Mukesh Tiwari

Conference Publication

[1] Toby Murray, Mukesh Tiwari, Gidon Ernst, and David A. Naumann. Assume but Verify: Deductive Verification of Leaked Information in Concurrent Applications. In Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, CCS '23, Copenhagen, Denmark, 26-30 Nov. `https://github.com/mukeshtiwari/IFMachine/`. (In this work, I formally verified the case studies: location-server, federated machine learning server, auction-server, and email-server in SecureC; project duration: 1.6 years).

[2] Thomas Haines, Rajeev Goré, and Mukesh Tiwari. Machine-checking Multi-Round Proofs of Shuffle:Terelius-Wikstrom and Bayer-Groth. 32nd USENIX Security Symposium (USENIX 2023), Anaheim, California, USA, August 9-11, 2023. `https://github.com/mukeshtiwari/secure-e-voting-with-coq`. (co-developer with Thomas Haines. I proved facts related to zero-knowledge proof and knowledge soundness in the Coq theorem prover; project duration: 2 years).

[3] Nadim Kobeissi, Georgio Nicolas, and Mukesh Tiwari. Verifpal: Cryptographic Protocol Analysis for the Real World. In Karthikeyan Bhargavan, Elisabeth Oswald, and Manoj Prabhakaran, editors, Progress in Cryptology - INDOCRYPT 2020, pages 151–202, Cham, 2020. Springer International Publishing. (co-developer with Georgio Nicolas. I worked on proofs related to Verifpal model in Coq; project duration: 8 months)

[4] Thomas Haines, Rajeev Goré, and Mukesh Tiwari. Verified Verifiers for Verifying Elections. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS '19, page 685–702, New York, NY, USA, 2019. Association for Computing Machinery. `https://github.com/mukeshtiwari/secure-e-voting-with-coq`. (co-developer with Thomas Haines. I worked on efficient finite field arithmetic, required for efficient zero-knowledge proof validation of well-formedness of a ballot; project duration: 1 year)

[5] Thomas Haines, Dirk Pattinson, and Mukesh Tiwari. Verifiable Homomorphic Tallying for the Schulze Vote Counting Scheme, In Verified Software: Theories, Tools, and Experiments. Springer, 2019. `https://github.com/mukeshtiwari/EncryptionSchulze/tree/master/code/Workingcode` (lead developer, project duration: 2 years)

+44-7824648138
mt883@cam.ac.uk, mukesh.tiwari@anu.edu.au, mukeshtiwari.iiitm@gmail.com
mukeshtiwari.github.io/ • *mukesh-tiwari-3609486/*
mukeshtiwari

[6] Milad K. Ghale, Rajeev Goré, Dirk Pattinson, and Mukesh Tiwari. Modular Formalisation and Verification of STV Algorithms. In Robert Krimmer, Melanie Volkamer, Véronique Cortier, Rajeev Goré, Manik Hapsara, Uwe Serdültt, and David Duenas-Cid, editors, Electronic Voting, pages 51–66, Cham, 2018. Springer International Publishing. `https://github.com/mukeshtiwari/Modular-STVCalculi`. (co-developer with Milad K. Ghale. I proved some of the critical theorems required for extracting an OCaml code; project duration: 8 months)

[7] Lyria Bennett Moses, Rajeev Goré, Ron Levy, Dirk Pattinson, and Mukesh Tiwari. No More Excuses: Automated Synthesis of Practical and Verifiable Vote-Counting Programs for Complex Voting Schemes. In Robert Krimmer, Melanie Volkamer, Nadja Braun Binder, Norbert Kersting, Olivier Pereira, and Carsten Schürmann, editors, Electronic Voting, pages 66–83, Cham, 2017. Springer International Publishing. `https://github.com/mukeshtiwari/formalized-voting/tree/master/SchulzeOCaml` (lead developer, project duration: 8 months)

[8] Dirk Pattinson and Mukesh Tiwari. Schulze Voting as Evidence Carrying Computation. In Mauricio Ayala-Rincón and César A, Muñoz, editors, *Interactive Theorem Proving*, pages 410–426. Cham, 2017. Springer International Publishing. `https://github.com/mukeshtiwari/formalized-voting/blob/master/paper-code` (lead developer, project duration: 1 year)

[9] Mukesh Tiwari, Karm V. Arya, Rahul Choudhari, and Kumar S. Choudhary. Designing Intrusion Detection to Detect Black Hole and Selective Forwarding Attack in WSN Based on Local Information. In 2009 Fourth International Conference on Computer Sciences and Convergence Information Technology, pages 824–828, Nov 2009. (lead developer, project duration: 1 year)

[10] Rahul Choudhari, Karm V. Arya, Mukesh Tiwari, and Kumar S. Choudhary. Performance Evaluation of SCTP-Sec: A Secure SCTP Mechanism. In 2009 Fourth International Conference on Computer Sciences and Convergence Information Technology, pages 1111–1116, Nov 2009. (co-developer with Rahul Choudhari, project duration: 1 year)

Workshop Publications

[1] Mukesh Tiwari and Dirk Pattinson. Machine Checked Properties of the Schulze Method. 7th Workshop on Hot Issues in Security Principles and Trust 2021.

[2] Mukesh Tiwari. Towards Leakage-Resistant Machine Learning in Trusted Execution Environments. Program Analysis and Verification on Trusted Platforms (PAVeTrust) Workshop 2021.

[3] Nadim Kobeissi, Georgio Nicolas, and Mukesh Tiwari. Verifpal: Cryptographic Protocol Analysis for the Real World. Proceedings of the 2020 ACM SIGSAC Conference on Cloud Computing Security Workshop 2020.

Work in Progress

+44-7824648138

mt883@cam.ac.uk, mukesh.tiwari@anu.edu.au, mukeshtiwari.iiitm@gmail.com

mukeshtiwari.github.io/ • *mukesh-tiwari-3609486/*

mukeshtiwari

[1] Formally Verified Verifiable Group Generators. In this work, we develop a formally verified algorithm that can be used to bootstrap a democratic election (sole author) (work in progress). https://github.com/mukeshtiwari/Formally_Verified_Verifiable_Group_Generator.

[2] Modelling Networking Protocols Mathematically. In this work, we develop a formally verified framework that a network protocol designers can use to verify the properties of their protocols (joint work Timothy Griffin. In this work, I formally verified generalised graph algorithm on semiring algebra in the Coq theorem prover). (ongoing and planning to submit to CAV 2024). https://github.com/mukeshtiwari/Semiring_graph_algorithm

[3] An Algebraic Framework for Multi-Objective Optimisation. In this work, we develop a formally verified framework in the Coq theorem prover that can be used to model various multi-objective optimisation problem as a graph algorithm in the Semiring framwork. (joint work Timothy Griffin but I am leading the project). (ongoing). https://github.com/mukeshtiwari/Formally-Verified-MultiObjective-Optimisation

[4] Theorem Provers to Protect Democracies. In this work, we are formalising all the cryptographic components written in Java of SwissPost in the Coq theorem prover. Our goal is to replace the SwissPost Java implementations (https://bit.ly/3EODmnF) with mathematically proven correct Coq implementations to write an independent verifier for the scrutiny sheet of elections conducted by Swiss Post software programs (sole author) (work in progress and planning to submit to IEEE S&P 2024). https://github.com/mukeshtiwari/Dlog-zkp.

[5] Machine Checked Properties of the Schulze Method. In this work, we are formally verifying all the social choice properties of the Schulze method. (lead developer, joint work with Dirk Pattinson) (work in progress). https://github.com/mukeshtiwari/Schulzeproperties.

+44-7824648138
mt883@cam.ac.uk, mukesh.tiwari@anu.edu.au, mukeshtiwari.iiitm@gmail.com
mukeshtiwari.github.io/ • *mukesh-tiwari-3609486/*
mukeshtiwari