

Mobile Code

CSH5 Chapter 17
 “Mobile Code”
 Robert Gezelter

1

Copyright © 2011 M. E. Kabay. All rights reserved.

Topics

- Introduction to Mobile Code
- Mobile Code from the Web
- Motivations and Goals
- Design and Implementation Errors



CSH5 Chapter 17
 Mobile Code

2

Copyright © 2011 M. E. Kabay. All rights reserved.

Mobile Code Defined

- Instructions delivered to remote computer from outside an *enclave*
- *Enclave* is system under unitary control by single authority
- Dynamic execution (execution on demand)
- Fundamental problems
 - ❑ Mobile code may perform unauthorized functions
 - ❑ Growing spectrum of devices using mobile code
 - ✓ PDAs
 - ✓ Mobile phones



3

Copyright © 2011 M. E. Kabay. All rights reserved.

Mobile Code from the WWW

- Definition
 - ❑ Executable code delivered by Web server
 - ❑ Or by e-mail
 - ❑ For execution on client computer
 - ❑ Not including HTML or XML
- Examples of problematic content
 - ❑ HTML-enabled e-mail with embedded code
 - ❑ Pop-ups in browsers
 - ✓ May access unexpected Web pages
 - ✓ Julie Amero, CT teacher, convicted of using classroom computer for inappropriate content due to popups



4

Copyright © 2011 M. E. Kabay. All rights reserved.

Effects of Mobile Code

- System / application crashes
 - ❑ Obvious effects include
 - ✓ Denial of service
 - ✓ Corruption (integrity problems)
- Covert effects more dangerous
 - ❑ Access to e-mail addresses → spam
 - ❑ Keyloggers
 - ❑ Rootkits
- Hephtrati case in Israel (2005) showed how mobile code could be used for industrial espionage
 - ❑ Varda Raziel-Jacont & Amnon Jacont's MS for "L is for Lies" appeared on Internet sites
 - ❑ Former son-in-law Michael Hephtrati responsible using implanted mobile code



5

Copyright © 2011 M. E. Kabay. All rights reserved.

Motivations & Goals

- Shift in motivations
 - ❑ Pranks → vengeful → vindictive → criminal
- Goals differ
 - ❑ Amusement
 - ❑ Blackmail
 - ❑ Corporate espionage
 - ❑ Financial fraud/theft
- Misappropriation of computer resources
 - ❑ Creation of botnets
 - ❑ Applications to DDoS & spam
 - ❑ Involvement in information warfare



6

Copyright © 2011 M. E. Kabay. All rights reserved.

Design & Implementation Errors



- Range of errors
 - ❑ Simple design/coding errors cause mistakes in function
 - ✓ Usually predictable
 - ✓ Often noticeable
 - ❑ But mistakes in security architecture particularly serious
- Security architectures for mobile code may be flawed in
 - ❑ Creation of sandbox
 - ❑ Method of code authentication



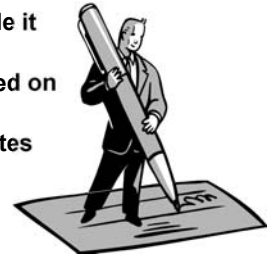
7

Copyright © 2011 M. E. Kabay. All rights reserved.

Signed Code



- Principle (belief):
 - ❑ If you know who wrote code it must be OK
 - ❑ Use digital signatures based on Public Key Cryptosystem
 - ❑ Apply ANSI X.509 Certificates
- But signing *bad* code doesn't remove the flaws
- Topics discussed below:
 - ❑ Authenticode
 - ❑ Limitations of Signed Code
 - ❑ Problems with ActiveX Security Model
 - ❑ Case Studies



8

Copyright © 2011 M. E. Kabay. All rights reserved.

Authenticode



- Microsoft method
 - ❑ Developers obtain digital certificate from Microsoft Certificate Authority (CA)
 - ❑ Sign their code with their private key
 - ❑ Users (systems) check validity of code at execution time using public key
- Components
 - ❑ PKI, X.509, CA
 - ❑ Control over private keys used to sign code
 - ✓ Known as the Software Publishing Certificate
 - ❑ Valid method for verifying digital signatures
- No protection against bad code



9

Copyright © 2011 M. E. Kabay. All rights reserved.

Limitations of Signed Code



- All-or-nothing approach to trust
 - ❑ Signed items assumed to be perfect
 - ❑ No concept of partial trust
 - ❑ "Digital signature does not ... provide any guarantee of benevolence or competence." – CERT/CC®
- Organizations signing their code must strictly control access to the signing key
 - ❑ But widespread practice in software development shops tolerates shared accounts and passwords
 - ❑ Serious question about value of signing code



10

Copyright © 2011 M. E. Kabay. All rights reserved.

Problems with ActiveX Security Model



- Importing and installing controls
 - ❑ Signing does not guarantee safety
- Running controls
 - ❑ ActiveX control has no limitations on actions
 - ❑ Runs with same privileges as user
 - ❑ Typically users run as *root* on their Windows PCs
 - ❑ No basis for deciding whether particular control is safe or not in specific context
- Scripting concerns
 - ❑ Individual developers have no systematic method for evaluating safety of their code
 - ❑ No equivalent to a *sandbox* for testing



Mr. Bunny's Guide to ActiveX
by
Carlton Egremont III

11

Copyright © 2011 M. E. Kabay. All rights reserved.

Case Studies (1)



- Internet Exploder (1996)
 - ❑ Fred McLain wrote demonstrate
 - ❑ ActiveX control to illustrate excessive control of systems
 - ❑ Shut down user's computer (with permission of user!)
- Chaos Computer Club Demo (1997)
 - ❑ ActiveX control subverted Quicken accounting package
 - ❑ Made Quicken create transfer order for money
 - ❑ Filmed demonstration for German television



12

Copyright © 2011 M. E. Kabay. All rights reserved.

Case Studies (2)

- VeriSign Issues Certificates to Imposters (2001)
 - ❑ Class 3 Digital Certificates for signing ActiveX controls
 - ❑ Issued to someone impersonating MS employee
 - ❑ Allowed signing code as if it came from MS
- Problems
 - ❑ No Certificate Revocation List (CRL)
 - ❑ Would need to verify date of every MS certificate to identify fraudulent issued ones
- Caution to avoid overreacting
 - ❑ 1st error discovered in >500,000 issued certificates



13

Copyright © 2011 M. E. Kabay. All rights reserved.

Restricted Operating Environments

- At simplest level, users should not execute code that affects entire system – restricted to their own processes
 - ❑ Process is unique instance of execution of particular code by specific user on particular machine at specific instant
- Concept of *privileges* determines what a process can accomplish
 - ❑ Supervisory or root privileges allow full access
- Restricted operating environment
 - ❑ Developed since earliest multi-user systems
 - ✓ MULTICS, OS/360, UNIX...
 - ❑ See CSH5 Chapter 24, Operating System Security
 - ❑ *Sandbox* is an example of restricted operating environment



14

Copyright © 2011 M. E. Kabay. All rights reserved.

Java

- Developed by Sun Microsystems
 - ❑ Platform independence
 - ❑ Typically used in Web browser
- Includes virtual machine (JVM)
 - ❑ Plus Java Run Time Environment
- Code known as *applets*
 - ❑ May be signed
 - ❑ Restricted access to system resources
 - ✓ Known as the Java sandbox
- But bugs have allowed Java applets to leave sandbox on occasion



15

Copyright © 2011 M. E. Kabay. All rights reserved.

Asymmetric & Transitive Trust

- *Asymmetry* in power can cause opportunity for mass infection
 - ❑ E.g., large customer can force small suppliers to conform to its standards
 - ❑ Force use of unsafe mobile code
 - ❑ Can resist damage by enforcing principle of *least privilege* in execution of all code
- *Transitive* trust results from assumption that trusted sites must have trustworthy code
 - ❑ Essential to enforce tight security on all mobile code regardless of source
 - ❑ ActiveX security model thus fundamentally flawed because it relies solely on transitive trust

16

Copyright © 2011 M. E. Kabay. All rights reserved.

Misappropriation & Subversion

- Mobile code targets have changed
 - ❑ From individual target machines
 - ❑ To entire populations of targets
- John Schiefer (2007) caught by Bot Roast II
 - ❑ FBI operation against botnet operators
 - ❑ Schiefer (“acidstorm”), 26 years old, pled guilty
 - ✓ 250,000 systems
 - Infected with spybots for capture of userID and passwords
 - Used to subvert PayPal & other accounts
 - ✓ 150,000 systems infected to support Dutch criminal Internet advertising company
 - ❑ Sentenced to 4 years in federal prison



17

Copyright © 2011 M. E. Kabay. All rights reserved.

Multidimensional Threat

- Signing code leaves other issues
 - ❑ Integrity of signing process
 - ❑ Integrity of the PKI
 - ❑ Safety or validity of code not addressed
- Individual controls or applets may function correctly BUT
 - ❑ Interactions that were not or could not be tested may cause failures
 - ❑ E.g., attempts to use same Windows registry key in conflicting ways
 - ❑ Complexity of operating environment may preclude provable safety



18

Copyright © 2011 M. E. Kabay. All rights reserved.

Client Responsibilities (1)

- Fundamental issue: Browser running mobile code may allow change to persistent state of operating system
- Use nonprivileged account wherever possible when browsing Web

Windows XP SP2 offers Data Execution Prevention™ (DEP) 

Monitors use of memory to restrict access to protected areas

Windows 7

Stronger partition between administrator privileges and normal user mode

AppLocker™ allows admin control over allowable execution of code



Client Responsibilities (2)

- Virtual Machine (VM) technology
 - Significant potential for security improvement
 - Can instantiate a desktop for specific purpose
 - Complete isolation from rest of system
- Expendable WWW browser
 - Run on VM desktop
 - Delete when finished
 - Bad effects of harmful code could be
 - Isolated
 - Evanescent



Server Responsibilities

- Avoid use of active code for trivial purposes
- Minimize use of ActiveX in favor of Java, JavaScript and other less dangerous tools
 - E.g., for shopping carts, changing appearance of screen
 - But updating Windows client will likely required ActiveX
- Apply standards of secure programming to all mobile code
 - See CSH5 Chapter 38, "Writing Secure Code"



Recommendations for Mobile Code Security

- Client systems
 - Bar acceptance of unsigned controls and applets
 - Restrict use of ActiveX
 - Restrict acceptance of pop-ups
- Developers
 - Follow good software engineering practices
 - Grant minimum necessary privileges & access
 - Use defensive programming
 - Limit privileged access
 - Ensure integrity of code-signing process
 - Protect signing keys against compromise



DISCUSSION