Capturing System-Wide Information for Malware Detection and Analysis

The work of:

Heng Yin, Dawn Song, Manuel Egele, Christopher Kruegel, and Engin Kirda

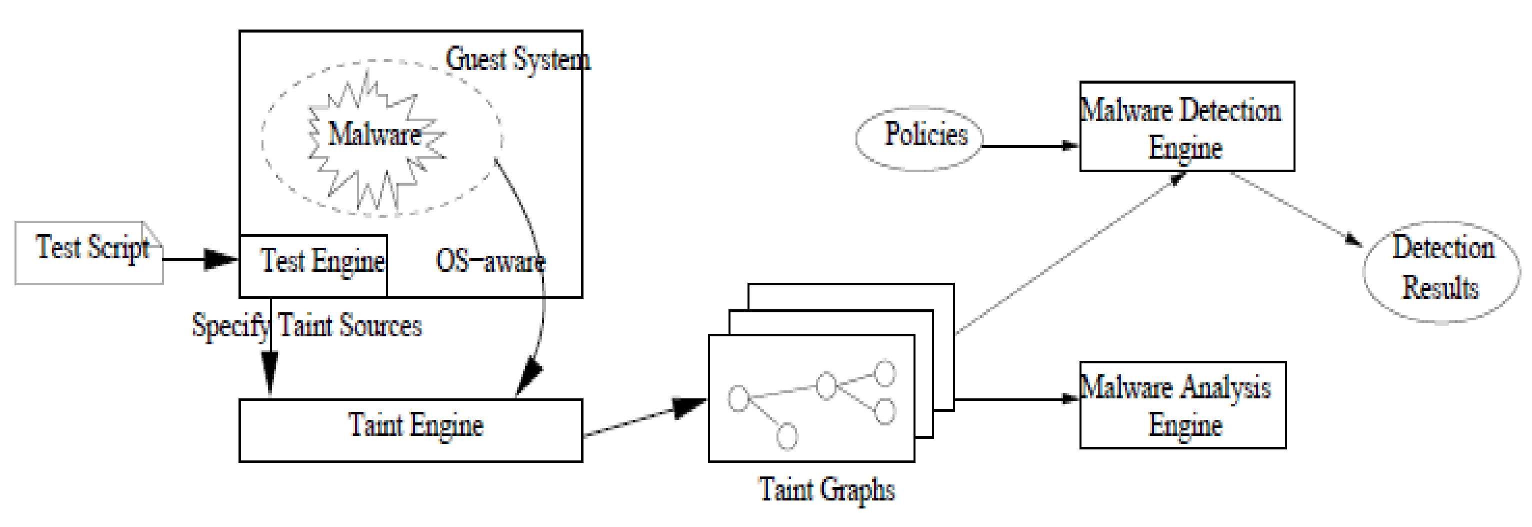
CCS 07: Proceedings of the 14th ACM conference on Computer and communications security.

Identification vs Detection

- Signatures rely on pattern recognition
- Signature based detection only works once a threat has been detected.
- What about threats that hide themselves?
- Malware is typically analyzed manually
- Need a behavior based malware detection approach
- Other approaches:
  - Don't address kernel attacks
  - Monitor system calls rather than data access

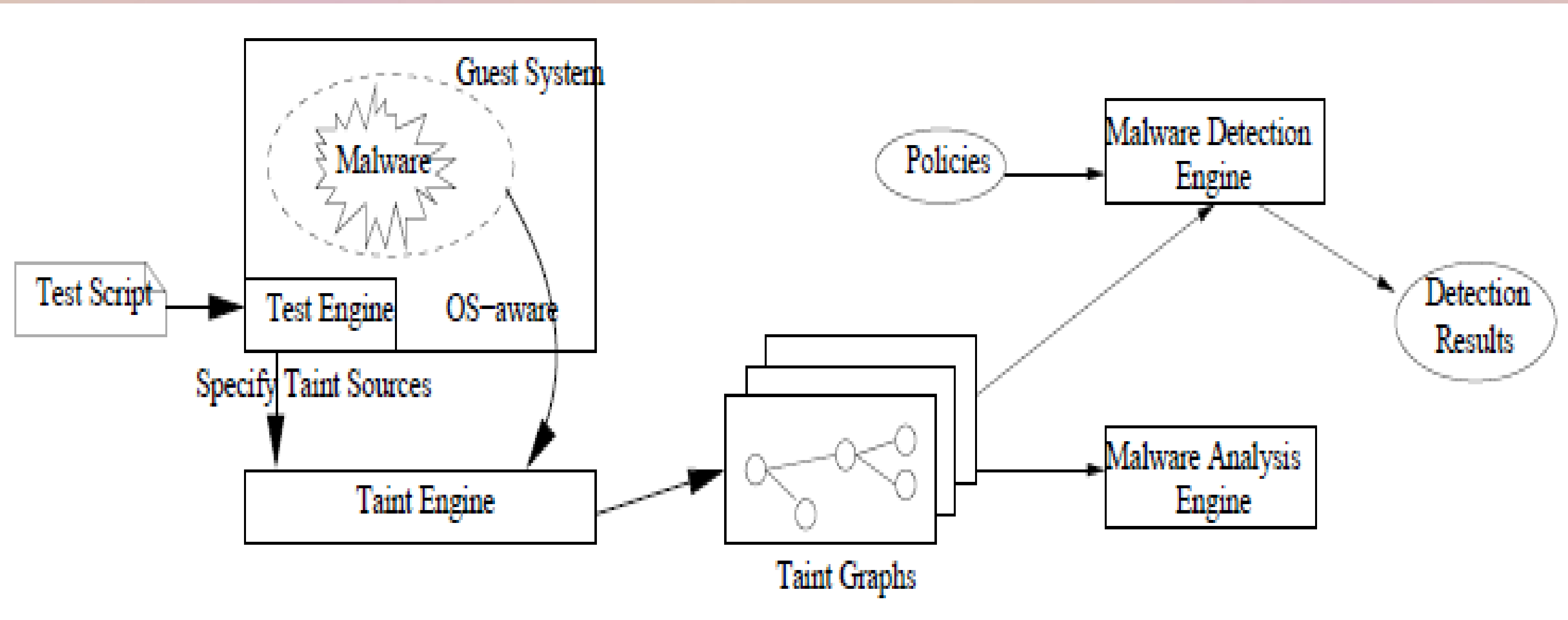
Goals

- Develop an automated process
- Offline analysis
- Identify many different forms of malware



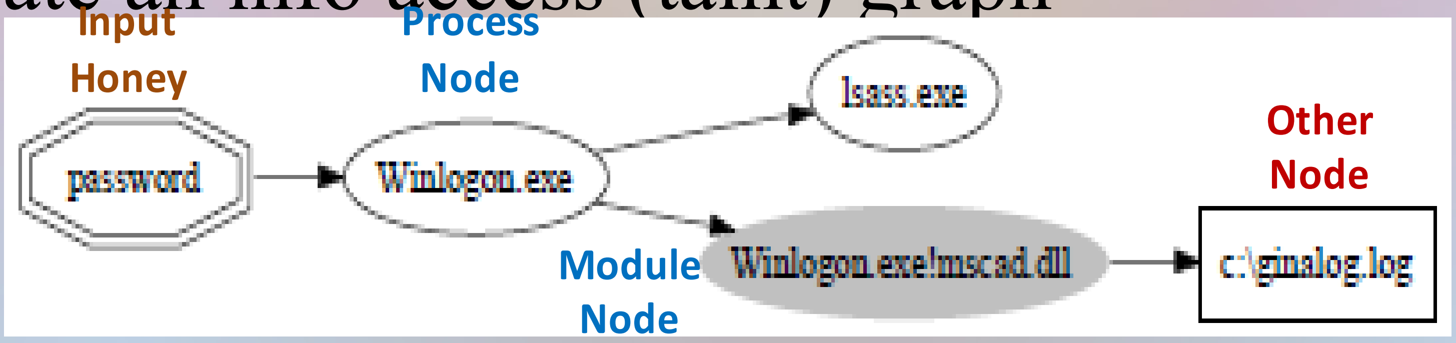
Detection Approach

- Tracking information access
- Generate a directed graph
- Analyze the results



Tracking Approach

- Hardware is used to track the information access
  - OS aware – ID what process is doing the accessing
- Track the information as it is accessed (type, value)
  - type ::= taint_source | os_object
  - taint_source ::= text | password | HTTP | HTTPS| FTP| ICMP | document | directory
  - os_object ::= process | module | network | file
- Create an info access (taint) graph



Taint Graph Example - Grabbing a password

Classify Suspicious Behavior

- Categorize three kinds of anomalous behavior
  - Anomalous information access
  - Anomalous information leakage
  - Excessive information access

- Anomalous information access behavior
  - Any secondary access is highly suspicious behavior
    - Keyloggers, password thieves, network sniffers, and stealth backdoors

Information Leakage

- Anomalous information leakage behavior
  - Acceptable for the samples to access them locally, but unacceptable to leak the information to third parties
  - Some secondary access is OK (local only)
  - Trackers, spyware/adware
    - HTTP
    - HTTPS
    - Documents
    - URL

Excessive Access

- Excessive information access behavior
  - Occasional access is typical
  - Malware will access information excessively to achieve their malicious intent
    - Rootkit behavior
      - Privileged hidden access
    - Filesystem request interception
      - File concealment

Test Stimulus

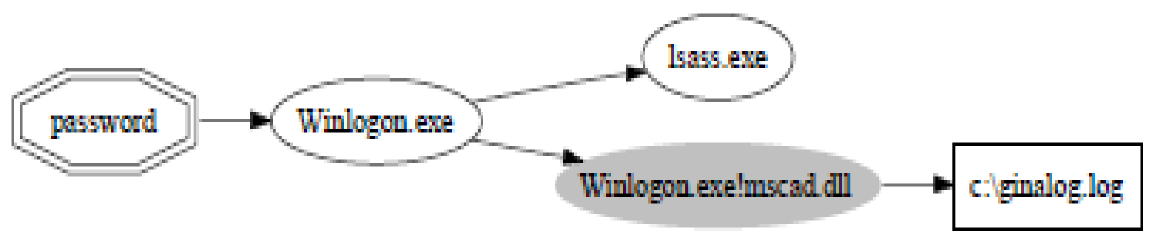
- Honey sources:
  - Keyboard
    - Text, password, and URL
  - Network
    - HTTP, HTTPS, FTP, ICMP, and UDP
  - Disk
    - Document and directory input

Detection Conditions (Policies)

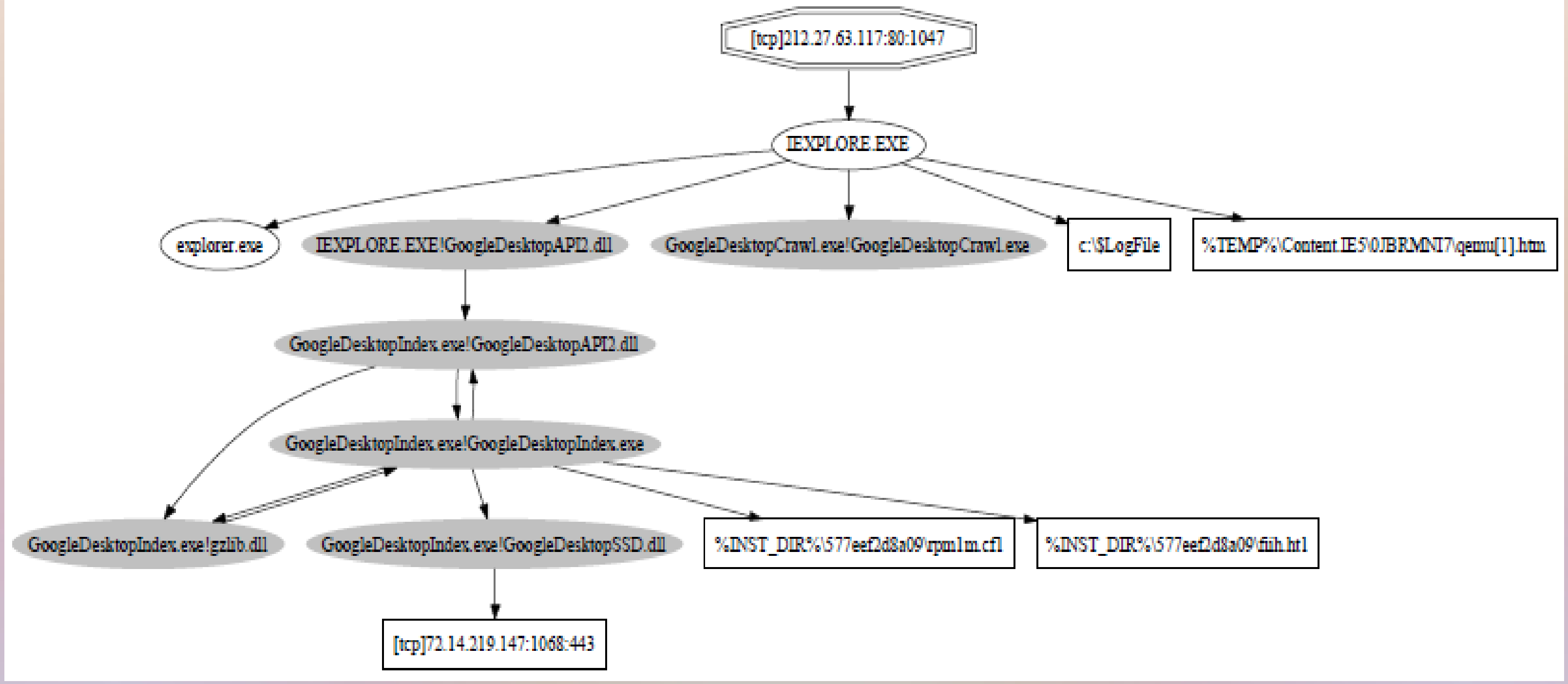
- Text, password, FTP, UDP and ICMP inputs can not be accessed by the samples
- URL, HTTP, HTTPS and document inputs cannot be leaked by the samples
- Directory inputs cannot be accessed excessively by the samples

Taint Graph

- Taint graph will show if the sample has accessed any input information (suspicious behavior)
- Graph will show what the suspect has done with the data
  - How it is intercepted
  - Which process grabs it
  - Where it goes
  - What is done with it



Google Desktop

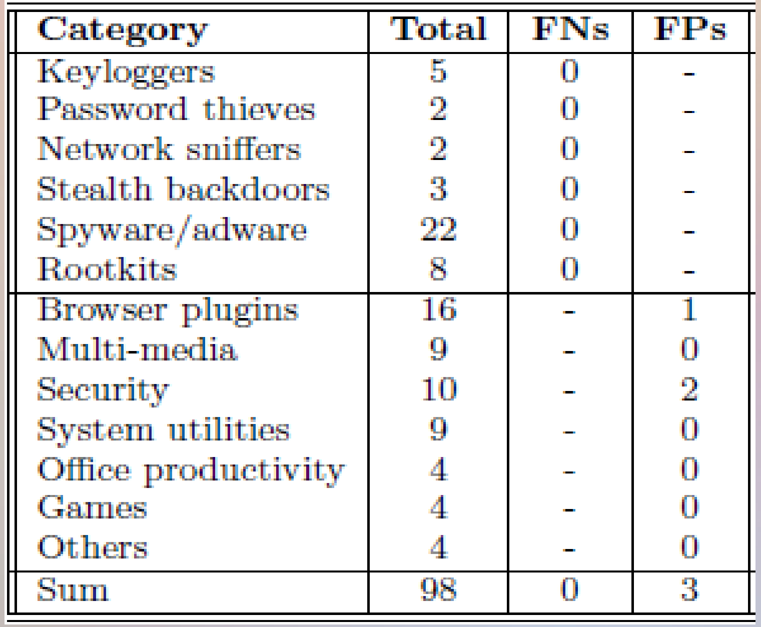

[tcp]212.27.63.117:80:1047
IEXPLORE.EXE
explorer.exe
IEXPLORE.EXE!GoogleDesktopAPI2.dll
GoogleDesktopCrawl.exe!GoogleDesktopCrawl.exe
c:\$LogFile
%TEMP%\Content.IE5\0JBRMNI7\qemu[1].htm
GoogleDesktopIndex.exe!GoogleDesktopAPI2.dll
GoogleDesktopIndex.exe!GoogleDesktopIndex.exe
GoogleDesktopIndex.exe!gzlib.dll
GoogleDesktopIndex.exe!GoogleDesktopSSD.dll
%INST_DIR%\577eef2d8a09\rpm1m.cfl
%INST_DIR%\577eef2d8a09\fiih.htl
[tcp]72.14.219.147:1068:443

Evaluation

- Panorama ran on a Linux machine with a dual-core 3.2 GHz Pentium 4 CPU and 2GB RAM
- On top of Panorama: Windows XP Professional with 512M of allocated RAM
- Malware samples (42)
  - Anit-Virus Company
  - Academia
  - Web (rootkit.com)
- Google Desktop as a case study

Results

- Benign sample source
  - Fresh downloads from www.download.com
  - Freeware, 56 samples
- 3 False Positives
  - 1 Browser accelerator
    - Web page prefetch
  - 2 Firewall programs
    - Network traffic monitor
  - Behave like malware
  - Panorama observes behavior not intent



Category	Total	FNs	FPs
Keyloggers	5	0	-
Password thieves	2	0	-
Network sniffers	2	0	-
Stealth backdoors	3	0	-
Spyware/adware	22	0	-
Rootkits	8	0	-
Browser plugins	16	-	1
Multi-media	9	-	0
Security	10	-	2
System utilities	9	-	0
Office productivity	4	-	0
Games	4	-	0
Others	4	-	0
Sum	98	0	3

Cost

- Average slowdown of 20 times
  - Speed was not a design goal
- Suggested performance improvements
  - Different execution technique
    - Virtual & emulation approach
  - Dynamic binary instrumentation
    - Software only approach – 4% overhead (qualified)
  - Use of Error Correcting Code memory
    - Data authentication

Evasion

- Info leak concealment
  - Unauthorized info access still detected
- Conditional launch mechanism (unresolved)
  - Timer triggers
  - Application specific
  - Emulation detection
- Interfering malware
  - Fix the malware (bug) exploit

Strengths of this approach

- Implemented outside the subject system
- Captures the info access & processing technique of the malware
- Uses a hardware approach for detection
- Rootkit and hidden file detection

Questions?