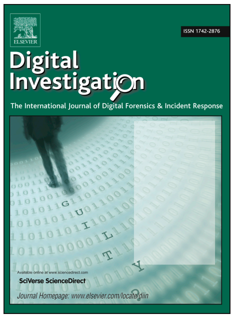

# Scanning memory with Yara

Michael Cohen

PII: S1742-2876(17)30059-2

DOI: 10.1016/j.diin.2017.02.005

Reference: DIIN 673

To appear in: *Digital Investigation*

Received Date: 1 September 2016

Revised Date: 5 January 2017

Accepted Date: 18 February 2017

# Scanning Memory with Yara.

Michael Cohen[a]

[a] *Google Inc., 747 6th St. Kirkland, Washington, USA*

**Abstract**

Memory analysis has been successfully utilized to detect malware in many high profile cases. The use of signature scanning to detect malicious tools is becoming an effective triaging and first response technique. In particular, the Yara library and scanner has emerged as the defacto standard in malware signature scanning for files, and there are many open source repositories of yara rules. Previous attempts to incorporate yara scanning in memory analysis yielded mixed results. This paper examines the differences between applying Yara signatures on files and in memory and how yara signatures can be developed to effectively search for malware in memory. For the first time we document a technique to identify the process owner of a physical page using the Windows PFN database. We use this to develop a context aware Yara scanning engine which can scan all processes simultaneously using a single pass over the physical image.

## 1. Introduction

Memory Scanning has been used as a quick and powerful way to detect anomalies or malicious software running on a system. For example, pool scanning techniques have been used to detect remnants of kernel objects such as exited processes, file handles and other kernel data structures - even after these have been freed from the active set (Sylve et al., 2016; Schuster, 2006). Scanning techniques can be used to identify and isolate encryption keys from process memory (Hargreaves and Chivers, 2008), and detect unique signatures for malware families (Oktavianto and Muhardianto, 2013).

There are a number of modes of applying scanning techniques - one can scan the process's virtualized view of memory, or the physical address space directly (i.e. the raw memory image itself). In general, scanning the physical address space tends to be faster because IO throughput is optimized (in the case where the user wants to exhaustively scan all processes). However scanning the Virtual Address Space may be more efficient when the user only wants to scan a targeted subset of running processes.

The Yara library and scanner has emerged as the defacto standard for communicating signatures used to identify malware files (Alvarez, 2016; Various, 2016). Popular memory forensic frameworks, have provided the capabilities for applying Yara signatures directly on memory images (The Volatility Foundation, 2015; The Rekall Team, 2016).

In this paper we evaluate the existing state of the art in applying yara signatures within the memory analysis domain. In particular we consider the practical difference of scanning in the Virtual Process Address space, as opposed to scanning the Memory image directly.

We describe for the first time a technique, dubbed "Context Aware Scanning", which uses the Windows PFN database to rapidly identify the owner of each physical page, and where that page is mapped in it's virtual address space.

Using this technique provides sufficient context about each physical address to be able to associate related hits in a single coherent signature - even when the scan is performed over the physical address space. We demonstrate this technique as applied to the Yara scanning engine by implementing a powerful new context aware scanning methodology.

The novel scanning technique dubbed "Context-Aware" scanning, employs detailed understanding of the address translation process with optimized scanning of the physical address space, we are able to gain performance advantage over existing techniques and efficiently scan multiple processes simultaneously. Finally we suggests guidelines for constructing more robust, memory-centric signatures.

Finally we discuss the practical differences between the different scanning techniques discussed and their applicability in effective malware identification.

## 2. Background

### 2.1. Malware identification through signature scanning

Identifying malware in files is a very common and established technique (Sathyanarayan et al., 2008). There are a number of approaches. On the one end of the scale the NSRL facilitates hash comparison analysis (Flaglien et al., 2011). This produces a high level of confidence if a hash matches that the file belongs to the suspected set. However, exact hash matching is very sensitive to small variations in the underlying file.

Commonly malware samples are not exactly identical, but rather are customized or are built from common source trees. Therefore malware samples can be clustered into malware *families*, suggesting that several samples are related to one another, although not identical.

Similarity hash matching is less sensitive to small variations in specific files and can be used to classify malware samples into respective families. However, calculating the similarity