Failures & Redundancy

Ennan Zhai

Computer Science at Yale University
ennan.zhai@yale.edu
About our labs, final project and midterm
About our labs, final project and midterm

- Building a simple and elegant Peerster
About our labs, final project and midterm

- Building a simple and elegant Peerster
- Making git log clear
About our labs, final project and midterm

• Building a simple and elegant Peerster
• Making git log clear
• Midterm
About our labs, final project and midterm

- Building a simple and elegant Peerster
- Making git log clear
- Midterm
- Lab1 grading
Lecture Outline

- Failures
- Correlated failures in decentralized systems
Lecture Outline

• Failures

• Correlated failures in decentralized systems
Failures

• What is the failure?
Failures

- What is the failure?
- Why?
Failures

- What is the failure?
- Why?
- Real evidences?
Evidence
### Evidence

<table>
<thead>
<tr>
<th>Category</th>
<th>Failure types</th>
<th>Diagnosis &amp; Repair</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software 21%</td>
<td>Link layer loop</td>
<td>Find and fix bugs</td>
<td>19%</td>
</tr>
<tr>
<td></td>
<td>Imbalance → overload</td>
<td></td>
<td>2%</td>
</tr>
</tbody>
</table>
## Evidence

<table>
<thead>
<tr>
<th>Category</th>
<th>Failure types</th>
<th>Diagnosis &amp; Repair</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software 21%</td>
<td>Link layer loop</td>
<td>Find and fix bugs</td>
<td>19%</td>
</tr>
<tr>
<td></td>
<td>Imbalance → overload</td>
<td></td>
<td>2%</td>
</tr>
<tr>
<td>Hardware 18%</td>
<td>FCS error</td>
<td>Replace cable</td>
<td>13%</td>
</tr>
<tr>
<td></td>
<td>Unstable power</td>
<td>Repair power</td>
<td>5%</td>
</tr>
</tbody>
</table>
## Evidence

<table>
<thead>
<tr>
<th>Category</th>
<th>Failure types</th>
<th>Diagnosis &amp; Repair</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software 21%</td>
<td>Link layer loop</td>
<td>Find and fix bugs</td>
<td>19%</td>
</tr>
<tr>
<td></td>
<td>Imbalance $\rightarrow$ overload</td>
<td></td>
<td>2%</td>
</tr>
<tr>
<td>Hardware 18%</td>
<td>FCS error</td>
<td>Replace cable</td>
<td>13%</td>
</tr>
<tr>
<td></td>
<td>Unstable power</td>
<td>Repair power</td>
<td>5%</td>
</tr>
<tr>
<td>Unknown 23%</td>
<td>Switch stops forwarding</td>
<td>N/A</td>
<td>9%</td>
</tr>
<tr>
<td></td>
<td>Imbalance $\rightarrow$ overload</td>
<td></td>
<td>7%</td>
</tr>
<tr>
<td></td>
<td>Lost configuration</td>
<td></td>
<td>5%</td>
</tr>
<tr>
<td></td>
<td>High CPU utilization</td>
<td></td>
<td>2%</td>
</tr>
</tbody>
</table>
## Evidence

<table>
<thead>
<tr>
<th>Category</th>
<th>Failure types</th>
<th>Diagnosis &amp; Repair</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software 21%</td>
<td>Link layer loop</td>
<td>Find and fix bugs</td>
<td>19%</td>
</tr>
<tr>
<td></td>
<td>Imbalance → overload</td>
<td></td>
<td>2%</td>
</tr>
<tr>
<td>Hardware 18%</td>
<td>FCS error</td>
<td>Replace cable</td>
<td>13%</td>
</tr>
<tr>
<td></td>
<td>Unstable power</td>
<td>Repair power</td>
<td>5%</td>
</tr>
<tr>
<td>Unknown 23%</td>
<td>Switch stops forwarding</td>
<td>N/A</td>
<td>9%</td>
</tr>
<tr>
<td></td>
<td>Imbalance → overload</td>
<td></td>
<td>7%</td>
</tr>
<tr>
<td></td>
<td>Lost configuration</td>
<td></td>
<td>5%</td>
</tr>
<tr>
<td></td>
<td>High CPU utilization</td>
<td></td>
<td>2%</td>
</tr>
<tr>
<td>Configuration 38%</td>
<td>Errors on multiple switches</td>
<td>Update configuration</td>
<td>32%</td>
</tr>
<tr>
<td></td>
<td>Errors on one switch</td>
<td></td>
<td>6%</td>
</tr>
</tbody>
</table>
Top 10 Failure Events in Clouds

WORST CLOUD OUTAGES OF 2013 (SO FAR)
Failure Models
Failure Models

- Crash failure
Failure Models

- Crash failure
- Timing failure
Failure Models

- Crash failure
- Timing failure
- Response failure
Failure Models

- Crash failure
- Timing failure
- Response failure
- Byzantine failure
Solutions
Solutions

• Overcoming failures after the outage occurs:
  - Diagnosis system
Solutions

- Overcoming failures after the outage occurs:
  - Diagnosis system
  - Accountability system
Solutions

• Overcoming failures after the outage occurs:
  - Diagnosis system
  - Accountability system
  - Fault tolerance system
Solutions

• Overcoming failures after the outage occurs:
  - Diagnosis system
  - Accountability system
  - Fault tolerance system

• Overcoming failures before the outage occurs:
  - Redundancy
Lecture Outline

• Failures

• Correlated failures in decentralized systems
Lecture Outline

- Failures

- Correlated failures in decentralized systems
Example
Example

Peer A

Email App

Peer B
Example

Peer A

Email App

Peer B

Third-party infrastructure components
Example

Third-party infrastructure components
Example

Email App

Peer A

Peer B

Third-party infrastructure components

ISP A

ISP B

ISP C

Power Source
Example

Peer A

Peer B

ISP A

ISP B

ISP C

Third-party infrastructure components

Email App

Power Source
Example

Become unavailable!

Peer A
ISP A

Peer B
ISP B

Third-party infrastructure components

ISP C

Power Source

Example
Example

Lightning strikes Amazon's European cloud

Summary: The lightning strike damaged a power company's transformer, causing disruption to Amazon Web Services's European cloud, and may have affected Microsoft's BPOS as well.

The outage, which Amazon Web Services (AWS) acknowledged on Sunday evening, affected its Dublin-based Elastic Compute Cloud (EC2) and Relational Database Service (RDS) cloud services, among others. The damage to the electricity infrastructure may have affected Microsoft's Business Productivity Online Services (BPOS) cloud as well, Microsoft said in a separate statement.
Existing Efforts

- Service providers allocate or tolerate failures via:
  - diagnosis systems, e.g., Sherlock.
  - fault-tolerant systems, e.g., F10, Skute.
Existing Efforts

• Service providers allocate or tolerate failures via:
  - diagnosis systems, e.g., Sherlock.
  - fault-tolerant systems, e.g., F10, Skute.

• Solving the problem after the outage occurs
Existing Efforts

• Service providers allocate or tolerate failures via:
  - diagnosis systems, e.g., Sherlock.
  - fault-tolerant systems, e.g., F10, Skute.

• Solving the problem after the outage occurs

• We want to prevent the problem before the outage occurs
Existing Efforts

• Service providers allocate or tolerate failures via:
  - diagnosis systems, e.g., Sherlock.
  - fault-tolerant systems, e.g., F10, Skute.

• Solving the problem after the outage occurs

• We want to prevent the problem before the outage occurs

• Recommending truly independent redundancy services when deploying applications
What kind of system we want to build?
Consumer

Node A

Node B

Node C
Select two nodes for redundancy

Consumer

Node A    Node B    Node C
A and B?

Consumer

Node A

Node B

Node C
Consumer

B and C?
Consumer

A and C?

Node A

Node B

Node C
Select two nodes for redundancy: A&B? B&C? or A&C?

- Node A
- Node B
- Node C
Assessing independence by the # of overlapping components between nodes

Consumer

Recommend

Node A

Node B

Node C
Recommender

ISP A  
Power B

ISP B  
Power C

Cloud A, C  
0

Cloud B, C  
1

Cloud A, B  
2

Deployment  
| ⌂ |

ISP A  
Power A

ISP B  
Power A

ISP B  
Power C

Node A

Node B

Node C

ISP A  
Power A

Power B

ISP B

Power C

Consumer
ISP A
Power A
Power B

ISP B
Power A
Power B

ISP B
Power C

Cloud A, C
0

Cloud B, C
1

Cloud A, B
2

Deployment
| n |

Node A

Node B

Node C

ISP A

Power A

Power A

Power B

Power C

Consumer

Recommender
Recommender

ISP A
Power A
Power B

ISP B
Power A
Power B

ISP B
Power B
Power C

Node A
Node B
Node C

\[ \bigcap \neq 1 \]

Deployment

<table>
<thead>
<tr>
<th>n</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Node B, C</td>
<td>1</td>
</tr>
<tr>
<td>Node A, B</td>
<td>2</td>
</tr>
</tbody>
</table>

Consumer
Node A, C
Node B, C
Node A, B

Deployment
<table>
<thead>
<tr>
<th></th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node A, C</td>
<td>0</td>
</tr>
<tr>
<td>Node B, C</td>
<td>1</td>
</tr>
<tr>
<td>Node A, B</td>
<td>2</td>
</tr>
</tbody>
</table>
Recommender

1. Node A, C            0
2. Node B, C            1
3. Node A, B            2

Consumer

Node A

Node B

Node C

Deployment

| |n| |
|---|---|
| 1. Node A, C | 0 |
| 2. Node B, C | 1 |
| 3. Node A, B | 2 |
But, it is not so easy.
Solution 1

Consumer → Recommender → Node A
Node B
Recommender → Node C
Solution 1

Privacy Concern!

Node A

Node B

Node C
Solution 2

- Consumer
- Node A
- Node B
- Node C
- Trusted Third Party
Solution 2

Hard to find!

Node A

Node B

Node C
Solution 3

Consumer

Secure Multiparty Computation

Node A

Node B

Node C
Solution 3

SMPC is difficult to scale!
Intersection cardinality does help

Example
Our solution - iRec
Our solution - iRec

• The first independence recommender sys:
  - achieving our goal
  - preserving privacy of each node
  - practical
Our solution - iRec

- The first independence recommender sys:
  - achieving our goal
  - preserving privacy of each node
  - practical

Preliminary background: P-SOP
P-SOP: Private Jaccard Similarity

- Allows $k$ parties to compute the intersection, union cardinality and Jaccard similarity, without learning other information.

$$J(S_1, S_2, ..., S_n) = \frac{|S_1 \cap S_2 \cap ... \cap S_n|}{|S_1 \cup S_2 \cup ... \cup S_n|}$$
P-SOP: Private Jaccard Similarity

- Allows $k$ parties to compute the intersection, union cardinality and Jaccard similarity, without learning other information.

\[
J(S_1, S_2, ..., S_n) = \frac{|S_1 \cap S_2 \cap ... \cap S_n|}{|S_1 \cup S_2 \cup ... \cup S_n|}
\]
P-SOP: Private Jaccard Similarity

- Allows $k$ parties to compute the intersection, union cardinality and Jaccard similarity, without learning other information.
P-SOP: Private Jaccard Similarity

- Allows $k$ parties to compute the intersection, union cardinality and Jaccard similarity, without learning other information.
P-SOP: Private Jaccard Similarity

- Allows $k$ parties to compute the intersection, union cardinality and Jaccard similarity, without learning other information.
P-SOP: Private Jaccard Similarity

- Allows $k$ parties to compute the intersection, union cardinality and Jaccard similarity, without learning other information.
P-SOP: Private Jaccard Similarity

- Allows $k$ parties to compute the intersection, union cardinality and Jaccard similarity, without learning other information.

But I do not know which elements are in union.

But I do not know which elements are in union.
P-SOP: Private Jaccard Similarity

- Allows \( k \) parties to compute the intersection, union cardinality and Jaccard similarity, without learning other information.

\[
J(S_1, S_2, ..., S_n) = \frac{|S_1 \bigcap S_2 \bigcap ... \bigcap S_n|}{|S_1 \bigcup S_2 \bigcup ... \bigcup S_n|}
\]
P-SOP: Private Jaccard Similarity

- Allows $k$ parties to compute the intersection, union cardinality and Jaccard similarity, without learning other information.

$$J(S_1, S_2, ..., S_n) = \frac{|S_1 \cap S_2 \cap ... \cap S_n|}{|S_1 \cup S_2 \cup ... \cup S_n|}$$
Each party maintains a commutative encryption key
Each party maintains a commutative encryption key
Each party maintains a commutative encryption key.

Commutative encryption holds: $K_x(K_y(E)) = K_y(K_x(E))$
Each party encrypts each item of elements in its dataset through the key
Each party shuffles the encrypted elements in its own dataset.
Each party sends its own encrypted dataset to its successor party.
Each party sends its own encrypted dataset to its successor party.
Each party encrypts each item of elements in the received dataset using its own key.
Shuffle too.
Each party sends its current encrypted dataset to its successor party.
Each party sends its current encrypted dataset to its successor party.
Each party encrypts each item of elements in the received dataset using its own key.
Shuffle.
Each party sends its current encrypted dataset to its successor party.
OK. Now, each party has received its own original dataset which has been encrypted by all the parties.
\[ K_x(K_y(E)) = K_y(K_x(E)) \]
\[ K_x(K_y(E)) = K_y(K_x(E)) \]
I know the # of intersection is 1, and union is 7.
Jaccard = 1/7

Frank

Jaccard = 1/7

K_e(K_d(K_f(3)))
K_e(K_d(K_f(7)))

K_f

K_d

David

Jaccard = 1/7

K_d(K_f(K_e(3)))
K_d(K_f(K_e(5)))
K_d(K_f(K_e(1)))
K_d(K_f(K_e(20)))

K_e

Eve

Jaccard = 1/7

K_f(K_e(K_d(3)))
K_f(K_e(K_d(10)))
K_f(K_e(K_d(11)))
Select two nodes for redundancy: A&B? B&C? or A&C?
iRec: Step 1

Consumer → iRec

Node A → Node B → Node C

ISP A → Power A

Power B

ISP B

Power C
iRec: Step 2

Consumer

ISP A
Power A

Node A

ISP B
Power B

Node B

ISP C
Power C

Node C
iRec: Step 3

Consumer

iRec

ISP A
Power A
Power B

ISP B
Power A
Power B

ISP B
Power C

Node A

Node B

Node C

ISP A
Power A

Power B

ISP B

Power C
iRec: Step 3

Consumer

ISP A
Power A
Power B

ISP B
Power A
Power B

ISP B
Power C

Node A

Node B

Node C

P-SOP

ISP A
Power A

Power B

ISP B
Power C

Power B

ISP B
iRec: Step 3

\[ \text{ISP A} \sqcap \text{ISP B} = 2 \]

Node A

Node B

Node C

ISP A

Power A

Power B

ISP B

Power A

Power B

ISP B

Power C

ISP B

Power C

Consumer

iRec
iRec: Step 3

Consumer

ISP A
Power A
Power B

ISP B
Power A
Power B

ISP B
Power C

Node A

Node B

Node C

ISP A
Power A
Power B

Power A
Power B

Power C

iRec

Node A

Node B

Node C

ISP B
iRec: Step 3

\[ J = \frac{2}{4} \]

ISP A
- Power A
- Power B

ISP B
- Power A
- Power B

ISP B
- Power C

Node A

Node B

Node C

Consumer

iRec
iRec: Step 3

J = 2 / 4

<table>
<thead>
<tr>
<th>Deployment</th>
<th>J</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node A, B</td>
<td>0.5</td>
</tr>
</tbody>
</table>
iRec: Step 3

<table>
<thead>
<tr>
<th>Deployment</th>
<th>J</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node A, B</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Node A

Node B

Node C

ISP A

Power A

ISP B

Power A

ISP B

Power C

ISP A

Power A

ISP B

Power B

ISP B

Power B

ISP B

Power C

Node A

Node B

Node C

P-SOP

Cloud A, C 0

Cloud B, C 0.25

Cloud A, B 0.5

Deployment

iRec

Consumer
iRec: Step 3

<table>
<thead>
<tr>
<th>Deployment</th>
<th>$J$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node A, B</td>
<td>0.5</td>
</tr>
</tbody>
</table>

$\bigcap = 1$
iRec: Step 3

<table>
<thead>
<tr>
<th>ISP</th>
<th>Power</th>
<th>ISP</th>
<th>Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A</td>
<td>B</td>
<td>A</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td></td>
<td>B</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Node</th>
<th>ISP</th>
<th>Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>B</td>
<td>C</td>
</tr>
</tbody>
</table>

Deployment | J  
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Node A, B</td>
<td>0.5</td>
</tr>
</tbody>
</table>

\( J = 1 \quad \cup = 4 \)
iRec: Step 3

<table>
<thead>
<tr>
<th>Deployment</th>
<th>$J$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node A, B</td>
<td>0.5</td>
</tr>
</tbody>
</table>

$\cup = 4$
$J = 1/4$
iRec: Step 3

<table>
<thead>
<tr>
<th>Deployment</th>
<th>J</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node B, C</td>
<td>0.2</td>
</tr>
<tr>
<td>Node A, B</td>
<td>0.5</td>
</tr>
</tbody>
</table>

\[
\bigcup = 1 \\
\bigcap = 4 \\
J = 1/4
\]
iRec: Step 3

<table>
<thead>
<tr>
<th>Deployment</th>
<th>J</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node B, C</td>
<td>0.2</td>
</tr>
<tr>
<td>Node A, B</td>
<td>0.5</td>
</tr>
</tbody>
</table>

ISP A
Power A
Power B

ISP B
Power C

Node A

Node B

Node C

Consumer

iRec

P-SOP
iRec: Step 3

Consumer

ISP A
Power A
Power B

Node A

ISP B
Power C

Node B

Node C

\[ \bigcap = 0, \]

<table>
<thead>
<tr>
<th>Deployment</th>
<th>( J )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node B, C</td>
<td>0.2</td>
</tr>
<tr>
<td>Node A, B</td>
<td>0.5</td>
</tr>
</tbody>
</table>

ISP A
Power A

Power B

ISP B

Power C
iRec: Step 3

\[ \cap = 0, \quad \cup = 5, \]

<table>
<thead>
<tr>
<th>Deployment</th>
<th>( J )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node B, C</td>
<td>0.2</td>
</tr>
<tr>
<td>Node A, B</td>
<td>0.5</td>
</tr>
</tbody>
</table>

ISP A
- Power A
- Power B

ISP B
- Power C

ISP A
- Power A

ISP B
- Power B

ISP B
- Power C

Node A

Node B

Node C

Consumer

iRec
iRec: Step 3

$\bigcap = 0, \quad \bigcup = 5, \quad J = 0/5$

<table>
<thead>
<tr>
<th>Deployment</th>
<th>$J$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node B, C</td>
<td>0.2</td>
</tr>
<tr>
<td>Node A, B</td>
<td>0.5</td>
</tr>
</tbody>
</table>

ISP A
- Power A
- Power B

ISP B
- Power C

Node A

Node B

Node C

Cloud A, C
Cloud B, C
Cloud A, B
iRec: Step 3

\[ \bigcap = 0, \quad \bigcup = 5, \quad J = 0/5 \]

<table>
<thead>
<tr>
<th>Deployment</th>
<th>( J )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node A, C</td>
<td>0</td>
</tr>
<tr>
<td>Node B, C</td>
<td>0.25</td>
</tr>
<tr>
<td>Node A, B</td>
<td>0.5</td>
</tr>
</tbody>
</table>

ISP A
- Power A
- Power B

ISP B
- Power C

ISP C
- Power C

Node A

Node B

Node C

Consumer

iRec
**iRec: Step 4**

<table>
<thead>
<tr>
<th>Deployment</th>
<th>J</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node A, C</td>
<td>0</td>
</tr>
<tr>
<td>Node B, C</td>
<td>0.25</td>
</tr>
<tr>
<td>Node A, B</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Diagram showing the connection between the Consumer, iRec, Node A, Node B, Node C, ISP A, Power A, Power B, ISP B, and Power C.
iRec: Step 5

<table>
<thead>
<tr>
<th>Deployment</th>
<th>J</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node A, C</td>
<td>0</td>
</tr>
<tr>
<td>Node B, C</td>
<td>0.25</td>
</tr>
<tr>
<td>Node A, B</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Deployment J

- Node A, C: 0
- Node B, C: 0.25
- Node A, B: 0.5
iRec: Step 6

<table>
<thead>
<tr>
<th>Deployment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node A, C</td>
</tr>
<tr>
<td>Node B, C</td>
</tr>
<tr>
<td>Node A, B</td>
</tr>
</tbody>
</table>

Ranking List

Consumer

ISP A  
Power A

ISP B  
Power B

ISP C  
Power C
Failure is a very important topic

It is very hard to solve in decentralized system
Thanks!

Questions?