Chapter 23
Massively Threaded Digital Forensics Tools

Lodovico Marziale
University of New Orleans, USA

Golden G. Richard III
University of New Orleans, USA

Santhi Movva
Wayne State University, USA

Vassil Roussev
University of New Orleans, USA

Loren Schwiebert
Wayne State University, USA

ABSTRACT

Digital forensics comprises the set of techniques to recover, preserve, and examine digital evidence, and has applications in a number of important areas, including investigation of child exploitation, identity theft, counter-terrorism, and intellectual property disputes. Digital forensics tools must exhaustively examine and interpret data at a low level, because data of evidentiary value may have been deleted, partially overwritten, obfuscated, or corrupted. While forensics investigation is typically seen as an off-line activity, improving case turnaround time is crucial, because in many cases lives or livelihoods may hang in the balance. Furthermore, if more computational resources can be brought to bear, we believe that preventative network security (which must be performed on-line) and digital forensics can be merged into a common research focus. In this chapter we consider recent hardware trends and argue that multicore CPUs and Graphics Processing Units (GPUs) offer one solution to the problem of maximizing available compute resources.

INTRODUCTION

The complexity of digital forensic analysis continues to grow in lockstep with the rapid growth of the size of forensic targets—as the generation of digital content continues at an ever-increasing rate, so does the amount of data that ends up in the forensic lab. According to FBI statistics (Federal Bureau of Investigation, 2007), the average amount of data examined per criminal case has been growing at an average annual rate of 35%—from 83GB in 2003 to 277GB in 2007. However, this is just the tip of the iceberg—the vast majority of forensic analyses are in support of either civil cases or internal investigations and can easily involve the examination of terabyte-scale data sets.

DOI: 10.4018/978-1-4666-2136-7.ch023

Ultimately, a tiny fraction of that information ends up being relevant—the proverbial 'needle in a haystack'—so there is a pressing need for high-performance forensic tools that can quickly sift through the data with increasing sophistication. As an illustration of the difficulty of the problem, consider the 2002 Department of Defense investigation into a leaked memo with Iraq war plans. It has been reported (Roberts, 2005) that a total of 60TB of data were seized in an attempt to identify the source. Several months later, the investigation was closed with no results. The Enron case involved over 30TB of raw data and took many months to complete. While these examples might seem exceptional, it is not difficult to come up with similar, plausible scenarios in a corporate environment involving large amounts of data. As media capacity continues to double every two years, such huge data sets will be increasingly the norm, not the exception.

Current state-of-the-art forensic labs use a private network of high-end workstations backed up by a Storage Area Network as their hardware platform. Almost all processing for a case is done on a single workstation—the target is first pre-processed (indexed) and subsequently queried. Current technology trends (Patterson, 2004) unambiguously render such an approach as unsustainable: I/O capacity increases at a significantly faster rate than corresponding improvements in throughput and latency.

This means that, in relative terms, we are falling behind in our ability to access data on the forensic target. At the same time, our raw hardware capabilities to process the data have kept up with capacity growth. The basic problem that we have is two-fold: a) current tools do a poor job of maximizing compute resource usage; b) the current index-query model of forensic computation effectively neutralizes most of the gains in compute power by traversing the I/O bottleneck multiple times.

Before we look at the necessary changes in the computational model, let us briefly review recent hardware trends. Starting in 2005, with the introduction of a dual-core Opteron processor by AMD, single-chip multiprocessors entered the commodity market. The main reason for their introduction is that chip manufacturing technologies are approaching fundamental limits and the decades-old pursuit of speedup by doubling the density every two years, a.k.a. keeping up with Moore's Law, had to make a 90 degree turn. Instead of shrinking the size and increasing the clock rate of the processor, more processing units are packed onto the same chip and each processor has the ability to simultaneously execute multiple threads of computation. This is an abrupt paradigm shift towards massive CPU parallelism and existing forensic tools are clearly not designed to take advantage of it.

Another important hardware development that gives us a peek into how massively parallel computation on the desktop will look in the near future is the rise of Graphics Processing Units (GPUs) as a general-purpose compute platform. GPUs have evolved as a result of the need to speedup graphics computations, which tend to be highly parallelizable and follow a very regular pattern. As a result, GPU architectures have followed a different evolution from that of the CPU. Instead of having relatively few, very complex processing units and large caches, GPUs have hundreds of simpler processing units and very little cache on board.

For many years, GPU manufacturers produced very specialized, idiosyncratic hardware that was narrowly tailored for its graphics workload and was incompatible across generations. This all changed in 2007 with the introduction of the NVIDIA G80 processor and the accompanying release of the CUDA development kit. CUDA enabled general-purpose computation in the GPU using a dialect of the C language. The general-purpose use of GPUs is a significant development. In terms of raw computational power, current generation GPUs can achieve a theoretical rate of 1000 GFLOPS, whereas CPUs max out around 50 GFLOPS. Figures 1 and 2 illustrate the peak computational speed