Overview of Research in the Dependable Computing Systems Lab

Saurabh Bagchi
Assistant Professor
Dept. of ECE
Purdue University

August, 2003

Why Dependable Computing?

- Simple:
  - We need systems that we can trust our life on: Medical diagnostics, fly-by-wire aircrafts
  - We need systems that we can trust our money on: Banking sector, financial investment sector, electronic commerce
- Why is it more of an issue today than ever before?
  - Ubiquitous computing: Your automobile has more computing power than the fastest supercomputer of 1970's
  - Tera-scale integration: Moore's law has meant more chips on a wafer, more transistors on a chip
  - Add security in the mix

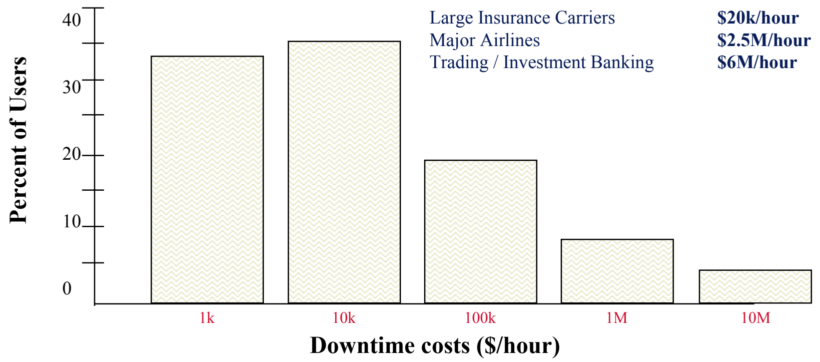
Who are the culprits?

- Hardware failure
  - Can cause degradation of performance or unavailability of data or devices
- Software system failure
  - Causes system crash
  - May or may not be reproducible
- Operational downtime (or, operator error)
- Application software failure
- Maintenance: Backups, Software or hardware upgrades
- Environmental problems: power supply, communication lines, etc.

Some Not So Pleasant Memories

- June 4, 1996: Maiden flight of space shuttle Ariane 5 crashed in France
  - Reason: Attempt to stuff the horizontal velocity in a 16 bit variable causing overflow
- February 19, 2001: AT&T's ATM network outage for 4 hours
  - Reason: Lucent WAN switch sent out a firestorm of network management messages
- October 21, 2002: Distributed Denial of Service (DDoS) attack against root DNS servers
  - Reason: "Ping" attack launched from multiple machines that were compromised

What Does It Cost?



- Survey of 450 Fortune 1000 companies
- Per hour of network outage costs an average of $82,500, higher end $6M

Research in Dependable Computing Systems Lab

- Framework for distributed disruption tolerant system
- Self-checking network protocols
- Dependable ad-hoc and sensor networks
- Hardware architecture support for enhancing software reliability

Project #1: Distributed Disruption Tolerant System

- Distributed e-commerce platform subjected to natural failures and malicious attacks to services
- Disruptions = Attacks + Failures
- Objective is to tolerate disruptions, not just detect
- Different phases:
  - Detection
  - Diagnosis
  - Containment
  - Response
- Project Members:
  - Here: Arif Ghafoor, Eugene Spafford, Yu-Sung Wu, Yongguo Mei, Bingrui Foo, Blake Matheny
  - Outside: Tim Tsai, Sachin Garg (Avaya Labs)

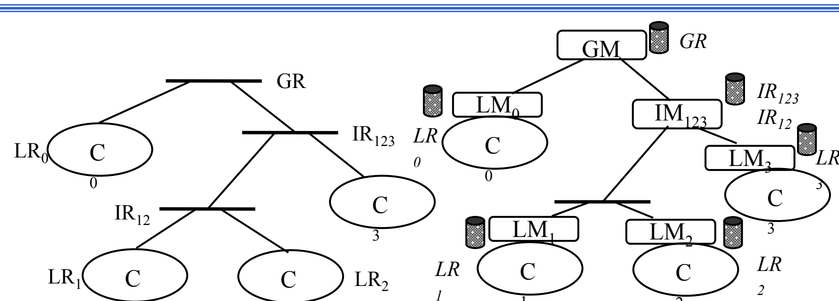
Project #1: Distributed Disruption Tolerant System

- Story So Far:
  - Collaborative Intrusion Detection System built: Combined alerts from multiple detectors for efficient and accurate detection
  - Paper accepted for publication in Advanced Computer Security Applications Conference (ACSAC), December 2003.
  - Design of data structure and algorithm for containment and determination of whether to take response
- What's Next:
  - Containment and Response system will be implemented
  - Paper to be submitted to IEEE Symposium on Security and Privacy (Deadline: November 5)

Project #2: Self-Checking Network Protocols

- Goal is to provide highly available network services (e.g., SIP, reliable multicast) in distributed environment
- Challenges in today's distributed systems
  - Large number of network protocol participants
  - No access to source code or machine on which code is running
  - Often soft real-time guarantees
- Our Approach:
  - Distributed monitor to observe external interactions and diagnose misbehavior or malfunction
  - A rulebase using temporal logic and fast matching algorithms

Project #2: Self-Checking Network Protocols



Legend

C: Cluster; LR: Local Rule; IR: Intermediate Rule; GR: Global Rule; LM: Local Monitor; IM: Intermediate Monitor; GM: Global Monitor; : Rule repository

- Project Members
  - Here: Gunjan Khanna, Padma Varadharajan
  - Outside: Ravi Iyer, Zbigniew Kalbarczyk (UIUC)

Project #2: Self-Checking Network Protocols

- Story So Far:
  - Reliable Multicast Protocol called TRAM made more robust: TRAM++
  - Paper submitted to Synposium on Reliable Distributed Systems (SRDS). To be resubmitted to PRDC (Deadline: September 5)
  - Formal specification language for rules identified
- What's Next:
  - Rules for TRAM++ and SIP applications
  - Implementation of hierarchical monitor for these two applications
  - Paper to be submitted to IEEE Intl. Conference on Dependable Systems and Networks (Deadline: November 5)

Project #3: Dependable Ad-hoc and Sensor Networks

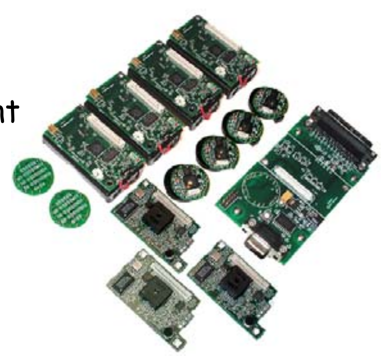
- Ad-hoc and sensor networks built of unreliable components and deployed in hostile or uncertain environments
- Goal is to provide middleware that provides a robust platform keeping environment constraints in mind
  - Energy constraint
  - Computational power constraint
  - Security constraint
- Project Members:
  - Here: Mikhail Atallah, Ness Shroff, Nipoon Malhotra, Serdar Cabuk, Longbi Lin, Issa Khalil

Project #3: Dependable Ad-hoc and Sensor Networks

- Mobility to help network characteristics
  - Intelligent mobility patterns to improve connectivity, coverage, diameter
- Robust data aggregation from sensor nodes to base station
  - Robust to failures of intermediate nodes and compromised nodes
  - Sensitive to energy budget of each node
- Secure message communication in sensor networks
  - Efficient protocol for encryption of messages
  - Scalable and energy parsimonious key distribution protocol

Project #3: Dependable Ad-hoc and Sensor Networks

- Testbed set up with small sensor nodes called Berkeley motes
- Story So Far:
  - Intel donated equipment
  - NSF funded 3 year project on Sensors and Sensor Networks
  - 2 papers published, 3 submitted



- What's Next:
  - Middleware development on the testbed
  - Paper to be submitted to DSN, 2004 (Deadline: Nov 14, 2003)

Project #4: Architecture Approach to Software Robustness

- Goal: Use idle hardware resources, such as additional execution contexts in SMT or CMP, for checking software
- Memory checks are biggest bang for buck
  - Large class of software errors
  - Easy to automate
- Approach
  - Devise detection routines
  - Keep synchronization between detection and application routine to a minimum
  - Devise hardware extensions that enable fast information transfer from one to the other

Project #4: Architecture Approach to Software Robustness

- Story so far:
  - SMT based simulator created for simple monitoring routines
  - Performance results show substantial improvement over baselines – all monitoring in software running in same execution context
- Project Members: Prof. T. N. Vijaykumar, Yen-Shiang Shue, Jin-Yi Wang, Yu-Sung Wu

Interested in any of this research?

- Bagchi, EE 329, sbagchi@purdue.edu
  WF 10:30-11:30
- Or, by appointment