Chaos Engineering:

Building Reliable Systems Through Experimentation

Hareesh Iyer, Solutions Architect, AWS

https://www.linkedin.com/in/hareeshiyer/

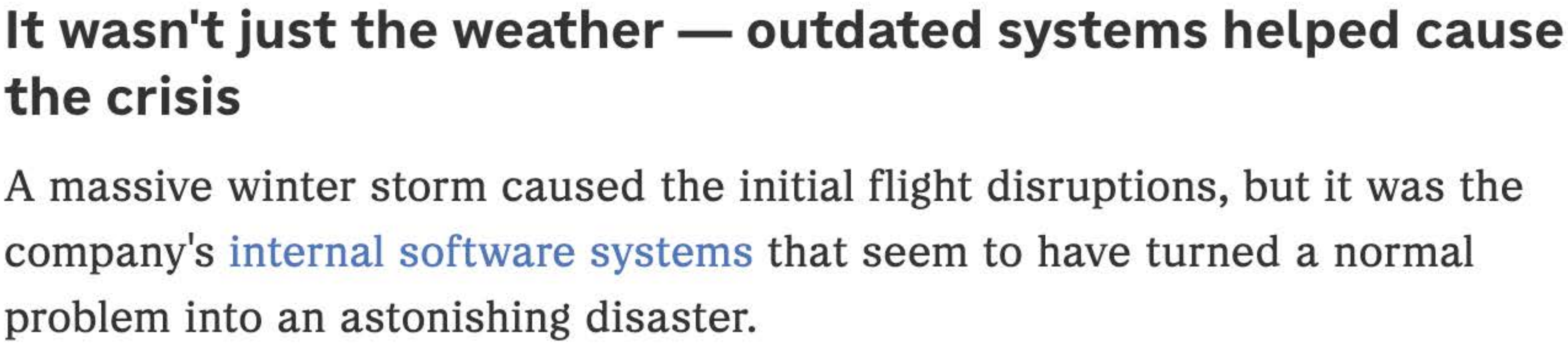
Agenda

Chaos Engineering

- *Why*
- *What*
- *How*

Why Chaos Engineering?

https://www.reviewjournal.com/business/business-columns/inside-gaming/



It wasn't just the weather — outdated systems helped cause the crisis

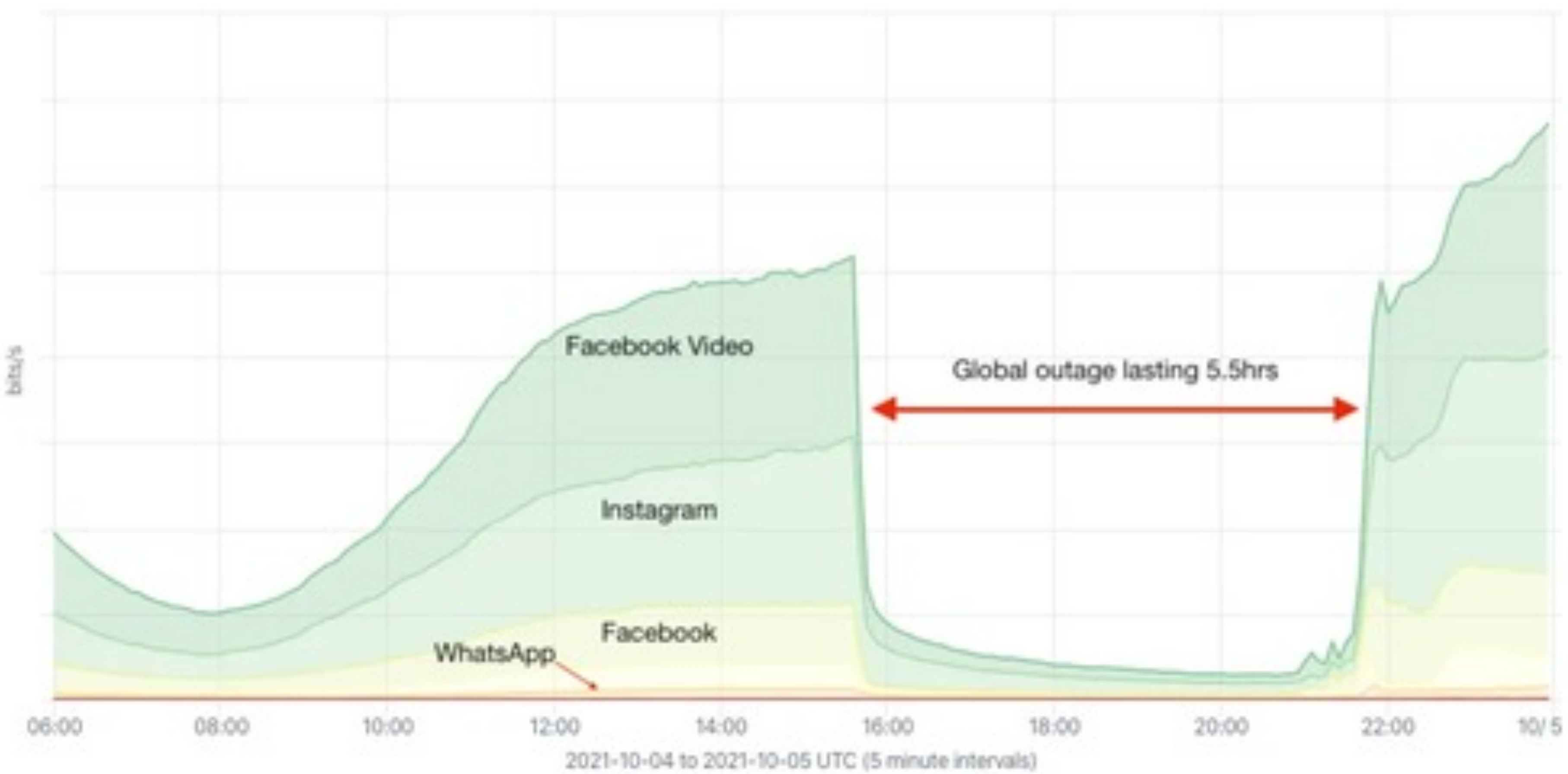
A massive winter storm caused the initial flight disruptions, but it was the company's internal software systems that seem to have turned a normal problem into an astonishing disaster.

https://www.npr.org/2022/12/30/1146377342/5-things-to-know-about-southwests-disastrous-meltdown

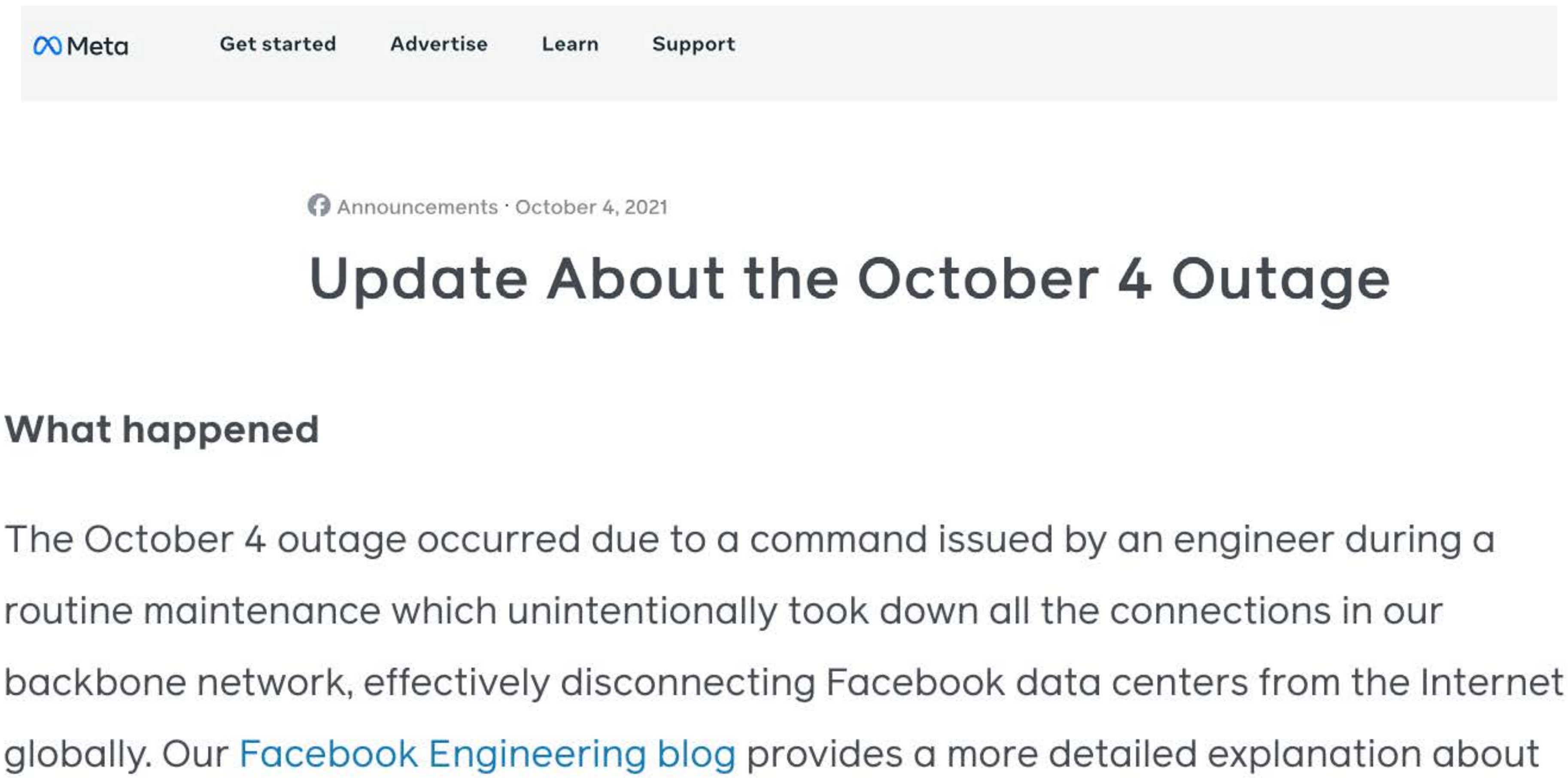
Internet Traffic served by Facebook
Global outage 4-Oct-2021

Top OTT Service by Average bits/s

Oct 04, 2021 06:00 to Oct 05, 2021 00:00 (18h)



https://en.wikipedia.org/wiki/2021_Facebook_outage



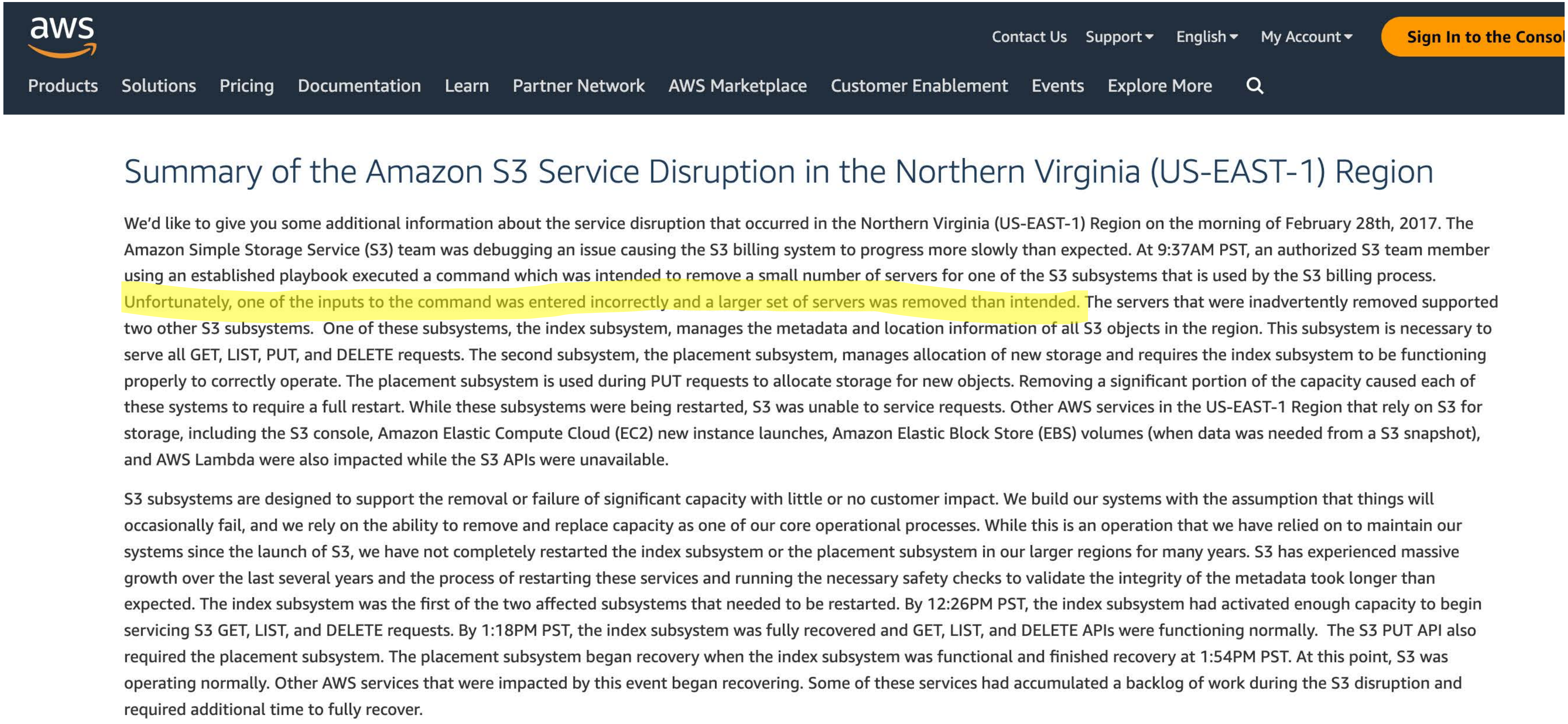
Announcements · October 4, 2021

Update About the October 4 Outage

What happened

The October 4 outage occurred due to a command issued by an engineer during a routine maintenance which unintentionally took down all the connections in our backbone network, effectively disconnecting Facebook data centers from the Internet globally. Our Facebook Engineering blog provides a more detailed explanation about

https://www.facebook.com/business/news/update-about-the-october-4th-outage

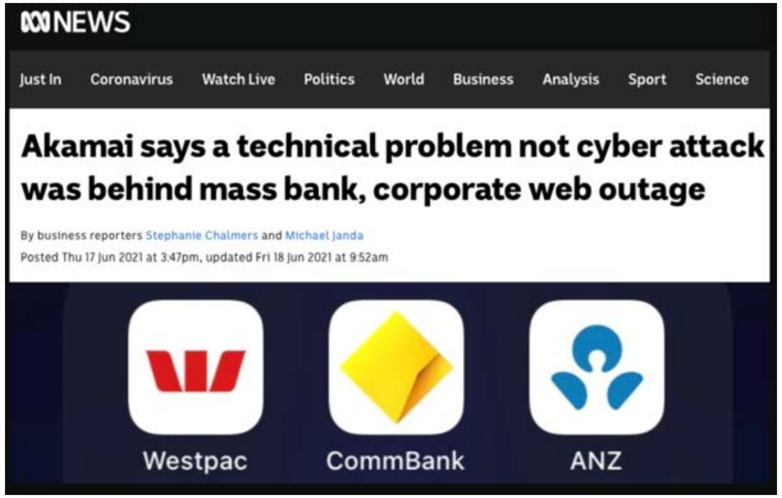


Summary of the Amazon S3 Service Disruption in the Northern Virginia (US-EAST-1) Region

We'd like to give you some additional information about the service disruption that occurred in the Northern Virginia (US-EAST-1) Region on the morning of February 28th, 2017. The Amazon Simple Storage Service (S3) team was debugging an issue causing the S3 billing system to progress more slowly than expected. At 9:37AM PST, an authorized S3 team member using an established playbook executed a command which was intended to remove a small number of servers for one of the S3 subsystems that is used by the S3 billing process. Unfortunately, one of the inputs to the command was entered incorrectly and a larger set of servers was removed than intended. The servers that were inadvertently removed supported two other S3 subsystems. One of these subsystems, the index subsystem, manages the metadata and location information of all S3 objects in the region. This subsystem is necessary to serve all GET, LIST, PUT, and DELETE requests. The second subsystem, the placement subsystem, manages allocation of new storage and requires the index subsystem to be functioning properly to correctly operate. The placement subsystem is used during PUT requests to allocate storage for new objects. Removing a significant portion of the capacity caused each of these systems to require a full restart. While these subsystems were being restarted, S3 was unable to service requests. Other AWS services in the US-EAST-1 Region that rely on S3 for storage, including the S3 console, Amazon Elastic Compute Cloud (EC2) new instance launches, Amazon Elastic Block Store (EBS) volumes (when data was needed from a S3 snapshot), and AWS Lambda were also impacted while the S3 APIs were unavailable.

S3 subsystems are designed to support the removal or failure of significant capacity with little or no customer impact. We build our systems with the assumption that things will occasionally fail, and we rely on the ability to remove and replace capacity as one of our core operational processes. While this is an operation that we have relied on to maintain our systems since the launch of S3, we have not completely restarted the index subsystem or the placement subsystem in our larger regions for many years. S3 has experienced massive growth over the last several years and the process of restarting these services and running the necessary safety checks to validate the integrity of the metadata took longer than expected. The index subsystem was the first of the two affected subsystems that needed to be restarted. By 12:26PM PST, the index subsystem had activated enough capacity to begin servicing S3 GET, LIST, and DELETE requests. By 1:18PM PST, the index subsystem was fully recovered and GET, LIST, and DELETE APIs were functioning normally. The S3 PUT API also required the placement subsystem. The placement subsystem began recovery when the index subsystem was functional and finished recovery at 1:54PM PST. At this point, S3 was operating normally. Other AWS services that were impacted by this event began recovering. Some of these services had accumulated a backlog of work during the S3 disruption and required additional time to fully recover.

https://aws.amazon.com/message/41926/



Akamai says a technical problem not cyber attack was behind mass bank, corporate web outage

By business reporters Stephanie Chalmers and Michael Janda
Posted Thu 17 Jun 2021 at 3:47pm, updated Fri 18 Jun 2021 at 9:52am

Starbucks' app is down and coffee drinkers are freaking out

BY TAYLOR SOPER on May 19, 2022 at 8:24 am

British Airways IT failure caused by 'uncontrolled return of power'

Cause of outage has not been revealed but BA says it was not due to an IT shutdown and was not linked to job outsourcing

THE OUTAGE OF CNN AND AMAZON WAS BECAUSE OF A SLEEPING BLOG POST ON FASTLY.
FASTLY PROVIDES SERVICES TO FORIEGN WEBSITES SO THEY LOAD FASTER IN THE U.S.

Business impact of resilience is bigger than ever

$1.25B to $2.5B

Annual Fortune 1,000 application downtime costs (IDC)

$474K

Average cost/hour of downtime (Ponemon Institute)

$500K to $1M

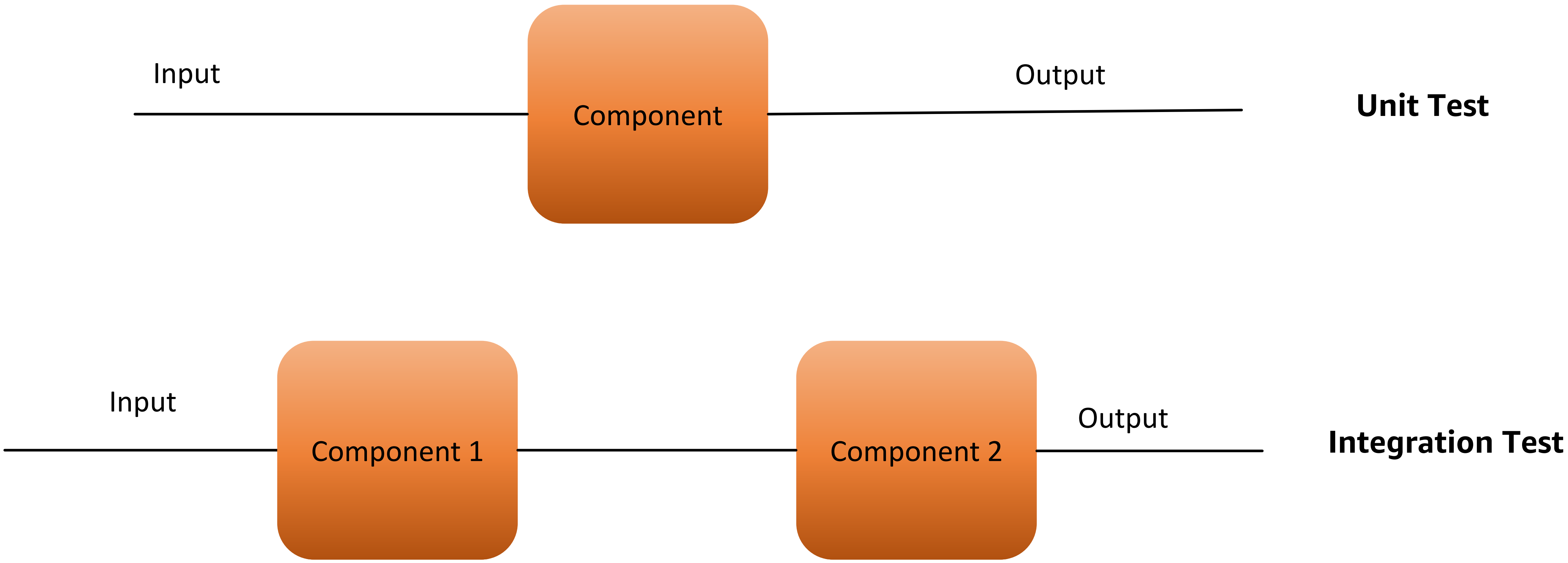
Cost/hour of a critical application failure (IDC)

$100K

Average cost/hour of an infrastructure failure (IDC)

Why are these issues not surfaced during testing?

Testing



Input

Component

Output

Unit Test

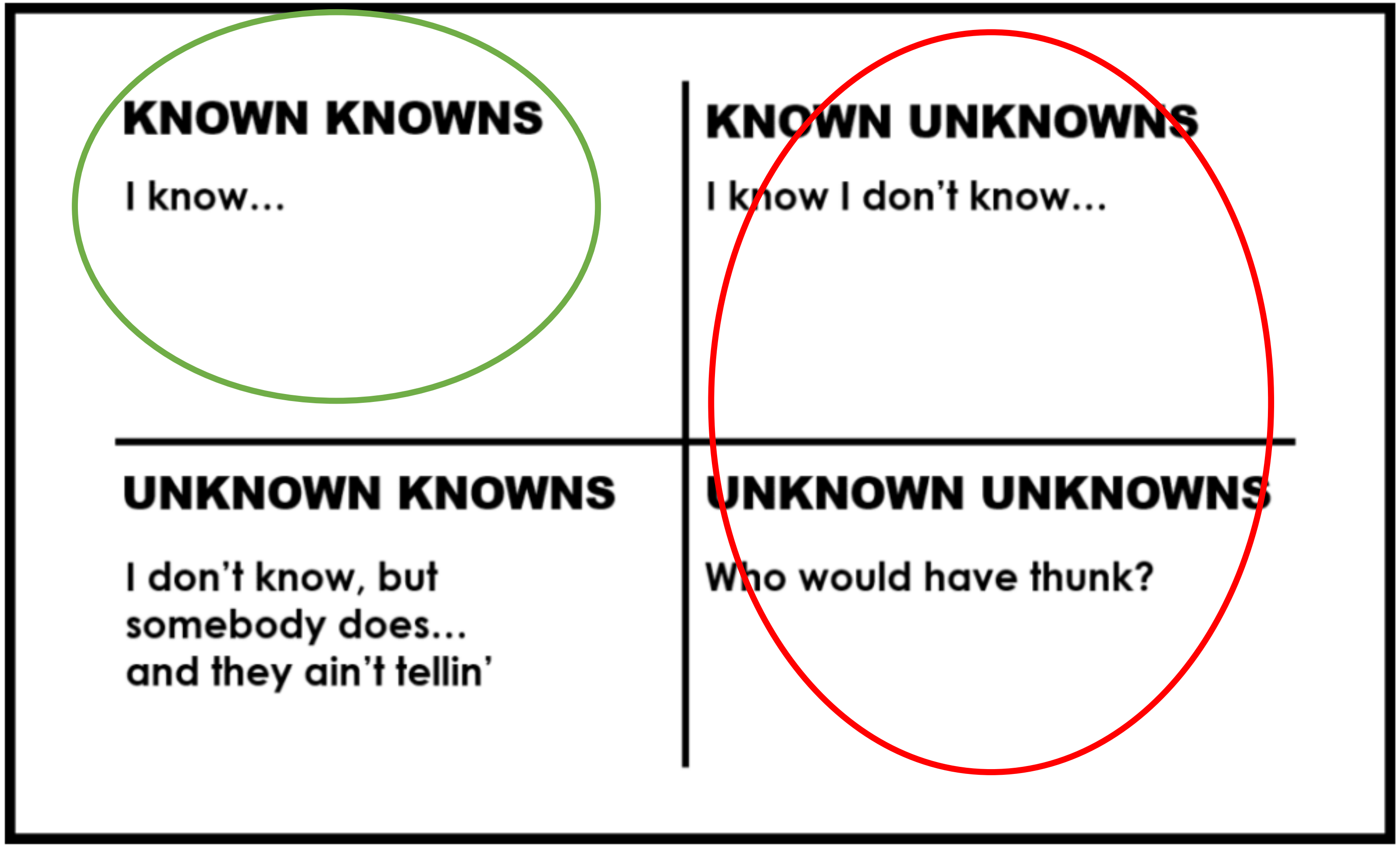
Input

Component 1

Component 2

Output

Integration Test



KNOWN KNOWNS I know…	**KNOWN UNKNOWNS** I know I don’t know…
UNKNOWN KNOWNS I don’t know, but somebody does… and they ain’t tellin’	**UNKNOWN UNKNOWNS** Who would have thunk?

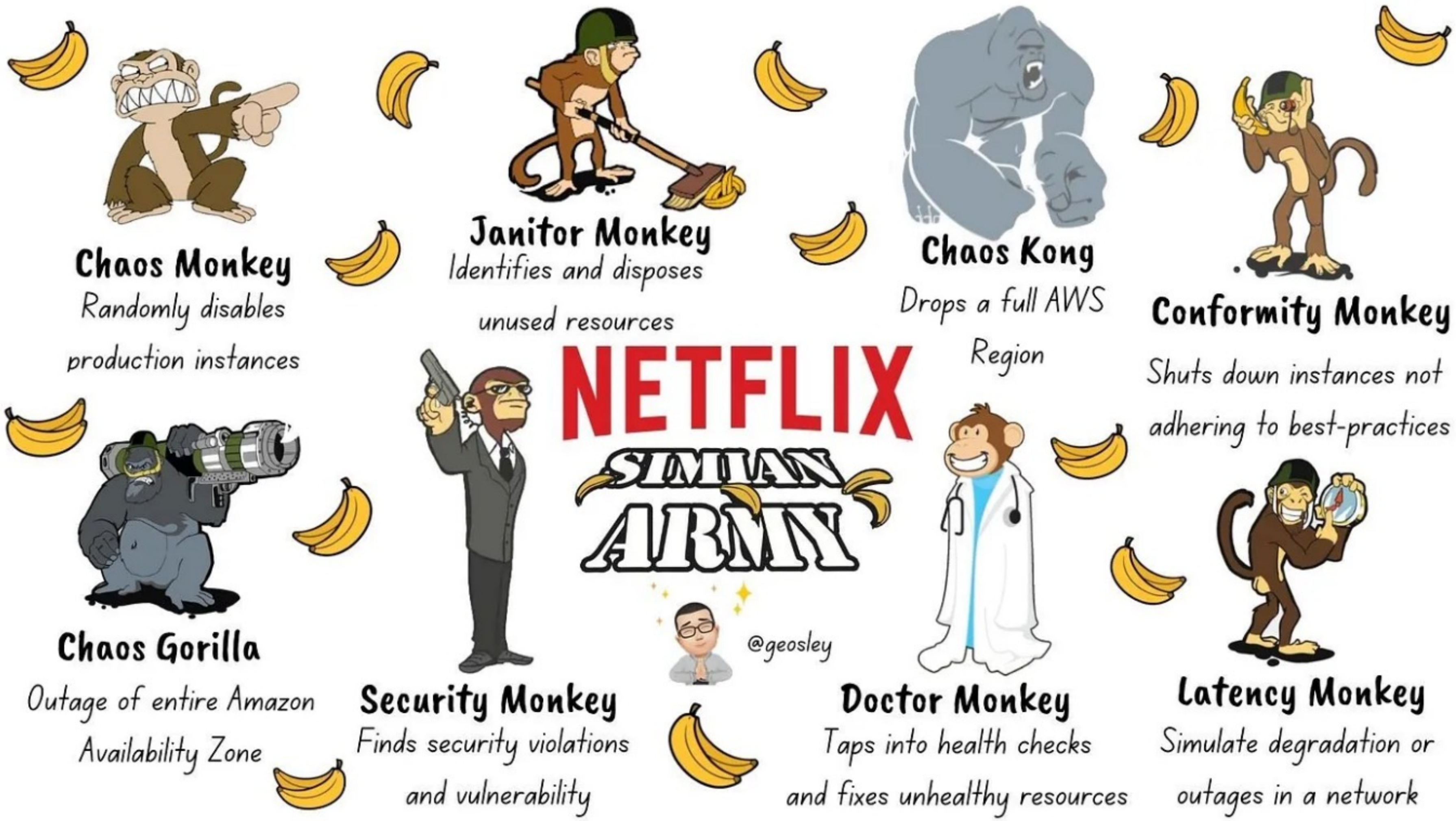
https://thepersimmongroup.com/how-to-use-the-knowns-and-unknowns-technique-to-manage-assumptions/

The solution: Chaos Engineering

What is Chaos Engineering?

A bit of history…

amazon web services
CHAOS MONKEY



Chaos Monkey
Randomly disables production instances

Janitor Monkey
Identifies and disposes unused resources

Chaos Kong
Drops a full AWS Region

Conformity Monkey
Shuts down instances not adhering to best-practices

Chaos Gorilla
Outage of entire Amazon Availability Zone

Security Monkey
Finds security violations and vulnerability

Doctor Monkey
Taps into health checks and fixes unhealthy resources

Latency Monkey
Simulate degradation or outages in a network

https://medium.com/@paulosejithu/unleashing-the-chaos-monkey-netflixs-ingenious-approach-to-building-resilient-systems-7e715b5f1b0f

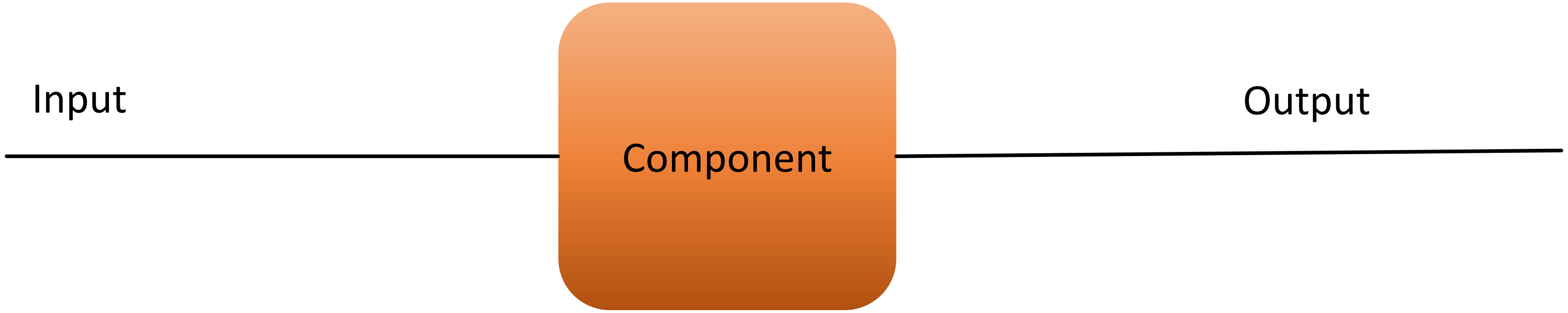
Chaos Engineering is the discipline of experimenting on a system in order to build confidence in the system’s capability to withstand turbulent conditions in production.

https://principlesofchaos.org/

Chaos Engineering is the discipline of experimenting on a system in order to build confidence in the system’s capability to withstand turbulent conditions in production.

https://principlesofchaos.org/

Testing vs Experiments

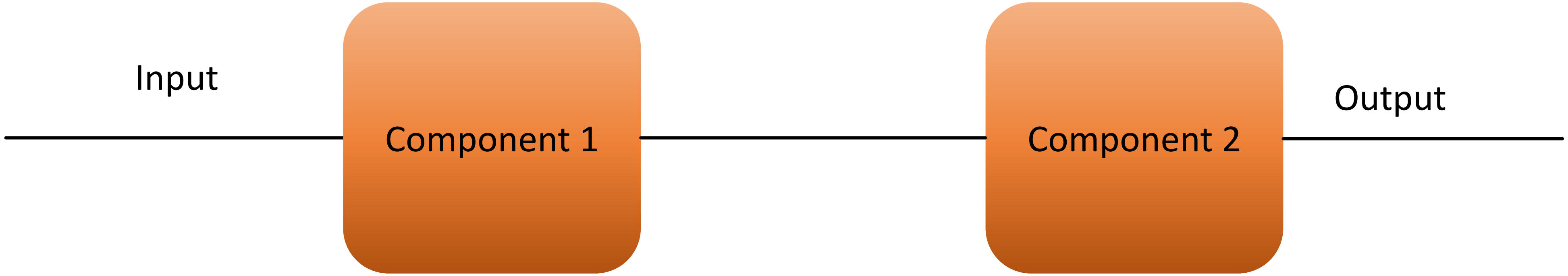


Input

Component

Output

Unit Test



Input

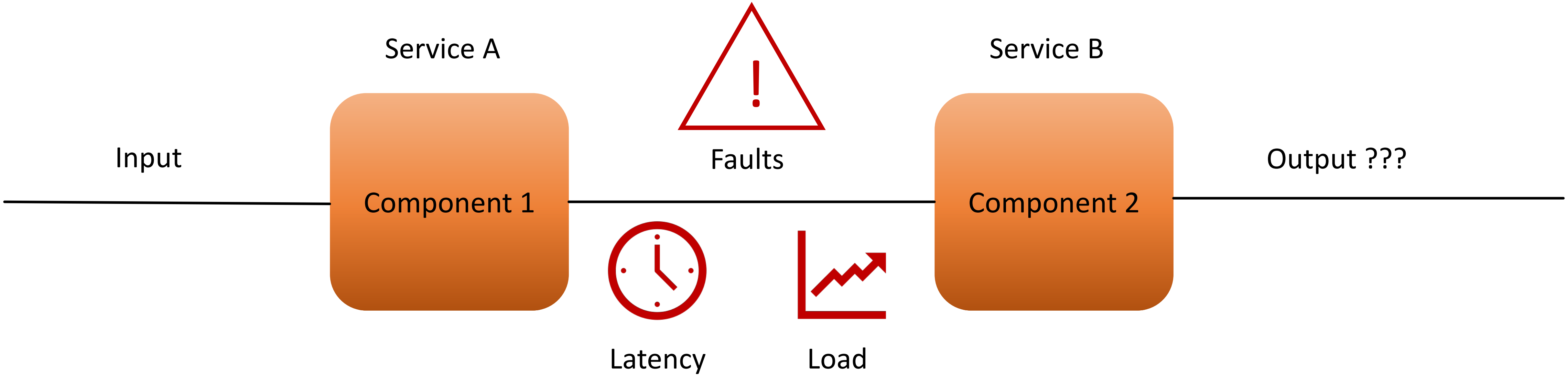
Component 1

Component 2

Output

Integration Test

Testing vs Experiments



Chaos Engineering is the discipline of experimenting on a system in order to build confidence in the system’s capability to withstand turbulent conditions in production.

https://principlesofchaos.org/

Chaos engineering is like a vaccine. Inject something harmful to build immunity

- gremlin, thoughtworks

Chaos engineering Is **Not** about creating chaos or breaking things in production

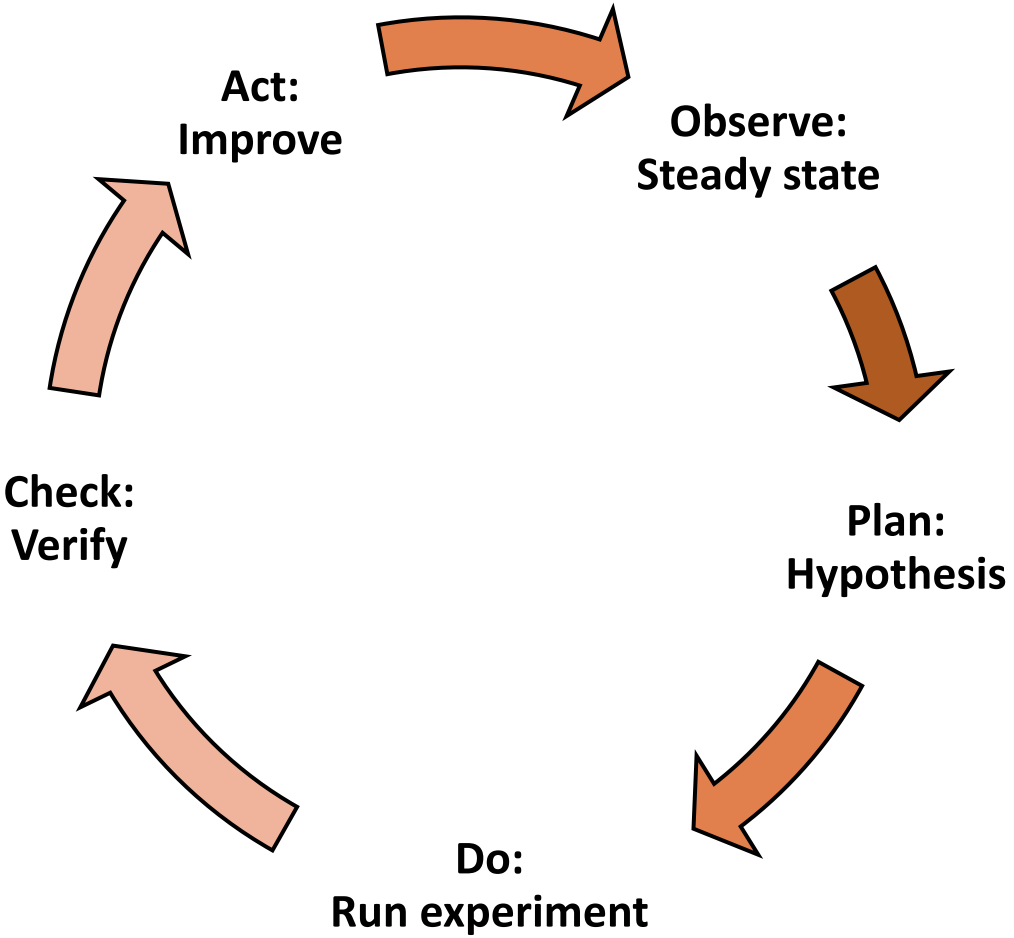
Chaos engineering Is about **uncovering** chaos inherent in the system

Chaos Engineering is the discipline of experimenting on a system in order to build confidence in the system’s capability to withstand turbulent conditions in production.

https://principlesofchaos.org/

Chaos Engineering: How To

Chaos Engineering Experiments



Act:
Improve

Observe:
Steady state

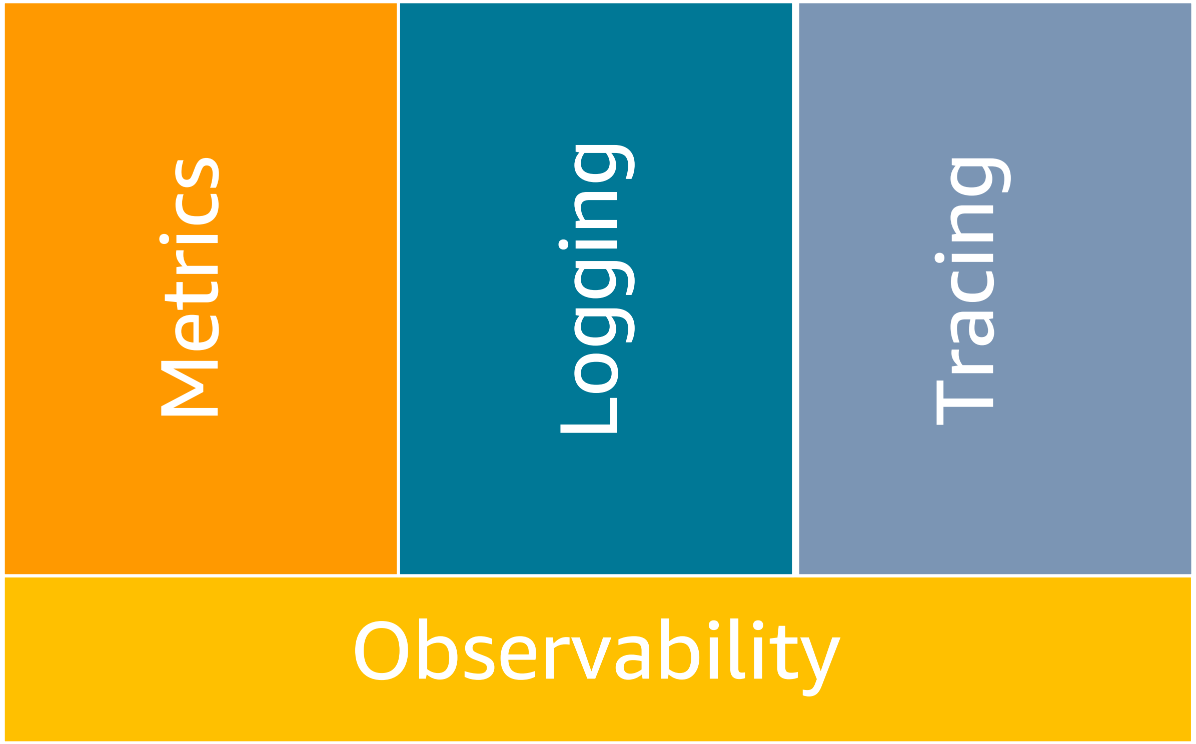
Plan:
Hypothesis

Do:
Run experiment

Check:
Verify

#1: Observe Steady State

Find the needle in the haystack



Metrics

Logging

Tracing

Observability

#2: Plan Hypothesis around the steady state

Under ______ circumstances, customers still have a good time. Availability Hypothesis

Under ______ circumstances, the security team is notified. Security Hypothesis

https://learning.oreilly.com/library/view/chaos-engineering/9781492043850/ch03.html#build_a_hypothesis_around_steady_state

#3: Run Experiments

Single points of failure

Lack of redundancy or fault tolerance

Excessive load

Not having sufficient capacity/ resources/ limits

Excessive latency

Not responding in the expected time

Misconfiguration and bugs

Incorrect execution

Shared fate

Violating intended fault isolation

#3: Run Experiments – Minimize Blast Radius

Start small	Build Confidence	Iterate
Limited scope, lower environment, Build Guardrails	Run Experiments in Production	Increase the scope

#3: Run Experiments – Continuous Resiliency

automation is an advanced mechanism to explore the solution space of potential vulnerabilities, and to reify institutional knowledge about vulnerabilities by verifying a hypothesis over time knowing that complex systems will change.

https://learning.oreilly.com/library/view/chaos-engineering/9781492043850/ch03.html#automate_experiments_to_run_continuousl

#4: Verify and Act

- Findings through chaos engineering experiments should be prioritized based on the level of impact they may cause
- Findings that involve the resilience or security of your workload should have priority over new features, as if not addressed timely, can impact your customers
- Find an executive sponsor that can help you address the priority if needed

Chaos Engineering: Tools

AWS Fault Injection Service - https://aws.amazon.com/fis/

Gremlin - https://www.gremlin.com/

Azure Chaos Studio - https://azure.microsoft.com/en-us/products/chaos-studio

Litmus - https://litmuschaos.io/

Thank you!