UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

ORCA SECURITY LTD

Petitioner

v.

WIZ, INC.

Patent Owner

Inter Partes Review Case No. IPR2025-01087

U.S. Patent No. 12,003,529

DECLARATION OF EREZ ZADOK

TABLE OF CONTENTS

I. INTRODUCTION *6*

A. Background and Qualifications **6**

II. Overview and Legal Standards *19*

A. Person of Ordinary Skill in the Art **19**

B. Obviousness **21**

C. Analogous Art **25**

D. Claim Construction **26**

III. Background of Technology *27*

A. Cloud Computing **27**

B. Artificial Intelligence Systems **27**

C. Cybersecurity **28**

1. Common Vulnerabilities and Exposures (CVEs) *29*

2. Graph Databases in Security *29*

IV. Overview of the '529 Patent *30*

A. The '529 Patent **30**

V. Overview of THE RELEVANCE OF THE Prior Art *34*

A. Shua **34**

1. The Shua Patent and the Shua Provisional *34*

2. The Orca Security "SideScanning" Technical Brief *36*

B. Lang **41**

VI. Opinions Regarding Ground I *43*

A. Motivations to combine Shua and Lang **43**

B. Claim 1 ..... **45**

1. Claim 1[P]: "A method for detecting a cybersecurity risk of an artificial intelligence (AI)" *46*

2. Claim 1[a]: "generating an inspectable disk based on an original disk of a resource deployed in a computing environment" *48*

3. Claim 1[b]: "inspecting the inspectable disk for an AI model" *50*

4. Claim 1[c] : "generating a representation of the AI model in a security database, the security database including a representation of the computing environment"..... 53
5. Claim 1[d]: "inspecting the AI model for a cybersecurity risk" 59
6. Claim 1[e]: "generating a representation of the cybersecurity risk in the security database, the representation of the cybersecurity risk connected to the representation of the AI model in response to detecting the cybersecurity risk"..... 62
7. Claim 1[f] "initiating a mitigation action based on the cybersecurity risk"..... 67

C. Claim 2: "The method of claim 1, further comprising: detecting an artifact of the AI model on the inspectable disk" 70

D. Claim 3: "The method of claim 1, further comprising: inspecting the AI model to detect an AI model configured to execute a code object"..... 72

E. Claim 4: "The method of claim 1, further comprising: detecting a metadata of the AI model, wherein the metadata indicates that the AI model is a cybersecurity risk" 75

F. Claim 5: "The method of claim 1, further comprising: detecting in the AI model any one of: a secret, a certificate, a code, and any combination thereof"..... 77

G. Claim 6: "The method of claim 1, further comprising: generating a lateral movement path, wherein the lateral movement path includes the AI model, and a representation of a resource, wherein the resource is accessible utilizing the AI model" 82

H. Claim 7: "The method of claim 1, further comprising: applying a policy on the representation of the computing environment"..... 87

I. Claim 8: "The method of claim 7, further comprising: initiating the mitigation action based on the policy."..... 89

J. Claim 9..... 90
1. Claim 9[P]: "non-transitory computer-readable medium storing a set of instructions for detecting a cybersecurity risk of an artificial intelligence (AI), the set of instructions comprising: one or more instructions that, when executed by one or more processors of a device, cause the device to"..... 90
2. Claim 9[a]: "generate an inspectable disk based on an original disk of a resource deployed in a computing environment" 91
3. Claim 9[b]: "inspect the inspectable disk for an AI model" 91

4. Claim 9[c]: "generate a representation of the AI model in a security database, the security database including a representation of the computing environment" *92*
5. Claim 9[d]: "inspect the AI model for a cybersecurity risk" *92*
6. Claim 9[e]: "generate a representation of the cybersecurity risk in the security database, the representation of the cyber-security risk connected to the representation of the AI model in response to detecting the cybersecurity risk" *92*
7. Claim 9[f]: "initiate a mitigation action based on the cybersecurity risk" *92*

K. Claim 10 ..... **92**
1. Claim 10[P]: "A system for detecting a cybersecurity risk of an artificial intelligence (AI) comprising: a processing circuitry; a memory, the memory containing instructions that, when executed by the processing circuitry, configure the system to" *92*
2. Claim 10[a]: "generate an inspectable disk based on an original disk of a resource deployed in a computing environment" *94*
3. Claim10[b]: "inspect the inspectable disk for an AI model" *94*
4. Claim 10[c]: "generate a representation of the AI model in a security database, the security database including a representation of the computing environment" *94*
5. Claim 10[d]: "inspect the AI model for a cybersecurity risk" *95*
6. Claim 10[e]: "generate a representation of the cybersecurity risk in the security database, the representation of the cyber-security risk connected to the representation of the AI model in response to detecting the cybersecurity risk" *95*
7. Claim 10[f]: "initiate a mitigation action based on the cybersecurity risk" *95*

L. Claim 11: "The system of claim 10, wherein the memory contains further instructions which when executed by the processing circuitry further configure the system to: detect an artifact of the AI model on the inspectable disk" **95**

M. Claim 12: "The system of claim 10, wherein the memory contains further instructions which when executed by the processing circuitry further configure the system to: inspect the AI model to detect an AI model configured to execute a code object" **95**

N. Claim 13: "The system of claim 10, wherein the memory contains further instructions which when executed by the processing circuitry

further configure the system to: detect a metadata of the AI model, wherein the metadata indicates that the AI model is a cybersecurity risk" 96

O. Claim 14: "The system of claim 10, wherein the memory contains further instructions which when executed by the processing circuitry further configure the system to: detect in the AI model any one of: a secret, a certificate, a code, and any combination thereof" 96

P. Claim 15: "The system of claim 10, wherein the memory contains further instructions which when executed by the processing circuitry further configure the system to: generate a lateral movement path, wherein the lateral movement path includes the AI model, and a representation of a resource, wherein the resource is accessible utilizing the AI model" 96

Q. Claim 16: "The system of claim 10, wherein the memory contains further instructions which when executed by the processing circuitry further configure the system to: apply a policy on the representation of the computing environment" 96

R. Claim 17: "The system of claim 16, wherein the memory contains further instructions which when executed by the processing circuitry further configure the system to: initiate the mitigation action based on the policy" 97

VII. CONCLUSION 97

I, Erez Zadok, declare as follows:

I. INTRODUCTION

1. My name is Erez Zadok, and I am over 21 years of age and otherwise competent to make this Declaration. I make this Declaration based on facts and matters within my own knowledge and on information provided to me by others, and if called as a witness, I could and would competently testify to the matters set forth herein.

2. I have been retained as an expert in this matter by Counsel for Orca Security Ltd. ("Petitioner") to provide my independent opinions on certain issues requested by Counsel for Petitioner relating to the accompanying petition for Inter Partes Review of U.S. Patent No. 12,003,529 ("the '529 Patent").

3. The compensation my company receives in this matter is not based on the substance of my opinions or the outcome of this matter. I have no financial interest in Petitioner. I am being compensated at my standard hourly rate for my analysis and testimony in this case.

A. Background and Qualifications

4. I am a Professor in the Computer Science Department at Stony Brook University (part of the State University of New York ("SUNY") system). I direct the File-systems and Storage Lab (FSL) at Stony Brook's Computer Science Department. My research interests include file systems and storage systems,

operating systems, transactional systems including database technologies, information technology and system administration, security/privacy and information assurance, networking, energy efficiency, performance and benchmarking, virtualization, cloud systems, compilers, applied machine learning, and software engineering.

5. I studied at a professional high school in Israel, focusing on electrical engineering ("EE"), and graduated in 1982. I spent one more year at the high school's college division, receiving a special Certified Technician's degree in EE. I then went on to serve in the Israeli Defense Forces for three years (1983–1986). I received my Bachelor of Science degree in computer science ("CS") in 1991, my Master's degree in CS in 1994, and my PhD in CS in 2001—all from Columbia University in New York.

6. When I began my undergraduate studies at Columbia University, I also started working as a student assistant in the various campus-wide computer labs, eventually becoming an assistant to the head labs manager, who was managing all public computer labs on campus. During that time, I also became more involved with research within the CS Department at Columbia University, conducting research on operating systems, file and storage systems, distributed and networked systems, security, and other topics. I also assisted the CS department's computer

administrators in managing the department's computers, which included storage, IT, networking, and cyber-security related duties.

7. In 1991, I joined Columbia University's CS department as a full-time systems administrator, studying towards my MS degree part-time. My MS thesis topic is related to file system reliability, fault tolerance, replication, and failover in mobile networked storage systems using file virtualization. My main duties as a systems administrator involved installing, configuring, and managing many networked servers, proxies, and desktops running several operating systems, as well as network devices setup; this included many software and hardware upgrades, device upgrades, and BIOS firmware/chipset updates/upgrades. My duties also included ensuring reliable, secure, authenticated access to networked systems/storage and licensed software, as well as software updates, security and bug fixes. Examples of servers and their protocols included email (SMTP), file transfer (FTP), domain names (DNS), network file systems (NFS), network news systems (NNTP), and Web (HTTP).

8. In 1994, I left my systems administrator position to pursue my doctoral studies at Columbia University. My PhD thesis topic was on versatile file system development using stackable (virtualized) file systems, with examples in the fields of security and encryption, efficiency, reliability, and failover. I continued to work part-time as a systems administrator at the CS department, and eventually I was

asked to serve as manager to the entire information technology ("IT") staff. From 1991 to 2001, I was also a member of the faculty-level Facilities Committee that oversaw all IT operations at the CS department.

9. As part of my PhD studies at Columbia, I collaborated on projects to develop advanced AI-like techniques to detect previously unknown viruses (a.k.a. "zero-day malware"), using data mining and rule-based detection. This work led to several highly cited papers (over 1,600 citations for one of the papers alone) and two patents. I also became a Teaching Assistant ("TA") for a first-ever Computer Security course given at Columbia University's CS department with Dr. Matt Blaze as instructor.

10. From 1990 to 1998, I consulted for SOS Corporation and HydraWEB Technologies as a systems administrator and programmer, managing data storage use and backup/restore duties, databases, web servers, as well as information assurance and cyber-security (*e.g.*, malware protection, software licensing). From 1994 to 2000, I led projects at HydraWEB Technologies, and then became the Director of Software Development—overseeing the development of several products and appliances such as stateful firewalls and HTTP load-balancers, utilizing network-virtualization and high-availability techniques. From 2009 to 2019, I have consulted for Packet General Networks, a startup specializing in secure, virtualized, network storage and applications' data security in the cloud.

11. In 2001, I joined the faculty of Stony Brook University, a position I have held since that time. In 2002, I joined the Operations Committee, which oversees the IT operations of the CS department at Stony Brook University. From 2006 to 2010, I was the Director of IT Operations of the CS department. My day-to-day duties included setting policies regarding computing, hiring and training new staff, assisting any staff with topics of my specialty, defining requirements for new software/hardware, and purchasing. From 2010 to 2015, I had served as the Co-Chair to the Operations Committee. From 2016 to 2019, I oversaw the IT Operations as the Chair of the Operations Committee. A significant component of these duties included defining and helping implement policies for data management, to ensure the security of users and their data, and data reliability and availability, while minimizing the inconvenience and performance impact to users. I personally helped setup and maintain an initial virtual-host infrastructure in the department. Since late 2019, I've been a member of the department's Executive Committee that also oversees all IT operations.

12. In 2017, I became the department's Graduate Academic Adviser, advising all Master students (over 400 annually on average) and many other graduate students on an assortment of academic matters. In 2024, I took over as the department's Graduate Program Director, overseeing the entire graduate CS program (700-800 students annually on average).

13. Since 2001, I have personally configured and managed my own research lab's network. This includes setting up and configuring multiple storage systems (*e.g.*, NFS, CIFS/SMB, NAS), virtual and physical environments, applications such as database (*e.g.*, MySQL, PostgreSQL), Web servers (*e.g.*, Apache), and mail servers; user access control (*e.g.*, NIS, LDAP, Active Directory), backups and restores, snapshot policies, and more. I have personally installed, configured, changed, replaced parts, and upgraded components in numerous devices including mobile devices, laptops, desktops, and servers, both physical and virtual.

14. Since 1995, I have taught courses on operating systems, storage and file systems, advanced systems programming in Unix/C, systems administration, data structures, data/software security, and more. My courses often use storage, file systems, distributed systems, and system/network security as key teaching principles and practical examples for assignments and projects. I have taught these concepts and techniques to my students, both to my direct advisees as well as in my courses. For example, in my graduate Operating Systems course, I often cover Linux's kernel mechanisms to protect users, applications, and data files, virtual file systems, as well as distributed storage systems (*e.g.*, NFS). And in the System Administration undergraduate course, I covered many topics such as networking, storage, backups, and configuring complex applications such as mail, web, and database servers.

15. My research often investigates computer systems from many angles: security, efficiency, energy use, scalability, reliability, portability, survivability, usability, ease-of-use, versatility, flexibility, and more. My research gives special attention to balancing five often-conflicting aspects of computer systems: performance, reliability, energy use, security, and ease-of-use.

16. Since joining Stony Brook University in 2001, my group in the File-systems and Storage Lab ("FSL") has developed many file systems and operating system extensions; examples include a highly-secure cryptographic file system, a portable copy-on-write ("COW") versioning file system, a tracing file system useful to detect intrusions, a replaying file system useful for forensics, a snapshotting and sandboxing file system, a namespace unification file system (that uses stackable, virtualized, file-based COW), an anti-virus file system, an integrity-checking file system, a load balancing and replication/mirroring file system, network file system extensions for security and performance, distributed secure cloud-based storage systems, transactional key-value stores and file systems, OS-level embedded databases, a compiler to convert user-level C code to in-kernel efficient yet safe code, GCC plugins, stackable file system templates, and a Web-based backup system. Many of these projects used one form of virtualization or another (storage, network, host, etc.). I continue to maintain and release newer versions of some of these file systems and software.

17. I have published over 120 refereed publications (in ACM, IEEE, USENIX, and more). To date, my publications have been cited more than 10,000 times (as per Google Scholar as of May 30, 2025). My papers cover a wide range of related technologies such as file systems, storage systems, transactional systems, security, clouds and virtualization, performance benchmarking and optimization, energy efficiency, system administration, web systems, and more. I also published a book titled "Linux NFS and Automounter Administration" (Sybex, 2001), covering systems administration topics related to network storage and data security.

18. Some of my research has led to public software releases that have been used worldwide. I have publicly maintained the Amd Berkeley Automounter in a package called "am-utils" since 1992; this software helps administrators manage the multitude of file system mounts on dozens of different Unix systems, especially helping to automate access to multiple NFS/NAS storage volumes. Since 1997, I have maintained and released several stackable (virtualized) file system software projects for Linux, FreeBSD, and/or Sun Solaris, in a package called FiST. One of my stackable file system encryption projects, called Cryptfs, became the basis for IBM's public release of eCryptfs, now part of Linux. Packet General Networks, for whom I have provided consulting services between 2009 and 2019, licensed another encryption file system called Ncryptfs. Another popular file system released in 2003, called Unionfs, offers virtual namespace unification, transparent shadow copying

(a.k.a. copy-on-write or COW), file system snapshotting (*e.g.*, useful for forensics and disaster recovery), and the ability to save disk space by sharing a read-only copy of data across several computers, among other features.

19. My research and teaching make extensive use of data security features. For example, each time I taught the graduate operating system course, the first homework assignment includes the creation of a new system call that performs new or added functionality, often for encrypting a file or verifying its integrity; many of my other assignments cover topics of user/process access control, anti-virus filtering, and more. Since 2001, over 1,000 graduate students were exposed to these principles directly through my teaching and research at Stony Brook University.

20. Moreover, in an undergraduate course titled "Advanced Systems Programming in Unix/C," I cover many topics of system security and vulnerabilities, such as the structure of UNIX processes, and memory segments such as the heap and stack. This course covers details of several hundred Linux system calls. Often, the first assignment for this course is to develop a tool to encrypt/decrypt files using advanced ciphers, use digital signatures to certify the cipher keys used, and reliably recover files in case of failures. Since 2001, several hundred undergraduate students were exposed to these principles directly through my teaching and research at Stony Brook University.

21. In another undergraduate course, System Administration, I taught network configuration, security, and storage configuration and reliability. In a special topics course on Storage Systems, I covered many topics such as data deduplication, RAID, transactional storage, storage hardware including modern Flash based ones, virtual storage, backup/restore, snapshots and continuous data protection ("CDP"), NAS and SAN, and NFS.

22. Overall, in addition to the aforementioned experience, my technical experience relevant to the '549 Patent at the time of the alleged invention included the following. With respect to storage and disks, I've conducted extensive research, published, taught, managed systems, and released software related to storage media (*e.g.*, disks), file systems, and networked and cloud based storage systems. This included copy-on-write (CoW) and snapshotting systems. Much of my work relates to security of data, including anti-virus scanning of file systems, encrypted storage, integrity checking, and more. With respect to AI experience, I first studied AI in the late 1980s. In the late 1990s, I collaborative on early generation AI-assisted research to detect zero-day malware; this work led to several highly cited publications as well as two patents. Since then I integrated AI techniques into my research and teaching, including applied machine learning (*e.g.*, neural networks, decision trees, and more), meta-heuristics such genetic algorithms and simulated annealing, statistical regression-based techniques, and more. In 2018 I was part of a National Science

Foundation (NSF) multi-day summit to chart the course of AI and ML techniques in systems and storage research (a report was published). As AI grew more in popularity, I began to collaborate and even lead industry, state, and federally funded projects involving AI/ML techniques. In one example, I led a team that used advanced Natural Language Processing (NLP) techniques to solve complex networked file/storage systems; as part of that work, the team integrated Large Language Models (LLMs) into the project. Several of my publications are in the fields of ML systems, NLP, and visual analytics.

23. My research has been supported by many federal and state grants as well as industry awards, including an NSF CAREER award, two IBM Faculty awards, two NetApp Faculty awards, a Western Digital award, a Facebook award, several Dell-EMC awards, and several equipment gifts. I received the 2008 SUNY Chancellor's Excellence in Teaching award, and the 2022 SUNY Chancellor's Award for Excellence in Scholarship and Creative Activities (both awards can be given only once in a lifetime). In 2021, I was named an ACM Distinguished Member for "Outstanding Scientific Contributions to Computing."

24. My service record to the community includes serving as the co-chair for the USENIX Annual Technical Conference in 2020 (ATC'20); serving as the co-chair for USENIX File and Storage Technologies (FAST'15) in 2015 and on the FAST Conference Steering Committee from 2015 to 2023; serving on the ACM

HotStorage Steering Committee since 2021; and serving as the co-chair in 2012 and on the Steering Committee of the ACM SYSTOR conference since 2012. I have served as an Associate Editor to the ACM Transactions on Storage ("TOS") journal from 2009 to 2022; in 2022, I was named the Editor-in-Chief for ACM's TOS journal.

25. I am a named inventor on four patents, two titled "Systems and Methods for Detection of New Malicious Executables" (U.S. Patent No. 7,487,544, issued February 3, 2009; and U.S. Patent No. 7,979,907, issued July 12, 2011); and two more titled "Multi-Tier Caching," (U.S. Patent No. 9,355,109, issued May 31, 2016; and U.S. Patent 9,959,279, issued May 1, 2018).

26. I have been disclosed as a testifying expert in 19 cases (including *inter partes* review ("IPR") proceedings) in the past four years. I have been deposed 13 times and testified in trial twice.

27. A complete copy of my curriculum vitae, including further details on my education, experience, publications, patents, and other qualifications to render an expert opinion, is attached as Appendix A.

28. In writing this declaration, I have considered my own knowledge and experience, including my work experience in the fields of software engineering, cybersecurity, artificial intelligence and machine learning, and computer networking; my experience teaching those subjects; my experience consulting in those subjects;

and my experience creating standards and certifications for those subjects. In reaching my opinions in this matter, I have focused on the following references and materials, which I have considered from the viewpoint of a POSITA as of the priority date of the ’529 Patent:

Exhibit	Description
1001	U.S. Patent No. 12,003,529 to Cohen et al.
1002	File History of the ’529 Patent
1003	Declaration of Dr. Erez Zadok
1004	U.S. Patent No. 11,489,863 to Shua
1005	U.S. Provisional Patent App. No. 63/180,048 to Shua
1006	U.S. Patent Application Publication No. 2024/02022405A1 to Lang et al.
1007	Orca Security SideScanning Technical Brief, Orca Security, 2020
1008	Understanding Cloud Computing Basics (https://online.eou.edu/resources/article/understanding-cloud-computing-basics/), retrieved May 29, 2025
1009	The NIST Definition of Cloud Computing, NIST SP 800-145, September 2011
1010	NIST Cloud Computing Standards Roadmap, NIST SP 500-291, V2, July 2013
1011	What is Artificial Intelligence? (https://www.ibm.com/think/topics/artificial-intelligence), retrieved June 2, 2025
1012	What is Machine Learning? (https://www.ibm.com/think/topics/machine-learning), retrieved June 2, 2025
1013	NIST AI Risk Management Framework, NIST AI 100-1, January 2023
1014	What is Cybersecurity? (https://www.cisco.com/site/us/en/learn/topics/security/what-is-cybersecurity.html), retrieved May 29, 2025
1015	What is Cloud Security? (https://www.ibm.com/think/topics/cloud-security), retrieved June 2, 2025
1016	NIST Guide to Using Vulnerability Naming Schemes, NIST SP 800-51, R1, February 2011

1017	Graph Database for CyberSecurity (https://www.nebula-graph.io/posts/how-to-use-graphs-for-cybersecurity), retrieved June 2, 2025
1018	Neo4j Graph Database Concepts (https://neo4j.com/docs/getting-started/appendix/graphdb-concepts/), retrieved June 2, 2025
1019	About Archive and Standard Disk Snapshots (https://cloud.google.com/compute/docs/disks/snapshots), retrieved June 2, 2025
1020	PyTorch GitHub Page (https://github.com/pytorch/pytorch), retrieved June 2, 2025

II. OVERVIEW AND LEGAL STANDARDS

29. In formulating my opinions, I have been instructed to apply certain legal standards. I am not a lawyer. I do not expect to offer any testimony regarding what the law is. Instead, the following sections summarize the law as I have been instructed to apply it in formulating and rendering my opinions found later in this declaration. I understand that, in an inter partes review proceeding, patent claims may be deemed unpatentable if it is shown that they were anticipated or rendered obvious in view of the prior art. I understand that prior art in an inter partes review is limited to patents or printed publications that predate the priority date of the patent at issue. I understand that questions of claim clarity (definiteness) and enablement cannot be considered as a ground for considering the patentability of a claim in these proceedings.

A. Person of Ordinary Skill in the Art

30. My opinions are provided based on what a person of ordinary skill in the art in the technical field of the invention would have understood at the time of

the purported invention of the ’529 patent. I have been asked to assume the date of the purported invention for the ’529 patent is February 22, 2024 (the “earliest possible priority date”).

31. I have been informed and understand that the content of a patent and prior art should be interpreted the way a person of ordinary skill in the art would have interpreted those references at the time of the invention of the patent. I understand that the person of ordinary skill is a hypothetical concept referring to one who thinks along the lines of conventional wisdom at the time. I also understand that the factors considered in determining the ordinary level of skill in the art may include: (i) the levels of education of the inventor; (ii) the types of problems encountered in the art; (iii) prior art solutions to those problems; (iv) the rapidity with which innovations are made; (v) the sophistication of the technology; and (vi) the educational level of persons working in the field.

32. I was asked to provide an opinion as to the level of one of ordinary skill in the art pertinent to the subject matter set forth in the ’529 patent at the time of the earliest possible priority date.

33. A POSITA at the time of the ’529 Patent would have had a bachelor’s degree in computer engineering, computer science, electrical engineering, or a similar degree, with about 2-3 years of experience in the fields of cybersecurity and

artificial intelligence; additional education or experience might substitute for these requirements.

34. As of February 22, 2024, I had more than ordinary skill in the art. I am, however, familiar with the skills and knowledge possessed by those I would have considered to be of ordinary skill in the art as of that date. For example, I knew people working in the field while I earned my academic degrees and contributed to the field through literature and inventions, as I discussed in my Background (§I.A above) and demonstrated in my CV (attached as Appendix A).

35. My opinions provided in this declaration would not change in view of minor modifications to this level of ordinary skill.

B. Obviousness

36. I have been informed and understand that subject matter claimed in a patent is obvious under 35 U. S. C. §103 if a person of ordinary skill in the art at the time the alleged invention was made would have had reason to combine or modify the disclosures of one or more prior art references to arrive at the claimed subject matter. A claim may be unpatentable even if each and every claim limitation is not present or disclosed in a single prior-art item.

37. I have been informed and understand that, under the doctrine of obviousness, a claim is unpatentable if the differences between the invention and the prior art are such that the subject matter as a whole would have been obvious at the

time the invention was made to a person having ordinary skill in the art to which the subject matter pertains. A person of ordinary skill in the art is presumed to know about the relevant prior art at the time of the claimed invention.

38. I have been informed and understand that obviousness is based on the scope and content of the prior art, the differences between the prior art and the claim, the level of ordinary skill in the art, and secondary indicia of obviousness and non-obviousness to the extent such indications exist.

39. I have been informed and understand the determination of whether the asserted claims would have been obvious to a person of ordinary skill in the art and, therefore, invalid, is not governed by any rigid test or formula. A determination that a claim is obvious is, instead, based on a common-sense determination that the claimed invention is merely a combination of known limitations to achieve predictable results. Any of the following rationales are acceptable justifications to conclude that a claim would have been obvious: (1) the claimed invention is simply a combination of known prior art methods to yield predictable results; (2) the claimed invention is a simple substitution of one known element for another to obtain predictable results; (3) the claimed invention uses known techniques to improve similar devices (methods, or products) in the same way; (4) the claimed invention applies a known technique to a known device (method, or product) ready for improvement to yield predictable results; (5) the claimed invention was "obvious

to try"—choosing from a finite number of identified, predictable solutions, with a reasonable expectation of success; (6) there is known work in one field of endeavor that may prompt variations of it for use in either the same field or a different one based on design incentives or other market forces if the variations would have been predictable to one of ordinary skill in the art; or, (7) there is some teaching, suggestion, or motivation in the prior art that would have led one of ordinary skill in the art to modify the prior art reference to combine prior art teachings to arrive at the claimed inventions.

40. I have been informed and understand that a claim may be obvious in light of a single reference without the need to combine references and that any elements of the claim that are not found in the reference (if any) can be supplied by the common sense or knowledge of one of ordinary skill in the art or taught in different areas of the single reference.

41. I have been informed and understand that an analysis of whether a claimed invention is obvious must not rely on a hindsight combination of prior art. The analysis must proceed in the context of the time of the invention or claimed priority date and consider whether the invention as a whole would have been obvious to a person of ordinary skill in the art, taking into consideration any interrelated teachings of the prior art, the effects of demands known to the design community or present in the marketplace, and the background knowledge possessed by a person

having ordinary skill in the art, all to determine whether there was an apparent reason to combine any known elements in the fashion claimed by the patent at issue.

42. I have been informed that it is improper to combine references where the references teach away from their combination. I am informed that a reference may be said to teach away when a person of ordinary skill in the relevant art, upon reading the reference, would be discouraged from following the path set out in the reference, or would be led in a direction divergent from the path that was taken by the patent applicant. In general, a reference will teach away if it suggests that the line of development flowing from the reference's disclosure is unlikely to be productive of the result sought by the patentee. I am informed that a reference teaches away, for example, if (1) the combination would produce a seemingly inoperative device, or (2) the references leave the impression that the product would not have the property sought by the patentee. I also am informed, however, that a reference does not teach away if it merely expresses a general preference for an alternative invention but does not criticize, discredit, or otherwise discourage investigation into the invention claimed.

43. I have been informed that even if a prima facie case of obviousness is established, the final determination of obviousness must also consider "secondary considerations" if presented. In most instances, the patentee raises these secondary considerations of non-obviousness. In that context, the patentee argues that an

invention would not have been obvious in view of these considerations, which include: (a) commercial success of a product due to the merits of the claimed invention; (b) a long-felt, but unsatisfied need for the invention; (c) failure of others to find the solution provided by the claimed invention; (d) deliberate copying of the invention by others; (e) unexpected results achieved by the invention; (f) praise of the invention by others skilled in the art; (g) lack of independent simultaneous invention within a comparatively short space of time; (h) teaching away from the invention in the prior art.

44. I have further been informed that secondary considerations evidence is only relevant if the offering party establishes a connection, or nexus, between the evidence and the claimed invention. The nexus cannot be based on prior art features. The establishment of a nexus is a question of fact. While I understand that the Patent Owner here has not offered any secondary considerations at this time, I reserve the right to supplement my opinions in the event that the Patent Owner raises secondary considerations during the course of this proceeding.

45. Finally, I understand that obviousness in an IPR must be proven by a preponderance of the evidence.

C. Analogous Art

46. I have been informed that for a prior art reference to be proper for use in an obviousness analysis, the reference must be "analogous art" to the claimed

invention. I have been informed that a reference is analogous art to the claimed invention if: (1) the reference is from the same field of endeavor as the claimed invention (even if it addresses a different problem); or (2) the reference is reasonably pertinent to the problem faced by the inventor (even if it is not in the same field of endeavor as the claimed invention). In order for a reference to be "reasonably pertinent" to the problem, it must logically have commended itself to an inventor's attention in considering his problem. In determining whether a reference is reasonably pertinent, one should consider the problem faced by the inventor, as reflected either explicitly or implicitly, in the specification. I believe that all of the references that my opinions in this IPR are based upon are well within the range of references a person having ordinary skill in the art would consult to address the type of problems described in the Challenged Claims.

D. Claim Construction

47. I understand that, for purposes of my analysis in this inter partes review proceeding, the terms appearing in the patent claims should be interpreted according to their "ordinary and customary meaning." In determining the ordinary and customary meaning, I understand that the words of a claim are first given their plain meaning that those words would have had to a person of ordinary skill in the art at the time of the alleged invention. I also understand that the structure of the claims, the specification and file history may be used to better construe a claim. Moreover,

I understand that even treatises and dictionaries may be used, albeit under limited circumstances, to determine the meaning attribute by a person of ordinary skill in the art to a claim term.

III. BACKGROUND OF TECHNOLOGY

A. Cloud Computing

48. Cloud computing is computing services delivered over the Internet. Cloud computing providers maintain networks of physical and virtual servers that contain the software and services to be provided. EX1008, *Understanding Cloud Computing Basics*, p. 1. The National Institute of Standards and Technology (NIST) defines cloud computing as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction." EX1009, *The NIST Definition of Cloud Computing*, p. 6. The NIST also published a Cloud Computing Standards Roadmap that outlines how to securely and effectively adopt the Cloud Computing model. EX1010, *NIST Cloud Computing Standards Roadmap*, p. 17.

B. Artificial Intelligence Systems

49. Artificial intelligence (AI) systems enable computers and machines to simulate human learning, comprehension, problem solving, decision making,

creativity and autonomy. EX1011, *What is Artificial Intelligence?*, p. 1. Machine Learning (ML) is a branch of AI focused on enabling computers and machines to imitate the way that humans learn, to perform tasks autonomously, and to improve their performance and accuracy through experience and exposure to more data. EX1012, *What is Machine Learning?*, p. 1. AI models are used for machine learning and can be implemented using deep neural networks. *Id.*

50. AI technologies have significant potential to transform society and people's lives. EX1013, *NIST AI Risk Management Framework*, p. 1. AI technologies can drive inclusive economic growth and support scientific advancements that improve the conditions of our world. *Id.* AI technologies pose risks that can negatively impact individuals, groups, organizations, communities, etc. *Id.* Without proper controls, AI systems can amplify, perpetuate, or exacerbate inequitable or undesirable outcomes. *Id.* AI risks include data poisoning, data tampering, data breaches, unauthorized manipulation of AI models, and operational risks, such as model drift. EX1011, *What is Artificial Intelligence?*, p. 6.

C. Cybersecurity

51. In general, cybersecurity is the practice of protecting systems, networks, and programs from digital attacks. EX1014, *What is Cybersecurity*?, p. 1. Successful cybersecurity systems have multiple layers of protection spread across computers, networks, programs, or data that one intends to keep safe. *Id.* A specific

type of cybersecurity is cloud security. *Id.* Cloud security is a collection of procedures and technologies designed to address external and internal threats to business security. EX1015, *What is Cloud Security?*, p. 1. Organizations need cloud security as they incorporate cloud-based tools and services as part of their infrastructure. *Id.* Security threats explicitly target cloud computing providers due to an organization's overall lack of visibility in data access and movement. *Id.,* p. 3.

1. Common Vulnerabilities and Exposures (CVEs)

52. The CVE vulnerability naming scheme is for a dictionary of unique, common names for publicly known software flaws. EX1016, *NIST Guide to Using Vulnerability Naming Schemes*, p. 2. CVE provides a comprehensive list of publicly known software flaws, a globally unique name to identify each vulnerability, a basis for discussing priorities and risks of vulnerabilities, and a way for a user of disparate products and services to integrate vulnerability information. *Id.* A CVE vulnerability entry consists of a unique identifier number, for example "CVE-2000-0001", a short description of the vulnerability, and references to public advisories on the vulnerability. *Id.*

2. Graph Databases in Security

53. Graph databases can provide a holistic view of an infrastructure, including each item on a network. EX1017, *Graph Database for Cybersecurity*, p. 1.

Graphs represent data as nodes and edges. *Id.* Using graph data structures and visualization techniques can be analyzed to unearth cybersecurity issues. *Id.*

54. Neo4j is a property graph database model consisting of nodes that can be connected by edges. EX1018, *Neo4j Graph Database Concepts*, p. 1. Each node can have a label and properties, while each edge can have a type and properties. *Id.*

IV. OVERVIEW OF THE ’529 PATENT

A. The ’529 Patent

55. The ’529 Patent relates generally to “detecting artificial models,” and more specifically to “implementing cybersecurity monitoring solutions on an AI development pipeline of a cloud computing environment.” EX1001, *’529 Patent*, 1:8-11. For example, the ’529 Patent describes “a method for detecting a cybersecurity risk of an artificial intelligence (AI)” in a computing environment. *Id.*, 1:61-64. The method may include inspecting the computing environment, generating a representation of the AI model in a security database, inspecting the AI model for a cybersecurity risk, generating a representation of the cybersecurity risk in a security database, and/or initiating a mitigation action. *Id.*, 1:61-2:12. The ’529 Patent illustrates said method in FIG. 6 below:

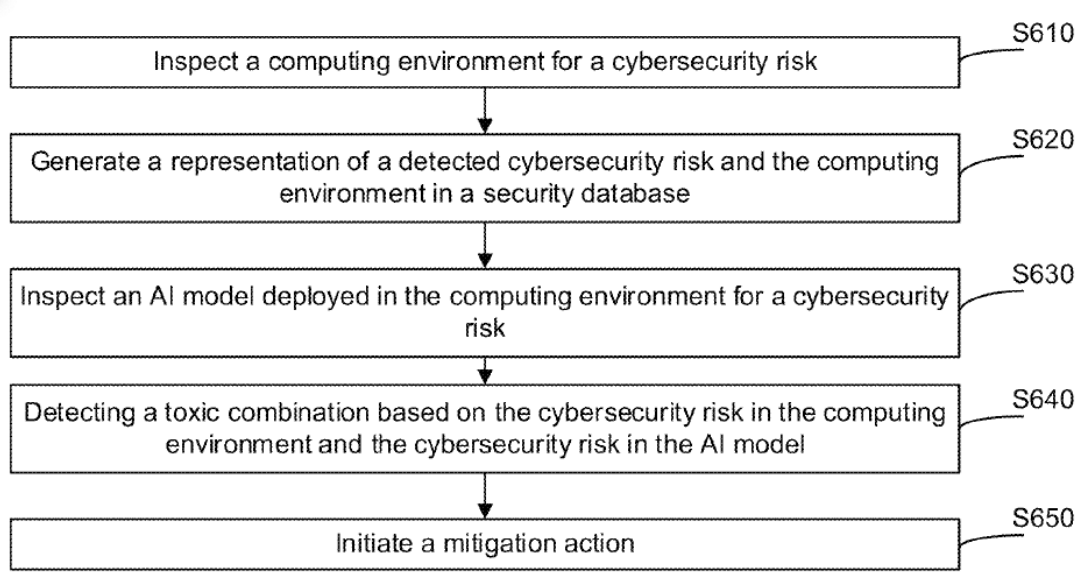


FIGURE 6

EX1001, *'529 Patent*, FIG. 6 above.

56. As shown in FIG. 6 above, the method begins with "[i]nspect[ing] a computing environment for a cybersecurity risk" with respect to an AI model, at step S610. *Id.*, 15:26-36. Step S610 also includes the process of "generating" and inspecting "an inspectable disk based on an original disk," where the "inspectable disc" may be a clone of an original disk, a snapshot of an original disk, or a copy of an original disk. *Id.*, 15:45-51, 15:32-36.

57. At step S620, "a representation is generated for a resource," including the AI model and computing environment, are each "stored in a security database[.]" *Id.*, 7:46-60, 15:61-66. The '529 Patent teaches that the "security database is implemented as a graph database," and "includes a plurality of nodes and connections between the nodes." *Id.*, 8:34-39. As represented in FIG. 2 below:

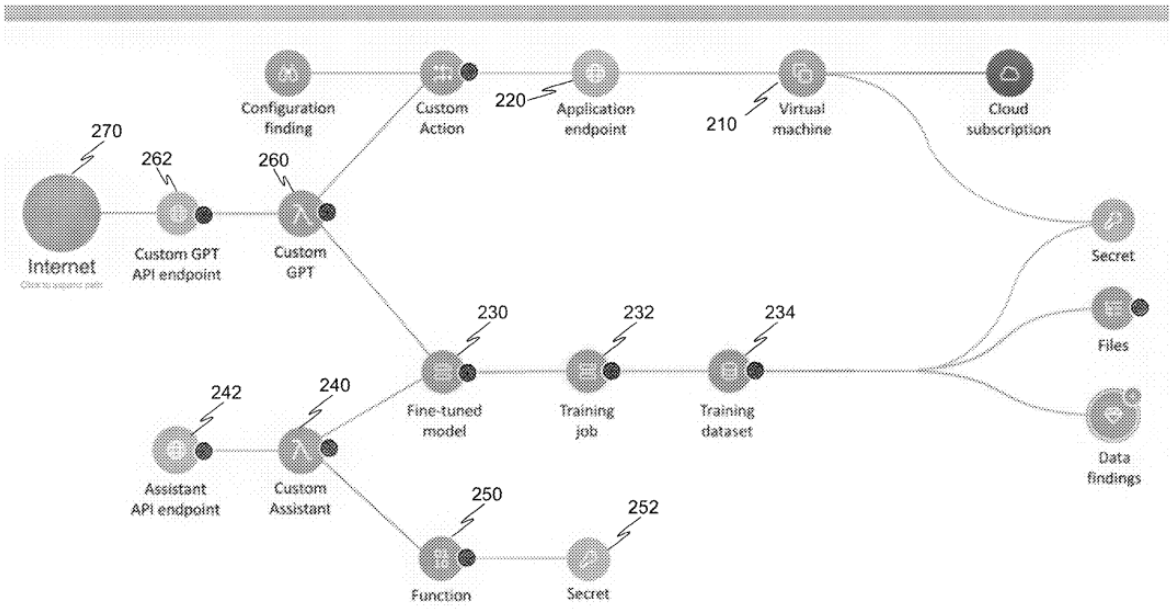


FIGURE 2

Id., FIG. 2 above.

58. Next, at step S630, "an AI model is inspected for [] a cybersecurity risk." EX1001, *'529 Patent*, 16:9-15, FIG. 6. The "cybersecurity risks" can be identified by matching "a version, identifier, and the like" of an AI model previously determined to be a cybersecurity risk, or "detecting a secret, a code, … which the AI model, when prompted with an appropriate prompt, is configured to output, leading to leaked data, leaked secrets, etc." *Id.*, 16:21-24, 16:28-31.

59. In sum and at a high level, the '529 Patent describes a minor modification of the well-known and widely practiced cloud cybersecurity technique of scanning a customer's cloud footprint to generate a graph of the interconnections between the cloud assets and identify known vulnerabilities that might allow

unauthorized access to sensitive data. Shua, as described in §V.A below, is one such example of a system for scanning a customer's cloud footprint to generate a graph of the interconnections between the cloud assets and identify (and mitigate) known vulnerabilities that might allow unauthorized access to sensitive data:

> Embodiments of the present disclosure may deliver instant-on, workload level visibility across cloud infrastructure assets without running opcodes on the target scanned workloads, helping organizations to: (1) Detect risks such as vulnerabilities, malware, misconfigurations, lateral movement risk, and unsecured sensitive data; (2) See cloud inventory at every layer: infrastructure, operating system, applications, and data; (3) Discover and see previously missed assets.

EX1004, *Shua*, 2:66-3:7; *see also generally*, EX1004, *Shua*.

60. The purported advancement of the '529 Patent is scanning for and identifying vulnerabilities in AI models, in addition to doing so for other cloud assets. EX1001, *'529 Patent*, 1:15-35. However, locating AI models and identifying security vulnerabilities in AI models was also well known before the filing date of the '529 Patent. Lang (*see* §V.B below) describes one system for locating and scanning AI models. *See* EX1006, *Lang*, [0030] (locating AI models in virtual machine environments); [0021] (identifying vulnerabilities in AI models). Particularly, given the emphasis in cloud scanning systems on enumerating and mapping all of customers' cloud assets (*see* EX1005, *Shua Provisional*, [0014], [0029], [0069]), it is my opinion that the claims of the '529 Patent represent an

obvious modification of a well-known prior art system to scan a well-known class of workloads. I lay out the detailed rationale for this opinion next.

61. Throughout this Declaration, emphases in quoted passages are in bold unless otherwise noted.

V. OVERVIEW OF THE RELEVANCE OF THE PRIOR ART

A. Shua

62. "Shua," as used in this Declaration, generally refers to the joint disclosures of EX1004, U.S. Patent No. 11,489,863 to Shua, and EX1005, U.S. Provisional Patent Application No. 63/180,048, unless the context indicates otherwise. The Shua Patent incorporates the disclosures of the Shua Provisional by reference. EX1004, *Shua*, 1:6-10. I have been informed that, when a patent application clearly evidences intent to incorporate another document by reference (such as by using the phrase "incorporated by reference"), the disclosures of the incorporated document are treated as part of the text of the incorporating document. Under this instruction, I treat disclosures of Shua and the Shua Provisional as a single prior art reference, although I cite to the exhibits individually.

1. The Shua Patent and the Shua Provisional

63. Shua broadly relates to "systems and methods for securing cloud infrastructure." EX1004, *Shua*, 1:20-21. The Shua Provisional provides additional details, including the goal: to scan an organization's complete inventory of cloud

assets (such as virtual machines) for vulnerabilities, misconfigurations, and lateral movement risk. EX1005, *Shua Provisional,* [0001].

64. Shua discloses an out-of-band process to reach cloud workloads through the runtime storage layer, combining with metadata gathered from cloud providers. EX1005, *Shua Provisional*, [0029]; *See also*, EX1004, *Shua*, 3:8-12, 17:38-41. Shua describes how the process advantageously covers 100% of an organization's assets. *Id.* To achieve the aforementioned process, Shua discloses performing a side scanning function to protect against potential vulnerabilities. EX1004, *Shua*, 22:44-46. As a part of this process, after connecting with a cloud provider API, the system creates snapshots of virtual machine storage disks and performs scans for network access or security information to detect potential vulnerabilities. *Id.*, 12:45-48; 22:48-58. Shua's scanning system may also analyze data related to the cloud infrastructure and generate a map representing the systems and devices in the cloud. EX1004, *Shua*, 9:53-64.

65. Like the '529 Patent, Shua is directed to the field of endeavor of security in cloud computing environments. For example, the '529 Patent states that "The present disclosure relates … specifically to implementing cybersecurity monitoring solutions on an AI development pipeline of a cloud computing environment." Similarly, Shua is directed to "cloud cyber security" including for a customer's "CI/CD [Continuous Integration/Continuous Deployment (a common

software development practice)] pipeline." EX1005, *Shua Provisional*, *Title;* [0012]. I would thus understand both the '529 Patent and Shua to be in the field of endeavor of "cybersecurity for cloud computing." It is thus my opinion that Shua would be analogous art to the '529 Patent at least because it is in the same field of endeavor.

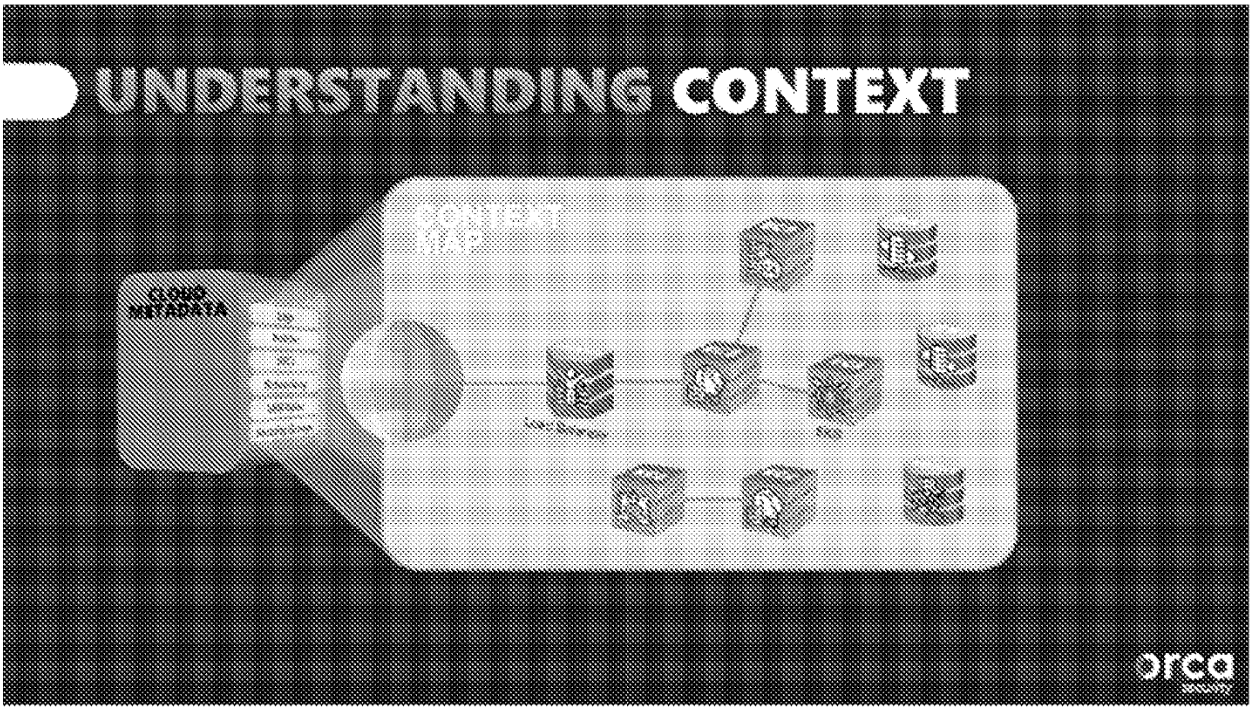
66. Furthermore, Shua "provide[s] technical solutions to challenges associated with cloud cybersecurity." EX1004, *Shua,* 6:33-34. Because Shua provides technical solutions to cloud cybersecurity challenges, it is highly relevant to the problem purportedly being solved by the inventors of the '529 Patent: "that of implementing cybersecurity monitoring solutions on … a cloud computing environment." EX1001, '529 *Patent*, *1:8-11.* It is thus my opinion that Shua would be analogous art to the '529 Patent at least because it is reasonably pertinent to the problem being solved by the inventors as well.

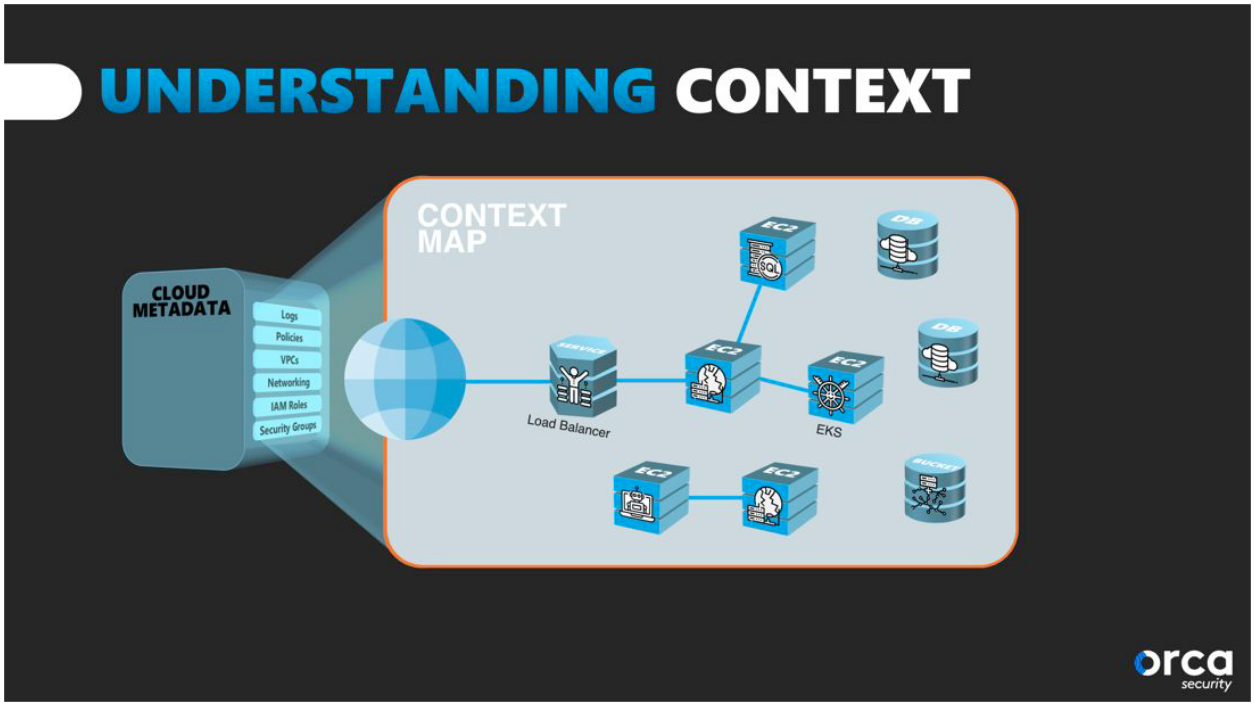
2. The Orca Security "SideScanning" Technical Brief

67. The Shua Provisional includes several figures that are of low image quality. *See, e.g.,* EX1005, *Shua Provisional,* pp. 15, 25, 30. In my experience, this is common when color images are included in documents processed by Patent Office systems: the colors are often replaced with black-and-white or grayscale images of lower resolution. I have been provided with a document entitled "SideScanning Technical Brief," which includes higher quality versions of these images. EX1007, *Orca Security SideScanning Technical Brief*, pp. 9, 17, 15. I have not relied upon

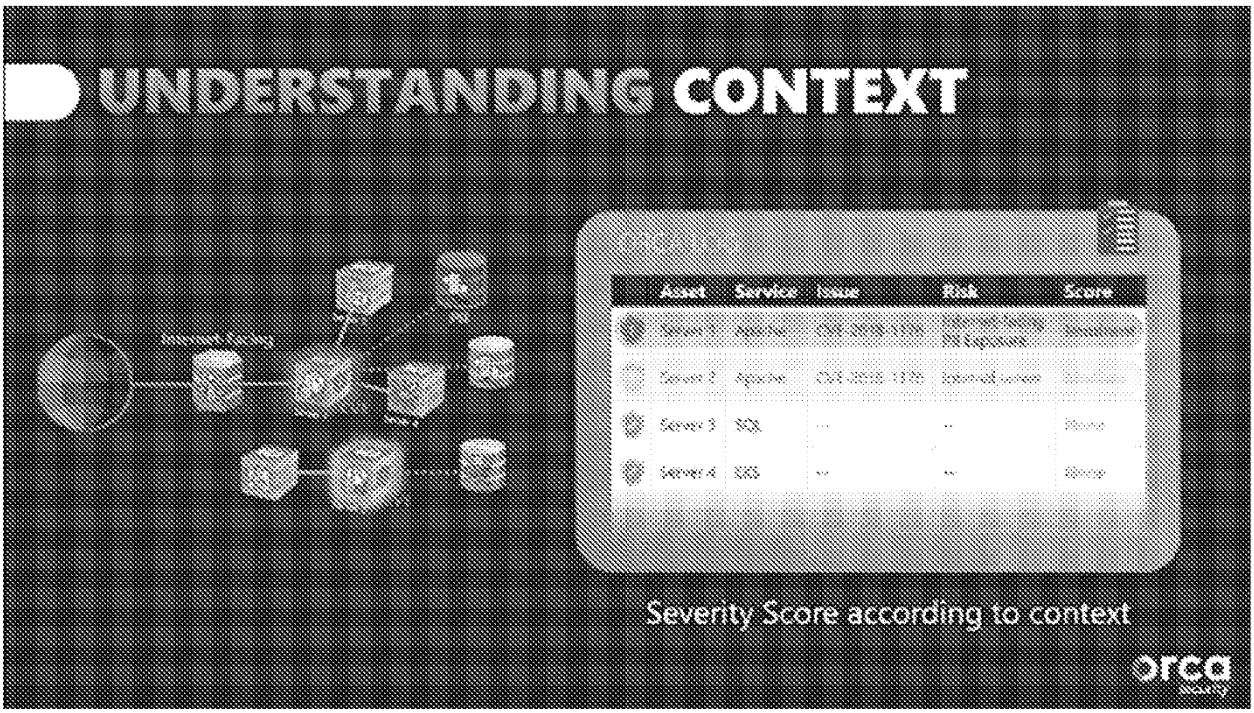
this document as a basis for my opinions as to the teachings of Shua and/or the Shua Provisional; however, I have examined and compared the figures appearing in the *Shua Provisional* and the *Orca Security SideScanning Technical Brief,* and it is my opinion that certain figures in *Orca Security SideScanning Technical Brief* are identical but for image quality to the corresponding figures appearing in the *Shua Provisional*.

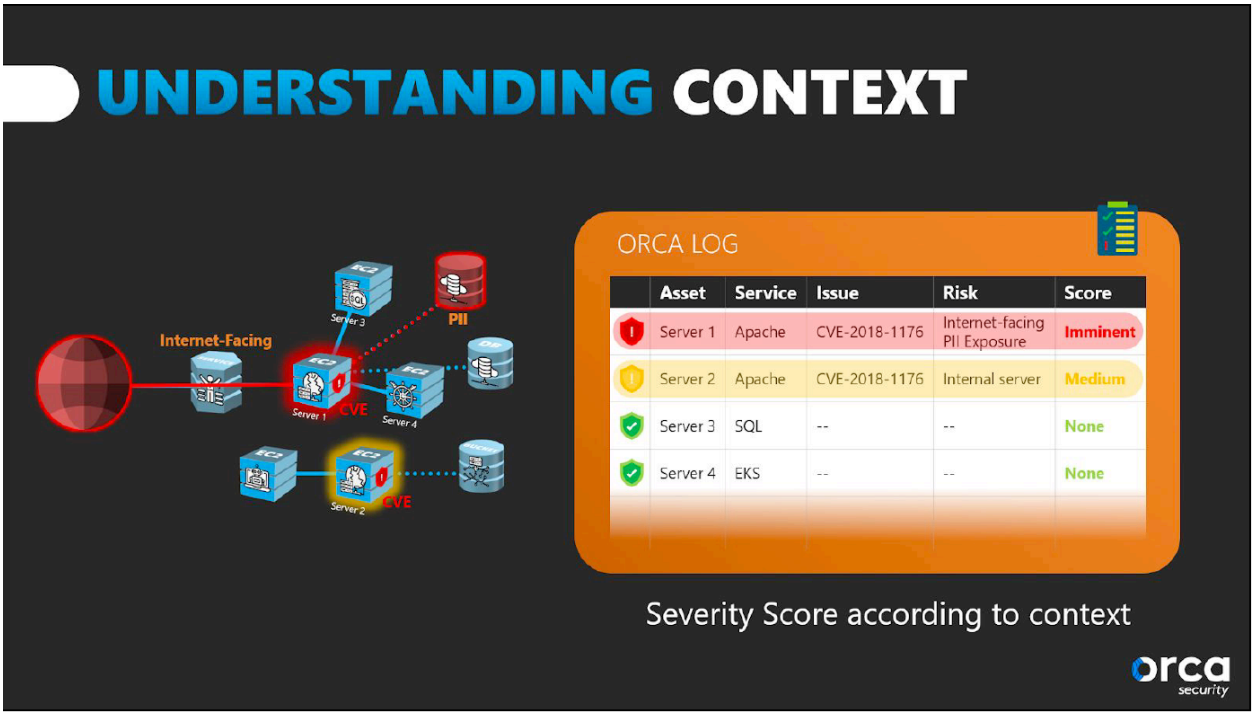
68. For the sake of readability of this Declaration, I reproduce below the figures from the *Orca Security SideScanning Technical Brief* in this Declaration. However, for completeness, I reproduce below the low-quality versions of the images from the *Shua Provisional* and the corresponding higher-quality images from the *Orca Security SideScanning Technical Brief* below, and it is the images from the *Shua Provisional* that form the basis of my opinions:



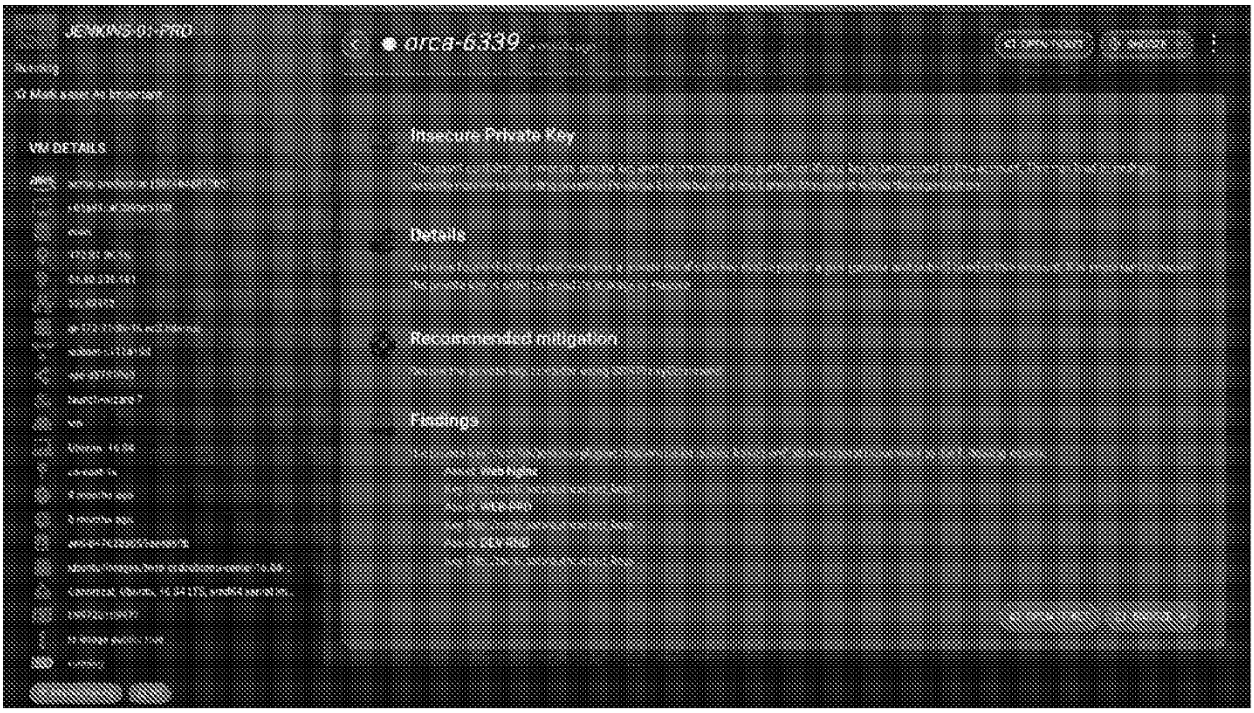


EX1005, *Shua Provisional*, p. 15 (top); EX1007, *Orca Security SideScanning Technical Brief*, p. 9 (bottom).





EX1005, *Shua Provisional*, p. 25; EX1007, *Orca Security SideScanning Technical* Brief, p. 17.



EX1005, *Shua Provisional*, p. 30; EX1007, *Orca Security SideScanning Technical Brief*, p. 13.

B. Lang

69. Lang relates to a method for analyzing computing systems, including AI systems. EX1006, *Lang*, [0003]. More specifically, Lang discloses "automatically and semiautomatically training, detecting, analyzing, monitoring, protecting, and hardening AI systems and their components, including dependencies, binaries, source code, data, etc., to improve the system's trust, reliability, accuracy, fairness, security, etc., and protect it from potential vulnerabilities, weaknesses, threats, limitations, etc." *Id.*

70. Like the '529 Patent and Shua, Lang is directed to the field of endeavor of security in cloud computing and (like the '529 Patent) Lang is specifically focused on security for AI systems. Broadly, Lang's system is directed to:

> automatically and semi-automatically training, **detecting, analyzing, monitoring, protecting, and hardening AI systems** and their components, including dependencies, binaries, source code, data, etc., to improve the system's trust, reliability, accuracy, fairness, security, etc., **and protect it from potential vulnerabilities**, weaknesses, threats, limitations, etc.

Lang, [0003].

71. For example, Lang specifies that "[b]efore, during and/or after deployment, [the disclosed Deployment and/or Monitoring System] may **check the system for cybersecurity risks**, **vulnerabilities**, endpoint vulnerabilities (if applicable), and/ or network vulnerabilities, etc." EX1006, *Lang,* [0274]. Lang also

specifies that "**the trained AI systems may be deployed in** several environments, including but not limited to, **the Cloud**, edge devices, a pipeline (*e.g.*, a MLaaS pipeline, a MLOps pipeline, **and/or CI/CD pipeline**, etc.)." *Id.,* [0124]. As such, it is my opinion that Lang shares the field of endeavor of "cybersecurity for cloud computing" with the '529 Patent and Shua. It is my opinion that Lang would thus be analogous art to the '529 Patent at least because it is in the same field of endeavor.

72. Furthermore, as Lang points out, "AI systems are being increasingly used across many industries, making these systems more susceptible to attacks and widening the attack surface." *Id.*, [0010]. At the time of the invention of the '529 Patent, this would have been well understood by a POSITA.

73. Accordingly, Lang's techniques for monitoring, protecting, and hardening AI models would have been highly relevant to the problem being solved by the inventors of the '529 Patent, that of implementing cybersecurity monitoring solutions on an AI development pipeline of a cloud computing environment. *Compare* EX1001, *'529 Patent*, 1:8-11 ("The present disclosure relates generally to detecting artificial intelligence models, and specifically to implementing cybersecurity monitoring solutions on an AI development pipeline of a cloud computing environment.") *with* EX1006, *Lang,* [0003] ("**embodiments of the invention are directed at automatically** and semi-automatically training, **detecting**, analyzing, **monitoring, protecting, and hardening AI systems** and their

components, including dependencies, binaries, source code, data, etc., to **improve the system's** trust, reliability, accuracy, fairness, **security, etc., and protect it from potential vulnerabilities,** weaknesses, threats, limitations, etc.").

74. It is thus my opinion that Lang would be analogous art analogous art to the '529 Patent at least because it is reasonably pertinent to the problem being solved by the inventors as well.

VI. OPINIONS REGARDING GROUND I

A. Motivations to combine Shua and Lang

75. In my opinion, for at least three reasons, a POSITA would have been motivated and able to modify Shua's system to include Lang's specific teachings of scanning for and identifying cybersecurity risks in AI models. Additional reasons for specific modifications to Shua's system are given below, in connection with specific claim elements. This section provides an overview of the compatibility and combinability of the teachings of Shua and Lang.

76. **First**, Shua teaches that it is desirable to scan "100% of an organization's assets." EX1005, *Shua Provisional*, [0029]; *see also* [001] ("The cloud estate of an organization may include its complete inventory of cloud assets including running and stopped workloads of all types… Organizations are searching for effective ways to scan their cloud estate to look for risks stemming from vulnerabilities…"); [0038]; [0014]; [0032]; [0069]. A POSITA at the time of the

invention would have understood, the AI models described by the '529 Patent to be simply another workload to be scanned for cybersecurity risks and vulnerabilities. This is reinforced by Lang's teachings that AI systems are "prone to adversarial attacks," but "difficult to audit." EX1006, *Lang*, [0006].

77. **Second**, many of Shua's teachings directly apply to workloads that include AI models. For example, Shua's system "gathers library versions, software versions and other identifying characteristics, and in a later phase tries matching them to known vulnerabilities in its vulnerability database." EX1005, *Shua Provisional*, [0042].

78. Similar to Shua's teachings, but in the specific context of AI models, Lang teaches a database of AI source code snippets that includes "version numbers of libraries used" to be used for tasks including "vulnerability analysis." EX1006, *Lang*, [0302]. However, Lang goes into further details about identifying risks specific to AI models by, for example, "analyzing intermediate results of the model, analyzing training data as it is being read into the analysis system, detecting poisoning attacks, low quality explanations, detecting biases, analyzing the model for overfitting and/or underfitting, etc." *Id.*, [0326]. A POSITA at the time of the invention would have understood that it would be beneficial to identify these AI-model-specific risks in cloud workload as a part of Shua's system.

79. **Third**, a POSITA would have been readily able to modify Shua's system to include Lang's teachings. Such a combination would have constituted the use of a known technique (Lang's scanning of AI models) to improve a similar device (Shua's system for scanning an organizations assets) in the same way (Shua's system would be improved in the same way as Lang's, because the additional cybersecurity risks present in the AI models would be detected). Incorporating Lang's scanning of AI models into Shua's scanning system for all of an organization's assets would have required at most minor modifications in software. Accordingly, a POSITA would have enjoyed a reasonable expectation of success in making such a combination.

B. Claim 1

80. As laid out in detail next, it is my opinion that claim 1 is obvious over Shua in view of Lang. In particular, Shua's system teaches scanning for cybersecurity risks in a cloud computing environment, and Lang's system teaches detecting cybersecurity risks of AI models, specifically, including in a cloud computing environment. EX1004, *Shua*, 2:66-3:7; EX1006, *Lang*, *Abstract,* [0006]. The combination of Shua's scanning system and Lang's techniques for analyzing AI models would be a natural (and obvious) one for a POSITA to make.

1. Claim 1[P]: "A method for detecting a cybersecurity risk of an artificial intelligence (AI)"

81. To the extent the preamble is limiting, it is my opinion that limitation 1[P] is obvious over Shua in view of Lang.

82. In particular, Shua discloses techniques for detecting risk in a cloud environment (*i.e.*, a method for detecting a cybersecurity risk): "Embodiments of the present disclosure may deliver instant-on, workload level visibility across cloud infrastructure assets … helping organizations to: [d]etect risks such as vulnerabilities, malware, misconfigurations, lateral movement risk, and unsecured sensitive data[.]" EX1004, *Shua*, 2:66-3:7. Lang discloses "a method for analyzing a computing system for properties of a machine learning model" in order "to ensure AI systems are working as intended," because they are "difficult to audit." EX1006, *Lang*, *Abstract;* [0006]; *see also* [0020] (it is an object of Lang's invention to "determine … vulnerabilities" in AI computing systems). Thus, the combined Shua-Lang system would detect cybersecurity risks in artificial intelligence systems.

83. A POSITA at the time of the invention would have understood that Lang's "machine learning model" to be an example of instances of the "AI models" of the '529 Patent. For example, Lang defines "machine learning" to be a particular "[s]ubfield of AI." *Id.*, [0102]. Lang also teaches that "AI components may be related to machine learning models…" *Id.*, [0155]. For these specific reasons as well as the general use of the terms "machine learning model and "AI model" in

Lang, a POSITA would have understood that Lang's "machine learning model" falls under the term "AI model" as used by the '529 Patent.

84. A POSITA would further understand that these machine learning models (and the systems running them) would be part of "the complete inventory of cloud assets" for an organization, as Shua uses that term. For example, Lang specifically references AI components residing on many of the same cloud computing platforms that Shua describes. *See* EX1005, *Shua Provisional*, [0014] ("Orca delivers instant-on, work-load level visibility across 100% of **AWS [Amazon Web Services]**, **Azure**, and **GCP [Google Cloud Platform]** assets") *with* EX1006, *Lang*, [0019] ("AI components may come from a MLaaS pipeline, including but not limited to, **Amazon** Sagemaker, **Azure**, **Google Cloud** Machine Learning, **Google** AI Platform, **Google Cloud** AutoML, **Microsoft Azure** MLOps, IBMWatson, BigML, and/or WhizzML, etc."). As such, a POSITA at the time of the invention would have understood that it would be consistent with Shua's stated goal of scanning "100% of an organization's assets" to scan Lang's machine learning models. *See* EX1005, *Shua Provisional*, [0029].

85. **In sum**, Shua in view of Lang teaches *a method for detecting a cybersecurity risk* (Shua's teaching of providing visibility across an organization's cloud assets to detect risks) *of an artificial intelligence (AI)* (Lang's teaching of analyzing the properties of a machine learning model to determine vulnerabilities).

2. Claim 1[a]: "generating an inspectable disk based on an original disk of a resource deployed in a computing environment"

86. It is my opinion that limitation 1[a] is obvious over Shua in view of Lang.

87. First, a POSITA at the time of the invention would have understood, that Shua's teaching of taking snapshots of data volumes falls within the claimed step of *generating an inspectable disk based on an original disk.* This would be the case at least because the '529 specification gives creating a "snapshot of the original disk" as one example of creating "an inspectable disk based on an original disk of a resource." EX1001, '*529 Patent*, 15:47-51.

88. A POSITA would also understand this to be the case because taking a snapshot of a data volume is a standard practice when inspecting data volumes in order to generate a consistent disk image at a single point in time. For example, I note the EX1019, a documentation article for GCP, entitled About Archive and Standard Disk Snapshots and downloaded from https://cloud.google.com/compute/docs/disks/snapshots, notes that "[a] snapshot captures the contents of a disk whether or not the disk is attached to a running virtual machine (VM) instance" and that archive snapshots are "data related to compliance [and] audits." EX1019, *About Archive and Standard Disk Snapshots*, pp. 1; 3. I

note that this is consistent with Lang's teaching that AI components may be analyzed "during compliance audits." EX1006, *Lang*, [0119].

89. Indeed, snapshots are commonly used to capture the contents of a running virtual machine. *Id.*, p. 1. Given that EX1019 is documentation for Google Cloud Platform, a POSITA would have understood that the virtual machines being discussed here include at least virtual machines running on GCP, and a POSITA would have understood that Shua's reference to "the machine" to refer to a virtual machine running in a cloud computing environment such as GCP; a POSITA would also understand this to be a "resource" as the term is used by the '529 Patent. As such, a POSITA at the time of the invention would have understood from Shua's teaching that "[a] snapshot of the disk state may be created through AWS or other cloud providers while the machine is running" to teach or disclose *generating an inspectable disk based on an original disk of a resource deployed in a computing environment.*

90. **In sum**, Shua in view of Lang teaches *generating an inspectable disk* (taking a snapshot) *based on an original disk* (the data volumes) *of a resource* (cloud asset) *deployed in a computing environment* (the customer's cloud account or multi-cloud deployment).

3. Claim 1[b]: "inspecting the inspectable disk for an AI model"

91. It is my opinion that limitation 1[b] is obvious over Shua in view of Lang.

92. First, Shua teaches *inspecting the inspectable disk*:

> Orca Security may involve a one-time, essentially instantaneous, impact-free integration into AWS, Azure, or GCP. Following its one-time integration, Orca scans the configuration, network layout, and security configuration. (This may be performed via a scanner account.) It does so while also reading into virtual machines, disks, databases, and datastores, as well as logs for cloud assets. It then analyzes the data and builds a full-stack inventory. Next it automatically assesses the security state of discovered asset [sic] throughout the technology stack, including all four cloud layers: I/S, OS, apps, and data.

EX1005, *Shua Provisional*, [0030].

93. A POSITA would have understood that Shua's teaching that the system "analyzes the data" to indicate that Shua teaches inspecting the snapshots. *See, e.g.,* EX1005, *Shua Provisional*, [0033] (describing a process of creating snapshots of block storage for subsequent analysis). Shua further emphasizes that "every asset in the account may be enumerated in all regions." *Id.*, [0035]; *see also* [0033] (Orca builds a map of the customer's "entire estate"); EX1004, *Shua*, 9:53-94. Shua discusses analyzing the snapshot (*i.e.*, inspecting the inspectable disk) for assets and

services such as clusters, repositories, functions, and CloudFront services, and discusses "Vulnerability Scanning." EX1005, *Shua Provisional*, [0039].

94. Further, Lang teaches inspecting file systems from virtual machines to extract and reverse AI components therefrom:

> FIG. 32 depicts the analysis system for extracting and reversing AI components from containers and/or virtual machines (VM). the [sic] container and/or VM may be ingested by the analysis system and AI components are extracted, such as (but not limited to) analyzing files on the file system, reversing binaries and extracting AI components, detecting installed AI dependencies, analyzing AI dependency versions and metadata, etc.

EX1006, *Lang*, [0331]; *see also* [0195] ("FIG. 12 illustrates how AI models may be detected, understood and/or reversed"); [0030]-[0032].

95. Lang expressly discloses that this analysis system can operate on "VM files, VM hard disk files and/or any other data related to containers or VMs that may or may not contain AI information, and/or other information pertaining to containers or VMs." *Id.*, [0332]. A POSITA at the time of the invention would have understood, that Lang's teachings of "VM files, VM hard disk files and/or any other data related to containers or VMs that may or may not contain AI information," would include snapshots of VM block storage as captured by Shua's system. Accordingly, Lang teaches a process of *inspecting the inspectable disk for an AI model*.

96. Furthermore, in addition to the motivations discussed in §VI.A above, it would have been obvious to a POSITA at the time of the '529 patent to modify the inspection of assets and services on an inspectable disk, as in Shua, to include the inspection specifically for an AI model as taught by Lang. Both Shua and Lang relate to security analyses of assets in a cloud environment, and it is my opinion that a POSITA would be motivated to modify Shua to include AI systems in view of increased needs in the industry "to ensure AI systems are working as intended." EX1006, *Lang*, [0006].

97. Lang states that AI systems are "prone to adversarial attacks, wherein an attack manipulates the AI model or data (incl. training or inference data) in a way that affects the model's behavior and decision-making." *Id.* Lang also states that AI models are "highly proprietary" and "frequently provide information about crucial systems and their data and behavior," and are thus "a target for theft." *Id.* Accordingly, it is my opinion that a POSITA would be motivated to assess cybersecurity risks for AI services (as taught by Lang) in Shua's system for assessing risks of assets and services in a cloud computing environment at least because of the cybersecurity risks of AI models described by Lang. *See also* limitation 1[P] (§VI.B.1 above).

98. **In sum**, Shua in view of Lang teaches *inspecting* (Shua's analyzing data and/or Lang's detecting AI models and extracting AI components) *the*

inspectable disk (Shua's snapshot) *for an AI model* (Lang's AI models or AI components).

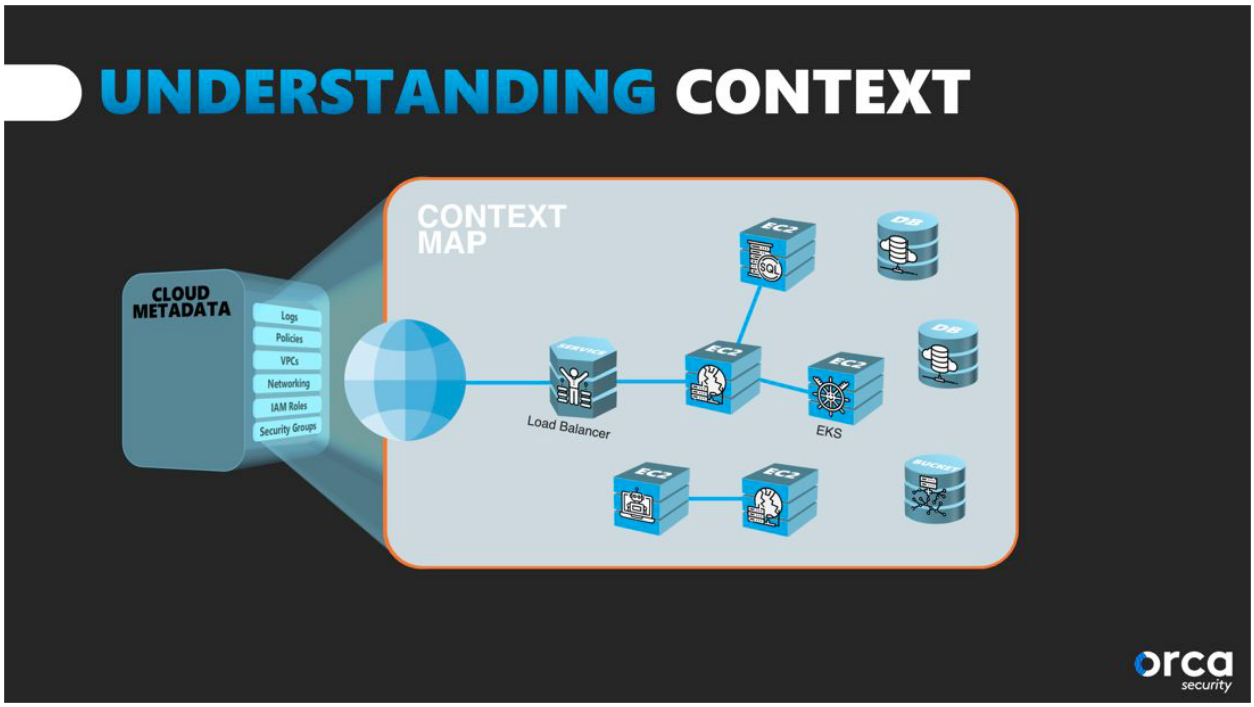
4. Claim 1[c] : "generating a representation of the AI model in a security database, the security database including a representation of the computing environment"

99. It is my opinion that limitation 1[c] is obvious over Shua in view of Lang.

100. First, a POSITA would have understood that the term "security database" in the '529 Patent encompassed a stored data structure representing the computing environment as nodes, edges, and attributes. The '529 Patent gives "graph database, such as Neo4j®" as one example of a security database. EX1001, '*529 Patent*, 7:58-60. A POSITA would have understood that a graph database is a form of stored data structure where (1) entities are represented as "nodes," (2) connections and/or relationships between entities are represented by "edges" connecting the nodes, and (3) properties of nodes and edges are represented by "attributes" attached to the nodes and edges. EX1018, *Neo4j Graph Database Concepts*; s*ee also* §III.C.2 above. Neo4j, given as an example of a security database, is such a stored data structure.

101. Second, it is my opinion that Shua teaches generating a representation of the computing environment via scanning the customer's "entire environment" to "build a map" of it. EX1005, *Shua Provisional*, [0033]. This map is shown as a

graph that includes nodes representing assets, edges representing the connections between them, and attributes representing the properties of the nodes and/or connections. This map can be presented to the user as a visualization, as shown below:



EX1007, *Orca Security SideScanning Technical Brief*, p. 9, corresponding to the lower-quality image appearing at EX1005, *Shua Provisional*, p. 15.

102. A POSITA would have understood that this map includes nodes corresponding to assets such as a service providing load balancing, and various EC2 instances (which a POSITA would have understood to refer to Amazon's Elastic Compute Cloud service providing cloud-based virtual machine servers) providing other assets such as web servers and database servers. The map additionally includes

edges representing (networking or logical) connections between the nodes and attributes describing the function of various nodes (such as the "Load Balancer" attribute attached to the service node and the "EKS" [Amazon's Elastic Kubernetes Service] attribute attached to one of the EC2 nodes). This is further supported by other disclosures of Shua:

> In step 227, scanning system 101 may generate a map of cloud infrastructure 106. In some embodiments, **this map may be in a form of graph: a plurality of interconnected vectors connecting the plurality of systems and devices,** based on the networking configuration.

EX1004, *Shua*, 13:1-5.

103. This graph is also annotated with attributes representing properties of the assets:

> Generating the map in step 227 may include recording information such a region identifier, site identifier, datacenter identifier, physical address, network address, workload name, or any other identifier which may be acquired via an API provided through a cloud service provider's system.

Id., 13:36-40.

104. Finally, a POSITA at the time of the invention would have understood, that Shua discloses generating a stored data structure of this representation, at least because generating it includes "recording information" (*id.*) and because Shua

contemplates comparing the original map to a "new map" generated subsequent to a change:

> **In some embodiments, scanning system 101 may monitor cloud change,** or "delta," logs (e.g., Cloud event logs such as AWS CloudTrail, **monitoring** a network-related change in state, **a trust-related change in state**, or a disk configuration-related change in state in at least one of the primary asset group or a secondary asset group) and contextualize associated risks discovered within a customer cloud account. **In some embodiments, scanning system 101 may generate a new map of assets based on a review of all systems or devices in cloud infrastructure** 106 (e.g., as described above with respect to step 203), **and compare the new map to the map generated earlier** in step 203 to determine which assets have changed. In some embodiments, this comparison may require comparing data, metadata, network connections, software configurations, firewall configurations, or any other aspects related to cloud infrastructure 106, including, for example, determining new assets that are created or assets for which their configuration has changed.

Id., 11:3-20.

105. Third, Lang teaches *generating a representation of the AI model*. For example, Lang teaches generating different representations of the AI model located via inspection, including an "abstract AI model" suitable for import into another system (such as Shua's system):

> An abstract AI model 1260 module may process and/or abstract AI model functions into semantical representations for example to allow for easier import into a system… [¶] **Abstracting AI model functions into semantical representations may be implemented using approaches such as** (but not limited to): semantic reasoning about detected AI tasks, mathematical comparison between the run binaries and/or abstracted functions, comparison to known and/or common AI model designs and/or formats such as neural network architectures, and/or other **approaches to create an abstract representation of the AI model.** If abstraction is successful, **the output of an abstract AI model 1260 module may include an abstracted AI model, which may be written to a storage such as a storage device, and/or networking device, etc.**

EX1006, *Lang*, [0210]-[0211].

106. This step is performed as part of the method of Lang's FIG. 12, where AI models are detected in a computer binary. *Id.,* [0195]. Finally, a POSITA at the time of the invention would have understood, that such a computer binary would be present on the snapshots of the disk captured from virtual machines running the AI models.

107. Furthermore, a POSITA at the time of the invention would have understood that, in the Shua-Lang system, the representation generated by Lang's system "for easier import into a system" would be imported into, and stored with, Shua's map. This would be the case at least because of Lang's teachings that the

abstract representation is designed "for easier import into a system" such as an analysis system (EX1006, *Lang*, [0210]) and Shua's stated desire to map the customer's entire cloud estate (EX1005, *Shua Provisional*, [001]). In the Shua-Lang system, Shua's would capture a snapshot of a disk from a virtual machine in the customer's cloud computing account, and Lang's system would inspect files on the snapshot to identify the AI model (*see* limitation 1[b] in §VI.B.3 above) and generate an abstract representation of the AI model that can be stored in Shua's map representing the customer's computing environment.

108. In addition to the motivations discussed in §VI.A above, it is my opinion that a POSITA would be motivated to add Lang's representations of the identified AI models into Shua's map based on Shua's teaching that the map should include the customer's "entire environment," at least because the AI models would represent cloud assets that are part of the customer's "entire environment." EX1005, *Shua Provisional*, [0033].

109. **In sum**, Shua in view of Lang teaches *generating a representation of the AI model* (adding a node in Shua's map corresponding to the AI model detected by Lang's analysis) *in a security database* (Shua's map)*, the security database including a representation of the computing environment* (Shua's map including nodes representing the customer's cloud assets, edges representing the connections

between them, and attributes representing the properties of the nodes and/or connections).

5. Claim 1[d]: "inspecting the AI model for a cybersecurity risk"

110. It is my opinion that limitation 1[d] is obvious over Shua in view of Lang.

111. First, Shua generally teaches inspecting all of the customer's assets for vulnerabilities:

> Orca Security may involve a one-time, essentially instantaneous, impact-free integration into AWS, Azure, or GCP. Following its one-time integration, Orca scans the configuration, network layout, and security configuration. (This may be performed via a scanner account.) It does so while also reading into virtual machines, disks, databases, and datastores, as well as logs for cloud assets. **It then analyzes the data and builds a full-stack inventory. Next it automatically assesses the security state of discovered asset** [sic] **throughout the technology stack, including all four cloud layers: I/S, OS, apps, and data.**

EX1005, *Shua Provisional*, [0030]; [0042].

112. A POSITA would have understood that AI models, as detected by Lang, would be a part of this technology stack; as such, their "security state" would be assessed with that of the other assets. This security state assessment would include a vulnerability assessment:

> In **performing vulnerability assessment**, Orca may extract the OS packages, installed software applications, libraries, and program language libraries such as Java archives, Python packages, Go modules, Ruby gems, PHP packages, and Node.js modules, or any other software applications. It gathers library versions, software versions and other identifying characteristics, and **in a later phase tries matching them to known vulnerabilities in its vulnerability database** (e.g. data structure of known vulnerabilities).

EX1005, *Shua Provisional*, [0042].

113. Furthermore, Lang teaches specifically *inspecting the AI model for a cybersecurity risk,* as a part of the analysis of detected AI models:

> For example, the analysis action may provide a variety of analyses and/or provide a variety of results. This may include, but is not limited to, analyzing the accuracy of models, detecting poisoning in data, binary analysis, source code analysis, detecting adversarial attacks, assessing adversarial robustness, analyzing reliability, finding privacy violations, analyzing safety and/or security of the system, analyzing the system against compliance controls, detecting vulnerabilities and/or risks, evaluating the models against OOD data, analyzing explainability (and transparency) aspects of the model, analyzing bias, analyzing fairness, analyzing flexibility, and/or testing the security of endpoints, etc.

Lang, [0120]; *see also* [0270] (giving a "poisoning attack" as one example of "vulnerabilities and/or attacks" on AI models).

114. A POSITA would have understood that the analysis described here (such as detecting vulnerabilities) to include inspecting for a cybersecurity risk, as such vulnerabilities are examples of cybersecurity risks. *See also* limitation 1[P] (§VI.B.1 above). A POSITA at the time of the invention of the '529 Patent would have understood that the AI models would be identified assets in the Shua-Lang system, and that Lang's vulnerabilities and/or attacks would represent cybersecurity risks of those assets.

115. In addition to the motivations discussed in §VI.A above, it is my opinion that it would be obvious to modify Shua's system to add Lang's inspection of the identified AI models for cybersecurity risks based on Shua's teaching that the system should "identify risks 803 associated with each of the identified assets." This is the case at least because a POSITA would understand from Shua's teachings that it is desirable to identify all of the risks across the customer's cloud estate. EX1005, *Shua Provisional*, [001]. Additionally, it is a goal of Shua to present a "unified view of [the customer's] assets in order to detect and prioritize vulnerabilities." *Id*., [0116]. A POSITA would have understood that such a unified view would include cybersecurity risks associated with AI models and been motivated to incorporate Lang's teachings.

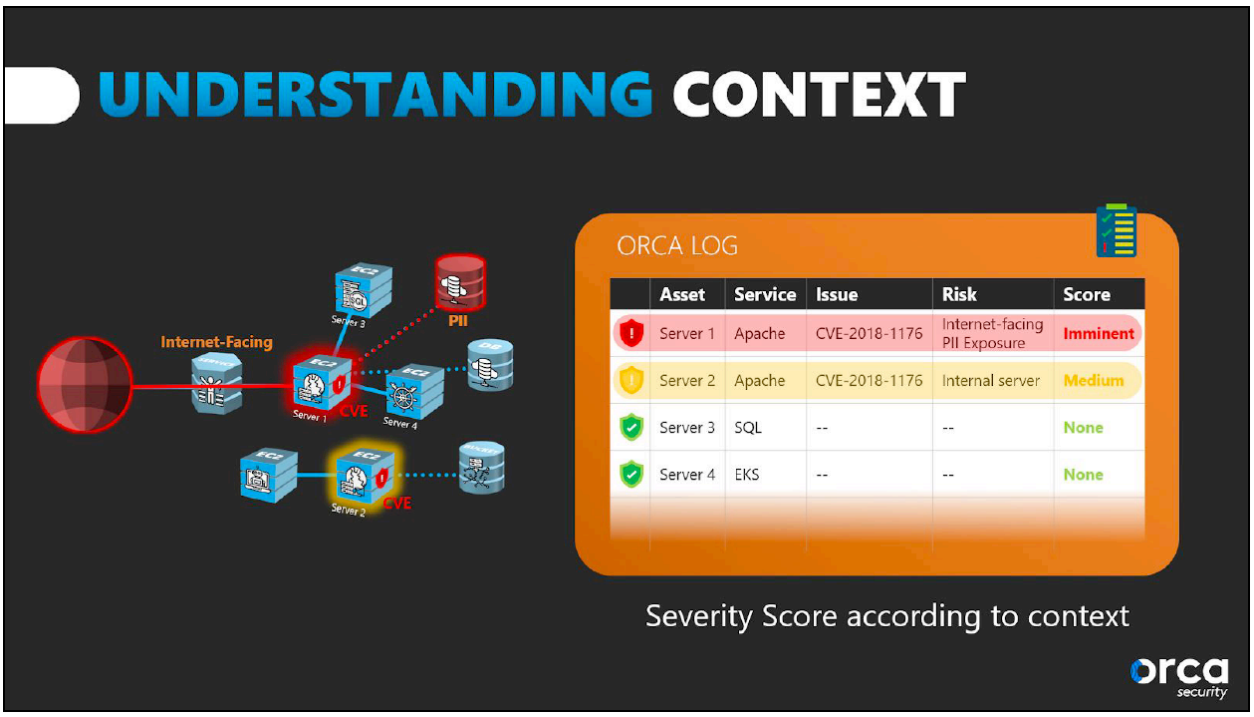
116. **In sum**, Shua in view of Lang teaches *inspecting the AI model* (Lang's analysis of AI models and/or components) *for a cybersecurity risk* (Lang's poisoning

detection, adversarial attack detection, finding privacy violations, detecting vulnerabilities and/or risks, etc.).

6. Claim 1[e]: "generating a representation of the cybersecurity risk in the security database, the representation of the cybersecurity risk connected to the representation of the AI model in response to detecting the cybersecurity risk"

117. It is my opinion that limitation 1[e] is obvious over Shua in view of Lang.

118. First, Shua teaches that representations of cybersecurity risks are added to its map (*security database*) to "contextualize assets and their associated risks." EX1005, *Shua Provisional*, [0036]. For example, Shua states that its system "**combines conclusions** from different environmental perspectives (*e.g.*, metadata) **into a single model. It takes the asset map** from the control plane—where the service meshes and containers talk to each other—**and enriches this information with the risk data** gathered from the workload deep scanning." *Id.*, [0061]. This process of enriching the map can be seen by comparing the original map (*see* §VI.B.4 above) with the updated map:



EX1007, *Orca Security SideScanning Technical Brief*, p. 17, corresponding to the reduced-quality version appearing at EX1005, *Shua Provisional*, p. 25.

119. Comparing these two visualizations of Shua's map, a POSITA would have understood that "CVE" attributes have been added to the nodes representing "Server 1" and "Server 2," indicating the presence of security vulnerabilities for those nodes. A POSITA would have understood that this at least because a POSITA would recognize that "CVE" stands for "Common Vulnerabilities and Exposures," a well-known program to catalog publicly disclosed cybersecurity vulnerabilities, and that the "Issue" annotations in the "Orca Log" are consistent with CVE identifiers. *See* §III.C.1 above.

120. A POSITA at the time of the invention of the ’529 Patent would have also understood that these attributes are each a “representation … in the security database” because they are each an attribute of the corresponding node and are therefore stored as an attribute (*representation*) of the node (affected asset such as Server 1) in the graph database representing Shua’s map (*security database*). This association between the nodes and the attribute representing the vulnerability is visualized just above, in the depicted “Orca Log” showing that “Server 1” is affected by the vulnerability “CVE-2018-1176.”

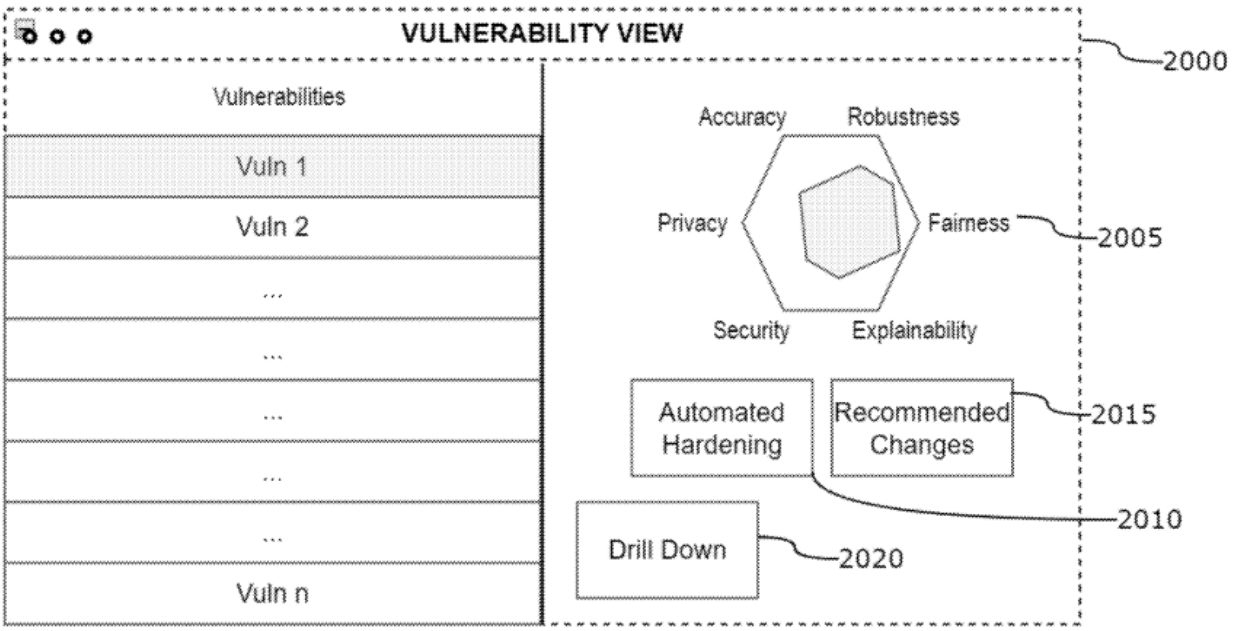
121. Next, Shua teaches that these attributes are added to Shua’s map *in response to detecting the cybersecurity risk*. This is the case at least because Shua’s system “takes the asset map … and enriches this information with **the risk data gathered from the workload deep scanning**.” EX1005, *Shua Provisional*, [0061]. A POSITA at the time of the invention of the ’529 Patent would have understood that this data is stored in the map after (*i.e.*, in response to) detecting the risks in order to present them (as depicted in the figure just above):

> **Orca detects risks** such as vulnerabilities, malware, misconfiguration, and lateral movement risk in the cloud. It sees what other security tools miss. **Following data gathering and analysis, Orca reports all issues in full context**, including where each risk resides and its true level of risk. This lets its customers prioritize their mitigation actions while mitigating cloud risk.

Id., [0070].

122. A POSITA would have further understood that these attributes are additionally added to Shua's map in response to detecting the cybersecurity risk, at least because Shua discloses that a new map is generated "based on" detected changes in security-related state (*e.g.*, trust-related changes) and compared to the previous map. EX1004, *Shua*, 11:3-20. Thus, a POSITA at the time of the invention of the '529 Patent would have understood, that the new map would be created, including adding the vulnerability attribute representations thereto, "responsive to" detecting a new vulnerability.

123. Next, Lang specifically teaches generating a representation of an identified vulnerability and associating identified vulnerabilities for an AI model with that AI model. For example, Lang discloses the user interface in its FIG. 20:



EX1006, *Lang,* FIG. 20.

124. A POSITA at the time of the invention of the ’529 Patent would have understood that the depicted “Vuln 1,” “Vuln 2,” etc., are each a “representation of the cybersecurity risk” at least because they represent a particular vulnerability. For example, Lang states that “Vulnerabilities may be selected on the left side to display deeper analysis results.” A POSITA would have understood this to indicate that the depicted labels correspond to stored representations of the underlying vulnerability insofar as the “deeper analysis results” are accessed via the displayed buttons (*e.g.*, “drill down” unit 2020).

125. Further, a POSITA at the time of the invention of the ’529 Patent would have understood that these representations would be stored “connected to” (*e.g.* linked to) the representation of the model, at least to enable the depicted “automated hardening,” “recommend changes,” and “drill down” functions. For example, if the vulnerabilities were not stored connected to the affected model, it would not be possible to automatically harden the affected AI model against the vulnerability affecting it.

126. Finally, in addition to the motivations discussed in §VI.A above, it is my opinion that it would be obvious to modify Shua’s system to add Lang’s associations of AI models with associated cybersecurity risks in Shua’s map based on Shua’s teaching that the system should “contextualize assets and their associated

risks discovered within the customer cloud account," including AI assets. EX1005, *Shua Provisional*, [0036]. This is the case at least because a POSITA would understand from Shua's teachings that it is desirable to identify all of the risks across the customer's cloud estate. EX1005, *Shua Provisional*, [001]. Additionally, it is a goal of Shua to present a "unified view of [the customer's] assets in order to detect and prioritize vulnerabilities." *Id*., [0116]. A POSITA would have understood that such a unified view would include cybersecurity risks associated with AI models and been motivated to incorporate Lang's teachings.

127. **In sum**, Shua in view of Lang teaches *generating a representation of the cybersecurity risk* (an attribute indicating the vulnerability) *in the security database* (Shua's map)*, the representation of the cybersecurity risk* (the attribute) *connected to the representation of the AI model* (attached to the node representing the affected asset) *in response to detecting the cybersecurity risk* (as a part of updating the map after performing the security assessment).

7. Claim 1[f] "initiating a mitigation action based on the cybersecurity risk"

128. It is my opinion that limitation 1[f] is obvious over Shua in view of Lang.

129. First, Shua teaches mitigating the discovered vulnerabilities (cybersecurity risks):

> Upon gathering network accessibility information and the identified port to identify one or more vulnerabilities susceptible to attack from outside the workload, **a disclosed embodiment would have the functionality to** perform processes to gather, display, and **mitigate a discovered vulnerability in order to minimize the likelihood and effectiveness of a cyber threat** outside of and attempting to access a workload through a known or previously encountered type of cyber threat. **This functionality may include** collecting and organizing the vulnerabilities according to type or category of vulnerability, displaying the gathered data for an end user or maintainer, and **implementing security features automatically** or manually by a user or maintainer such as security patches, password or passcode changes and suggestions for users to do the same, and malicious code eradication.

EX1004, *Shua*, 24:61-25:9; s*ee also id.*, 30:31-33.

130. Similarly, Lang teaches performing mitigation actions based on cybersecurity risks in AI models, specifically:

> Analysis and/or hardening may work closely with one or more monitoring actions. For example, monitoring may include continuously analyzing the AI systems 105 for vulnerabilities. If vulnerabilities are discovered, then a hardening action may be used to harden the system against those vulnerabilities. Monitoring may include continuously assessing logs, inputs, and/or outputs, etc., of the model. It may include reporting on any issues, and/or it may include making mitigations. Mitigations may include, but are not limited to,

> blocking access to specific components of the systems, shutting down entire systems, blocking access by a specific user, blocking specific types of inputs, emailing users, texting users, alert users, and/or trigger a siren, etc.

EX1006, *Lang*, [0122].

131. A POSITA at the time of the invention of the ’529 Patent would have understood that Lang’s “vulnerabilities,” “hardening actions,” and “mitigations” are all specific to cybersecurity risks in AI models. This is the case at least because Lang discusses “continually analyzing the AI systems” immediately before discussing the vulnerabilities,” “hardening actions,” and “mitigations,” and because Lang is generally directed to security analysis of machine learning and AI models. *Id.,* [0003].

132. In addition to the motivations discussed in §VI.A above, it is my opinion that it would be obvious to modify Shua’s system to add Lang’s teaching of mitigating vulnerabilities in AI models into Shua’s more general teachings of mitigating vulnerabilities based on Shua’s teaching that mitigating vulnerabilities is desirable. EX1004, *Shua*, 64:36-43. This is because a POSITA at the time of the invention of the ’529 Patent would have understood that mitigating identified vulnerabilities is generally desirable to improve security in cloud-based systems.

133. **In sum**, Shua in view of Lang teaches *initiating a mitigation action* (Shua's mitigating a discovered vulnerability using Lang's hardening actions) *based on the cybersecurity risk* (Lang's vulnerability in an AI system).

C. Claim 2: "The method of claim 1, further comprising: detecting an artifact of the AI model on the inspectable disk"

134. It is my opinion that claim 2 is obvious over Shua in view of Lang, as well as render obvious claim 1 (*see* §VI.B above).

135. First, the '529 patent explains that an artifact of an AI model is "a file, a software library, a software binary, a folder name, various combinations thereof, and the like, which indicate that a known (*e.g.*, previously determined) AI model is installed on a disk, resource, and the like, which is being inspected." EX1001, *'529 Patent*, 13:24-32. For example, the inspector of the '529 patent is "configured to detect a PyTorch framework." *Id.*, 13:33-34.

136. A POSITA at the time of the invention of the '529 Patent would have understood that PyTorch (given by the '529 Patent as an example of an artifact of an AI model) is a Python script-programming package that provides software libraries suitable for AI models. For example, the web-based source code repository for PyTorch, states "At a granular level, PyTorch is a library…" and "PyTorch is a Python package that provides … deep neural networks…." EX1020, *PyTorch GitHub Page*, p. 4 (last visited on June 2, 2025). A POSITA would have understood that such neural networks are the basis of many "AI models." *See* §III.B above.

From this, a POSITA would have understood that a PyTorch library, when detected as a part of an AI model, is an artifact of the AI model.

137. Lang expressly teaches detecting artifacts of AI models, including the same PyTorch framework given as an example by the '529 Patent:

> A detect always present model functions 1315 module may scan an input computer binary 1305 (potentially received from a detect model headers and or file extensions 1310 module) for other data and/or functions that are always present in known AI formats. **Such scanning may for example (but not limited to) include scanning for known** preprocessing functions that allow the binary to set up, known **importing functions such as library imports that describe known AI libraries such as** SciKit Learn, TensorFlow, **PyTorch**, Caffe, CNTK, MXNet, H20, Core ML, Shogun, etc., and/or other known AI frameworks, and/or otherwise known functions that are always present for known AI frameworks, etc.

EX1006, *Lang*, [0226].

138. Lang further discloses that such artifacts are detected on an *inspectable disk* (*i.e.*, the disk captured from a virtual machine). *Id.*, [0029]-[0030] ("Models may be analyzed in a consistent manner; detecting and/or analyzing AI components in virtual machines and containerized and virtual environments;").

139. In addition to the motivations discussed in §VI.A above, it is my opinion that it would be obvious to modify Shua's system to add Lang's teaching of collecting artifacts of AI models specifically into Shua's process of collecting

software libraries more generally based on Shua's teaching that all software libraries should be collected to perform vulnerability analysis for those packages. *See* EX1005, *Shua Provisional,* [0042]. This is the case at least because Lang teaches that "**there may be vulnerabilities** within the model architectures and/or **in specific AI libraries used**." EX1006*, Lang*, [0308]. A POSITA would be motivated by (1) Shua's teachings that it is desirable to identify all of the risks across the customer's cloud estate and (2) Shua's goal to present a "unified view of [the customer's] assets in order to detect and prioritize vulnerabilities" to identify and present risks of AI models. EX1005, *Shua Provisional*, [001]; [0116]. A POSITA would have understood that Shua's unified view would include cybersecurity risks associated with AI models and would have thus been motivated to incorporate Lang's teachings to better identify such risks.

140. **In sum**, Shua in view of Lang teaches *detecting an artifact of the AI model* (including Lang's detecting of software libraries known to be associated with AI models) *on the inspectable disk* (Shua's snapshot).

D. Claim 3: "The method of claim 1, further comprising: inspecting the AI model to detect an AI model configured to execute a code object"

141. It is my opinion that claim 3 is obvious over Shua in view of Lang, as well as render obvious claim 1 (*see* §VI.B above).

142. First, Lang teaches *inspecting the AI model*. *See* limitation 1[d] (§VI.B.5 above). Lang further teaches that the AI models are inspected for preprocessing or importing functions that allow the AI model to import library code objects for execution:

> A detect always present model functions 1315 module may scan an input computer binary 1305 (potentially received from a detect model headers and or file extensions 1310 module) for other data and/or functions that are always present in known AI formats. **Such scanning may for example (but not limited to) include scanning for known preprocessing functions that allow the binary to set up, known importing functions such as library imports that describe** known AI libraries such as SciKit Learn, TensorFlow, PyTorch, Caffe, CNTK, MXNet, H20, Core ML, Shogun, etc., and/or other known AI frameworks, and/or **otherwise known functions** that are always present for known AI frameworks, etc.

EX1006, *Lang*, [0226].

143. A POSITA at the time of the invention of the ’529 Patent would have understood that software libraries (*e.g.*, for Python) include executable code objects or execute under a script processor, and that software (such as the AI models including the preprocessing or importing functions disclosed by Lang) that imports such software libraries is *configured to execute a code object* embodied in the software library.

144. In addition to the motivations discussed in §VI.A above, it is my opinion that it would be obvious to modify Shua's system to add Lang's teaching of inspecting AI models configured to execute code objects specifically into Shua's process of inspecting cloud assets more generally, based on Shua's teaching that cloud assets should be inspected to perform vulnerability analysis for those packages, and Lang's teaching that "source code analysis" in a part vulnerability analysis for AI models. *See* EX1005, *Shua Provisional,* [0042]; EX1006, *Lang*, [0120]. This is the case at least because Lang teaches that there may be "specific **vulnerabilities**, etc. **in the source code**." EX1006, *Lang*, [0276]. A POSITA would be motivated by (1) Shua's teachings that it is desirable to identify all of the risks across the customer's cloud estate and (2) Shua's goal to present a "unified view of [the customer's] assets in order to detect and prioritize vulnerabilities" to identify and present risks of AI models. EX1005, *Shua Provisional*, [001]; [0116]. A POSITA would have understood that Shua's unified view would include cybersecurity risks associated with AI models and would have thus been motivated to incorporate Lang's teachings to better identify such risks.

145. **In sum**, Shua in view of Lang teaches *inspecting the AI model* (Lang's scanning of binaries with detected AI model headers) *to detect an AI model* (Lang's detected AI model) *configured to execute a code object* (Lang's AI software libraries).

E. Claim 4: "The method of claim 1, further comprising: detecting a metadata of the AI model, wherein the metadata indicates that the AI model is a cybersecurity risk"

146. It is my opinion that claim 4 is obvious over Shua in view of Lang, as well as render obvious claim 1 (*see* §VI.B above).

147. First, Lang expressly teaches extracting metadata of AI models when analyzing AI components from virtual machines:

> FIG. 32 depicts the analysis system for extracting and reversing AI components from containers and/or virtual machines (VM). the [sic] container and/or VM may be ingested by the analysis system and AI components are extracted, such as (but not limited to) analyzing files on the file system, reversing binaries and extracting AI components, detecting installed AI dependencies, analyzing AI dependency versions and **metadata**, etc.

EX1006, *Lang*, [0331].

148. Lang further describes how metadata is used in the analysis of AI models, including functional analysis:

> A **detected AI model functions 1445 module may include functions related to AI**, including for example aggregated functions of all functions from a computer binary related to AI (incl. e.g., AI processes, and/or tasks, etc. etc. **This may include** one or more of binary, disassembled, semantic representations, **metadata,** and/or reports etc., and/ or may be a file such as a file (e.g., JSON, CSV, XML file, binary file, and/or code file etc.), which may be stored on a storage (e.g., physical media) and/or communicated over a network (e.g., via API).

Id., [0241].

149. Finally, Lang describes how metadata, including software library version numbers, may indicate that a particular AI model is a cybersecurity risk:

> The analysis system may include original and permutated code segments that are stored in a source code segments 2820 database. This database may include metadata, such as (but not limited to) the language, version numbers of libraries used, and/or dates, etc. Source code segments may be used in a variety of analyses, for example (but not limited to), they may be used for: vulnerability analysis where each segment is assessed for vulnerabilities; similarity analysis to determine similar code segments that may be vulnerable; and/or testing the overall analysis system etc.

Id., [0302]; *see also id.,* [0308] ("For example, there may be vulnerabilities within the model architectures and/or in specific AI libraries used.").

150. In addition to the motivations discussed in §VI.A above, it is my opinion that it would be obvious to modify Shua's system to add Lang's teaching of inspecting metadata of AI models specifically into Shua's process of inspecting cloud assets more generally based on Shua's teaching that cloud assets should be inspected to perform vulnerability analysis for those packages and Lang's teaching that metadata analysis in a part vulnerability analysis for AI models. *See* EX1005, *Shua Provisional,* [0042]; EX1006, *Lang*, [0302], [0308]. This is the case at least because Lang teaches that metadata may include information such as "version

numbers of libraries." *Id.*, [0302]. Lang further teaches that that "**there may be vulnerabilities** within the model architectures and/or **in specific AI libraries used**." EX1006*, Lang*, [0308]. A POSITA would thus understand that metadata may be indicative of the use of vulnerable AI libraries and be motivated by (1) Shua's teachings that it is desirable to identify all of the risks across the customer's cloud estate and (2) Shua's goal to present a "unified view of [the customer's] assets in order to detect and prioritize vulnerabilities" to identify and present risks of AI models. EX1005, *Shua Provisional*, [001]; [0116]. A POSITA would have understood that Shua's unified view would include cybersecurity risks associated with AI models and been motivated to incorporate Lang's teachings to better identify such risks.

151. **In sum**, Shua in view of Lang teaches *detecting a metadata* (a version number of a software library used) *of the AI model* (Lang's detected AI model)*, wherein the metadata* (software library version number) *indicates that the AI model is a cybersecurity risk* (particular AI libraries indicate that the AI model has a vulnerability).

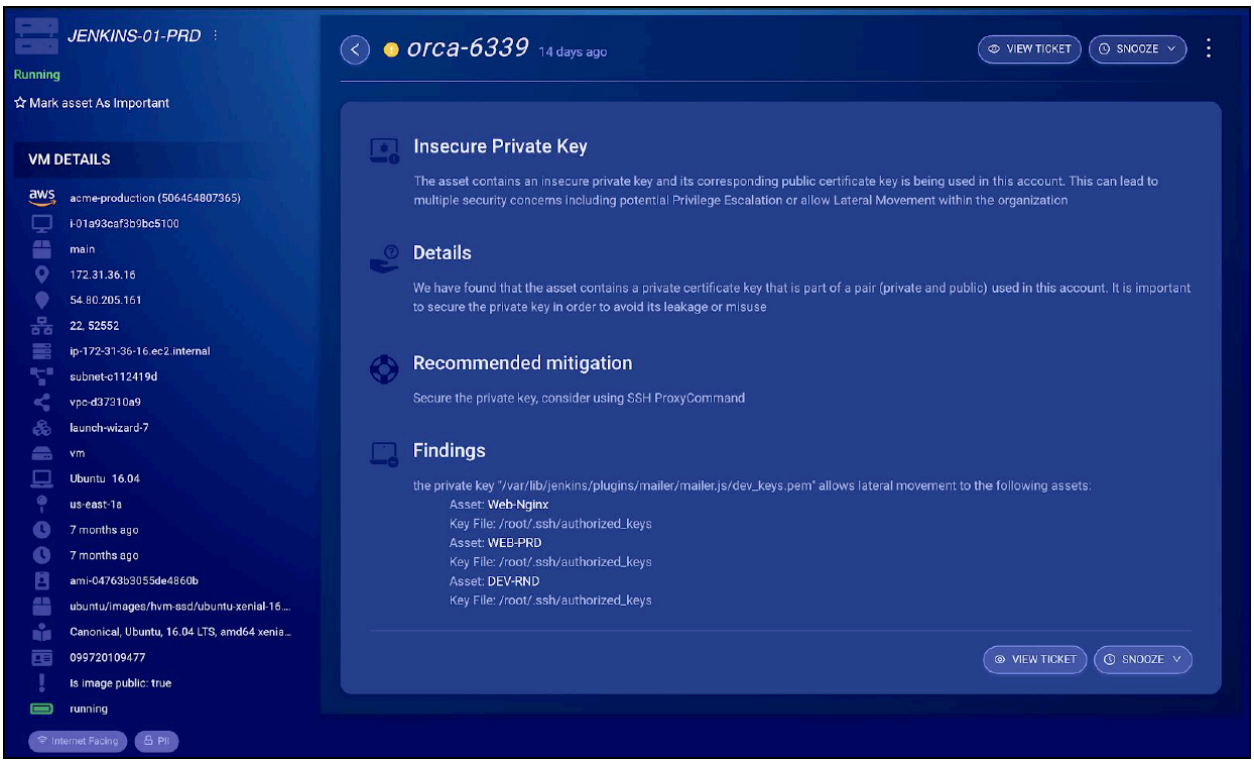
F. Claim 5: "The method of claim 1, further comprising: detecting in the AI model any one of: a secret, a certificate, a code, and any combination thereof"

152. It is my opinion that claim 5 is obvious over Shua in view of Lang, as well as render obvious claim 1 (*see* §VI.B above).

153. First, the ’529 Patent gives, as one example of a “secret,” a credential “utilized by an administrator account to access a storage in a cloud computing environment.” EX1001, *’529 Patent*, 14:62-67. A POSITA at the time of the invention of the ’529 Patent would have understood that such credentials include passwords (allowing a user to log onto an asset) or the private portions of certificates’ cryptographic key pairs (in scenarios where the corresponding public key is included in the authorized keys configuration for an asset). *See, e.g.,* EX1005, *Shua Provisional,* [0085].

154. Shua teaches detecting credentials enabling lateral movement (such as private keys, passwords) in the inspectable disk:

> Orca Security scans your entire cloud account in a holistic manner. **It peers into each machine’s filesystem for private keys**, such as on server A, taking a hash of each found. It then checks the authorized key configuration across all other assets for public keys. If any are found, such as on server B, Orca calculates hashes and compares the private key hash to find a match.



From the image above we see that Orca provides: The path to the insecurely stored key: /var/lib/jenkins/plugins/mailer/mailer.js/ dev_keys.pem. A recommended solution. The identity of the machines the key allows access to e.g., (Web-Nginx, WEBPRD, DEV-RND). The user account on the machine in this case (root user for all machines in this example). **Other lateral movement risks Orca finds are: Passwords in Shell History. Sensitive Info in Git Repository. Sensitive AWS Keys on Systems**.

EX1005, *Shua Provisional,* [0085]-[0086]; EX1007, *Orca Security SideScanning Technical Brief*, p. 13, corresponding to the lower-quality figure version appearing at EX1005, *Shua Provisional*, p. 30).

155. Shua also teaches detecting "certificates" that enable trust between a customer's cloud assets:

> Another embodiment contemplated may comprise a method for identifying relationships 305 between at least some of the identified assets 801, the relationships including at least one of a trust 807A, a network connectivity 807B, or a mechanism of network proxying 807C. A trust 807A may establish a relationship and may involve cryptography, digital signatures, electronic certificates, or similar method of establishing a trusting relationship.

EX1004, *Shua*, 49:36-43.

156. A POSITA at the time of the invention of the '529 Patent would have understood that Shua's disclosure of "identifying relationships," where the relationship is established by a "trust" involving "electronic certificates," teaches the claim limitation of "detecting … a certificate."

157. Similarly, Lang teaches analyzing components of AI (including files used by AI components) in the aid of vulnerability analysis:

> The analysis system may include extracted AI information 3240 that represents output of an of the analysis system depicted in FIG. 32 for analyzing containers and VMs. **This may include, but is not limited to**, AI components, a report of detected AI, **a bill of materials of AI components, a list of directory and file paths to AI components,** reversed AI components, and/or source code segments, etc. **Such output may then be used by the analysis system (or elsewhere), for** example for (but not limited to) source code analysis, **adversarial attack analysis**, and/or bias analysis, etc.

EX1006, *Lang*, [0339].

158. A POSITA at the time of the invention of the ’529 Patent would have understood that the passwords, keys, and certificates that Shua scans for in snapshots (*see* ¶¶154-155 above) would be found in files of AI components. *See*, *e.g.,* EX1005, *Shua Provisional,* [0085] (scanning the file system of a virtual machine for private keys).

159. A POSITA at the time of the invention of the ’529 Patent would have further understood that this sensitive information would be present in the file system of virtual machines because it is common for virtual machines to be given access to an environment without realizing that the environment contains sensitive information. *Id.*, [0053] (“[I]t is not uncommon for an entire production environment repository to be cloned, with no one remembering the copy contains sensitive information.”).

160. Thus, a POSITA at the time of the invention of the ’529 Patent would have understood that the Shua-Lang system, when scanning each machines file system (as taught by Shua) and examining the list of file paths used by AI model (as taught by Lang), would detect the secrets and certificates expressly scanned for by Shua’s system when they are a component used by the AI model, at least because Lang teaches identifying all file paths “to AI components.” EX1006, *Lang*, [0339].

161. In addition to the motivations discussed in §VI.A above, it is my opinion that it would be obvious to modify Shua’s system to add Lang’s teaching of

searching AI model file and directory paths into Shua's process of inspecting each machine's file system for sensitive information at least based on Shua's teaching that "it's essential to know where sensitive information, such as insecurely stored keys, is located across your entire cloud environment." *See* EX1005, *Shua Provisional,* [0084].

162. **In sum**, Shua in view of Lang teaches *detecting in the AI model* (the file system of Lang's detected AI model) *any one of: a secret* (the passwords and certificated scanned for by Shua)*, a certificate* (the electronic certificate detected by Shua)*, a code, and any combination thereof.*

G. Claim 6: "The method of claim 1, further comprising: generating a lateral movement path, wherein the lateral movement path includes the AI model, and a representation of a resource, wherein the resource is accessible utilizing the AI model"

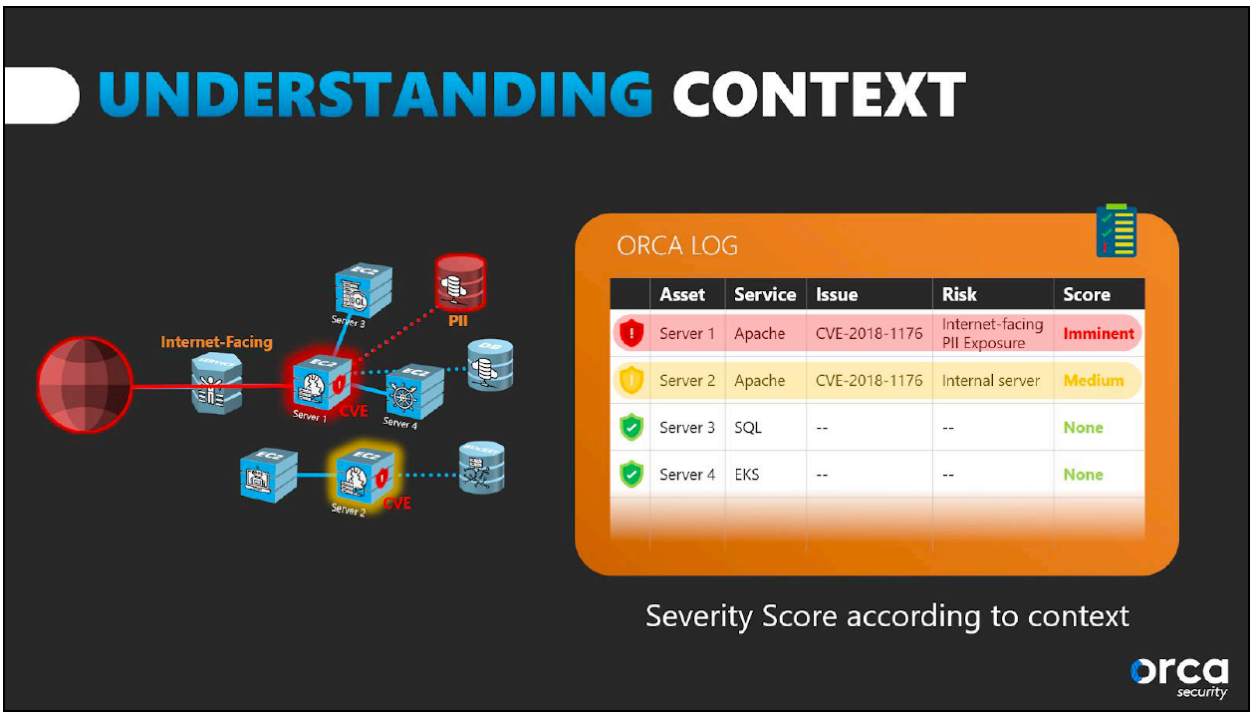
163. It is my opinion that claim 6 is obvious over Shua in view of Lang, as well as render obvious claim 1 (*see* §VI.B above). The '529 Patent does not define "lateral movement," but a POSITA would have understood that the term "lateral movement" refers to an attacker moving from one vulnerable asset inside a cloud computing environment (*e.g.*, one host or VM broken into) to another asset in that cloud computing environment (*e.g.*, another host or VM on the same network). For example, Shua states, "[a]n attacker who establishes a network foothold usually attempts to move laterally from one resource to another in search of rich targets such as valuable data." EX1004, *Shua*, 15:5-8.

164. First, Shua teaches generating a lateral movement path including an asset or service and a representation of a resource accessible utilizing the asset or service, at least via its detection of lateral movement risk and its map representing cloud resources (databases with exposed PII [Personally Identifiable Information]) accessible using other assets and services with connections to risks (*e.g.*, CVEs):

> For example, based on the contextual map, the system may perform a forward analysis of the specific asset under identification to identify at least one possible internet-originating attack vector to the asset. Alternatively or additional, **the system may perform a backward analysis of the specific asset to identify exposure risk to assets downstream of the specific asset, wherein the downstream exposure risk includes an identification of an exposed asset, an entry point to the exposed asset, and lateral movement risks associated with the exposed asset.** Such analysis may be presented to a user via graphical user interface.

EX1005, *Shua Provisional*, [0061].

165. Shua gives an example of such a lateral movement path, as presented by the user:



EX1007, *Orca Security SideScanning Technical Brief*, p. 17, corresponding to the reduced-quality version appearing at EX1005, *Shua Provisional*, p. 25).

166. A POSITA at the time of the invention of the '529 Patent would have understood, this figure to depict (in red) a lateral movement path in a customer network, as generated by Shua's system. The depicted lateral path begins at the public internet, moves via "Server 1" with a CVE, to the exposed database with PII. As discussed in §§VI.B.4 and VI.B.6 above, Server 1 and the exposed database each have a corresponding representation of a resource in Shua's map. As such, Shua teaches generating a lateral movement path (shown in red in the figure just above) including an asset or service (Server 1), and a representation of a resource (the

database with exposed PII) accessible utilizing the asset or service (as shown in the lateral movement path).

167. Next, Lang teaches that it is well-known that AI models include vulnerabilities, and that these vulnerabilities can allow unauthorized access to resources:

> There has been a significant increase in research into the risks and vulnerabilities of AI systems, which include adversarial attacks that attempt to trick the system into revealing proprietary data or affecting its decision-making. These attacks often include small perturbations to input data that are difficult for a human to detect but drastically affect the AI system's decision-making process. An attack can take place before, during or after training, and before, during or after inference. For example, an attacker may introduce malicious data to the training data to introduce a backdoor to the system, reduce the model's accuracy, introduce biases, etc.

EX1006, *Lang*, [0010].

168. As such, a POSITA at the time of the invention of the '529 Patent would have understood that in the Shua-Lang system, the process of lateral path detection would include detecting lateral paths that include AI models. This is the case at least because Shua teaches that the disclosed system "looks for everything an attacker searches for and enumerates that in detailed reports." EX1005, *Shua Provisional*, [0048]; *see also* [0047] (attackers search for lateral movement paths).

169. A POSITA at the time of the invention of the '529 Patent would have further understood that once a lateral movement risk in an AI model is identified in the Shua-Lang system, by scanning the AI model file paths per the teaching of Lang, the step of generating a lateral movement path would proceed per the teaching of Shua even when the lateral movement path includes the AI model. This is the case at least because, again, Shua teaches that the system "looks for **everything** an attacker searches for and enumerates that in detailed reports." EX1005, *Shua Provisional*, [0048].

170. In addition to the motivations discussed in §VI.A above, it is my opinion that it would be obvious to modify Shua's system to add lateral movement paths including AI model vulnerabilities (as identified per Lang's teachings) into Shua's process that generates a master map, putting all vulnerabilities in context (including the lateral movement risk associated with each vulnerability) at least based on Shua's teaching that "[a]ll such context [including vulnerabilities] is merged into the 'master map' discussed earlier, producing a map that includes the relationships between all of the workloads, discrete and containerized alike." *See* EX1005, *Shua Provisional,* [0067].

171. **In sum**, Shua in view of Lang teaches *generating a lateral movement path* (Shua's lateral movement risk)*, wherein the lateral movement path includes the AI model* (Lang's detected AI model)*, and a representation of a resource* (Lang's

proprietary data)*, wherein the resource (proprietary data) is accessible utilizing the AI model* (*e.g.*, because of the vulnerability enables tricking the system into revealing the proprietary data).

H. Claim 7: "The method of claim 1, further comprising: applying a policy on the representation of the computing environment"

172. It is my opinion that claim 7 is obvious over Shua in view of Lang, as well as render obvious claim 1 (*see* §VI.B above).

173. First, the '529 patent explains that a "policy" might be "a rule, a conditional rule, a plurality of each, a combination thereof, and the like, which [is] applied to the representation of the computing environment, the representation of the AI model, a representation of a cybersecurity object, a combination thereof, and the like." EX1001, *'529 Patent*, 15:20-25. The '529 Patent goes on to give, as an example of a policy, "a rule to detect a toxic combination. For example, in an embodiment, a policy includes a rule applied to the representation to detect a finding [*i.e.*, a detected cybersecurity risk] connected to a resource, and an AI model connected to the resource." *Id.*, 17:15-19. Thus, a POSITA would have understood that a "policy" in the language of the '529 Patent includes at least a rule to detect a configuration of resources that includes a vulnerability condition ("the toxic combination").

174. Shua teaches such rules to detect a configuration of resources that includes a vulnerability condition:

> Orca combines conclusions from different environmental perspectives (e.g., metadata) into a single model. It takes the asset map from the control plane-where the service meshes and containers talk to each other-and enriches this information with the risk data gathered from the workload deep scanning. **For example, Orca takes a customer's instance, maps the running services on it, then takes into consideration the vulnerability data collected in the data path. Orca contextualizes the information to establish whether an asset is internet-facing and easily accessible to attackers, or if it's private and hence a less important vulnerability. … [¶] Consider a vulnerability in a web service. Orca scores it as: High risk if it's connected to the internet, either directly or indirectly via a load balancer or reverse proxy;** Medium severity if it's only accessible internally; Low severity if it's blocked by a security group configuration of the cloud provider.

EX1005, *Shua Provisional*, [0061].

175. Shua thus includes a policy that detects a configuration of resources (the web service and the load balancer/reverse proxy) that includes a vulnerability condition (the vulnerability in the web server). Shua teaches applying this policy (rule) on the map (representation of the computing environment) to identify and categorize the configuration as high, medium, or low severity.

176. **In sum**, Shua in view of Lang teaches *applying a policy* (Shua's rule for categorizing risk of a detected publicly accessible server with a vulnerability) *on the representation of the computing environment* (Shua's map).

I. Claim 8: "The method of claim 7, further comprising: initiating the mitigation action based on the policy."

177. It is my opinion that claim 8 is obvious over Shua in view of Lang, as well as render obvious claims 1 (*see* §VI.B above) and 7 (*see* §VI.H above).

178. First, as discussed for claim 7 in §VI.H above, the '529 Patent teaches that a "policy" includes at least a rule to detect a configuration of resources that includes a vulnerability condition (the "toxic combination"). The '529 Patent further specifies that mitigation actions may be triggered by applying a rule. *Id.*, 17:20-23 ("In an embodiment, **a mitigation action is initiated based on a toxic combination**, the detected cybersecurity object, an AI model type, a combination thereof, and the like.").

179. Like the '529 Patent, Shua specifies that policies, when applied, can trigger mitigation actions:

> Upon gathering network accessibility information and the identified port to identify one or more vulnerabilities susceptible to attack from outside the workload, a disclosed embodiment would have the functionality to perform processes to gather, display, and mitigate a discovered vulnerability in order to minimize the likelihood and effectiveness of a cyber threat outside of and attempting to access a workload through a known or previously encountered type of cyber threat.

EX1004, *Shua*, 24:61-25:2; *see also id.*, 31:12-27, 32:25-48.

180. **In sum**, Shua in view of Lang teaches *initiating the mitigation action* (Shua's mitigating a discovered vulnerability) *based on the policy* (Shua's rule categorizing the risk as "high severity" based on the vulnerabilities susceptible to attack from outside the workload).

J. Claim 9

181. It is my opinion that claim 9 is obvious over Shua in view of Lang as set forth next. *See also generally* claim 1 (§VI.B above).

1. Claim 9[P]: "non-transitory computer-readable medium storing a set of instructions for detecting a cybersecurity risk of an artificial intelligence (AI), the set of instructions comprising: one or more instructions that, when executed by one or more processors of a device, cause the device to"

182. To the extent the preamble is limiting, it is my opinion that limitation 9[P] is obvious over Shua in view of Lang. *See* limitation 1[P] (§VI.B.1 above).

183. Additionally, Shua teaches computer-readable media storing instructions to perform Shua's disclosed techniques:

> Embodiments described herein may refer to **a non-transitory computer readable medium containing instructions that when executed by at least one processor, cause the at least one processor to perform a method.** Non-transitory computer readable medium may include any medium capable of storing data in any memory in a way that may be read by any computing device with a processor to carry out methods or any other instructions stored in the memory.

EX1004, *Shua*, 4:48-55.

184. Lang similarly teaches computer-readable media storing instructions to perform Lang's disclosed techniques:

> Further, many examples are described in terms of sequences of actions to be performed by, for example, elements of a computing device. It will be recognized that various actions described herein can be performed by specific circuits (e.g., Application Specific Integrated Circuits (ASICs), Field Programmable Gate Arrays (FPGA), Graphics Processing Units (GPU)), by program instructions being executed by one or more processors, or by a combination of both. Additionally, **these sequences of actions described herein can be considered to be embodied entirely within any form of computer readable storage medium having stored therein a corresponding set of computer instructions that upon execution would cause an associated processor to perform the functionality described herein.**

EX1006, *Lang*, [0076].

2. Claim 9[a]: "generate an inspectable disk based on an original disk of a resource deployed in a computing environment"

185. It is my opinion that limitation 9[b] is obvious over Shua in view of Lang. *See* limitation 1[a], (§VI.B.2 above).

3. Claim 9[b]: "inspect the inspectable disk for an AI model"

186. It is my opinion that limitation 9[b] is obvious over Shua in view of Lang. *See* limitation 1[b] (§VI.B.3 above).

4. Claim 9[c]: "generate a representation of the AI model in a security database, the security database including a representation of the computing environment"

187. It is my opinion that limitation 9[c] is obvious over Shua in view of Lang. *See* limitation 1[c] (§VI.B.4 above).

5. Claim 9[d]: "inspect the AI model for a cybersecurity risk"

188. It is my opinion that limitation 9[d] is obvious over Shua in view of Lang. *See* limitation 1[d] (§VI.B.5 above).

6. Claim 9[e]: "generate a representation of the cybersecurity risk in the security database, the representation of the cyber-security risk connected to the representation of the AI model in response to detecting the cybersecurity risk"

189. It is my opinion that limitation 9[e] is obvious over Shua in view of Lang. *See* limitation 1[e] (§VI.B.6 above).

7. Claim 9[f]: "initiate a mitigation action based on the cybersecurity risk"

190. It is my opinion that limitation 9[f] is obvious over Shua in view of Lang. *See* limitation 1[f] (§VI.B.7 above).

K. Claim 10

191. It is my opinion that claim 10 is obvious over Shua in view of Lang, as set forth next. *See also generally* claim 1 (§VI.B above).

1. Claim 10[P]: "A system for detecting a cybersecurity risk of an artificial intelligence (AI) comprising: a processing circuitry; a memory, the memory containing instructions

that, when executed by the processing circuitry, configure the system to"

192. To the extent the preamble is limiting, it is my opinion that limitation 10[P] is obvious over Shua in view of Lang. *See* limitation 1[P] (§VI.B.1 above).

193. Additionally, Shua discloses a "system" comprising "a memory" and "instructions" executed by "processing circuitry":

> The application program may be uploaded to, and executed by, a machine having any suitable architecture. Preferably, the machine may be implemented on **a computer platform having hardware such as one or more central processing units ("CPUs"), a memory,** and input/output interfaces. **The computer platform may also include** an operating system and **microinstruction code. The various processes and functions described in this disclosure may be either part of the microinstruction code** or part of the application program, or any combination thereof, **which may be executed by a CPU**, whether or not such a computer or processor is explicitly shown.

EX1004, *Shua*, 4:62-5:6.

194. Lang similarly teaches a system including "one or more processors" and "computer readable storage medium having stored therein a corresponding set of computer instructions":

> Further, many examples are described in terms of sequences of actions to be performed by, for example, elements of a computing device. It will be recognized that various actions described herein can be performed by specific circuits (e.g., **Application Specific Integrated**

Circuits (ASICs), Field Programmable Gate Arrays (FPGA), Graphics Processing Units (GPU)), by program instructions being executed by **one or more processors**, or by a combination of both. Additionally, **these sequences of actions described herein can be considered to be embodied entirely within any form of computer readable storage medium having stored therein a corresponding set of computer instructions that upon execution would cause an associated processor to perform the functionality described herein.**

EX1006, *Lang*, [0076].

2. Claim 10[a]: "generate an inspectable disk based on an original disk of a resource deployed in a computing environment"

195. It is my opinion that limitation 10[a] is obvious over Shua in view of Lang. *See* limitation 1[a] (§VI.B.2 above).

3. Claim10[b]: "inspect the inspectable disk for an AI model"

196. It is my opinion that limitation 10[b] is obvious over Shua in view of Lang. *See* limitation 1[b] (§VI.B.3 above).

4. Claim 10[c]: "generate a representation of the AI model in a security database, the security database including a representation of the computing environment"

197. It is my opinion that limitation 10[c] is obvious over Shua in view of Lang. *See* limitation 1[c] (§VI.B.4 above).

5. Claim 10[d]: "inspect the AI model for a cybersecurity risk"

198. It is my opinion that limitation 10[d] is obvious over Shua in view of Lang. *See* limitation 1[d] (§VI.B.5 above).

6. Claim 10[e]: "generate a representation of the cybersecurity risk in the security database, the representation of the cyber-security risk connected to the representation of the AI model in response to detecting the cybersecurity risk"

199. It is my opinion that limitation 10[e] is obvious over Shua in view of Lang. *See* limitation 1[e] (§VI.B.6 above).

7. Claim 10[f]: "initiate a mitigation action based on the cybersecurity risk"

200. It is my opinion that limitation 10[f] is obvious over Shua in view of Lang. *See* limitation 1[f] (§VI.B.7 above).

L. Claim 11: "The system of claim 10, wherein the memory contains further instructions which when executed by the processing circuitry further configure the system to: detect an artifact of the AI model on the inspectable disk"

201. It is my opinion that claim 11 is obvious over Shua in view of Lang. *See* claims 10 (§VI.K above) and 2 (§VI.C above).

M. Claim 12: "The system of claim 10, wherein the memory contains further instructions which when executed by the processing circuitry further configure the system to: inspect the AI model to detect an AI model configured to execute a code object"

202. It is my opinion that claim 12 is obvious over Shua in view of Lang. *See* claims 10 (§VI.K above) and 3 (§VI.D above).

N. Claim 13: "The system of claim 10, wherein the memory contains further instructions which when executed by the processing circuitry further configure the system to: detect a metadata of the AI model, wherein the metadata indicates that the AI model is a cybersecurity risk"

203. It is my opinion that claim 13 is obvious over Shua in view of Lang.

See claims 10 (§VI.K above) and 4 (§VI.E above).

O. Claim 14: "The system of claim 10, wherein the memory contains further instructions which when executed by the processing circuitry further configure the system to: detect in the AI model any one of: a secret, a certificate, a code, and any combination thereof"

204. It is my opinion that claim 14 is obvious over Shua in view of Lang.

See claims 10 (§VI.K above) and 5 (§VI.F above).

P. Claim 15: "The system of claim 10, wherein the memory contains further instructions which when executed by the processing circuitry further configure the system to: generate a lateral movement path, wherein the lateral movement path includes the AI model, and a representation of a resource, wherein the resource is accessible utilizing the AI model"

205. It is my opinion that claim 15 is obvious over Shua in view of Lang.

See claims 10 (§VI.K above) and 6 (§VI.G above).

Q. Claim 16: "The system of claim 10, wherein the memory contains further instructions which when executed by the processing circuitry further configure the system to: apply a policy on the representation of the computing environment"

206. It is my opinion that claim 16 is obvious over Shua in view of Lang.

See claims 10 (§VI.K above) and 7 (§VI.H above).

R. Claim 17: "The system of claim 16, wherein the memory contains further instructions which when executed by the processing circuitry further configure the system to: initiate the mitigation action based on the policy"

207. It is my opinion that claim 17 is obvious over Shua in view of Lang. *See* claims 16 (§VI.K above) and 8 (§VI.I above).

VII. CONCLUSION

208. I declare that all statements made herein of my knowledge are true, and that all statements made on information and belief are believed to be true, and that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

Dated: June 3, 2025 By:

Erez Zadok, PhD

APPENDIX A

EREZ ZADOK, CURRICULUM VITAE

Erez Zadok
Computer Science Department
349 New Computer Science
Stony Brook University
Stony Brook, NY 11794-2424

Phone: +1 631 632 8461 (office)
Fax: +1 631 632 8243
Timezone: US/Eastern
Email: ezk@cs.stonybrook.edu
Web: https://www.cs.sunysb.edu/~ezk

RESEARCH INTERESTS

Operating systems with a special focus on file systems, storage, clouds, big data, hardware/architecture, encryption, security, benchmarking, performance analysis and optimization, energy efficiency, and system administration.

EDUCATION

May 2001 Ph.D., Computer Science, Columbia University, New York, NY,
FiST: A System for Stackable File-System Code Generation.

Sep 1997 M.Phil., Computer Science, Columbia University, New York, NY

Oct 1994 M.S., Computer Science, Columbia University, New York, NY
Discovery and Hot Replacement of Replicated Read-Only File Systems, with Application to Mobile Computing

May 1991 B.S., Computer Science, Columbia University, New York, NY

May 1982 Certified Technician, Electrical Engineering, Holtz College, Israel

PROFESSIONAL EXPERIENCE

Aug 2024–present	Graduate Program Director, Computer Science Department, Stony Brook University
Jul 2024–present	Senator, University Faculty Senate, representative of the Computer Science Department, Stony Brook University
Aug 2017–Jul 2024	Graduate Academic Adviser, Computer Science Department, Stony Brook University
Jan 2016–present	Professor, Computer Science Department, Stony Brook University
Jan 2007–2015	Associate Professor, Computer Science Department, Stony Brook University
Jan 2001–Jan 2007	Assistant Professor, Computer Science Department, Stony Brook University
2013–present	Director, Smart Energy Technologies (SET) Faculty Cluster, Stony Brook University
2025–present	President; Zadok Consulting LI Inc., DBA "Zadoks Consulting Services"
2013–present	Managing Member; Zadoks Consulting, LLC.
2009–2010	Consultant; CTERA Networks, Inc.
2009–2019	Consultant; Packet General Networks, Inc.
1991–2000	Graduate Research Assistant, Computer Science Department, Columbia University
1999–2000	Director of Software Development, HydraWEB Technologies, Inc.
1994–1998	Project Leader, HydraWEB Technologies, Inc.

1990–1998	Consultant, SOS Corporation
1997	Manager of Computing Facilities, Computer Science Department, Columbia University
1991–1998	Technical Staff Member, Computer Science Department, Columbia University
1989–1991	Assistant Lab Manager, Academic Information Systems, Columbia University
1987–1989	Student Consultant, Academic Information Systems, Columbia University
1984–1986	National Army Service, Israeli Air Force, Israel
1982–1984	Programmer, Commodore Israel, Tel-Aviv, Israel
1981–1984	Computer Lab Manager, Holtz College, Tel-Aviv, Israel

PERSONAL

Born December 4, 1964, Tel-Aviv, Israel.

Married, one child.

Citizenships: U.S.A and Israel

Fluent in English and Hebrew

Member: ACM, ACM SIGOPS, IEEE, IEEE Computer Society, USENIX

Affiliate: Storage Systems Research Center (SSRC), Jack Baskin School of Engineering, University of California, Santa Cruz, California.

Member: The I/O Traces, Tools and Analysis (IOTTA) Technical Work Group (TWG), part of the the Storage Networking Industry Association (SNIA).

FUNDING

Oct 2022–present	*CNS Core: Large: Systems and Verifiable Metrics for Sustainable Data Centers*. NSF. $1,500,000 (SBU share $928,403), 4 years. Co-PI with four other SBU faculty, collaborative with Binghamton University and Penn State University.
Oct 2022–present	*Collaborative Research: CyberTraining: Implementation: Medium: FOUNT: Scaffolded, Hands-On Learning for a Data-Centric Future*. NSF. $996,548 (SBU share $174,905), 3 years. Sole PI at SBU. Collaborative with U. Chicago (lead), NYU, Northern Illinois U., and UC San Diego (6 collaborators total).
2022	*CNS Core: Medium: Optimizing Storage Caches via Adaptive and Reconfigurable Tiering*. NSF Research Experiences for Undergraduates (REU) supplement. $16,000. Lead-PI with one other SBU faculty, collaborative with Emory University.
2022	*SCC-IRG Track 1: Smart Aging: Connecting Communities Using Low-Cost and Secure Sensing Technologies*. NSF Research Experiences for Undergraduates (REU) supplement. $24,000. Co-PI with four other SBU faculty.
2022	*CNS Core: Medium: Optimizing Storage Caches via Adaptive and Reconfigurable Tiering*. NSF Research Experiences for Undergraduates (REU) supplement. $16,000. Lead-PI with one other SBU faculty, collaborative with Emory University.

2022	*CNS Core: III: Medium: Collaborative Research: Optimizing and Understanding Large Parameter Spaces in Storage Systems*. NSF Research Experiences for Undergraduates (REU) supplement. $16,000. Lead PI with one other.
Oct 2021–present	*CNS Core: Medium: Optimizing Storage Caches via Adaptive and Reconfigurable Tiering*. NSF. $800,000 (SBU share $533,333), 3 years. Lead-PI with one other SBU faculty, collaborative with Emory University.
Oct 2021–present	*CNS Core: Medium: Secure, Reliable, and Efficient Long-Term Storage*. NSF. $1,198,842 (SBU share $717,303), 4 years. Lead-PI with one other SBU faculty, collaborative with UC Santa Cruz.
2021–2022	*Kubernetes/Containerizations on GPU Clusters for AI*. SUNY-IBM AI Collaborative Research Alliance. $100,000. Co-PI with two others.
2020–2021	*Automated Cross-Validation of TLS 1.3 Implementations*. Facebook Faculty Research Award. $50,000. Lead PI with two others.
Oct 2020–present	*SCC-IRG Track 1: Smart Aging: Connecting Communities Using Low-Cost and Secure Sensing Technologies*. NSF Smart and Connected Communities. $1,700,126, 4 years. Co-PI with four other SBU faculty.
Sep 2020–present	*Recruitment and Retention of Women Undergraduates*. Center for Inclusive Computing Diversity Initiative, by Northeastern University and Pivotal Ventures. $295,742, 4 years. Co-PI with four other SBU faculty.
2020	*FMitF: Track I: NLP-Assisted Formal Verification of the NFS Distributed File System Protocol*. NSF Research Experiences for Undergraduates (REU) supplement. $16,000. Lead PI with two others.
2020	*CNS Core: III: Medium: Collaborative Research: Optimizing and Understanding Large Parameter Spaces in Storage Systems*. NSF Research Experiences for Undergraduates (REU) supplement. $8,000. Lead PI with one other.
Oct 2019–present	*CNS Core: III: Medium: Collaborative Research: Optimizing and Understanding Large Parameter Spaces in Storage Systems* NSF. $1,088,017 (SBU share $823,142), 4 years. Lead-PI with one other SBU faculty, collaborative with Harvey Mudd College.
Oct 2019–present	*FMitF: Track I: NLP-Assisted Formal Verification of the NFS Distributed File System Protocol* NSF. $748,300, 3 years. Lead-PI with two other SBU faculty.
2019	*CI-SUSTAIN: National File System Trace Repository*. NSF Research Experiences for Undergraduates (REU) supplement. $8,000. Sole PI.
2019	*Study of a Novel Non-Wearable Respiration and Heart Rate Sensor in Cardiopulmonary Exercise Testing*. Stony Brook College of Engineering and Applied Sciences, SEED grant. $15,000, Co-PI with five others.
2019	*Realizing the Full Performance and Parallelization Potential of Modern Storage Architectures for Big Data Applications*. Stony Brook Research Foundation, SEED grant. $60,000, Co-PI with Anshul Gandhi.
2018	*Storage/Deduplication research*. Dell-EMC Corporation. $25,000, Single PI.

Jun 2017–present	*CI-SUSTAIN: National File System Trace Repository*. NSF. $129,867 (SBU share), 3 years. Co-PI with lead institution Harvey Mudd College.
2017–2022	*I/UCRC Phase II: Center for Visual and Decision Informatics (CVDI) Site at SUNY Stony Brook*. NSF. $400,000, 4 years. Co-PI with A. Kaufman, K. Mueller, H. Schwartz, and D. Samaras.
2017	Dell-EMC Corporation. *Storage/Deduplication research*, $25,000, Single PI.
2016–2021	*NRT-DESE: Interdisciplinary Graduate Training to Understand and Inform Decision Processes Using Advanced Spatial Data Analysis and Visualization (STRIDE)*. NSF. $2,993,930, 5 years. Senior personnel.
2016–2019	*Early Detection of User-impersonating Attackers using Multilayer Tripwires*, U.S. Office of Naval Research (ONR). $586,215, 3 years. Co-PI with Nick Nikiforakis.
Jun 2016–2019	*EAGER: Elastic Multi-layer Memcached Tiers* NSF. $257,165, 2 years. Co-PI with Anshul Gandhi.
2016	EMC Corporation. *Storage/Deduplication research*, $25,000, Single PI.
2015	EMC Corporation. *Storage/Deduplication research*, $25,000, Single PI.
Feb 2015	*Student Travel Support for the 13^{th} USENIX Conference on File and Storage Technologies (FAST 2015)*. NSF. $20,000, 1 year. Sole PI.
Sep 2014–2017	*Adaptive Runtime Verification and Recovery for Mission-Critical Software*. U.S. Air Force Office of Scientific Research (AFOSR). $620,861, 3 years. Co-PI with Scott A. Smolka and Scott D. Stoller. (Collaboration with NASA JPL.)
Jan 2014–2016	*Smarter Electric Grid Research, Innovation, Development, Demonstration, Deployment Center (SGRID3)*. Brookhaven Science Associates LLC (BNL), $236,397, 1 year. Lead-PI with one other Stony Brook Co-PI.
2014	EMC Corporation. *Storage/Deduplication research*, $25,000, Sole PI.
Jun 2013–2017	*CSR: Medium: Collaborative Research: Workload-Aware Storage Architectures for Optimal Performance and Energy Efficiency*. NSF. $513,900 (SBU share, total budget $1,000,000), 3 years. Lead-PI with one other Stony Brook Co-PI, and two more institutions (Harvard U. and Harvey Mudd College).
Jan 2013–2017	*BIGDATA: Small: DCM: Collaborative Research: An efficient, versatile, scalable, and portable storage system for scientific data containers*. NSF. $444,267 (SBU share, total budget $746,290), 3 years. Lead-PI with two other Stony Brook Co-PIs, and two more institutions (Brandeis U. and Louisiana State U.).
Sep 2013–2017	*CRI-CI-ADDO-EN: National File System Trace Repository*. NSF. $37,018 (SBU share, total budget $167,817), 3 years. Co-PI with lead institution Harvey Mudd College.
2013	Western Digital Research award. *Shingled Magnetic Recording Disks Benchmarking*, $50,000, Single PI.
Sep 2012 – 2016	*NFS4Sec: An Extensible Security Layer for Network Storage*. NSF. $486,783, 3 years. Lead-PI with one other Co-PI.

2012-2013	*Server-Class Performance vs. Energy Optimizations*. Government of Israel (GoI), Mission to the USA. $47,152, 1 year. Lead PI with one other Co-PI.
2011	NetApp Research award. *Dedup Workload Modeling, Synthetic Datasets, and Scalable Benchmarking*, $40,000, Single PI.
2010	NetApp Research award. *A Study of Network Storage Benefits using FLASH Hardware with Indexing Workloads*, $40,000, Single PI.
Nov 2010 – 2016	*Long Island Smart Energy Corridor*. Department of Energy (DOE), LIPA, and New York State. Collaboration between Stony Brook University, SUNY Farmingdale, and LIPA. $2,822,638, Co-PI.
Sep 2009 – Aug 2013	*Collaborative Research: Performance- and Energy-Aware HEC Storage Stacks*. NSF. $652,000, 3 years. Co-PI with Geoff Kuenning (Harvey Mudd College)
Sep 2009 – Aug 2013	*Collaborative Research: Secure Provenance in High End Computing Systems*. NSF. $564,972, 3 years. Co-PI with Radu Sion. Collaborative project with Patrick McDaniel (Penn State U.) and Marianne Winslett (UIUC).
Apr 2009 – Nov 2012	*Survivable Software*. U.S. Air Force Office of Scientific Research (AFOSR). $881,691, 39 months. Co-PI with Scott A. Smolka, Radu Grosu, Scott D. Stoller, and Klaus Havelund (NASA JPL).
Feb 2010	*Student Travel Support for the First USENIX Workshop on Sustainable Information Technology (SustainIT 2010)*. NSF. $10,000, 1 year. Lead PI.
2009	Network Appliance Research award. *Power use in Storage Servers.* $30,000 Single PI.
2008	*The Impact of Storage Software and Aging on Power Consumption*, **IBM Faculty award** (IBM T.J. Watson Labs). $20,000, one year. Single PI.
2008	Network Appliance Equipment gift. *A Study of User File Access Patterns.* $91,083 Single PI.
Sep 2007 – 2015	*Center for Information Protection: A Multi-University Industry/University Collaborative Research Center*. NSF. $250,147, 5 years. Co-PI with R. Sekar (PI), Tzi-Cker Chiueh, Scott Stoller, and Radu Sion.
Sep 2006 – Aug 2009	*CT-ISG: N3S: Networked Secure Searchable Storage with Privacy and Correctness Assurances*. NSF. $300,000, 3 years. Co-PI with Radu Sion.
Aug 2006 – Aug 2010	*File System Tracing, Replaying, Profiling, and Analysis on HEC Systems*. NSF. $760,252, 3 years. Lead PI with Klaus Mueller (Stony Brook) and Ethan Miller (UC Santa Cruz).
Jul 2006	*End-to-End File Server Security*, **IBM Faculty award** (IBM Haifa Research Labs). $20,000, one year. Single PI.
Jun 2006 – Aug 2010	*CSR—PDOS: Support for Atomic Sequences of File System Operations*. NSF. $561,727, 3 years. Lead PI with Margo Seltzer (Harvard University).
Jan 2006 – Dec 2006	*Secure File Systems*, NY State "Millennium" award, $204,528, one year. Co-PI with R. Sekar (PI), Tzi-Cker Chiueh, CR Ramakrishnan, Radu Sion, and Scott D. Stoller.

Jul 2005 – Aug 2010	*CSR—AES: Runtime-Monitoring and Model Checking for High-Confidence Systems Software*. NSF. $830,000, 4 years. Lead PI with Radu Grosu, Y. Annie Liu, Scott Smolka, and Scott D. Stoller.
Sep 2005 – Aug 2004	*I/UCRC: A Plan for Developing a Multi-University Industry/University Collaborative Research Center on Cyber Security*. NSF. $9,987, one year. Co-PI with R. Sekar, Radu Sion Scott D. Stoller, and Tzi-Cker Chiueh.
Sep 2004 – Aug 2009	*Federal Cyber Service: Scholarship for Service (SFS)*. NSF. $2,459,061, 4 years. Co-PI with R. Sekar, Scott D. Stoller, I. V. Ramakrishnan, and Tzi-Cker Chiueh.
Sep 2003 – Aug 2008	*A Layered Approach to Securing Network File Systems*. NSF Trusted Computing Program (TC). $400,000, 3 years. Single PI.
Sep 2003 – Aug 2005	*Collaborative Research: Capacity Expansion in Information Assurance*. NSF Collaborative Research Proposal (CAP). $199,883, 2 years. Co-PI with R. Sekar, Scott D. Stoller, and I. V. Ramakrishnan.
Sep 2002 – Aug 2007	**CAREER**: *An In-Kernel Runtime Execution Environment for User-Level Programs.* NSF Next Generation Software Program (NGS). $400,000, 5 years. Single PI.
Jan 2003 – Dec 2003	*Assessing the Technological Basis for Enterprise Protection*, NIJ (CyberScience Lab) CSL. $90,563, one year. Co-PI with R. Sekar, Tzi-cker Chiueh, and Scott D. Stoller.
Sep 2002 – May 2003	New York State Strategic Partnership for Industrial Resurgence (SPIR), with Packet General Networks. *S3: Secure Shared Storage.* $94,581. Single PI.
May 2002 – Aug 2002	New York State Strategic Partnership for Industrial Resurgence (SPIR), with Packet General Networks. *A Secure and Scalable Network Appliance.* $55,676. Single PI.
Apr 2003	Microsoft Tablet PC Seed Award. $4,169. Single PI.
2002	HP/Intel IA-64/IPF Second Generation Equipment gift. *Linux Application Performance and File System Security.* $131,529. Single PI.
2001	HP/Intel IA-64/IPF Equipment gift. *Linux Network Scalability and File System Reliability.* $22,490. Single PI.
2001	Red Hat University software award.

AWARDS AND HONORS

2023	Best paper award. A. Merenstein, V. Tarasov, A. Anwar, S. Guthridge, and E. Zadok. F3: Serving Files Efficiently in Serverless Computing. In Proceedings of the 16th ACM International Systems and Storage Conference (SYSTOR '23).
2023	Elected as IEEE Senior Member
2021–2022	Chancellor's Award for Excellence in Scholarship and Creative Activities, State University of New York (SUNY).
2022	Provost's Excellent Mentor award, Stony Brook University.

2021 Elected as ACM Distinguished Member for "Outstanding Scientific Contributions to Computing."

2021 Dean's "Millionaire" award (given lead PIs who secure more than one-million dollars of new funding in a given year), College of Engineering and Applied Sciences, Stony Brook University.

2020 Inducted as member of the Stony Brook University Chapter of the National Academy of Inventors (NAI).

2020 Best paper award. M. Wajahat, A. Yele, T. Estro, A. Gandhi, and E. Zadok. Distribution fitting and performance modeling for storage traces. In 27th IEEE International Symposium on the Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS 2019).

2019 Dean's "Millionaire" award (given lead PIs who secure more than one-million dollars of new funding in a given year), College of Engineering and Applied Sciences, Stony Brook University.

Sep 2011 Best Paper Award. S. D. Stoller and E. Bartocci and J. Seyster and R. Grosu and K. Havelund and S. A. Smolka and E. Zadok. Runtime verification with state estimation. In proceedings of the 2nd International Conference on Runtime Verification (RV'11).

Dec 2009 LISTnet's "Top 20 techies of Long Island" award.

Aug 2008 Service Award (for 2006–2008), Stony Brook University, Computer Science Department

2008 IBM Faculty award (IBM T.J. Watson Labs)

2007–2008 Chancellor's Award for Excellence in Teaching, State University of New York (SUNY).

2007–2008 President's Award for Excellence in Teaching, Stony Brook University.

Aug 2006 Research Excellence Award (for 2005–2006), Stony Brook University, Computer Science Department

Jul 2006 IBM Faculty award (IBM Haifa Research Labs)

Nov 2005 Best Short Paper Award. N. Joukov, A. Kashyap, G. Sivathanu, E. Zadok. Kefence: An Electric Fence for Kernel Buffers. In proceedings of the first ACM International Workshop on Storage Security and Survivability (StorageSS 2005), "The Paradigm Shift to Info-Centric Protection," held in conjunction with the 12th ACM Conference on Computer and Communications Security (CCS 2005).

May 2005 Best Paper Award. N. Joukov, A. Rai, and E. Zadok. Increasing Distributed Storage Survivability with a Stackable RAID-like File System. In proceedings of the 2005 IEEE/ACM Workshop on Cluster Security, in conjunction with the Fifth IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid).

Aug 2004 Graduate Teaching and Research Award (for 2003–2004), Stony Brook University, Computer Science Department

May 2003 Promising Inventor Award, The State University of New York, Research Foundation

Sep 2002 NSF CAREER award, National Science Foundation

Jun 2001 Best Student Paper. M. G. Schultz, E. Eskin, E. Zadok, M. Bhattacharyya, and S. J. Stolfo. MEF: Malicious Email Filter — A UNIX Mail Filter that Detects Malicious Windows Executables. In *Proceedings of the Annual USENIX Technical Conference, FreeNIX Track*, pages 245–252, JUNE 2001.

Jun 2000 Usenix Student Stipend award

Jun 1999 Usenix Student Stipend award

1991 Dean's List

1991 NCR Best Undergraduate Research Award

1989–1990 Gussman Scholar

1988–1989 National Dean's List

1988–1989 Scheuer Foundation Presidential Scholar

1988 Dean's List

PUBLICATIONS

Books

E. Zadok. *Linux NFS and Automounter Administration*. Sybex, Inc., May 2001.

Journal and Magazine Articles

Mário Antunes, Tyler Estro, Pranav Bhandari, Anshul Gandhi, Geoff Kuenning, Yifei Liu, Carl Waldspurger, Avani Wildani, and Erez Zadok. Kneeliverse: A universal knee-detection library for performance curves. *SoftwareX*, 30:102161, 2025.

Tyler Estro, Mário Antunes, Pranav Bhandari, Anshul Gandhi, Geoff Kuenning, Yifei Liu, Carl Waldspurger, Avani Wildani, and Erez Zadok. Accelerating multi-tier storage cache simulations using knee detection. *Performance Evaluation*, 164:102410, May 2024.

S. Hussain, P. McDaniel, A. Gandhi, K. Ghose, K. Gopalan, D. Lee, Y. Liu, Z. Liu, S. Mu, and E. Zadok. Verifiable sustainability in data centers. *IEEE Security & Privacy*, (01):2–15, March 2024.

Anshul Gandhi, Dongyoon Lee, Zhenhua Liu, Shuai Mu, Erez Zadok, Kanad Ghose, Kartik Gopalan, Yu David Liu, Syed Rafiul Hussain, and Patrick Mcdaniel. Metrics for sustainability in data centers. *SIGENERGY Energy Inform. Rev.*, 3(3):40–46, oct 2023.

Ibrahim "Umit" Akgun, Ali Selman Aydin, Andrew Burford, Michael McNeill, Michael Arkhangelskiy, and Erez Zadok. Improving storage systems using machine learning. *ACM Transactions on Storage (TOS)*, 19(1):1–30, Jan 2023.

Erez Zadok and Ada Gavrilovska. Running virtual pc (cpc) meetings. *;login: The USENIX Magazine*, 45(2):54–56, July 2020.

Muhammad Wajahat, Aditya Yele, Tyler Estro, Anshul Gandhi, and Erez Zadok. Analyzing the distribution fit for storage workload and internet traffic traces. *Performance Evaluation*, pages 102–121, 2020.

George Amvrosiadis, Ali R. Butt, Vasily Tarasov, Erez Zadok, and Ming Zhao. Selected results of the workshop on data storage research 2025. *;login: The USENIX Magazine*, 44(4):24–28, December 2019.

Bharath Kumar Reddy Vangoor, Prafful Agarwal, Manu Mathew, Arun Ramachandran, Swaminathan Sivaraman, Vasily Tarasov, and Erez Zadok. Performance and resource utilization of FUSE user-space file systems. *ACM Transactions on Storage (TOS)*, 15(2), May 2019.

Zhen "Jason" Sun, Geoff Kuenning, Sonam Mandal, Philip Shilane, Vasily Tarasov, Nong Xiao, and Erez Zadok. Cluster and single-node analysis of long-term deduplication patterns. *ACM Transactions on Storage (TOS)*, 14(2), May 2018.

Ming Chen, Geetika Bangera, Dean Hildebrand Farhaan Jalia, Geoff Kuenning, Henry Nelson, and Erez Zadok. vNFS: Maximizing NFS performance with compounds and vectorized I/O. *ACM Transactions on Storage (TOS)*, 13(7), September 2017.

Vasily Tarasov, Erez Zadok, and Spencer Shepler. Filebench: A flexible framework for file system benchmarking. *;login: The USENIX Magazine*, 41(1):6–12, March 2016.

Z. Li, M. Chen, A. Mukker, and E. Zadok. On the trade-offs among performance, energy, and endurance in a versatile hybrid drive. *ACM Transactions on Storage (TOS)*, 11(3), July 2015.

Ming Chen, Dean Hildebrand, Geoff Kuenning, Soujanya Shankaranarayana, Bharat Singh, and Erez Zadok. Is NFSv4.1 ready for prime time? *;login: The USENIX Magazine*, 40(3):6–12, June 2015.

Zhiyuan Zhang, Kevin T. McDonnell, Erez Zadok, and Klaus Mueller. Visual correlation analysis of numerical and categorical data on the correlation map. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 21(2), February 2015.

Michael A. Bender, Martin Farach-Colton, Rob Johnson, Russell Kraner, Bradley C. Kuszmaul, Dzejla Medjedovic, Pablo Montes, Pradeep Shetty, Richard P. Spillane, and Erez Zadok. Don't thrash: How to cache your hash on flash. 5(11):1627–1637, August 2012.

J. Seyster, K. Dixit, X. Huang, R. Grosu, K. Havelund, S. A. Smolka, S. D. Stoller, and E. Zadok. InterAspect: Aspect-oriented instrumentation with GCC. *Formal Methods in System Design*, August 2012.

X. Huang, J. Seyster, S. Callanan, K. Dixit, R. Grosu, S. A. Smolka, S. D. Stoller, and E. Zadok. Software monitoring with controllable overhead. *International Journal on Software Tools for Technology Transfer (STTT)*, 14(3):327–347, 2012.

Priya Sehgal, Vasily Tarasov, and Erez Zadok. Optimizing energy and performance for server-class file system workloads. *ACM Transactions on Storage (TOS)*, 6(3), September 2010.

E. Zadok, V. Tarasov, and P. Sehgal. The case for specialized file systems, or, fighting file system obesity. *;login: The USENIX Magazine*, 35(1):38–40, February 2010.

M. T. Dougherty, M. J. Folk, E. Zadok, H. J. Bernstein, F. C. Bernstein, K. W. Eliceiri, W. Benger, and C. Best. Unifying biological image formats with hdf5. *Communications of the ACM (CACM)*, 52(10):42–47, October 2009.

A. Traeger and E. Zadok. Notes on a nine year study of file system and storage benchmarking. *Byte and Switch*, July 2009. www.byteandswitch.com/storage/storage-management/notes-on-a-nine-year-study-of-file-system-and-storage-benchmarking.php.

A. Traeger, E. Zadok, E. L. Miller, and D. D. E. Long. Findings from the first annual storage and file systems benchmarking workshop. *;login: The USENIX Magazine*, 33(5):113–117, October 2008.

Avishay Traeger, Erez Zadok, Nikolai Joukov, and Charles P. Wright. A nine year study of file system and storage benchmarking. *ACM Transactions on Storage (TOS)*, 4(2):25–80, May 2008.

C. P. Wright, R. Spillane, G. Sivathanu, and E. Zadok. Extending ACID semantics to the file system. *ACM Transactions on Storage (TOS)*, 3(2):1–42, June 2007.

Erez Zadok, Rakesh Iyer, Nikolai Joukov, Gopalan Sivathanu, and Charles P. Wright. On incremental file system development. *ACM Transactions on Storage (TOS)*, 2(2):161–196, 2006.

C. P. Wright, J. Dave, P. Gupta, H. Krishnan, D. P. Quigley, E. Zadok, and M. N. Zubair. Versatility and unix semantics in namespace unification. *ACM Transactions on Storage (TOS)*, 2(1):1–32, February 2006.

C. P. Wright and E. Zadok. Unionfs: Bringing file systems together. *Linux Journal*, 2004(128):24–29, December 2004.

E. Zadok. Using the Amd automounter. *Linux Journal*, 10(114):52–55, October 2003.

E. Zadok. Writing stackable file systems. *Linux Journal*, 05(109):22–25, May 2003.

W. Lee, W. Fan, M. Miller, S. Stolfo, and E. Zadok. Toward cost-sensitive modeling for intrusion detection and response. *Journal of Computer Security*, 10(1–2):5–22, January 2002.

Refereed Conference and Workshop Papers

Dongsheng Luo, Raju Rangaswami, Amir Rahmati, and Erez Zadok. GitTemporalAI: Leveraging temporal knowledge graphs and LLMs for multi-agent repository intelligence. 2025. The First MARW: Multi-Agent AI in the Real World Workshop at AAAI 2025.

Mohammad Saqib Hasan, Sayontan Ghosh, Dhruv Verma, Geoff Kuenning, Erez Zadok, Scott Smolka, and Niranjan Balasubramanian. Handling open-vocabulary constructs in formalizing specifications: Retrieval augmented parsing with expert knowledge. In *Proceedings of the First Conference on Language Modeling (COLM 2024)*, Philadelphia, PA, October 2024.

Christopher Smith, Maliha Tabassum, Soumya Chowdary Daruru, Gaurav Kulhare, Arvin Wang, Ethan Miller, and Erez Zadok. Secure archival is hard... really hard. In *Proceedings of the 16th ACM Workshop on Hot Topics in Storage and File Systems (HotStorage '24)*, Santa Clara, CA, July 2024. ACM.

Alex Merenstein, Xinran Wang, Vasily Tarasov, Prajjawal Agarwal, Scott Guthridge, Kapil Thakkar, Katherine Wu, Ali Anwar, and Erez Zadok. Balancing costs and durability for serverless data. In *Proceedings of the 38th International IEEE Symposium on Mass Storage Systems and Technologies (MSST '24)*, Santa Clara, California, June 2024. IEEE.

Debajyoti Halder, Manas Acharya, Aniket Malsane, Anshul Gandhi, and Erez Zadok. Empirical evaluation of ML models for per-job power prediction. In *Proceedings of the 7th ICPE Workshop on Hot Topics in Cloud Computing Performance (HotCloudPerf 2024)*, pages 181–188, London, UK, May 2024. ACM.

Yifei Liu, Manish Adkar, Gerard Holzmann, Geoff Kuenning, Pei Liu, Scott Smolka, Wei Su, and Erez Zadok. Metis: File system model checking via versatile input and state exploration. In *Proceedings of the 22nd USENIX Conference on File and Storage Technologies (FAST '24)*, pages 123–140, Santa Clara, CA, February 2024. USENIX Association. Received all 3 Artifact-Evaluation badges.

Tyler Estro, Mário Antunes, Pranav Bhandari, Anshul Gandhi, Geoff Kuenning, Yifei Liu, Carl Waldspurger, Avani Wildani, and Erez Zadok. Guiding simulations of multi-tier storage caches using knee detection. In *31st Annual International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS '23)*, Stony Brook, NY, October 2023. IEEE Computer Society.

Peter Desnoyers, Andy Rudoff, Ian Adams, Tyler Estro, Anshul Gandhi, Geoff Kuenning, Mike Mesnier, Carl Waldspurger, Avani Wildani, and Erez Zadok. Persistent memory research in the

post-optane era. In *1st Workshop on Disruptive Memory Systems (DIMES 2023)*, October 2023. https://dl.acm.org/doi/10.1145/3609308.3625268.

Yifei Liu, Gautam Ahuja, Geoff Kuenning, Scott Smolka, and Erez Zadok. Input and output coverage needed in file system testing. In *Proceedings of the 15th ACM Workshop on Hot Topics in Storage and File Systems (HotStorage '23)*, Boston, MA, July 2023. ACM.

Alex Merenstein, Vasily Tarasov, Ali Anwar, Scott Guthridge, and Erez Zadok. F3: Serving files efficiently in serverless computing. In *Proceedings of the 16th ACM International Systems and Storage Conference (SYSTOR '23)*, Haifa, Israel, June 2023. ACM. Won Best Paper Award.

Ibrahim "Umit" Akgun, Santiago Vargas, Michael Arkhangelskiy, Andrew Burford, Michael McNeill, Aruna Balasubramanian, Anshul Gandhi, and Erez Zadok. Predicting network buffer capacity for BBR fairness. In *36th Conference on Neural Information Processing Systems (NeurIPS 2022) Workshop on ML for Systems*, December 2022.

Anjul Tyagi, Tyler Estro, Geoff Kuenning, Erez Zadok, and Klaus Mueller. PC-Expo: A metrics-based interactive axes reordering method for parallel coordinate displays. *IEEE Transactions on Visualization and Computer Graphics*, October 2022.

Anshul Gandhi, Kanad Ghose, Kartik Gopalan, Syed Hussain, Dongyoon Lee, David Liu, Zhenhua Liu, Patrick McDaniel, Shuai Mu, and Erez Zadok. Metrics for sustainability in data centers. In *Proceedings of the 1st Workshop on Sustainable Computer Systems Design and Implementation (HotCarbon'22)*, San Diego, CA, USA, July 2022. USENIX.

Sayontan Ghosh, Amanpreet Singh, Meyer "Alex" Merenstein, Wei Su, Scott A. Smolka, Erez Zadok, and Niranjan Balasubramanian. SpecNFS: A challenge dataset towards extracting formal models from natural language specifications. In *Proceedings of the 13th Language Resources and Evaluation Conference (LREC'22)*, pages 2166–2176, Marseille, France, June 2022.

Wei Su, Yifei Liu, Gomathi Ganesan, Gerard Holzmann, Scott Smolka, Erez Zadok, and Geoff Kuenning. Model-checking support for file system development. In *Proceedings of the 13th ACM Workshop on Hot Topics in Storage and File Systems (HotStorage '21)*, pages 103–110, Virtual, July 2021. ACM.

Ibrahim 'Umit' Akgun, Ali Selman Aydin, Aadil Shaikh, Lukas Velikov, and Erez Zadok. A machine learning framework to improve storage system performance. In *Proceedings of the 13th ACM Workshop on Hot Topics in Storage and File Systems (HotStorage '21)*, pages 94–102, Virtual, July 2021. ACM.

Alex Merenstein, Vasily Tarasov, Ali Anwar, Deepavali Bhagwat, Julie Lee, Lukas Rupprecht, Dimitris Skourtis, Yang Yang, and Erez Zadok. CNSBench: A cloud native storage benchmark native storage. In *Proceedings of the 19th USENIX Conference on File and Storage Technologies (FAST '21)*, Virtual, February 2021. USENIX Association.

Alex Merenstein, Vasily Tarasov, Ali Anwar, Deepavali Bhagwat, Lukas Rupprecht, Dimitris Skourtis, and Erez Zadok. The case for benchmarking control operations in cloud native storage. In *Proceedings of the 12th USENIX Workshop on Hot Topics in Storage (HotStorage '20)*, Boston, MA, July 2020. USENIX.

Tyler Estro, Pranav Bhandari, Avani Wildani, and Erez Zadok. Desperately seeking ... optimal multi-tier cache configurations. In *Proceedings of the 12th USENIX Workshop on Hot Topics in Storage (HotStorage '20)*, Boston, MA, July 2020. USENIX.

Wei Su, Akshay Aurora, Ming Chen, and Erez Zadok. Supporting transactions for bulk NFSv4 compounds. In *Proceedings of the 13th ACM International Systems and Storage Conference (SYSTOR '20)*, Haifa, Israel, June 2020. ACM.

Ibrahim Umit Akgun, Geoff Kuenning, and Erez Zadok. Re-animator: Versatile high-fidelity storage-system tracing and replaying. In *Proceedings of the 13th ACM International Systems and Storage Conference (SYSTOR '20)*, pages 61–74, Haifa, Israel, June 2020. ACM.

Elinor Schoenfeld, Fan Ye, Bing Zhou, Zongxing Xie, Erez Zadok, Xi Cheng, Jacqueline B. Mondros, Patricia Bruckenthal, and Jignesh Patel. Smart aging: Utilizing low-cost, non-wearable and secure sensing technologies to support the health and wellbeing of community dwelling older adults. In *Proceedings of the HIMSS 20 Preconference Session on Aging and Tech Forum: Addressing the Silver Tsunami*, Orlando, FL, March 2020.

Ibrahim Umit Akgun, Ali Selman Aydin, and Erez Zadok. KMLib: Towards machine learning for operating systems. In *Proceedings of the 2020 On-Device Intelligence Workshop, co-located with the MLSys Conference*, February 2020. Refereed abstract+poster.

Zhen Cao, Geoff Kuenning, and Erez Zadok. Carver: Finding important parameters for storage system tuning. In *Proceedings of the 18th USENIX Conference on File and Storage Technologies (FAST '20)*, Santa Clara, CA, February 2020. USENIX Association.

Muhammad Wajahat, Aditya Yele, Tyler Estro, Anshul Gandhi, and Erez Zadok. Distribution fitting and performance modeling for storage traces. In *27th IEEE International Symposium on the Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS 2019)*, Rennes, France, October 2019. IEEE Computer Society. (**Won best student paper award**).

Anjul Tyagi, Zhen Cao, Tyler Estro, Klaus Mueller, and Erez Zadok. ICE: Interactive configuration explorer for high dimensional categorical parameter spaces. In *IEEE Conference on Visual Analytics Science and Technology (VAST 2019)*, October 2019.

Zhen Cao, Geoff Kuenning, Klaus Mueller, Anjul Tyagi, and Erez Zadok. Graphs are not enough: Using interactive visual analytics in storage research. In *Proceedings of the 11th USENIX Workshop on Hot Topics in Storage (HotStorage '19)*, Renton, WA, July 2019. USENIX.

Ming Chen and Erez Zadok. Kurma: Secure geo-distributed multi-cloud storage gateways. In *Proceedings of the 12th ACM International Systems and Storage Conference (SYSTOR '19)*, Haifa, Israel, June 2019. ACM.

Zhen Cao, Vasily Tarasov, Sachin Tiwari, and Erez Zadok. Towards better understanding of black-box auto-tuning: A comparative analysis for storage systems. In *Proceedings of the Annual USENIX Technical Conference*, Boston, MA, July 2018. USENIX Association. Data set at `http://download.filesystems.org/auto-tune/ATC-2018-auto-tune-data.sql.gz`.

S. Archak, S. Dixit, R.P. Spillane, and E. Zadok. Multi-tier caching, May 2018. US Patent 9,959,279.

Erez Zadok, Dean Hildebrand, Geoff Kuenning, and Keith Smith. POSIX is dead! Long live... errr... what exactly? In *Proceedings of the 9th USENIX Workshop on Hot Topics in Storage (HotStorage '17)*, Santa Clara, CA, July 2017. USENIX. WACI track.

Ming Chen, Dean Hildebrand, Henry Nelson, Jasmit Saluja, Ashok Subramony, and Erez Zadok. vNFS: Maximizing NFS performance with compounds and vectorized I/O. In *Proceedings of the 15th USENIX Conference on File and Storage Technologies (FAST '17)*, pages 301–314, Santa Clara, CA, February-March 2017. USENIX Association. **(Nominated for best paper award)**.

Bharath Kumar Reddy Vangoor, Vasily Tarasov, and Erez Zadok. To FUSE or not to FUSE: Performance of user-space file systems. In *Proceedings of the 15th USENIX Conference on File and Storage Technologies (FAST '17)*, pages 59–72, Santa Clara, CA, February-March 2017. USENIX Association.

Zhen Cao, Vasily Tarasov, Hari Raman, Dean Hildebrand, and Erez Zadok. On the performance variation in modern storage stacks. In *Proceedings of the 15th USENIX Conference on File and Storage Technologies (FAST '17)*, pages 329–343, Santa Clara, CA, February-March 2017. USENIX Association.

Ming Chen, Arun Vasudevan, Kelong Wang, and Erez Zadok. SeMiNAS: A secure middleware for wide-area network-attached storage. In *Proceedings of the 9th ACM International Systems and Storage Conference (ACM SYSTOR '16)*, Haifa, Israel, June 2016. ACM.

S. Archak, S. Dixit, R.P. Spillane, and E. Zadok. Multi-tier caching, May 2016. US Patent 9,355,109.

Zhen Sun, Geoff Kuenning, Sonam Mandal, Philip Shilane, Vasily Tarasov, Nong Xiao, and Erez Zadok. A long-term user-centric analysis of deduplication patterns. In *Proceedings of the 32nd International IEEE Symposium on Mass Storage Systems and Technologies (MSST '16)*, pages 1–7, Santa Clara, California, May 2016. IEEE.

Sonam Mandal, Geoff Kuenning, Dongju Ok, Varun Shastry, Pilip Shilane, Sun Zhen, Vasily Tarasov, and Erez Zadok. Using hints to improve inline block-layer deduplication. In *Proceedings of the 14th USENIX Conference on File and Storage Technologies (FAST '16)*, pages 315–322, Santa Clara, CA, February 2016. USENIX Association.

Erez Zadok, Aashray Arora, Zhen Cao, Akhilesh Chaganti, Arvind Chaudhary, and Sonam Mandal. Parametric optimization of storage systems. In *HotStorage '15: Proceedings of the 7th USENIX Workshop on Hot Topics in Storage*, Santa Clara, CA, July 2015. USENIX, USENIX.

Vasily Tarasov, Abhishek Gupta, Kumar Sourav, Sagar Trehan, and Erez Zadok. Terra incognita: On the practicality of user-space file systems. In *HotStorage '15: Proceedings of the 7th USENIX Workshop on Hot Topics in Storage*, Santa Clara, CA, July 2015. USENIX, USENIX.

Ming Chen, Dean Hildebrand, Geoff Kuenning, Soujanya Shankaranarayana, Bharat Singh, and Erez Zadok. Newer is sometimes better: An evaluation of NFSv4.1. In *Proceedings of the 2015 ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS 2015)*, Portland, OR, June 2015. ACM.

M. Chen, D. Hildebrand, G. Kuenning, S. Shankaranarayana, V. Tarasov, A. Vasudevan, E. Zadok, and K. Zakirova. Linux NFSv4.1 performance under a microscope. In *Proceedings of USENIX LISA*. USENIX Association, November 2014. Extended Abstract.

Vasily Tarasov, Deepak Jain, Geoff Kuenning, Sonam Mandal, Karthikeyani Palanisami, Philip Shilane, Sagar Trehan, and Erez Zadok. Dmdedup: Device mapper target for data deduplication. In *Proceedings of the Linux Symposium*, pages 83–95, Ottawa, Canada, July 2014.

Z. Li, A. Mukker, and E. Zadok. On the importance of evaluating storage systems' $costs. In *Proceedings of the 6th USENIX Conference on Hot Topics in Storage and File Systems*, HotStorage'14, 2014.

Vasily Tarasov, Deepak Jain, Dean Hildebrand, Renu Tewari, Geoff Kuenning, and Erez Zadok. Improving I/O performance using virtual disk introspection. In *Proceedings of the USENIX Workshop on Hot Topics in Storage and File Systems (HotStorage)*, June 2013.

Vasily Tarasov, Dean Hildebrand, Geoff Kuenning, and Erez Zadok. Virtual machine workloads: The case for new benchmarks for NAS. In *Proceedings of the USENIX Conference on File and Storage Technologies (FAST '13)*, San Jose, CA, February 2013. USENIX Association.

Pradeep Shetty, Richard Spillane, Ravikant Malpani, Binesh Andrews, Justin Seyster, and Erez Zadok. Building workload-independent storage with VT-trees. In *Proceedings of the USENIX Conference on File and Storage Technologies (FAST '13)*, pages 17–30, San Jose, CA, February 2013. USENIX Association.

Z. Li, A. Desai, C. Bhatt, and E. Zadok. vATM: vSphere adaptive task management. In *In Proceedings of the Seventh International Workshop on Feedback Computing (FC'12)*, September 2012.

David S. H. Rosenthal, Daniel C. Rosenthal, Ethan L. Miller, Ian F. Adams, Mark W. Storer, and Erez Zadok. The economics of long-term digital storage. In *The Memory of the World in the Digital age: Digitization and Preservation*. United Nations Educational, Scientific and Cultural Organization (UNESCO), September 2012.

E. Bartocci, R. Grosu, A. Karmarkar, S. A. Smolka, S. D. Stoller, E. Zadok, and J. Seyster. Adaptive runtime verification. In *Proc. 3rd International Conference on Runtime Verification (RV'12)*, Istanbul, Turkey, September 2012.

Michael A. Bender, Martin Farach-Colton, Rob Johnson, Russell Kraner, Bradley C. Kuszmaul, Dzejla Medjedovic, Pablo Montes, Pradeep Shetty, Richard P. Spillane, and Erez Zadok. Don't thrash: How to cache your hash on flash. In *Proceedings of the 38th International Conference on Very Large Data Bases (VLDB)*, Istanbul, Turkey, August 2012. Morgan Kaufmann.

Vasily Tarasov, Gyumin Sim, Anna Povzner, and Erez Zadok. Efficient I/O scheduling with accurately estimated disk drive latencies. In *In Proceedings of the 8th Annual Workshop on Operating Systems Platforms for Embedded Real-Time Applications (OSPERT'12)*, pages 36–45, Pisa, Italy, 2012. The George Washington University.

V. Tarasov, A. Mudrankit, W. Buik, P. Shilane, G. Kuenning, and E. Zadok. Generating realistic datasets for deduplication analysis. In *Proceedings of the Annual USENIX Technical Conference*, Boston, MA, June 2012. USENIX Association.

Z. Li, K. M. Greenan, A. W. Leung, and E. Zadok. Power consumption in enterprise-scale backup storage systems. In *Proceedings of the Tenth USENIX Conference on File and Storage Technologies (FAST '12)*, San Jose, CA, February 2012. USENIX Association.

V. Tarasov, K. S. Kumar, J. Ma, D. Hildebrand, A. Povzner, G. Kuenning, and E. Zadok. Extracting flexible, replayable models from large block traces. In *Proceedings of the Tenth USENIX Conference on File and Storage Technologies (FAST '12)*, San Jose, CA, February 2012. USENIX Association.

M.G. Schultz, E. Eskin, E. Zadok, M. Bhattacharyya, and S.S. J. Systems and methods for detection of new malicious executables, July 2011. US Patent 7,979,907.

Richard P. Spillane, Pradeep J. Shetty, Erez Zadok, Shrikar Archak, and Sagar Dixit. An efficient multi-tier tablet server storage architecture. In *Proceedings of the 2nd ACM Symposium on Cloud Computing (SOCC'11)*, Cascais, Portugal, October 2011.

J. Seyster, P. Radhakrishnan, S. Katoch, A. Duggal, S. D. Stoller, and E. Zadok. Redflag: A framework for analysis of kernel-level concurrency. In *Proc. of the 11th International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP'11)*, Melbourne, Australia, October 2011.

S. D. Stoller, E. Bartocci, J. Seyster, R. Grosu, K. Havelund, S. A. Smolka, and E. Zadok. Runtime verification with state estimation. In *Proc. 2nd International Conference on Runtime Verification (RV'11)*, San Francisco, CA, September 2011. (**Won best paper award**).

Z. Li, R. Grosu, K. Muppalla, S. A. Smolka, S. D. Stoller, and E. Zadok. Model discovery for energy-aware computing systems: An experimental evaluation. In *Proceedings of the 1st Workshop on Energy Consumption and Reliability of Storage Systems (ERSS'11)*, Orlando, FL, July 2011.

Michel A. Bender, Martin Farach-Colton, Rob Johnson, Bradley C. Kuszmaul, Dzejla Medjedovic, Pablo Montes, Pradeep Shetty, Richard P. Spillane, and Erez Zadok. Don't thrash: How to cache your hash on flash. In *HotStorage '11: Proceedings of the 3rd USENIX Workshop on Hot Topics in Storage*, Portland, OR, June 2011. USENIX Association.

Z. Li, R. Grosu, P. Sehgal, S. A. Smolka, S. D. Stoller, and E. Zadok. On the energy consumption and performance of systems software. In *Proceedings of the 4th Israeli Experimental Systems Conference (ACM SYSTOR '11)*, Haifa, Israel, May/June 2011. ACM.

Vasily Tarasov, Saumitra Bhanage, Erez Zadok, and Margo Seltzer. Benchmarking file system benchmarking: It *IS* rocket science. In *Proceedings of HotOS XIII:The 13th USENIX Workshop on Hot Topics in Operating Systems*, Napa, CA, May 2011.

J. Seyster, K. Dixit, X. Huang, R. Grosu, K. Havelund, S. A. Smolka, S. D. Stoller, and E. Zadok. Aspect-oriented instrumentation with GCC. In *Proc. of the 1st International Conference on Runtime Verification (RV 2010)*, Lecture Notes in Computer Science. Springer, November 2010.

R. Spillane, S. Dixit, S. Archak, S. Bhanage, and E. Zadok. Exporting kernel page caching for efficient user-level I/O. In *Proceedings of the International IEEE Symposium on Mass Storage Systems and Technologies (MSST)*, Incline Village, Nevada, May 2010. IEEE.

Priya Sehgal, Vasily Tarasov, and Erez Zadok. Evaluating performance and energy in file system server workloads. In *Proceedings of the USENIX Conference on File and Storage Technologies (FAST '10)*, pages 253–266, San Jose, CA, February 2010. USENIX Association.

Patrick McDaniel, Kevin Butler, Stephen Mclaughlin, Radu Sion, Erez Zadok, and Marianne Winslett. Towards a secure and efficient system for end-to-end provenance. In *Proceedings of the Second USENIX workshop on the Theory and Practice of Provenance (TAPP '10)*, San Jose, CA, February 2010. USENIX Association.

M.G. Schultz, E. Eskin, E. Zadok, M. Bhattacharyya, and S.J. Salvatore. System and methods for detection of new malicious executables, February 2009. US Patent 7,487,544.

D. J. Dean, S. Callanan, and E. Zadok. The visual development of GCC plug-ins. In *Proceedings of the 2009 GCC Developers' Summit*, Montreal, Canada, June 2009.

C. Yalamanchili, G. Sivathanu, K. Vijayasankar, and E. Zadok. DHIS: Discriminating hierarchical storage. In *Proceedings of the Second ACM Israeli Experimental Systems Conference (SYSTOR '09)*, Haifa, Israel, May 2009. ACM.

Rachita Kothiyal, Vasily Tarasov, Priya Sehgal, and Erez Zadok. Energy and performance evaluation of lossless file data compression on server systems. In *Proceedings of the Second ACM Israeli Experimental Systems Conference (SYSTOR '09)*, Haifa, Israel, May 2009. ACM.

R. Spillane, R. Sears, C. Yalamanchili, S. Gaikwad, M. Chinni, and E. Zadok. Story Book: An efficient extensible provenance framework. In *Proceedings of the First USENIX workshop on the Theory and Practice of Provenance (TAPP '09)*, San Francisco, CA, February 2009. USENIX Association.

R. P. Spillane, S. Gaikwad, E. Zadok, C. P. Wright, and M. Chinni. Enabling transactional file access via lightweight kernel extensions. In *Proceedings of the Seventh USENIX Conference on File and Storage Technologies (FAST '09)*, pages 29–42, San Francisco, CA, February 2009. USENIX Association.

S. Sundararaman, G. Sivathanu, and E. Zadok. Selective versioning in a secure disk system. In *Proceedings of the 17th USENIX Security Symposium*, pages 259–274, San Jose, CA, July-August 2008. USENIX Association.

A. Traeger, I. Deras, and E. Zadok. DARC: Dynamic analysis of root causes of latency distributions. In *Proceedings of the 2008 International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS 2008)*, pages 277–288, Annapolis, MD, June 2008. ACM.

D. P. Quigley, K. Albrecht, and E. Zadok. PLEASE: Policy language for easy administration of SELinux. In *Proceedings of the Second annual Computer Security Conference (CSC 2008)*, Myrtle Beach, SC, April 2008.

S. Callanan, D. J. Dean, M. Gorbovitski, R. Grosu, J. Seyster, S. A. Smolka, S. D. Stoller, and E. Zadok. Software monitoring with bounded overhead. In *Proceedings of the 2008 NSF Next Generation Software Workshop, in conjunction with the 2008 International Parallel and Distributed Processing Symposium (IPDPS 2008)*, Miami, FL, April 2008.

E. Zadok. The layers are coming, the layers are coming. In *The 2008 Linux Storage and Filesystem Workshop (LSF'08)*, San Jose, CA, February 2008. USENIX Association.

K. Vijayasankar, G. Sivathanu, S. Sundararaman, and E. Zadok. Exploiting type-awareness in a self-recovering disk. In *Proceedings of the Third ACM Workshop on Storage Security and Survivability (StorageSS 2007)*, pages 25–30, Alexandria, VA, October 2007. ACM.

A. Traeger, K. Thangavelu, and E. Zadok. Round-trip privacy with NFSv4. In *Proceedings of the Third ACM Workshop on Storage Security and Survivability (StorageSS 2007)*, pages 1–7, Alexandria, VA, October 2007. ACM.

N. Joukov, A. M. Krishnakumar, C. Patti, A. Rai, S. Satnur, A. Traeger, and E. Zadok. RAIF: Redundant array of independent filesystems. In *Proceedings of 24th IEEE Conference on Mass Storage Systems and Technologies (MSST 2007)*, pages 199–212, San Diego, CA, September 2007. IEEE.

S. Callanan, D. J. Dean, and E. Zadok. Extending GCC with modular GIMPLE optimizations. In *Proceedings of the 2007 GCC Developers' Summit*, pages 31–37, Ottawa, Canada, July 2007.

Richard P. Spillane, Charles P. Wright, Gopalan Sivathanu, and Erez Zadok. Rapid file system development using ptrace. In *Proceedings of the Workshop on Experimental Computer Science (ExpCS), in conjunction with ACM FCRC*, page 22, San Diego, CA, June 2007.

J. Sipek, Y. Pericleous, and E. Zadok. Kernel support for stackable file systems. In *Proceedings of the 2007 Ottawa Linux Symposium (OLS 2007)*, volume 2, pages 223–227, Ottawa, Canada, June 2007.

A. D. A. Martin and E. Zadok. Autofs - an automounting file system for freebsd 6.x. In *Proceedings of the 2007 BSDCan Technical BSD Conference*, Ottawa, ON, Canada, May 2007.

S. Callanan, R. Grosu, J. Seyster, S. A. Smolka, and E. Zadok. Model predictive control for memory profiling. In *Proceedings of the 2007 NSF Next Generation Software Workshop, in conjunction with the 2007 International Parallel and Distributed Processing Symposium (IPDPS 2007)*, Long beach, CA, March 2007.

J. Sipek and E. Zadok. Making Linux stacking-friendly. In *The 2007 Linux Storage and Filesystem Workshop (LSF'07)*, San Jose, CA, February 2007. USENIX Association.

G. Sivathanu, S. Sundararaman, and E. Zadok. Type-safe disks. In *Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2006)*, pages 15–28, Seattle, WA, November 2006. ACM SIGOPS.

Nikolai Joukov, Ashivay Traeger, Rakesh Iyer, Charles P. Wright, and Erez Zadok. Operating system profiling via latency analysis. In *Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2006)*, pages 89–102, Seattle, WA, November 2006. ACM SIGOPS.

N. Joukov, H. Papaxenopoulos, and E. Zadok. Secure deletion myths, issues, and solutions. In *Proceedings of the Second ACM Workshop on Storage Security and Survivability (StorageSS 2006)*, pages 61–66, Alexandria, VA, October 2006. ACM.

A. Traeger, N. Joukov, J. Sipek, and E. Zadok. Using free web storage for data backup. In *Proceedings of the Second ACM Workshop on Storage Security and Survivability (StorageSS 2006)*, pages 73–77, Alexandria, VA, October 2006. ACM.

D. P. Quigley, J. Sipek, C. P. Wright, and E. Zadok. UnionFS: User- and community-oriented development of a unification filesystem. In *Proceedings of the 2006 Linux Symposium*, volume 2, pages 349–362, Ottawa, Canada, July 2006.

S. Callanan, R. Grosu, X. Huang, S. A. Smolka, and E. Zadok. Compiler-assisted software verification using plug-ins. In *Proceedings of the 2006 NSF Next Generation Software Workshop, in conjunction with the 2006 International Parallel and Distributed Processing Symposium (IPDPS 2006)*, Rhodes Island, Greece, April 2006. DOI 10.1109/IPDPS.2006.1639579.

S. Callanan, R. Grosu, A. Rai, S. A. Smolka, M. R. True, and E. Zadok. Runtime verification for high-confidence systems: A Monte Carlo approach. In *Proceedings of the Second Workshop on Model Based Testing (MBT 2006), held in conjunction with the European Joint Conferences on Theory and Practice of Software (ETAPS 2006)*, pages 41–53, Vienna, Austria, March 2006.

Nikolai Joukov, Timothy Wong, and Erez Zadok. Accurate and efficient replaying of file system traces. In *Proceedings of the USENIX Conference on File and Storage Technologies (FAST '05)*, pages 337–350, San Francisco, CA, December 2005. USENIX Association.

N. Joukov and E. Zadok. Adding secure deletion to your favorite file system. In *Proceedings of the third international IEEE Security In Storage Workshop (SISW 2005)*, pages 63–70, San Francisco, CA, December 2005. IEEE Computer Society.

N. Joukov, A. Kashyap, G. Sivathanu, and E. Zadok. Kefence: An electric fence for kernel buffers. In *Proceedings of the First ACM Workshop on Storage Security and Survivability (StorageSS 2005)*, pages 37–43, FairFax, VA, November 2005. ACM. (**Won best short paper award**).

G. Sivathanu, C. P. Wright, and E. Zadok. Ensuring data integrity in storage: Techniques and applications. In *Proceedings of the First ACM Workshop on Storage Security and Survivability (StorageSS 2005)*, pages 26–36, FairFax, VA, November 2005. ACM.

N. Joukov, A. Rai, and E. Zadok. Increasing distributed storage survivability with a stackable raid-like file system. In *Proceedings of the 2005 IEEE/ACM Workshop on Cluster Security, in conjunction with the Fifth IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid 2005)*, pages 82–89, Cardiff, UK, May 2005. IEEE. (**Won best paper award**).

C. P. Wright, N. Joukov, D. Kulkarni, Y. Miretskiy, and E. Zadok. Auto-pilot: A platform for system software benchmarking. In *Proceedings of the Annual USENIX Technical Conference, FREENIX Track*, pages 175–187, Anaheim, CA, April 2005. USENIX Association.

E. Zadok, S. Callanan, A. Rai, G. Sivathanu, and A. Traeger. Efficient and safe execution of user-level code in the kernel. In *Proceedings of the 2005 NSF Next Generation Software Workshop, in conjunction with the 2005 International Parallel and Distributed Processing Symposium (IPDPS 2005)*, page 221.1, Denver, CO, April 2005.

A. Kashyap, S. Patil, G. Sivathanu, and E. Zadok. I3FS: An in-kernel integrity checker and intrusion detection file system. In *Proceedings of the 18th USENIX Large Installation System Administration Conference (LISA 2004)*, pages 69–79, Atlanta, GA, November 2004. USENIX Association.

Y. Miretskiy, A. Das, C. P. Wright, and E. Zadok. Avfs: An on-access anti-virus file system. In *Proceedings of the 13th USENIX Security Symposium (Security 2004)*, pages 73–88, San Diego, CA, August 2004. USENIX Association.

K. Muniswamy-Reddy, C. P. Wright, A. Himmer, and E. Zadok. A versatile and user-oriented versioning file system. In *Proceedings of the USENIX Conference on File and Storage Technologies (FAST '04)*, pages 115–128, San Francisco, CA, March/April 2004. USENIX Association.

Akshat Aranya, Charles P. Wright, and Erez Zadok. Tracefs: A file system to trace them all. In *Proceedings of the USENIX Conference on File and Storage Technologies (FAST '04)*, pages 129–143, San Francisco, CA, March/April 2004. USENIX Association.

E. Zadok, J. Osborn, A. Shater, C. P. Wright, K. Muniswamy-Reddy, and J. Nieh. Reducing storage management costs via informed user-based policies. In *Proceedings of the 12th NASA Goddard, 21st IEEE Conference on Mass Storage Systems and Technologies (MSST 2004)*, pages 193–197, College Park, MD, April 2004. IEEE.

C. P. Wright, J. Dave, and E. Zadok. Cryptographic file systems performance: What you don't know can hurt you. In *Proceedings of the Second IEEE International Security In Storage Workshop (SISW 2003)*, pages 47–61, Washington, DC, October 2003. IEEE Computer Society.

C. P. Wright, M. Martino, and E. Zadok. NCryptfs: A secure and convenient cryptographic file system. In *Proceedings of the Annual USENIX Technical Conference*, pages 197–210, San Antonio, TX, June 2003. USENIX Association.

Amit Purohit, Charles P. Wright, Joseph Spadavecchia, and Erez Zadok. Cosy: Develop in user-land, run in kernel-mode. In *Proceedings of the 2003 ACM Workshop on Hot Topics in Operating Systems (HotOS IX)*, pages 109–114, Lihue, Hawaii, May 2003. USENIX Association.

R. Grosu, E. Zadok, S. A. Smolka, R. Cleaveland, and Y. A. Liu. High-confidence operating systems. In *Proceedings of the Tenth SIGOPS European Workshop: "Can we really depend on an OS?"*, pages 205–208, Saint-Emilion, France, September 2002.

J. Spadavecchia and E. Zadok. Enhancing NFS cross-administrative domain access. In *Proceedings of the Annual USENIX Technical Conference, FREENIX Track*, pages 181–194, Monterey, CA, June 2002. USENIX Association.

E. Zadok. Overhauling AMD for the '00s: A case study of GNU autotools. In *Proceedings of the Annual USENIX Technical Conference, FREENIX Track*, pages 287–297, Monterey, CA, June 2002. USENIX Association.

E. Zadok, J. M. Anderson, I. Bădulescu, and J. Nieh. Fast indexing: Support for size-changing algorithms in stackable file systems. In *Proceedings of the Annual USENIX Technical Conference (ATC)*, pages 289–304, Boston, MA, June 2001. USENIX Association.

E. Zadok and J. Nieh. FiST: A language for stackable file systems. In *Proceedings of the Annual USENIX Technical Conference*, pages 55–70, San Diego, CA, June 2000. USENIX Association.

E. Zadok, I. Bădulescu, and A. Shender. Extending file systems using stackable templates. In *Proceedings of the Annual USENIX Technical Conference*, pages 57–70, Monterey, CA, June 1999. USENIX Association.

E. Zadok and I. Bădulescu. A stackable file system interface for Linux. In *LinuxExpo Conference Proceedings*, pages 141–151, Raleigh, NC, May 1999.

E. Zadok and A. Dupuy. HLFSD: Delivering Email to your $HOME. In *Proceedings of the Seventh USENIX Systems Administration Conference (LISA VII)*, pages 243–254, Monterey, CA, November 1993. USENIX Association.

E. Zadok and D. Duchamp. Discovery and hot replacement of replicated read-only file systems, with application to mobile computing. In *Proceedings of the Summer USENIX Technical Conference*, pages 69–85, Cincinnati, OH, June 1992. USENIX Association.

M. G. Schultz, E. Eskin, E. Zadok, M. Bhattacharyya, and S. J. Stolfo. MEF: Malicious Email Filter — A UNIX Mail Filter that Detects Malicious Windows Executables. In *Proceedings of the Annual USENIX Technical Conference, FREENIX Track*, pages 245–252, Boston, MA, June 2001. (**Won best student paper award**).

M. G. Schultz, E. Eskin, E. Zadok, and S. J. Stolfo. Data mining methods for detection of new malicious executables. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 38–49, Oakland, CA, May 2001.

W. Lee, W. Fan, M. Miller, S. Stolfo, and E. Zadok. Toward cost-sensitive modeling for intrusion detection and response. In *Workshop on Intrusion Detection and Prevention, Seventh ACM Conference on Computer Security (CCS)*, pages 1–22, Athens, Greece, November 2000.

Other Publications

Maliha Tabassum. Integrity protections for secure long-term storage. Technical Report FSL-24-05, Computer Science Department, Stony Brook University, December 2024.

Yifei Liu. Towards efficient, scalable, and versatile file system model checking. Technical Report FSL-24-04, Computer Science Department, Stony Brook University, November 2024.

Christopher Smith. Long-term secure archival using proactively secret-shared datastores. Technical Report FSL-24-03, Computer Science Department, Stony Brook University, September 2024.

Alex Merenstein. *Techniques for Storage Performance Measurement and Data Management in Container-Native and Serverless Environments*. PhD thesis, May 2024.

Christopher Smith, Maliha Tabassum, Soumya Chowdary Daruru, Gaurav Kulhare, Arvin Wang, Ethan Miller, and Erez Zadok. Plugging the leaks in secure archival systems. Non-archival Poster presentation at Stony Brook University Department of Computer Science Graduate Research Day, March 2024.

Christopher Smith, Maliha Tabassum, Soumya Chowdary Daruru, Gaurav Kulhare, Arvin Wang, Ethan Miller, and Erez Zadok. Plugging the leaks in secure archival systems. Work in Progress

presentation at the 22nd USENIX Conference on File and Storage Technologies (FAST'24), February 2024.

Christopher Smith, Maliha Tabassum, Soumya Chowdary Daruru, Gaurav Kulhare, Arvin Wang, Ethan Miller, and Erez Zadok. Plugging the leaks in secure archival systems. Non-archival Poster presentation in the 22nd USENIX Conference on File and Storage Technologies (FAST'24), February 2024.

Alex Merenstein. Moving beyond host based virtualization: New techniques for performance measurement and data management in cloud native environments. Technical Report FSL-24-01, Computer Science Department, Stony Brook University, January 2024.

Mário Antunes, Tyler Estro, Pranav Bhandari, Anshul Gandhi, Geoff Kuenning, Yifei Liu, Carl Waldspurger, Avani Wildani, and Erez Zadok. (multi)knee/elbow point detection library, December 2023.

Syed Rafiul Hussain, Patrick McDaniel, Anshul Gandhi, Kanad Ghose, Kartik Gopalan, Dongyoon Lee, Yu David Liu, Zhenhua Liu, Shuai Mu, and Erez Zadok. Verifiable sustainability in data centers, July 2023. https://arxiv.org/abs/2307.11993.

Tyler Estro, Mário Antunes, Pranav Bhandari, Anshul Gandhi, Geoff Kuenning, Yifei Liu, Carl Waldspurger, Avani Wildani, and Erez Zadok. Guiding simulations of multi-tier storage caches using knee detection. Technical Report FSL-23-01, Computer Science Department, Stony Brook University, July 2023.

Elinor Schoenfeld, Tracy Trimboli, Givenchy Ayisi-Boahene, Patricia Bruckenthal, Shelley Horwitz, Erez Zadok, and Fan Ye. Smart aging: Engaging older adults to guide sensor development to support aging in place. In *Issues and experiences of integrating patient and community stakeholder perspectives in aging research*, volume 7, pages 583–583, Tampa, FL, Dec 2023.

Elinor Schoenfeld, Tracy Trimboli, Kaylyn Finnerty, Givenchy Ayisi-Boahene, Patricia Bruckenthal, Shelley Horwitz, Erez Zadok, and Fan Ye. Engaging older adults to guide low-cost non-wearable sensor technology development to age in place: Survey findings. In *Gerontology Society of America Annual Meeting (GSA 2023)*, volume 7, pages 692–692, Tampa, FL, Dec 2023.

Yifei Liu, Gerard Holzmann, Geoff Kuenning, Scott Smolka, and Erez Zadok. The case for model checking emerging file systems. Non-archival poster presentation in the 17th USENIX Symposium on Operating Systems Design and Implementation (OSDI '23), July 2023.

Tyler Estro, Kamalnath Polakam, and Erez Zadok. Using bash notebooks for systems education (user meeting 23). `https://www.chameleoncloud.org/experiment/share/82fbba82-1b34-4562-9866-b7adaec94b19`.

Ibrahim "Umit" Akgun. *Using Machine Learning to Improve Operating Systems' I/O Subsystems*. PhD thesis, December 2022.

Ibrahim "Umit" Akgun. Using machine learning to improve operating systems' I/O subsystems. Technical Report FSL-22-02, Computer Science Department, Stony Brook University, June 2022.

Sayontan Ghosh, Amanpreet Singh, Alex Merenstein, Wei Su, Scott Smolka, Erez Zadok, and Niranjan Balasubramanian. SpecNFS: A challenge dataset towards extracting formal models from natural language specifications. Non-archival poster presentation in the Workshop on Natural Language Processing for Programming in the ACL'21 conference, August 2021.

Tyler Estro. Accelerating multi-tier cache evaluations with intelligent MRC point selection. Technical Report FSL-21-01, Computer Science Department, Stony Brook University, January 2021.

Wei Su. Supporting transactions for bulk NFSv4 compounds. Technical Report FSL-20-02, Computer Science Department, Stony Brook University, December 2020.

Alex Merenstein. CNSBench: A cloud native storage benchmark. Technical Report FSL-20-01, Computer Science Department, Stony Brook University, December 2020.

Ibrahim "Umit" Akgun. Re-animator: Versatile high-fidelity system-call tracing and replaying. Technical Report FSL-19-02, Computer Science Department, Stony Brook University, May 2019.

George Amvrosiadis, Ali R. Butt, Vasily Tarasov, Erez Zadok, Ming Zhao, Irfan Ahmad, Remzi H. Arpaci-Dusseau, Feng Chen, Yiran Chen, Yong Chen, Yue Cheng, Vijay Chidambaram, Dilma Da Silva, Angela Demke-Brown, Peter Desnoyers, Jason Flinn, Xubin He, Song Jiang, Geoff Kuenning, Min Li, Carlos Maltzahn, Ethan L. Miller, Kathryn Mohror, Raju Rangaswami, Narasimha Reddy, David Rosenthal, Ali Saman Tosun, Nisha Talagala, Peter Varman, Sudharshan Vazhkudai, Avani Waldani, Xiaodong Zhang, Yiying Zhang, and Mai Zheng. Data storage research vision 2025: Report on NSF visioning workshop held may 30–june 1, 2018. Technical report, National Science Foundation, February 2019. `https://dl.acm.org/citation.cfm?id=3316807`.

Zhen Cao. *A Practical Auto-Tuning Framework for Storage Systems*. PhD thesis, Stony Brook University, January 2019.

Zhen Cao. A practical, real-time auto-tuning framework for storage systems. Technical Report FSL-18-01, Computer Science Department, Stony Brook University, April 2018.

M. Chen. *Kurma: Efficient and Secure Multi-Cloud Storage Gateways for Network-Attached Storage*. PhD thesis, Computer Science Department, Stony Brook University, April 2017. Technical Report FSL-17-01.

Bharath Kumar Reddy Vangoor. To FUSE or not to FUSE? analysis and performance characterization of the FUSE user-space file system framework. Master's thesis, Stony Brook University, December 2016. Technical Report FSL-16-02, `http://www.fsl.cs.sunysb.edu/docs/fuse/bharath-msthesis.pdf`.

Zhen Cao. Parametric optimization of storage systems. Technical Report FSL-16-01, Computer Science Department, Stony Brook University, January 2016.

Sonam Mandal. Design and implementation of an open-source deduplication platform for research. Technical Report FSL-15-03, Computer Science Department, Stony Brook University, December 2015.

Ming Chen. Kurma: Geo-distributed secure middlewares for cloud-backed network attached storage. Technical Report FSL-15-02, Computer Science Department, Stony Brook University, November 2015.

Arun Olappamanna Vasudevan. Finding the right balance: Security vs. performance with network storage systems. Master's thesis, Stony Brook University, May 2015. Technical Report FSL-15-01, `http://www.fsl.cs.sunysb.edu/docs/arun-msthesis/arun-msthesis.pdf`.

Benixon Arul Dhas, Erez Zadok, James Borden, and Jim Malina. Evaluation of Nilfs2 for shingled magnetic recording (SMR) disks. Technical Report FSL-14-03, Stony Brook University, September 2014.

M. Chen, D. Hildebrand, G. Kuenning, S. Shankaranarayana, V. Tarasov, A. Vasudevan, E. Zadok, and K. Zakirova. Linux NFSv4.1 performance under a microscope. Technical Report FSL-14-02, Stony Brook University, August 2014.

Z. Li. *GreenDM: A Versatile Tiering Hybrid Drive for the Trade-Off Evaluation of Performance, Energy, and Endurance*. PhD thesis, Computer Science Department, Stony Brook University, May 2014.

M. Palmur, Z. Li, and E. Zadok. Cpuidle from user space. Technical Report FSL-13-05, Computer Science Department, Stony Brook University, December 2013.

V. Tarasov. *Multi-dimensional Workload Analysis and Synthesis for Modern Storage Systems*. PhD thesis, Computer Science Department, Stony Brook University, December 2013.

Zhichao Li. System-aware resource scheduling for performance, energy, and reliability in tiered storage systems. Technical Report FSL-13-03, Computer Science Department, Stony Brook University, May 2013.

V. Tarasov. Multi-dimensional workload analysis and synthesis for modern storage systems. Technical Report FSL-13-02, Computer Science Department, Stony Brook University, April 2013.

M. Chen and E. Zadok. Greendm: A versatile hybrid drive for energy and performance. Technical Report FSL-13-01, Computer Science Department, Stony Brook University, April 2013.

J. Seyster. *Runtime Verification of Kernel-Level Concurrency Using Compiler-Based Instrumentation*. PhD thesis, Computer Science Department, Stony Brook University, December 2012. Technical Report FSL-12-05, `www.fsl.cs.sunysb.edu/docs/jseyster-dissertation/redflag.pdf`.

Pradeep J. Shetty, Richard P. Spillane, Ravikant R. Malpani, Binesh Andrews, Justin Seyster, and Erez Zadok. Unifying file systems and databases efficiently. Technical Report FSL-12-06, Stony Brook University Computer Science, September 2012.

A. Mudrankit. A context aware block layer: The case for block layer deduplication. Master's thesis, Stony Brook University, May 2012. Technical Report FSL-12-04.

P. J. Shetty. From tuples to files: a fast transactional system store and file system. Master's thesis, Stony Brook University, May 2012. Technical Report FSL-12-03.

S. Archak, S. Dixit, R.P. Spillane, and E. Zadok. Multi-tier caching, March 2012. US Patent App. 13/159,039.

R. P. Spillane. *Efficient, Scalable, and Versatile Application and System Transaction Management for Direct Storage Layers*. PhD thesis, Computer Science Department, Stony Brook University, Jan 2012. Technical Report FSL-12-02.

J. Seyster. Runtime verification of kernel-level concurrency using compiler-based instrumentation. Technical Report FSL-12-01, Computer Science Department, Stony Brook University, January 2012. `www.fsl.cs.sunysb.edu/docs/jseyster-proposal/redflag.pdf`.

A. Duggal. Stopping data races using Redflag. Master's thesis, Stony Brook University, May 2010. Technical Report FSL-10-02, `http://www.fsl.cs.sunysb.edu/docs/abhinav-thesis/abhinav_thesis.pdf`.

P. Sehgal. Optimizing energy and performance for server-class file system workloads. Master's thesis, Stony Brook University, May 2010. Technical Report FSL-10-01.

S. Callanan. *Flexible Debugging with Controllable Overhead*. PhD thesis, Computer Science Department, Stony Brook University, August 2009. Technical Report FSL-09-01, `http://www.fsl.cs.sunysb.edu/docs/callanan09phdthesis/callanan09phdthesis.pdf`.

R. P. Spillane. SAMT performance proof. Technical Report FSL-09-03, Stony Brook University Computer Science, June 2009. www.fsl.cs.sunysb.edu/~rick.

D. Dean. The visual development of GCC plug-ins with GDE. Master's thesis, Stony Brook University, April 2009. Technical Report FSL-09-04.

R. Kothiyal. Energy and performance evaluation of lossless file data compression on computer systems. Master's thesis, Stony Brook University, April 2009. Technical Report FSL-09-02.

R. P. Spillane, S. Gaikwad, E. Zadok, C. P. Wright, and M. Chinni. Enabling transactional file access via lightweight kernel extensions. Technical Report FSL-08-05, Stony Brook University Computer Science, August 2008.

A. Traeger. *Analyzing Root Causes of Latency Distributions*. PhD thesis, Computer Science Department, Stony Brook University, August 2008. Technical Report FSL-08-04.

J. Seyster. Techniques for visualizing software execution. Technical Report FSL-08-03, Computer Science Department, Stony Brook University, March 2008. www.fsl.cs.sunysb.edu/docs/debugging-tr/software-visualization-tr.pdf.

S. Callanan. Remote debugging with controllable overhead. Technical Report FSL-08-02, Computer Science Department, Stony Brook University, March 2008. www.fsl.cs.sunysb.edu/docs/debugging-tr/debugging-proposal.pdf.

G. Sivathanu. *End-to-End Abstractions for Application-Aware Storage*. PhD thesis, Computer Science Department, Stony Brook University, May 2008. Technical Report FSL-08-01.

D. P. Quigley. PLEASE: Policy language for easy administration of SELinux. Master's thesis, Stony Brook University, May 2007. Technical Report FSL-07-02, www.fsl.cs.sunysb.edu/docs/dquigley-msthesis/dquigley-msthesis.pdf.

N. Joukov. *Versatile, Portable, and Efficient File System Profiling*. PhD thesis, Computer Science Department, Stony Brook University, December 2006. Technical Report FSL-06-05.

C. P. Wright. *Extending ACID Semantics to the File System via ptrace*. PhD thesis, Computer Science Department, Stony Brook University, May 2006. Technical Report FSL-06-04.

N. Joukov, A. Traeger, R. Iyer, C. P. Wright, and E. Zadok. Latency as a viable metric for OS profiling. Technical Report FSL-06-03, Computer Science Department, Stony Brook University, April 2006.

R. Spillane, C. P. Wright, G. Sivathanu, and E. Zadok. Rapid file system development using ptrace. Technical Report FSL-06-02, Computer Science Department, Stony Brook University, January 2006.

C. P. Wright, R. Spillane, G. Sivathanu, and E. Zadok. Extending ACID semantics to the file system. Technical Report FSL-06-01, Computer Science Department, Stony Brook University, January 2006. www.fsl.cs.sunysb.edu/docs/amino-tr/amino.pdf.

G. Sivathanu and E. Zadok. A versatile persistent caching framework for file systems. Technical Report FSL-05-05, Computer Science Department, Stony Brook University, December 2005. www.fsl.cs.sunysb.edu/docs/cachefs-tr/cachefs.pdf.

N. Joukov, A. Traeger, C. P. Wright, and E. Zadok. Benchmarking file system benchmarks. Technical Report FSL-05-04, Computer Science Department, Stony Brook University, December 2005. www.fsl.cs.sunysb.edu/docs/fsbench/fsbench.pdf.

S. Satnur. Storage virtualization with a stackable file system. Master's thesis, Stony Brook University, December 2005. Technical Report FSL-05-03, www.fsl.cs.sunysb.edu/docs/sunil-msthesis/sunil-msthesis.pdf.

E. Zadok, S. Callanan, R. Grosu, A. Rai, S. Smolka, and M. R. True. Runtime verification for high-confidence OS kernels: A Monte Carlo approach. Technical Report FSL-05-02, Computer Science Department, Stony Brook University, November 2005.

A. Rai and E. Zadok. On the role of static analysis in operating system checking and runtime verification. Technical Report FSL-05-01, Computer Science Department, Stony Brook University, May 2005. www.fsl.cs.sunysb.edu/docs/kgcc-rpe/kgcc.pdf.

E. Zadok, C. Wright, A. Aranya, A. Das, Y. Miretskiy, K.K. Muniswamy-Reddy, and A. Himmer. Stackable file systems and methods thereof, December 2005. US Patent App. 10/862,212.

A. Kashyap. File system extensibility and reliability using an in-kernel database. Master's thesis, Stony Brook University, December 2004. Technical Report FSL-04-06, www.fsl.cs.sunysb.edu/docs/kbdbfs-msthesis/kbdbfs.pdf.

A. Aranya. Versatile file system tracing with Tracefs. Master's thesis, Stony Brook University, August 2004. Technical Report FSL-04-05, www.fsl.cs.sunysb.edu/docs/tracefs-msthesis/tracefs.pdf.

G. Sivathanu, C. P. Wright, and E. Zadok. Enhancing file system integrity through checksums. Technical Report FSL-04-04, Computer Science Department, Stony Brook University, May 2004. www.fsl.cs.sunysb.edu/docs/nc-checksum-tr/nc-checksum.pdf.

A. Traeger, A. Rai, C. P. Wright, and E. Zadok. NFS file handle security. Technical Report FSL-04-03, Computer Science Department, Stony Brook University, May 2004. www.fsl.cs.sunysb.edu/docs/nfscrack-tr/nfscrack.pdf.

C. P. Wright and E. Zadok. Operating system support for extensible secure file systems. Technical Report FSL-04-02, Computer Science Department, Stony Brook University, May 2004. www.fsl.cs.sunysb.edu/docs/secfs-rpe/secfs.pdf.

C. P. Wright, J. Dave, P. Gupta, H. Krishnan, E. Zadok, and M. N. Zubair. Versatility and Unix semantics in a fan-out unification file system. Technical Report FSL-04-01b, Computer Science Department, Stony Brook University, October 2004. www.fsl.cs.sunysb.edu/docs/unionfs-tr/unionfs.pdf.

K. Muniswamy-Reddy. Versionfs: A versatile and user-oriented versioning file system. Master's thesis, Stony Brook University, December 2003. Technical Report FSL-03-03, www.fsl.cs.sunysb.edu/docs/versionfs-msthesis/versionfs.pdf.

C. P. Wright, J. Dave, and E. Zadok. Cryptographic file systems performance: What you don't know can hurt you. Technical Report FSL-03-02, Computer Science Department, Stony Brook University, August 2003. www.fsl.cs.sunysb.edu/docs/nc-perf/perf.pdf.

E. Zadok, J. Osborn, A. Shater, C. P. Wright, K. Muniswamy-Reddy, and J. Nieh. Reducing storage management costs via informed user-based policies. Technical Report FSL-02-02, Computer Science Department, Stony Brook University, September 2002. www.fsl.cs.sunysb.edu/docs/equota-policy/policy.pdf.

A. Purohit, J. Spadavecchia, C. Wright, and E. Zadok. Improving application performance through system call composition. Technical Report FSL-02-01, Computer Science Department, Stony Brook University, June 2003. www.fsl.cs.sunysb.edu/docs/cosy-perf/.

O. C. Leonard, J. Nieh, E. Zadok, J. Osborn, A. Shater, and C. Wright. The design and implementation of elastic quotas: A system for flexible file system management. Technical Report CUCS-014-02, Computer Science Department, Columbia University, June 2002. www.cs.columbia.edu/~library.

E. Zadok. *FiST: A System for Stackable File System Code Generation*. PhD thesis, Computer Science Department, Columbia University, May 2001. www.fsl.cs.sunysb.edu/docs/zadok-phd-thesis/thesis.pdf.

Wenke Lee, Matthew Miller, Salvatore Stolfo, Kahil Jallad, Christopher T Park, Erez Zadok, and Vijay Prabhakar. Toward cost-sensitive modeling for intrusion detection. Technical Report CUCS-002-00, Computer Science Department, Columbia University, 2000. http://academiccommons.columbia.edu/download/fedora_content/download/ac:110332/CONTENT/cucs-002-00.pdf.

Erez Zadok, Johan M Andersen, Ion Badulescu, and Jason Nieh. Performance of size-changing algorithms in stackable file systems. Technical Report CUCS-023-00, Computer Science Department, Columbia University, 2000. http://academiccommons.columbia.edu/download/fedora_content/download/ac:110385/CONTENT/cucs-023-00.pdf.

E. Zadok. Stackable file systems as a security tool. Technical Report CUCS-036-99, Computer Science Department, Columbia University, December 1999. www.cs.columbia.edu/~library.

E. Zadok and I. Bădulescu. Usenetfs: A stackable file system for large article directories. Technical Report CUCS-022-98, Computer Science Department, Columbia University, July 1998. www.cs.columbia.edu/~library.

E. Zadok, I. Bădulescu, and A. Shender. Cryptfs: A stackable vnode level encryption file system. Technical Report CUCS-021-98, Computer Science Department, Columbia University, June 1998. www.cs.columbia.edu/~library.

E. Zadok. *FiST: A File System Component Compiler*. PhD thesis, Computer Science Department, Columbia University, April 1997. www.fsl.cs.sunysb.edu/docs/zadok-thesis-proposal.

E. Zadok. Discovery and hot replacement of replicated read-only file systems, with application to mobile computing. Master's thesis, Computer Science Department, Columbia University, October 1997. Technical Report CUCS-036-94, www.cs.columbia.edu/~library.

E. Zadok and A. Lih. PGMAKE: A Portable Distributed Make System. Technical Report CUCS-035-94, Computer Science Department, Columbia University, July 1994. www.cs.columbia.edu/~library.

Manuals

C. P. Wright and E. Zadok. *The Auto-pilot Benchmarking Suite User Manual*, 2.1 edition, July 2005. www.filesystems.org/docs/auto-pilot/.

J. S. Pendry, N. Williams, and E. Zadok. *Am-utils User Manual*, 6.1b3 edition, July 2003. www.am-utils.org.

PATENTS

Multi-Tier Caching, U.S. Patent 9,959,279, issued May 1, 2018.

Multi-Tier Caching, U.S. Patent 9,355,109, issued May 31, 2016.

Systems and methods for detection of new malicious executables, U.S. Patent 7,979,907B2, issued July 12, 2011.

System and methods for detection of new malicious executables, U.S. Patent 7,487,544B2, issued February 3, 2009.

ACADEMIC SERVICE

Program Committee Chair

2019-2020	The 2020 USENIX Annual Technical Conference (ATC 2020) program committee co-chair.
2015	The 13th USENIX Conference on File and Storage Technologies (FAST 2015) program committee co-chair.
2012	The 5th Israeli Experimental Systems Conference (SYSTOR 2012), program committee co-chair.
2010	The First Usenix Workshop on Sustainable Information Technology (SustainIT 2010), program committee co-chair.
2008	The First File-systems and Storage Benchmarking workshop, program chair (hosted at the Storage Systems Research Center at the University of California at Santa Cruz).
2006	ACM Storage Security and Survivability (StorageSS) workshop, program committee co-chair.
2005	Usenix Annual Conference 2005, Invited Talks program co-chair
2004	Usenix Security 2004, Works-in-Progress program chair
2004	Usenix Security 2004, Poster presentations program chair
2003	USENIX FreeNIX 2003 Annual Conference, program chair

Panels

Oct 2021	NSF CAREER review panel
June 2013	Panelist, EMC University Day, "University/Industry Relationships"
Jan 2012	NSF Sustainability Research Network (SRN) program
May 2010	Department of Energy, Computational and Technology Research program
Feb 2009	Panel moderator, The First Usenix Workshop on Sustainable Information Technology (SustainIT 2010), "The Present and Future of Sustainability R&D"
Oct 2009	Panelist, Workshop on Hot Topics in Storage and File Systems (HotStorage'09), "Rethinking File Systems"
Oct 2009	NSF CAREER (CSR+CNS) review panel

Apr 2007	NSF Computer Systems Research (CSR) Parallel and Distributed Operating Systems (PDOS) review panel
Aug 2004	NSF CyberTrust PI meeting panelist, "Software Assurance," Pittsburgh, PA
Aug 2004	Workshop on Information Assurance Education, Stony Brook University (WIAED 2004), panel chair, "How to establish your own IA Curriculum"
Apr 2004	NSF Distributed Systems and Compilers (DSC) review panel
Apr 2001	NSF ITR Operating Systems and Compilers (OSC) review panel, small grants

Editor-in-Chief

Nov 2012–present	Editor-in-Chief, ACM Transactions on Storage (TOS). `https://dl.acm.org/journal/tos/`

Editor

2014	Editorial Board, USENIX Journal of Education in System Administration (JESA). `www.usenix.org/jesa/`
Aug 2009–2012	Associate Editor, ACM Transactions on Storage (TOS). `https://tos.acm.org/`
Jul 2001–Jan 2009	Bulletin Editor, Webmaster, and lists maintainer for IEEE Technical Committee on Operating Systems and Application Environments (TCOS). `www.tcos.org`.

Book Reviews

Summer 2014	Technical reviewer for *The Design and Implementation of the FreeBSD Operating System (2nd Edition)*, Addison-Wesley, 2014, by Marshall Kirk McKusick, George Neville-Neil, and Robert N.M. Watson.
Summer 2005	Technical reviewer for *Understanding the Linux Kernel*, O'Reilly, 3rd ed., by P. Bovet and M. Cesati
Summer 2005	Technical reviewer for *The Design and Implementation of the FreeBSD Operating System*, Addison-Wesley, 2005, by Marshall Kirk McKusick and George V. Neville-Neil (two chapters)
Summer 2002	Technical reviewer for *Understanding the Linux Kernel*, O'Reilly, 2nd ed., by P. Bovet and M. Cesati
January 1994	Chapter 14 in *UNIX Systems for Modern Architectures* by Curt Schimmel (Addison-Wesley, 1994)

Program Committee Member

2019	The 2019 USENIX Annual Technical Conference (ATC 2019), Renton, WA.
2018	The 10th USENIX Workshop on Hot Topics in Storage and File Systems (HotStorage '18), Boston, MA.
2018	The 9th Annual Non-Volatile Memories Workshop (NVME 2018), San Diego, CA.
2017	The 4th FAST/Linux Summit, Santa Clara, CA.
2016–2017	The 15th USENIX Conference on File and Storage Technologies (FAST '17), Santa Clara, CA.

2016	The 3rd FAST/Linux Summit, Santa Clara, CA.
2015	The 2nd FAST/Linux Summit, Santa Clara, CA.
2014	USENIX 2nd Summit for Educators in System Administration (SESA 2014), Seattle, Washington (co-located with LISA'14).
2014	The 7th Israeli ACM Experimental Systems Conference (ACM SYSTOR 2014), Haifa, Israel.
2014	The 12th USENIX Conference on File and Storage Technologies (FAST '14), Santa Clara, CA.
2014	The 1st FAST/Linux Summit, Santa Clara, CA.
2013	USENIX 1st Summit for Educators in System Administration (SESA 2013), Seattle, Washington (co-located with LISA'13).
2013	Interactions of NVM/Flash with Operating-Systems and Workloads Workshop (INFLOW 2013), Farmington, PA (co-located with SOSP'13).
2013	International Conference on Parallel and Distributed Systems (ICPADS 2013), Seoul, Korea.
2011	Operating System Support for Next Generation Large Scale NVRAM (NVRAMOS'11), Jeju Island, South Korea.
2011	The 4th Israeli Experimental Systems Conference (SYSTOR 2011), Haifa, Israel.
2011	The 27th IEEE Symposium on Massive Storage Systems and Technologies (MSST2011), Research Track. Colorado.
2011	The 2011 USENIX Annual Technical Conference
2010	The 26th IEEE Symposium on Massive Storage Systems and Technologies (MSST2010), Research Track. Incline Village, Nevada.
2010	The 3rd Israeli Experimental Systems Conference (SYSTOR 2010), Haifa, Israel.
2009	The 21st International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD 2009).
2009	The first Usenix workshop on the theory and practice of provenance (TAPP '09).
2009	The 2009 ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS 2009), New York, NY.
2009	The 2nd Israeli Experimental Systems Conference (SYSTOR 2009), Haifa, Israel.
2008	Fifth USENIX Conference on File and Storage Technologies (FAST '08), San Jose, CA.
2007	The Third ACM International Workshop on Storage Security and Survivability (StorageSS 2007), Alexandria, VA
2007	Fifth USENIX Conference on File and Storage Technologies (FAST '07)
2006	CCGrid Cluster Security workshop (in conjunction with CCGrid 2006, the Sixth IEEE/ACM International Symposium on Cluster Computing and the Grid), Singapore, Korea.

2005	The 3rd International IEEE Security in Storage Workshop (co-sponsored by Usenix), San Francisco, CA
2005	The First ACM International Workshop on Storage Security and Survivability (StorageSS), "The Paradigm Shift to Info-Centric Protection," Fairfax, VA
2004–2005	CCGrid Cluster Security workshop (in conjunction with CCGrid 2005, the Fifth IEEE/ACM International Symposium on Cluster Computing and the Grid), Cardiff, UK
2004	Workshop on Information Assurance Education, Stony Brook University (WIAED 2004), organizer
2002	USENIX FreeNIX 2002 Annual Conference
2001	First USENIX Conference on File and Storage Technologies (FAST '02)

Session Chair

Oct 2020	Keynote by Richardo Biancini, the 13th ACM International System and Storage Conference
Jul 2019	Co-chair, ACM Workshop on on Hot Topics in Storage and File Systems (HotStorage), "File Systems"
Jan 2015	The annual NSF Secure and Trustworthy Computing (SaTC) program PI meeting (SaTC-PI 2015), "Benchmarking and workloads"
Feb 2014	The 12th USENIX Conference on File and Storage Technologies (FAST '14), "Performance and Efficiency"
June 2011	The 2011 Usenix Annual Technical Conference (Usenix ATC '11), "Storage Deduplication"
July 2013	The 6th Israeli Experimental Systems Conference (SYSTOR '13), "Flash Optimization"
May 2011	The 4th Israeli Experimental Systems Conference (ACM SYSTOR '11), "Adaptation and performance"
May 2009	The 2nd Israeli Experimental Systems Conference (ACM SYSTOR '09), "Deduplication"
Feb 2009	The first Usenix workshop on the theory and practice of provenance (TAPP '09)
May 2009	Fifth USENIX Conference on File and Storage Technologies (FAST '07), "Beyond the Machine Room"
Nov 2005	ACM International Workshop on Storage Security and Survivability (StorageSS), "Cryptographic Storage Security"
	Usenix Security 2004, Works-in-Progress session chair
Aug 2004	Usenix Security 2004, Works-in-Progress session chair
Aug 2004	Usenix Security 2004, Poster presentations session chair
Jun 2002	USENIX FreeNIX Annual Conference, "File Systems"

Jun 2001	USENIX FreeNIX Annual Conference, “Scripting”

Referee

Feb 2022	USENIX FAST 2022
Nov 2020	ACM Transactions on Storage (TOS)
Oct 2018	IEEE Transactions on Parallel and Distributed Systems (TDPS)
Aug 2017	ACM Transactions on Storage (TOS)
2016	Runtime Verification (RV) 2016
2011	Sustainable Computing Informatics and Systems journal
Nov 2010	ACM SIGMOD 2010
Oct 2010	IEEE Internet Computing
Jan 2010	ACM Transactions on Computers (TOCS)
Dec 2009	IEEE Transactions on Parallel and Distributed Systems (TPDS)
Feb 2009	USENIX FAST 2009
Feb 2007	USENIX Annual conference 2007
Jan 2007	Sigmetrics 2007
Jul 2006	Elsevier’s Information Sciences journal
Feb 2006	USENIX Annual conference 2006
Aug 2005	USENIX FAST 2005
Aug 2004–present	ACM Transactions on Storage (TOS) journal
Jun 2004	USENIX OSDI 2004
Nov 2003	USENIX FAST 2004
Dec 2002	USENIX FREENIX 2003
Jun 2002	USENIX OSDI 2002
May 2002	ACM Mobile Networks and Applications (MONET)
Mar 2002	16th Annual ACM International Conference on Supercomputing (ICS)
Mar 2002	LCTES/SCOPES ’02: ACM SIGPLAN Joint Conference on Languages, Compilers, and Tools for Embedded Systems and Software and Compilers for Embedded Systems.
Jan 2002	USENIX FreeNIX Annual Conference
Mar 2001	8th Workshop on Hot Topics in Operating Systems (HotOS-VIII)
Jan 2001	USENIX Annual Conference
Jan 2001	USENIX FreeNIX Annual Conference
May 2000	USENIX LISA 14
Dec 2000	USENIX Annual Conference

May 1999	USENIX LISA 13
May 1999	ACM Twelfth Annual Symposium on User Interface Software and Technology (UIST '99)
Mar 1996	2nd International Mobile Computing Conference
1994	Trends in Software: Software Configuration Management

Committee Service

2021–present	Tiger Team lead, "Data Storage, Data Management, Distributed Ledger Technologies, Cybersecurity, and Biometrics," Stony Brook University.
2020–present	Steering Committee member, USENIX Annual Technical Conference (USENIX ATC)
2020–present	Steering Committee member, ACM Workshop on on Hot Topics in Storage and File Systems (HotStorage)
2021–2022	Co-Chair, President's "One Campus IT Operations" (OCITO) working group, Stony Brook University.
2017–2020	Member, Provost Committee on Faculty Mentorship, Stony Brook University
2017	Member, ad-hoc SL-2 hiring committee for CS Staff Assistant, Computer Science Department, Stony Brook University
2016–2020	Member, STRIDE project advisory committee, Institute for Advanced Computational Science (IACS) and Computer Science Department, Stony Brook University
2016–2019	Chair, Research and IT/Building Operations committee, Computer Science Department, Stony Brook University (note: also official mentor to all junior CS faculty).
2016-2017	Member, ad-hoc hiring committee for CEAS Grant Writer, College of Engineering and Applied Science, Stony Brook University
2016	Member, ABET funding cmte, College of Engineering and Applied Science, Stony Brook University
2015–present	Steering Committee member, USENIX Filesystem and Storage Technologies Conference (USENIX FAST)
2014	Member, Executive committee, Computer Science Department, Stony Brook University
2014	Member, ad-hoc committee for selection of departmental award, Computer Science Department, Stony Brook University
2014	Member, Provost's Outstanding Lecturer Selection Committee, Stony Brook University
2013	Member, Provost's Interdisciplinary Faculty Cluster Selection Committee, Stony Brook University
2013–present	Undergraduate Curriculum Committee, Computer Science Department, Stony Brook University
2013–present	Member, Provost's Massive Open Online Courses (MOOCS) Task Force, Stony Brook University

2012–present	Steering Committee member, The Israeli ACM Experimental Systems Conference (ACM SYSTOR)
2011–2015	Co-Chair, IT Operations committee, Computer Science Department, Stony Brook University
2011–present	Member, University Senate Information Technology Committee, Stony Brook University
2011–present	Member, President's Information Technology Steering Committee on Email and Collaboration, Stony Brook University
2006–2011	Director of IT Operations, Computer Science Department, Stony Brook University
2010–present	Advanced Energy Research and Technology Center (AERTC), Center Director Search Committee member, Stony Brook University
2012	Member, Grant Writer Recruiting committee, Computer Science Department, Stony Brook University
2011	Staff Recruiting committee chair (Windows Server administrator), Computer Science Department, Stony Brook University
2010	Staff Recruiting committee chair (Unix/Networking administrator), Computer Science Department, Stony Brook University
2010	Departmental 5-year Strategic Planning Committee member, Computer Science Department, Stony Brook University
2010	Faculty Recruiting Committee, Computer Science Department, Stony Brook University
2010	Staff Recruiting committee member (Network Administrator), CEWIT Center, Stony Brook University
2010	President's Teaching Excellence Award Selection Committee, Stony Brook University
2009	Staff Recruiting committee chair (Windows client administrator), Computer Science Department, Stony Brook University
2009	Staff Recruiting committee chair (CVC manager), Computer Science Department, Stony Brook University
2009–present	Advanced Energy Research and Technology Center (AERTC), scientific advisor board member, Stony Brook University
2009	President's Teaching Excellence Award Selection Committee, Stony Brook University
2007–2008	Staff Recruiting committee chair (webmaster), Computer Science Department, Stony Brook University
2008–present	New CS building committee, Computer Science Department, Stony Brook University
2007	Staff Recruiting committee chair (Unix administrator), Computer Science Department, Stony Brook University
2007–2010	Co-Chair, IT Operations committee, Computer Science Department, Stony Brook University

2006–present	Center of Excellence in Wireless and Information Technologies (CEWIT) faculty advisory board member, Computer Science Department, Stony Brook University
2006–2007	Chair, Resources and Facilities sub-committee, part of an 8-year departmental review committee. Computer Science Department, Stony Brook University.
2003–present	Visibility (Ranking) committee, Computer Science Department, Stony Brook University
2003–present	Graduate Committee, Computer Science Department, Stony Brook University
2003–present	Permanent Graduate Quals committee, Computer Science Department, Stony Brook University
2003–present	Center of Excellence in Wireless and Information Technologies (CEWIT) committee, Computer Science Department, Stony Brook University
2002–present	Web Site Design committee, Computer Science Department, Stony Brook University
2002–present	Operations Committee, Computer Science Department, Stony Brook University
2001–2006	Faculty Recruiting Committee, Computer Science Department, Stony Brook University
2002–2006	Graduate Recruiting Poster Committee, Computer Science Department, Stony Brook University
2005	SFS Program Director Recruiting committee, Computer Science Department, Stony Brook University
2004	Staff Recruiting committee, Computer Science Department, Stony Brook University
2004	Undergraduate Curriculum Review ad-hoc committee, Computer Science Department, Stony Brook University
2003	PI Award Interface (PIAI) Review committee, Stony Brook University
2003	Graduate Brochure committee, Computer Science Department, Stony Brook University
Spring 2002	Graduate Student Admissions committee, Computer Science Department, Stony Brook University
2002–2003	Graduate Quals Revisions committee, Computer Science Department, Stony Brook University
Spring 2002	Graduate Student Admissions Committee, Computer Science Department, Stony Brook University
Fall 2001	Undergraduate Curriculum Committee, Computer Science Department, Stony Brook University
Fall 2001	Domestic Graduate Student Recruiting Committee, Computer Science Department, Stony Brook University
Fall 2001	Quality of Life Committee, Computer Science Department, Stony Brook University
2001	Ph.D. and M.S. Recruiting Committee, Computer Science Department, Stony Brook University

1998–2000	Ph.D. Committee, Columbia University
1998–1999	Ph.D. Admissions Committee, Columbia University
1990–1999	Facilities Committee, Columbia University

Ph.D. Dissertation Defense Committee

Dec 2021	*Appearance Aware Visualization of High Dimensional Data*, Jenny Hyunjung Lee, Computer Science Department, Stony Brook University.
Aug 2020	Committee Chair, Muhammad Wajahat, *Cost-Efficient Dynamic Management of Cloud Resources via Supervised Learning*, Computer Science Department, Stony Brook University.
Jul 2019	Committee chair, *Analytical Approaches for Dynamic Scheduling in Cloud Environments*, Seyyed Ahmad Javadi, Computer Science Department, Stony Brook University.
Jan 2019	Adviser, *A Practical Auto-Tuning Framework for Storage Systems*, Zhen Cao, Computer Science Department, Stony Brook University.
Oct 2017	*Optimizing System Software with B^ε Trees*, William Jannen, Computer Science Department, Stony Brook University.
May 2017	Committee Chair, Yifeng Sun, *Protection Mechanisms for Virtual Machines on Virtualized Servers*, Computer Science Department, Stony Brook University.
Apr 2017	Adviser, *Kurma: Efficient and Secure Multi-Cloud Storage Gateways for Network-Attached Storage*, Ming Chen, Computer Science Department, Stony Brook University.
Sep 2014	Committee Chair, *Achieving Regulatory Compliance in Data Management*, Sumit Bajaj, Computer Science Department, Stony Brook University.
May 2014	Adviser, *GreenDM: A Versatile Tiering Hybrid Drive for the Trade-Off Evaluation of Performance, Energy, and Endurance*, Zhichao Li, Computer Science Department, Stony Brook University.
May 2014	Committee Chair, *Efficient Implementation Techniques for Block-Level Cloud Storage Systems* Dilip Simha, Computer Science Department, Stony Brook University.
Dec 2013	Adviser, *Multi-dimensional Workload Analysis and Synthesis for Modern Storage Systems*, Vasily Tarasov, Computer Science Department, Stony Brook University.
May 2012	Committee Chair, *Practical Oblivious Outsourced Data Processing*, Peter Williams, Computer Science Department, Stony Brook University.
Jan 2012	Adviser, *Efficient, Scalable, and Versatile Application and System Transaction Management for Direct Storage Layers*, Richard P. Spillane, Computer Science Department, Stony Brook University.
Aug 2009	*Virtualization Mechanisms for Mobility, Security and System Administration*, Shaya Potter, Computer Science Department, Columbia University.
Aug 2009	Adviser, *Flexible Debugging with Controllable Overhead*, Sean Callanan, Computer Science Department, Stony Brook University.
Sep 2008	Committee chair, *Repairable File and Storage Systems*, Ningning Zhu, Computer Science Department, Stony Brook University.

Aug 2008 — Adviser, *Analyzing Root Causes of Latency Distributions*, Avishay Traeger, Computer Science Department, Stony Brook University.

Aug 2008 — *Programmable Ethernet Switches and Their Applications*, Srikant Sharma, Computer Science Department, Stony Brook University.

May 2008 — Committee chair, *Practical Information Flow Based Techniques to Safeguard Host Integrity"*, Weiqing Sun, Computer Science Department, Stony Brook University.

Feb 2008 — Adviser, *End-to-End Abstractions for Application-Aware Storage*, Gopalan Sivathanu (advisee), Computer Science Department, Stony Brook University.

Dec 2006 — Adviser, *Versatile, Portable, and Efficient File System Profiling*, Nikolai Joukov (advisee), Computer Science Department, Stony Brook University.

Jul 2006 — Committee chair, *Availability, Fairness, and Performance Optimization in Storage Virtualization Systems*, Gang "Jason" Peng, Computer Science Department, Stony Brook University.

Jun 2006 — *Implementation Techniques for Scalable, Secure and QoS-Guaranteed Enterprise-Grade Wireless LANs*, Fanglu Guo, Computer Science Department, Stony Brook University.

May 2006 — Adviser, *Extending ACID Semantics to the File System via* `ptrace`, Charles P. Wright (advisee), Computer Science Department, Stony Brook University.

Aug 2002 — *Authoring, Compression, and Playback of Active Video Content*, Anindya Neogi, Computer Science Department, Stony Brook University.

M.S. Thesis Defense Committee

May 2021 — *Transfer Learning Techniques for Sequence Labeling in Network File System Specifications*, Amanpreet Singh, Computer Science Department, Stony Brook University.

Dec 2016 — Adviser, *To FUSE or not to FUSE? Analysis and Performance Characterization of the FUSE User-Space File System Framework*, Bharath Kumar Reddy Vangoor, Computer Science Department, Stony Brook University.

May 2015 — Adviser, *Finding the Right Balance: Security vs. Performance with Network Storage Systems*, Arun Olappamanna Vasudevan, Computer Science Department, Stony Brook University.

May 2012 — Adviser, *A Context Aware Block Layer: The Case for Block Layer Deduplication*, Amar Mudrankit, Computer Science Department, Stony Brook University.

May 2012 — Adviser, *From Tuples to Files: a Fast Transactional System Store and File System*, Pradeep Shetty, Computer Science Department, Stony Brook University.

December 2010 — *TRACECUT: a mechanism of providing history based pointcuts on top of InterAspect*, Ketan Dixit, Computer Science Department, Stony Brook University.

May 2010 — Adviser, *Stopping Data Races Using Redflag*, Abhinav Duggal, Computer Science Department, Stony Brook University.

May 2010 — Adviser, *Optimizing Energy and Performance for Server-Class File System Workloads*, Priya Sehgal, Computer Science Department, Stony Brook University.

May 2010 — *Model Checking the Kaminsky DNS Cache-Poisoning Attack Using PRISM*, Tushar Sudas Deshpande, Computer Science Department, Stony Brook University.

May 2010 — Committee chair, *LFSM (Log Structured Flash Storage Manager)* Gautham Meruva, Computer Science Department, Stony Brook University.

Apr 2009 — Adviser, *The Visual Development of GCC Plug-ins with GDE* Daniel J. Dean, Computer Science Department, Stony Brook University.

Apr 2009 — Adviser, *Energy and Performance Evaluation of Lossless File Data Compression on Computer Systems* Rachita Kothiyal, Computer Science Department, Stony Brook University.

May 2007 — Adviser, *PLEASE: Policy Language for Easy Administration of SELinux*, David P. Quigley, Computer Science Department, Stony Brook University.

Jun 2006 — *Towards the Minimal TCP Offload Architecture*, Lakshmi Kumar Tiruppukuzhi, Computer Science Department, Stony Brook University.

Jun 2006 — *An Approach to Protect System Integrity from Untrusted Applications*, Gaurav Poothia, Computer Science Department, Stony Brook University.

Dec 2005 — Adviser, *Storage Virtualization with a Stackable File System*, Sunil Satnur, Computer Science Department, Stony Brook University.

Dec 2004 — Adviser, *File System Extensibility and Reliability Using an in-Kernel Databases System*, Aditya Kashyap, Computer Science Department, Stony Brook University.

Aug 2004 — Adviser, *Versatile File System Tracing with Tracefs*, Akshat Aranya, Computer Science Department, Stony Brook University.

Mar 2004 — *System and Network Management Using State Machines*, Mohan-Krishna Channa-Reddy, Computer Science Department, Stony Brook University.

Dec 2003 — Adviser, *Versionfs: A Versatile and User-Oriented Versioning File System*, Kiran-Kumar Muniswamy-Reddy (advisee), Computer Science Department, Stony Brook University.

Aug 2003 — *WattProbe: Software-Based Empirical Extraction of Hardware Energy Models*, Manish Prasad, Computer Science Department, Stony Brook University.

Jun 2003 — Adviser, *A System for Improving Application Performance Through System Call Composition*, Amit Purohit, Computer Science Department, Stony Brook University.

Ph.D. Thesis-Proposal (Prelim) Committee

Jun 2022 — Adviser, *Using Machine Learning to Improve Operating Systems' I/O Subsystems*, Ibrahim "Umit" Akgun, Computer Science Department, Stony Brook University.

Apr 2022 — *Exploring Large Parameter Spaces with Machine Learning and Data Visualization*, Anjul Tyagi, Computer Science Department, Stony Brook University.

Apr 2019 — *Cost-Efficient Dynamic Management of Cloud Resources through Supervised Learning*, Muhammad Wajahat, Computer Science Department, Stony Brook University.

Dec 2018 — *Predicting the Future of Archival Storage Using Simulation*, James Byron, Computer Science Department, UC Santa Cruz.

Nov 2018	Committee chair, *Analytical Approaches for Dynamic Scheduling in Cloud Environments*, Seyyed Ahmad Javadi, Computer Science Department, Stony Brook University.
Jun 2018	Committee chair, *Appearance Aware Visualization of High Dimensional Data*, Jenny Hyunjung Lee, Computer Science Department, Stony Brook University.
Apr 2018	Adviser, *A Practical, Real-Time Auto-Tuning Framework for Storage Systems*, Zhen Cao, Computer Science Department, Stony Brook University.
Apr 2017	*Optimizing System Software with B^ε Trees*, William Jannen, Computer Science Department, Stony Brook University.
Nov 2015	Adviser, *KURMA: Geo-Distributed Secure Middlewares for Cloud-Backed Network Attached Storage*, Ming Chen, Computer Science Department, Stony Brook University.
Dec 2014	*Economic Modeling of Long-Term Digital Storage*, Preeti Gupta, Computer Science Department, the University of California at Santa Cruz.
Oct 2013	*Achieving Regulatory Compliance in Data Management*, Sumit Bajaj, Computer Science Department, Stony Brook University.
Aug 2013	Committee chair, *Implementation Techniques for Software-Defined Distributed Storage Systems*, Dilip Simha, Computer Science Department, Stony Brook University.
May 2013	Adviser, *System-Aware Resource Scheduling for Performance, Energy, and Reliability in Tiered Storage Systems*, Zhichao Li, Computer Science Department, Stony Brook University.
Apr 2013	Adviser, *Multi-dimensional Workload Analysis and Synthesis for Modern Storage Systems*, Vasily Tarasov, Computer Science Department, Stony Brook University.
Jan 2012	Adviser, *Runtime Verification of Kernel-Level Concurrency Using Compiler-Based Instrumentation*, Justin Seyster, Computer Science Department, Stony Brook University.
Aug 2011	Adviser, *Efficient, Scalable, and Versatile Application and System Transaction Management for Direct Storage Layers* Richard P. Spillane, Computer Science Department, Stony Brook University.
May 2011	*Practical Oblivious Outsourced Data Processing* Peter Williams, Computer Science Department, Stony Brook University.
Dec 2008	*Organizing, Indexing, and Searching Large-Scale File Systems* Andrew Leung, Computer Science Department, the University of California at Santa Cruz.
Mar 2008	Adviser, *Remote Debugging with Controllable Overhead* Sean Callanan, Computer Science Department, Stony Brook University.
Feb 2008	Adviser, *Analyzing Root Causes of Latency Distributions*, Avishay Traeger, Computer Science Department, Stony Brook University.
Oct 2007	*Practical Information Flow Based Techniques to Safeguard Host Integrity"*, Weiqing Sun, Computer Science Department, Stony Brook University.
Aug 2007	Committee chair, *OS-level Virtualization and Its Applications*, Yang Yu, Computer Science Department, Stony Brook University.
May 2007	Adviser, *End-to-End Abstractions for Application-Aware Storage*, Gopalan Sivathanu, Computer Science Department, Stony Brook University.

Jun 2006 — *Building Enterprise-Grade Wireless LAN with High Mobility, Capacity and Security*, Fanglu Guo, Computer Science Department, Stony Brook University.

Apr 2006 — Adviser, *Versatile, Portable, and Efficient File System Profiling*, Nikolai Joukov, Computer Science Department, Stony Brook University.

Jan 2006 — Adviser, *Extending ACID Semantics to the File System*, Charles P. Wright, Computer Science Department, Stony Brook University.

Aug 2005 — Committee chair, *Repairable File and Storage Systems*, Ningning Zhu, Computer Science Department, Stony Brook University.

Dec 2004 — Committee chair, *Programmable Ethernet switch networks and their applications*, Srikant Sharma, Computer Science Department, Stony Brook University.

Sep 2004 — Committee chair, *Availability Support and Performance Optimization in Stonehenge*, Gang "Jason" Peng, Computer Science Department, Stony Brook University.

Mar 2002 — *Stonehenge: A High Performance Virtualized Storage Cluster With QoS Guarantees*, Lan Huang, Computer Science Department, Stony Brook University.

Oct 2001 — *A System for Collaborative Web Resource Categorization and Ranking*, Maxim Lifantsev.

May 2001 — *Authoring, Compression, and Playback of Active Video Content*, Anindya Neogi, Computer Science Department, Stony Brook University.

Ph.D. Research Proficiency Exam (RPE) Committee

Apr 2022 — Committee chair, *Optimizing Near-Data Processing for Spark*, Sri Pramodh Rachuri, Computer Science Department, Stony Brook University.

Jan 2022 — Adviser, *Model-Checking Support for File System Development*, Yifei Liu, Computer Science Department, Stony Brook University.

Aug 2020 — *An Experimental Evaluation of Cache-Adaptive Algorithms*, Abiyaz Chowdhury, Computer Science Department, Stony Brook University.

May 2019 — Adviser, *Re-Animator: Versatile High-Fidelity System-Call Tracing and Replaying* Ibrahim "Umit" Akgun, Computer Science Department, Stony Brook University.

Aug 2018 — Committee chair, *Reducing tail latency in web applications by addressing variability in service times*, Amoghavarsha Suresh, Computer Science Department, Stony Brook University.

Apr 2018 — Committee chair, *Realizing an Elastic Memcached via Cached Data Migration*, Ubaid Ullah Hafeez, Computer Science Department, Stony Brook University.

Sep 2016 — Dynamic Interference-Aware Load Balancing, Seyyed Ahmad Javadi, Computer Science Department, Stony Brook University.

May 2016 — Committee chair, *Leveraging Machine Learning for Black-Box Autoscaling*, Muhammad Wajahat, Computer Science Department, Stony Brook University.

Apr 2016 — *Spatial Big Data Management and Analytics on Cloud environments and modern computing infrastructures* Hoang Vo, Computer Science Department, Stony Brook University.

Jan 2016	Adviser, *Parametric Optimization of Storage Systems* Zhen Cao, Computer Science Department, Stony Brook University.
Dec 2015	Adviser, *Design and Implementation of an Open-Source Deduplication Platform for Research* Sonam Mandal, Computer Science Department, Stony Brook University.
Sep 2012	*The Design and Application of Content Addressable Storage Systems*, William Jannen Computer Science Department, Stony Brook University.
Aug 2012	Committee chair, *Interaction for Visualization of Multidimensional Data*, Puripant Ruchikachorn, Computer Science Department, Stony Brook University.
Nov 2011	Committee chair, *Trusted Hardware in Secure Data Management*, Sumit Bajaj, Computer Science Department, Stony Brook University.
Sep 2011	Committee chair, *Isolated and Efficient Execution of Unmodified Network Device Drivers*, Yifeng Sun, Computer Science Department, Stony Brook University.
Sep 2011	Committee chair, *The Art of Data Deduplication*, DilipSimha N M, Computer Science Department, Stony Brook University.
Sep 2011	Adviser, *Power and Performance in Compression Systems: A Control Theoretical Approach with Evaluation*, Zhichao Li, Computer Science Department, Stony Brook University.
Sep 2011	Committee chair, *Isolated and Efficient Execution of Unmodified Network Device Drivers*, Sun Yifeng, Computer Science Department, Stony Brook University.
Sep 2011	*Defensive Techniques for Untrustworthy Operating Systems*, Committee chair, Rui Qiao, Computer Science Department, Stony Brook University.
Aug 2011	*Computational Modeling and Analysis of Cardiac Arrhythmia*, Abhishek Murthy, Computer Science Department, Stony Brook University.
Feb 2010	Adviser, *An End-to-End Performance Evaluation of NFSv4 Implementations*, Vasily Tarasov, Computer Science Department, Stony Brook University.
Aug 2009	*Attacks and Defenses: Unix File-System race conditions*, Xiang Cai, Computer Science Department, Stony Brook University.
Mar 2008	Adviser, *Techniques for Visualizing Software Execution*, Justin Seyster, Computer Science Department, Stony Brook University.
May 2006	Committee chair, *Challenges of Long-Term Digital Archiving*, Maohua Lu, Computer Science Department, Stony Brook University.
Feb 2006	*A Survey of Program Transformation Languages and Systems*, Michael Gorbovitski, Computer Science Department, Stony Brook University.
Jun 2005	*Peer to Peer File Download and Streaming*, Gang Wu, Computer Science Department, Stony Brook University.
May 2005	Adviser, *Debugging and Optimizing Systems With Runtime Instrumentation*, Sean Callanan, Computer Science Department, Stony Brook University.
May 2005	Adviser, *File System Benchmarking: Fallacies and Pitfalls*, Avishay Traeger, Computer Science Department, Stony Brook University.

May 2005 — Adviser, *On the Role of Static Analysis in Operating System Checking and Runtime Verification*, Abhishek Rai, Computer Science Department, Stony Brook University.

May 2005 — Adviser, *Ensuring Data Integrity in Storage: Techniques and Applications*, Gopalan Sivathanu, Computer Science Department, Stony Brook University.

Feb 2005 — *A Survey on Virtualization Techniques*, Susanta Nanda, Computer Science Department, Stony Brook University.

Sep 2004 — *Enterprise Digital Rights Management: Solutions against Information Theft by Insiders*, Yang Yu, Computer Science Department, Stony Brook University.

May 2004 — Adviser, *Operating System Support for Extensible Secure File Systems*, Charles P. Wright, Computer Science Department, Stony Brook University.

Feb 2004 — *Traffic Analysis: From Stateful Firewall to Network Intrusion Detection System*, Fanglu Guo, Computer Science Department, Stony Brook University.

Feb 2004 — *Damage Recovery from Malicious Code and Operator Errors*, Weiqing Sun, Computer Science Department, Stony Brook University.

Sep 2003 — Adviser, *Internet Worms as Internet-Wide Threat*, Nikolai Joukov, Computer Science Department, Stony Brook University.

Mar 2003 — *Cache Oblivious Algorithms: Theory and Practice*, Piyush Kumar, Computer Science Department, Stony Brook University.

Jan 2003 — Committee chair, *CDN: Content Distribution Network*, Gang "Jason" Peng, Computer Science Department, Stony Brook University.

Aug 2001 — *Pattern Based Intrusion Detection Systems*, Prem Uppuluri, Computer Science Department, Stony Brook University.

Other One-Time Committee Service

2019 — Office of Vice-President for Research, Seed Grant review committee Bi-annual Awards committee, Stony Brook University.

Aug 2012 — Bi-annual Awards committee, Computer Science Department, Stony Brook University.

Nov 2011 — Member, Steering Committee on High-Performance Computing, Stony Brook University and Brookhaven National Labs.

Oct 2010 — Annual Merit Raise committee, Computer Science Department, Stony Brook University.

Aug 2010 — Bi-annual Awards committee, Computer Science Department, Stony Brook University.

Oct 2009 — Annual Merit Raise committee, Computer Science Department, Stony Brook University.

Sep 2008 — Annual Merit Raise committee, Computer Science Department, Stony Brook University.

Sep 2007 — Annual Merit Raise committee, Computer Science Department, Stony Brook University.

Jan 2007 — Examiner for CS Ph.D. Qualifying Exams, Computer Science Department, Stony Brook University.

Jul 2006	Bi-annual Awards committee, Computer Science Department, Stony Brook University.
Apr 2006	Judge, *Compilers* session. Graduate Research Conference (GRC), Computer Science Department, Stony Brook University.
Sep 2005	Annual Merit Raise committee, Computer Science Department, Stony Brook University.
Aug 2005	Ad-hoc Steering Committee on Strategic Planning, Computer Science Department, Stony Brook University.
May 2004	Judge, *Security* session. Graduate Research Conference (GRC), Computer Science Department, Stony Brook University.
May 2004	Examiner for CS Ph.D. Qualifying Exams, Computer Science Department, Stony Brook University.
Jan 2004	Examiner for CS Ph.D. Qualifying Exams, Computer Science Department, Stony Brook University.
May 2003	Judge, *Security* session. Graduate Research Conference (GRC), Computer Science Department, Stony Brook University.
Oct 2003	Ph.D. qualifying exam chair (Hardware exam), Computer Science Department, Stony Brook University.
Oct 2003	Examiner for CS Ph.D. Qualifying Exams, Computer Science Department, Stony Brook University.
May 2003	Examiner for CS Ph.D. Qualifying Exams, Computer Science Department, Stony Brook University.
May 2002	Examiner for CS Ph.D. Qualifying Exams, Computer Science Department, Stony Brook University.
May 2001	Examiner for CS Ph.D. Qualifying Exams, Computer Science Department, Stony Brook University.

Other Departmental Service

2008–present	Course Coordinator, Systems Administration, (ISE-311)
2003–present	Course Coordinator, Introduction to Programming in C (CSE-130)
2002–present	Course Coordinator, Advanced Systems Programming in Unix/C (CSE-376)
2001–present	Course Coordinator, Operating Systems (CSE-506)
2003–2005	Course Coordinator, Intermediate Programming in C and C++ (CSE-230)

Community Work

Nov 2008	"How Computers Work," guest speaker at Mount Elementary 4th grade (teacher: Jo-Ann Hudecek), Stony Brook Three Village School District, Stony Brook, NY.
2003	Stony Brook Child Care Services Inc. (SBCCSI), fund-raising committee member
2002–present	Faculty sponsor of the Stony Brook Linux Users Group (SBLUG)
1997–present	Maintainer of FiST, a stackable file system template system

1992–present	Maintainer of Amd, the Berkeley Automounter (am-utils)

Shepherding

2016–2017	The 15th USENIX Conference on File and Storage Technologies (FAST '17) (1 paper)
2011	The 2011 Usenix Annual Technical Conference (Usenix ATC 2011) (1 paper)
2009	The 2009 ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS 2009) (2 papers)
2009	Fifth USENIX Conference on File and Storage Technologies (FAST '09) (1 paper)
2007	USENIX FAST Annual Conference (2 papers)
2003	USENIX FreeNIX Annual Conference (2 papers)
2002	USENIX FreeNIX Annual Conference (4 papers)
2001	First USENIX Conference on File and Storage Technologies (FAST '02) (1 paper)

Software Released or Maintained

2010–present	Wrapfs Stackable File System for Linux
2007–present	GCC Plugin Infrastructure.
2005–present	Auto-Pilot: A Suite of Tools for Conducting Benchmarks, `www.filesystems.org/project-autopilot.html`
2004–present	Unionfs: A Stackable Unification File System, `https://unionfs.filesystems.org`. (As of January 2007, Unionfs has been a part of Andrew Morton's Linux Kernel source tree. Andrew is the person in charge of supplying new patches to Linus Torvalds for new kernel releases, therefore Unionfs is on its way to becoming a part of the Linux kernel.)
1997–present	FiST: Stackable File System Language and Templates, `www.filesystems.org/`
1992–present	Am-Utils: The Berkeley Automounter Suite of Utilities, `www.am-utils.org`
2004–2005	I^{3}FS: In-Kernel Integrity Checker and Intrusion Detection File System, `www.fsl.cs.sunysb.edu/project-i3fs.html`
2003–2004	KBDB: In-Kernel Berkeley DB Databases NFScrack: NFS File Handle Security, KBDBFS: File System Extensibility and Reliability Using an in-Kernel Database, `www.fsl.cs.sunysb.edu/project-kbdb.html`
2002	ENF: Enhancing NFS Cross-Administrative Domain Access, `ftp://ftp.fsl.cs.sunysb.edu/pub/enf/`
1999	PGmake: Parallel GNU Make, `www.fsl.cs.sunysb.edu/project-pgmake.html`

TALKS

Invited Talks

May 2017	*Oh, what a tangled web we weave: practice and deception in system analysis*, Performance Evaluation and Analysis Meetup, IBM Research–Haifa, Haifa, Israel.
May 2017	*vNFS: Maximizing NFS Performance with Compounds and Vectorized I/O*, ACM SYSTOR 2017 (highlights paper track), Haifa, Israel.

Feb 2017	*Oh, what a tangled web we weave: practice and deception in system analysis*, Tintri, Inc., Mountain View, CA.
Dec 2016	*Oh, what a tangled web we weave: practice and deception in system analysis*, Harvey Mudd College, Claremont, CA.
Jun 2016	*Oh, what a tangled web we weave: practice and deception in system analysis* Keynote address at inaugural workshop: The 1st International Workshop on System Analytics and Characterization (SAC'16, co-located with ACM SIGMETRICS), Antibes Juan-Les-Pins, France.
Feb 2016	*Parametric Optimization of Storage Systems*, VMware Inc., Palo Alto, CA.
Feb 2006	*File System and Storage Benchmarking: On Formalizing Filebench Workloads*, a Birds-of-Feather (BoF) talk presented at the 2016 Usenix FAST conference, Santa Clara, CA.
Jun 2012	*Storage Security Overview*, CA/CEWIT center inauguration workshop, Tel Aviv University, Tel Aviv, Israel.
May 2012	*Synthesizing Realistic Workload Models and Dedup Datasets*, IBM Haifa Research Labs, Tel Aviv, Israel.
May 2012	*Optimizing Performance and Energy Use of Large Data Systems*, NSF Industry/University Center (I/UCRC) for Dynamic Data Analytics (CDDA) 2012 workshop, Stony Brook, NY.
Feb 2012	*Scaling OS Storage Stack Performance Using NVRAM Technologies*, Fusion-io, San Jose, CA
Nov 2011	*Scaling and Understanding Large Data I/O Performance*, NSF Industry/University Center (I/UCRC) for Dynamic Data Analytics (CDDA) 2011 workshop, Rutgers, NJ.
Nov 2011	*Scaling I/O Performance for Large Data Sets*, 2011 CEWIT Center Conference, Hauppauge, NY.
Nov 2011	*Scaling OS Storage Stack Performance Using NVRAM Technologies*, Operating System Support for Next Generation Large Scale NVRAM (NVRAMOS'11), Jeju Island, South Korea.
Aug 2011	*Optimizing Server Performance and Energy*, High-End Computing Filesystems and I/O Workshop (HEC FSIO 2011), Arlington, VA.
May 2011	*On the Energy Consumption and Performance of Systems Software*, The 4th Israeli Experimental Systems Conference (ACM SYSTOR '11), Haifa, Israel.
Feb 2011	*FSL Storage Research Overview*, Riverbed, Inc, Menlo Park, CA.
Nov 2010	*Server-Class Energy and Performance Evaluations, or, Mommy, Daddy, where did my Joules go?*, An SBCS Faculty Symposium series talk, sponsored by the Stony Brook Computer Society's ACM Chapter. Stony Brook University, Computer Science Department, Stony Brook, NY.
May 2010	*Server-Class Energy and Performance Evaluations*, The Third Israeli Experimental Systems Conference (ACM SYSTOR '10), Haifa, Israel.

Feb 2010	*On the Science of Power Management: Encouraging Sustainability R&D*, First USENIX Workshop on Sustainable Information Technology (SustainIT'10), San Jose, CA.
Feb 2010	*FSL Research Overview*, Data Domain, Inc, Santa Clara, CA.
Nov 2009	Priya Sehgal and Erez Zadok, *Improving Energy & Performance For Server Workloads*, The third Advanced Energy Research & Technology Center (AERTC) conference, Hauppauge, NY.
July 2009	*Energy and Performance Characterization of Storage Stacks*, IBM Watson Research Labs, Hawthorne, NY.
May 2009	*DHIS: Discriminating Hierarchical Storage*, The Israeli Experimental Systems Conference (ACM SYSTOR '09), Haifa, Israel.
May 2009	*Energy and Performance Evaluation of Lossless File Data Compression on Server Systems*, The Israeli Experimental Systems Conference (ACM SYSTOR '09), Haifa, Israel.
February 2009	*How to Cheat in Benchmarking*, a Birds-of-Feather (BoF) talk presented at the 2009 Usenix FAST conference, San Francisco, CA.
Nov 2008	*Mommy, Daddy, Why is My New Laptop So Dog Slow?!*, Parents' Day lecture, Stony Brook University, Stony Brook, NY.
May 2007	*Why is my new laptop so dog slow?!*, An SBCS Faculty Symposium series talk, sponsored by the Stony Brook Computer Society's ACM Chapter. Stony Brook University, Computer Science Department, Stony Brook, NY.
July 2006	*Securing Applications using Operating System Transactions*, Center for Information Protection, NSF I/UCRC organizational workshop, Stony Brook, NY.
June 2006	*Adding security to file systems, one layer at a time*, IBM's Storage Systems Strategy meeting, Mountain View, CA.
June 2006	*File System Benchmarking: Fallacies, Pitfalls, and Beyond*, VMware, Inc., Palo Alto, CA.
December 2005	*File System Benchmarking and Tools: Past, Present, and Future*, a Birds-of-Feather (BoF) talk presented at the 2006 Usenix FAST conference, San Francisco, CA.
December 2005	*File System Benchmarking: Fallacies, Pitfalls, and Beyond*, Penn State University, Computer Science Department, State College, PA.
October 2005	*File System Benchmarking: Fallacies, Pitfalls, and Beyond*, IBM Watson Research Labs, Hawthorne, NY.
October 2005	*File System Benchmarking: Fallacies, Pitfalls, and Beyond*, CSE-600 talk, Stony Brook University, Computer Science Department, Stony Brook, NY.
September 2005	*File System Benchmarking: Fallacies, Pitfalls, and Beyond*, UC Santa Cruz, Computer Science Department, Santa Cruz, California.
June 2005	*Adding security to file systems, one layer at a time*, Second Security Summit, Storage Networking Industry Association (SNIA), Pittsburgh, Pennsylvania
May 2005	*File System Security*, 35th Anniversary Celebration, Stony Brook University, Computer Science Department (CS@35), Stony Brook, New York

April 2005	*Efficient and Safe Execution of User-Level Code in the Kernel*, the 2005 NSF Next Generation Software Workshop, in conjunction with the 2005 International Parallel and Distributed Processing Symposium (IPDPS), Denver, Colorado
October 2004	*File System and OS Security*, talk given to Korean delegation led by Dr. Ho, Stony Brook University, Stony Brook, New York
October 2004	*File System and OS Security*, Computer Associates, Islandia, New York
June 2003	*ASK: Active Spam Killer*, Annual Usenix Conference, FREENIX Track, San Antonio, Texas
February 2003	*File System Security*, New York State Cyber-Security Symposium, Utica, New York
April 2000	*FiST: A System for File System Code Generation*, Computer Science Department, University of Chicago
March 2000	*FiST: A System for File System Code Generation*, Computer Science Department, University of Rochester
March 2000	*FiST: A System for File System Code Generation*, Computer Science Department, Stony Brook University
March 2000	*FiST: A System for File System Code Generation*, Computer Science Department, University of California and Santa Barbara
March 2000	*FiST: A System for File System Code Generation*, Computer Science Department, Penn State University
February 2000	*FiST: A System for File System Code Generation*, Computer Science Department, Bell Labs, New Jersey

Interviews

November 4, 2005	LIA magazine, with Ellen Sterling; what small businesses can do to protect themselves.
June 29, 2001	Dr. Dobb's Technetcast, with Phillippe Lourier; my NFS and Automounters book, status of Linux file system development.

Other Presentations

Mar 2012	Erez Zadok, Steering Committee Town Hall meeting, presenting Google Apps for Education, Stony Brook University.
Feb 2012	Daniel C. Rosenthal, University of California, Santa Cruz; David S. H. Rosenthal, Stanford University Libraries; Ethan L. Miller and Ian F. Adams, University of California, Santa Cruz; Mark W. Storer, NetApp; Erez Zadok, Stony Brook University. *Toward an Economic Model of Long-Term Storage* WIP presented at the *Tenth USENIX Conference on File and Storage Technologies (FAST '12)*, San Jose, CA.
Feb 2012	Zhichao Li, Stony Brook University; Kevin M. Greenan and Andrew W. Leung, EMC Corporation; Erez Zadok, Stony Brook University. *Power Consumption in Enterprise-Scale Backup Storage Systems* Poster presented at the *Tenth USENIX Conference on File and Storage Technologies (FAST '12)*, San Jose, CA.
Feb 2012	V. Tarasov and S. Kumar, Stony Brook University; J. Ma, Harvey Mudd College; D. Hildebrand and A. Povzner, IBM Almaden Research; G. Kuenning, Harvey Mudd College; E. Zadok, Stony Brook University *Extracting Flexible, Replayable Models from*

Large Block Traces Poster presented at the *Tenth USENIX Conference on File and Storage Technologies (FAST '12)*, San Jose, CA.

Nov 2011 — Zhichao Li, R. Grosu, S. A. Smolka, S. D. Stoller, and E. Zadok, Stony Brook University. *Power and Performance in Compression Systems: A Control Theoretical Approach with Evaluation* Poster presented at the *Eighth CEWIT conference (CEWIT '11)*, Hauppauge, NY.

Oct 2011 — Zhichao Li, R. Grosu, S. A. Smolka, S. D. Stoller, and E. Zadok, Stony Brook University. *Power and Performance in Compression Systems: A Control Theoretical Approach with Evaluation* Poster presented at the *Fifth AERTC conference (AERTC '11)*, Buffalo, NY.

May 2011 — Vasily Tarasov, Koundinya Santhosh Kumar, and Erez Zadok, Stony Brook University; Geoff Kuenning, Harvey Mudd College. *T2M: Converting I/O Traces to Workload Models*, Poster presented at the *13th USENIX Workshop on Hot Optics in Operating Systems (HotOS '11)*, Napa, CA.

Feb 2011 — Vasily Tarasov, Koundinya Santhosh Kumar, and Erez Zadok, Stony Brook University; Geoff Kuenning, Harvey Mudd College. *T2M: Converting I/O Traces to Workload Models*, WIP and Poster presented at the *Ninth USENIX Conference on File and Storage Technologies (FAST '11)*, San Jose, CA.

Feb 2011 — Erez Zadok, Stony Brook University and Geoff Kuenning, Harvey Mudd College. *Benchmarking and Tracing: New Horizons*, Birds-of-a-feather (BOF) presented at the *Ninth USENIX Conference on File and Storage Technologies (FAST '11)*, San Jose, CA.

Oct 2011 — Justin Seyster, Abhinav Duggal, Prabakar Radhakrishnan, Scott D. Stoller, and Erez Zadok, Stony Brook University. *Redflag: Detailed Runtime Analysis of Kernel-Level Concurrency*, Poster presented at the *9th USENIX Symposium on Operating Systems Design and Implementations (OSDI '10)*, Vancouver, BC, Canada.

Feb 2010 — Vasily Tarasov, Sujay Godbole, and Erez Zadok. *NSFv4 Implementations: Who Performs Better, When, and Why*, poster presented at the *Eighth USENIX Conference on File and Storage Technologies (FAST '10)*, San Jose, CA.

Feb 2010 — Rick Spillane and Erez Zadok. *Down with the VFS and His Insidious Caches!*, WIP presented at the *Eighth USENIX Conference on File and Storage Technologies (FAST '10)*, San Jose, CA.

Jan 2010 — Erez Zadok and Radu Sion. *Energy and Cost Analysis of Cloud Computing*, NSF Industry/University Center (I/UCRC) for Dynamic Data Analytics (CDDA) planning workshop, New York, NY.

Oct 2009 — Priya Sehgal, Vasily Tarasov, and Erez Zadok. *Server Energy and Performance Evaluations*, poster presented at the *2009 CEWIT Center Conference*, Islandia, NY.

Dec 2008 — Rick Spillane, Chaitanya Yalamanchili, Sachin Gaikwad, Manjunath Chinni, and Erez Zadok. *Honor: A Serializing On-Disk Writeback Buffer*. WIP presented at the *Eighth Symposium on Operating Systems Design and Implementation (OSDI '08)*, San Diego, CA.

Mar 2007 — D. Dean, S. Callanan, and E. Zadok. *Developing GCC Transformations Faster with*

Gimple Viz Graduate Research Conference (GRC), Computer Science Department, Stony Brook University.

Mar 2007 — J. Seyster, S. Callanan, R. Grosu, S. Smolka, and E. Zadok. *Memcov A High Performance Memory Profiling Tool*. Graduate Research Conference (GRC), Computer Science Department, Stony Brook University.

Mar 2007 — R. Spillane and E. Zadok. *A Lot for a Little: Transactional File Systems*. Graduate Research Conference (GRC), Computer Science Department, Stony Brook University.

Apr 2006 — R. Spillane, C. P. Wright, and E. Zadok, *Goanna* (Best Paper Award), Graduate Research Conference (GRC), Computer Science Department, Stony Brook University.

Apr 2006 — N. Joukov, H. Papaxenopoulos, and E. Zadok, *Secure Deletion Myths, Issues, and Solutions*. Graduate Research Conference (GRC), Computer Science Department, Stony Brook University.

Sep 2005 — E. Zadok, *A Layered Approach to Securing File Systems*, poster presented at the NSF CyberTrust PI meeting, Newport Beach, CA.

Oct 2005 — N Joukov, R. Iyer, A. Traeger, C. P. Wright, and E. Zadok. Versatile, Portable, and Efficient OS Profiling via Latency Analysis. Poster presented at the *20th ACM Symposium on Operating Systems Principles (SOSP'05)*, Brighton, UK.

Jul 2005 — D. Quigley, C. P. Wright, and E. Zadok. *UnionFS: Namespace Management*. Cyber Corps Summer Symposium 2005, Syracuse, New York.

Apr 2005 — D. Quigley, C. P. Wright, and E. Zadok. *UnionFS*. URECA (undergraduate research) presentation at Stony Brook.

Aug 2004 — Y. Miretskiy, A. Das, C. P. Wright, and E. Zadok. Avfs: An On-Access Anti-Virus File System. Poster presented at the *Workshop on Information Assurance Education* (WIAED 2004), Stony Brook University

Aug 2004 — C. P. Wright, M. C. Martino, and E. Zadok. NCryptfs: A Secure and Convenient Cryptographic File System. Poster presented at the *Workshop on Information Assurance Education* (WIAED 2004), Stony Brook University

May 2004 — P. Gupta, H. Krishnan, C. P. Wright, and E. Zadok. Versatility and Unix Semantics in a Fan-Out Unification File System. Graduate Research Conference (GRC), Computer Science Department, Stony Brook University.

May 2004 — A. Kashyap, J. Dave, H. Krishnan, C. P. Wright, M. Nayyer Zubair, and E. Zadok. Using Berkeley DB in the Linux Kernel. Graduate Research Conference (GRC), Computer Science Department, Stony Brook University.

May 2004 — Y. Miretskiy, A. Das, C. P. Wright, and E. Zadok. Avfs: An On-Access Anti-Virus File System (Best Paper Award). Graduate Research Conference (GRC), Computer Science Department, Stony Brook University.

May 2004 — A. Aranya, C. P. Wright, and E. Zadok Tracefs: A File System to Trace Them All. Graduate Research Conference (GRC), Computer Science Department, Stony Brook University

Apr 2004 — C. P. Wright and E. Zadok. A Stackable Smorgasbord. Poster presented at the *Third USENIX Conference on File and Storage Technologies (FAST '04)*, San Francisco, CA.

Apr 2004	C. P. Wright and E. Zadok. *Avfs: An On-Access Anti-Virus File System* WIP presented at the *Third USENIX Conference on File and Storage Technologies (FAST '04)*, San Francisco, CA.
Apr 2004	K. Muniswamy-Reddy, and E. Zadok. Elastic Quotas. Poster presented at the *12th NASA Goddard, 21st IEEE Conference on Mass Storage Systems and Technologies (MSST 2004)*, College Park, MD.
Nov 2003	C. P. Wright and E. Zadok. Secure and Mobile File Systems. Poster presented at the *CEWIT 2003* conference, Woodbury, NY.
June 2003	P. Gupta, H. Krishnan, K. Muniswamy-Reddy, C. P. Wright, and E. Zadok. Unionfs: A Flexible, and Portable Unioning File System. WIP presented at the *Annual USENIX Technical Conference*, San Antonio, TX.
June 2003	A. Himmer, K. Muniswamy-Reddy, C. P. Wright, and E. Zadok. Versionfs: A Transparent and Portable Versioning File System. WIP presented at the *Annual USENIX Technical Conference*, San Antonio, TX.
May 2003	A. Himmer and E. Zadok. Versionfs: A Transparent and Portable Versioning File System. Graduate Research Conference (GRC), Computer Science Department, Stony Brook University.
May 2003	E. Zadok, J. Osborn, A. Shater, C. P. Wright, K. Muniswamy-Reddy, and J. Nieh. Elastic Quotas: Stretch Your Disk Space. Graduate Research Conference (GRC), Computer Science Department, Stony Brook University.
May 2003	C. P. Wright, M. C. Martino, and E. Zadok. NCryptfs: A Secure and Convenient Cryptographic File System. Graduate Research Conference (GRC), Computer Science Department, Stony Brook University.
May 2003	A. Purohit, C. P. Wright, J. Spadavecchia, and E. Zadok. Cosy: Develop in User-Land, Run in Kernel-Mode. Graduate Research Conference (GRC), Computer Science Department, Stony Brook University.
April 2003	M. Prasad and E. Zadok. Transparent Page Cache Coherence Support for Linux-based Stackable File Systems. WIP presented at the *Second USENIX Conference on File and Storage Technologies (FAST '03)*, San Francisco, CA.
Fall 2002	E. Zadok. Linux Kernel Internals. Guest lecture in CSE-306.
December 2002	A. Purohit, J. Spadavecchia, C. Wright, and E. Zadok. Improving Application Performance Through System Call Composition WIP presented at the *Fifth Symposium on Operating Systems Design and Implementation (OSDI '02)*, Boston, MA.
December 2002	C. Wright, M. Martino, and E. Zadok. NCryptfs: A Secure and Convenient Cryptographic File System. WIP presented at the *Fifth Symposium on Operating Systems Design and Implementation (OSDI '02)*, Boston, MA.
December 2002	E. Zadok, J. Osborn, C. P. Wright, K. Muniswamy-Reddy. Elastic Quotas. WIP presented at the *Fifth Symposium on Operating Systems Design and Implementation (OSDI '02)*, Boston, MA.
June 2002	A. Purohit, J. Spadavecchia, C. Wright, and E. Zadok. Improving Application Performance Through System Call Composition. WIP presented at the *Annual USENIX Technical Conference*, Monterey, CA.

June 2002 — O. C. Leonard, J. Nieh, E. Zadok, J. Osborn, A. Shater, and C. Wright. The Design and Implementation of Elastic Quotas: A System for Flexible File System Management. WIP presented at the *Annual USENIX Technical Conference*, Monterey, CA.

January 2002 — J. Spadavecchia and E. Zadok. Enhancing NFS Cross-Administrative Domain Access. WIP presented at the *First USENIX Conference on File and Storage Technologies (FAST '02)*, Monterey, CA.

December 1999 — E. Zadok and J. Nieh. FiST: A Language for Stackable File Systems. Poster presented at the *17th ACM Symposium on Operating Systems Principles (SOSP)*, Kiawah Island, SC.

TEACHING

Instructor

Spring 2022 — Research in Computer Science (CSE-487), Computer Science Department, Stony Brook University

Spring 2022 — Advanced Systems Programming in Unix/C (CSE-376), Computer Science Department, Stony Brook University

Spring 2022 — Advanced Project in Computer Science I (CSE-523), Computer Science Department, Stony Brook University

Spring 2022 — Advanced Project in Computer Science I (CSE-524), Computer Science Department, Stony Brook University

Spring 2022 — Graduate Practicum in Teaching (CSE-698), Computer Science Department, Stony Brook University

Spring 2022 — Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University

Fall 2021 — Advanced Project in Computer Science I (CSE-523), Computer Science Department, Stony Brook University

Fall 2021 — Advanced Project in Computer Science II (CSE-524), Computer Science Department, Stony Brook University

Fall 2021 — Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University

Fall 2021 — M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University

Fall 2021 — Seminar in Concurrency (CSE-643), Computer Science Department, Stony Brook University

Fall 2021 — Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University

Summer 2021 — M.S. Internship in Research (CSE-596), Computer Science Department, Stony Brook University

Summer 2021 — Internship in Research (CSE-696), Computer Science Department, Stony Brook University

Spring 2021	Advanced Systems Programming in Unix/C (CSE-376), Computer Science Department, Stony Brook University
Spring 2021	Advanced Project in Computer Science I (CSE-523), Computer Science Department, Stony Brook University
Spring 2021	Advanced Project in Computer Science I (CSE-524), Computer Science Department, Stony Brook University
Spring 2021	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Spring 2021	Graduate Practicum in Teaching (CSE-698), Computer Science Department, Stony Brook University
Spring 2021	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Fall 2020	Research in Computer Science (CSE-487), Computer Science Department, Stony Brook University
Fall 2020	Senior Honors Research Project II (CSE-499), Computer Science Department, Stony Brook University
Fall 2020	Advanced Project in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Fall 2020	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Fall 2020	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Summer 2020	Research in Computer Science (CSE-487), Computer Science Department, Stony Brook University
Summer 2020	Senior Honors Research Project I (CSE-495), Computer Science Department, Stony Brook University
Summer 2020	M.S. Internship in Research (CSE-596), Computer Science Department, Stony Brook University
Spring 2020	Advanced Systems Programming in Unix/C (CSE-376), Computer Science Department, Stony Brook University
Spring 2020	Advanced Project in Computer Science I (CSE-523), Computer Science Department, Stony Brook University
Spring 2020	Advanced Project in Computer Science I (CSE-524), Computer Science Department, Stony Brook University
Spring 2020	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Spring 2020	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University

Fall 2019	Advanced Project in Computer Science I (CSE-523), Computer Science Department, Stony Brook University
Fall 2019	Advanced Project in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Fall 2019	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Fall 2019	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Summer 2019	M.S. Internship in Research (CSE-596), Computer Science Department, Stony Brook University
Summer 2019	Internship in Research (CSE-696), Computer Science Department, Stony Brook University
Spring 2019	Graduate Level *Operating Systems* (CSE-506), Computer Science Department, Stony Brook University
Spring 2019	Advanced Project in Computer Science I (CSE-523), Computer Science Department, Stony Brook University
Spring 2019	Advanced Project in Computer Science I (CSE-524), Computer Science Department, Stony Brook University
Spring 2019	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Spring 2019	Graduate Practicum in Teaching (CSE-698), Computer Science Department, Stony Brook University
Winter 2019	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Fall 2018	Research in Computer Science (CSE-487), Computer Science Department, Stony Brook University
Fall 2018	Advanced Project in Computer Science I (CSE-523), Computer Science Department, Stony Brook University
Fall 2018	Advanced Project in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Fall 2018	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Fall 2018	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Summer 2018	Advanced Project in Computer Science I (CSE-523), Computer Science Department, Stony Brook University
Summer 2018	M.S. Internship in Research (CSE-596), Computer Science Department, Stony Brook University

Summer 2018	Internship in Research (CSE-696), Computer Science Department, Stony Brook University
Spring 2018	Research in Computer Science (CSE-487), Computer Science Department, Stony Brook University
Spring 2018	Graduate Level *Operating Systems* (CSE-506), Computer Science Department, Stony Brook University
Spring 2018	Advanced Project in Computer Science I (CSE-523), Computer Science Department, Stony Brook University
Spring 2018	Advanced Project in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Spring 2018	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Spring 2018	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Fall 2017	Advanced Project in Computer Science I (CSE-523), Computer Science Department, Stony Brook University
Fall 2017	Advanced Project in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Fall 2017	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Summer 2017	Advanced Project in Computer Science I (CSE-523), Computer Science Department, Stony Brook University
Summer 2017	M.S. Internship in Research (CSE-596), Computer Science Department, Stony Brook University
Summer 2017	Internship in Research (CSE-696), Computer Science Department, Stony Brook University
Spring 2017	Advanced Systems Programming in Unix/C (CSE-376), Computer Science Department, Stony Brook University
Spring 2017	Advanced Project in Computer Science I (CSE-523), Computer Science Department, Stony Brook University
Spring 2017	Advanced Project in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Spring 2017	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Fall 2016	Graduate Level *Operating Systems* (CSE-506), Computer Science Department, Stony Brook University
Fall 2016	Advanced Project in Computer Science I (CSE-523), Computer Science Department, Stony Brook University

Fall 2016	Advanced Project in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Fall 2016	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Fall 2016	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University
Fall 2016	Graduate Practicum in Teaching (CSE-698), Computer Science Department, Stony Brook University
Fall 2016	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Summer 2016	M.S. Internship in Research (CSE-596), Computer Science Department, Stony Brook University
Summer 2016	Internship in Research (CSE-696), Computer Science Department, Stony Brook University
Spring 2016	Advanced Systems Programming in Unix/C (CSE-376), Computer Science Department, Stony Brook University
Spring 2016	Research in Computer Science (CSE-487), Computer Science Department, Stony Brook University
Spring 2016	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University
Spring 2016	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Spring 2016	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University
Spring 2016	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Fall 2015	Graduate Level *Operating Systems* (CSE-506), Computer Science Department, Stony Brook University
Fall 2015	Senior Honors Research Project II (CSE-496), Computer Science Department, Stony Brook University
Fall 2015	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University
Fall 2015	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Fall 2015	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Fall 2015	Graduate Teaching Practicum (CSE-698), Computer Science Department, Stony Brook University

Fall 2015	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Summer 2015	M.S. Internship in Research (CSE-596), Computer Science Department, Stony Brook University
Summer 2015	Internship in Research (CSE-696), Computer Science Department, Stony Brook University
Spring 2015	Topics in Computer Science: Storage Systems (CSE-593), Computer Science Department, Stony Brook University
Spring 2015	Senior Honors Research Project I (CSE-495), Computer Science Department, Stony Brook University
Spring 2015	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University
Spring 2015	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Spring 2015	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Spring 2015	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University
Spring 2015	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Fall 2014	Advanced Systems Programming in Unix/C (CSE-376), Computer Science Department, Stony Brook University
Fall 2014	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University
Fall 2014	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Fall 2014	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Fall 2014	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University
Fall 2014	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Summer 2014	M.S. Internship in Research (CSE-596), Computer Science Department, Stony Brook University
Summer 2014	Internship in Research (CSE-696), Computer Science Department, Stony Brook University
Spring 2014	Graduate Level *Operating Systems* (CSE-506), Computer Science Department, Stony Brook University

Spring 2014	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University
Spring 2014	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Spring 2014	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Spring 2014	Practicum in Teaching (CSE-698), Computer Science Department, Stony Brook University
Fall 2013	Advanced Systems Programming in Unix/C (CSE-376), Computer Science Department, Stony Brook University
Fall 2013	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University
Fall 2013	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Fall 2013	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Fall 2013	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Summer 2013	M.S. Internship in Research (CSE-596), Computer Science Department, Stony Brook University
Summer 2013	Internship in Research (CSE-696), Computer Science Department, Stony Brook University
Summer 2013	FT Summer Research (CSE-800), Computer Science Department, Stony Brook University
Spring 2013	Graduate Level *Operating Systems* (CSE-506), Computer Science Department, Stony Brook University
Spring 2013	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University
Spring 2013	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Spring 2013	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Spring 2013	Topics in Computer Science: Storage Systems (CSE-595), Computer Science Department, Stony Brook University
Spring 2013	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Fall 2012	*Operating Systems* (CSE-306), Computer Science Department, Stony Brook University
Fall 2012	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University

Fall 2012	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Fall 2012	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Fall 2012	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Summer 2012	M.S. Internship in Research (CSE-596), Computer Science Department, Stony Brook University
Summer 2012	Internship in Research (CSE-696), Computer Science Department, Stony Brook University
Summer 2012	FT Summer Research (CSE-800), Computer Science Department, Stony Brook University
Spring 2012	Graduate Level *Operating Systems* (CSE-506), Computer Science Department, Stony Brook University
Spring 2012	Special Project in Computer Science (CSE-522), Computer Science Department, Stony Brook University
Spring 2012	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University
Spring 2012	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Spring 2012	Topics in Computer Science: Storage Systems (CSE-595), Computer Science Department, Stony Brook University.
Spring 2012	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University
Spring 2012	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Fall 2011	Advanced Systems Programming in Unix/C (CSE-376), Computer Science Department, Stony Brook University
Fall 2011	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Fall 2011	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Fall 2011	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University
Fall 2011	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Summer 2011	Information Systems Internship (ISE-488), Computer Science Department, Stony Brook University

Summer 2011	M.S. Internship in Research (CSE-596), Computer Science Department, Stony Brook University
Summer 2011	Internship in Research (CSE-696), Computer Science Department, Stony Brook University
Spring 2011	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University
Spring 2011	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Spring 2011	*M.S. Thesis Research* (CSE-599), Computer Science Department, Stony Brook University
Spring 2011	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Fall 2010	Graduate Level *Operating Systems* (CSE-506), Computer Science Department, Stony Brook University
Fall 2010	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Fall 2010	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Fall 2010	Practicum in Teaching (CSE-698), Computer Science Department, Stony Brook University
Fall 2010	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Summer 2010	M.S. Internship in Research (CSE-596), Computer Science Department, Stony Brook University
Summer 2010	Internship in Research (CSE-696), Computer Science Department, Stony Brook University
Summer 2010	FT Summer Research (CSE-800), Computer Science Department, Stony Brook University
Spring 2010	Systems Administration (CSE-311), Computer Science Department, Stony Brook University
Spring 2010	Systems Administration (ISE-311), Computer Science Department, Stony Brook University
Spring 2010	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University
Spring 2010	*M.S. Thesis Research* (CSE-599), Computer Science Department, Stony Brook University
Spring 2010	Practicum in Teaching (CSE-698), Computer Science Department, Stony Brook University

Spring 2010	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Fall 2009	Graduate Level *Operating Systems* (CSE-506), Computer Science Department, Stony Brook University
Fall 2009	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Fall 2009	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Fall 2009	*M.S. Thesis Research* (CSE-599), Computer Science Department, Stony Brook University
Fall 2009	Practicum in Teaching (CSE-698), Computer Science Department, Stony Brook University
Fall 2009	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Spring 2009	Advanced Systems Programming in Unix/C (CSE-376), Computer Science Department, Stony Brook University
Spring 2009	Undergraduate Teaching Practicum (CSE-475), Computer Science Department, Stony Brook University
Spring 2009	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University
Spring 2009	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Spring 2009	Proficiency Requirement in Computer Science (CSE-587), Computer Science Department, Stony Brook University
Spring 2009	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Spring 2009	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University
Spring 2009	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Fall 2008	Graduate Level *Operating Systems* (CSE-506), Computer Science Department, Stony Brook University
Fall 2008	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Fall 2008	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Fall 2008	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University

Fall 2008	Practicum in Teaching (CSE-698), Computer Science Department, Stony Brook University
Fall 2008	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Summer 2008	FT Summer Research (CSE-800), Computer Science Department, Stony Brook University
Spring 2008	Advanced Systems Programming in Unix/C (CSE-376), Computer Science Department, Stony Brook University
Spring 2008	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University
Spring 2008	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Spring 2008	Proficiency Requirement in Computer Science (CSE-587), Computer Science Department, Stony Brook University
Spring 2008	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Spring 2008	Practicum in Teaching (CSE-698), Computer Science Department, Stony Brook University
Spring 2008	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Fall 2007	Graduate Level *Operating Systems* (CSE-506), Computer Science Department, Stony Brook University
Fall 2007	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University
Fall 2007	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Fall 2007	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Fall 2007	Practicum in Teaching (CSE-698), Computer Science Department, Stony Brook University
Fall 2007	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Summer 2007	FT Summer Research (CSE-800), Computer Science Department, Stony Brook University
Spring 2007	Advanced Systems Programming in Unix/C (CSE-376), Computer Science Department, Stony Brook University
Spring 2007	Undergraduate Teaching Practicum (CSE-475), Computer Science Department, Stony Brook University

Spring 2007	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University
Spring 2007	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Spring 2007	Topics in Computer Science: Secure Storage (CSE-590), Computer Science Department, Stony Brook University. Co-taught with Prof. Radu Sion.
Spring 2007	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Spring 2007	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University
Spring 2007	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Fall 2006	Graduate Level *Operating Systems* (CSE-506), Computer Science Department, Stony Brook University
Fall 2006	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University
Fall 2006	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Fall 2006	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Fall 2006	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University
Fall 2006	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Summer 2007	FT Summer Research (CSE-800), Computer Science Department, Stony Brook University
Spring 2006	Advanced Systems Programming in Unix/C (CSE-376), Computer Science Department, Stony Brook University
Spring 2006	Topics in Computer Science: File Systems (CSE-594), Computer Science Department, Stony Brook University
Spring 2006	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University
Spring 2006	Proficiency Requirement in Computer Science (CSE-587), Computer Science Department, Stony Brook University
Spring 2006	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University
Spring 2006	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University

Fall 2005	Graduate Level *Operating Systems* (CSE-506), Computer Science Department, Stony Brook University
Fall 2005	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Fall 2005	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Fall 2005	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University
Fall 2005	Practicum in Teaching (CSE-698), Computer Science Department, Stony Brook University
Fall 2005	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Spring 2005	Research in Computer Science (CSE-487), Computer Science Department, Stony Brook University
Summer 2005	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Summer 2005	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Summer 2005	FT Summer Research (CSE-800), Computer Science Department, Stony Brook University
Spring 2005	Advanced Systems Programming in Unix/C (CSE-376), Computer Science Department, Stony Brook University
Spring 2005	Advanced Operating Systems (CSE-624), Computer Science Department, Stony Brook University
Spring 2005	Research in Computer Science (CSE-487), Computer Science Department, Stony Brook University
Spring 2005	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University
Spring 2005	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Spring 2005	Proficiency Requirement in Computer Science (CSE-587), Computer Science Department, Stony Brook University
Spring 2005	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Spring 2005	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Fall 2004	Graduate Level *Operating Systems* (CSE-506), Computer Science Department, Stony Brook University

Fall 2004	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Fall 2004	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University
Fall 2004	Practicum in Teaching (CSE-698), Computer Science Department, Stony Brook University
Fall 2004	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Spring 2004	Graduate Level *Operating Systems* (CSE-506), Computer Science Department, Stony Brook University
Spring 2004	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Spring 2004	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Spring 2004	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University
Spring 2004	Practicum in Teaching (CSE-698), Computer Science Department, Stony Brook University
Spring 2004	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Fall 2003	Advanced Systems Programming in Unix/C (CSE-376), Computer Science Department, Stony Brook University
Fall 2003	Introduction to Software Engineering and Project Plan (CSE-523), Computer Science Department, Stony Brook University
Fall 2003	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Fall 2003	Proficiency Requirement in Computer Science (CSE-587), Computer Science Department, Stony Brook University
Fall 2003	Independent Study in Computer Science (CSE-593), Computer Science Department, Stony Brook University
Fall 2003	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University
Summer 2003	Research in Computer Science (CSE-487), Computer Science Department, Stony Brook University
Summer 2003	Intro to Software Engineering (CSE-523), Computer Science Department, Stony Brook University
Spring 2003	Graduate Level *Operating Systems* (CSE-506), Computer Science Department, Stony Brook University

Spring 2003	Senior Honors Research Project II (CSE-496), Computer Science Department, Stony Brook University
Spring 2003	Intro to Software Engineering (CSE-523), Computer Science Department, Stony Brook University
Spring 2003	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Spring 2003	Independent Study in Computer Science (CSE-587), Computer Science Department, Stony Brook University
Spring 2003	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University
Spring 2003	Graduate Teaching Practicum (CSE-698), Computer Science Department, Stony Brook University
Fall 2002	Advanced Systems Programming in Unix/C (CSE-376), new course developed, Computer Science Department, Stony Brook University
Fall 2002	Senior Honors Research Project I (CSE-495), Computer Science Department, Stony Brook University
Fall 2002	Intro to Software Engineering (CSE-523), Computer Science Department, Stony Brook University
Fall 2002	Lab in Computer Science II (CSE-524), Computer Science Department, Stony Brook University
Fall 2002	Independent Study in Computer Science (CSE-587), Computer Science Department, Stony Brook University
Fall 2002	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University
Summer 2002	Intro to Software Engineering (CSE-523), Computer Science Department, Stony Brook University
Spring 2002	Graduate Level *Operating Systems* (CSE-506), Computer Science Department, Stony Brook University
Spring 2002	Undergraduate Teaching Practicum (CSE-475), Computer Science Department, Stony Brook University
Spring 2002	Research in Computer Science (CSE-487), Computer Science Department, Stony Brook University
Spring 2002	Intro to Software Engineering (CSE-523), Computer Science Department, Stony Brook University
Spring 2002	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University
Spring 2002	Internship in Research (CSE-698), Computer Science Department, Stony Brook University

Spring 2002	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Fall 2001	*Special Topics in Software Engineering: Systems Programming* (CSE/ISE-391), Computer Science Department, Stony Brook University
Fall 2001	M.S. Thesis Research (CSE-599), Computer Science Department, Stony Brook University
Fall 2001	Ph.D. Dissertation Research (CSE-699), Computer Science Department, Stony Brook University
Spring 2001	Graduate Level *Operating Systems* (CSE-506), Computer Science Department, Stony Brook University
Spring 2001	*Advanced File Systems*, seminar, Computer Science Department, Stony Brook University
November 1999	Tutorial, *Using Amd and the Automounter Utilities*, USENIX LISA 13, Seattle, Washington
October 1999	*NFS: Protocols, Programming and Implementation* (guest lecture in Internet Systems Programming), Computer Science Department, Columbia University
1997–1999	*Operating System Vulnerabilities* (guest lecture in Network Security), Computer Science Department, Columbia University
Fall 1995	*Data Structures in C*, Computer Science Department, Columbia University
1986	*Programming in BASIC*, A.A. Computers, Holon, Israel
1986	*Programming in Logo*, A.A. Computers, Holon, Israel

Teaching Assistant

Spring 1995	*Computer Security*, Computer Science Department, Columbia University
Fall 1994	*Operating Systems*, Computer Science Department, Columbia University
Spring 1992	*Object-Oriented Programming and C++*, Computer Science Department, Columbia University
Spring 1989	*Data Structures*, Computer Science Department, Columbia University
Fall 1989	*Software Design Lab,* Computer Science Department, Columbia University
Fall 1988	*Introduction to Programming in Pascal*, Computer Science Department, Columbia University

GRADUATE STUDENTS (PH.D.)

Ph.D. Students Graduated

May 2014–Jan 2019	Zhen Cao Dissertation title: A Practical Auto-Tuning Framework for Storage Systems Current Position: Google, Inc. (Mountain View, CA)
May 2012–Apr 2017	Ming Chen. Dissertation title: Kurma: Efficient and Secure Multi-Cloud Storage Gateways for Network-Attached Storage Current Position: Google, Inc. (New York, NY)

Sep 2009–May 2014	Zhichao Li. Dissertation title: GreenDM: A Versatile Tiering Hybrid Drive for the Trade-Off Evaluation of Performance, Energy, and Endurance Current Position: VMware, Inc. (Palo Alto, CA)
Jan 2008–Nov 2013	Vasily Tarasov (IBM Ph.D. Fellow). Dissertation title: Multi-dimensional Workload Analysis and Synthesis for Modern Storage Systems Current Position: IBM Research–Almaden, Inc. (San Jose, CA)
Jan 2006–Dec 2013	Justin Seyster (domestic) Dissertation title: Runtime Verification of Kernel-Level Concurrency Using Compiler-Based Instrumentation Current Position: Apple Computer, Inc. (Cupertino, CA)
Jan 2008–Feb 2012	Richard Spillane (domestic). Dissertation title: Efficient, Scalable, and Versatile Application and System Transaction Management for Direct Storage Layers. Current Position: Apple Computer, Inc. (Cupertino, CA)
Sep 2007–May 2011	Vinay Pai (domestic). Dissertation title: Incentive Mechanisms for Peer-to-Peer Streaming. Current Position: MuCash (New York, NY)
Sep 2003–Aug 2009	Sean Callanan (domestic). Dissertation title: Flexible Debugging with Controllable Overhead. Current Position: Apple Computer, Inc. (Cupertino, CA)
Sep 2003–Aug 2008	Avishay Traeger (domestic) (IBM Ph.D. Fellow). Dissertation title: Analyzing Root Causes of Latency Distributions. Current Position: Research Staff Member, IBM Haifa Research Labs (Tel-Aviv, Israel)
Sep 2003–Feb 2008	Gopalan Sivathanu. Dissertation title: End-to-End Abstractions for Application-Aware Storage. Current position: Google, Inc. (Mountain View, CA).
May 2003–Dec 2006	Nikolai Joukov. Dissertation title: Versatile, Portable, and Efficient File System Profiling. Current position: Research Staff Member, IBM T. J. Watson Research Center (Hawthorne, NY).
May 2003–May 2006	Charles P. Wright (domestic). Dissertation title: Extending ACID Semantics to the File System via `ptrace`. Current position: Research Staff Member, IBM T. J. Watson Research Center (Hawthorne, NY).

Current Ph.D. Students

Jan 2020–present	Yifei Liu
May 2019–present	Meyer “Alex” Merenstein
May 2018–present	Tyler Estro
Sep 2017–present	Ibrahim “Umit” Akgun

Past Ph.D. Students Advised

May 2019–Dec 2021	Wei Su
Nov 2014–Dec 2017	Sun (Jason) Zhen (visiting scholar from China)
Jun 2014–Dec 2016	Sonam Mandal
May 2004–Nov 2004	Anand Kashyap (He was Samir Das student. I advised him on one project and paper that came out of a class project.)
Sep 2001–May 2002	Li Aili (Student came to program interested in systems work, but became interested in graphics and switched areas altogether. Currently advised by Klaus Mueller.)
May 2001–May 2002	Joseph Spadavecchia (domestic) (I advised and funded student for one year. Student dropped out of program to go work and study in Scotland, for personal reasons. We co-authored one paper together.)
May 2002–Aug 2002	Zhenghong "Sam" Yang (Student already dropped from PhD to MS before I funded him under SPIR. Student graduated and now works for IBM.)
Sep 2001–May 2002	Hui Zhang (Student came to program interested in systems work, but became interested in hardware systems. Currently advised by Yuanyuan Yang.)

GRADUATE STUDENTS (M.S.)

Current M.S. Students

May 2022–present	Pradeep Nalluri
Jun 2021–present	Archit Saxena

Past M.S. Students

Jun 2021–May 2022	Manish Adkar
Jun 2021–May 2022	Nasratullah Sultany
Jun 2021–Jun 2022	Pei Liu
Jun 2021–Dec 2021	Rohan Chhabra
Jun 2021–May 2022	Rishabh Srivastava
Jun 2021–May 2022	Shushanth Madhubalan
Jun 2021–May 2022	Tejeshwar Gurram
Jan 2021–Dec 2021	Aakarsh Duvvuru
Sep 2020–Dec 2021	Aadil Shaikh
Jun 2020–May 2021	Dheeraj Ramchandani
Jan 2020–Dec 2020	Abhiraj Smit
Jan 2020–May 2021	Gomathi Ganesan
Jan 2020–Dec 2020	Vigneshwaran Mylsamy
Jan 2020–Dec 2020	Yatna Verma

Aug 2019–May 2020	Aneesh Joshi
Jun 2019–Dec 2019	Sagar Jeevan
Jan 2019–Dec 2019	Akshay Aurora
Jan 2019–Dec 2019	Jatin Sood
Jan 2019–Dec 2019	Krapi Ravindra Shah
Jan 2019–Dec 2019	Mehul Jain
Jan 2019–Dec 2019	Pragesh Jagnani
Jan 2019–Dec 2019	Shobhit Khandelwal
Sep 2018–Dec 2019	Aayush Sureka
Sep 2018–May 2019	Dhivahar Perumal
Sep 2018–Dec 2019	Ritika Nevatia
May 2018–Dec 2019	Nilesh Somani
Jan 2018–Dec 2019	Prafful Agarwal
Jan 2018–Dec 2018	Ankit Aggarwal
Jan 2018–Dec 2018	Dhanashri Patil
Jan 2018–Dec 2018	Manu Mathew
Jan 2018–Dec 2018	Prateek Roy
Jan 2018–Dec 2018	Rahul Rane
Jan 2018–Dec 2018	Rohit Kumar
Jan 2018–Dec 2018	Saish Sali
Jan 2018–Dec 2018	Siddesh Shinde
Jan 2018–Dec 2018	Vineeth Ramesh
Aug 2017–Dec 2018	Noopur Anil Maheshwari
Sep 2017–Apr 2018	Tyler Estro
Jun 2017–May 2018	Mayur Jadhav
May 2017–May 2018	Maryia Maskaliova
Jan 2017–Dec 2017	Darshan Godhia
Jan 2017–Dec 2017	Farhaan Jalia
Jan 2017–Dec 2017	Geetika Babu Bangera
Jan 2017–Dec 2017	Kunal Shah
Jan 2017–Dec 2017	Nehil Shah
Jan 2017–Dec 2017	Nidhi Panpalia

Jan 2017–Dec 2017	Rushabh Shah
Jan 2017–Dec 2017	Sagar Shah
Jan 2017–Dec 2017	Swaminathan Sivaraman
Jan 2017–Dec 2017	Tushar Jain
Sep 2016–Dec 2017	Vinothkumar Raja
Aug 2016–Dec 2017	Arun Ramachandran
Aug 2016–Dec 2017	Mukul Sharma
Aug 2016–Dec 2017	Sachin Tiwari
Aug 2016–Dec 2017	Shivanshu Goswami
Aug 2016–Dec 2016	Vishal Sahu
Apr 2016–May 2017	Leixiang Wu
Jan 2016–Dec 2016	Praveen Kumar Morampudi
Jan 2016–Dec 2016	Vithiya Muthukumar
Jan 2016–Dec 2016	Harshkumar Patel
Jan 2016–Dec 2016	Deepika Peringanji
Jan 2016–Dec 2016	Venkatakrishnan Rajagopalan
Jan 2016–Dec 2016	Vishnu Vardhan Rajula
Jan 2016–Dec 2016	Hari Prasath Raman
Sep 2015–Dec 2016	Shubhi Rani
Jan 2016–Dec 2016	Jasmit Saluja
Sep 2015–Dec 2016	Ashok Sankar Harihara Subramony
Aug 2015–Dec 2016	Bharath Kumar Reddy Vangoor
Sep 2014–May 2016	Dongju Ok
Mar 2015–May 2016	Garima Gehlot
Sep 2015–Dec 2015	Vivek Tiwari
May 2014–Dec 2015	Kumar Sourav
Sep 2014–Dec 2015	Bharat Singh
Sep 2014–Dec 2015	Varun Shastry
May 2014–Dec 2015	Abhishek Gupta
Sep 2014–Dec 2015	Arvind Chaudhary
Sep 2014–Dec 2015	Aashray Arora
Sep 2013–May 2015	Arun Olappamanna Vasudevan

Jan 2014–May 2015	Akhilesh Chaganti
May 2014–May 2015	Udit Kaushik Chitalia
Apr 2014–May 2015	Kelong Wang
Sep 2013–Dec 2014	Soujanya Shankaranarayana
May 2014–Dec 2014	Li Mengyang
Sep 2013–May 2014	Amanpreet Mukker
Jun 2013–May 2014	Sonam Mandal
Sep 2012–Dec 2013	Deepak Jain
Sep 2012–Dec 2013	Sagar Trehan
Sep 2012–Dec 2013	Madhu Srikar Palmur
Sep 2012–May 2013	Benixon Arul Dhas
May 2012–Jun 2013	Karthikeyani Palanisami
Jan 2012–Dec 2012	Binesh Andrews
Jan 2012–Dec 2012	Atul Karmarkar
Sep 2011–Dec 2012	Mandar S. Joshi
Sep 2011–Dec 2012	Ravikant R. Malpani
Sep 2011–May 2012	Rajesh Aavuty
Jan 2011–May 2012	Amar Mudrankit
Jan 2011–Dec 2011	Siddhi Tadpatrikar
Sep 2010–Dec 2011	Samriti Katoch
Sep 2010–Dec 2011	Koundinya Santhosh Kumar
Aug 2010–May 2012	Pradeep Shetty
Jan 2011–May 2011	Koundinya Muppalla
Jan 2010–Dec 2010	Prabakar Radhakrishnan
Jan 2010–Dec 2010	Gyumin Sim
Sep 2009–Dec 2010	Shrikar Archak
Sep 2009–Dec 2010	Saumitra Bhanage
Sep 2009–Dec 2010	Sagar Dixit
Jan 2009–May 2010	Priya Sehgal
Jan 2009–Dec 2009	Karthikeyan Srinaivasan
Sep 2008–May 2010	Abhinav Duggal
Sep 2008–Dec 2009	Sujay Godbole

Jan 2008–Dec 2008	Yamini Allu
Jan 2008–Dec 2008	Manjunath Chinni
Jan 2008–Dec 2008	Sachin Gaikwad
Sep 2007–Jun 2009	Daniel J. Dean (domestic)
Sep 2007–Dec 2008	Himanshu Kanda
Sep 2007–May 2009	Rachita Kothiyal
Sep 2007–Dec 2008	Chaitanya Yalamanchili
Jan 2007–Dec 2007	Kimberly Albrecht (domestic, SFS scholar)
Jan 2007–Dec 2007	Ramya Edara
Jan 2007–Dec 2007	Ivan Deras Tabora
Jan 2007–Dec 2007	Kumar Thangavelu
Sep 2006–Dec 2007	Gopala Suryanarayana
Sep 2006–Dec 2007	Kiron Vijayasankar
Jun 2006–Dec 2007	Richard Spillane (domestic)
Jan 2006–August 2007	Swaminathan Sundararaman
Sep 2006–June 2007	Yiannis Pericleous
Sep 2005–May 2007	David Quigley (domestic, SFS scholar)
Jan 2006–March 2007	Harry Papaxenopoulos (domestic, SFS scholar)
Jan 2006–Dec 2006	Chaitanya Patti
Jan 2005–Dec 2005	Dan Ottavio (domestic)
Jan 2005–Dec 2005	Arun Kumar
Sep 2004–Dec 2005	Naveen Gupta
Sep 2004–Dec 2005	Rakesh Narayan Iyer
Sep 2004–Dec 2005	Sunil Satnur
May 2004–Nov 2004	Swapnil V. Patil
Jan 2004–Dec 2004	Mohammad Nayyer Zubair
Sep 2003–Aug 2005	Abhishek Rai
Sep 2003–Dec 2004	Aditya Kashyap
Sep 2003–Dec 2004	Devaki Pandurang Kulkarni
May 2003–Aug 2002	Akshat Khandelwal
May 2003–Aug 2002	Deepak Rao
May 2003–Aug 2004	Akshat Aranya

May 2003–Dec 2003	Alexander Butler (domestic)
May 2003–Dec 2003	Jay Pradip Dave
May 2003–Dec 2003	Mohan-Krishna Channa-Reddy
May 2003–Dec 2003	Salil Gokhale
May 2003–Jan 2004	Abhijith Das
May 2003–Dec 2004	Miretskiy “Eugene” Yevgeniy (domestic)
Jan 2003–Dec 2003	Harikesavan Pathangi Krishnan
Jan 2003–Dec 2003	Puja Gupta
May 2002–Dec 2002	Delia Paval
May 2002–Dec 2002	Nitin Khosla
May 2002–Dec 2002	Rongqing “Frank” Tu
May 2002–May 2003	Jeffrey R. Osborn (domestic)
May 2002–May 2003	Michael Martino (domestic)
May 2002–May 2003	Zhenghong “Sam” Yang
Sep 2002–Dec 2002	Nishant Nagalia
Sep 2002–Dec 2002	Zhou Zhang
Sep 2002–Dec 2003	Swaroop Karunakara
Sep 2002–May 2003	Ana Centeno
Sep 2002–May 2003	Andrew Himmer (domestic)
Sep 2002–May 2003	Sheshadri Sreenath
Mar 2002–Apr 2003	Manish Prasad
Jan 2002–May 2004	Kiran-Kumar Muniswamy-Reddy
Sep 2001–May 2003	Amit Purohit

UNDERGRADUATE STUDENTS

Jun 2021–present	Michael Arkhangelskiy
Jun 2021–present	Andrew Burford
Jun 2021–present	Michael McNeill

Past Undergraduate Students

Jun 2021–Jun 2022	Pei Liu (domestic, BS program, CSE program)
Jun 2020–Dec 2020	Haolin Yu (BS program, CSE program)
Jun 2020–May 2021	Haewon Julie Lee (BS program, CSE program)
Jun 2020–May 2021	Yang Yang (domestic, BS program, CSE program)

Aug 2019–May 2020	Yinuo Zhang (BS program, CSE program)
Jul 2019–Dec 2019	Abraham Spitalny, (domestic, BS program, CSE program)
May 2018–Dec 2018	Amrith Arunachalam (domestic, BS program, CSE program)
May 2017–May 2018	Kevin Sun (domestic, BS program, CSE program)
Feb 2015–Mar 2016	Leixiang Wu (domestic, BS program, CSE program)
July 2012–Dec 2012	Edgardo Linero, (domestic, BS program, CSE program)
May 2011–May 2012	Ryan Matthew Walsh, (domestic, BS program, ISE program)
May 2010–Aug 2011	Christopher Jutting, (domestic, BS program, ISE program)
Jan 2009–July 2009	Ryan Leif Walsh, (domestic, BS/MS program, Honors program)
Jan 2004–2009	Adam David Alan Martin (domestic)
Sep 2005–May 2008	Josef "Jeff" Sipek (domestic)
Jun 2006–August 2007	Daniel J. Dean (domestic)
Jan 2005–May 2006	Richard Spillane (domestic, BS/MS program, Honors program, Stony Brook URECA summer internship)
Jan 2005–Jul 2005	Tim Wong (domestic)
Jan 2005–Aug 2005	David Quigley (domestic)
May 2003–Dec 2003	Mohammad Nayyer Zubair (Honors program, Stony Brook URECA summer internship)
Jan 2002–May 2002	Evan Chan (domestic)
Jan 2002–May 2002	Reynold Mercado (domestic)
Jan 2002–May 2002	Jeffrey Osborn (domestic)
Jan 2002–May 2002	Julio Salazar (domestic)
Jan 2002–May 2002	Ariye Shater (domestic)
Jan 2002–May 2003	Charles P. Wright (Honors program advisor, domestic)
Jan 2002–May 2002	Yuri Yanpolski (domestic)

HIGH-SCHOOL STUDENTS

Past High-School Students

Sep 2015–Aug 2017	Henry Nelson (domestic), "NFSv4 Compounds (vNFS)" project
June 2005–August 2005	Jordan Hoch (domestic), Simons summer internship, project title "Visual Debugging of Operating Systems State.")

Stony Brook, New York, June 3, 2025