



INCIDENT RESPONSE

**SPEED CAN MEAN THE DIFFERENCE BETWEEN
SUCCESS AND FAILURE**

When it comes to investigations and incident response speed is all important. The faster an examiner gets the data in a consumable form the faster an incident can be diagnosed and resolved. Even when time to resolution isn't sensitive, response speeds can be critical to ensure that the relevant data is preserved. Every minute critical data sits on a target machine the odds that the data will dissolve, be deleted, over-written, or altered increases dramatically. This is particularly true in the case of incriminating data, malicious code or evidence of hacking, because that data is generally located in volatile memory (RAM), which is constantly changing. Frankly, though, even when the evidence is on the disk the people it incriminates are generally working relatively hard to get rid of it, which tends to dramatically reduce its shelf life. Much like a homicide case, where the first 48 hours are critical to finding the killer, in incident response, the first few hours, and even minutes, are often the difference between success and failure.

Despite the importance of speed in the context of an investigation, it is actually a very difficult concept to define. The reason for the complexity is that there are many types of investigations, and each one has its own set of factors that determine the overall speed. For example an investigation of the volatile data of 10 computers follows a very different path and is subject to very different performance metrics than the investigation of 100 computers' hard drives. All seasoned examiners know that there is no average investigation, no common case, and therefore no one single measure of performance. That said, at least from a technological stand point, there are a few common themes that run through all investigations, which allow us to have a relatively productive conversation about the key factors that must be considered when evaluating investigative products.

In every meaningful investigation there is some degree of data gathering that must take place and some degree of data processing. Data gathering may simply be the act of acquiring full disk images and RAM dumps or it could entail a targeted search of machines to locate data reactive to specific criteria. Processing can mean as little as presenting the acquired information in an easily consumable format or as much as breaking down complex file types, extracting text, and building an index. While the importance of these phases may vary significantly from case to case, the fact is they are both required phases that represent the key technical hurdles in all meaningful investigations. As a result, when evaluating the speed of incident response software, in order to make true and valid comparisons, it is essential to break the analysis down into both the speed at which a solution can acquire data and the speed at which it can process that data.

Data Identification and Acquisition Speed

Data acquisition speed is the speed at which a system is able to identify and acquire the data from a specific machine or group of machines. The relevant factors in data acquisitions are the performance of the target machines, the speed of the network in between the target machines and the examiner machine, and the architecture and performance of the software being used. Since we are only interested in developing a metric for evaluating the software, we will ignore the other two factors, but before we do, it is important to note that either one of them can radically change the overall speed of data acquisition. It simply doesn't matter how optimized the software is; if the network can't move data quickly, acquisition times will be slow.

The two critical questions to ask when evaluating the relative acquisition speeds of different solutions are the following: "How quickly can the solution acquire data from a single system?" and "How well do those times scale when acquiring data from multiple systems?" Starting first with the acquisition of data from a single machine, we need to look at both how quickly a system can identify data and then how quickly it can actually transport it from the target computer to the desired location. How quickly a system can identify data is generally a function of how smart the agent is. If as part of an investigation the system needs to locate all files with a specific hash, or that have a specific attribute, how well the agent is able to achieve that is critical to the ultimate data acquisition speeds.

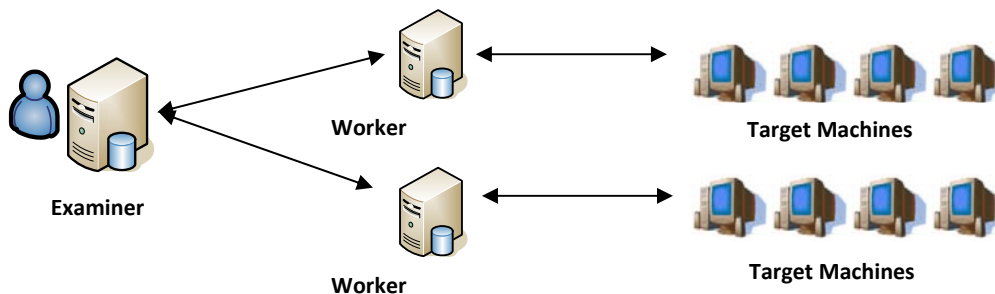
What purchasers of incident response systems need to understand is exactly what happens at the agent level versus what happens at the examiner level. Many common investigative solutions will say that everything happens at the agent, when in fact all the agent does is move data back and forth between the computer and the investigator. This may be a viable solution in the investigation of a small number of machines, but it doesn't scale, and systems that utilize this methodology are invariably unable to meaningfully investigate more than a few systems at any one time. For truly scalable performance an investigative solution needs to have an agent that is effectively file system aware. Meaning the agent must understand the file system of the computer it is running on so it can rapidly and independently execute basic investigative functions on the machine.

Once the agent has located the relevant data, the key factor is not necessarily how quickly it can pass it back to the examiner, since that is largely governed by the speed of the network, but what options it presents to the examiner for the handling of that data. The key option that a solution must offer is the ability to redirect the collected data either to a staging point or to local media. This may seem counterintuitive, since we suggested the goal was to get the data back to the examiner, but the simple fact is that objective is not always practical or even possible. Imagine trying to acquire data from a small remote office that is linked to you via a VPN connection over modem speeds. Obviously that won't work. What is truly important in those situations is that the examiner has the ability to ensure the data is preserved and can be captured at a later date. As a result, the key option that any enterprise-class solution must have is the

ability to acquire data from one computer and instead of bringing it back to the examiner pass that data to a local network share or other storage device. Other lesser, but still important options that the solution should have are the ability to compress the collected data prior to transport, which can provide as much as a 20% reduction in size, and the option to encrypt or not encrypt the collected data. Encryption is obviously important when dealing with sensitive data, but not all incident response data is sensitive, and you can pay a significant performance tax by taking the time to encrypt the data.

While single computer acquisition speeds are important, what is truly critical is multi-machine acquisition speed. The key factors that affect these speeds are the solution's architecture and the sophistication of the agent. To collect from a large number of machines at a single time without suffering significant performance degradation, that solution must support a distributed architecture. Meaning not only must the solution have agents that can effectively operate independently, collecting data without the support of the examiner's machine, but the solution must also have an intermediary level of distributed workers that can interface with and manage both the agents and the data they send back. No matter how powerful the agents are, an examiner's machine will quickly be overcome by the data being sent back and by the task of managing the agents. The architecture that we are suggesting is illustrated in the simple figure below and frankly is not uncommon among software applications, although as of yet it is not supported by many incident response solutions.

FIGURE 1: Intelligent agents identifying and collecting data without examiner machine support and using intermediary workers to manage those agents and the collected data.



It is worth noting that this architecture is less important if your incident response process deals only with volatile data. When dealing with volatile data, the amount of data gathered is generally small on a per computer basis. With only a little amount of data to gather having an intermediary staging point is of less value. Of course that dynamic is rapidly changing as the size of memory on average computers is literally exploding. The architecture is also overkill if your incident response process is effectively mobile and the responders actually travel to the affected network to deal with the issue locally. When that process is employed, the whole premise of a collection worker is removed, making the additional infrastructure overly cumbersome relative to the use case.

As Figure 1 shows this architecture allows the examiner to send commands to collection workers that in turn distribute that information to the agents they manage. The collection workers then act as an intermediary for collected or responsive data before it is sent back to the examiner. Utilizing this type of architecture, a single investigator can conduct thorough and broad investigations of thousands, even hundreds of thousands of computers with relatively little performance degradation.

Processing Speed

Once the target data has been collected, an effective solution needs to now process that data and put it into usable form. Depending on the case and type of data, that operation can be simple or it can be extremely complex. However, analyzing the processing speeds of various solutions is relatively straight forward. There are two ways to do processing. A solution can process the data entirely in memory or it can utilize a database. The advantage of utilizing a memory-only solution is that it is very fast for small sets of data. The problem with it is that it falls over when the data size begins to grow large. The problem with utilizing a database is that there is a fixed amount of overhead that a database brings to the system, so for very small sets of data processing speeds can be slower than they would with a memory-driven solution. The two advantages that a database solution has are both related to scale. The first is that as the data grows in size the processing speeds grow in a linear fashion, and there is no limit to the amount of data that can be processed. The second advantage is that, assuming the solution is architected correctly, a database solution should be able to utilize distributed processing to split the processing load across multiple computers and thereby reduce the speeds in proportion to the amount of processors being utilized.

It is very important to know whether the solution you are evaluating is memory-driven only or offers database-driven architecture, because attempting to compare a memory-driven solution to a database-driven solution is like comparing apples and oranges.

Processing speeds are extremely sensitive and are radically affected not only by the size of the data, but by the number of items being processed and the types of documents being processed. Processing speeds also vary radically depending on what is being done during processing. For example, simple stream analysis is quick, so solutions that only perform that operation generate great speeds, while tools that provide the option of building an index will be slower when that option is selected, and tools that offer extensive data carving during processing will be even slower when *that* operation is selected. What is ideal is to find a solution that offers the user the flexibility to select the processing options that meet his or her needs on a case by case basis. The option to select indexing and carving can be key in many large cases, but can also be unnecessary overhead for smaller simpler cases.

Given the difficulty associated with comparing processing numbers, the best thing to do is to first understand the methodology being utilized by the system, then look for flexibility in the processing options, and finally and most importantly, drag test different systems against your own data. Only by matching your needs to the capabilities of the solution can you really get what you need.

It should be obvious by this point that when discussing incident response solutions, determining the speed of a product is not nearly as simple as it might otherwise seem. There is no one magic number that allows two systems to be intelligently compared, but rather a number of different factors that affect different types of investigations. That said our hope is that we have armed you with sufficient information to break down the question of speed into the few key categories that truly matter and then analyze the merits of each product you are evaluating, based on its capabilities in those key areas. Remember there is no clear cut answer to the question of investigative speed, but if you are happy with all of the major aspects of a product's data acquisition speed and its processing speed, you will be happy with the solution as a whole.

Flexible Identification and Acquisition Options – Portability or Enterprise Architecture

From an identification and acquisition standpoint AccessData technology was designed from the ground up with flexibility in mind. There are too many use cases and workflows employed throughout the incident response market to constrain a user to a particular acquisition architecture or methodology. As a result, AccessData technology supports all of the acquisition methods discussed in this paper. With our solution the examiner can either utilize distributed collection workers or bypass them and collect from the target node directly. This allows some users to set up lightweight portable systems that the responder can carry from location to location, while others can employ a true enterprise-wide architecture, capable of collecting from nodes anywhere on the network. The solution also supports some very important acquisition features, such as the redirection of acquired data, the ability to control compression and encryption, the ability to stop and restart an acquisition without losing data or having to recollect from the beginning, and the ability to throttle the collection speed to reduce the impact on the network or target machine.

From a speeds and feeds perspective we have performed a number of tests using our solution on our own network and have compiled a fair amount of data. Since this type of information is entirely dependent on a number of factors, including the network, the data being acquired, the target machine and the examiner machine, we are providing it for reference only and would encourage you to test our system on your own network to determine the performance you can expect.

Scaling Out and Scaling Up

Given the enterprise architecture of the AccessData solution, no other investigative product on the market can come close from a performance and flexibility perspective. It is the only solution that can scale out and scale up during identification, collection and processing operations to achieve the speed and investigative reach organizations require. With the distributed worker architecture, flexibility in how given operations are executed and its smart agent technology, the AccessData incident response solution can identify and collect data rapidly and in a very scalable fashion to meet the needs of the most demanding environments.

AccessData Performance Information

The following information is the best and most relevant information we can provide regarding the performance of AccessData's incident response solution. These metrics are based on tests performed with version 2.2 of AccessData's incident response technology. Preliminary testing has shown a further increase in processing speeds when using version 3.0.

Acquisition examples performing full disk acquisition:

| | Acquisition with Encryption | Acquisition w/o Encryption | Volatile |
|-----------------|-----------------------------|----------------------------|--------------|
| Desktop Machine | 12.8 GB/hr | 41 GB/hr | 1 min 38 sec |
| Server | 25.2 GB/hr | 47 GB/hr | 3 min 40 ec |

No performance degradation was seen on these numbers when the collection was scaled to over 100 target nodes simultaneously per collection worker.

Identifying files by a hash:

| Target | Identification and Collection Speed |
|-----------------|-------------------------------------|
| Desktop Machine | 300 files per second |

The scan operation happens concurrently across the nodes, is executed by the workers, and the actual hashing/filtering takes place on the agent. Each worker during a file hash identification operation can handle hundreds of nodes or more at a time with the proper hardware. The solution allows you to scale the workers out and up, which means you can effectively scan 800 nodes simultaneously at a time with a few workers.

Identifying and collecting files by a certain type (exe)

| Target | Identification and Collection Speed |
|-----------------|-------------------------------------|
| Desktop Machine | 8GB per hour |

Again as with the other examples no performance degradation is seen as the number of nodes is increased.

The hardware used in these tests was as follows:

Hardware Specifications

| Desktop Machine | Server | Examiner |
|--------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|
| Windows XP Intel Core 2 Duo P8400@2.26 GHZ 3.45 GB of RAM 149 GB Hard Drive | Windows Server 2003 Standard AMD Opteron Processor 275 @ 2.19 GHZ 4 GB of RAM 640GB RAID 1 (4 Drives) | Windows Server 2003 Enterprise x64 Intel Xeon X5450 @ 3.00 GHZ 8 GB of RAM 68.3 GB RAID 1 (OS) 410 GB RAID 0 Hard Drive (4 Drives) |

Data Processing

While data processing may not be an immediate concern to the traditional incident responder, the term “incident response” means different things to different people, and in many cases, the processing and analysis of hard drive data is required to effectively investigate and remediate a variety of “incidents”. With that said, on the data processing side, AccessData has pursued a database approach. As we discussed, this has some downsides on smaller cases, but the upside in terms of performance and scalability are in our view well worth it. Not only does the solution process large amounts of data extremely quickly, but it fully supports distributed processing, allowing many machines to be utilized on larger cases to ensure the data is made accessible and usable as quickly as possible.

As with acquisition speeds, when building our processing solution we stressed flexibility and allow people to select a wide variety of processing options that radically affect speeds. As the data below shows, the key variables are clearly indexing and carving, and both can be turned on or off. Again this data is meant for reference purposes only, and we would encourage you to test the solution against your own data before making conclusions about absolute processing times.

NOTE: All tests below used the same 120GB image, containing 815,000 items.

Figure 3: Processing times comparing versions 1 and 2, using various systems, as well as various processing options. Preliminary testing of FTK 3.0 has shown that processing times are reduced even more when using version 3.

| Version/ Computer/ Database Location | Product | CPU | # of Core s | RAM | OS | Oracle | Temp Space | Image | Case | Total Time | Total Time | Total Time | Indexing | Carving | Total |
|-------------------------------------------------|---------|------------------------|-------------------|----------|---------------------|--------------------------------------|----------------------------------|------------------------------|-----------------------------------|---------------------------------------------|-----------------------------------|--------------------|----------|---------|-------|
| | | | | | | | | | | Stream Analysis w/ Carving & Indexing | Stream Analysis w/ Indexing | Stream Analysis | | | |
| i7 w/ 12 GB RAM - 5805 8 drive RAID10 | 1.18.3 | Intel i7 | 8 | 12 GB | 10K Raptor | | Eight 15K SAS Drives RAID0 | Two - 10K Raptor | Eight 15K SAS Drives RAID10 | 19.51 | 4.32 | 1.35 | 2.97 | 15.19 | 19.51 |
| Dual Quads 16GB RAM - Single Velociraptor | 1.18.3 | Dual Quad | 8 | 16 GB | 10K Velociraptor | | 10K Velociraptor (Same as OS) | Two 7200 Dynamic Disks | Two 7200 Dynamic Disks | 25.80 | 5.10 | 1.43 | 3.67 | 20.70 | 25.80 |
| Quad w/ 8GB RAM - 5405 4 drive RAID0 | 1.18.3 | Quad Q9450 2.66 Ghz | 4 | 8 GB | 10K Raptor | | Four 7200 Drives RAID0 | Two - 10K Raptor | Four 7200 Drives RAID0 | 28.70 | 5.45 | 1.50 | 3.95 | 23.25 | 28.70 |
| Quad 4GB RAM - Single 7200 | 1.18.3 | Quad Q9450 2.66 Ghz | 4 | 4 GB | 10K Velociraptor | | 10K Velociraptor (Same as OS) | NAS | Single 7200 | 32.50 | 6.30 | 1.90 | 4.40 | 26.20 | 32.50 |
| Core2Duo 3GB RAM - USB 7200 | 1.18.3 | Core2Duo 2.5 Ghz | 2 | 3 GB | Internal 7200 | | Internal 7200 (Same as OS) | USB 7200 | USB 7200 | 38.25 | 8.45 | 2.10 | 6.35 | 29.80 | 38.25 |
| i7 w/ 12 GB RAM - 5805 8 drive RAID10 | 2.2.1 | Intel i7 | 8 | 12 GB | 10K Raptor | 5805 - 8 15K SAS Drives RAID10 | Eight 15K SAS Drives RAID0 | Two - 10K Raptor | Eight 15K SAS Drives RAID10 | 4.27 | 2.77 | 1.10 | 1.67 | 1.50 | 4.27 |
| Dual Quads 16GB RAM - Single Velociraptor | 2.2.1 | Dual Quad | 8 | 16 GB | 10K Velociraptor | 10K Velociraptor (Same as OS) | 10K Velociraptor (Same as OS) | Two 7200 Dynamic Disks | Two 7200 Dynamic Disks | 8.75 | 3.49 | 2.10 | 1.39 | 5.26 | 8.75 |
| Quad w/ 8GB RAM - 5405 4 drive RAID0 | 2.2.1 | Quad Q9450 2.66 Ghz | 4 | 8 GB | 10K Raptor | 5405 - 4 7200 Drives RAID0 | Four 7200 Drives RAID0 | Two - 10K Raptor | Four 7200 Drives RAID0 | 11.29 | 3.27 | 1.55 | 1.72 | 8.02 | 11.29 |
| Quad 4GB RAM - Single 7200 | 2.2.1 | Quad Q9450 2.66 Ghz | 4 | 4 GB | 10K Velociraptor | Single 7200 | 10K Velociraptor (Same as OS) | NAS | Single 7200 | 14.01 | 5.40 | 2.60 | 2.80 | 8.61 | 14.01 |

| | | | | | | | | | | | | | | | |
|-------------------------------------|-------|---------------------|---|------|------------------|---------------|-------------------------------|----------|----------|-------|------|------|------|-------|-------|
| Core2Duo 3GB RAM - Internal 7200 | 2.2.1 | Core2Duo 2.5 Ghz | 2 | 3 GB | Internal 7200 | Internal 7200 | Internal 7200 (Same as OS) | USB 7200 | USB 7200 | 18.75 | 6.20 | 3.60 | 2.60 | 12.55 | 18.75 |
|-------------------------------------|-------|---------------------|---|------|------------------|---------------|-------------------------------|----------|----------|-------|------|------|------|-------|-------|

Figure 3a: Visual of the above metrics, showing the processing time in hours for various hardware systems, performing various processing functions.

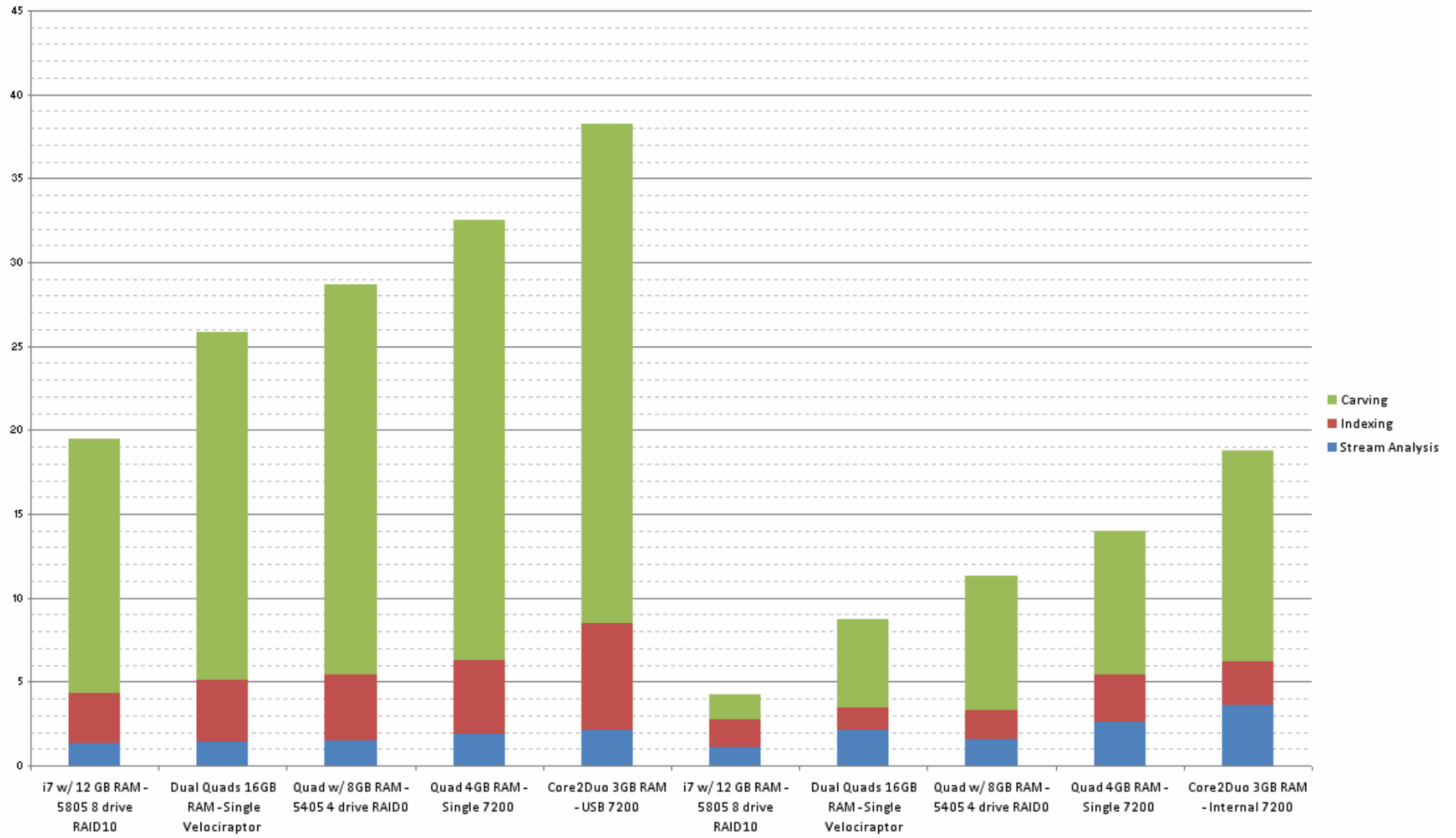


Figure 4: The following chart illustrates performance time (hrs:min:sec) using ideal system specifications and utilizing various processing options, including simply acquiring data in Field Mode with no advanced processing.

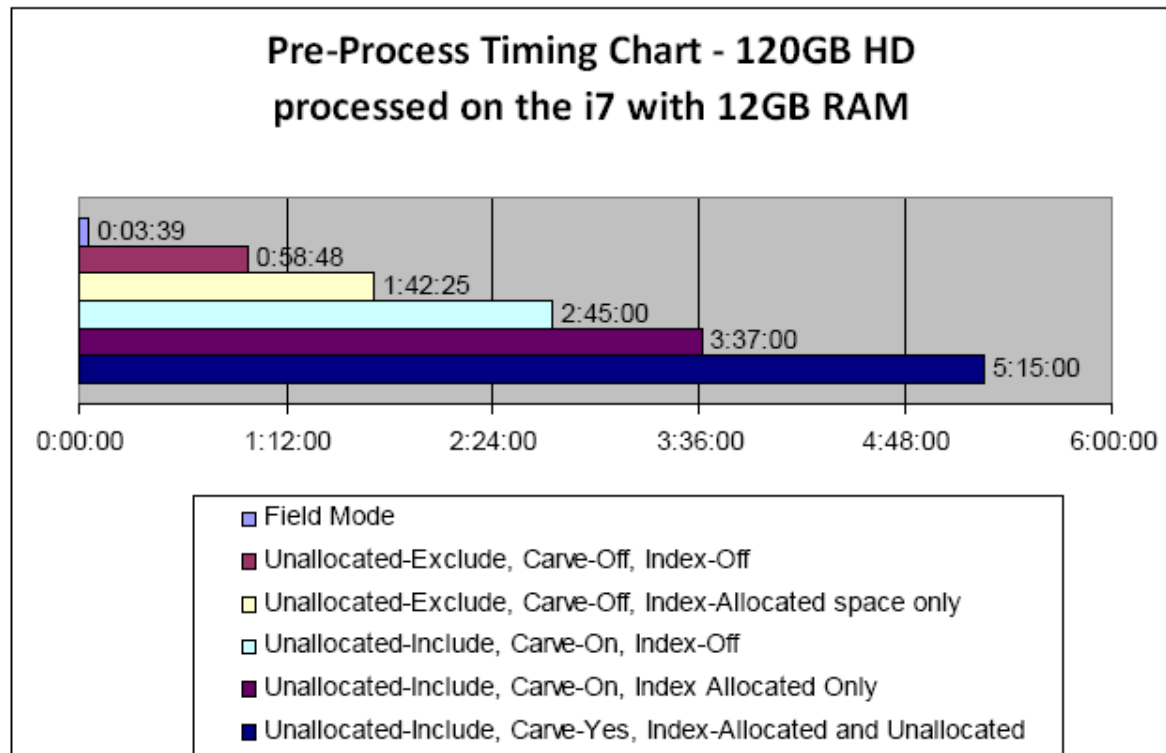


Figure 2 metrics illustrate the impact various processing options have when using ideal system specifications (Intel i7 processor with 12 GB of RAM).

