

DECLARATION OF GERARD P. GRENIER

I, Gerard P. Grenier, am over twenty-one (21) years of age. I have never been convicted of a felony, and I am fully competent to make this declaration. I declare the following to be true to the best of my knowledge, information and belief:

1. I am Senior Director of Publishing Technologies of The Institute of Electrical and Electronics Engineers, Incorporated (“IEEE”).
2. IEEE is a neutral third party in this dispute.
3. Neither I nor IEEE itself is being compensated for this declaration.
4. Among my responsibilities as Senior Director of Publishing Technologies, I act as a custodian of certain records for IEEE.
5. I make this declaration based on my personal knowledge and information contained in the business records of IEEE.
6. As part of its ordinary course of business, IEEE publishes and makes available technical articles and standards. These publications are made available for public download through the IEEE digital library, IEEE Xplore.
7. It is the regular practice of IEEE to publish articles and other writings including article abstracts and make them available to the public through IEEE Xplore. IEEE maintains copies of publications in the ordinary course of its regularly conducted activities.
8. The article below has been attached as Exhibit A to this declaration:

A.	A. Chander; J.C. Mitchell; I. Shin “Mobile code security by Java bytecode instrumentation”, DARPA Information Survivability Conference and Exposition II, June 12-14, 2001.
----	---

9. I obtained a copy of Exhibit A through IEEE Xplore, where it is maintained in the ordinary course of IEEE’s business. Exhibit A is a true and correct copy of the Exhibit, as it existed on or about August 2, 2018.
10. The article abstract from IEEE Xplore shows the date of publication. IEEE Xplore populates this information using the metadata associated with the publication.
11. A. Chander; J.C. Mitchell; I. Shin “Mobile code security by Java bytecode instrumentation”, was published in the DARPA Information Survivability Conference

and Exposition II. The DARPA Information Survivability Conference and Exposition II was held from June 12-14, 2001. Copies of the conference proceedings were made available no later than the last day of the conference. The article is currently available for public download from the IEEE digital library, IEEE Xplore.

12. I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true, and further that these statements were made with the knowledge that willful false statements and the like are punishable by fine or imprisonment, or both, under 18 U.S.C. § 1001.

I declare under penalty of perjury that the foregoing statements are true and correct.

Executed on: 7-Aug-2018

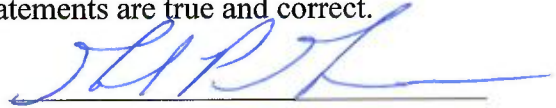


EXHIBIT A

Access provided by:
IEEE Staff
Sign Out

[Browse](#)

[My Settings](#)

[Get Help](#)

Browse Conferences > Proceedings DARPA Information...

[Back to Results](#)

Mobile code security by Java bytecode instrumentation

<< Results

[View Document](#)

11
Paper
Citations

18
Patent
Citations

164
Full
Text Views

Related Articles

Analysis of potential deadlock in Java multithreaded object-oriented programs

A Teaching Path for Java Object Oriented Programming

JR: flexible distributed programming in an extended Java

[View All](#) [View All](#)

3
Author(s) | A. Chander J.C. Mitchell I. Shin | A. Chander ; J.C. Mitchell ; I. Shin

[View All Authors](#)

Abstract

[Authors](#)

[Figures](#)

[References](#)

[Citations](#)

[Keywords](#)

Abstract: Mobile code provides significant opportunities and risks. Java bytecode is used to provide executable content to Web pages and is the basis for dynamic service configuration in the Jini framework. While the Java Virtual Machine includes a bytecode verifier that checks bytecode programs before execution, and a bytecode interpreter that performs run-time tests, mobile code may still behave in ways that are harmful to users. We present techniques that insert run-time tests into Java code, illustrat... [View more](#)

Metadata

Abstract:

Mobile code provides significant opportunities and risks. Java bytecode is used to provide executable content to Web pages and is the basis for dynamic service configuration in the Jini framework. While the Java Virtual Machine includes a bytecode verifier that checks bytecode programs before execution, and a bytecode interpreter that performs run-time tests, mobile code may still behave in ways that are harmful to users. We present techniques that insert run-time tests into Java code, illustrating them for Java applets and Jini proxy bytecodes. These techniques may be used to contain mobile code behavior or, potentially, insert code that is appropriate to profiling or other monitoring efforts. The main techniques are class modification, involving subclassing non-final classes, and method-level modifications that may be used when control over objects from final classes is desired.

Published in: Proceedings DARPA Information Survivability Conference and Exposition II. DISCEX'01

Date of Conference: 12-14 June 2001

INSPEC Accession Number: 6991891

Date Added to IEEE Xplore: 07 August 2002

DOI: 10.1109/DISCEX.2001.932157

Print ISBN: 0-7695-1212-7

Publisher: IEEE

Conference Location: Anaheim, CA, USA, USA

[Contents](#) [Contents](#)



[Authors](#)[References](#)[Citations](#)[Keywords](#)[Related Articles](#)[Footnotes](#)[Back to Top](#)[Download PDF](#)[Download PDF](#)[Download Citation](#)[Download Citation](#)[View References](#)[View References](#)[Email](#)[Email](#)[Print](#)[Print](#)[Request Permissions](#)[Request Permissions](#)[Export to Collabratec](#)[Export to Collabratec](#)[Alerts](#)[Alerts](#)

SECTION 1. Introduction

Since its early beginnings in the Green project, the Java language [26] has come a long way in its applicability and prevalence. While its initial adoption was fuelled by the ability to add “active content” to web pages, Java has also become a predominant system and application development language, providing useful capabilities over and above the language features through an extensive set of application programming interfaces (APIs). The APIs' simplify programming by providing a rich set of domain-dependent libraries, as well as enabling new programmatic and computational paradigms. As an example, the Java Cryptography API makes it possible for applications to easily implement security protocols for their own needs, while the Jini API provides a specification for Java bytecode based distributed programming. One of the keys to Java's success and appeal is its platform independence, achieved by compilation of source code to a common intermediate format, namely Java Virtual Machine (JVM) bytecode, which can then be interpreted by various platforms. The ability to transport byte code between JVMs is most commonly encountered while browsing the net, and Java's platform independence ensures a client-independent experience.

Although previous language implementations, such as Pascal and Smalltalk systems, have used intermediate byte code, the use of bytecode as a *medium of exchange* places Java bytecode in a new light. A networked computer can import and execute Java bytecode in ways that are invisible or partly invisible to the user. For example, a user (or his browser) may execute a Java applet embedded within a page as part of the HTTP protocol, or a client may execute a lookup service proxy as it prepares to join a Jini community. To protect against execution of erroneous or intentionally malicious code, the JVM verifies bytecode properties before execution and performs additional checks at run time. However, these checks only enforce some type correctness conditions and basic resource access control. For example, these tests will not protect against large classes of undesirable run-time behavior, including denial-of-service, compromise of integrity, and loss of sensitive information from password or credit card information files, say. The introduction

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts



Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research



With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips



Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.