RECOVERY-ORIENTED COMPUTING

Why do Internet services fail, and what can be done about it?

David Oppenheimer,
Archana Ganapathi, and David Patterson

Computer Science Division
University of California at Berkeley

4th USENIX Symposium on Internet Technologies and Systems
March 2003

Motivation

- **Internet service availability is important**
  - email, instant messenger, web search, e-commerce, …
- **User-visible failures are relatively frequent**
  - especially if use non-binary definition of "failure"
- **To improve availability, must know what causes failures**
  - know where to focus research
  - objectively gauge potential benefit of techniques
- **Approach: study failures from real Internet svcs.**
  - evaluation includes impact of humans & networks

Outline

- **Describe methodology and services studied**
- **Identify most significant failure root causes**
  - **source**: type of component
  - **impact**: number of incidents, contribution to TTR
- **Evaluate HA techniques to see which of them would mitigate the observed failures**
- **Drill down on one cause: operator error**
- **Future directions for studying failure data**

Methodology

- **Obtain "failure" data from three Internet services**
  - two services: problem tracking database
  - one service: post-mortems of user-visible failures

Methodology

- **Obtain "failure" data from three Internet services**
  - two services: problem tracking database
  - one service: post-mortems of user-visible failures
- **We analyzed each incident**
  - failure root cause
    - » hardware, software, operator, environment, unknown
  - type of failure
    - » "component failure" *vs.* "service failure"
  - time to diagnose + repair (TTR)

Methodology

- **Obtain "failure" data from three Internet services**
  - two services: problem tracking database
  - one service: post-mortems of user-visible failures
- **We analyzed each incident**
  - failure root cause
    - » hardware, software, operator, environment, unknown
  - type of failure
    - » "component failure" *vs.* "service failure"
  - time to diagnose + repair (TTR)
- **Did not look at security problems**

Comparing the three services

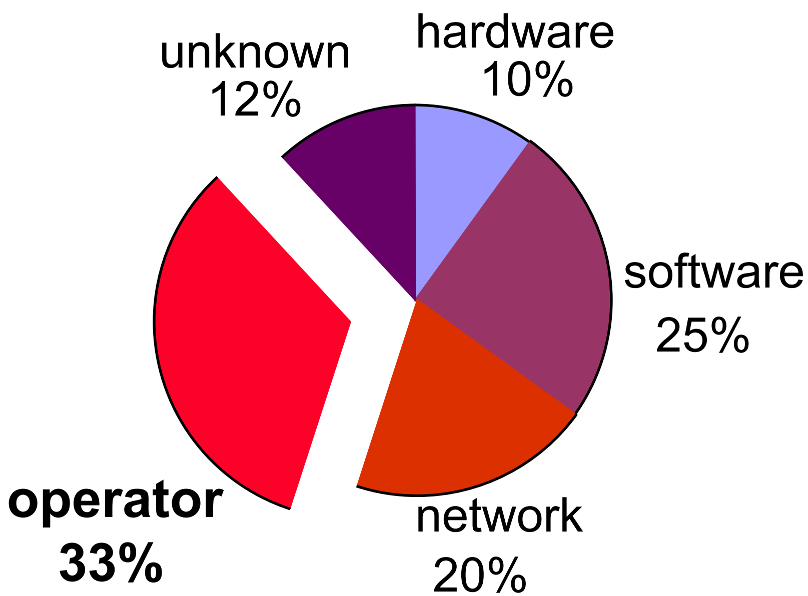
characteristic	*Online*	*ReadMostly*	*Content*
hits per day	~100 million	~100 million	~7 million
# of machines	~500 @ 2 sites	> 2000 @ 4 sites	~500 @ ~15 sites
front-end node architecture	custom s/w; Solaris on SPARC, x86	custom s/w; open-source OS on x86	custom s/w; open-source OS on x86;
back-end node architecture	Network Appliance filers	custom s/w; open-source OS on x86	custom s/w; open-source OS on x86
period studied	7 months	6 months	3 months
# component failures	296	N/A	205
# service failures	40	21	56

Outline

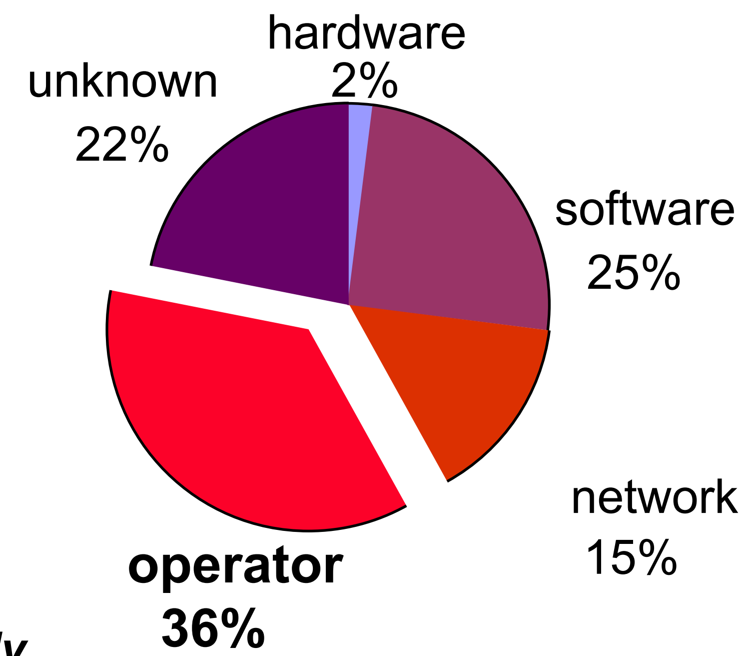
- Describe methodology and services studied
- Identify most significant failure root causes
  - **source**: type of component
  - **impact**: number of incidents, contribution to TTR
- Evaluate HA techniques to see which of them would mitigate the observed failures
- Drill down on one cause: operator error
- Future directions for studying failure data

Failure cause by % of service failures

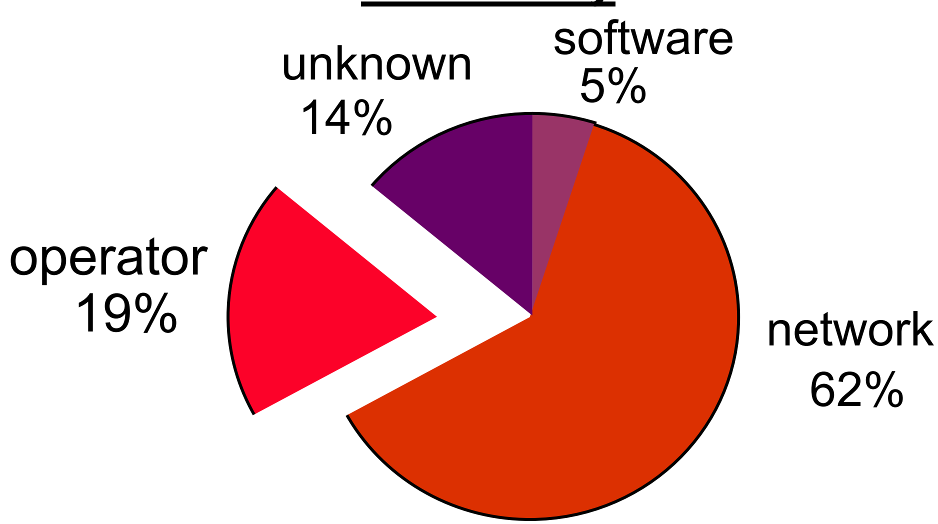
Online



Content

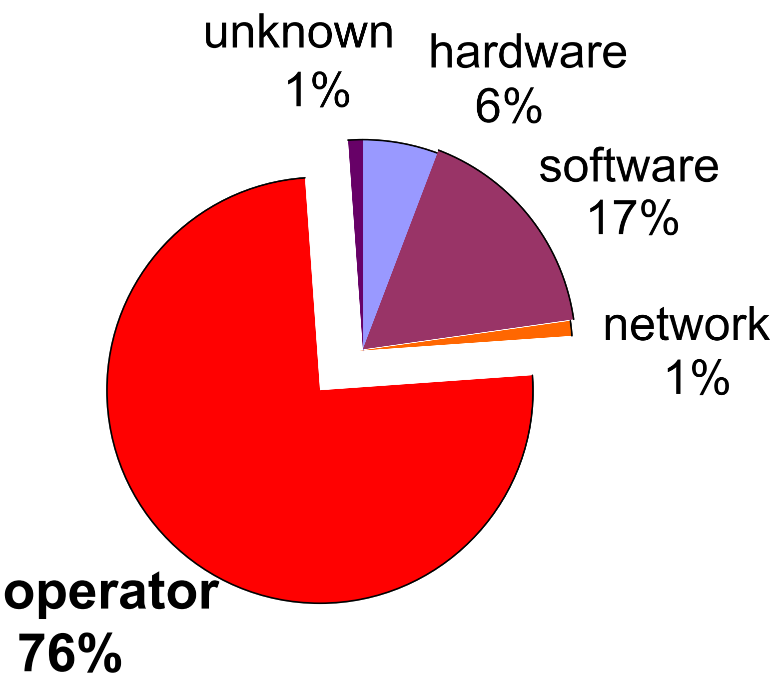


ReadMostly

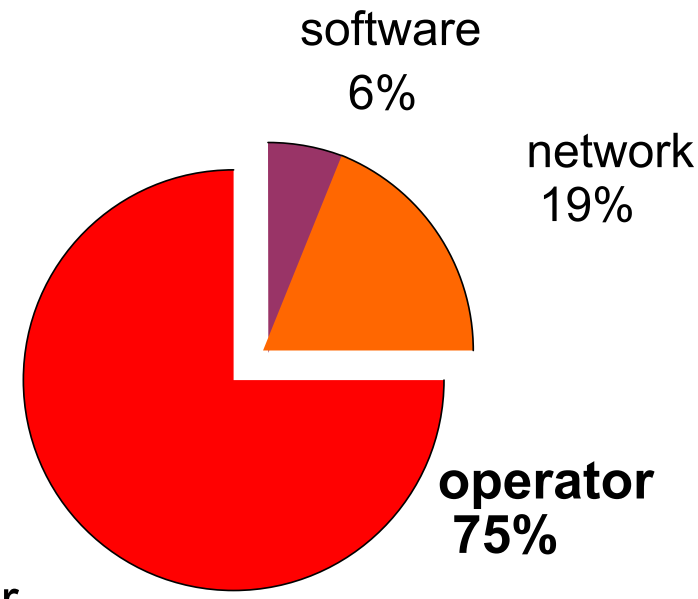


Failure cause by % of TTR

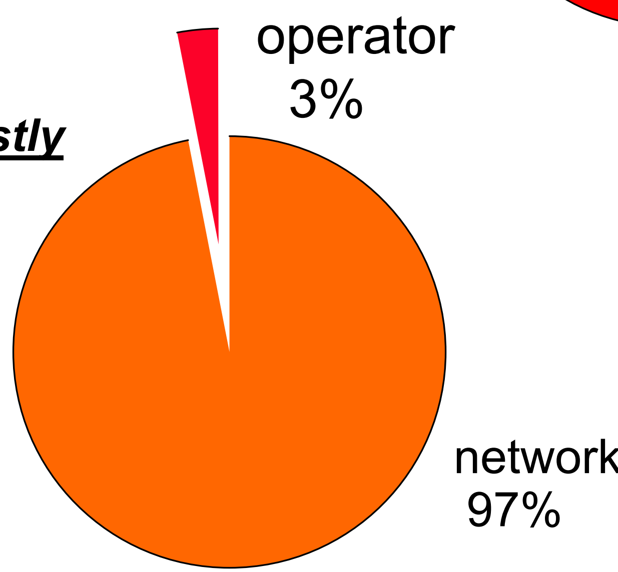
Online



Content



ReadMostly



Most important failure root cause?

- **Operator error generally the largest cause of service failure**
  - even more significant as fraction of total "downtime"
  - configuration errors > 50% of operator errors
  - generally happened when making changes, not repairs
- **Network problems significant cause of failures**

Related work: failure causes

- **Tandem systems** (Gray)
  - 1985: **Operator 42%,** software 25%, hardware 18%
  - 1989: Operator 15%, software 55%, hardware 14%
- **VAX** (Murphy)
  - 1993: **Operator 50%,** software 20%, hardware 10%
- **Public Telephone Network** (Kuhn, Enriquez)
  - 1997: **Operator 50%,** software 14%, hardware 19%
  - 2002: **Operator 54%,** software 7%, hardware 30%

Outline

- Describe methodology and services studied
- Identify most significant failure root causes
  - **source**: type of component
  - **impact**: number of incidents, contribution to TTR
- Evaluate HA techniques to see which of them would mitigate the observed failures
- Drill down on one cause: operator error
- Future directions for studying failure data

Potential effectiveness of techniques?

technique
post-deployment correctness testing*
expose/monitor failures*
redundancy*
automatic configuration checking
post-deploy. fault injection/load testing
component isolation*
pre-deployment fault injection/load test
proactive restart*
pre-deployment correctness testing*

* *indicates technique already used by* Online

Potential effectiveness of techniques?

technique	failures avoided / mitigated
post-deployment correctness testing*	26
expose/monitor failures*	12
redundancy*	9
automatic configuration checking	9
post-deploy. fault injection/load testing	6
component isolation*	5
pre-deployment fault injection/load test	3
proactive restart*	3
pre-deployment correctness testing*	2

(40 service failures examined)

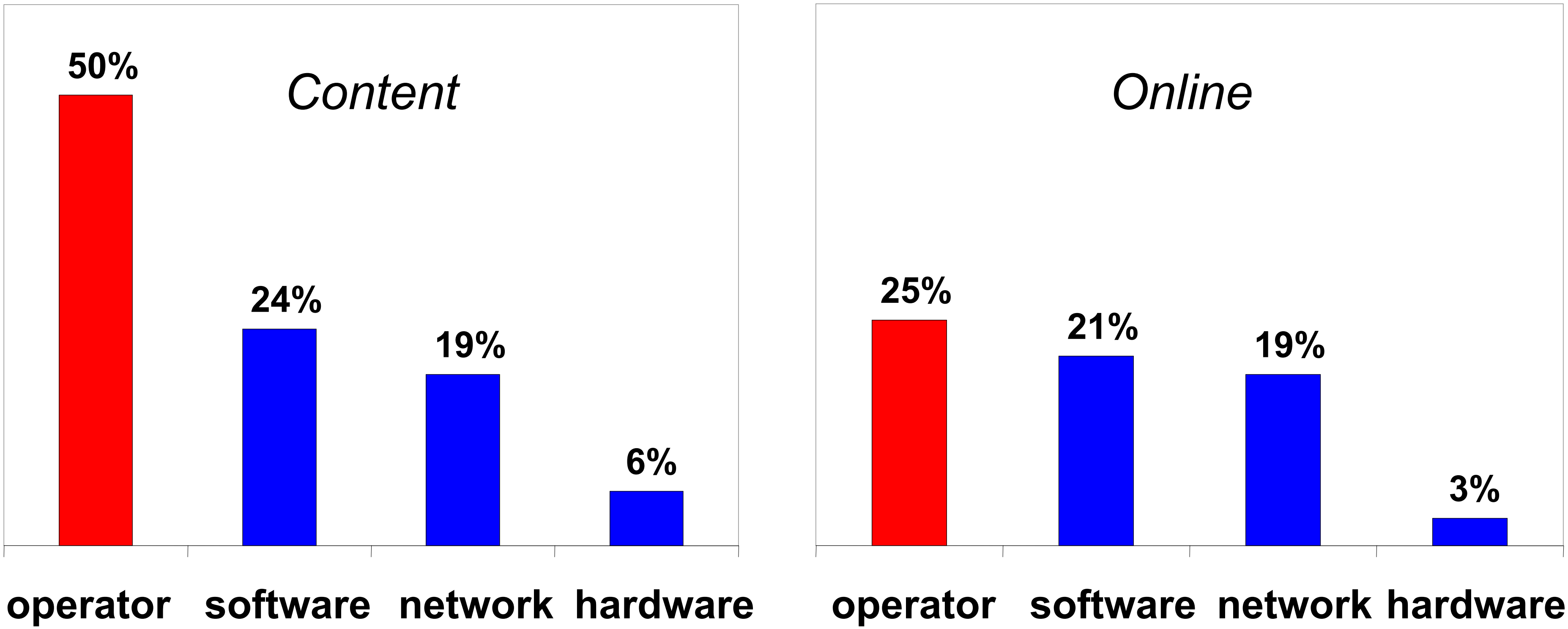
Outline

- Describe methodology and services studied
- Identify most significant failure root causes
  - **source**: type of component
  - **impact**: number of incidents, contribution to TTR
- Evaluate existing techniques to see which of them would mitigate the observed failures
- Drill down on one cause: operator error
- Future directions for studying failure data

Drilling down: operator error

Why does operator error cause so many svc. failures?

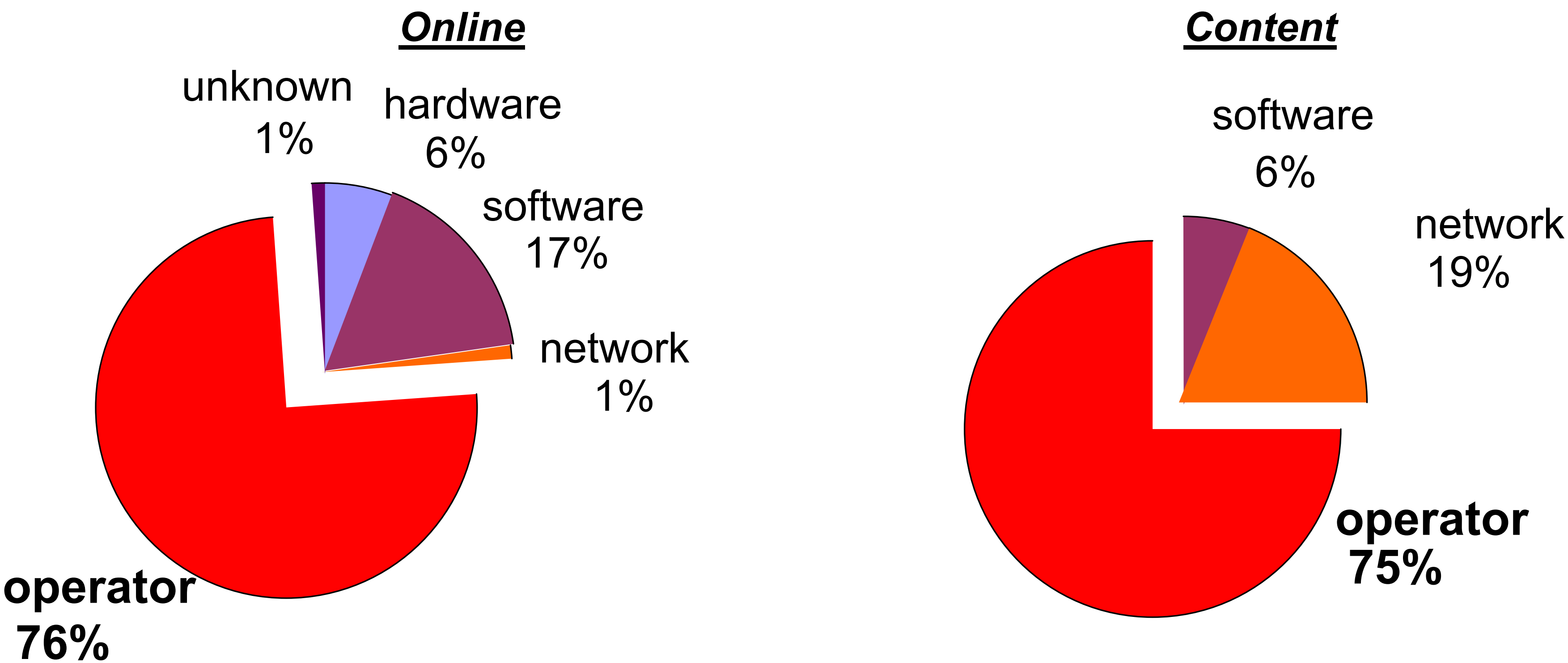
% of component failures resulting in service failures



Existing techniques (*e.g.*, redundancy) are minimally effective at masking operator error

Drilling down: operator error TTR

Why does operator error contribute so much to TTR?



Detection and diagnosis difficult because of non-failstop failures and poor error checking

Future directions in studying failures

- **Quantify impact of of operational practices**
- **Study additional types of sites**
  - transactional, intranets, peer-to-peer
- **Create a public failure data repository**
  - standard taxonomy of failure causes
  - standard metrics for impact
  - techniques for automatic anonymization
  - security (not just reliability)
  - automatic analysis (mining for trends, fixes, attacks, …)
- **Perform controlled laboratory experiments**

Conclusion

- **Operator error large cause of failures, downtime**
- **Many failures could be mitigated with**
  - better post-deployment testing
  - automatic configuration checking
  - better error detection and diagnosis
- **Longer-term: concern for operators must be built into systems from the ground up**
  - make systems robust to operator error
  - reduce time it takes operators to detect, diagnose, and repair problems

Willing to contribute failure data, or information about problem detection/diagnosis techniques?

http://roc.cs.berkeley.edu/projects/faultmanage/

davidopp@cs.berkeley.edu