
EDINBURGH
JUNE 16-21
2019

# Improving the Efficiency of Dynamic Malware Analysis with Temporal Syscall Measure

**Dr. Chih-Hung Lin**

**Taiwan Computer Emergency Response Team/ Coordination Center (TWCERT/CC)**

**Taiwan Network Information Center (TWNIC)**

twcertcc

# Outline

- Introduction
- Efficiency mechanics for dynamic malware analysis
  - Virtual time controller
  - Information measure for early stopping
  - Resistance to virtual time controller & Information measurement
- Experiment
- Conclusion

# Introduction

# Malware Analysis

- Static Malware analysis – e.g., grep, pattern match
  - To find malicious patterns from given codes
  - Effective in detecting known threats
  - Advanced attacks can easily bypass static method
- Dynamic Malware analysis – e.g., sandbox
  - A file is placed in a controlled environment & its behavior patterns are examined when executed
  - Practical way to defeat the code obfuscation attempts
  - Hide the malicious behavior
    - Launching its malicious behavior when certain conditions are met
    - Timer trigger [Dinaburg et al., 2008], Event trigger
  - Execution-stalling loop detection [Kolbitsch et al., 2011]
  - Time consuming (3-5 minutes/file)

1(hr/day) x 24 (hr) x 60 (min/hr) / 3 (min/file)
= 480 (files/day) → Inefficient !!

# Common Ways to Improve the Efficiency

- More Computers
  - Use numerous physical machines simultaneously to perform parallel computation
  - More physical space for such enormous machines
  - Costly & needs more resources
- More VMs
  - Most commonly used method
  - Use numerous virtual machines (VMs) simultaneously to perform parallel computation
  - Less physical space, resources or cost
  - Still takes minutes to analyze a file

# Conventional System Clock Speedup

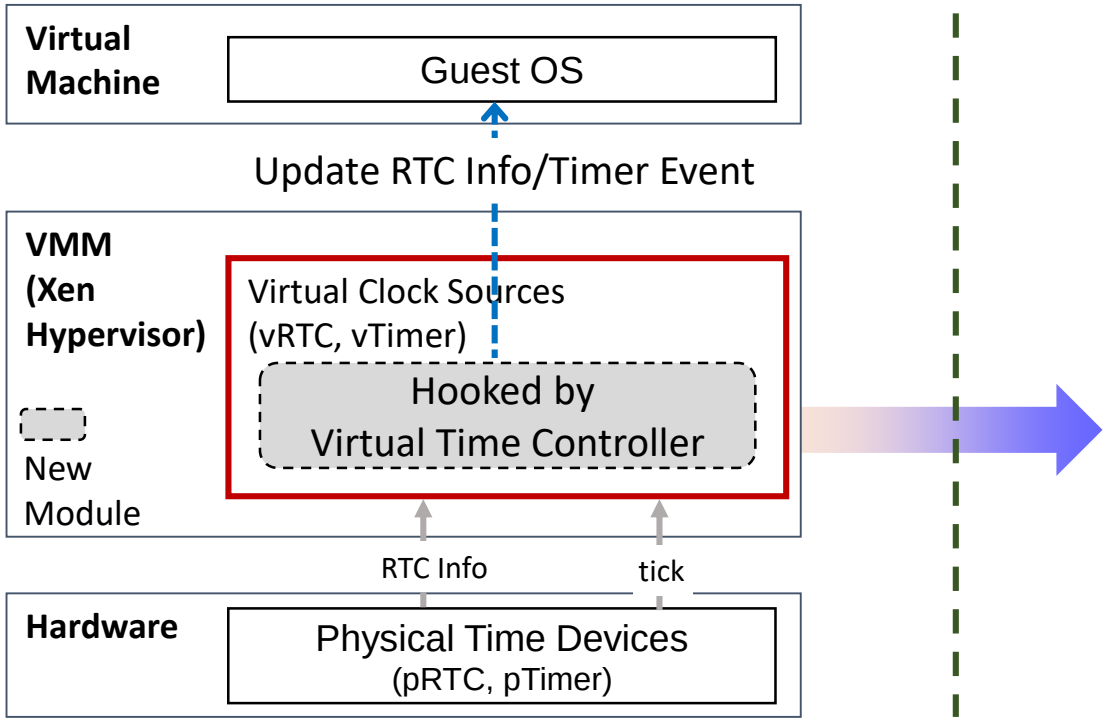
- Reduction of the latency of dynamic analysis
  - System clock speedup is a feasible solution
- Modify time parameters inside the OS kernel [Kobayashi, 2010]
  - Different OSs need to be separately modified
  - Restricted to OSs, which open their source code (e.g., Linux)
- Adopts time-related API hooking mechanisms for the OS and modify the time parameters [Gray-Donald and Price, 2013]
  - Requires the acquisition in advance of all functions relating to time
  - In cases a function is modified, concealed, or unmodifiable, the system becomes incomplete and its effectiveness is therefore reduced

# Efficiency mechanics for dynamic malware analysis

- Virtual time controller (VTC)
- Information measure for early stopping
- Resistance to VTC & Information measurement

# Virtual Time Controller

- The VTC hooks virtual clock sources and then alters the period of the virtual RTC (vRTC) and virtual timer (vTimer)




$$new\_vRTC\_p = \frac{vRTC\_p}{speed\_ratio}$$

$$new\_vTick\_p = \frac{vTick\_p}{speed\_ratio}$$

[C.-H. Lin et al., Computer & Security, 2018]

# Temporal Syscall Measurement for Early Stopping (1)

- System call vector

  $s_i$ (process ID, name, arguments)

- Shannon entropy $H(S)$
  - To measure how diverse the system calls are
- Relative entropy $D(S^t \,||\, S^{t-1})$
  - To measure how different between the distributions of system calls in this & next moment
  - Small value of relative entropy
    - the lists of system calls and their distributions in this & in the next moments are similar

Information measurement

$$H(S) = -\sum_i P(S = s_i) \log P(S = s_i)$$

$$D(S^t \parallel S^{t-1}) = \sum P(S^t) \log \frac{P(S^t)}{P(S^{t-1})} \geq 0$$

if $P(s_i^{t-1})$=0, then $P(s_i^{t-1})$=0.0001

# Temporal Syscall Measurement for Early Stopping (2)

- Terminated
  - While malware complete its execution, the VTC will receive unitary NtClose system calls
  - $H \sim D \sim 0$
  - Accompany NtClose system calls for continued time slots

  => Early stopping
- Execution-stalling loop
  - VTC will continue receiving system calls and producing nonzero entropy values
  - $H > 0$
  - $D \sim 0$

  => Increase the speed ratio

# Anti-VTC

- Compare the differences in epochs with various clock sources
  - System clock sources

  => All time sources in the guest VM are altered
  - Determine the correct time via NTP

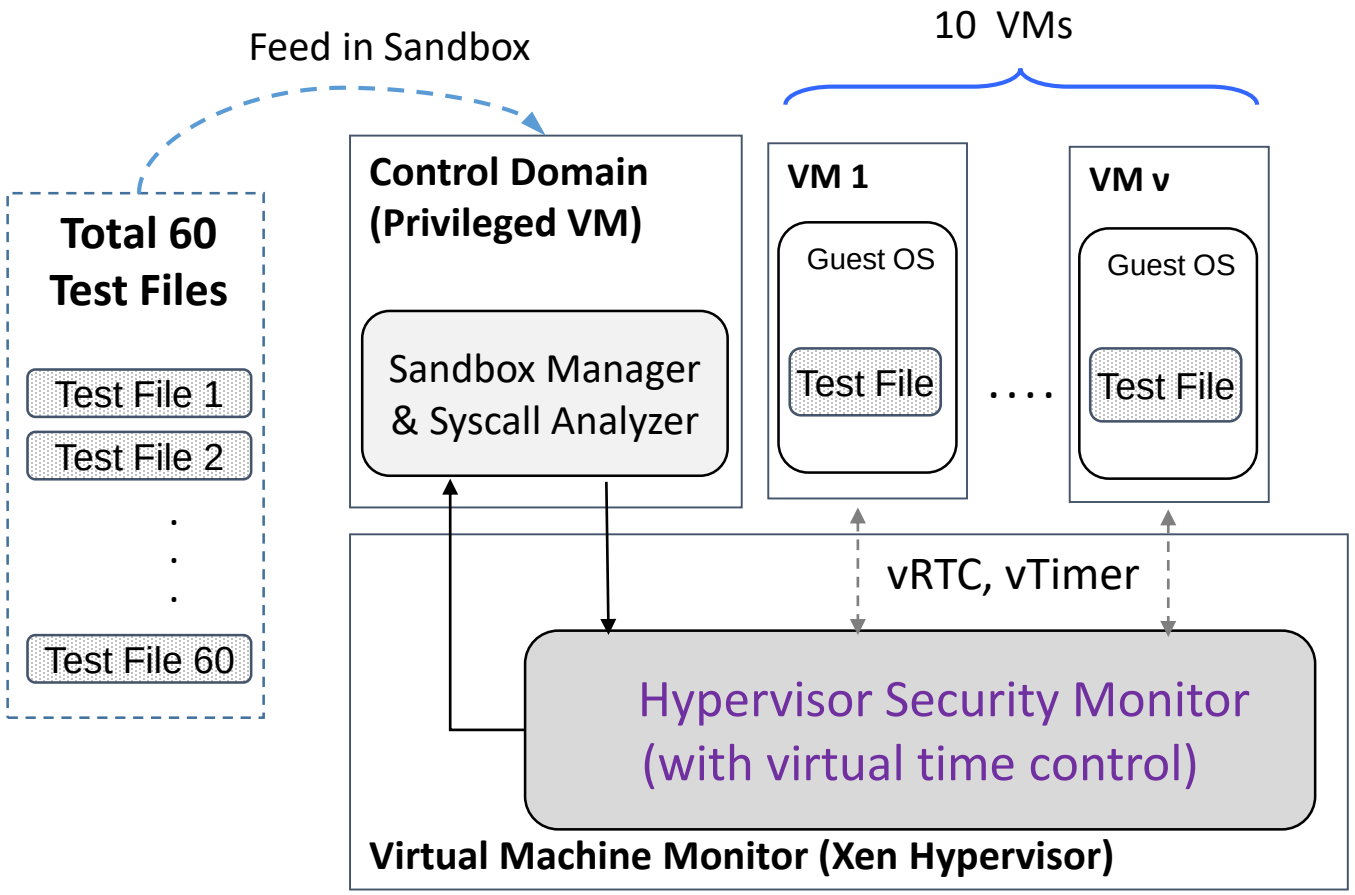
  => Build a fake NTP service in Sandbox
  - No information that malware can use to resist VTC mechanism inside a guest VM
- Escape from the border of the guest VM [Luan, 2016] and then detect the deviation of the clock
  - VTC can be detected by these highly sophisticated malware
  - VTC mechanism will still be effective to deal with less sophisticated malware

# Anti-Temporal Syscall Measurement

- Disturb the information measurement
- Malicious adversary may create and release system resources to produce large NtClose system calls
- Frequently creating and releasing system resources will make the malware noisy and easily detectable in the sandbox

# Experiment

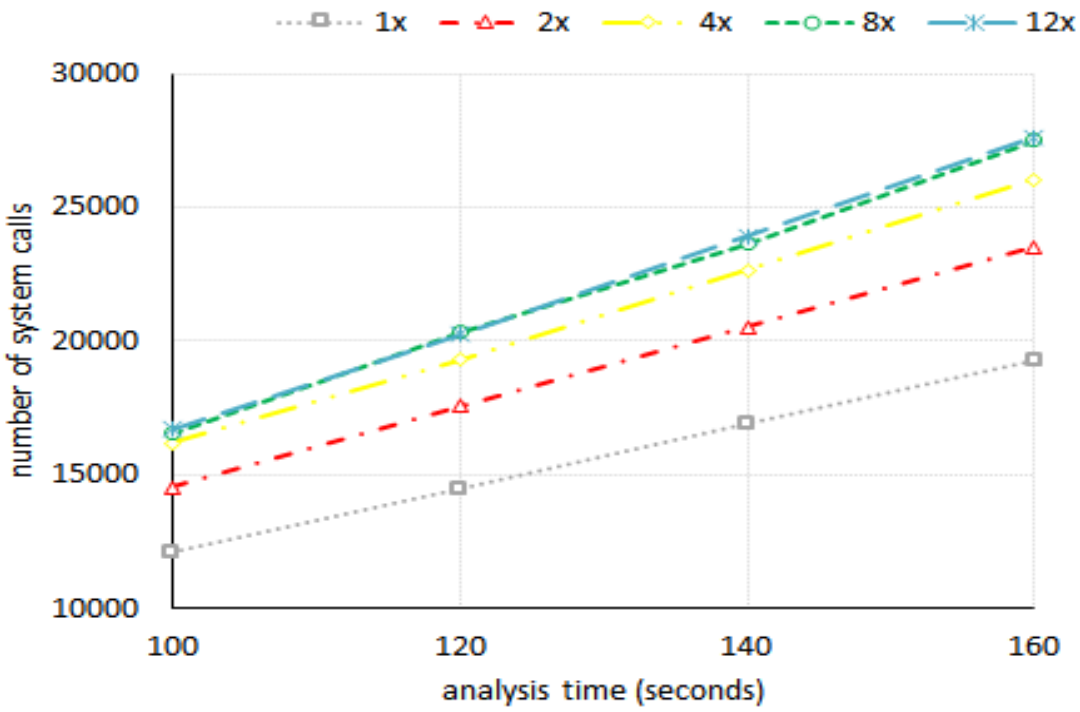
# Experiment Setup




- The sandbox system had executed the ten VMs in parallel in the VTC environment until all 60 test files were analyzed

# Effectiveness of VTC

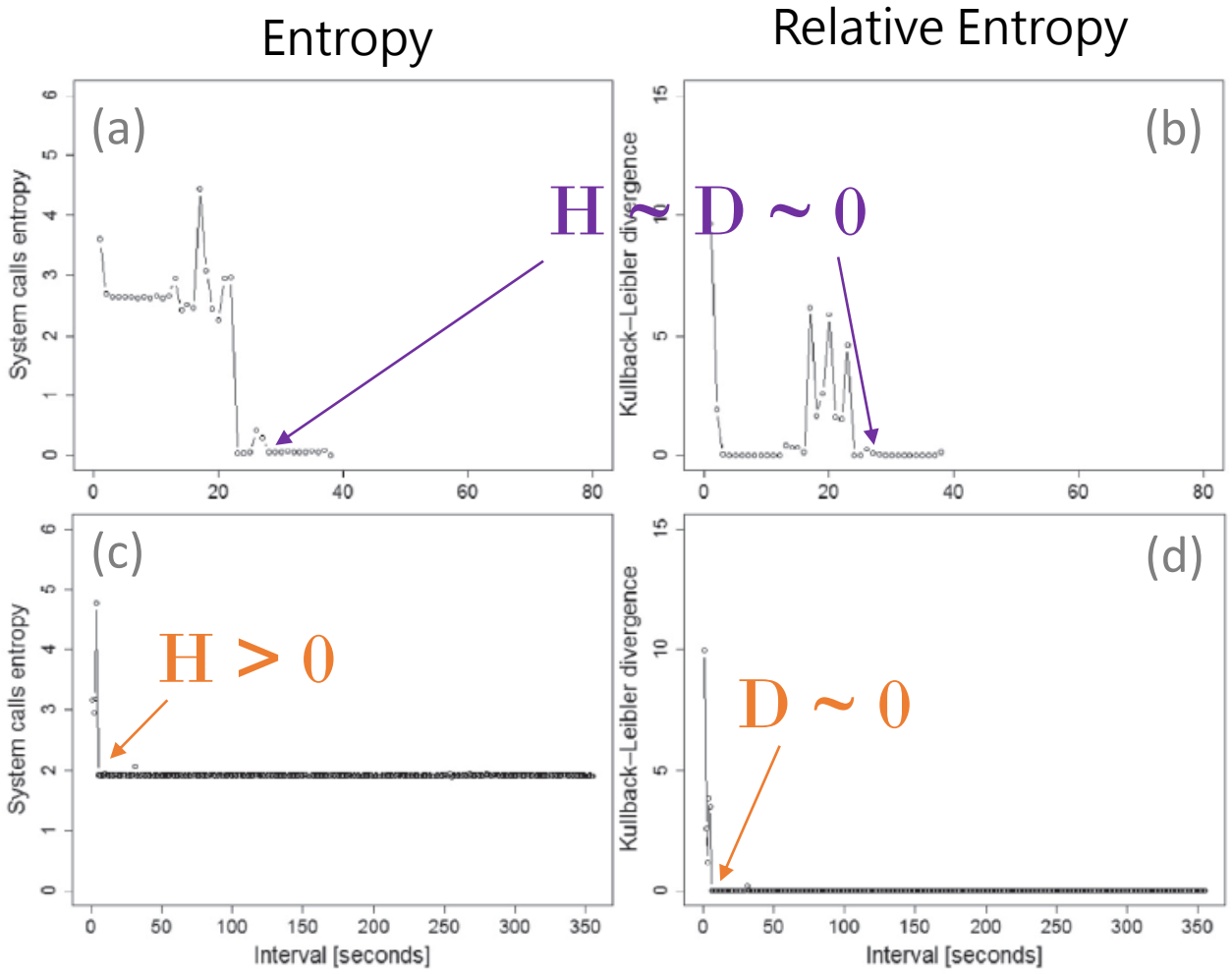
- In a computer system, the longer a program executes, the more system calls can be logged
- More specifically, if we can obtain more system calls in the same period, then the Sandbox is more efficient
- There is a significant positive correlation between speed ratio and the number of system calls
- The number of system calls noticeably increases for time-speed ratio 1x (non-VTC) to 4x, and then increases mildly from 8x to 12x



# Early Stopping Experiment

DarkHotel
@8x

Srizbi
@8x



# Conclusion

- We present a sandboxing-based method to reduce the latency of dynamic analysis using virtual time speedup and entropy-based measurement
- cyber security researchers can easily root out potential security problems in minimum analysis time
- To counter sophisticated malware with timing-based evasion technologies, VTC can be combined with existing techniques for further research.

# References


- C.-H. Lin, H.-K. Pao, and J.-W. Liao, “Efficient dynamic malware analysis using virtual time control mechanics,” *Computers & Security*, vol. 73, pp. 359 – 373, 2018.
- S. Luan, “Exploit two xen hypervisor vulnerabilities.” Blackhat US 2016, 3-4 Aug. 2016.
- A. Dinaburg, P. Royal, M. Sharif, and W. Lee, “Ether: malware analysis via hardware virtualization extensions,” in *Proceedings of the 15th ACM conference on Computer and communications security*,pp. 51–62, ACM, 2008.
- C. Kolbitsch, E. Kirda, and C. Kruegel, “The power of procrastination: detection and mitigation of execution-stalling malicious code,” in *Proceedings of the 18th ACM conference on Computer and communications security*, pp. 285–296, ACM, 2011.
- Y. Kobayashi, “Linux kernel acceleration for long-term testing,” in *CELF Embedded Linux Conference Europe*, (Cambridge, UK), 27-28 Oct. 2010.
- T. A. Gray-Donald and M. W. Price, “Date and time simulation for time-sensitive applications,” Jan. 8 2013. US Patent 8,352,922.

# Thank you!