Disk Level Virus Detection

Adrienne Felt
Nathanael Paul, David Evans, Sudhanva Gurumurthi
What is a virus?

• Self-replicating program
• Adds its own code to the host’s files
• Runs without permission
• Seeks out new hosts
• Destroys data
• Annoys the user
Where do viruses come from?

- Motivation
  - “Show off”
  - Financial gain
- Compete with anti-virus software
  - Try to circumvent AV protection
How to catch a virus

• String scanning
  – Compare files against a database of known viruses

• All files stored as bits on a disk
  – `MOVE.W D4 D5`\[0011110000000100\]

• Signatures are strings of bits
  – `00000110010001000000000000000001001`
Do virus scanners work?

Norton Anti-Virus detection rates

- WildList viruses: 100%
- Zoo threats: 97%
- Heuristic detection:
  - 1-month-old signatures: 22%
  - 2-month-old signatures: 8%
- Outbreak response time: 10-12 hrs

*From AV-test.org, an independent testing agency. Published in PC Mag.*
How viruses don’t get caught

• “Morphing” viruses
  – Change their own code between generations
  – For example:
    • x = x+1;  a = x;
    • a = a + 1;
    • x = a;
  – Now it won’t match the signature!
    • ...00110011000001000111011000100...
    • ...0010000010110010010000110000010...

05.FELT  disk level  virus detection
The war against morphing viruses

- Emulation
  - Slow
  - Imperfect
- Custom methods
  - Hard to create
  - W95/ZMist - a week or more
Our solution

- Behavior-based signatures
  - Harder to change behavior than code structure
- Watch behavior from a different place
  - Use the disk processor
  - Viruses access files stored on the hard drive
  - Disk processor sees all reads and writes to the disk
1. User opens a file
2. Operating system asks for data
3. Disk processor retrieves the data
4. Data given to the OS
5. Anti-virus scanner
1. User opens a file

2. Operating system asks for data

3. Disk processor retrieves the data

3.5 Disk processor watches for viruses

4. Data given to the OS

Our model
My research

- Goal: show that idea is feasible
  - Create “low-level” behavioral signatures
  - Difficult-to-detect viruses
- Experimental overview
  - Run the viruses
  - Record their behavior
  - Looked for patterns
• Future disk processors
  – File names, opens, closes, offsets

• Current disk processors
  – Reads, writes, block numbers

1. Virus accesses a file
2. OS asks for data
3. Disk processor retrieves the data
4. Data given to the OS
Disk requests -> signatures

READ 1636.1672 14:27:20:984 <NO NAME>
   block= 530 ## 5Êå~Éf«É RtlCreateActivationContextSXS: %s…

WRITE 1636.1672 14:27:20:984 EFISHNC.EXE
   block= 15 ## <@KERNEL32.dllUSER32.dllExitProcessWriteProce…

• Can see behavior from these requests!
  – Use “goat” files to make it clearer
  – Run lots of traces
  – Patterns emerge
What makes it hard?

- Precision and accuracy
- False positives
  - Misidentifying a “good” application as a virus
  - Costly to consumers
- False negatives
  - Not catching a virus
  - Have to catch all generations
EFishNC is a complex virus
  – Encrypted morphing virus
  – Hard to detect using normal methods

Infection sequence:
1. Drops a file named "Explorer.exe" into C:\Windows
2. Explorer.exe executes
3. Explorer.exe infects the entire hard drive
4. Attacks the network
Beginning of the signature

• The original virus executing
  – READ block 0
  – CLEANUP apphelp.dll
  – CLEANUP apphelp.dll
  – CLEANUP sysmain.sdb

• Explorer being dropped
  – CLEANUP Explorer.exe
  – WRITE block 0 Explorer.exe
  – CLEANUP Explorer.exe
  – READ block 0 Explorer.exe
Evaluation of the signature

• False negatives
  – Generations & variants
  – Signature caught all of them
• False positives
  – Initially: 0 false positives
    • Unrealistic
  – Probability argument
    • Modeling normal workloads
Conclusion

- Anti-virus software is not perfect
  - Tricked by morphing viruses
- Disk level virus detection
  - Behavioral instead of code-based
- Promising results
  - Created a signature for a complex virus
  - Future work on probabilistic analysis

- felt@virginia.edu
- www.cs.virginia.edu/malware