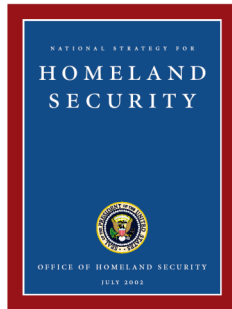


## Homeland Security Red Teaming

- Directs intergovernmental coordination
- Specifies “Red Teaming”
  - Viewing systems from the perspective of a potential adversary
  - Target hardening
  - Looking for weakness in existing security measures



COMP 6370 – Web Security – Lecture 11



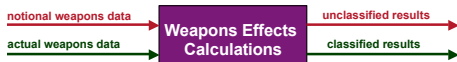
Unreasonable Security Fears



COMP 6370 – Web Security – Lecture 11



## Typical DoD Security Model Nuclear Weapons Training



- Assumptions
  - Calculating nuclear and chemical weapons effects are already well known, only the actual weapons capabilities are classified
  - The calculations themselves do not reveal sensitive information about training, tactics and procedures used in nuclear/chemical targeting

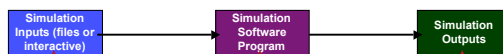


COMP 6370 – Web Security – Lecture 11



## Attacking Simulations as Software

- OBJECTIVES
  - Look for the underlying models the simulation is constructed from
  - Compromise training, tactics and procedures
  - Compromise weapons and systems performance data



- |   |  |  |
|---|--|--|
| <ol style="list-style-type: none"> <li>1. Experimentation w/"open source" system data</li> <li>2. Privilege escalation via buffer overflows</li> <li>3. Analysis of bounds checking if implemented</li> </ol> | <ol style="list-style-type: none"> <li>1. Exploitation of OS vulnerabilities</li> <li>2. Analysis of installed files</li> <li>3. Decompilation, disassembly of targeted executables</li> </ol> | <ol style="list-style-type: none"> <li>1. Sensitivity Analysis of output based on input changes</li> <li>2. One "off" test cases to examine relationships</li> </ol> |
|---|--|--|



COMP 6370 – Web Security – Lecture 11



## Background

- Separation of program and data is once more abolished by web-based applications
  - Content providers embed executable content in documents to create interactive web pages that can process user input
- Computation is moved back to the client
  - Documents include executable code
  - Clients run on quite powerful machines
    - Servers free themselves by offloading computation to clients
    - Clients need protection from rogue content providers
- Mobile code moves from machine to machine collecting information from different places or looking for spare computing resources
- Users are forced to become sys admins and make policy decisions



COMP 6370 – Web Security – Lecture 11



## Browsers

- Provide “bells and whistles” to support attractive presentation of content
- Is a service layer for web applications
- Includes the protocols to communicate with web servers
- Manage security relevant information for the client



COMP 6370 – Web Security – Lecture 11



## Browsers in the Trusted Computer Base

- Browsers handle web traffic
  - has to indicate return address as a minimum
  - lack of privacy protection as the server can build up a database about its clients
- Browsers manage default settings and preferences
  - default settings include the location of executables
  - security prefs indicate the protection clients want to apply to their web session
- Browsers keep a cache of recently visited pages
  - this is convenient for the user
  - consider using a terminal in an airport lounge
- Browsers often run in “system mode” with full access to all system resources



COMP 6370 – Web Security – Lecture 11



7

## Browsers in the Trusted Computer Base (cont.)

- Web security applications use encryption and digital signatures
  - When performed for the client, browser entrusted with the client's private keys
  - Browsers today come with the root verification keys of major certification bodies
  - Browser must protect
    - verification keys from modification
    - signature keys from disclosure
    - encryption keys from disclosure
- Browsers integrate other comm services like email
  - Unnecessary use of a complex program to run email
  - Email messages can exploit browser bugs
  - Unexpected interactions
- Overall, browsers being used for functions they were not intended for



COMP 6370 – Web Security – Lecture 11



8

## CGI Scripts

- Common Gateway Interface
  - Metalanguage (middleware)
  - translates URLs or HTML forms into runnable programs
  - Scripting languages used
    - Perl, TCL, etc.
    - Server Side Includes (SSIs)
    - SSI in-lines
  - example: page counter



COMP 6370 – Web Security – Lecture 11



9

## Sample CGI Attack

- A script for sending a file to a client may look like:  

```
cat thefile | mail clientaddress
```

  - where thefile is the name of the file and clientaddress is the mail address of the client
- When a malicious user enters:  

```
cat thefile | mail user@address | rm -rf/
```

  - as the mail address the server will execute and after mailing the file to the user, delete all files the script has permission to delete



COMP 6370 – Web Security – Lecture 11



10

## Options for aliasing CGI on a UNIX Server

- Script-aliased CGI:
  - all CGI scripts are put into one directory
  - e.g. ./cgi-bin in the web server root directory, e.g. /var/httpd
  - EASIER to find and track all CGI scripts
- Non-script-aliased CGI:
  - all CGI scripts are identified by their extension, e.g. .cgi.



COMP 6370 – Web Security – Lecture 11



11

## Securing CGI Scripts on UNIX

- Need UID for web server program
  - do not run web server program as “root”
- Create a special web server UID and carefully control its access rights
  - Do not share UID with other services
- Conduct code review of installed CGI scripts
  - use public resources for checking
  - Different issues between say ENS and Earthlink
- EXEC operator with argument cmd  

```
<#!#exec cmd = "myprogram myparameters" ->
```

  - passes the string myprogram myparameters to /bin/sh for execution
    - malice can come from the program or the parameters, particularly if myparameters contains a shell escape
  - Options Includes NOEXEC
  - Unescape operation gets rid of shell escapes in input coming from the client by commenting out escape characters



COMP 6370 – Web Security – Lecture 11



12

## Cookies

- Where web servers store information about their customers
  - searching large customer databases on server costly
- HTTP requests do NOT automatically identify individual users
  - Thus easier to use a cooperating browsers' customer side
  - Server requests browser to store a cookie that contains information the server will use the next time the client calls
    - .netscape/cookies
- Cookies give browsers the chance to create stateful HTTP sessions
- Privacy
  - cookies stored by the browser create client profiles



COMP 6370 – Web Security – Lecture 11



13

## UNIX Cookie Example (.netscape/cookies)

- www.marion-institute.org FALSE / FALSE 2137622427 CFID 475137
- www.marion-institute.org FALSE / FALSE 2137622427 CFTOKEN 2642479
- .bravenet.com TRUE / FALSE 1373583329 HASCOOKIES 1
- .bravenet.com TRUE / FALSE 1293837163 BNUC366777 1058309425
- marionmilitary.edu FALSE / FALSE 2137622427 CFID 475161
- marionmilitary.edu FALSE / FALSE 2137622427 CFTOKEN 92849103



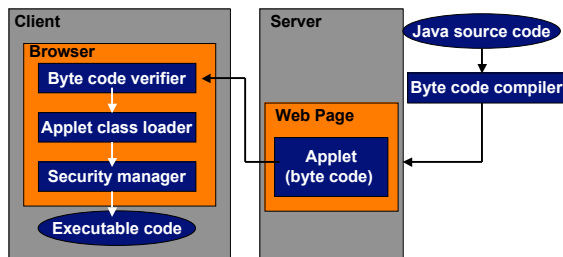
COMP 6370 – Web Security – Lecture 11



14

## Java Sandbox

- Executable Content (applets) from remote web sites



COMP 6370 – Web Security – Lecture 11



15

## Environment for Applets

- Users cannot rely on prior acquaintance and trust relationship with the source of an applet
- Few users are willing to rule personally on each access request made by an applet
- Client's operating system cannot be expected to offer any protection



COMP 6370 – Web Security – Lecture 11



16

## Language Design Decisions (Java)

- The language itself should make it more difficult for programs to create damage.
- The execution environment provides mechanisms for access control
- The security policies enforced by the execution environment have to be set correctly



COMP 6370 – Web Security – Lecture 11



17

## Security for Executable Java Applets (Objectives)

- Applets do not get access to the user's file system
- Applets cannot obtain information about the user's name, email address, machine configuration, etc.
- Applets may make outward connections only back to the server they came from
- Applets can only pop-up windows that are marked "untrusted"
- Applets cannot reconfigure the system, e.g. by creating a new class loader or a new security manager



COMP 6370 – Web Security – Lecture 11



18

## Byte Code Verifier

- **Checks for:**
  - the class file is in the proper format
  - stacks will not overflow
  - all operands have the correct type
  - there will be no data conversion between types
  - all references to other classes are legal
- **Byte code verifier reduces the workload on the interpreter**
  - guaranteed code properties do not have to be checked again
- **However, security still depends on the run-time environment**



COMP 6370 – Web Security – Lecture 11



19

## Applet Class Loader

- **Class loader protects the integrity of the run-time environment**
- **Applets must not be allowed to create their own class loaders**
  - Applets are handled by the applet class loader
- **Java comes with its own class library**
  - The CLASSPATH environment variable specifies the location of built-in classes
  - The security issues associated with altering CLASSPATH should be obvious
- **“Spoofing” of the CLASSPATH can be avoided by:**
  - If the applet class loader first searches the built-in classes in the local name space
  - Then expand search to the class making the request



COMP 6370 – Web Security – Lecture 11



20

## Security Manager

- **Reference Monitor of the Java Security Model**
  - Performs run-time checks on ‘dangerous’ methods
- **Java classes are grouped into packages**
  - packages facilitate rudimentary access control to classes
- **Variables and methods can be declared as follows:**
  - **Private:** only the class creating the variable or method has access
  - **Protected:** only the class creating the variable or method and its subclasses have access
  - **Public:** all classes have access
  - **None of the above:** only classes within the same package have access



COMP 6370 – Web Security – Lecture 11



21

## Summation of Web Threats

	Threats	Consequences	Countermeasures
<b>Integrity</b>	<ul style="list-style-type: none"> <li>•Modification of user data</li> <li>•Trojan horse browser</li> <li>•Modification of memory</li> <li>•Modification of message traffic in transit</li> </ul>	<ul style="list-style-type: none"> <li>•Loss of information</li> <li>•Compromise of machine</li> <li>•Vulnerability to all other threats</li> </ul>	<ul style="list-style-type: none"> <li>•Cryptographic checksums</li> </ul>
<b>Confidentiality</b>	<ul style="list-style-type: none"> <li>•Eavesdropping on the Net</li> <li>•Info theft from server</li> <li>•Info theft from client</li> <li>•Info about network configuration</li> <li>•Info about which clients talk to server</li> </ul>	<ul style="list-style-type: none"> <li>•Loss of Information</li> <li>•Loss of Privacy</li> </ul>	<ul style="list-style-type: none"> <li>•Encryption,</li> <li>•Web Proxy</li> </ul>
<b>Denial of Service</b>	<ul style="list-style-type: none"> <li>•Killing of user threads</li> <li>•Flooding machine with bogus threats</li> <li>•Filling up disk or memory</li> <li>•Isolating machines by DNS attack</li> </ul>	<ul style="list-style-type: none"> <li>•Disruptive</li> <li>•Annoying</li> <li>•Prevent user from getting work done</li> </ul>	<ul style="list-style-type: none"> <li>•Difficult to prevent</li> </ul>
<b>Authentication</b>	<ul style="list-style-type: none"> <li>•Impersonation of legitimate users</li> <li>•Data Forgery</li> </ul>	<ul style="list-style-type: none"> <li>•Misrepresentation of user</li> <li>•Belief that false information is valid</li> </ul>	<ul style="list-style-type: none"> <li>•Cryptographic techniques</li> </ul>



COMP 6370 – Web Security – Lecture 11



22