Application Aware SoftWare Anomaly Treatment

Pradeep Ramachandran, Siva Kumar Sastry Hari, Manlap Li, Sarita Adve, Shobha Vasudevan

Resilient Theme Task # 1.2.2.5

GSRC Annual Symposium

Sep 3-4, 2009

Motivation

Technology scaling ⇒ Increased in-the-field failures for commodity systems

Wear-out, infant mortality, design defects, etc.

Need low-cost in-field techniques for detection, diagnosis, recovery, repair

SWAT - SoftWare Anomaly Treatment

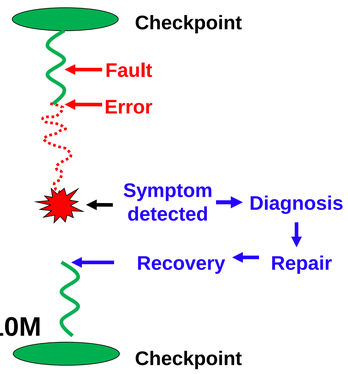
Strategy

Watch for anomalous software behavior ⇒ Symptom

Zero/low cost "always-on" monitors

Diagnose fault after detection

Rarely invoked => may incur higher overheads



Previous results for SPEC

95% of faults detected in 10M instructions

⇒ Recovery needs checkpoint/output buffer window of 10M

0.8% of faults result in SDCs

This work: Application-aware methods to improve SDCs, recovery window

Using Application-Awareness for SDCs, Recovery Window

- **Low-Cost Address Out-of-Bounds detector**
- **Application-aware SDC and recovery window analysis**
- **Baseline SWAT on new I/O intensive client/server apps for I/O analysis**

Results: Orders of magnitude improvement in SDC rate, recovery window, output buffer size

Application-Aware Address Out-of-Bounds Detector

Amortize resiliency cost for HW/SW faults

SW bug detection uses such detectors

Low-cost detector that monitors bounds

HW faults ⌫invalid/unallocated addr

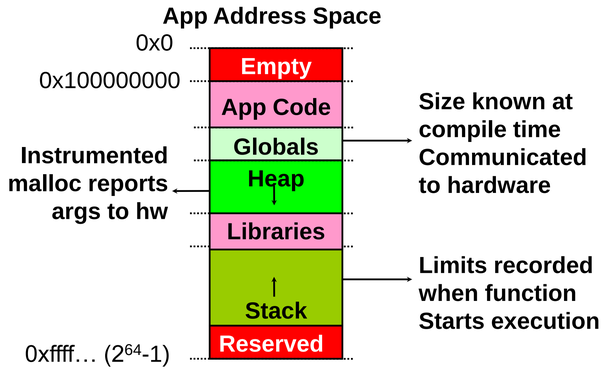
HW/SW coordination to identify legal bounds

Results

50% faults detected by new detector

Dramatic reduction in recovery window

Reduces system state corruption by half



Application-Aware SDC Analysis

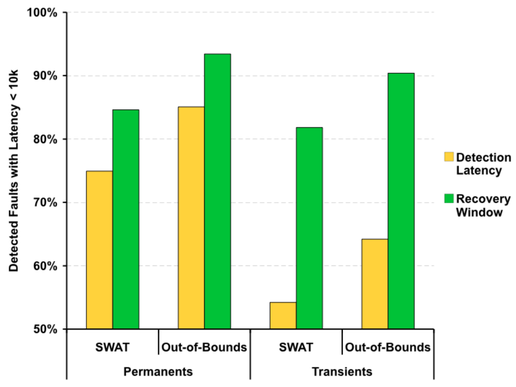
Fault corrupts output produced by application traditionally ⇒ SDC

But some applications, even SPEC, tolerate errors in outputs!

Fault activation influences detection ⇒ Round-robin scheduling ↓ SDCs

Application-Aware SDC Rate of 8 SPEC Apps				
Output error tolerance	**SWAT**	**App-Aware**	**SWAT**	**App-Aware**
	Permanent Faults		*Transient Faults*	
< 0.1%	54	2	14	7
< 1%	54	1	14	0

Application-Aware Recovery of Detected Faults



Fault unrecoverable only *after* corrupting SW state

SW recoverable with corrupted arch state

***Detection latency* - Arch state corruption → detection**

***Recovery window* - SW state corruption → detection**

Results:

>80% in SWAT recoverable in <10k instructions

>90% in Out-of-Bounds recoverable in <10k instr

Detector reduced latency, recovery window

Implications of Recovery Window for I/O and Recoverability

Larger recovery window ⇒ Overhead for buffering I/O, user perception

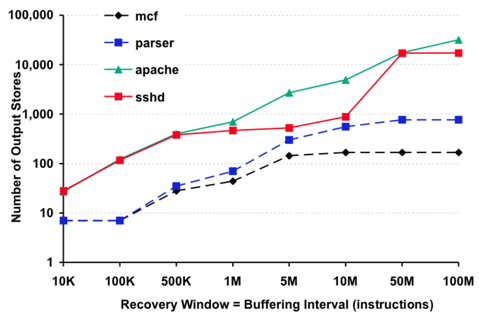
Results:

10M instruction window needs 80KB buffer

New 10K instruction window needs only 30 stores!

Can be buffered using Store buffer

New techniques have dramatic implications for recovery



Conclusions and Future Work

Application Awareness ⇒ Lower SDC rate, shorter recovery window w/ less I/O buffering

Future Work: App-aware SDC analysis of distributed client/server applications

Low overhead recovery techniques for short latency