The Next Generation of Robust Linux Memory Acquisition Techniques via Consequential Timestamp Dumps

Saeed Shafiee Hasanabadi, Arash Habibi Lashkari, Ali A. Ghorbani

Canadian Institute for Cybersecurity (CIC), University of New Brunswick (UNB)

CIC

UNB

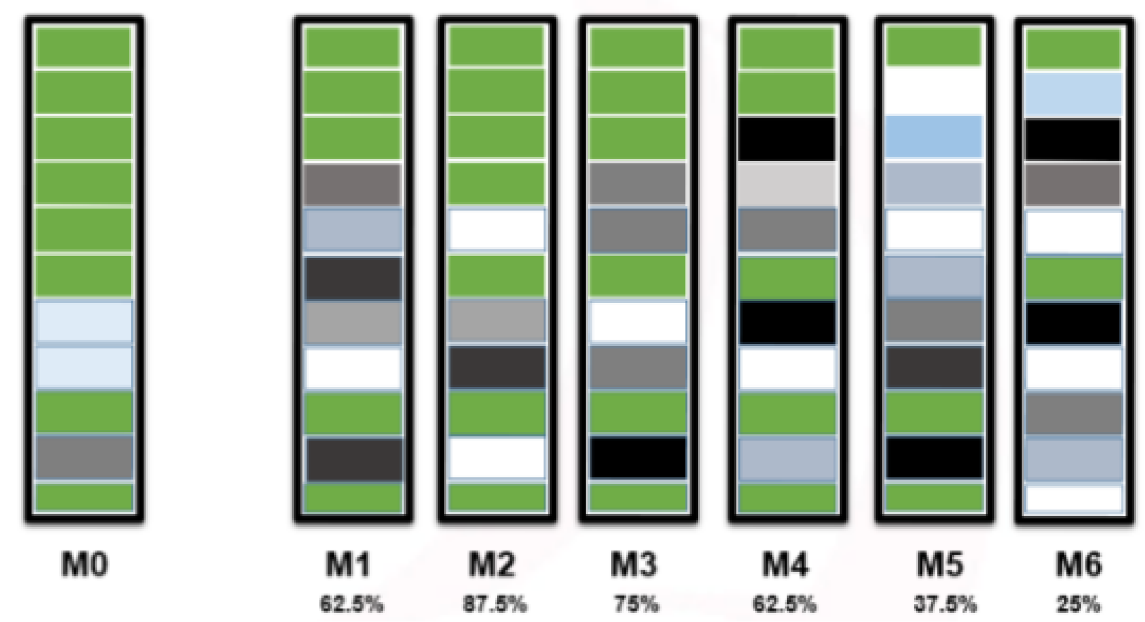
Cybersecurity investigators use memory forensics to detect evidence of attacks. Selected methodologies and techniques for memory analysis are important; however, the accuracy of performed analysis is dependent on the dumped memory and whether the image of the physical memory is consistent, atomic, and reliable. The increment of the volume of physical memories, the elapsed time for memory acquisition, the tampering and page smearing effects, and anti-forensics techniques are current challenges for memory acquisition and even memory analysis techniques. Here we addressed these challenges by proposing an approach to determine and to approximate how much the content of consequent images can help to reduce required I/O operations for image acquisition, tampering, page smearing, and anti-forensics effects, and also speed up image acquisition and rebuild specific parts of sequence images from available images. The proposed idea is applicable to software-based and hardware-based memory acquisition methods. According to the experiments' results, the proposed idea shows how much, approximately, a digital investigator can use previous memory dumps of a specific operating system with specific architecture to find similar parts in a new image memory. Also, it can help administrators or automatic systems to estimate how much timestamp thresholds can approximate the desired value for a consequential memory dump mechanism based on the available architecture and operating system.

Limitations of Memory Acquisition Techniques

CATEGORY	YEAR	REF	LIMITATION
Software-based Acquisition Techniques	2005	[21]	The impact of memory acquisition applications on the target machine
	2007	[16]	Elapsed time for memory acquisition
	2007	[29]	Page smearing
	2012	[25]	Tampering effects on the content of RAM
	2016	[10]	Increasing amounts of RAM and page smearing
	2016	[11]	Trail obfuscation and artifact wiping
	2017	[9]	Memory forensics framework crashing because of smear and malicious tampering
Hardware-based Acquisition Techniques	2006	[26]	Unnecessary memory-to-memory copies, disk I/O, write operation
	2006	[26]	Data reduction during memory acquisition, and volume of target machines
	2007	[14]	The dumped memory has the same size as the RAM
	2011	[37]	Issues of anti-forensic techniques
	2013	[39]	Atomicity of acquired RAM
Independant Research	2004	[6]	Tampering effects on the content of RAM
	2007	[28]	Tampering effects on the content of RAM
	2010	[41]	Tampering effects on the content of RAM

.

Comparison of Timestamped Memory Dumps



Machine-user and Human-user based Scenarios

Configuration of Installed Virtual Machines			
OS	Architecture	Processors	RAM
Fedora (27-1.6)	32-bit and 64-bit	3	1024
OpenSUSE	32-bit and 64-bit	3	1024
Debian (9.3.0)	32-bit and 64-bit	3	1024
Ubuntu (16.04.3)	32-bit and 64-bit	3	1024
CentOS (70)	32-bit and 64-bit	3	1024

Information on Memory Dumps for Machine-based Stress	
Time Window	Count
2 Minute	19
3 Minute	14
5 Minute	9
10 Minute	5

Algorithm

Algorithm 1 Pseudo code for written C module

procedure MEMORY ACQUISITION, STRESS-NG EXECUTION AND MD5 CALCULATION
  $l \leftarrow$ *NUMBER-OF-PREDETERMINED-DUMP-FILES*
  $m \leftarrow$ *SIZE-OF-PHYSICAL-MEMORY*
  $n \leftarrow$ *Stress-NG-TIME*
  if $fork() == 0$ **then** Stress-NG("−vm-bytes m −vm-method all −verify -t n -v")
loop:
  if $i <= l$ **then**
    if Memory acquisition module is inserted **then** remove it
    Insert memory acquisition module
  $i \leftarrow i + 1$.
  goto *loop*.
  Compute hash for captured memory dump
  close;

Results

Evaluation Result				
OS	2Min	3Min	5Min	10Min
Fedora (27-1.6)	**6.48**%	5.68%	5.81%	4.90%
OpenSUSE	**16.79**%	10.84%	14.36%	9.39%
Ubuntu (16.04.3)	11.68%	16.60%	**17.08**%	10.20%
Debian (9.3.0)	19.71%	**22.96**%	13.58%	6.00%

TABLE 2: Common block percentage among consequent images for machine-based stress on 32-bit OS

Evaluation Result				
OS	2Min	3Min	5Min	10Min
Fedora (27-1.6)	**9.10**%	8.40%	7.75%	6.06%
OpenSUSE	10.82%	10.99%	**11.71**%	9.69%
Ubuntu (16.04.4)	**8.89**%	3.83%	3.85%	4.94%
Debian (9.3.0)	5.46%	**8.90**%	5.05%	5.88%
CentOS (7)	7.62%	6.79%	**8.60**%	7.92%

TABLE 3: Common block percentage among consequent images for machine-based stress on 64-bit OS

Evaluation Result				
OS	2Min	3Min	5Min	10Min
Fedora (27-1.6)	**8.85**%	8.83%	8.29%	8.69%
OpenSUSE	**8.71**%	8.19%	7.52%	6.83%
Ubuntu (16.04.3)	10.06%	**11.00**%	8.34%	9.13%
Debian (9.3.0)	8.08%	12.32%	6.87%	**12.55**%

TABLE 4: Common block percentage among consequent images for human-based mode stress on 32-bit

Evaluation Result				
OS	2Min	3Min	5Min	10Min
Fedora (27-1.6)	12.86%	**23.41**%	13.52%	3.86%
OpenSUSE	8.26%	10.40%	**9.35**%	8.38%
Ubuntu (16.04.4)	8.37%	8.51%	9.92%	**10.51**%
Debian (9.3.0)	6.42%	6.69%	**8.05**%	6.37%
CentOS (7)	9.71%	8.52%	**10.36**%	9.53%

TABLE 5: Common block percentage among consequent images for human-based mode stress on 64-bit