ANDREY FALKO - March 2019

Fault Tree Analysis Applied to Apache Kafka®



Agenda

The Challenge: Quantify Kafka Reliability

Introduction to Fault Tree Analysis

Kafka Fault Trees

- **Availability**
- **Data Durability**

Conclusion

The Challenge: Quantify Kafka Reliability

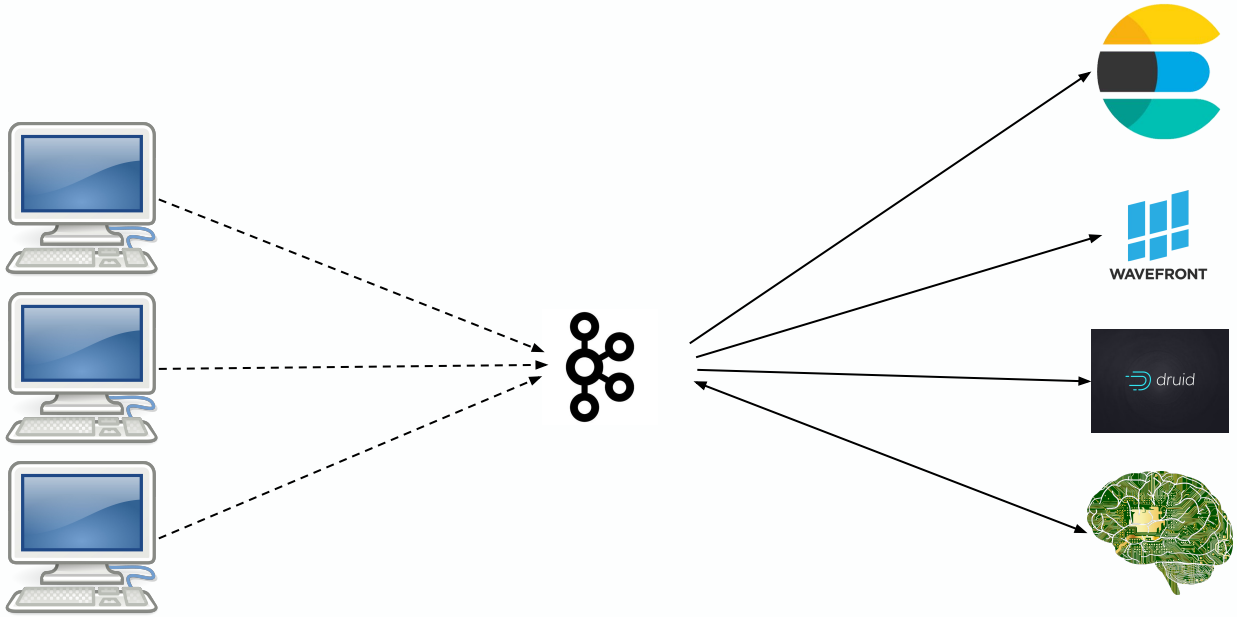
What are we trying to do?

Kafka is a “reliability tool”

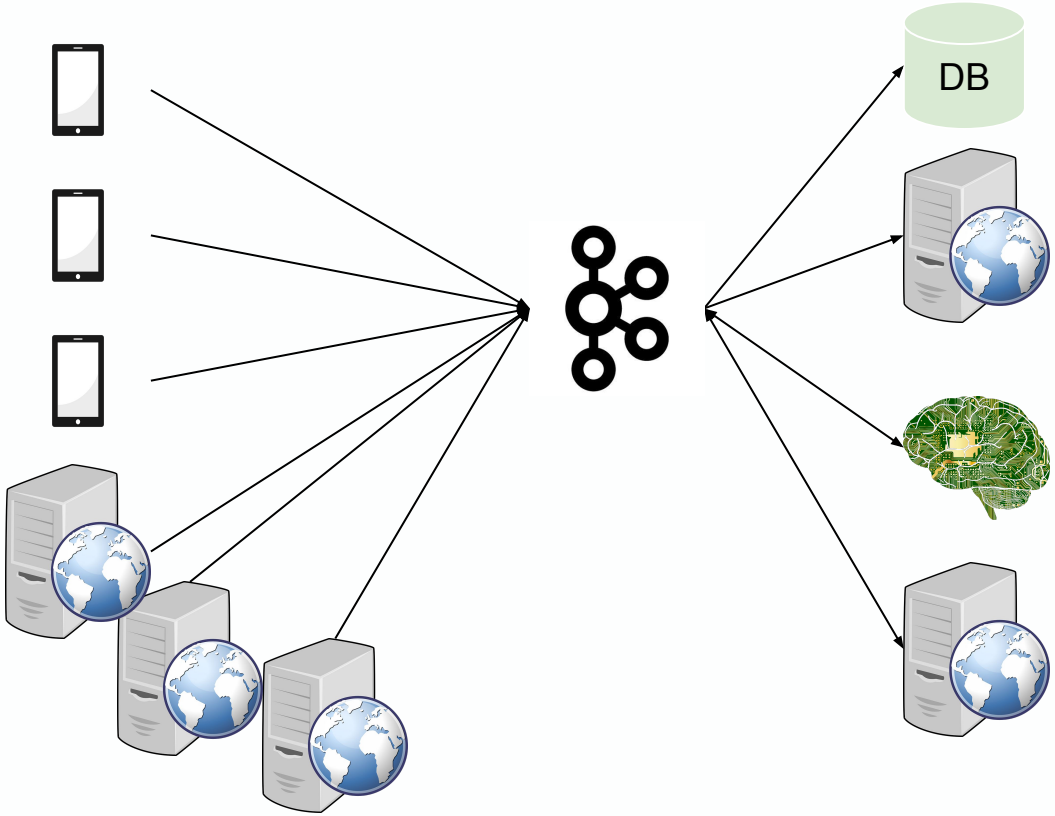
Move data without lossiness

High stakes usage

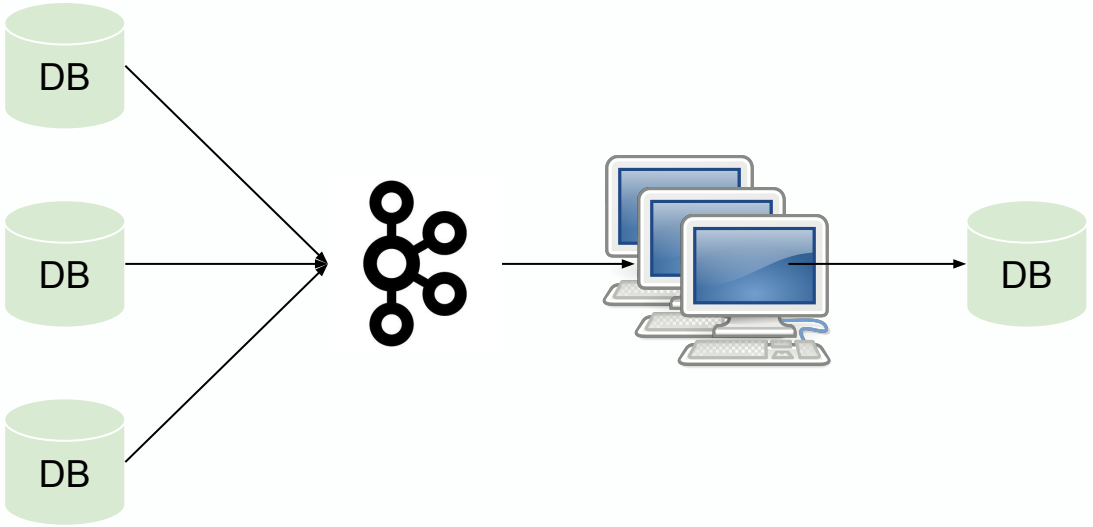
Observability Data



Event Streaming



Change Data Capture



Why Quantify?

Determine probability of success

Find opportunities to trim cost

Defining SLOs

Need to define Service Level Objectives

Availability

Durability

Latency

Quantifying SLOs

Availability

What is the probability that writes or reads fail?

How long do we tolerate downtime?

Quantifying SLOs

Durability

What is the probability that we'll lose data?

How much will we lose?

Quantifying SLOs

Latency

How long are transactions allowed to take?

Introduction to Fault Tree Analysis

What is Fault Tree Analysis?

Deductive Failure Analysis

Invented in 1962 for Minuteman I ICBM Launch Control System

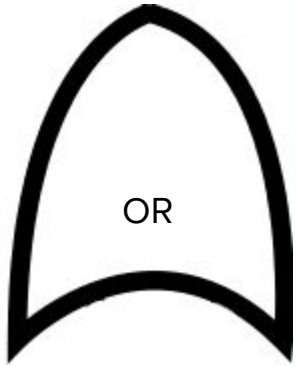
Industry wide adoption

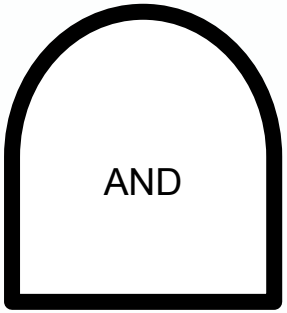
- **Aerospace**
- **Military**
- **Petrochemical**
- **Et al.**

Fault Tree Analysis: Event Symbols

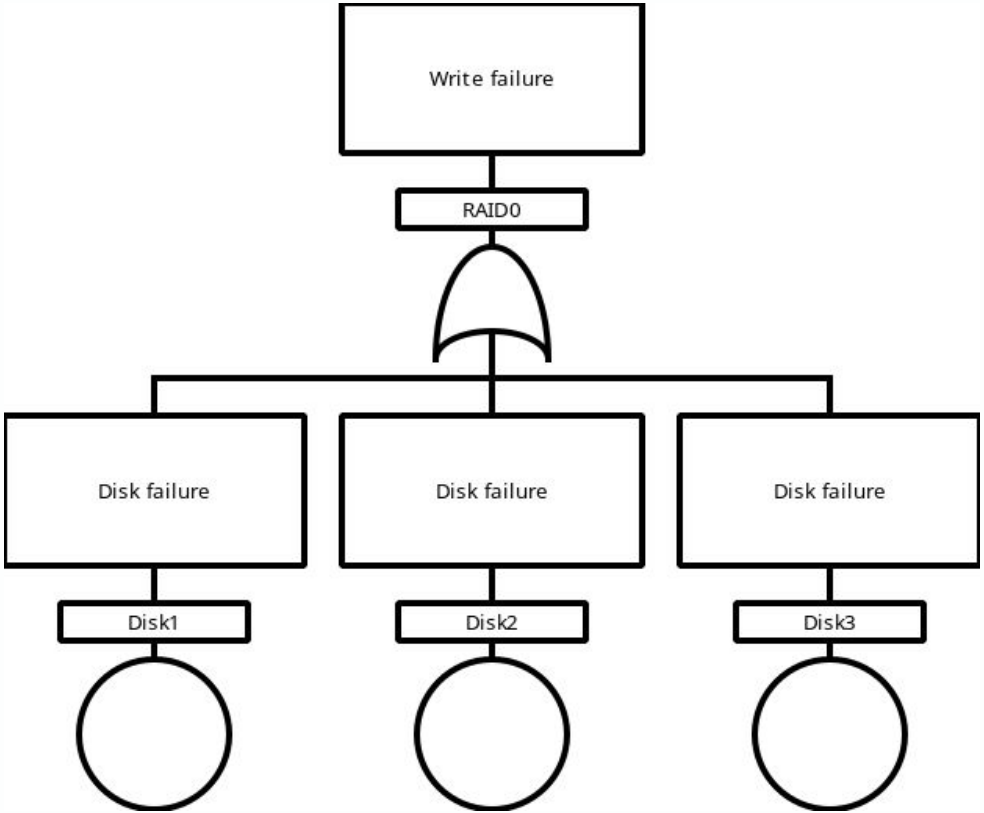


Fault Tree Analysis: Gate Symbols



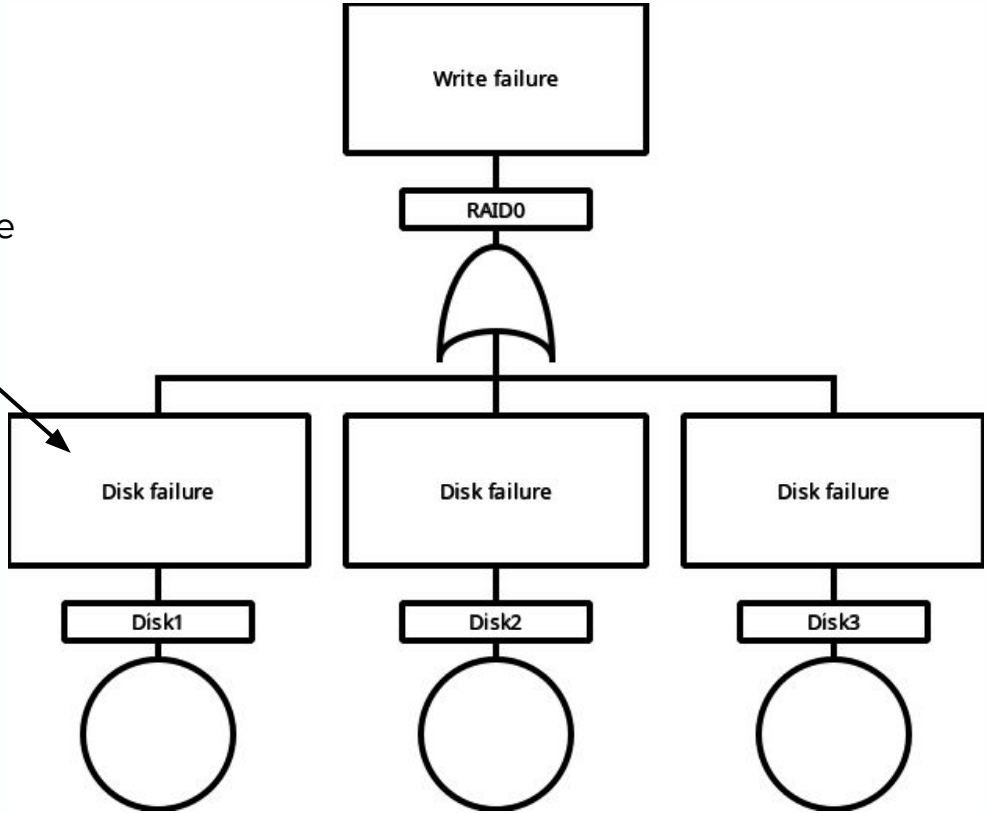


Fault Tree Analysis: OR Example



Fault Tree Analysis: OR Example

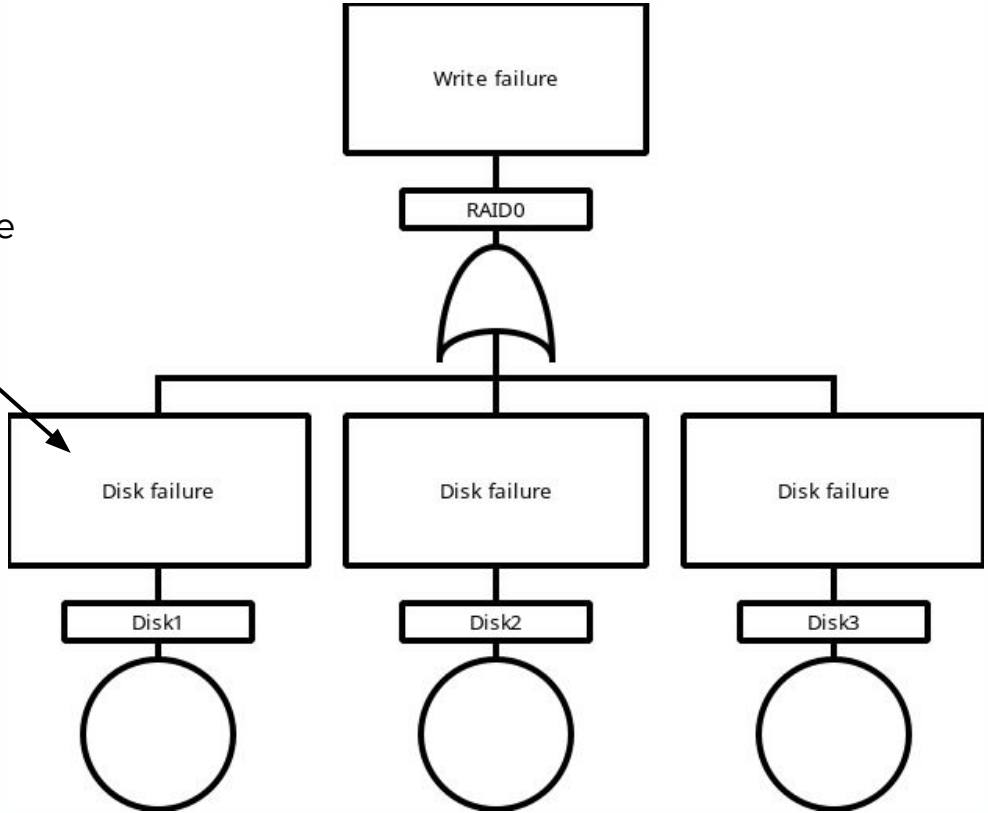
4% probability of failure



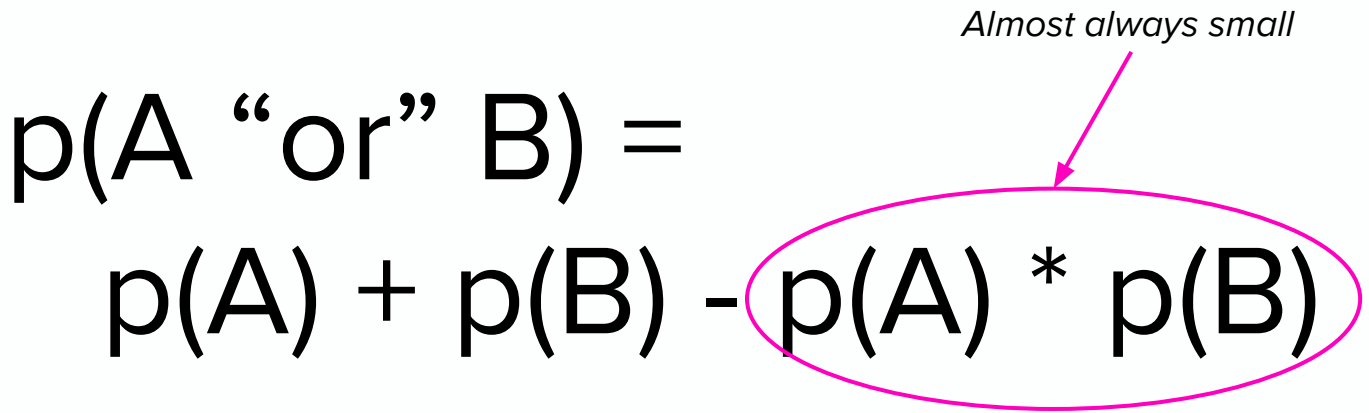
Fault Tree Analysis: OR Example



4% probability of failure annualized

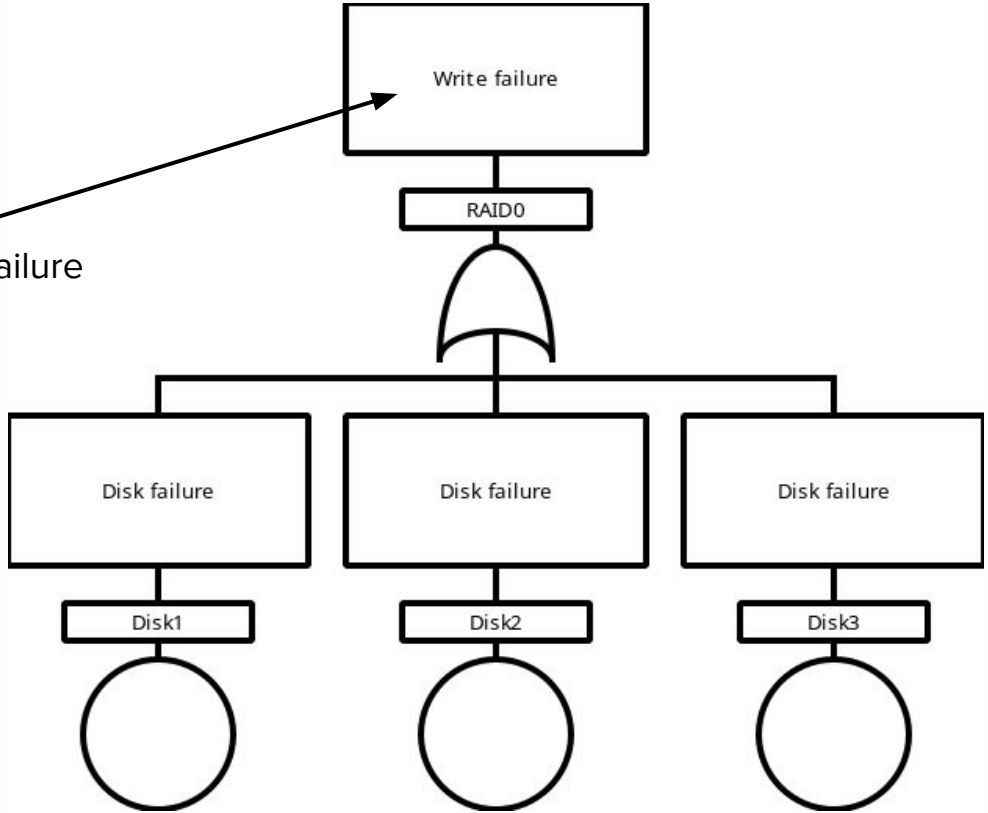


Fault Tree Analysis: OR Example

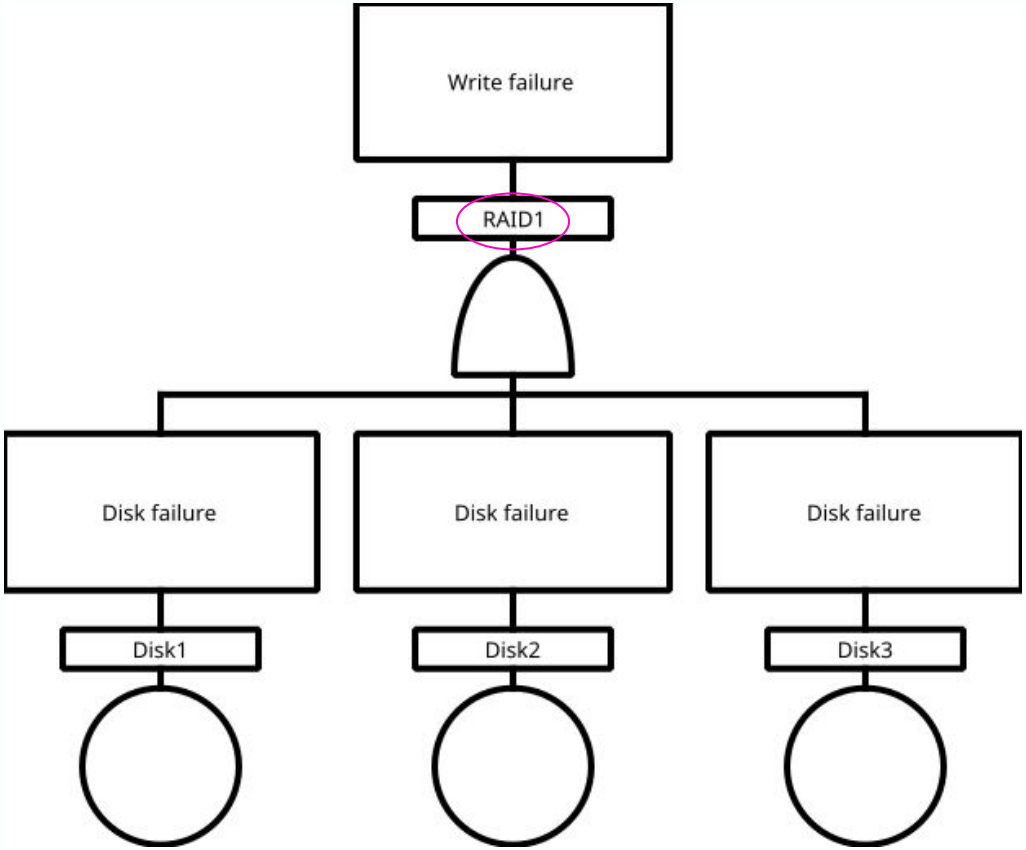


Fault Tree Analysis: OR Example

12% chance of failure annualized



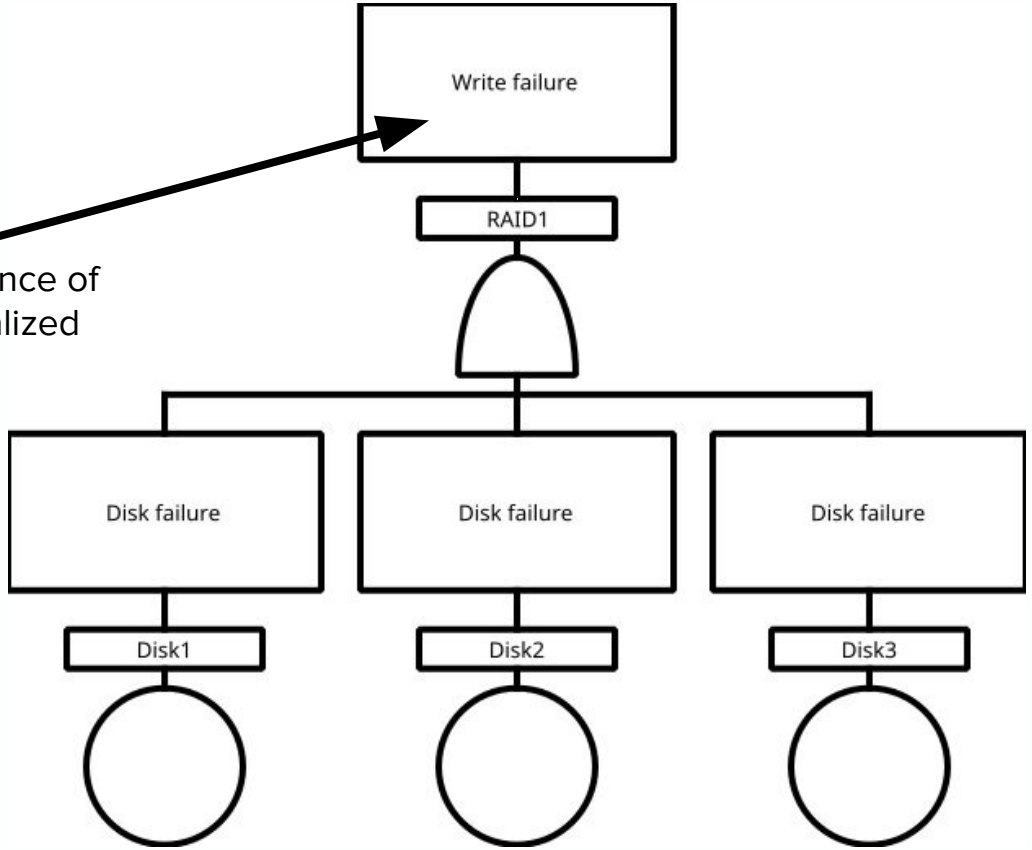
Fault Tree Analysis: AND Example



Fault Tree Analysis: AND Example

$$p(A \text{ and } B) = p(A) * p(B)$$

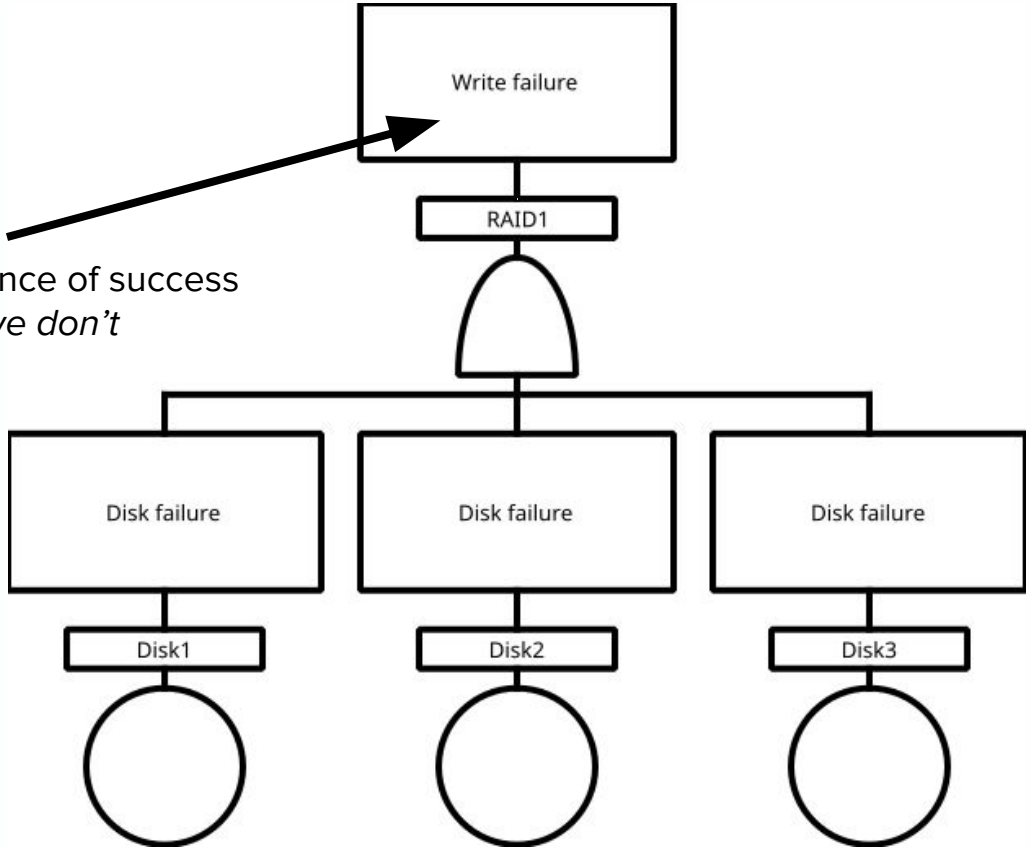
Fault Tree Analysis: AND Example



99.9936% chance of success annualized

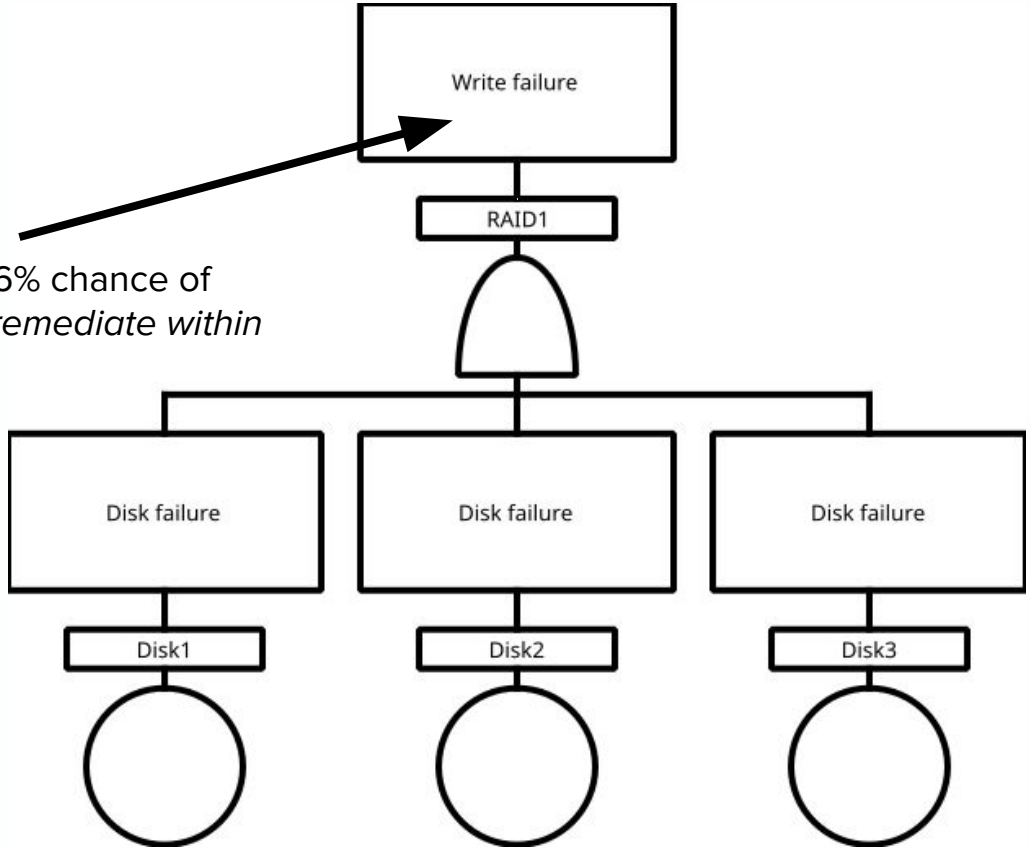
Fault Tree Analysis: AND Example

99.9936% chance of success annualized *if we don't remediate*



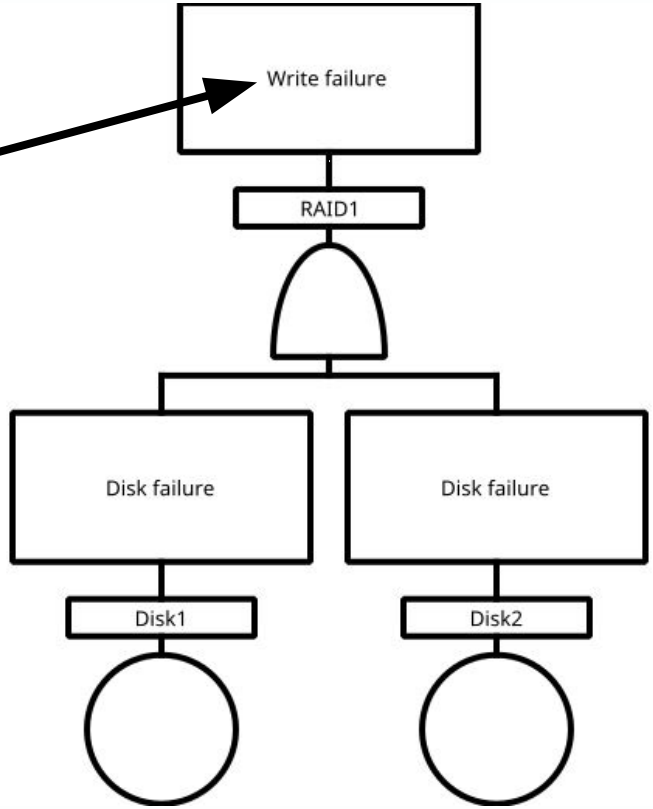
Fault Tree Analysis: AND Example

99.999999996% chance of success *if we remediate within 3 days*



Fault Tree Analysis: AND Example

99.99999% chance of success *if we remediate within 3 days*



Fault Tree Analysis: AND Example

$$p(A \text{ and } B) =$$
$$p(A) * p(B) =$$
$$(1 - e^{-p(A)*t}) * (1 - e^{-p(B)*t})$$

Where t = time to remediate

If p(A) and p(B) < .1, approximate to $\mathbf{p(A)*p(B)*t^2}$

Kafka Fault Trees

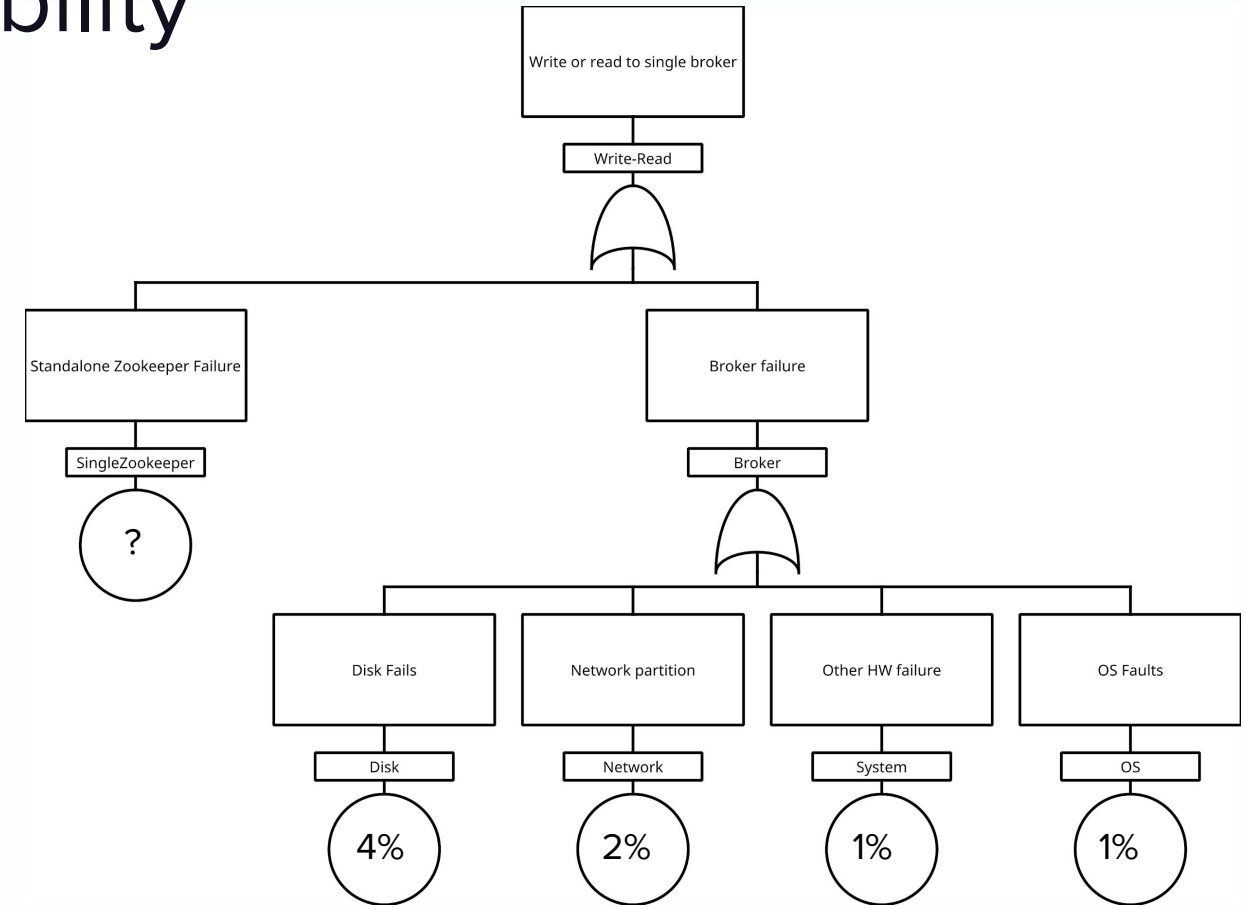
Availability

Can we write or read to a Kafka cluster?

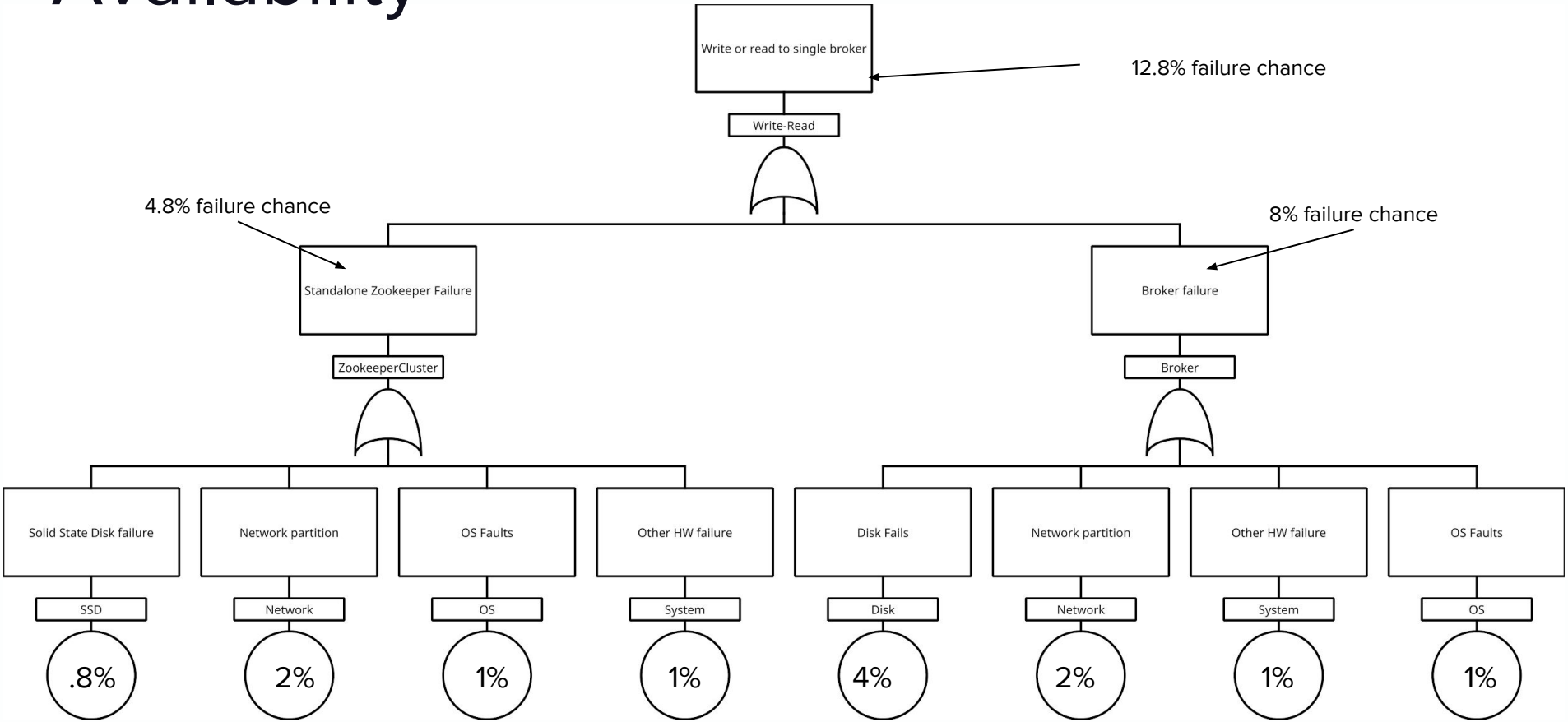
Service Level Objective (SLO):

99.99% success rate per year

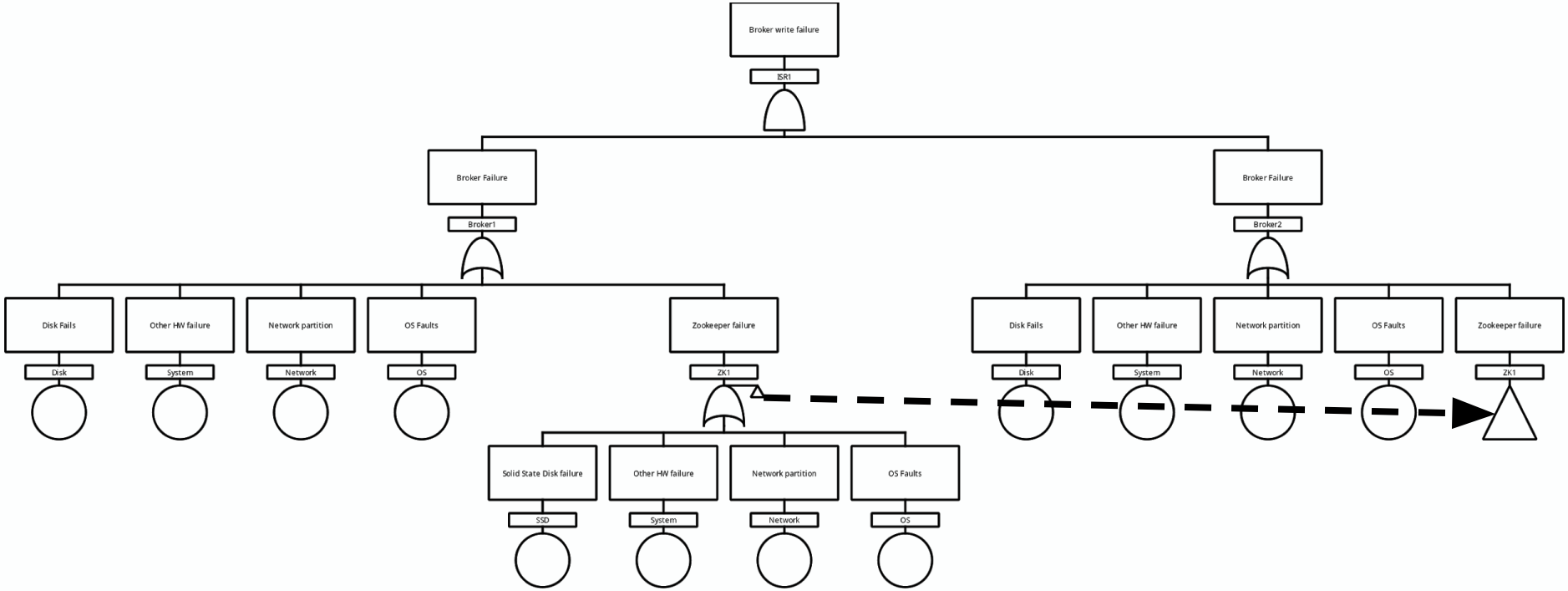
Availability



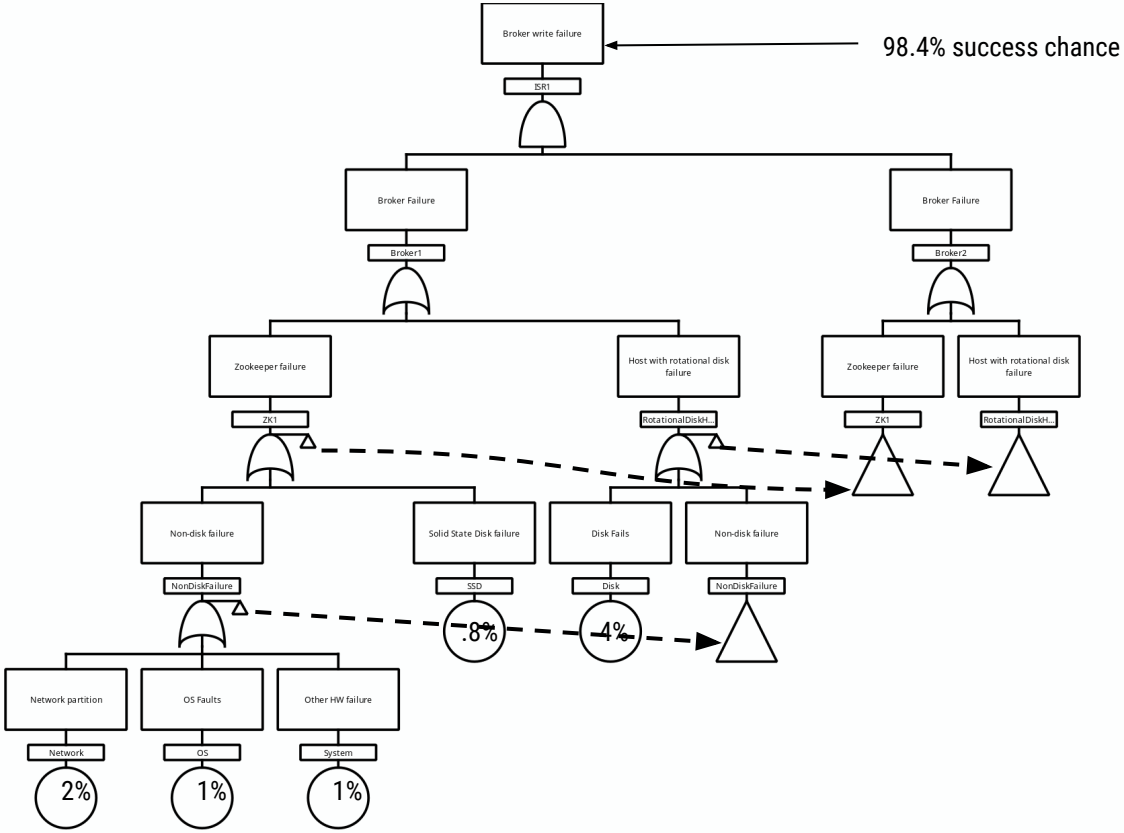
Availability



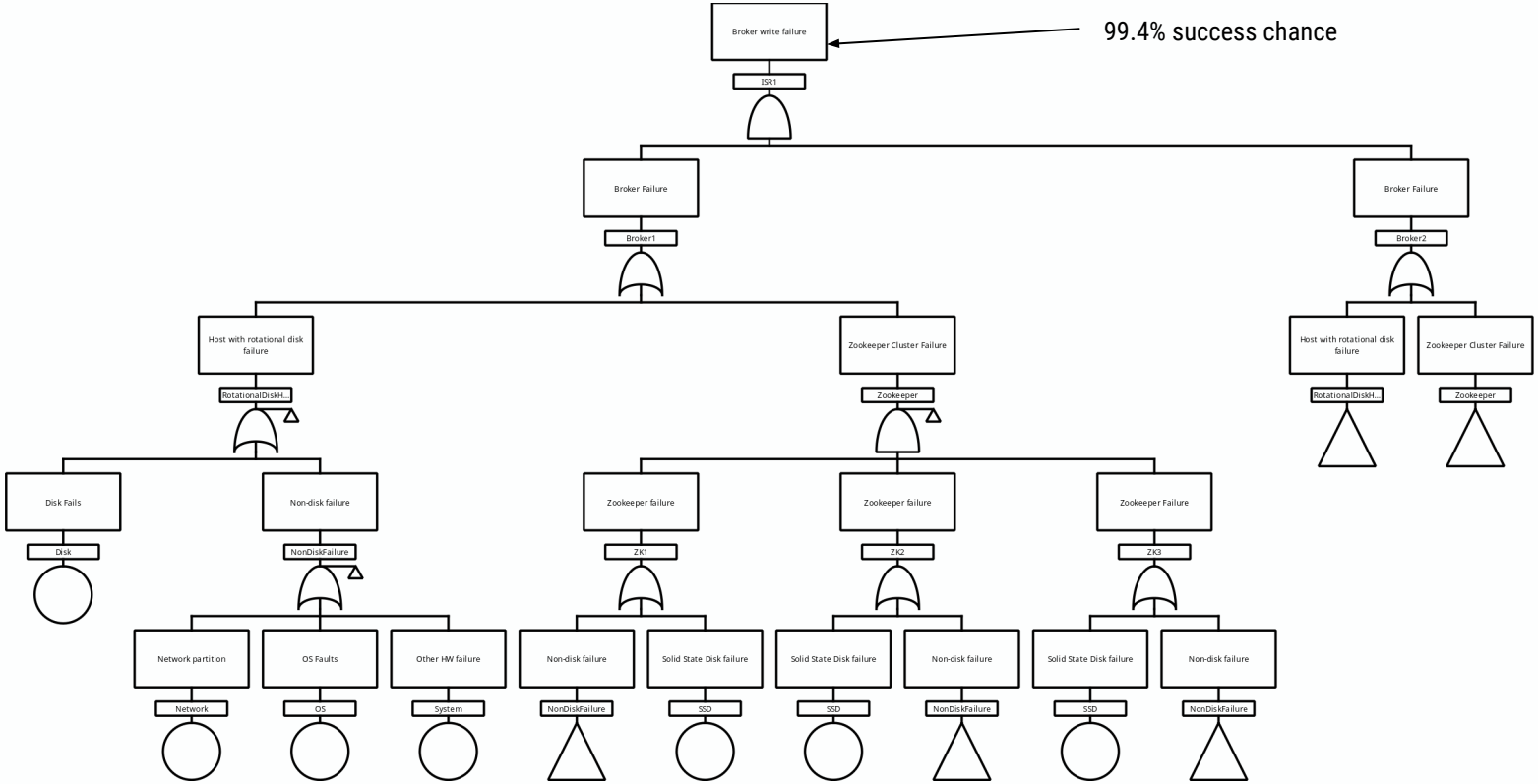
Availability - Two brokers, single ZK



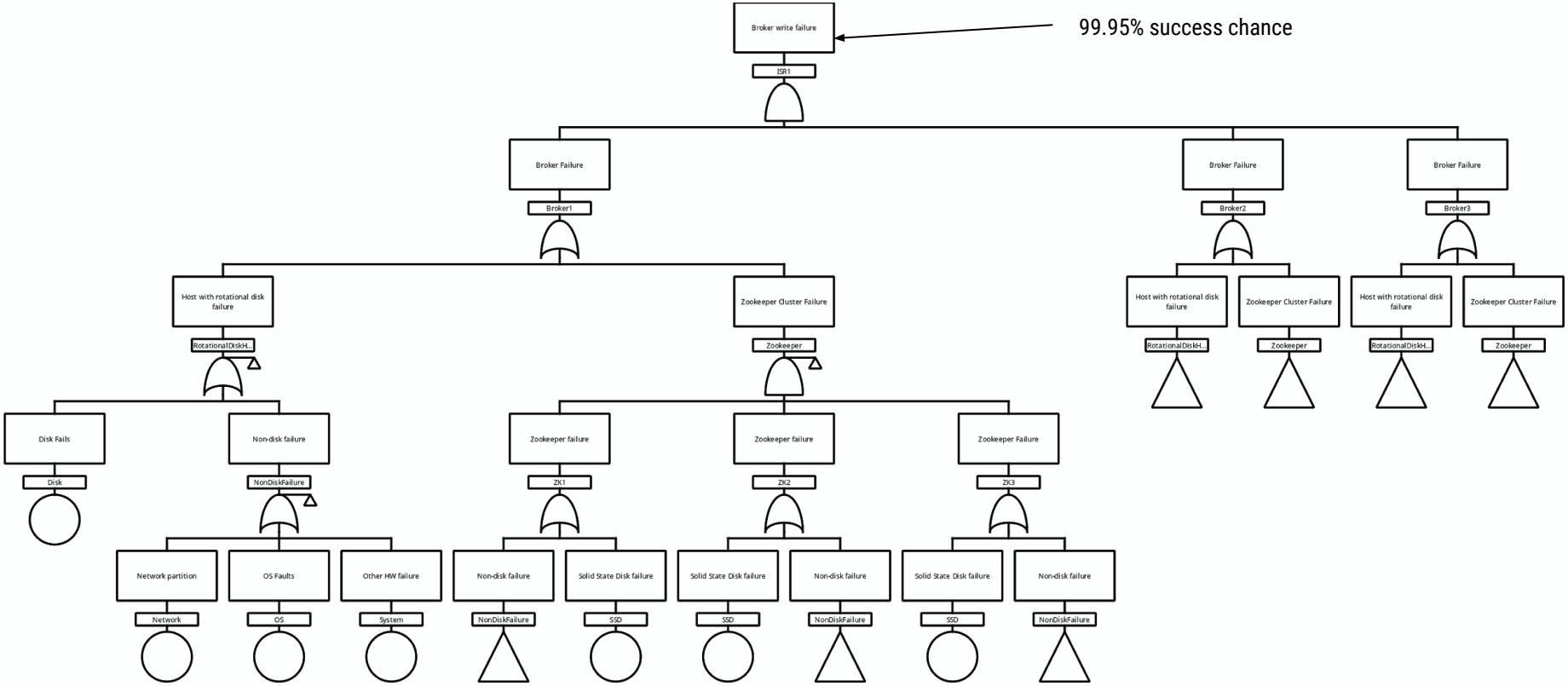
Availability - Collapse Host Faults



Availability - Multiple ZKs



Availability - Three Brokers



Availability - Summary

	Success Probability	Cost Per Nine*
Standalone	87.2%	n/a
Two brokers, ISR=1, One ZK	98.36%	2
Two brokers, ISR=1, Three ZKs	99.36%	2
Two brokers, ISR=1, Five ZKs	99.36%	3
Three brokers, ISR=1, Three ZKs	99.95%	1.5
Three brokers, ISR=2, Three ZKs	99.36%	2.25

* Cost is computed in “disk units” / “number of nines”:
Kafka Broker Rotational Disk = .5
Zookeeper SSD Disk = 1
Lower is better

Availability - Broker SSD

	Success Probability	Cost Per Nine*
Standalone	90.4%	1
Two brokers, ISR=1, One ZK	99.08%	1.5
Two brokers, ISR=1, Three ZKs	99.77%	2.5
Two brokers, ISR=1, Five ZKs	99.77%	3.5
Three brokers, ISR=1, Three ZKs	99.99%	1.5
Three brokers, ISR=2, Three ZKs	99.77%	3

* SSD Disk = 1

Availability - Broker EBS

	Success Probability	Cost Per Nine*
Standalone	91.6%	1.5
Two brokers, ISR=1, One ZK	99.29%	2.25
Two brokers, ISR=1, Three ZKs	99.82%	3.75
Two brokers, ISR=1, Five ZKs	99.82%	5.25
Three brokers, ISR=1, Three ZKs	99.99%	4.5
Three brokers, ISR=2, Three ZKs	99.82%	2.25

* EBS disk units:
EBS SSD Disk = 1.5

Assumption that EBS fails at .2%

Durability

What are the chances of losing data?

Service Level Objective (SLO):

99.99999% durability per year

Durability

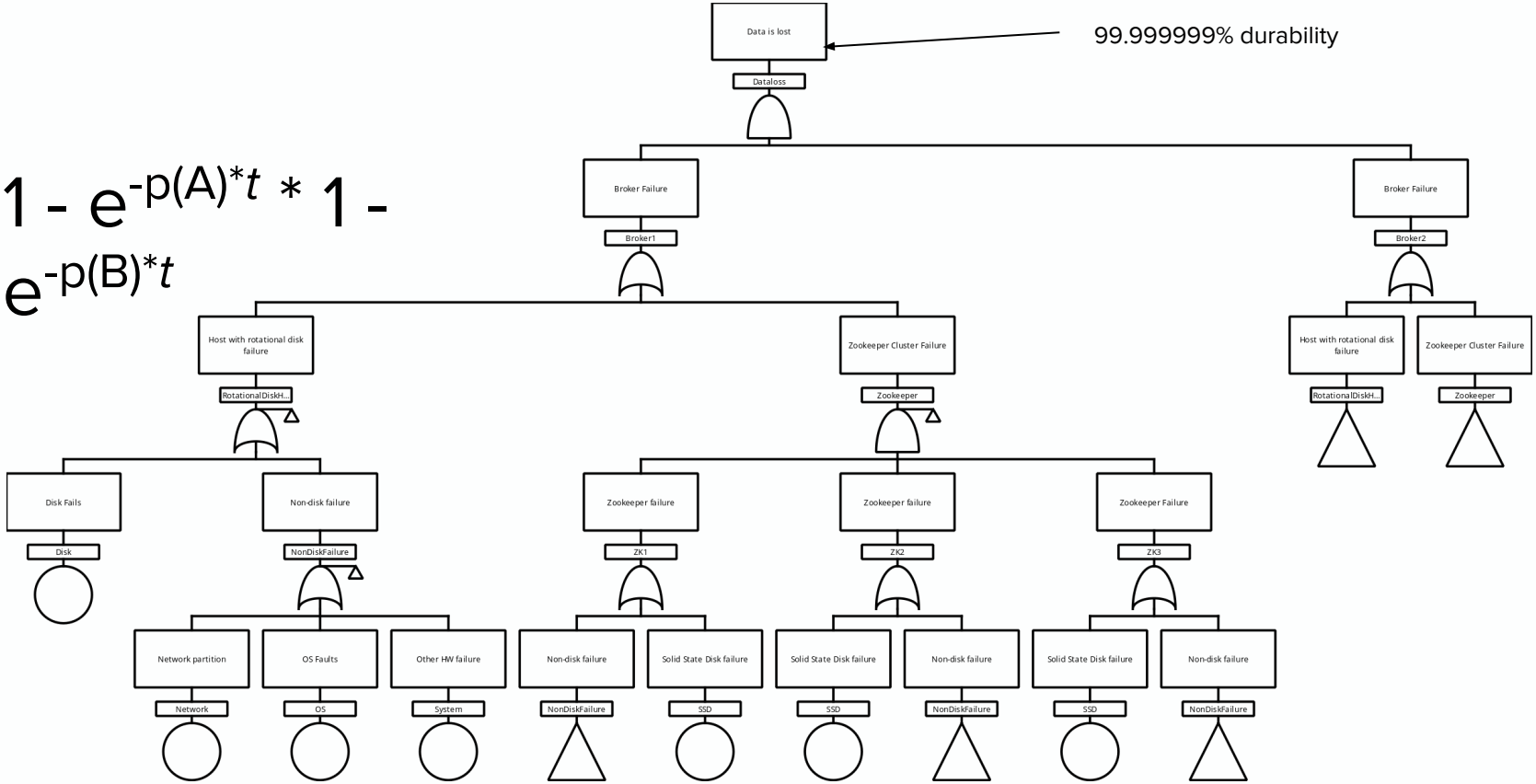
We lose data when all hosts with replicas go down

Assumptions:

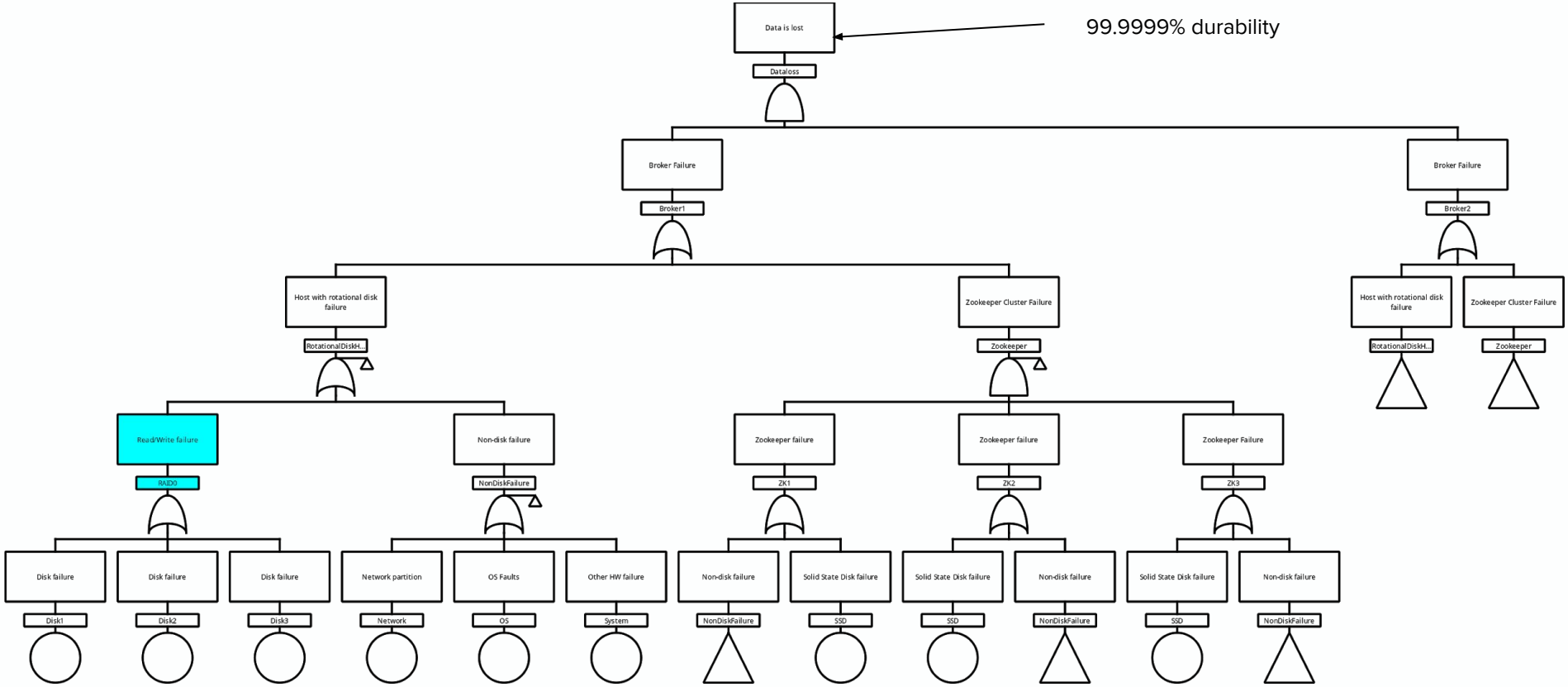
- **6TB per broker (2TB per disk w/ RAID)**
- **70MB/s replication rate**
- **~24 hours to replicate full broker**
- **We replace bad hosts almost immediately**

Durability - Two brokers - One 6TB Disk

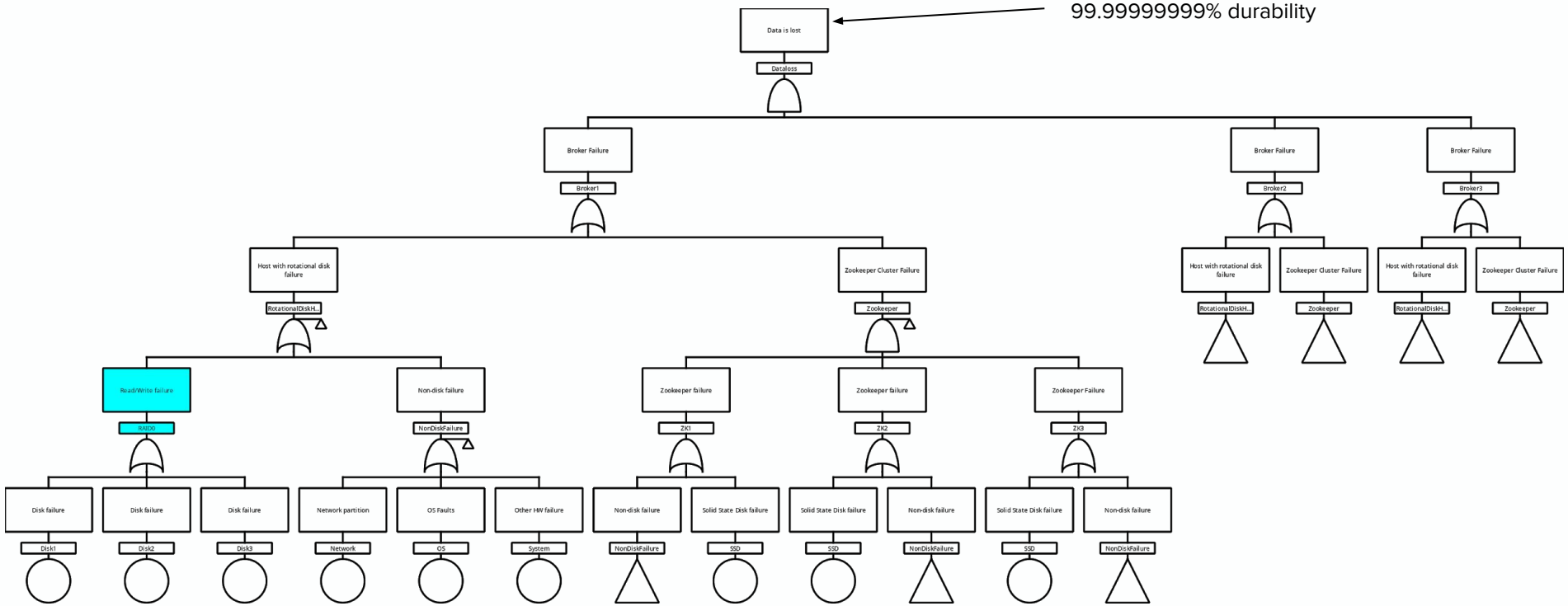
$$1 - e^{-p(A)*t} * 1 - e^{-p(B)*t}$$



Durability - Add Raid0



Durability - Three brokers

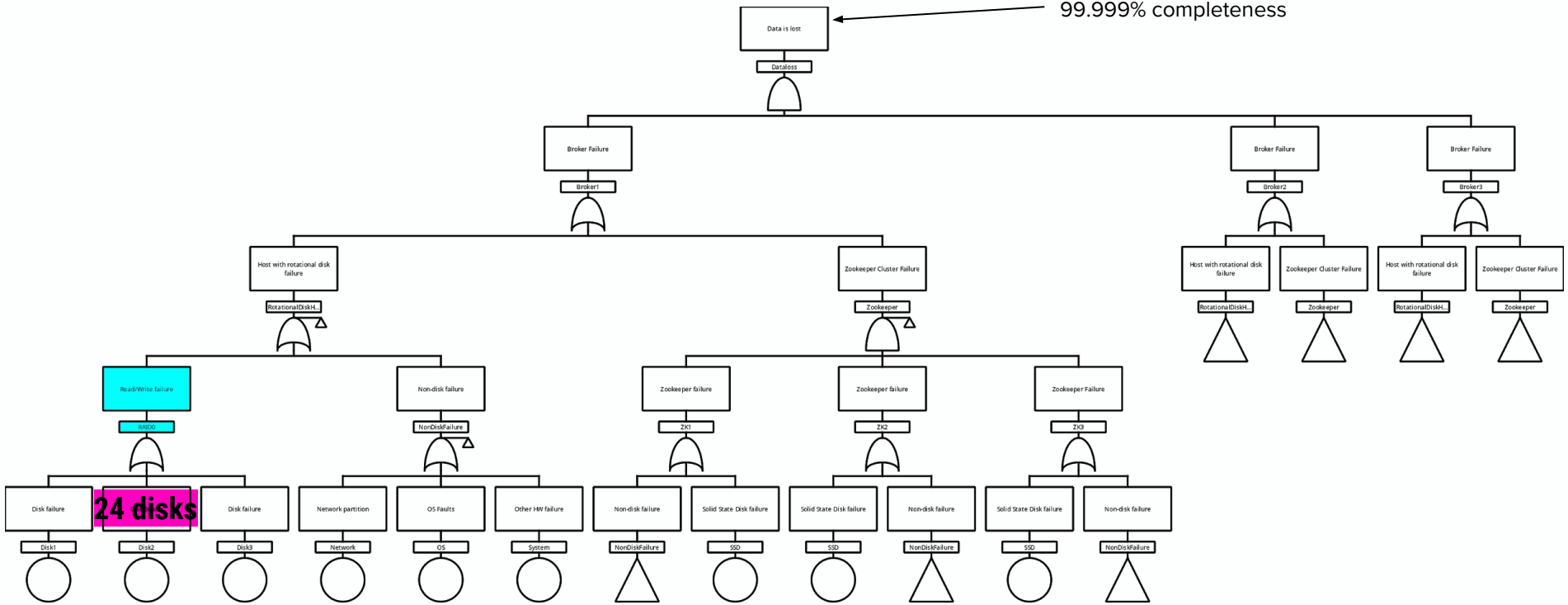


Durability

Assumption:

- **48TB per broker**
- **70MB/s replication rate**
- **~8 days to replicate full broker**
- **We replace bad hosts almost immediately**

Durability - Three brokers 24 disks



Durability - Summary

	Data completeness	Cost Per Nine*
Standalone	99.99%	.125
Two brokers	99.999999%	.5
Two brokers RAID0	99.9999%	.22
Three brokers RAID0	99.99999999%	.15
Three brokers RAID0 - 48TB	99.999%	1

* Cost is computed in “disk units” / “number of nines”:
Single non-raid disk = .5
Raid0 = .167
Zookeeper SSD Disk = 1
Lower is better

Latency

FTA focused on failures

Latency is not an inherent failure

Experiment with worst-case scenarios

Conclusion

Tools and References

Fault Tree Models: github.com/afalko/fta-kafka

OSS tool to draw and compute models: github.com/rakhimov/scram

How Not to Go Boom: Lessons for SREs from Oil Refineries by Emil Stolarsky

Fault Tree Analysis - A History by Clifton A. Ericson II

Fault Tree Handbook with Aerospace Applications by Dr. Michael Stamatelatos and Mr. José Caraballo

Failure Trends in a Large Disk Drive Population by Eduardo Pinheiro, Wolf-Dietrich Weber and Luiz Andre Barroso

Solving Data Loss in Massive Storage Systems by Jason Resch

Failures at Scale and How to Ride Through Them by James Hamilton

Takeaways

FTA can be applied to:

- **Kafka Availability and Durability SLOs**
- **Find cost savings**
- **Uncover decisions that reduce reliability**

Further Work

Kafka on Kubernetes analysis

Improve scram-pra

Better FTA inputs via Distributed Tracing

Thank you!

github.com/afalko/fta-kafka

Andrey Falko <afalko@lyft.com>

Thank You