

1 Background

1.1 Topic of research

Mathematical quantitative tools (like probability theory and statistics) have played an increasing role both in the theory and practice of most sciences. Theory based reasoning and analysis of software systems has largely relied on logics and has made relatively limited use of quantitative mathematics. However, some fundamental notions in theoretical computer science could benefit from a quantitative study.

Consider *interference* [10, 21] between program variables: informally, the capacity of variables to affect the values of other variables. Non-interference, i.e. absence of interference, is often used in proving that a system is well-behaved, whereas interference can lead to obscure (mis-)behaviours. However misbehaviour in the presence of interference will generally happen only when there is *enough* interference. Think in terms of electric current: non-interference between variables X, Y is the absence of a circuit involving X, Y ; interference is the existence of a circuit; this however doesn't imply that there is enough "current" in the circuit to affect the behaviour of the system.

A concrete example is a software system with *access control*. To enter such a system the user has to pass an identification stage; whether subsequently authorised or failed, some information has been leaked so these systems present interference [10]. However if the interference in such systems is *small* we can be confident in the security of the system.

Possible applications of a quantitative theory of interference go beyond access control systems as the next subsection explores.

We propose to do foundational work in measuring interference within software systems, applying well understood information theoretic ideas to the development of methods for reasoning about the quantities of information flow within programs and other software systems. We will be guided in this by our future aim of developing analyses on the basis of these methods.

1.2 Academic and industrial context

Interference in a security setting has been studied by academics for many years in the guise of its dual: *non-interference* (see section 1.3). Applications of this work can be found in large scale, mission critical, security related domains

such as the military, banks and health care. Within these domains the man-power cost of ensuring non-interference has often been acceptable, however security concerns have moved in recent years into the public domain. The need for a cost effective, domain-appropriate approach to security is the motivation to develop methods that support an *automatic* analysis approach. Imagine a future user wishing to install a trusted application going to, say, "www.trustcertificates.com" and a certificate for the application then being automatically checked on the client side as part of the process.

In the long term we see many potential applications of this work such as certification of spyware, refinements to access control systems, improved control of statistical inference from databases, measurement of leakage of keys in cryptographic protocols, and improved reasoning about the probability of failures of critical components in safety or security related systems.

Consider the example of spyware, or adware, which is an increasing problem for ordinary users. Suppliers may self-certify that only statistical information (and no private information) is transmitted, but there is a tension between the users' desire for the "free" software and their uncertainty with respect to the validity of the "vendors" claim.

Our proposed research would enable the development of static analyses for a wide class of programs allowing measurement of the amount of information leaked into different variables or ports in a given program, as in figure 1.

| security class | example | leakage allowed |
|----------------|---|---|
| very high | credit card numbers passwords pin numbers | extremely little e.g. 1 billionth of a bit |
| high | salary informations tax details address | very little e.g. postcode ok |
| low | amount of free memory processor type | all |

Figure 1: possible security classes

1.3 Related work

In the last decade or so there has been significant activity around the question of how to construct programs that demonstrably have

the non-interference property. Two approaches have been an a priori one via type systems and an a posteriori one via program analysis. Volpano and Smith have successfully applied a types based approach to a variety of systems e.g. [24]. Examples of work that use the program analysis approach include Bodei’s work with the Nielsons [2], Malacaria and Hankin’s work using control flow analysis via Games [17], Clark, Hankin and Hunt’s application of flow logic to Idealised Algol [3].

Our research, in contrast, is concerned with measuring *the amount of information flow* caused by interference between variables by using information theoretic quantities. We have established the feasibility of our approach and argued the appropriateness of our measures in a number of workshop papers [4, 5, 6].

A precursor for this work was that of Dorothy Denning in the early 1980’s. In [9] she gives examples and intuitions about imperative program constructs. Also in [9] she gives an information theoretic definition for measuring interference between program variables¹: $\mathcal{H}(x_s|y_s) - \mathcal{H}(x_s|y_{s'})$. (As discussed in [6] this definition is actually incorrect.)

Other closely related work is that of McLean and Gray and McIver and Morgan. McLean presented a very general Flow Model in [19], related to Sutcliffe’s approach to information flow in [25]. On the basis of this model, McLean gave a probabilistic definition of security (with respect to flows) of a system. Gray presented a less general and more detailed elaboration of McLean’s flow model in [14], making an explicit connection with information theory through his definition of the channel capacity of flows between confidential and non-confidential variables. Recently McIver and Morgan devised a different information theoretic definition of information flow (and hence channel capacity) to Gray’s [18], aiming at a different security property than the one embodied in the Flow Model. They then use this to derive a non-probabilistic characterisation of the security property for an simple imperative language.

Although our work is currently the only such that *measures* the amount of information (in an information theoretic sense) that flows along channels of interference in a program, other researchers have recently taken quantitative approaches to information flow.

¹The formula measures the difference between the uncertainty for the possible values for x at the state s given knowledge of the possible values for y at the state s and the uncertainty for the possible values for x at s given knowledge of the possible values for y at s'

In [20] Di Pierro, Hankin and Wiklicky define probabilistic measures on flows in a probabilistic concurrent constraint setting where the interference comes via probabilistic operators. They use this to derive a quantitative measure of the similarity between agents written in this probabilistic concurrent constraint language. However, in contrast to our work, they do not measure quantities of information. Gavin Lowe has measured information flow in CSP by counting refusals [15] and Volpano and Smith have relaxed strict non-interference and developed a type system in which a well typed program will not leak its secret in polynomial time [27].

2 Programme/methodology

2.1 Aim of this research

We have already carried out a quantitative study in the context of a simple programming language and established results [6] relating classical information theory measures and interference between program variables. We have used these theoretical results as a basis for developing a static analysis which gives safe bounds on the amount of interference for a program in a simple **while** language [5].

We aim to widen the current areas of application of our methods of reasoning and analysis so that we can consider more sophisticated security attacks, such as timing attacks and statistical attacks, and reason about more complex systems that may be reactive, object-oriented, or incorporate probabilistic aspects.

We further aim to improve the *quality* of our analyses by calculating tighter bounds on the amounts of information that flows..

We have identified the following areas of investigation as being key to these aims: *expressivity* (extension of our work to a wider class of languages), *time* (extension to reasoning about and using time), and *equivalence relations* (using relations as a reasoning tool). These three areas are dealt with in more detail in the program of work given below.

A successful outcome of the research in the above three threads will provide conceptual tools for quantitative analyses of information flow in a typed higher-order object-oriented multi-threaded language. Such research would be sufficient for the analysis of software applications mentioned in the introduction. It is possible that in the time span of the project we will produce some software prototype, however, the primary aims are theoretical and we *do not* plan to produce software implementations.

2.2 Methodology

The methodology will be based on information theoretical, program analysis and program semantics techniques. In the following we list the techniques we plan to use (a + sign means that we have already used that particular technique)

Information theory methods: Venn diagrams(+), Fano inequality(+), relative entropy, Markov models, network of random variables. *Program analysis:* program dependency graph(+), abstract interpretation, slicing. *Program semantics:* denotational semantics(+), PER models(+), Game semantics, trace based semantics.

2.3 Timeliness and novelty

The foundational work for new analyses that we propose in this application is very novel.

The proposed collaboration is the only one in the world (to our knowledge) working to establish and develop a bridge between information theory and program analysis.

Because of the open system nature of modern computing (e.g. the internet, the grid) language based analyses are becoming increasingly important security tools. In this respect we argue that our approach is very timely.

2.4 Programme of work

The project consists of 12 work packages divided into three groups of equal size: **Expressivity (group A)** investigated by Malacaria, **Time (group B)** investigated by Clark, and **Equivalence Relations (group C)** investigated by Hunt. We plan to explore the three threads concurrently to exploit synergies between them.

The project will begin with some establishing work on the part of each P.I. After six months we will advertise for PhD students who will be trained in their first year through suggested courses, exposure to the establishing work, and attendance at weekly meetings of the whole group (all P.I.s and students). These regular meetings have been the main forum for conducting our joint research in the past. We intend that they provide a good research context for the students.

Near the end of the project we will organise a workshop to aid in the dissemination of the results of the project. The three work packages are outlined below and a plan for implementing them is given in the Gantt chart.

We expect to publish at least one paper for each non-training work package.

2.4.1 A: Expressivity

We have shown that conditional mutual information and conditional entropy are equivalent for deterministic languages²[5]. We aim to extend our work to a wider class of systems and languages, in particular probabilistic, interactive systems and OO languages.

Work package A1: *Treatment of probabilistic and interactive systems* will demand the more fundamental definition of interference given in terms of conditional mutual information. This will require the development of new techniques based on for instance, a Venn diagram style analysis of random variables [28] or the use of information theory in Bayesian style networks.

Work package A2: *Training PhD student*

Work package A3: *The analysis of probabilistic and OO languages.* This will be based on Game semantics [13, 1, 8] in the spirit of previous work by Malacaria and Hankin [16, 17] relating Games and program analysis. Preliminary unpublished results show that it is possible to plug quantitative analysis on top of Game based control flow graphs of higher order languages.

Work package A4: *Conditional mutual information.* The quality of the global analysis in both settings (probabilistic and OO) will depend on the quality of the analysis of language constructs. So far our best estimates for interference use Fano's inequality to get an upper bound on the quantity of implicit flows (as in if-statements). Our analysis of *explicit flows* is, in comparison, poor and we plan to improve it for example by studying conditional mutual information of random variables obeying a specific distribution.

2.4.2 B: Time

Our work to date has been highly conservative in its treatment of loops and recursion, e.g. [5]. To gain precision, and to analyse time-based covert channels, we need to account for time at some suitable level of abstraction. This would permit a more accurate measurement of *absolute leakage* from a loop by summing leakage per iteration. It would also allow measurement of *rates of leakage* for programs containing loops, such knowledge sometimes being more useful or more appropriate than knowledge of absolute leakage. Note that the classical information theory notion of rates of transmission may not be appropriate because of the variation of information

² I.e. for X, Y, Z random variables such that, $Z = f(X, Y)$, where f is any function $\mathcal{H}(Z|Y) = \mathcal{I}(X; Z|Y)$.

on the same *channel* at different *times*: e.g. a loop which leaks an arbitrarily large amount of information initially, and then loops indefinitely with no further leakage, has an *asymptotic* leakage rate of zero.

Work package B1: *Terminating programs via a functional semantics.* In many security settings it may be sufficient to be able to analyse bounded iterations. This can be done by unwinding loops into nested if statements and applying Fano’s inequality as in our earlier work [4, 5]. The accuracy of the bounds can be improved with respect to existing work by incorporating information about the context of expressions rather than analysing them as if they were not in the branch of an if statement. This approach, still based on end-to-end analysis as in [4, 5], will provide useful benchmarks for more general treatments that allow unbounded iteration.

Work package B2: *Training PhD student*

Work package B3: *Possibly non-terminating programs via a functional semantics.* Any extension of the denotational semantics approach of existing work would then need to define measures of information for the images of *partial functions*. At this point in the work program we will not yet be able to define or measure rates of leakage.

Work package B4: *Time-based measures via an operational semantics.* First develop a trace based, operational semantics at a level of abstraction still to be determined. This will enable the definition of a “per step” measurement of leakage. One possibility is to adapt the work of [23] which takes an operational semantics approach and defines various forms of partial probabilistic bisimulations on behaviours. Investigate and define useful definitions of leakage based on the previous stage, for instance a notion of “instantaneous” rates of leakage.

2.4.3 C: Relations and PER models

Some previous, qualitative, work on interference has made use of equivalence relations and partial equivalence relations (PERs) to re-formulate and generalise the original definitions of non-interference [22]. In [6] we have shown that the relational presentation provides a quite general setting, in which both traditional non-interference and our quantitative approach may be unified.

We intend to develop this work, forming a bridge to a large body of research on semantic models [21], type systems [27, 26] and static program analysis [7, 11, 12, 22]. The aim is to

enable the adaptation of these theories and techniques (traditionally applied in purely logical, qualitative settings) to the quantitative analysis of interference.

Work package C1: *Quantitative analysis of declassification systems.* Apply quantitative analysis to systems allowing *declassification* of confidential data [29]. In such systems, at certain points confidential data is deliberately allowed to leak, in order to allow such operations as password checking. The work of [29] does not address the question of how much of the declassified data is leaked (the informal justification of password declassification is, of course, that the password checking operation leaks very little).

After some initial preparatory work, it is envisaged that this thread will serve as an ongoing source of case-studies for packages C3 and C4.

Work package C2: *Training PhD student*

Work package C3: *Type-based analysis.* Investigate and develop type systems which provide quantitative guarantees about the flow of confidential information. The starting point will be a relational analysis of the type system of [27], in which it is shown that a (tightly constrained) password checking operation cannot leak the password in a time polynomial in the length of the password. Weaknesses of [27] which we aim to address are that it does not obviously generalise and that it provides only asymptotic limits on leakage.

Work package C4: *Abstract interpretation-based analysis* First, together with thread **B** adapt the work of [23] to provide a quantitative analysis of information flow based on *operational* models. Generalise this work to allow for systems which leak within well-defined quantitative bounds.

Then develop an abstract interpretation for a simple imperative language which can form the basis of a qualitative analysis of information flow. The intention here is to separate flow analysis into a qualitative stage (using an abstract interpretation abstracting over lattices of relations) and a quantitative stage (applying information theoretical techniques).

3 Relevance to Beneficiaries

The increased understanding of information theory in the context of measuring interference will benefit both academia and (in the longer term) industry.

We believe our research will impact on the academic community looking for quantitative methods to be used both in security systems

and large scale commercial applications. Spyware software analysis is an example of this. More generally we believe our foundational work will open the door for applying many techniques from the field of information theory to the analysis of software systems.

4 Dissemination and exploitation

Results will mainly be made public through the usual channels i.e. journal papers as well as conferences and workshops such as POPL, ETAPS, SAS, QAPL and WITS. For complete sections of research we aim to publish in journals such as Theoretical Computer Science, the Journal of Logic and Computation, and the Journal of Computer Security.

A web site publicising the aims and progress of the project and any related publications and events will be developed, hosted and maintained at the lead site.

We will also investigate if and when it will be appropriate to seek the attention of a wider public with a publication in a broad spectrum journal like Journal of ACM or even try to seek the interest of writers for New Scientist or similar publications.

Towards the end of the project (as shown on the Gantt chart) we will organise a workshop which, apart from aiding dissemination, will give feedback and assist in planning where to take the research in the post-project future.

5 Justification of resources

We ask for three PhD students, one for each site. The sites are all in close proximity within a single city and the P.I.s have an on-going research relationship, offering an excellent opportunity for the necessary cross-disciplinary training.

We estimate an average of a transatlantic trip and three European trips to conferences a year per site.

We request equipment funding sufficient to purchase a workstation for each PhD student and a laptop per site to be shared between the P.I. and the student for conference presentations etc.

We request funds for one site to organise the workshop.

As standard we request a percentage of funding for technical and administrative support. Slightly more technical support is requested for

the lead site to support the proposed project web site.

References

- [1] S. Abramsky, R. Jaghadeesan, and P. Malacaria. Full abstraction for pcf. *Information and Computation*, 163:409–470, December 2000.
- [2] C. Bodei, P. Degano, H. Riis Nielson, and F. Nielson. Static analysis for secrecy and non-interference in networks of processes. In *Proc. PACT'01*, number 2127 in Lecture Notes in Computer Science, pages 27–41. Springer-Verlag, 2001.
- [3] D. Clark, C. Hankin, and S. Hunt. Information flow for algol-like languages. *Computer Languages (Special Issue: Computer Languages and Security)*, 28(1):3–28, April 2002.
- [4] David Clark, Sebastian Hunt, and Pasquale Malacaria. Quantitative analysis of the leakage of confidential data. In Alessandra Di Pierro and Herbert Wiklicky, editors, *Electronic Notes in Theoretical Computer Science*, volume 59. Elsevier, 2002.
- [5] David Clark, Sebastian Hunt, and Pasquale Malacaria. Quantified interference for a while language. In *Proceedings of The Workshop on Quantitative Aspects of Programming Languages*, Barcelona, April 2004.
- [6] David Clark, Sebastian Hunt, and Pasquale Malacaria. Quantified interference: Information theory and information flow. In *Proceedings of the Workshop on Issues in the Theory of Security 2004*, Barcelona, April 2004.
- [7] P. Cousot and R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Conference Record of the Fourth Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 238–252, Los Angeles, California, 1977. ACM Press, New York, NY.
- [8] Vincent Danos and Russell Harmer. Probabilistic game semantics. In *Logic in Computer Science*, pages 204–213, 2000.
- [9] D. E. R. Denning. *Cryptography and Data Security*. Addison-Wesley, 1982.

- [10] J. Goguen and J. Meseguer. Security policies and security models. In *IEEE Symposium on Security and Privacy*, pages 11–20. IEEE Computer Society Press, 1982.
- [11] Sebastian Hunt. Pers generalise projections for strictness analysis (extended abstract). In *Proc. 1990 Glasgow Workshop on Functional Programming*, Workshops in Computing, Ullapool, 1991. Springer-Verlag.
- [12] Sebastian Hunt and David Sands. Binding time analysis: a new perspective. In *Proc. PEPM'91: Symposium on Partial Evaluation and Semantics-Based Program Manipulation*. ACM Press, 1991.
- [13] J. M. E. Hyland and C.-H. L. Ong. On full abstraction for pcf: I. models, observables and the full abstraction problem, ii. dialogue games and innocent strategies, iii. a fully abstract and universal game model. *Information and Computation*, 163:285–408, December 2000.
- [14] III J. W. Gray. Toward a mathematical foundation for information flow security. In *Proceedings of the 1991 IEEE Symposium on Security and Privacy*, pages 21–34, Oakland, California, 1991.
- [15] Gavin Lowe. Quantifying information flow. In *Proceedings of the Workshop on Automated Verification of Critical Systems*, 2001.
- [16] P. Malacaria and C. Hankin. Generalised flowcharts and games. In *Proceedings of the 25th International Colloquium on Automata, Languages and Programming*, pages 363–374. Springer-Verlag, 1998.
- [17] P. Malacaria and C. Hankin. Non-deterministic games and program analysis: an application to security. In *Proceedings of Logic in Computer Science (LICS)*. IEEE Press, 1999.
- [18] Annabelle McIver and Carroll Morgan. A probabilistic approach to information hiding. In *Programming methodology*, pages 441–460. Springer-Verlag New York, Inc., 2003.
- [19] J. McLean. Security models and information flow. In *Proceedings of the 1990 IEEE Symposium on Security and Privacy*, Oakland, California, 1990.
- [20] Alessandra Di Pierro, Chris Hankin, and Herbert Wiklicky. Approximate non-interference. In Iliano Cervesato, editor, *CSFW'02 – 15th IEEE Computer Security Foundation Workshop*. IEEE Computer Society Press, June 2002.
- [21] J. C. Reynolds. Syntactic control of interference. In *Conf. Record 5th ACM Symp. on Principles of Programming Languages*, pages 39–46, Tucson, Arizona, 1978. ACM, New York.
- [22] Andrei Sabelfeld and David Sands. A per model of secure information flow in sequential programs. In *Proc. European Symposium on Programming*, Amsterdam, The Netherlands, March 1999. ACM Press.
- [23] Andrei Sabelfeld and David Sands. Probabilistic noninterference for multi-threaded programs. In *Proc. 13th IEEE Computer Security Foundations Workshop*, Cambridge, England, July 2000. IEEE Computer Society Press.
- [24] Geoffrey Smith and Dennis Volpano. Secure information flow in a multi-threaded imperative language. In *Proc. 25th ACM Symposium on Principles of Programming Languages*, pages 355–364, San Diego, CA, Jan 1998.
- [25] D. Sutherland. A model of information. In *Proceedings of the 9th National Computer Security Conference*, 1986.
- [26] Dennis Volpano and Geoffrey Smith. A type-based approach to program security. In *Proceedings of TAPSOFT '97 (Colloquium on Formal Approaches in Software Engineering)*, number 1214 in Lecture Notes in Computer Science, pages 607–621, Lille, France, 1997.
- [27] Dennis Volpano and Geoffrey Smith. Verifying secrets and relative secrecy. In *Proc. 27th ACM Symposium on Principles of Programming Languages*, pages 268–276, Boston MA, Jan 2000.
- [28] Raymond W. Yeung. A new outlook on shannon's information measures. *IEEE Transactions on Information Theory*, 37(3), May 1991.
- [29] S. Zdancewic and A. C. Myers. Robust declassification. In *Proceedings of the 14th IEEE Computer Security Foundations Workshop*, pages pages 15–23, Cape Breton, Nova Scotia, Canada, June 2001.