NetPilot: Automating Datacenter Network Failure Mitigation

Xin Wu, Daniel Turner, Chao-Chih Chen, David A. Maltz, Xiaowei Yang, Lihua Yuan, Ming Zhang
Failures are Common and Harmful

- Network failures are common

10,000+ switches
Failures are Common and Harmful

• Network failures are common

• Failures cause long down times
Failures are Common and Harmful

Six-month failure logs of production datacenters

25% of failures take 13+ hours to repair

Time from detection to repair (minutes)
Failures are Common and Harmful

• Failures are common due to VERY large datacenters

• Failures cause long down times

• Long failure duration → large revenue loss
Failures are Common and Harmful

Data Center Outages Generate Big Losses

Downtime in a data center can cost an average of $505,500 per incident, according to a Ponemon Institute study.

By Chandler Harris InformationWeek
May 12, 2011 01:22 PM

Sure data center failures are costly, but how costly? Try an average of $5,600 per minute, according to a study of outages at U.S.-based data centers by the Ponemon Institute.

• Long failure duration → large revenue loss
How to Shorten Failure Recovery Time?
Previous Work

• Conventional failure recovery takes 3 steps

Detection → Diagnosis → Repair

SNMP

passive

ping

active
Previous Work

• Conventional failure recovery takes 3 steps

Detection → Diagnosis → Repair

• Failure localization/diagnosis
  – [M. K. Aguilera, SOSP’03]
  – [M. Y. Chen, NSDI’04]
  – [R. R. Kompella, NSDI ’05]
  – [P. Bahl, SIGCOMM’07]
  – [S. Kandula, SIGCOMM’09]...
Automating Failure Diagnosis is Challenging

• Root causes are deep in network stack

• Diagnosis involves multiple parties
<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>Configuration 38%</td>
<td>Errors on multiple switches</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Errors on one switch</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. Root causes are deep in the network stack

- Six-month failure logs from several production DCNs
<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</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Switch stops forwarding</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unknown 23%</td>
<td>Imbalance</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Overload</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Lost configuration</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>High CPU utilization</td>
<td></td>
<td></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>

- **Six-month failure logs from several production DCNs**

**1. Root causes are deep in the network stack**

**2. Diagnosis involves multiple parties**
<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>Link layer 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></td>
<td></td>
<td>9%</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</td>
<td>Update configuration</td>
<td>32%</td>
</tr>
<tr>
<td></td>
<td>switches</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Errors on one switch</td>
<td></td>
<td>6%</td>
</tr>
</tbody>
</table>

- Six-month failure logs from several production DCNs

Failure Diagnosis Requires Human Intervention!
Can we do something other than failure diagnosis?
NetPilot: Mitigating rather than Diagnosing Failures

- **Mitigate** failure symptoms ASAP, at the cost of reduced capacity
NetPilot: Mitigating rather than Diagnosing Failures

- **Mitigate** failure symptoms ASAP, at the cost of reduced capacity
NetPilot Benefits

• Short recovery time
• Small network disruption
• Low operation cost
Failure Mitigation is Effective

• Most failures can be mitigated by **simple** actions

• Mitigation is feasible due to **redundancy**
<table>
<thead>
<tr>
<th>Category</th>
<th>Failure types</th>
<th>Mitigation</th>
<th>Repair</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software 21%</td>
<td>Link layer loop</td>
<td>Deactivate port</td>
<td>Find and fix bugs</td>
<td>19%</td>
</tr>
<tr>
<td></td>
<td>Imbalance-triggered overload</td>
<td>Restart switch</td>
<td></td>
<td>2%</td>
</tr>
<tr>
<td>Hardware 18%</td>
<td>FCS error</td>
<td>Deactivate port</td>
<td>Replace cable</td>
<td>13%</td>
</tr>
<tr>
<td></td>
<td>Unstable power</td>
<td>Deactivate switch</td>
<td>Repair power</td>
<td>5%</td>
</tr>
<tr>
<td>Unknown 23%</td>
<td>Switch stops forwarding</td>
<td>Restart switch</td>
<td></td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>Imbalance-triggered overload</td>
<td>Restart switch</td>
<td></td>
<td>7%</td>
</tr>
<tr>
<td></td>
<td>Lost configuration</td>
<td>Restart switch</td>
<td></td>
<td>5%</td>
</tr>
<tr>
<td></td>
<td>High CPU utilization</td>
<td>Restart switch</td>
<td></td>
<td>2%</td>
</tr>
<tr>
<td>Configuration 38%</td>
<td>Errors on multiple switches</td>
<td>n/a</td>
<td>Update configuration</td>
<td>32%</td>
</tr>
<tr>
<td></td>
<td>Errors on single switch</td>
<td>Deactivate switch</td>
<td></td>
<td>6%</td>
</tr>
<tr>
<td>Category</td>
<td>Failure types</td>
<td>Mitigation</td>
<td>Repair</td>
<td>%</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------------------</td>
<td>----------------------</td>
<td>-----------------</td>
<td>----</td>
</tr>
<tr>
<td>Software 21%</td>
<td>Link layer loop</td>
<td>Deactivate port</td>
<td>Find and fix bugs</td>
<td>19%</td>
</tr>
<tr>
<td></td>
<td>Imbalance-triggered overload</td>
<td>Restart switch</td>
<td></td>
<td>2%</td>
</tr>
<tr>
<td>Hardware 18%</td>
<td>FCS error</td>
<td>Deactivate port</td>
<td>Replace cable</td>
<td>13%</td>
</tr>
<tr>
<td></td>
<td>Unstable power</td>
<td>Deactivate switch</td>
<td>Repair power</td>
<td>5%</td>
</tr>
<tr>
<td>Unknown 23%</td>
<td>Switch stops forwarding</td>
<td>Restart switch</td>
<td>N/A</td>
<td>9%</td>
</tr>
<tr>
<td></td>
<td>Imbalance-triggered overload</td>
<td>Restart switch</td>
<td></td>
<td>7%</td>
</tr>
<tr>
<td></td>
<td>Lost configuration</td>
<td>Restart switch</td>
<td></td>
<td>5%</td>
</tr>
<tr>
<td></td>
<td>High CPU utilization</td>
<td>Restart switch</td>
<td></td>
<td>2%</td>
</tr>
<tr>
<td>Configuration 38%</td>
<td>Errors on multiple switches</td>
<td>n/a</td>
<td>Update configuration</td>
<td>32%</td>
</tr>
<tr>
<td></td>
<td>Errors on single switch</td>
<td>Deactivate switch</td>
<td></td>
<td>6%</td>
</tr>
<tr>
<td>Category</td>
<td>Failure types</td>
<td>Mitigation</td>
<td>Repair</td>
<td>%</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------------</td>
<td>--------------------</td>
<td>--------------</td>
<td>----</td>
</tr>
<tr>
<td>Software 21%</td>
<td>Link layer loop</td>
<td>Deactivate port</td>
<td>Find and fix bugs</td>
<td>19%</td>
</tr>
<tr>
<td></td>
<td>Imbalance-triggered overload</td>
<td>Restart switch</td>
<td></td>
<td>2%</td>
</tr>
<tr>
<td>Hardware 18%</td>
<td>FCS error</td>
<td>Deactivate port</td>
<td>Replace cable</td>
<td>13%</td>
</tr>
<tr>
<td></td>
<td>Unstable power</td>
<td>Deactivate switch</td>
<td>Repair power</td>
<td>5%</td>
</tr>
<tr>
<td>Unknown 23%</td>
<td>Switch stops forwarding</td>
<td>Restart switch</td>
<td></td>
<td>7%</td>
</tr>
<tr>
<td></td>
<td>Lost configuration</td>
<td>Restart switch</td>
<td></td>
<td>5%</td>
</tr>
<tr>
<td></td>
<td>High CPU utilization</td>
<td>Restart switch</td>
<td></td>
<td>2%</td>
</tr>
<tr>
<td>Configuration 38%</td>
<td>Errors on multiple switches</td>
<td>n/a</td>
<td>Update configuration</td>
<td>32%</td>
</tr>
<tr>
<td></td>
<td>Errors on single switch</td>
<td>Deactivate switch</td>
<td></td>
<td>6%</td>
</tr>
</tbody>
</table>
Mitigation Made Possible by Redundancy

- Redundancy → deactivation unlikely to partition / overload the network
Mitigation Made Possible by Redundancy

- Redundancy $\Rightarrow$ deactivation unlikely to partition / overload the network
Outline

• Automating failure diagnosis is challenging

• Failure mitigation is effective

• How to automate mitigation?

• NetPilot evaluations

• Conclusion
A Strawman NetPilot: Trial-and-error

1. Network failure
2. Localization
   - Roll back if necessary
   - Execute an action
   - Failure mitigated?
      - No
      - Yes
         - End
NetPilot: Challenges & Solutions

1. Blind trial-and-error takes a long time
NetPilot: Challenges & Solutions

1. Blind trial-and-error takes a long time

Network failure

Localization

Roll back if necessary

Execute an action

Failure mitigated?

Yes

End

No
NetPilot: Challenges & Solutions

1. Network failure

   Localization

   Estimate impact

   Roll back if necessary

   Execute an action

   Failure mitigated?

   Yes

   End

   No

2. Partition/overload network

Impact estimation
NetPilot: Challenges & Solutions

Network failure

Localization

2. Partition/overload network

Roll back if necessary

Execute an action

Failure mitigated?

Yes

End

No
NetPilot: Challenges & Solutions

Network failure

Localization

Estimate impact

Rank actions

Execute an action

Roll back if necessary

Failure mitigated?

End

3. Different actions have different side-effects

Rank actions based on impact
Failure Specific Localization

• Limited # of failure types
• Domain knowledge improves accuracy

<table>
<thead>
<tr>
<th>Failure types</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Link layer loop</td>
</tr>
<tr>
<td>2. Imbalance-triggered overload</td>
</tr>
<tr>
<td>3. FCS error</td>
</tr>
<tr>
<td>4. Unstable power</td>
</tr>
<tr>
<td>5. Switch stops forwarding</td>
</tr>
<tr>
<td>6. Imbalance-triggered overload</td>
</tr>
<tr>
<td>7. Lost configuration</td>
</tr>
<tr>
<td>8. High CPU utilization</td>
</tr>
<tr>
<td>9. Errors on multiple switches</td>
</tr>
<tr>
<td>10. Errors on single switch</td>
</tr>
</tbody>
</table>
Example: Frame Check Sequence (FCS) Errors

- 13% of all the failures
- Cut-through switching
  - Forward frames before checksums are verified
- Increase application latency
Localizing FCS Errors

error frames seen on $L$

frames corrupted by $L$

$e_l = \sum_{k \neq l} p_k x_k m_{kl} + p_l x_l$

frames corrupted by other links & traverse $L$

- $x_L$: link corruption rate
- # of variables = # of equations = # of links
- Corrupted links: $x_L > 0$
NetPilot Overview

1. Network failure
2. Localization
3. Estimate impact
4. Rank actions
5. Execute an action

Roll back if necessary

Failure mitigated?

End

No

Yes
Impact Metrics

• Derived from *Service Level Agreement* (SLA)
  – Availability: *online_server_ratio*
  – Packet loss: *total_lost_pkt*
  – latency: *max_link_utilization*
    • Small link utilization $\Rightarrow$ small (queuing) delay

• *Total_lost_pkt* & *max_link_utilization* derived from *utilization of individual links*
Estimating Link Utilization

- # of flows >> redundant paths
  - Traffic evenly distributed under ECMP
- Estimate the load contributed by each flow on each link
- Sum up the loads to compute utilization
Link Utilization Estimation is Highly Accurate

- 1-month traffic from a 8000-server network
  - Log socket events on each server
- Ground truth: SNMP counters

[Diagram showing CDF distribution with values labeled 0, 0.25, 0.5, 0.75, and 1 along the y-axis and (estimation - switch counter) / capacity along the x-axis.]
NetPilot Overview

Network failure

Localization

Estimate impact

Choose the action with the least impact

Roll back if necessary

Failure mitigated?

Yes

End

No

Rank actions

Execute an action
Outline

• Automating failure diagnosis is challenging

• Failure mitigation is effective

• How to automate mitigation?
  – Localization → impact estimation → ranking

• NetPilot evaluations
  – Mitigating load imbalance
  – Mitigating FCS errors
  – Mitigating overload

• Conclusion
Load Imbalance

- $\text{Agg}_a$ stops receiving traffic
- Localize to 4 suspects
Mitigating Load Imbalance

Load evenly splitted
Mitigating Load Imbalance

Load (Gbps)

0 5 10 15 20 25 30 35

Time (minutes)

0:00 0:05 0:10 0:15 0:20 0:25

core_a -> agg_a
core_b -> agg_a
core_a -> agg_b
core_b -> agg_b

Agg_a stops receiving traffic
Mitigating Load Imbalance

Diagram showing network nodes `core_a`, `core_b`, `Agg_a`, and `Agg_b` with data flow arrows `core_a -> agg_a`, `core_b -> agg_a`, `core_a -> agg_b`, and `core_b -> agg_b`. There is a detected & rebooted core `core_b` indicated with an oval and arrow.
Mitigating Load Imbalance

Diagram showing the network topology with core_a and core_b, Agg_a, and Agg_b, and the load distribution over time. The diagram illustrates the load (in Gbps) on different links: core_a -> agg_a, core_a -> agg_b, core_b -> agg_a, and core_b -> agg_b. The graph highlights the load peaks and the proposed solution of rebooting core_a to mitigate the load imbalance.
Mitigating Load Imbalance

[Diagram showing network topology and load distribution over time with specific events annotated, such as 'Reboot Agg_a']
Mitigating Load Imbalance

- core_a
- core_b
- Agg_a
- Agg_b

Load (Gbps)

Time (minutes)

Mitigation confirmed
Fast FCS Error Mitigation

Human operator: after 11 trials in 3.5 hours, 2 out of 28 ports are deactivated

NetPilot: deactivates 2 links in 1 trial within 15 minutes
Fast FCS Error Mitigation

**Human operator:** after 11 trials in **3.5 hours**, 2 out of 28 ports are deactivated

**NetPilot:** deactivates 2 links in 1 trial within **15 minutes**
Mitigating Link Overload

- Mitigate overload by **deactivating** healthy links
Mitigating Link Overload

- Mitigate overload by **deactivating** healthy links
Mitigating Link Overload

- Mitigate overload by **deactivating** healthy links
  - Many candidate links in production networks
  - Choose the link(s) with the least impact

```
+-----------------+-----------------+-----------------+-----------------+-----------------+
|     core₁      |     core₂      |     core₁      |     core₂      |     core₁      |
| 1.5            | 1.5            | 1              | 1.5            | 0              |
| agg            | agg            | agg            | agg            | agg            |
| 3              | 3              | 3              | 3              | 3              |
```

```
lost 0.5
```
Action Ranking Lowers Link Utilization

- Replay 97 overload incidents due to link failures
Conclusion

• Mitigation reduces failure recovery time
  – Simple actions are effective
  – Made possible by redundancy

• NetPilot: automating failure mitigation
  – Recovery time: hour → minutes
  – Several mitigation scenarios deployed in Bing
Thank You!

Detection → **NetPilot: Automated Mitigation** → Diagnosis → Repair

netpilot@microsoft.com
NetPilot Shortens Recovery Time

• Time from detection to mitigation
  – 6 months, many production datacenters

NetPilot mitigate 3 types of failures all with in 30 minutes

Operators work around 50% failures in 2 HOURS