CHAPTER-2

LITERATURE REVIEW

2.1 Computer as a Digital Forensics

The investigation of digital devices such as portable music players and mobile phones are common part of any digital forensic investigation process. These devices can store gigabytes of data, and in most cases can be used to store and transfer uninformed binary data. Though, their operating systems are becoming increasingly secure [10][11] and internal memory is frequently attached directly to the chipsets, which can prevent analysts from obtaining low-level copies of data without untrusted processes which can physically extinguish the evidence [12].

Digital forensic includes mostly digital equipment like Mobile Phone, Computer System, CD (Compact Disc), DVD(Digital Versatile Disc) etc.

Computer is mainly used to commit cyber-criminal activities because of its ease and compactness.

Computer system supports many user friendly software tools for criminal activities. Hence, most of the cyber-crime activist uses computer as a weapon to commit crimes.

2.1.1 Reasons to acquire Digital Forensic Evidences

There are various reasons to acquire digital forensic evidences from computer systems. Few mentioned below
- Deleted files cannot be securely deleted. Recover those deleted files and the date and time when it was deleted.
- Rename the files to avoid detection is useless.
- Disk Formatting may not wipe all data.
- Web-based messages can be somewhat recouped straightforwardly from a PC framework.
- Files transferred over a secured network can be reconstructed and used as evidence.
- Uninstallation of various applications may leave footprints in the computer system.
- Anti-forensics software may breach the privacy. Hence, those tools are mostly broken.
- Volatile data can be recovered if power supply is still not discontinued.
- Data remnants can be fetched from previously executed programs.
- Encryption may also useless if it would not be decrypted. Hence, at the end of process plain text data must be available.
As deleted files cannot be permanently wipe out from physical storage devices, if it is not forcefully overwritten. It is also possible to recover deleted files from storage devices. So, criminals cannot destroy their all the evidences. Renaming of file to hide or destroy evidences is not a smart task because it is easily caught by an investigator using some technologies.

Formatting of storage media is also not possible to destroy data completely. Data can be recovered successfully from it even if it is formatted.

Web-based emails can also be partially recovered directly from a computer system using some latest software tools and technologies. Transferred files or data can be resembled from a network and evidence can be identified. Criminals use software tools to commit crime and uninstall that software after committing a crime. If software programs completely uninstalled successfully, still data remnants can be recovered from computer system.

To hide criminal activities from forensic investigation process, some criminals use Anti-forensics tools that may breach privacy of users. Hence, they are reluctant to use such anti-forensic tools. Raw data can also be used, created during
execution of applications. To hide criminal activities, data encryption techniques may also be used. But if they want to use that data decryption must take place. So, it is very hard to kill the data from a computer system.

2.2 Electronic Evidence Processing Guidelines

Various risks and hazards exist identified with the processing and handling of computer evidence, as it varies by nature, it is entirely friable and can conceivably be easily destroyed, even by the computer systems’ simple operation. Subsequently, evidence preparation rules must be pursued to guarantee that no sensitive data is decimated. Of at-most significance are the evidences that are covered up in the slack space of record or the unallocated space or likewise in the Windows swap file.

The accompanying evidence processing guidelines were created to reveal electronic evidence in a forensically thorough way and are put together and extend with respect to the United States Department of Justice search and seizure guideline, Searching and Seizing Computers and Obtaining Electronic Evidence.

Step 1: Turn off the Computer

Turning off the computer system generally consist of directly remove the computer's plug from the electrical outlet or utilizing applicable commands to shut down a network computer. If the computer is already shut down, the forensic investigator may choose to take snapshots of the screen image at his or her decision. It is significant for the advanced forensic investigator to think about that the PC may have possibly harming procedures running out of sight in the memory. In addition, depending on the operating system installed, it may be possible that a password protected screen saver appear, possibly making it more difficult to turn off the computer. Time is a critical factor in the midst of the shutdown process and forensic analysts must be careful
in order to perform this task as quick and beneficially as could be the normal shutdown process being done.

Step 2: Execute Chain of Custody

An appropriate chain of custody must be pursued, that includes formation of evidence marks and starting a detailed list of people who is the accountable for evidence anytime, from its extraction to its conclusion. When the chain of custody has been finished, the computer system can be transferred to a protected area, where the genuine evidence could be handled. Snapshots of the computer screen and physical devices from various angles should be taken to prepare formal document of the system hardware components and their connections. All these things should be accomplished before the computer is actually dismantled. Furthermore, each wire should be properly tagged, so that it would be easily reconnected after the system configuration has been re-established.

Step 3: Make Bit Stream Backups

To be careful any computer evidence is important to the legal examination process. Consequently, the computer system ought not to be turn on and evidences ought not to be prepared until bit stream backups have been made of all storage media connected to the computers. Bit stream backups are used in a way like a protection approach and are important to process any computer evidence. All proof ought to be prepared on a restored copy of the bit stream backup rather than the original computer system, and the essential evidence should stay flawless except if irrefutable conditions exist.

Step 4: Mathematically Authenticate Data

The mathematical validation process based on the utilization of an MD5 sum is employed to demonstrate that any evidence has not been modified after the PC
has gone under the custody of the forensic investigator. And also original bit stream backup was made. This process is employed for authentication purpose.

Step 5: Document the System Date & Time

The accurate dates and times associated with computer files are often plays vital role in forensic investigation process. It is an essential part of investigation process, to document the system date and time since the system is taken into consideration. Hence, file time stamps will reflect the same time as the system clock.

Step 6: Make a List of Key Word Search

It is unlikely for forensic investigators to physically decide and see the significance of each registry/record of Computer hard disk. As most computer hard disk drives are very bigger. Therefore, forensic examiners may utilize mechanized content search tools to reveal important evidences dependent on data gathered from people associated with the case. Proper list of keywords should be made in order to perform text searching task. These keywords are utilized to look through the whole storage media.

Step 7: Evaluate the Windows Swap File

The Windows Swap File regularly assumes crucial job of evidence and potential leads and its assessment is frequently automated using explicit forensic tools. In any case, while operating system like Windows 95 or 98 is working on the computer being referred to, the swap document may set to be powerfully made as the PC is worked, and the swap record is deleted when the PC is turned off.

Step 8: Evaluate File Slack Space
Most PC clients don't know about File Slack Space, an data storage area that frequently contains significant amount of data, including raw memory dumps that happen as files are closed all through work sessions. Any information dumped from the PC's memory is put away towards the end of allocated files, which is normally not known to the PC user. From time to time, forensic tools are important to inspect and analyse File Slack Space, as it can give important data and extra analytical pieces of information. It is important to assess the File Slack Space, as it might have applicable keywords to supplement the recently distinguished identified keywords.

Step 9: Evaluate Unallocated Space (Erased Files)

Typically, when a computer user deletes a stored file, user accept that the file has been altogether deleted from storage media. However, the DOS and Windows "delete" function does not altogether delete either record names or document content from storage media. Rather, the extra room related with such files are unallocated and accessible to be overwritten with new records or files. Such unallocated space is frequently a noteworthy source of significant data that incorporates deleted data and File Slack Space related with these files that have been "deleted". Besides, usually conceivable to use the DOS Undelete program to restore the prior deleted file, gives increasingly applicable keywords and leads.

Step 10: Search Files, File Slack Space, & Unallocated Space

Recently recognized keywords ought to be used to explore all speculate PC hard disk drives and floppy disks and other storage media. At the point when noteworthy evidence is distinguished as a result of keyword search, it ought to be reported. The recognized information ought to be totally dissected to uncover extra keywords. Besides, when the new keywords are recognized because of the investigation, it ought to be added to the total keyword list and another pursuit ought to be directed.

Step 11: Document File Names, Dates, & Times
Record names, last changed dates, creation dates and times are colossally significant to forensic investigation process. Moreover, they ought to be appropriately recorded for all allocated and revealed "deleted" documents.

Step 12: Identify File, Program, & Storage Inconsistencies

Compressed, Encrypted and graphical files store data in a binary format and, thus, a text search program can't distinguish text stored in such files. Subsequently, these files must be assessed manually. Many software forensic tools can distinguish the graphical file formats and archive document formats, which can be utilized to distinguish files that require extra extensive manual assessment.

The seized hard disk drives might be apportioned in different volumes that may have hidden partitions. These hidden partitions may be formatted using various software tools. At the point when hidden partitions are found, they ought to be investigated for evidences and their existence ought to be reported. Besides, the records contained in the Recycle Bin ought to likewise be assessed. The Recycle Bin is a registry of records that the computer user has erased. On the off chance that applicable records are discovered, it ought to be completely reported.

Step 13: Evaluate Program Functionality

Depending on the various application software involved, it might be important to run certain programs to get familiar with their purpose. Subsequently, unsafe procedures might be revealed that are associated with pertinent evidences, which can frequently demonstrate the fact. These destructive procedures can be associated to the execution of normal working directions that are associated with the operating system or applications being used by the computer system.

Step 14: Document Your Findings
All discoveries ought to be archived, as should the majority of the product and its form numbers used over the span of the forensic investigation.

### 2.3 Digital Forensic Investigations

Digital forensic investigation incorporates extraction and investigation of evidences. This procedure is separated into two classifications. Traditional forensic investigation includes post-hoc duplication and examination of storage media to remove evidence and reason about various events after a computer system has been turned off [11]. Investigators can keep up the integrity of the data and confirm their outcomes by performing repeatable activities on copies of the original storage medium while keeping up a record of the cryptographic hash (for example MD5, SHA) of documents [12].

When a system cannot be powered off, because of legal, technical, or other reasons, analysts must perform a live forensic analysis. It has been shown that different results can be obtained from live analysis than traditional forensic evidence extraction [13]. Volatile information such as open network connections, lists of running processes, and more recently keys from full-drive encryption software can be recovered using live analysis [14], but would be lost if the system was powered down. However, there are inherent risks to live forensic analysis. Results for operations such as listing the contents of a folder or displaying running processes could be modified by malicious or malfunctioning code intercepting system calls. Whilst hardware solutions have been suggested, Carrier [15] notes that better operating system design and virtualisation can alleviate the problem. Regardless of whether the examiner is taking a traditional or live approach, acquiring and analysing evidence in a forensically sound manner is paramount to the success of an investigation and the acceptance of evidence in court.

To be considered forensically sound, processes must meet the following criteria [16]:

---

32
• The meaning of electronic evidence should not be altered by the process.
• Any errors should be identified and “satisfactorily explained.”
• Processes should be available for, and stand up to, independent verification.
• Analysts should have sufficient and relevant experience.

Casey [17] states that a strict way to deal with forensic soundness, where any modification to prove alteration it prohibited, is unhelpful. When performing the live investigation, the condition of a computer system is fundamentally changed, and this is steady with other forensic domains.

2.3.1 Traditional Evidence Extraction

File hashing, which has long been used to identify the integrity of digital evidence throughout the chain of custody, coupled with file fingerprinting allow us to rapidly identify the presence of known files on subject media. Recent research has identified mechanisms that significantly speed up the operations behind hashing and fingerprinting techniques, reducing the memory footprint while still maintaining a high level of accuracy. Roussev [19] applies bloom filters to file hashing and fingerprinting. Haggerty [18] approaches the problem differently, determining that selecting sixteen points from a random block of a set of sample files is sufficient to determine their presence on a hard drive by ignoring the file system and searching the binary data directly. This is an efficient alternative to using whole or partial file hashing, and makes it significantly harder for suspects to disguise files by altering the data. However, as with cryptographic hashes, if a file has been resized or encoded in a different format it is unlikely that a comparison can be made. There is always the possibility of entirely unexpected malfunctions in techniques that were previously presumed to be correct. Byers and Shahmehri [20] discuss one such flaw that affects drive imaging tools on Linux, where one faulty sector on a drive causes a failure to acquire a series of adjacent
sectors. They conducted their experiments using known good drives, on which they simulated an error using an ATA command, and received identical results for all the imaging tools they tested. An analysis showed that Linux always tries to read an entire memory page of sectors, and if any sector in the page fails to read, it will fail to read the page. On 32-bit systems this means a run of eight 512-byte sectors will not be read, and on future platforms the failure will have a bigger impact. Their solutions range from optimal, rewriting the software to use direct I/O, to hacks involving hooking system calls for specific applications and forcing them to open all files in direct mode.

2.3.2 Live Evidence Extraction

The Primary Memory or Random Access Memory (RAM) also called as a main or real memory is typically considered to be as a volatile resource. The data residing on the memory is less trustworthy as power is dissuaded from it. Data stored in RAM persists temporarily if power is not revitalized [21]. Few factors that affect the persistence of data in memory [22] include:

- System Activity
- Size of Main memory
- Data Type
- Operating System

Both operating system kernel and user-space applications utilize this type of capacity to store data and execution codes. Since all executions leave leftovers on the primary memory, there is a lot of in-memory data at any time. Examining a snapshot of memory can open or recreate the condition of a computer system when the memory snapshot was captured. This does excludes just current running processes but rather likewise terminated processes as the segments of memory utilized by them (the terminated processes) won't be removed when these bits are never again required for current operation. [23]
The digital forensic investigators’ will likely to find digital evidences that proves committed crime. There are different kinds of digital crimes in the computing environment. Each electronic crime has a few likely digital evidences. For instance, the sorts of digital evidences related with identity theft incorporate Inventory Management Software, Banking Records, forged documents and website transaction records (On the Scene Reference for First Responders, November 2009).

The detailed description of the structure comprises of a depiction of the logical and physical structures. The depiction of the logical structure contains investigation of five fundamental registry keys, as those keys can be gotten to utilizing windows registry editors in all windows operating systems. The description of the physical structure incorporates where and how registry hives are stored in the physical memory.

Throughout the years since Microsoft deployed the Windows Registry in their operating systems, it has turned out to be certain that it contains important data for the legal forensic examiners (Carvey and Kleiman, 2007). Carvey said that "Realizing where to search inside the registry, and how to decipher what you find, will go far towards giving you knowledge into activity that happened on the system". He dissected the Registry structure of Windows XP and worked admirably of the examination of the registry structure inside the hive files in physical memory. He likewise gave helpful data about the signature of hive files in memory. These signatures can be utilized by a crime scene investigation person to cut registry keys and their values from the unallocated space of a image or from a dump of the RAM. The fundamental center is the registry analysis and the extensive amount of significant information that are recognized for the crime scene investigation examiner inside the Windows Registry. For instance time zone data, wireless SSIDs, audit policy, locations of auto-start programs, mounted devices, and user activities.

Thomas and Marris said that "When a USB flash drive is connected to a Windows XP computer system; various registry settings and log records are
automatically updated for the utilization of the USB flash drive” (Thomas and Marris, 2008). The purpose behind their work was to fathom information that perceives a USB flash drive that has been used in the PC and to recognize where the crime scene examination person should want to acquire this digital evidence.

Traditional forensic tools assemble evidence from persistent storage devices, for example, hard drives. Interestingly, more up to date forensic tools additionally gather transient evidence from the memory of a running PC. Tools, for example, Volatility deconstruct and decode the raw memory dumps and quest for evidence of intrigue. While these tools can discover evidence, for example, the procedure list, open network sockets and open files, which are straightforwardly identified with the running programs, they are regularly unfit to give profound semantic knowledge into the internal operations of the running programs. Without the utilization of tedious manual examination or explicitly developed tools, the forensic investigator can't briefly get to or translate the entirety of the significant evidence. Current strategies to concentrate evidence from these memory dumps incorporate the utilization of debuggers, quick file carving methods and raw pattern recognition tools, for example, grep, strings. Entropy investigation tools are likewise used to recognize scrambled information. A few tools, for example, KOP can give an item guide of kernel objects, however intermittently they need expert information and just deal with static memory dumps.

2.4 Volatile Data on Running Computers can Provide Crucial Evidence

Computers necessitate that a specific measure of computer memory called "Random Access Memory" (RAM) be utilized by the operating system and its applications when the computer is in activity. The computer uses this RAM to write the running processes it is utilizing as a type of a virtual clipboard. The data is there for prompt reference and use by the process. This sort of information is classified "volatile
information” since it just leaves and is inaccessible when the computer is off. Volatile information put away in the RAM can contain data important to the investigator. This data could incorporate as appeared as follows:

- Processes running on workstation.
- Commands executed on console.
- Passwords shown in readable text.
- Data that in unencrypted.
- Instant messages (IMs).
- Internet Protocol addresses.
- Presence of a Trojan Horse.

There are different sorts of unpredictable information that could be mulled over for evidence important to an examination process. This conceivably exculpatory data may likewise basically "go away" when the system is killed or loses the power. This kind of volatile data as potential evidence can in like manner be accumulated from a running Microsoft Windows PC. The extra information that can be gathered may include:

- Who is signed into the workstation.
- Open ports and listening applications.
- List present running procedures.
- Registry data.
- System data.
- Devices connected to workstation.

2.5 Extracting persistent data from Windows Registry

Windows registry keys can be used for the forensic purpose. The accompanying area features a portion of the significant registry keys of the Windows
XP (Service Pack 2) operating system by Microsoft and its use in scientific examination process.

HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\OpenSaveMRU

MRU is the abbreviation for “Most Recently Used”. This key keeps up a list of as of recently opened or saved records via typical Windows Explorer-style common dialog boxes (i.e. Open dialog box and Save dialog box) (Microsoft, 2002). For instance, files (for example .txt, .htm, .pdf, .jpg) that are as of recently opened or saved from an internet browser (counting IE-Internet Explorer and Firefox) are maintained. However, documents that are opened or saved by means of Microsoft Office programs are not kept up.

Subkey * contains the full path to the 10 most as of recently opened/saved records. Different subkeys in OpenSaveMRU contain more sections identified with recently opened or saved records (counting the 10 latest ones), which are assembled in like manner to their extensions.

HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\LastVisitedMRU

This key gives data about the past OpenSaveMRU key to give additional data. At the point when another entry is added to the past OpenSaveMRU key, library value is made or refreshed in this key. Every binary registry value under this key contains an as of recently used program executable filename and the path of a file to which the program has been utilized to open or save it. In the event that a file is saved, the file path alludes to the saved file destination path. In the event that a file is opened, the folder path alludes to the file source path. New registry value might be made to this key, if no current registry values contain the program executable filename. Be that as it
may, if there is a relevant executable filename in the current values, just the folder path area of the related registry value is refreshed.

HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs

This key likewise keeps up list of documents as of recently executed or opened through Windows Explorer. This key compares to %USERPROFILE%\Recent (My Recent Documents). The key contains local or remote files that are as of recently opened and stores the filename in binary format. It has similar grouping as the past OpenSaveMRU key, opened records are sorted out as per file extension under individual subkeys. Moreover, the Subkey Folder contains the folder (without drive letter and parent folder) of the as of recently open records. Subkey NetHood which relates to %USERPROFILE%\NetHood, contains just LAN shared folder folder path(folder and server name) which the file was accessed or opened. Be that as it may, erasing this RentDocs key does not evacuated the substance in the two folders %USERPROFILE%\Recent and %USERPROFILE%\NetHood.

HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\RunMRU

This key keeps up a list of sections (for example full file path or commands like cmd, regedit, compmgmt.msc) executed utilizing the Start>Run commands. The MRUList value keeps up a list of letters in order which allude to the respective values. The letters in are written according to the order of the entries is being added, services.msc, which connects to e.g. is the most as of most recently added entry, while .taskmgr, contains the most latest entries. Be that as it may, most as of recently added entries does not conclude about committed crimes. Because, suspects may have re-executed past commands which are not related to any crime. Windows does not alter the key LastWrite time or MRUList if there is a current entry in the key. In the event that a record is executed through Run command (Windows utility program), it will leaves the artefacts in the past two keys OpenSaveMRU and RecentDocs. Erasing the subkeys in RunMRU does not evacuate the history list in Run command box right away.
Notwithstanding, when either button Start>Log Off or Turn Off Computer is clicked (without really logging off or shutdown), the respective entries in Run history list are then expelled. By utilizing Windows .Recent Opened Documents, Clear List highlight by means of Control Panel>Taskbar and Start Menu, suspect can expel the Run command history list. Truth be told, executing the Clear List capacity will expel the accompanying registry keys and their subkeys:

- HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs\n- HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\RunMRU\n- HKCU\Software\Microsoft\Internet Explorer\TypedURLs\n- HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\OpenSaveMRU\n- HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\LastVisitedMRU

2.6 Digital Forensics Investigation Methodology

This section introduces the traditional techniques used in digital forensics investigations process. Also, focuses on few limitations of existing methodology of digital forensic investigation process. Various popular digital forensics tools are used in digital forensics investigations process, though have certain limitations to analyse evidences.

2.6.1 Existing Methodology

Existing techniques for cyber forensics investigation is as follows:
Figure 2.2 Existing Methodology of Cyber Forensics Investigation

Steps involves are mentioned below:

Acquire Image: Bit-by-bit duplication is performed on storage media. Various digital forensics tools are available in market like dd, FTK Imager, Encase etc. Few tools use proprietary format which cannot be further used.

Authenticate: Mathematical calculation is applied to verify the image copy of hard drive. There are various hashing techniques such as md5, RSA are available to generate hash value that could be used to authenticate the original evidence.

Analyze: Image copy of hard drive is to be further analysed in the forensic lab using various forensic tools and cyber forensics investigator discovers evidences to prove committed crime.

Documentation: Making reports about how cyber forensics investigation is in progress and how the evidences are handled (Chain of Custody). Documentation also
specifies description about digital forensics evidences which are extracted by an investigator.

2.6.2 Some Popular Cyber Forensics tools

- Vinetto
- Pasco
- Encase
- FTK Access Data
- The Sleuth Kit(TSK)
- Nmap
- Helix
  - Advanced Registry Tracer[28]
  - Windows Registry Analyzer[29]
- The Volatility Framework: Volatile memory artifact extraction utility framework[30]
- Computer Online Forensic Evidence Extractor (COFEE)[31]
- Memoryze™[32]
- The Pmem Memory acquisition suite[33]
- Windows Memory Reader™[34]
- Paraben[26]
- Dd[35]
- Safe Back Version 2.0[36]
- SnapBack Dat Arrest Version 4.12
- Portable Evidence Recovery Unit
- Rapid Action Imaging Device[26,27]

2.6.3 Limitations of Existing Popular Cyber Forensics tools

42
**Vinetto**: Vinetto is a criminology investigation device which extricates the thumbnails and related metadata from the Thumbs.db files.

**Disadvantages**: It can just concentrates image files not different evidences of the computer system.

**Pasco**: Forensic tool for reading the index.dat file that are created by Internet Explorer.

**Disadvantages**: It can only extract Browsing history from Internet Explorer, not from other from Browsers.

**Encase**: It is a multipurpose forensic tool

**Disadvantages**: File is acquired in a proprietary format.

**Forensic ToolKit**: Multi-purpose device, usually used to list obtained media.

**Disadvantages**: It is disk imaging tool which breaks down the computer system and gathers information however it will require a time to collect all data from the devices.

**The Sleuth Kit(TSK)**: It is used only to investigate disk images not on original copy. It is a library and collection of command line tools that allow you to investigate disk images.

**Disadvantages**: It is utilized just to research disk images not on original evidence. On the off chance, multiple images with higher capacity of storage media will waste the time of investigator.
**NMap**: NMap is particularly associated with network security. NMap is a port scanner tool that helps find open ports on a remote machine. What separates NMap from other tools is its ability to evade source machine identity and to work without causing any Intrusion Detection System (IDS) alarms to go off.

**Disadvantages**:  
It can only scans through network based environment not about computer system and its file system.

It is only used for scanning purpose. Like Port Scan, OS Detection etc.

**Helix**: Freely downloadable as ISO file and is a CD based forensics tool.

**Disadvantages**:  
It does not auto-mount disks.

It mounts disks read-only to preserve evidence integrity.

**F.I.R.E: Forensics Incident Response Environment**  
It is bootable Linux-based CD.

It contains wide variety of forensics tools.

Capabilities include forensics, incident response, and vulnerability assessments.
Disadvantages: Need to analyse each and every computer system in order to find out suspicious computer.

LADS: List Alternate Data Stream

ADS is an advance feature of NTFS file systems where a user hide their file which is going to be undetected.

LADS is a forensic tool which list out those data which is hidden by user.

Disadvantages: Only analyse ADS of NTFS file system.

Log Parser: Microsoft depicts Logparser as an incredible, adaptable instrument that gives universal query access to text-based data, for example, log documents, XML files and CSV records, just as key information sources on the Windows operating system, for example, the File System, the Event Log, the Registry, and Active Directory.

Disadvantages: It only analyse IIS log file and produces output in different format. For extracting other evidences need another technique.

Limitations of Existing Methodology of Cyber Forensics Investigations

- Need Higher Storage media to perform backup of evidence.
- Requires more time to acquire image copy of storage media.
- Harder to examine higher capacity of storage media.
- Need more labor to analyze captured data
- Harder to keep up the privacy and secrecy of the data.
- Harder to handle physically extracted and related evidences because of high cost of preservation.
2.7 Work done on existing methodology

"First Responders Guide to Computer Forensics" By Richard Nolan, Colin O'Sullivan, Jake Branson, Cal Waits, This paper clarifies in this time of interconnected systems, in any case, the examination and gathering of volatile information is winding up increasingly significant. Volatile information is any information that is put away in random access memory; it will be lost when the PC loses control or is controlled off. Volatile information lives in cache, registries and random access memory (RAM). Since the nature of volatile information is “go away”, accumulation of this data will probably need to happen in forensic examination process. [37]

"Techniques and Tools for Recovering and Analyzing Data from Volatile Memory" By Kristine Amari Adviser: Carlos Cid in 26 March 2009, This Paper centers around, there are numerous moderately most recent developed tools to recuperate and dismember the data that can be accumulated from volatile memory. But since this is a generally new and quickly developing field many forensic investigators don't have the precise idea or get advantage of these resources. Volatile memory may contain many leftovers of data appropriate to a forensic examination, for example, cryptographic keys, passwords and other applicable valuable information. The knowledge and forensic tools expected to recuperate the forensic information is essential, and this ability is ending up progressively, increasingly important as hard drive encryption and other security mechanisms make traditional hard disk forensic additionally challenging. This paper has secured the hypothesis behind volatile memory examination; including reason of its significance, what sorts of information can be recouped. [38]

"Extracting Windows command line details from physical memory" By Richard M. Stevens, Eoghan Casey in Elsevier 2010, this paper analyzes the data structures of the command prompt history and gives forensic experts a tool for
recreating the Windows command history from a Windows XP memory dump. Simultaneously, it also exhibits a methodology that can be summed up to concentrate user-entered information on different products of Windows [39].

"Analyzing Multiple Logs for Forensic Evidence" By Ali Reza Arasteh, Mourad Debbabi, Assaad Sakha,and Mohamed Saleh, The Digital Forensic Research Conference DFRWS 2007 USA". This paper proposed a model checking approach to the formalization of the forensic examination of logs. A lot of logs are displayed as a tree whose names are events extracted from the logs. So as to give a structure to these events, they expressed every event as a term of algebra. The signature of the algebra is painstakingly picked to incorporate all significant data important to lead the investigation. Properties of the model, attack situations, and event sequences are communicated as equations of a logic having dynamic, linear, worldly, and modular qualities. Additionally, they have given a tableau-based proof system for this logic whereupon a model checking calculation can be created. Also, utilized proposed model in the case study to show how events driving to a SYN attack can be reproduced from various system logs. [39]

"The Value of Metadata in Digital Forensics" by Fahad Alanazi, Software Technology Research Laboratory, De Montfort University, Leicester LE19BH, UK and Andrew Jones, Cyber Security Center, Center for Computing and Social Responsibilty, De Montfort University, Edith Cowan University, Leicester LE19BH, UK, Perth, Australia, 2015 European Intelligence and Security Informatics Conference." It centers on adequacy of evidence metadata for advanced forensic examination process. These evidences are not in every case promptly presumptive; subsequently, examiners must guarantee the accumulation of such information and approval of it in court. This paper significantly centers on possibility of metadata for prosecution process [40].

“Automating Disk Forensic Processing with SleuthKit, XML and Python” by Garfinkel, Simson L., IEEE/SADFE 2009”. In his paper, more focus given on data to
be extracted from a computer system. Researcher has not focused on information that could be extracted and can be helpful. [42]

“Impacts of increasing volume of digital forensic data: A survey and future research challenges” By Darren Quick*, Kim-Kwang Raymond Choo, elsevier 2014 have compared various digital forensic models based on their characteristics. [41]
2.8 Digital Forensic Models comparison

<table>
<thead>
<tr>
<th>Year</th>
<th>Author</th>
<th>Data mining</th>
<th>Data reduction</th>
<th>DFasS</th>
<th>Distributed parallel</th>
<th>Intelligence</th>
<th>Machine learning</th>
<th>Triage</th>
<th>Visualization</th>
<th>Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010</td>
<td>Aghil et al.</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2010</td>
<td>Rihan et al.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2011</td>
<td>Marrison</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2012</td>
<td>Al-Taliby et al.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2012</td>
<td>Rhodjung et al.</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2012</td>
<td>Gherbi et al. 2012a</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2012</td>
<td>Gherbi et al. 2012b</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2012</td>
<td>Jones et al.</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td>Roussel and Quares</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td>Ahmadi et al.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td>Casey et al.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td>Koepmann and James</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td>Martinez and Taccon</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td>Overill et al.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td>Paulin</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td>Quick and Choo</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td>Raghavan</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td>Rosseel et al.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td>Shaw and Browne</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td>Stirling et al.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td>Zimmerman</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2014</td>
<td>Reiteringer and Rosseel</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2014</td>
<td>Reiteringer et al.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2014</td>
<td>Casey</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2014</td>
<td>Noel and Peterson</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2014</td>
<td>Pringle and Burgess</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2014</td>
<td>Quick and Choo</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2014</td>
<td>Quick et al.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2014</td>
<td>van Staal et al.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2014</td>
<td>Vidar et al.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 2.3 Comparison between various digital forensics models

2.9 Importance of MetaData in Digital Forensics

Digital data contains enormous amount of advanced forensic information that would be broken down so as to foresee the carried out offense. All things considered, this monstrous amount of digital forensic information has metadata that is known as information about information, which could be utilized to find the facts. For instance, a record/document has different properties like name of document, size of document, document owner; authorizations of document, shared files and so on. Henceforth, this metadata could be utilized that decrease the forensic investigators’ efforts to demonstrate carried out crime.
Fahad Alanazi and Andrew Jones has distributed an exploration paper on "The Value of Metadata in Digital Forensics", 2015 European Intelligence and Security Informatics Conference to help the reality about how metadata can perform significant job in advanced legal sciences examination process. [40]

Metadata itself implies abundance of data of the advanced forensic information that are being examined. Moreover, the advanced forensic investigator can utilize computerized metadata to extricate data, for instance, the creator of file, the date and time of file creation, the number of time the file has been modified, including when the change took places. Commonly used programs, for example, MS Office and OpenOffice.org contain features like track changes and client remarks that give a way to clients to see the different modifications that have been made to the file. File System Metadata incorporated into the classification of file system metadata is data about allocated data units, record size and access to date or time stamps of the record. File Metadata is contained in media records including, Graphic Interchange Format (GIF), JPEG, and in music: MP3 and AAC and tagged picture document Format. Metadata is additionally incorporated into the archive records, for example, Microsoft office, Open Office, office open XML group (MS Office 2007) and Portable document format. In this manner, metadata can be valuable to determine lawful debates, since it very well may be utilized as proof for the endorsement or dissatisfaction with different evidences set forward in a court case. Furthermore, file metadata can be kept in different destinations inside documents. Agents can amass data from this metadata that connects to proprietorship and potential proprietors [40].

2.10 Accessibility of MetaData

Different scientists have directed research on metadata and its role in digital forensic examination process.
Digital Forensics Tool Integration by Simson L. Garfinkel, This examination paper is concentrating on different procedures of storing digital forensics evidences and limitations of it. So as to take care of issues if storing all digital evidences new methods ought to be required. This system may be help to forensics investigators to perform analysis of digital crime in limited time span. The data which accumulated for examination are stored in various formats, so every single tool might need its proprietary structure for specific forensic data investigation. Hence, there ought to be a standard structure to store all data in specific manner and same format. Securing all those forensics data is the best practice with regards to evidences secure from unauthorized access [66] .

We can utilize any devices to extract digital forensic data and furthermore utilize another forensic examination tools for investigation procedure of extracted evidences. Thus, digital forensic evidences can be dissected by applying different forensics devices. Here, forensic investigators can get preferred standpoint of utilizing proposed approach by guideline of "Extract data once, consume it for all" [62].

Suggested Practice:" Creating Cyber Forensics Plans for Control Systems by Mark Fabro1 Eric Cornelius.” Cyber forensics has been in the well-known standard for quite a while, and has developed into an information-technology capability that is normal among present day information security programs. The objective of cyber forensics is to assist the components of investigating, checking, recuperation, and the security of sensitive information. In addition, in the event of a crime being committed, cyber forensics is additionally the way to deal with gathering, investigating, and documenting information as evidence in a court of law. Although adaptable to many information technology areas, particularly present day corporate structures, cyber forensics can be challenging when being applied to non-traditional situations, which are not included current information technologies or are planned with technologies that don't give sufficient information storage or audit abilities. Moreover, further complexity has been applied if the forensic environments are designed using proprietary solutions
and protocols, thus limiting the ease of which modern forensic methods can be utilized [67].

“Lawful Aspects of Collecting and Preserving Computer Forensic Evidence” by Franklin Witter. This exploration paper focuses on forensic evidences of computer systems. Different phases of forensic examination process incorporate Preservation, investigation, and collecting digital evidence would require suitable techniques to demonstrate digital evidences in court. This exploration paper discuss on best practices to deal with these digital evidences [68].

“Selecting storage media for long-term preservation”, Author: Adrian Brown, Head of Digital Preservation Research, Issue Date: August 2008, This paper gives information to the managers and administrators of digital evidence records about the determination of physical storage media with regards to long term preservations. The extent of this exploration is limited to removable storage media. There are numerous concerns to be viewed as while choosing physical storage media that delay past the quick and clear prerequisites of the situation. This exploration should help data architects to settle on knowledgeable choices dependent on the regularly changing decisions available [69].

Use of AFF4 “Chain of custody”- Methodology for Foolproof Computer Forensics Operation NITHESH K.NANDHAKUMAR, UJJWAL AGARWAL, FAIZAL H. IT department, Salalah Colege of Technology, This paper published in International Journal of Communication and Networking System Volume: 01 Issue: 01 January- June 2012, JUN_12_IJCNES_006, This paper studies how AFF4 can be extended to accommodate the new challenges faced during digital evidence custodial processes. In this respect, the use of new standards of Advance Forensic format (AFF) for Forensic analysis is suggested that comprises of evidence acquisition, handling and decision making. AFF4 implementation is easier to design, as it supports diverse specifications including the digital evidence management framework (DEMF). AFF does not satisfy all required data structure for some kind of attacks like data theft using
any digital device. AFF was proposed by Garfinkel that was defined format using a single archived file along with its container. They have studied on AFF due to its open source and exclusive format, although the numbers of applications are not limited to this only. The investigation analysis incorporates the advanced AFF4 which will enable us to work on the problem at hand in a distributed manner [70].

Digital Forensics XML and the DFXML Toolset By. Simson Garfinkel, at Naval Postgraduate School, 900 N. Glebe, Arlington, VA 22203, This paper discuss Digital Forensics XML (DFXML), a XML language for digital forensics look into on digital evidence metadata and exchange of it. DFXML is intended for compatible arrangement between digital forensic software. The abstractions represented in DFXML have been specifically chosen to represent digital forensic processing steps, allowing for ease to generate and ingest DFXML objects. Digital Forensics XML (DFXML) is the XML language intended to signify to a wide scope of forensic data properties and forensic examination results. By coordinating its reflections to the necessity of forensics instruments and examination, DFXML permits the sharing of organized data of digital evidences between free forensic devices and associations. Since the underlying work done in 2007, DFXML has been utilized to store the aftereffects of forensic examination steps, decrease the prerequisite for re-preparing of digital evidences, and as a compatible organization, permitting marked digital forensic data to be disseminated between research partners [61].

2.11 Various Standards for MetaData

There are various standards to store digital forensics metadata. It includes AFF(Advance Forensic Format) and DFXML(Digital Forensic Format).

2.11.1 Advanced Forensics Format (AFF)
The Advanced Forensics Format (AFF) is an extensible open configuration to store disk images and related forensic metadata. It was created by Simson Garfinkel [71]. This organization can store different kinds of forensic information contains Disk images and Exported files. AFF is an open and extensible file format to store disk images and related forensic evidence metadata. Utilizing AFF, the user isn't secured in a proprietary format that may restrict how the legal person may investigate it. An open standard empowers investigators to rapidly and proficiently utilize their favoured tools to illuminate crimes, accumulate knowledge, and resolve security occurrences.

Utilization of proprietary file formats implies changing over starting with one format then onto the next to utilize various tools. Conversion between different formats may put on risk of data corruption if the formats are not surely known. Metadata might be lost if all formats don't support similar types of metadata [72].

Extensibility Design

AFF can be utilized to store different kind of advanced metadata, for example, chain of custody data, GPS coordinates or some other user oriented.

AFF bolsters the meaning of discretionary metadata by putting away all data as name and value sets, called segments. A few segments store the disk data and others store metadata. On account of this general structure, any metadata can be characterized by essentially making another key and value pair. Every one of the segments can be compacted to diminish the size of drive images, and cryptographic hashes can be determined for each segment to guarantee data integrity [73].

Adaptable Design

For adaptability, there are three varieties of AFF files – AFF, AFD and AFM – and freely accessible tools to effortlessly change over between the varieties of formats.
The first AFF design is a solitary record that contains different sections with drive information and metadata. Its substance can be packed to decrease the size, however it very well may be very bigger as the information as on present day hard disks frequently came to 1 TB in size.

The first AFF format is a solitary file that contains segments with drive data and metadata. Its substance can be packed, however it very well may be very huge as the data on present day hard disks capacity frequently reached more than 100GB.

For simplicity of exchange, substantial AFF records can be broken into various AFD group documents. The littler AFD documents can be promptly moved around a FAT32 record framework which limits records to 2GB or put away on DVDs, which have comparable size limitations.

For simplicity of move, substantial AFF files can be broken into different AFD format files. The tiny AFD files can be promptly moved around a FAT32 file system which limits files to 2GB or stored on DVDs, which have comparative size confinements.

The AFM format stores the evidence metadata in an AFF file, and the disk data in a different raw file format. This format permits forensic analysis tools that help the raw format to access the data, however without losing the metadata. Hence, it would be easier to access forensic data [69].

Compression and Encryption

AFF introduces two compression algorithms: zlib, which is quick and sensibly proficient, and LZMA, which is slower yet significantly progressively effective. zlib is a similar compression algorithm utilized by EnCase. Accordingly, AFF files packed with zlib are generally a similar size as the identical EnCase file. AFF files
can be recompressed utilizing the LZMA algorithm. These files are somewhere in the range of 1/2 to 1/tenth the size of the first AFF/EnCase file.

AFF2.0 underpins encryption of disk images. Dissimilar to the password implemented by the EnCase tool, encoded images can't be accessed without the vital encryption key. FTK Imager/FTK included help for this encryption in form 3.0 and can make and access AFF encoded images.

2.11.2 AFF Tools

- affix
- affuse
- afinfo
- afstats
- afxml
- afsegment
- ident
- afcat
- afcompare
- afconvert

2.11.3 Advantages of AFF

- Storage of arbitrary digital metadata
- Storage of arbitrary types of digital data
- Compression levels at a broader range
- Ability to create images 1/4 to 1/2 the size of EnCase images using LZMA compression
- Size limits are not existing (64-bit)
- It is Open Source[71]
2.12 Researches in Cyber Forensics

A noteworthy problem to digital forensic investigation is the rapidly developing volume of digital information seized and to be dissected those evidences. This is because of expanding limit of storage media, different electronic devices and propelling of various distributed storage administrations, and an expansion in the quantity of media devices seized per case. Likewise it is seen that there is an extraordinary development of the rate of digital crime. This prompted expanding overabundances of digital evidences anticipating for examination, frequently numerous months to years, influences even the major computerized forensic research facilities. Over the earlier years, there has been an assorted variety of research started in connection to the volume challenge. Previous researches diverted to information decrease, information mining, computerized forensic, conveyed preparing, and other propelled strategies [74]. It gives overview about different distributed research and the proposed procedure. It is reasoned that there is as yet a requirement for further research with an emphasis on genuine world admissibility of a process or techniques to address the computerized forensic information volume and examination challenge.

The volume of advanced forensic evidence is hurriedly developing, prompting extensive accumulations [75]. The research paper, “a Digital Forensic Data Reduction and Data Mining Framework” is proposed. Introductory research with test information from South Australia Police Electronic Crime Section and Digital Corpora Forensic Images utilizing the proposed structure brought about noteworthy decrease in the capacity necessities — the diminished subset is just 0.196 percent and 0.75 percent separately of the first information volume. The suggested framework isn't recommended to supplant full investigation, yet helps to give a quick triage, accumulation, intelligence analysis, review and storage procedure to help the different stages of digital forensic examinations. Forensic agencies that can attempt quick evaluation of caught digital evidences can all the more adequately target explicit criminal issues. The framework is likewise competent to give a more prominent planned insight gain from investigation of evidences and flow and authentic forensic information in a convenient way, and the
capacity to embrace research of patterns after some time. A future research extension is to contemplate the potential advantages of having devoted shrewd digital forensic examination and research of computerized forensic information including the utilization of specific information mining strategies to separate insight from the organized and unstructured digital information normal to advanced scientific information subsets.

This paper on “Machine learning for post-event timeline reconstruction” we present a novel methodology for post-event timeline recreation utilizing machine learning procedures. Post-event timeline recreation assumes a basic job in forensic examination and serves as evidence in the digital crime. A variety of digital forensic tools have been created during most recent two decades to help computer forensic investigators for digital timeline analysis yet the vast majority of them can't deal with huge volumes of data in an effective way. The focal point of this paper is to layout the viability of utilizing machine learning techniques for computer forensic analysis by following past file-system activities and setting up a timeline of the events. Proposed methodology comprises of monitoring the file-system accesses, taking file-system snapshots at discrete interims of time by running various applications and utilizing this data to prepare a recurrent neural network to perceive the execution pattern of the individual applications. The trained variant of the network could thusly be utilized for producing post-event timeline of a seized hard disk to verify the execution of different applications at different time intervals [76].

Another methodology, in light of the k-Nearest Neighbour (kNN) classifier, is utilized to group program behaviours ordinary or intrusive. Short sequences of system calls have been utilized by others to portray a program's ordinary behaviour in past. Be that as it may, separate databases of short system call sequences must be worked for various programs, and learning program profiles includes tedious training and testing processes. With the kNN classifier, the frequencies of system calls are utilized to portray the program behaviour. Text categorization systems are received to change over each process to a vector and measure the similitude between two program profiles. Since there is no compelling reason to learn individual program profiles separately, the
calculation included is to a great extent decreased. Preliminary experiments with 1998 DARPA BSM review data show that the kNN classifier can adequately recognize nosy attacks and accomplish a low false positive rate [77].

Another research published on a host-based Intrusion Detection System (IDS) for Microsoft Windows. The core of the research is the algorithm that recognizes the attacks on a host machine by searching for unusual access to the Windows Registry. The key thought is to initially prepare a model of normal registry behaviour on a windows host, and utilize this model to recognize irregular registry access at run-time. The ordinary model is prepared utilizing clean (attack free) information. At run-time the model is utilized to check each access to the registry progressively to decide if the conduct is anomalous and (potentially) compares to an attack. The proposed system is powerful to recognize the activities of malignant software while at the same time keeping up a low rate of false alerts [78].

2.13 Digital Forensics XML (DFXML)

Digital Forensics XML (DFXML) is the effort to make a XML schema to permit interoperability between various digital forensics tools [59].

The strategy given is:

- To develop a set of standard XML tags and forensic purpose data representations for existing XML tools.
- To revise the current forensic tools to deliver proposed XML structure.
- To develop a standard structure (DTD) and schema to permit XML validation [60].
Digital Forensics XML (DFXML) is the XML language intended to represent a wide scope of digital forensic information and forensic results. By coordinating its deliberations to the requirements of forensics tools and experts, DFXML permits the sharing of structured information between autonomous tools and organizations. Since the underlying work in 2007, DFXML has been utilized to archive the consequences of forensic processing steps, reduce the requirement for re-processing digital evidence, and as an interchange format, enabling labelled forensic information to be shared between various researchers of forensic domains. DFXML is also the basis of a Python module (dfxml.py) that makes it easy to create sophisticated forensic processing programs (or “scripts”) with little efforts.

Forensic devices can be promptly adjusted to create and expend DFXML as an option digital information representation format. For instance, the PhotoRec carver (Grenier, 2011) and the MD5DEEP hashing application (Kornblum, 2011) were both changed appropriately, that produces DFXML files. The DFXML yield contains the recognized records, its physical location inside the disk image (on account of PhotoRec), and their cryptographic hashes. Since these projects now both produce perfect DFXML, their yield can be handled by an aggregate arrangement of devices.

DFXML can likewise store the report source, the PC on which the application program was ordered, the dynamic linked libraries, and the runtime environment. Such source can be valuable both in research and in getting ready court declaration.

DFXML’s minimal use of XML features means that the forensic abstractions, APIs and representations described, can be readily migrated to other object-based serializations, including JSON (Zyp and Court, 2010), Protocol Buers (Google, 2011) and the SQL schema implemented in SleuthKit 3.2 (Carrier, 2010). Certainly, it is quite possible to easily be convertible between all given formats.
A large number of the digital forensic tools use XML to store digital evidences for digital forensic environment in light of the fact that there is enormous amount of information would be stored in other storage media. XML is truly adaptable to store the information and we can without much of a stretch produce and devour those information for investigation purpose.

DFXML is a methodology to produce and expend digital evidence in such a typical way for every single forensic device. It characterizes a typical approach to store evidence in XML that is called as DFXML. Utilizing this process for storing digital forensic evidence using basic structure, we can accomplish interoperability between different forensic tools.

We can utilize any forensic tools to bring digital forensic evidence and furthermore utilize some other forensic tools to analyse that forensic information. Consequently, it is simpler to utilize any forensic tool whenever for examination purpose. Henceforth, examiners can get extraneous advantage of utilizing proposed approach to develop forensic software tool like just a “Aggregate once, consume for all”.

The quick pace of creation in digital advances presents broad difficulties to digital forensics. Introduction of new storage and memory devices and improvements in existing ones gives difficulties at exponential rate to the securing of digital evidence. The expansion of contending file formats and communication protocols challenges examiners’ capacity to extract essentialness from the arrangement of one’s and zero’s inside. Fundamentally, overarching these challenges are the concerns of maintaining the integrity of any forensic evidence found, and reliably explaining any conclusions drawn based on the evidences [26].

Forensic Researchers and specialists in the field of digital forensics have reacted to these difficulties by developing forensic tools for evidence acquisition and examination. Nowadays, these efforts have brought about an assortment of ad-hoc and
proprietary formats for storing evidence data, examination results, and evidence metadata, for example, worthiness and provenance data. Conversion of evidence formats used and produced by the present software forensic tools is quite complicated. This procedure needs expansive amount of time and manual intervention, and there exists the potential that it might deliver incorrect evidence information, or lose metadata (DFRWS 2005). Approval of the outcomes created is stuck by absence of format standards [79].

It is the concerns that call have been made for a universal container to capture digital evidence. Lately, the expression "Digital evidence Bags" was introduces to allude to a container for digital evidences, evidence metadata, integrity of data, and access and usage audit reports (Turner 2005). In this way, the Digital Forensics Research Workshop (DFRWS) recently framed a working group with an objective of characterizing standardized Common Digital Evidence Storage Format (CDESF) for storing digital evidence and related metadata (DFRWS, 2005) [74]

Additional source of multifaceted nature related to the ad-hoc delegated nature of forensic devices is the lack of a common representational format for evidence metadata. This is genuinely not a minor issue due to the possibility of the forensics domains that deals with massive conceptual complexity within multiple layers of abstraction. The biggest challenge here is to identify a suggests that decouples the models of evidence and evidence metadata used by software forensics tools from the utilization justification of these tools. Also, this ought to be developed in a manner that empowers the establishment of provenance and preserve the integrity.

This illustrative issue isn't simply limited to the challenge of tool interoperability. In representing the "Big Computer Forensic Challenges", Spafford sees that pros and researchers in the field of digital forensics don't use standard terminology (Palmer, 2001). It is not surprising that we discovered limited attention paid to the formal definition of taxonomies or ontologies describing this domain [80].
It proposes the use of ontologies as a solution to these terminological and representational issues. They have introduced a number of basic ontologies modelling that is the domain of digital evidence acquisition, computer networks and computer hardware, and described these ontologies using the Web Ontology Language (OWL) (McGuinness, 2001) [81].

Inclusions of logical orders, ontologies are described by Gruber (1993) as an explicit specification of a conceptualization. Even more engagingly, ontology is a strategy for domain of discourse, similar to thoughts, properties, and relationship of various entities. Ontology languages, the inspiration driving which are to depict ontologies, hold the assurance of empowering machines to have increasingly conspicuous ability to reason over and analysis of evidences, commonly of sharing an ordinary understanding of the present information (Under coffer et. al., 2004). Furthermore, it encourages knowledge sharing and reuse within a domain, which has the potential to lead towards a convergence of vocabulary in the forensics domain [82].

2.13.1 The need for DFXML

The present digital forensic tools need uniformity. Instead of being arranged with the Unix approach of devices that can be related together to deal with immense forensic issues, most routinely used forensic devices are monolithic systems expected to ingest couple of data types (typically disk images and hash sets) and produce a compelled set of data types (conventionally individual records and final reports). This is certified that both of devices with limited capability to function (for instance simple file carvers), as well as the complex GUI-based forensic tools that incorporates integrated scripting languages. The nonappearance of closeness has entrapped automation and tool validation efforts, and in the process has unassumingly bound the advancement of digital forensics research [79].

Regardless of the way that there is existing file formats and a different XML structures used in digital forensics today, they are bound to specific applications and
limited domains. The absence of systematized abstraction makes it difficult to compare results conveyed by different forensic tools and algorithms. This absence of standardized formats may impact forensic software developers who ought to regularly include functionalities in their software. Unfortunately, it is redundant functionalities in existing forensic tools [79].

2.13.2 Usages of DFXML

DFXML improves closeness by giving a language to delineating traditional forensic process (e.g., cryptographic hashing), forensic software products (e.g., the location of documents on a hard drive), and metadata (e.g., filenames and time stamps).

Different model DFXML executions have been used by the researchers since 2007 for various purposes:

A device reliant on SleuthKit called fiwalk (x5.1) ingests disk images and reports the location and related file system metadata of each file in the disk image. This tool was used by students for Masters' thesis and a project that associated machine learning to computer forensics (Garfinkel, Parker-Wood, Huynh, and Migletz, 2010).

A DFXML file was made for each disk image in a corpus of more than 2000 disk images acquired from all over the world (Garfinkel, Farrell, Roussev, and Dinolt, 2009). Each DFXML file contains information concerning the disk's purchase, physical characteristics, imaging process, allocated and deleted records, and extracted metadata from those files (e.g., Microsoft Office document properties, extracted JPEG EXIF information).

The DFXML Python modules (x4.1) make it possible to form smaller programs that perform complex forensic tasks with these files. Alternately, the desire to learn and adjust for tools, for instance, EnCase EnScript (Guidance Software, 2007) and SleuthKit (Carrier, 2010) can be exceptionally steep.
The XML files make it definitely less difficult to share data to various organizations. On occasion it has recently been imperative to share the XML files, rather than the disk images themselves. This is progressively successful, as the records are significantly humbler than the disk images. It also helps to verify the security of the data subjects.

The XML format makes it easy to perceive and redact personal information. The resulting redacted XML files can be shared without the requirement for Institutional Review Board (IRB) or Ethics Board approval; they can even be published on the Internet.

At long last, in light of the way that the DFXML file records all information about the forensic tools version used to produce DFXML evidences. It is definitely not hard to have tools normally reprocess disk images when the toolset improves [83].

2.13.3 Contributions

There are a couple of responsibilities to the field of digital forensics. At first, it represents the motivation and plan goals for DFXML. Second, it also presents an example of how DFXML can be used to depict forensic artefact. These points of reference make it basic for developers of the present forensic tools to adopt their tools to deliver and ingest DFXML as an enhancement to their current file formats. Next, it shows an API that considers the rapid prototyping and improvement of forensic applications. Finally, it depicts how the DFXML abstractions can be used as a structure for making new automated forensic processes [84].

2.13.4 Goals of DFXML

DFRWS moved an errand in 2007 to make standardized abstractions for digital forensics; this effort was left incomplete due to lack of support and financing (Common Digital Evidence Storage Format Working Group, 2007). In perspective on
the DFRWS experience, it seems, by all accounts, to be reasonable that any efforts to make a XML language for digital forensic investigation should be proposed as a low-cost project that nevertheless can produce significant savings or provide new capabilities. The following goals are compatible with such financial support [79].

Existing Forensic Formats

Rather than replacing existing formats, the new language ought to extend them. This is developed by making it easy to adapt between legacy and new associations, and by making systems with the objective that the new formats can be used to annotate legacy data.

Simple to produce

It must be anything besides harder to alter existing forensic tools to create the new representations. An open source C and C++ library helps in the change method.

Simple to ingest

Also, it must be anything besides harder to modify existing forensic tools to read and process DFXML. An open source DFXML Python module reliant on the Python SAX XML parser makes it possible to capably examine and process DFXML files [80].

Human Readability

A forensic analyst without any training should have the ability to look at conforming DFXML file and comprehend it without the prerequisite for the trained forensic investigator. To this end, various forensic tools produce DFXML that is pretty-printed.
Free, open, and extensible

Both the reference and representation of DFXML structure must be available for all without any charge. Software Forensic tools developers should be able to add new tags without the need for central coordination (need to use XML namespaces).

Adaptable

The representation must be usable for both side of forensic scale. Small portion of information must have short description, while it must be possible to efficiently process XML files tens of gigabytes in size (which may come about due to planning multi-terabyte drives). In that limit, it must be possible to process DFXML using event based XML parsers (e.g., Python Software Foundation (2010); Cameron, Herdy, and Lin (2008); Zhang and van Engelen (2006)), instead of requiring the usage of tree-based parsers, for instance, those subject to the Document Object Model [85].

Obey existing practices and principles.

DFXML would seek after existing benchmarks instead of making new ones. As there are different conflicting benchmarks exist, DFXML ought to implement those standards that are the most gainful and appropriate for the forensic investigation process.

2.13.5 Overall design

- DFXML is expected to represent to the accompanying sorts of forensic information:
• Metadata outlining the source disk image file or any other input. Normally this is the name of the image file, but it may represent other information as well.
• Detailed information about the forensic tools that performed forensic processing (e.g., Version number and the program name, where the program was compiled, Linked Library).
• The state of the computer system on which the acquired evidence processing performed (e.g., the name of the computer system, the dynamic libraries that were used, the timestamp of the program was run).
• The forensic evidence or information that was extracted, how the extraction being carried out, and the location of that evidences found.
• Cryptographic hash values of byte sequences.
• Operating System (OS) related information beneficial for forensic examination.
• To provide the Standard Structure for forensic evidences

2.14 Standard Structured format of DFXML for various digital forensics objects

Various attributes are available as a metadata of various digital forensics objects. Different evidences are available in structured format as given below.

2.14.1 File/Directory Object:

There are following attributes of file/directory object.

• Filename : It specifies name of file/directory
- Location: File/Directory path that locates the object
- Filesize: The size of file
- Filetype: Type of file whether file/directory
- Shared: file/directory is shared or not
- Contains: It represents that the subdirectories are present in the given directory, if it contains then number of subdirectories are specified.
- Created: Date and time of file/directory object is created
- Modified: Date and time of modification of file/directory object
- Accessed: Accessed date and time
- Attribute: Specifies attributes of file e.g. readonly, hidden, encrypted or ready to archive
- Revisionno: Number of times the file has been modified or saved
- Hashdigest: Footprint of file using some algorithms like md5, SHA-1 or any other algorithm

2.14.1.1 DTD for File/Directory Object

<!DOCTYPE fileInformation [ 

<!ELEMENT fileobject (filename, location, filesize, filetype, shared, contains+, signature, created, modified, accessed, attribute+, revisionno, hashdigest)> 

<!ELEMENT filename (#PCDATA)> 

<!ELEMENT location (#PCDATA)> 

<!ELEMENT filesize (#PCDATA)> 

69
<!ELEMENT  filetype (#PCDATA)> 

<!ELEMENT  shared (#PCDATA)> 

<!ELEMENT  contain (nooffile, noofdirectory)> 

<!ELEMENT  nooffiles (#PCDATA)> 

<!ELEMENT  noofdirectory (#PCDATA)> 

<!ELEMENT  signature (#PCDATA)> 

<!ELEMENT  created (#PCDATA)> 

<!ELEMENT  modified (#PCDATA)> 

<!ELEMENT  accessed (#PCDATA)> 

<!ELEMENT  attribute (readonly, hidden, encrypted, readytoarchive)> 

<!ELEMENT  readonly (#PCDATA)> 

<!ELEMENT  hidden (#PCDATA)> 

<!ELEMENT  encrypted (#PCDATA)> 

<!ELEMENT  readytoarchive (#PCDATA)> 

<!ELEMENT  revisionno (#PCDATA)> 

70
The File/Directory object’s DFXML file will look as shown below.

```xml
<?xml version="1.0"?>
<fileobject>
  <location>E:\ME\SEM4\Final Dissertation\Demo\fileInformation.py</location>
  <filesize>3237</filesize>
  <creationDateTime>20 Jun 2013 10:07:52</creationDateTime>
  <modifiedDateTime>21 Jun 2013 11:04:37</modifiedDateTime>
  <accessedDateTime>20 Jun 2013 10:07:52</accessedDateTime>
  <hashDigest type="md5">7b46b864229e7b87bf009d9084dc7c3c</hashDigest>
</fileobject>

<fileobject>
  <location>E:\ME\SEM4\Final Dissertation\Demo\FileInformation.xml</location>
  <filesize>0</filesize>
  <creationDateTime>20 Jun 2013 16:52:48</creationDateTime>
  <modifiedDateTime>21 Jun 2013 19:37:42</modifiedDateTime>
  <accessedDateTime>20 Jun 2013 16:52:48</accessedDateTime>
  <hashDigest type="md5">f02be8dd5cf1df42e2801c3ca1de6a</hashDigest>
</fileobject>

<fileobject>
  <location>E:\ME\SEM4\Final Dissertation\Demo\main.py</location>
  <filesize>5883</filesize>
  <creationDateTime>20 Jun 2013 11:30:23</creationDateTime>
  <modifiedDateTime>20 Jun 2013 22:18:13</modifiedDateTime>
  <accessedDateTime>20 Jun 2013 11:30:23</accessedDateTime>
  <hashDigest type="md5">f9c3c867e2edf941f8f4f0302fa76360</hashDigest>
</fileobject>

<fileobject>
  <filename>Persistent&VolatileInformation.py</filename>
  <location>E:\ME\SEM4\Final Dissertation\Demo\Persistent&VolatileInformation.py</location>
  <filesize>1041</filesize>
  <creationDateTime>20 Jun 2013 10:19:58</creationDateTime>
</fileobject>
```

Figure. 2.4 DFXML file for file object evidence

2.14.2 Process Object

It represents information related to running process in the operating system. It mostly used to recognize malicious processes that act like innocent process. There are following attributes of Process object are:

- **PId**: It represents Process ID

71
2.14.2.1 DTD for Process Objects

<!DOCTYPE processes [

<!ELEMENT processes (process+)>

<!ELEMENT process (id, name, description, memory-usage)>

<!ELEMENT pid (#PCDATA)>

<!ELEMENT ppid (#PCDATA)>

<!ELEMENT status (#PCDATA)>

<!ELEMENT name(#PCDATA)>

<!ELEMENT description (#PCDATA)>

<!ELEMENT memory-usage(#PCDATA)>

]>}

The Process object’s DFXML file will look as shown below.
<xml version="1.0"?>
  - <processes>
    - <process>
      <pid>0</pid>
      <ppid>0</ppid>
      <pname>System Idle Process</pname>
      <status>running</status>
      <memory-usage>24576</memory-usage>
    </process>
    - <process>
      <pid>4</pid>
      <ppid>0</ppid>
      <pname>System</pname>
      <status>running</status>
      <memory-usage>2625536</memory-usage>
    </process>
    - <process>
      <pid>284</pid>
      <ppid>4</ppid>
      <pname>smss.exe</pname>
      <status>running</status>
      <memory-usage>667648</memory-usage>
    </process>
    - <process>
      <pid>392</pid>
      <ppid>376</ppid>
      <pname>csrss.exe</pname>
      <status>running</status>
      <memory-usage>3096576</memory-usage>
    </process>
    - <process>
      <pid>452</pid>
      <ppid>376</ppid>
      <pname>wininit.exe</pname>
      <status>running</status>
      <memory-usage>3051520</memory-usage>
    </process>
  </processes>

Figure 2.5 DFXML file for process evidence

### 2.14.3 Registry Object

It is used to represent attributes of Windows Registry that is the storage structure in the windows operating system. It contains different hives that store information used to execute programs in the operating system. Following attributes used to store such information about windows registry data.
- Msregistry: Microsoft registry
- Hive: It describes hive information
- Key: Represents main key path of registry
- Subkey: Represents subkey information path of registry
- Name: name of registry data
- Type: Data Type of registry
- Data: Raw data of registry

### 2.14.3.1 DTD for Registry Object

```xml
<!DOCTYPE msregistry [ 

<!ELEMENT msregistry (hive+)> 

<!ELEMENT hive (key+)> 

<!ELEMENT key (subkey+)> 

<!ELEMENT subkey (name, type, data)> 

<!ELEMENT name (#PCDATA)> 

<!ELEMENT type (#PCDATA)> 

<!ELEMENT data (#PCDATA)> 

]> 

The Windows Registry object’s DFXML file will look as shown below.
2.14.4 System Object

This object is used to describe information about computer system that is used as evidence. Acquired storage media belongs to this computer system. Hence, information about computer system may play the key role to prove committed crime. There are various attributes used to store data about the computer system are as follows.

- system-info: It describes all system information
- computer-name: Describes the computer name
- workgroup: Describes the name of workgroup to which computer belongs
- IP_address: Describes computer IP address
• MAC-address: Describes the physical address of the computer system
• Processor: Describes the information about the information
• RAM: Describe the information about RAM (Random Access Memory)
• OS-name: Describes the name of Operating System installed in the computer system
• OS-edition: Describes the edition of Operating system
• OS-type: Describes the type of Operating System
• Storage-disk: Describes detailed information about storage media used
• Volume-name: Drive letter of storage media
• File-system: Type of file system used by Operating System
• Devicename: Name of storage device
• attached-device: Description about various attached storage devices to the computer system
• serial-no: Serial number of all attached devices

2.14.4.1 DTD for system information object

<!DOCTYPE system-info [ 

<!ELEMENT system-info (computer-name, workgroup, IP-Address, MAC-address, processor, RAM, OS-Name, OS-edition, OS-type, storage-disk+, attached-devices+)>

<!ELEMENT (computer-name (#PCDATA)> 

<!ELEMENT workgroup (#PCDATA)> 

76
<!ELEMENT IP_Address (#PCDATA)>

<!ELEMENT MAC-address (#PCDATA)>

<!ELEMENT processor(#PCDATA)>

<!ELEMENT RAM(#PCDATA)>

<!ELEMENT OS-Name(#PCDATA)>

<!ELEMENT OS-edition(#PCDATA)>

<!ELEMENT OS-type(#PCDATA)>

<!ELEMENT Storage-disk(volume-name, filesystem, devicename)>

<!ELEMENT volume-name(#PCDATA)>

<!ELEMENT filesystem(#PCDATA)>

<!ELEMENT devicename(#PCDATA)>

<!ELEMENT attached-devices(name, volume, serial-no)>

<!ELEMENT name(#PCDATA)>

<!ELEMENT volume(#PCDATA)>

<!ELEMENT serial-no(#PCDATA)>
The System structure object’s DFXML file will look as shown below.

```xml
<?xml version="1.0"?>
<system-info>
  <computer-name>Esan</computer-name>
  <ipaddress>172.0.0.1</ipaddress>
  <OS-Name>Windows 7</OS-Name>
  <processor>x86 Family 6 Model 37 Stepping 2, GenuineIntel</processor>
  <OSversion>6.1.7600</OSversion>
  <currentuser>Esan</currentuser>
  <volumename>['C', 'D', 'E', 'F', 'I']</volumename>
  <OS-Architecture>x86</OS-Architecture>
  <bytes_sent>0</bytes_sent>
  <bytes_recv>0</bytes_recv>
  <packets_sent>0</packets_sent>
  <packets_recv>0</packets_recv>
  <cpu_count>4</cpu_count>
  <RAM>
    <Total_capacity>3077468160</Total_capacity>
    <Available_Memory>1708097536</Available_Memory>
    <Usage_Percentage>44.5</Usage_Percentage>
    <Used_Memory>1369366528</Used_Memory>
    <Free_Memory>1708097536</Free_Memory>
  </RAM>
  <MAC_Address>f0:7b:cb:36:60:2f</MAC_Address>
</system-info>
```

Figure 2.7 DFXML for System Information evidence

### 2.14.5 Event Object

There are various events occurred in the operating system. These events may be triggered due to user interaction, system event or any other external sources. All these events play a crucial role in forensic investigation process. Windows Operating system creates a log of various events generated by any sources. Hence, this log should be analyzed and stored in structure format in order to analyze using forensic tools. The various attributes of such events are as described below.

- Event: Describe information about event viewer tool
- EventID: It represents the event ID
• Ename: Describes the name of Event
• Execution_ProcessID: The process ID that triggered the event
• ThreadID: It represents the thread ID of particular process
• Channel: Describe the main category of path
• Username: Describes the OS user of related event
• Edate_time: Describes date and time of triggered event
• source: Describes the OS user who triggered the event
• level: Describes the type of event – error, information, warning
• process_id: Represents the process id that has triggered the event
• key_word: Status of event -- audit success, audit failure, classic
• task_category: Describes type of task -- logon, general, special logon

The Event object’s DFXML file will look as shown below.

```xml
<?xml version="1.0"?>
<event>
  <time>00/16/16 14:30:08</time>
  <eventID>2020</eventID>
  <eventType>EVENTLOG_INFORMATION_TYPE</eventType>
  <eventRecord>1594335</eventRecord>
  <eventSource>Service Control Manager</eventSource>
  <eventMessage>(Yes), u'the Windows search service entered the running state.

  <time>00/16/16 14:30:04</time>
  <eventID>2020</eventID>
  <eventType>EVENTLOG_INFORMATION_TYPE</eventType>
  <eventRecord>1594334</eventRecord>
  <eventSource>Service Control Manager</eventSource>
  <eventMessage>(Yes), u'the IPsec Policy Agent service entered the running state.

  <time>00/16/16 14:30:08</time>
  <eventID>2020</eventID>
  <eventType>EVENTLOG_INFORMATION_TYPE</eventType>
  <eventRecord>1594333</eventRecord>
  <eventSource>Service Control Manager</eventSource>
  <eventMessage>(Yes), u'the Remote Access Connection Manager service entered the running state.

  <time>00/16/16 14:30:06</time>
  <eventID>2020</eventID>
  <eventType>EVENTLOG_INFORMATION_TYPE</eventType>
  <eventRecord>1594332</eventRecord>
  <eventSource>Service Control Manager</eventSource>
  <eventMessage>(Yes), u'the Secure Socket Tunneling Protocol Service service entered the running state.

</event>
```

Figure 2.8 DFXML for Event logs
2.15 Archival and preservation of proposed structure:

Archival and Preservation is the most important for digital forensic investigation data. Acquired evidences must be preserved in order to maintain its integrity. Hence, the standard structure is developed to store forensic evidence attributes in dfxml file. Therefore, acquired evidences remain untouched because attributes are stored as evidence.

Following standard structure format used to preserve and archive the forensic evidence.

2.15.1 Information of proposed structure:

Initially, the case information should be documented in order to maintain its authenticity and authorization. Hence, various information are used like, case details, investigators’ data, incident description, location of evidence and its hash value to verify integrity are needed to store in structured format. Following attributes are used to store such case details.

- **case**: Describes archival case file
- **case_info**: Describes the detailed information about the case
- **case_id**: Describes the case ID
- **case_name**: Describe the name of case
- **case_type**: Describe the type of case( Data theft, Identity theft etc.)
- **investigator_info**: Describes information about forensic investigators
- **Inv_id**: Describes the unique ID of investigator
- **Inv_name**: It describes the name of investigator
- **Investigation_info**: Describes investigation information
- **InvS_DT**: Describes start date and time of investigation process
- **InvE_DT**: Describes end date and time of investigation process
- **Inv_Duration**: Describes duration taken by investigation process
- **Incident_info**: Describes detailed information about the incident
- **Incident_date**: Date and time when the incident happen
- **Incident_location**: Describes the location of incident
- **afileInfo**: Describes archival file information
- **afname**: Describes archival file name
- **afhashvalue**: Describes the hash value of archived files
- **afpath**: Describes archival file storage path details

### 2.15.2 Standard structure for archive and preserve data:

```xml
<Case>
  
  <Case_info>
    
    <case_id>038</case_id>
    
    <case_name>Rai University</case_name>
    
    <case_type>Data Theft</case_type>
    
  </Case_info>
  
  <Investigator_info>
    
    <Inv_ID>E002</Inv_ID>
    
  </Investigator_info>

</Case>
```
<Inv_Name>Esan</Inv_Name>

<Inv_contact>esan@gmail.com</Inv_contact>

</Investigator_info>

<Investigation_info>

<InvS_DT>12-03-2017 02:40:00</InvS_DT>

<InvE_DT>12-03-2017 03:40:00</InvE_DT>

<Inv_Duration>60</Inv_Duration>

</Investigation_info>

<Incident_info>

<Incident_date>13-02-2017</Incident_date>

<Incident_location>Ahmedabad</Incident_location>

</Incident_info>

<afInfo>

<afname>case43.dfxml</afname>

<afhashvalue type="MD5"> </afhashvalue>
2.15.2.1 DTD to archive and preserve forensic data

<!DOCTYPE Case [

<!ELEMENT Case (Case_info+, Investigator_info+, Investigation_info+, Incident_info+, afileInfo+)>

<!ELEMENT Case_info(case_id,case_name,case_type)>

<!ELEMENT case_id (#PCDATA)>

<!ELEMENT case_name (#PCDATA)>

<!ELEMENT case_type (#PCDATA)>

<!ELEMENT Investigator_info(Inv_ID, Inv_Name, Inv_contact)>

<!ELEMENT Inv_ID (#PCDATA)>

<!ELEMENT Inv_Name (#PCDATA)>

<!ELEMENT Inv_contact (#PCDATA)>
2.16 Clustering

Clustering can be considered the most noteworthy unsupervised learning problem; therefore, as one another issue of this sort, it solves the problem of finding a structure in the collection of unlabelled data. A rough definition could be "the process of
organizing objects into groups whose members are similar in some way”. A cluster is subsequently a collection of objects which are "similar" among them and are "dissimilar" to the objects belongs to other clusters.

It can be visually represented as shown below:

![Clustering Graphical Example](image)

Figure 2.9 Clustering Graphical Example

In above case, we can easily classify the 4 clusters into which the data can be distributed; the similarity criterion is distance: multiple objects belong to the same cluster if they are “close” according to a given distance (in this case geometrical distance). It is called “distance-based clustering”.

Another type of clustering is “conceptual clustering”, two or more objects belong to the same cluster if this one defines a concept common to all that objects. In other words, objects are grouped according to their fit to descriptive concepts, not according to simple similarity measures.

2.16.1 K-means clustering Algorithm

K-means clustering is a system for vector quantization, at first from signal processing, that is well known for cluster analysis in data mining. K-means clustering purpose to fragment n partitions into k clusters in which each data point has a cluster
with the nearest mean, can be demonstrated as a cluster. This results in an allotting of the data space into Voronoi cells.

The issue is computationally more costly (NP-hard); in any case, there are successful heuristic algorithms that are commonly used and converge very quickly to a local optimum. These are commonly like the expectation-maximization algorithm for mixes of Gaussian distributions through an iterative refinement approach used by both k-means and Gaussian mixture modeling. Besides, both of them use group centers to model the data; nevertheless, k-means clustering will by and large find clusters of equivalent spatial extent, while the expectation-maximization mechanism empowers clusters to have different shapes.

The estimation has not strong relationship to the k-nearest neighbor classifier, a notable machine learning technique for classification that is as often as possible problem for k-means because of the “k” in the name. One can apply the 1-nearest neighbor classifier to the center of cluster achieved by k-means to classify new data into the existing clusters. This is known as nearest centroid classifier or Rocchio algorithm.

2.16.1.1 History

The word "k-means" was first used by James MacQueen in 1967,[86] anyway the idea comes back to Hugo Steinhaus in 1957 [87]. The standard computation was first proposed by Stuart Lloyd in 1957 as a technique for pulse code modification, anyway it wasn't circulated outside of Bell Labs until 1982 [88]. In 1965, E. W. Forgy published fundamentally a comparable methodology, which is the reason it is now and again suggested as Lloyd-Forgy [89].

It has a cluster with unsupervised data that can be used to take care of problem of similarities and dissimilarities between the data. It conceives given dataset according to their similar characteristics and attributes in number of clusters. The
The central idea is to pick number of clusters and centroids for each group. These centroids should be chosen using a couple of calculations and to set far isolated from each other to measure the progress.

The further development is to pick each point and find the nearest centroid beginning there and include it to that group and repeat it for all data points of the given dataset. Now, re-learn the centroids and assign data point centers. New K-centroids are arranged at better positions diverged from past clusters. Continue with this methodology until we get no any changes in clusters centroids. Consequently, the crucial focus of this algorithm is to minimize an objective function. The objective function knows as squared error is shown below

\[ J(V) = \sum_{i=1}^{C} \sum_{j=1}^{C_i} (\|x_i - v_j\|^2) \]

where,

‘\|x_i - v_j\|’ is the Euclidean distance between xi and vj.

‘ci’ is the number of data points in ith cluster.

‘c’ is the number of cluster centers.

Three key highlights of k-means which make it productive are regularly viewed as its greatest drawbacks:

Euclidean distance is used as variance and metric is used as measure of cluster scatter. The number of clusters k is an input parameter: a wrong choice of k may yield poor results. That is the reason, when performing k-means, it is important to
perform diagnostic checks for choosing the amount of clusters in the data list. Convergence to a local minimum may convey irrational ("wrong") results.

A key restriction of k-means is its cluster model. The idea depends on spherical clusters that are separable with the goal that the mean converges towards the cluster center. The clusters are relied upon to be of comparable size, with the goal that the assignment to the closest cluster focus is the right task.

When for instance applying k-means with a value of k=3 onto the outstanding Iris flower data set, the outcome frequently neglects to isolate the three Iris species contained in the data set. With k=2, the two noticeable clusters (one containing two species) will be found, while with k=3 one of the two clusters will be part into two even parts. Truth be told, k=2 is progressively fitting for this data set, regardless of the data set is containing 3 classes. Likewise with some other clustering algorithm, the k-means result makes suppositions that the data fulfil certain criteria. It functions admirably on certain data sets, and flops on others.

The aftereffect of k-means can be viewed as the Voronoi cells of the cluster means. Since data is part somewhere between clusters means, this can prompt imperfect parts as can be found in the "mouse" model. The Gaussian models utilized by the expectation-maximization algorithm (apparently a generalization of k-means) are increasingly adaptable by having the two statistical terms variance and covariance. The EM result is along these, ready to suit clusters of variable size obviously superior to k-means just as correlated clusters (not in this model). K-means is firmly identified with nonparametric Bayesian modelling [90].

2.16.1.2 Complexity

As to multifaceted nature, finding the ideal answer for the k-means clustering issue for observations in d measurements is:
NP-hard as a rule Euclidean space d dimensions notwithstanding for 2 clusters [91][92][93][94],

NP-hard for a general number of clusters k even in the plane [95],

if k and d (the measurement) are fixed, the issue can be actually fathomed in time $O(n^{(dk+1)})$, where n is the quantity of elements to be grouped [96].

In this manner, an assortment of heuristic calculations, for example, Lloyd's calculations that are given above are commonly used.

The running time of Lloyd's estimation (and most varieties) is $O(nkdi)$, [97][98] where n is the amount of d-dimensional vectors, k the amount of clusters and i the amount of iterations required until convergence. On data that has a clustering structure, the amount of iterations until the convergence is pretty much nothing, and results simply improve hardly after the underlying multiple iterations. Lloyd's algorithm is thusly consistently seen starting at "linear" complexity in practice, for all intents and purposes, disregarding the way that it is in the worst case super-polynomial when performed until convergence [99].

Following are a bit of the on-going problems into this observed complex nature of algorithm. In the most worst case situation, Lloyd's computation needs $i=2^{o(n^2)}$ iterations, with the objective that the most critical situation worst case complexity of Lloyd's figuring is excessively polynomial [100].

Lloyd's k-means algorithm has polynomial smoothed running time. It is showed up for discretionary set of n centers in $[0,1]^d$, if each point is independently disturbed by a normal distribution with variance $\sigma^2$ and mean 0, by then the expected running time of k-means algorithm is restricted by $O(n^{34} k^{34} d^8 \log^4(n)/\sigma^6)$, which is a polynomial in n, k, d and 1/$\sigma$. 

89
Better bounding points are exhibited for fundamental cases. For example, in [101] it is exhibited that the running time of k-means clustering algorithm is restricted by \( O(dn^4M^2) \) for \( n \) centers in an entire number cross segment \( \{1, \ldots, M\}^d \).

Lloyd's struggle was the standard technique for this issue. In any case, it contributes a huge amount of time to compute distances between all of the k cluster centers and the n data points. Since data points usually reside same cluster after few iterations, a considerable amount of this work is pointless, making the naïve implementation very inefficient. A few executions use caching and the triangle inequality so as to make bounds and accelerate Lloyd's algorithm [102][103][104][105].

2.17 Linear Regression

In statistics, linear regression is a linear way to deal with demonstrating the connection between the dependent and independent variables. The instance of one informative variable is called basic linear regression. For more than one illustrative variable, the procedure is called multi-linear regression [106]. This term is particular from multivariate linear regression, where different interrelated dependent variables can be predicted, as contrasting to a single scalar variable [107].

2.17.1 Simple and Multi-linear Regression method

The least difficult instance of a single scalar predictor variable \( x \) and a single scalar predictor variable \( y \) is known as basic linear regression. The extension of various and additionally vector-value predictor variable, \( X \) is known as multiple linear regression, otherwise called multivariable linear regression.

About all real world regression models include different predictors, and brief descriptions of linear regression are frequently stated as far as the various regression model. Note, in any case, that in these cases the response variable \( y \) is as yet
a scalar. Another term, multivariate linear regression, alludes to case where $y$ is a vector, i.e., equivalent to general linear regression.

Figure 2.10 Data points versus Regression line Example

As shown in figure 2.10, Data is divided in two different parts- Training set and Testing Set. Data can be divided using various techniques such as k-fold technique and many more. Model is trained using training data points and plotted on the graph. Trained model is used to predict the values for testing data points and plotted on the graph. It can be observed from the figure that regression line that should be fitted from all data points such that it minimizes the error between actual and predicted values.
Figure 2.11 Distance between actual and predicted value

As shown in figure 2.11, regression line should be fitted such that it must minimize the error of model.

2.17.2 Cost Function

The cost function encourages us to make sense of the most ideal qualities for independent variables which would give the best fit line to the information focuses. Since we need the best qualities for various independent variables, we convert this pursuit issue into a minimization problem where we might want to limit the error between the anticipated value and the real value.
2.17.3 Use of Gradient Descent to reduce error

Gradient descent is a technique for changing the value of independent variables to diminish the Mean Squared Error (MSE). The thought is that we begin with certain value for independent variable and after that we change these values iteratively to lessen the expense. Gradient descent causes us on the best way to change the values.
Figure 2.14 Non Convex and Convex Function

The question is how to utilize gradient descent to modify value of independent variables. To modify values of such variables, we take gradients from the cost function. To discover these gradients, we take incomplete subsidiaries concerning independent variables.

\[
J = \frac{1}{n} \sum_{i=1}^{n} (\text{pred}_i - y_i)^2 \\
J = \frac{1}{n} \sum_{i=1}^{n} (a_0 + a_1 \cdot x_i - y_i)^2
\]

\[
\frac{\partial J}{\partial a_0} = \frac{2}{n} \sum_{i=1}^{n} (a_0 + a_1 \cdot x_i - y_i) \implies \frac{\partial J}{\partial a_0} = \frac{2}{n} \sum_{i=1}^{n} (\text{pred}_i - y_i) \\
\frac{\partial J}{\partial a_1} = \frac{2}{n} \sum_{i=1}^{n} (a_0 + a_1 \cdot x_i - y_i) \cdot x_i \implies \frac{\partial J}{\partial a_1} = \frac{2}{n} \sum_{i=1}^{n} (\text{pred}_i - y_i) \cdot x_i
\]
\[ a_0 = a_0 - \alpha \cdot \frac{2}{n} \sum_{i=1}^{n} (\text{pred}_i - y_i) \]

\[ a_1 = a_1 - \alpha \cdot \frac{2}{n} \sum_{i=1}^{n} (\text{pred}_i - y_i) \cdot x_i \]

Figure 2.15 Partial derivatives to update independent variables

The partial derivatives are the gradients and they are utilized to correct the estimations of \(a_0\) and \(a_1\). Alpha is the learning rate which is a hyperparameter that should be indicated. A lower learning rate could get you closer to the minima yet sets aside more effort to come to the minima, a higher learning rate unites sooner however quite possibly, could skip the minima.

2.18 Optimal String Alignment Distance Algorithm

In information theory and computer science, the Damerau–Levenshtein distance (named after Frederick J. Damerau and Vladimir I. Levenshtein [111][112][113]) is a string metric for measuring the edit distance between two sequences. Informally, the Damerau–Levenshtein distance between two words is the minimum number of operations (consisting of insertions, deletions or substitutions of a single character, or transposition of two adjacent characters) required to change one word into the other.

The Damerau–Levenshtein distance differs from the classical Levenshtein distance by including transpositions among its allowable operations in addition to the
three classical single-character edit operations (insertions, deletions and substitutions) [114].

In his seminal paper,[115] Damerau stated that these four operations correspond to more than 80% of all human misspellings. Damerau's paper considered only misspellings that could be corrected with at most one edit operation. While the original motivation was to measure distance between human misspellings to improve applications such as spell checkers, Damerau–Levenshtein distance has also seen uses in biology to measure the variation between protein sequences [116].

It can be computed using variation of Wagner–Fischer dynamic programming algorithm. It computes the total operations needed to make distinct string to be equal. It does not allow editing string multiple times. This algorithm uses Levenshtein distance that defines the minimum number of alterations needed to make two distinct string identical. For example, The distance between CA and ABC using optimal string alignment algorithm is 3 vide CA→A→AB→ABC.

It requires less overhead. Hence, It works best at runtime operations [117]
Figure 2.16: Pseudocode for OSA (Optimal String Alignment)

Example: To make two distinct strings “barking” and “dark”, few conversions may needed to make it identical

Barking -> barkin (deletion of g)

barkin -> barki (deletion of n)

barki -> bark (deletion of i)

bark -> dark (substitution of b)

Hence, the Levenshtein distance between these two words strings is 4 [118]

Distance with adjacent transpositions
The following algorithm computes the true Damerau–Levenshtein distance with adjacent transpositions; this algorithm requires as an additional parameter the size of the alphabet $\Sigma$, so that all entries of the arrays are in $[0, |\Sigma|)$:

\[
\begin{align*}
\text{algorithm DL-distance is} \\
\text{input: strings } a[1..\text{length}(a)], b[1..\text{length}(b)] \\
\text{output: distance, integer} \\
da := \text{new array of } |\Sigma| \text{ integers} \\
\text{for } i := 1 \text{ to } |\Sigma| \text{ inclusive do} \\
da[i] := 0 \\
\text{let } d[-1..\text{length}(a), -1..\text{length}(b)] \text{ be a 2-d array of integers, dimensions } length(a)+2, length(b)+2 \\
// \text{note that } d \text{ has indices starting at } -1, \text{ while } a, b \text{ and } da \text{ are one-indexed.} \\
\text{maxdist := length(a) + length(b)} \\
d[-1, -1] := \text{maxdist} \\
\text{for } i := 0 \text{ to length(a) inclusive do} \\
d[i, -1] := \text{maxdist}
\end{align*}
\]
\[ d[i, 0] := i \]

for \( j := 0 \) to \( \text{length}(b) \) inclusive do

\[ d[-1, j] := \text{maxdist} \]

\[ d[0, j] := j \]

for \( i := 1 \) to \( \text{length}(a) \) inclusive do

\[ db := 0 \]

for \( j := 1 \) to \( \text{length}(b) \) inclusive do

\[ k := d[a[i]] \]

\[ t := db \]

if \( a[i] = b[j] \) then

\[ \text{cost} := 0 \]

\[ db := j \]

else

\[ \text{cost} := 1 \]
\[ d[i, j] := \text{minimum}(d[i-1, j-1] + \text{cost}, \quad \text{//substitution} \]

\[ d[i, j-1] + 1, \quad \text{//insertion} \]

\[ d[i-1, j] + 1, \quad \text{//deletion} \]

\[ d[k-1, \ell-1] + (i-k-1) + 1 + (j-\ell-1)) \quad \text{//transposition} \]

\[ da[a[i]] := i \]

\[ \text{return } d[\text{length}(a), \text{length}(b)] \]

Figure 2.17: Pseudocode for Adjacent Transposition

To devise a proper algorithm to calculate unrestricted Damerau–Levenshtein distance, note that there always exists an optimal sequence of edit operations, where once-transposed letters are never modified afterwards. (This holds as long as the cost of a transposition, \( W_T \), is at least the average of the cost of an insertion and deletion, i.e., \( 2W_T \geq W_I + W_D \) [119] Thus, we need to consider only two symmetric ways of modifying a substring more than once: (1) transpose letters and insert an arbitrary number of characters between them, or (2) delete a sequence of characters and transpose letters that become adjacent after deletion. The straightforward implementation of this idea gives an algorithm of cubic complexity: \( O(M*N*\text{max}(M,N)) \), where \( M \) and \( N \) are string lengths. Using the ideas of Lowrance and Wagner,[117] this naive algorithm can be improved to be \( O(M*N) \) in the worst case.

Applications

Damerau–Levenshtein distance plays an important role in natural language processing. In natural languages, strings are short and the number of errors
(misspellings) rarely exceeds 2. In such circumstances, restricted and real edit distance differ very rarely. Oommen and Loke[120] even mitigated the limitation of the restricted edit distance by introducing generalized transpositions. Nevertheless, one must remember that the restricted edit distance usually does not satisfy the triangle inequality and, thus, cannot be used with metric trees.

DNA

Since DNA frequently undergoes insertions, deletions, substitutions, and transpositions, and each of these operations occurs on approximately the same timescale, the Damerau–Levenshtein distance is an appropriate metric of the variation between two strands of DNA. More common in DNA, protein, and other bioinformatics related alignment tasks is the use of closely related algorithms such as Needleman–Wunsch algorithm or Smith–Waterman algorithm[121]

Fraud detection

The algorithm can be used with any set of words, like vendor names. Since entry is manual by nature there is a risk of entering a false vendor. A fraudster employee may enter one real vendor such as "Rich Heir Estate Services" versus a false vendor "Rich Hier State Services". The fraudster would then create a false bank account and have the company route checks to the real vendor and false vendor. The Damerau–Levenshtein algorithm will detect the transposed and dropped letter and bring attention of the items to a fraud examiner.

Export control

The U.S. Government uses the Damerau–Levenshtein distance with its Consolidated Screening List API.