SRIRAM SUBRAMANIAN

5002 Sheboygan Ave, #332
Madison WI 53705
Ph : 650 391 5222
srirams@cs.wisc.edu

7361 CS&S, 1210 W Dayton St
Madison WI 53706
Office No : 608-262-6602
http://pages.cs.wisc.edu/~srirams

EDUCATION

University of Wisconsin, Madison Aug 2007 - Present
PhD in Computer Sciences
M.S. in Computer Science; GPA: 3.8

Birla Institute of Technology & Science, Pilani, India Aug 2001 - June 2005
B.E.(Hons) in Computer Science; GPA: 3.97

RESEARCH

Memory Corruption Impact Analysis for File Systems using Symbolic Execution June - Dec 2010
(submitted to DSN 2011)

In this paper we present a novel technique to understand the impact of memory corruptions in file systems. We employ selective symbolic execution to exhaustively explore the impact of memory corruptions on other in-memory data structures, disk and the user. Our technique emphasizes the importance of the location of corruption in addition to the corrupted data structure. We present a detailed case study of applying our technique to Ext2. We identify the data structures and code regions most sensitive to corruption and present corruption spectrums for each scenario. Thus, our technique offers developers the opportunity to improve file system reliability without sacrificing performance.

Making the Common Case the Only Case with Anticipatory Memory Allocation Jan - May 2010
(FAST 2011)

In this paper, we present Anticipatory Memory Allocation (AMA), a new method to build kernel code that is robust to memory allocation failures. AMA avoids the usual difficulties in handling allocation failures through a novel combination of static and dynamic techniques. Specifically, a developer, with assistance from AMA static analysis tools, determines how much memory a particular call into a kernel subsystem will need, and then pre-allocates said amount immediately upon entry to the kernel; subsequent allocation requests are served from the pre-allocated pool and thus guaranteed never to fail.

Membrane: Operating System Support for Restartable File Systems Jan - Sept 2009
(FAST 2010 - *Awarded Best Paper***)**

In this paper, we introduce Membrane, a set of changes to the operating system to support restartable file systems. Membrane allows an operating system to tolerate a broad class of file system failures and does so while remaining transparent to running applications; upon failure, the file system restarts, its state is restored, and pending application requests are served as if no failure had occurred. Membrane provides transparent recovery through a lightweight logging and checkpoint infrastructure, and includes novel techniques to improve performance and correctness of its fault-anticipation and recovery machinery.

Impact of Disk Corruption on Modern DBMS (ICDE 2010) Sept - Dec 2008

In this paper, we present a systematic study of the impact of disk corruptions on modern databases. Disk corruptions are commonplace today and no application can ignore their existence. We evaluate the impact of disk corruptions on MySQL and Postgress databases by injecting corruptions in the source code to simulate on-disk corruptions. We explore two classes of corruption: online and offline: Online corruptions are injected while the server is running and offline by corrupting the on-disk image. Overall we observed a large fraction of these injected faults result in the MySQL server being unavailable, returning incorrect results and corruption valid data on disk. The offline checker (myisamchk) is also incapable of detecting and correcting all errors, and may corrupt on-disk data in some cases.

WORK EXPERIENCE

Summer Intern, NetApp, Sunnyvale, CA June - Aug 2008

- **Image Backup User Level Tool** to browse the image file generated by the ONTAP Image Backup tool. The tool runs on any linux system independent of ONTAP. It can selectively display specific portions of the image file and allows the user to browse through the image as well as various WAFL snapshots as thought it was a regular filesystem. The tool shares code with the ONTAP kernel, but runs in user mode. The tool needs to be recompiled when IB code changes and it (almost) always starts working without requiring any rework. The original code had to be reorganized to support this feature.

Member Technical Staff, Database Security, Oracle India, Bangalore July 2005 - July 2007

- **Oracle Audit Vault** aims at helping enterprises achieve their regulatory compliance goals by consolidating (and archiving) the Audit Information from all over the enterprise.
  - **Alerts** are notifications that get fired when certain events occur. Involved in design and development of the alert evaluation and storage engine. Alert Creation and Evaluation involves translation of Alert Rules from Source Specific format to a Audit Vault Level generalized format. The translation process helps in creating a cache of data columns that are used in active alerts and faster evaluation of alert rules using the Oracle Rules Engine.
  - Design and Development of **Administrative Packages** and **Row-level Caching** of Audit Vault Schema. Administrative packages are employed to manage the various components of Audit Vault including audit sources, audit data collectors, audit data mapping etc. The read-only nature of this data can be exploited to improve performance by caching frequently accessed information.
  - As part of the development process, I also had to write extensive test suites in the ORATST framework of Oracle.
- **Oracle Fusion** - Oracle's next generation application development framework. Involved in development of lightweight users and XDB schema of Fusion architecture on Oracle 11g.

Research Intern, Insead Business School, Fontainebleau, France Jan - June 2005

- **NegoDeal** - Tool for simulating real-time games for the course Negotiation Analysis handled Prof Ayse Onculer. My work involved a transition from a quantitative to qualitative approach of handling the course, requiring several changes to interfaces and back-end of the tool that was developed in Delphi.
- **WebQuery** - Tool for extracting large volumes of data from lexis-nexis by simulating queries (*http post* and *get* requests) over the internet and analyzing their results. As lexis-nexis, the data provider, were moving their web interfaces, my worked involved moving the existing program to work with the new and more complex interface.

COURSE PROJECTS

Network Applications in the Voice-Over-Net Age Jan - May 2008

In a typical deployment of VoIP service, various network appliances coexists with VoIP traffic. We identified the impact of different applications on VoIP performance and call quality. Jitter, call setup delays and packet loss were quantified by taking packet traces.

Electronic Voting Jan - May 2008

Electronic Voting provides significant advantages over conventional voting, but poses several challenges as well. As part of this course project we explored the various security issues including establishing identities of voters, voting machines and the vote repository. We public key cryptography and Digital Certificates to developed a protocol that can satisfy these requirements.

Untangling ZFS Policies Oct - Dec 2007

We observed and quantitatively measured some of the striking policies of ZFS including block allocation and aggregation, intent logging under synchronous and asynchronous workloads of varied sizes. To obtain meaningful data, we developed a pseudo-device driver to perform semantic block analysis.

Recovery Strategies in Main Memory Databases Oct - Dec 2007

Designed and Implemented a simple multi-threaded main memory database based on a B+Tree with 2 phase locking, logging and transaction semantics, and studied the relative performance of two recovery strategies - recovery over network and recovery from check-point.

Cursive Handwritten Character Recognition Jan - Mar 2003

Developed in C using ALLEGRO graphics library involving Neural Networks. Project involved tokenization of cursive writing followed by feeding inputs to a back-propagating neural network.

Performance Analysis of Web Caching Algorithms Aug - Dec 2004

Analyzed the performance of various Cache Replacement Algorithms including FIFO, Least Recently Used etc by measuring the hit-ratios of these algorithms using the Squid Proxy Server.

WEPSIMS - Web Enabled Practice School Instruction Monitoring System Aug - Dec 2003

Designed and implemented web-interfaces and Java software modules to help the Practice School Division monitor instructors at various industrial centers.

COURSEWORK

Graduate Coursework - Advanced Operating Systems, Topics in Database Management Systems Implementation, Advanced Computer Networks, Advanced Natural Language Processing, Introduction to Information Security
Undergraduate Coursework - Operating Systems, Computer Networks, Network Programming, Programing Languages and Compiler Construction, Theory of Computation, Database Systems

SKILL SET

Programming Languages: C, C++,Java(beginner)
Markup & Scripting: Perl, XML, HTML, JavaScript
Development Platforms: Windows, Linux, Solaris(beginner)
Databases: Oracle

ACHIEVEMENTS

Topper of Computer Science Batch of 2005 (comprising of 120 students)
Recipient of BITS Merit Scholarship awarded to the Top 10 students of each batch (consisting of about 850 students) for 7 consecutive semesters.