Towards Disk-Level Malware Detection

Nathanael Paul
Sudhanva Gurumurthi
David Evans

University of Virginia

Cobassa Workshop, Nov. 7, 2005

Current Malware Detection

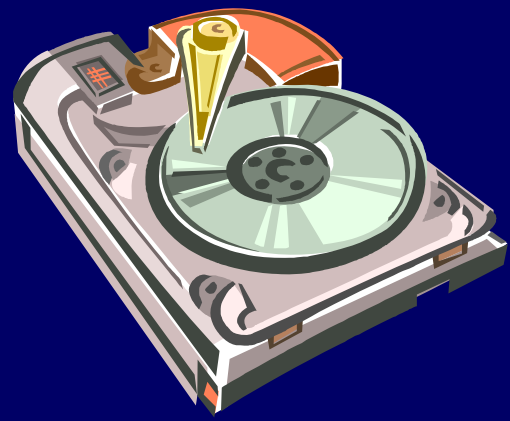
- String matching
  - Match bytes in .exe to virus signature (40% of CPU time in ClamAV [Silberstein '04])
- Emulation
  - Must emulate decrypt or decompress viruses to match against virus signature
- AV software 129% overhead on single processor [Uluski et al. '04]

Proposed AV Solutions

- Software
  - Symantec, McAfee, F-Secure, ...
- Hardware Coprocessor
  - Tarari
- Use **Disk Drive Processor**
  - Seagate Momentus, EMC checksums Oracle DB blocks
  - Let disk do some AV work

Disk-Drive Characteristics

- Data Parallelism (SIMD-type) system
- Can actively process data instead of simple data transfers
- Sees I/O traffic at lower-level



Disk-Drive Malware Detector

- ~60% of CPU time in ClamAV is disk I/O
- Extra processing capability with each drive (cheap and scales)
- Think PDA with 8-16 MB RAM

Workload Partitioning between host CPU and disk CPU

- Disk processors are about 2-3 generations behind current desktop CPUs
  - WinNT 4.0 (Pentium with 16 MB RAM)
- Disk CPU is underutilized
- Secure communication and updates problematic

Dynamic Analysis of Disk I/O

- Funlove virus
  - Only detected once disk had been infected
- Opportunity to protect data just before persistent state updated
  - Protection of critical OS data
  - Scanning should have low overhead

Dynamic Analysis Challenges

- Semantic information loss
  - Only will see reads and writes of disk blocks
  - Could bootstrap semantic information at OS installation
- Updates/communication a problem
  - May make use of Trusted Platform Module (TPM)

Rootkit Detection

- Normally comparison between high-level and low-level information
  - Registry, Master File Table, Kernel Process Information
- Scan without detection from rootkits

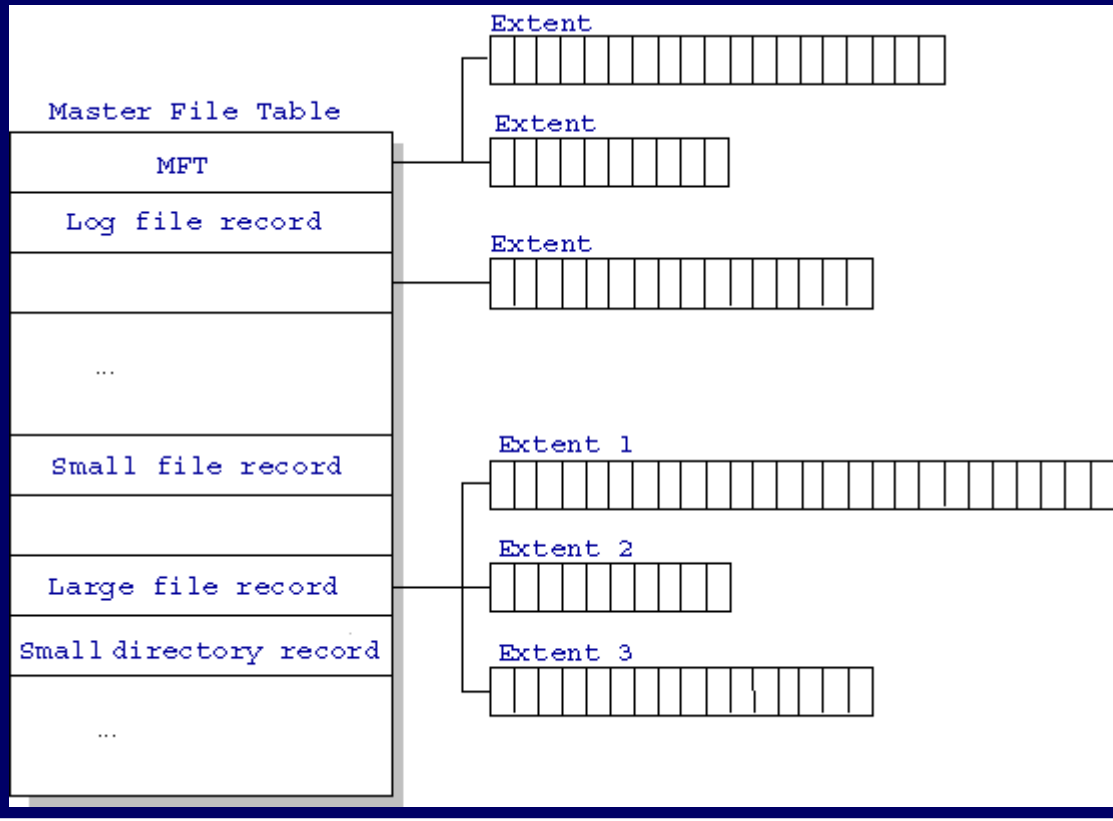
Example Scan

```
C:\WINDOWS\system32\cmd.exe

C:\research\harddrive\presentations\cobassa>dir /s /b
C:\research\harddrive\presentations\cobassa\clam.png
C:\research\harddrive\presentations\cobassa\disk-level-malware-detection-commented.ppt
C:\research\harddrive\presentations\cobassa\disk-level-malware-detection.ppt
C:\research\harddrive\presentations\cobassa\disk-level-malware-v2.ppt
C:\research\harddrive\presentations\cobassa\NTFS-MFT-structure.gif

C:\research\harddrive\presentations\cobassa>_
```

versus



Rootkit Challenges

- Communication a must
  - Disk or host must perform comparison
- Updating is again a problem
  - Without TPM, we “up the bar” of compromise

DADDIO: **D**ynamically **A**nalyze **D**isk **D**rive **I/O**

- Goal: Prevent corruption of OS while continuing safe execution
  - Recovery and disinfection
- DADDIO only has to process **writes** to disk
  - Software running on disk processor
  - Could ship on disk drive

DADDIO Challenges

- Recognizing malicious activity from low-level disk blocks
- 8-16MB of RAM
  - Performance may suffer if too much RAM used
- Will throttle execution if system under heavy load

Conclusion

- Opportunities exist for partitioning AV workload with disk CPU
- Will first look at what information can be learned without communication with host AV engine
- Recovery tactics

Questions?

nate@cs.virginia.edu
http://www.cs.virginia.edu/daddio