Mukesh Tiwari
Oxford, United Kingdom
+447824648138
mukesh.tiwari@cs.ox.ac.uk

To
The Hiring Committee
Rice University, USA

Application for the tenure-track Assistant Professor, Computer Science

Dear Hiring Committee,

My name is Mukesh Tiwari and I am a senior research associate at the University of Oxford, UK with expertise in formal methods, cybersecurity and privacy, and social choice theory. I am writing to apply for the tenure-track **Assistant Professor** job. I have extensive research experience in formal verification (Coq theorem prover), electronic voting, and cryptography, and my research experience makes me a valuable candidate for this post.

My research touches the lives of common people and solves real-world problems that matter to democracies and common people. For example, my paper (i) **Assume but Verify: Deductive Verification of Leaked Information in Concurrent Applications**, accepted in ACM CCS, develops a theory using the information-flow security principals for processing sensitive data –ethnic origin, political opinions, health-related data, and biometric data– of common people in secure enclave, e.g., Intel SGX, Arm TrustZone, etc. Moreover, we demonstrate the usability of our method by developing non-trivial case studies that handles sensitive data accompanied by the machine-checked mathematical proofs that none of them have unintended side-channel data leakage; (ii) **Machine-checking Multi-Round Proofs of Shuffle: Terelius-Wikstrom and Bayer-Groth**, published in USENIX Security, mathematically establishes a critical piece of code in the SwissPost voting software –used in legally binding elections in Switzerland– is correct (and debunks a decade old myth of the cryptographic community that Terelius-Wikstrom method is zero-knowledge proof. We have formally proved in the Coq theorem prover that it is a zero-knowledge argument and not a zero-knowledge proof); (iii) **Verifiable Homomorphic Tallying for the Schulze Vote Counting Scheme**, published in VSTTE, not only developes a publicly verifiable method to count encrypted ballots for a complex voting method but it is also proven correct in the Coq theorem prover to ensure that there is no gap between the pen-and-paper proof and the actual implementation; (iv) **Verified Verifiers for Verifying Elections**, published in ACM CCS, develops a mathematically proven correct tool in the Coq theorem prover to verify the elections conducted by the International Association for Cryptologic Research. We have used our tools to verify the integrity of IACR elections; (v) **Modular Formalisation and Verification of STV Algorithms**, published in E-Vote, develops a mathematically proven correct tool in the Coq theorem prover. We have used this tool to verify the results of Australian Senate election; (vi) **Verifpal: Cryptographic Protocol Analysis for the Real World**, published in INDOCRYPT, develops a tool that can used to model real work cryptographic protocol, and Verifpal has been used by many researchers to model security and privacy aspect of digital contact tracing during COVID, etc. At Cambridge, I developed a mathematically proven correct tool in the Coq theorem prover that can be used by networking researchers to model networking-protocols in the abstract setting of semirings (and we are in the process of submitting our paper in CAV 2024). Currently at Oxford, I am exploring the avenues to combining security protocol with their implementations using session types. Our goal is to produce a formal executable model of real-world (distributed) applications, e.g., **Signal** chat messenger, etc. In a nutshell, all my research so far has an impact on the lives on common people and researchers.

Although, as a person, I am slightly introvert, but I firmly believe interdisciplinary research is the key to solve challenging problem pertaining to society. Therefore, I like to chat and work with diverse set of researchers, which is evident from my projects involving myriad of concepts, e.g., formal method, cryptography, voting, social choice, separation logic, information-flow security, graph theory, session types, etc.

Your Sincerly,

Mukesh Tiwari

Mukesh Tiwari

Education

2016–2020 **PhD, Computer Science**, *Australian National University*, Canberra, Australia

2004–2009 **Integrated Postgraduate**, *Indian Institute of Information Technology & Management*, Gwalior, India

PhD thesis

Title *Formally Verified Verifiable Electronic Voting Scheme*

Supervisor Dirk Pattinson

Description We focussed on three main challenges posed by electronic voting: correctness, privacy, and verifiability. We addressed correctness by using a theorem prover to implement a vote-counting algorithm, privacy by using homomorphic encryption, and verifiability by generating a independently checkable scrutiny sheet. Our work had been formalised in the Coq theorem prover.

Employment

2021-2023 **Senior Research Associate**, *University of Oxford*, Oxford, United Kingdom

I am working on connecting symbolic models of cryptographic protocols and their verified implementations using session types. The goal is to obtain a mathematically proven correct distributed implementation that mimics the real-world situation.

2021-2023 **Senior Research Associate**, *University of Cambridge*, Cambridge, United Kingdom

I worked on formalising a networking-protocol framework based on semiring algebraic structure in the Coq theorem prover. The goal was to develop a mathematical proven correct framework so that a protocol designer could assess the properties of their protocols using my framework.

2020-21 **Research Associate**, *University of Melbourne*, Melbourne, Australia

I worked with Toby Murray on *Security Concurrent Separation Logic*. The aim was to mathematically reason about memory safety and information flow property of concurrent programs written in C.

2018-20 **Tutor**, *Australian National Univeristy*, Canberra, Australia

I was a tutor for a first year logic course and Haskell programming course. My role was to help students understand the concepts, clearing their doubts, and assisting them in homework.

2013–2015 **Lecturer**, *International Institute of Information Technology*, Bhubaneswar, India

This role was primarily teaching focussed, and the courses I taught were *C programming, Java Programming, Web Technologies, Compiler Design,* and *Cryptography*. In addition, every year I supervised two master's students in their final year project.

2012–2013 **Haskell Developer**, *Parallel Scientific*, Colorado, USA

In this role, my primary job was research and prototype high performance software programs, mainly linear algebra algorithms written in Haskell.

+44-7824648138

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

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

mukeshtiwari

2009–2012 **Technical Assistant**, *Government of India*, Kolkata, India

I worked as a developer for automating the day-to-day job, including enforcing the security policies of the organisation.

2008–2008 **Summer Intern**, *Arcelor-Mittal, Research & Development Technological Centre*, Avilés, Spain

I worked on formalising many business requirements into a linear programming problem and wrote a custom interface that interacted with Arcelor-Mittal's in-house linear programming solver.

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/`. (co-developer with Toby Murray and Gidon Ernst. In this work, I formally verified the case studies: differentially private location-server, federated machine learning server, aution-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 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 code extraction; 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.

Community Service (Reviewer)

[1] ACM Transactions on Privacy and Security

[2] Sadhana - Academy Proceedings in Engineering Sciences

[3] IEEE Security & Privacy

Skills

Coding Coq, Lean, Haskell, OCaml, Python, C, Racket

Language Hindi, English

Awards

HDR Fee Remission Merit Scholarship

ANU PhD Scholarship (International)

Full Scholarship to attend DeepSpec Summer School 2018, Princeton University

Travel Scholarship to attend Marktoberdorf Summer School 2019

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

References

- Dirk Pattinson, Research School of Computer Science, Australian National University, Canberra, dirk.pattinson@anu.edu.au
- Toby Murray, School of Computing and Information Systems, University of Melbourne, Melbourne, toby.murray@unimelb.edu.au
- Timothy Griffin, Computer Laboratory, University of Cambridge, Cambridge, tgg22@cam.ac.uk
- Thomas Haines, Research School of Computer Science, Australian National University, Canberra, thomas.haines@anu.edu.au

+44-7824648138

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

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

mukeshtiwari