Michael Hicks

Curriculum Vitae

I. Personal Information

I.A. UID, Last Name, First Name, Middle Name, Contact Information
UID: 112776114
Last Name: Hicks
First Name: Michael
Mailing Address: 5246 Iribe Center for Computer Science and Engineering (IRB), Department of Computer Science, University of Maryland, College Park, MD 20742
Email: mwh@cs.umd.edu
WWW: http://www.cs.umd.edu/~mwh/
Twitter: @michael_w_hicks
Blog: http://www.pl-enthusiast.net/

I.B. Academic Appointments at UMD
Professor, Department of Computer Science, July 2013–present
Associate Professor, Department of Computer Science, July 2008–2013
Affiliate Associate Professor, Department of Electrical and Computer Engineering, July 2008–2011
Assistant Professor, Department of Computer Science, July 2002–2008
Affiliate Assistant Professor, Department of Electrical and Computer Engineering, July 2005–2007.

I.C. Administrative Appointments at UMD
Associate Co-Chair for Undergraduate Education, Dept of Computer Science, September 2017–2020.
Director, Maryland Cybersecurity Center (MC2), November 2011–2013.

I.D. Other Employment
Consultant, Covington and Burling, LLP, August–November 2020.
Chief Technical Officer, Correct Computation, Inc., September 2018 to present.
Consultant, Stealth Software Technologies, Inc., June 2019–July 2020.
Consultant, Wilson Sonsini Goodrich and Rosati, February 2014–2017.
Visiting Researcher, Microsoft Research, Redmond, Washington, June–August 2015.
Consultant, Finnegan, Henderson, Farabow, Garrett & Dunner, LLP, February–October 2014.
Visiting Professor, *Ecole Normale Superieure*, Paris, France. October 2012.
Adjunct, IDA CCS, Bowie, Maryland. October 2006–March 2016.
Visiting Scholar, The Computer Laboratory, University of Cambridge, England. September 2008–August 2009.
Visiting Researcher, Microsoft Research, Cambridge, England. September 2008–November 2008.
Post-doctoral Research Associate, Department of Computer Science, Cornell University, Ithaca, New York. July 2001–July 2002.
Lecturer, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, Pennsylvania. August 1999–December 1999.
Scientist, NEC Research Institute, Princeton, New Jersey. May 1998–August 1999 (part time).
Software Engineer, ARINC, Inc., Annapolis, Maryland. June 1993–April 1997.

I.E. Educational Background
Ph.D. in Computer and Information Science, University of Pennsylvania, 2001 (Won ACM SIGPLAN

Doctoral Dissertation Award, 2002)
M.Sc. in Computer and Information Science, 1996.
B.Sc. in Computer Science with Honors; Pennsylvania State University, 1993 (Magna Cum Laude and Department Standard Bearer).

II. Research, Scholarly and Creative Activities

†Indicates a student or post-doc advised or co-advised by Dr. Hicks.
‡Indicates work performed in part while student advised or co-advised by Dr. Hicks.
⋆Indicates a student advised or co-advised by Dr. Hicks for this project.

II.B. Chapters
II.B.1 Books

1. Jose Manuel Calderon Trilla, Michael Hicks, Stephen Magill, Piotr Mardziel, and Ian Sweet†. Probabilistic Abstract Interpretation: Sound Inference and Application to Privacy. In *Foundations of Probabilistic Programming*, chapter 11, pages 361–389, Cambridge University Press, November 2020.

II.C. Articles in Refereed Journals

1. James Parker†, Michael Hicks, Andrew Ruef‡, Michelle L. Mazurek, Dave Levin, Daniel Votipka⋆, and Kelsey Fulton⋆. Build It, Break It, Fix It: Contesting Secure Development. *ACM Transactions on Privacy and Security (TOPS*), 23(2), April 2020.
2. Benjamin C. Pierce, Michael Hicks, Crista Lopes, and Jens Palsberg. ACM Conferences and the Cost of Carbon. *Communications of the ACM*, 63(3):35-37, March 2020.
3. Karla Saur†, Michael Hicks, and Jeffrey S. Foster. C-Strider: Type-Aware Heap Traversal for C. *Software, Practice, and Experience*, May 2015.
4. Christopher M. Hayden‡, Karla Saur†, Edward K. Smith†, Michael Hicks, and Jeffrey S. Foster. Efficient, general-purpose dynamic software updating for C. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 36(4):13, October 2014.
5. Piotr Mardziel†, Stephen Magill†, Michael Hicks, and Mudhakar Srivatsa. Dynamic enforcement of knowledge-based security policies using probabilistic abstract interpretation. *Journal of Computer Security*, 21:463–532, October 2013.
6. Christopher M. Hayden†, Edward K. Smith†, Eric A. Hardisty⋆, Michael Hicks, and Jeffrey S. Foster. Evaluating dynamic software update safety using efficient systematic testing. *IEEE Transactions on Software Engineering*, 38(6):1340–1354, December 2012.
7. Polyvios Pratikakis‡, Jeffrey S. Foster, and Michael Hicks. Locksmith: Practical static race detection for C. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 33(1):Article 3, January 2011.
8. Michael Hicks and Jeffrey S. Foster. SCORE: Agile research group management. *Communications of the ACM*, 53(10):30–31, October 2010.
9. Jeffrey A. Meister⋆, Jeffrey S. Foster, and Michael Hicks. Serializing C intermediate representations for efficient and portable parsing. *Software, Practice, and Experience*, 40(3):225–238, February 2010.
10. Peter Sewell, Gareth Stoyle⋆, Michael Hicks, Gavin Bierman, and Keith Wansbrough. Dynamic rebinding for marshalling and update, via redex-time and destruct-time reduction. *Journal of Functional Programming (JFP)*, 18(4):437–502, July 2008.

11. Saurabh Srivastava†, Michael Hicks, Jeffrey S. Foster, and Patrick Jenkins⋆. Modular information hiding and type safe linking for C. *IEEE Transactions on Software Engineering*, 34(3):1–20, May 2008.
12. Gareth Stoyle⋆, Michael Hicks, Gavin Bierman, Peter Sewell, and Iulian Neamtiu†. *Mutatis Mutandis*: Safe and flexible dynamic software updating (full version). *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 29(4), July 2007.
13. Nikhil Swamy†, Michael Hicks, Greg Morrisett, Dan Grossman, and Trevor Jim. Safe manual memory management in Cyclone. *Science of Computer Programming (SCP)*, 62(2):122–144, October 2006. Special issue on memory management.
14. Michael Hicks and Scott M. Nettles. Dynamic software updating. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 27(6):1049–1096, November 2005.
15. James Rose†, Nikhil Swamy†, and Michael Hicks. Dynamic inference of polymorphic lock types. *Science of Computer Programming (SCP)*, 58(3):366–383, December 2005. Special Issue on Con- currency and Synchronization in Java programs.
16. Michael Hicks, Angelos D. Keromytis, and Jonathan M. Smith. A secure PLAN. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 33(3):413–426, August 2003. Special Issue on Technologies Promoting Computational Intelligence, Openness and Programmability in Networks and Internet Services.
17. D. Scott Alexander, William A. Arbaugh, Michael Hicks, Pankaj Kakkar, Angelos Keromytis, Jonathan T. Moore, Carl A. Gunter, Scott M. Nettles, and Jonathan M. Smith. The SwitchWare active network architecture. *IEEE Network*, 12(3):29–36, 1998. Special issue on Active and Controllable Networks.

II.D. Published Conference Proceedings

II.D.1. Refereed Conference Proceedings

1. Michael Coblenz†, Michelle Mazurek, and Michael Hicks. Does the Bronze Garbage Collector Make Rust Easier to Use? A Controlled Experiment. In *Proceedings of the International Conference on Software Engineering (ICSE)*, May 2022. Acceptance rate 197/751 (26%).
2. Kelsey Fulton‡, Anna Chan‡, Dan Votipka, Michael Hicks, and Michelle Mazurek. Benefits and Drawbacks of Adopting a Secure Programming Language: Rust as a Case Study. In *Proceedings of the Symposium on Usable Privacy and Security (SOUPS)*, August 2021. Acceptance rate 36/136 (26%)
3. Ian Sweet†, David Darais, David Heath, Ryan Estes, William Harris, and Michael Hicks. Symphony: A Concise Language Model for MPC. In *Informal Proceedings of the Workshop on Foundations on Computer Secuirty (FCS)*, June 2021.
4. Kesha Hietala†, Robert Rand‡, Shih-Han Hung‡, Liyi Li†, and Michael Hicks. Proving Quantum Programs Correct. In *Proceedings of the Conference on Interative Theorem Proving (ITP)*, June 2021. Acceptance rate 28/51 (55%).
5. Souvik Bhattacherjee‡, Gang Liao‡, Michael Hicks, and Daniel J. Abadi. BullFrog: Online Schema Evolution via Lazy Evaluation. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD)*, June 2021. Acceptance rate 188/450 (41.8%).
6. Finn Voichick† and Michael Hicks. Toward A Quantum Programming Language for Higher-Level Formal Verification. In *Informal Proceedings of the Workshop on Programming Languages and Quantum Computing (PLanQC)*, June 2021.
7. Kesha Hietala†, Liyi Li†, Akshaj Gaur‡, Aaron Green‡, Robert Rand, Xiaodi Wu, and Michael Hicks. Applying and Expanding the VOQC Toolkit.In *Informal Proceedings of the Workshop on Programming Languages and Quantum Computing (PLanQC)*, June 2021.
8. Kesha Hietala†, Robert Rand†, Shih-Han Huang‡, Xiaodi Wu, and Michael Hicks. A Verified Optimizer for Quantum Circuits. In *Proceedings of the ACM Conference on Principles of*

Programming Languages (POPL), January 2021. Acceptance rate 61/258 (23.6%). **Distinguished Paper** (7/61).

9. Ian Sweet†, David Darais, and Michael Hicks. Short Paper: Probabilistically Almost-Oblivious Computation. In *Proceedings of the ACM SIGPLAN Workshop on Programming Languages and Analysis for Security (PLAS)*, November 2020.
10. Yiyun Liu†, James Parker†, Patrick Redmond, Lindsey Kuper, Michael Hicks, and Niki Vazou. Verifying Replicated Data Types with Typeclass Refinements in Liquid Haskell. In *Proceedings of the ACM Conference on Object-Oriented Programming Languages, Systems, and Applications (OOPSLA)*, October 2020.
11. Daniel Votipka⋆, Kelsey Fulton⋆, James Parker†, Matthew Hou†, Michelle L. Mazurek, and Michael Hicks. Understanding security mistakes developers make: Qualitative analysis from Build It, Break It, Fix It. In *Proceedings of the USENIX Security Symposium (USENIX SEC)*, August 2020. Acceptance rate 157/977 (16.1%). **Distinguished Paper** (11/157).
12. David Darais‡, Ian Sweet†, Chang Liu‡, and Michael Hicks. A Language for Probabilistically Oblivious Computation. In *Proceedings of the ACM Conference on Principles of Programming Languages (POPL)*, January 2020. Acceptance rate 68/247 (27.5%)
13. Leonidas Lampropoulos†, Michael Hicks, and Benjamin C. Pierce. Coverage Guided, Property Based Testing. In *Proceedings of the ACM Conference on Object-Oriented Programming Languages, Systems, and Applications (OOPSLA)*, October 2019. Acceptance rate 72/201 (36%)
14. Robert Rand†, Kesha Hietala†, and Michael Hicks. Formal Verification vs. Quantum Uncertainty. In *3rd Summit on Advances in Programming Languages (SNAPL)*, May 2019. Acceptance rate 9/21 (42%).
15. Aseem Rastogi‡, Nikhil Swamy, and Michael Hicks. Wys*: A DSL for Verified Secure Multi-party Computations. In *Proceedings of the Symposium on Principles of Security and Trust (POST)*, April 2019. Acceptance rate 10/37 (37%)
16. Andrew Ruef†, Leonidas Lampropoulos†, Ian Sweet†, David Tarditi, and Michael Hicks. Achieving Safety Incrementally with Checked C. In *Proceedings of the Symposium on Principles of Security and Trust (POST)*, April 2019. Acceptance rate 10/37 (37%)
17. Luis Pina, Anastasios Andronidis, Michael Hicks, and Cristian Cadar. MVEDSUa: Higher availability dynamic software updates via multi-version execution. In *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, April 2019. Acceptance rate 74/350 (21.1%)
18. James Parker†, Niki Vazou†, and Michael Hicks. LWeb: Information Flow Security for Multi-Tier Web Applications. In *Proceedings of the ACM Conference on Principles of Programming Languages (POPL)*, January 2019. Acceptance rate 77/267 (28.8%)
19. Shih-Han Hung⋆, Kesha Hietala†, Shaopeng Zhu⋆, Mingsheng Ying, Michael Hicks, and Xiaodi Wu. Quantitative Robustness Analysis of Quantum Programs. In *Proceedings of the ACM Conference on Principles of Programming Languages (POPL)*, January 2019. Acceptance rate 77/267 (28.8%)
20. George T. Klees†, Andrew Ruef†, Benjamin Cooper†, Shiyi Wei†, and Michael Hicks. Evaluating Fuzz Testing. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, October 2018. Acceptance rate 134/809 (16.6%) **Winner of the NSA's 2018 Best Scientific Cybersecurity Paper Competition**
21. Archibald Samuel Elliott⋆, Andrew Ruef†, Michael Hicks, and David Tarditi. Checked C: Making C Safe by Extension. In Proceedings of the *IEEE Conference on Secure Development (SecDev)*, September 2018. Acceptance rate 14/29 (48%)
22. Ian Sweet†, Jose Manuel Calderon Trilla, Chad Scherrer, Michael Hicks, and Stephen Magill. What's the over/under? probabilistic bounds on information leakage. In *Proceedings of the Symposium on Principles of Security and Trust (POST)*, April 2018. Acceptance rate 14/45 (31%)

23. Shiyi Wei†, Piotr Mardziel, Andrew Ruef†, Jeffrey S. Foster, and Michael Hicks. Evaluating Design Tradeoffs in Numeric Static Analysis for Java. In *Proceedings of the European Symposium on Programming (ESOP)*, April 2018. Acceptance rate 36/114 (32%)
24. ThanhVu Nguyen, Timos Antopoulos, Andrew Ruef†, and Michael Hicks. A counterexample-guided approach to finding numerical invariants. In *Proceedings of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE)*, September 2017. Acceptance rate 72/295 (24%)
25. Timos Antonopoulos, Paul Gazzillo, Michael Hicks, Eric Koskinen, Tachio Terauchi, and Shiyi Wei†. Decomposition Instead of Self-Composition for Proving the Absence of Timing Channels. In *Proceedings of the ACM Conference on Programming Language Design and Implementation (PLDI)*, June 2017. Acceptance rate 47/322 (14.6%)
26. Mário S. Alvim, Piotr Mardziel, and Michael Hicks. Quantifying vulnerability of secret generation using hyper-distributions. In *Proceedings of the Symposium on Principles of Security and Trust (POST)*, April 2017. Acceptance rate 14/40 (35%)
27. Andrew Ruef†, Michael Hicks, James Parker†, Dave Levin, Michelle L. Mazurek, and Piotr Mardziel†. Build It, Break It, Fix It: Contesting Secure Development. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, October 2016. Acceptance rate 137/831 (16.5%)
28. Karla Saur†, Tudor Dumitras, and Michael Hicks. Evolving NoSQL databases without downtime. In *Proceedings of the International Conference on Software Maintenance and Evolution (ICSME)*, October 2016. Acceptance rate 37/127 (29%)
29. Luis Pina† and Michael Hicks. *Tedsuto: A General Framework for Testing Dynamic Software Updates.* In *Proceedings of the International Conference on Software Testing (ICST)*, April 2016. Acceptance rate 35/130 (26.9%)
30. Karla Saur†, Joseph Collard‡, Nate Foster, Arjun Guha, Laurent Vanbever, and Michael Hicks. *Safe and Flexible Controller Upgrades for SDNs.* In *Proceedings of the Symposium on SDN Research (SOSR)*, March 2016. Acceptance rate 17/69 (24.6%)
31. Matthew Hammer†, Kyle Headley†, Nicholas Labich‡, Jeffrey S. Foster, Michael Hicks, David Van Horn, and Joshua Dunfield. Incremental Computation with Names. In *Proceedings of the ACM Conference on Object-Oriented Programming Languages, Systems, and Applications (OOPSLA)*, October 2015. Acceptance rate 53/210 (25.2%)
32. Andrew Ruef†, Michael Hicks, James Parker†, Dave Levin, Atif Memon, Jandelyn Plane, and Piotr Mardziel†. Build It Break It: Measuring and Comparing Development Security. In *Proceedings of the USENIX Workshop on Cyber Security Instrumentation and Test (CSET)*, August 2015.
33. Chang Liu†, Michael Hicks, Austin D Harris, Mohit Tiwari, Martin Maas, and Elaine Shi. Ghostrider: A hardware-software system for memory trace oblivious computation. In *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, March 2015. Acceptance rate 48/278 (17.3%). **Winner of best paper award.**
34. Luıs Pina†, Luıs Veiga, and Michael Hicks. Rubah: Dynamic Software Updating for Java on a stock JVM. In *Proceedings of the ACM Conference on Object-Oriented Programming Languages, Systems, and Applications (OOPSLA)*, October 2014. Acceptance rate 52/181 (28.7%).
35. Piotr Mardziel†, Mário S. Alvim, and Michael Hicks. Adversary gain vs. defender loss in quantified information flow. In *(Unofficial) Proceedings of the International Workshop on Foundations of Computer Security (FCS)*, July 2014.
36. Matthew Hammer†, Yit Phang Khoo†, Michael Hicks, and Jeffrey S. Foster. Adapton: Composable, demand-driven incremental computation. In *Proceedings of the ACM Conference on Programming Language Design and Implementation (PLDI)*, June 2014. Acceptance rate 52/287 (18.1%)

37. Chang Liu†, Yan Huang, Elaine Shi, Jonathan Katz, and Michael Hicks. Automating efficient RAM- model secure computation. In *Proceedings of the IEEE Symposium on Security and Privacy (Oak- land)*, May 2014. Acceptance rate 44/324 (13.6%)
38. Aseem Rastogi†, Matthew A. Hammer†, and Michael Hicks. Wysteria: A programming language for generic, mixed-mode multiparty computations. In *Proceedings of the IEEE Symposium on Security and Privacy (Oakland)*, May 2014. Acceptance rate 44/324 (13.6%)
39. Piotr Mardziel†, Mario Alvim, Michael Hicks, and Michael Clarkson. Quantifying information flow for dynamic secrets. In *Proceedings of the IEEE Symposium on Security and Privacy (Oakland)*, May 2014. Acceptance rate 44/324 (13.6%).
40. Andrew Miller†, Michael Hicks, Jonathan Katz, and Elaine Shi. Authenticated data structures, generically. In *Proceedings of the ACM Conference on Principles of Programming Languages (POPL)*, January 2014. Acceptance rate 51/220 (23.2%).
41. Aseem Rastogi†, Piotr Mardziel†, Matthew Hammer†, and Michael Hicks. Knowledge inference for optimizing secure multi-party computation. In *Proceedings of the ACM SIGPLAN Workshop on Programming Languages and Analysis for Security (PLAS)*, June 2013.
42. Chang Liu†, Michael Hicks, and Elaine Shi. Memory trace oblivious program execution. In *Proceedings of the Computer Security Foundations Symposium (CSF)*, June 2013. Acceptance rate 19/73 (26%). **Winner of the NSA's 2014 Best Scientific Cybersecurity Paper Competition**
43. Luis Pina and Michael Hicks. Rubah: Efficient, general-purpose dynamic software updating for Java. In *Proceedings of the Workshop on Hot Topics in Software Upgrades (HotSWUp)*, June 2013.
44. Yit Phang Khoo,†Jeffrey S. Foster, and Michael Hicks. Expositor: Scriptable time-travel debugging with first class traces. In *Proceedings of the International Conference on Software Engineering (ICSE)*, May 2013. Acceptance rate 85/461 (18.5%).
45. Mudhakar Srivatsa and Michael Hicks. Deanonymizing mobility traces: Using a social network as a side-channel. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, October 2012. Acceptance rate 80/423 (18.9%).
46. Christopher M. Hayden†, Edward K. Smith†, Michail Denchev⋆, Michael Hicks, and Jeffrey S. Foster. Kitsune: Efficient, general-purpose dynamic software updating for C. In *Proceedings of the ACM Conference on Object-Oriented Programming Languages, Systems, and Applications (OOPSLA)*, October 2012. Acceptance rate 57/228 (25%).
47. Stephen Magill⋆, Michael Hicks, Suriya Subramanian⋆, and Kathryn S. McKinley. Automating object transformations for dynamic software updating. In *Proceedings of the ACM Conference on Object-Oriented Programming Languages, Systems, and Applications (OOPSLA)*, October 2012. Acceptance rate 57/228 (25%).
48. Piotr Mardziel†, Michael Hicks, Jonathan Katz, and Mudhakar Srivatsa. Knowledge-oriented secure multiparty computation. In *Proceedings of the ACM SIGPLAN Workshop on Programming Languages and Analysis for Security (PLAS)*, June 2012. Acceptance rate 9/12 (75%). Number of submissions lower than usual due to location (China).
49. Edward K. Smith†, Michael Hicks, and Jeffrey S. Foster. Towards standardized benchmarks for dynamic software updating systems. In *Proceedings of the Workshop on Hot Topics in Software Upgrades (HotSWUp)*, pages 11–15, June 2012. Acceptance rate 9/13 (69%).
50. Christopher M. Hayden†, Karla Saur†, Michael Hicks, and Jeffrey S. Foster. A study of dynamic software update quiescence for multithreaded programs. In *Proceedings of the Workshop on Hot Topics in Software Upgrades (HotSWUp)*, pages 6–10, June 2012. Acceptance rate 9/13 (69%).
51. Christopher M. Hayden†, Stephen Magill†, Michael Hicks, Nate Foster, and Jeffrey S. Foster. Specifying and verifying the correctness of dynamic software updates. In *Proceedings of the International Conference on Verified Software: Theories, Tools, and Experiments (VSTTE)*, pages 278–293, January 2012. Acceptance rate 20/54 (37%).

52. Nataliya Guts†, Michael Hicks, Nikhil Swamy, and Daan Leijen. A demo of Coco: a compiler for monadic coercions in ML. In *Informal proceedings of the ML Workshop*, September 2011.
53. Nikhil Swamy, Nataliya Guts†, Daan Leijen, and Michael Hicks. Lightweight monadic programming in ML. In *Proceedings of the ACM International Conference on Functional Programming (ICFP)*, pages 15–27, September 2011. Acceptance rate 33/92 (35.9%). Featured on the *Lambda the Ultimate* blog, July 2011
54. Kin-Keung Ma†, Yit Phang Khoo†, Jeffrey S. Foster, and Michael Hicks. Directed symbolic execution. In Eran Yahav, editor, *Proceedings of the Static Analysis Symposium (SAS)*, volume 6887 of *Lecture Notes in Computer Science*, pages 95–111. Springer, September 2011. Acceptance rate 22/67 (32.8%).
55. Piotr Mardziel†, Stephen Magill†, Michael Hicks, and Mudhakar Srivatsa. Dynamic enforcement of knowledge-based security policies. In *Proceedings of the Computer Security Foundations Symposium (CSF)*, pages 114–128, June 2011. Acceptance rate 21/81 (25.9%).
56. Christopher M. Hayden†, Edward K. Smith†, Michael Hicks, and Jeffrey S. Foster. State transfer for clear and efficient runtime upgrades. In *Proceedings of the Workshop on Hot Topics in Software Upgrades (HotSWUp)*, pages 179–184, April 2011. Acceptance rate 9/17 (52.9%).
57. Jong hoon (David) An†, Avik Chaudhuri⋆, Jeffrey S. Foster, and Michael Hicks. Position paper: Dynamically inferred types for dynamic languages. In *Informal Proceedings of the Workshop on Scripts to Programs (STOP)*, January 2011.
58. Jong hoon (David) An†, Avik Chaudhuri⋆, Jeffrey S. Foster, and Michael Hicks. Dynamic inference of static types for Ruby. In *Proceedings of the ACM Conference on Principles of Programming Languages (POPL)*, pages 459–472, January 2011. Acceptance rate 49/209 (23.4%).
59. Piotr Mardziel† , Adam Bender⋆, Michael Hicks, Dave Levin⋆, Mudhakar Srivatsa, and Jonathan Katz. Secure sharing in distributed information management applications: problems and directions. In *Proceedings of the Annual Conference of the International Technology Alliance (ACITA)*, September 2010.
60. Jean-Phillipe Martin, Michael Hicks, Manuel Costa, Periklis Akritidis, and Miguel Castro. Dynamically checking ownership policies in concurrent C/C++ programs. In *Proceedings of the ACM Con- ference on Principles of Programming Languages (POPL)*, pages 457–470, January 2010. Acceptance rate 39/206 (19%).
61. Yit Phang Khoo†, Jeffrey S. Foster, Michael Hicks, and Vibha Sazawal. Triaging checklists: a substitute for a PhD in static analysis. In *Proceedings of the Workshop on the Evaluation and Usability of Programming Languages and Tools (PLATEAU)*, October 2009.
62. Yit Phang Khoo†, Michael Hicks, Jeffrey S. Foster, and Vibha Sazawal. Directing javascript with arrows. In *Proceedings of the ACM SIGPLAN Dynamic Languages Symposium (DLS)*, pages 49–58, October 2009. Acceptance rate 10/27 (37%).
63. Michael Furr⋆, Jong hoon (David) An⋆, Jeffrey S. Foster, and Michael Hicks. The Ruby intermediate language. In *Proceedings of the ACM SIGPLAN Dynamic Languages Symposium (DLS)*, pages 89– 98, October 2009. Acceptance rate 10/27 (37%).
64. Nikhil Swamy, Michael Hicks, and Gavin S. Bierman. A theory of typed coercions and its applications. In *Proceedings of the ACM International Conference on Functional Programming (ICFP)*, pages 329–340, August 2009. Acceptance rate 26/85 (31%).
65. Pavlos Papageorge†, Justin McCann⋆, and Michael Hicks. Passive aggressive measurement with MGRP. In *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)*, pages 279–290, August 2009. Acceptance rate 27/270 (10%).
66. Brian J. Corcoran†, Nikhil Swamy‡, and Michael Hicks. Cross-tier, label-based security enforcement for web applications. In *Proceedings of the ACM SIGMOD International*

Conference on Management of Data (SIGMOD), pages 269–282. June 2009. Acceptance rate 63/397 (16%).

67. Iulian Neamtiu‡ and Michael Hicks. Safe and timely dynamic updates for multi-threaded programs. In *Proceedings of the ACM Conference on Programming Language Design and Implementation (PLDI)*, pages 13–24. June 2009. Acceptance rate 41/196 (21%).
68. Suriya Subramanian⋆, Michael Hicks, and Kathryn S. McKinley. Dynamic software updates for Java: A VM-centric approach. In *Proceedings of the ACM Conference on Programming Language Design and Implementation (PLDI)*, pages 1–12. June 2009. Acceptance rate 41/196 (21%).
69. Michael Furr⋆, Jong hoon (David) An⋆, Jeffrey S. Foster, and Michael Hicks. Static type inference for Ruby. In *Proceedings of the ACM Symposium on Applied Computing, Object-oriented Programming Languages and Systems Track (OOPS)*, pages 1859–1866, March 2009. Acceptance rate 4/9 (44%) for OOPS track, 315/1084 (29%) overall.
70. Dave King⋆, Boniface Hicks, Michael Hicks, and Trent Jaeger. Implicit flows: Can't live with 'em, can't live without 'em. In R. Sekar and Arun K. Pujari, editors, *Proceedings of the International Conference on Information Systems Security (ICISS)*, volume 5352 of *Lecture Notes in Computer Science*, pages 56–70. Springer, December 2008. Acceptance rate 21/81 (26%).
71. Yit Phang Khoo†, Jeffrey S. Foster, Michael Hicks, and Vibha Sazawal. Path projection for user-centered static analysis tools. In *Proceedings of the ACM Workshop on Program Analysis for Software Tools and Engineering (PASTE)*, pages 57–63, November 2008. Acceptance rate 13/26 (50%).
72. Polyvios Pratikakis†, Jeffrey S. Foster, Michael Hicks, and Iulian Neamtiu†. Formalizing soundness of contextual effects. In Otmane Aıt Mohamed, Cesar Munoz, and Sofiene Tahar, editors, *Proceedings of the International Conference on Theorem Proving in Higher Order Logics (TPHOLs)*, volume 5170 of *Lecture Notes in Computer Science*, pages 262–277. Springer, August 2008. Acceptance rate 16/45 (36%).
73. Nikhil Swamy† and Michael Hicks. Verified enforcement of automaton-based information release policies. In *Proceedings of the ACM SIGPLAN Workshop on Programming Languages and Analysis for Security (PLAS)*, pages 21–32, June 2008. Chosen as one of two **Distinguished papers** for the workshop. Acceptance rate 13/24 (54%).
74. Nikhil Swamy†, Brian Corcoran†, and Michael Hicks. Fable: A language for enforcing user-defined security policies. In *Proceedings of the IEEE Symposium on Security and Privacy (Oakland)*, pages 369–383, May 2008. Acceptance rate 28/249 (11.2%).
75. Iulian Neamtiu†, Michael Hicks, Jeffrey S. Foster, and Polyvios Pratikakis†. Contextual effects for version-consistent dynamic software updating and safe concurrent programming. In *Proceedings of the ACM Conference on Principles of Programming Languages (POPL)*, pages 37–50. January 2008. Acceptance rate 35/212 (16.5%).
76. Brian Corcoran†, Nikhil Swamy†, and Michael Hicks. Combining provenance and security policies in a web-based document management system. In *On-line Proceedings of the Workshop on Principles of Provenance (PrOPr)*, November 2007. http://homepages.inf.ed.ac.uk/jcheney/propr/.
77. Nick L. Petroni, Jr.†and Michael Hicks. Automated detection of persistent kernel control-flow attacks. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 103–115, October 2007. Acceptance rate 55/303 (18%).
78. Nikhil Swamy†, Michael Hicks, and Simon Tsang. Verified enforcement of security policies for cross-domain information flows. In *Proceedings of the 2007 Military Communications Conference (MILCOM)*, October 2007.
79. Jeffrey S. Foster, Michael W. Hicks, and William Pugh. Improving software quality with static analysis. In *Proceedings of the ACM Workshop on Program Analysis for Software Tools and*

Engineering (PASTE), pages 83–84, June 2007. Acceptance rate 4/4 (100%) for research presentations category.

80. Trevor Jim, Nikhil Swamy†, and Michael Hicks. Defeating scripting attacks with browser-enforced embedded policies. In *Proceedings of the International World Wide Web Conference (WWW)*, pages 601–610, May 2007. Acceptance rate 10/63 (16%) for the security track, 111/740 (15%) overall.
81. Saurabh Srivastava†, Michael Hicks, and Jeffrey S. Foster. Modular information hiding and type safety for C. In *Proceedings of the ACM Workshop on Types in Language Design and Implementation (TLDI)*, pages 3–14, January 2007. Acceptance rate 6/12 (50%).
82. Polyvios Pratikakis†, Jeffrey S. Foster, and Michael Hicks. Existential label flow inference via CFL reachability. In Kwangkeun Yi, editor, *Proceedings of the Static Analysis Symposium (SAS)*, volume 4134 of *Lecture Notes in Computer Science*, pages 88–106. Springer-Verlag, August 2006. Acceptance rate 23/80 (29%).
83. Nikhil Swamy†, Michael Hicks, Stephen Tse⋆, and Steve Zdancewic. Managing policy updates in security-typed languages. In *Proceedings of the Computer Security Foundations Workshop (CSFW)*, pages 202–216, July 2006. Acceptance rate 25/96 (26%). Note: though termed a workshop, based on citation rate and number of submissions, CSFW has effectively been a conference for many years; as of 2007, CSFW has been renamed CSF and is termed a symposium.
84. Iulian Neamtiu†, Michael Hicks, Gareth Stoyle⋆, and Manuel Oriol†. Practical dynamic software updating for C. In *Proceedings of the ACM Conference on Programming Language Design and Implementation (PLDI)*, pages 72–83, June 2006. Acceptance rate 36/174 (21%).
85. Polyvios Pratikakis†, Jeffrey S. Foster, and Michael Hicks. Context-sensitive correlation analysis for detecting races. In *Proceedings of the ACM Conference on Programming Language Design and Implementation (PLDI)*, pages 320–331, June 2006. Acceptance rate 36/174 (21%).
86. Boniface Hicks, Dave King, Patrick McDaniel, and Michael Hicks. Trusted declassification: high- level policy for a security-typed language. In *Proceedings of the ACM SIGPLAN Workshop on Programming Languages and Analysis for Security (PLAS)*, pages 65–74, June 2006. Acceptance rate 10/18 (56%).
87. Michael Hicks, Jeffrey S. Foster, and Polyvios Pratikakis†. Inferring locking for atomic sections. In *On-line Proceedings of the ACM SIGPLAN Workshop on Languages, Compilers, and Hardware Support for Transactional Computing (TRANSACT)*, June 2006. http://www.cs.purdue.edu/ homes/jv/events/TRANSACT/transact-06.tgz. Acceptance rate 10/19 (53%).
88. Michael Hicks, Boniface Hicks, Stephen Tse, and Steve Zdancewic. Dynamic updating of information- flow policies, March 2005. In *Proceedings of the International Workshop on Foundations of Computer Security (FCS)*, pages 7–18, June 2005. Acceptance rate 11/30 (37%).
89. Iulian Neamtiu†, Jeffrey S. Foster, and Michael Hicks. Understanding source code evolution using abstract syntax tree matching. In *Proceedings of the International Workshop on Mining Software Repositories (MSR)*, pages 1–5, May 2005. Acceptance rate 22/38 (58%).
90. Manuel Oriol† and Michael Hicks. Tagged sets: a secure and transparent coordination medium. In Jean-Marie Jacquet and Gian Pietro Picco, editors, *Proceedings of the International Conference on Coordination Models and Languages (COORDINATION)*, volume 3454 of *Lecture Notes in Computer Science*, pages 252–267. Springer-Verlag, April 2005. Acceptance rate 19/88 (22%).
91. Pavlos Papageorgiou† and Michael Hicks. Merging network measurement with data transport (extended abstract). In *Proceedings of the IEEE Passive/Active Measurement Workshop (PAM)*, volume 3431, pages 368–371. Springer-Verlag, March 2005. Acceptance rate 36/84 (43%).
92. Gareth Stoyle⋆, Michael Hicks, Gavin Bierman, Peter Sewell, and Iulian Neamtiu†. *Mutatis Mutandis*: Safe and flexible dynamic software updating. In *Proceedings of the ACM Conference*

on Principles of Programming Languages (POPL), pages 183–194, January 2005. Acceptance rate 31/172 (18%).

93. Polyvios Pratikakis†, Jaime Spacco⋆, and Michael Hicks. Transparent proxies for Java futures. In *Proceedings of the ACM Conference on Object-Oriented Programming Languages, Systems, and Applications (OOPSLA)*, pages 206–223, October 2004. Acceptance rate 27/173 (16%).
94. Michael Hicks, Greg Morrisett, Dan Grossman, and Trevor Jim. Experience with safe manual memory management in cyclone. In *Proceedings of the ACM International Symposium on Memory Management (ISMM)*, pages 73–84, October 2004. Acceptance rate 15/43 (35%).
95. James Rose†, Nikhil Swamy†, and Michael Hicks. Dynamic inference of polymorphic lock types. In *Proceedings of the ACM Conference on Principles of Distributed Computing (PODC) Workshop on Concurrency and Synchronization in Java Programs (CSJP)*, pages 18–25, July 2004. Acceptance rate 11/24 (46%).
96. Gavin Bierman, Michael Hicks, Peter Sewell, Gareth Stoyle, and Keith Wansbrough. Dynamic rebinding for marshalling and update with destruct-time λ. In *Proceedings of the ACM International Conference on Functional Programming (ICFP)*, pages 99–110, August 2003. Acceptance rate 24/95 (25%).
97. Gavin Bierman, Michael Hicks, Peter Sewell, and Gareth Stoyle. Formalizing dynamic software updating. In *On-line Proceedings of the Second International Workshop on Unanticipated Software Evolution (USE)*, April 2003. http://www.informatik.uni-bonn.de/~gk/use/2003/Papers/papers.html.
98. Michael Hicks, Adithya Nagarajan†, and Robbert van Renesse. User-specified adaptive scheduling in a streaming media network. In *Proceedings of the IEEE Conference on Open Architectures and Network Programming (OPENARCH)*, pages 87–96. April 2003. Acceptance rate 12/50 (24%).
99. Seo-Kyu Song, Stephen Shannon, Michael Hicks, and Scott Nettles. Evolution in action: Using active networking to evolve network support for mobility. In James Sterbenz, Osamu Takada, Christian Tschudin, and Bernhard Plattner, editors, *Proceedings of the Fourth International Working Conference on Active Networks (IWAN)*, volume 2546 of *Lecture Notes in Computer Science*, pages 146–161. Springer-Verlag, December 2002. Acceptance rate 20/53 (38%).
100. Trevor Jim, Greg Morrisett, Dan Grossman, Michael Hicks, James Cheney, and Yanling Wang. Cy- clone: A safe dialect of C. In *Proceedings of the USENIX Annual Technical Conference*. USENIX, pages 275–288. June 2002. Acceptance rate 25/107 (23%).
101. Dan Grossman, Greg Morrisett, Trevor Jim, Michael Hicks, Yanling Wang, and James Cheney. Region-based memory management in Cyclone. In *Proceedings of the ACM Conference on Programming Language Design and Implementation (PLDI)*, pages 282–293. ACM, June 2002. Acceptance rate 28/169 (17%).
102. Michael Hicks, Jonathan T. Moore, David Wetherall, and Scott Nettles. Experiences with capsule- based active networking. In *Proceedings of the DARPA Active Networks Conference and Exposition (DANCE)*, pages 16–24. IEEE, May 2002.
103. Michael Hicks, Jonathan T. Moore, and Scott Nettles. Compiling PLAN to SNAP. In Ian W. Mar- shall, Scott Nettles, and Naoki Wakamiya, editors, *Proceedings of the Third International Working Conference on Active Networks (IWAN)*, volume 2207 of *Lecture Notes in Computer Science*, pages 134–151. Springer-Verlag, October 2001. Acceptance rate 10/22 (45%).
104. Michael Hicks, Jonathan T. Moore, and Scott Nettles. Dynamic software updating. In *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*, pages 13–23. ACM, June 2001. Acceptance rate 30/144 (21%).
105. Jonathan T. Moore, Michael Hicks, and Scott Nettles. Practical programmable packets. In *Proceedings of the Twentieth IEEE Computer and Communication Society INFOCOM Conference*, pages 41–50. IEEE, April 2001. Acceptance rate 192/830 (23%).

106. Michael Hicks and Scott Nettles. Active networking means evolution (or enhanced extensibility required). In Hiroshi Yashuda, editor, *Proceedings of the Second International Working Conference on Active Networks (IWAN)*, volume 1942 of *Lecture Notes in Computer Science*, pages 16–32. Springer-Verlag, October 2000.
107. K. G. Anagnostakis, M. W. Hicks, S. Ioannidis, A. D. Keromytis, and J. M. Smith. Scalable resource control in active networks. In Hiroshi Yashuda, editor, *Proceedings of the Second International Working Conference on Active Networks (IWAN)*, volume 1942 of *Lecture Notes in Computer Science*, pages 343–358. Springer-Verlag, October 2000.
108. Michael Hicks, Stephanie Weirich, and Karl Crary. Safe and flexible dynamic linking of native code. In Robert Harper, editor, *Types in Compilation*, volume 2071 of *Lecture Notes in Computer Science*. Springer-Verlag, September 2000. Selections from the Third ACM SIGPLAN workshop of the same name.
109. Pankaj Kakkar, Michael Hicks, Jonathan T. Moore, and Carl A. Gunter. Specifying the PLAN network programming language. In *Higher Order Operational Techniques in Semantics (HOOTS)*, volume 26 of *Electronic Notes in Theoretical Computer Science*, pages 87–104. Elsevier, September 1999.
110. Michael Hicks, Suresh Jagannathan, Richard Kelsey, Jonathan T. Moore, and Cristian Ungureanu. Transparent communication for distributed objects in java. In *Proceedings of the ACM SIGPLAN Java Grande Conference*, pages 160–170. ACM, June 1999. Acceptance rate 20/38 (53%).
111. Michael Hicks and Angelos D. Keromytis. A secure PLAN. In Stefan Covaci, editor, *Proceedings of the First International Working Conference on Active Networks IWAN)*, volume 1653 of *Lecture Notes in Computer Science*, pages 307–314. Springer-Verlag, June 1999. Reprinted with extensions in DARPA Active Networks Conference and Exposition (DANCE).
112. Michael Hicks, Jonathan T. Moore, D. Scott Alexander, Carl A. Gunter, and Scott Nettles. PLANet: An active internetwork. In *Proceedings of the Eighteenth IEEE Computer and Communication Society INFOCOM Conference*, pages 1124–1133. IEEE, March 1999. Acceptance rate 184/600 (31%).
113. Michael Hicks, Luke Hornof, Jonathan T. Moore, and Scott Nettles. A study of large object spaces. In *Proceedings of the ACM SIGPLAN International Symposium on Memory Management (ISMM)*, pages 138–145. ACM, October 1998.
114. Michael Hicks, Pankaj Kakkar, Jonathan T. Moore, Carl A. Gunter, and Scott Nettles. PLAN: A packet language for active networks. In *Proceedings of the Third ACM SIGPLAN International Conference on Functional Programming Languages (ICFP)*, pages 86–93. ACM, September 1998. Acceptance rate 30/77 (39%).
115. D. Scott Alexander, Michael W. Hicks, Pankaj Kakkar, Angelos D. Keromytis, Marianne Shaw, Jonathan T. Moore, Carl A. Gunter, Trevor Jim, Scott M. Nettles, and Jonathan M. Smith. The SwitchWare Active Network Implementation. In *Notes of the ACM SIGPLAN Workshop on ML*, pages 67–76, September 1998.
116. Michael Hicks, Pankaj Kakkar, Jonathan T. Moore, Carl A. Gunter, and Scott Nettles. Network programming with PLAN. In Luca Cardelli, editor, *Proceedings of the IEEE Workshop on Internet Programming Languages*, volume 1686 of *Lecture Notes in Computer Science*, pages 127–143. Springer-Verlag, May 1998.
117. Michael W. Hicks, Jonathan T. Moore, and Scott M. Nettles. The measured cost of copying garbage collection mechanisms. In *Proceedings of the ACM SIGPLAN International Conference on Functional Programming (ICFP)*, pages 292–305. ACM, June 1997. Acceptance rate 25/78 (32%).

II.D.4. Other

118. Andrew Ruef† and Michael Hicks, Build it, break it, fix it: Competing to build secure systems, *The Next Wave*, 21(1), 2015. Invited article.
119. Gilles Barthe, Michael Hicks, Florian Kerschbaum, and Dominique Unruh. The Synergy Between Programming Languages and Cryptography (Dagstuhl Seminar 14492). *Dagstuhl Reports*, 4(12):29-47, 2015.
120. Michael Hicks. POPL'12 program chair's report (or, how to run a medium-sized conference). *SIGPLAN Notices*, 47(4), April 2012.
121. Christopher M. Hayden, Eric A. Hardisty, Michael Hicks, and Jeffrey S. Foster. Efficient systematic testing for dynamically updatable software. In *Proceedings of the Workshop on Hot Topics in Software Upgrades (HotSWUp)*, October 2009. Invited article.
122. Jonathan T. Moore, Michael Hicks, and Scott M. Nettles. Chunks in PLAN: Language support for programs as packets. In *Proceedings of the 37th Annual Allerton Conference on Communication, Control, and Computing*, September 1999. Invited article.
123. Dan Grossman, Michael Hicks, Greg Morrisett, and Trevor Jim. Cyclone: a type-safe dialect of C. *C/C++ Users Journal*, 23(1), January 2005. Invited article.

II.E. Conferences, Workshops, and Talks
II.E.2. Invited Talks

1. "Increasing the Impact of PL Research." Invited talk at the Programming Languages Mentoring Workshop (PLMW) at ACM SIGPLAN Internatinoal Conference on Functional Programming (ICFP), August 23, 2021. (See https://icfp21.sigplan.org/home/plmw-icfp-2021#program)

2. "From Verified Compilation to Shor's Algorithm," invited talk, Second International Workshop on Programming Languages for Quantum Computing (PLanQC 2021), June 22, 2021. Held virtually. (See https://pldi21.sigplan.org/home/planqc-2021#program)

3. "A Verified Optimizer for Quantum Circuits," invited talk, University of Massachusetts Systems Lunch, October 23, 2020. Held virtually. (See https://www.cics.umass.edu/event/systems-lunch-2)

4. "Evaluating Fuzz Testing," keynote, Meeting on Hot Topics on the Science of Security (HotSOS), September 24, 2020. Held virtually. (See https://cps-vo.org/node/71596)

5. "Contesting Secure Software Development," keynote, Computer Security Foundations Symposium, June 23, 2020. (See https://www.ieee-security.org/TC/CSF2020/program.html)

6. "Wysteria: A Programming Language for Generic, Mixed-Mode Multiparty Computations," Johns Hopkins Applied Physics Lab (JHUAPL), January 17, 2020.

7. "Contesting Secure Development to Understand Security Mistakes." UPenn CIS Department Seminar, November 5, 2019. (See https://events.seas.upenn.edu/event/cis-seminar-contesting-secure-development-to-understand-security-mistakes/)

8. "Coverage Guided, Property Based Testing," Microsoft Research Seminar, July 31, 2019. (See https://www.microsoft.com/en-us/research/video/coverage-guided-property-based-testing/)

9. "What is Programming Languages Research?" Invited talk at the Programming Languages Mentoring Workshop (PLMW) at ACM SIGPLAN Symposium on Principles of Programming Languages (POPL), January 2019. (See https://popl19.sigplan.org/track/PLMW-2019-

papers#program and https://www.youtube.com/watch?v=70_DFytfnWw&t=3s)

10. "Evaluating Design Tradeoffs in Numeric Static Analysis for Java," Invited talk, Workshop on Dependable and Secure Software Systems, ETH Zurich, Oct 19-20, 2018. See https://eth-sri.github.io/workshop2018
    a. Also gave a shorter version of this talk at the European Symposium on Programming (ESOP), April 15, 2018, Thessaloniki, Greece. See https://www.etaps.org/2018/esop

11. "Building Security In: Programming Language-based Techniques for Ensuring Software Security", Invited lecture, 9th International Summer School on Information Security and Protection, Canberra, Australia, July 9, 2018. See https://cs.anu.edu.au/cybersec/issisp2018/schedule.html

12. "Evaluating Fuzz Testing",
    a. **Best scientific cybersecurity paper award** acceptance talk, National Security Agency, October 10, 2019.
    b. Invited lecture, 9th International Summer School on Information Security and Protection, Canberra, Australia, July 10, 2018. See https://cs.anu.edu.au/cybersec/issisp2018/schedule.html
    c. Keynote talk, Cornell Programming Languages Retreat, September 7, 2018. See https://www.cs.cornell.edu/courses/cs7190/2018fa/pl-retreat-2018.html
    d. Colloquium talk, Department of Computer and Communications Sciences, October 22, 2018. See https://memento.epfl.ch/event/ic-colloquium-evaluating-fuzz-testing-an-adventu-2/

13. "Languages for Oblivious Computation," invited talk at the ACM Workshop on Programming Languages and Analyses for Security (PLAS), co-located with the ACM SIGSAC Conference on Computer and Communications Security (CCS), October 30, 2017. http://plas2017.cse.buffalo.edu/

14. "Programming Languages Meet Cryptography," series of three invited lectures at the Cornell, Maryland, Max Planck Pre-doctoral Research School, August 8-13, 2017 in Saarbruecken, Germany. http://cmmrs.mpi-sws.org/

15. "Rubah: DSU for Java on a Stock JVM," invited presentation to the Microsoft HDInsight Team, Microsoft, Redmond, WA, June 29, 2017.

16. "Dynamic Software Updating with Kitsune," invited presentation to the Microsoft SQL server team, Microsoft, Redmond, WA, Jun 27, 2017.

17. "Scientifically evaluating security-conscious development using the Build it, Break it, Fix it programming contest," invited talk at Google Security and Privacy Research Summit, Google main campus, Mountain View, CA, June 6, 2017.

18. "Dynamic Enforcement of Knowledge-based Security Policies," invited talk at Immuta, Inc., a startup company whose product performs in private data management, College Park, MD, March 22, 2017. (See https://twitter.com/ImmutaData/status/844709889851174912)

19. "What is research and how to do it?" Invited talk, Programming Languages Mentoring Workshop (PLMW)

a. at ACM SIGPLAN Symposium on Principles of Programming Languages (POPL), January 2017. (See http://popl17.sigplan.org/track/PLMW-2017)
b. at ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI), June 2017.

20. "Tackling Software Insecurity with Abstract Interpretation," Invited talk, Workshop on the Next 40 Years of Abstract Interpretation (N40AI), collocated with POPL, January 2017. (See http://popl17.sigplan.org/track/N40AI)

21. "Build it, Break it, Fix it: Contesting Secure Development,"
    a. Seminar talk, Maryland Cybersecurity Center Symposium, Dec 5, 2016
    b. Seminar talk, Imperial College, London, UK, June 27, 2016.
    c. Invited talk, Microsoft Research, Redmond, July 11, 2016. See https://www.microsoft.com/en-us/research/video/build-it-break-it-fix-it-contesting-secure-development/

22. "PL and Crypto: Case studies in their combination," Seminar talk, University of California, San Diego, March 15, 2016.

23. "Oblivious Computation," NSF Workshop on Formal Methods for Security, November 2015.

24. "Build It Break It: Measuring and Comparing Development Security," UMD Computer Science Research Seminar Series, October 2015.

25. "Build it, break it, fix it: A security-minded programming contest," RISE Seminar talk, Microsoft Research, Redmond, July 2015.

26. "Authenticated Data Structures, Generically,"
    a. Dagstuhl Seminar on the Synergy between Programming Languages and Cryptography, Wadern, Germany, December 1, 2014.
    b. Colloquium talk, Max Planck Institute for Software Systems (MPI-SWS), Saarbrucken, Germany, December 5, 2014.

27. "Memory-Trace Oblivious Program Execution,"
    a. Dagstuhl Seminar on the Synergy between Programming Languages and Cryptography, Wadern, Germany, December 1, 2014.
    b. NSA Best Scientific Cybersecurity Award Paper presentation, September 18, 2014.

28. "Build-it, break-it, fix-it: Recapping the results," 5th Annual NIST National Initiative on Cyber-security Education (NICE) Workshop on "Shaping the Future of Cybersecurity Education," Columbia, Maryland, November 5, 2014.

29. "Rubah: DSU for Java on a stock JVM,"
    a. Invited talk, Laboratory for Telecommunications Sciences, June 2014.
    b. Invited talk, Microsoft Research, Redmond, August 2014.

30. "Adapton: Composable, Demand-Driven Incremental Computation," invited talk, Microsoft Research, August 2014.

31. "Wysteria: A Programming Language for Generic, Mixed-Mode Multiparty Computations"
    a. OOPSLA PC Workshop talk, May 14, 2014.

b. Invited talk, Computer Science Department Seminar, Stevens Institute of Technology, February 24, 2014. See http://www.stevens.edu/news/content/cs-department-seminar-michael-hicks-university-maryland-wysteria-programming-language
c. USUKITA "bootcamp" talk, London, England, January 5, 2014.

32. "Expositor: Scriptable time-travel debugging with first class traces," Software Reliability Group Seminar, Imperial College, London, UK, January 8, 2014. http://srg.doc.ic.ac.uk/seminars/14-01-hicks/

33. "Authenticated Data Structures, Generically," joint PL/Systems seminar, Microsoft Research, Cambridge, November 22, 2013. See http://research.microsoft.com/apps/video/dl.aspx?id=205246

34. "Secure Computation: Combining PL and Crypto Research," University of Maryland Undergraduate Research Seminar series, October 25, 2013.

35. "Build-it, break-it, fix-it: A new security contest," 4th Annual NIST National Initiative on Cybersecurity Education (NICE) Workshop on "Shaping the Future of Cybersecurity Education," Gaithersburg, Maryland, September 18, 2013.

36. "On-line patching for better security, and other security challenges," Colloquium talk, Johns Hopkins Applied Physics Lab, May 10, 2013. Cf. http://www.youtube.com/watch?v=eF1NBkSKUzw

37. "Expositor: Scriptable time-travel debugging with first class traces,"
    a. Colloquium talk, Max Planck Institute for Software Systems (MPI-SWS), Saarbrucken, Germany, November 30, 2012.
    b. Seminar talk, Microsoft Research, Redmond, WA, December 3, 2012.See http://research.microsoft.com/apps/video/default.aspx?id=179198

38. "Directed Symbolic Execution," seminar talk, Ecole Normale Superieur, Paris, France, October 11, 2012.

39. "Kitsune: Efficient, General-purpose Dynamic Software Updating for C,"
    a. Colloquium talk, Johns Hopkins University, Department of Computer Science, November 12, 2012.
    b. Seminar talk, Microsoft Research, Redmond, July 13, 2012. See http://research.microsoft.com/apps/video/default.aspx?id=169667
    c. Invited talk at Symantec Research Labs, Herndon, VA, March 21, 2012.
    d. Invited Tech. talk at Cyberpoint, Inc., Baltimore, MD, March 6, 2012. "Polymonads: reasoning and inference," seminar talk, Madrid Institute for Advanced Studies (IMDEA), Madrid, Spain, June 1, 2012.

40. "Cybersecurity: Past, Present, Looking ahead," Keynote, Maryland Francis King Carey School of Law Symposium on *Cybersecurity: Safeguarding Information in a Digital Age*, March 30, 2012.

41. "Dynamic Enforcement of Knowledge-based Security Policies,"
    a. Georgia Institute of Technology ISC invited talk, April 12, 2013
    b. Dagstuhl Seminar on Quantitative Security Analysis, Wadern, Germany, November 27, 2012

c. Seminar talk, Microsoft Research, Cambridge, UK, October 12, 2012. See http://research.microsoft.com/apps/video/default.aspx?id=174740
d. Dagstuhl Seminar on Web Application Security, Wadern, Germany, October 4, 2012
e. Seminar talk, Ecole Normale Superieur, Paris, France, May 30, 2012
f. Technical talk, First Maryland Cybersecurity Center (MC2) Symposium, May 17, 2012
g. Colloquium talk, George Washington University, March 20, 2012
h. USUKITA "bootcamp" talk, Weybridge, England, June 22, 2011
i. Guest speaker at University of Maryland CyberClub, Oct 27, 2011

42. "Dynamic inference of static types for Ruby,"
    a. Dagstuhl Seminar on Foundations of Scripting Languages, Wadern, Germany, January 2012
    b. Colloquium talk, Department of Computer Science, Cornell University, October 14, 2011
    c. Systems Seminar, University of Delaware, December 3, 2010
    d. Computing Laboratory seminar, University of Oxford (United Kingdom), September 17, 2010

43. "Software Synthesis for cost-effective development of correct, efficient software," given at the ISAT outbrief to DARPA program managers and the deputy director on October 4, 2011. I co-chaired this ISAT study with Armando Solar-Lezama of MIT.

44. "Lightweight monadic programming in ML,"
    a. Seminar, Institut National de Recherche en Informatique et en Automatique (INRIA) Paris- Rocquencourt, June 24, 2011.
    b. Automated Reasoning Group seminar, University of Cambridge Computer Laboratory, July 1, 2011.

45. "Evaluating dynamic software update safety using efficient systematic testing," Computer Science Department seminar, University of Texas at Austin, April 15, 2010.

46. "Secure Provenance Policies in SELinks," Workshop on Provenance in Secure and Advanced Computer Systems, University of Edinburgh, Scotland, May 13, 2009.

47. "Systematic Testing for Dynamically Updatable Software", Systems Research Group Seminar, University of Cambridge Computer Laboratory, May 7, 2009.

48. "A Theory of Typed Coercions and its Applications",
    a. Departmental Seminar, Chalmers Institute of Technology (Gothenburg, Sweden), April 28, 2009
    b. Automated Reasoning Group seminar, University of Cambridge Computer Laboratory, March 17, 2009

49. "Type-directed coercion insertion for security enforcement", Dagstuhl Seminar on Web Application Security, Wadern, Germany, April 2, 2009.

50. "Practical Dynamic Software Updating for C",
    a. Departmental Seminar (Colloquium talk), University of Kent at Canterbury Computer Laboratory, March 10, 2009.
    b. Departmental Wednesday Seminar (Colloquium talk), University of Cambridge Computer Laboratory, October 15, 2008.

51. “Cross-tier, Label-based Security Enforcement for Web Applications”, invited talk, Security and Privacy Day @ Stony Brook, SUNY Stony Brook, May 30, 2008.

52. “Building Secure Web Applications with SELinks”, invited talk, IARPA STONESOUP workshop, May 9, 2008.

53. “Automated Detection of Persistent Kernel Control-Flow Attacks”, invited talk, Microsoft Research, Silicon Valley, October 2007.

54. “SELINKS: A language for provably secure web applications”, Distinguished Lecturer talk, IBM T.J. Watson Research Laboratory, July 2007.

55. “CMOD: Modular Type Safety and Information Hiding for C”
    a. Seminar talk, Department of Computer Science, The University of Wisconsin, Madison, March 2007.
    b. Seminar talk, Department of Computer Science, Stanford University, February 2007.
    c. Microsoft Research, Redmond, Washington, July 2006.
    d. Triforce Programming Language Research Group Seminar, Harvard University, November 2006.

56. “LOCKSMITH: Context-sensitive Correlation Analysis for Detecting Races”
    a. Colloquium talk, Department of Computer Science, University of Massachusetts (Amherst), November 2006.
    b. Seminar talk, Department of Computer Science, Cornell University, April 2007.

57. “Making Concurrent Software Safer”, Summer School on Language-based Techniques for Concurrent and Distributed Software, July 2006.

58. “Static Analysis to Improve Software Quality” (with Jeff Foster), National Research Council of Canada, Institute for Information Technology invited talk, June 2006.

59. “Cyclone: A Safe Dialect of C”, Trust and Security Seminar, Department of Computer Science, University of Illinois Urbana-Champaign, November 2005.

60. “Practical Dynamic Software Updating for C”
    a. Seminar talk, Department of Computer Science and Engineering, University of California, Berkeley, February 2007.
    b. Colloquium talk, Department of Computer Science, the University of Virginia, December 2006.
    c. MIT Programming Language Group Seminar, November 2006.
    d. Colloquium talk, Department of Computer Science, the Pennsylvania State University, November 2005.
    e. IFIP Working Group 2.4 meeting (invited participant), Jackson’s Mill, West Virginia, October 2005.
    f. Microsoft Research, Redmond, Washington, August 2005. (See http://research.microsoft.com/apps/video/default.aspx?id=104614.)
    g. IBM T.J. Watson Research Lab, Hawthorne, New York, June 2005.

61. “Analyzing code that changes on the fly”, Dagstuhl Seminar on Multi-version Program Analysis, Wadern, Germany, June 2005.

62. "Safe and Predictable Dynamic Software Updating"
    a. Principles of Programming Seminar, Carnegie-Mellon University, Pittsburgh, July 2004.
    b. NTT/Docomo Research Laboratory, San Jose, California, August 2004.

63. "Transparent Proxies for Java Futures", Seminar talk, the Johns-Hopkins University, May 2004.

64. "Safe and Flexible Memory Management in Cyclone"
    a. Microsoft Research, Redmond, Washington, February 2004.
    b. Short talk at the Second Workshop on Semantics, Program Analysis, and Computing Environments for Memory Management (SPACE), 2004.
    c. Dagstuhl Seminar on Language-based Security, Wadern, Germany, October 2003.
    d. Colloquium talk, Department of Computer Science, Purdue University, August 2003.

65. "Dynamic Software Updating"
    a. Colloquium talk, The Computer Laboratory at Cambridge University, Cambridge, England, October 2001.
    b. Microsoft Research Lab, Cambridge, England, October 2001.
    c. ULTRA Seminar, Heriot-Watt University, Edinburgh, Scotland, October 2001.
    d. Principles of Programming Seminar, Carnegie-Mellon University, Pittsburgh, December 2000.

II.E.3. Refereed Presentations

1. Robert Maxwell and Michael Hicks. Using students to pen test your network (for credit). Presentation at *EDUCAUSE Security Professionals Conference*, Saint Louis, MO, April 2013.
2. Amol Deshpande and Michael Hicks. Toward on-line schema evolution for non-stop systems. Presented at the 11th High Performance Transaction Systems Workshop, September 2005.

II.E.5. Refereed Posters

1. Yit Phang Khoo, Michael Hicks, Jeffrey S. Foster, and Vibha Sazawal. Directing javascript with arrows (poster summary). In *Poster Proceedings of the ACM International Conference on Functional Programming (ICFP)*, September 2008.
2. Iulian Neamtiu† and Michael Hicks. Dynamic Software Updating for the Linux Kernel. Work-in-progress poster/presentation at the *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, November 2006.
3. Iulian Neamtiu†, Michael Hicks, Gareth Stoyle⋆, and Manuel Oriol⋆. Ginseng: A System for Dynamic Software Updating. Poster presented at the *ACM Conference on Programming Language Design and Implementation (PLDI)*, June 2006.
4. Pavlos Papageorgiou† and Michael Hicks. Merging network measurement with data transport. Poster presented at the *IEEE Passive/Active Measurement Workshop (PAM)*, March 2005.
5. Carl Gunter, Michael Hicks, Pankaj Kakkar, Jonathan Moore, Scott Nettles, and Jonathan Smith. PLAN: Language-Based Safety and Security for Active Networks. Work-in-progress poster/presentation at the *ACM Symposium on Operating Systems Principles (SOSP)*, October 1997.

II.E.9. Non-Refereed Posters

1. Michael Hicks. Programming Languages for Reliable, Available, and Secure Software. Invited poster at the 25th National Academy of Sciences Kavli Frontiers of Science Symposium, November 7–9, 2013.

II.F. Professional Publications

II.F.1. Reports and Non-Refereed Monographs

1. David Darais, Chang Liu, Ian Sweet, and Michael Hicks. A Language for Probabilistically Oblivious Computation. *CoRR*, abs/1711.09305, November 2017.
2. Aseem Rastogi, Nikhil Swamy, and Michael Hicks. Wys*: A Verified Language Extension for Secure Multi-Party Computations. *CoRR*, abs/1711.06467, November 2017.
3. Piotr Mardziel, Mario Alvim, Michael Hicks, and Michael Clarkson. Quantifying information flow for dynamic secrets (extended version). Technical Report CS-TR-5035, Department of Computer Science, the University of Maryland, College Park, May 2014.
4. Aseem Rastogi, Matthew A. Hammer, and Michael Hicks. Wysteria: A programming language for generic, mixed-mode multiparty computations (extended version). Technical Report CS-TR-5034, Department of Computer Science, the University of Maryland, College Park, May 2014.
5. Matthew Hammer, Yit Phang Khoo, Michael Hicks, and Jeffrey S. Foster. Adapton: Composable, demand-driven incremental computation. Technical Report CS-TR-5027, Department of Computer Science, the University of Maryland, College Park, July 2013.
6. Yit Phang Khoo, Jeffrey S. Foster, and Michael Hicks. Expositor: Scriptable Time-Travel Debugging with First-Class Traces. Technical Report CS-TR-5021, Department of Computer Science, University of Maryland, College Park, February 2013.
7. Nataliya Guts, Michael Hicks, Nikhil Swamy, Daan Leijen, and Gavin Bierman. Polymonads. Technical Report, University of Maryland Department of Computer Science, July 2012.
8. Christopher M. Hayden, Stephen Magill, Michael Hicks, Nate Foster, and Jeffrey S. Foster. Specifying and verifying the correctness of dynamic software updates. Technical Report CS-TR-4997, University of Maryland Department of Computer Science, November 2011. Extended version of VSTTE'12 paper with proofs of theorems and additional discussion.
9. Christopher M. Hayden, Edward K. Smith, Eric A. Hardisty, Michael Hicks, and Jeffrey S. Foster. Evaluating dynamic software update safety using efficient systematic testing. Technical Report CS- TR-4993, University of Maryland, Department of Computer Science, September 2011.
10. Jonathan Turpie, Elnatan Reisner, Jeffrey S. Foster, and Michael Hicks. Multiotter: Multiprocess symbolic execution. Technical Report CS-TR-4982, University of Maryland Department of Computer Science, August 2011.
11. Nikhil Swamy, Nataliya Guts, Daan Leijen, and Michael Hicks. Lightweight monadic programming in ML. Technical Report MSR-TR-2011-039, Microsoft Research, May 2011.
12. Kin-Keung Ma, Yit Phang Khoo, Jeffrey S. Foster, and Michael Hicks. Directed symbolic execution. Technical Report CS-TR-4979, University of Maryland Department of Computer Science, April 2011.
13. Piotr Mardziel, Stephen Magill, Michael Hicks, and Mudhakar Srivatsa. Dynamic enforcement of knowledge-based security policies. Technical Report CS-TR-4978, University of Maryland Depart- ment of Computer Science, April 2011. Extended version contains proofs of theorems.
14. Jong hoon (David) An, Avik Chaudhuri, Jeffrey S. Foster, and Michael Hicks. Dynamic inference of static types for Ruby. Technical Report CS-TR-4965, University of Maryland Department of Computer Science, July 2010. Extended version of POPL 2011 paper contains proofs of theorems.
15. Michael Hicks and Jeffrey S. Foster. Adapting Scrum to managing a research group. Technical Report CS-TR-4966, University of Maryland, Department of Computer Science, September 2010.

16. Christopher M. Hayden, Eric A. Hardisty, Michael Hicks, and Jeffrey S. Foster. A testing based empirical study of dynamic software update safety restrictions. Technical Report CS-TR-4949, University of Maryland, Department of Computer Science, November 2009.
17. Yit Phang Khoo, Michael Hicks, Jeffrey S. Foster, and Vibha Sazawal. Directing javascript with arrows (functional pearl). Technical Report CS-TR-4923, University of Maryland, Department of Computer Science, August 2008. Extended version of ICFP 2008 poster summary.
18. Yit Phang Khoo, Jeffrey S. Foster, Michael Hicks, and Vibha Sazawal. Path projection for user-centered static analysis tools (long version). Technical Report CS-TR-4919, University of Maryland, Department of Computer Science, August 2008.
19. Nikhil Swamy† and Michael Hicks. Verified enforcement of automaton-based information release policies. Technical Report CS-TR-4906, University of Maryland, Department of Computer Science, 2008. Full version of PLAS 08 paper.
20. Nick L. Petroni, Jr.†and Michael Hicks. Automated detection of persistent kernel control-flow attacks. Technical Report CS-TR-4880, Department of Computer Science, University of Maryland, October 2007.
21. Iulian Neamtiu†, Michael Hicks, Jeffrey S. Foster, and Polyvios Pratikakis†. Contextual effects for version-consistent dynamic software updating and safe concurrent programming (extended version). Technical Report CS-TR-4875, University of Maryland, Department of Computer Science, July 2007.
22. Nikhil Swamy†and Michael Hicks. Fable: A language for enforcing user-defined security policies (extended version). Technical Report CS-TR-4876, University of Maryland, Department of Computer Science, July 2007.
23. Saurabh Srivastava†, Michael Hicks, and Jeffrey S. Foster. Appendix to CMod: Modular information hiding and type-safe linking for C. Technical Report (TBD), University of Maryland, College Park, 2007.
24. Michael Hicks, Nikhil Swamy†, and Simon Tsang. Toward specifying and validating cross-domain policies. Technical Report CS-TR-4870, Department of Computer Science, University of Maryland, April 2007.
25. Nikhil Swamy†, Michael Hicks, Stephen Tse, and Steve Zdancewic. Managing policy updates in security-typed languages (extended version). Technical Report CS-TR-4793, Department of Computer Science, University of Maryland, August 2006. Contains full formal development and proofs for CSFW 06 paper.
26. Saurabh Srivastava†, Michael Hicks, Jeffrey S. Foster, and Bhargav Kanagal⋆. Defining and enforcing C's module system. Technical Report CS-4816, Department of Computer Science, University of Maryland, July 2006. Contains full formal development and proofs for TLDI 07 paper.
27. Polyvios Pratikakis†, Jeffrey S. Foster, and Michael Hicks. Context-sensitive correlation analysis for detecting races (extended version). Technical Report CS-TR-4789, Department of Computer Science, University of Maryland, June 2006. Extends PLDI 2006 paper with full formal development.
28. Boniface Hicks, Dave King, Patrick McDaniel, and Michael Hicks. Trusted declassification: high- level policy for a security-typed language (extended version). Technical Report NAS-TR-033-2006, Department of Computer Science and Engineering, the Pennsylvania State University, June 2006. Extended version of the PLAS 2006 paper with full formal development.
29. Michael W. Hicks, Pankaj Kakkar, Jonathan T. Moore, Carl A. Gunter, and Scott M. Nettles. PLAN: A packet language for active networks (extended version). Unpublished manuscript, May 2006. Unifies and consolidates ICFP 98, IPL 98, and Allterton 99 papers.
30. Iulian Neamtiu†, Michael Hicks, Gareth Stoyle⋆, and Manuel Oriol†. Practical dynamic software updating for C (extended version). Technical Report CS-TR-4790, Department of Computer Science, University of Maryland, March 2006. Extended version of PLDI 06 paper.

31. Polyvios Pratikakis†, Michael Hicks, and Jeffrey S. Foster. Existential label flow inference via CFL reachability. Technical Report CS-TR-4700, Department of Computer Science, University of Mary- land, July 2005. Contains full formal development and proofs of SAS 06 paper.
32. Michael Hicks, Greg Morrisett, Dan Grossman, and Trevor Jim. Safe and flexible memory management in Cyclone. Technical Report CS-TR-4514, University of Maryland Department of Computer Science, July 2003.
33. Michael Hicks, Adithya Nagarajan†, and Robbert van Renesse. User-specified adaptive scheduling in a streaming media network. Technical Report CS-TR-4430, Department of Computer Science, University of Maryland, March 2003. Extends OPENARCH '03 paper with full algorithmic description and analysis.
34. Dan Grossman, Greg Morrisett, Trevor Jim, Michael Hicks, Yanling Wang, and James Cheney. For- mal type soundness for Cyclone's region system. Technical Report CS 2001-1856, Cornell University, November 2001. Proves formal results of PLDI '02 paper.
35. Michael Hicks and Stephanie Weirich. A calculus for dynamic loading. Technical Report MS-CIS- 00-07, University of Pennsylvania, April 2000. Formalism and proof for system presented in *Types in Compilation* paper.
36. Michael Hicks. Types and Intermediate Representations. Technical Report MS-CIS-98-05, Department of Computer and Information Science, University of Pennsylvania, April 1998.
37. Michael Hicks. PLAN system security. Technical Report MS-CIS-98-25, Department of Computer and Information Science, University of Pennsylvania, April 1998.
38. Jonathan T. Moore, Michael Hicks, and Scott Nettles. General-purpose persistence using flash memory. Unpublished manuscript, April 1997.

II.F.5. Other

1. Michael Hicks. *Dynamic Software Updating*. PhD thesis, Department of Computer and Information Science, University of Pennsylvania, August 2001. **Winner of the 2002 ACM SIGPLAN Doctoral Dissertation award.**

II.H. Completed Creative Works

II.H.8. Software and Applications

These software packages were developed as part of research I supervised or participated in. Most are freely distributed, or are available upon request.

1. **VOQC and SQIR**, developed 2018 – present; framework for writing and optimizing quantum circuit programs, proved correct in the Coq proof assistant. https://github.com/inQWIRE/SQIR
2. **MVEDSUa**, developed 2017 – 2019; framework for fault-tolerant dynamic software updating of C programs, combining my prior **Kitsune** framework with another one called Varan.
3. **Checked C**, developed 2016 – present; extension to Clang/LLVM implementation of C for enforcing spatial safety. Available at https://github.com/Microsoft/checkedc
4. **Build it, Break it, Fix it** *(BIBIFI)* developed 2012 – present; a web platform for hosting novel programming-meets-hacking contests. Hosted at https://builditbreakit.org with code available upon request.
5. **LWeb**, developed 2012 – present; a information flow-secure platform for developing web applications in Haskell. Basis for BIBIFI. Code available upon request.
6. **Java-Fuzz**, developed August 2016 – March 2018; a random, mutation-based testing tool for Java programs. Available upon request.
7. **Jana**, developed April 2015 – present; an abstract interpreter for Java bytecode programs, aiming to prove numeric properties. Foundation of the **Blazer** analysis for finding timing side channels (or proving their absence). Available at https://github.com/plum-umd/JANA

8. **Prob**, developed June 2011 – present; a static analysis for probabilistic programs, for the purposes of enforcing knowledge-based security policies. Available at https://github.com/GaloisInc/TAMBA (code/prob subdirectory)
9. **Wys***: A Verified Language Extension for Secure Multiparty Computations, developed March 2015 – 2016; a new programming language for developing secure applications, distributed as part of F*, a programming language developed in collaboration with Microsoft Research. Based on Wysteria. Available at https://www.fstar-lang.org/
10. **KVolve**: An extension to the redis key-value store to support lazily-implemented dynamic updates the store's contents, developed since April 2015 – August 2016. Available at https://github.com/plum-umd/kvolve
11. **C-Strider**, A toolkit for building heap traversal-based tools for C programs, developed April 2014 – August 2016. Available at https://github.com/plum-umd/c-strider
12. **ObliVM**: A privacy-preserving computation framework that allows one or multiple organizations to perform secure data analysis without disclosing their private data, developed March 2015 – August 2016. Available at http://www.oblivm.com/
13. **Rubah**, developed October 2012 – October 2014; a library and program rewriting system to support dynamic software updating services in Java programs (follow-on to Kitsune). Available at https://github.com/plum-umd/rubah
14. **Kitsune**, developed October 2010 - 2016; a library and compiler for support dynamic software updating services in C programs. Available at http://kitsune-dsu.com
15. **Expositor**, developed 2013 - 2014; a framework for scriptable, time-travel debugging. Available at https://bitbucket.org/khooyp/expositor
16. **Adapton**, developed 2013 - present; a framework for demand-driven, incremental computation. Available at https://bitbucket.org/khooyp/adapton.ocaml.
17. **Wysteria**, developed 2013 - 2016; a programming language for secure multiparty computation. Available at https://bitbucket.org/aseemr/wysteria/wiki/Home
18. **LambdaAuth**, developed 2014 - 2015; a programming language for authenticated data structures. Available at http://amiller.github.io/lambda-auth/
19. **LockSmith**, a tool for statically discovering race conditions in C programs, at http://www.cs.umd.edu/projects/PL/locksmith/, developed 2005 - 2009 (released June 2006). LockSmith checks that the program consistently follows the *guarded-by pattern*, in which some mutual exclusion lock is held consistently by any thread that accesses a location that may simultaneously be accessed by other threads.
20. **Rubydust**, developed October 2009 - 2013. Rubydust is a Ruby library that can be used to infer types for Ruby methods based on information gathered during testing.
21. **Measurement Manager Protocol (MGRP)**, at http://www.cs.umd.edu/projects/MGRP/, developed 2007 - 2009 (released May 2010). MGRP is a Linux-kernel extension that permits network measurement tools to be written so that they transparently piggyback on top of existing traffic, to infer high-quality measurements at lower overhead.
22. **Path Projection**, at http://www.cs.umd.edu/projects/PL/PP/, developed 2008 - 2012 (released June 2008). Path Projection is novel user interface toolkit that helps users visualize, navigate, and understand program paths, a common component of many static analysis tools' error reports.
23. **Arrowlets**, at http://www.cs.umd.edu/projects/PL/arrowlets/, developed 2008 - present (released September 2008). Arrowlets is a JavaScript library that uses the concept of arrows to ease the task of composing event-handlers, a common need in Javascript programming.
24. **SELinks**, a secure web programming language, at http://www.cs.umd.edu/projects/PL/selinks, developed 2007 - 2010 (released May 2008). {\sc SElinks extends the Links web programming language type system with support for specifying security policies so that type checking is equivalent to security checking: type-correct programs correct enforce their security policies.
25. **CMod**, a module system for C programs, at http://www.cs.umd.edu/projects/PL/CMod, developed 2006 - 2010 (released October 2006). CMod wraps the standard C compiler and linker

to enforce programming patterns that effectively define a module system for legacy C programs; type safe linking and information hiding are provable consequences.

26. **BEEP**, browser modifications to support Browser-enforced Embedded Policies, at http://www.research.att.com/~trevor/beep.html, developed 2006 - 2007 (released February 2007). BEEP is a system by which a web server can embed a security policy within its pages, implemented as a Javascript function. While rendering a page, the browser will only execute script programs that are approved by this function, defeating script injection attacks.
27. **Ginseng**, a system for dynamically updating C programs, at http://www.cs.umd.edu/projects/dsu/, developed 2004 - 2011 (released June 2006). Ginseng compiles a C program to accept a dynamic patch that can replace functions, global variables, and type definitions on the fly; Ginseng generates dynamic patches mostly automatically from the deployed and updated source code for the program.
28. **Cyclone**, a safe dialect of C for more reliable and secure systems programming, at http://cyclone.thelanguage.org/, developed 2001 - 2007. A hallmark of Cyclone is type safety alongside a high degree of control over data layout and memory management.
29. **MediaNet**, an overlay network for user-specified adaptations to streaming media, at http://www.cs.umd.edu/projects/medianet/, developed 2002 - 2004. MediaNet is written in Cyclone and employs many of its advanced memory management features.
30. **Java Transparent Proxy Framework**, a toolkit for proxy programming, at http://www.cs.umd.edu/~polyvios/proxyc/, developed 2004. The framework uses static analysis to track the flow of proxies beginning from user-annotated program points. The analysis results are used to re-write the program, inserting code to wrap and unwrap proxies as necessary.
31. **The Flashed updgradeable webserver** and an implementation for dynamically updateable applications, developed 2000 - 2001. Superceded by Ginseng; this is the dynamic updating implementation I developed as part of my doctoral dissertation, on top of the Popcorn compiler for Typed Assembly Language.
32. **The SNAP Active (Programmable) Network**, at http://www.cis.upenn.edu/~dsl/SNAP/, developed 2001 - 2002. SNAP defines a bytecode language for constructing programs that reside in network packets, e.g., to define their routing or quality-of-service. SNAP is distinguished in being efficient with tight controls over worst-case resource usage.
33. **The PLANet Active (Programmable) Network**, at http://www.cis.upenn.edu/~dsl/PLAN/, developed 1998 - 2001. PLANet's packet language PLAN is an ML-like language that can be compiled to SNAP.
34. **The Oscar Garbage Collection Testbed**, at http://www.cis.upenn.edu/~oscar/, developed 1997 - 1998. Defines a heap snapshot and replay system for measuring the costs of garbage collection mechanisms.

II.J. Sponsored Research

II.J.1. Grants

1. AFOSR FA95502110051 *Software Assurance for Quantum Programs*, $218,875, January 2021-2024. PI: Michael Hicks. (This is part of a $506,651 multi-institutional grant with UChicago, and co-PI Robert Rand; UMD is the lead.)
2. NSF CCF-1955610: *Collaborative Research: SHF: Medium: Bringing Python Up to Speed*, $374,390, July 2020-2023. PI: Michael Hicks. (This is part of a $1.2M multi-institution grant with UPenn and UMass Amherst; UMD is the lead institution.)
3. DARPA FA87501910008: *Symbolic Input Unification and Minimization*, $34,802, Nov 2018 – Dec 2018. PI: Michael Hicks.
4. NSF CNS-1801545 *SaTC: CORE: Medium: Collaborative: Understanding Security in the Software Development Lifecycle: A Holistic, Mixed-Methods Approach*, $698,957, Sep 2018 –

2022. Lead PI: Michelle Mazurek, co-PI: Michael Hicks. (This is part of a multi-institution grant with University of South Florida.)

5. DOE ASCR grant: *Efficient and Reliable Mapping of Quantum Computations onto Realistic Architectures*, $4,467,131, Sep 2018 – 2022. Lead PI: Andrew Childs. Co-PIs: Michael Hicks, Xiaodi Wu, Alexey Gorshkov.
6. NSF CNS-1563722 *TWC: Medium: Collaborative: New Protocols and Systems for RAM-Based Secure Computation*, $464,196. May 2016—2019. Lead PI: Jonathan Katz, co-PI: Michael Hicks. (This is part of a multi-institution grant with George Mason University.)
7. DARPA FA87501520104: *SOUCIS: Sound Over- & Under-Approximations of Complexity & Information Security*, $1,209,032. April 2015—2018. Lead PI: Michael Hicks, co-PI: David Van Horn and Jeffrey S. Foster. (This is part of a $2.52M multi-institution grant with UC Berkeley and Yale, each with one additional co-PI; UMD is the lead institution.)
8. DARPA FA8750-16-C-0022: *TAMBA: Testing and Modeling of Brandeis Artifacts*, $521,524. October 2015—October 2019. PI: Michael Hicks. (This is part of a multi-institution grant led by Galois, Inc.; Hicks is the only UMD PI.)
9. NSF CNS-1314857 *TWC: Medium: Collaborative: DIORE: Digital Insertion and Observation Resistant Execution*, $799,499. August 2013-2018. PI: Elaine Shi, co-PIs: Michael Hicks and Bobby Bhattacharjee. (This is part of a $1.2M multi-institution grant with one other PI at UT Austin. Bhattacharjee became UMD PI when Shi left UMD.)
10. NSF CNS-1111599 *TC: Large: Collaborative Research: Practical Secure Two-Party Computation: Techniques, Tools, and Applications*, $1M. August 2011-2018. PI: Jonathan Katz, co-PI: Michael Hicks. (This is part of a multi-institution grant, with UVA as lead institution.)
11. *Establishing a Science of Security Research Lablet at the University of Maryland*, NSA, $1,487,608. February 2014–February 2017. Lead PI: Jonathan Katz, co-PI: Michael Hicks (among 11 others).
12. *Secure Information Flows in Hybrid Coalition Networks*, US Army Research Lab and UK Ministry of Defence International Technology Alliance in Network and Information Science (USUKITA) program BPP15, $244,233 PI: Michael Hicks, co-PI Jonathan Katz. May 2015–2016.
13. NSF EDU-1319147 *EDU: Competing to Build Secure Systems*, $300,000. September 2013-2016. PI: Michael Hicks, co-PIs: Atif Memon, David M. Levin, and Jandelyn D. Plane.
14. *Secure Information Flows in Hybrid Coalition Networks*, US Army Research Lab and UK Ministry of Defence International Technology Alliance in Network and Information Science (USUKITA) program BPP13, $356,615 PI: Michael Hicks, co-PI Jonathan Katz. May 2013–2015.
15. University of Maryland Partnership with the Laboratory of Telecommunications Sciences, Contract Number H9823013D00560002, *Protecting against Malware on Android*, $371K. April 5, 2012–September 30, 2014. PI: Jeffrey S. Foster, co-PI: Michael Hicks.
16. NSF CCF-0910530 *SHF: Large: Collaborative Research: Ever Ready: Perpetually Available Software Systems*, $642K. August 2009–2014. PI: Michael Hicks. (This is part of a multi-institutional team grant with two other PIs for a total award of $2.3M.)
17. NSF CCF-0915978 *SHF: Small: User-Centered Software Analysis Tools*, $500K. September 2009– 2013. PI: Jeffrey S. Foster, co-PI: Michael Hicks.
18. *Securing Information Flows*, US Army Research Lab and UK Ministry of Defence International Technology Alliance in Network and Information Science (USUKITA) program BPP11, $333K. co-PIs: Michael Hicks and Jonathan Katz. May 2011–2013.
19. University of Maryland Partnership with the Laboratory for Telecommunications Sciences (LTS), *Dynamic Software Update Automation for Servers and Event-Driven Programs*, $407K. January 2010– 2013. PI: Michael Hicks, co-PI: Jeff Foster.

20. NSF CNS-0905419 RECOVERY: *TC: Medium: Collaborative Research: Techniques to Retrofit Legacy Code with Security*, $300K. September 2009–2012. PI: Michael Hicks. (This is part of a multi-institutional team grant with three other PIs for a total award of $1.2M.)
21. NSF CCF-0541036 *Scalable, Precise, and Effective Analyses for Detecting Race Conditions*, $360K. September 2006–2010. PI: Michael Hicks, co-PI: Jeffrey S. Foster.
22. US Army Research Lab and UK Ministry of Defence International Technology Alliance in Network and Information Science (USUKITA) program BPP09, $197K. co-PIs: Michael Hicks and Jonathan Katz. May 2009–2011.
23. NSF CNS-0346989 *CAREER: Programming Languages for Reliable and Secure Low-level Systems*, $550K. June 2004-2010. PI: Michael Hicks.
24. University of Maryland Partnership with the Laboratory for Telecommunications Sciences (LTS), *Safe and Robust Dynamic Software Updating for Real-time Embedded Applications*, $327K, August 2007–January 2010. PI: Michael Hicks, co-PI: Jeff Foster.
25. Army Research Lab Communications and Networking Collaborative Technology Alliance (ARL C&N CTA) program TIP-07-4, *End-to-End Security of Cross-Domain Information Flows*, $245K (funds to Maryland). October 2006–October 2009. PI: Michael Hicks.
26. ARL C&N CTA program (supported by DARPA DSO), *Provably Enforcing Decentralized, Multi-level Security Policies in Distributed, Web-based Applications*, $83K. March 1, 2007–September 30, 2009. PI: Michael Hicks.
27. NSF CCF-0524036 *Collaborative Research: CT-T: Flexible, Decentralized Information-flow Control for Dynamic Environments*, $280K. September 2005-2009. PI: Michael Hicks. (This is part of a multi-institutional team grant with three other PIs for a total award of $1.1M.)
28. DARPA #HR00110610019 *Computer Science Study Group Panelist*, $77K. March 2006-2007. PI: Michael Hicks.
29. NSF IIS-0613601 *SoD-HCER: Evaluation of Complex Designs–A Comparative Study*, $100K. July 2006–2008. PI: Michael Hicks (transferred from Vibha Sazawal in January 2007).

II.K. Fellowships, Gifts and Other Funded Research

II.K.2. Gifts

1. Advancing Checked C, Microsoft Research Award (gift), $119,477. March 2019-December 2020. PI: Michael Hicks.
2. Contesting Secure Development Effectiveness, Google Research Award (gift), $55,256. March 2017–2018. PI: Michelle Mazurek, co-PI: Michael Hicks.
3. Boombox: Dynamic Software Updates for Software Defined Networks, Google Research Award (gift), $51,037. September 2014–2015. PI: Michael Hicks.
4. Connecting the Theory and Practice of Incremental Computation via Servo, gift from Mozilla Corporation, $88,294. May 2014–2015. co-PIs: Michael Hicks and Jeff Foster.
5. Improving the efficiency of static analysis of C source code using multiple tools, Cisco Systems Inc (Unrestricted gift), $35,000. 2007. PI: Jeffrey S. Foster, co-PI: Michael Hicks.

II.M. Centers for Research, Scholarship, and Creative Activities

II.M.2. Centers Directed

1. Maryland Cybersecurity Center (MC2), (first) Director from November 2011 to 2013.

II.M.3. Symposia Organized (*through Maryland Cybersecurity Center*)

1. Maryland Cybersecurity Center Symposium, May 2012.
2. Maryland Cybersecurity Center Symposium, May 2013.

II.N. Patents

II.N.1. Device

1. Chang Liu, Michael Hicks, and Elaine Shi, US 62/017,758, *Memory Trace Oblivious Program Execution and Method*, US PTO, Submitted, 2014 July.

II.O. Other Research/Scholarship/Creative Activities

1. **Build-it, Break-it, Fix-it, a security-oriented programming contest**. My role: Co-inventor, director. First offered August 2014, then again in Spring and Fall of 2015, and Fall 2016. The nationwide contest pits student "builder" teams against "breaker" teams, where the former tries to build software resilient against attack from the latter. Intended outcomes are to (a) impress upon students the value of "building security in," and (b) to gather data to research common errors, and to learn what most often correlates with success. https://builditbreakit.org/. The latter two offerings were also part of a Coursera Cybersecurity Capstone project.

III. Teaching, Mentoring and Advising.

III.A. Courses Taught

1. CMSC 631, Program Analysis and Understanding, 30 students, Fall 2021.
2. CMSC 330, Organization of Programming Languages, 145 students, Spring 2021.
3. CMSC 330, Organization of Programming Languages, 145 students, Spring 2019.
4. CMSC 330, Organization of Programming Languages, 145 students, Spring 2018.
5. CMSC 631, Program Analysis and Understanding, 28 students, Fall 2017.
6. CMSC 330, Organization of Programming Languages, 215 students, Spring 2017.
7. CMSC 396H, Honors Seminar, 13 students, Fall 2016.
8. CMSC 838G, Mechanized Proof and Verified Software, 8 students, Spring 2016.
9. CMSC 330, Organization of Programming Languages, 129 students, Fall 2015.
10. CMSC 330, Organization of Programming Languages, 130 students, Spring 2015.
11. CMSC 433, Programming Language Technologies and Paradigms, 50 students, Fall 2014
12. CMSC 838G, Software Security, 10 students, Spring 2014
13. CMSC 433, Programming Language Technologies and Paradigms, 50 students, Fall 2013
14. CMSC 631, Program Analysis and Understanding, 26 students, Spring 2013
15. CMSC 330, Organization of Programming Languages, 113 students, Spring 2013
16. CMSC 498L, Cybersecurity lab (co-taught with 4 other profs, 25 students, Fall 2012
17. CMSC 498B, Secure Maryland (penetration testing), 7 students, Spring 2012
18. CMSC 631, Program Analysis and Understanding, 13 students, Fall 2011
19. CMSC 838G, Software Security, 5 students, Spring 2011
20. CMSC 433, Programming Language Technologies and Paradigms, 40 students, Fall 2010
21. CMSC 330, Organization of Programming Languages, 66 students, Spring 2010
22. CMSC 631, Program Analysis and Understanding, 9 students, Fall 2009
23. CMSC 631, Program Analysis and Understanding, 16 students, Fall 2007
24. CMSC 412, Operating Systems, 42 students, Spring 2007
25. CMSC 631, Program Analysis and Understanding, 20 students, Fall 2006
26. CMSC 433, Programming Language Technologies and Paradigms, 44 students, Spring 2006
27. CMSC 412, Operating Systems, 50 students, Fall 2005
28. CMSC 838Z, Language-based Security, 7 students, Spring 2005
29. CMSC 412, Operating Systems, 45 students, Fall 2004
30. CMSC 838Z, Tools and Techniques for Software Dependability, 8 students, Spring 2004
31. CMSC 433, Programming Language Technologies and Paradigms, 48 students, Fall 2003

32. CMSC 838Y, Agile and Adaptive Programming, 10 students, Spring 2003
33. CMSC 433, Programming Language Technologies and Paradigms, 47 students, Fall 2002

III.B. Teaching Innovations

III.B.3. Software, Applications, Online Education, etc.

1. **Software Security** - I developed a massively-open on-line course (MOOC) under the UMD-Coursera agreement, on this topic, as part of a specialization on cybersecurity. I fully developed the content for this course, including videos, lecture notes, and projects. See https://www.coursera.org/course/softwaresec. As of January 2019, this course has had 35,000+ participants.
2. **Cybersecurity Capstone –** I developed the Capstone project for the UMD specialization on cybersecurity, based on our Build-it, Break-it, Fix-it contest. We hosted two capstones in 2015 and one in 2016.

III.B.4. Instructional Workshops and Seminars Established

1. **Using LLVM for Program Analysis and Transformation** (co-presented with Andrew Ruef): This tutorial was given at the Conference on Programming Language Design and Implementation (PLDI) in June 2013, having 47 attendees (the most of any tutorial that year), and again in June 2014. It was organized to give programming language researchers and implementers with information about how to use the Low Level Virtual Machine (LLVM) project to approach program transformation and analysis. Cf. http://pldi2013.ucombinator.org/tutorials.html

2. **Using Symbolic Execution to Find Software Vulnerabilities**: This tutorial was given at the Second Maryland Cybersecurity Center Symposium in May 2013, having 25 attendees. It presented an overview of symbolic execution technology and a tutorial on how to use the KLEE tool in particular. Cf. http://www.cs.umd.edu/~mwh/se-tutorial/

III.B.5. Course or Curriculum Development

1. **CMSC 330 - Organization of Programming Languages** In 2017, I redesigned much of the material and slides, to further streamline connections (and support in-class exercises), and supervised the development of several new projects. In 2018, I swapped out the use of Prolog (introduced by me in 2013) and replaced it with Rust, a new, security-oriented programming language. In 2015, I developed new content on writing secure software, including new lectures and a new project. In 2013, I reintroduced logic programming to the course, and redesigned projects and lecture material to make better connections between course elements, including this new unit.

2. **CMSC 388N - Build it, Break it, Fix it: Competing to Secure Software** (2019) This short winter course employs the Build it, Break it, Fix it contest which I co-developed to teach students about how to build secure software.

3. **CMSC 498B - Secure Maryland** (2012) This is a new course on penetration testing. Seven students piloted the course. We train them in scanning, vulnerability assessment, and exploitation, and educate them in the ethics of penetration testing. The most novel feature of the course is that they are permitted to attack the University's own networks in a supervised setting via public-facing interfaces. When vulnerabilities are discovered, they contact the system owners to work to fix the problems. In The 2012 iteration of the course revealed literally hundreds of vulnerabilities, the majority of which were fixed before the semester concluded.

4. **CMSC 412 - Operating Systems** (2004,2005,2007) I substantially revised the content of the course—I created new lecture notes, new projects, and shifted focus more toward core operating systems and systems-building concepts (and away from networking and distributed systems, taught in other courses). I added an introductory project that uses Cyclone, a research programming language, and extended the remaining projects so that Cyclone could be used optionally; several students used Cyclone for their projects and two started independent study projects involving Cyclone with me afterward.

5. **CMSC 433 - Programming Language Technologies and Paradigms** (2002,2003,2006,2010,2013,2014) I substantially reworked the course content twice. In 2006, I focused the material on advanced programming techniques for sequential, concurrent, and distributed software. I initially revised most of the lecture notes, and selectively created new lecture notes and new projects with each offering of the course. In 2010, I focused the material almost entirely on concurrent/parallel programming, with some material on distributed programming. I completely redesigned all projects and lecture notes.

6. **CMSC 631 - Program Analysis and Understanding** (2006,2007,2009,2011,2013,2017) This course focuses on programming language semantics, and foundations and implementation techniques of program analyses. Some of the material of the course was inherited from my earlier course, CMSC 838Z (see below). For the first offering I added some new lecture material, and for the second I reorganized the material to be more bottom-up. In both cases I designed new projects and homework assignments. For the 2013 offering, I developed new material involving the use of a *proof assistant* to ensure that students' formalizations and proofs are checked for correctness automatically. In the 2017 offering, I moved the course to be based in its majority on the use of a proof assistant.

7. **CMSC 838Y - Agile and Adaptive Programming** (2003) This course focuses on novel techniques for building reusable, reconfigurable, and adaptable software components and systems. I developed the lecture material and helped students design research projects.

8. **CMSC 838Z - Tools and Techniques for Software Dependability** (2004), Language-based Security (2005) The theme of these courses is programming language-oriented techniques for building reliable and secure software. I developed the course content, lectures, and several substantial projects. Most of the project materials and some of the lecture materials from the 2004 offering were subsequently included in *CMSC 631, Program Analysis and Understanding*.

III.C. Advising: Research or Clinical

III.C.1. Undergraduate

1. Guido Ambasz, September 2021 – present.
2. Aaron Green, September 2020 – June 2021.
3. Paaras Bhandari, June 2020 – December 2020.
4. Jerry Peng, May 2018 – December 2020.
5. Deena Postol, January 2020 – August 2020 (co-mentored with David Van Horn); joined UMD PhD program September 2020.
6. Technica+Research team advisor, Nov 8-10 2019, for a team of six undergraduate women: Nhi Ngo, June Tian, Aishwarya Jayan, Amel Hassan (Tufts), Qundeel Rafiq, and Gagandeep Samra (the rest from UMD).

7. George Klees, September 2018 – May 2021. (Supervised George when he was a high school student, also, June – September 2017. Work we did together eventually won the **2018 NSA Best Scientific Cybersecurity Paper competition**.)
8. Yiyun Liu, May 2018 – August 2020; **2020 CRA outstanding undergraduate researcher, honorable mention** (see https://cra.org/about/awards/outstanding-undergraduate-researcher-award/#2020); joined UMD MS program September 2020.
9. Aaron Eline, October 2019 – December 2019.
10. Ivan Quiles, June 2019 – December 2019.
11. Shriraj Gandhi, January 2019 – June 2019.
12. Hasan Touma, August 2018 – August 2019.
13. Pei-Jo Yang, May 2018 – June 2019.
14. Matthew Hou, January 2018 – June 2018 (CMSC 499 in Spring'18)
15. Gavriel Epstein, January 2018 – June 2018.
16. Adam Fiergang, June 2017 – August 2017.
17. Benjamin Cooper, June 2017 – June 2018 (including CMSC 499 in Fall'17).
18. Ryan Forsyth, January-May 2016 (CMSC 499, Spring'16).
19. Jerry Tan, January 2016-June 2017.
20. Steven Sosnick, September 2015-December 2015 (co-mentored with Atif Memon)
21. Kyle Headley, June 2014–August 2015 (co-mentored with Jeff Foster)
22. Ilse Haim, January 2014–December 2014
23. Andrew Ruef, June 2010–2013 (he is now my Ph.D. student)
24. James Parker, Spring 2012–2013
25. Edward (Ted) Smith, Summer 2009–2013 (co-mentored with Jeff Foster)
26. Ryan Sims, November 2010–December 2011
27. Michail Denchev, June 2010–December 2011
28. QUEST-program mentor, Fall 2010, for a team of five undergraduates: Mitchell Kochman, Sebastian Gomez, Melinda Jih, Brad Klein, Jeremy Prince
29. Andrew Ferguson, Winter 2010–Summer 2010
30. QUEST-program mentor, Fall 2009, for a team of six undergraduates: Jeremy Erdman, Andrew Duch, Cameron Rose, Nicole Thomas, Haren Arcot, and Akshay Goyal
31. Elliott Sprehn, December 2008–May 2008
32. Jeffrey S. Meister, Summer 2007–Summer 2008 (co-mentored with Jeff Foster).
33. Neha Gupta, Spring 2005
34. Hariharan Sivaramakrishnan (ECE), Spring 2004–Spring 2005
35. Christopher Wamble, Summer 2003.This was as part of the Ronald E. McNair Post-Baccalaureate Achievement Program, which aims to increase the participation of under-represented groups in graduate school
36. Martha Gebremichael, Spring 2003
37. Michael Nelson, Spring 2003

III.C.2. Master's

1. Paaras Bhandari, January 2021 – June 2021. Co-advised by Leo Lampropoulos.
2. Yiyun Liu, August 2020 – present. (Previously supervised as UMD undergraduate student.)
3. James Parker, *LMonad: Information Flow Control for Haskell Web Applications*. Defended and graduated December 2014. My role: Advisor.
   - Part-time research programmer 2015-2018; returned as PhD student in 2018.
4. Nate Parsons, *Implementing and Typing a Core Calculus for Mixed-mode Secure Multi-party Computations*. Graduated December 2013. My role: advisor.
5. Jong hoon (David) An, *Dynamic Inference of Static Types for Ruby*, defended July 2010, graduated August 2010. My role: co-advisor (with Jeff Foster).

6. Elnatan Reisner. Graduated August 2011. My role: co-advisor (with Jeff Foster)
7. Jonathan Turpie. Graduated August 2011. My role: co-advisor (with Jeff Foster). Currently a software engineer at Amazon.
8. Surupa Biswas, *Memory Overflow Protection for Embedded Systems using Run-time Checks, Reuse and Compression*, Department of Electrical and Computer Engineering. Defended Au- gust 2004, graduated August 2004. My role: committee member
9. Adithya Nagarajan (SE). Graduated Spring 2003. My role: Co-advisor.
10. James Rose. Graduated Spring 2004. My role: Advisor.

III.C.3. Doctoral

1. Le Chang. Fall 2021-present.
2. Shaobo Cui. Fall 2021-present.
3. Finn Voichick. Fall 2020-present.
4. Kesha Hietala. *A Verified Software Toolchain for Quantum Programming.* My role: advisor, Fall 2016–present. Advanced to candidacy September 2020.
5. Ian Sweet. My role: advisor, Fall 2016–present. (Also supervised during Spring/Summer 2016 as post-undergraduate research assistant).
6. Sankha Guria. *Synthesis of Web Apps through Lightweight Abstractions*. My role: proposal committee member. Advanced to candidacy December 2021.
7. Sangyung Hong. *Practical Hardware Attacks on Deep Learning*. My role: proposal and thesis committee member. Advanced to candidacy May 2020. Defended July 2021.
8. Eddie Schoute. *Architecture-Respecting Quantum Circuit Transformations*. My role: proposal and thesis committee member. Advanced to candidacy April 2020. Defended October 2021.
9. Shih-Han Hung. *Classical proofs for quantum computation*. My role: proposal and thesis committee member. Advanced to candidacy April 2020. Defended June 2021.
10. Milod Kazerounian, *Towards More Expressive and Usable Types for Dynamic Languages.* My role: proposal and thesis committee member. Advanced to candidacy April 2020. Defended May 2021.
11. Daniel Votipka, *A Human-Centric Approach to Software Vulnerability Discovery*. My role: proposal and dissertation committee member. Advanced to candidacy, April 2019. Defended November 2020. Currently an Assistant Professor at Tufts University.
12. Aravind Machiry (University of California, Santa Barbara), *Protecting Smart Devices from the Bottom-up*. My role: co-advisor (Aravind was an intern in my lab June-August 2019, and I served on his dissertation committee), June 2019 – August 2020; defended August 2020. Currently a post-doc at UPenn (August-December 2020); starts as an Assistant Professor at Purdue in January 2021.
13. James Parker, *Advanced Language-based Techniques for Correct, Secure Networked Systems*. My role: advisor, January 2019—June 2020. (Previously MS student, see above.) Advanced to candidacy, June 2019. Defended May 2020. Currently a Research Engineer at Galois, Inc.
14. Andrew Glaudell (University of Maryland Dept of Physics). My role: dissertation committee member. Defended December 2019.
15. Phuc (Phil) Nguyen, *Higher-order Symbolic Execution*. My role: proposal and dissertation committee member. Advanced to candidacy January 2019. Defended October 2019. Currently a Software Engineer at Google.
16. Danny Kim, *Improving Existing Static and Dynamic Malware Detection Techniques with Instruction-level Behavior*. My role: Dean's representative (ECE). Defended February 2019.
17. Richard Johnson, *Implicit CPU-GPU Parallelization and Hybrid Memory Hierarchies for Next Generation HPC Systems*. My role: Proposal committee member. Advanced to candidacy December 2018.

18. Andrew Ruef. *Tools and Experiments for Software Security*. My role: advisor, Spring 2014–Fall 2018. Advanced to candidacy October 2017, defended October 2018. Currently a researcher at IDA/CCS.
19. Willem Wyndham. My role: advisor, Fall 2016-2018.
20. Xiao Wang. My role: dissertation committee member. Defended June 2018. Currently an assistant professor at Northwestern.
21. David Darais, *Mechanizing Abstract Interpretation*. My role: Proposal and dissertation committee member. Advanced to candidacy October 2016, defended May 2017. Currently an assistant professor at U. Vermont.
22. Ahmed Kosba, *Verifiable Computation in Practice: Tools and Protocols*. My role: Proposal committee member. Advanced to candidacy April 2017.
23. Kartik Nayak, *Efficient Oblivious Computation*. My role: Proposal and dissertation committee member. Advanced to candidacy April 2017, defended June 2018. Currently an assistant professor at Duke University.
24. Andrew Miller, *Provable Security and Safely Composable Abstractions for Cryptocurrencies*. My role: Proposal and dissertation committee member. Advanced to candidacy December 2015, defended July 2016. Currently an assistant professor at UIUC.
25. Chang Liu. *Trace Oblivious Program Execution.* My role: co-advisor (with Elaine Shi), Fall 2012–present. Advanced to candidacy Fall 2015, defended July 2016. Post-doc at UC Berkeley 2016 -- 2018. Currently a quant researcher at Citadel Securities, since September 2018.
26. Alex Malozemoff, *Efficient Secure Computation For Real-World Settings and Security Models*. Defended May 2016. My role: dissertation committee member. Currently a researcher at Galois, Inc.
27. Kristopher Micinski, *Interaction-Based Security Policies for Mobile Apps*. Advanced to candidacy March 2016. My role: proposal committee member. Currently a visiting lecturer at Haverford College.
28. Luis Pina, *Practical Dynamic Software Updating (for Java)*. My role: co-advisor (Luis was a Ph.D. student at *Instituto Superior Tecnico*, Lisbon, Portugal, visiting my group), Fall 2012–Spring 2015; defended January 2016 (left early to take a research position). Currently an assistant professor at the University of Illinois, Chicago (UIC).
29. Aseem Rastogi, *Language-Based Techniques for Practical and Trustworthy Secure Multi-Party Computations*. My role: advisor, Fall 2012–present. Advanced to candidacy Fall 2014, defended January 2016. Currently a researcher at Microsoft Research, India.
30. Karla Saur, *Dynamic Upgrades for High Availability Systems*. Advanced to candidacy May 2014, defended September 2015. My role: co-advisor (with Jeff Foster), Spring 2012–Fall 2015. Currently a researcher at Intel Labs, Portland.
31. Jeff Stuckman, *Developing and Validating Security Vulnerability Indicator Metrics*. Advanced to candidacy May 2013. My role: Proposal committee member.
32. Piotr Mardziel, *Modeling, Quantifying, and Limiting Adversary Knowledge*. Advanced to candidacy March 2012, defended December 2014, graduated January 2015. My role: advisor, Summer 2009–January 2015. Currently a post-doc at Carnegie Mellon.
33. Peter Fontana, *Towards a Unified Theory of Timed Automata*. Advanced to candidacy April 2012, defended February 17, 2014. My role: Proposal and dissertation committee member. Currently at NIST.
34. Arnar Birgisson (Chalmers University of Technology, Sweden), *Tracking Dependencies for Security and Privacy*. Defended November 2013. My role: opponent. (The opponent is a special role in Swedish defenses: he begins the defense by presenting context on the research area, and then engages the candidate with roughly one hour of probing questions following the candidate's talk.) Currently at Google.

35. Khoo Yit Phang, *User-Centered Program Analysis Tools.* Advanced to candidacy May 2010, defended June 2013, graduated August 2013. My role: co-advisor (with Jeff Foster), Spring 2007–Summer 2013. Currently at Mathworks.
36. Aaron Schulman, *Add Wireless Broadcast to the Internet*. Advanced to candidacy May 2012, Defended July 2013. My role: Proposal and dissertation committee member. Won the ACM SIGCOMM Dissertation Award. Currently an assistant professor at UCSD.
37. Justin McCann, *Automating Performance Diagnosis in Networked Systems*. Advanced to candidacy December 2010, defended April 2012, graduated August 2012. My role: Advisor, Spring 2010–May 2012.
38. Chris Hayden, *Clear, Correct, and Efficient Dynamic Software Updates*. Advanced to candidacy May 2010, defended April 2012, graduated August 2012. My role: co-advisor (with Jeff Foster), Spring 2007–May 2012. Currently in industry.
39. Kin Keung (Martin) Ma, *Improving Program Testing and Understanding via Symbolic Execution*. Defended December 2011, graduated December 2011. My role: co-advisor (with Jeff Foster), Spring 2009–December 2011. Currently at Google.
40. Adam Bender, *An Accountability Architecture for the Internet*. Defended November 2010, graduated December 2010. My role: dissertation committee member. Currently at Google.
41. Matthew Simpson (UMD ECE), *Optimizing Memory Safety Checks Using Dynamic Pointer Disambiguation*. Proposed August 2008, defended November 3, 2010. My role: dissertation committee member.
42. Nathaniel Ayewah. *Static Analysis in Practice*. Defended July 2010, graduated August 2010. My role: dissertation committee member. Currently at Microsoft.
43. Saurabh Srivastava, *Satisfiability-based Program Reasoning and Synthesis*. Advanced to candidacy Winter 2008, defended April 2010, graduated Spring 2010. My role: co-advisor (with Jeff Foster and Sumit Gulwani), Spring 2005–2010. Co-founder of 20^n. Previously a post-doc at Berkeley.
44. Suriya Subramanian (University of Texas at Austin CS), *Dynamic Software Updates: a VM-centric Approach*. Defended April 2010, graduated May 2010. My role: Co-advisor (with Kathryn McKinley), August 2007 – May 2010. Previously at Microsoft Research, India. Now at a startup company.
45. Michael Furr. *Combining Static and Dynamic Typing in Ruby*. Defended October 2009, graduated December 2009. My role: dissertation committee member. Currently at Amazon.
46. Pavlos Papageorgiou (UMD ECE), *The Measurement Manager: Modular and Efficient End-to-end Measurement Services*. Advanced to candidacy May 2007, defended November 2008, graduated December 2008. My role: advisor, Summer 2004–Fall 2008. Now at Google, Inc.
47. Iulian Neamtiu, *Practical Dynamic Software Updating*. Received Masters degree Spring 2005. Advanced to candidacy Fall 2006, defended July 2008, graduated Summer 2008. My role: advisor, Fall 2003–August 2008. Previously an Associate Professor at UC Riverside. Now at the New Jersey Institute of Technology.
48. Polyvios Pratikakis, *Sound, Precise and Efficient Static Race Detection for Multi-Threaded Programs*. Advanced to candidacy Fall 2007, defended July 2008, graduated Summer 2008. My role: Co-advisor (with Jeff Foster), Summer 2003–Summer 2008. Now at FORTH (a Greek research institute) and the University of Crete as an assistant professor.
49. Nikhil Swamy, *Language-Based Enforcement of User-Defined Security Policies as Applied to Multi-tier Web Programs.* Advanced to candidacy Spring 2007, defended July 2008, graduated Summer 2008. My role: advisor, Spring 2004–Summer 2008. Now a Principal Researcher at Microsoft Research.
50. Nick L. Petroni, Jr, *Property-based integrity monitoring of operating system kernels*. Advanced to candidacy Spring 2006, defended December 2007, graduated Spring 2008. My role: co-advisor (with Bill Arbaugh), Spring 2006–December 2007. Thesis one of two nominated by

the University of Maryland for the ACM Doctoral Dissertation Award. Previously at IDA/CCS (government research lab). Currently at a startup.

51. Boniface Hicks (Pennsylvania State University, Dept CSE), *Bridging the Semantic Gap in Security Policy*. Proposed December 2005, defended August 3, 2007. My role: external dissertation committee member. Now at St. Vincent's Archabbey.
52. Yu David Liu (Johns Hopkins University, Dept. CS), *Assemblages: Modules for Large-scale Distributed Software*. Proposed May 2004, defended September 2007, graduated December 2007. My role: Proposal and dissertation committee external member. Currently an Associate Professor at SUNY Binghamton.
53. Chadd Williams, *Using Historical Data from Source Code Revision Histories to Detect Source Code Properties*. Defended July 2006, graduated August 2006. My role: dissertation committee member. Currently an Associate Professor at Pacific University.
54. Jeremy Manson. *The Java Memory Model*. Defended September 2004, graduated Winter 2004. My role: dissertation committee member. Currently at Google, Inc.
55. Rajiv Gandhi, *Broadcast Scheduling*. Defended May 2003, graduated August 2003. My role: dissertation committee member. Currently an Associate Professor at Rutgers, Camden.
56. Jonathan Turpie. My role: advisor, Winter 2010–August 2011. Currently at Amazon.
57. Brian Corcoran. My role: advisor, Summer 2007–Summer 2009.
58. Eric Hardisty. My role: co-advisor (with Jeff Foster), Winter 2008–Summer 2009.
59. Gary Jackson. My role: co-advisor (with Jeff Foster), Fall 2005–Fall 2007.
60. Mujtaba Ali. My role: advisor, Spring 2004–Spring 2005.

III.C.4. Post-doctoral

1. Liyi Li, November 2020 – present. Co-advised by Xiaodi Wu.
2. Michael Coblenz, September 2020 – present. Co-advised by Adam Porter.
3. Leonidas Lampropoulos, September 2018 – July 2020. Co-advised by Benjamin Pierce (UPenn).
4. Robert Rand, September 2018 – July 2020. Co-advised by Xiaodi Wu.
5. Tamara Rezk, April 3, 2018. Served on Jury of her *Habilitation à diriger des Recherches* from *Université Côte d'Azur*. This is a post-doctoral thesis awarded 10 years after the PhD, summarizing work to that point.
6. Paul Gazzillo, June 2017---April 2018. Co-advised with Eric Koskinen (Stevens Tech). To start as Assistant Professor at UCF, July 2017.
7. Shiyi Wei, September 2015—August 2017. Currently an assistant professor at UT Dallas.
8. Piotr Mardziel, January 2015–June 2016. Currently a post-doc at CMU.
9. Matthew Hammer, September 2012–August 2015. Currently an assistant professor at CU Boulder.
10. Nataliya Guts, January 2011–July 2012.
11. Stephen Magill, January 2010–August 2011. Currently a researcher at Galois, in Portland, OR.
12. Manuel Oriol, Spring 2004–Spring 2005. Currently a senior lecturer at the University of York, United Kingdom.

III.C.5. Other Research Directions (*K-12 Interactions*)

1. Eric Wang (Poolesville High School), Summer 2021.
2. Jason An (Montgomery Blair High School), Summer 2021.
3. Akshaj Gaur (Poolesville High School), Summer 2020.
4. Yael Pinsky (Montgomery Blair High School), Summer 2019.
5. George Klees (Montgomery Blair High School), Summer 2017 (predominantly) – April 2018 (part-time). Now at UMD as an undergraduate in CS. Research project was foundation of CCS 2018 paper which later won the NSA's Best Scientific Cybersecurity Paper Award.

6. Arun Dillipan (Poolesville High School), Summer 2014.
7. Edward (Ted) Smith (Walt Whitman High School), Summer 2008. Went to UMD for undergraduate school, and then to graduate school at UMass, Amherst. Now at Google, Inc.
8. William C. Burton (Montgomery Blair High School), Summer 2007. Went to Princeton for undergraduate school, and then to graduate school at MIT.
9. Richard Matthew McCutchen (Montgomery Blair High School), Fall 2005. Attended UMD for his undergraduate degree; currently a graduate student at MIT.

III.D. Mentorship

III.D.1. Junior Faculty

1. Leonidas Lampropoulos, July 2020—present.
2. Huaishu Peng (mentor), January 2019—present
3. John Dickerson (mentor), September 2016—present.
4. David Van Horn (mentor), January 2013—2018 (tenured in 2019).

III.G. Other Teaching Activities

1. CMSC 396H: Guest lecture and discussion lead, 2015 Nov, This class exposes undergraduates to research. I conducted a guest lecture on a cybersecurity-related research paper and organized the class discussion.
2. CMSC 798F: How to conduct great research, 2015 Nov, 2018 April, 2021 April, I gave a guest lecture on how to write a good research paper, which included a hands-on exercise

IV. Service and Outreach

IV.A. Editorships, Editorial Boards, and Reviewing Activities

IV.A.1. Editorships

1. Founder and Editor in Chief, *PL Perspectives*, the blog of the ACM Special Interest Group on Programming Languages (SIGPLAN), June 2019 – June 2021. (Added one co-editor in December 2020.)
2. Journal guest co-editor, Logical Methods in Computer Science (LMCS), Special issue on the best papers of the 2010 European Symposium on Programming
3. Associate Editor, ACM Transactions on Programming Languages and Systems (February 2012–November 2016)

IV.A.3. Reviewing Activities for Journals and Presses

1. Communications of the ACM (CACM)
2. ACM Transactions on Programming Languages and Systems (TOPLAS)
3. ACM Transactions on Code Optimization (TACO)
4. IEEE Transactions on Software Engineering (TSE)
5. IEEE Security and Privacy Magazine (S&P)
6. ACM Transactions on Modeling and Computer Simulation (TOMACS)
7. Elsevier Information Processing Letters (IPL)
8. Elsevier Theoretical Computer Science (TCS)
9. Elsevier Journal of Systems Software (JSS)
10. Elsevier Journal of Object Technology (JOT)
11. Springer Acta Informatica

12. Software: Practice and Experience (SPE)

IV.A.4. Reviewing Activities for Agencies and Foundations

1. NSF SATC panelist, April 2021.
2. Computing Innovation Fellows 2020 Reviewer
3. NSF CCF panelist, Spring 2019
4. External reviewer, NSF SATC, Fall 2018
5. NSF FMitF panelist, Spring 2018
6. Expert Reviewer, Software Security Knowledge Area, Cyber Security Body of Knowledge project (http://www.cybok.org), 2017-2018
7. NSF CCF panelist, Spring 2016
8. NSF SaTC panelist, Fall 2013
9. NSF SaTC panelist, Spring 2012
10. NSF CCF panelist, Spring 2010
11. NSF CCF panelist, Spring 2008
12. NSF CNS panelist, Spring 2006
13. NSF CCF panelist, Spring 2005
14. NSF CCF panelist, Spring 2004
15. NSF SBIR panelist, Fall 2003
16. NSF SBIR panelist, Fall 2002
17. External reviewer, NSF CCR INT panel, Fall 2002

IV.A.5. Reviewing Activities for Conferences

1. Program Committee (PC) member, IEEE Symposium on Security and Privacy, May 2022.
2. Review Committee (RC) member, ACM SIGPLAN Conference on Object-Oriented Programming, Languages, and Applications (OOPSLA), October 2021.
3. PC member, ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI), June 2021.
4. External Review Committee (ERC), ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), March 2021.
5. ERC member, ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI), June 2020.
6. **Co-organizer**, ACM SIGPLAN Workshop on Programming Languages and Quantum Computing (PLanQC), January 2020.
7. **Associate Chair** (1 of 6), ACM Conference on Computer and Communications Security (CCS), October 2019.
8. PC member**,** IEEE Conference on Cybersecurity Development (SecDev), September 2019.
9. PC member, Summit on Advances in Programming Languages (SNAPL), May 2019
10. ERC member, ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), March 2019.
11. Member, POPL Student Research Competition (SRC) judging committee, January 2019
12. **Associate Chair** (1 of 4), IEEE Symposium on Security and Privacy, May 2018.
13. PC member**,** IEEE Conference on Cybersecurity Development (SecDev), September 2018.
14. PC member, ETAPS Symposium on Principles of Security and Trust (POST), April 2018.
15. PC member, ACM Conference on Computer and Communications Security (CCS), October 2017.
16. PC member**,** IEEE Conference on Cybersecurity Development (SecDev), September 2017.
17. PC member, USENIX Workshop on Advances in Security Education (ASE), August 2017.
18. PC member, International Symposium on Engineering Secure Software and Systems (ESSoS), July 2017.

19. PC member, IEEE Symposium on Security and Privacy, May 2017.
20. **Program Chair**, IEEE Conference on Cybersecurity Development (SecDev), November 2016.
21. Best Paper Awards Committee member, Cybersecurity Awareness Week (CSAW), November 2016.
22. PC member, USENIX Workshop on Advances in Security Education (ASE), August 2016.
23. **Program Co-Chair**, Computer Security Foundations Symposium (CSF), June 2016.
24. PC member, ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI), June 2016.
25. **Program Co-Chair**, Computer Security Foundations Symposium (CSF), June 2015.
26. PC member, Summit on Advances in Programming Languages (SNAPL), May 2015.
27. PC member, IEEE Symposium on Security and Privacy, May 2015.
28. Selection Committee, 2014 IBM "PL Day", October 2014.
29. PC Member, ACM SIGPLAN Conference on Object-Oriented Programming, Languages, and Applications (OOPSLA), October 2014.
30. PC member, Computer Security Foundations Symposium (CSF), June 2014.
31. PC member, ACM SIGPLAN “Off the Beaten Track” Workshop, January 2014.
32. PC member, ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI), June 2013.
33. ERC member, ACM SIGPLAN-SIGACT Symposium on the Principles of Programming Languages (POPL), January 2013.
34. PC Member, International Workshop on Hot Topics in Software Upgrades, June 2012.
35. **Program Chair**, ACM SIGPLAN-SIGACT Symposium on the Principles of Programming Languages (POPL), January 2012. PC Chair precluded from submitting to the conference.
36. PC Member, ACM SIGPLAN Conference on Object-Oriented Programming, Languages, and Applications (OOPSLA), October 2011.
37. **Co-organizer (PC Chair and General Chair)**, International Workshop on Hot Topics in Software Upgrades, April 2011.
38. PC Member, ACM SIGPLAN Workshop on Types in Language Design and Implementation (TLDI), January 2011.
39. PC Member, ACM SIGPLAN International Conference on Functional Programming (ICFP), August 2010.
40. ERC Member, ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI), June 2010.
41. PC Member, ACM SIGPLAN-SIGSOFT Workshop on Program Analysis for Software Tools and Engineering (PASTE), June 2010.
42. PC Member, European Symposium on Programming (ESOP), March 2010.
43. Member, PLDI Student Research Competition (SRC) judging committee, June 2009
44. PC Member, IEEE Symposium on Security and Privacy (Oakland), May 2009.
45. PC Member, ACM SIGPLAN-SIGACT Symposium on the Principles of Programming Languages (POPL), January 2009. PC members were held to a higher standard for paper acceptance.
46. PC Member, ACM SIGSAC Conference on Computer and Communications Security (CCS), 2008.
47. PC Member, First Workshop on Compiler and Architectural Techniques for Application Reliability and Security (CATARS), 2008 (in conjunction with the International Conference on Dependable Systems and Networks (DSN 2008))
48. RC member, International Symposium on Memory Management (ISMM), 2008.
49. PC Member, International Conference on Coordination Models and Languages (COORDINATION), 2008.
50. PC Member, ACM SIGPLAN Conference on Object-Oriented Programming Languages, Systems, and Applications (OOPSLA), 2007. PC members were held to a higher standard for paper acceptance.

51. PC Member, International Conference on Coordination Models and Languages (COORDINATION), 2007.
52. PC Member, ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI), 2007. PC members were precluded from submitting to the conference.
53. **General Chair and Technical Program Chair**, Workshop on Programming Languages and Analysis for Security (PLAS), 2007.
54. PC Member, Workshop on Programming Languages and Analysis for Security (PLAS), 2006.
55. PC Member, Workshop on Formal Techniques for Java-like Languages (FTfJP), 2006.
56. PC Member, ACM SIGPLAN Workshop on Semantics, Program Analysis, and Computing Environments for Memory Management (SPACE), 2006.
57. PC Member, ACM Symposium on Applied Computing (SAC), Object-oriented Programming Systems (OOPS) track, 2005.
58. PC Member, ACM Workshop on Synchronization and Concurrency in Object-Oriented Languages (SCOOL) 2005.
59. PC Member, ACM/USENIX Virtual Execution Environments Conference (VEE) 2005.
60. PC Member, Workshop on Formal aspects of Unanticipated Software Evolution (FUSE) 2004.
61. PC Member, International Conference on Parallel Processing (ICPP) 2004.
62. PC Member, International Working Conference on Active Networking (IWAN) 2004.
63. PC Member, International Working Conference on Active Networking (IWAN) 2003.
64. PC Member, Workshop on Unanticipated Software Evolution (USE) 2003.
65. PC Member, International Working Conference on Active Networking (IWAN) 2002.
66. PC Member, Workshop on Unanticipated Software Evolution (USE) 2002.
67. PC Member, International Working Conference on Active Networking (IWAN) 2001.

IV.A.6. Historical Editorships, etc. (10+ years ago)

1. External reviewer, International Conference on Concurrency Theory (CONCUR)
2. External reviewer, IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)
3. External reviewer, ETAPS International European Conference on Object Oriented Programming (ECOOP).
4. External reviewer, ACM International Conference on Functional Programming (ICFP)
5. External reviewer, International Conference on Parallel Processing (ICPP)
6. External reviewer, IEEE International Parallel and Distributed Processing Symposium (IPDPS)
7. External reviewer, International Working Conference on Active Newtorks (IWAN)
8. External reviewer, ACMConferenceonObject-OrientedProgrammingSystems,LanguagesandApplications(OOP- SLA)
9. External reviewer, USENIX Symposium on Operating System Design and Implementation (OSDI)
10. External reviewer, USENIX Annual Technical Conference
11. External reviewer, USENIX Security Symposium
12. External reviewer, ACM Conference on Programming Language Design and Implementation (PLDI)
13. External reviewer, ACM Conference on the Principles of Programming Languages (POPL)
14. External reviewer, ACM Conference of the Special Interest Group in Communications (SIGCOMM)
15. External reviewer, ACM Symposium on Operating Systems Principles (SOSP)

IV.B. Committees, Professional & Campus Service

IV.B.1. Campus Service – Department

1. Chair, Teaching Evaluation Committee, Fall 2021.
2. Member (elected), Department Council, 2021 - 2022.
3. Member, Hiring Committee for Associate Program Director of Undergraduate Studies, April 2021—August 2021.
4. Chair, Hiring Committee for PTK lecturer, April 2021—August 2021.
5. **Associate Co-Chair of Undergraduate Education** (focus: Curriculum), August 2017 – September 2021 (pause during 2020 while on sabbatical).
6. Co-Chair, Teaching Evaluation Committee, Fall 2019.
7. Member, Tenure and Promotion committee for Prof. Michelle Mazurek, Associate Professor, 2019.
8. Chair, Tenure and Promotion committee for Prof. David Van Horn, Associate Professor, 2018.
9. Chair, Programming Languages, Software Engineering, and Human-Computer Interaction Field Committee, 2018 – 2019.
10. Member (elected), Department Council, 2018 - 2019.
11. Member, Merit Pay Committee, 2017 – 2018.
12. Member (elected), Department Council, 2017 - 2018.
13. Chair, Programming Languages, Software Engineering, and Human-Computer Interaction Field Committee, 2017 – 2018.
14. Chair, Honors Program, Fall 2016.
15. Member (elected), Department Council, 2016 - 2017.
16. Member, Undergraduate Scholarships and awards committee, 2016 - 2017.
17. Member, Undergraduate Education Committee, CS Faculty retreat, May 2016.
18. Chair, Undergraduate Scholarships and awards committee, 2015 - 2016.
19. Member, Department Council, 2015 - 2016.
20. Chair, Tenure and Promotion committee for Dr. Jandelyn Plane, Principal Lecturer, 2016.
21. Co-Director (with Rich Gerber), CS Education for Tomorrow, 2015 - 2016.
22. Chair, Strategic planning subcommittee on undergraduate education, 2014 - 2015.
23. Co-Director (with Adam Porter), CS Education for Tomorrow, 2014 - 2015.
24. Member, Department Council, 2014 - 2015.
25. Chair, Undergraduate Scholarships and awards committee, 2014 - 2015.
26. Member, Admissions Committee, 2014
27. Member, Committee to develop curriculum for *Advanced Cybersecurity Experience for Students* (ACES) honors program, June 2012 – November 2013.
28. Member, Teaching Evaluation Committee, 2011 - 2013.
29. Member, Tenure Committee for Assistant Professor Nick Feamster, Fall 2012.
30. Member, Faculty hiring committee for positions in cybersecurity and software systems, Fall 2011 - Spring 2012.
31. Member, Teaching Evaluation Committee, 2011 - 2012.
32. Member, Faculty hiring committee for director of Maryland Cybersecurity Center, Fall 2010 - Spring 2011.
33. Member, Faculty hiring committee for positions in cybersecurity, Fall 2010 - Spring 2011.
34. Member, Honors Program committee, Fall 2010 - Spring 2011.
35. Member, Teaching Evaluation Committee, 2010 - 2011.
36. Elected Member, Department Council, 2010 - 2011. 6 professors are elected annually by the rest of the faculty to sit on this committee.
37. Member, Honors Program committee, Fall 2009 - Spring 2010.
38. Member, Friday Faculty Lunch committee, Fall 2009 - Spring 2010.
39. Member, Teaching Evaluation Committee, 2009 - 2010.
40. Elected Member, Department Council, 2009 - 2010.
41. Member, Graduate Admissions committee, Spring 2008.

42. Faculty speaker, undergraduate recruiting day, March 13, 2008.
43. Elected Member, Department Council, 2007 - 2008.
44. Chair, Graduate Student Evaluation committee, 2008.
45. Member, Faculty panel to advise undergraduates interested in pursuing graduate school, November 8, 2007.
46. Member, Undergraduate Curriculum Evaluation committee, Spring 2007–2008.
47. Chair, Graduate Student Evaluation committee, 2007.
48. Chair, Faculty Retreat Committee on Undergraduate Education, Fall 2005 - Spring 2006.
49. Chair, Graduate Student Evaluation committee, 2006.
50. Member, Graduate Admissions committee, Spring 2005.
51. Chair, Graduate Student Evaluation committee, 2005.
52. Member, Faculty Retreat Committee on Graduate Student Recruiting, Admissions, and Retention, Fall 2003 - Fall 2004.
53. Member, Friday Faculty Lunch committee, Fall 2003 - Spring 2004.
54. Faculty judge, High-school programming contest, Spring 2003.
55. Member, Graduate Admissions committee, Spring 2003.
56. Member, Education sub-committee to revise the introductory undergraduate curriculum for computer science, Fall 2002.
57. Member, Lab committee, Fall 2002 - Spring 2003.
58. Member, Education committee, Fall 2002 - present.
59. Member, Programming Languages, Software Engineering, and Human-Computer Interaction Field Committee, Fall 2002 - present.

IV.B.2. Campus Service – College

1. Member, UMIACS Steering Committee, 2020 - 2021.
2. Member, UMIACS APT Committee, 2016 - 2017.
3. Member, Hiring committee for the Maryland Cybersecurity Center Senior position (hosted by UMIACS), 2015 - 2016.
4. Member, UMIACS APT Committee, 2015 - 2016.
5. Member, Hiring committee for the Maryland Cybersecurity Center Director (hosted by UMIACS), January - October 2013.
6. Member, Committee to develop Oral communications class in response to new University requirement, August 2010—October 2010.
7. Member, UMIACS APT Committee, 2009 - 2010.
8. Member, UMIACS APT Committee, 2006 - 2007.
9. Member, Business Office Coordinator search committee, July 2005.

IV.B.3. Campus Service – University

1. Chair, Provost’s Computing Science Working Group, April 2019 – December 2019.
2. Chair, Security Enhancements Subcommittee, President's Task Force on Cybersecurity, April - June 2014
3. Chair, Infrastructure Focus Area, IT Strategic Planning task force. One of four Chairs; output was a document presenting the University's strategic plan for IT, 2013.
4. Director, Maryland Cybersecurity Center (MC2) (a joint effort of the CMNS and Engineering colleges), Oct 2011 - 2013.
5. Team mentor, QUEST program (jointly run by the Smith school, the Engineering school, and CMNS), Fall 2010.
6. Team mentor, QUEST program, Fall 2009.
7. Panelist for CAREER proposal preparation workshop, June 13, 2005.

IV.B.6. Offices and Committee Memberships

1. **Steering committee Vice Chair**, Developing Secure Systems Summit (DS3), since November 2019.
2. Steering committee, ACM SIGPLAN Workshop on Principles of Secure Compilation (PriSC), since February 2019.
3. **Steering committee Chair**, ACM SIGPLAN-SIGACT Symposium on the Principles of Programming Languages (POPL); member, July 2015 – February 2021.
4. Steering committee, IEEE Computer Security Foundations Symposium, October 2014 – December 2019.
5. Steering committee, IEEE Conference on Cybersecurity Development (SecDev), September 2017 – December 2019.
6. Member of advisory board, Proceedings of the ACM on Programming Languages (PACMPL) journal, January 2016 – present.
7. **Chair, ACM Special Interest Group on Programming Languages (SIGPLAN)**, July 2015 – July 2018 (elected position). Serving in role of **Past Chair** from July 2018 – 2021.
8. Steering committee, ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI), July 2015 – 2018.
9. Steering committee, ACM SIGPLAN Conference on Systems, Programming, Languages and Applications: Software for Humanity (SPLASH)**,** July 2015 – July 2018.
10. Steering committee, ACM SIGPLAN International Conference on Functional Programming (ICFP)**,** July 2015 – July 2018.
11. Member, ACM SIGPLAN ad hoc committee on Empirical Evaluations, August 2017—present. See http://sigplan.org/Resources/EmpiricalEvaluation/
12. Member, ACM SIGPLAN ad hoc committee on Climate Impact, March 2017—present. See http://www.sigplan.org/Resources/Climate/
13. Member, Program Review Committee, Institute for Defense Analyses (IDA) Center for Computing Sciences (CCS), Bowie, Maryland. October 2013–March 2016.
14. Invited participant, National Science Foundation (NSF) Workshop on Formal Methods and Security, College Park, MD, November 19-20, 2015.
15. Steering Committee, ACM SIGPLAN-SIGACT Symposium on the Principles of Programming Languages (POPL), January 2011 – 2014.
16. Invited participant, National Academy of Sciences Kavli Frontiers of Science Symposium (one of five computer scientists invited, out of more than 100 participants), November 2013.
17. Invited participant, National Science Foundation (NSF) Workshop on Formal Methods: Future Directions and Transitions to Practice, San Diego, CA, December 5 - 6, 2012.
18. Steering Committee, 1st Workshop on Theory and Practice of Provenance (TaPP), February 2009.
19. Member of DARPA Information Processing Technology Office (IPTO) Futures Panel, Fall 2007. This panel consists of a few select faculty gathered to help IPTO chart its research goals and to develop potential programs.
20. Member of DARPA Computer Science Study Panel (CS2P), 2006 - 2009. This panel consists of a dozen select junior faculty from U.S. institutions, convening four times during 2006, and twice during 2007.
21. Participant, National Science Foundation, Computer Systems Research Future Directions Workshop, March 2010.
22. Invited participant, NSF workshop on Virtual Execution Environments for Scientific Computing, September 2010.
23. Invited participant, NSF workshop on Trustworthy Computing, November 2010.

24. Member of DARPA Information Science and Technology Board (ISAT), September 2009 - 2012. This panel consists of 30 individuals (largely faculty and researchers) from U.S. institutions which advises DARPA by conducting studies on topics of increasing relevance and impact.
25. Member, CyberMaryland 2.0 Task Force, April - October 2012. Panel commissioned out of the Maryland Department of Business and Economic Development (DBED) to examine progress toward goals for increasing Maryland's impact and standing in the cybersecurity arena.

IV.B.7. Leadership Roles in Meetings and Conferences

1. Moderator, Panel on the POPLmark Challenge 15 Year Retrospective, co-located with POPL, January 21, 2020 (see https://popl20.sigplan.org/track/POPL-2020-poplmark-15-year-retrospective-panel).
2. Co-organizer, Conference on Advancing Interdisciplinary Integration of Computational Thinking in Science, supported by National Science Foundation grant 1812860 and 1812916. Held May 2-5, 2019, in College Park, MD.
3. Co-organizer, Dagstuhl Seminar on the synergy between programming languages and cryptography, November 30 - Dec 5, 2014. http://www.dagstuhl.de/en/program/calendar/semhp/?semnr=14492
4. Co-organizer, ISAT workshop on Fostering Programming Language Adoption, February 2013.
5. Organizer, Principles of Programming Languages Mini-Workshop, held at the University of Maryland, October 2, 2011.
6. Local arrangements chair, ACM SIGPLAN International Conference on Functional Programming (ICFP), September 2010.
7. Co-organizer, ISAT study on Software Synthesis, August 2010 - 2011
8. Tutorials Chair, ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI), June 2010.
9. Co-organizer, the Mid-Atlantic Programming Languages Seminar (MAPLS), held in conjunction with the New Jersey Programming Languages Seminar (NJPLS) at the University of Maryland, November 2007.
10. Co-organizer, the Mid-Atlantic Programming Languages Seminar (MAPLS), held in conjunction with the New Jersey Programming Languages Seminar (NJPLS) at the University of Maryland, November 2005.

IV.B.8. Other Non-University Committees, Memberships, Panels, etc.

1. Invited participant, White House Office of Science and Technology panel on policies for public access to government funded research, March 17, 2020.
2. Invited participant, NSA-sponsored workshop on Secure Coding, September 10-11, 2018.
3. Member, USM Chancellor's Task Force on Cyber Security, November 2010–February 2011.
4. Member, Panel on Research Evaluation, European Computer Science Summit, October 24-26, 2016, Budapest, Hungary.

IV.C. External Service and Consulting

IV.C.1. Community Engagements, Local, State, National, International
IV.C.2. International Activities
IV.C.3. Corporate and Other Board Memberships
IV.C.4. Entrepreneurial Activities
IV.C.5. Consultancies *(to local, state and federal agencies; companies; organizations)*
IV.C.6. Historical External Service and Consulting (10+ years ago)
IV.C.7. Other

IV.D. Non-Research Presentations
IV.D.1. Outreach Presentations

1. Career day speaker, Eastern Middle School, May 2, 2014; May 6, 2016.

IV.D.2. Other

IV.E. Media Contributions
IV.E.1. Internet
IV.E.2. TV
IV.E.3. Radio
IV.E.4. Digital Media
IV.E.5. Print Media
IV.E.6. Blogs

1. The Programming Languages Enthusiast, http://www.pl-enthusiast.net. Launched June 2014; 300K total pageviews as of December 2017. (Co-edited and authored with Swarat Chaudhuri, Rice University, 2014-2015.)

IV.F Community & Other Service

1. Committee Chair, Cub Scout Pack 1250, Laurel, MD, August 2012–March 2015.
2. Youth Soccer coach, Greater Laurel United Soccer Club, August 2013–May 2016.

V. Awards, Honors and Recognition
V.1. Research Fellowships, Prizes and Awards

1. Co-author of Distinguished paper at the ACM Symposium on Principles of Programming Languages, 2021.
2. Co-author of Distinguished paper at USENIX Security Symposium, 2020.
3. Co-author of paper winning the NSA's Best Scientific Cybersecurity Competition, 2018.
4. IEEE Technical Committee on Security and Privacy Community Service award, 2017
5. University of Maryland Distinguished Scholar-Teacher, 2015-2016.
6. Co-author of paper winning the ASPLOS Conference's Best Paper Award, 2015.
7. Co-author of paper winning the NSA's Best Scientific Cybersecurity Competition, 2013.
8. NSF CAREER Award, 2004.
9. ACM SIGPLAN Doctoral Dissertation award, 2002.
10. Information Assurance Institute (IAI) Postdoctoral Research Fellowship, Department of Computer Science, Cornell University, 2001.
11. ARINC Technical Excellence Award, 1995.
12. Member of Phi Beta Kappa National Honors Society, 1993.

V.2 Teaching Awards

1. Department of Computer Science Faculty Teaching Award, Spring 2014.
2. Department of Computer Science Faculty Teaching Award, Spring 2007.
3. Department of Computer Science Faculty Teaching Award, Spring 2005.
4. Department of Computer Science Faculty Teaching Award, Spring 2003