Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 3, Issue 7, 2025

RISK DETECTION AND AVOIDANCE FOR THE SELECTION AND DEVELOPMENT OF AN EXTRACT TRANSFORM LOAD TOOL

Muhammad Daud Khan[1]*, Amir Ullah Barki[2], Dr. Hayat Ullah[3]

[1]*Lecturer Department of Computing & Information Technology Gomal University DIKhan- Email: daudkhan@qurtuba.edu.pk*

[2]*Lecturer Govet Degree College DIKhan- Email: amirbarki785@gmail.com*

[3]*Lecturer Department of Computing & Information Technology Gomal University DIKhan- Email: Hayyat.ullah2468@gmail.com*

**Corresponding Author)- daudkhan@qurtuba.edu.pk*

DOI: https://doi.org/

Keywords (*ETL, Data warehouse, Harmful Scripts, Cyber Security, Detection, Avoidance)*

Article History
Received on 08 June 2025
Accepted on 29 June 2025
Published on 26 July 2025

Corresponding Author: *
Muhammad Daud Khan

Abstract

Detecting malware has emerged as one of the most important challenges in the field of computer security. As Time bomb scripts (Techniques) are capable of working as well as run automatically on a predefine date and time during data operations, or during the Extract Transform Load (ETL) process in a data warehouse, a malicious script is being uploaded in which time bomb scripts will have the ability to work. The malware data oriented is hidden in the data warehouse by the hacklers and can be run on the hacker's chosen date and time which can lead to huge losses and huge financial losses, so there should be a solution to filter the data uploaded in the data warehouse Whether it's a plain or clean file because today's communications infrastructure is subject to intrusion by various malware infection tactics and attacks. One of the most difficult tasks in the field of computer security is detecting malware. This vulnerability will lead to the intrusion (and attack) of a successful computer system, on the other hand, leading to the success of more sophisticated attacks such as the Distributed Daniel of Service (DDoS) attack. Data mining methods can be used to eliminate the limitations of signature-based techniques in detecting zero-day malware. This study discusses in detail the malware and malware detection systems that use existing methods, such as data mining techniques, to identify known and undetected malware patterns.

INTRODUCTION

Most of the approaches that we surveyed adapted traditional testing and evaluation approaches to the area of data warehouse testing, these approaches consisted of three modules Extract, Transform and Load (ETL), we identified that thoroughly file checking was not performed during the data loading phase so the malicious script might have hidden in files easily and can corrupt the database, consume resources or cause other security and performance issues during the data loading phase. It was also a lack of checking for any form of risk in the uploading file content, which is a significant danger for any form of data storage server and can result in significant damage to a business dependent on data warehouse, the researcher added a new module that is divided into two sub-modules (Detection and Avoidance). Data from various sources will be extracted from the sources first, as in a normal ETL process, but data will not be processed to the transform phase until it is checked through the researcher modified proposed model. Obtaining the necessary business insight for expansion takes a lot of time, effort, and the correct data analytics tools. Many data reporting systems do not have the storage capacity to keep all of the data from multiple studies. You frequently wind up with wrong or missing data, which can interrupt your business operations and reduce income. A business intelligence product with ETL capabilities or a dedicated ETL tool can be an effective solution.

Extract, Transform, and Load (ETL) are the three major phases in data integration and migration. The ETL process can help you develop a data warehouse, a data lake, or a data hub by synthesizing silos of data from different sources and assuring an accurate, dependable, and optimized data flow. At times, the inclusive test bed is unavailable. There is an insufficient flow of business information. During the ETL procedure, data loss is possible. Several confusing software requirements exist. The ETL process can help you develop a data warehouse, a data lake, or a data hub by synthesizing silos of data from different sources and assuring an accurate, dependable, and efficient data flow. While ETL can be beneficial to your organization, it also has its own set of issues that you should be aware of. Different hackers can use various scripts to cause inefficiencies, performance issues, and operational downtime [6]. The sheer amount of data available is one of the most prominent ETL problems. It's not rare for ETL tool maker to make mistakes, as well as hackers, when working with large data sets. Because some procedures in the transformation step may not have been executed correctly, you may end up with data loss, corruption, or irrelevant data. You may also encounter numerous bottlenecks as a result of limited memory or CPU power.

ETL also faces the issue of disparate data sources from various time bomb oriented scripts from the hackers. Not every source database and destination system are aligned, which means their coded mappings aren't the same. In such instances, you might need to do a variety of data transformations. That negates the purpose of ETL entirely.

Any type of malicious scripts or malicious code that hooks into private information, computer system data, computing operations, or (and) simply to perform the hacker's malicious goals without the authentic permission of that computer user or database administrator. The anti-malware tools can't detect malicious codes installed on the system because some hackers install time bomb scripts that can't be detected or removed by the anti-malware tools. Data mining techniques are used to identify both known and unknown malware samples. Data mining approaches can be used to overcome the limitations of signature-based techniques in terms of detecting zero-day malware. Malware and malware detection systems are addressed in depth, with modern technologies such as data mining techniques being used [9][14].

2. Data Warehousing:

The term "Data Warehouse" is derived from the phrases "Data" and "Warehouse." The raw facts and figures are referred to as data, and a warehouse is a structure used to store data. Furthermore, an information warehousing framework can be defined as a collection of strategies, systems, and tools that assist the maintained information laborer (someone who works primarily with data or creates and uses information in the workplace: for example, a corporate director or an information investigator) in making decisions by converting data into information. [4]. The ever-increasing volume of data within a corporation or organization necessitates the necessity for data storage because each department inside the company prefers to maintain data relevant to their immediate requirements, many details have surfaced. Departmental information, on the other hand, may not meet the largest requirement when it comes to employing organizational data in the larger picture. When information is dispersed across departments, it can be more difficult and time-consuming to extract data from a number of information sources. As a result, delays in decision-making can occur, as well as the squandering of opportunities and the inability to identify where the business might generate money or where targeted efforts may be made. Massive amounts of data, data analysis, and business intelligence are all problems that data repositories solve. In this situation, data usage and consumption may be derived from a single storage point that gives all data and storage equal value. The use of a database will need a coordinated effort between current systems and the implementation of a corporate database. An old data strategy should be updated to reflect current requirements and how they should be addressed. The requirement for a single data storage facility is critical in this respect.

By connecting the various information spread throughout the firm, the types of data held by Data Warehouse (DW) are utilized in report management, different business enquiries, decision support systems, management information systems, and data mining applications. All of these advantages may be obtained by utilizing a company data repository. (Abiodun Bamidele Obisesan, 2021).

3. Extract, Transform, Load:

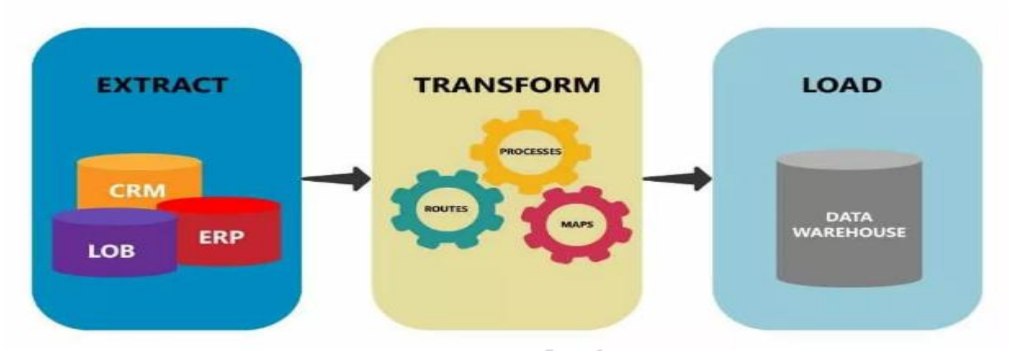


Figure 1. Extract Transform Load

Extract, Transform, Load (ETL) is a time-consuming process, especially when dealing with a large, heterogeneous network. Many suppliers make operating systems and apps for computers easier to use, and customers are prompted to fill out information for each of the platforms they want to use. ETL is an automated process that takes data from a variety of sources, extracts the knowledge needed for analysis from the data, converts the extracted information into a format that may suit the business's needs, and loads it into a data warehouse. By doing particular types of analysis, ETL frequently summarizes data to reduce its size and increase efficiency. After you've built an ETL infrastructure, you'll need to integrate data sources and plan and test rigorously to ensure that source data is transformed appropriately into the data warehouse. Because ETL is the cornerstone of the data warehouse, it is not feasible to build one without it. This ETL process is important because it determines the quality of data in the data warehouse, which is necessary for the data warehouse to be used for business intelligence or other analytical purposes. The data warehouse is divided into numerous phases, each of which is interactive, including cleaning the data, Integration of data, Selection of data, Data Transformation, Data Processing, Pattern Evaluation, and Knowledge Presentation. Users are involved either directly or indirectly through the knowledge domain. An ETL process is successful if it includes numerous steps, including extracting data from a source, preserving data quality, applying standard rules, and presenting data in multiple formats for decision-making [15][1].

ETL collects data from unique RDBMS source structures and transforms it using commercial industry common sense, concatenation, and other techniques. Then, in the Data Warehouse, load these statistics. The data is put into the Data Warehouse as reality tables and dimensions. The source structures are particularly useful for extracting records over a given time period. This is a fraction of the time it takes to load all of the data. As a result, the staging location allows you to collect data from the supply device and keep it in the staging site until the time window expires. A data warehouse is a topic-oriented, integrated, time-varying, on-risk collection of records that aids in the decision-making process of control.

An ETL tool extracts data from various types of heterogeneous data sources, transforms it (using calculations, joining fields, keys, and deleting incorrect statistical fields, for example), and inserts it into a Data Warehouse. Using this data, you may create useful reports, dashboards, and visualizations using a variety of business intelligence (BI) tools. Extraction-Transformation-Loading (ETL) equipment is a set of specialist tools for dealing with warehouse homogeneity, cleansing, and loading issues. ETL (data purification) and ETL (data integration) tools are expected to account for at least one-third of the effort and costs in a warehouse project's budget, and this might increase to 80% of the development time [35]

The data is segregated from many sources in the primary stage, which is referred to as source data stores. OLTP (Online Transaction Processing), legacy frameworks, data from websites, spreadsheet entries, basic content documents, photographs, and even video streaming are examples of these. The data is then relocated to a designated area in the stockroom known as the Data Staging Area (DSA) in the second stage. The generated data is double-checked for business rules and respectability needs, as well as mapping adjustments, to ensure that the data is compatible with the Data Warehouse's goal. Separating and varied checks are among the adjustments. The next step is to load this modified data into the central Data Warehouse (DW) along with all of its partners, such as information stores and data sees. The traditional Data Warehouse setup needed the ETL technique to enliven the data during inactive or low-stacks, and the jobs to be completed within a defined timeframe [26].

4. Extract

Data extraction refers to the process of extracting data from a variety of sources. Each source of data has its own set of characteristics that must be handled and integrated into the ETL system in order for data extraction to be efficient. The ETL process necessitates the integration of systems that differ in the following ways: Database management systems and operating systems are two types of systems. Prior to the translation of physical data, hardware components, protocols, and a logical map of data are required [5]. The substrate surrounding the patch is simply removed, either completely or partially, to create heterogeneous substrates. By suppressing the surface waves, this method has been utilised to boost the gain of a Microstrip antenna. The antenna has also undergone substrate removal, which has significantly improved bandwidth and efficiency.

5. Transform

After extracting data from various sources, the second part of the ETL process is transformation. The data retrieved from sources may not be useable immediately after extraction; it must be cleansed, mapped, and changed before it can be used. To detect errors in the data, procedures such as data quality validation and data auditing are frequently used during this process. Consider the following scenario: A distinct spelling of a comparable person, such as Jon, John, and so on. A company's name can be expressed in several ways, for as by using alternative names like Cleaveland or Cleveland. Diverse applications for a same customer may create distinct record numbers. Particular records that are essential for certain types of information are kept clear. Mixed-ups can occur when an invalid item is collected at the POS as a manual passage. Cleaning, rebuilding, duplication elimination, configuration and joining, and splitting processes all happen during the transformation process. (N & V, 2016).

6. Load:

Data is moved from the staging area to the data warehouse's data storage tables during loading, the final stage of etl processing. The load part makes use of data manipulation language (dml) operations like truncate, update, insert, and merge to add new records to data storage tables. In any data storage environment, a loading strategy controls how data is delivered between the platform and the target tables for storage. The etl process structure, data amount, loading schedule, and target table type all have an impact on this approach. To ensure that the etl process is not interrupted, it is necessary to design a loading strategy that minimizes the impact on the data warehouse. This also ensures that other data storage procedures and business reporting requests may continue as usual [11].

7. Algorithm:

An algorithm is a well-defined technique for solving a problem that consists of a series of stages that take a set of input values and produce a set of output values. It's a specific definition of how to solve the problem, and it specifically outlines the technique so that a computer can implement it. It is critical that the method is valid and that it completes in an acceptable amount of time. This may necessitate a mathematical examination of the method in order to establish its accuracy and efficiency, as well as to demonstrate that it is terminated within a reasonable timescale. If there are multiple algorithms to solve a problem, the simplest approach (e.g. fastest/most efficient) should be chosen [30].

7. Malware/ Malicious Codes/ Malicious Scripts/ Harmful Scripts:

Malware, often known as malicious code, is software that is designed to carry out an attacker's malicious and harmful intent on a victim. This is done in order to acquire access to network resources, personal computers, and personal information without the authorization of the system owner. Malware comes in many forms, including Trojans, Viruses, Worms, Rootkits, Spyware, Adware, and so on. More than 10,000 codes have been reported to date, and the number is growing every day. Attackers take advantage of flaws in online browsers, operating systems, and social engineering techniques to persuade users to run malicious programs on their computers in order to propagate malware. To avoid detection by existing protections such as antivirus, firewalls, and gateways that use signature-based processes and cannot detect invisible malware, malware authors employ a variety of obfuscation techniques such as registry insertion, command replacement, dead code insertion, code modification, and so on [46].

8. Literature Review:

Since last decade one of the most difficult tasks in computer security is detecting malware. Data mining techniques can be used to circumvent the limitations of signature-based techniques in detecting zero-day malware. This research looks into malware and malware detection systems that use existing techniques like data mining to find known and undetected malware patterns. Malware, sometimes known as malicious software, is a prevalent sort of hacker attack. Crypto miners, viruses, ransomware, worms and time bomb scripts techniques are examples of malware families. Despite the fact that global hackers attack detections remained unchanged from the previous year, there was a notable movement from consumer to business targets. Hackers mostly used to damage the date warehouse date through the 5 Most Common Types of Malwares i.e., Malware, Crypto mining, Mobile malware, Botnet, Info stealers, Trojans. There malware, Protection and time bomb scripts insertion. In order to overcome the limits of signature-based techniques in zero-day malware detection, data retrieval techniques can be applied. Many researchers are working to overcome as well as reduce the hackers' attacks through various algorithms, ETL techniques specially to prevent data by time bomb scripting techniques.

9. ETL in Data Warehouse:

In order for the data warehouse to help with business analysis, data must be constantly added to it. Data must be extracted and copied into the data warehouse, which will be used for future analysis and decisions, from one or more functioning systems. Massive amounts of data from disparate systems must be joined, restructured and combined to create a new unified knowledge base for business intelligence, which is the most difficult task in a data warehouse scenario. Extraction, transformation, and loading (ETL) is a term used to describe the process of gathering data from many sources and operating systems and transferring it to a warehouse. ETL is a general word that covers a wide range of activities rather than three discrete phases in the project life cycle. Data must be transferred between applications or systems in order to integrate them and provide at least two apps with an analogous image of the world. The majority of this data sharing was handled via techniques that were similar to what we now refer to as ETL [8].

With the help of ETL, companies may carry out in-depth analyses of their company data to make critical business decisions. Transactional databases will be unable to address complex business concerns that ETL will be able to handle. One is a conventional store of data, while the other is used to move information from different sources into one. As data sources evolve, the data warehouse will be updated on a regular basis. For a data warehouse project to be successful, an ETL system that is both well-designed and well-documented is very essential. The norms for transformation, accumulation and calculation of knowledge should be confirmed. The ETL technique allows for sample data comparison between the source and the target systems. When using ETL, complex transformations can be carried out, and as a result, more storage space is required. ETL makes it easier to move data into a data warehouse. Ensure you have at least one solid system by converting to a number of formats and styles. To retrieve and transfer data from one database to another, an ETL technique must be followed. ETL in the data warehouse gives rich historical context for business and decision-making. It improves efficiency by encoding and reusing data without requiring a lot of technical know-how on your part [7].

10. Cyber Security

The introduction of computer technology has had a significant impact on humanity. Information technology, as a component of computer technology, has had a significant impact on making things easier when dealing with large amounts of data. Despite the effectiveness of computer technology, reliable files saved on computers, cell phones, and the Internet remain subject to hackers and other forms of illegal cyber access, necessitating the implementation of effective cyber security procedures. The Internet is a modern example of communication technology usage. Since its beginning, the Internet has been capable of displaying the hopes and anxieties of its users. The internet allows one to join the world of real insects without having to smell them. The internet has altered how we live, work, learn, and operate our enterprises. We've turned the planet into a global village with interconnected land. We can have casual discussions, send text messages, and do real-time commerce with people in another nation in the shortest time feasible. Personal computers, mobile technology, simple internet access, and the rising market for new related communication devices have transformed how we spend our leisure time and conduct business. The method criminals commit crimes has also evolved as a result of technological advancements. For the dishonest, global digital discovery has opened up new options. Victims are now being harassed and subjected to violent attacks using computers and other network-related devices. Even terrorist operations that pose a threat to our collective well-being have been exacerbated by technological advancements [31].

11. Malicious Code or Scripting Attacks:

Malicious code is software that carries out an attacker's malicious goal. This is done with the intention of acquiring access to network resources, personal computers, and personal information without the knowledge of the system owner. Malicious code is developed for a variety of malware kinds, including Trojans, Viruses, Worms, Rootkits, Spyware, and Adware. Approximately 10,000 harmful scripts have been reported up to this point. Attackers take advantage of vulnerabilities in online browsers, operating systems, and social engineering techniques to persuade users to run malicious programs on their computers in order to propagate malware. Malware authors employ a number of obfuscation techniques, including registry insertion, command replacement, dead code insertion, code modification, and others, to prevent detection by signature-based safeguards such as antivirus, firewalls, and gateways [46].

12. Security risks to the data warehouse due to harmful data and malicious scripts:

The number of security cyber-attacks on data warehouses has been steadily increasing. To meet the aforementioned three requirements, numerous techniques have been offered. Data security refers to three primary issues: privacy, reliability, and accessibility to data. Dedicated data security techniques safeguard data against security breaches, attacks, and other dangers in advance. Data masking, encryption, and data access standards were all employed. After a security assault or a security problem, responsive data security techniques respond quickly and effectively. i.e., these tactics are only effective once a security breach has occurred. These monitor and study user behavior in order to determine whether they are hazardous or not, thereby protecting data that has eluded the protection of protective security procedures [23].

13. Malicious code / Malware Detection during ETL process:

Malicious code detection before and after the ETL process is a very difficult topic nowadays, and each detection approach has its own set of benefits and drawbacks. As a result, certain approaches alone are unable to entirely tackle the malicious code detection problem. Malicious code can refer to many different types of harmful or intrusive software, such as worms, trojan horses and rootkits. It can also refer to malware that has the ability to compromise a computer or network. Cybercriminals can steal personal information and privacy from computer users without their knowledge using malicious code. It may be possible to control prohibited computer systems and cyber sources while also destroying the trustworthiness, integrity, or availability of computer networks to counteract this threat [53].

13. Malicious Code or Malware Detection approaches:

It must first gather information linked to malicious code in order to detect and prevent malware. It provides information assistance for critical system analysis for the malicious analysis by gathering system performance at various aspects and levels. Semantic enhancement, API monitoring, integrity checking, coding testing, and operational performance are some of the important technologies employed in the data collection process. A program's hazardous nature must be identified and used as a basis for making a bad programmer selection before the severity of the programmer can be determined. Some of the structural aspects of research accomplishments both at home and abroad include call features, standard code features, and N-gram features, flow control graphs (CFG: Control Flow Graph features), command sequence features and characteristics, and more in file format. To assess if a system is malicious, use the malicious behavior judgment technique, which compares a known harmful system to the unknown one. Since the safe use of harmful code security has emerged in recent years, researchers in many fields have developed various inventive ways to get insight, such as machine learning, data mining, and graph theory [27]

14. Research Study:

Extracting data from source data, turning it into a format suited for study and analysis, and finally uploading it to a data warehouse are all part of the Extract-Transform-Load (ETL) data storage process. ETL operations can include sophisticated alterations that incorporate resources and objectives that use a variety of methods, databases, and technologies, which can lead to incorrect ETL implementation. Researchers present a method for validating ETL operations based on automated measurement tests that look for several types of differences between source and target data in their study. Completeness, consistency, and syntactic legitimacy are three forms of structural formality that researchers do during testing. Our method employs the rules specified in the ETL definitions to generate focused resource planning that ensures the proven production of each asset. Researchers concentrated on balancing tests, data quality, software testing, test assertions, and mutation analysis, but there is no method in place to detect, avoid, or remove rogue scripts. (Hajar, Ghosh, Ray 2018).

In their study, the researchers concentrated on the efficacy of various methodologies, loading issues, and tools that deal with activities that occur in the background of data warehouse facilities during the ETL process, such as data conversion, cleanliness, integration, homogenization, and data purification. Controlled data is thoroughly evaluated to identify business rules and integrity issues, as well as schema updates, to verify that the data matches the data storage requirements. In their study, they also give a suggested architecture for improving the effectiveness of data integration tools with business intelligence, but no technique for detecting, avoiding, or removing hazardous dangerous scripts is offered [26].

14. Research Methodology:

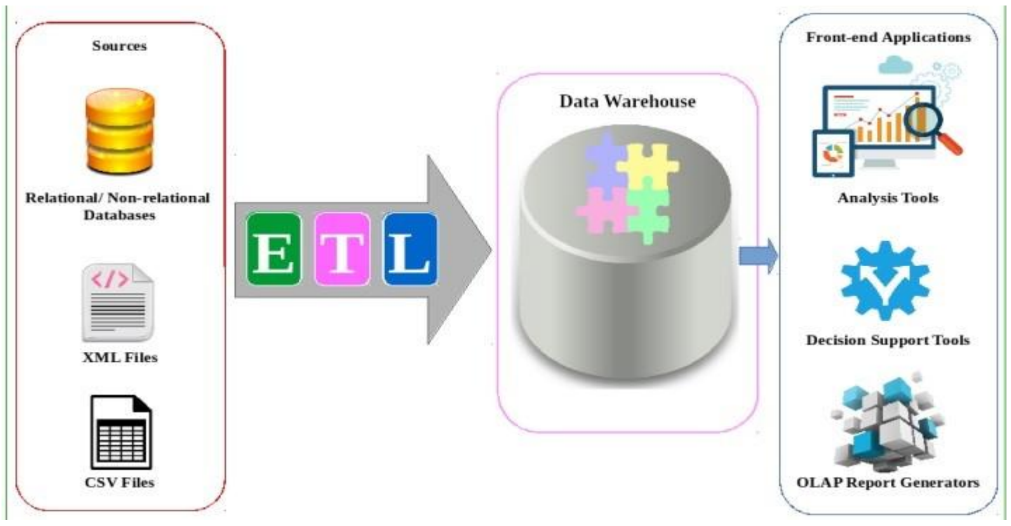
The Research Methodology was employed in order to achieve the desired outcome of the current study. We looked into existing methods for testing and evaluating each data warehouse component. We started by describing the components of a data warehouse using real-world examples from a health data warehouse project. Using multiple functional and non-functional testing and evaluation activities, we devised a classification framework that takes into account which components of a data warehouse have been tested. On to existing approaches for component testing and evaluation. The bulk of the approaches we examined utilized data warehouse testing techniques from traditional testing and assessment procedures. Answers to the observed gaps in the literature, as well as suggested research directions. This chapter covers research methodologies, data gathering, and algorithms, as well as tools for analyzing hazardous data or hazardous scripts, as well as a pilot study on a virtual machine to ensure the desired outcomes.

14. Research context:

As we all know, companies use data repositories to collect information from a variety of data analysis and research sources. Because most organizational decisions are dependent on data kept in a database, all aspects of the database must be continuously monitored. (Hajar, Gosh, and Roy 2018) start by presenting an in-depth analysis of data testing methodologies, then create and test an automated testing technique to validate the Extract-Transform-Load (ETL) process, which is a common data retention function. Hajar, Gosh, and Roy (2018) provide a classification framework that distinguishes the monitoring and assessment functions utilized in various data storage units in their study. Dynamic analysis, static testing, and manual tests are all examples of these methodologies.

The classification framework takes into account what has been evaluated in terms of the certified asset storage component, as well as how it has been evaluated using various testing and assessment methodologies. (Hajar, Gosh, and Roy 2018) talked about the specific obstacles and open-ended problems for each item, as well as research indicators. The ETL process entails extracting data from resource information, turning it into a format suitable for study and analysis, and then loading and storing the information. ETL procedures can employ complicated dynamics from one to another, as well as a variety of other resources and objectives that rely on schemes, data, and a variety of technologies. ETL techniques must be thoroughly checked because wrong usage of any ETL measure can result in inaccurate data in the data repository. Researchers suggest automated measuring tests to uncover anomalies between data in resource information and where it is stored in (Hajar, Gosh, and Roy 2018). The balancing test ensures that data extracted from source data is neither lost nor erroneously altered during the ETL process.

To begin, (Hajar, Gosh, and Roy 2018) classified and defined a set of qualities that would be assessed in the measurement test. (Hajar, Gosh, and Roy 2018) identify the various sorts of potential conflicts between source and targeted data, as well as validate three structural categories to be examined during testing, namely completeness, compatibility, and performance. Following that, (Hajar, Gosh, and Roy 2018) use the ETL modification rules provided in the placement to automatically identify the desired map construction. They list each of the many, many, many tables, records, and symbols that are used in ETL conversion. They construct test authentication automatically in order to validate test balance structures. To generate validation for each asset, they use the source map to the target. Guaranteed verifies qualities by comparing data in targeted data storage with associated data from sources. (Hajar, Gosh, and Roy 2018) investigated a health data storage system based on many data sources and several data models working on various platforms. They demonstrate that their method can discover true faults that were previously undetectable in ETL performance. (Hajar, Gosh, and Roy 2018) additionally provide a test modification to change the default changes in suggested tests in order to assess the ability to detect errors or flaws. (Hajar, Gosh, and Roy 2018) used transformation analysis to show that their automated statements may find errors in data within the targeted database while using erroneous ETL scripts on fake source data. Previously, (Hajar, Gosh, and Roy 2018) proposed a module for ETL Operations in which a file is uploaded to a data warehouse without being tested for any malicious script or malware, which can be harmful to data on the servers, infect more files, consume more and more system resources, cause bit losses, and increase query execution time.



Model by (Hajar, Gosh and Roy)

In our research work we have modified the module which was suggested by (Hajar, Gosh and Roy 2018). In this model we will add a new module to enhance the efficiency and capability as it was lack in above mentioned model it contain 2 sub-Models (Detection and Avoidance) this model will pretest any type of file which will loaded later on to Data warehouse (Server) where it will not only detect the malicious file but also will detect malicious content in the uploading file and also this 2nd module will avoid such detected file and it will refuse to further process the file into the Data warehouse (will not allow the file to be uploaded until the malicious content will be removed). This module also detects each line of the file content or any block that is being uploaded and will notify the system about these risks.

14. Proposed Model:

This section of the study is about the proposed methodology, by giving the sequence of steps for a designer to follow, during the construction of the data warehouse. Each step of this methodology is presented in terms of our ETL example (also hoping that it clarifies the nature of the employed modeling entities).

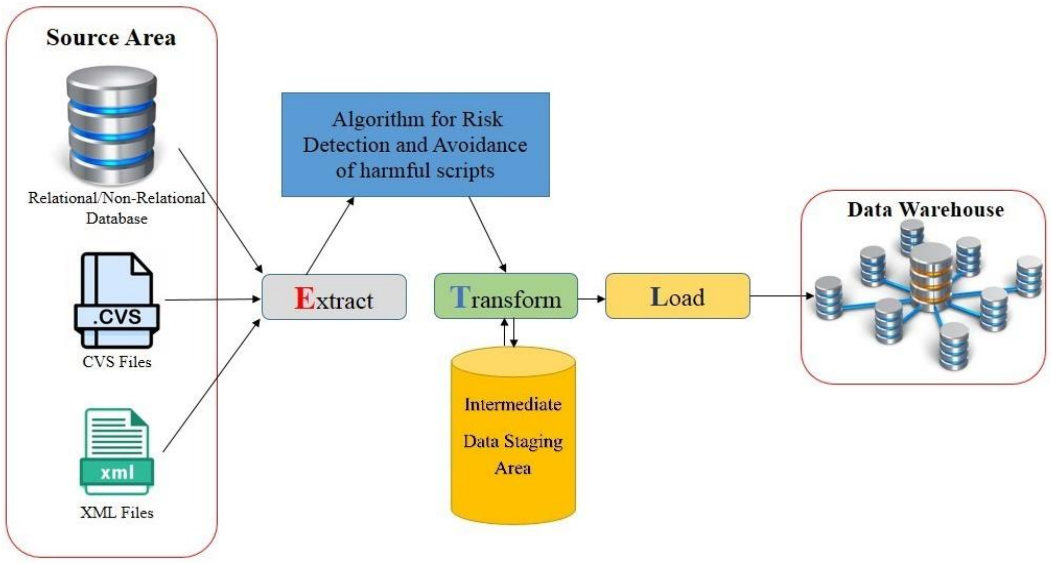

Source Area
Relational/Non-Relational Database
.CVS
CVS Files
xml
XML Files
Algorithm for Risk Detection and Avoidance of harmful scripts
Extract
Transform
Load
Intermediate Data Staging Area
Data Warehouse

Figure 3.2 Proposed Model by the Researcher (Part-1)

The above model was adapted from the model of (Hajar, Gosh, and Roy 2018), which consisted of three modules (Extract, Transform and Load). Because security, reliability, usability, and stress testing were not performed, the malicious script might easily corrupt the database or cause security and performance issues during the data loading phase. It was also a lack of checking for any form of risk in the uploading file content, which is a significant danger for any form of data storage server and can result in significant damage to a business. In the above proposed model, the researcher added a new module that is divided into two sub-modules (Detection and Avoidance). Data from various sources will be extracted from the sources first, as in a normal ETL process, but data will not be processed to the transform phase until it is checked through the researcher modified proposed model.

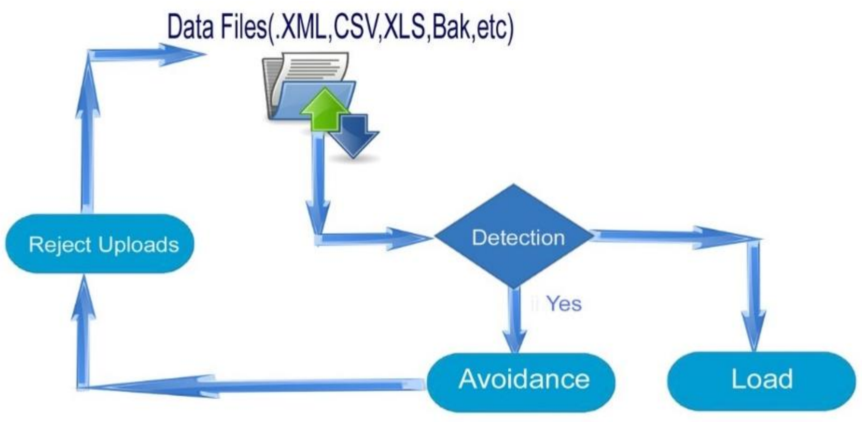


Figure 3.3 Proposed Model by the Researcher (Part-I1)

The first sub module in the above model, which is used for detecting malicious files or content, is a new addition to the (Hajar, Gosh, and Roy 2018) Model. It will pretest the files coming from various sources that are uploading to the server (Data warehouse), and it will not only read the file but also read the content in the newly uploading file before processing, and it will produce reports regarding any type of malicious files or content. This module will automate the identification of dangerous code, which was not possible with the previous paradigm (Hajar, Gosh, Roy 2019). The algorithm utilized in this sub-module is based on unsupervised learning method. In this phase below code block will pretest the file for check the data type validation where it's a valid type of data which should be allowed for the next phase of the model presented by the researcher.

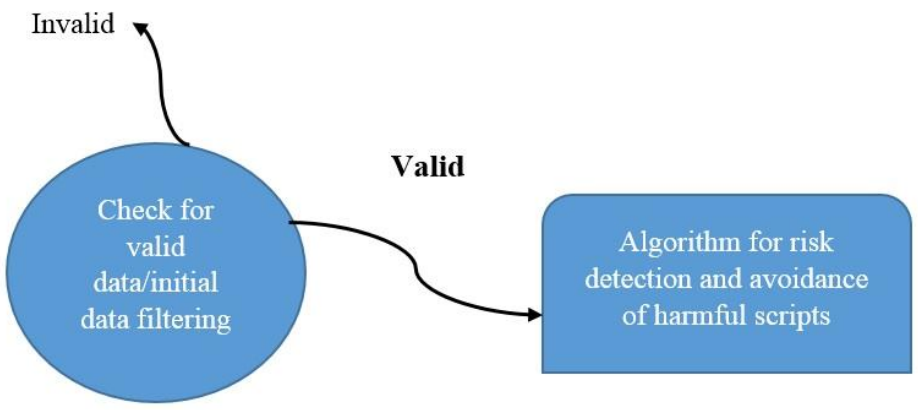

Invalid
Valid
Check for valid data/initial data filtering
Algorithm for risk detection and avoidance of harmful scripts

Figure 3.4 Check Data Validation

In the above figure initial the model will test the data which is valid for the ETL process or not after validation then it will be put up for next phase which is the actual phase of testing harmful script in the valid data to be uploaded in the data ware house.

We have seen how attacks can be made possible via the files by embedding code into it. This helps exploit the reader and target systems. To avoid this, there are various analysis methods which could be used to identify malicious scripts from the non-malicious ones. These processes can be difficult to implement and consume time, but with the growing attacks and their consequences, it becomes more and more important to invest this time and find the documents produced by the attackers. We have collected a dataset of over 24000 files comprising of 9000 clean PDF files and 15000 Malicious PDF files collected over the internet.

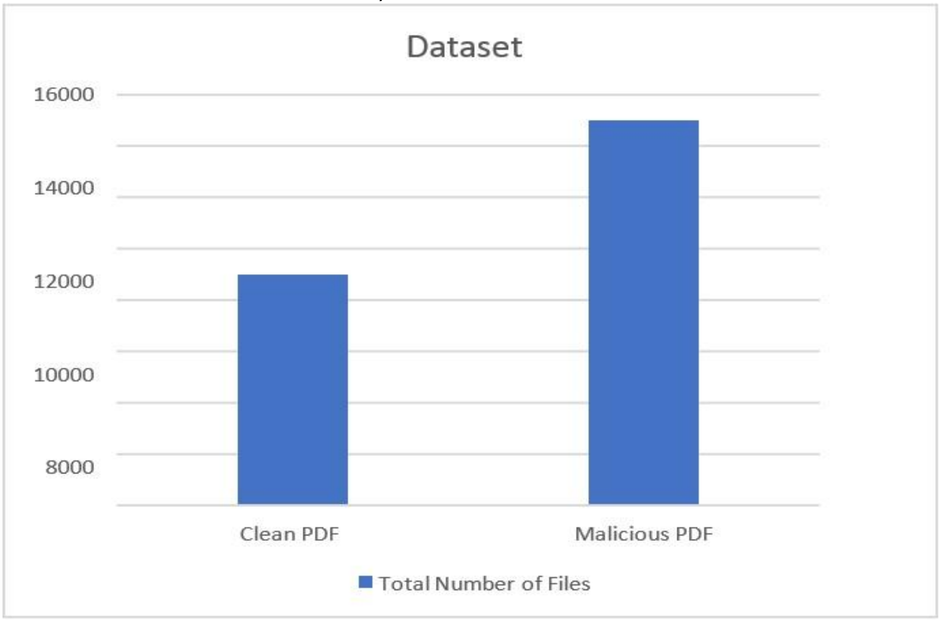
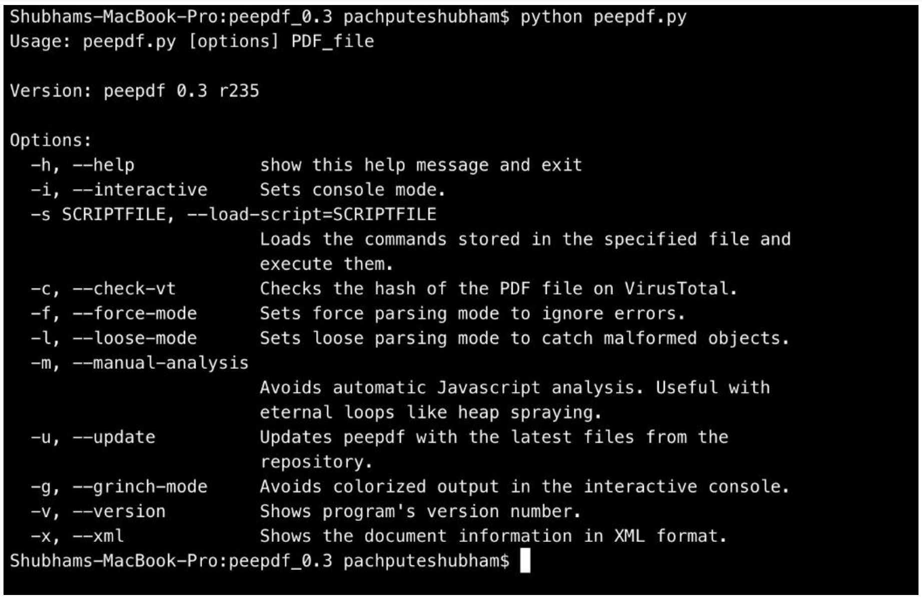


Figure 3.5 Dataset 1

It is an open-source python tool which helps to analyze a given file. It can help a security researcher give valuable information and acquire all the necessary objects required by the researcher.

Usgae:



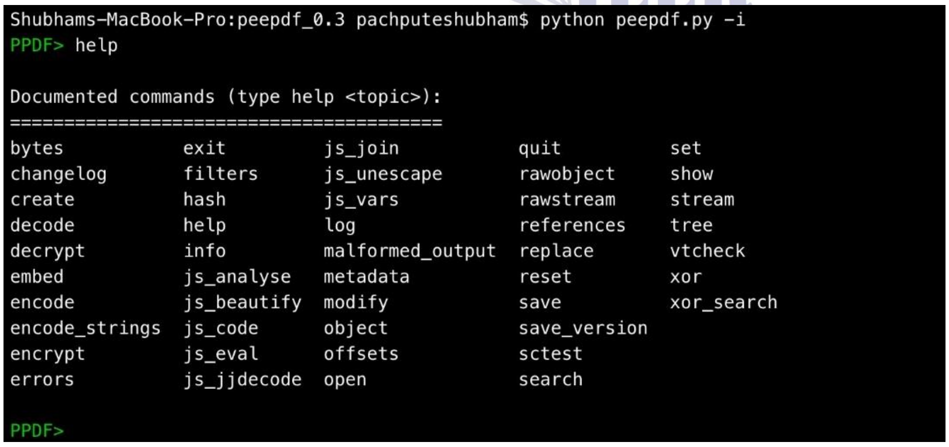
```
Shubhams-MacBook-Pro:peepdf_0.3 pachputeshubham$ python peepdf.py
Usage: peepdf.py [options] PDF_file

Version: peepdf 0.3 r235

Options:
  -h, --help            show this help message and exit
  -i, --interactive     Sets console mode.
  -s SCRIPTFILE, --load-script=SCRIPTFILE
                        Loads the commands stored in the specified file and
                        execute them.
  -c, --check-vt        Checks the hash of the PDF file on VirusTotal.
  -f, --force-mode      Sets force parsing mode to ignore errors.
  -l, --loose-mode      Sets loose parsing mode to catch malformed objects.
  -m, --manual-analysis
                        Avoids automatic Javascript analysis. Useful with
                        eternal loops like heap spraying.
  -u, --update          Updates peepdf with the latest files from the
                        repository.
  -g, --grinch-mode     Avoids colorized output in the interactive console.
  -v, --version         Shows program's version number.
  -x, --xml             Shows the document information in XML format.
Shubhams-MacBook-Pro:peepdf_0.3 pachputeshubham$
```

3.6 Usage of peepdf tool

It provides multiple options which can help a researcher dig deep and understand the structure and contents of the file.



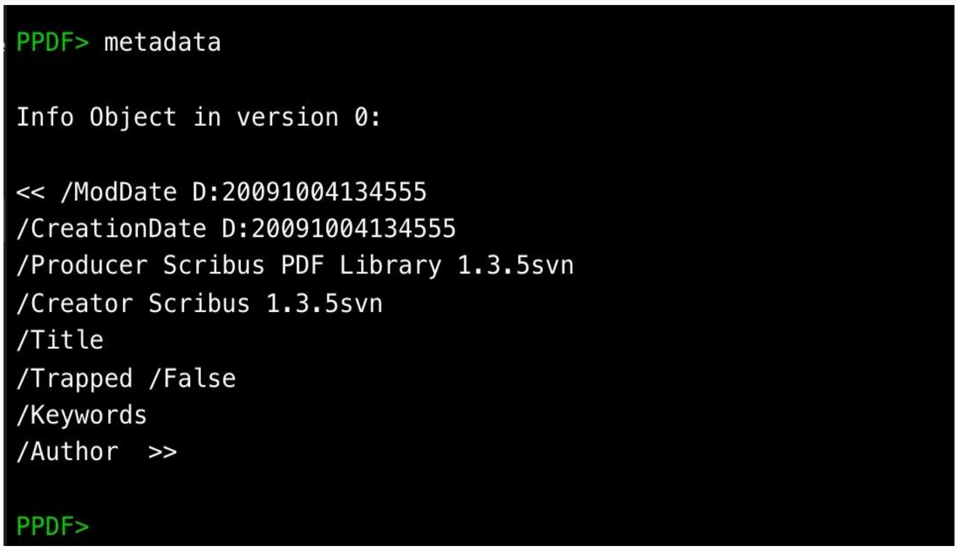
```
Shubhams-MacBook-Pro:peepdf_0.3 pachputeshubham$ python peepdf.py -i
PPDF> help

Documented commands (type help <topic>):
========================================
bytes           exit        js_join          quit           set
changelog       filters     js_unescape      rawobject      show
create          hash        js_vars          rawstream      stream
decode          help        log              references     tree
decrypt         info        malformed_output replace        vtcheck
embed           js_analyse  metadata         reset          xor
encode          js_beautify modify           save           xor_search
encode_strings  js_code     object           save_version
encrypt         js_eval     offsets          sctest
errors          js_jjdecode open             search

PPDF>
```

Figure 3.7 Tools provided by peepdf

It is an open-source python tool which helps to analyze a given file. It can help a security researcher give valuable information and acquire all the necessary objects required by the researcher.



```
PPDF> metadata

Info Object in version 0:

<< /ModDate D:20091004134555
/CreationDate D:20091004134555
/Producer Scribus PDF Library 1.3.5svn
/Creator Scribus 1.3.5svn
/Title
/Trapped /False
/Keywords
/Author  >>

PPDF>
```

Figure 3.8 Metadata of PDF Document

The process to discover these malicious documents could be forensic analysis or learning based detection.

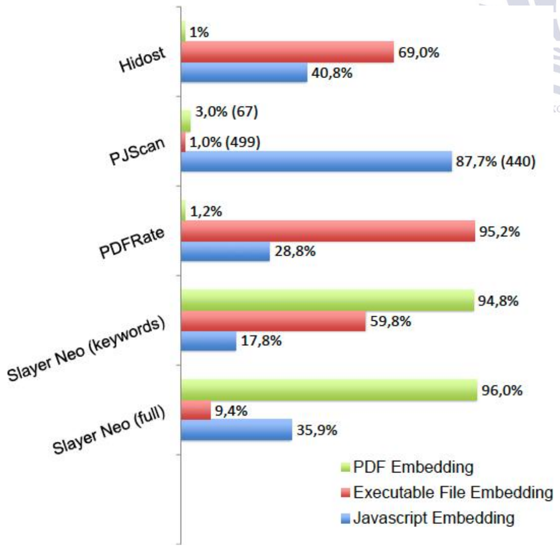
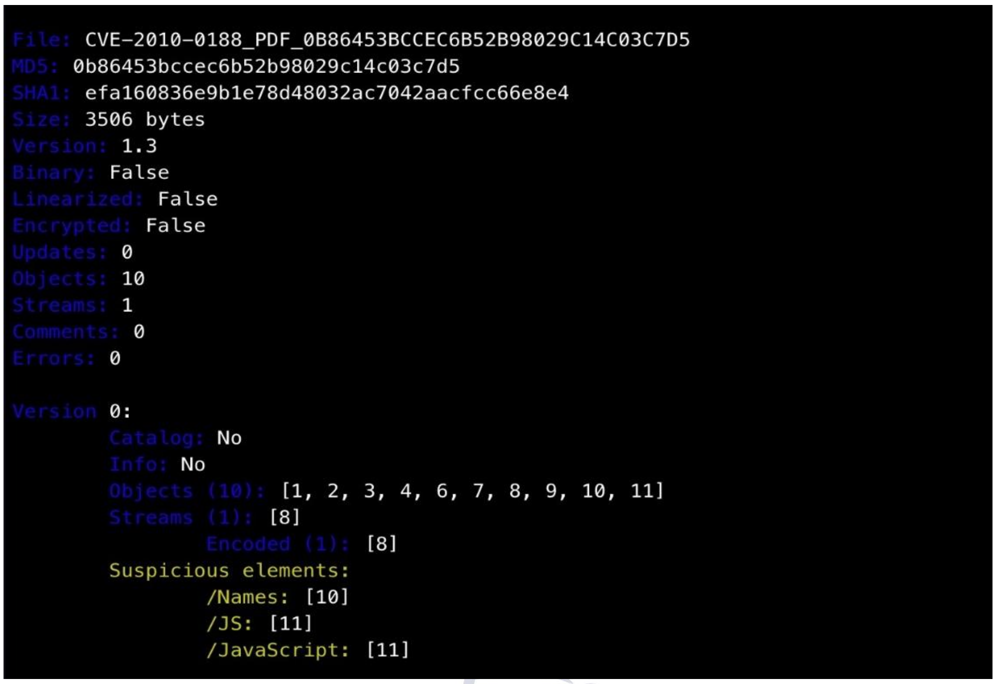


Figure 3.9 Detection rate of PDF malware

S. Pai in talks about forensic analysis as a method where we can use machine learning techniques to identify the important aspects that could classify a document as malicious or by focusing on the source of the document. If the source was a spam email or phishing website, then the content can be analyzed. Further analysis would explore the actions performed by the malicious code and their consequences on the system or the data.



```
File: CVE-2010-0188_PDF_0B86453BCCEC6B52B98029C14C03C7D5
MD5: 0b86453bccec6b52b98029c14c03c7d5
SHA1: efa160836e9b1e78d48032ac7042aacfcc66e8e4
Size: 3506 bytes
Version: 1.3
Binary: False
Linearized: False
Encrypted: False
Updates: 0
Objects: 10
Streams: 1
Comments: 0
Errors: 0

Version 0:
        Catalog: No
        Info: No
        Objects (10): [1, 2, 3, 4, 6, 7, 8, 9, 10, 11]
        Streams (1): [8]
                Encoded (1): [8]
        Suspicious elements:
                /Names: [10]
                /JS: [11]
                /JavaScript: [11]
```

Figure 3.10 Analysis of PDF document

A. Damodaran in talks about keyword-based analysis. This depends on finding the keyword objects in the document as /JavaScript or /JS. If the keyword is found, the analyst could further study the stream objects and decompress them to understand their real function. If there are no keywords found, then the document can be considered to be safe for the user to access.

15. Multiple techniques exist that evade different detection mechanisms.

Polymorphic (or at least often recompiled) malware can defeat signature-based detection, obfuscation of code flow can evade heuristics based detection, conditional statements based on environmental checks can detect and bypass sandboxes, encoding or encryption of sensitive information can help bypass signature-based detection as well as heuristic detection.

16. Generating shellcode

We will use Metasploit to generate some malicious shellcode - let it be bind TCP shell. msfvenom -p windows/shell_bind_tcp LPORT=4444 -f c Shellcodes are pieces of machine code designed to run local or remote system shell (hence the name). They are mainly used during exploitation of software vulnerabilities - when an attacker is able to control program's execution flow he needs some universal payload to execute desired action (usually shell access). This applies to both local exploitation (e.g. for privilege escalation) and remote exploitation (for gaining RCE on a server). Shellcode is a bootstrap code that leverage known platform-specific mechanics to execute specific actions (create a process, initiate TCP connection etc.). Windows shellcodes usually use TEB (Thread Environment Block) and PEB (Process Environment Block) to find address of loaded system libraries (kernel32.dll, kernelbase.dll or ntdll.dll) and then "browse" them to find addresses of LoadLibrary and GetProcAddress functions which then can be used to locate other functions. Generated shellcode can be included in the binary as a string. Classic execution of char array involves casting this array to a pointer to a function like this:

```
if file exists(path_to_file) then:
        open (path_to_file)
for each line in file: print the line of the file
```

Pseudocode for 1st checking the path of file

However, the researcher found it impossible to execute data on stack due to data execution prevention mechanisms (especially data on stack is protected from execution). While it is easy to achieve using GCC (with -fno-stack-protector and -z execstack flags) I didn't manage to do it using Visual Studio and MSVC compiler. Well, it's not that relevant. Note: It may seem pointless to execute shellcode in an application, especially since we can just implement its features in python However there are situations when custom shellcode loader or injector needs to be implemented (for example to run shellcode generated by other tool). Besides executing known malicious code (like Metasploit shellcode) is a good proof-of-concept to test detection mechanisms and bypasses.

17. Executing shellcode

The actual way to execute shellcode is a bit different. We need to: allocate a new memory region using VirtualAlloc (or VirtualAllocEx for remote processes) Windows API function,fill it with the shellcode bytes (e.g. with RtlCopyMemory function which is basically a memcpy wrapper),create a new thread using CreateThread or CreateRemoteThread function, respectively. Shellcode can be also executed using char array to function pointer cast, as long as the memory region which the shellcode resides in is marked as executable:

18. Dynamic analysis of malware

Dynamic analysis of an executable may be performed either automatically by a sandbox or manually by an analyst. Malicious applications often use various methods to fingerprint the environment they're being executed in and perform different actions based on the situation. Automated analysis is performed in a simplified sandbox environment which may have some specific traits, particularly it may not be able to emulate all nuances of the real environment. Manual analysis is usually performed in a virtualized environment and specific additional tools may be encountered (debugger, other analytic software). Both automated and manual analysis have common characteristics, in particular they are usually performed in a virtualized environment which can be easily detected if not configured (hardened) properly. Most sandbox/analysis detection techniques revolve around checking specific environment attributes (limited resources, indicative device names) and artifacts (presence of specific files, registry keys). However, there are several specific detections for automated sandboxes and other specific for virtual environments used by malware analysts.

19. Testing detection

To bypass some static detections, the application targets x64 architecture and is signed with a custom certificate. This time however we use reverse shell shellcode: msfvenom -p windows/x64/shell_reverse_tcp LPORT=4444 LHOST=192.168.200.102 -f raw We will check if the IP address of reverse shell handler (which is a very basic IoC in this case) will be extracted during dynamic analysis.

20. Detecting virtualized environment

Both sandboxes and analyst's virtualized OSes usually can't 100% accurately emulate actual execution environment (like typical user workstation). Virtualized environments have limited resources (corresponding device names can also provide useful information), may have VM-specific tools and drivers installed, often look like a fresh Windows installation and sometimes use hardcoded user or computer names. We can take advantage of that.

21. Detecting breakpoints by checking memory pages' permissions

Checking memory pages' permissions can help us detect software breakpoints placed by a debugger. First we need to find the number of pages in the process working set and allocate large enough buffer to store all the information. Then we enumerate memory pages and check their permissions - we are only interested in executable pages. For each executable page we check if it is shared with any other process. By default, memory pages are shared by all processes. When a write occurs (for example by placing an INT 3 instruction in the code), a copy of this page is mapped to the process' virtual memory (copy on write mechanism) and thus is no longer shared.

$$D(p) = \begin{cases} malicous\ if\ p\ contains\ malcode \\ bengin\ \ if\ p\ is\ a\ normal\ program \\ undecidable\ \ if\ D\ fails\ to\ determine\ p \end{cases}$$

Figure 3.11 checking memory pages

22. Exception handlers

In general, exceptions are handled first by a debugger. If we could add new or modify exception processing routines (to execute arbitrary code) we would be able to discover a debugger presence, since our code will be executed only if there's no debugger to catch the exception first. Structured Exception Handling (SEH) is a Windows mechanism for, well, exception handling. When an exception is raised and no other facility was able to process it, the exception is passed to SEH. Manipulating SEH functions during runtime can be used to detect a debugger. In x86 environment exception handlers are present in a form of linked list and the first element address is stored at the beginning of the TEB. We can add a custom handler and link it to the beginning of the list. The custom exception handler can indicate that the application is not being debugged. However, in x64 environment SEH operations are done in kernel mode (this protects the SEH data from being overwritten on the stack via a buffer overflow attack) so the aforementioned technique is generally not applicable. However, if none of the handlers is able to process the exception, it is passed to kernel32. UnhandledExceptionFilter function (which is the last resort of exception handling). It is possible to set a custom filter function that will be called from UnhandledExceptionFilter using SetUnhandledExceptionFilter function. Interestingly, our custom unhandled exception filter will be called only if the application is not being debugged. This happens because UnhandledExceptionFilter checks for the presence of a debugger using pNtQueryInformationProcess function with ProcessDebugPort flag (same as in the technique described before). So, we can register arbitrary unhandled exception filter function which will indicate absence of a debugger. Performance of the model is evaluated using the confusion matrix to test various evaluation parameters as recall and area under ROC. Following are the results and visual representations of the obtained results.

23. Results

True Positives: 10780
True Negative: 10835
False Positive: 1
False Negative: 664
Total: 22280
Accuracy: 97.0153
F-Score: 97.0078741
Recall: 94.1978
Precision: 99.9907

The performance results are computed using the following formulas:

Accuracy = (T P + T N) / T P + F N + F P + T N
F-Score = 2 ∗ (P ∗ R) / (P + R)
Recall = T P / (T P + F N)
Precision = T P / (T P + F P)

$$D(p) = \begin{cases} malicous & if\ p\ contains\ malicous\ code \\ bengin & otherwise \end{cases}$$

Evaluation Metric results obtained from the confusion matrix table show that the Accuracy and F-measure are the metrics which are used for the evaluation of the classifier performance. The F- measure is defined in terms of Recall (R) and Precision (P). If the evaluation

True Negative (TN): Malicious URL samples are labelled as a Malicious.
False Positive (FP): Malicious URL samples are labelled as a Legitimate(non-malicious).
False Negative (FN): Legitimate(non-malicious) URL samples are labelled as Malicious.
True Positive (TP): Legitimate(non-malicious) URL samples are labelled as a Legitimate (non-malicious). Metrics have higher value, then the classifier is best suitable for the data set The evaluation metrics results described effectively by the confusion matrix are:

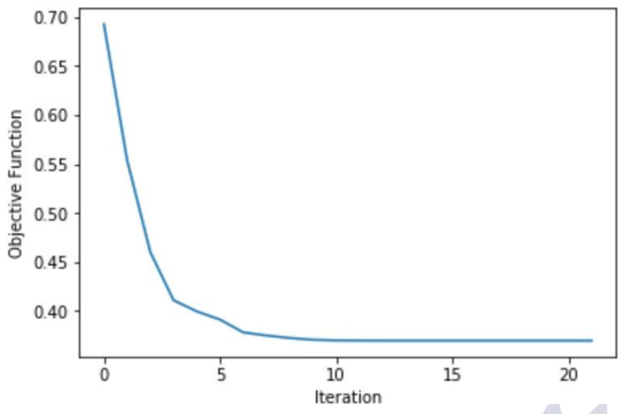


Figure 3.12. Convergence Curve

Figure 3.12 gives the Convergence curve of the objective function plotted against the number of iterations for the logistic regression method applied to URL dataset.

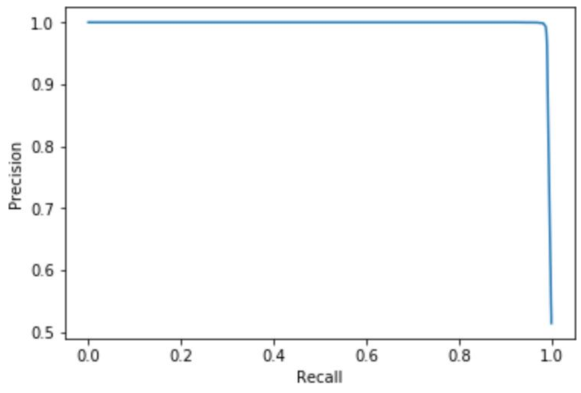


Figure 3.13. Convergence Curve

Figure 3.13 gives the precision-recall curve which shows the relationship between the precision (positive prediction value) and recall (sensitivity) for every possible cut-off. The graph is compute as:

X-axis: Recall = sensitivity = TP / (TP + FN)

Y-axis: Precision = positive predicted value = TP / (TP + FP).

```
areaUnderROC: 0.9956972387097107
```

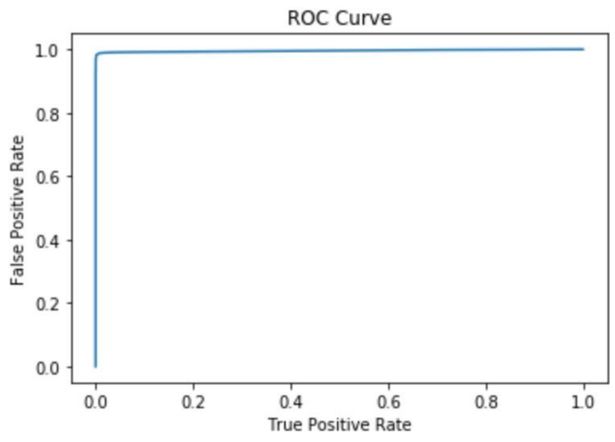


Figure 3.14. AUC-ROC

Figure 3.14 shows the AUC-ROC (area under curve - receiver operating characteristic curve) which graphically illustrates the diagnostic ability of the logistic regression model, which has varied discrimination threshold. The ROC curve is created by plotting the false positive rate (FPR) known as fall-out or calculated as 1-specificity, against the true positive rate(TPR) known as sensitivity or recall, at various threshold settings. Python tool which analyses a file to determine the maliciousness of a file is implemented to give a verdict on a given input. This script reads through the content of a given file and identifies the objects which are suspicious and gives the verdict based on the same. The results from scanning a number of mixed, benign and malicious files is given below:

Figure 3.14 shows th

Thus, we can see that the tool is working with 92.92% accuracy on identifying the malicious files and 92.46% accuracy on identifying the benign files when the entire input set is a mixture of both malicious and benign files.

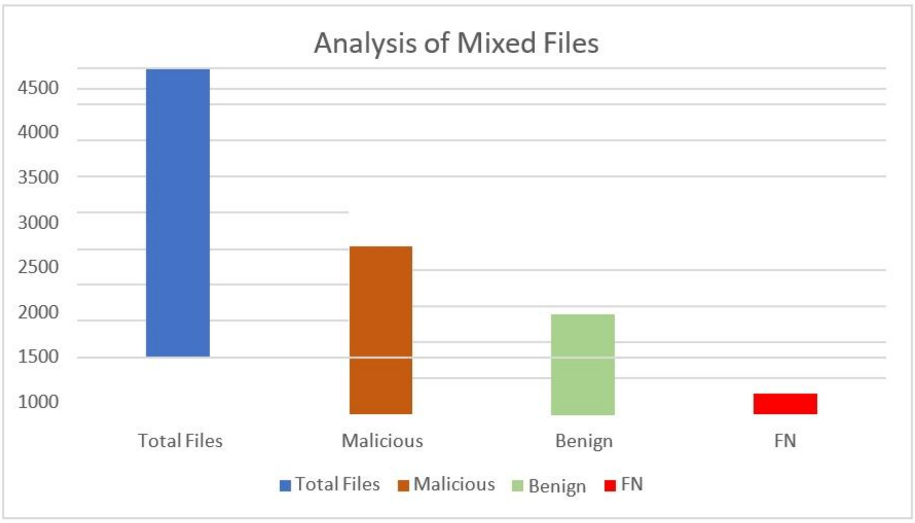


Figure 3.15: Analysis of Mixed Files

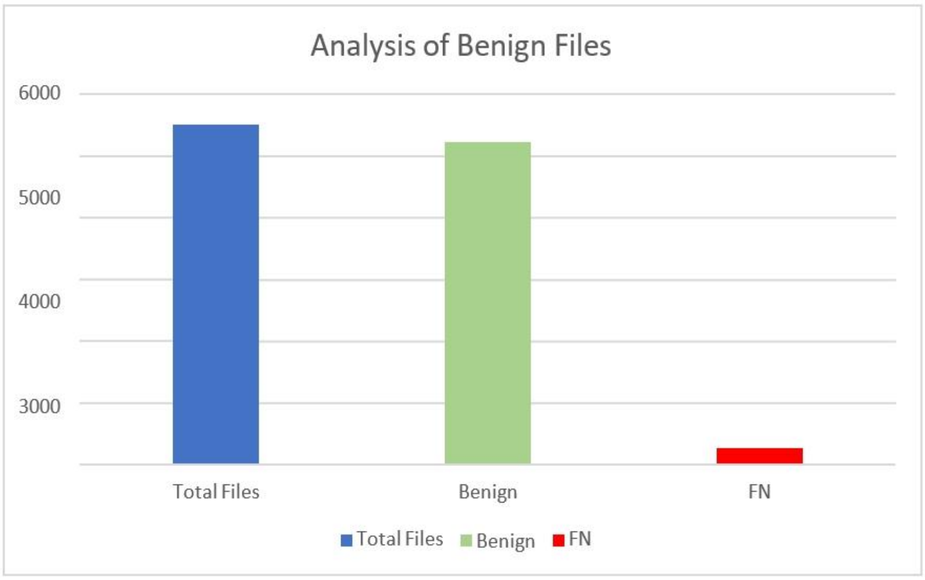


Figure 3.16: Analysis of Benign Files

Thus, we can see that the tool is working with 95.07% accuracy on identifying the benign files when the entire input set is of benign files.

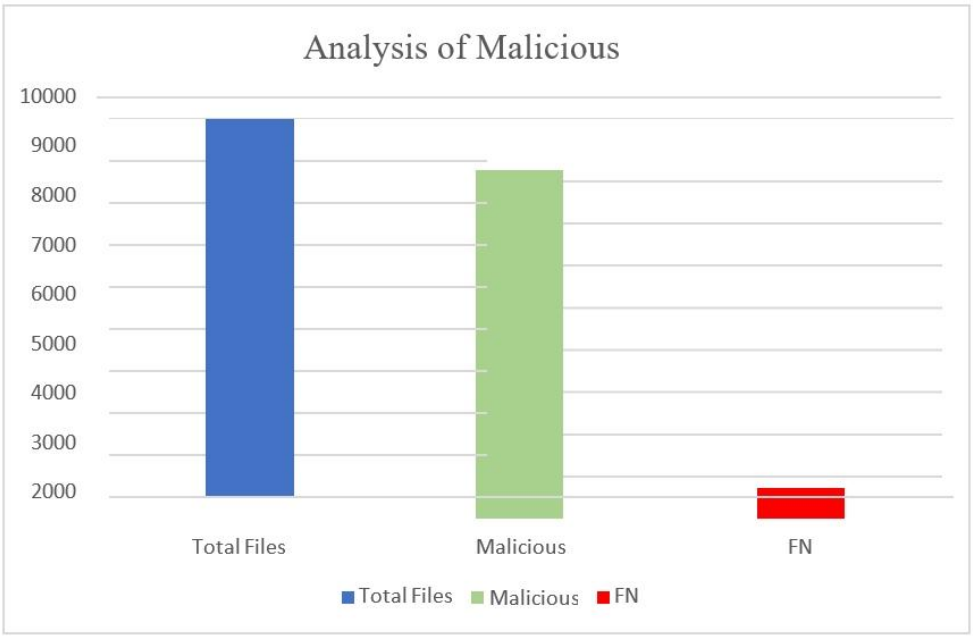


Figure 3.17. Analysis of Malicious Files

Thus, we can see that the tool is working with 92.07% accuracy on identifying the malicious files when the entire input set is of malicious files.

23. Results & Discussions

Here we elaborate the results and initially code test under file type, line evaluated in code, code evaluation time, risk avoidance percentage during code execution and at the end observed embedded script line for test my overall all results. Perform all my test on CCSV, Xlx, PDF etc. file type from date wager house.

Now a day's server plays an important role in the situation where data have been stored and manipulate for various business decision and also where it provide variety of merits it's also suffers from various malicious attacks like data theft denial of services where the invaders used different ways to harm the data servers, in this study the researcher developed a module to test and detect various malicious activities in code before loading data into Data warehouse. Initially the module accepts data from the source and then it tests for the malicious code like harmful script that blast on the server to harm normal activity of the server or may cause to damage the data. The researcher initially tested it with code fit in the file of XML sheet to test whether the module properly extract or split each line of code and later on to analyze the lines one by one, in the initial phase the efficiency was measure which was around 70% accuracy but in due time.

File Type	Lines evaluated	Evaluated time	Risk Avoidance %age	Embedded Scripts line for testing
CSV	2000	1.6 sec	69.98	300
Xlx	2100	1.4 sec	70.14	350
PDF	2500	1.9 sec	67.32	400

Table showing result of evaluating the Model in initial Phase

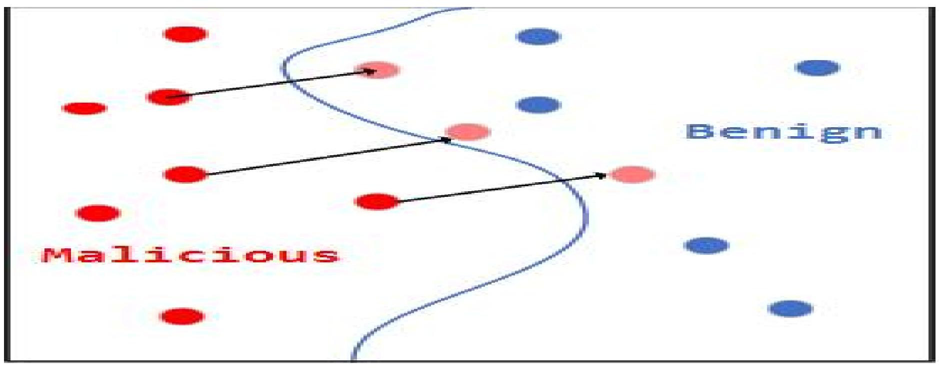


Figure 4.1. Decision Boundary

File Type	PDF File Size Range	Dimension
CSV	<10 kB	32
Xlx	10 kB – 30 kB	64
PDF	30 kB – 60 kB	128

in the above table the results showing that each file type was analyzed where in CSV, XLX and PDF files total lines were added 2000, 2100 and 2500 respectively which took 1.6, 1.4 and 1.9 respectively Sec where risk avoidance rate was 69.98, 70.14 and 67.32 respectively where harmful script was added intentional to test the model for the initial phases which give amount of percentage which is fair enough but not acceptable because risk still there in the scripts. After the researcher modified the algorithm to increase the level of accuracy by parsing each block line by line and not to skip the inverted quotes in the code like comments where the invaders or attackers may add some sort of lines to harm the servers.

File Type	Lines evaluated	Evaluated time	Risk Avoidance %age	Embedded Scripts line for testing
CSV	2000	1.9 sec	89.50	300
Xlx	2100	1.8 sec	92.56	350
PDF	2500	2.6 sec	91.50	400

Table showing result of evaluating the Model after modifying the initially flaws

After the modification of the algorithm researcher again tested module on the data /files containing the malicious code /Script, the detection and avoidance ratio of modified module was 85% to 92.56% in between the ranges with a little bit process time compromise because of line by line file checking for any sort of code which can be harmful for data warehouse during ETL process or after determined time by the hacker. Before modifying the model purposed by (Hajar, Gosh, and Roy 2018) the process time was few second less but wasn't reliable and able to test the uploading content to the data warehouse (servers) for hidden malicious codes / scripts which was quite danger but now with a little bit increase of a few seconds module is reliable and will undoubtedly make users feel more at ease due to automation, and users will no longer have to worry about malicious and harmful data because our purposed model will scan files and content within them, and if any malicious content is detected, the model will not only detect, notify the system, but will also prevent this data from being uploaded to the data warehouse. In the proposed the model the algorithm checks and detect the malicious code and alert the system about the malicious files. PDF document is more and more flexible to embed malwares into it. We studied the various functions which can help embed malware into PDF. Malicious URLs, JavaScript and Action Script code are the most exploited features of the PDF document to embed the malware.

In this report, we have also looked at some examples of how malware is embedded into the document to take advantage of the vulnerabilities found in the reader system. We have also seen it with respect to the formation of the document giving us better insights. We have seen some methods that can be used to analyze and stop such attacks. However, attackers find loopholes in these techniques and are successful in obfuscating the actual embedded code to evade from such analysis findings. With the economic gains and usage of refined methods to carry out attacks, we need to provide security guarantees for the usage of documents and thus require the development of algorithms which can help stop these attacks and prevent them. There are few vulnerabilities in the machine learning techniques which the attackers' sooner or later will exploit.

24. Results Summary and Conclusion

In current study we discussed about Computer history, Data and Information, Databases, ETL process, importance of ETL in Data warehouses, Data warehouse security, malicious code, type of malicious codes, ways of spreading code, spreading techniques and detection methods. As we know ETL process is a piece of software which plays a very important role in a data warehouse for data cleaning, integration, extraction, transformation and loading in to data warehouse, but data with malicious code also being uploaded to the data warehouse during extraction phase is a very critical and important problem.

Our current study is based on this problem, we investigate that this problem wasn't focused in previous research (Hajar, Gosh and Roy 2018). This problem may increase the risk of malicious code uploading to the data warehouse during data extraction from different sources, reliability, usability, and stress testing were also not performed which can cause severe damage to the data warehouse. The malicious code / scripts might easily corrupt the database or cause security and performance issues during or after the data loading phase. We modify the previous module and proposed two sub modules solution for malicious codes detection and avoidance which increased the efficiency, security, reliability and usability also reduced the risk of harmful scripts being uploaded to the data warehouse.

These 2 sub modules also automate the malicious code detection and avoidance process line by line go through inside a file to detect malicious codes hidden in image, doc, text, cvs and xls type of files etc. It reduces cost, maintenance, data corruption, and time and system resources consummation. Researcher's modified model is reliable, stress tested and user friendly. During Researcher's module testing phase it wasn't easy to implement in real time environment for the Researchers because of lack of resources, it needs fully functioned data warehouse with huge amount of memory, processing and data storage, that's why we tested this module on a virtual environment.

Data Availability Statement: The data is available as per request.
Acknowledgments:
Conflicts of Interest: The authors declare no conflict of interest.

References

Aleem, S., Capretz, L. F., & Ahmed, F. (2015). Data security approaches and solutions for data warehouse. 9.

Ali, I., Adil, S. H., &Ebrahim, M. (2020). Intrusion detection framework for SQL injection. ArXiv, (November).

AvantikaYadav, 2019 UNIT-II: Cyber Security Threats UNIT-II: CYBER SECURITY. 1–27.

Bhatia, P. (2019). Data Warehouse. Data Mining and Data Warehousing, 388–404.

Bindal, P., & Khurana, P. (2015). ETL Life Cycle. 6(2), 1787–1791.

Chiew, K. L., Sheng, K., Yong, C., & Tan, C. L. (2018). A Survey of Phishing Attacks: Their Types, Vectors and Technical Approaches. Expert Systems with Applications.

Costello, T. (n.d.). Prepare Your Data for Tableau. 2020

Deshpande, B. (2020). A Proposal of Scala Script Generation Tool for Extract Transform Load (ETL) Operations. International Journal for Research in Applied Science and Engineering Technology, 8(7), 264–268.

El Boujnouni, M., Jedra, M., &Zahid, N. (2016). New malware detection framework based on N-grams and Support Vector Domain Description. Proceedings of the 2015 11th International Conference on Information Assurance and Security, IAS 2015, 123–128.

Fukushima, Y., Sakai, A., Hori, Y., & Sakurai, K. (2010). A behavior based malware detection scheme for avoiding false positive. 2010 6th IEEE Workshop on Secure Network Protocols, NPSec 2010, 79–84.

Gorhe, S. (2020). ETL in Near-Real Time Environment: Challenges and Opportunities.

Gosain, A., & Arora, A. (2015). Security Issues in Data Warehouse: A Systematic Review. Procedia Computer Science, 48, 149–157. doi:10.1016/j.procs.2015.04.164

Gupta, S. G. B. B. (2015). XSS-SAFE : A Server-Side Approach to Detect and Mitigate Cross-Site Scripting (XSS) Attacks in JavaScript Code.

Heena & Mehtre, B. M. (2021). Advances in Malware Detection- An Overview.

Hendayun, M., Yulianto, E., Febrian, J., Setiawan, A., & Ilman, B. (2021). MethodsX Extract transform load process in banking reporting system. MethodsX, 8(February), 101260.

Hill, R. K. (2016). What an Algorithm Is. Philosophy and Technology, 29(1), 35–59.

Homayouni, H., Ghosh, S., & Ray, I. (2018). An approach for testing the extract-transform-load process in data warehouse systems. ACM International Conference Proceeding

Series, (June), 236–245. https://doi.org/10.1145/3216122.3216149

Jena, S., Kulkarni, E., Annaldas, N., Yalameli, P., &Shingade, K. (2015). Security Applications for Malicious Code Detection Using Data Mining. 3(1), 56–61.

Khan, N., Abdullah, J., & Khan, A. S. (2017). Learning Classifiers. 2017

Khouri, S., &Bellatreche, L. (2017). Design Life-Cycle-Driven Approach for Data Warehouse Systems Configurability. Journal on Data Semantics, 6(2), 83–111.

Kurmi, J. (2017). A Reassessment on Security Tactics of Data Warehouse and Comparison of Compression Algorithms. 10(5), 847–854.

Malik, A., Ahsan, A., Shahadat, M. M. Z., &Tsou, J. C. (2019). Man-in-the-middle-attack: Understanding in simple words. International Journal of Data and Network Science, 3(2), 77–92.

Mr. Aman, Dr. Jaiteg Singh Chakraborty, S. (2016). A Comparison study of Computer Virus and Detection Techniques. Research Journal of Engineering and Technology, 8(1), 49. https://doi.org/10.5958/2321-581x.2017.00008.3

Mr. Aman, Dr. Jaiteg Singh Chakraborty, S. (2017). A Comparison study of Computer Virus and Detection Techniques. Research Journal of Engineering and Technology, 8(1), 49.

Muhammad Naseer, C. (2021). Malware Detection: Issues and Challenges.

N, M. M., & V, R. K. (2016). Methods to Enhance Transformation in Near Real Time ETL Methods to Enhance Transformation in Near Real Time ETL. (March), 1–6. https://doi.org/10.5120/ijca2016908733

Nissim, N., Yahalom, R., &Elovici, Y. (2017). USB-Based Attacks. Computers & Security.

O'Regan, G. (2018). World of Computing. In World of Computing.

Obotivere, B. A., &Nwaezeigwe, A. O. (2020). Cyber Security Threats on the Internet and Possible Solutions. Ijarcce, 9(9), 92–97

Or-Meir, O., Nissim, N., Elovici, Y., &Rokach, L. (2019). Dynamic malware analysis in the modern era—A state of the art survey. ACM Computing Surveys, 52(5)

P.S, S., S, N., & M, S. (2018). Overview of Cyber Security. Ijarcce, 7(11), 125–128.

Rabia Tahir (2018). A Study on Malware and Malware Detection Techniques. International Journal of Education and Management Engineering, 8(2), 20–30.

Raghuraman, M. (2021). An assessment of ETL TOOLS.

Ragulan, B., & Subash, R. (2021). Designing a Data Warehouse System for Sales and Distribution Company. February.

Razak, M. F. A., Anuar, N. B., Salleh, R., &Firdaus, A. (2016). The rise of "malware": Bibliometric analysis of malware study. Journal of Network and Computer Applications, 75, 58–76.

36. S. Swapna, P. Niranjan, B. Srinivas and R. Swapna, "Data cleaning for data quality," 2016 3rd International Conference on Computing for Sustainable Global Development 2016, pp. 344-348.

Sarangdevot, P. S. S., &Tanwar, G. (2010). Improve Performance of Extract, Transform and Load (ETL) in Data Warehouse. (May 2014).

Schmidt, N., Rosa, M., Garcia, R., Reyna, R., & the, J. G. (2015). A Survey of ETL Tools Abstract: 2(5), 20–27.

Smarking, A. (2020). Beware of Exploits in ETL Exploits in Data Files.

Som, S., Sinha, S., &Kataria, R. (2016). STUDY ON SQL INJECTION ATTACKS: MODE, 1(8), 23–29.

Sood, A. K., &Zeadally, S. (2016). Drive-By Download Attacks: A Comparative Study. IT Professional, 18(5), 18–25.

Souibgui, M., Atigui, F., Zammali, S., Cherfi, S., & Yahia, S. Ben. (2019). Data quality in ETL process: A preliminary study. Procedia Computer Science, 159, 676–687.

Sreelakshmi, K. V. (2020). Malicious Code Variant Detection: A Survey. 7(1), 245–251.

Sureddy, M. R., &Yallamula, P. (2020). Data Quality Architecture for Data Warehouses. International Journal of Research Culture Society, 4(6), 95–100.

Talib, R., Hanif, M. K., Fatima, F., & Ayesha, S. (2016). A Multi-Agent Framework for Data Extraction, Transformation and Loading in Data Warehouse. 7(11), 5–8.

Tandon, Rajat. (2020). A Survey of Distributed Denial of Service Attacks and Defenses. (August).

Vol, I. (2019). A study of Security threats on Cyber Security and cyber Law's MUKESH. 10(6), 86–89.

Zhang, B. Y., Yan, X. A., & Tang, D. Q. (2018). Survey on Malicious Code Intelligent Detection Techniques.

Zimba, A. (2017). Malware-Free Intrusion: A Novel Approach to Ransomware Infection Vectors. 15(2), 317–325.