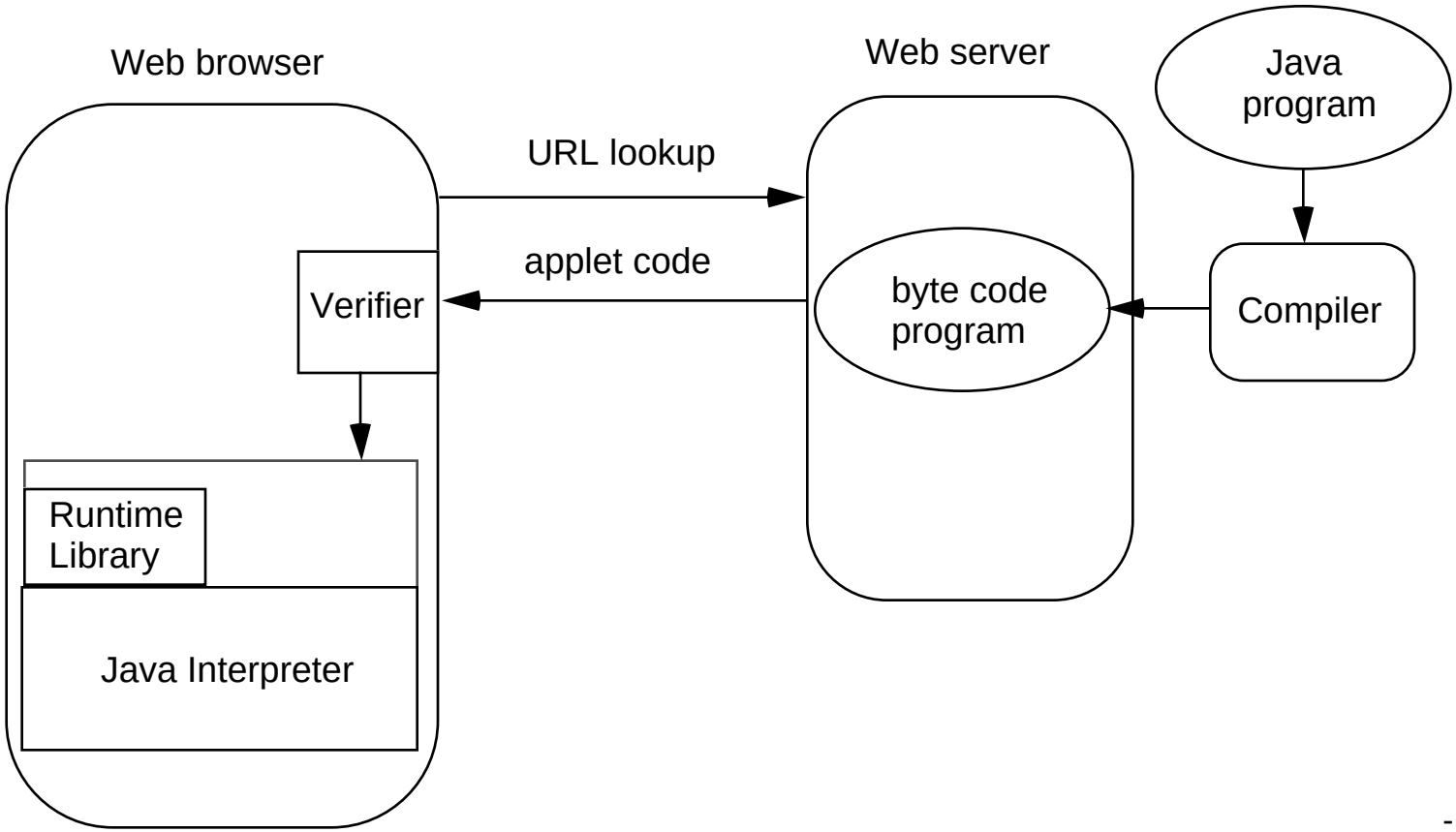
# Java Security: From HotJava to Netscape and Beyond

Drew Dean, Ed Felten, Dan Wallach

Safe Internet Programming Group

Princeton University

# The Java Model



# Security in Java

relies on type-safe language

code loaded from local filesystem trusted

applets cannot directly make system calls

- SecurityManager module approves dangerous operations

applets forbidden to

- access the filesystem
- open sockets, except back to their home
- interfere with other applets
- learn about the local environment

# History of Java Security Problems

Nov 95: many flaws found in HotJava alpha release

Feb 96: DNS flaw in Java 1.0

Mar 96: pathname flaw in Java 1.0 [Hopwood]

Mar 96: verifier and ClassLoader flaw in Java 1.0.1

several other problems: details in paper

# HotJava (alpha) Problems

covert channels: URLs, DNS

applets can learn private information

- mailcap files
- environment variables

denial/degradation of service

man-in-the-middle attack

- set HTTP proxy server

Sun's response: "fixed in the next release"

- covert channels not fixed
- denial of service considered unimportant

# Java 1.0 Problems

applets can interfere with each other

denial/degradation of service attacks

unchecked sprintf in disassembler

bytecode verifier bugs

# More Java 1.0 Problems

## DNS problem

- Java 1.0 trusted remote DNS servers to tell the truth
- attacker could
  - set up a DNS server that lies
  - write applet that conspires with DNS server to connect to anywhere
  - attack any site on the net
    - possibly behind a firewall

## pathname problem [Hopwood]

- applet can load ANY file on the system as trusted code
- attacker could load code (as data) into browser's cache first

# Java 1.0.1 Problem

applet can run arbitrary machine code

exploits two problems

- Verifier allows constructors that catch exceptions thrown by superclass constructor
- no constraints on what ClassLoaders can do

how the attack works

- make a ClassLoader by exploiting Verifier problem
- break Java's type system by exploiting ClassLoader
  - can treat any type as any other type
  - can access interpreter's internal data structures
- write machine code into memory and jump to it

# Deeper Problems

no formal security policy

no logging

no identified Trusted Computing Base

"single line of defense"

# Java Language Problems

Java is great for writing programs, but not great for security.

example: difficult to restrict access to locks

example: name resolution process vulnerable to attack

# Other Problems

language vs. bytecode differences led to many bugs

- no significant advantage to using bytecode

need stronger module system

- nested modules
- interfaces that export more than one type
- ability to hide some methods and variables of a class
- automatic security checks when crossing module boundary

no proof of type soundness

# Managing Flexibility

future releases will allow more flexibility to applets

- file access
- network access
- device access

access control decisions up to users

how will ordinary users manage?

- user interface is key

digitally signed applets add new complexities

# Mitigating Denial of Service Attacks

denial of service attacks make "net vandalism" possible

to reduce their scope, borrow ideas from operating systems

- protect access to locks and system objects by default
- do accounting to detect wasteful applets
- let user display list of running applets, and kill undesired ones

# Solving A New Problem

Java is the first Internet-friendly, language-based operating system.

Java is an excellent programming language, but not an excellent operating system.

Java's weaknesses don't make other plug-in mechanisms (ActiveX, JavaScript, VBScript, ...) secure.