.

UNIVERSITI PUTRA MALAYSIA

PROOF-CARRYING CODE FOR VERIFYING CONFIDENTIALITY OF MOBILE CODE THROUGH SECURE INFORMATION FLOW ANALYSIS

ABDULRAHMAN AHMAD ABDU MUTHANA

FSKTM 2008 20

PROOF-CARRYING CODE FOR VERIFYING CONFIDENTIALITY OF MOBILE CODE THROUGH SECURE INFORMATION FLOW ANALYSIS

By

ABDULRAHMAN AHMAD ABDU MUTHANA

Thesis Submitted to the School of Graduate Studies, Universiti Putra Malaysia, in Fulfillment of the Requirements for the Degree of Doctor of Philosophy

November 2008

Dedicated to my wife, Adeeba;
to my kids, Omar and Zahra;
to my family.

Abstract of thesis presented to the Senate of Universiti Putra Malaysia in fulfillment of the requirements for the Degree of Doctor of Philosophy

PROOF-CARRYING CODE FOR VERIFYING CONFIDENTIALITY OF MOBILE CODE THROUGH SECURE INFORMATION FLOW ANALYSIS

By

ABDULRAHMAN AHMAD ABDU MUTHANA

November 2008

Chairman: **Associate Professor Abdul Azim Abdul Ghani, PhD.**

Faculty: **Computer Science and Information Technology**

The growing dependence of our society and economy on networked information systems makes it essential to protect our confidential data from being leaked by malicious code. Downloading and executing code (possibly from untrusted sources) has become a daily event. Modern operating systems load code for adding new functionalities; web browsers download plug-ins and applets; end-users download untrusted code for doing some useful tasks. Certification that the untrusted code respects the confidentiality of data it manipulates is essential in these situations. Thus it is necessary to analyze how information flows within that program.

This thesis presents an approach to enable end-users to determine whether untrusted mobile code will respect pre-specified confidentiality policies by statically analyzing the untrusted code for secure information flow. The approach is based on adapting of a well-known approach, proof-carrying code (PCC) to information flow security and basing the security policy of PCC on a security-type system, which enforces information flow policy, namely noninterference security policy in RISC-style

assembly programs. The untrusted code is then analyzed for secure information flow based on the idea of PCC. The proofs that untrusted code does not leak confidential information are generated by the *code producer* and checked by the *code consumer*. If the proofs are valid, then the end-users (*code consumer*) can install and execute the untrusted mobile code safely.

The proposed approach benefits from distinctive features that make it a very appropriate for security checking. First, it operates directly on object code produced by general-purpose off-the-shelf compilers. Second, it exploits the benefits that both type systems and proof-carrying code approaches offer and combines their strengths. Type systems provide an appealing option for implementing security policies, and thus represent a natural enabling technology of proof-carrying code. Meanwhile, proof-carrying code is an efficient approach for assembly code verification. Third, the explicit machine-checkable proofs serve as a certificate to distrustful users and give them more confidence in the security approach.

The proposed security approach represents one point in the design space for mobile code security systems; it is well suited to typical Internet users. It enforces information flow policy with low preparation cost on the part of the *code producer* and no runtime overhead cost on the part of the *code consumer*. The security approach provides end-users with an adequate assurance of protecting the confidentiality of their confidential data.

Abstrak tesis yang dikemukakan kepada Senat Universiti Putra Malaysia sebagai memenuhi keperluan untuk Ijazah Doktor Falsafah

KOD MEMBAWA-BUKTI UNTUK MENENTUSAH KERAHSIAAN KOD MOBIL MELALUI ANALISIS ALIRAN MAKLUMAT

Oleh

ABDULRAHMAN AHMAD ABDU MUTHANA

November 2008

Pengerusi: Profesor Madya Abdul Azim Abdul Ghani, Ph.D.

Fakulti: Sains Komputer dan Teknologi Maklumat

Pertumbuhan kebergantungan masyarakat dan ekonomi ke atas sistem maklumat terangkai menyebabkan ianya penting untuk mengawal data rahsia daripada kebocoran oleh kod hasad. Memuat turun dan melaksanakan kod (mungkin daripada sumber tidak boleh dipercayai) telah menjadi amalan harian. Sistem pengoperasian moden memuat kod untuk menambah fungsian baharu; pelayar web memuat turun *plug-in* dan *applets*; pengguna akhir memuat turun kod tidak boleh dipercayai untuk melakukan beberapa tugasan penting. Pensijilan yang kod tidak boleh dipercayai menghormati kerahsiaan data yang dimanipulasi adalah penting dalam situasi begini. Oleh itu adalah perlu untuk menganalisa bagaimana maklumat mengalir dalam program tersebut.

Tesis ini mempersembahkan satu pendekatan yang membenarkan pengguna akhir menentukan sama ada kod mobil yang tidak boleh dipercayai akan menghormati polisi prapenentu kerahsiaan melalui penganalisisan secara statik kod yang tidak boleh dipercayai untuk aliran maklumat yang selamat. Pendekatan ini berdasarkan penyesuaian pendekatan yang terkenal, *kod membawa-bukti* (PCC) ke keselamatan

aliran maklumat dan mendasarkan polisi keselamatan PCC ke atas sistem keselamatan-jenis yang menguatkuasa polisi aliran maklumat, khususnya polisi keselamatan tidak campur tangan dalam program himpunan stail RISC. Kod yang tidak boleh dipercayai kemudian dianalisis untuk keselamatan aliran maklumat berdasarkan ide PCC. Bukti bahawa kod yang tidak boleh dipercayai tidak membocorkan maklumat rahsia dijana dan diperiksa. Jika bukti adalah sah, maka pengguna akhir boleh memasang dan melaksana kod mobil yang tidak dipercayai secara selamat.

Cadangan pendekatan ini mendapat manfaat daripada fitur tersendiri yang menjadikannya sangat sesuai untuk pemeriksaan keselamatan. Pertama, ia beroperasi secara terus ke atas kod objek terhasil melalui pengkompil *off-the-shelf* kegunaan umum. Kedua, ia mengeksploitasikan manfaat yang ditawarkan oleh kedua-dua pendekatan sistem jenis dan kod membawa-bukti dan menggabungkan kekuatan mereka. Sistem jenis menyediakan suatu opsyen yang menarik untuk melaksana polisi keselamatan, dan dengan itu mewakili teknologi kod membawa-bukti terboleh secara semula jadi. Sementara itu kod membawa-bukti adalah suatu teknik yang efisyen untuk penentusah kod himpunan. Ketiga, bukti semakan mesin yang eksplisit digunakan sebagai sijil kepada pengguna yang dicuragai dan memberi mereka keyakinan yang lebih dalam pendekatan keselamatan.

Pendekatan cadangan keselamatan ini mewakili satu titik dalam ruang reka bentuk sistem keselamatan kod mobil; ianya sangat sesuai untuk pengguna tipikal Internet. Ia menguatkuasakan polisi aliran maklumat dengan kos penyediaan rendah ke atas penghasil kod dan tiada kos overhed masa larian ke atas pengguna kod. Pendekatan

keselamatan menyediakan pengguna akhir dengan jaminan kukuh pengawalan kerahsiaan data sulit mereka.

ACKNOWLEDGEMENTS

In the name of ALLAH, the Beneficent, the Compassionate and who giving me strength, patience, and motivation to complete this research work. I would like to take this opportunity to record my gratitude towards the great peoples who they were an important support during the phases of this research; particularly those who help me during the time I was doing my Ph.D. research. My deepest appreciation and gratitude go to the research committee leads by Associate Prof. Dr. Abdul Azim Abdul Ghani, who has always take time to listen to my ideas, and he has patiently answered my questions, invaluable guidance, fruitful discussion, patience and continued encouragement supply me at every stage of this work and who always provides the gold recommendations and suggestions to my inquiries tranquilly and accurately. Also I would like to introduce my great thanks to all the member of my Ph.D. supervision committee; Associate Prof. Dr. Ramlan Mahmod for his support, attentions during my research work and the guidance in each discussion during all steps of this work and Associate Prof. Mohd. Hasan Selamat for his virtuous guidance, encouragement and help during the time of doing the research.

Great thanks to My country, Yemen. Great thanks to Malaysia: the country and the people. I would like to express my deep thanks to Malaysian government that gave me this opportunity to study PhD in Malaysia; its support was of a great significance. Finally, I am so grateful to Dr. Gaber Assanabani who gave me his full support and encouragement. Dr. Gaber is one of few people who branded in my memory.

ABDULRAHMAN AHMAD ABDU MUTHANA
November 2008

I certify that an Examination Committee has met on 12 November 2008 to conduct the final examination of Abdulrahman Ahmad Abdu Muthana on his Doctor of Philosophy thesis entitled "Proof-Carrying Code for Verifying Confidentiality of Mobile Code through Secure Information Flow Analysis " in accordance with Universiti Pertanian Malaysia (Higher Degree) Act 1980 and Universiti Pertanian Malaysia (Higher Degree) Regulations 1981. The Committee recommends that the candidate be awarded the relevant degree. Members of the Examination Committee are as follows:

Associate Professor
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Chairman)

Associate Professor
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Internal Examiner)

Associate Professor
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Internal Examiner)

Professor
(External Examiner)

_____

HASANAH MOHD. GHAZALI, PhD
Professor / Deputy Dean
School of Graduate Studies
Universiti Putra Malaysia

Date:

This thesis was submitted to the Senate of Universiti Putra Malaysia and has been accepted as fulfillment of the requirement for the degree of Doctor of Philosophy. The members of the Supervisory Committee were as follows:

Abdul Azim Abdul Ghani, PhD
Associate Professor
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Chairman)

Ramlan Mahmod, PhD
Associate Professor
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Member)

Hj. Mohd. Hasan Selamat
Associate Professor
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Member)

_____

HASANAH MOHD. GHAZALI, PhD
Professor and Dean
School of Graduate Studies
Universiti Putra Malaysia

Date: 12 February 2009

DECLARATION

I declare that the thesis is my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously and is not concurrently submitted for any other degree at Universiti Putra Malaysia or at any other institution.

_____

ABDULRAHMAN AHMAD ABDU MUTHANA

Date :01 August 2008

TABLE OF CONTENTS

		Page
DEDICATION		ii
ABSTRACT		iii
ABSTRAK		v
ACKNOWLEDGEMENTS		viii
APPROVAL		ix
DECLARATION		xi
LIST OF TABLES		xv
LIST OF FIGURES		xvi
LIST OF APPENDICES		xix
LIST OF ABBREVIATIONS		xx
CHAPTER		
1	**INTRODUCTION**	
1.1	Background	1
1.2	Problem Statement	3
1.3	Objectives of the Research	5
1.4	Scope of the Research	6
1.5	Importance of the Research	7
1.6	Structure of Thesis	8
2	**LITERATURE REVIEW**	
2.1	Introduction	11
2.2	Code Mobility	11
2.3	Mobile Code Security	14
	2.3.1 Security Components of a Computing System	15
	2.3.2 Mobile Code Security Threats	16
2.4	Approaches to Mobile Code Security	17
	2.4.1 Traditional Approaches	17
	2.4.2 Language-based Approaches	20
2.5	Background: Information Flow Security	23
	2.5.1 Information Flow Policies and Security Lattices	25
	2.5.2 Noninterference	28
2.6	Related Work	35
	2.6.1 Static Analysis Approaches	37
	2.6.2 Dynamic Analysis Approaches	53
	2.6.3 Concluding Remarks	54
2.7	Summary	56
3	**RESEARCH METHODOLOGY**	
3.1	Introduction	58
3.2	Methodology to Develop an Approach for Information Flow Analysis	59
	3.2.1 Defining the Research Problem	61
	3.2.2 Exploring Language-based Approaches to Mobile Code Security	62

3.2.3 Adapting Standard Proof-Carrying Code to Information Flow Security 64
3.2.4 Validating the Proposed Approach 67
3.2.5 Prototyping Implementation 69
3.3 Summary 70

4 PCC-SIF: AN APPROACH FOR SECURE INFORMATION FLOW ANALYSIS

4.1 Introduction 71
4.2 Standard Proof-Carrying Code Approach 72
4.2.1 A Certifying Compiler 74
4.2.2 Safety Policy 75
4.2.3 Theorem Prover 77
4.2.4 Proof Checker 77
4.3 Proof-Carrying Code for Secure Information Flow Analysis (PCC-SIF) 78
4.3.1 Assembly Language 81
4.3.2 Information Flow Policy 85
4.3.3 Proofs Generation and Validation 144
4.4 Steps of Secure Information Flow Analysis 153
4.4.1 Defining Information Flow policy 156
4.4.2 Generating the annotated code 156
4.4.3 Recovering High-Level Control Flow Structures 158
4.4.4 Generating the Verification Condition 159
4.4.5 Proving the Verification Condition 159
4.4.6 Checking the Proof 161
4.5 Soundness of PCC-SIF Approach 161
4.5.1 Soundness of Information Flow Policy 166
4.5.2 Adequacy of the LF Representation of Proofs 178
4.6 Prototype Implementation 179
4.6.1 Translation from SAL to SPARC 183
4.6.2 The SPARC Prototype 185
4.6.3 Case Studies 186
4.7 Summary 190

5 RESULTS AND DISCUSSION

5.1 Introduction 191
5.2 Results 192
5.3 Discussion 194
5.3.1 Perspective 196
5.3.2 Implications 197
5.3.3 Applications 198
5.3.4 PCC-SIF vs. Standard PCC 201
5.3.5 PCC-SIF vs. Related Work 203
5.3.6 Issues with Information Flow Checking of Assembly Programs 217
5.4 Summary 217

6 CONCLUSIONS AND FUTURE WORKS

6.1 Conclusions 218

6.2 Contributions 220
6.3 Future Works 223
6.3.1 Hybrid Analysis 223
6.3.2 Flow-sensitive Type System 224
6.3.3 Relaxing Noninterference 225

REFERENCES 227
APPENDICES 236
BIODATA OF STUDENT 309
LIST OF PUBLICATIONS 310

LIST OF TABLES

Table		page
2.1	Common mobile code paradigms	12
3.2	Software Requirements for Prototype Implementation	70

LIST OF FIGURES

Figure		Page
2.1	Linear ordered Lattice	26
2.2	Lattice of subsets of X = $\{\ell_1, \ell_2, \ell_3\}$	27
2.3	A diagram showing the position of our research work with respect to computer security area	38
4.1	Standard Proof-Carrying Code Framework	74
4.2	The structure of PCC-SIF Framework	79
4.3	SAL Instruction Set	82
4.4	Operational Semantics of SAL Language (part 1). Standard Assembly Instructions	84
4.5	Operational Semantics of SAL Language (part 2). Abstract file read and write and message sending instructions	85
4.6	A grammar of type expressions	88
4.7	Typing rules of information flow analysis of function F (part 1)	91
4.8	Information flow through aliasing	93
4.9	Mutual recursion functions	99
4.10	A recursive function rec and its corresponding SAL program	100
4.11	Typing rules of information flow analysis of function F (part 2)	102
4.12	Security automaton: “No messages send after reading a file”	105
4.13	Syntax of logic specified by code consumer	108
4.14	Axioms and proof rules of logic specified by the code consumer	111
4.15	Algorithm to computing all control dependence regions in control flow graph of each function.	118
4.16	Algorithm to constructing control flow graph	122
4.17	Algorithm to computing semi postdominators of each node in a control flow graph	123
4.18	Algorithm to computing all postdominators of each node in control flow graph	124
4.19	Algorithm to computing all control dependence regions for each conditional node in control flow graph.	125
4.20	Stack management in SAL	131
4.21	Definition of verification condition generator for function F (part 1). Standard assembly instructions	134

4.22	Definition of verification condition generator for function F (part 2). Abstract file read and write and send data instructions	135
4.23	Definition of functions *CopyIn* and C*opyOut*	138
4.24	The syntax of Edinburgh LF	146
4.25	Fragment of LF signature corresponding to logic in Figure 4.11	147
4.26	LF signature for information flow policy proof rules in Figure 4.12	149
4.27	Specification of function main	156
4.28	Sample source code for secure information-flow program	157
4.29	The assembly form of the program in Figure 4.26	157
4.30	Control Flow Graph of an assembly code in Figure 4.27 showing Control Dependence Regions	158
4.31	The verification condition for the code in Figure 4.27	160
4.32	Fragment of the proof of the verification condition in Figure 4.29	160
4.33	Abstract Machine for Soundness Proof of the SAL programs	163
4.34	Flowchart of PCC-SIF Prototype	182
4.35	SPARC register windows for three successive functions	184
4.36	The translation table from SAL to SPARC architecture	185
4.37	Characteristics of test cases and number of security conditions	188
4.38	Performance Time	189
4.39	Sizes of assembly code, verification conditions, and proofs (in bytes)	190
5.1	High-level structure of PCC-SIF Architecture	197
C.1	A control flow graph	265
C.2	Control Dependences of control flow graph in Figure C.1	265
C.3	A simple program (a) and its control flow graph (b)	266
C.4	Control Dependences of program in Figure C.3	266
C.5	Program segment	267
C.6	Control flow graph (a) and control dependence subgraph (b) of program segment in Figure C.5	267
C.7	Control Dependences of program of program segment in Figure C.5	268
F.1	Program10.c source code file	282
F.2	SPARC assembly code file of program10.c in Figure F.1	290

F.3	User interface of PCC-SIF SPARC prototype showing the verification of the SPARC assembly code in Figure F.2	291
F.4	Verification condition file of SPARC assembly program in Figure F.2	298
F.5	Security proof of verification conditions in Figure F.4	308

LIST OF APPENDICES

Appendix		Page
A	Soundness of Information Flow Type System	236
B	Soundness of the Axiomatic System	259
C	Evaluation of Control Dependence Region Calculator Algorithm	263
D	Soundness of Verification Condition Generator	269
E	Twelf Signature for Secure Information Flow Analysis	276
F	Sample of Test Case Programs and Their Proofs	279

LIST OF ABBREVIATIONS

AST	Abstract Semantics Tool
CDR	Control Dependence Region
CLI	Common Language Interface
CPS	Continuation Passing Style
DL	Dynamic Logic
ECC	Efficient Code Certification
EM	Execution Monitor
IFD	Immediate Forward Dominator
JVM	Java Virtual Machine
JVML	Java Virtual Machine Language
LF	Logical Framework
LOC	Lines of code
PCC	Proof-Carrying Code
PCC-SIF	Proof-Carrying Code for Secure Information Flow Analysis
PSP	Perfect Security Property
RIP	Region Inclusion Property
RISC	Reduced Instructions Set Computing
SFI	Software Fault Isolation
SPARC	Scalable Processor ARChitecture
TAL	Typed Assembly Language
TCB	Trusted Computing Base
TIL	Typed Intermediate Languages
VC	Verification Condition
VCG	Verification Condition Generator

CHAPTER 1

INTRODUCTION

1.1 Background

The growing dependence of our society and economy on networked information systems makes the organizations as well as individuals potential targets to computer security attacks. Moreover, the number of sources and targets of these attacks are growing fast day after day. Not only has the advancement of Internet complicated the task of protection mechanisms against computer security attacks but also made performing such attacks much easier than ever. Performing computer security attacks today does not need one to be a security expert because one can simply exploit existing tools and software available on the Internet (Sabelfeld, 2001). One of the computer security attacks that target organizations as well as users today is the attack of confidentiality, in which the malicious programs attempt to leak confidential data to intended parties. For the remainder of this thesis, when the term security is used, it means confidentiality.

Recent years have witnessed a significant growth of interest in protecting confidentiality of information of organizations as well as individuals more particularly in presence of mobile code. Standard security mechanisms such as access control mechanisms, cryptography, antivirus and digital signature fail to provide a complete assurance of protecting confidentiality of information, and thus do not provide end-to-end security. Though access control mechanism is the normal

way to protect confidentiality of information, it is simple and in some cases is of limited use and tends to be restrictive. Cryptography provides an assurance about the source of the downloaded code and that it has not been compromised during transition but cannot ensure that the downloaded code has secure information flow. Antivirus tools searches for viruses signatures and do not concern confidentiality problem (Sabelfeld & Myers, 2003).

The following motivating example demonstrates clearly the pitfall of access control mechanisms and shows their inadequacy to protect the confidentiality of information: assume that a piece of code has been downloaded off the network to perform some useful tasks. If this code has an access right to some user's confidential data and needs to communicate over the Internet connection, it may leak confidential information through the Internet connection. In order to protect the confidentiality of information, the access control mechanism will prevent the code from accessing the Internet or private data or prevent it from accessing both. The access control mechanisms are of limited use here and cannot prevent the program from accessing the private data because the program uses the user's access rights. Furthermore, this strategy is not suitable because it may prevent useful programs from doing their tasks and sacrifices some richness of the web. The crux of the problem is not in disabling the run of mobile code but how one can enjoy the functionalities provided by mobile code while protecting data confidentiality.

Language-based approaches to security are promising in protecting data confidentiality (Sabelfeld & Myers, 2003). As they can access the program's code, language-based security approaches can enforce fine-grained security policies and

express easily the behavior of the programs, and thus succeed where traditional access control mechanisms fail short. The class of language-based security approaches that can protect data confidentiality is called language-based approaches to information-flow security. These security approaches protects data confidentiality through analyzing the target programs for secure information flow. The concept of secure information flow is typically formalized in terms of what is known as noninterference (Goguen & Meseguer, 1982). Noninterference states that confidential data may not interfere with (affect) public data.

Unfortunately, much of works on language-based information-flow security have been devoted to high-level languages with relatively less interest given to assembly languages (Sabelfeld & Myers, 2003). High-level languages approaches suffer from a potential flaw—the use of the compiler to check information flow. The compiler is a big, complex, and cannot be assured to be free of bugs. To avoid this potential weakness, it is required to check the code produced by compiler directly. Moreover, much of the code is distributed in the form of executables files and convincing the code suppliers to provide a code in a form amenable to high-level security checking as source code is not successful.

1.2 Problem Statement

Protecting the confidentiality of information in the presence of mobile code is an increasing important problem (Sabelfeld & Myers, 2003). Mobile code refers to that sort of programs that are moved from one place to another over a network before being executed. Mobile code is normally shipped in low-level form (e.g., Windows executables), and hence it is appropriate to perform the security checking at assembly

level. Examples of mobile code are ActiveX, VBScript, and JavaScript. Mobile code is downloaded from the Internet (often from untrusted or partially trusted sources) adding new functionalities to modern computing systems or performing some useful tasks to end-users. However, the useful and powerful features that mobile code is offering come at a high price. Mobile code may leak sensitive information, and thus the computing systems that incorporate mobile code must protect their confidential information from being leaked by mobile code.

From the viewpoint of our research work, existing approaches that attempt to protect the confidentiality of information by analyzing mobile programs fall into two main groups. Research works that deal with RISC architecture (e.g., Yu & Islam, 2005; Medel et al., 2005; Bonelli et al., 2004) and those that deal with Java bytecode (e.g., Barthe et al., 2006; Barthe & Rezk 2005; De Francesco & Martini, 2007). The approaches belong to the first group assume that mobile programs being checked are generated by certifying compilers. As a result these approaches are difficult to use for checking mobile programs generated by general-purpose off-the-shelf compilers, and thus prevents end-users from benefiting from such approaches to check a wide range of existing mobile programs. On the other side, the approaches that deal with Java bytecode are not suitable for checking programs written in high-level languages other than Java. Furthermore, none of these approaches generates explicit proofs (certificates) for the programs acceptable by them, and thus they do not explain to users why these programs are secure. The explicit proof, however, is a convincing way that provides a confidence that a program execution will not leak confidential information. The security proof serves as evidence to the users that a given program is truly secure and should be allowed to execute.