Check before You Change:

Preventing Correlated Failures in Service Updates

Ennan Zhai — Alibaba Group 阿里巴巴集团

Ang Chen — RICE

Ruzica Piskac — LUX ET VERITAS

Mahesh Balakrishnan — facebook

Bingchuan Tian — 南京大學 NANJING UNIVERSITY

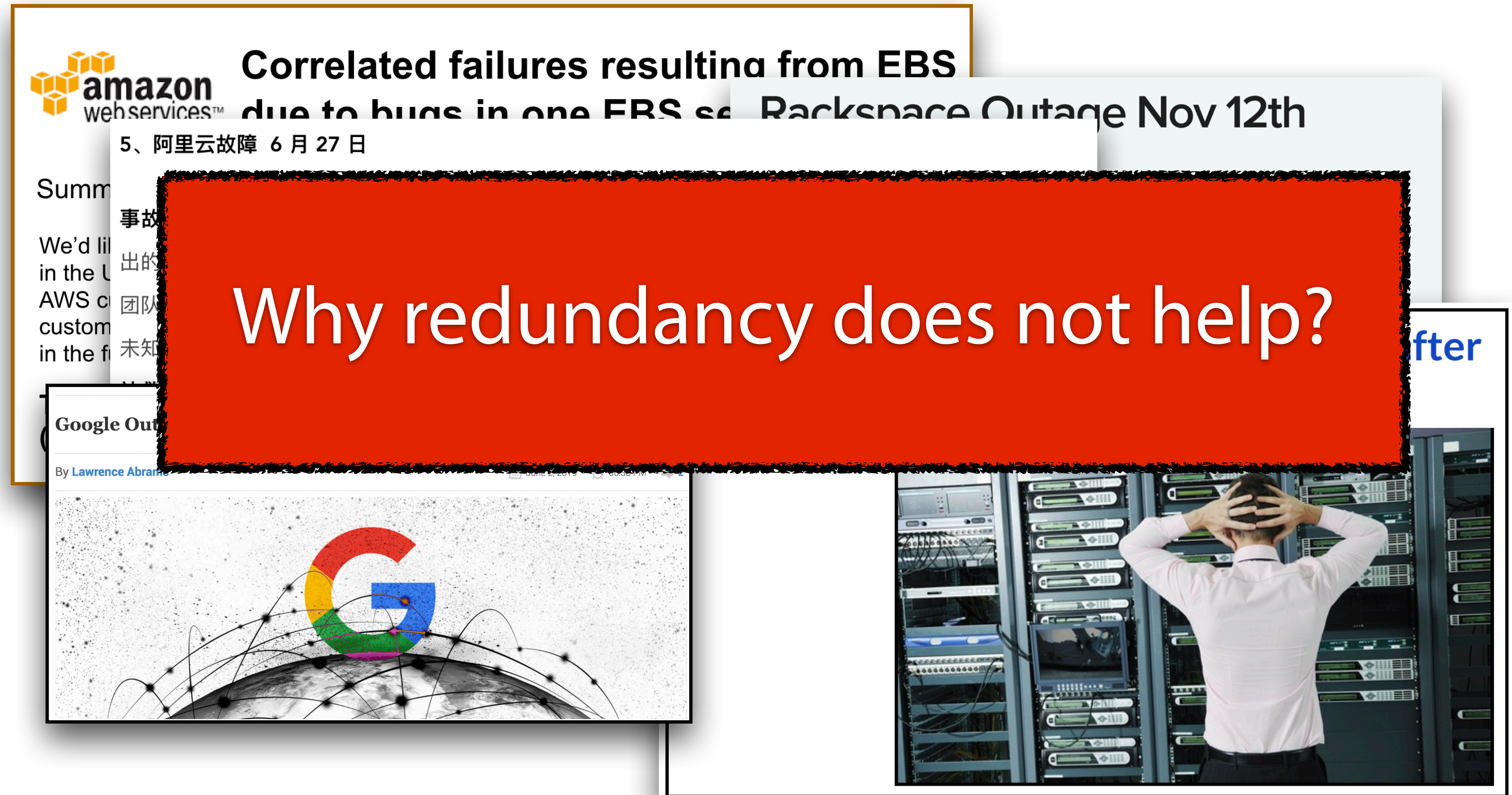
Bo Song, Haoliang Zhang — Google

Background

- Cloud services ensure reliability by redundancy:
  - Storing data redundantly
  - Replicating service states across multiple nodes

- Examples:
  - Amazon AWS, AliCloud, Google Cloud, etc. replicate their data and service states

However, cloud outages still occur



Why redundancy does not help?

An AWS Outage in 2018

AWS outage killed some cloudy servers, recovery time is uncertain

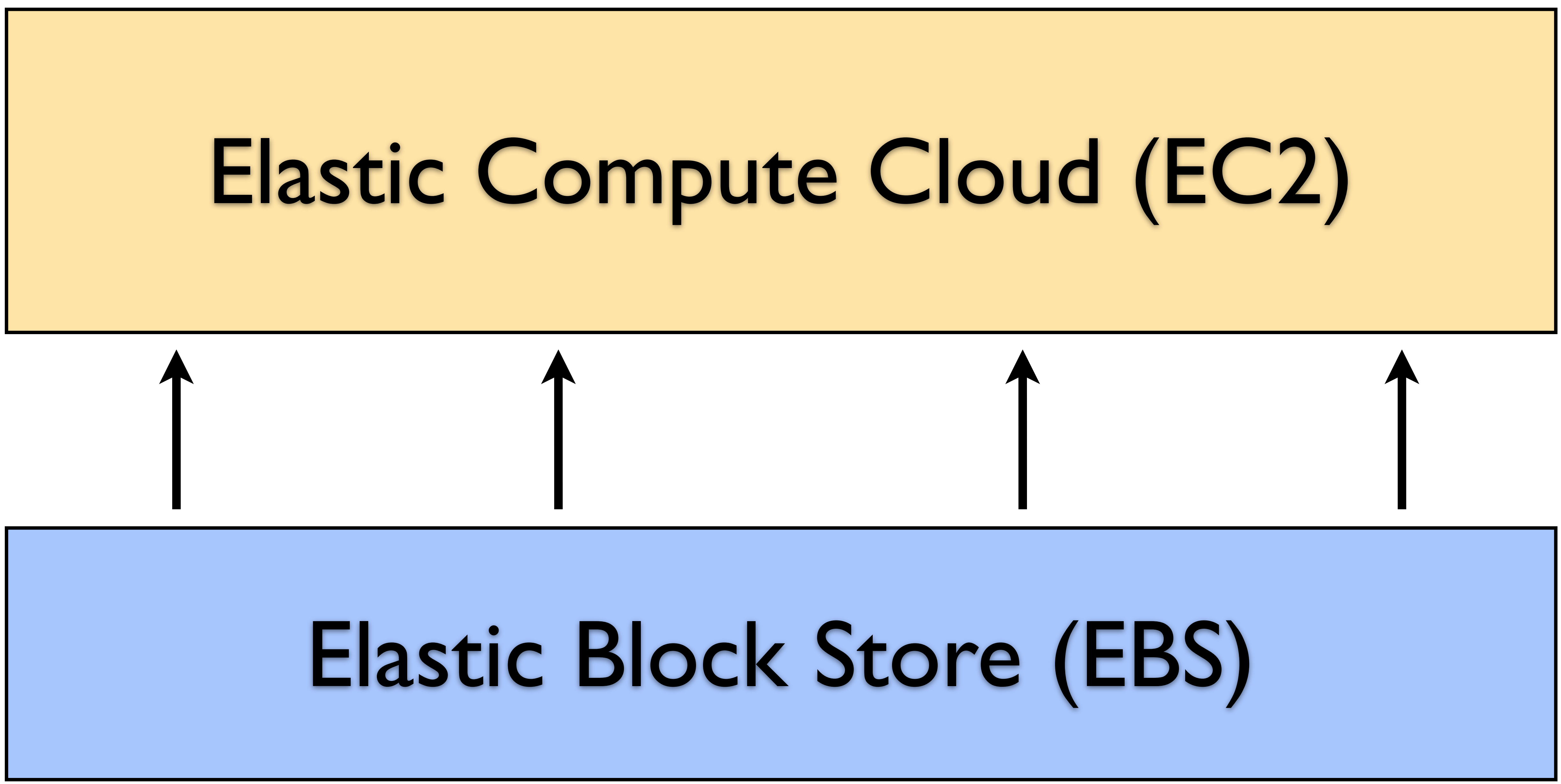
‘Power event’ blamed, hit subset of kit in US-EAST-1

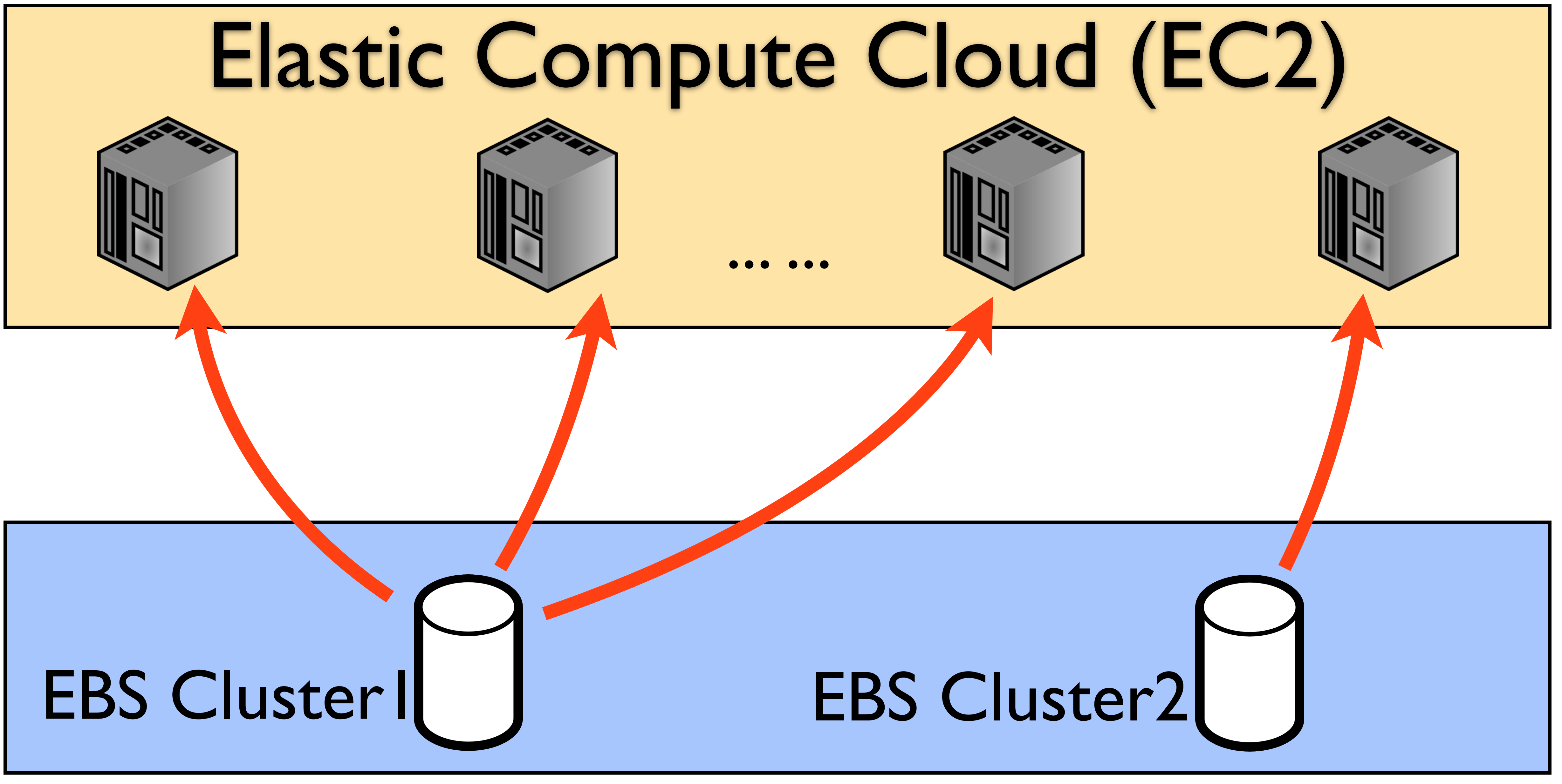
By Simon Sharwood 1 Jun 2018 at 00:48 16 SHARE ▼

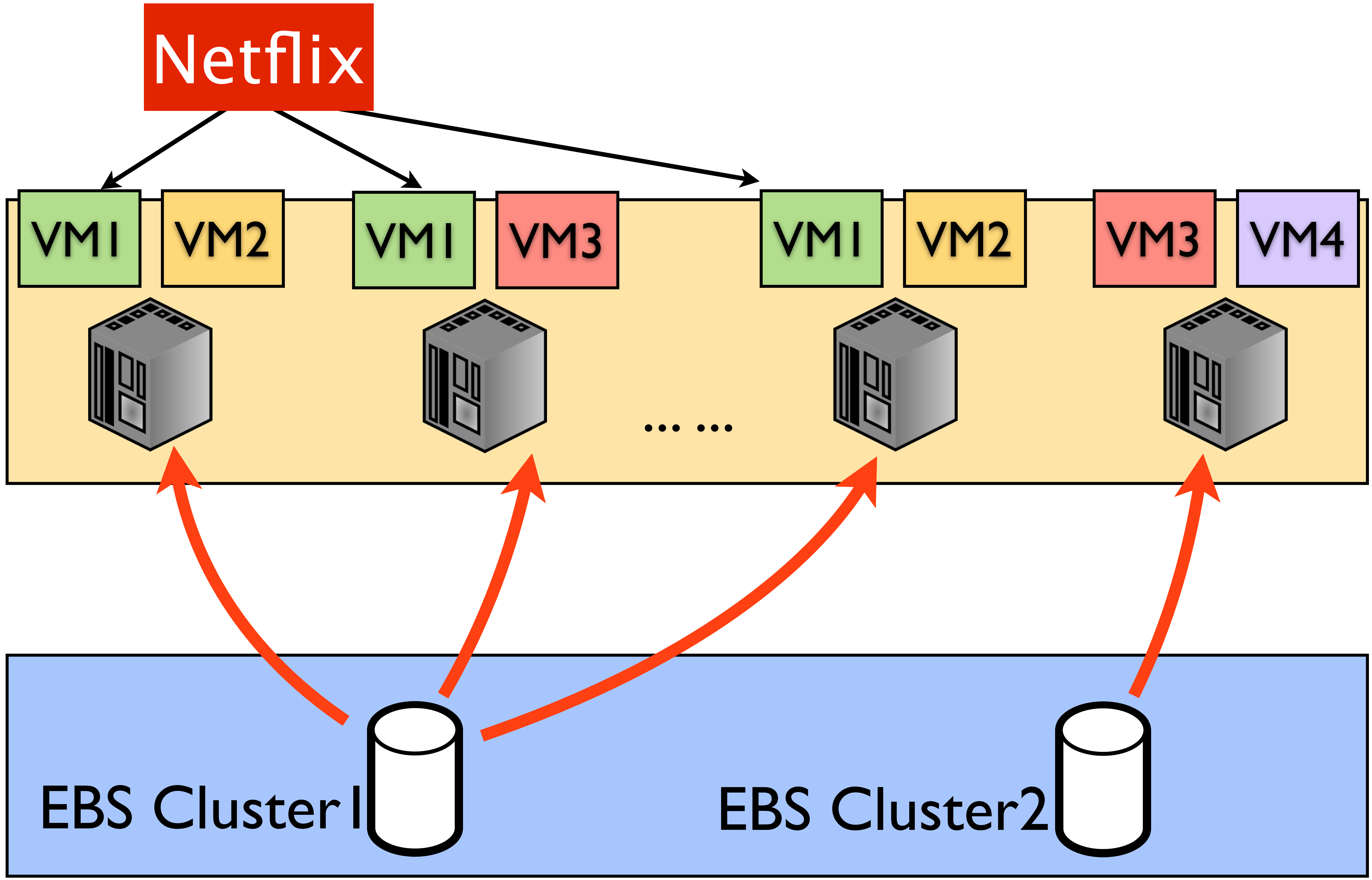
Updated Parts of Amazon Web Services' US-East-1 region have experienced about half an hour of downtime, but some customers' instances and data can't be restored because the hardware running them appears to have experienced complete failure.

The cloud colossus’ status page reports an investigation of “connectivity issues affecting some instances in a single Availability Zone in the US-EAST-1 Region” as of 3:13 PM PDT on Thursday, May 31.

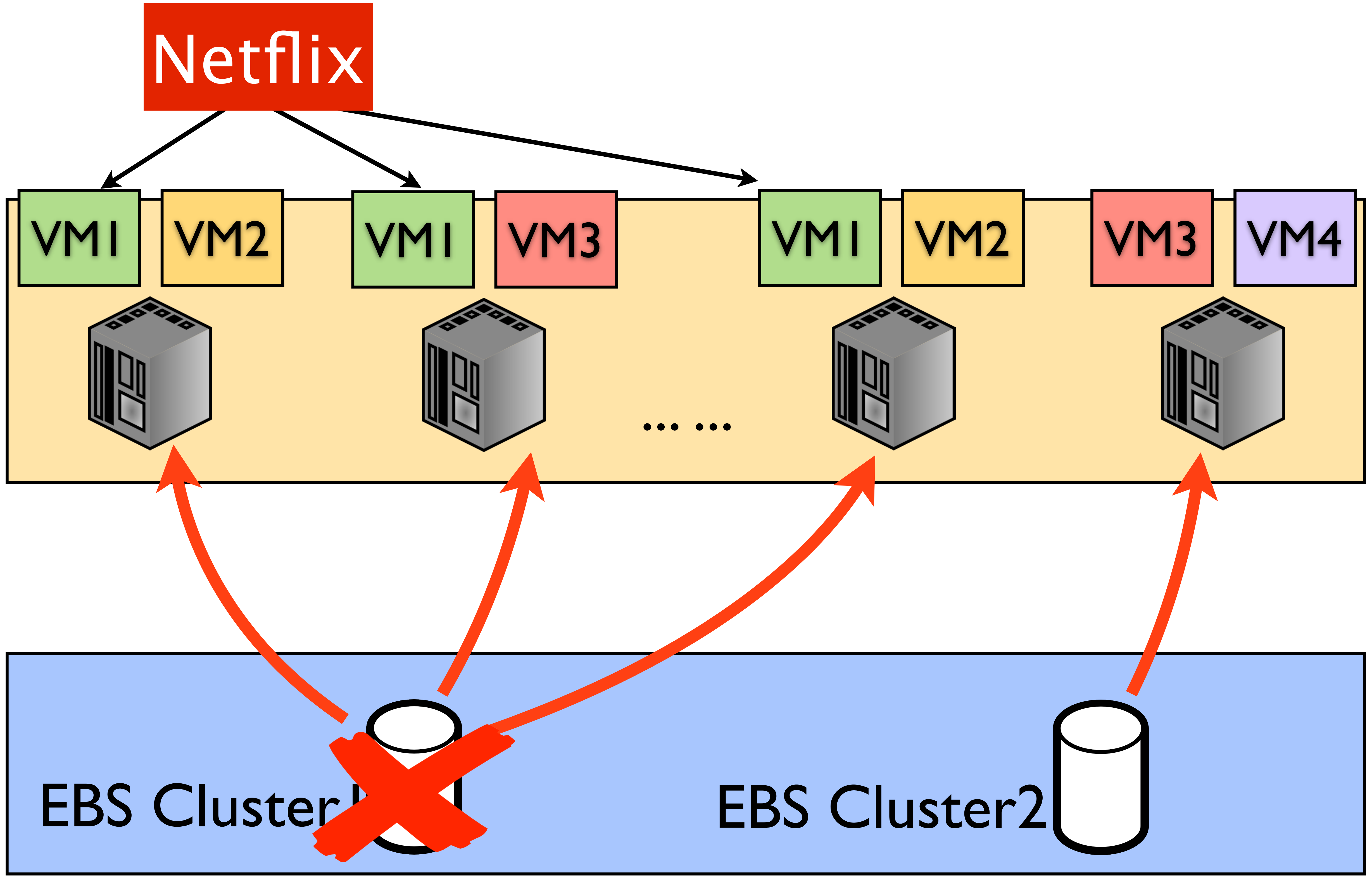
A 3:42 PM update confirmed “an issue in one of the datacenters that makes up one of US-EAST-1 Availability Zones. This was a result of a power event impacting a small percentage of the physical servers in that datacenter as well as some of the networking devices.”

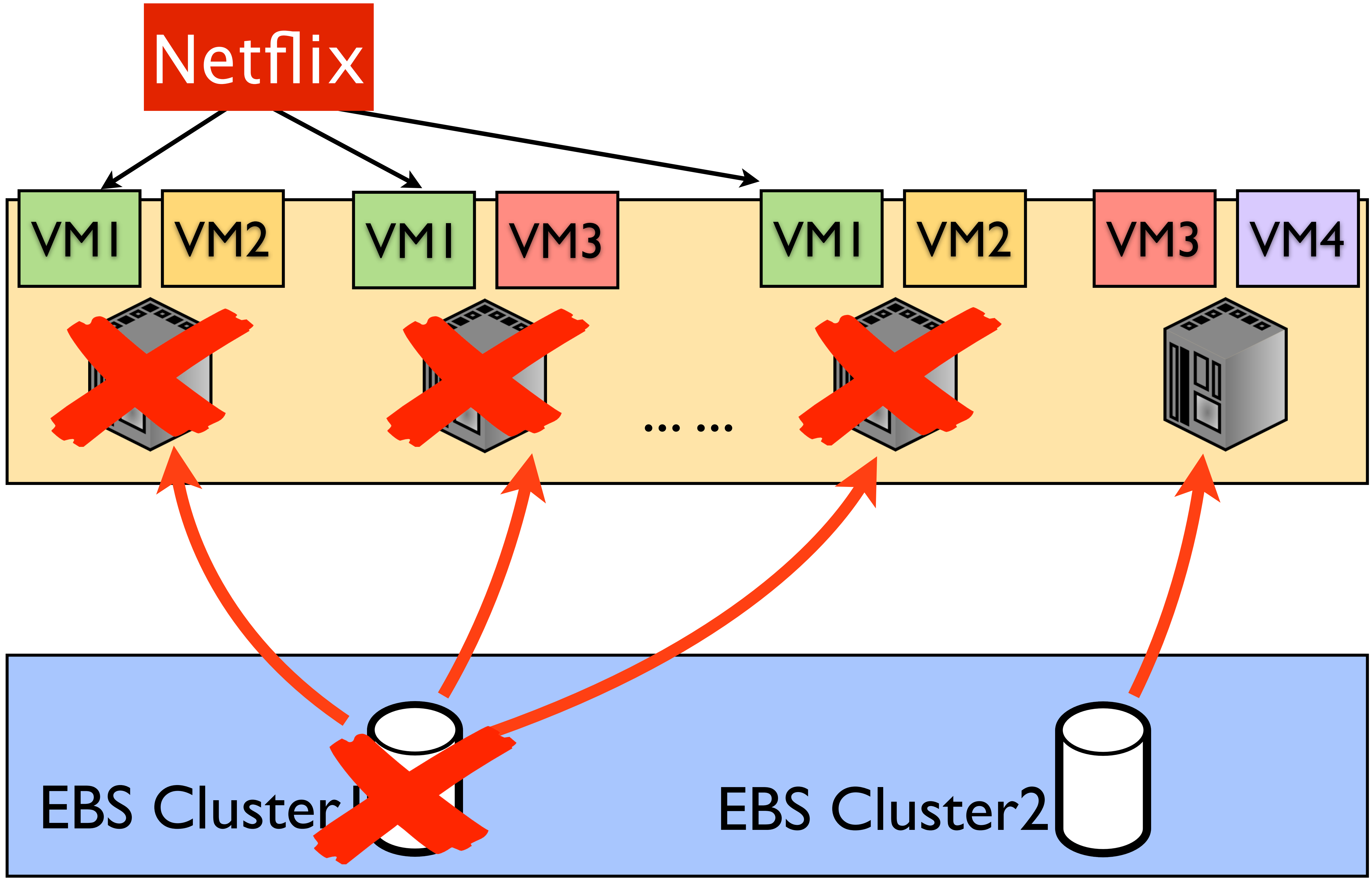

Elastic Compute Cloud (EC2)
Elastic Block Store (EBS)

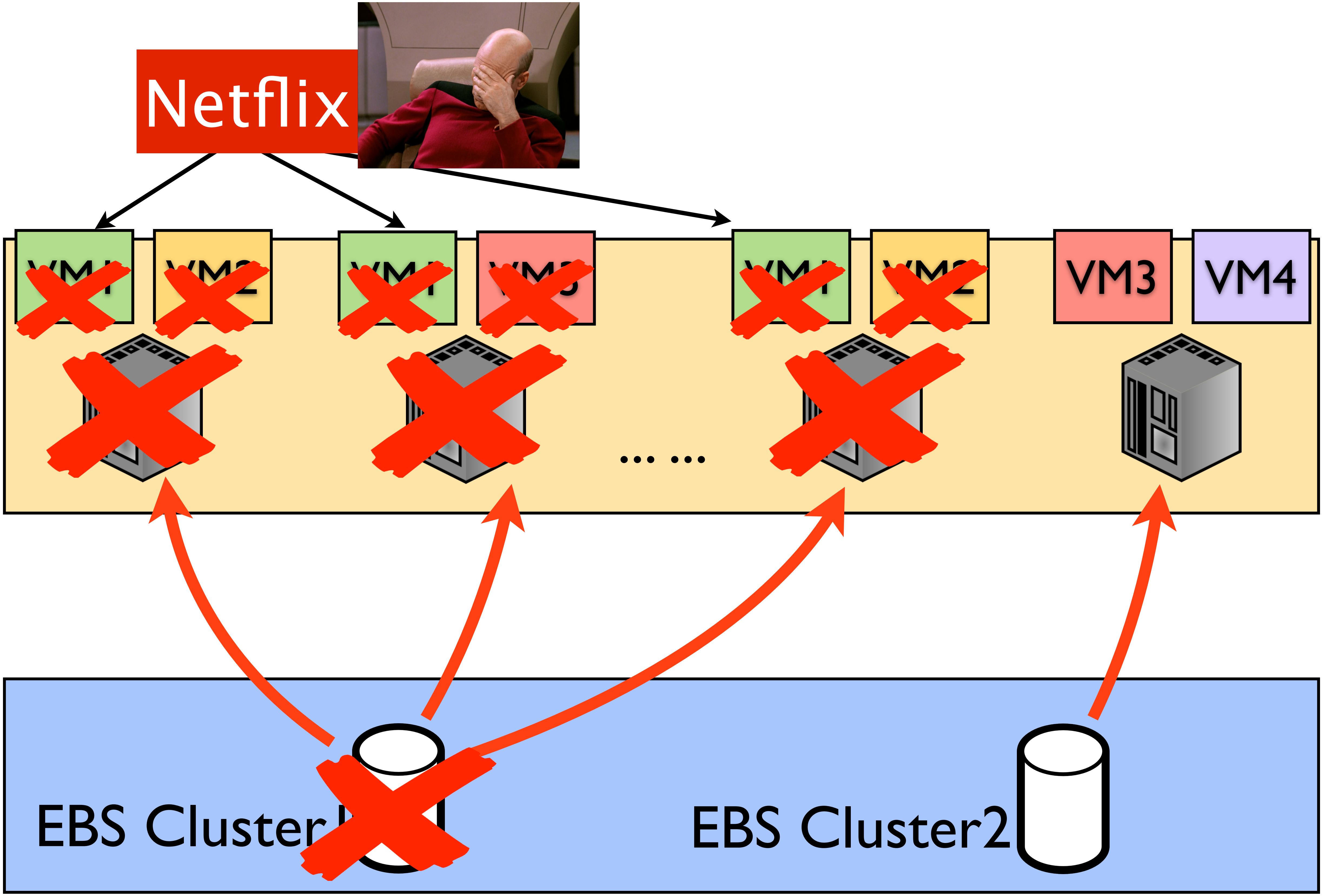

Elastic Compute Cloud (EC2)
... ...
EBS Cluster1
EBS Cluster2



Netflix
VM1
VM2
VM1
VM3
VM1
VM2
VM3
VM4
....
EBS Cluster1
EBS Cluster2


Netflix
VM1
VM2
VM1
VM3
VM1
VM2
VM3
VM4
… …
EBS Cluster1
EBS Cluster2

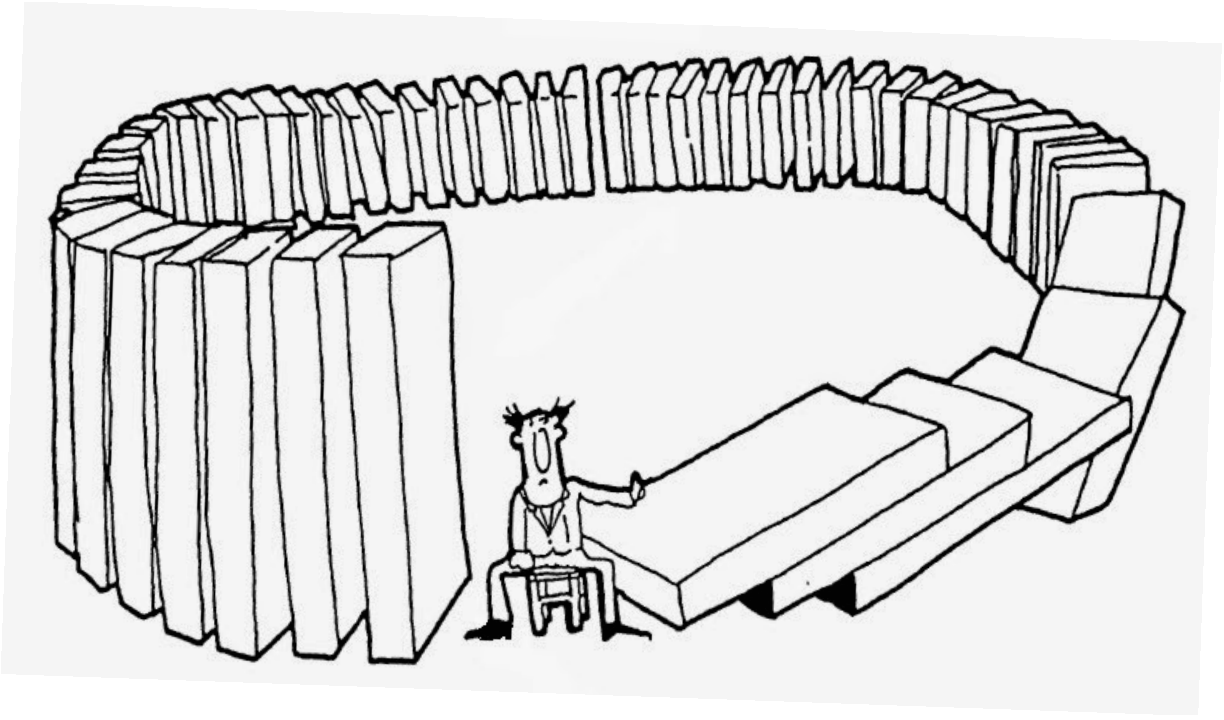

Netflix
VM1
VM2
VM1
VM3
VM1
VM2
VM3
VM4
... ...
EBS Cluster1
EBS Cluster2


Netflix
VM1
VM2
VM1
VM3
VM1
VM2
VM3
VM4
… …
EBS Cluster1
EBS Cluster2

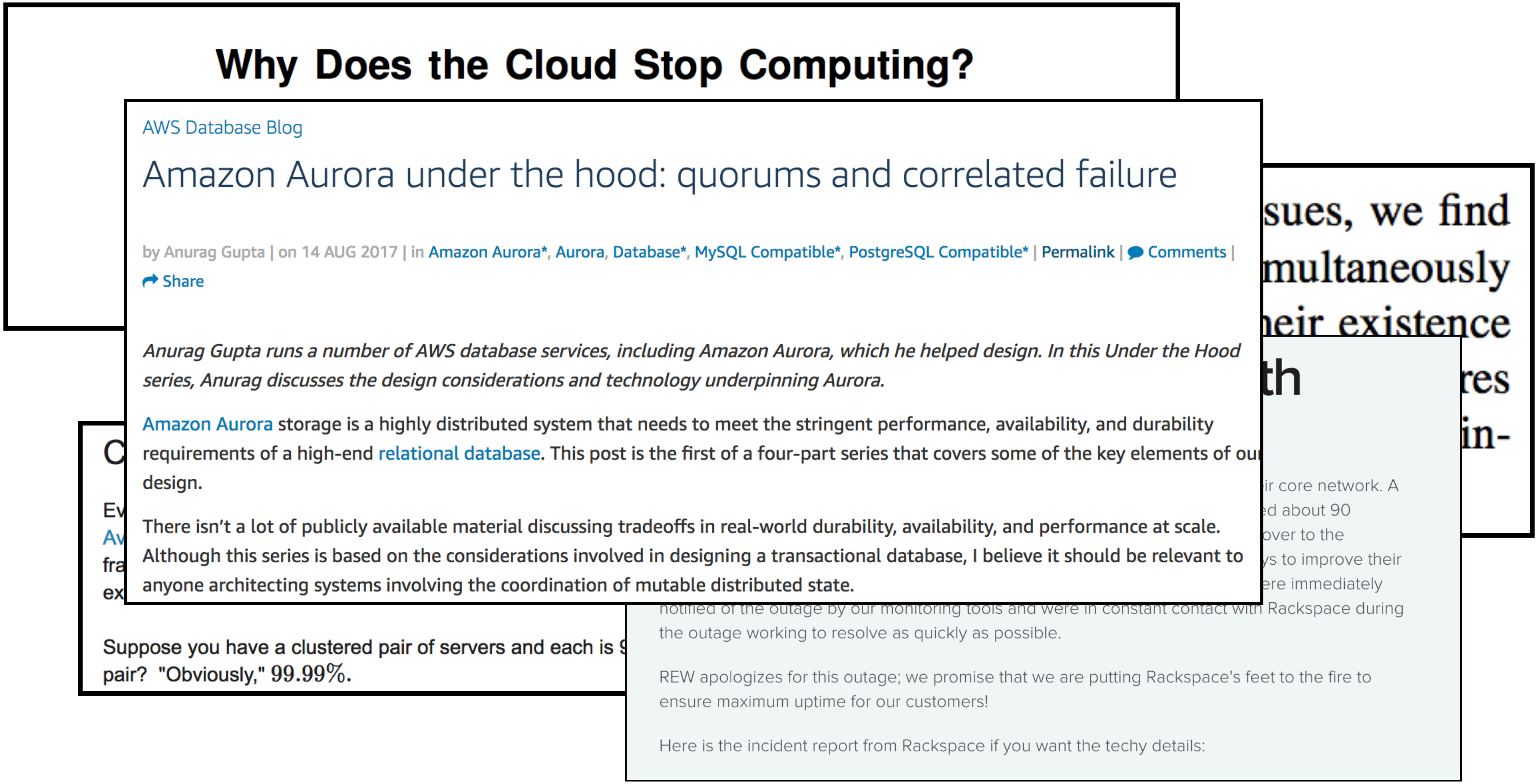

Netflix
Correlated failures resulting from deep dependencies
EBS Cluster1
EBS Cluster2

Correlated Failures

- Correlated failures are harmful and epidemic:
  - Propagated to all the redundant instances
  - Undermine redundancy and fault tolerance efforts



Correlated failures are prevalent



Why Does the Cloud Stop Computing?

AWS Database Blog

Amazon Aurora under the hood: quorums and correlated failure

by Anurag Gupta | on 14 AUG 2017 | in Amazon Aurora*, Aurora, Database*, MySQL Compatible*, PostgreSQL Compatible* | Permalink | Comments | Share

Anurag Gupta runs a number of AWS database services, including Amazon Aurora, which he helped design. In this Under the Hood series, Anurag discusses the design considerations and technology underpinning Aurora.

Amazon Aurora storage is a highly distributed system that needs to meet the stringent performance, availability, and durability requirements of a high-end relational database. This post is the first of a four-part series that covers some of the key elements of ou design.

There isn't a lot of publicly available material discussing tradeoffs in real-world durability, availability, and performance at scale. Although this series is based on the considerations involved in designing a transactional database, I believe it should be relevant to anyone architecting systems involving the coordination of mutable distributed state.

sues, we find multaneously heir existence th res in-

ir core network. A ed about 90 over to the ys to improve their ere immediately notified of the outage by our monitoring tools and were in constant contact with Rackspace during the outage working to resolve as quickly as possible.

REW apologizes for this outage; we promise that we are putting Rackspace's feet to the fire to ensure maximum uptime for our customers!

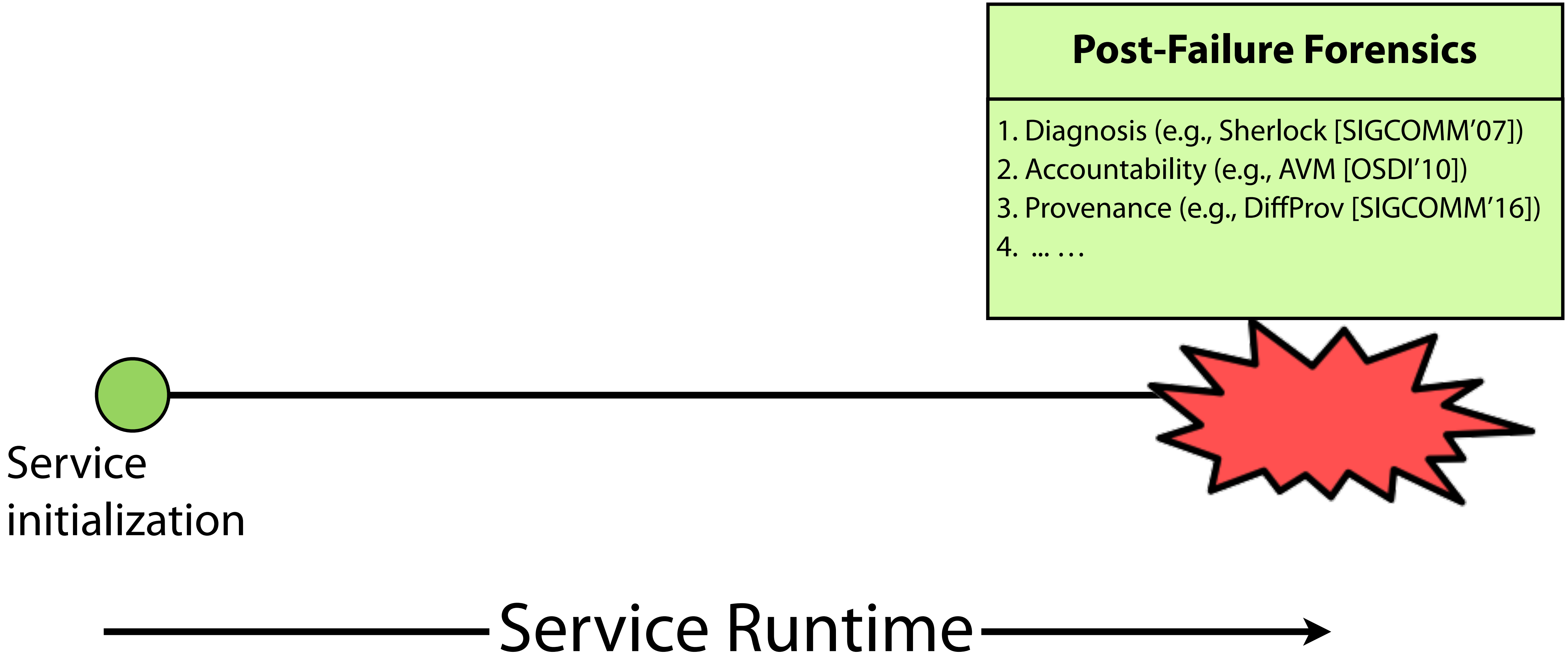
Here is the incident report from Rackspace if you want the techy details:

C Ev Av fra ex

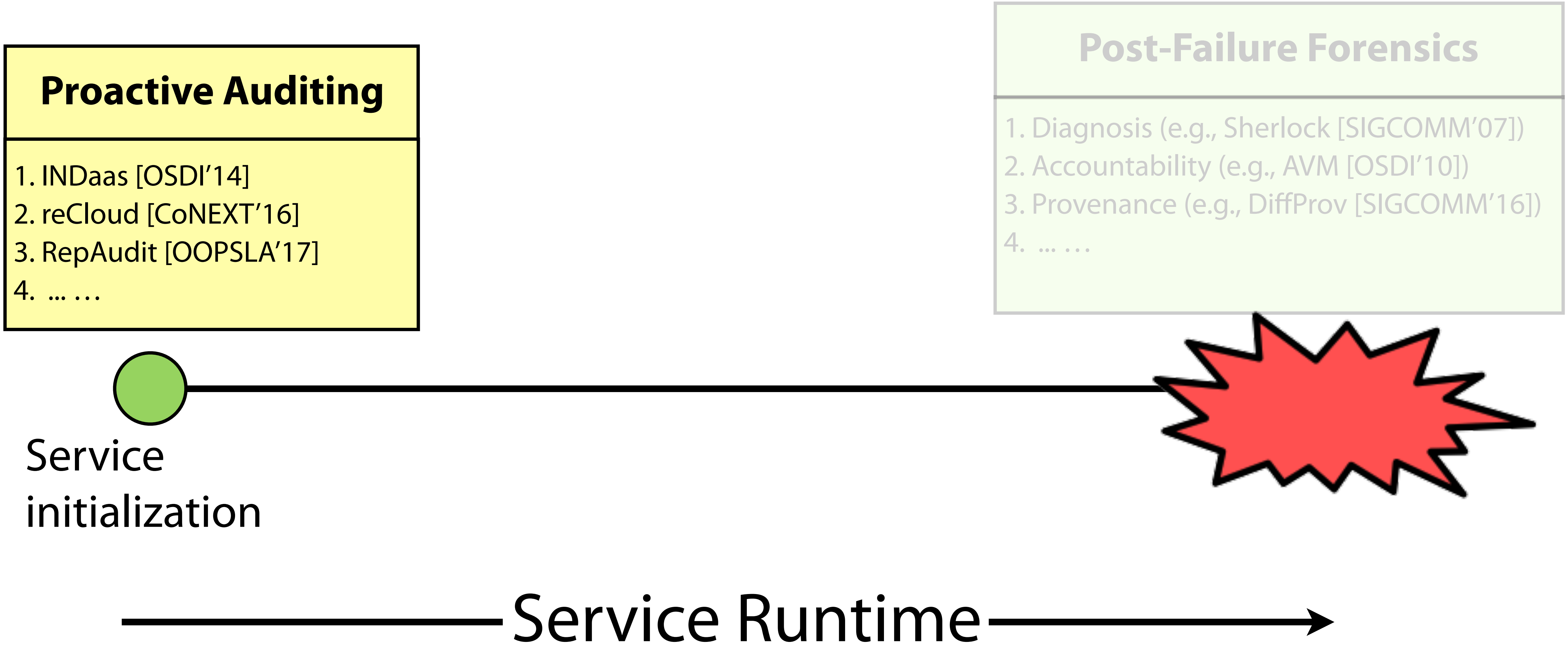
Suppose you have a clustered pair of servers and each is 9 pair? "Obviously," 99.99%.

State of the Art

State of the Art

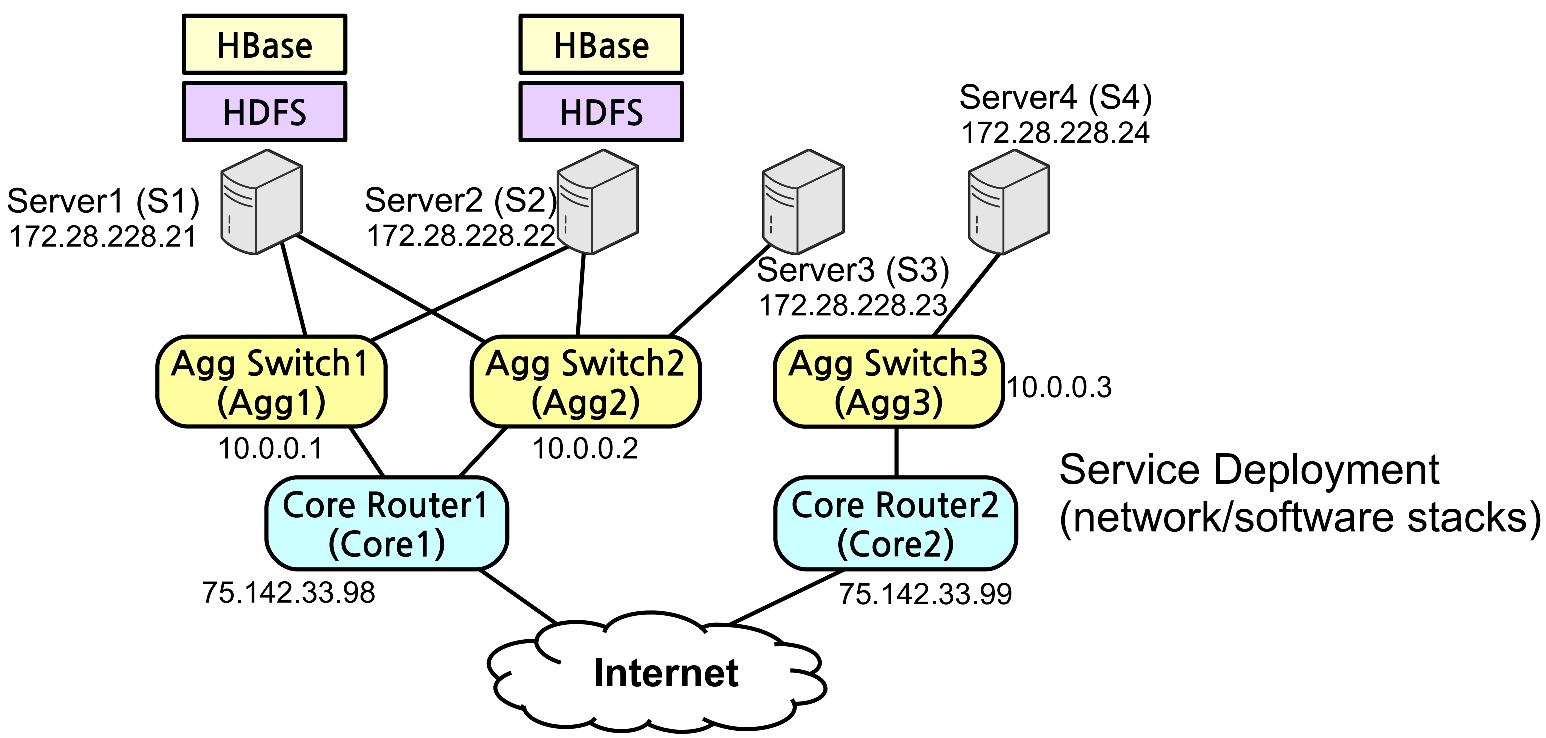


State of the Art



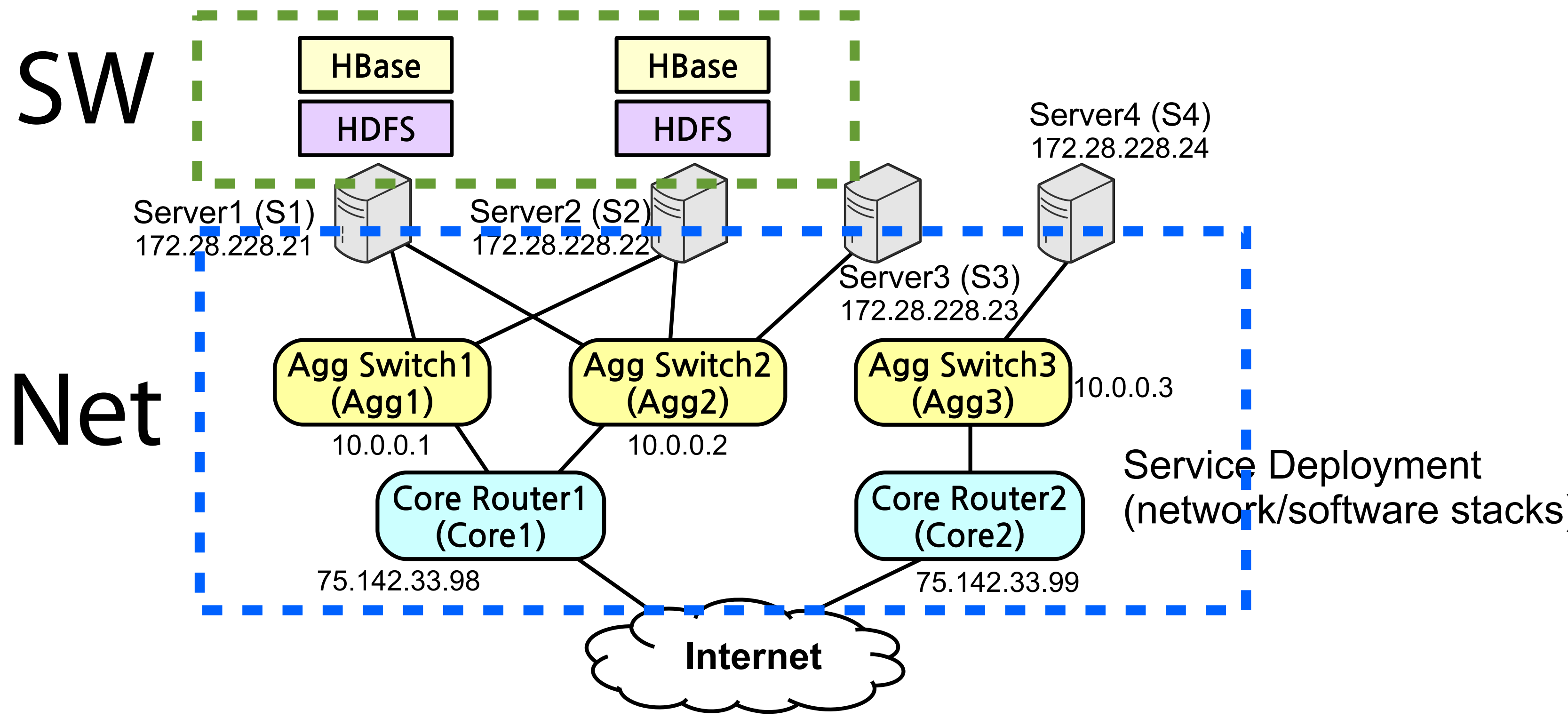
Proactive Auditing

- They did pre-deployment recommendations:



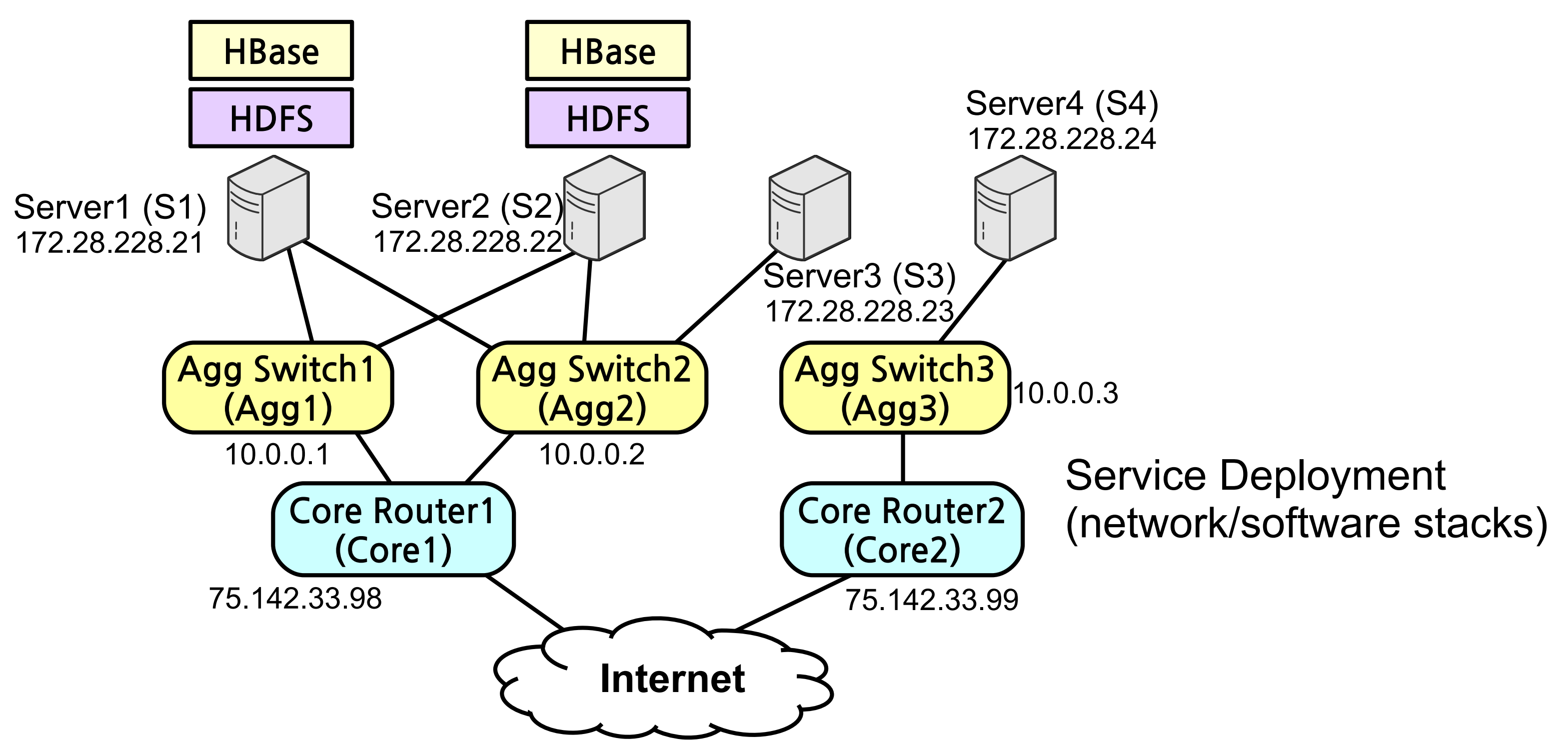
Proactive Auditing

- They did pre-deployment recommendations:
  - Step1: Automatically collecting dependency data



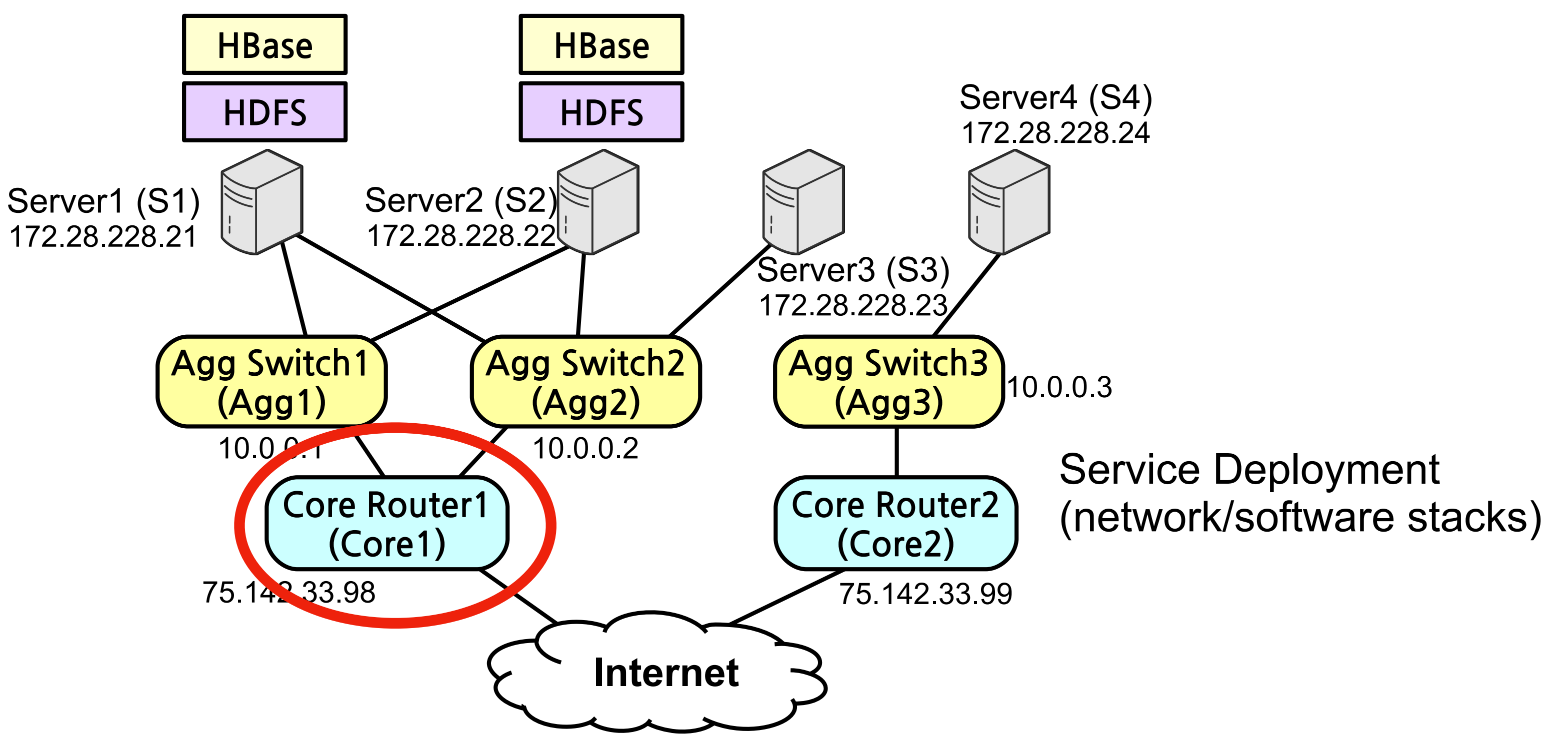
Proactive Auditing

- They did pre-deployment recommendations:
  - Step1: Automatically collecting dependency data
  - Step2: Modeling system stack in fault graph

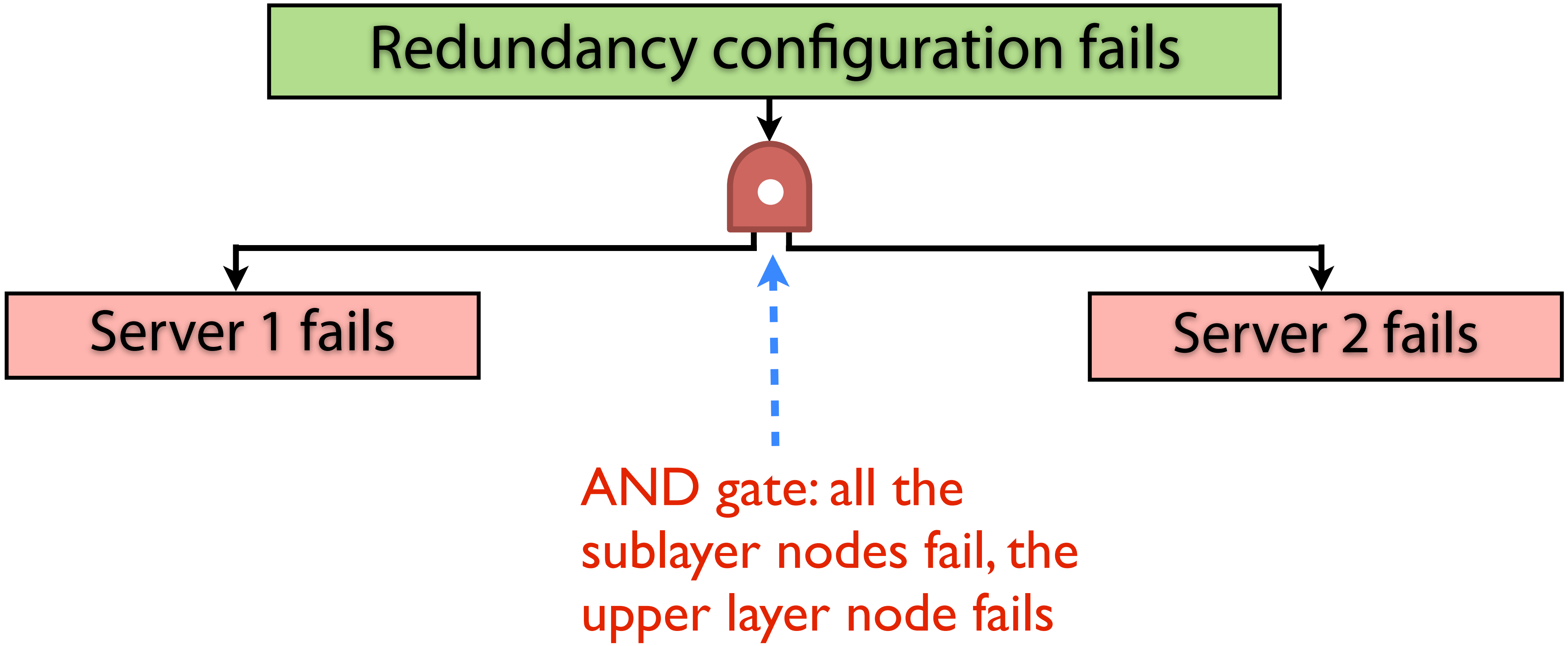


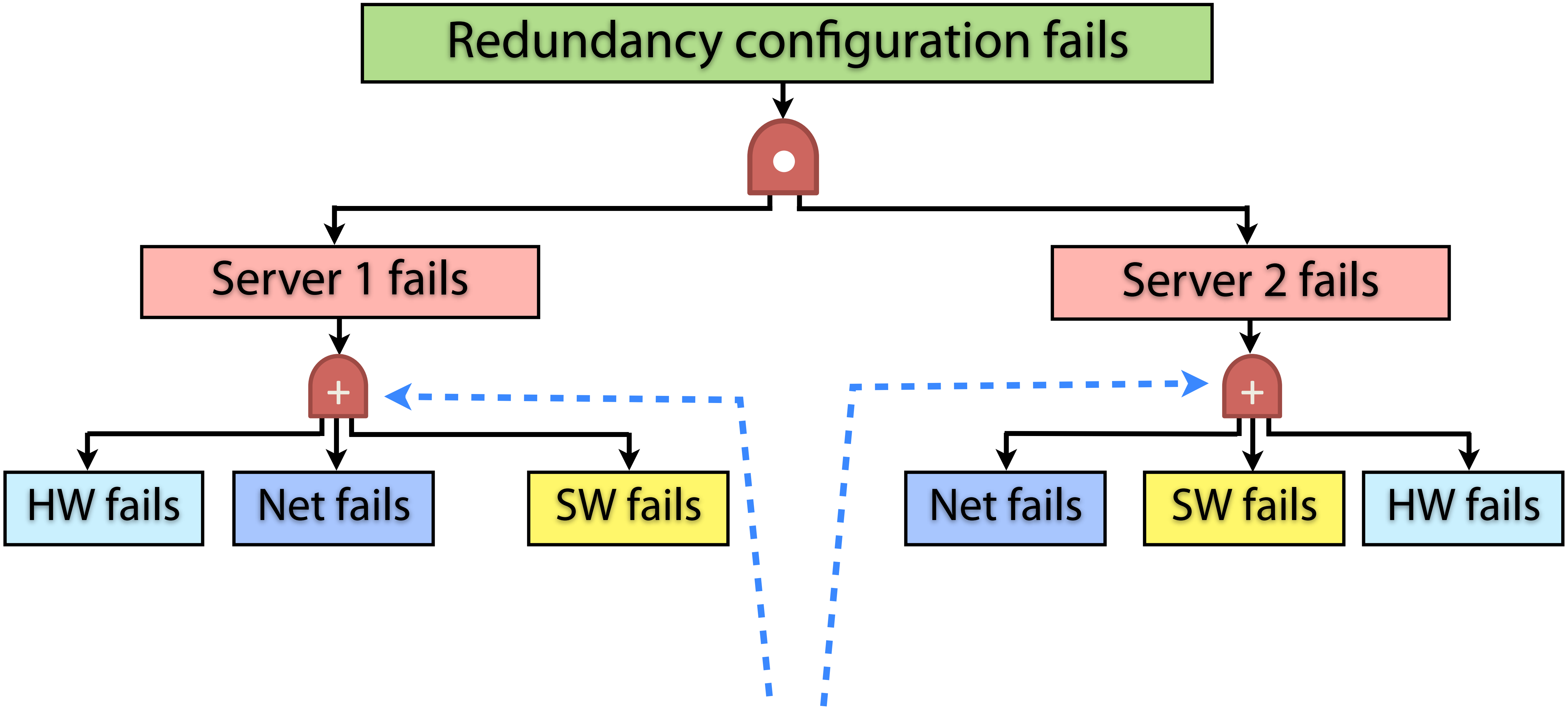
Proactive Auditing

- They did pre-deployment recommendations:
  - Step1: Automatically collecting dependency data
  - Step2: Modeling system stack in fault graph
  - Step3: Evaluating alternative deployments' independence

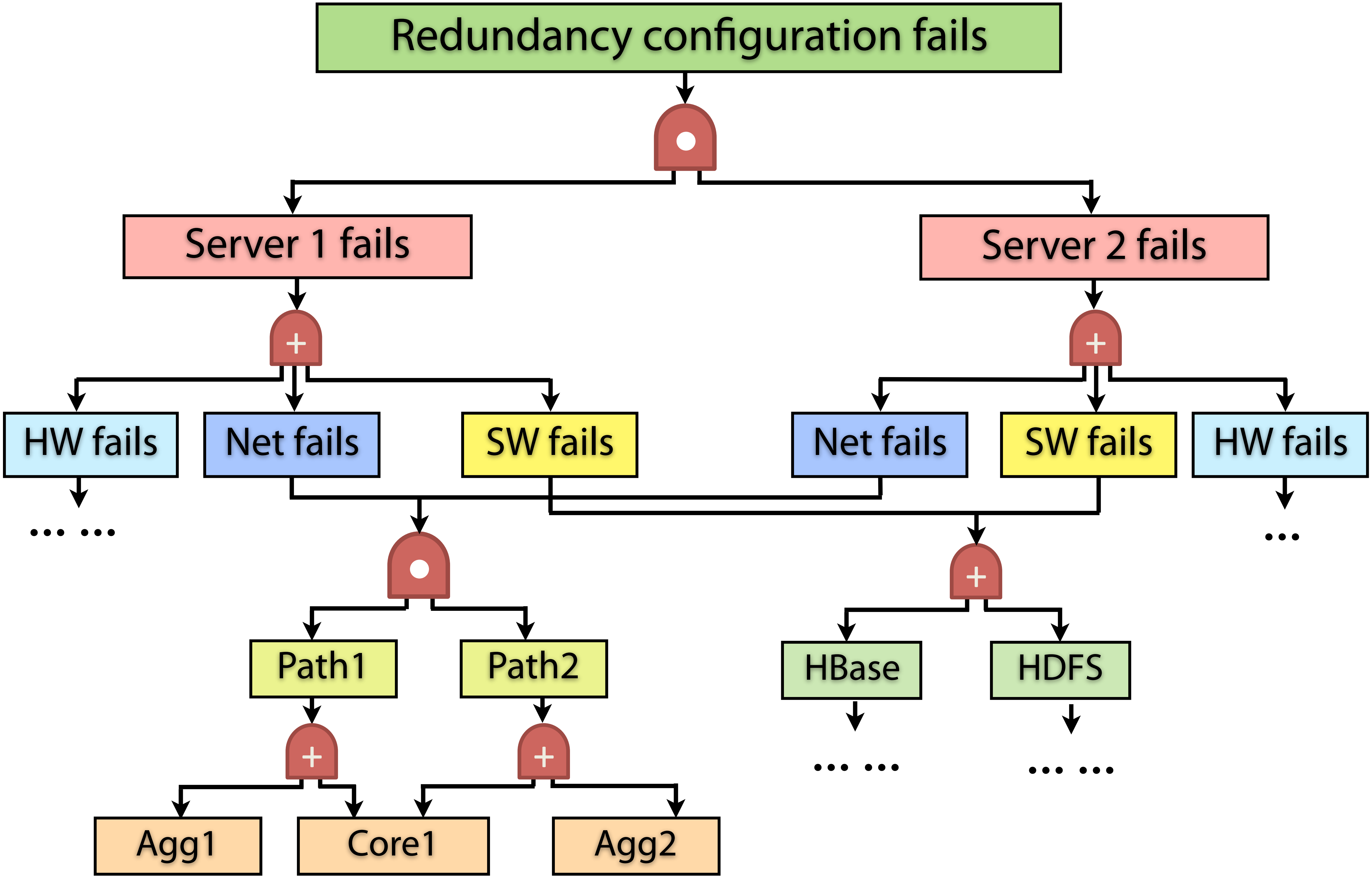


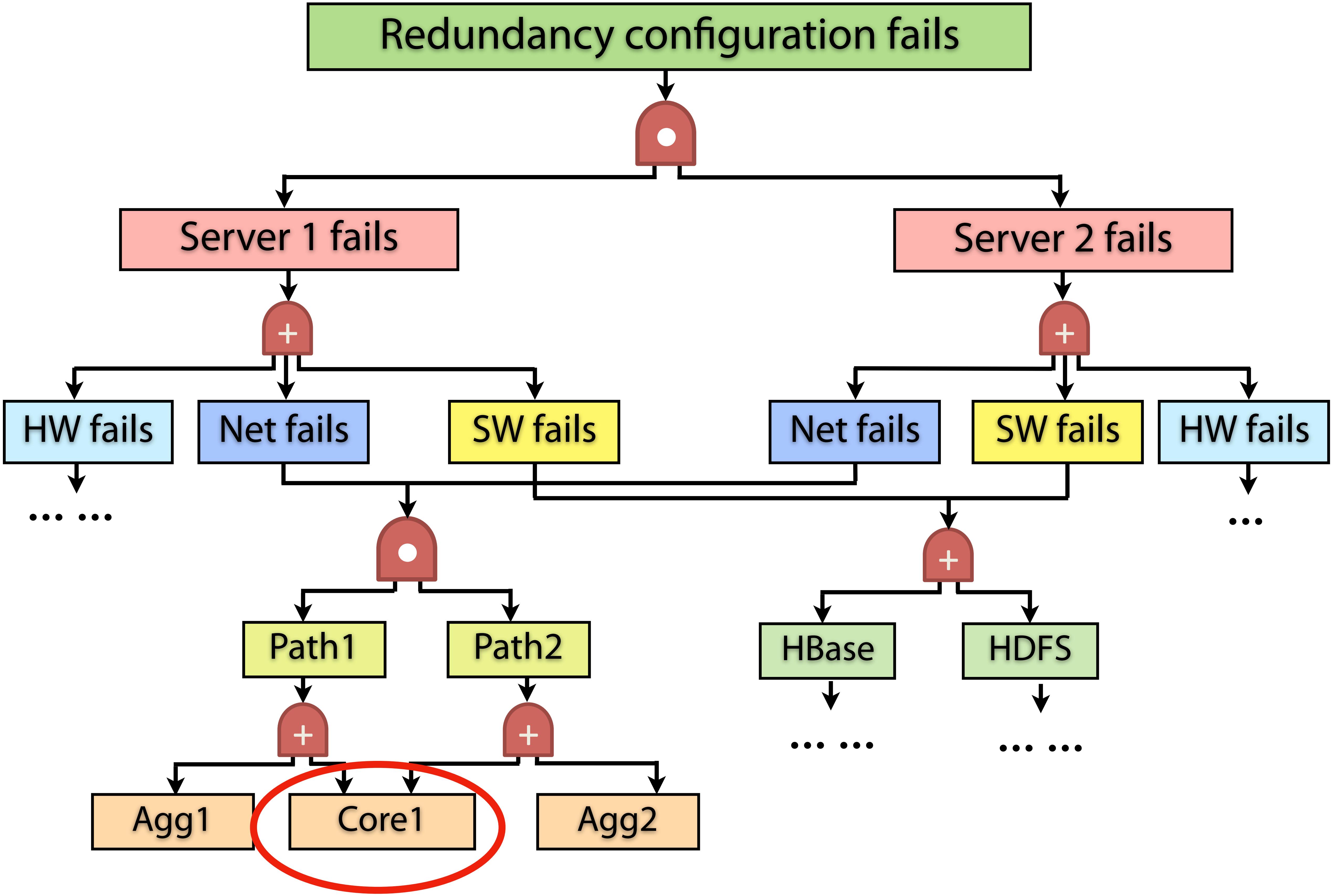
Redundancy configuration fails


Redundancy configuration fails
Server 1 fails
Server 2 fails
AND gate: all the sublayer nodes fail, the upper layer node fails

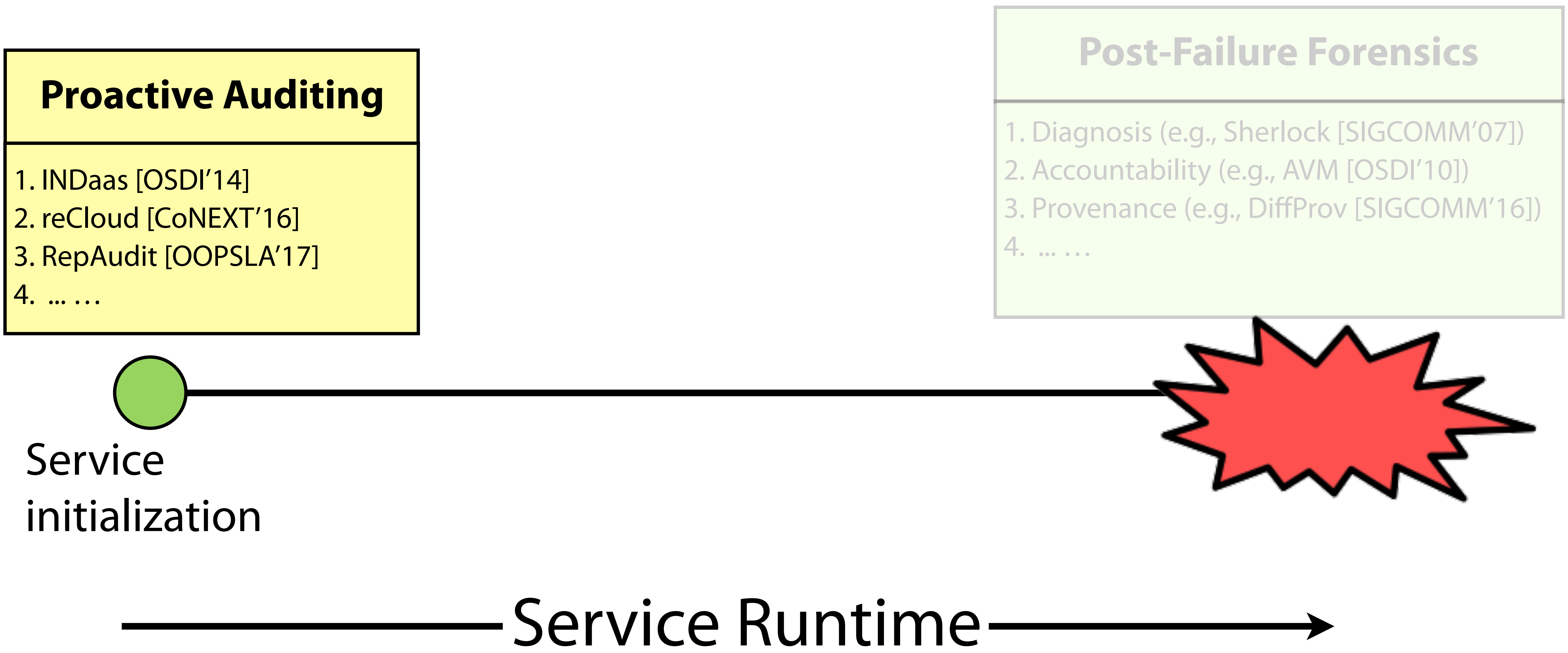


OR gate: one of the sublayer nodes fails,
the upper layer node fails


Redundancy configuration fails
Server 1 fails
Server 2 fails
HW fails
Net fails
SW fails
Net fails
SW fails
HW fails
Path1
Path2
HBase
HDFS
Agg1
Core1
Agg2


Redundancy configuration fails
Server 1 fails
Server 2 fails
HW fails
Net fails
SW fails
Net fails
SW fails
HW fails
Path1
Path2
HBase
HDFS
Agg1
Core1
Agg2

State of the Art



Correlated Failure Risks in Updates

Proactive Auditing

1. I
2. r
3. R
4. ..

Se
in

**Azure global outag
mangled domain re**

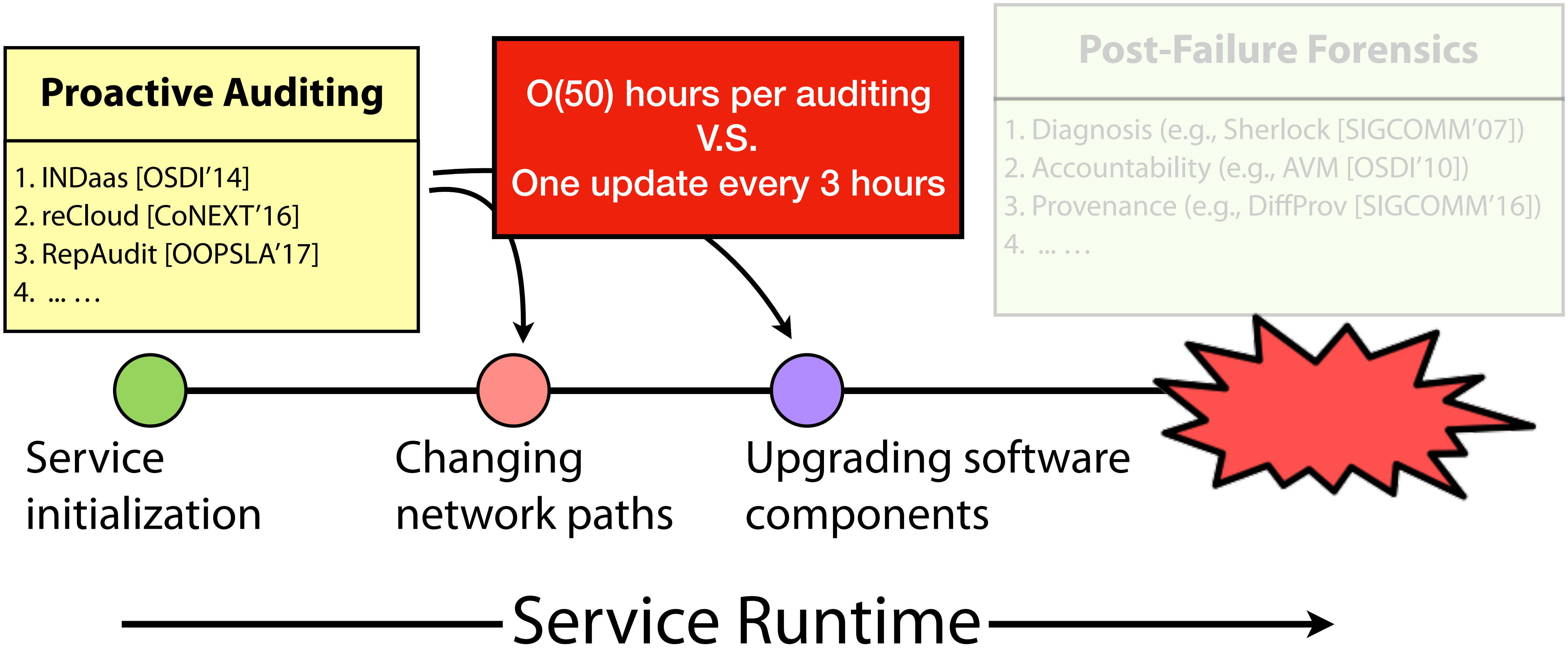
Microsoft Azure and Office 365 downed by DNS configuration blunder

Google Outage in Eastern U.S. Affecting Gmail, YouTube, and More

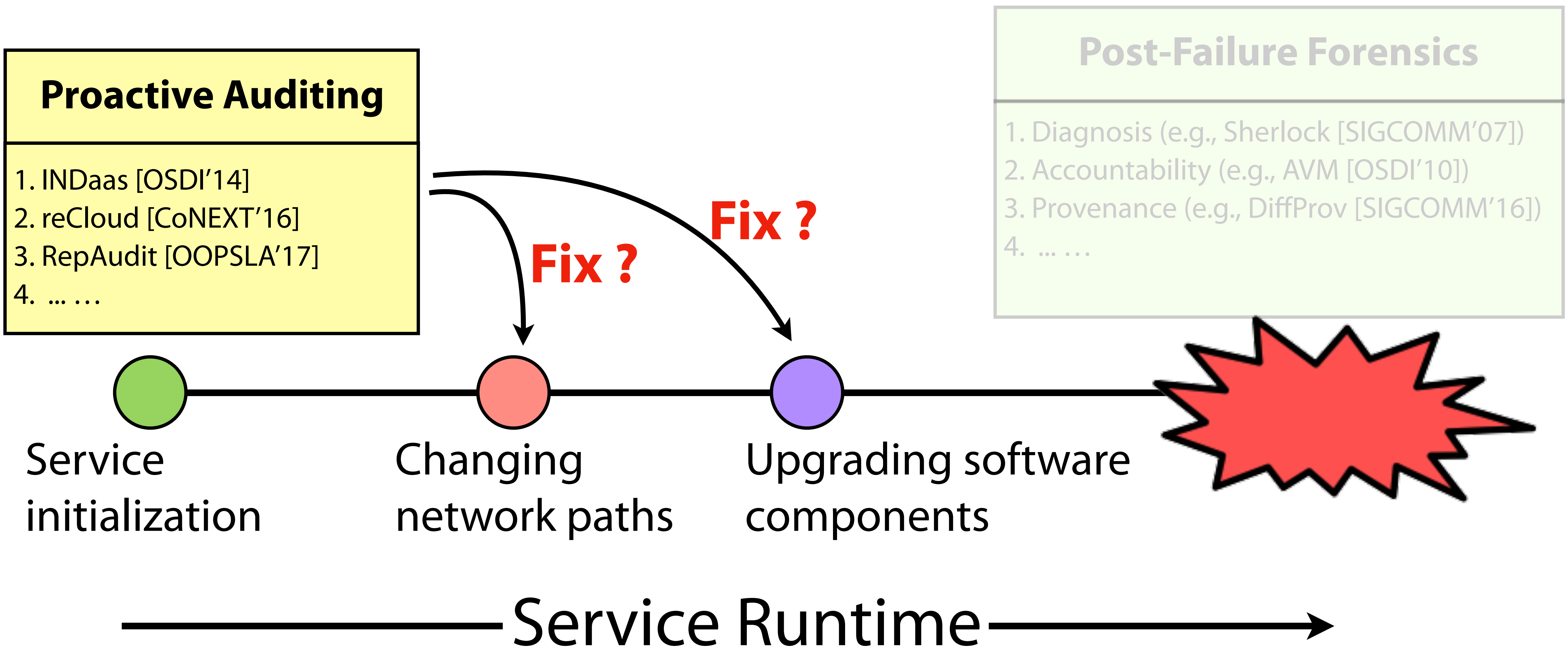
By Lawrence Abrams   June 2, 2019   06:38 PM   2

Benjamin Treynor Sloss, Google's VP of engineering, explained that the root cause of last Sunday's outage was a configuration change for a small group of servers in one region being wrongly applied to a larger number of servers across several neighboring regions.

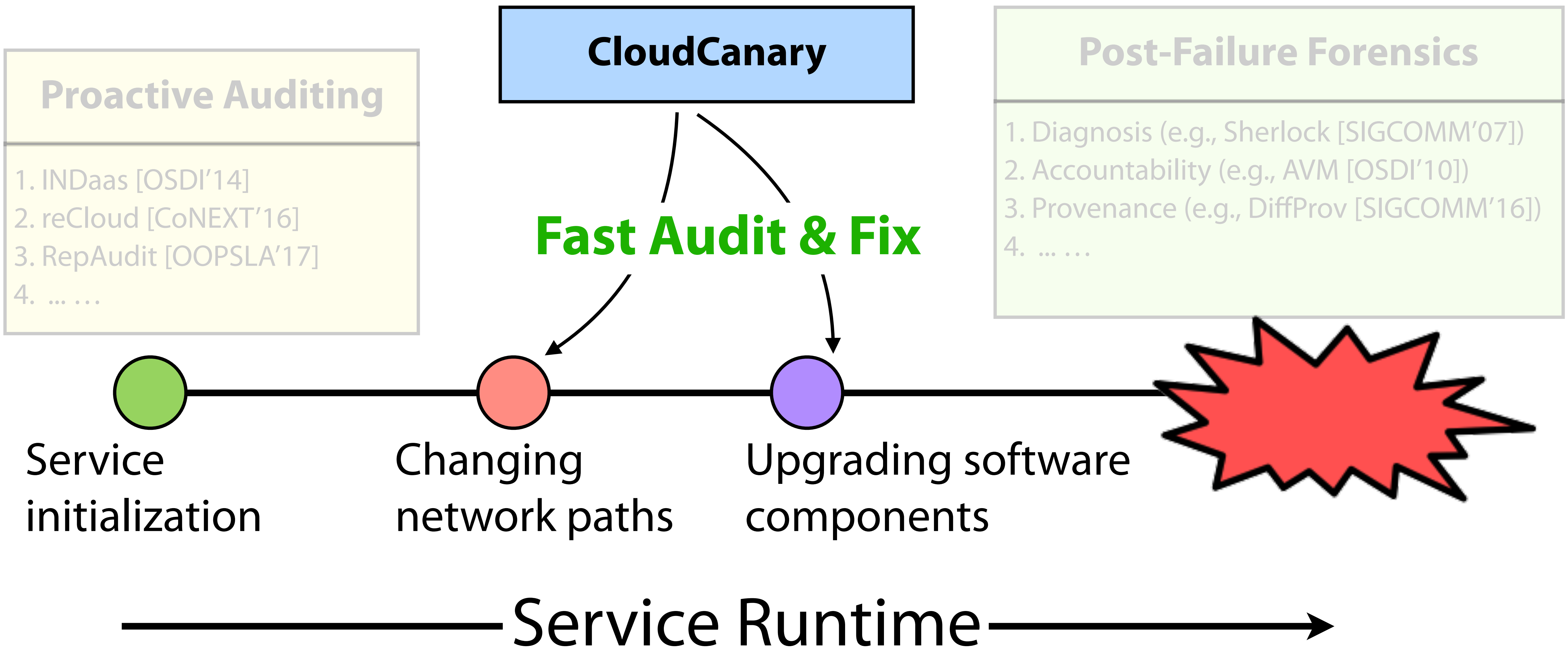
Problem 1: Inefficient Auditing in Updates



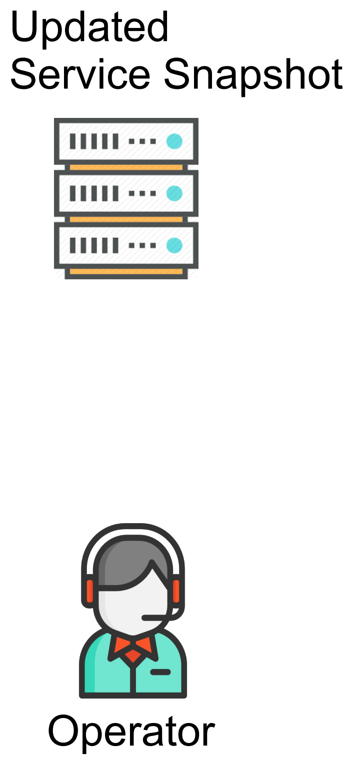
Problem 2: Lack of fixing risks



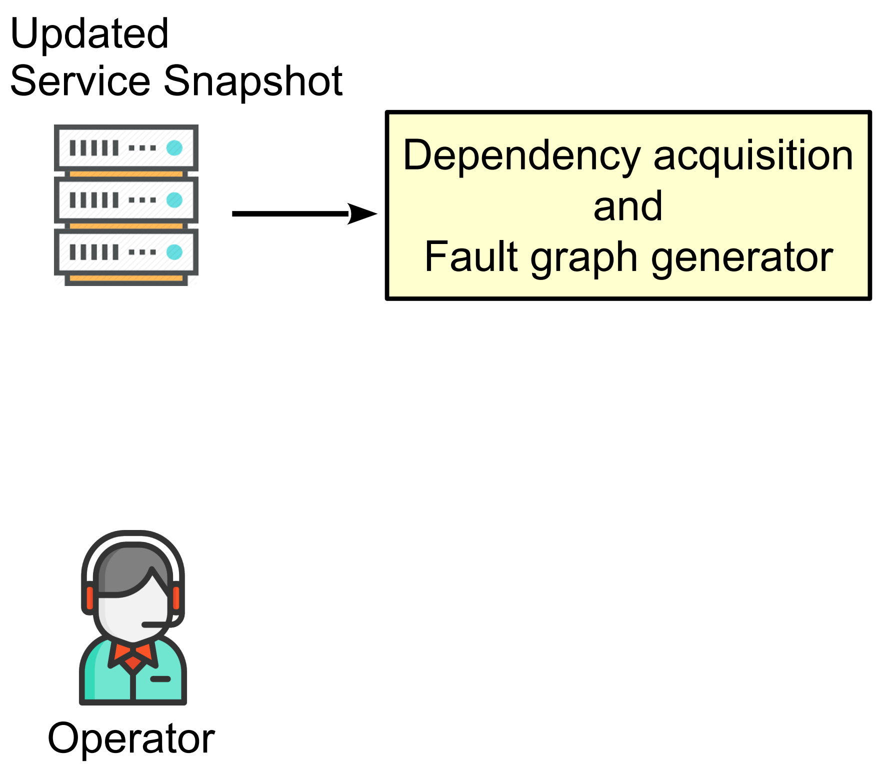
Our Contribution



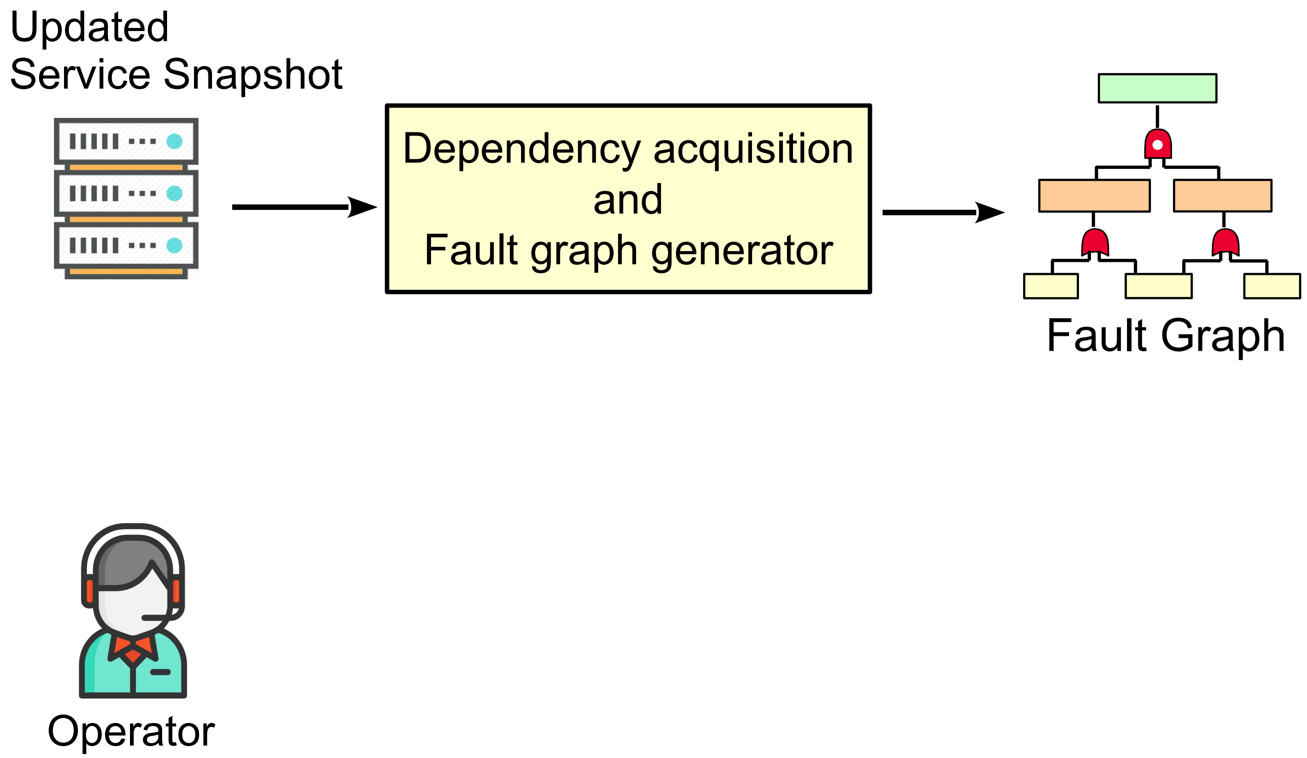
CloudCanary's Workflow



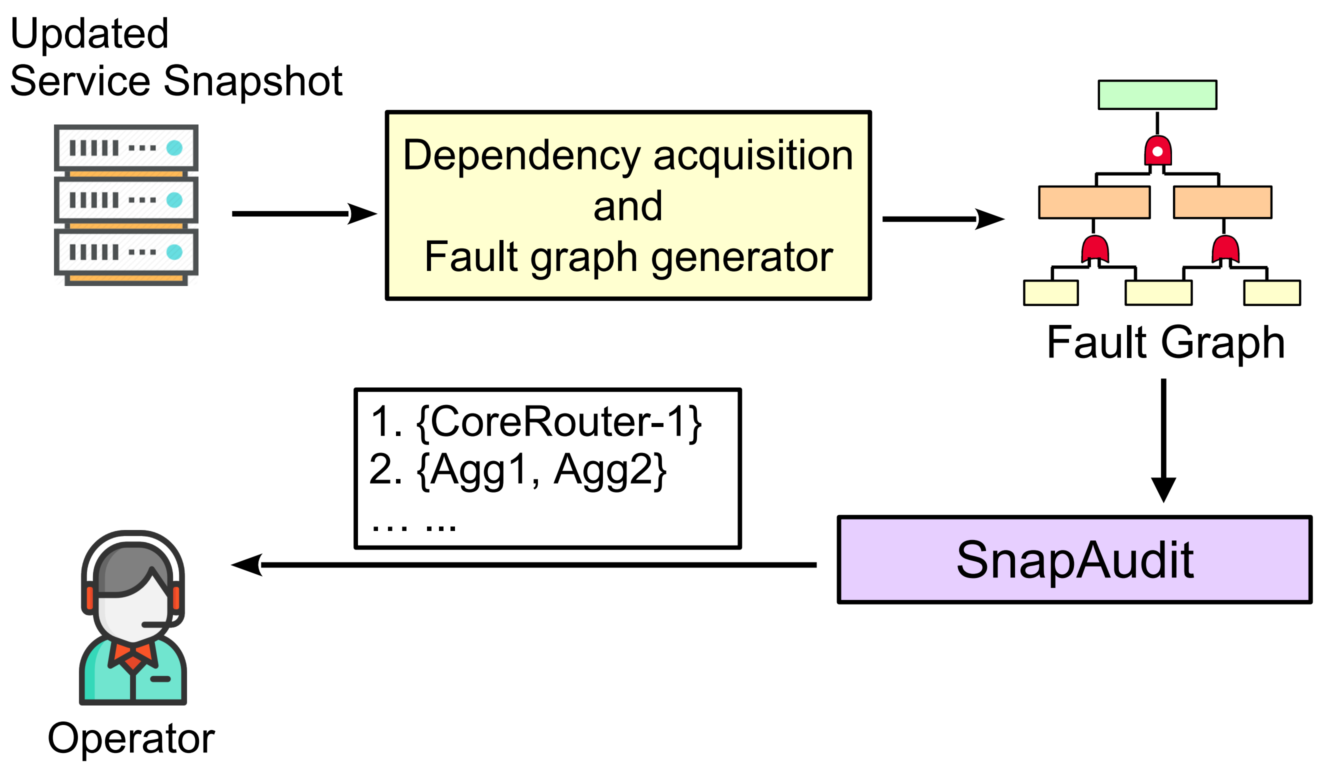
CloudCanary's Workflow



CloudCanary's Workflow

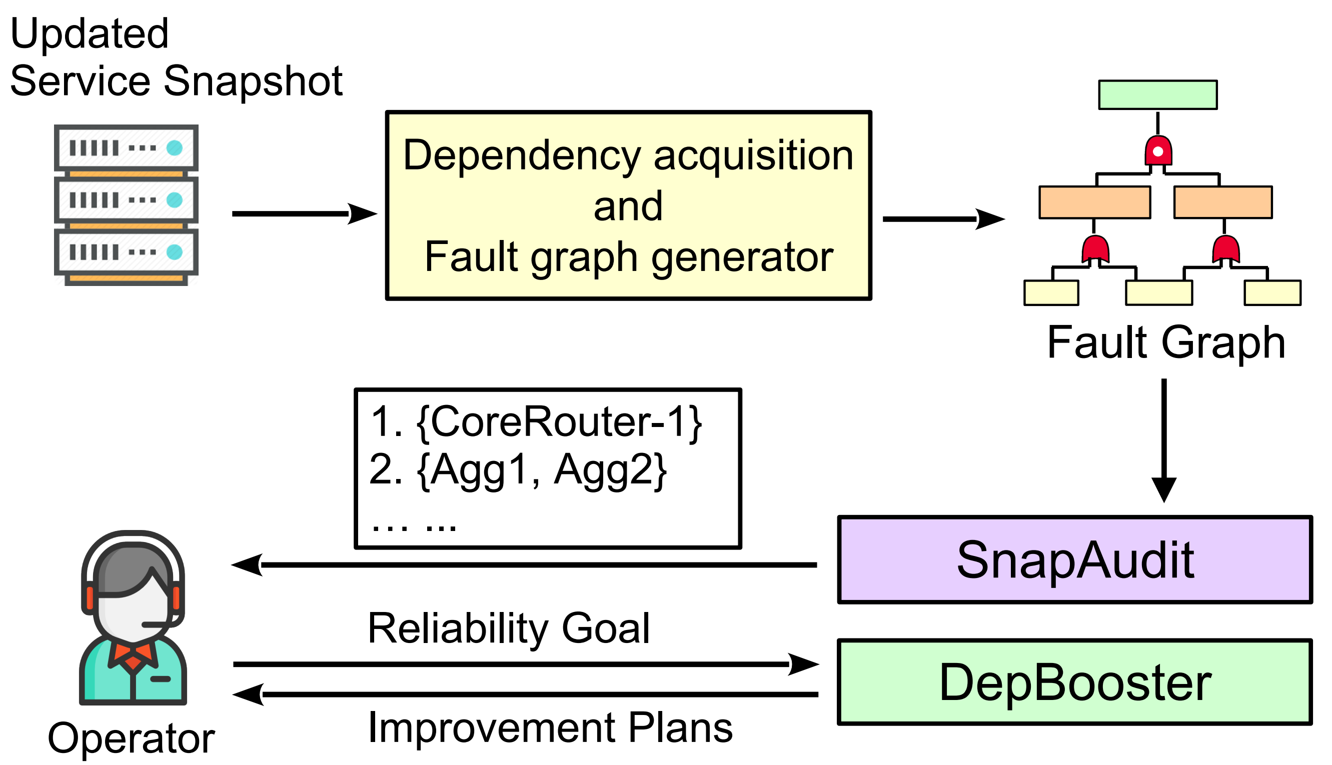


CloudCanary's Workflow



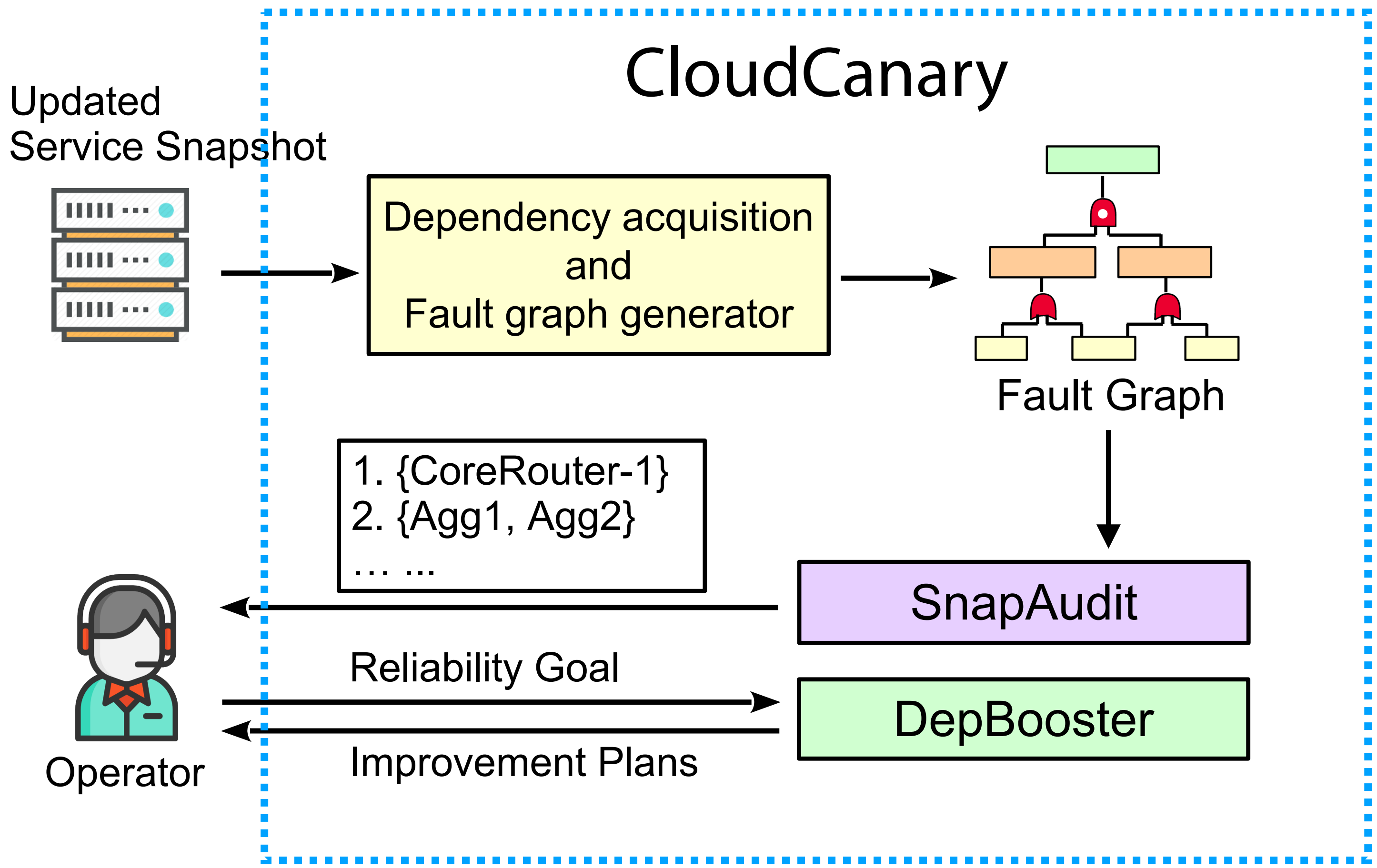
- Challenge 1: SnapAudit

CloudCanary's Workflow



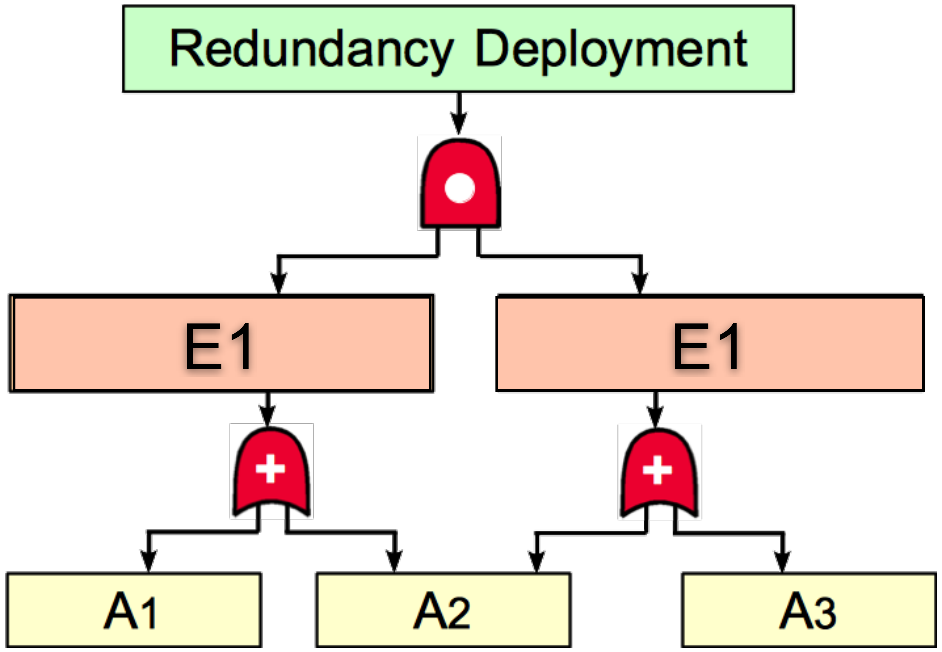
- Challenge 1: SnapAudit
- Challenge 2: DepBooster

CloudCanary's Workflow

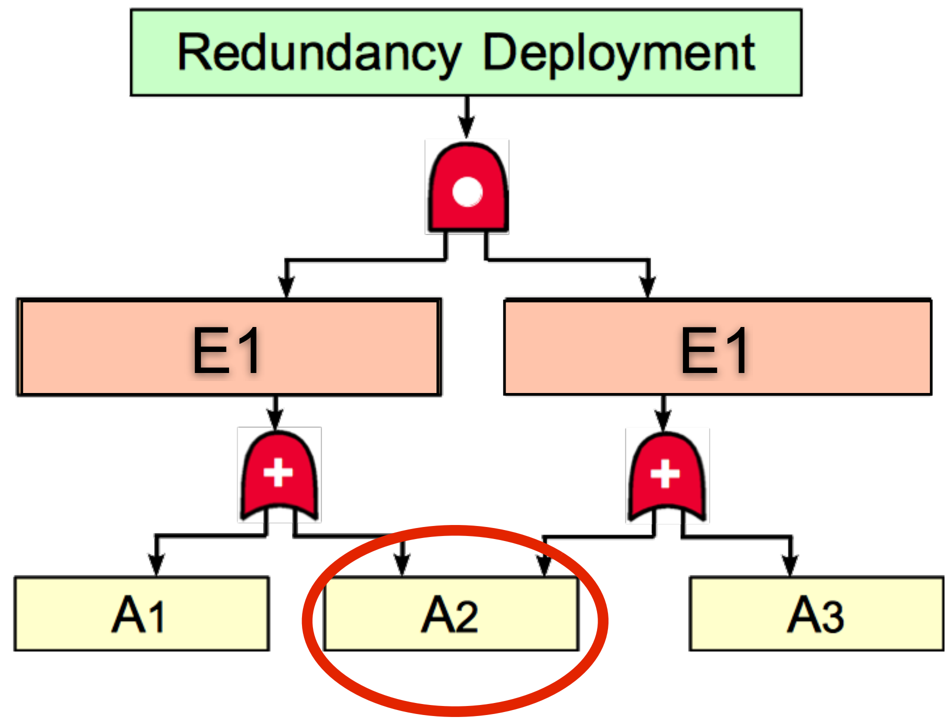


- Challenge 1: SnapAudit
- Challenge 2: DepBooster

A Fault Graph

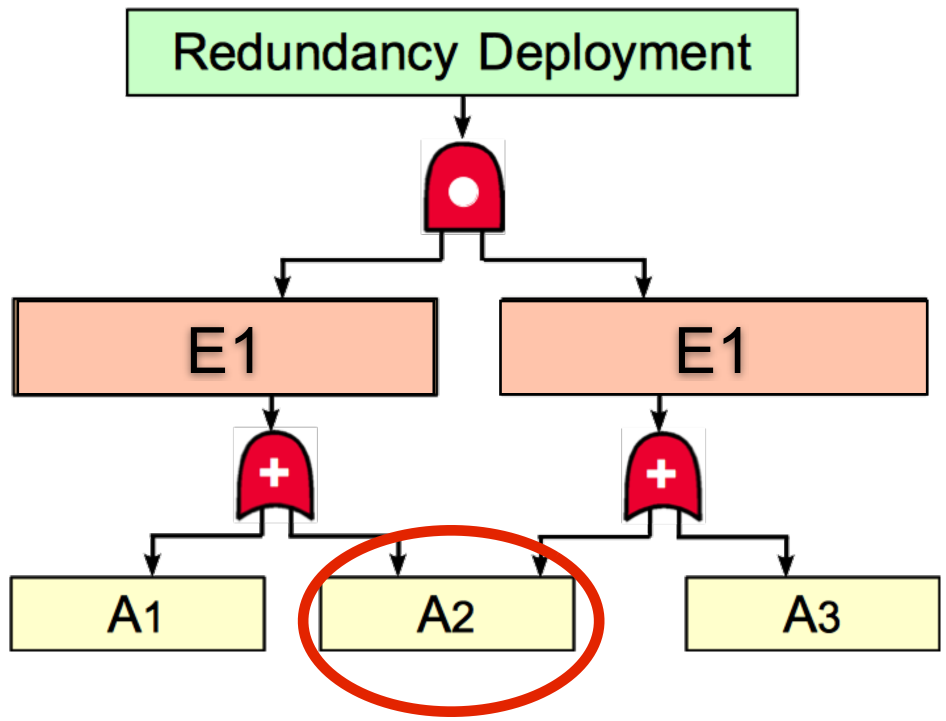


Risk Groups in Fault Graphs



- A risk group means a set of leaf nodes whose simultaneous failures lead to the failure of root node

Risk Groups in Fault Graphs

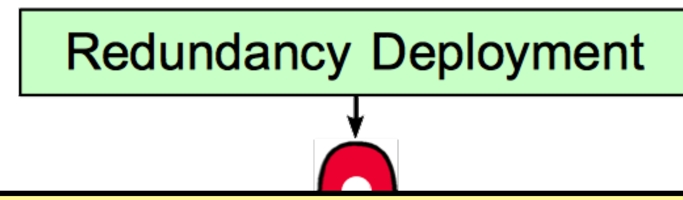


- A risk group means a set of leaf nodes whose simultaneous failures lead to the failure of root node

{A2} and {A1, A3} are risk groups

{A1} or {A3} is not risk group

Risk Groups in Fault Graphs



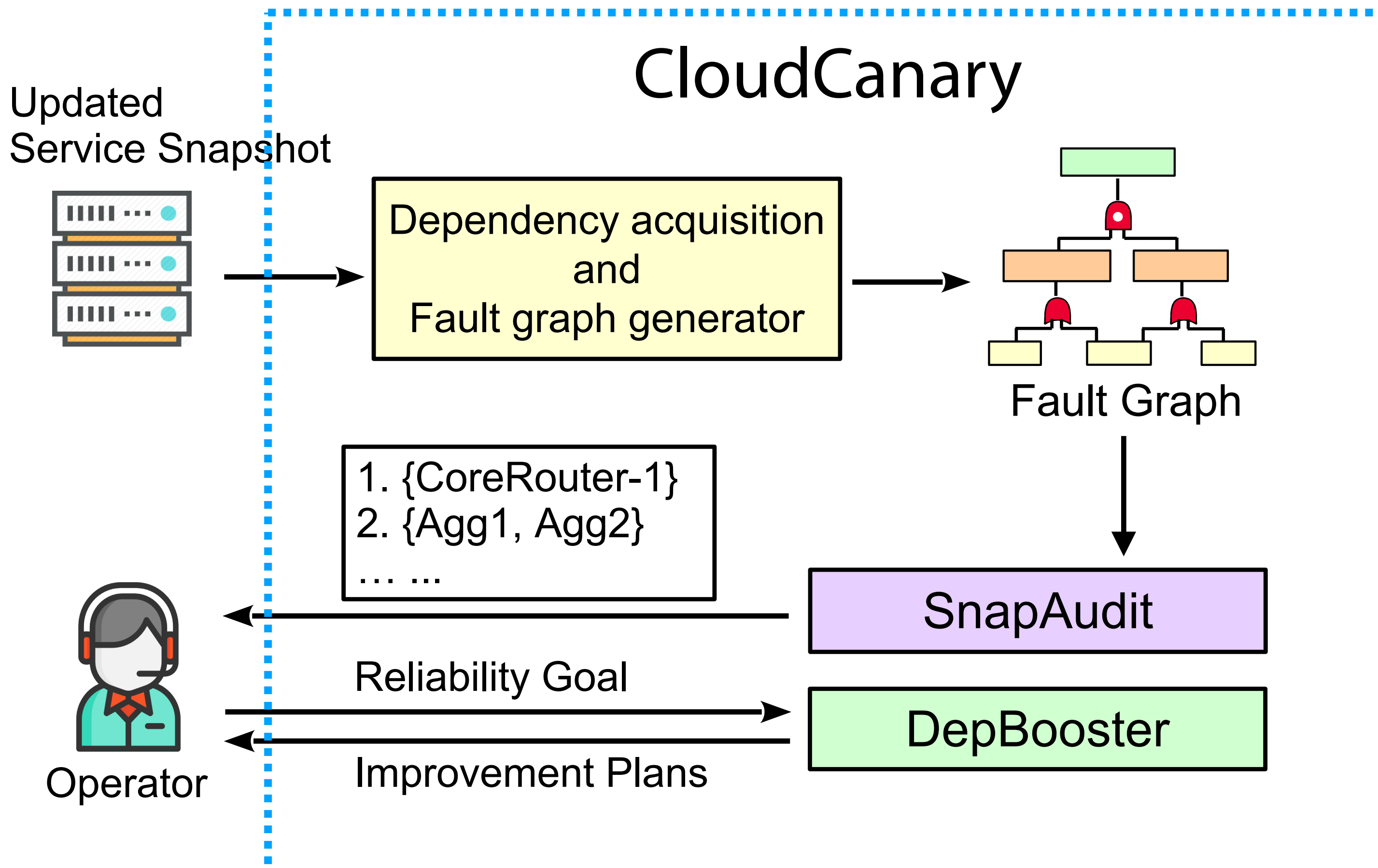
Identifying correlated failure risks can be reduced to the problem of finding risk groups in the fault graph.

However, analyzing risk groups is NP-complete problem

{A2} and {A1, A3} are risk groups

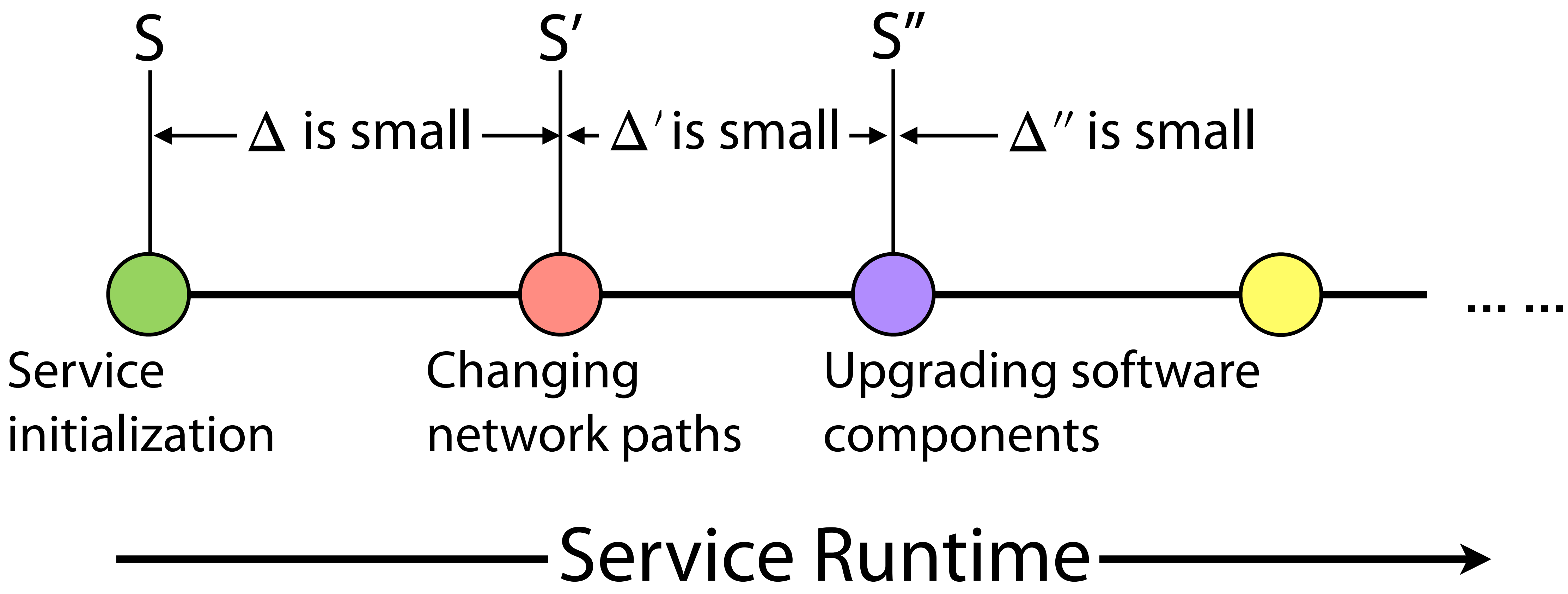
{A1} or {A3} is not risk group

CloudCanary's Workflow

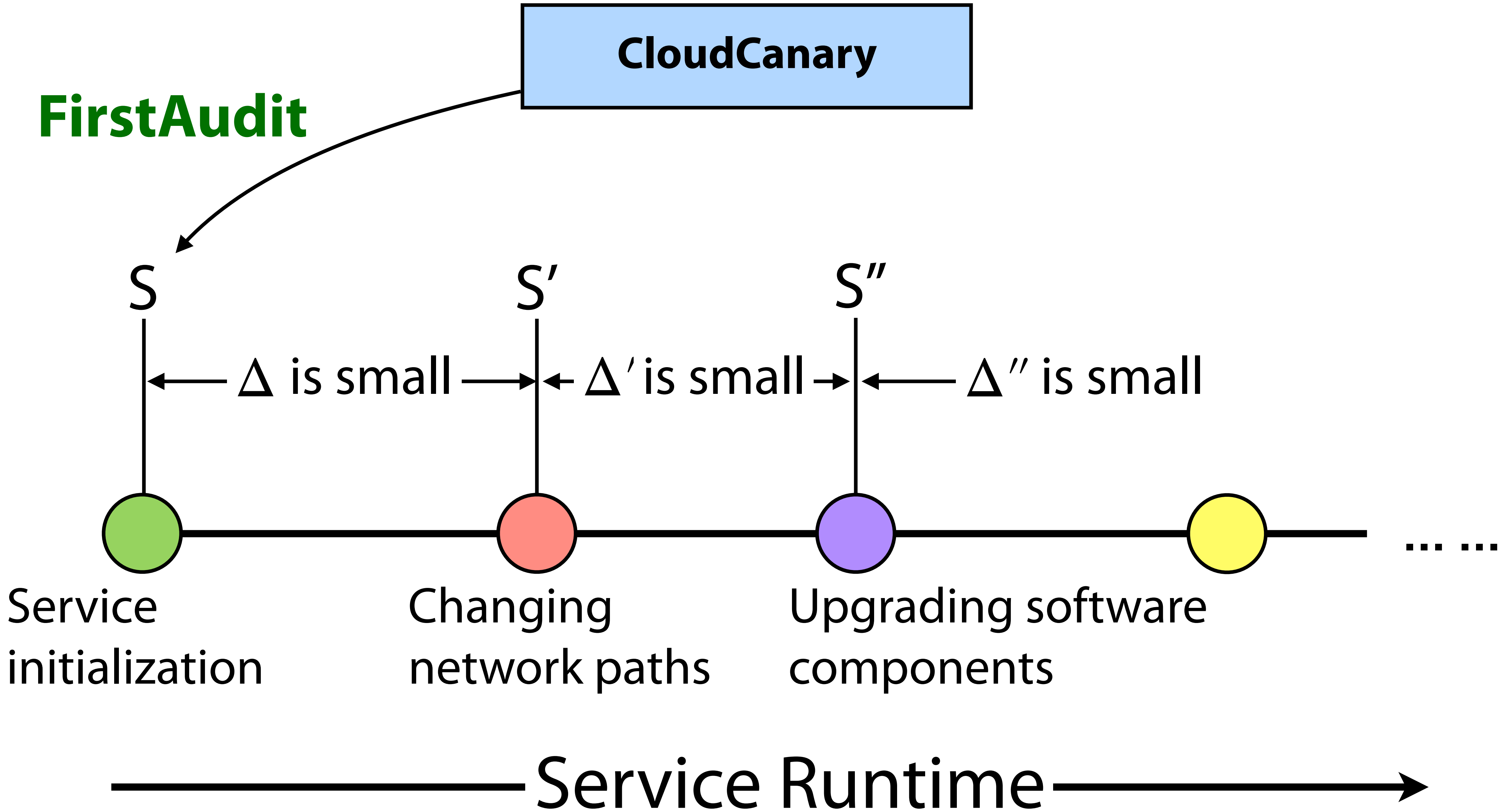


- Challenge 1: SnapAudit
- Challenge 2: DepBooster

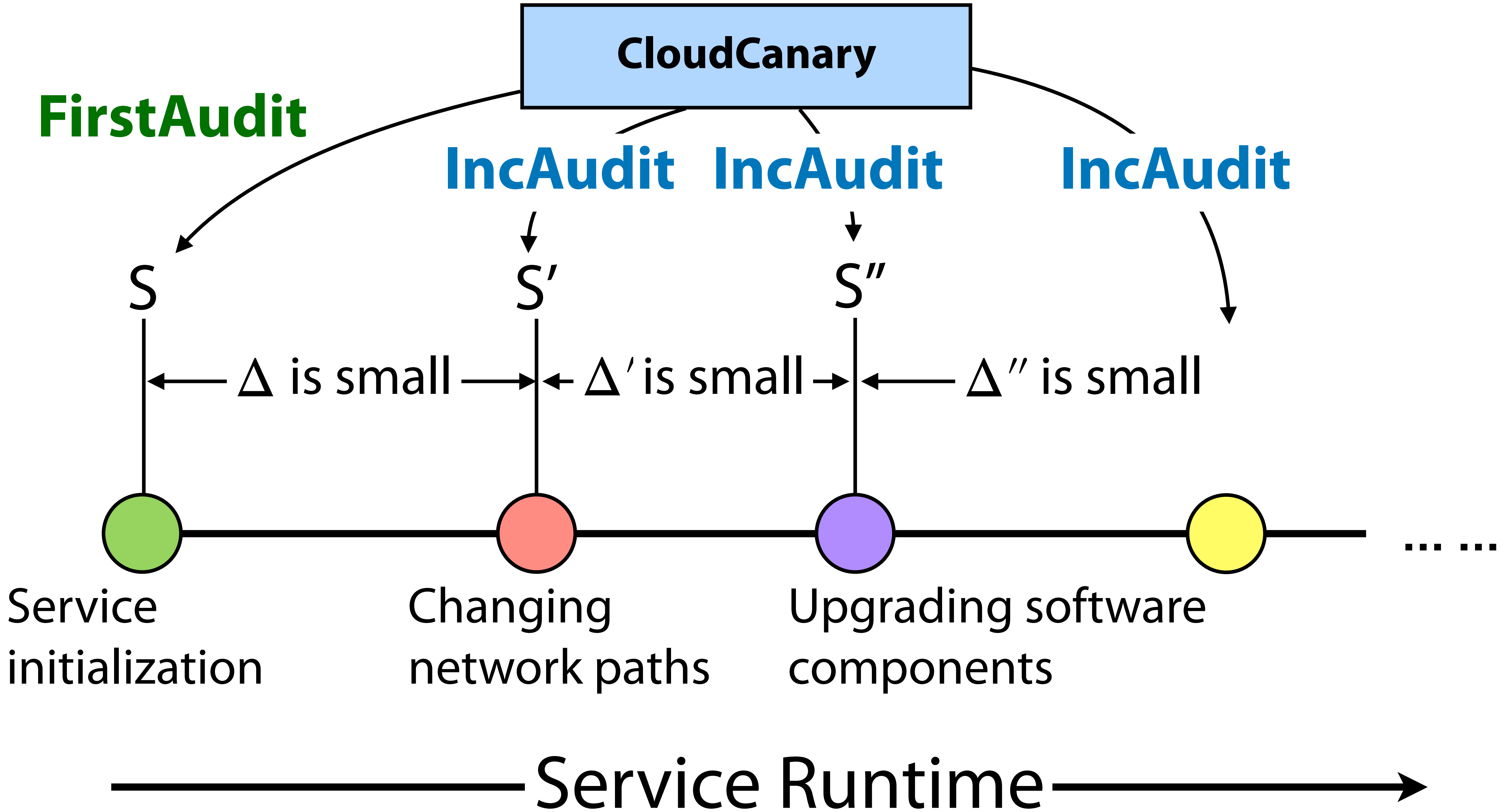
The Insight of SnapAudit



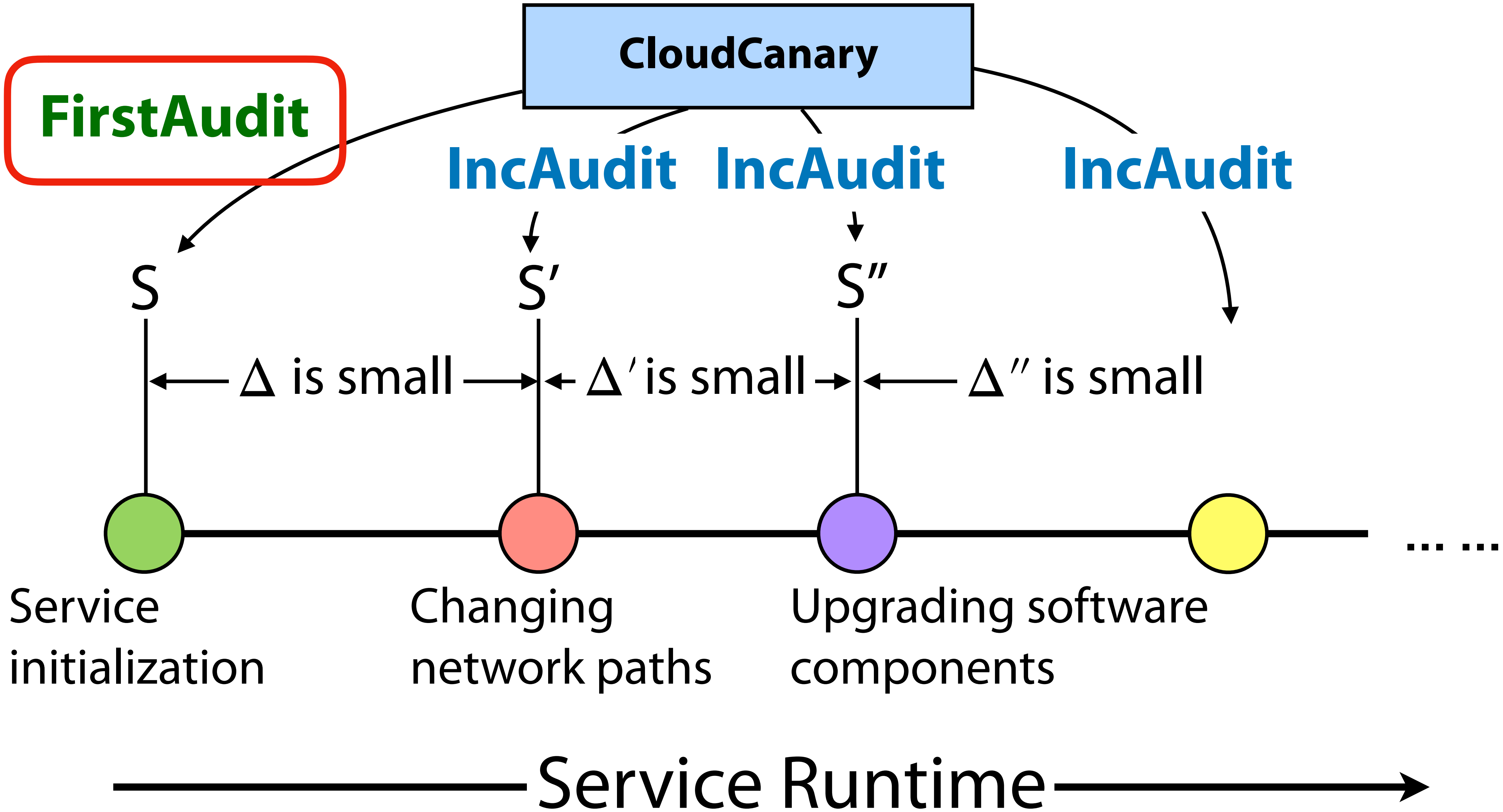
The Insight of SnapAudit



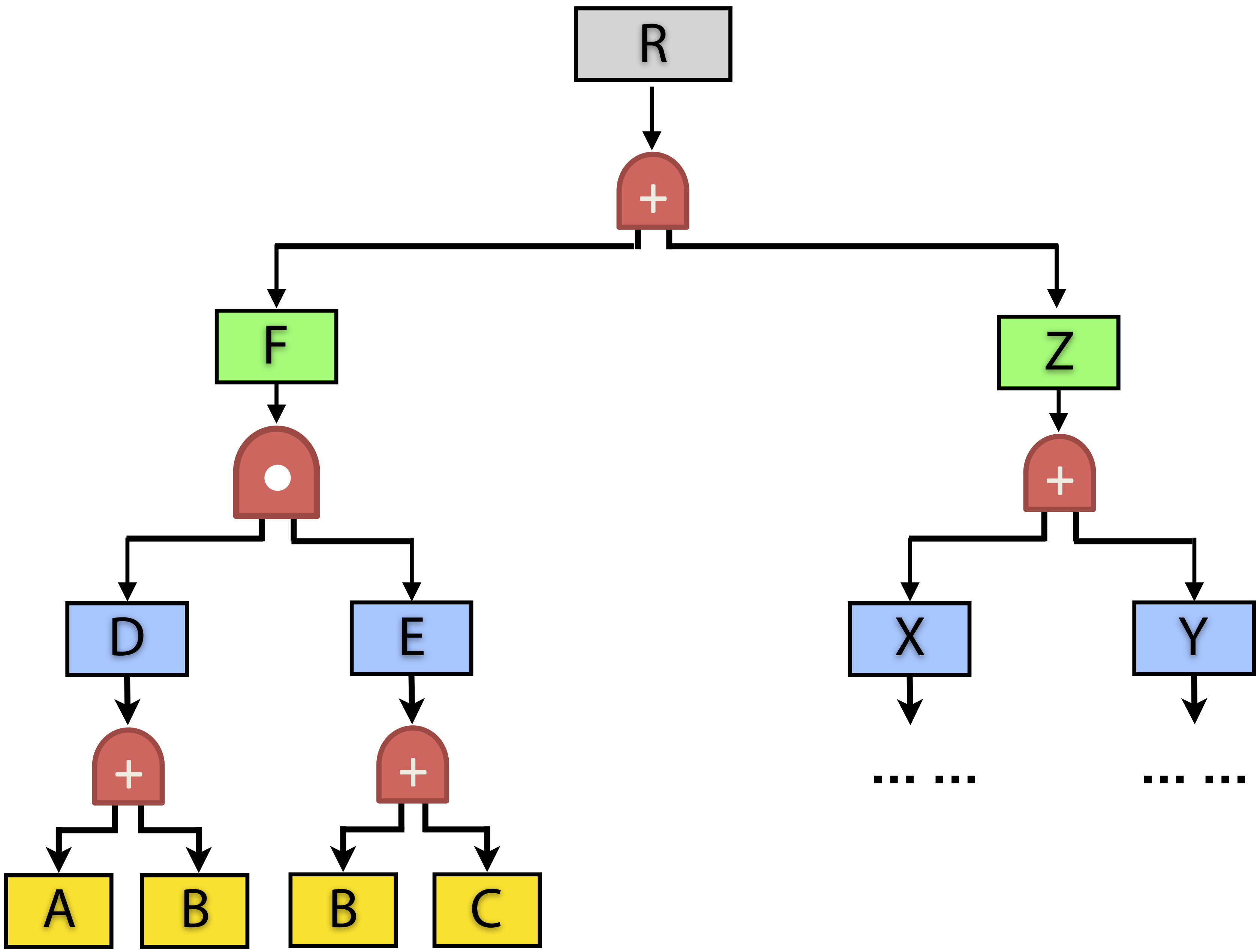
The Insight of SnapAudit



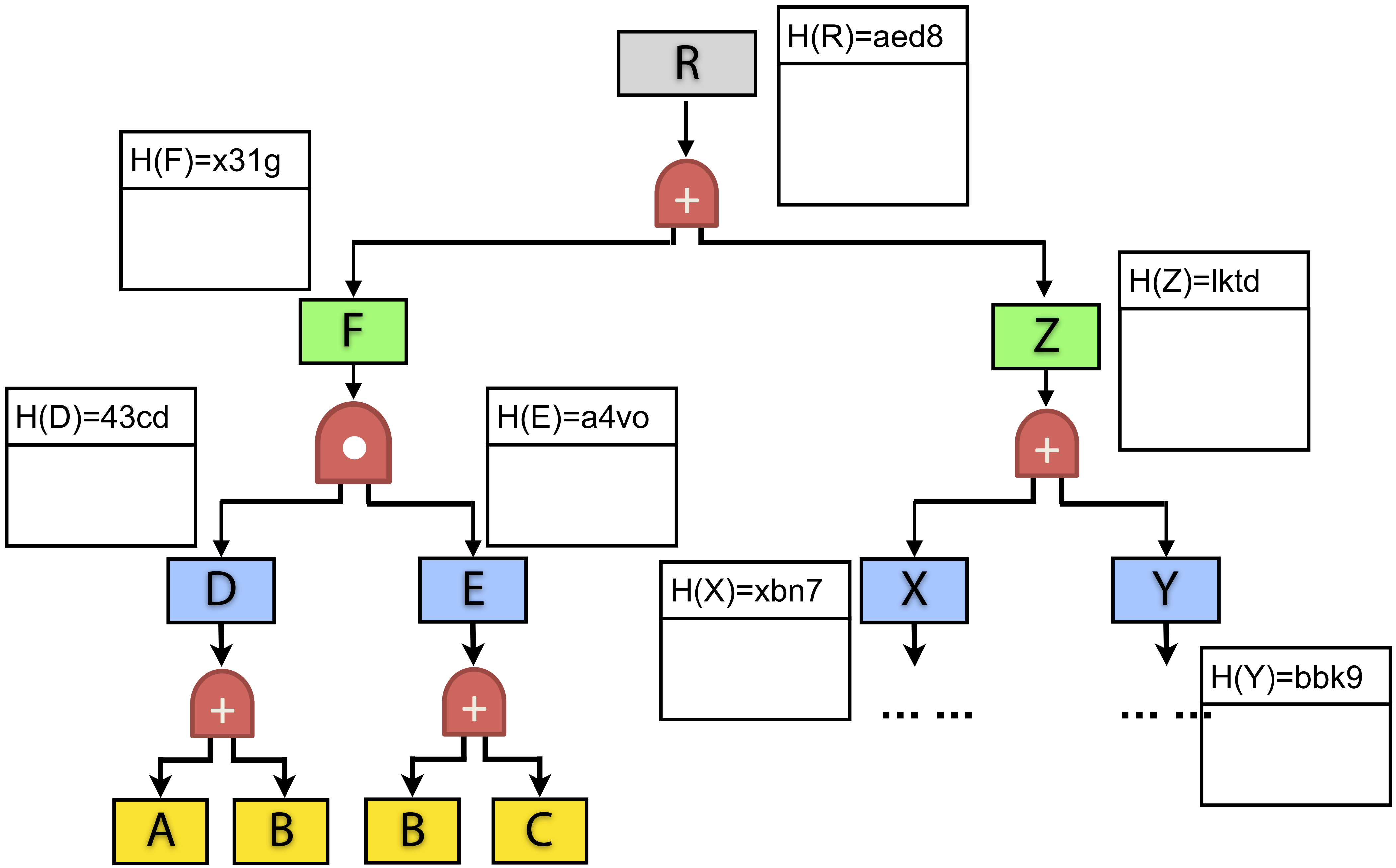
SnapAudit: FirstAudit & IncAudit



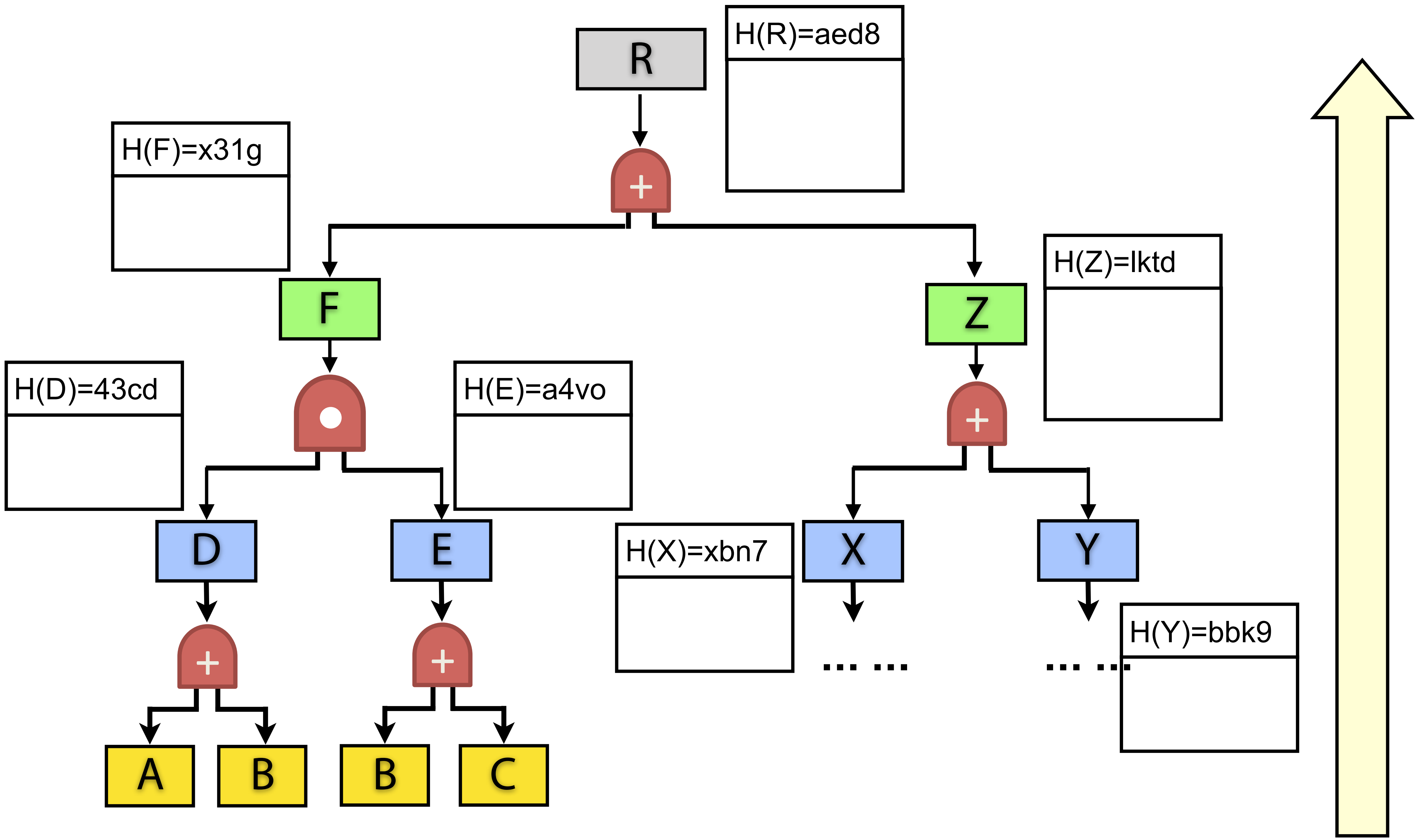
FirstAudit Primitive



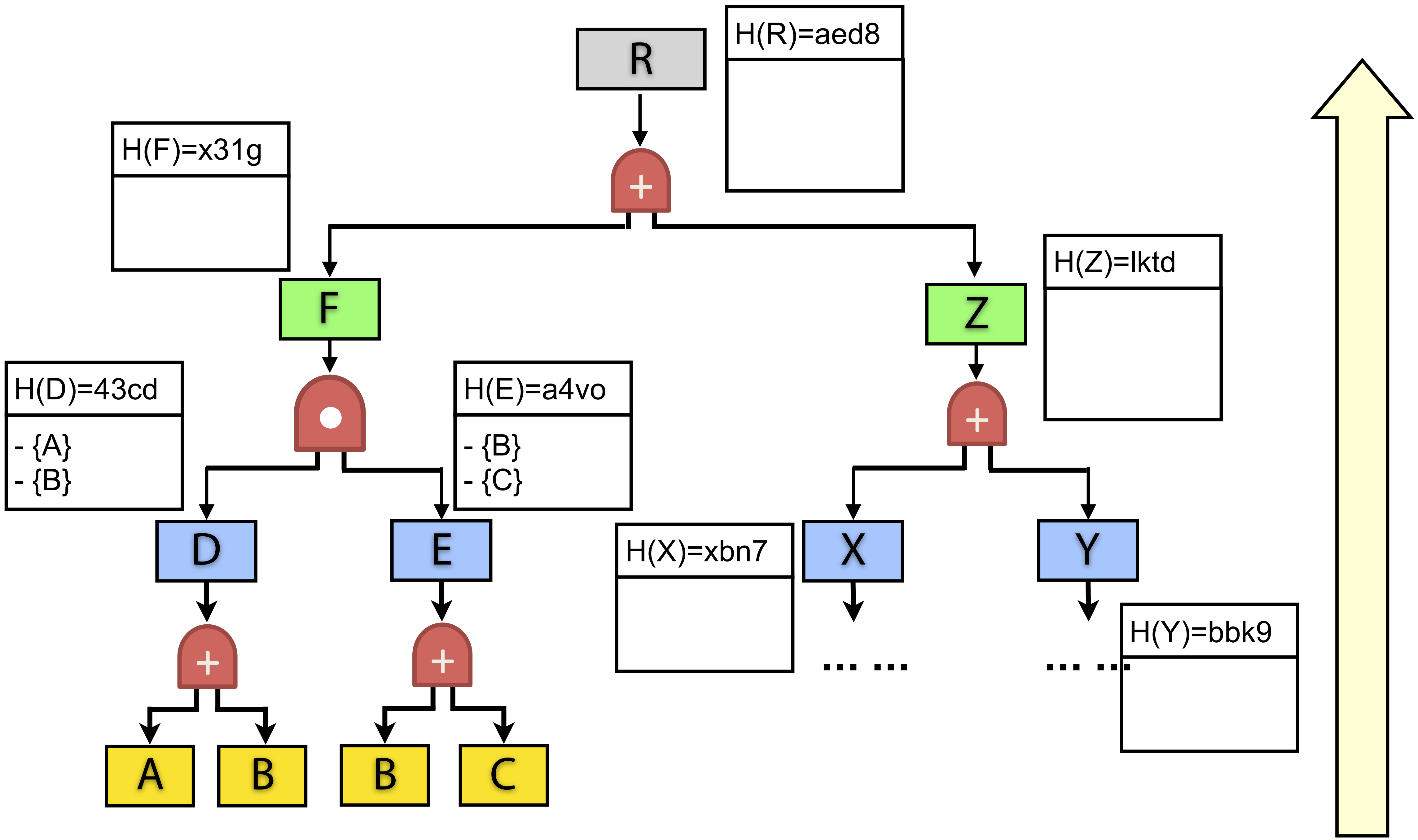
FirstAudit Primitive



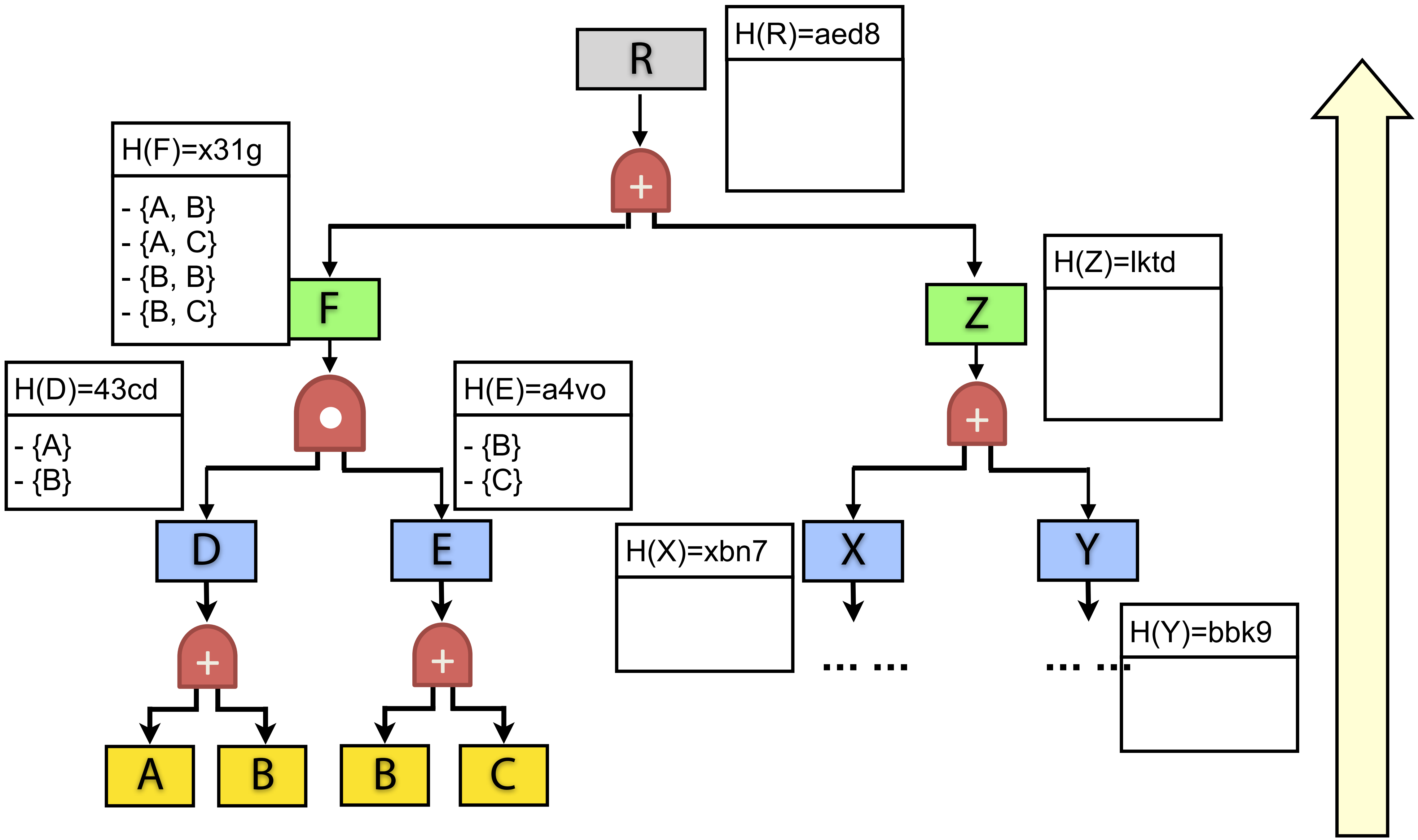
FirstAudit Primitive



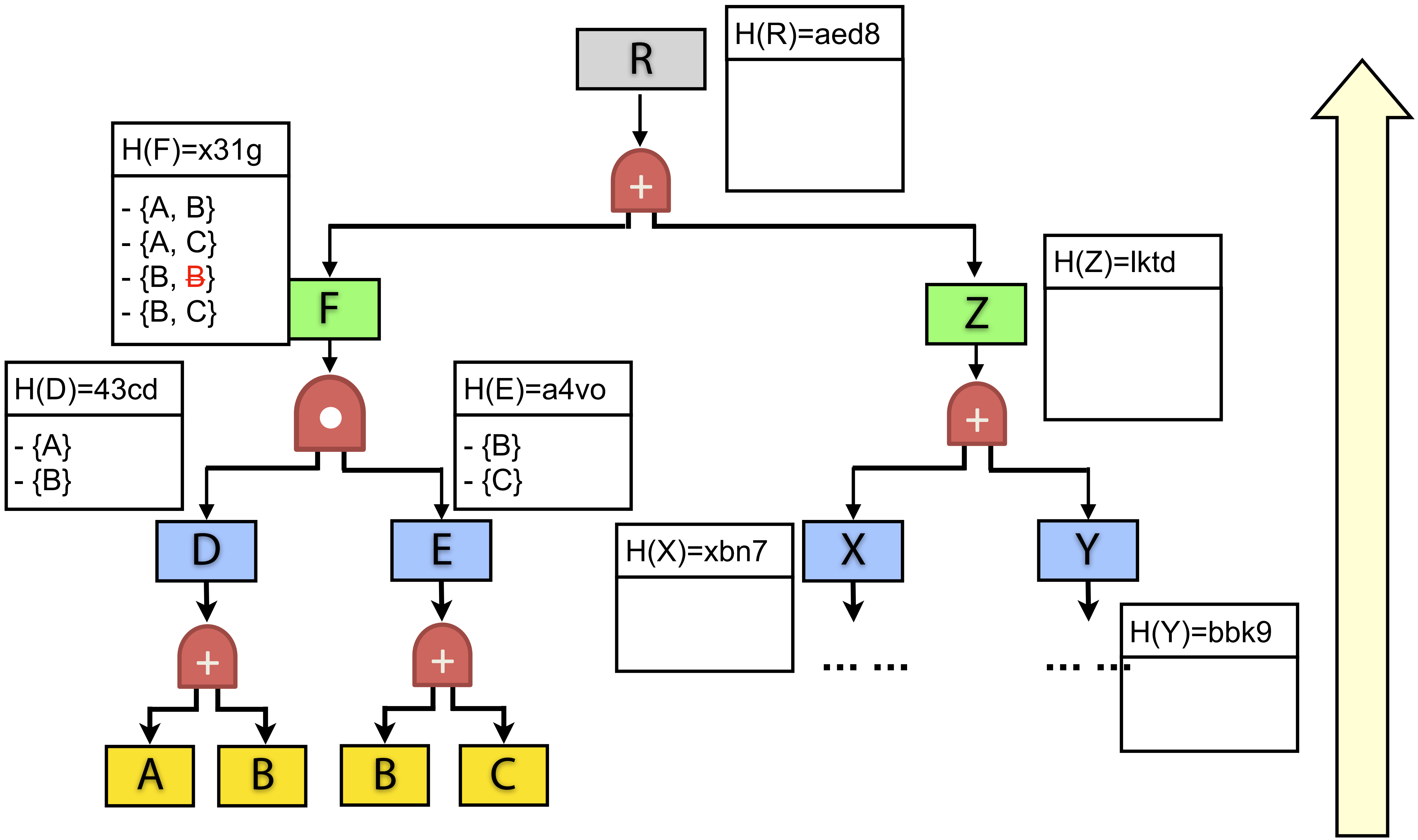
FirstAudit Primitive



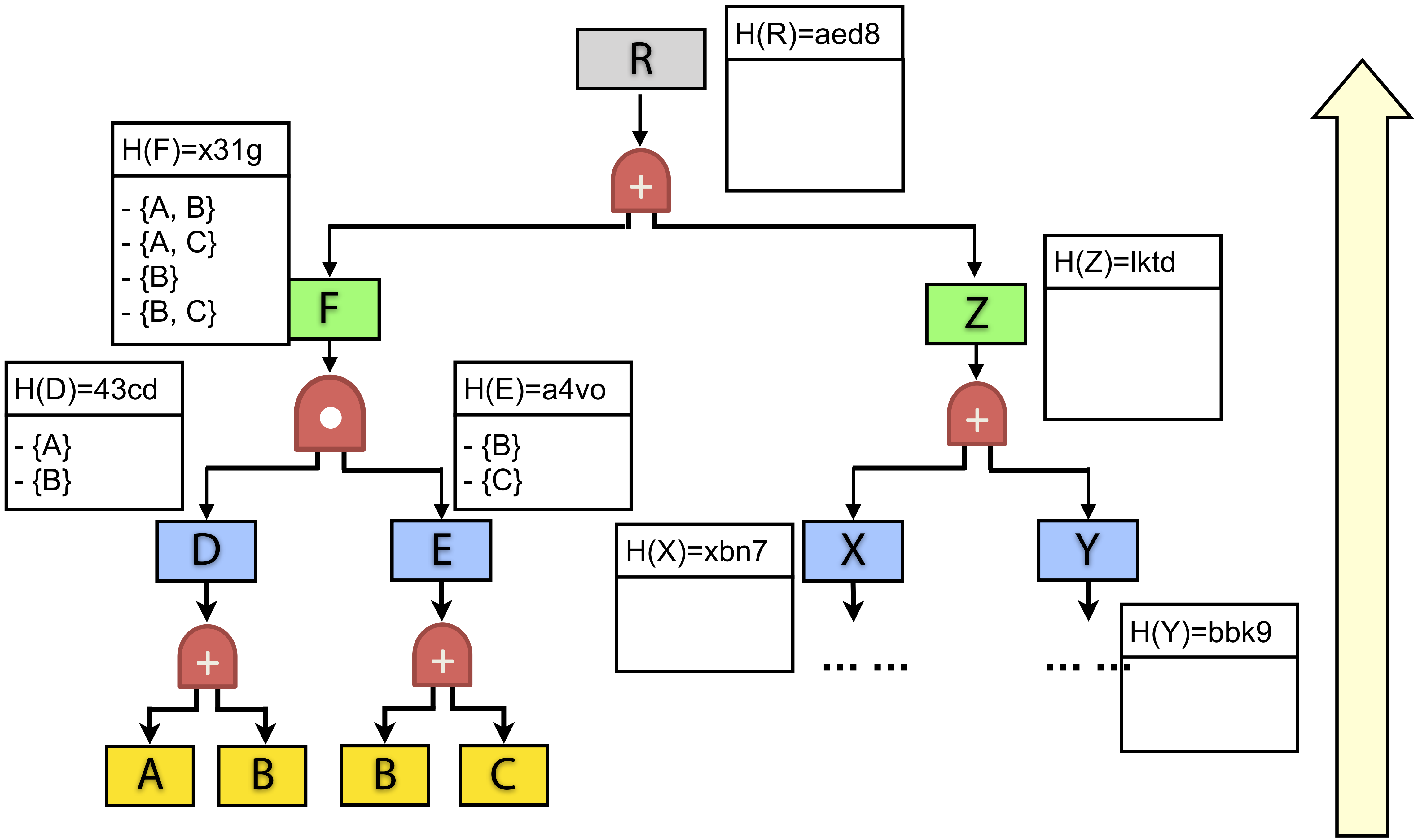
FirstAudit Primitive



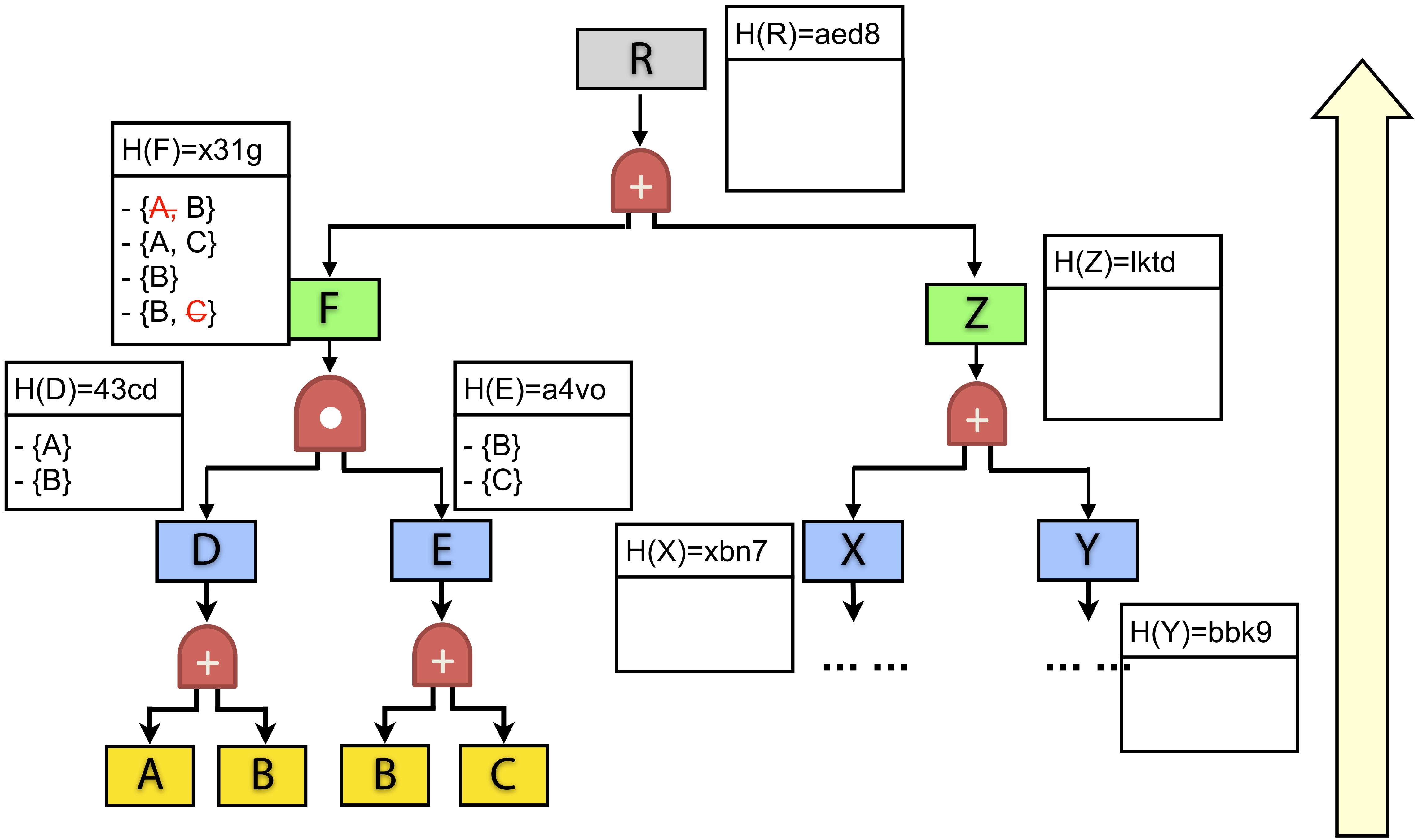
FirstAudit Primitive



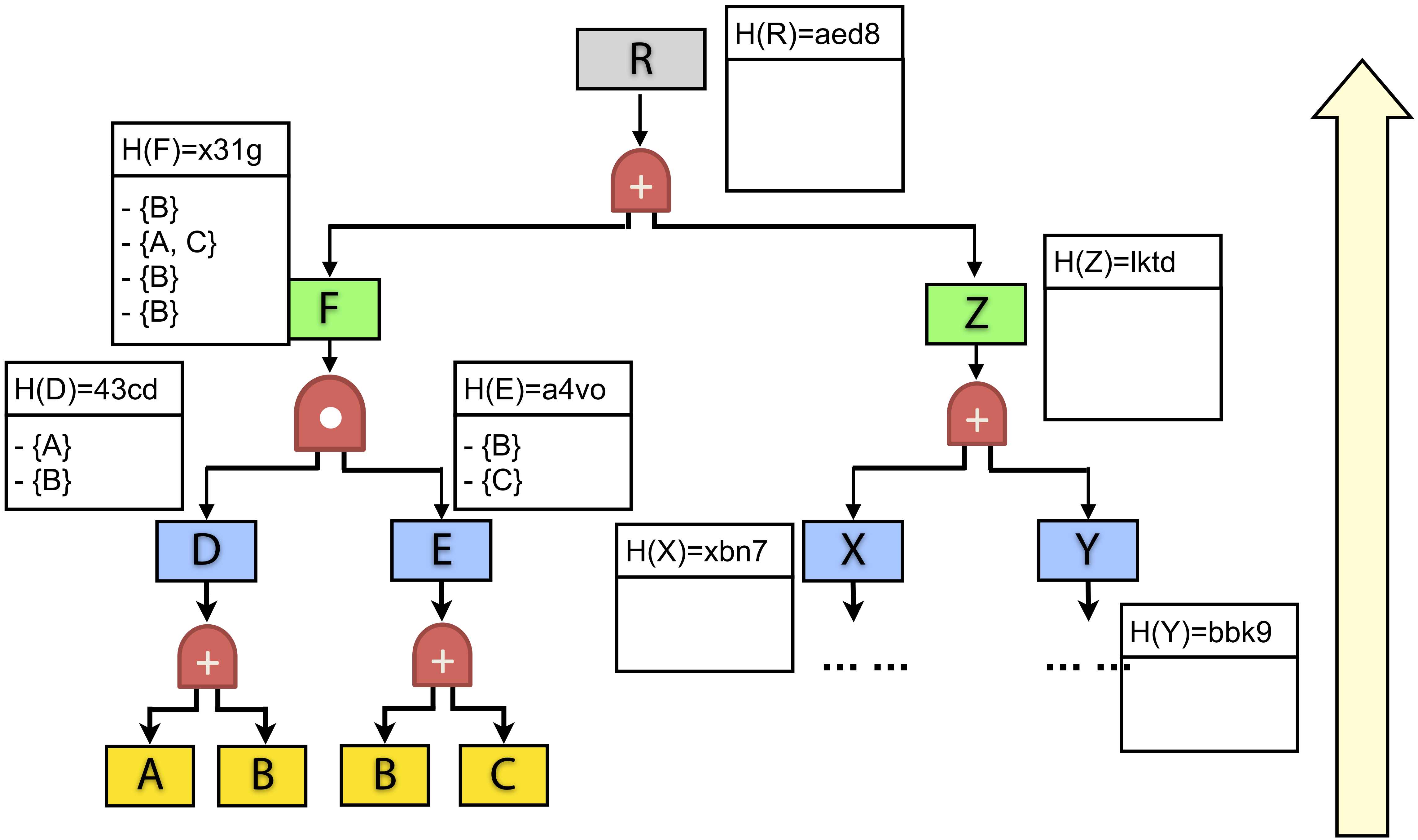
FirstAudit Primitive



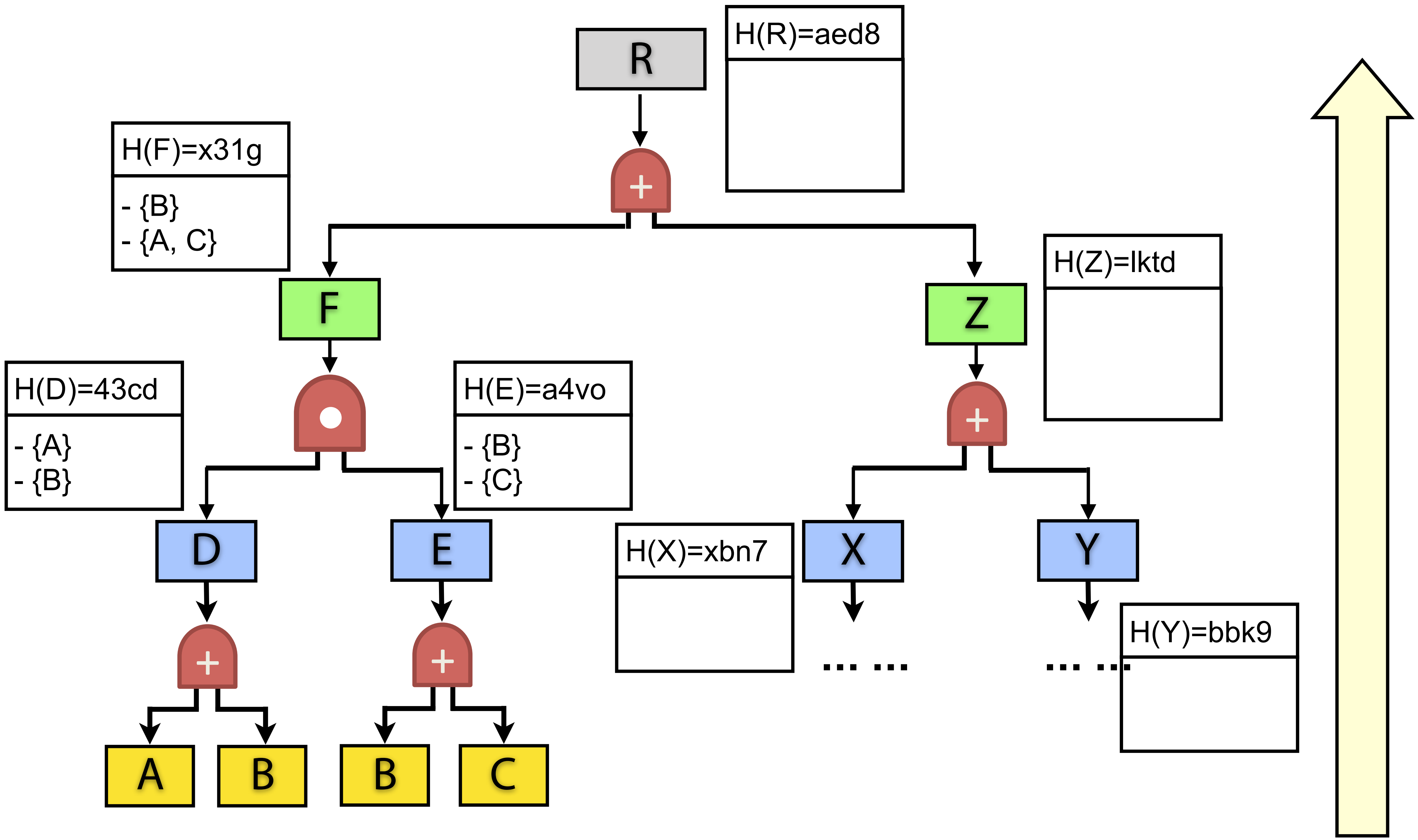
FirstAudit Primitive



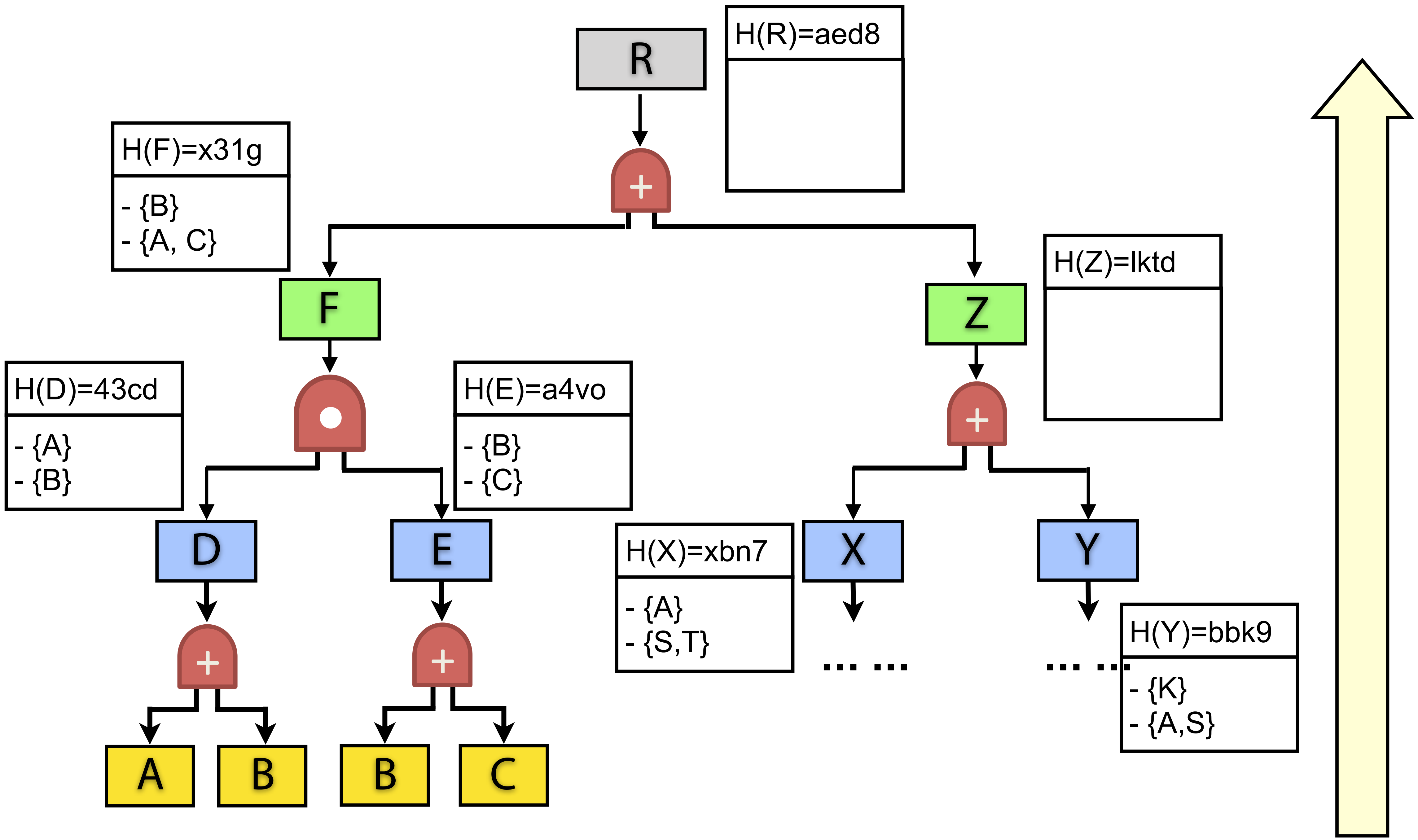
FirstAudit Primitive



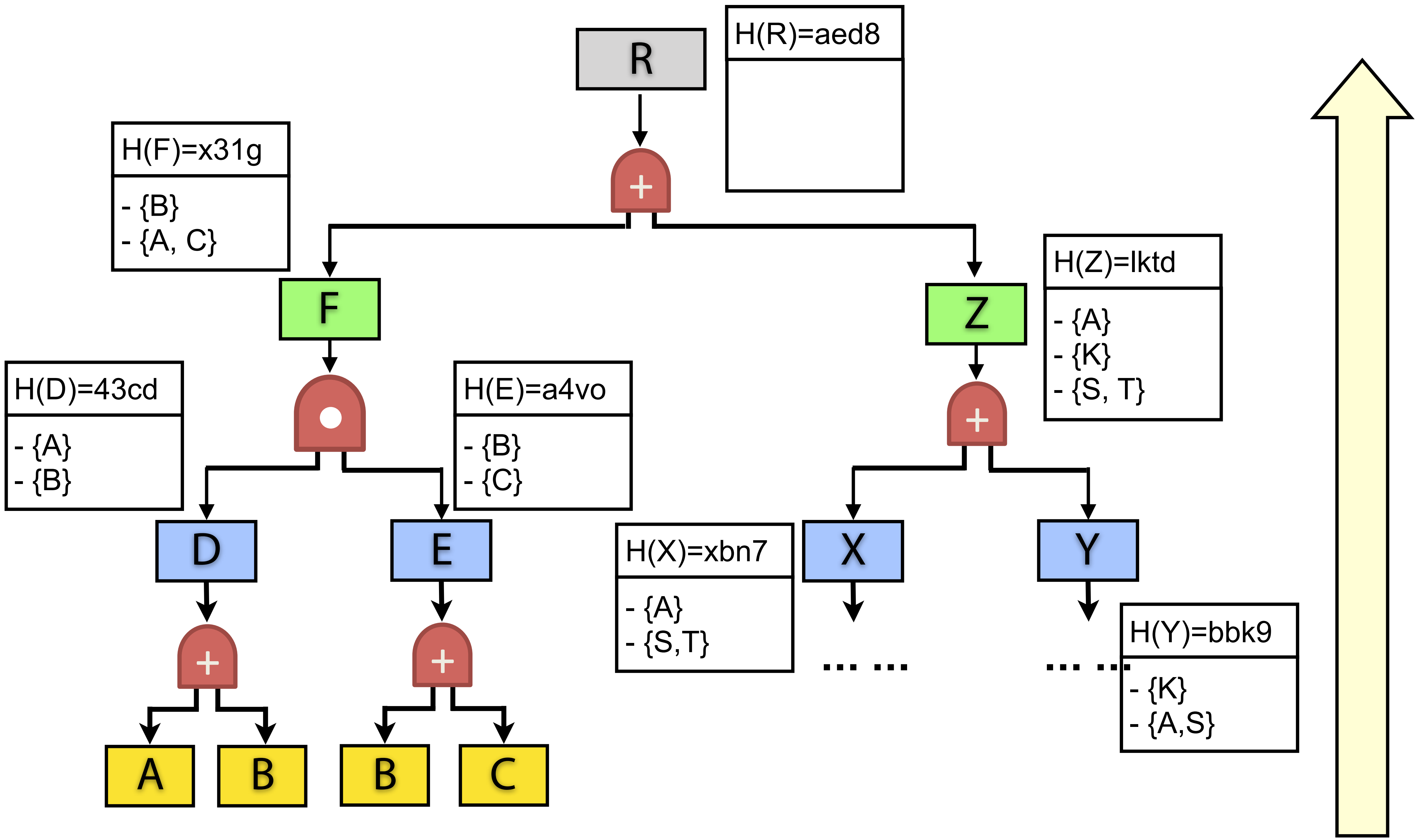
FirstAudit Primitive



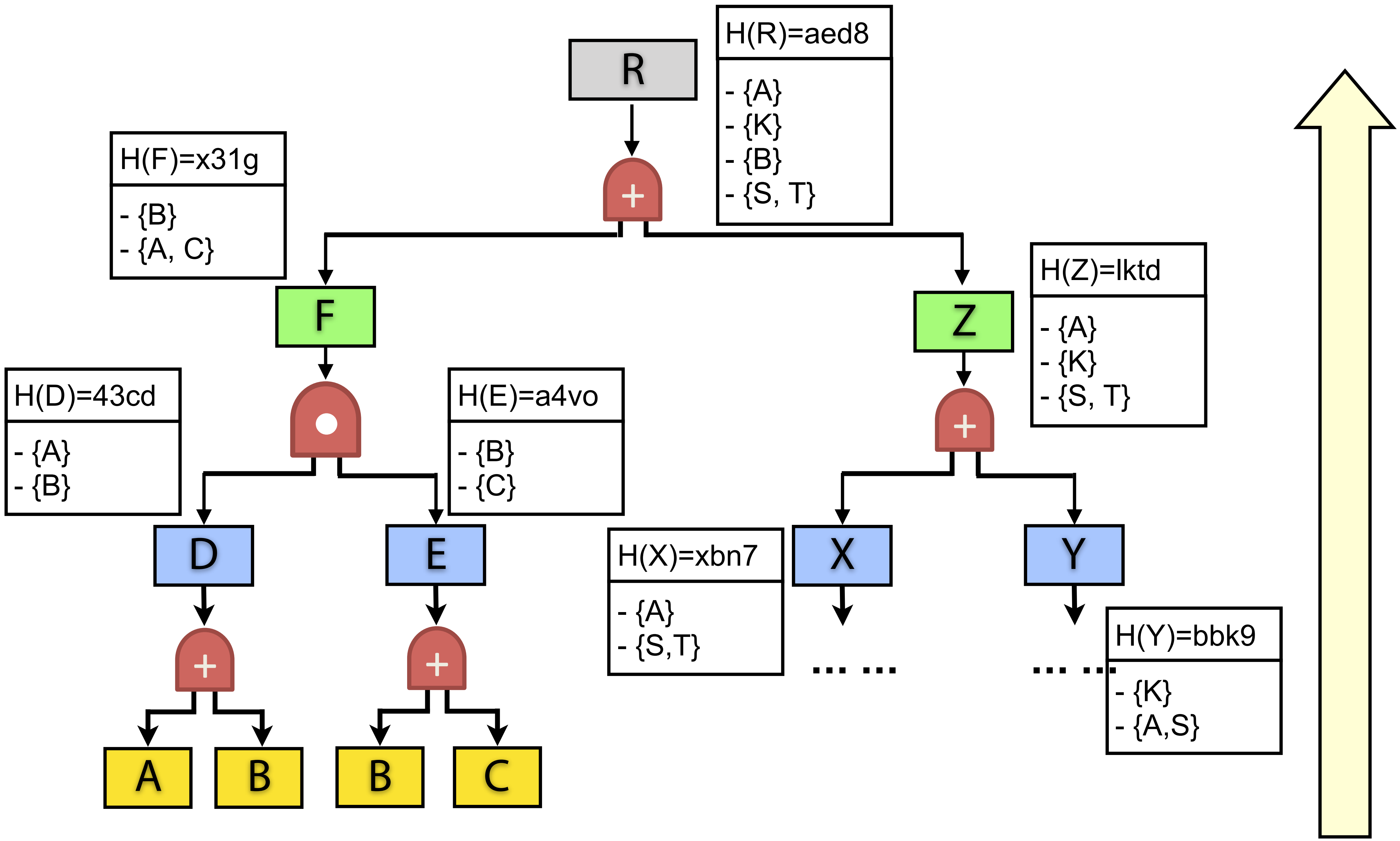
FirstAudit Primitive



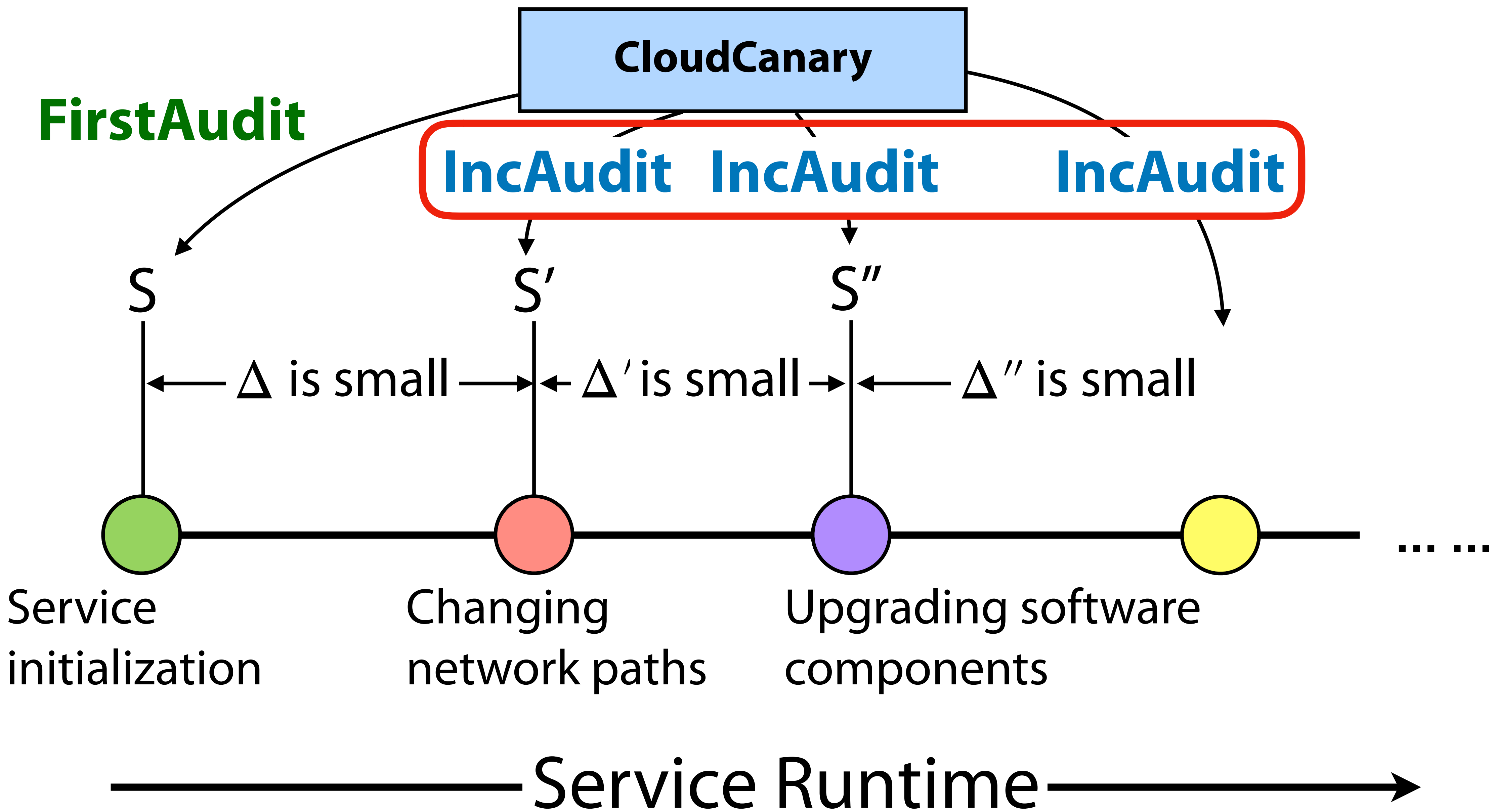
FirstAudit Primitive



FirstAudit Primitive

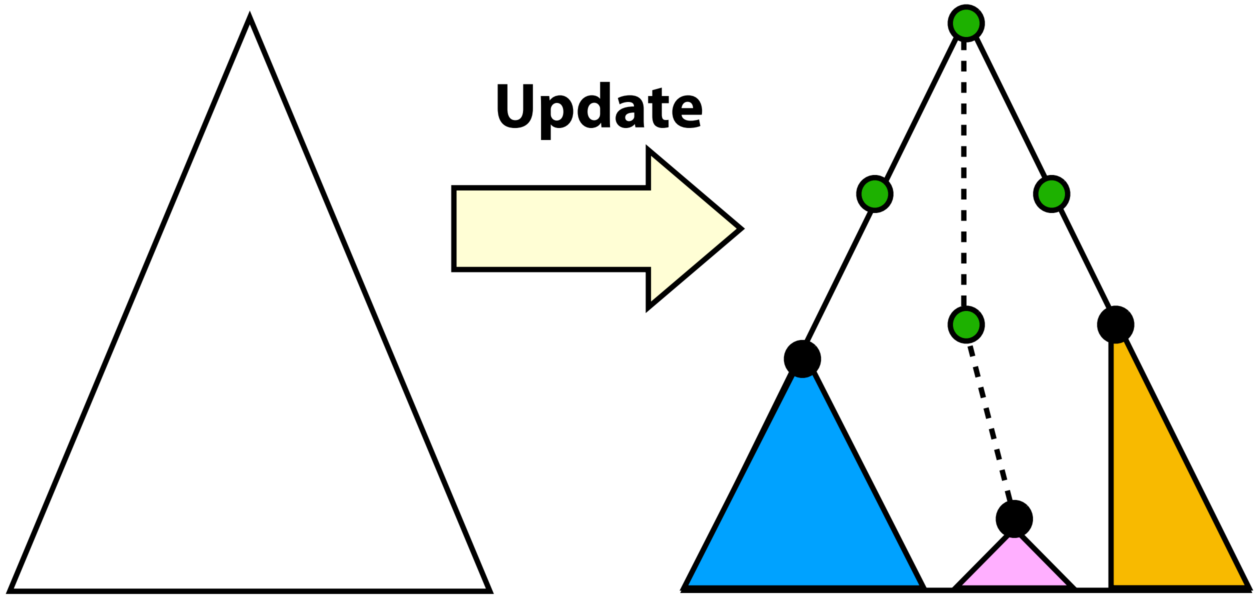


SnapAudit: FirstAudit & IncAudit

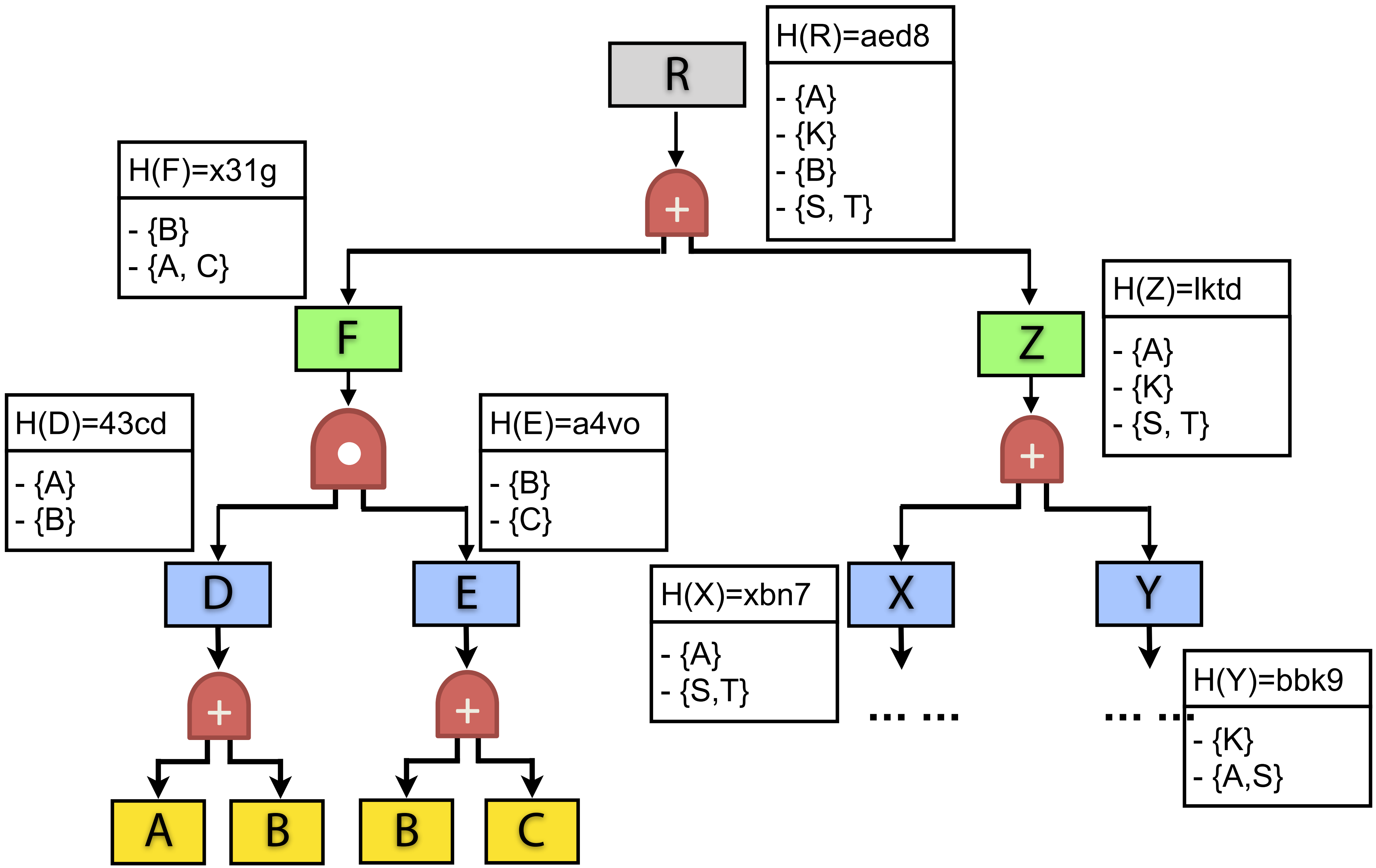


Our Insight

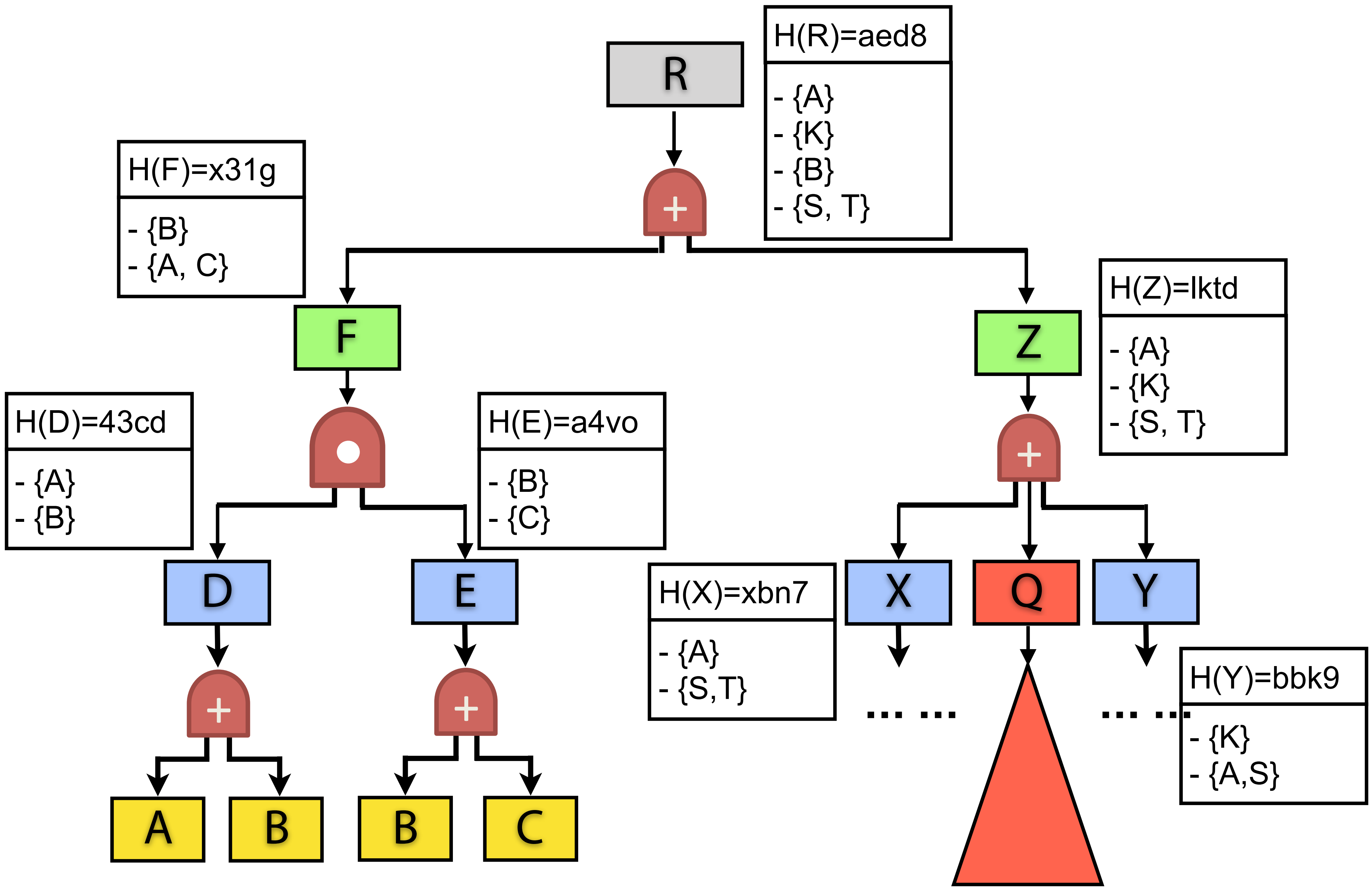
- Algorithm sketch:
  - Finding all the border nodes (black nodes)
  - Computing their risk groups
  - Merging these risk groups towards root



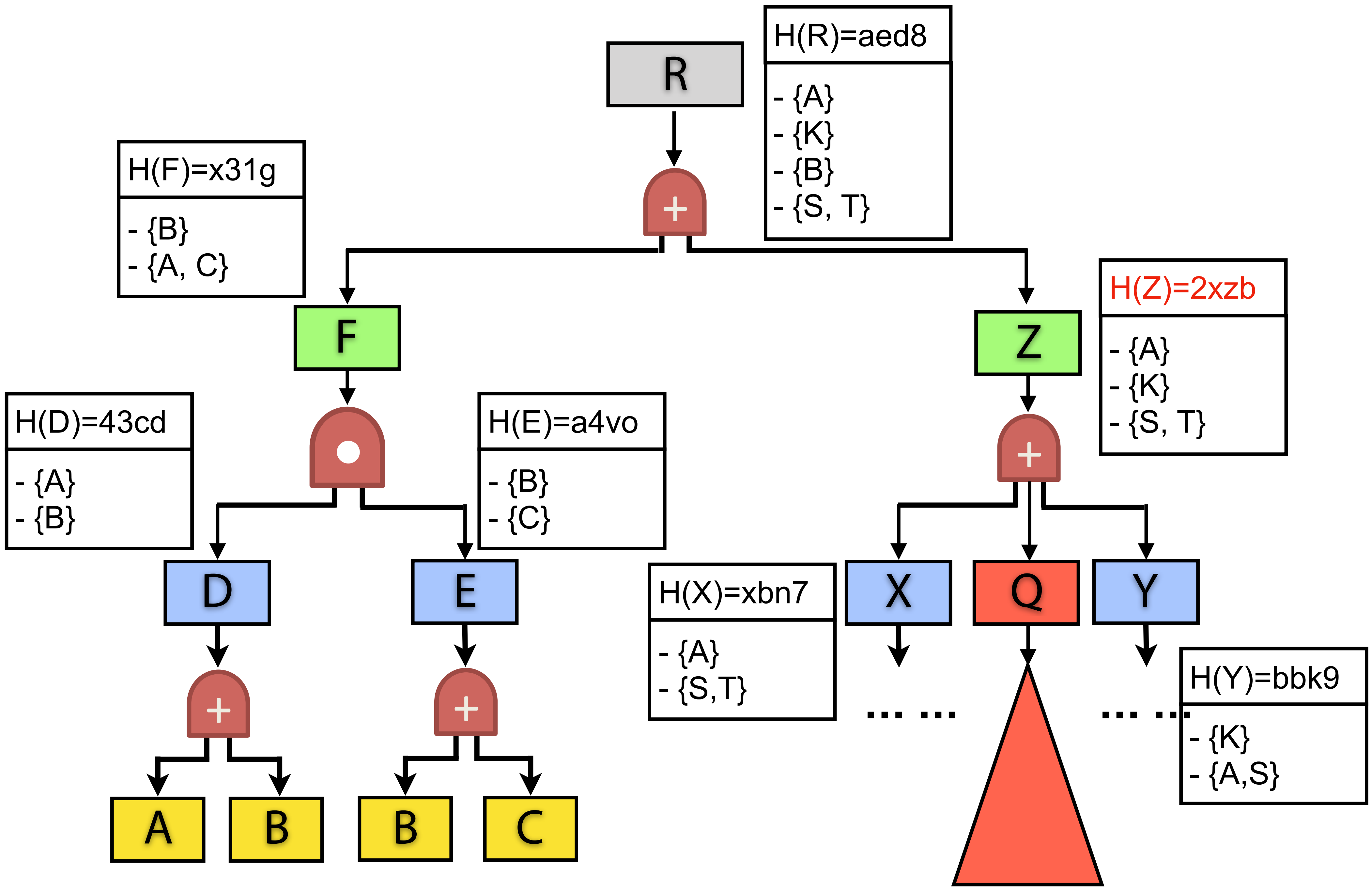
Original Deployment



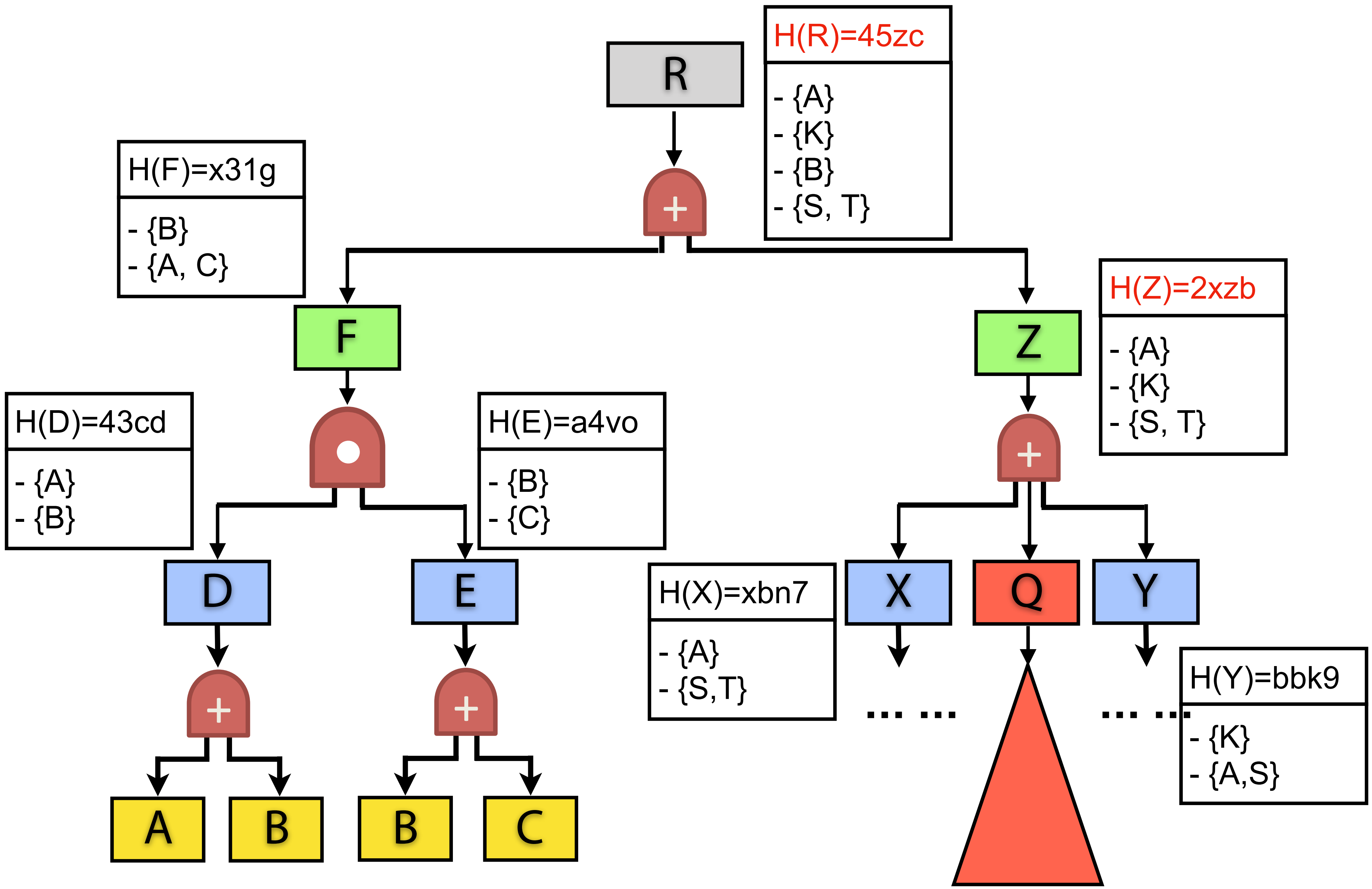
Updated Deployment



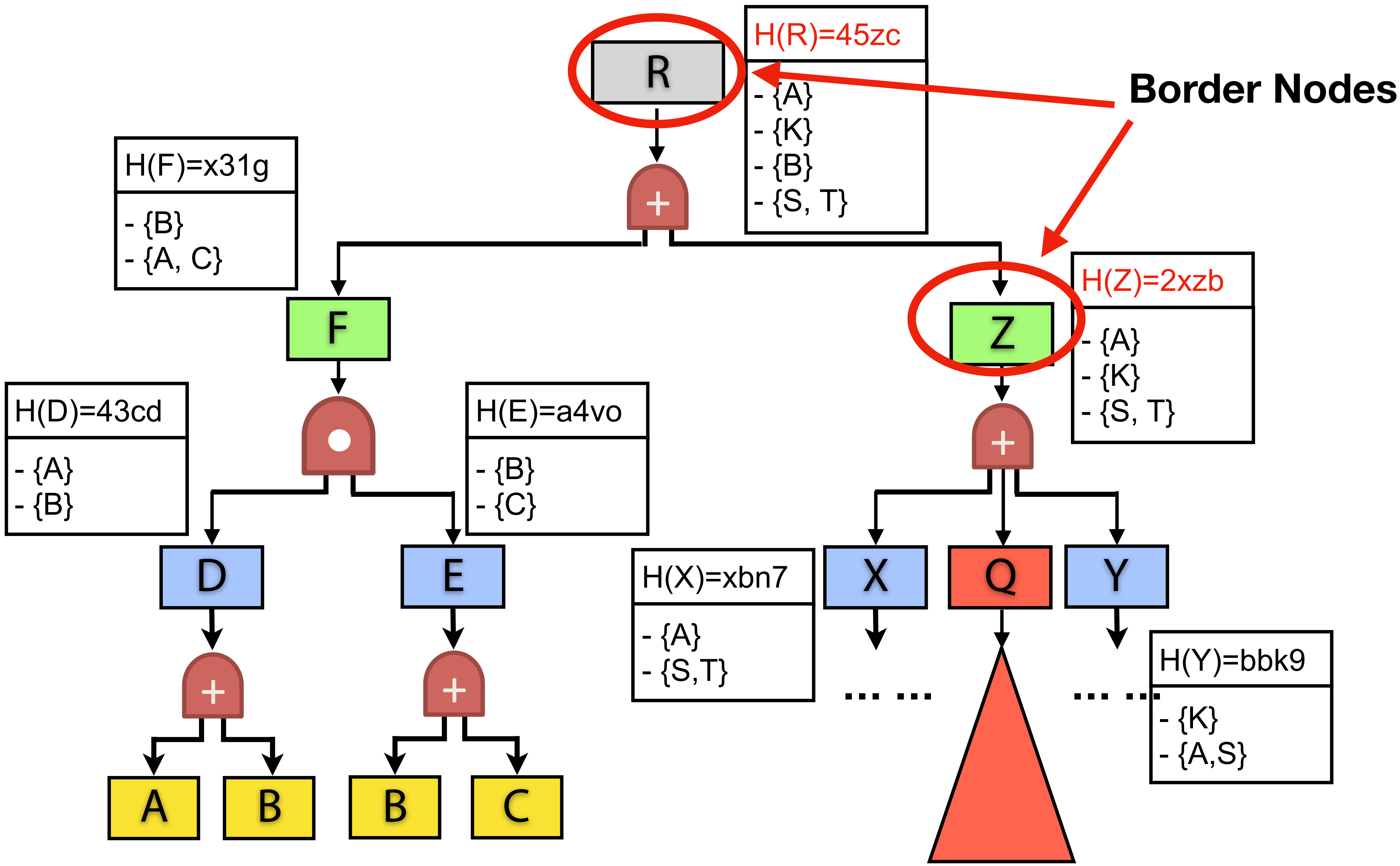
Updated Deployment



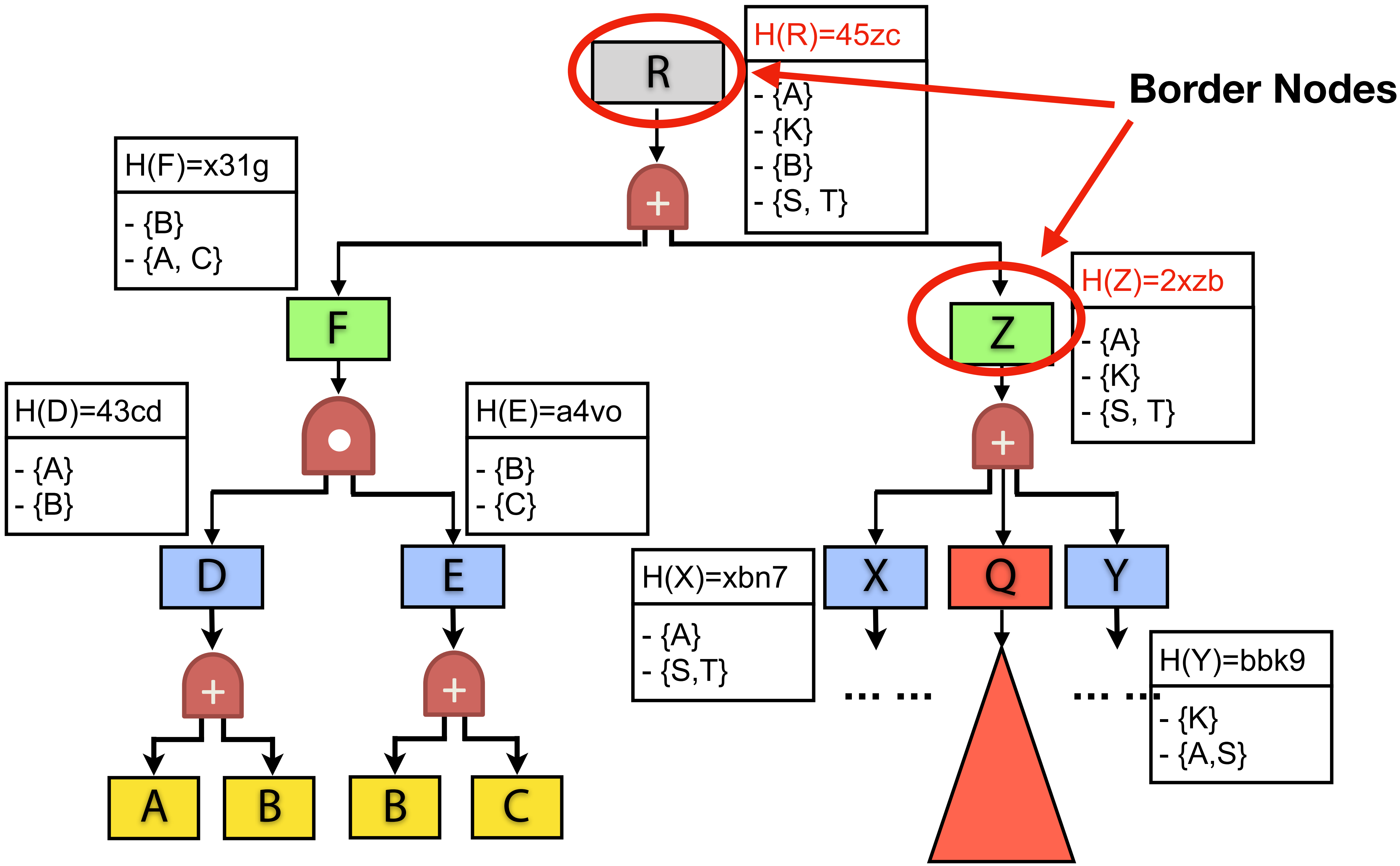
Updated Deployment



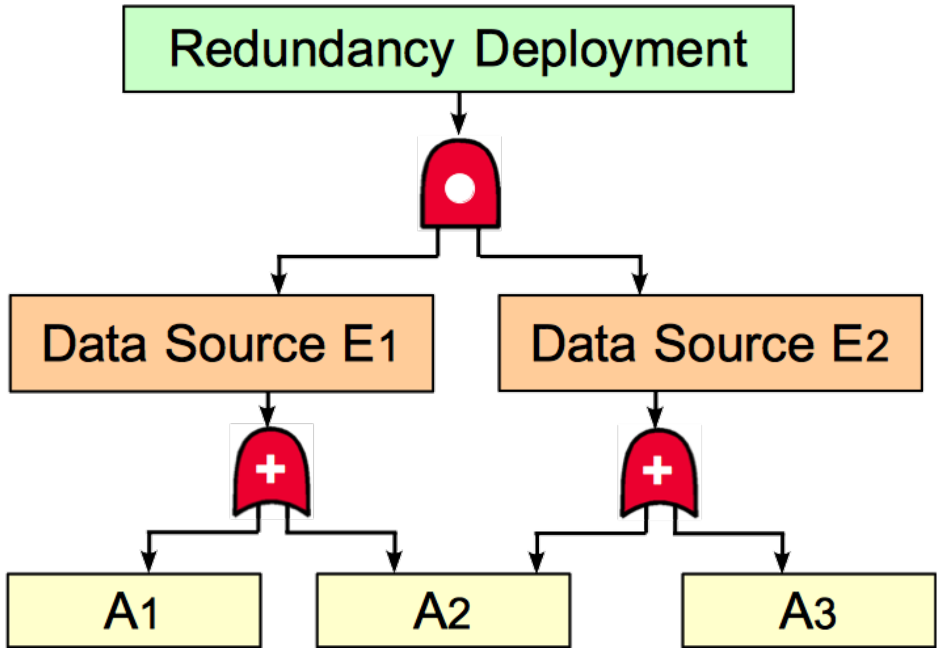
Step 1: Find Border Nodes



Step 2: Q's Risk Groups



Our Insight

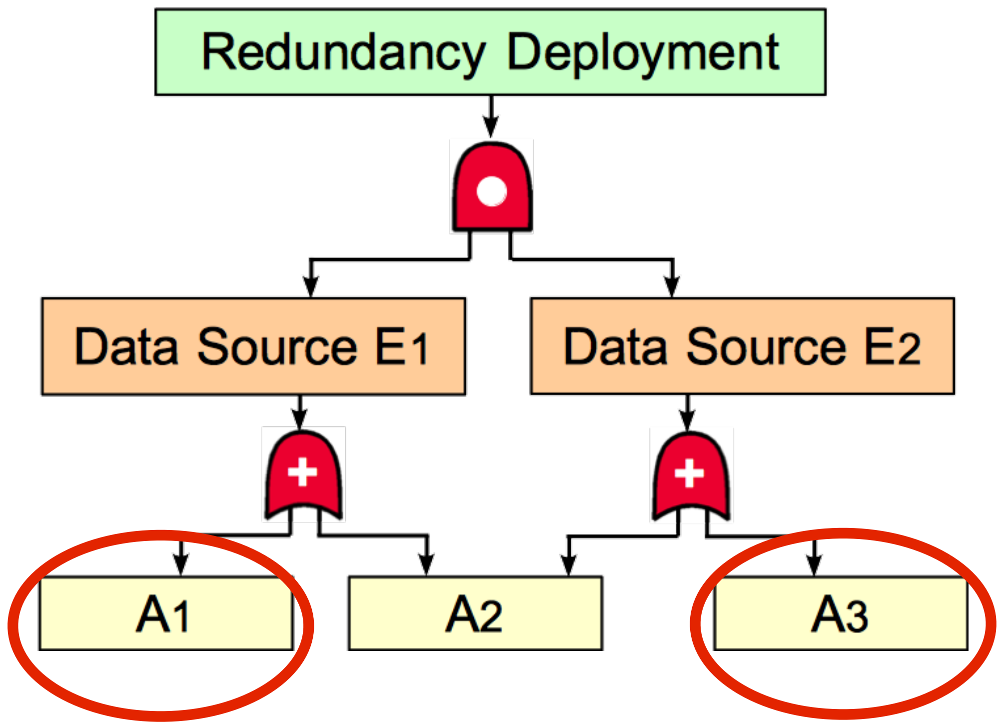


Boolean formula

$= E_1 \wedge E_2$

$= (A_1 \vee A_2) \wedge (A_2 \vee A_3)$

Our Insight



Boolean formula

$= E_1 \wedge E_2$

$= (A_1 \vee A_2) \wedge (A_2 \vee A_3)$

SAT solver

Satisfying assignment:
{A1=1, A2=0, A3=1}

Our Insight

- Problem:
  - Standard SAT solver outputs an arbitrary satisfying assignment

Our Insight

- Problem:
  - Standard SAT solver outputs an arbitrary satisfying assignment
  - What we want is top-k **critical (minimal)** risk groups

Identifying Risk Groups

- Using MinCostSAT solver
  - Satisfiable assignment with the least weights
  - Obtain the least $C = \sum c_i \cdot w_i$
  - Very fast with 100% accuracy

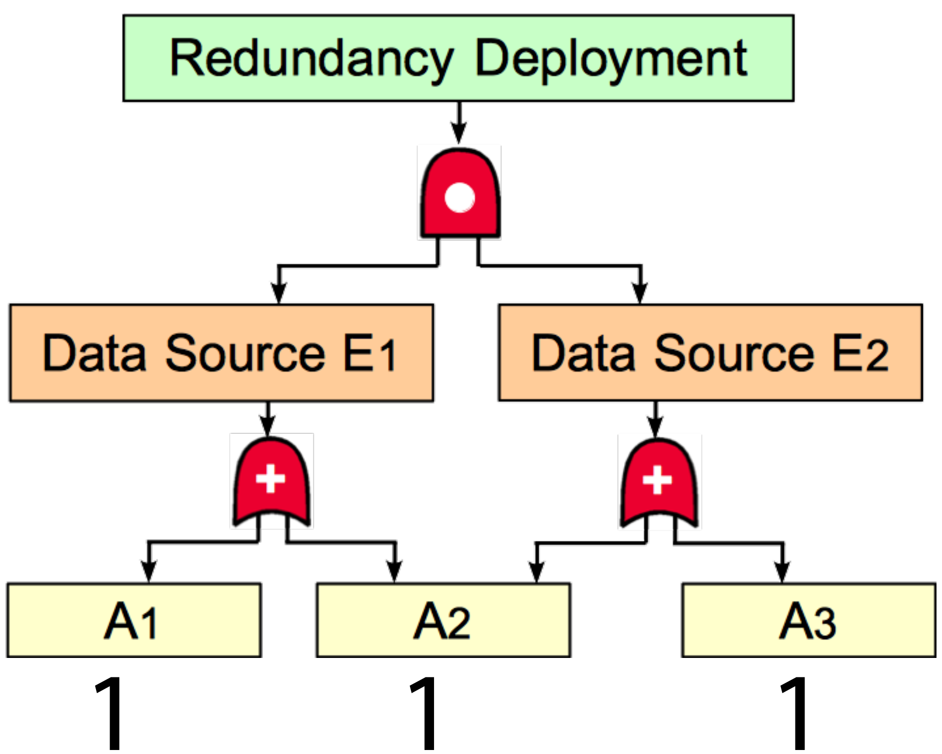
Identifying Risk Groups

- Using MinCostSAT solver
  - Satisfiable assignment with the least weights
  - Obtain the least $C = \sum c_i \cdot w_i$
  - Very fast with 100% accuracy

We set the values of all the leaf nodes (i.e., W_i) as 1

Identifying Risk Groups

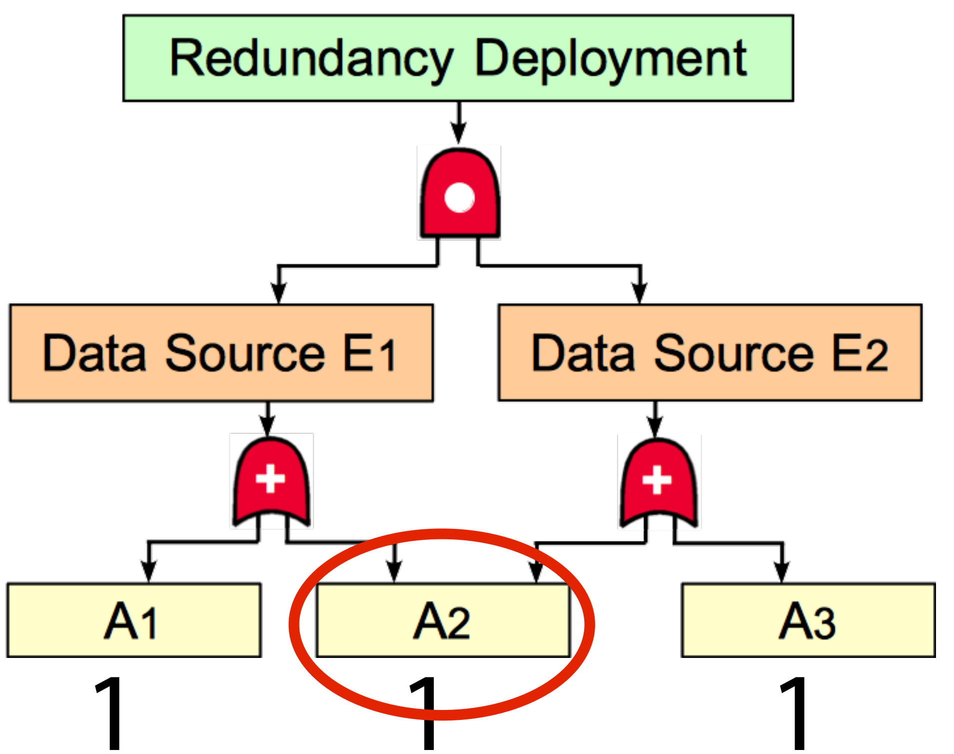
- Using MinCostSAT solver
  - Satisfiable assignment with the least weights
  - Obtain the least $C = \sum c_i \cdot w_i$
  - Very fast with 100% accuracy



A1	A2	A3	Weight
1	0	0	
0	1	0	1
0	0	1	
1	1	0	2
1	0	1	2
0	1	1	2
0	0	0	
1	1	1	3

Identifying Risk Groups

- Using MinCostSAT solver
  - Satisfiable assignment with the least weights
  - Obtain the least $C = \sum c_i \cdot w_i$
  - Very fast with 100% accuracy



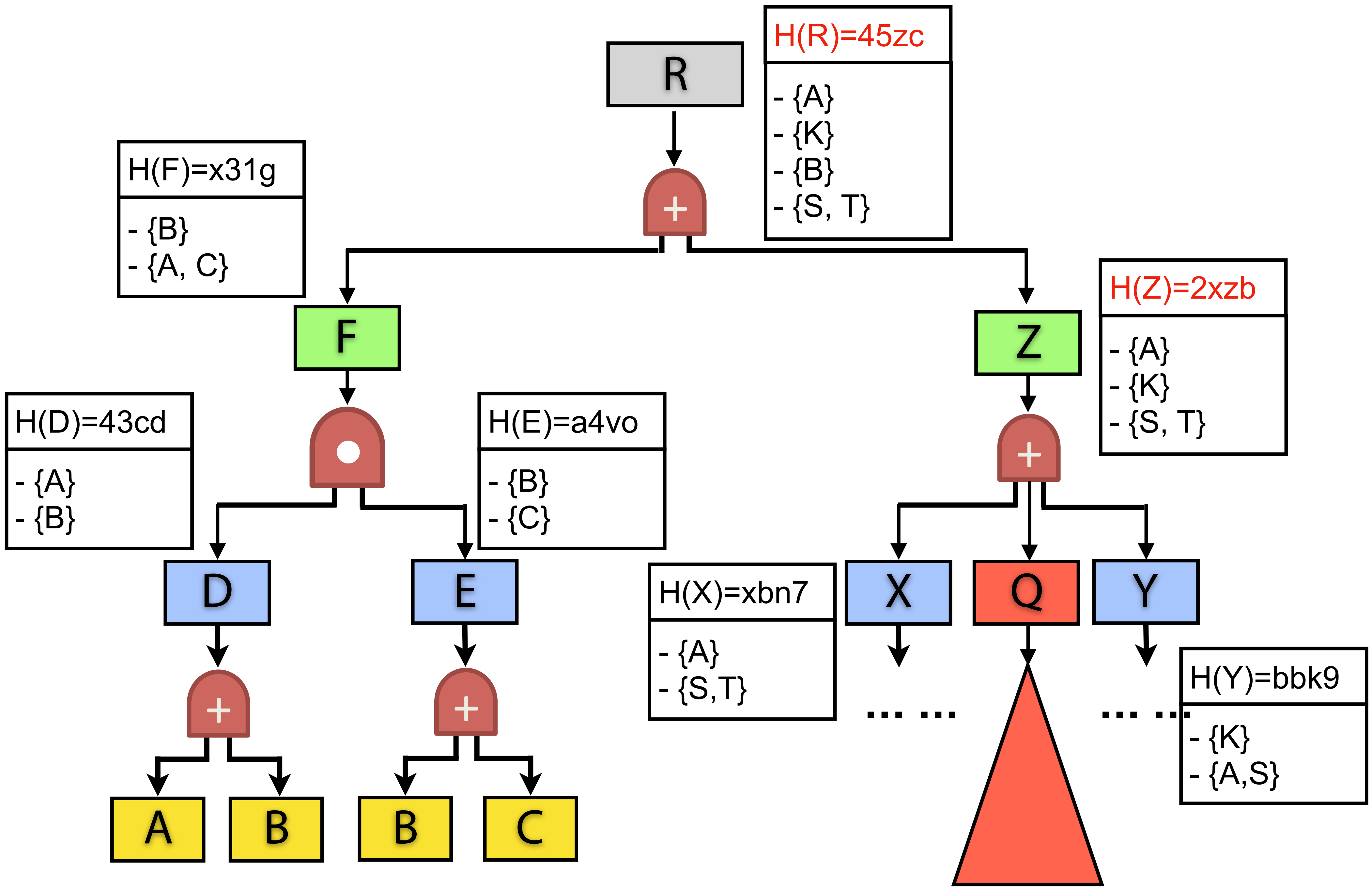
A1	A2	A3	Weight
1	0	0	
0	1	0	1
0	0	1	
1	1	0	2
1	0	1	2
0	1	1	2
0	0	0	
1	1	1	3

Identifying Risk Groups

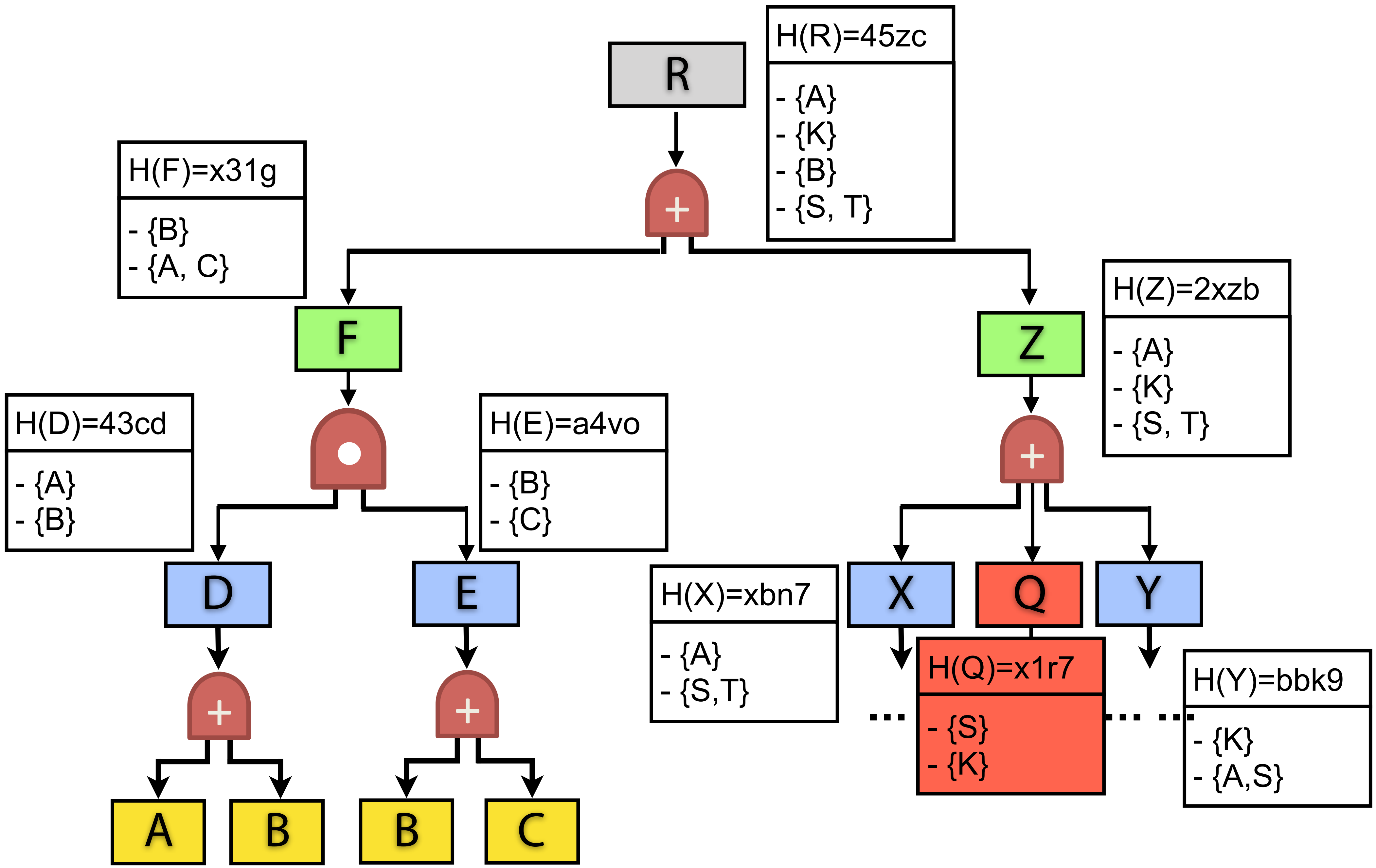
- Find out the top-k critical risk groups
  - Use a $\wedge$ to connect the current formula and the negation of the resulting assignment

$$(A_1 \vee A_2) \wedge (A_2 \vee A_3) \wedge \neg(\neg A_1 \wedge A_2 \wedge \neg A_3)$$

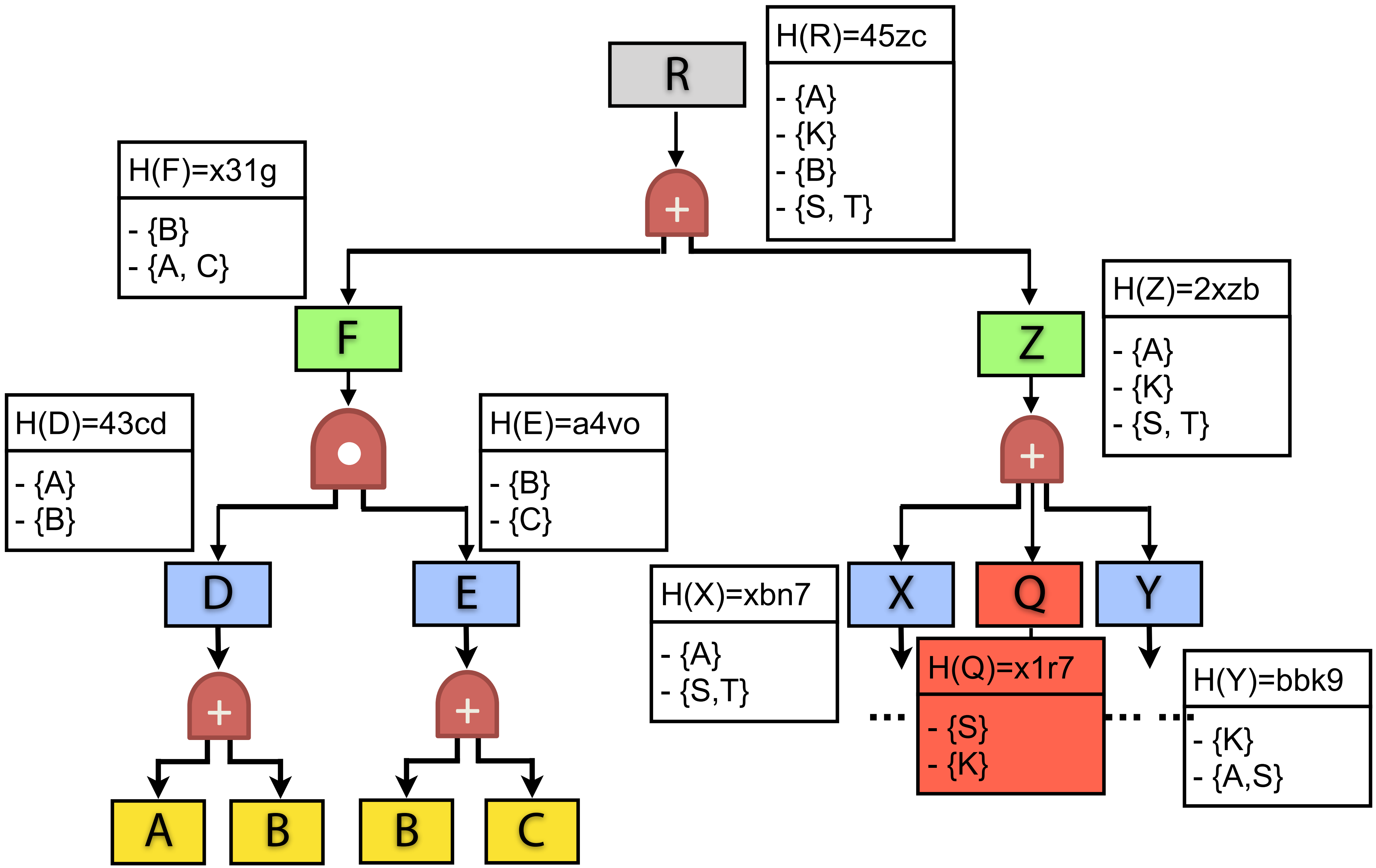
Step 2: Q's Risk Groups



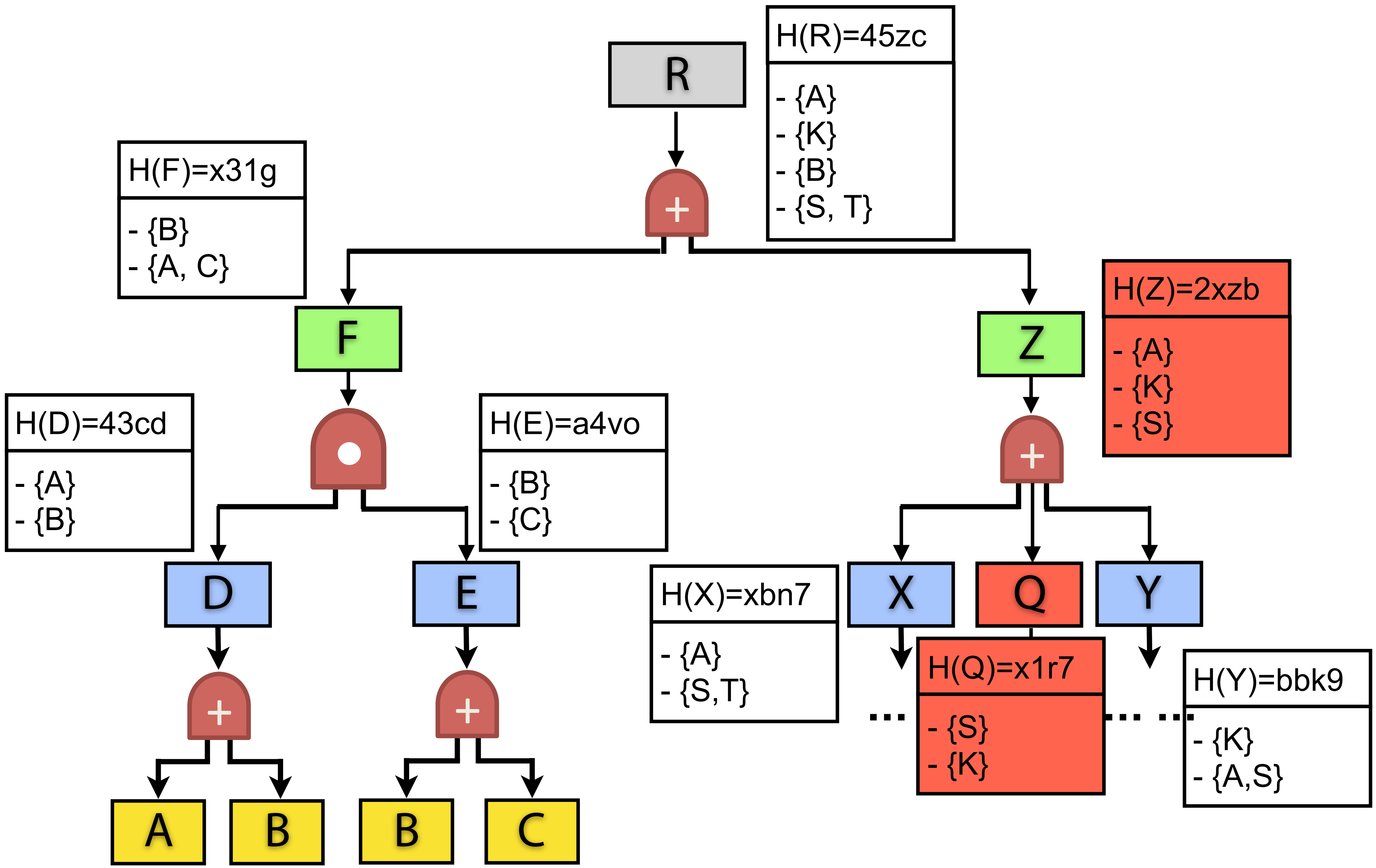
Step 2: Q's Risk Groups



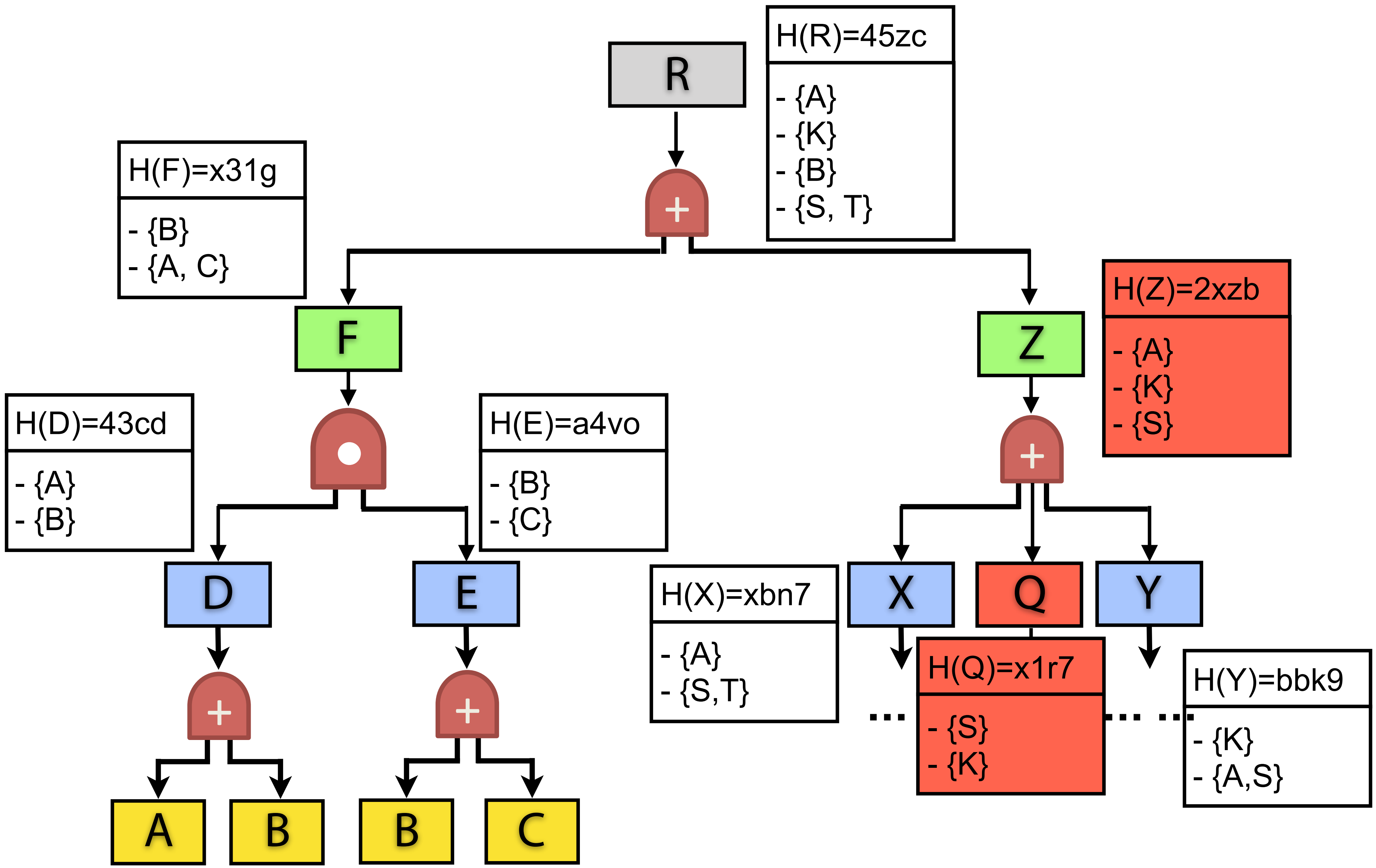
Step 3: Merging Changed Caches



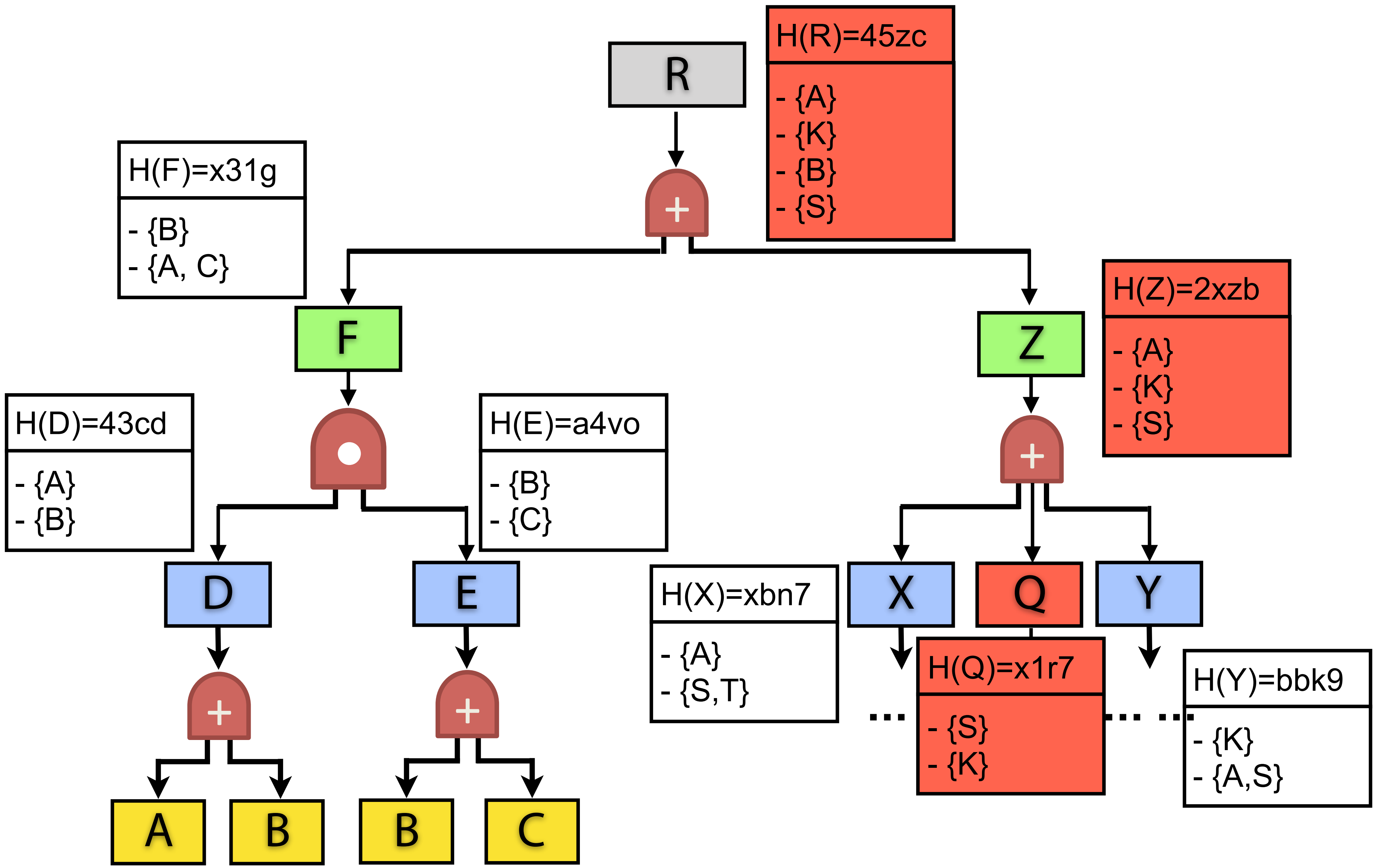
Step 3: Merging Changed Caches



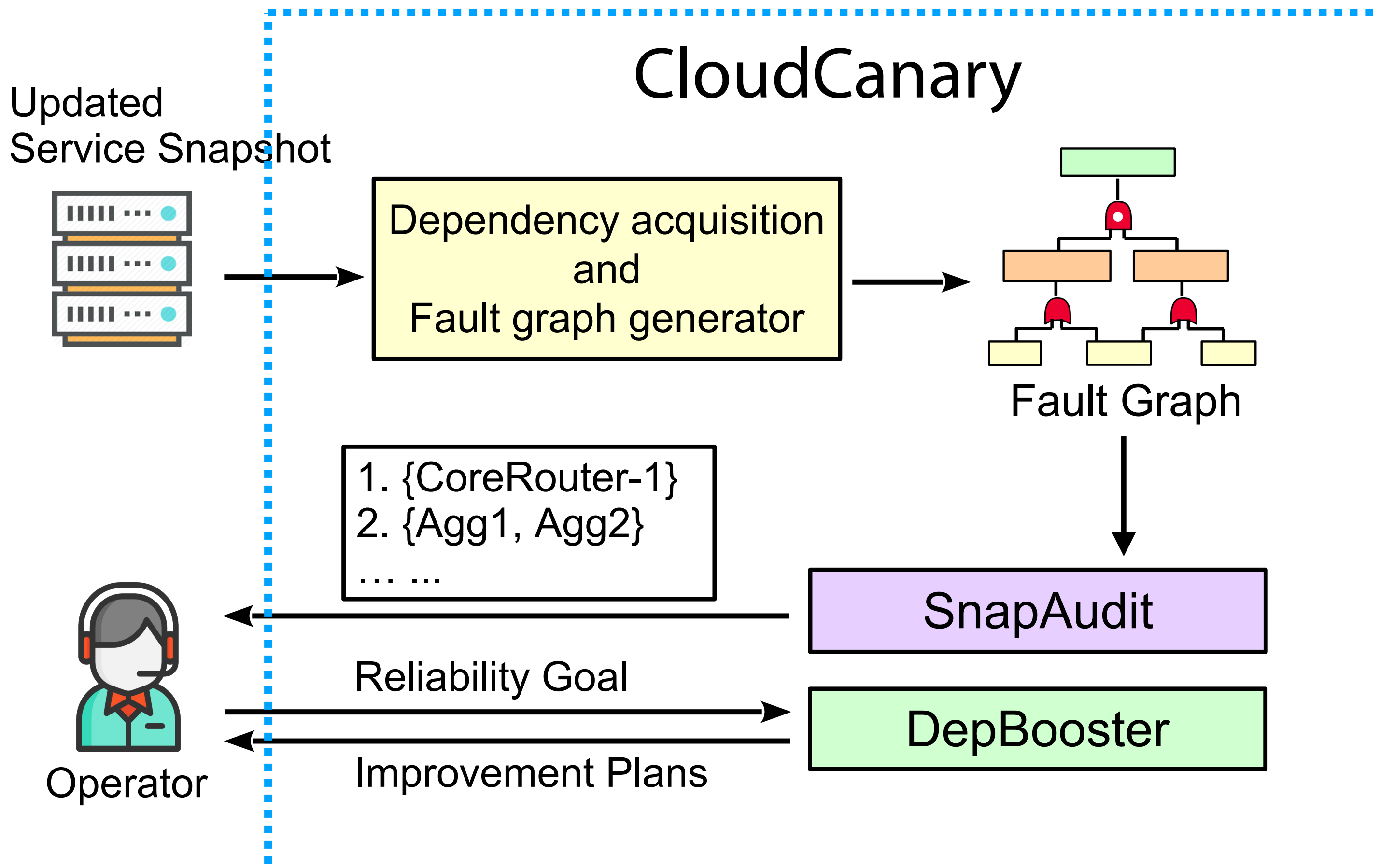
Step 3: Merging Changed Caches



Step 3: Merging Changed Caches

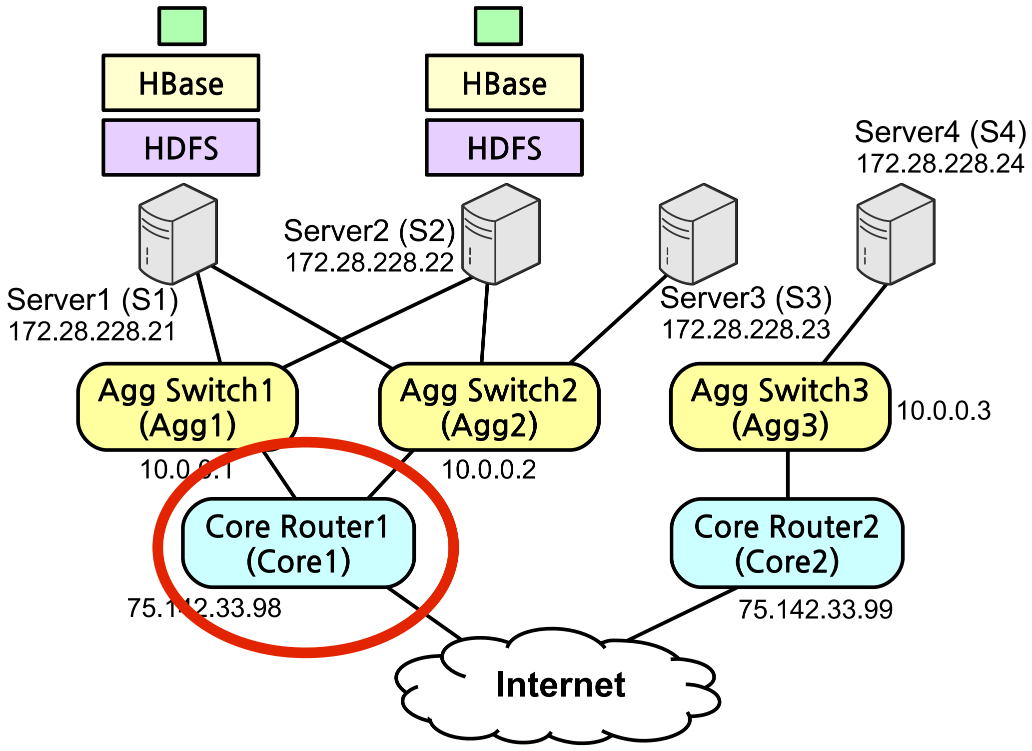


CloudCanary's Workflow

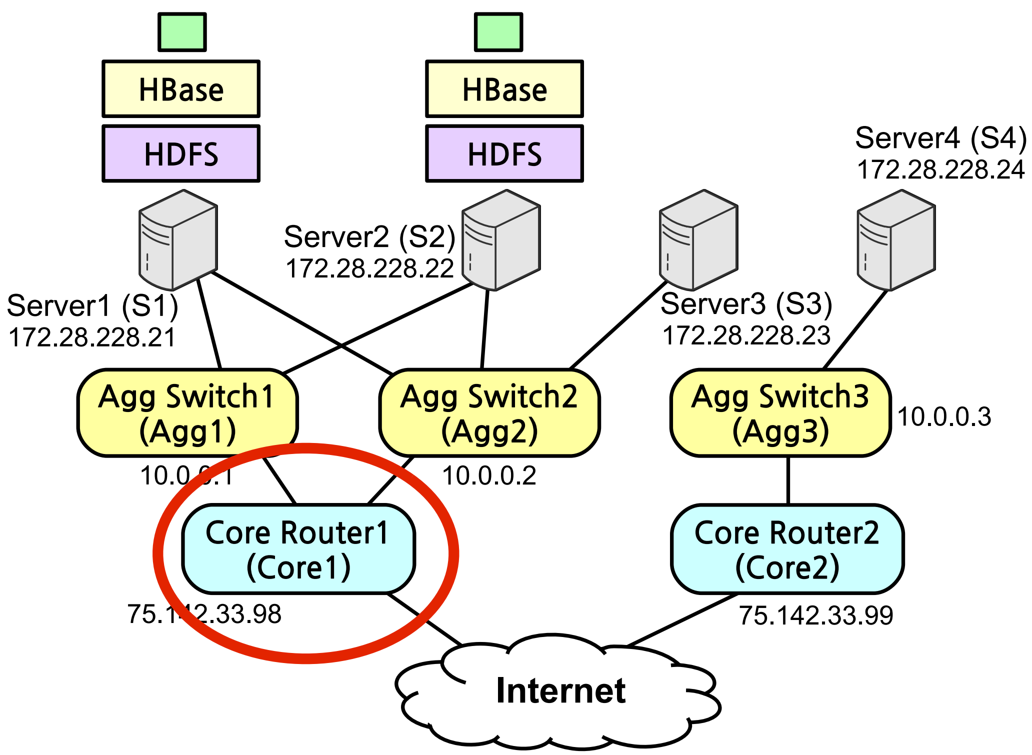


- Challenge 1: SnapAudit
- Challenge 2: DepBooster

Correlated Failure Risk Repairing



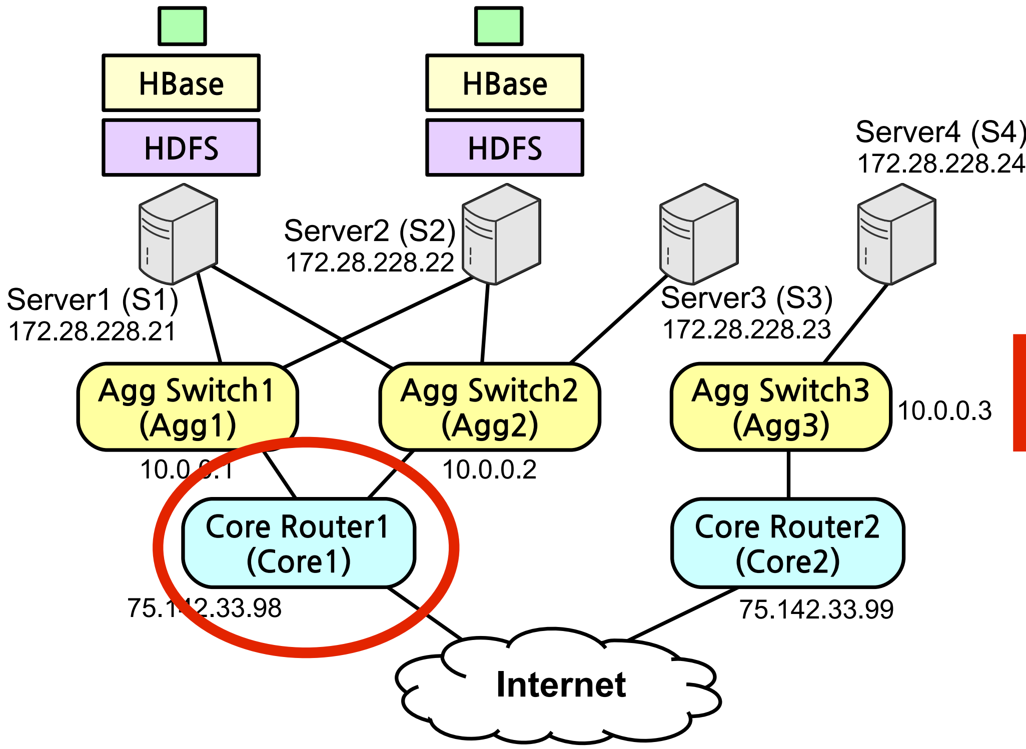
Correlated Failure Risk Repairing



Specification:

```
$Server -> 172.28.228.21, 172.28.228.22
goal(failProb(ft)<0.08 | ChNode | Agg3)
```

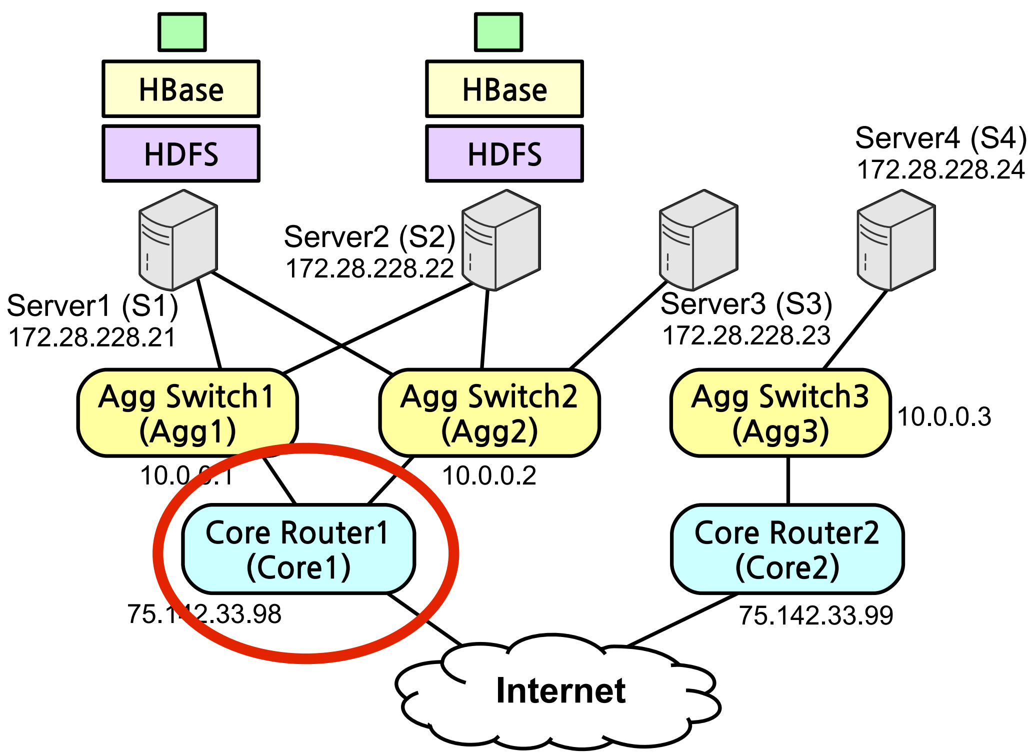
Correlated Failure Risk Repairing



Specification:

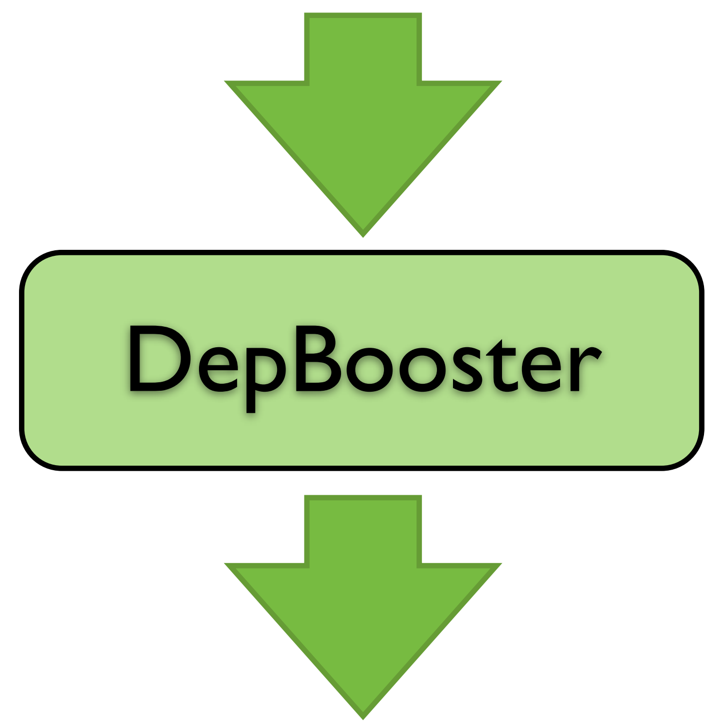
```
$Server -> 172.28.228.21, 172.28.228.22
goal(failProb(ft)<0.08 | ChNode | Agg3)
```

Correlated Failure Risk Repairing



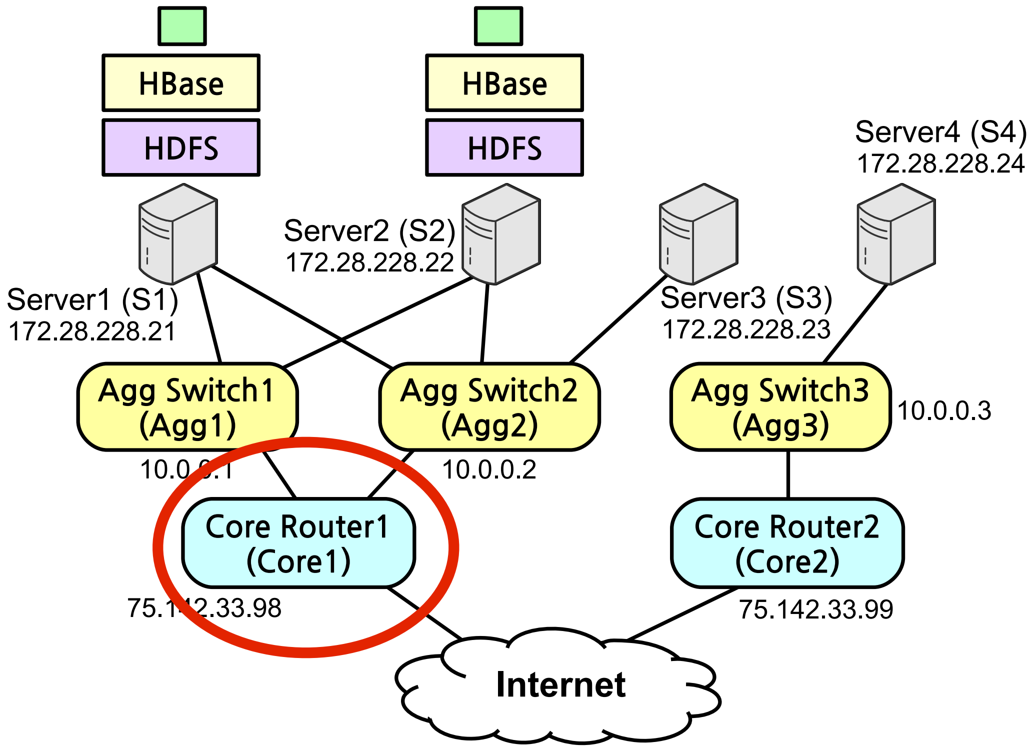
Specification:

```
$Server -> 172.28.228.21, 172.28.228.22
goal(failProb(ft)<0.08 | ChNode | Agg3)
```




```
Plan 1: Move replica from S1 -> S4
Plan 2: Move replica from S2 -> S4
```

Correlated Failure Risk Repairing

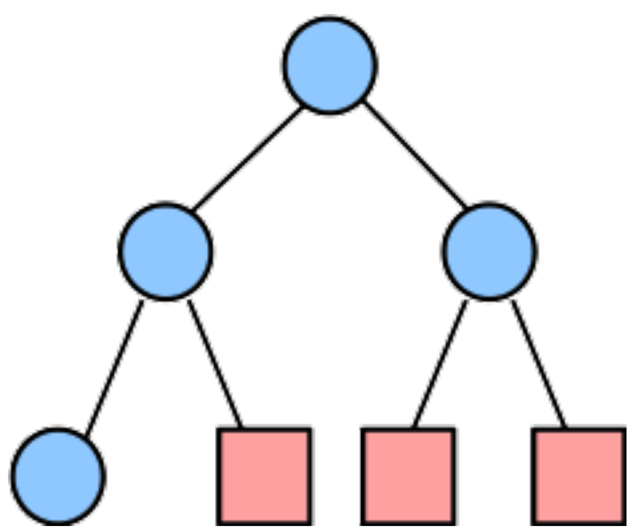


Specification:

```
$Server -> 172.28.228.21, 172.28.228.22
goal(failProb(ft)<0.08 | ChNode | Agg3)
```

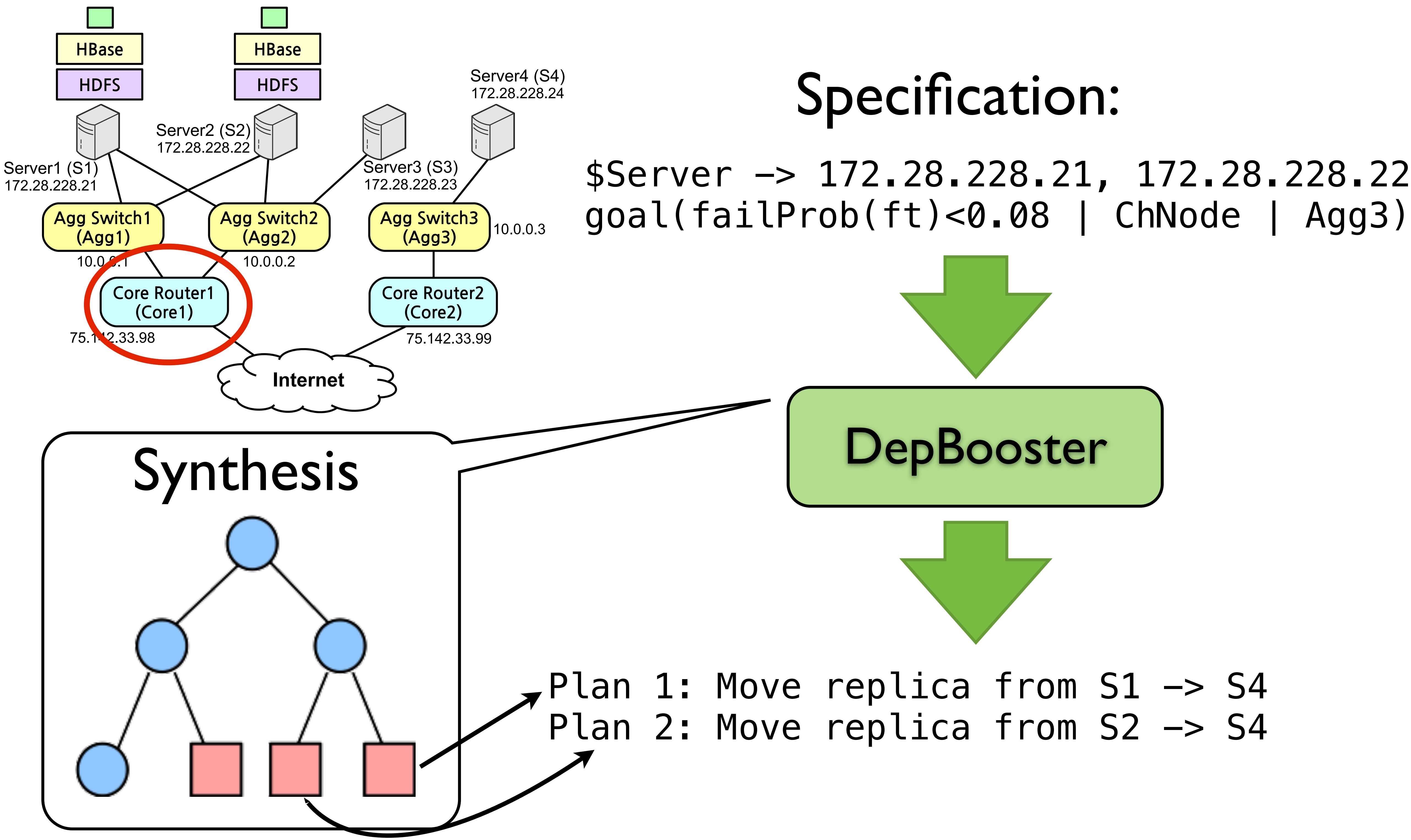
DepBooster

Synthesis

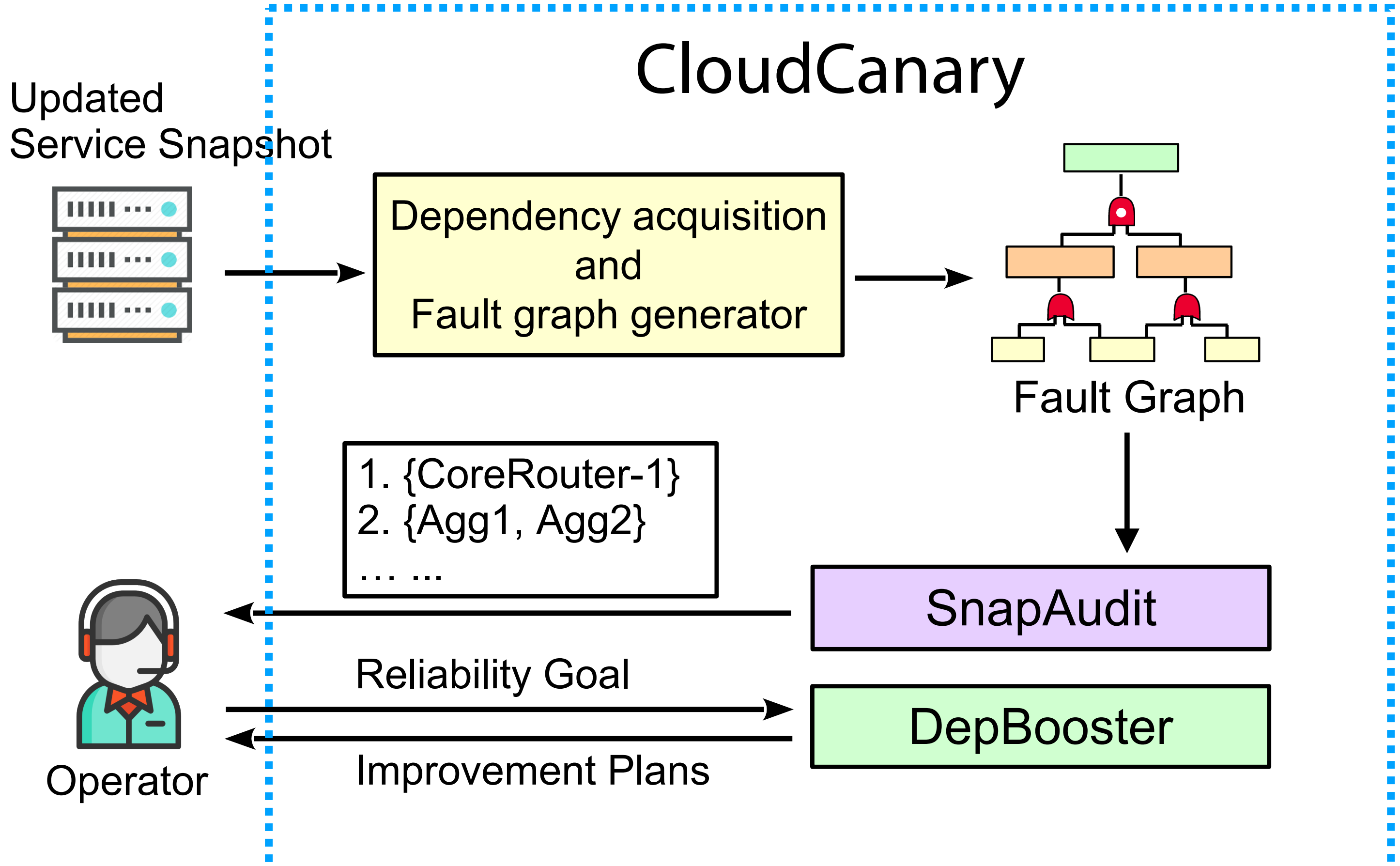


```
Plan 1: Move replica from S1 -> S4
Plan 2: Move replica from S2 -> S4
```

Correlated Failure Risk Repairing



CloudCanary's Workflow



- Challenge 1: SnapAudit
- Challenge 2: DepBooster

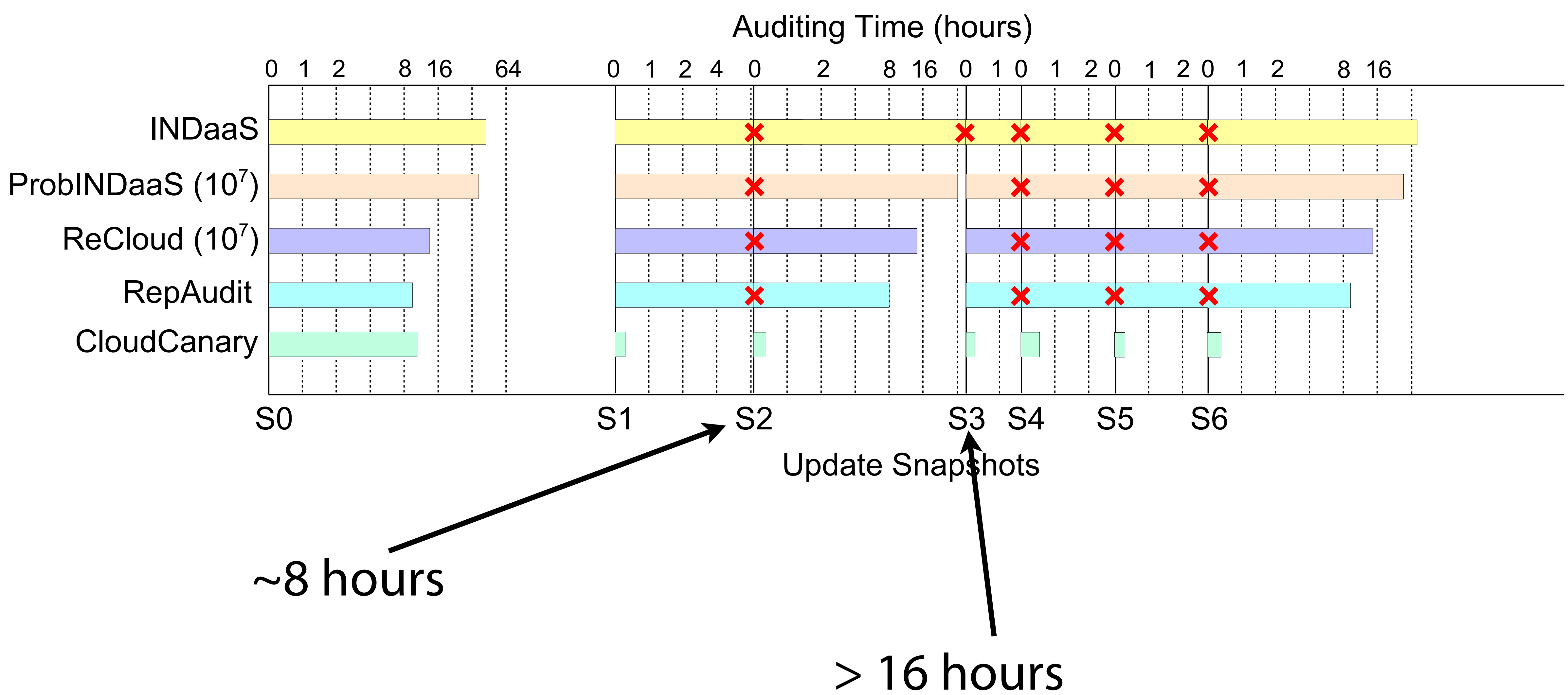
Evaluation

- Comparing CloudCanary with the state of the art
- Evaluating CloudCanary's practicality via real dataset

Evaluation

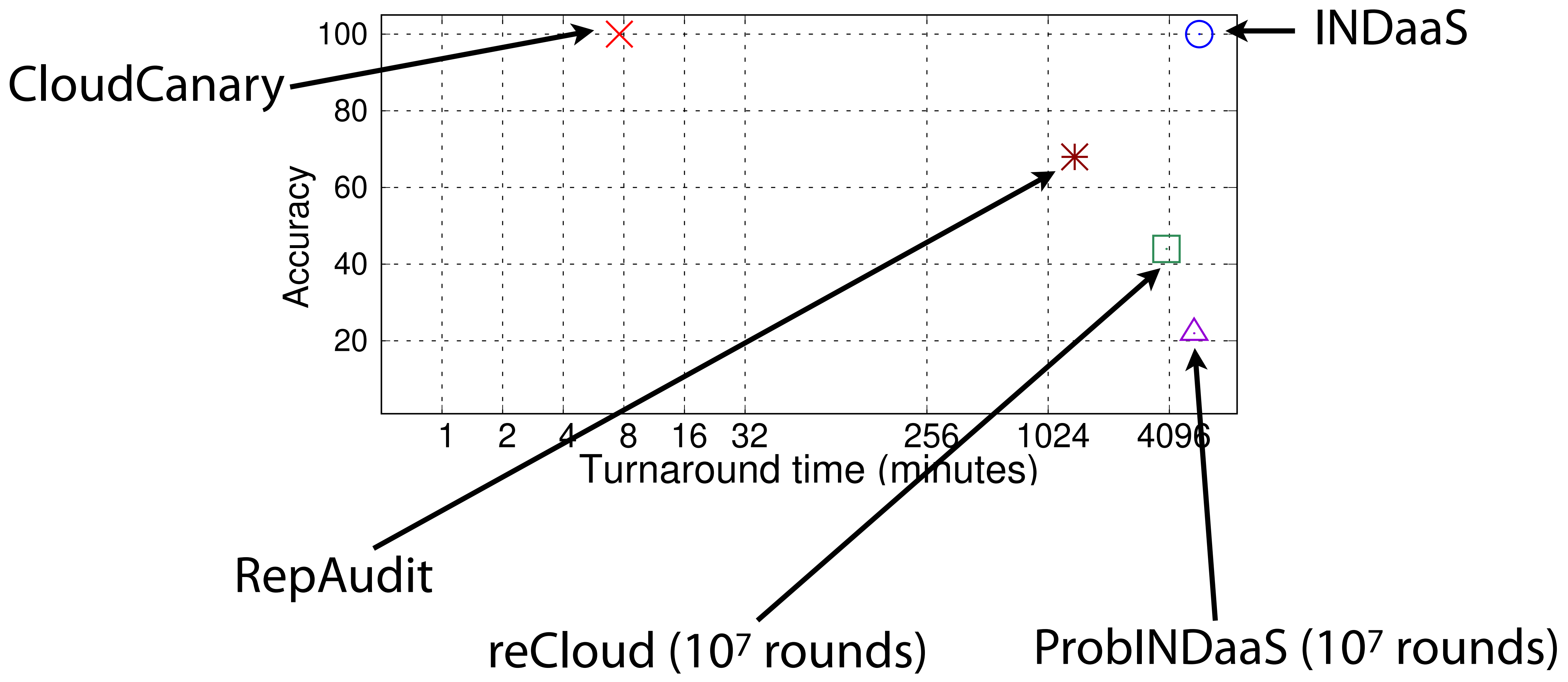
	Accuracy	Efficiency	Improvement
INDaaS [OSDI’14]	✔	✘	✘
ProbINDaaS [OSDI’14]	✘	✔✘	✘
reCloud [CoNEXT’16]	✘	✔✘	✘
RepAudit [OOPSLA’17]	✔✘	✔✘	✘
CloudCanary	✔	✔	✔

Efficiency Comparison

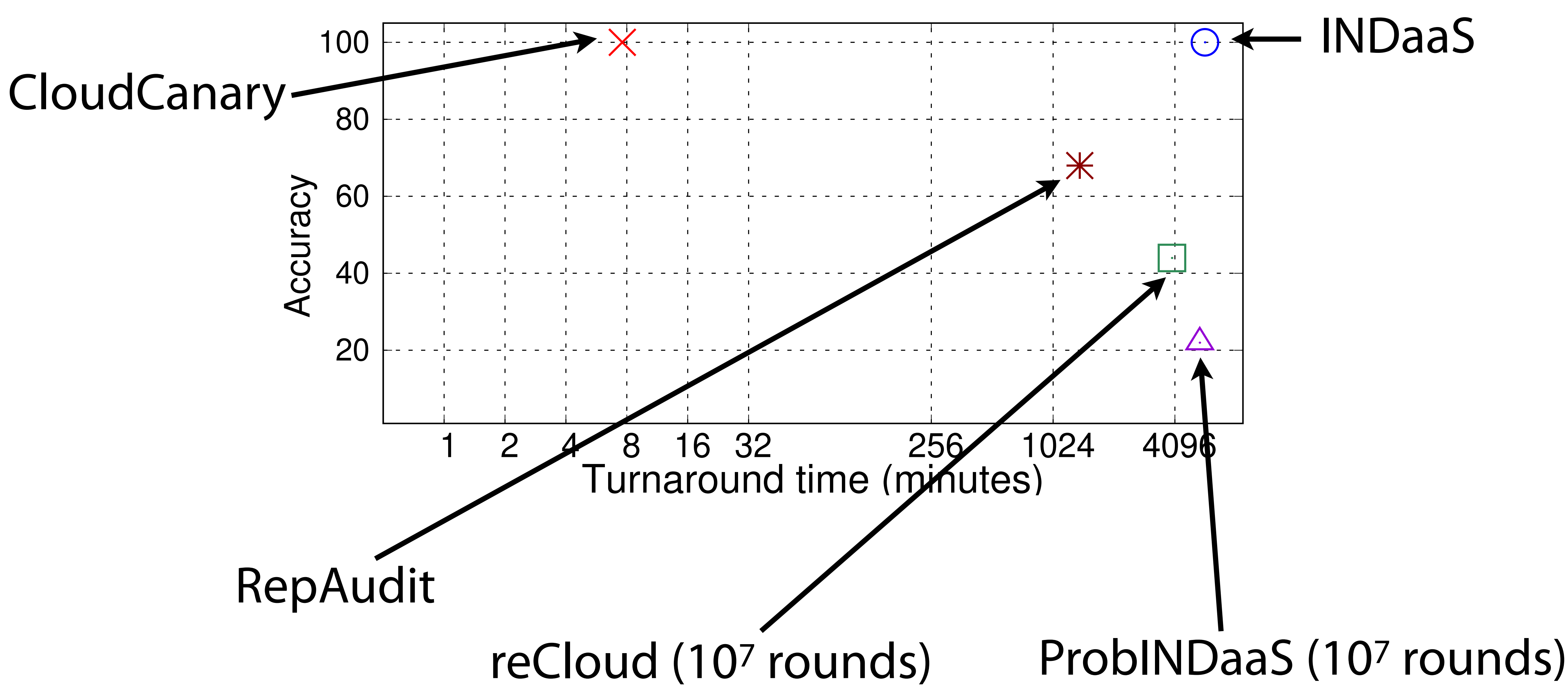


Accuracy V.S. Efficiency

- 20,608 switches; 524,288 servers; 638,592 software components
- Auditing a random update affecting 20% components



Our approach is 200x faster than state-of-the-arts, and offers 100% accurate results.



Evaluation

- We evaluated CloudCanary via real update trace:

	Detected Num	Confirmed	Examples
Microservices	50+	96%	Authentication and access control systems introduce most risk groups
Power Sources	10+	100%	Primary and backup power sources are carelessly assigned to multiple racks hosting a critical service
Network	30+	100%	Aggregation and ToR switches are easily updated to be risk groups

Conclusion

- CloudCanary is the first system for real-time auditing
  - SnapAudit primitive: Quickly auditing update snapshot
  - DepBooster: Quickly generating improvement plans

- We evaluated CloudCanary with real trace and large-scale emulations

Thanks, questions?

- CloudCanary is the first system for real-time auditing
  - SnapAudit primitive: Quickly auditing update snapshot
  - DepBooster: Quickly generating improvement plans

- We evaluated CloudCanary with real trace and large-scale emulations