Client-Side Security: The Dangers of Active Content

Lecture Outline

- Threats of Active Content
- Plug-ins and Helper Applications
- Java Applets
- JavaScript

Threats of Active Content

- Browsers can download and execute software (JavaScript, applets, etc) without warning
  - a potential security threat since the software could be anything.
- The security attack could be deliberate or accidental.
- Accidental attacks are usually the result of programs with bugs
  - these “bugs" could have been a feature of the software originally, but later used as a security loophole by other parties.

Traditional Threats

- *Trojan Horses*
  - software which pretends to do something else other than what it actually does
- *Viruses*
  - software capable of replicating itself and inserting themselves into executable files or other protions of the hard disk

Traditional Threats (cont'd)

- *Rabbits*
  - software when executed, replicates many copies of itself to fill up memory, disk space and perhaps crash the system.
  - Do not inflicted themselves into another file as viruses do
- *Worms*
  - software which spreads from one machine to another exploiting loopholes in network security of target machines

Threats from Helper Applications and Plug-ins

- Since helper applications and plug-ins execute as if they were programs on your local machine, they can potentially be made do something dangerous.
- First of all, the helper apps or plug-in themselves might have malicious behaviour
  - Don't download unverified helper applications and plug-ins from random sites.
  - Always download from official sites.
  - Always check with up-to-date virus checker.

Threats from Helper Applications and Plug-ins (cont'd)

- Well-intentioned helper apps and plug-ins can also be used to attack a system
  - how potentially dangerous it is depends on how powerful the helper apps and plug-ins are
  - Eg. using command interpreter (like DOS command.com or UNIX shells) as helper application give files types which uses these applications all the powers of the OS to do malicious things – AVOID!

Threats from Helper Applications and Plug-ins (cont'd)

- Also very dangerous are helper applications which has the power to launch other applications
  - eg. MS-PowerPoint can start-up other applications from inside a presentation.

Java Security

- Java **applications** are considered trusted software
  - Since they execute as local installed copies, so have the rights and privileges of any other software)file I/O, make network connections, etc).
  - You installed it, you live with it!
- Java **applets**, on the other hand, are not considered trusted software
  - You don't know what applets embedded in a web page will do.

Java Security (cont'd)

- There are heavy security restrictions (called the *sandbox* restrictions) on applets:
  - No local file input or output
  - No direct access to local resources like printers or drivers.
  - No access to environment information, like what OS they are on.
  - Cannot invoke external programs, or run system commands
  - Cannot make network connections to any other machine, EXCEPT to the server the applet came from (the *phone-home* restriction).

Enforcing Java Security

- The security model for Java applets is enforced in two ways:

  1. A *security manager* (in the form of an object) compiled into the program
     - oversees all security-sensitive system calls
     - if there is a violation by the applet, an exception is thrown and the applet is aborted.

  2. The Java interpreter (running the applet on the client machine) incorporates a package called *bytecode verifier*
     - verifies that the byte-code was generated by a valid Java compiler (one which following the language restrictions).

Problems with Java Restrictions

- Since Java applets are placed under such heavy restrictions, it is currently no more than just amusing animation and demo tools.
- Vendors are modifying the Java security model to address this
  - by allowing the user a choice to grant permission to an applet to side-step the "sandbox" restrictions in a controlled manner.
    - Eg allow access to certain directories, to print to certain printers, etc.
  - having the applets digitally signed so that it can be verified as a trusted applet

JavaScript Security

- JavaScript (the Netscape version) have received a lot of attention for some of its security holes
  - some of which have been patched.
- JavaScript security seems to not have been designed right from the start (like Java's), but evolved with time.
- There are some promise for JavaScript in terms of security
  - Netscape have announced future browsers will support JavaScript code signing (like the new Java security model) to verify trusted JavaScript code.

Example JavaScript Security Problems

- Some of the known major problems with JavaScript which were later fixed:
  - Ability to send email in the user's name - without the user knowing.
  - Ability to obtain directory listing of the client local file system
  - Ability to upload contents of a file
  - Ability to monitor other sites visited by the user, besides the one where the JavaScript resides
  - Ability to log images viewed by the user, by having JavaScript opened in one frame able to spy on contents in another frame

Other Annoying JavaScript Behaviours

- JavaScript have also been known to also cause some less malicious, but annoying outcomes:
  - Use large chunks of memory, and slows system to a crawl
  - Can open windows faster than you can close them
  - Cause the browser to quit.

Precautions Against Malicious Active Contents

- Setting User Privileges
  - Since active contents run under the privileges of the current user using the browser, it is restricted to whatever the current user have rights to do.
  - Do not:
    - Run browsers when logged in as privileged users (eg. root in UNIX or Administrator in WinNT)
    - Allow unprivileged users more access privileges than necessary

Precautions Against Malicious Active Contents (cont'd)

- Install and use up-to-date virus checkers .
- Verify the integrity of downloaded software
    - Software these days includes checksum or fingerprint (eg. MD5 hashes) information that can be used to verify that the downloaded software has not been tampered with.
- Software from home sites are generally safer than mirror sites.

Precautions Against Malicious Active Contents (cont'd)

- See week 13's tutorial on instructions to set active content settings in MSIE and Netscape Navigator.

- Blank Page -

- Blank Page -