2018

Rising Tide_Promotion & Tenure_Tenure

University of Maine Rising Tide Center

UNIVERSITY OF MAINE SYSTEM

Tenure and Promotion Application Form

I. FACE DATA

A. **NAME:** Vincent Weaver

B. **PRESENT RANK:** Assistant Professor

C. **COLLEGE / DEPARTMENT:** Electrical and Computer Engineering

D. **PROFESSIONAL EXPERIENCE:**

YEARS	EMPLOYER	POSITION
2012-	University of Maine	Assistant Professor
2010-2012	University of Tennessee	Postdoctoral Research Associate
2008-2009	Center for Applied Math, Cornell	TA Sysadmin
2006-2007	ECE, Cornell	Teaching Assistant
2005	Lawrence Livermore National Lab	Summer Intern
2003-2004	ECE Techshop, Cornell	TA Sysadmin
2002-2003	US Army SBCCOM / ORISE	Research Programmer
2000-2001	Frontpath, Inc.	Jr. Software Engineer
1995-2000	US Army SBCCOM / ORISE / SEAP	Summer/Winter Intern

E. **EDUCATIONAL BACKGROUND:**

YEARS	INSTITUTION	FIELD	DEGREE
2003-2010	Cornell University	Electrical & Computer Engineering	Ph.D.
2009	Cornell University	Electrical & Computer Engineering	M.S.
1996-2000	University of Maryland, College Park	Electrical Engineering	B.S.

II. RECORD OF ACTIONS

A. INITIAL PROBATIONARY APPOINTMENT

1. Date: 1 September 2012
2. Length of Initial Appointment: 1 year
3. Prior Experience Credited toward Tenure: N/A
4. Rank: Assistant Professor

B. REAPPOINTMENTS

DATE	LENGTH
1 September 2013	1 year
1 September 2014	1 year
1 September 2015	1 year
1 September 2016	1 year
1 September 2017	1 year

C. PROMOTIONS

EFFECTIVE DATE	TO RANK

D. RECOMMENDATIONS FOR:

RECOMMENDING BODY	RECOMMENDATION (YES/NO/ NO ACTION)	SIGNATURE	DATE
Peer Committee			
Department/Division Chair			
Dean			
Provost/VPAA			
President			

E. EXCEPTIONS TO BOARD OF TRUSTEES POLICY

None

F. TRANSMITTAL LETTERS

1. President
2. Provost/VPAA
3. Dean

III. CANDIDATE'S PROFILE

A. DOCUMENTATION OF TEACHING (including advising)

I devote 50% of my time to teaching.
Typical teaching load is two courses per semester.
Due to current university TA allocation policies my courses are often taught without TA support.

Teaching Philosophy

My teaching primarily involves courses in computer engineering. I usually teach 400-level computer engineering electives and graduate courses, but on two occasions I have taught sophomore-level classes (one of which was an electrical engineering circuits lab).

Computer Engineering is an area where hands-on learning is necessary for complete mastery of the material. Lecturing is not enough; the students need to be programming, building circuits, constructing networks, measuring performance, and undertaking other activities that reinforce what they have learned in class. My courses all incorporate regular homeworks that apply what is learned in lecture to actual real-world applications. These homeworks act as self-paced labs, allowing hands-on experimentation even when the courses do not include separate lab sections. In the graduate courses the students are given accounts on high-end Linux servers where they conduct experiments with actual modern hardware, including multi-core graphics processing units (GPUs) and high-speed networks.

Computer engineering is a rapidly changing field which requires constant attention to current events to stay on top of recent research developments. In addition to introducing students at all levels to contemporary topics, I also include my current research when appropriate. My research into performance analysis, Linux operating system programming, computer security, embedded systems, and cluster construction are all directly relevant to the courses I teach.

Student feedback is important, and I closely monitor the end-of-semester course reviews (a summary of which is shown in Section VIII). For student engagement it is important that an instructor be enthusiastic about the topic being taught. An instructor should also be well prepared (Q1, Q6) and should make sure to sufficiently answer student questions in a complete and respectful manner (Q9). The students should also feel that the instructor has an interest in the progress they are making in the class (Q12).

In addition to helping students in-class, it is important to be available to students while they are working on homeworks. I keep a standard set of office hours, but in addition I have an open-door policy where students can drop by with questions any time my office door is open. I also encourage the students to send e-mails with questions, especially when they are having programming issues. A quick piece of advice can save hours of frustration when struggling with a complicated problem.

Since starting at the University of Maine I have constructed completely new classes, such as ECE598 Advanced Operating Systems. I have also overhauled the existing ECE471 Embedded Systems and ECE574 Cluster Computing courses, refreshing the material as well as updating the computing hardware used. These classes now use the popular low-cost Raspberry Pi development board, which allows students to have individual hands-on experience with an actual embedded computing device. I also brought back ECE435 Network Engineering after a decade of not being taught, updating the course material for modern times.

Courses taught:

ECE214: Electrical Circuits Lab (2 credits)

Spring 2015 41 students

This course is the introductory analog circuits lab. The students learn to use test equipment and how to maintain lab notebooks. The labs start out with RC circuits, advance to op-amps, and culminate with the design and analysis of a DC–DC converter. I taught this to help cover for another professor who was on sabbatical.

ECE271: Microcomputer Architecture and Applications (3 credits)

Spring 2014 49 students (co-taught with Yifeng Zhu)

Introduction to low level computer hardware (timers, interrupts, GPIOs, etc), using STM32L discovery board and ARM assembly language.

ECE435: Network Engineering (3 credits)

Fall 2016 13 students
Fall 2017 17 students

This course covers computer networks from the physical layer up through the application layer, including socket programming and security issues. There are hands-on, lab-like homeworks throughout the semester, with a network-related final group project. I brought back this course after ten years of not being taught and developed all new course materials.

ECE471: Embedded Systems (3 credits)

Fall 2013	9 students	
Fall 2014	20 students	
Fall 2015	22 students	
Fall 2016	28 students	
Fall 2017	16 students	

This course covers embedded systems: ARM processors and assembly language, system busses (i2c, SPI, 1-wire), real-time concerns, code quality, operating systems (with a focus on embedded Linux), security, programming ethics, and low-power design. There are hands-on, lab-like homeworks throughout the semester with an open-ended final group project. I have completely restructured this class since I have started teaching it: this includes all new course material as well as a move to using Raspberry Pi devices for the homeworks and projects.

ECE498: Linux Assembly Language Programming (1 credit)

Fall 2012	6 students

This course covers ARM, x86, and 6502 assembly language. The final assignment involves writing a short game for the SNES gaming system in assembly language.
I developed all-new content for this course.

ECE571: Advanced Microprocessor-Based Design (3 credits)

Spring 2013	6 students	
Fall 2014	9 students	
Spring 2016	16 students	(12 local, 4 online)
Spring 2017	8 students	

This course (one of the required courses for a masters degree) covers advanced microprocessor topics: modern computer architecture, x86 and ARM processors, performance analysis, and power/energy design tradeoffs. Computer architecture is reviewed and then built upon with current issues in CPUs, GPUs, DRAM, and non-volatile memory. Homeworks involve performance and power measurements on actual high-end x86 and ARM hardware. There is a final group project. I have completely restructured and developed new content for this course.

ECE574: Cluster Computing (3 credits)

Fall 2015 10 students
Spring 2017 21 students

This course covers all aspects of high performance computing: cluster design, parallel programming (pthreads, OpenMP, MPI, GPGPU), and performance analysis. The students run code on high-end 16-core x86 servers and a 24-node Raspberry-Pi cluster. Homeworks involve writing parallel code; there is a final group project. I developed all new content for this course.

ECE598: Advanced Operating Systems (3 credits)

Spring 2015 12 students
Spring 2016 8 students

This course covers advanced operating system topics (device drivers, file systems, virtual memory, security) culminating with the students completing a simple UNIX-like custom operating system for a Raspberry Pi device. Homeworks involve step-by-step improving the custom operating system, and there is a final group project where the students add one advanced feature of their choice. This is a new class with all new content developed by me, including the custom operating system framework.

Advising:

I currently am the academic advisor for seven undergraduate students. This number was once closer to twenty, but has been gradually going down as my department transitions to a new cohort-based advising system. Eventually I will be assigned to an incoming class of students and co-advise them as they progress each year until graduation.

To enhance my advising skills, I attended the 2014 Academic Advising Workshop hosted by the CLAS Advising Center and Student Life.

I have advised and mentored both undergraduate and graduate students as researchers in my lab. The undergraduates are fully integrated into lab activities and they have appeared as co-authors on multiple papers. The funding for the students has been a mix of NSF grant money, internal grant money, departmental funds, and startup funds.

PhD Students (Advisor):

- Pascal Francis-Mezger, Fall 2017 –
  Automatic Validation of Performance Counters in Modern Computing Systems

Masters Students (Advisor):

- Yan Liu, Fall 2015 – December 2017 (expected)
  Optimizing PAPI for Low-overhead Counter Measurement
- Yanxiang Mao, Summer 2015 – December 2017 (expected)
  Evaluation of Embedded Boards in Low-cost Power Measurement
- Chad Paradis, Fall 2013 – August 2015
  Detailed Low-cost Energy and Power Monitoring of Computing Systems

Masters Students (on committee):

- Graduated: Tania Rahman (2015), Will Conklin (2015)
- Current: Qihao He, Xiang Guo

NSF Sensors! Research Experience for Undergraduates (REU) Students

This REU program is run by Professor Vetelino and Professor Emanetoglu in the ECE department; I contribute by advising some of the students. The students are a diverse group selectively picked from applicants nationwide. They conduct research under my guidance during the summer then write up their results in a paper which is presented at an end-of-summer workshop.

- Ben Schaff (UMaine), Summer 2016
  Sensing Power Consumption of Desktop Computer System Components
- Ian Wright (UMaine), Summer 2014
  Server DRAM Power Measurement
- Juan Mont (University of Puerto Rico Arecibo), Summer 2014
  Server Power Measurement
- Philipp-Alain Bastien (Syracuse University), Summer 2013
  Embedded Processor Power Measurement

Center for Undergraduate Research (CUGR) Student

I advised this student as he conduced research over the summer, wrote a final report, and presented it in poster form at the University of Maine Student Symposium.

- Spencer Desrochers, Summer 2015
  Computer Energy Measurement
  His work was featured in video form at the Maine Journal:
  `http://mainejournal.umaine.edu/spencer-desrochers/`

Other Undergraduate Students:

- Daniel Vaughn, Summer 2014, Summer 2015
  Embedded ROM Emulation, Parallel Raytracing
- Michael Cloutier, Summer 2014, Fall 2014
  Raspberry Pi Cluster

B. DOCUMENTATION OF SCHOLARSHIP AND PROFESSIONAL ACTIVITY

I devote 50% of my time to research.

1. Publications and Creative Works

Refereed Journal Publications

1. M.F. Cloutier, C. Paradis, V.M. Weaver. "**A Raspberry Pi Cluster Instrumented for Fine-Grained Power Measurement**", MDPI *Electronics* Journal, Special Issue on Raspberry Pi Technology, 2016, 5, 61.

Refereed Conference Publications

1. Q. He, B. Segee, V. Weaver. "**Raspberry Pi 2 B+ GPU Power, Performance, and Energy Implications**", The 2016 International Conference on Computational Science and Computational Intelligence, Las Vegas, Nevada, December 2016.
2. S. Desrochers, C. Paradis, V.M. Weaver. "**A Validation of DRAM RAPL Power Measurements**", The International Symposium on Memory Systems (MEMSYS'16), Washington, DC, October 2016.
3. I. Lopez, S. Moore, V.M. Weaver. "**A prototype sampling interface for PAPI**", Extreme Science Engineering Discovery Environment Conference (XSEDE'15), St. Louis, Missouri, July 2015.
4. V.M. Weaver. "**Self-monitoring Overhead of the Linux perf_event Performance Counter Interface**", IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS 2015), Philadelphia, Pennsylvania, March 2015, (33% acceptance rate).
5. V.M. Weaver, D. Terpstra, S. Moore. "**Non-Determinism and Overcount on Modern Hardware Performance Counter Implementations**", IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS 2013), Austin, Texas, April 2013. (26% acceptance rate).
6. J. Dongarra, H. Ltaief, P. Luszczek, V.M. Weaver. "**Energy Footprint of Advanced Dense Numerical Linear Algebra using Tile Algorithms on Multicore Architecture**", 2nd International Conference on Cloud and Green Computing (CGC 2012), Xiangtan, China, November 2012. (30% acceptance rate).

Refereed Workshop Publications

1. Y. Liu and V.M. Weaver. "**Enhancing PAPI with Low-Overhead `rdpmc` Reads**", The 6th Workshop on Extreme-Scale Programming Tools, Denver, Colorado, November 2017. (62% acceptance rate).
2. M.F. Cloutier, C. Paradis, V.M. Weaver. "**Design and Analysis of a 32-bit Embedded High-Performance Cluster Optimized for Energy and Performance**", Co-

HPC 2014: The 1st International Workshop on Hardware-Software Co-Design for High Performance Computing, New Orleans, Louisiana, November 2014.

3. V.M. Weaver. "**Linux perf_event Features and Overhead**", FastPath 2013: The 2nd International Workshop on Performance Analysis of Workload Optimized Systems, Austin, Texas, April 2013.
4. V. Weaver, M. Johnson, K. Kasichayanula, J. Ralph, P. Luszczek, D. Terpstra, S. Moore. "**Measuring Energy and Power with PAPI**", PASA'12: The 1st International Workshop on Power-Aware Systems and Architectures, Pittsburgh, Pennsylvania, September 2012.
5. M. Johnson, H. Jagode, S. Moore, P. Mucci, J. Nelson, D. Terpstra, V. Weaver, T. Mohan. "**PAPI-V: Performance Monitoring for Virtual Machines**", CloudTech-HPC 2012, Pittsburgh, Pennsylvania, September 2012.

Non-refereed Workshop Presentations

1. S. Desrochers, C. Paradis, V. Weaver. Work-in-progress presentation: "**Initial Validation of DRAM and GPU RAPL Power Measurements**", 4th Workshop on Extreme-Scale Programming Tools, Austin, TX, November 2015.

Posters

1. H. Jagode, A. YarKhan, A. Danalis, J. Dongarra, Y. Liu, V.M. Weaver. "**Recent Advances in the Performance API (PAPI)**", Supercomputing (SC'17), Denver, Colorado, November 2017. (58% acceptance rate, Nominated for Best Poster Award).
2. S. Desrochers and V. Weaver. "**Optimizing Power Usage of Modern Computing Systems**". 2016 Center For Undergraduate Research Academic Showcase, Bangor, Maine, April 2016.
3. V.M. Weaver, M. Johnson, K. Kasichayanula, J. Ralph, P. Luszczek, D. Terpstra, S. Moore, J. Nelson. "**PAPI 5.0: Measuring Power, Energy, and the Cloud**", ISPASS 2013, Austin, Texas, April 2013.

Articles (not peer-reviewed)

1. V.M. Weaver. "**Fuzzing perf_events**", Linux Weekly News, 5 August 2015, `http://lwn.net/Articles/653382/`.

2. **Scholarly and Creative Works in Progress**

My primary area of research is performance measurement of computing systems. This involves various subfields of Computer Engineering, including:

- Computer Architecture — the low-level design of computing systems,
- High Performance Computing (HPC) — more commonly known as supercomputing,
- Operating Systems — the low-level software that controls a computer, and
- Embedded Systems — the design of small computers found inside of automated devices.

I am well-known in the international community for my work with Hardware Performance Counters. Modern computers are incredibly complex and even their designers cannot predict how fast a program is going to run. Often things will go wrong: the new web-browser you downloaded runs slowly or leaks memory, a new app drains the battery in your phone too quickly, your fancy climate model is taking up too much time on the hundred-million dollar supercomputer the government just built. Hardware performance counters are the best way to find out what is going on inside the computer so you can fix, speed-up, reduce the power usage, or otherwise improve the running applications.

Hardware performance counters are found on most modern processors: usually there are from four to eight per core that can be programmed to measure interesting performance events. These events include cycle and instruction counts, as well as other important low-level architectural values such as cache misses, branch predictor misses, memory accesses, and various other more obscure features that impact performance.

Despite being critical for optimization, the counters have many issues that make them difficult to use. They are often not validated, are poorly documented, and may not work at all. Chip vendors do want accurate results, but they must balance the cost of providing accurate validated counters against the fact that the average consumer (who typically is not doing code optimization) might never use them. Counter bugs may simply be documented, leading to uneven event support across vendors and chip generations. While this complicates the use of the counters, these challenges can lead to interesting research opportunities.

Hardware counter research drives most of my grant applications and publications. I am considered to be one of the top experts worldwide, especially in the area of Linux performance counters. This has led to invited talks, program committee memberships, external collaborations, and paper reviewing opportunities.

My Hardware Performance Counter research is split among various research topics:

- **The Performance Application Programming Interface (PAPI)**

  PAPI is a well-established, widely-used library interface for accessing hardware performance counters. It is an open-source, cross-platform, portable interface allowing users on any desktop, server, or supercomputer to access the performance counters without having to know the details of the underlying system. Because of this it is widely used in high-performance computing, including many of the Top500 fastest supercomputers in the world. We support many users at national labs, and at their request many high performance computers shipped by companies such as IBM and Cray are required to have PAPI working on them as a condition for purchase. We also work with companies such as NVIDIA, AMD, Intel, Redhat, and VMware to make sure their hardware and software is fully supported.

  PAPI is primarily maintained by a group at the University of Tennessee, Knoxville. I initially worked on it there as a postdoctorate researcher and I continue to collaborate with the other PAPI developers. Currently I am a PI on a sustainable infrastructure NSF grant, in conjunction with the group at UTK, that provides funding for continued PAPI development.

  I also contribute changes to the libpfm4 project, which is the library PAPI uses for accessing the names of the various performance events. Every processor has its own, incompatible, names for the various events and they must be added manually. Many devices, including most AMD and ARM processors, are only supported due to my contributions.

  Among my recent PAPI related tasks are improving the validation tests, adding support for power measurement (RAPL), adding low-overhead read (rdpmc) support, new architecture support, and the addition of an advanced sampling interface. These changes make PAPI easier to test, make the counts more accurate, and add support for often-requested advanced measurement capabilities provided by newer processors. In addition to software development, I also help answer questions from around the world on the PAPI mailing lists. The PAPI work has led to numerous publications.

- **Linux Development**

  Linux is an open-source operating system used by millions of people around the world. It runs most supercomputers, internet servers, and every Android smartphone.

  I have made many contributions to Linux. My recent work focuses on the `perf_event` subsystem, which is the official Linux interface for interacting with hardware performance counters. A sampling of my contributions (including git commit reference) include: the Intel Xeon-Phi counter driver (e717bf4e4fe8), units fix for

the Broadwell-EP RAPL driver (33b88e708e7d), and Raspberry Pi counter support (edcb4d3c36a6).

I wrote and maintain a test suite, perf_event_test, that validates the perf_event interface. It catches regressions in new kernel releases and has led to many bug fixes. Various groups, including researchers at Intel, use this tool to watch for regressions in the perf_event code. Even though I am not an official perf_event maintainer, I have done enough work that I am usually carbon-copied on all new perf_event development patches by the Linux developers so that I can help review them.

In addition, I am the author and primary maintainer of the official perf_event_open manual page (a manual page is the primary documentation for Linux interfaces). This document is included on nearly every Linux system; you can see my work by running at the command line `man perf_event_open`.

- **System Call Fuzzing (Security)**

  Any complex computer interface needs to be well-written, otherwise unexpected inputs could cause the system to crash. Even worse, these invalid inputs might let a remote attacker take over the system.

  While working on perf_event and PAPI I found some bugs that could completely crash the entire system. These crashes were found accidentally at first, but I became interested in actively finding and fixing as many of these bugs as possible.

  Tools that generate automatically generate crashing test cases are known as "fuzzers". I initially worked on extending an existing fuzzer, Trinity, which is maintained by well-known Linux developer Dave Jones. The perf_event code I contributed to Trinity found numerous bugs, including a few major critical security vulnerabilities (CVEs).

  With my extensive knowledge of the perf_event interface I wrote my own fuzzer that used deep knowledge of the perf_event_open system-call to generate almost, but not quite, valid inputs. These were more likely to trigger bugs than the uniform random search used by Trinity. This succeeded greatly, and numerous bugs were found in Linux. Despite getting many fixes applied upstream, it is still possible to crash Linux by running this perf_fuzzer program, and the project is still ongoing.

  My fuzzer tool is widely known and used; it is particularly effective on android phones where the various vendors do a poor job of writing bug-free performance counter interfaces. The ARM Linux developers typically require a run of my tool before allowing performance unit patches into the Linux kernel.

  This work has led to various grant proposals, as well as being invited to speak at the Linux Plumber's Conference which is one of the highest regarded Linux technical conferences.

- **Power Measurement**

  Traditionally, performance (raw speed) has been the primary metric when optimizing computer code. Recently this has started to change and power usage has become just as important. This is especially true at the high end: modern supercomputers may have millions of processors; saving just one Watt of power from each quickly adds up to Megawatts of saved power and cooling costs. Power is also a concern at the low end: a common example is designing smartphones to provide maximize battery life.

  Power usage information is important, but actually gathering the measurements remains difficult. Often there is no built-in way to do this; one must custom instrument each system by hand. This often involves modifying the hardware to have a power sensor and then using an expensive external voltmeter to measure the power. While practical for small experiments, this methodology does not scale to large systems.

  My group has conducted extensive research on how to gather detailed power measurements using low-cost embedded boards. By using off-the-shelf low-cost boards, system power can be measured at a detailed, fine-grained level. This provides many opportunities for conducting power optimization in ways not previously available. Programmers can use this information to write power-aware programs that make the best use of limited battery life. Low-power programs also save electricity in desktops and servers, leading to lower power bills and cooling costs. These detailed measurements can be used to validate other power measurement methodologies, as described later.

- **Hardware Counter Validation**

  As mentioned previously, processor vendors do not fully test all possible counters. Many event counts are buggy, leading to overcounts or undercounts in the results. This can be hard to detect: some events, such as retired instruction count, have known correct values that can be easily tested and verified. Other, more complex, events that depend on the nondeterministic behavior of the underlying microarchitecture of modern out-of-order processors can vary in ways that even the chip designers cannot easily predict. This includes events involving the branch predictor, caches, and even cycle counts.

  I research the causes of overcount and nondeterminism on x86 processors, which are particularly problematic with respect to deterministic counts. I develop tools to aid in automating these tests. My publications in this area have been found valuable by other researchers, most notably the developers of the RR deterministic replay debugger from Mozilla.

- **Power Measurement Validation**

  Some newer chips from Intel have built-in power estimating circuitry that behave similarly to hardware performance counters. This is part of the Running Average Power Limit (RAPL) subsystem. The processor can provide estimated power info, but the results are not fully validated by the vendor. These values are much easier to obtain than actual power readings, which often require intrusive hardware modification, if possible at all. Users are eager to use these easy-to-obtain values when optimizing code, but the results should be validated first.

  I use the custom embedded power meters from our previously described power measurement research to gather actual real-world power readings. These are then used to validate the RAPL counters. I find that both CPU and DRAM memory results validate reasonably well to actual readings. I was the first to publish DRAM RAPL validation results; details can be found in my MEMSYS'16 conference paper.

  My research in this area has also turned up major CPU bugs. AMD processors have counters similar to the RAPL ones, and the energy measurement counters on AMD Fam16h processors are buggy. My report led to at least one official errata being added to the AMD documentation.

I also conduct research in some other areas that are not related to performance counters:

- **Code Density Exploration**

  I have a project that explores the code-density of modern processors. Different processor types have different encodings of the low level instructions they use. Some encodings are smaller than others; the ratio of the number of instructions versus the number of bytes it takes to hold them is called code-density.

  Dense code allows optimizing programs to be the smallest size possible. This can still be important in modern systems; small code that fits entirely into instruction cache will run faster. Also, there are still embedded systems that have extremely limited memory and storage sizes that benefit from small code. Finally, if code can be made smaller so that it fits in a smaller flash chip or disk, it can save money as a cheaper part can be used which is then multiplied in large product runs.

  My research involves coding the same program in over 30 different kinds of low-level assembly language for density comparison. I also find outside experts to perform critical review of the generated code.

  This is a long-term project originating over 10 years ago, but I have been continuously adding new architectures. The most recent addition is support for the new RISC-V architectures which appears to be the new standard for academic computer architecture research. I plan to seek publication of my updated recent results.

- **Raspberry Pi Cluster**

  I have constructed a Raspberry Pi cluster out of 24 embedded Raspberry Pi boards. The cluster has custom per-node power measurement circuitry as well as LED visualization displays. This project combines the research areas of embedded systems, power measurement, and high performance computing.

  The primary purpose of the cluster is to allow students to have hands-on experience working with a large many-node cluster. By building the cluster out of inexpensive boards it can have the core-count of a large server at a fraction of the cost. The students often use this as an inspiration to build their own low-cost computing clusters where they can more easily practice the incredibly difficult task of writing code for parallel systems. This research is documented in a journal article.

- **Embedded Display Projects**

  As part of my embedded system courses we discuss many kinds of embedded busses (I2C, SPI, 1-Wire, etc.). To serve as inspiration for student projects I have designed various displays and interfaces that make use of these busses. Typically these have various sensors as well as colorful LED displays. The resulting projects include various clocks, time circuit movie prop replicas, 1980s chiptune players, and wii-nunchuck powered video games. I have had this research published in various STEM hobbyist and maker publications.

- **Retro-computing**

  While researching the future of computing systems, it is important not to forget what came before. I make sure that the students in my classes obtain historical perspectives. Yesterday's high-end programming techniques work equally well on today's low-end embedded systems.

  I have various projects involving what is known as retro-computing: writing new software for obsolete architectures. My primary systems of interest are the old Apple II systems (late 1970s era) and SNES game systems (early 1990s). Much of this research is video game related as this interests the students. I have also done more practical work such as writing a webserver that runs on old Apple II hardware. This research has been published in various maker publications.

3. **Professional Presentations**

Invited Talks

1. "**All About the perf_fuzzer**", *Linux Plumbers Conference, Testing and Fuzzing track*, Santa Fe, New Mexico, 3 November 2016
2. "**The Challenge of Measuring Power on Modern Computing Systems**", *University of Maryland Alumnus Talk*, College Park, Maryland, 7 October 2016.
3. "**Hardware Performance Counter Accuracy on Modern Processors**", *Johns Hopkins Applied Physics Lab*, Columbia, Maryland, 26 June 2014
4. "**Tracking the Sources of Performance Measurement Variation**", *University of Waterloo*, Waterloo, Ontario, Canada, 6 June 2014. (A large crowd of nearly 80 students and professors attended this talk).

International Conferences Attended

1. Supercomputing SC'17, November 2017, Denver CO.
   (Presented refereed workshop paper and refereed poster)
2. MEMSYS'17, October 2017, Washington, DC.
3. Linux Plumbers, November 2016, Santa Fe, NM.
   (Presented invited talk)
4. MEMSYS'16, October 2016, Washington, DC.
   (Presented refereed paper)
5. Supercomputing SC'15, November 2015, Austin TX.
   (Presented workshop paper)
6. ISPASS, March 2015, Philadelphia PA.
   (Presented refereed paper, was Session Chair for Mobile Devices Session)
7. Supercomputing SC'14, November 2014, New Orleans LA.
   (Presented refereed workshop paper at Co-HPC workshop)
8. ISPASS, April 2013, Austin, TX.
   (Presented a refereed paper, a poster, and a refereed paper at the Fastpath Workshop)
9. ICPP-41, September 2012, Pittsburgh, PA.
   (Presented refereed papers at PASA and CloudTech-HPC Workshops)

4. **Other Scholarly Activity**

Professional Society Membership

Member of ACM, IEEE and USENIX.

National Meetings Attended and Sessions Chaired

See "International Conferences Attended" in Section III.B.3.

Program Committee Membership

2017 MEMSYS Conference

2017 IEEE International Symposium on Performance Analysis of Systems and Software

2015 ACM International Conference on Computing Frontiers

2014, 2015 Co-HPC Workshop

External Conference and Journal Paper Reviewer

2016 Simulation Modeling Practice and Theory (SIMPAT) Journal

2014 International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)

2014 Computing Frontiers conference

2014 February IEEE Computer Architecture Letters

2014 EuroPar conference

2014 ACM Transactions on Architecture and Code Optimization (TACO)

5. **Statement on the Status of Candidate's Scholarly and Creative Work**

In my area of computer engineering (computer architecture and high performance computing) the foremost way of publishing work is in high-profile, selective, peer-reviewed conferences. Journal articles are secondary, although they are also held in high regard. Papers published in journals are usually extended conference articles rather than completely original works.

The ISPASS conference where I have published multiple times is considered one of the higher profile venues for performance measurement research. My work in this area was recognized by my being asked to serve on the Program Committee for the most recent iteration of this conference.

I also have numerous peer-reviewed workshop and poster publications; these are not as selective as the conference publications but are still important venues for introducing and getting feedback on ongoing cutting-edge research. In the constantly changing field of computer architecture the quick turnaround time available in workshops can allow rapid dissemination of important work. My highest cited work (with over 100 cites) is a 2012 ICPP Workshop paper on PAPI power measurement.

Google Scholar reports that between September 2012 and September 2017 I have had publications cited 602 times, with an h-index of 12 (I have 12 publications with at least 12 citations) as well as an i10-index of 15 (I have 15 publications with at least 10 citations).

I have reviewed papers for ACM TACO, which is considered the top journal in computer architecture, as well as the ASPLOS conference, with is considered one of the top computer architecture conferences.

My work in the Linux community has led to my becoming well-known, especially in regards to the `perf_event` performance counter interface. I am the author of the manual page for the `perf_event_open()` system call which is included in every Linux distribution. I maintain a test suite that is used by various groups and companies to validate that the interface is working. I have contributed code that has been included in the upstream Linux kernel codebase. I am also the most frequent recent committer of code to the PAPI performance library.

I have written a system call fuzzer, `perf_fuzzer` that has found numerous bugs in the Linux kernel. Some of these are major security bugs, affecting things such as Android phones. Security researchers have used this tool to find bugs that resulted in them being paid bug bounties by Google. While my work in this area is highly visible in the security industry, I am still working on getting it recognized in more traditional academic venues.

My Linux work has been presented and published in Linux conferences and journals. While high-profile among Linux developers, these are not subject to the strict peer review expected in academia. One article was published in The *Linux Weekly News*, the most prominent and respected publication for Linux programmers. I was also invited to present a talk at the *Linux Plumbers Conference* which is a top conference attended by high-profile developers involved in the important low-level "plumbing" of the Linux environment.

The classes I teach often involve final projects. To give the students a better idea of the expectations for the class, I often do a final project myself as an example. Some of these projects, especially the ones for embedded systems and network engineering, have been announced on various electronics hobby and maker websites. A few have been interesting enough to obtain more prominent media attention. These projects include:

- **Portal Ported to the Apple II** (12 January 2017)
  featured in Hackaday, The Register, Gizmodo, the A.V. Club, and made the front page of HackerNews.
- **Apple II BASIC Webserver** (17 December 2016)
  featured in Hackaday and Call-A.P.P.L.E. magazine.
- **Raspberry Pi Chiptune Player** (4 September 2016)
  featured in Hackaday, Adafruit blog, and the official Raspberry Pi Foundation blog.
- **Kerbal Space Program for the Apple II** (31 May 2016)
  featured in Hackaday.
- **Raspberry Pi PS/2 Keyboard** (14 January 2016)
  featured in Hackaday and the Adafruit blog.
- **Raspberry Pi Time Circuits** (11 November 2015)
  featured in Hackaday.
- **Falling Blocks Embedded Project** (14 December 2013)
  featured in Hackaday.

C. DOCUMENTATION OF RESEARCH/TRAINING GRANTS

I have applied for $3.3 million in grants from external funding agencies, of which $282,828 has been awarded.

i. External Grants Awarded

a. National Science Foundation
SI2-SSI: Collaborative Proposal: Performance Application Programming Interface for Extreme-scale Environments (PAPI-Ex)
Weaver (PI), Dongarra (PI)
`https://www.nsf.gov/awardsearch/showAward?AWD_ID=1450122`
September 2015 - August 2019
$274,828
I am the PI on this grant. It is a collaborative grant in conjunction with a group at the University of Tennessee Knoxville ($2 million on their side). The grant provides funding for one graduate student for four years, with additional funds for travel and equipment.

b. National Science Foundation
SI2-SSI: REU Supplement
Weaver (PI)
25 March 2016
$8,000
This REU extension provided an $8,000 supplement to my existing NSF grant in order to hire an undergraduate researcher for the summer.

ii. External Grants Pending

a. National Science Foundation
CAREER: Fuzzing the Future
Weaver (PI)
19 July 2017
$538,405

This grant proposes extending my Linux system call fuzzing work to find more bugs and to extend it into a system for automatic generation of hardware and software testing platforms. It requests support for one graduate student and two undergraduate students for five years.

iii. **External Grants Not Awarded**

a. Google Faculty Research Awards

Google will provide a gift that covers the tuition of one student.
Typically only 16% of proposals are funded.

I have applied three times:

Eliminating perf_event Security Bugs with Fuzzing
15 October 2016
$37,256

Targeted Linux Systemcall Fuzzing
15 October 2015
$37,556

Automatic Analysis of Performance Counter Overhead and Accuracy
15 October 2014
$36,519

b. National Science Foundation – Early Career Development Program

Three attempts are allowed before obtaining tenure. My last attempt is currently pending. Typically about 20% of proposals are funded.

I have attended UMaine's day-long CAREER proposal workshop in 2015, 2016, and 2017.

CAREER: Ubiquitous Embedded Supercomputing
Weaver (PI)
15 July 2015
$547,252

CAREER: Ubiquitous Embedded Supercomputing
Weaver (PI)
21 July 2014
$668,768

c. National Science Foundation – Secure and Trustworthy Cyberspace
TWC: Small: Enhancing Linux Security with Targeted Fuzzing
Weaver (PI)
13 January 2015
$405,832

d. Department of Energy – Early Career Research Program

This is one of the few remaining grants that the DoE provides to individual researchers. Three attempts are allowed before tenure; I had planned to apply once more in November 2017 but due to budget issues with the new administration it appears that the grant will not be offered for Fiscal Year 2018. In 2016 only 8% (59/700) of proposals were funded.

Optimizing Hardware Performance Counter Interfaces to Enable Automatic Performance Tuning
Weaver (PI)
20 November 2014
$750,038
This topic made it past the initial preproposal screening and a full proposal was requested.

iv. **Internal Grants Awarded**

a. University of Maine
Pre-Tenure Research and Creative Activity Fellowship
Reducing Wasted Energy in High Performance Computers
25 March 2014
$24,986

b. BSB Travel Grant
Travel to the 2014 Supercomputing Conference in New Orleans
15 October 2014
$1,500

v. **Internal Grants Not Awarded**

a. BSB Travel Grant
Travel to Linux Plumbers, November 2016
$1,295

b. UMaine Graduate School
Chase Distinguished Research Assistantships, 2016
$14,600

vi. **Future Funding Plans**

I plan to continue to apply for funding both from industry and from federal sources.

The NSF has various programs with small-sized grants available that are applicable to my research. This includes Secure and Trustworthy Cyberspace (SaTC), Computer Systems Research (CSR) and Computer and Network Systems (CNS). These typically have December deadlines.

I also plan to continue collaborating with the PAPI group at the University of Tennessee. Our collaboration is eligible for various larger software development grants.

D. DOCUMENTATION OF DEPARTMENT/CAMPUS/COLLEGE SERVICE

- Helped advise the renovation of Barrows Hall room 208 into new Computer Engineering research lab space.
- Served as Judge at the 2016 UMaine Graduate and Undergraduate Student Research Symposium.
- Served as Proctor for the 2017, 2016 and 2012 IEEE IEEExtreme 24-hour Programming Competition.
- Served as Judge at the 2015 and 2013 Grad Student Expos.
- Hosted guest speaker Jason Sewall, a UMaine Grad and Intel Employee, March 2015.
- ECE Faculty Meeting minutes note-taker 2012-2013.

E. DOCUMENTATION OF PUBLIC SERVICE

Contractually I am to devote 0% of my time to public service.

Compensated and Uncompensated Public Service

I contribute to various open source Linux projects, which involves collaborating with researchers and programmers from around the world. These open-source projects involve contributing back the code written, for free, so that anyone can use it.

These volunteer code contributions can be used by large companies and industry, but they are also used by non-profit organizations, as well as by average users around the world. This provides access to high-end software that would have not been affordable otherwise.

Organizations, such as the Raspberry Pi Foundation, have the goal of getting more students involved in programming and STEM education. These types of initiatives can take full advantage of the free open-source operating system and utilities and build upon the contributions and support from open-source coders like me.

F. DOCUMENTATION OF SPECIAL RECOGNITION/AWARDS

- Poster at SC'17 nominated for Best Poster award. (Only 9 out of 98 total were nominated)
- MEMSYS'16 Conference: "Most Gratuitous Use of Umlauts in a Scientific Presentation" award
- 2015 University of Maine Engineering Early Career Teaching Award

IV. EVALUATIONS OF TEACHING

A. STUDENT EVALUATIONS OF TEACHING

This summary has been verified by:

_____

Title: _____ Date: _____

1. Summary of quantitative student evaluations

   My teaching evaluations are consistently above the departmental averages.

   A table presenting a summary of the quantitative evaluations can be found on the following page: "Summary of Courses Taught and Students' Evaluations".

SUMMARY OF COURSES TAUGHT & STUDENTS' EVALUATIONS

Term and Year	Course ID Number	Course Credit Hours	Section Enroll-ment	Mean Evaluation Rating: This Course Q13	Q22	Q1	Q6	Q9	Q12	College Average N***	Q13	Q22	Q1	Q6	Q9	Q12
F2012	ECE498	1	5	4.60	4.60	4.80	4.80	5.00	4.60	5	4.19	3.91	4.46	4.11	4.45	4.03
S2013	ECE571	3	6	5.00	4.60	5.00	4.80	4.80	4.80	5	4.09	3.86	4.42	4.09	4.48	4.05
F2013	ECE471	3	9	4.00	4.00	4.56	4.33	4.89	4.11	9	4.19	3.94	4.49	4.17	4.47	4.10
S2014	ECE271	3	49	4.80	4.47	4.69	4.77	4.86	4.74	35	4.09	3.84	4.46	4.11	4.48	4.08
F2014	ECE471	3	21	4.71	4.64	4.86	4.71	4.93	4.71	15	4.18	3.95	4.42	4.15	4.51	4.09
F2014	ECE571	3	9	4.38	4.12	4.50	4.25	4.88	4.14	8	4.18	3.95	4.42	4.15	4.51	4.09
S2015	ECE214	2	41	4.58	4.08	4.46	4.27	4.73	4.54	26	4.05	3.77	4.38	4.07	4.47	4.04
S2015	ECE598	3	12	4.40	3.80	4.20	4.00	4.80	4.20	5	4.05	3.77	4.38	4.07	4.47	4.04
F2015	ECE471	3	22	4.69	4.69	4.69	4.44	4.87	4.44	16	4.21	3.96	4.47	4.19	4.53	4.15
F2015	ECE574	3	10	5.00	4.80	4.80	4.80	5.00	4.90	10	4.21	3.96	4.47	4.19	4.53	4.15
S2016	ECE571	3	16	5.00	4.67	5.00	4.83	5.00	4.00	6	4.03	3.82	4.37	4.00	4.44	3.90
S2016	ECE598	3	8	5.00	4.86	4.86	4.86	5.00	4.86	7	4.03	3.82	4.37	4.00	4.44	3.90
F2016	ECE435	3	13	5.00	4.90	4.50	4.90	5.00	4.90	11	4.14	3.93	4.48	4.11	4.39	4.09
F2016	ECE471	3	28	4.84	4.67	4.79	4.58	4.74	4.58	20	4.14	3.93	4.48	4.11	4.39	4.09
S2017	ECE571	3	8	4.86	5.00	4.86	5.00	4.86	4.71	8	4.24	3.97	4.49	4.13	4.49	4.12
S2017	ECE574	3	21	4.89	4.67	4.89	4.67	4.94	4.72	18	4.24	3.97	4.49	4.13	4.49	4.12

Ratings for questions 13 (overall rating of Instructor) and 22 (overall rating of course) are required. If other than a standard University evaluation form is used, please provide ratings for the equivalent questions, note their numbers and append a copy of the alternate form.
Ratings for four (4) additional questions, of your choosing, are also required. Please indicate their numbers in the spaces provided in the table and reference the text of each question in your narrative section on teaching.
***Number of students in this course section responding.

College and Departmental ratings are available from the Office of Institutional Studies.

Q13 : Overall, how would you rate the instructor?

Q22 : What is your overall rating of the class?

Q1 : How prepared was the instructor for class?

Q6 : How concerned was he instructor for the quality of his or her teaching?

Q9 : Did the instructor show respect for the questions and opinions of the students?

Q12 : How concerned was the instructor with the student's progress?

2. Summary of qualitative student evaluations

What follows are selected student comments from the end-of-semester course reviews. The reviews are anonymous by default, but students may optionally sign their names and then their comments are attributed to them. All of the comments included are from signed forms. At the suggestion of the tenure application directions I have not included names, but they can be found in the copies of the reviews on file.

- **ECE574 Spring 2017**
  - Excellent. I like this class. It teaches me the skills for parallel computing which are needed in my research.
  - This was a super fun class. Thanks.
  - Give Weaver a promotion or raise. He deserves it. Best teacher.
  - Vince is an amazing teacher.
  - Vince is among the greatest instructors at this University.
- **ECE571 Spring 2017**
  - Dr. Weaver is excellent.
  - Best Class ever. 10/10
  - Why can't Weaver teach all my courses?
- **ECE471 Fall 2016**
  - Best teacher in department.
  - This was an awesome and fun course.
  - Vince is one of my favorite professors. He is an excellent communicator. Explanations are clear.
- **ECE598 Spring 2016**
  - Weaver is so smart and really helpful. This class has given me so much practical knowledge. Probably my favorite elective I've taken so far.
- **ECE471 Fall 2015**
  - Dr. Weaver is one of the two best professors I've had at UMaine, and I've had dozens.
- **ECE471 Fall 2014**
  - One of my favorite professors in the department. Always available for help.
  - I enjoyed the class and found all of the information interesting and informative. I personally like Weaver's teaching style.
- **ECE471 Fall 2013**
  - Great class and teacher. Vince didn't stress the little details and focuses on teaching the big picture and concepts/understanding.

B. OTHER EVALUATIONS OF TEACHING

1. Peer Evaluations of Teaching

   Not applicable

2. Teaching Awards

   2015 University of Maine Engineering Early Career Teaching Award

3. Teaching of graduate students in the classroom and thesis advising

   I teach a number of 500-level graduate courses. The course reviews and comments for these classes are presented in the overall teaching evaluation section. Even though these courses are graduate level, a large number of higher-level undergraduates take them as well.

   I have directly advised four graduate students. This has led to publications with the students as primary authors. This includes a conference paper (MEMSYS 2016), a journal paper (Electronics), three workshop presentations (ESPT 2017, ESPT 2015, and Co-HPC 2014) and a poster (SC'17).

V. DEPARTMENTAL PEER COMMITTEE EVALUATION

A. EVALUATION LETTER

1. Evaluation of Teaching

2. Evaluation of Scholarship

3. Evaluation of Service

B. RECOMMENDATION/RECOMMENDED ACTION

VI. LETTERS OF REVIEW

A. LETTERS INTERNAL TO THE UNIVERSITY OF MAINE

None

B. **LETTERS INTERNAL TO THE UNIVERSITY OF MAINE SYSTEM, BUT EXTERNAL TO THE UNIVERSITY OF MAINE**

None

C. LETTERS EXTERNAL TO THE UNIVERSITY OF MAINE SYSTEM AND THE UNIVERSITY OF MAINE

1. Bruce Jacob, University of Maryland

   I attended the University of Maryland as an undergraduate and had Bruce Jacob as professor in two classes (one in 1998, the other in 2000). He had no advisory role with me while I was there. In 2017 I served on the Program Committee for the MEMSYS'17 conference alongside Bruce Jacob, who organized the conference.

2. Tao Xie, San Diego State University

   No known connection.

3. Jun Wang, University of Central Florida

   No known connection.

APPENDIX A: CURRICULUM VITAE

APPENDIX B: CURRICULA VITAE OF EXTERNAL REVIEWERS