Institution: Imperial College London

Unit of Assessment: 11 Computer Science and Informatics

Title of case study: Infer: Scalable Static Analysis at Facebook and Beyond

Period when the underpinning research was undertaken: Oct 2005 – Jun 2012

Details of staff conducting the underpinning research from the submitting unit:

Name(s):	Role(s) (e.g. job title):	Period(s) employed by submitting HEI:
Cristiano Calcagno	Research Associate EPSRC Advanced Fellow Lecturer Honorary Lecturer	2003 – 2005 2005 – 2011 2008 – 2017 2017 – present

Period when the claimed impact occurred: Aug 2013 – Dec 2020

Is this case study continued from a case study submitted in 2014? No

1. Summary of the impact

Infer is an open-source static analysis tool for finding memory-safety bugs in Java, C++ and Objective-C code bases. It is based on fundamental advances in automated program reasoning made by Cristiano Calcagno (Imperial College London), Dino Distefano (Queen Mary, University of London), and collaborators between 2005 and 2011. In 2009, Calcagno and Distefano founded start-up company Monoidics to commercialise Infer. Infer and its development team were acquired by Facebook in 2013 for an undisclosed sum, with the team extending the tool to cater to large-scale use. As described in a recent Communications of the ACM article, Infer now runs continuously on the millions of lines of code underlying the apps and back-end servers of Facebook, Instagram and WhatsApp. It also has many users in government and industry, such as the National Cyber Security Centre (NCSC), Amazon, Mozilla, Spotify, Uber, Freefem, JD.com, Marks and Spencer, Sky, OLA, Vuo, CodeAI and Tile. Infer finds reliability and security bugs, typically within minutes of their introduction so that they can be rapidly fixed. As a direct result of the basic research underpinning this case study, Infer has led to more than 100,000 bugs in the Facebook, Instagram and WhatsApp code bases being found and fixed before they reached production, including many security-critical bugs that would otherwise have potentially affected billions of users.

2. Underpinning research

Nature of the research insights and findings relating to the impact: In a series of research papers between 2006 and 2011, Cristiano Calcagno at Imperial College London and collaborators developed fundamental theory, efficient algorithms and tools for reasoning about software at an industrial scale. Almost all software used in industry manages memory using *pointers*. However, pointer-manipulating programs suffer from difficulties posed by *aliasing*, where a program can access common regions of memory via many pathways. This makes it hard to reason about a program's behaviour: a modification to memory made by one part of the program may end up – due to aliasing – influencing seemingly unrelated parts of the program. In 2001, *separation logic* was proposed as a new theory to aid in the reasoning about pointer-manipulating programs. At first, it was used solely for pencil-and-paper reasoning; it was unclear whether it would be possible to create automated tools to analyse programs using separation logic, and whether such tools would have any hope of scaling to real world industrial systems.

Calcagno and collaborators demonstrated in their papers and tools that scalable reasoning *is* possible using *abstract symbolic execution*, a static analysis technique that reasons about the behaviour of a program for all potential inputs simultaneously and can thus establish a *proof* that the program is correct, or demonstrate that the program contains *bugs* related to memory safety. This line of research culminated in a Journal of the ACM paper [R6, 2011], showing that reasoning can be made to scale to very large code bases through a new technique called *bi-abduction*. Bi-abduction allows the analysis of a large code base to be broken down into many mini-analyses, one per procedure of the program. This (a) decouples the scalability of analysis from the overall code base size, and (b) allows analysis *re-use*: when a developer changes a small part of a large code base, only the procedures actually relevant to the change must be re-analysed. The Journal of the ACM paper is an extended version of a POPL 2009 conference paper that introduced bi-abduction and won the POPL Ten-Year Test-of-Time award.

Outline of underpinning research, and associated dates: The research was conducted between 2005 and 2011. The first academic semi-automated reasoning tool for pointer-manipulating programs, using *abstract symbolic execution* based on separation logic, was Smallfoot [R1, 2005]. This led to extensions to the theory to support pointer arithmetic, common in C-family languages [R2, 2006]. The problems with Smallfoot were that users had to annotate all functions with preconditions, and that it only worked with simple data structures, e.g., linked lists. This was partly mitigated by *precondition discovery* [R3, 2007] and work on composite data-structure reasoning [R4, 2007]. Now the method could be used to analyse thousands of lines of code for the first time, including Microsoft device drivers [R5, 2008]. The industrial breakthrough came in 2009 with the introduction of *bi-abduction* for highly automated inference of function preconditions [R6, 2011] (an extension of the POPL 2009 paper that introduced bi-abduction). The combination of abstract symbolic execution and bi-abduction unlocked scalability to millions of lines of code and has enabled Infer's widespread industrial impact.

Bi-abduction is at the heart of the Infer tool that forms the basis of this case study: the large-scale roll-out of Infer at Facebook has involved engineering work to optimise the tool, but the core analysis engine remains a faithful realisation of the basic fundamental research introduced in [R1–R6], while Calcagno was working full time at Imperial.

3. References to the research

[R1] Josh Berdine, Cristiano Calcagno, Peter W. O'Hearn: Smallfoot: Modular Automatic Assertion Checking with Separation Logic. FMCO 2005: 115-137. Google Scholar cites: 474. DOI: 10.1007/11804192_6.

[R2] Cristiano Calcagno, Dino Distefano, Peter W. O'Hearn, Hongseok Yang: Beyond Reachability: Shape Abstraction in the Presence of Pointer Arithmetic. SAS 2006: 182-203. Google Scholar cites: 74. DOI: 10.1007/11823230_13.

[R3] Cristiano Calcagno, Dino Distefano, Peter W. O'Hearn, Hongseok Yang: Footprint Analysis: A Shape Analysis That Discovers Preconditions. SAS 2007: 402-418. Google Scholar cites: 60. DOI: 10.1007/978-3-540-74061-2_25.

[R4] Josh Berdine, Cristiano Calcagno, Byron Cook, Dino Distefano, Peter W. O'Hearn, Thomas Wies, Hongseok Yang: Shape Analysis for Composite Data Structures. CAV 2007: 178-192. Google Scholar cites: 271. DOI: 10.1007/978-3-540-73368-3_22.

[R5] Hongseok Yang, Oukseh Lee, Josh Berdine, Cristiano Calcagno, Byron Cook, Dino Distefano, Peter W. O'Hearn: Scalable Shape Analysis for Systems Code. CAV 2008: 385-398. Google Scholar cites: 308. DOI: 10.1007/978-3-540-70545-1_36.

[R6] Cristiano Calcagno, Dino Distefano, Peter W. O'Hearn, Hongseok Yang: Compositional Shape Analysis by Means of Bi-Abduction. J. ACM 58(6): 26:1-26:66 (2011). Extended

version of POPL 2009 paper. Google Scholar cites (J. ACM and POPL paper combined): 531. DOI: 10.1145/2049697.2049700.

4. Details of the impact

The impact arising from this research during the REF period of 1 August 2013 and 31 December 2020 is related to Facebook acquiring *Infer*, a start-up company and software tool based on the fundamental research described above, and the wide industrial adoption of the Infer tool. Facebook has integrated the Infer tool with their continuous development workflow, discovering and fixing a large number of bugs in products used by billions of users. After making the technology available as open source, there has been broad adoption by companies across the whole software industry.

Background to the Facebook acquisition. Based on the success of the foundational research described above for enabling scalable reasoning about pointer-manipulating programs, Calcagno and Distefano (based at Imperial College London and Queen Mary University of London, respectively) incorporated a start-up company, Monoidics Ltd. (company no. 06805518), in January 2009, as directors. They developed the *Infer* tool, an implementation of the line of research (papers R1 – R6 above) that culminated in the Journal of the ACM article [R6]. Monoidics was funded by Quantum Wave Capital, received support from the CARP EU FP7 project (led by Imperial College London), and reported ARM, Airbus and Mitsubishi Electric as customers. In July 2013, Facebook acquired the assets of Monoidics for an undisclosed sum and hired the engineering team, including Calcagno [I1]. Calcagno has since worked as a Software Engineer at Facebook, remaining a Lecturer on Leave of Absence at Imperial until 2017, and now retains his connection with Imperial as an Honorary Lecturer.

Impact of Infer at Facebook. Facebook integrated Infer into its perpetual software development process [I2]. As a result, Infer is now used to *continuously* analyse the multi-million-line code bases of both mobile apps and back-end services at Facebook, as well as the software associated with WhatsApp and Instagram [I3]. Each time a developer commits any code change, Infer analyses the change and participates in the code review process, indicating potential memory safety problems that the programmer may have overlooked. This is possible due to the scalability afforded by bi-abduction [R6]: only the procedures related to the code change must be re-analysed, meaning that analysis can complete within minutes, providing a report of potential problems while the code changes are fresh in the mind of the developer.

At a talk at Imperial College London in 2015, Facebook CTO Mike Schroepfer said about Infer: *"It's a really key component of being able to build an app and ship it weekly to a billion people. If you want to do this without everything grinding to a halt, it's really helpful to have a system that automatically detects and helps with bugs"* [I4]. In a 2019 Communications of the ACM article, Facebook explains that Infer now runs continuously on millions of lines of code, and that so far **more than 100,000 bugs** found by Infer have been fixed before affecting end users. It reports a **high fix rate of 70%** associated with memory safety bugs flagged by Infer [I3].

The high degree of impact that Infer continues to have, and its underpinning by the original research of Calcagno et al., is summed up well in the letter from Facebook's Director of Engineering [I5]: *"Today within Facebook, the verification technology originally designed by Prof Dino Distefano, Dr Cristiano Calcagno and colleagues contributes to the quality of the products we serve to over 2 billion people every day. Currently Infer analyses every applicable code change within Facebook and its family of apps (Messenger, WhatsApp, Instagram, Oculus). Every month, thousands of issues are fixed before these apps are shipped to users. Thus, Infer has saved Facebook engineers innumerable hours that they would have had to spend debugging the problems it detected, had those problems reached production."*

The significance of the link between the fundamental research and its impact was recognised by the Infer team receiving the prestigious 2016 CAV Award, which *"recognizes their contribution to the development of Separation Logic, the theory that underpins the Infer static analyser"*.

Impact of Infer beyond Facebook. Infer was made open source in 2015 [I6], making it available to other companies. The open source repository is very active (>150 contributors) and popular (starred by >12,000 GitHub users, forked >1,600 times). Companies who have publicly acknowledged using Infer include Amazon Web Services, Mozilla, Oculus, Spotify, Uber, Freefem, JD.com, Marks and Spencer, Sky, Money Lover, LA, Netcetera, Vuo, CodeAI, and Tile [I7]. The National Cyber Security Centre (NCSC, part of GCHQ) also uses Infer. We detail here the four most significant uses of Infer in industry and government beyond Facebook: by Spotify, Uber, Muse Dev (a recent start-up company), and NCSC.

Spotify: The Android Infrastructure team at Spotify have extended Infer in order to incorporate it in their continuous integration process for Android. In an article describing the collaboration [I8], A Spotify engineer writes: *"In our quest to make our code even better, we started using Infer. Infer found several legitimate issues that other tools had missed. The Infer team was also very helpful in following a few false positives that we encountered, and we now have it running on our build servers."*

Uber: Uber have reported using Infer in a portfolio of tools for static detection of null pointer exceptions, claiming that this endeavour *"reduced the number of NPEs observed in our apps in production by an order of magnitude"* [I9].

MuseDev: A spin-off company from Galois, MuseDev is a new company providing tools and services for automated code reasoning. As detailed further in the supporting letter from MuseDev CEO [I10], MuseDev runs a cloud-based code analysis service that uses Infer as its flagship analyser: *"Via Muse, Infer is now running daily in the cloud on several open source repositories. And we are actively working with a variety of enterprises to deploy and run MuseDev in their environments, including Fortune 500 and fast-growing venture-backed technology companies."* The "cool math" link on the MuseDev website is to a review article on separation logic, with a section entitled "Bi-Abduction and Facebook Infer" that indicates the importance of the underpinning research. Magill further stresses the breakthrough nature of this research: *"The technology developed at Imperial and Queen Mary ushered in a new wave of static analysis, capable of finding subtle and important bugs while running quickly enough to fit into modern agile development workflows. In fact, in talking with customers, we often use the terms 'first wave' and 'second wave' to distinguish just how different these techniques are from what came before."*

National Cyber Security Centre (NCSC): This case study is accompanied by a confidential letter from the Head of High Assurance Engineering Research at NCSC (available to reviewers of the case study), outlining the use of Infer by NCSC and the significance of the fundamental research on which Infer is based [I11].

Beneficiaries of the impact: The companies that use Infer are direct beneficiaries of the impact: 18 large organisations publicly advertise that they use the tool [I7], and the open source. The improvement in software quality afforded by Infer is to the benefit of the users of the products that these companies build. Facebook alone report that Infer has led to more than 100,000 fixed bugs in the Facebook, WhatsApp and Instagram code bases [I3].

Nature of the impact: Software bugs are expensive: a report from the Consortium for IT Software Quality (CISQ) estimates that the cost of poor-quality software in the US alone is USD2.84 trillion. By finding bugs early during the software development process, Infer helps reduce these costs, leading to major economic impact. Billions of people

worldwide depend on products developed by the companies that use Infer. The large number of software defects in these products that have been fixed during development, after identification by Infer, and that would otherwise have reached production to the detriment of end users is evidence of the broad societal nature of the impact [I3].

5. Sources to corroborate the impact

[I1] Techcrunch.com: "Facebook Acquires Assets Of UK Mobile Bug-Checking Software Developer Monoidics", July 2013. (Supports the claim that Facebook acquired the Monoidics technology and hired the team.) Link archived here.

[I2] C. Calcagno et al.: "Moving Fast with Software Verification". Proc. NASA Formal Methods (NFM), 2015. DOI: 10.1007/978-3-319-17524-9_1. (Supports the claim that Infer is integrated into Facebook's perpetual software development process, and reinforces the underpinning of [R1] and [R6].) PDF available here.

[I3] D. Distefano et al.: "Scaling Static Analyses at Facebook". Commun. ACM 62(8): 62-70, 2019. DOI: 10.1145/3338112. (Supports claims that a 70% "fix rate" for issues reported by Infer has been achieved, and that the use of Infer has led to more than 100,000 bugs being fixed in Facebook, WhatsApp and Instagram code bases.) PDF available here.

[I4] Jon Narcross: "Facebook chief gives Imperial sneak peek into tech future", November 2015. (Supports the claim that Infer is used widely by Facebook.) Link archived here.

[I5] Letter from Director of Engineering at Facebook, confirming the extent of the impact associated with their use of Infer and emphasising that it is a direct result of basic research that underpins this case study.

[I6] Infer on GitHub: https://github.com/facebook/infer. (Supports claim that Facebook open-sourced Infer and that the project is highly active, with many forks/stars and contributors.) Link archived here.

[I7] Facebook's Infer web page (Link archived here) and research page on Infer (Link archived here) insists companies who self-identify as using Infer. (Supports claim that several companies outside Facebook use Infer.)

[I8] J. Villard: "Collaboration with Spotify", fbinfer.com, March 2016. (Supports the claim that Spotify are using Infer.) Link archived here.

[I9] M. Sridharan: "Engineering NullAway, Uber's Open Source Tool for Detecting NullPointerExceptions on Android", Uber Engineering, October 2017. (Supports the claim that Uber used Infer to reduce null pointer exceptions by an order of magnitude.) Link archived here.

[I10] Letter from CEO of MuseDev, confirming the extent of the impact associated with their use of Infer and emphasising that it is a direct result of basic research that underpins this case study.

[I11] Confidential letter from the NCSC Head of High Assurance Engineering Research, detailing the use of Infer by NCSC and emphasising the importance of the research on which Infer is based, available to reviewers of this case study.