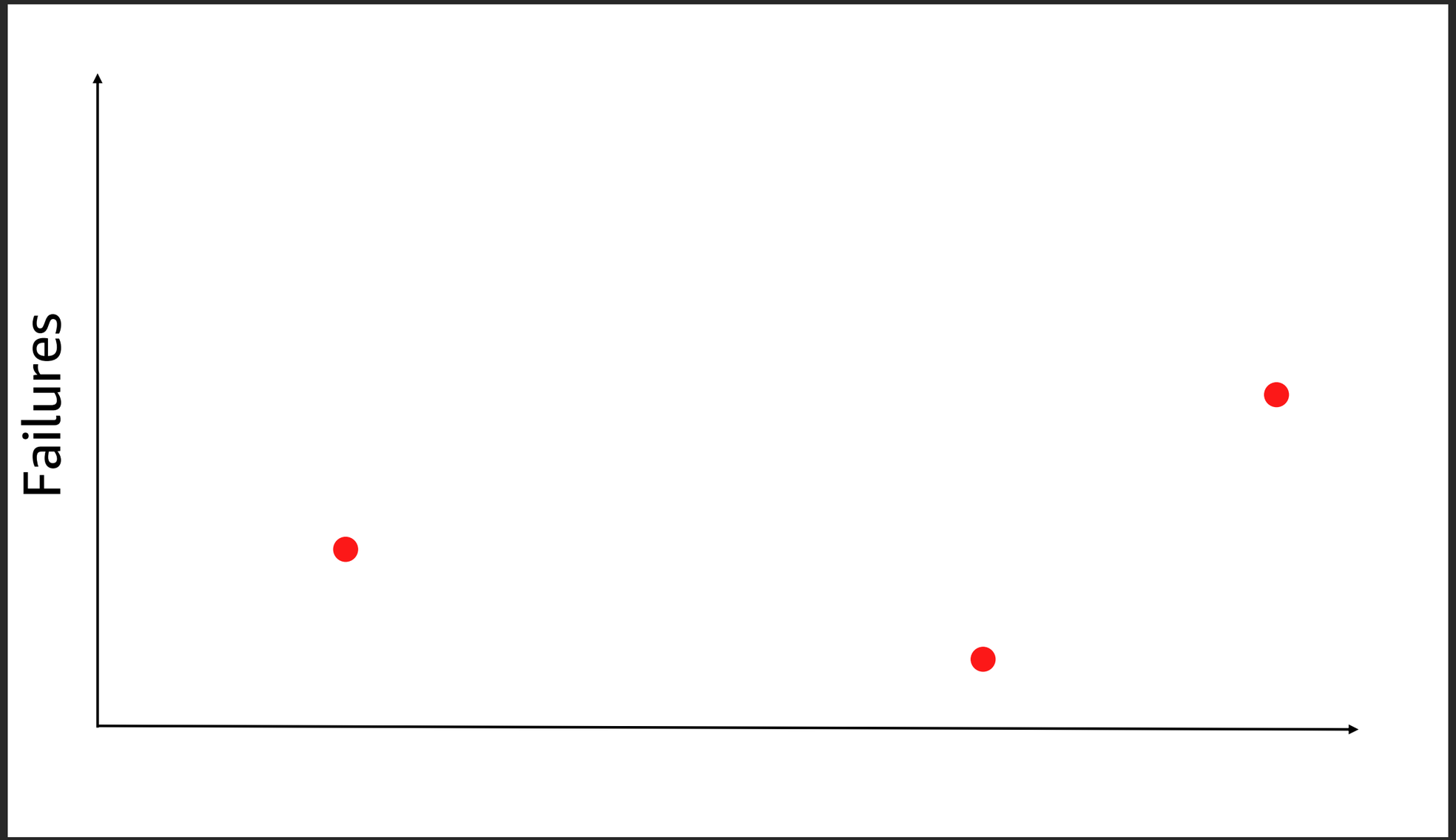

Volume
Volume
Time scale: ~one week

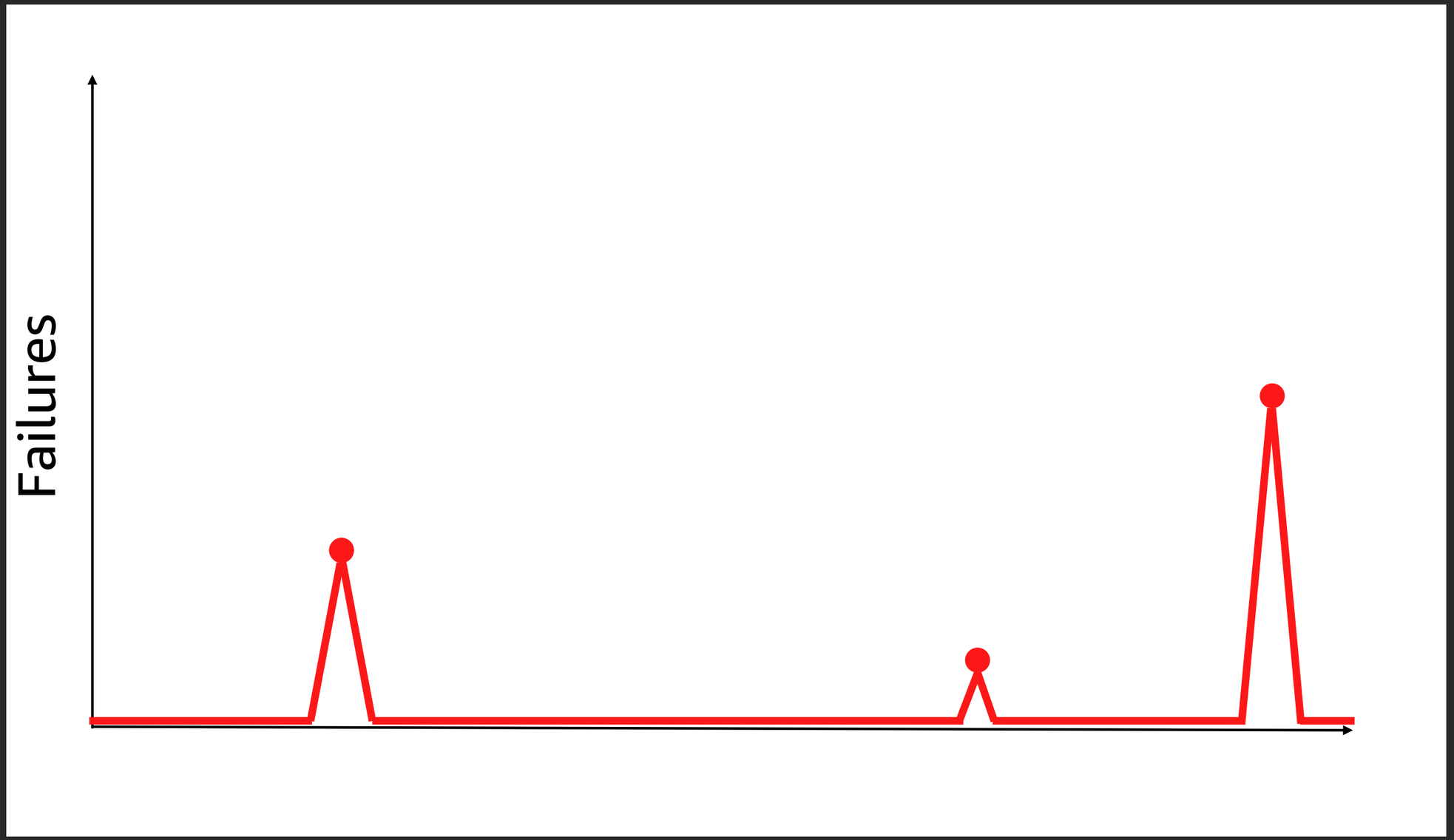

p99.9 latency
Volume
Time scale: ~one week

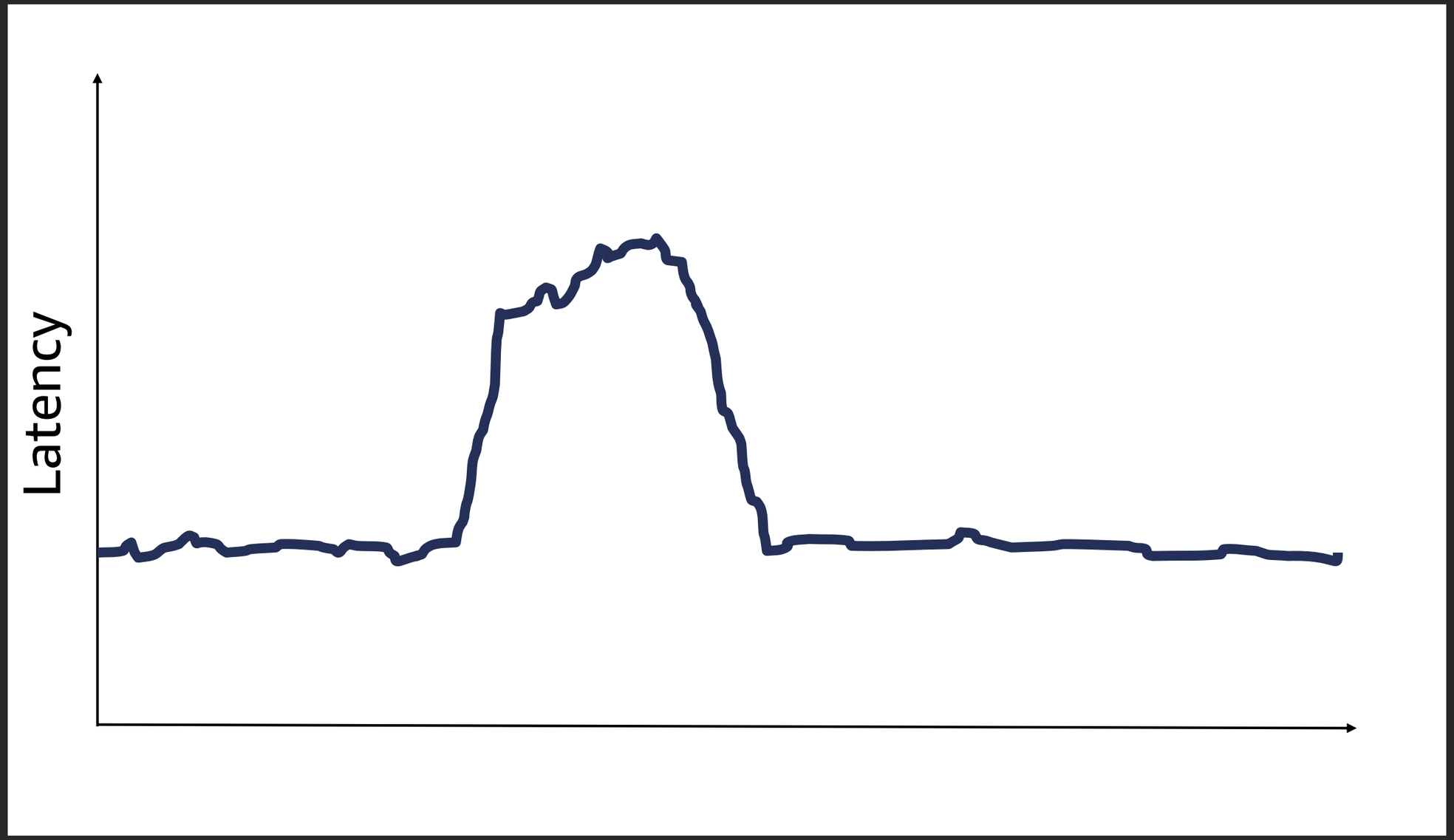

Volume
Time scale: Weeks/Months

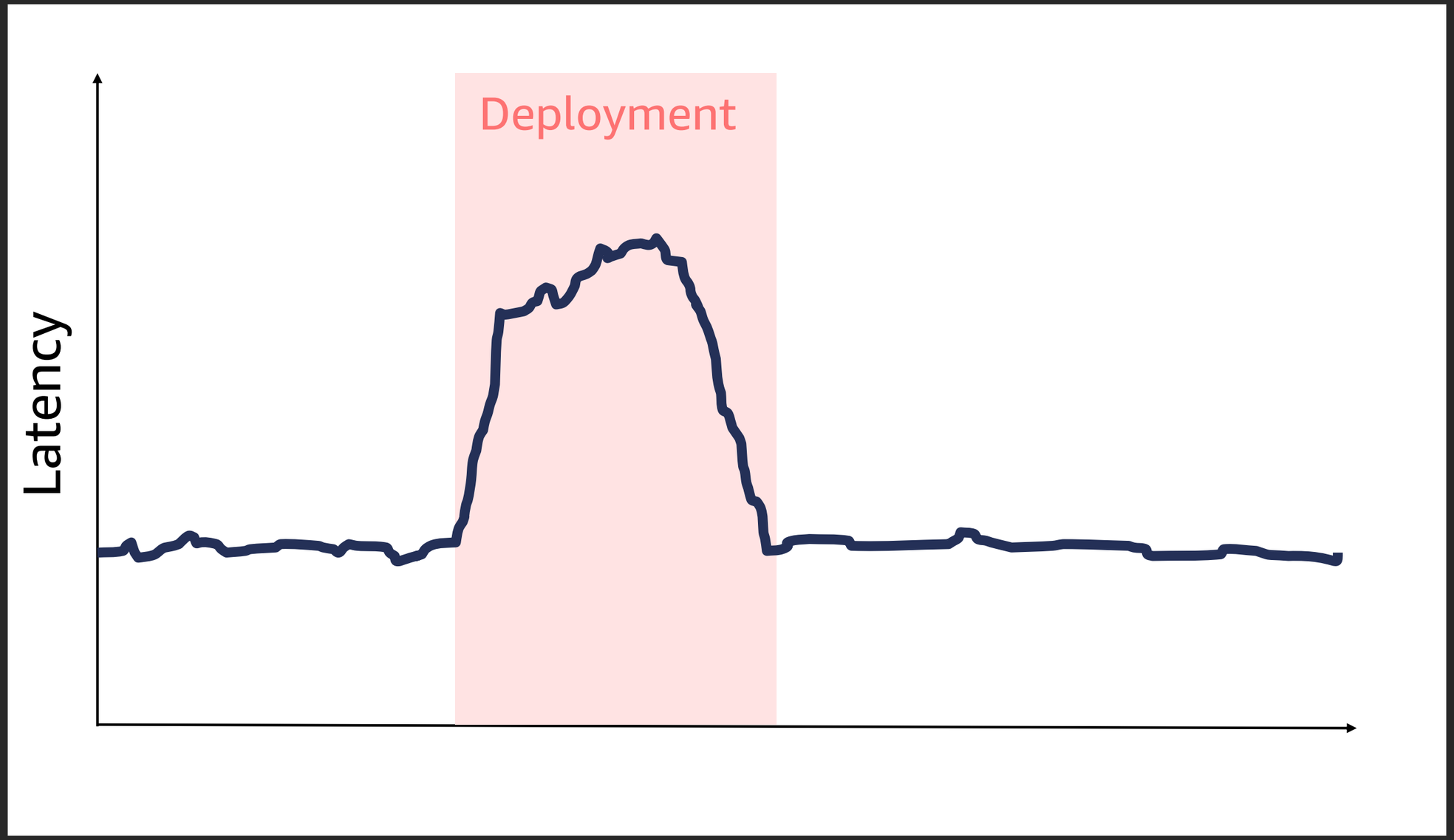

Higher is worse →
Time scale: Weeks/Months

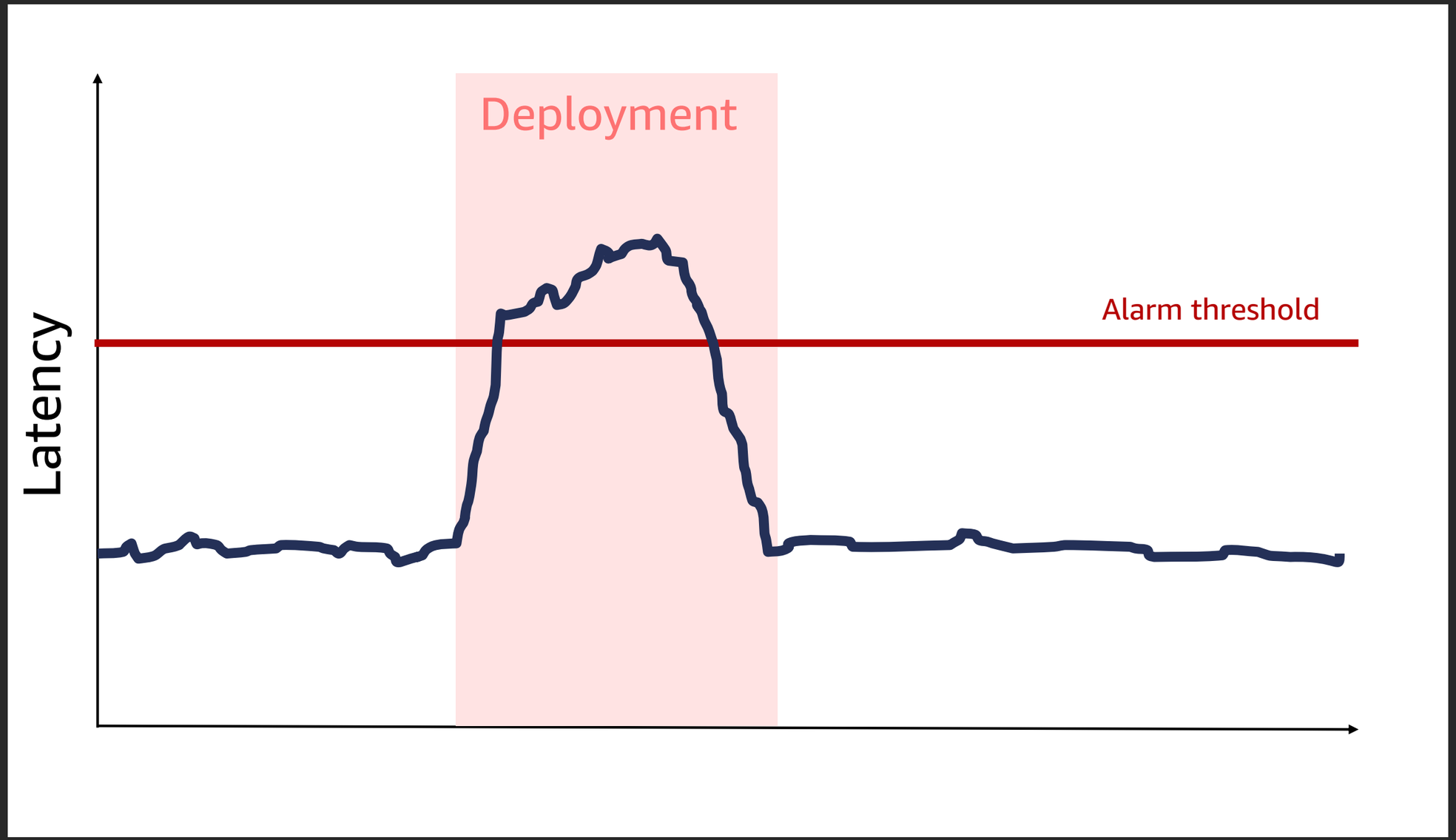

Alarm threshold
Latency
p99


Failures


Failures

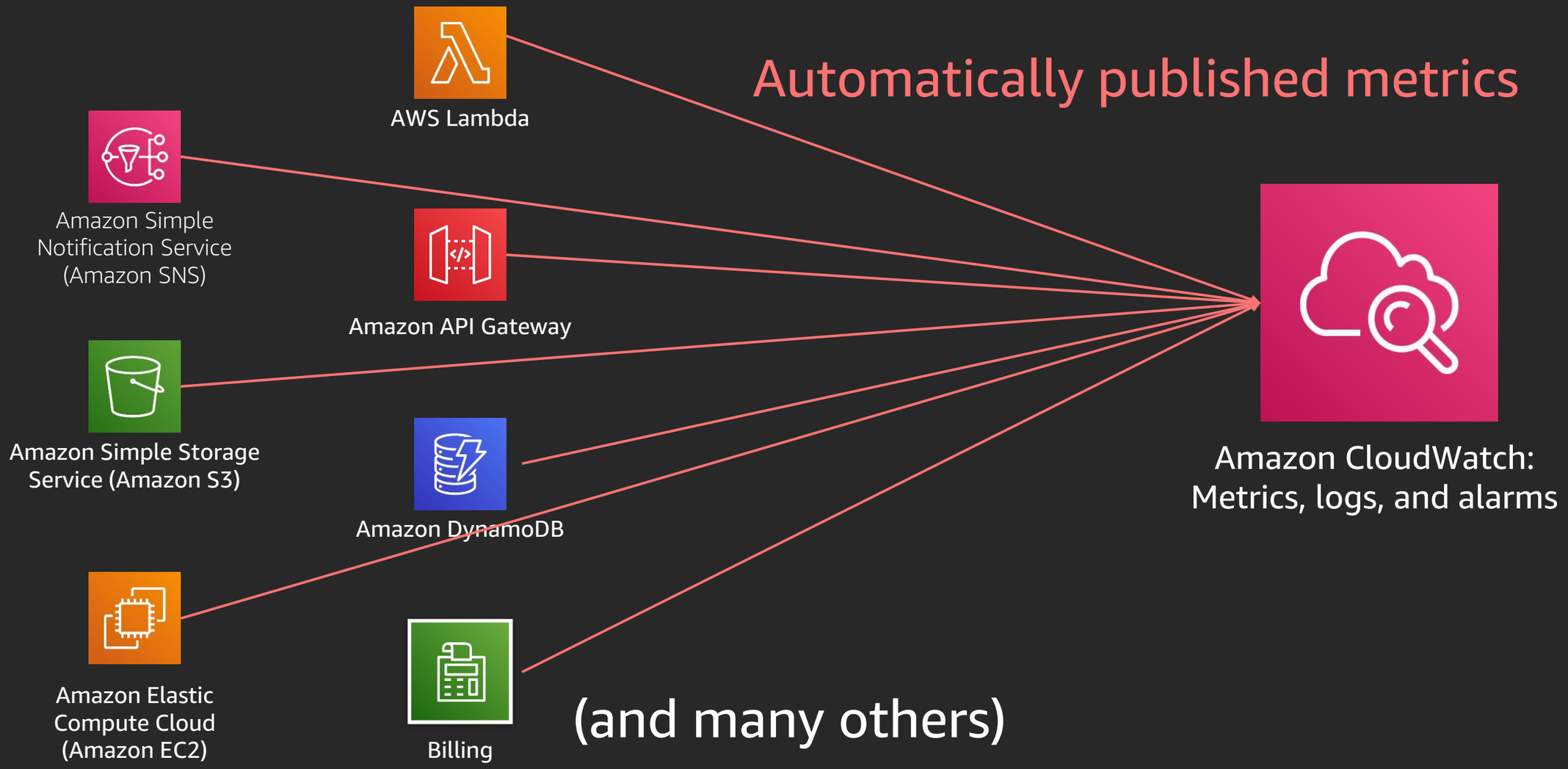

Latency


Deployment
Latency

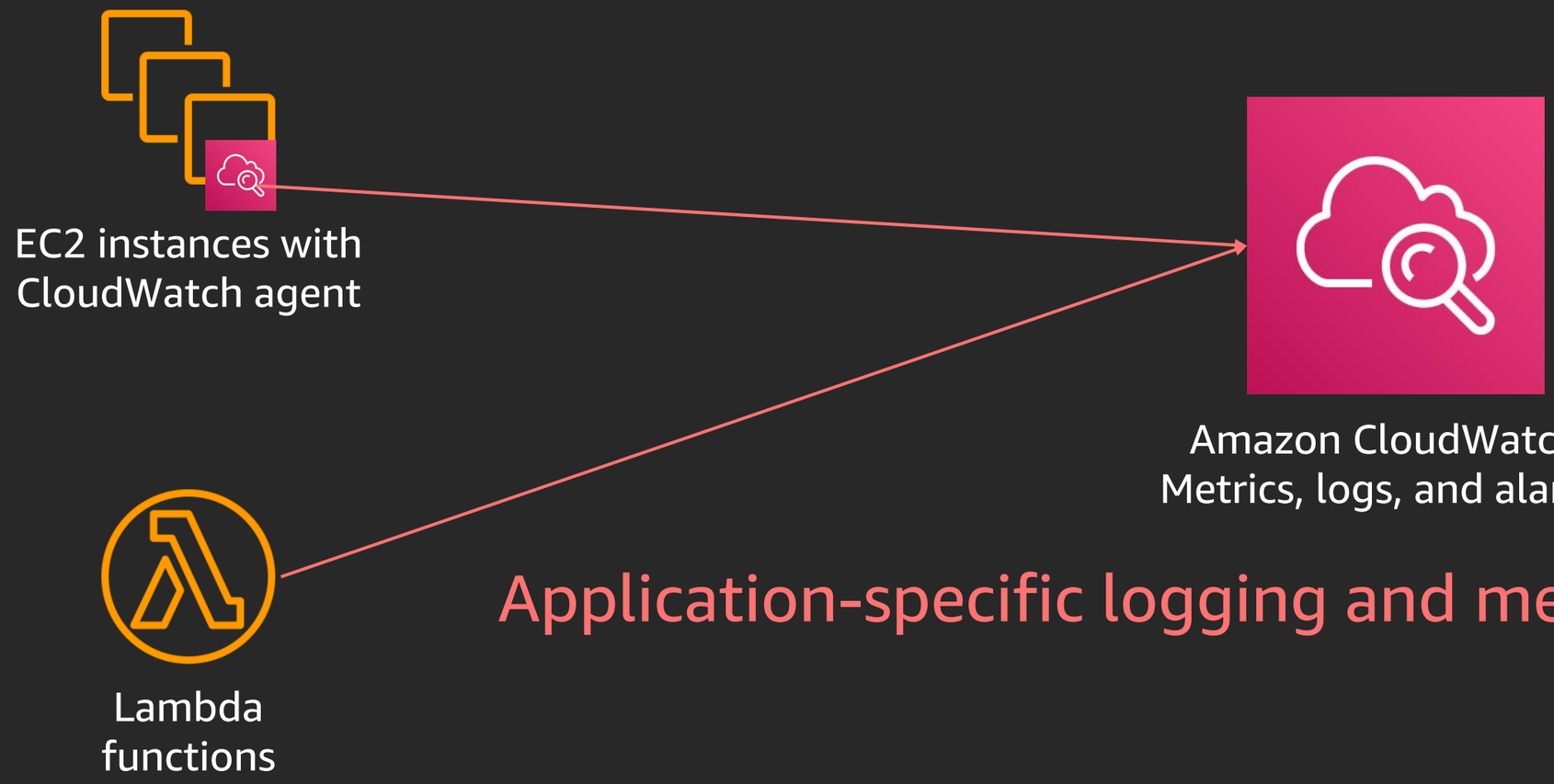

Deployment
Alarm threshold
Latency

How AWS can help you

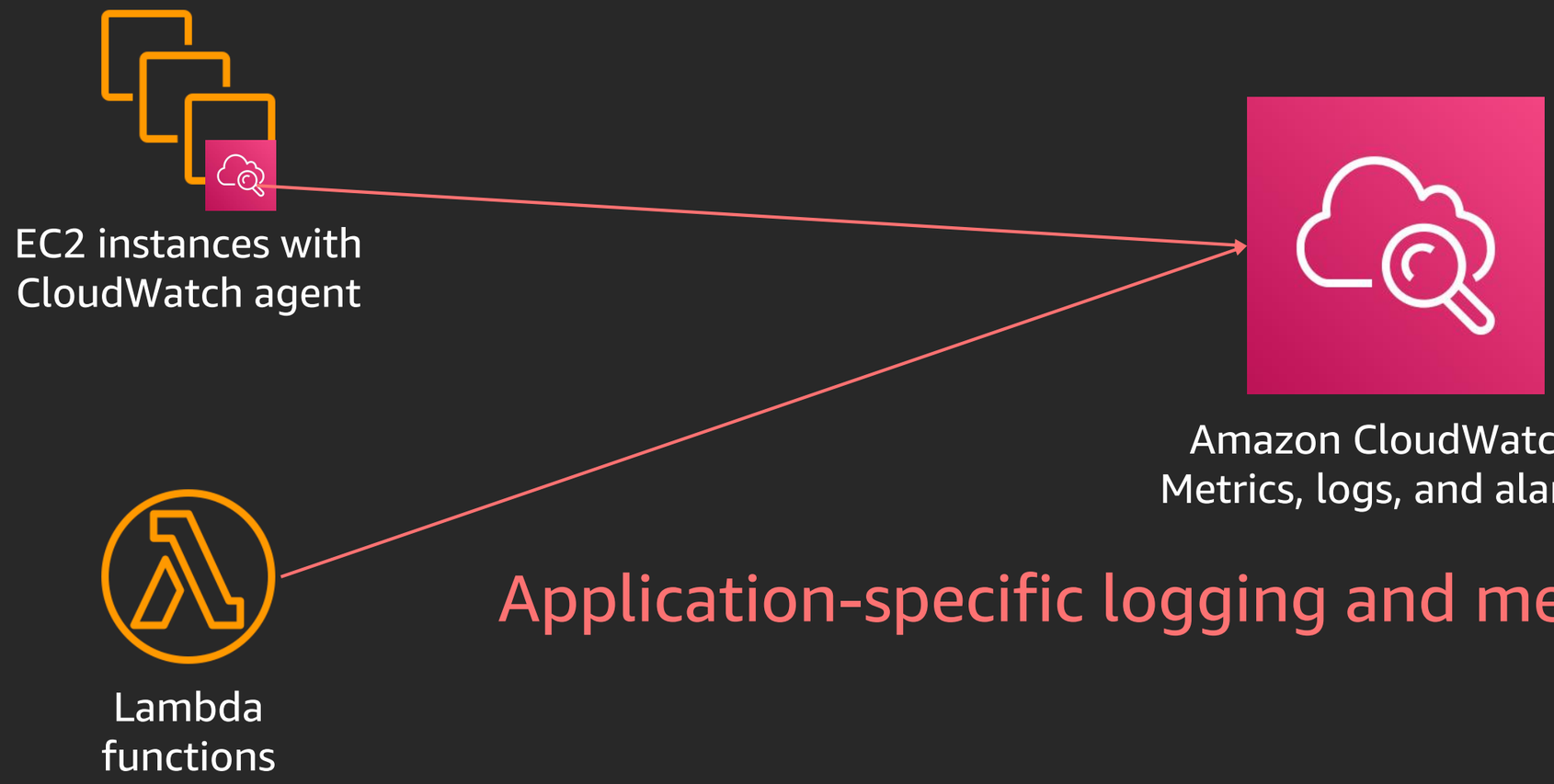
Key service: Amazon CloudWatch



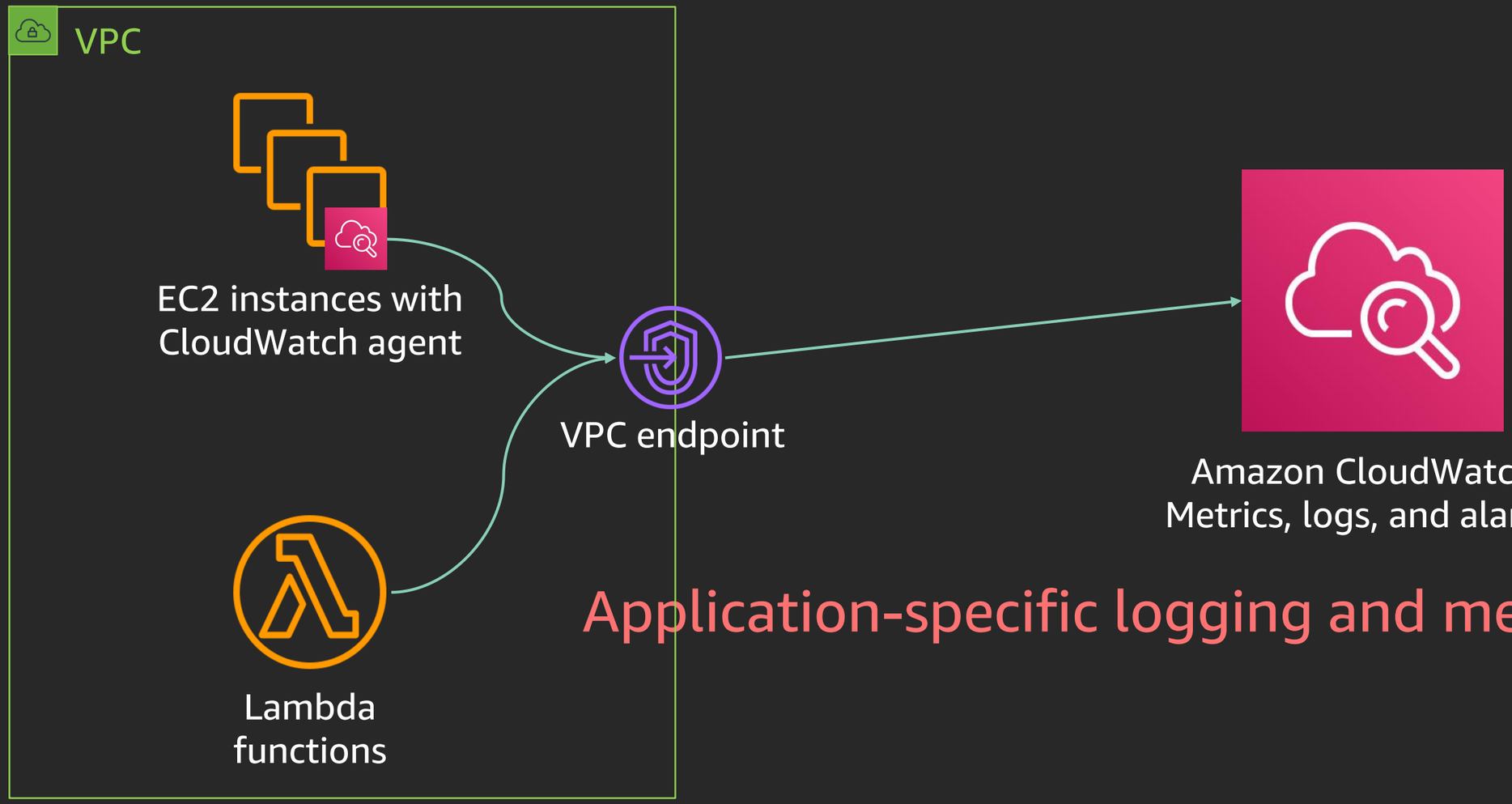
Key service: Amazon CloudWatch



Key service: Amazon CloudWatch



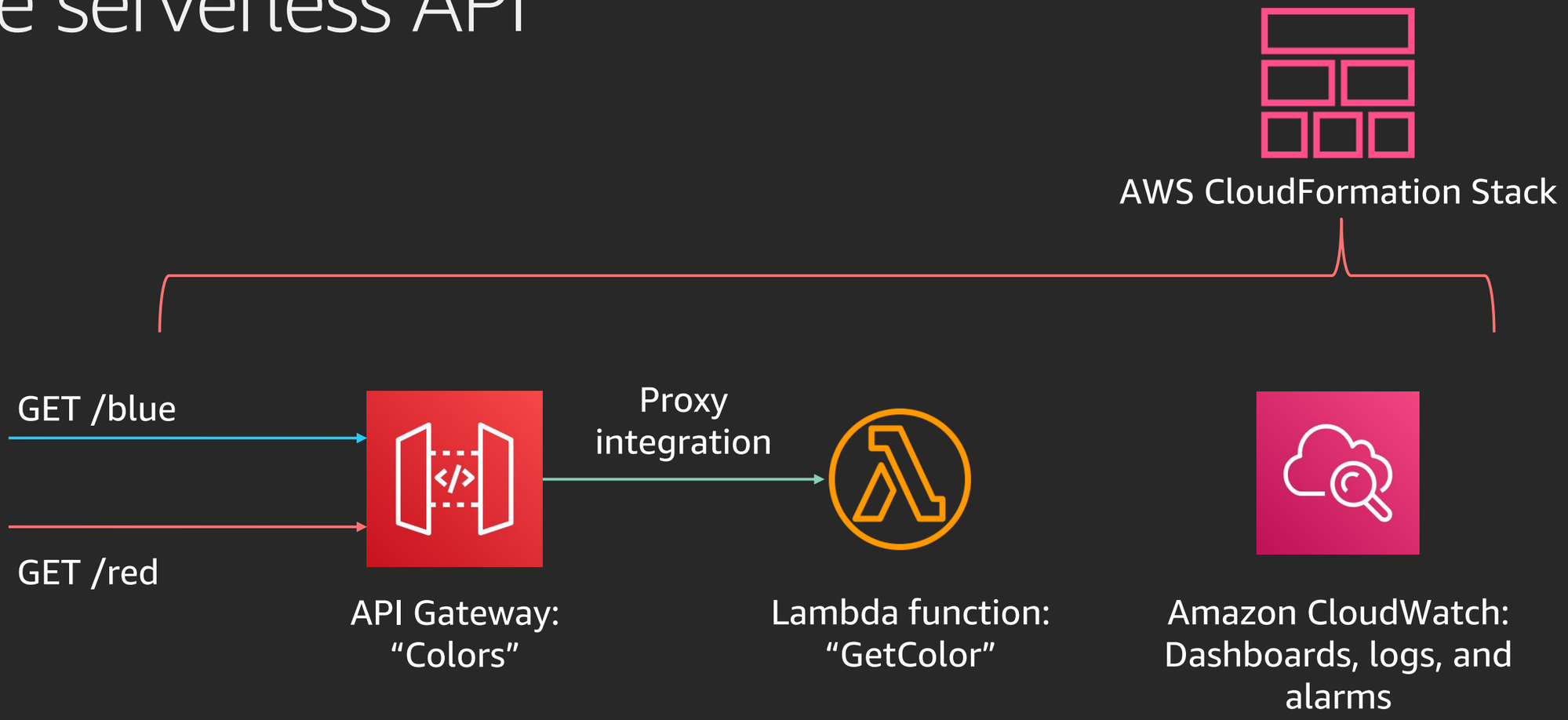
Key service: Amazon CloudWatch



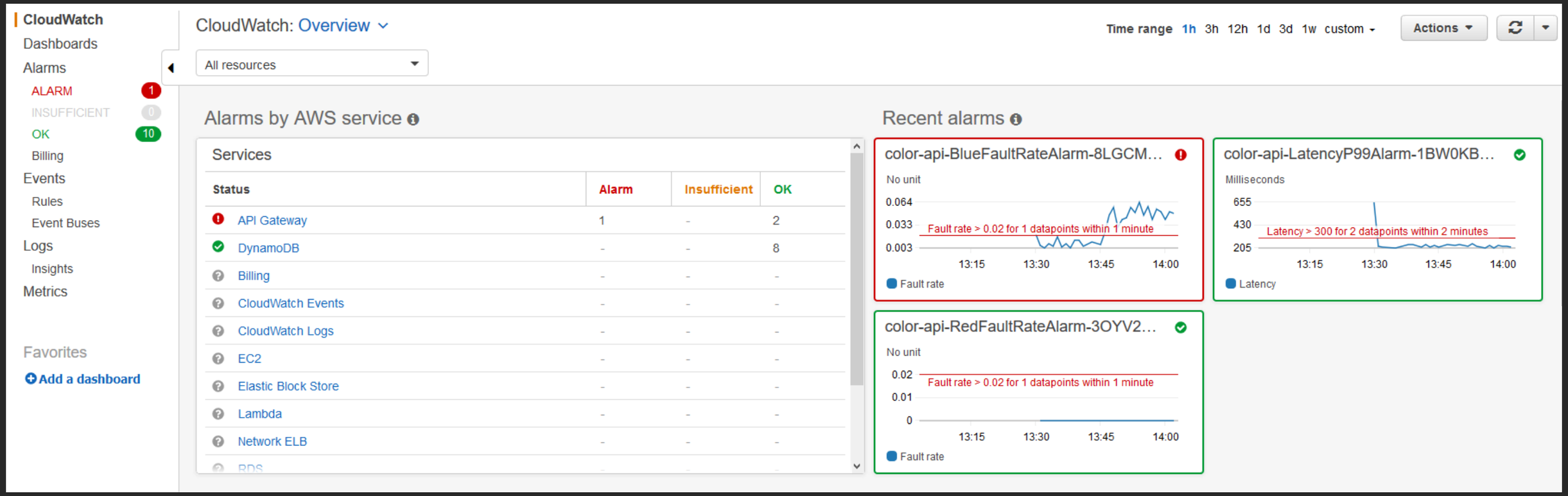
Application-specific logging and metrics

Example: Monitoring a serverless application

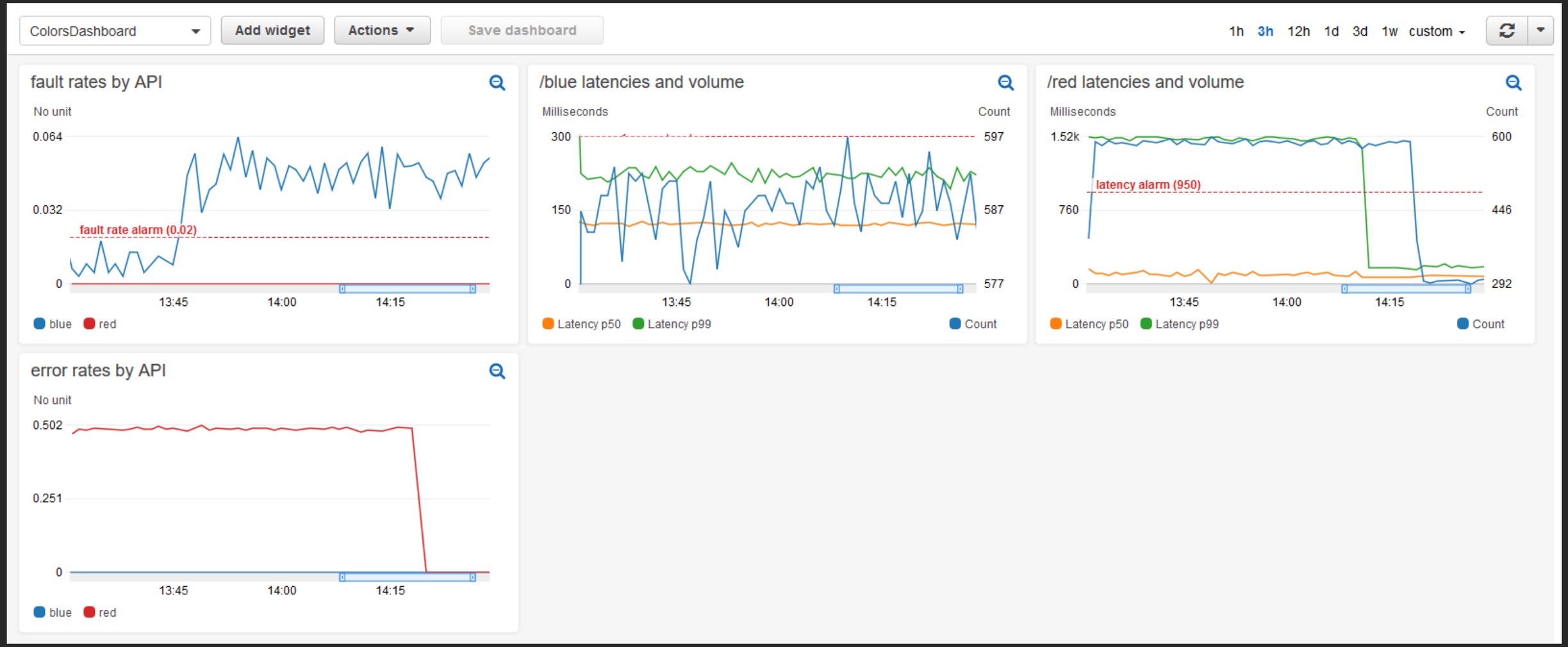
A simple serverless API



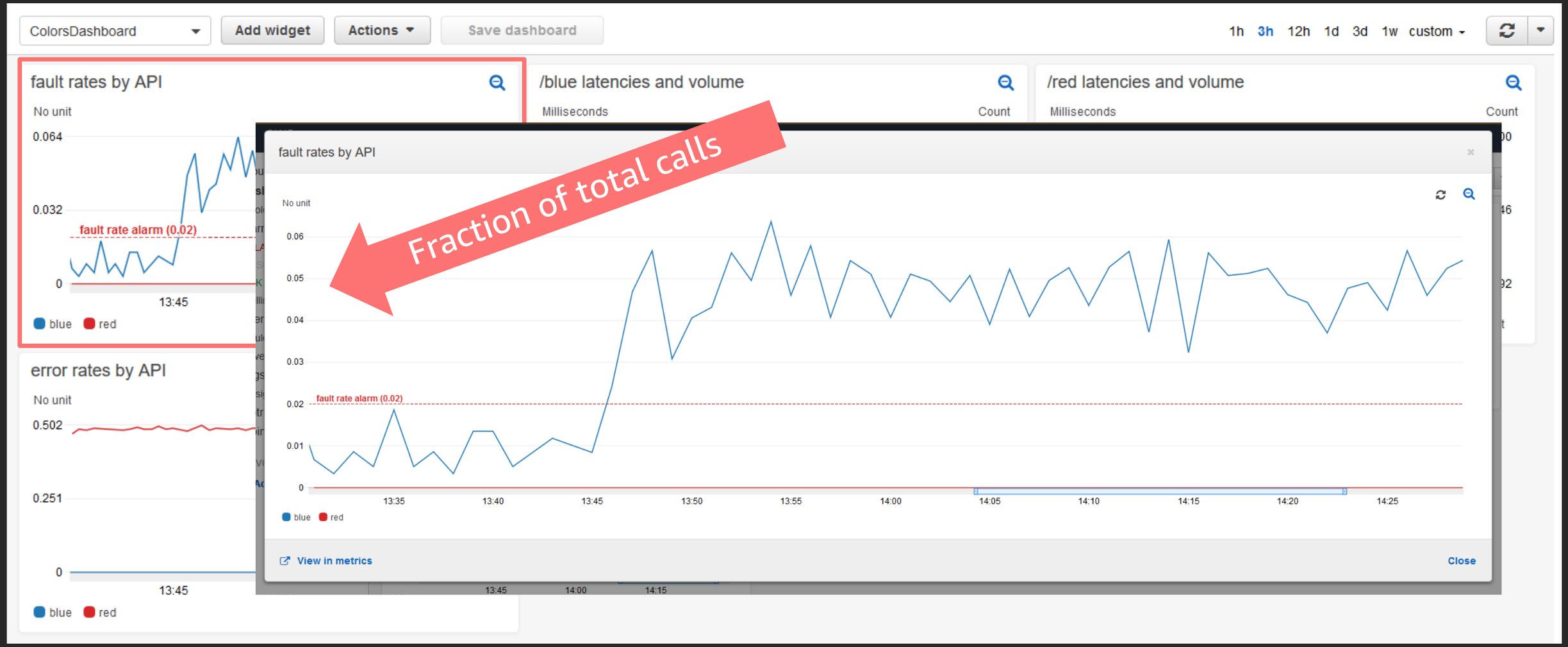
CloudWatch default view



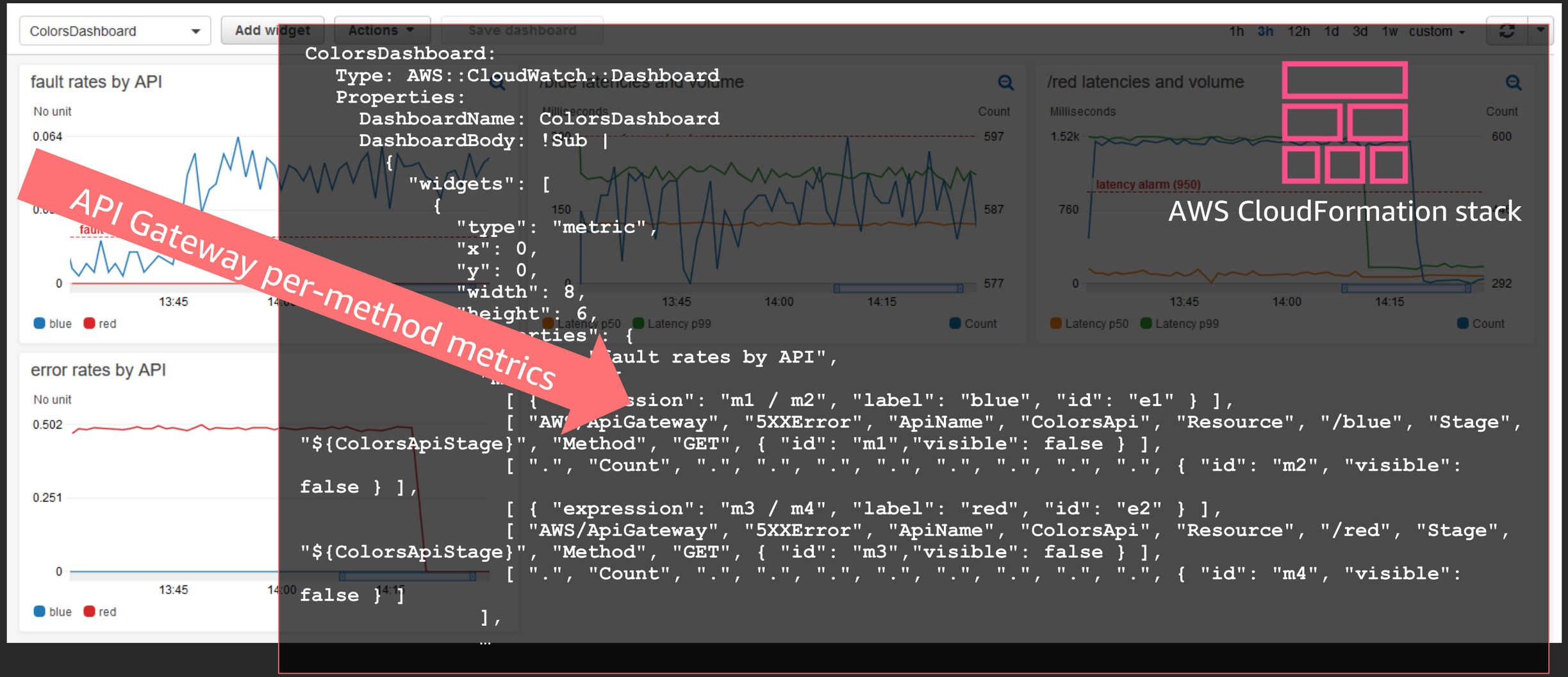
Custom dashboard



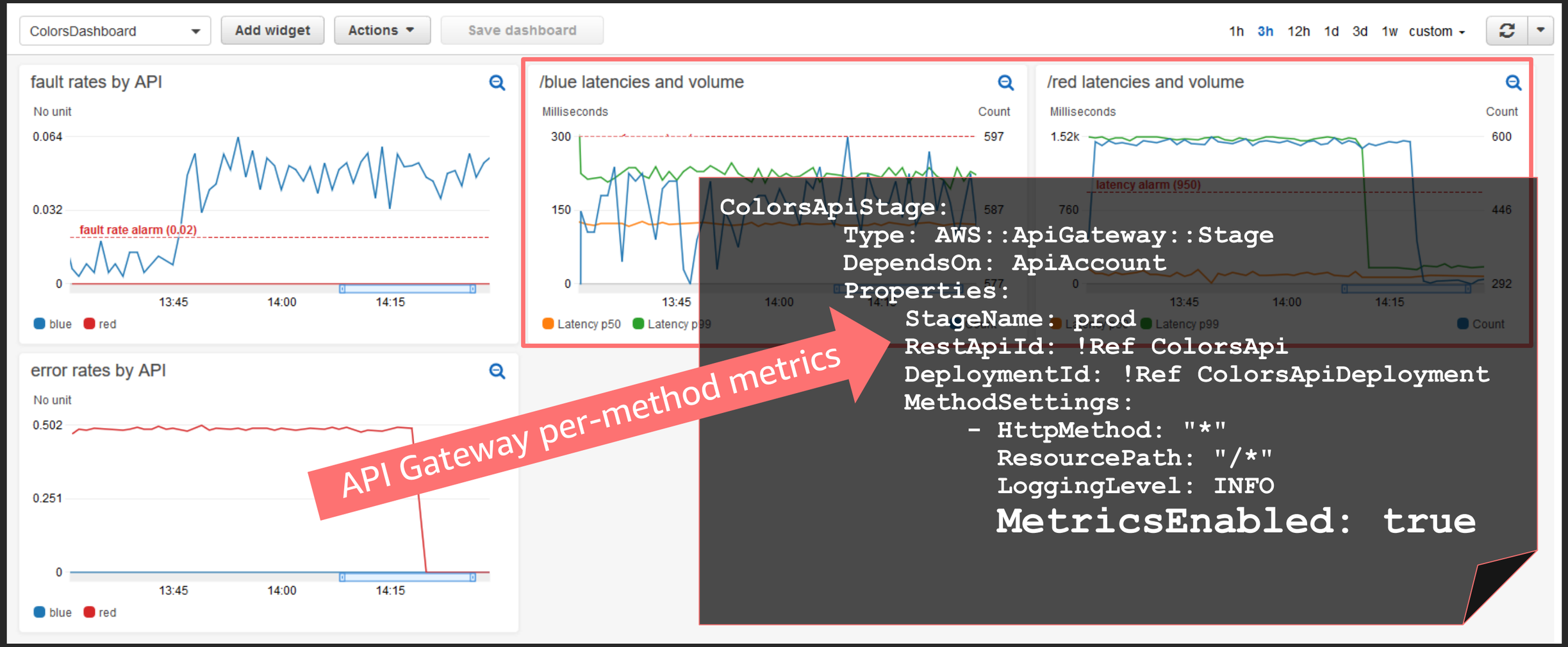
Custom dashboard: Metric math



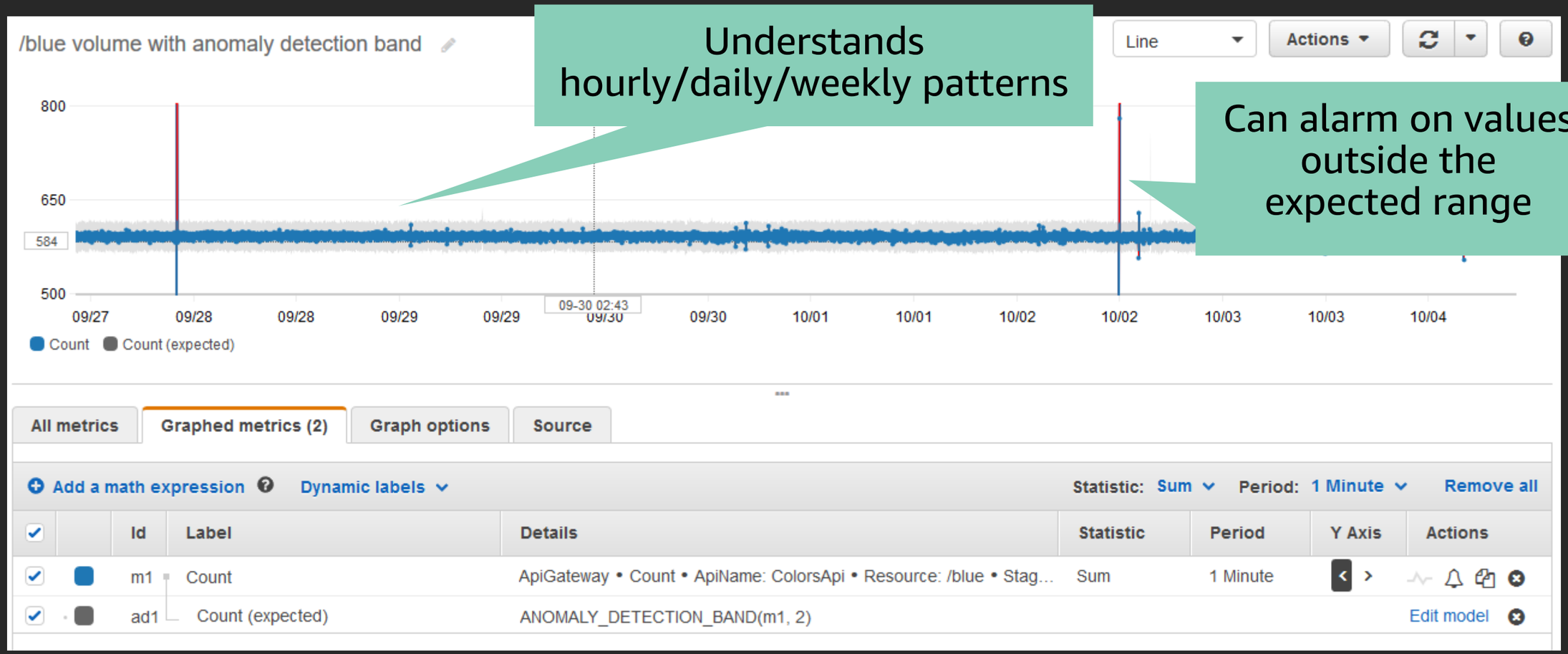
Provisioning dashboards with AWS CloudFormation

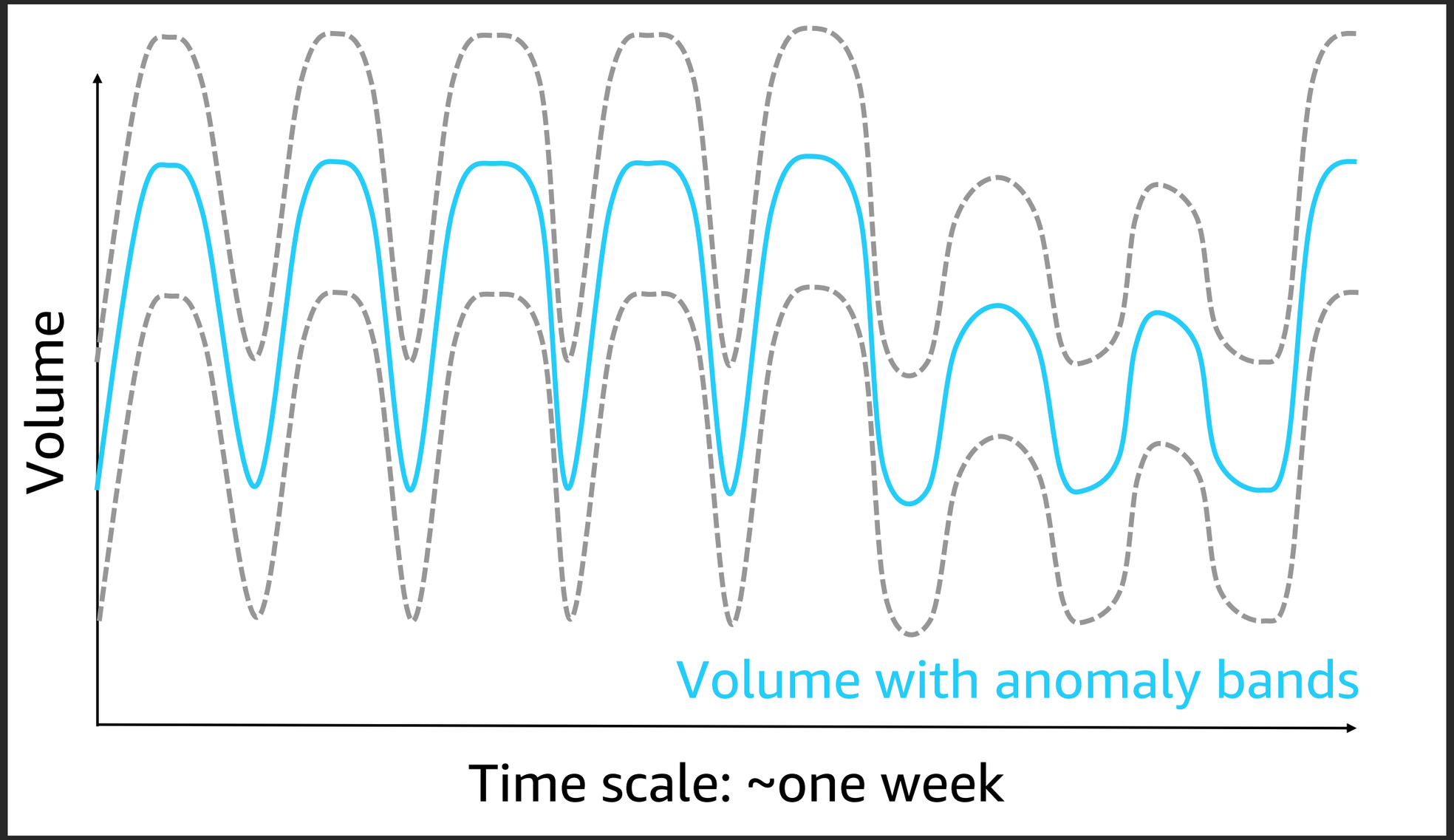


Custom dashboard: Breakdown by method

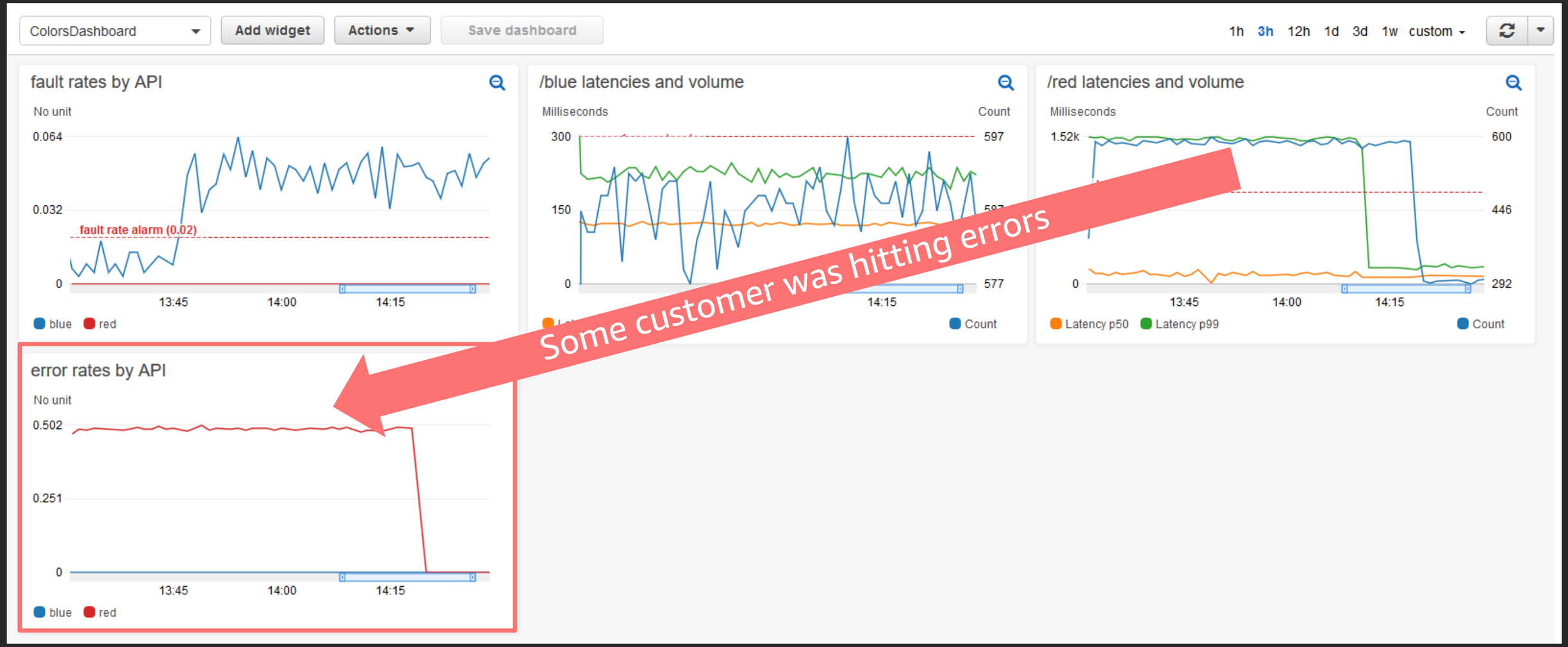


New: CloudWatch Anomaly Detection

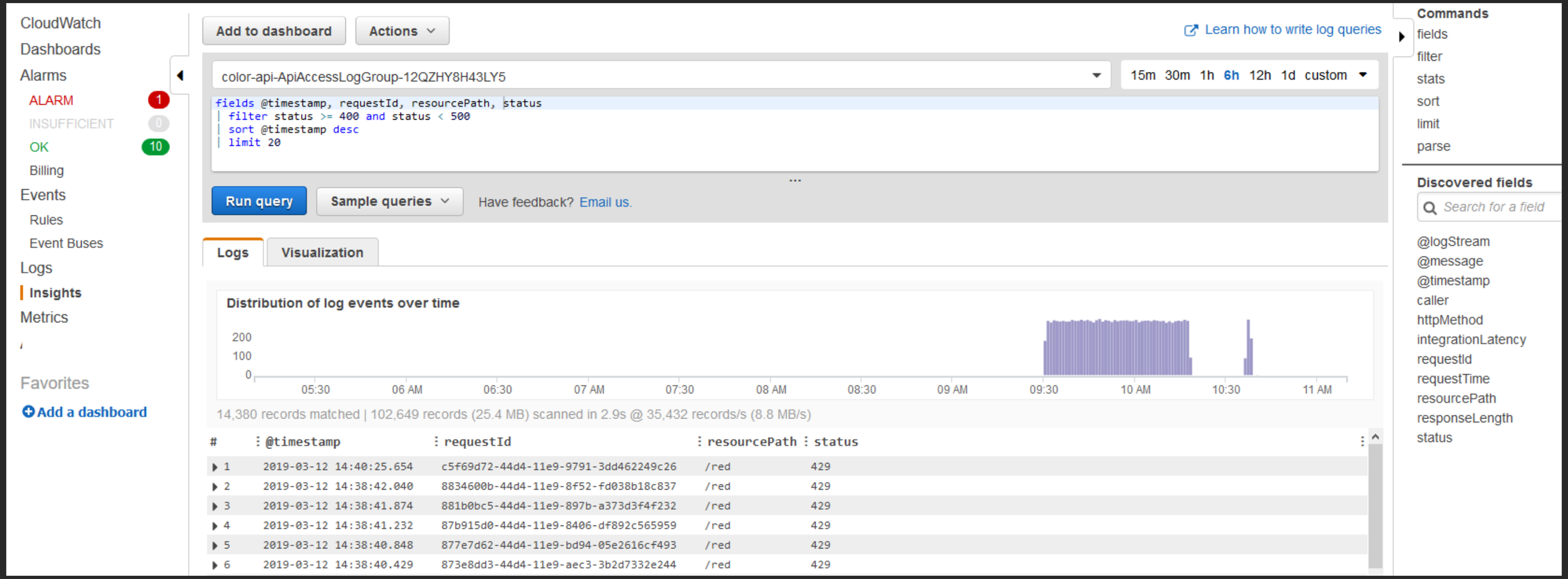



Volume
Volume with anomaly bands
Time scale: ~one week

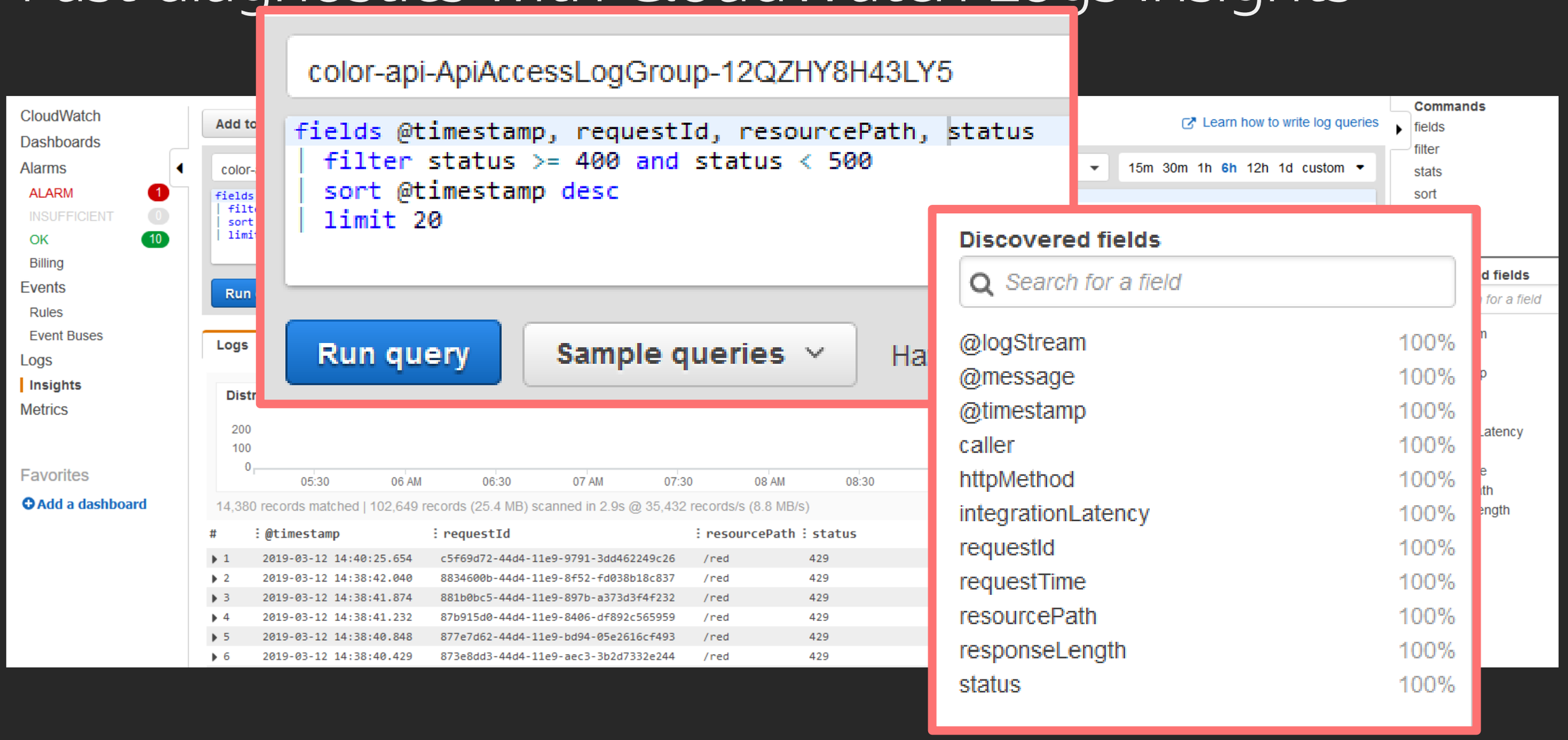
Diagnosing unexpected behavior



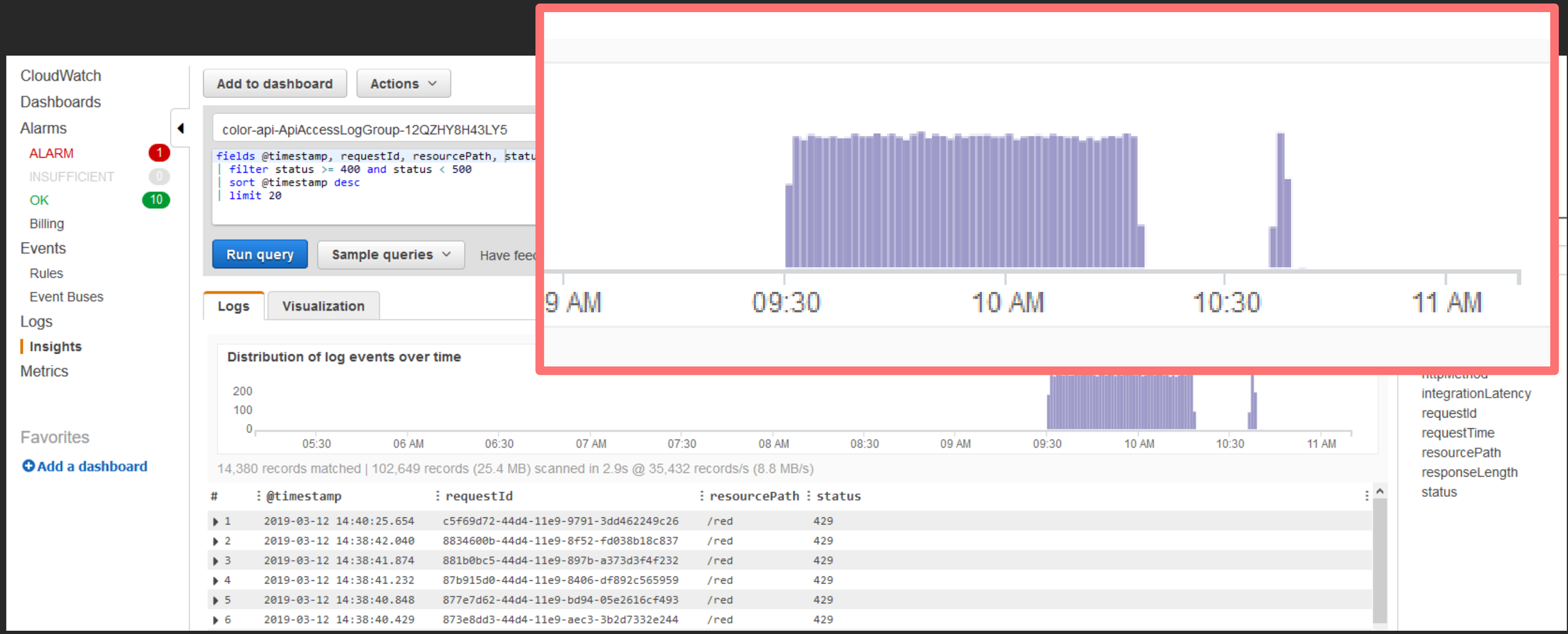
Fast diagnostics with CloudWatch Logs Insights



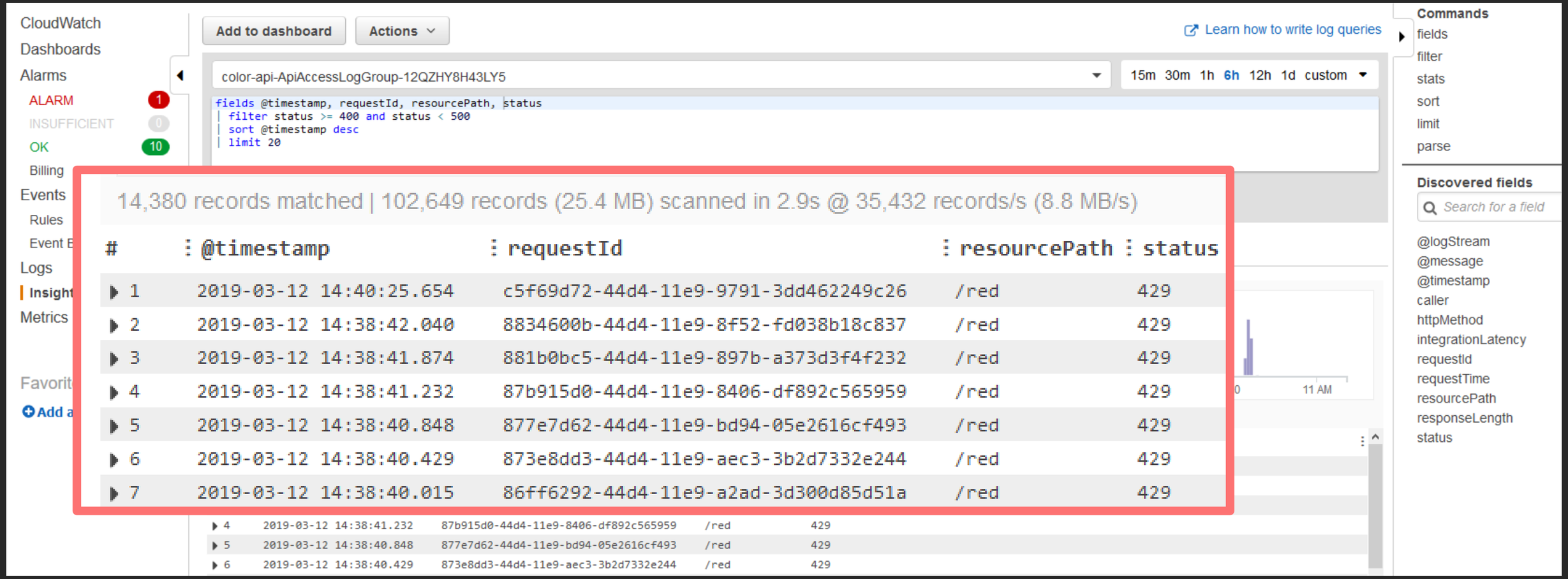
Fast diagnostics with CloudWatch Logs Insights



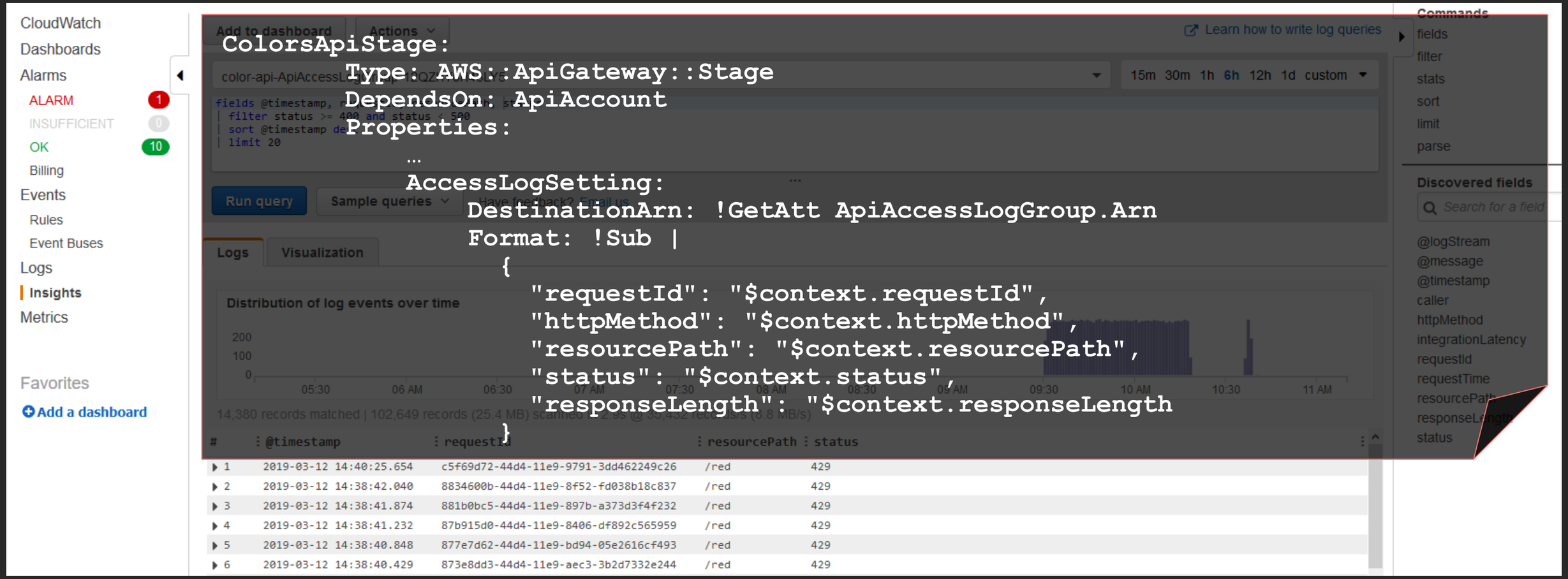
Fast diagnostics with CloudWatch Logs Insights



Fast diagnostics with CloudWatch Logs Insights



Fast diagnostics with CloudWatch Logs Insights




```
ColorsApiStage:
    Type: AWS::ApiGateway::Stage
    DependsOn: ApiAccount
    Properties:
        ...
        AccessLogSetting:
            DestinationArn: !GetAtt ApiAccessLogGroup.Arn
            Format: !Sub |
              {
                "requestId": "$context.requestId",
                "httpMethod": "$context.httpMethod",
                "resourcePath": "$context.resourcePath",
                "status": "$context.status",
                "responseLength": "$context.responseLength
              }
```

Wrap-up

Takeaways

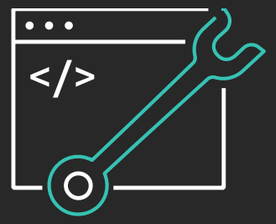
- Never waste a failure:
  Effective post-mortems

- Catch failures before your customers do:
  Effective dashboarding and metrics-reading

- Use AWS tools to gain visibility and insight into your application

Learn DevOps with AWS Training and Certification

Resources created by the experts at AWS to propel your organization and career forward



Take free digital training to learn best practices for developing, deploying, and maintaining applications

Classroom offerings, like DevOps Engineering on AWS, feature AWS expert instructors and hands-on activities

Validate expertise with the **AWS Certified DevOps Engineer - Professional** or **AWS Certified Developer - Associate** exams

Visit aws.amazon.com/training/path-developing/

aws training and certification

Thank you!

Becky Weiss

becky@amazon.com

AWS re:Invent

aws

Please complete the session survey in the mobile app.