Automated Diagnosis of Chronic Performance Problems in Production Systems

Soila P. Kavulya

Christos Faloutsos , *CMU*

Greg Ganger, *CMU*

Matti Hiltunen, *AT&T Labs*

Priya Narasimhan, *CMU (Advisor)*

Motivation

- Major outages are rare in production systems
  - Such blackouts also detected by alarms

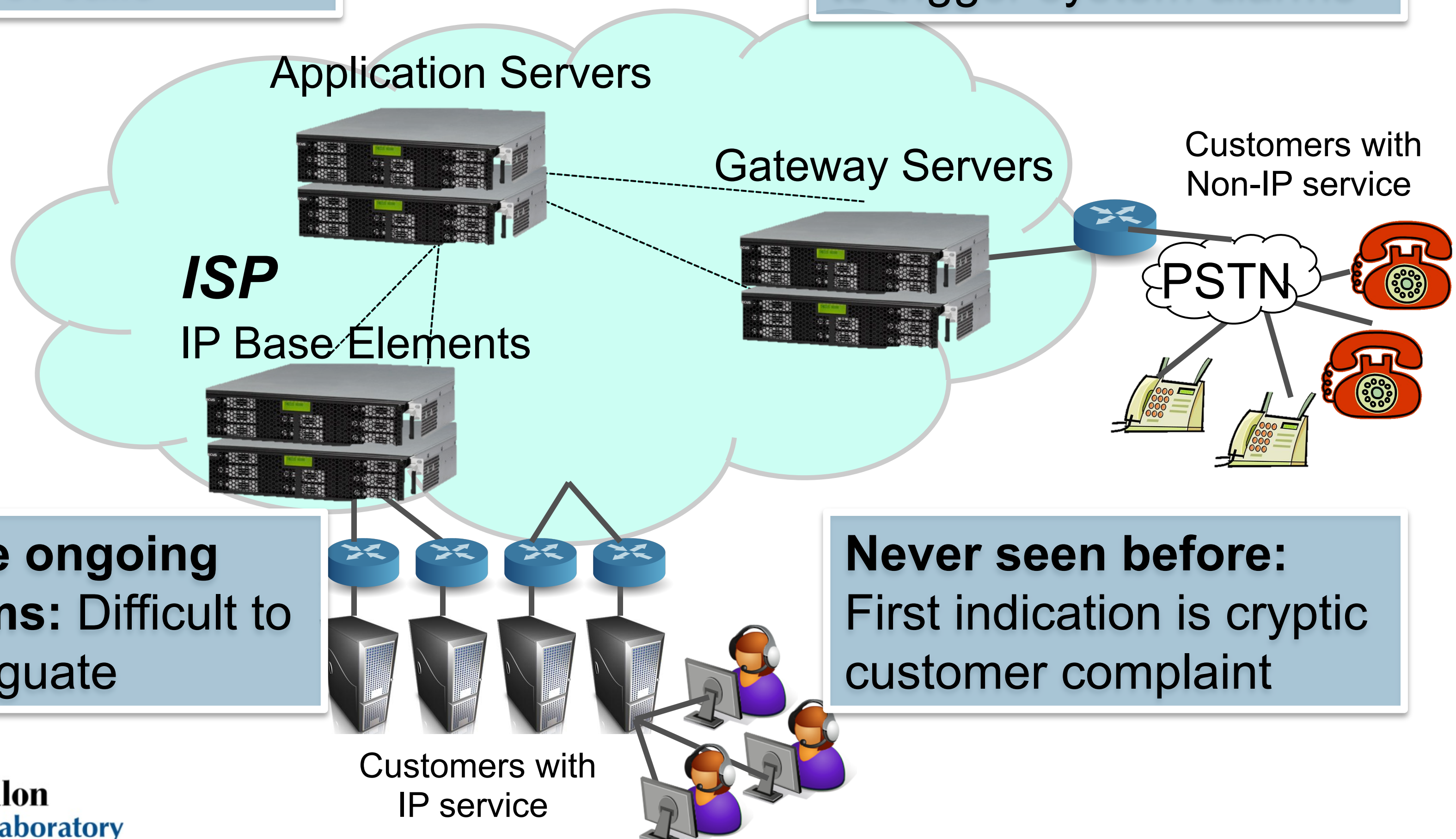
- **Chronic performance problems** (*brownouts*)
  - System still works, but with degraded performance
  - Problem typically affects subset of users/requests
  - Admins often unaware until the user complains

- Chronics are fairly common
  - Production VoIP system:
    – Source of 42% of failed calls in worst month of outages
  - Production Hadoop cluster (OpenCloud):
    – Source of 78% of reported problems in 11-month period

Challenges (1)

Scale: Thousands of network elements, millions of calls

Under radar: Symptoms often not severe enough to trigger system alarms



Multiple ongoing problems: Difficult to disambiguate

Never seen before: First indication is cryptic customer complaint

Challenges (2)

- **Labeled failure-data** not always available
  - Difficult to diagnose problems not encountered before
- **Desired level of instrumentation** might not be possible
  - Existing vendor instrumentation with limited control
  - Cost of adding instrumentation might be high
  - Instrumentation might be diverse, at different sampling rate

Outline

- Thesis statement
- Approach
  - Instrumentation
  - Anomaly detection
  - Problem localization
- Experimental evaluation
  - Fault injection
  - Case studies
- Extensions
- Conclusion

Thesis Statement

Diagnosis of chronic performance problems in production systems is possible through the analysis of common **white-box logs** to extract local behavior and system-wide dependencies, coupled with the analysis of common **black-box metrics** to identify the resource at fault.

Goals and Non-goals

- Goals of approach
  - Diagnosis using existing instrumentation in production systems
  - Anomaly detection in the absence of labeled failure-data
  - Differentiation of workload changes from anomalies

- Non-goals
  - Diagnosis of system-wide outages
  - Diagnosis of value faults and transient faults
  - Root-cause analysis at code-level

Assumptions

- Majority of the system is working correctly
- Problems manifest as observable behavioral changes
  - Exceptions or performance degradations
  - Visible to the end-user
- All instrumentation is locally time-stamped
- Clocks are synchronized to enable system-wide correlation of data
- Instrumentation faithfully captures system behavior

Target Systems for Validation

- VoIP system at large ISP
  - 10s of millions of calls per day
  - 1000s of network elements with heterogeneous hardware
  - 24x7 Ops team uses alarm correlation to diagnose outages
  - Separate team troubleshoots long-term chronics
  - Labeled traces available
- Hadoop: Open-source implementation of MapReduce
  - Diverse kinds of data-intensive workloads
    - Graph mining, language translation
  - Hadoop clusters have homogeneous hardware
    - 400-node Yahoo! M45, 64-node OpenCloud clusters
  - Controlled experiments in Amazon EC2 cluster
  - Long running jobs (> 100s): Hard to label failures

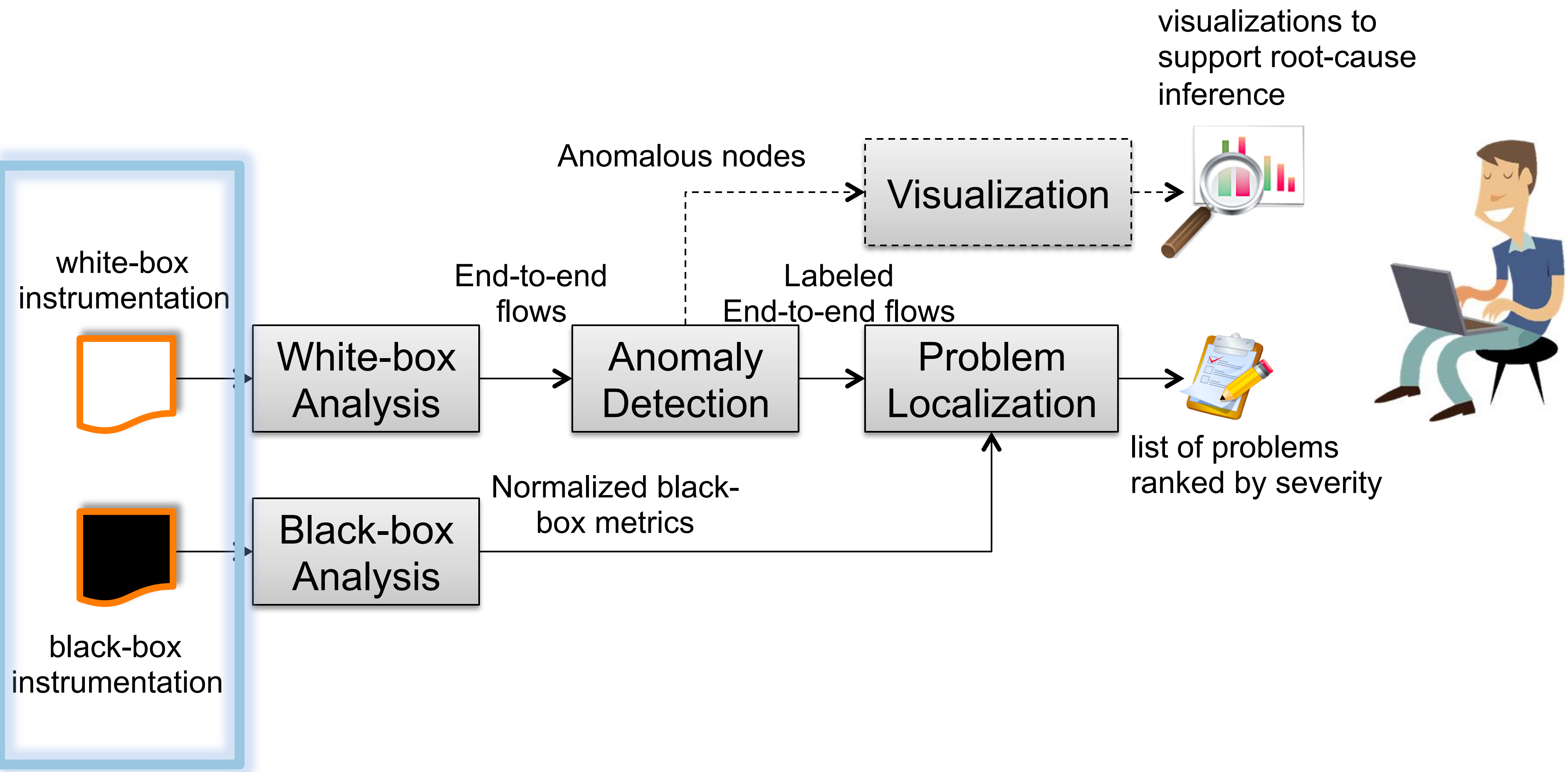
Contributions

	VoIP	HADOOP
Anomaly Detection	Heuristics-based	Peer comparison without labeled data
Problem Localization	Localize to customer/network-element/resource/error-code	Localize to node/task/resource
Types of chronics	Exceptions, performance degradation, single-source, multiple-source	Exceptions, performance degradation, single-source, multiple-source
Experimental Evaluation	Production VoIP system, 1000s of network elements	OpenCloud, 64 nodes
Publications	SLAML'11, OSR'11, DSN'12	WASL'08, HotMetrics'09, ISSRE'09, ICDCS'10, NOMS'10, CCGRID'10

Outline

- Thesis statement
- Approach
  - Instrumentation
  - Anomaly detection
  - Problem localization
- Experimental evaluation
  - Fault injection
  - Case studies
- Extensions
- Conclusion

Overview of Approach



Black-Box Instrumentation

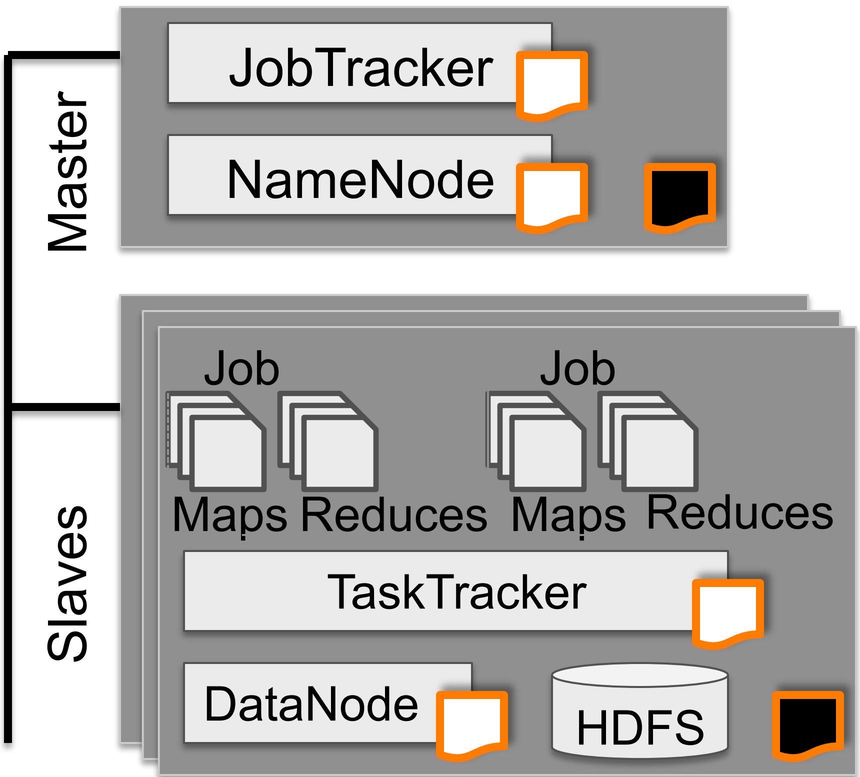
- For both Hadoop and VoIP
- Resource-usage metrics collected periodically from OS
- Monitoring interval varies from 1s to 15min
- Examples of metrics
  - CPU utilization, CPU run-queue size
  - Pages in, pages out
  - Memory used, memory free
  - Context-switches
  - Packets received, packets sent
  - Disk blocks read, disk blocks written

White-Box Instrumentation

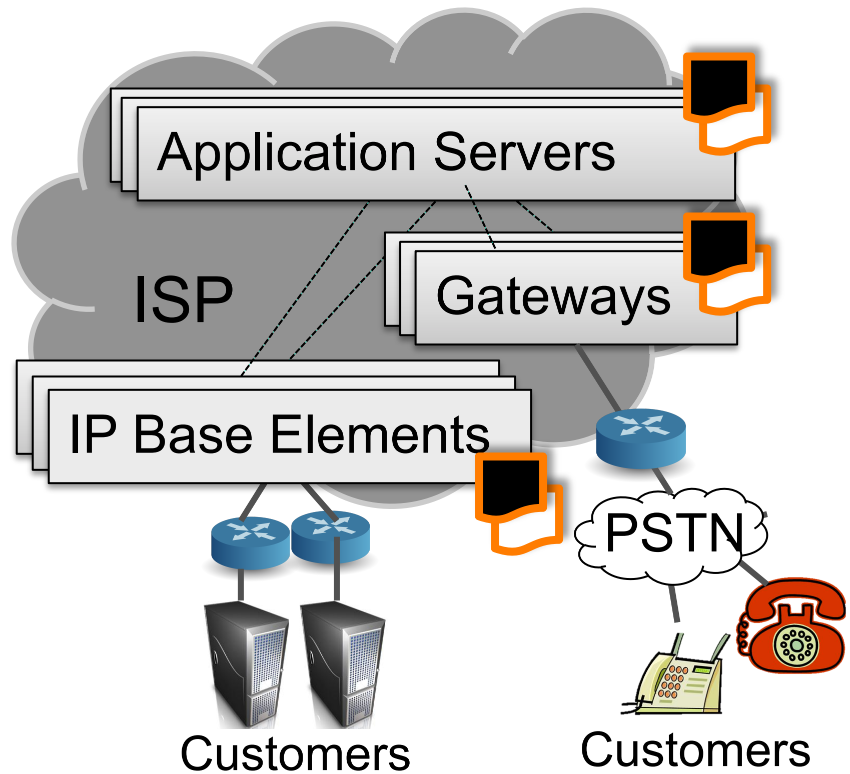
- Each node logs each request that passes through it
  - Timestamp, IP address, request duration/size, phone no., …
- Log formats vary across components
  - Application-specific parsers extract relevant attributes
- Construction of end-to-end traces
  - Extract control flow information
    - Control flow captures sequence of events executed
    - E.g., dependencies between Maps and Reduces in Hadoop
  - Extract data flow information
    - Data flow captures transfer of data between components
  - Stitch end-to-end flows using control and data flow information

Target Systems' Instrumentation

Hadoop Clusters
(OpenCloud, Yahoo! M45)

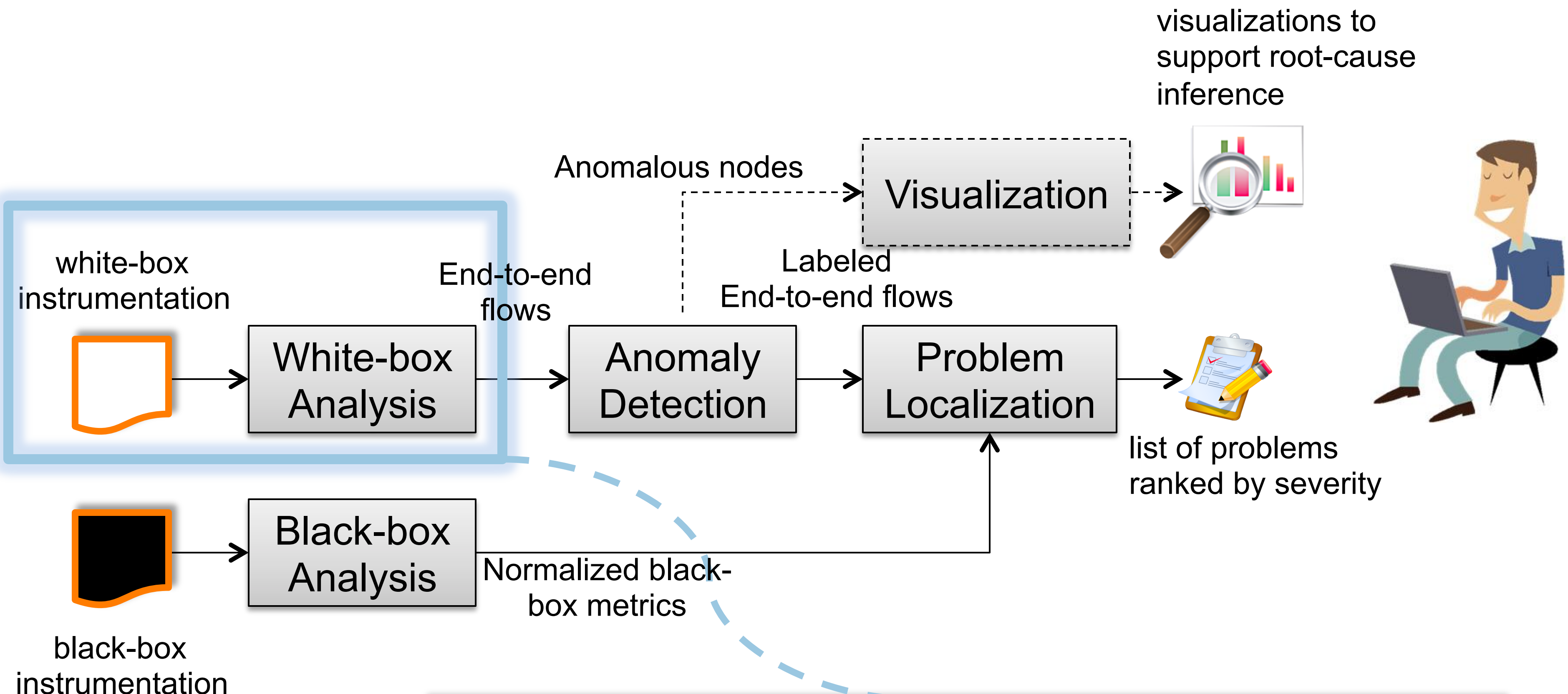


ISP's VoIP System



Carnegie Mellon
Parallel Data Laboratory

White-Box Analysis



Questions

- How do we extract local control- and data-flow?
- How do we infer dependencies with other components?
- How do we deal with missing dependency information?

White-Box Logs

Hadoop Logs

control flow

```
2011-10-12 23:59:55,625 INFO org.apache.hadoop.mapred.TaskTracker:
attempt_201106031747_9630_m_013846_0 0.027189009% Records R/W=208/1
2011-10-12 23:59:55,943 INFO org.apache.hadoop.mapred.TaskTracker: Sent
out 43000 bytes for reduce: 37 from map:
attempt_201106031747_9630_m_013677_0 given 43000
```

control and data flow

VoIP Call Detail Record

```
Mon Apr 29 07:30:14 2013
  NAS-Identifier = "other"
  Acct-Status-Type = "Accounting-off"
  NAS-IP-Address = 172.30.11.36
  Client-IP-Address = 172.30.11.36
  Acct-Input-Packets = 5
  Acct-Output-Packets = 5
  Acct-Input-Octets = 100
  Acct-Output-Octets = 2789
```

control flow

data flow

White-Box Analysis: Hadoop

1 Each node logs task (or block) info. locally

TaskTracker 6

```
10:03:59, MAP
LaunchTaskAction
task_188_m_98
```

2 Domain-specific knowledge used to extract attributes of interest

3 Infer dependency using exact key match on task ID.

TaskTracker 8

```
10:03:59, SHUFFLE
task_656_r_900
copying
task_188_m_98

10:03:59, REDUCE
task_656_r_900
192.168.22.3
```

4 Match on IP address within given time window

Datanode 34

```
10:04:01,
BLOCK_WRITE
blk_8987676
192.168.22.3 to
192.168.22.6
```

White-Box Analysis: VoIP

1 Each node logs call outcomes locally in Call Detail Record

IP Base Element 3

```
IPBE CDR
10:03:59, START
973-123-8888 to
409-555-5555
192.156.1.2 to
11.22.34.1
10:04:02, STOP
```

2 Domain-specific knowledge used to extract attributes of interest

3 Infer dependency using exact key match on phone no.

Application Server 5

```
AS CDR
10:04:05,
ATTEMPT
973-123-8888 to
409-555-5555\
```

4 Approx match on partial phone no. and time

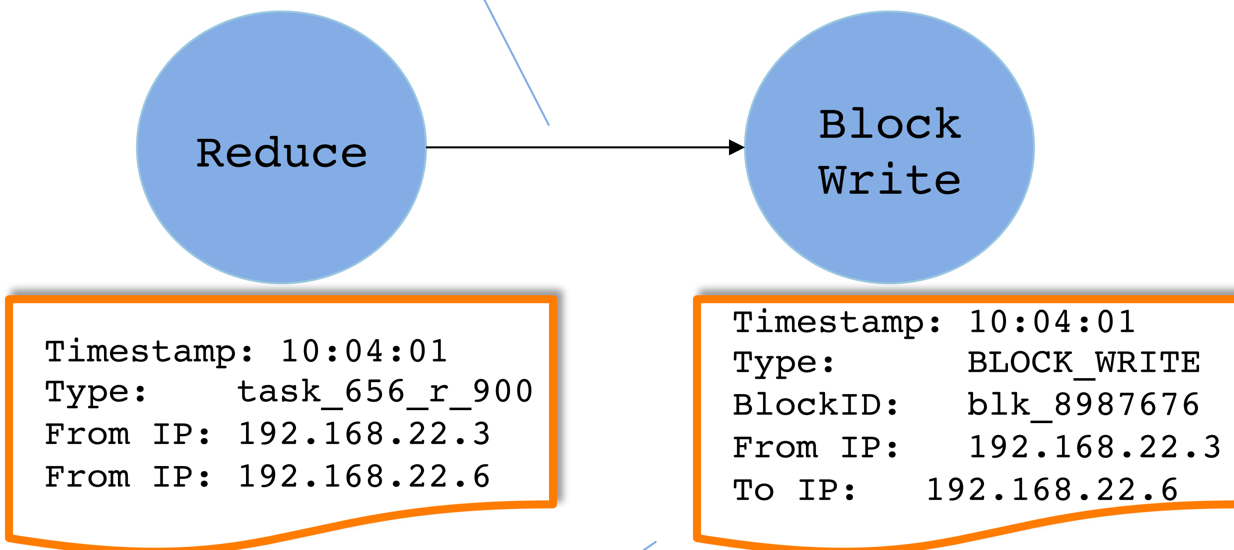
Gateway Server 12

```
GS CDR
10:04:10,
ATTEMPT
973-123-xxxx to
409-555-xxxx
```

Carnegie Mellon
Parallel Data Laboratory

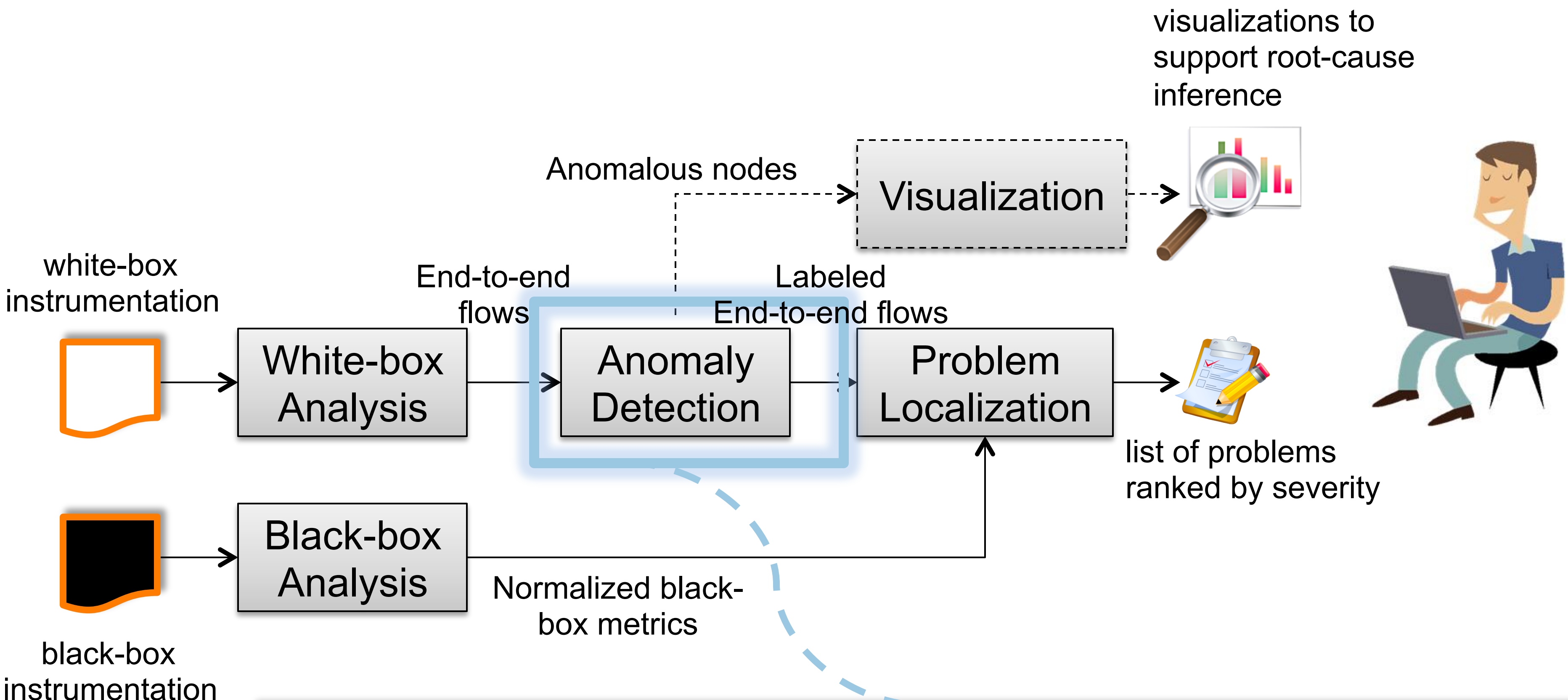
Output of White-box Analysis

Inferred dependencies between components



Unstructured logs transformed into structured log

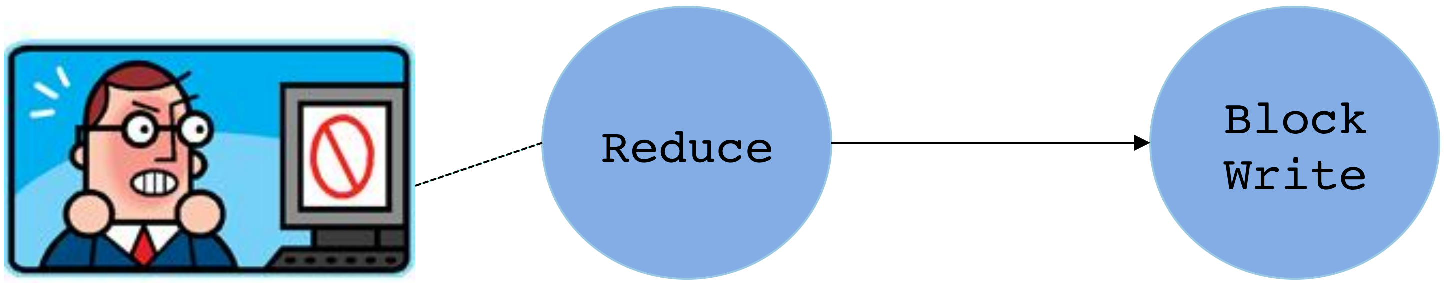
Anomaly Detection



Questions

- How to detect performance problems in the absence of labeled data?
- How to distinguish legitimate application behavior vs. problems?

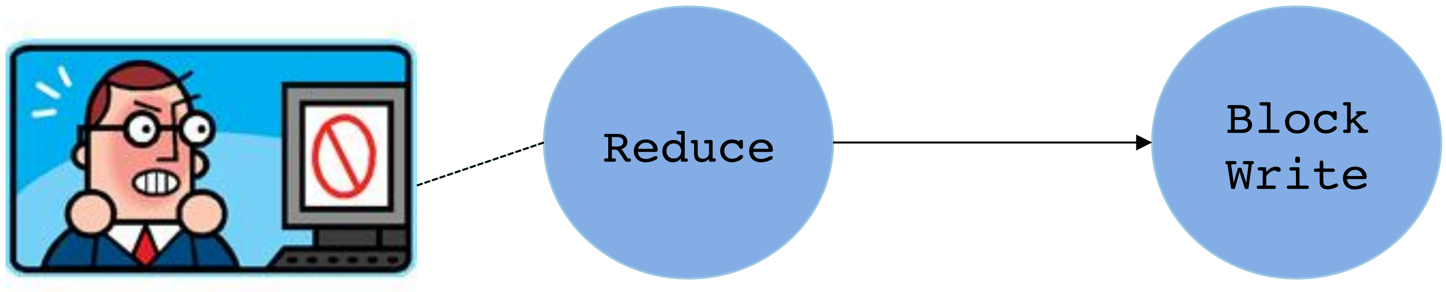
Anomaly Detection



- Some user-visible problems manifest as errors
  - Detected by extracting error codes from failed flows, or
  - Apply domain-specific heuristics
- Performance problems can be harder to detect
  - Exploit the notions of “peers” to detect performance problems
  - Determine what system behaviors can be considered equivalent (“peers”) under normal conditions
  - Significant deviation from “peers” is regarded anomalous

rika (Swahili), *noun*. peer, contemporary, age-set, undergoing rites of passage (marriage) at similar times.

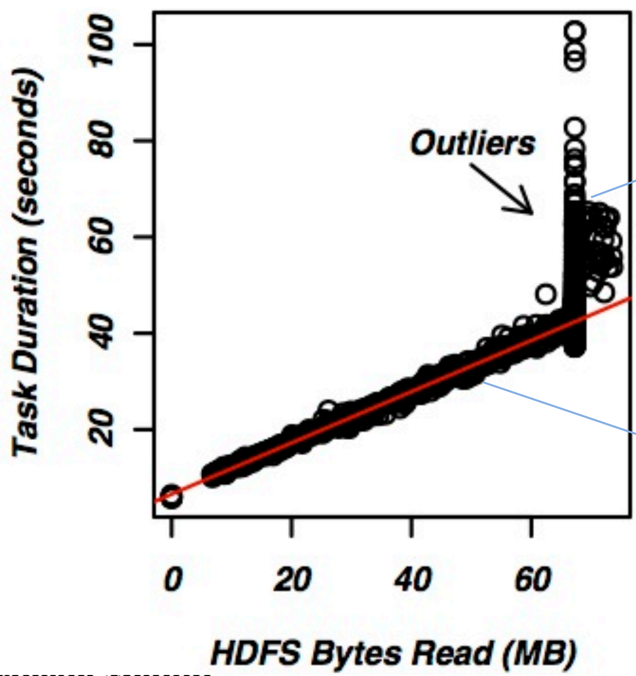
Anomaly Detection: Hadoop (1)



- Extract exceptions from failed and canceled tasks
- Detect performance problems using “peers”
  - Empirical analysis of production data to identify peers
    - 219,961 successful jobs (Yahoo! M45 and OpenCloud)
    - 89% of jobs had low variance in their Map durations
    - 65% of jobs had low variance in their Reduce durations
  - Designate tasks belonging to the same job as peers

Anomaly Detection: Hadoop (2)

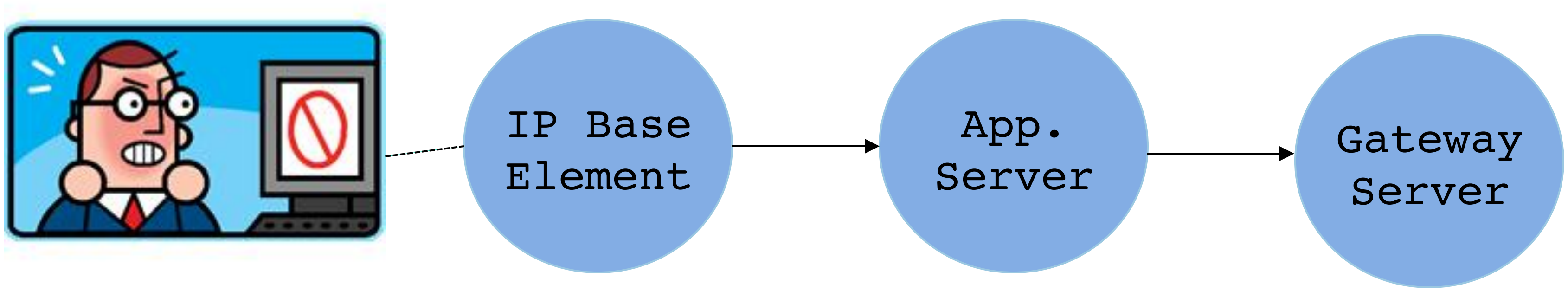
- At the same time, behavior amongst peers can legitimately diverge due to various application factors
  - Identified 12 such factors on OpenCloud
  - Example: HDFS bytes written/read



Flag tasks which do not fit regression model as anomalous

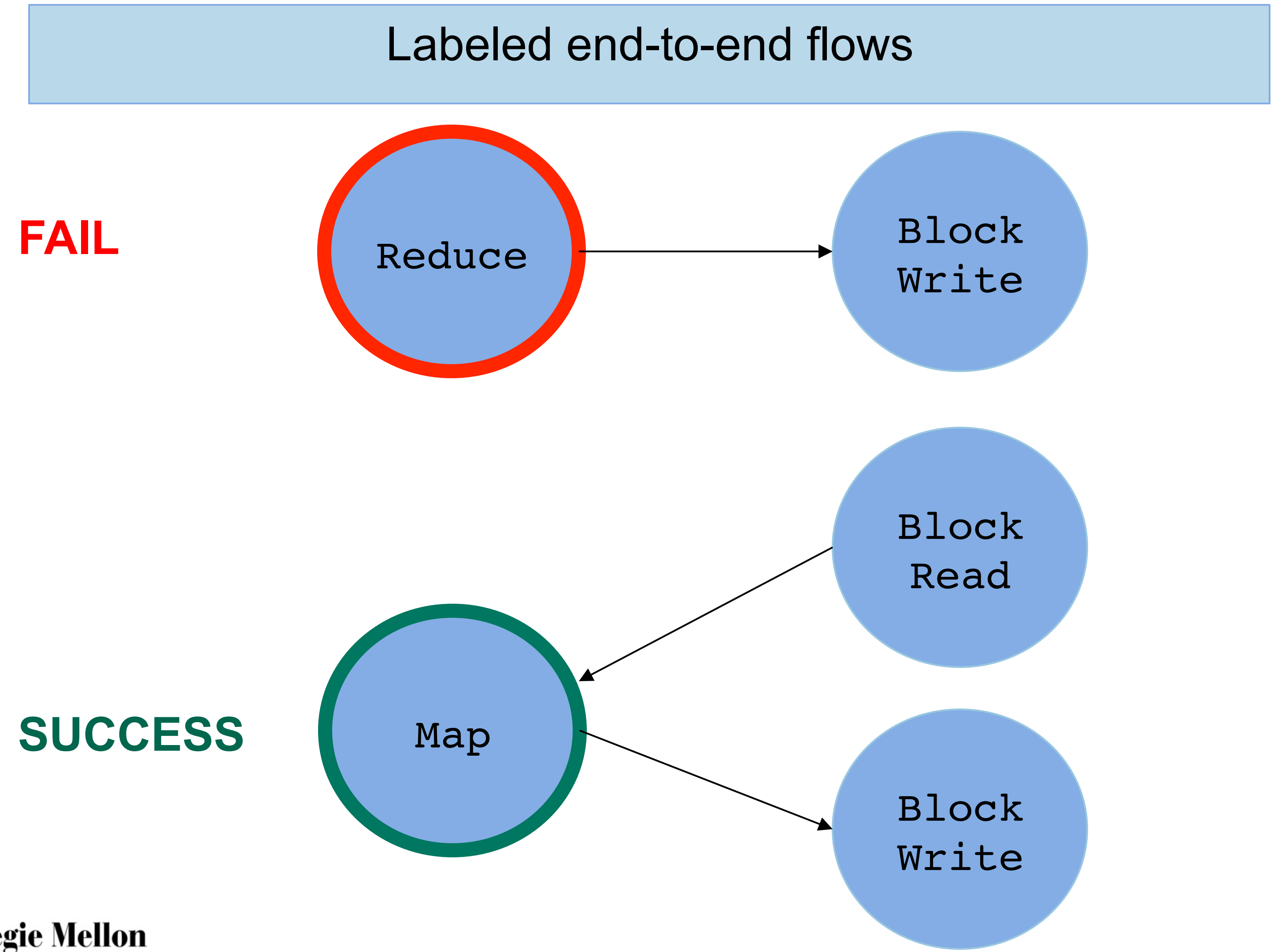
Exploit regression to automatically learn factors influencing task durations

Anomaly Detection: VoIP

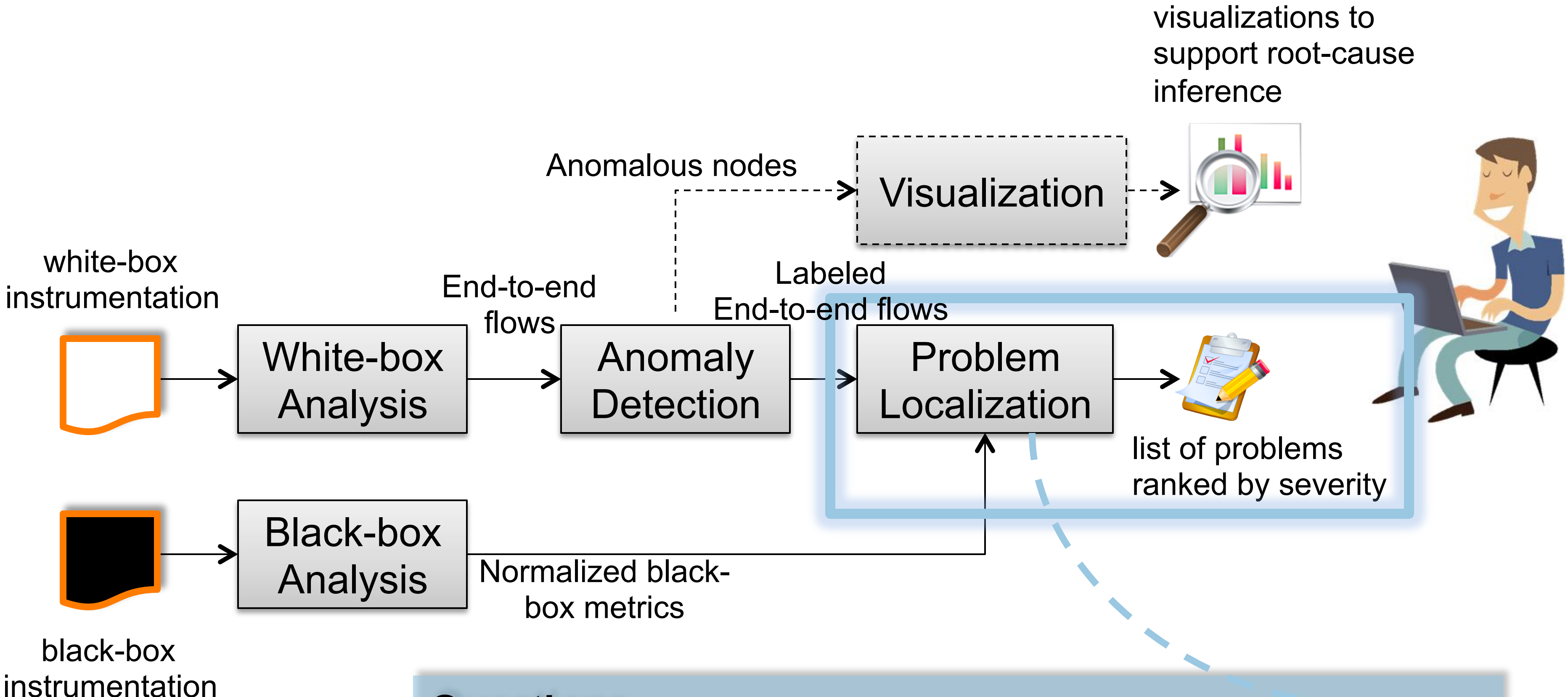


- Detecting blocked or dropped calls
  - Extract defect codes (e.g., timeout) from failed calls
  - Exploit domain-specific heuristics to detect problems
    - Examples: Callback soon after call end, zero talk-time
- Detecting performance problems
  - Designate peers to be calls belonging to the same service
  - Simple statistical technique to detect peers when deviate
    - Packet loss exceeds 90^{th} percentile of calls

Output of Anomaly Detection



Problem Localization



Questions

- How to identify problems due to combination of factors?
- How to distinguish multiple ongoing problems?
- How to find resource that caused the problem?
- How to handle “noise” due to flawed anomaly detection?

Identify Suspect Attributes (1)

STEP 1: Find individual attributes most likely to occur in failed flows

Labeled end-to-end traces generated by anomaly-detection

```
Task1: 09:31am,SUCCESS, Server1, Map2, BlockRead3,
Task2: 09:32am,FAIL, Server1, Map3, Server8, BlockRead5,
       :
       :
```

10s of thousands of attributes

10s of millions of flows

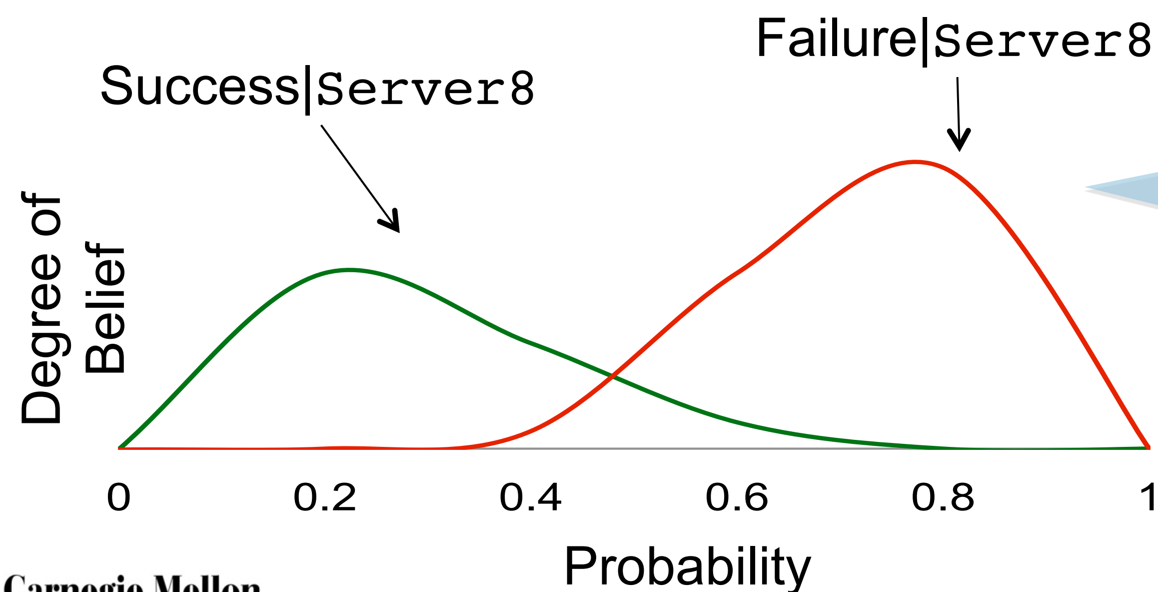
	Server1	Server8	Map2	Map3	Outcome
Task1	1	0	1	0	SUCCESS
Task2	1	1	0	1	FAIL

Binary representation supports scalable representation of attributes as sparse table.

Identify Suspect Attributes (2)

STEP 1: (contd)

- Estimate conditional probability distributions
  - *Prob*(Success|Attribute) vs. *Prob*(Failure|Attribute)
- Update belief on distribution with each flow seen



Anomaly score: Distance between distributions

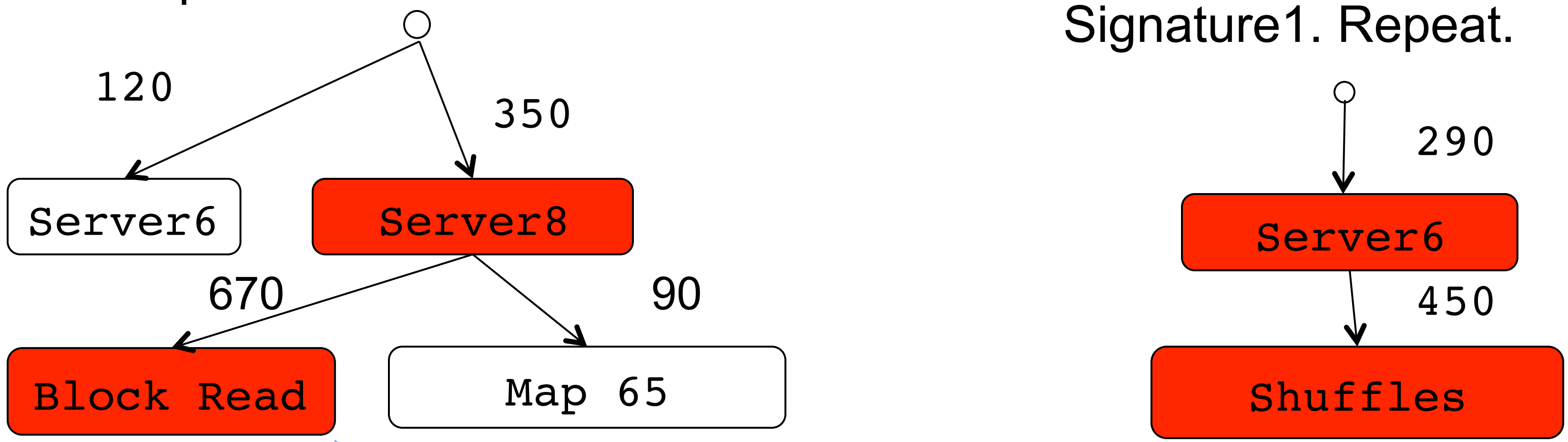
Find Attribute Combinations

STEP 2: Determine if chronic is triggered by combination of factors

- Find attribute combinations that maximize anomaly score
  - Greedy, iterative search limits combinations explored

Step 1: All flows

Step 2: Remove all flows that match Signature1. Repeat.



Indict path with highest anomaly score

Carnegie Mellon
Parallel Data Laboratory

Rank Problems by Severity

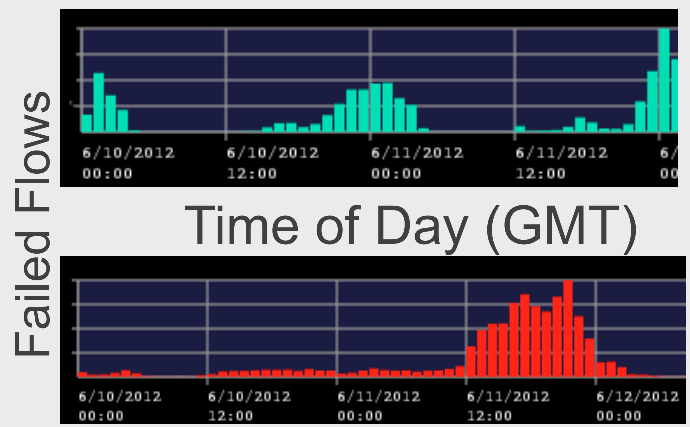
STEP 3: Rank list of identified problems by severity

- Rank problems based on number of flows affected
  - Spurious attributes introduced by noise receive low-rank

UI: Ranked list of chronics identified

1. Chronic signature1
`Server8`
`BlockRead`

2. Chronic signature2
`Server64`



Visualization allows operators to identify recurrent problems

Carnegie Mellon
Parallel Data Laboratory

Fusing Black-box Metrics

STEP 4: Determine if resource-usage metrics affected

Annotate flows associated with culprit nodes (and peers)

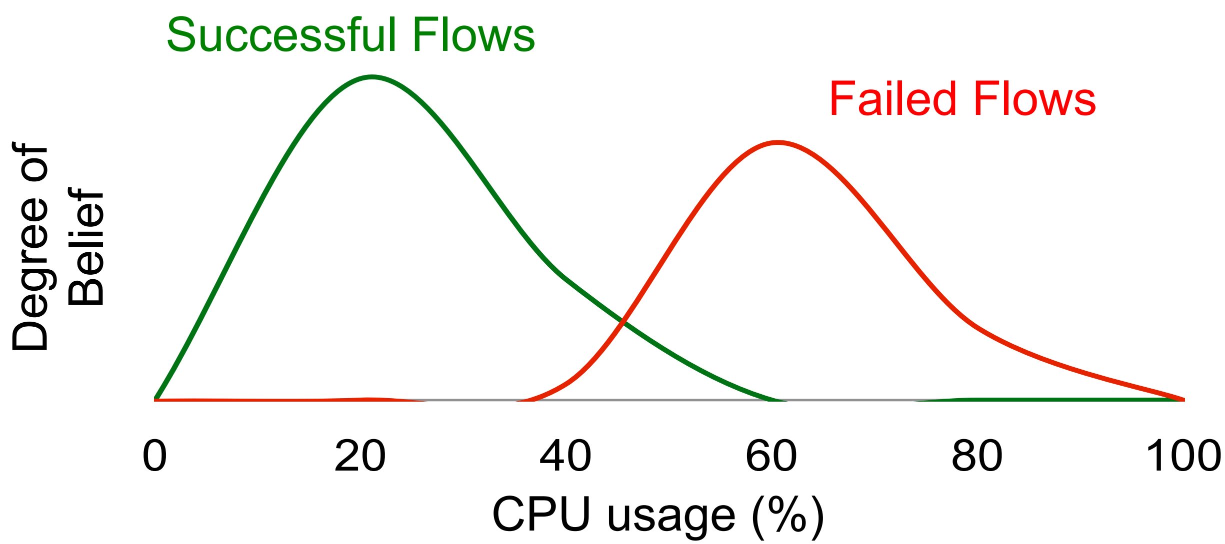
Culprit Node	Peer	Peer
Server 8	**Server 10**	**Server 13**
Time: 10:03:59, Map ID: task_188_m_98 Bytes Read: 7867 Duration: 25 seconds Status: FAILED	Time: 10:03:59, Map ID: task_188_m_76 Bytes Read: 7867 Duration: 3 seconds Status: SUCCESS	Time: 10:03:59, Map ID: task_188_m_85 Bytes Read: 6863 Duration: 2 seconds Status: SUCCESS
Mean CPU: 70.4% Mean Memory: 500MB Mean DiskUtil: 30KB	Mean CPU: 12.4% Mean Memory: 430MB Mean DiskUtil: 32KB	Mean CPU: 15.4% Mean Memory: 480MB Mean DiskUtil: 23KB

Mean resource-usage on node during event duration

Culprit Black-Box Metrics

STEP 4: (contd)

Compare distribution of each black-box metric for successful/failed flows



Indict metric if difference between distributions is statistically significant

Outline

- Thesis statement
- Approach
  - Instrumentation
  - Anomaly detection
  - Problem localization
- Experimental evaluation
  - Fault injection
  - Case studies
- Extensions
- Conclusion

Experimental Evaluation

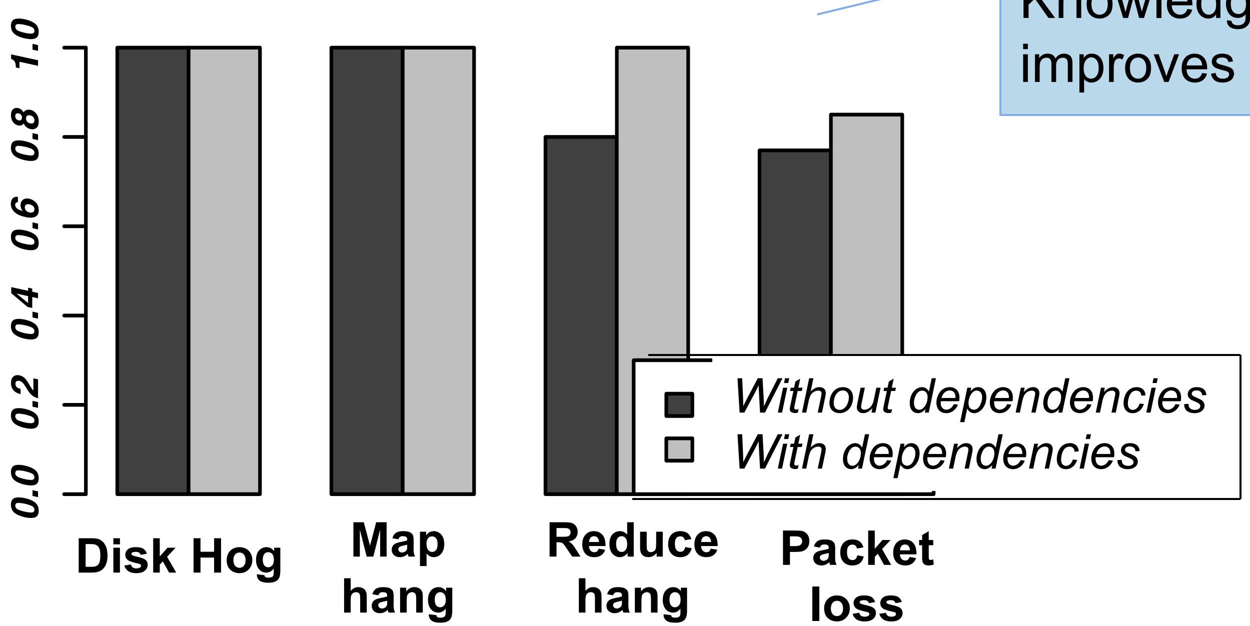
		HADOOP	VOIP
FAULT INJECTION	Workload	Gridmix cluster benchmark	One-week of ISP's call logs
	Injected faults	Resource hogs/Task hangs 10 iterations per fault	Server/Customer problems 1000 simulated faults in total
	Experimental setup	10-node EC2 cluster 2 1.2GHz cores, 7GB RAM	1 simulation node 2 2.4GHz cores, 16GB RAM
CASE STUDIES	Production Sytem	OpenCloud	Production VoIP system at ISP
	Status	Post-mortem offline analysis of real incidents	Used in production for 2 years since 2011

Expt #1: Impact of Dependencies

QUESTION: Does knowledge of dependencies affect diagnosis?

METHOD: Hadoop EC2 cluster, 10 nodes, fault injection.

- Apply problem localization with white-box metrics.
- Compare against approach without knowledge of dependencies.



Knowledge of dependencies improves diagnosis

Expt #2: Impact of Fusion

QUESTION: Does fusion of metrics provide insight on root-cause?

METHOD: Hadoop EC2 cluster, 10 nodes, fault injection.

- Apply problem localization with fused white/black-box metrics.

	Top Metrics Indicted		Insight on root-cause
Fault Injected	**White box**	**Black-box**	
Disk hog	Maps	Disk	✓
Packet-loss	Shuffles	-	✗
Map hang (Hang1036)	Maps	-	✓
Reduce hang (Hang1152)	Reduces	-	✓

Fusion of metrics provides insight on most injected faults

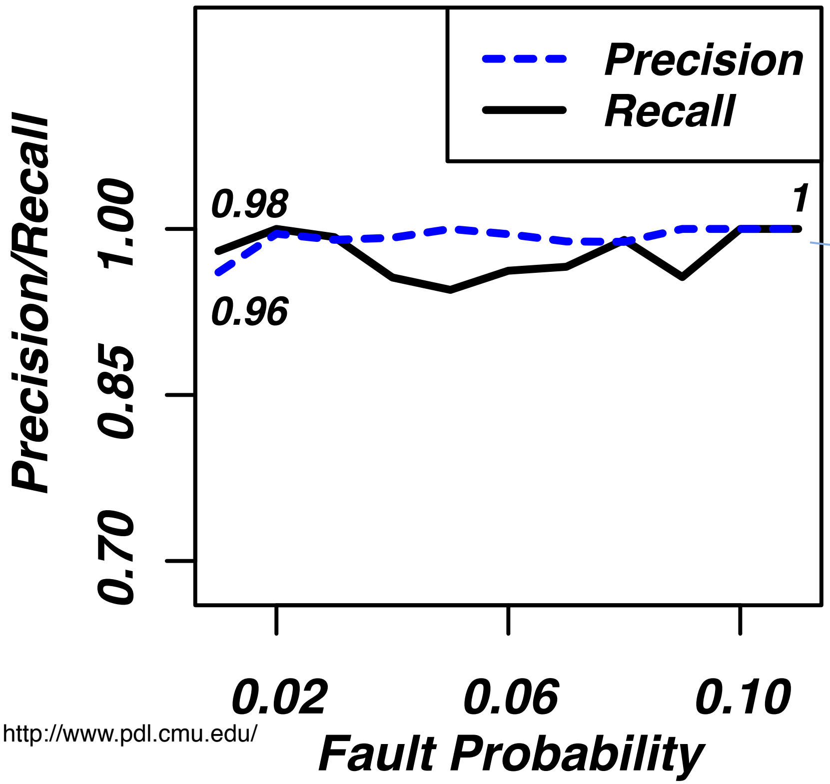
Carnegie Mellon
Parallel Data Laboratory

Expt #3: Impact of Fault Probability

QUESTION: Can we effectively diagnose low-probability faults?

METHOD: One week of ISP's call logs, 1 node, fault simulation.

- Randomly label 1-10% of flows with attributes of interest as faulty.
- Apply problem localization with white-box metrics.



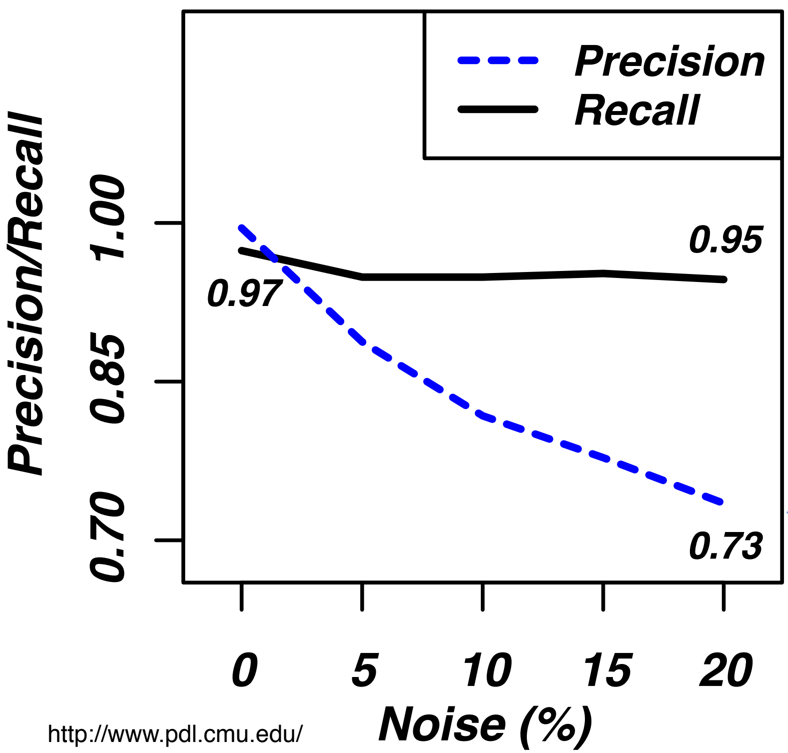
Robust to changes in fault probability

Expt #4: Impact of Noise

QUESTION: Does flawed anomaly-detection (noise) impact diagnosis?

METHOD: One week of ISP's call logs, 1 node, fault simulation.

- Mislabel 5-20% of failed flows.
- Apply problem localization with white-box metrics



Recall robust to noisy labels.

Precision drops due to spurious attributes.

Case #1: Multiple Hardware Issues

INCIDENT: Multiple hardware problems in OpenCloud cluster

- User experiences multiple job failures with cryptic exceptions.
- Administrators initially suspected memory configuration issue.
- Took a week to resolve. Bad disk and bad NIC on two nodes.

DIAGNOSIS APPLIED

- Apply problem-localization approach with white-box metrics.
- Correctly identified nodes with bad hardware in top-10 ranked list

Identified multiple simultaneous problems affecting user's job.

Case #2: Quality (QoS) Violations

INCIDENT: Calls in VoIP system experiencing high packet-loss

- Operators suspect issue with network elements at ISP.
- Took a weeks to resolve.

DIAGNOSIS APPLIED

- Flag calls whose packet-loss exceeds 85th percentile as faulty.
- Apply problem-localization approach with white-box metrics.
- Showed most QoS issues were tied to specific customers.

Operators alerted customers' of problem on their site.
Customer fixed problem and QoS violation resolved.

Case #3: Performance Problem

INCIDENT: Intermittent performance problem in VoIP system

- Intermittent performance problem with two application servers.
- Affected 0.1% of all calls passing through application servers.

DIAGNOSIS APPLIED

- Apply problem-localization using white/black-box metrics.
- Post-mortem analysis confirmed issue with application servers.
- Also flagged anomalous CPU and memory usage.

Successfully identified low-severity fault affecting multiple servers.

Outline

- Thesis statement
- Approach
  - Instrumentation
  - Anomaly detection
  - Problem localization
- Experimental evaluation
  - Fault injection
  - Case studies
- Extensions
- Conclusion

Lessons Learned (1)

- Synthesis of end-to-end causal traces possible
  - Local logs capture local control- and data-flow info
  - Approximate-matching infers implicit dependencies
- In absence of labeled data, peer-comparison is feasible approach for anomaly detection
  - Peers can be tasks (Hadoop), end-to-end flows, calls within the same service (VoIP)
- Regression can help to differentiate between
  - Legitimate application behavior (more bytes read/written) vs.
  - Anomalous behavior (task taking longer to run for other unexplained reasons)

Lessons Learned (2)

- Important to analyze both successful and failed flows
  - Limiting analysis to only failed flows might elevate common elements over causal elements
- Fusion of white+black-box data can provide more insight into source of problem
- Ranking problems by severity helps tolerate noise
  - Spurious labels receive lower ranking

Limitations

- No diagnosis for the Master node of a Hadoop cluster
  - Problems at master typically result in system-wide issues
- Peer-groups are defined statically
  - Need to automate identification of peers
- False positives occur if root-cause not in logs
  - Algorithm tends to implicate adjacent network elements
  - Need to incorporate more data to improve visibility
- Does not detect dormant problems that do not impact user-perceived system behavior
  - Examples: Blacklisted nodes in Hadoop

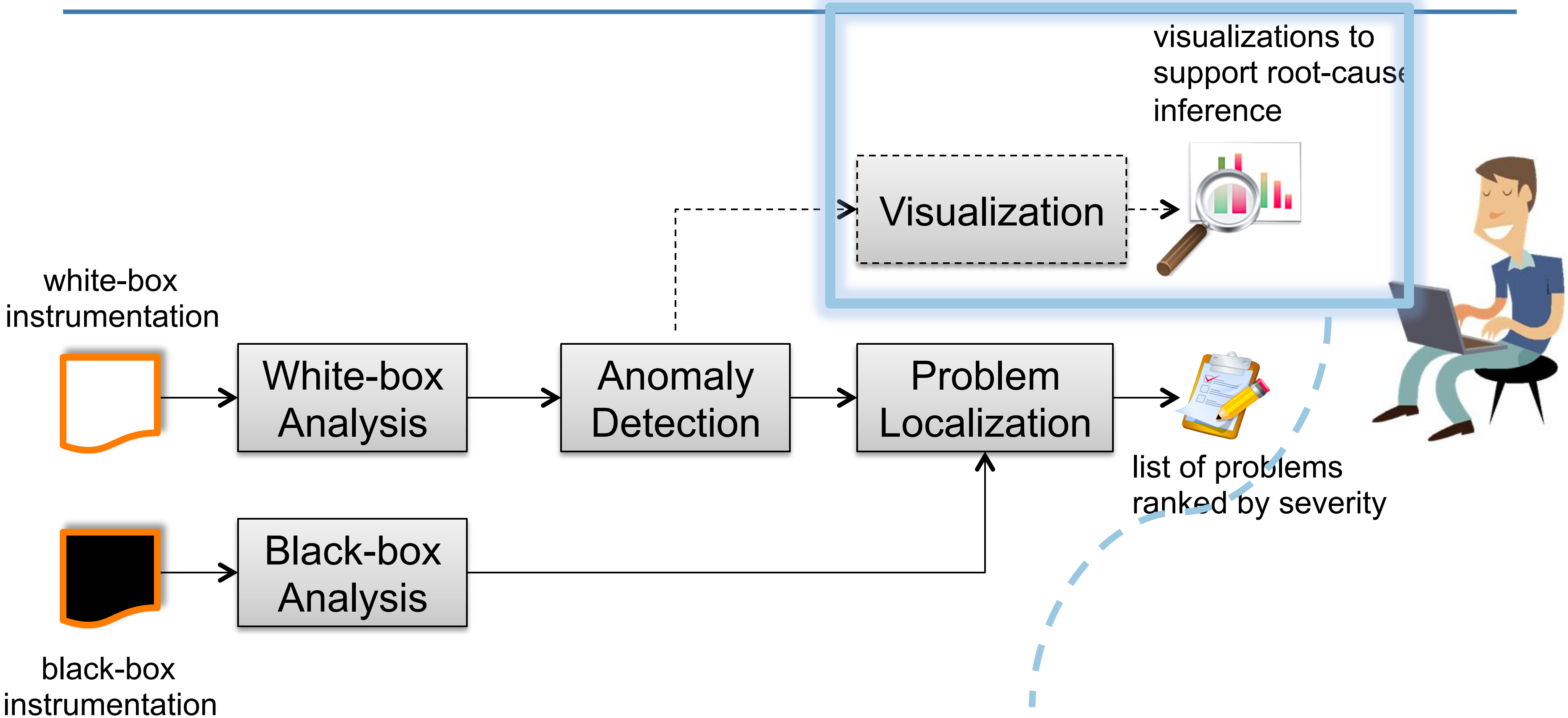
Extensions (Future Work)

- Visualization in heterogeneous systems
  - ✓ User study on diagnosis interfaces in Hadoop [CHIMIT11]
  - ✓ Visual signatures of problems in Hadoop [LISA12]
  - ✗ Visual signatures of problems in heterogeneous systems
  - ✗ Extensible visualization framework for diagnosis

Future Work

- Online monitoring and diagnosis
  - ✓ Generic framework for monitoring and diagnosis [WADS09]
  - ✓ Streaming implementation of problem-localization [DSN12]
  - ✗ Scalable monitoring and diagnostic framework

Visualization

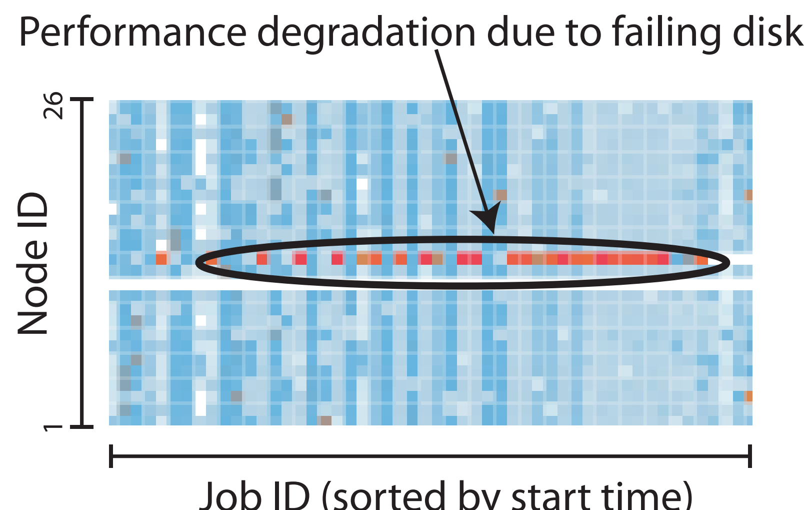


Questions

- How to develop compact visualizations for large clusters?
- Can visualizations help spot/discriminate different anomalies?

Theia: Visual Signatures of Problems

- Maps anomalies observed to broad problem classes
  - Hardware failures, application issue, data skew
- Supports interactive data exploration
  - Users drill-down from cluster- to job-level displays
  - Hovering over the visualization gives more context
- Compact representation for scalability
  - Can support clusters with 100s of nodes



*USENIX LISA 2012 Best Student-Paper Award

Conclusion

- Approach for diagnosis of chronic problems
  - Amenable for use in production systems
  - Infers dependencies from existing white-box logs
  - Uses heuristics and peer-comparison to detect anomalies
  - Localizes source of problem using statistical approach
  - Incorporates both white-box and black-box logs
- Demonstrated for two production systems
  - VoIP system at ISP (approach deployed for 2 years now)
  - OpenCloud Hadoop cluster
- Initial progress on extensions (visualization)

Collaborators & Thanks

- VoIP (AT&T)
  - **Matti Hiltunen**, Kaustubh Joshi, Scott Daniels
- Hadoop visualization
  - **Christos Faloutsos**, U Kang, Elmer Garduno, Jason Campbell (Intel), HCI 05-610 team
- OpenCloud
  - **Greg Ganger**, Garth Gibson, Julio Lopez, Kai Ren, Mitch Franzos, Michael Stroucken
- Hadoop diagnosis
  - Jiaqi Tan, Xinghao Pan, Rajeev Gandhi, Keith Bare, Michael Kasick, Eugene Marinelli

Publications (1)

Diagnosis in VoIP

1. S. P. Kavulya, S. Daniels, K. Joshi, M. Hiltunen, R. Gandhi, P. Narasimhan. Draco: Statistical Diagnosis of Chronic Problems in Large Distributed Systems. IEEE Dependable Systems and networks (DSN'12), Boston, MA, Jun 2012.
2. S. P. Kavulya, K. Joshi, M. Hiltunen, S. Daniels, R. Gandhi, P. Narasimhan. Practical Experiences with Chronics Discovery in Large Telecommunications Systems. Best Papers from SLAML in Operating Systems Review (OSR'12), 2012.
3. S. P. Kavulya, K. Joshi, M. Hiltunen, S. Daniels, R. Gandhi, P. Narasimhan. Practical Experiences with Chronics Discovery in Large Telecommunications Systems. Workshop on Managing Large-Scale Systems via the Analysis of System Logs and the Application of Machine Learning Techniques (SLAML'11), 2011.

Visualization, User studies, Surveys

4. E. Garduno, S. Kavulya, J. Tan, R. Gandhi and P. Narasimhan. Theia: Visual Signatures for Problem Diagnosis in Large Hadoop Clusters. In Large Installation System Administration Conference (LISA) 2012, San Diego, CA, Dec 2012. *Best Student Paper Award.*
5. S. P. Kavulya, K. Joshi, F. Di Giandomenico, P. Narasimhan. Failure Diagnosis of Complex Systems.Book on Resilience Assessment and Evaluation (RAE'12). Wolter, 2012.
6. J. Campbell, A. Ganesan, B. Gotow, S. Kavulya, J. Mulholland, P. Narasimhan, S. Ramasubramanian, M. Shuster, and J. Tan. Understanding and Improving the Diagnostic Workflow of MapReduce Users. In 5th ACM Symposium on Computer Human Interaction for Management of Information Technology (CHIMIT), Boston, MA, Dec 2011.
7. S. Kavulya, J. Tan, R. Gandhi, P. Narasimhan. An Analysis of Traces from a Production MapReduce Cluster. 10th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid) 2010, Melbourne, Victoria, Australia, May 2010.

White-box diagnosis

8. J. Tan, S. Kavulya, R. Gandhi, P. Narasimhan. Visual, Log-based Causal Tracing for Performance Debugging of MapReduce Systems. 30th IEEE International Conference on Distributed Computing Systems (ICDCS) 2010, Genoa, Italy, Jun 2010.
9. J. Tan, X. Pan, S. Kavulya, R. Gandhi, P. Narasimhan. Mochi: Visual Log-Analysis Based Tools for Debugging Hadoop. USENIX Workshop on Hot Topics in Cloud Computing (HotCloud '09), San Diego, CA, Jun 2009.
10. J. Tan, X. Pan, S. Kavulya, R. Gandhi, P. Narasimhan. SALSA: Analyzing Logs as State Machines. USENIX Workshop on Analysis of System Logs (WASL'08), San Diego, CA, Dec 2008.

Black-box diagnosis

11. J. Tan, S. Kavulya, R. Gandhi, P. Narasimhan. Lightweight Black-box Failure Detection for Distributed Systems. In Workshop on Management of Big Data systems (MBDS) 2012, co-located with the International Conference on Autonomic Computing, San Jose, SA, Sep 2012.
12. X. Pan, S. Kavulya, J. Tan, R. Gandhi, P. Narasimhan. Ganesha: Black-Box Diagnosis for MapReduce Systems. Workshop on Hot Topics in Measurement & Modeling of Computer Systems (HotMetrics), Seattle, WA, Jun 2009.

Publications (2)

Black-box + White box diagnosis

13. J. Tan, X. Pan, S. Kavulya, E. Marinelli, R. Gandhi, P. Narasimhan. Kahuna: Problem Diagnosis for MapReduce-based Cloud Computing Environments. 12th IEEE/IFIP Network Operations and Management Symposium (NOMS) 2010, Osaka, Japan, Apr 2010.
14. X. Pan, J. Tan, S. Kavulya, R. Gandhi, P. Narasimhan. Blind Men and the Elephant: Piecing Together Hadoop for Diagnosis. 20th IEEE International Symposium on Software Reliability Engineering (ISSRE), Industrial Track, Mysuru, India, Nov 2009.
15. S. Kavulya, R. Gandhi, P. Narasimhan. Gumshoe: Diagnosing Performance Problems in Replicated File-Systems. IEEE Symposium on Reliable Distributed systems (SRDS'08), Naples, Italy, October 2008.
16. S. Pertet, R. Gandhi, P. Narasimhan. Fingerpointing Correlated Failures in Replicated Systems. SysML, April 2007.

Questions?


CONGRATULATIONS
YOU ARE NOW AT
UHURU PEAK TANZANIA 5895M AMSL.
AFRICA'S HIGHEST POINT
WORLD'S HIGHEST FREE-STANDING
ONE OF WORLD'S LARGEST VOLCAN
WELCOME

Related Work (1)

- M. Attariyan, M. Chow, and J. Flinn, X-ray: Automating root-cause diagnosis of perfor- mance anomalies in production software. *USENIX Symposium on Operating Systems Design and Implementation (OSDI'12),* Hollywood, CA, October 2012.
- P. Bodík, M. Goldszmidt, A. Fox, D. B. Woodard, H. Andersen. Fingerprinting the datacenter: automated classification of performance crises. EuroSys 2010.
- **[Cohen05]:** Capturing, indexing, clustering and retrieving system history. Ira Cohen, Steve Zhang, Moises Goldszmidt, Julie Symons, Terence Kelly, Armando Fox. SOSP, 2005.
- **[Kandula09]:** Detailed diagnosis in enterprise networks. Srikanth Kandula, Ratul Mahajan, Patrick Verkaik, Sharad Agarwal, Jitendra Padhye, Paramvir Bahl. SIGCOMM 2009.
- **[Kasick10]:** Black-Box Problem Diagnosis in Parallel File Systems. Michael P. Kasick, Jiaqi Tan, Rajeev Gandhi, Priya Narasimhan. FAST 2010.

Related Work (2)

- **[Kiciman05]:** Detecting application-level failures in component-based Internet Services. Emre Kiciman, Armando Fox. IEEE Trans. on Neural Networks 2005.
- **[Mahimkar09]:** Towards automated performance diagnosis in a large IPTV network. Ajay Anil Mahimkar, Zihui Ge, Aman Shaikh, Jia Wang, Jennifer Yates, Yin Zhang, Qi Zhao. SIGCOMM 2009.
- **[Sambasivan11]:** Diagnosing Performance Changes by Comparing Request Flows. Raja R. Sambasivan, Alice X. Zheng, Michael De Rosa, Elie Krevat, Spencer Whitman, Michael Stroucken, William Wang, Lianghong Xu, and Gregory R. Ganger. NSDI 2011.
- **[Tati12]** S. Tati, B.-J. Ko, G. Cao, A. Swami, and T. F. L. Porta, Adaptive algorithms for diagnosing large-scale failures in computer networks. *IEEE Conference on Dependable Systems and Networks (DSN),* Boston, MA, Jun. 2012