# Verified Trustworthy Software Systems

Modern society is faced with a fundamental problem: the reliability of complex, evolving software systems on which society critically depends cannot be guaranteed by the established, non-mathematical computer engineering techniques such as informal prose specification and ad-hoc testing. The situation is worsening: modern companies are moving fast, leaving little time for code analysis and testing; the correctness of concurrent and distributed programs cannot be adequately assessed using traditional testing methods; users of mobile applications often neglect to apply software fixes; and malicious users increasingly exploit even simple programming errors, causing major security disruptions. Building trustworthy, reliable software is becoming harder and harder to achieve, whilst new business and cyber-security challenges make it of escalating critical importance.

The challenge is to bring program specification and verification to the heart of the software design process. Most code validation is based on out-dated ideas of trusting that the internal, unpublished procedures of a company is robust and the assumption that the developer is not malicious. High-grade industry players, such as Airbus, do use sophisticated internal processes and tools to establish some degree of trust in their code, which can then certified by government bodies such as the UK National Cyber Security Centre. We should, however, be able to do better than this. Software should be judged on fundamental scientific principles, with proper answers to questions such as: 'What does this software do and what does it not do?'; 'Are the behaviours exhibited by the software the ones we want?'; and 'How do we assess that the software does what it says it does?'. It should be possible to bring proper, mathematically rigorous, scientific method to our software development, in line with standard engineering practice.

The specialist academic and industrial research community is ready to tackle this challenge: proof assistants area mature; symbolic testing and verification techniques and tools are tractable; and real-world programs are specified and verified in academia and industry. On the one hand, academics and software companies such as Facebook, Amazon and Google are developing industrial-strength symbolic testing and verification tools that help the general programmer uncover simple bugs, such as memory overruns and null-pointer exceptions. Such tool development is, however, currently limited. Indeed, Facebook has the saying 'this journey is 1% finished' [Facebook Infer tool for C and Java, open-source announcement, 2015]. On the other hand, academics and specialist software companies are building fully specified and verified software systems: Altran UK have delivered fully verified software for UK air-traffic control with contractual guarantees to fix any bugs found; the DARPA $70M project 'HACMS: High-assurance Cyber Military Systems' have developed fully verified system software stacks for autonomous vehicles which passed strict security testing with flying colours; and the DeepSpec project is a $10M NSF Expedition in Science to develop the specification, testing and verification of the system software stack and to create an undergraduate curriculum to train future software engineers in specification and verification. Such fully verified systems are, however, closed and technically complicated. This approach is unable to handle the mainstream software systems being used today.

The Royal Society meeting on 'Verified Trustworthy Software Systems' was unique, bringing together verification experts from academia and industry, systems and security experts interested in verification, industrial scientists and engineers wanting to use verification, and employees of government bodies thinking about cyber security and the certification of software. The speakers were international experts in systems, security and verification with the remit to present new long-term verification challenges, describe real-world software verification projects, discuss new application areas, and assess how to trust verified software. The speakers were:

- Sir Tony Hoare FRS from Microsoft Research (MSR), Cambridge, a founder of the field who opened and closed the meeting;
- J. Strother Moore from the University of Texas, who pioneered tractable verification in industry by developing ACL2 and using it to verify floating-point arithmetic in AMD processors;
- Cédric Fournet from MSR Cambridge and the MSR-INRIA Joint Centre in Paris, who developed the verified reference implementation miTLS for the TLS protocol;
- Kathleen Fisher from Tufts University, USA, who created the DARPA project HACMS;
- Neil White, head of engineering at Altran UK;
- Daniel Kroenig from Oxford, who built the CBMC model-checker, and is the CEO and founder of the start-up DiffBlue;
- Gerwin Klein from Data61 in Australia, who developed the verified kernel sel4 which underpins the HACMS project with his colleague Gernot Heisser;

- Nickolai Zeldovich from MIT, who has developed a verified operating system with his colleague Adam Chlipala from the DeepSpec project;
- Mark Batty from the University of Kent, who has developed weak memory models for x86, Power and ARM CPUs, and has significantly influenced the C11 language specification with Peter Sewell at Cambridge;
- Peter O'Hearn, engineering director at Facebook London who leads the team working on the INFER verification tool;
- Philippa Gardner from Imperial College London, who explores the fundamental theory of abstract specification for concurrent programs and web programs;
- Fred Schneider from Cornell, who is an expert in cybersecurity and previously worked in verification;
- Marco Pistoia from IBM Research in New York, who combines static analysis and machine learning to provide information-flow security;
- Michael Backes from Saarland University, Saarbrücken, who works on information security and privacy and is part of the ERC Synergy Grant imPACT;
- Alexey Gotsman from IMDEA, Madrid, who works on the fundamental theory of distributed systems;
- Xavier Leroy from INRIA in Paris, who developed the efficient verified compiler CompCert which underpins the HACMS project; and
- Greg Morrisett from Cornell, who aims to build secure, reliable, high-performance software systems.

I would like to thank personally all the speakers and people in the audience for making the Royal Society Scientific meeting on 'Verified Trustworthy Software Systems' such a fun meeting. In this accompanying publication, some of the speakers and their co-authors describe their current and future research ideas in verification, aiming to make their papers accessible to a general audience interested in verification. There are some absolute gems. Enjoy reading!

Philippa Gardner, Imperial College London