Towards Automated Characterization of Malware's High-level Mechanism Using Virtual Machine Introspection

Shun Yonamine
Laboratory for Cyber Resilience
Nara Institute of Science and Technology

Background: How to infer the intention of malware automatically?

- It is essential to analyze the relationship between each APIs.
  - Since malicious activities are conducted through the series of low-level actions.
- Basically, inferring relies on the manual efforts of the human analysts.
- Requirements
  - Reconstruct the semantics of the series of low-level actions
  - Define the policies to detect the maliciousness

Observed Item	Type
cat	Process
meterpreter	Process
sh	Process
/etc/passwd	File
sys_read	System call
sys_sendto	System call
sys_write	System call
192.168.124.131	IP address
44261	Port
...	...

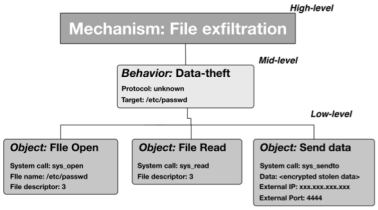
What kind of intention is hidden in those observed items?

C2? Data theft? Encryption? Whatever...

Even if every IOC can be captured, the relationship between them remains obscure.

Methodology: Automatically track malicious processing through data flow to gather clues.

- Our standpoint
  - **Extract the series of low-level actions that relate to each other** which execute on the attacker's intention
  - **We focus on the file access** by malware since **the way of manipulating data depends on the attacker's intention**.



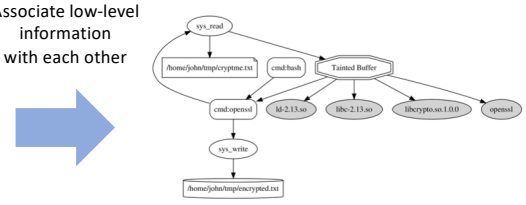
As for ***data-theft***, the malware **opens** the file and **reads** the data before **sending** those data outside the network.
It can be said that **there is the relationship** between each of low-level actions **as long as they handle the data** derived from **the same data source.**

- Our approach
  - Focus on **the flow of data**
  - **Monitor "what" file is accessed**
    - Use API monitoring
  - **Track "how" the file-data is processed**
    - Start taint analysis when a targeted file is opened

Observed Item	Type
openssl	Process
sh	Process
/home/john/tmp/cryptme.txt	File
/home/john/tmp/encrypted.txt	File
sys_read	System call
sys_write	System call
libcrypto.so.1.0.0	Shared library
libc-2.13.so	Shared library
...	...

Those low-level actions are independent information.

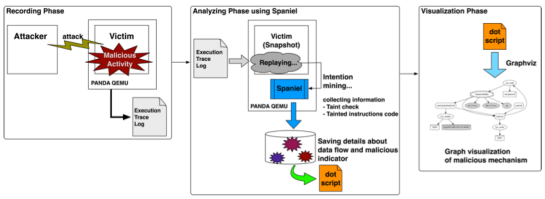
Associate low-level information with each other



- Key features of our approach
  1. **File monitoring based** analysis
  2. Automatically **associate with each low-level action**
  3. Visualization
     - **Extract malicious behavior** as the abstracted format

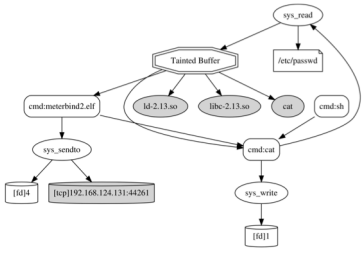
The intention of File encryption can be seen clearly.
File data is handled by the encryption library ("*libcrypto.so*") before being written into another file.

Experiment & Evaluation: Data-flow tracking can gather the intention of malware.

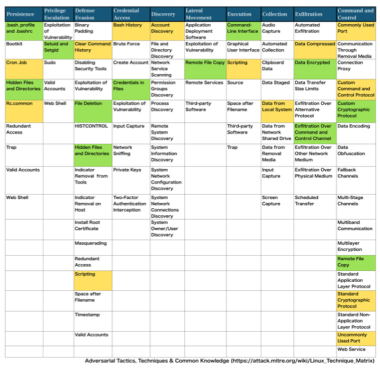


Three phase (Our implementation is in analyzing phase)

1. Execute malware in QEMU
   - Record execution-trace-log
2. Analyze the trace using our implementation
   - **Hook the file-read** API and start tainting
   - Taint checking and **identify the intention**
     - **When send() buffer is tainted, it is data-theft.**
     - **When write() buffer is tainted, it is file tampering.**
   - Inspect tainted instructions
   - Leverage virtual machine introspection (VMI) to reconstruct semantics
3. Visualize runtime-behavior of malware



This characterizes the intention of "File exfiltration".



Leverage ATT&CK matrix to evaluate.

- Traits for detecting the malicious intention
  - Tainted buffers
  - OS utilities used by an attacker
    - shared libraries
    - os commands
- Observed **low-level items can be connected through tainting**.

Summary

- We proposed an approach to **automatically extract the intention** of the attacker.
- **By tracking data-flow of the file, intention analysis works in practice**.
- Our prototype is capable of analyzing each stage of the cyber kill chain. We evaluated using ATT&CK models