Chapter 4
Modeling of Advanced Intrusion Detection System

4.1 Introduction

In recent days, IDSs have become vital and important part of any organization’s information infrastructure. Even business organizations and government agencies are deploying and developing enterprise-wide IDSs. However, as they move towards practical implementation of IDSs, these organizations are facing hurdles related to IDS management, IDS deployment, Data Collection and data correlation. During the literature review, it is noticed that to make any IDS /IPS effective, various systems and components are deployed, used by business organization to do sophisticated task of IDS/IPS. All these machines, devices, tools, applications and processes define the security architecture of the system.

During the implementation of IDS, the issue raised with IDS is that why IDS architecture need to studied and designed properly in comparison with other components of IDS. The answer lies in IDS architecture itself since IDS architecture is the most important component in intrusion detection implementation. An effective architecture is one in which,

- Each component, machines and processes performs role in an effective and in coordinated manner.
- Enough and correct information is processed, analyzed and produce output correctly.
- Appropriate and accurate results meet the business needs, operational requirements of an organization.

In contrast, poorly designed, implemented architecture can create network havoc, can give opportunity for intruders to compromise security and even can increase burden on system administrators with undesirable consequences, unavailability of proper data, network slowdowns, and or improper responses.

Before discussing on system architecture design of research work, following section covers various tiered models, their advantages and disadvantages to give directions for proper next system architecture design. For this research work, the system architecture is modeled
and designed with the two major components: Intrusion Detection Phase and Prevention Phase. Intrusion Detection Phase includes input data/testing dataset, training attack dataset, Existing Attack Database, Signature-based intrusion detection process, Anomaly-based intrusion detection process; Detected attacks types either ‘known’ or ‘unknown’, New attack signature generation process, Attack Data Correlation, Attack database updating process and generation of alerts process.

Prevention Phase consists of list of prevention policies provided by the proposed system, prevention policy selection strategies for system administrator and graphical user interface component. It is observed that in existing IDSs, very rarely multiple detection methodologies integrated together to cover maximum attack types. However, the preventions from detected attack found with additionally deployed IPS into information networks. So, the proposed system architecture integrates the signature based and anomaly based detection methodologies along with prevention strategies provision into a single unit.

4.2 Existing IDS Architectures

This section covers intrusion detection architectures, and will look at tiered models to give the details about how servers are deployed for intrusion detection, how various sensors are function to capture and do data collection required for intrusion data analysis, and the roles and functionality of management consoles in an intrusion detection systems and in the information infrastructure of business organization. Generally, IDS architectures are classified into single-tiered, multi-tiered, and peer-to-peer architectures which are introduced here as given in Ankit Phadia(2007).

4.2.1 Single-Tiered IDS Architecture

It is most commonly used architecture in which in an IDS or IPS collects data and processes it, rather than passing the output to another set of components which they produced. The example of a single-tiered architecture is adopted from K Scarfone and Peter Mell (2007) as host-based intrusion detection system that takes the output of system logs which are further used to identify attack patterns and to detect attacks.
This architecture offers advantages like simplicity, low cost in terms of freeware tools running on individual host terminals, and works independently. At the same time, however, a single-tiered architecture usually uses those components that are not aware of each other, reducing considerably the potential for efficiency and specific functionality.

### 4.2.2 Multi-Tiered IDS Architecture

In multi-tiered IDS architecture involves multiple components that pass information to each other and coordinate each other. Existing developed multiple IDSs, mainly consist of three primary components namely: sensors, analyzers and or a manager console. The details of each component are given below.

*Sensors* are used to perform data collection. For example, network sensors are often programmed to capture data from network interfaces. These sensors are useful even to collect data from system logs and other sources, such as personal firewalls and TCP (Transmission Control Protocol) wrappers.

Sensors collect data and pass this information to *analyzers* known as agents also, which monitors which individual host activity. These sensors and analyzers are installed, deployed and configured in particular operating environment. Analyzers are specialized to perform only certain functions. For example, an agent may use to examine TCP traffic, to examine only FTP (File Transfer Protocol) connections and connection attempts. Additionally, network-monitoring tools, neural networks, and connection-tracing tools can be used in scalable environment. Once analyzer determines an attack has determined, it sends information to the *manager console*, which can perform functions like:

- Collects and display alerts on a console
- Store information regarding attacks in a database
- Retrieve information relevant to the attack if required
- Send information to host that stops its execution
- Send configuration details for network interfaces once the access rights are changed
- Provide management console interface to network administrators.
This manager console offers advantages such that, Manager console provides greater ease in analyzing logs since log details are available at particular place. Finally, a management console provides flexibilities to IDSs to change policies and parameters, to destroy log files after they are archived, and perform other important functions related to intrusion detection.

An advantage of a multi-tiered IDS architecture includes greater efficiency and in-depth data analysis. Each component of this architecture performs the sophisticated function which is independent on other components; a properly designed multi-tiered IDS architecture can be more effective which is not possible with the simpler single-tiered IDS architecture. Even it provides complete picture of the security situation of an organization’s entire network and the terminals connected to it as compared to a single-tiered IDS architecture. However, this architecture has major drawbacks of increased cost and complexity. Additionally, this architecture has difficulty in setting up multiple components involved in it as well as its maintenance and troubleshooting challenges.

4.2.3 Peer-to-Peer IDS Architecture

Peer-to-peer IDS architecture involves sharing of information of intrusion detection and intrusion prevention between peer components, which exhibits same functionalities. This architecture is often deployed in cooperating firewalls, cooperating routers, switches. Simplicity is the key advantage of this architecture. The major problem is a lack of certain functionality since no specialized component to do operation.

4.2.4 Reasons behind Studying IDS Architectures

- Architectures make a critical difference in terms of effectiveness that intrusion detection and intrusion prevention technology produces.

- At the same time, it is important to understand that “Rome was not built in a day.” If organization is not sufficed with financial resources, use of simple single-tiered architecture is best option to develop IDS / IPS architecture. As organization grows, more resources are available, functionalities of intrusion detection and intrusion prevention efforts become clearly visualize, then architecture change is possible.
- A peer-to-peer architecture is suitable for those organizations that have enough investment to obtain and deploy cooperating firewalls but cannot offer the deployment of IDS/IPS.

- Apart from these financial views behind using certain architecture to be used by organization. Moreover, Architecture provides foundation and guidelines to design and implement next generation IDS/IPS architectures.
4.3 Architectural Design of Proposed Research Work

During the study of existing IDS architectures, it is observed that not a single architecture can be made as generic since each one is having their own merits and demerits. So, the following section gives the details about the system architecture designed and used for implementation for the research work.

Initially, the theoretical formulation is illustrated which gives an overview about the steps carried out while designing the architecture of research work. Subsequently, the mathematical basis for this architectural design is introduced and mathematical model is developed. To simplify the functionality details of architecture, it is broken into following modules: Input Data and Data Preprocessing Module, Intrusion Detection Module (including known & unknown attack detection sub modules), Data Correlation Module, Intrusion Prevention Module, and Management Console Module for system administrator. The complete details of each module are explained in following section. Finally, the chapter concludes with description about Evaluation Test Bed for proposed IDS along with evaluation parameter details.

4.3.1 Theoretical Formulation

- Issues associated with existing IDS development technologies have been identified.

- Input Data required for IDS evaluation is captured, analyzed and its preprocessing has been modeled. The details of training data, testing data have been modeled and attack details have been elaborated.

- Integration of Signature based IDS and Anomaly based IDS is modeled and its general base is formed along with execution flow details.

- The use of Data Mining Algorithm *i.e.* A priori algorithm is modeled and new attack signature generation algorithm is introduced based on modified A priori algorithm.

- The correlation between identified known and unknown attacks has been defined.
• The prevention phase for detected intrusion has been modeled with the prevention policies usages and benefits.

• To respond immediately to the generated attack once it is reported, management console is designed and proposed for system administration has been discussed.

• Evaluation Test Bed for proposed IDS has been selected and evaluation parameters for IDS evaluation are described and finalized.

4.3.2 Mathematical Formulation for Proposed Research Work

There are various ways to implement the mathematical models for scientific and engineering problems. Mathematical model aims to describe the different aspects of the intrusion detection system and its use in real world, their component interaction, and their dynamic aspects through mathematics. The approach chosen to form mathematical model for the research work is based on set theory. This set theory based model clarifies and visualize the inputs, processes or functionalities, and outputs of the given problem definition. Additionally, with the help of Venn diagram and state transition diagram, it is possible to understand the flow and operations of the designed system. The details of this model are described in next section. Before proceeding for actual model built for the research work, the importance of classification model of data mining to build intrusion detection system are elaborated by Europeanjournalofscientificresearch.com (2013) in the section 4.3.2.1.

In this section, the classification problem studied by W Lee et al. (1999), is illustrated and explains the rationale of using classification model for intrusion detection. In our experiments, information on building classification rules for anomaly detection is described.

4.3.2.1 Classification Model

In many applications, e.g. pattern recognition, we need to classify data items into one of a discrete set of possible class. These classification tasks typically require the construction of a classifier, an operation that allocates a class tag to every data item described by a set of attributes.
The field of intrusion detection requires the same application of classification while analyzing the intrusions in captured network traffic as well as in standard datasets which are unlabeled and unstructured in original nature.

In classification, specifically, let \( A = \{ A_0, A_1, \ldots, A_n \} \), where each attributes is either one of two types, discrete or numerical. Let \( C \) be the set of class labels. Let \( I \) represents the universe of all possible data items in the problem domain. Then each data item will be represented as \( x \in I \) as a vector \( [v_0, v_1, \ldots, v_n] \) such that each \( v_i \) is a value of \( A_i \). So here it is found that each data item belongs to a category, it can be assigned a class label i.e., there is a function \( f \) from \( I \) to \( C \) such that
\[
f(x) = c
\]
Where \( c \) is a valid value of \( C \). Function (or model) \( f \) is normally unknown, and is the subject of inductive learning.

**Learning of Classification Model**

Let \( D \) be the given set of pre-labeled training items, where each \( d_i \in D \) is a vector \( [x_i, c_i] \). Here, \( x_i \in I \) and \( c_i \) is the Class label of \( x_i \) as known one. Inductive learning is used for constructing an approximation \( f' \) of \( f \). That is, here we want the classifications made by the learned classifier, i.e., \( f'(x_i) = c'_i \) for each \( d_i \), to agree with the true class labels, i.e. \( c_i \), as much as possible. It is often desirable that \( f \) has generalization ability, that is, the ability to make predictions beyond the observed training data. The accuracy of a learned model can therefore be refined into two separate measurements: the training accuracy on data it was trained from, and the generalization accuracy on unseen data.

A set of ‘set-aside’ testing items, \( T \) that has the same class distribution as \( D \) and \( D \cap T = \emptyset \), can be used to estimate the generalization accuracy of the learned model. A model with higher generalization accuracy is normally preferred.

There are several machine learning approaches for computing classification models, for example, decision tree learning, rule induction, neural networks, Bayesian learning, etc. Each approach uses a different model representation, e.g. a decision tree or a set of rules, etc. and a different search strategy and heuristic for traversing the space of possible models.
Classification Rules

In many applications, it is desirable to learn a set of propositional if-then rules that jointly define the target function. Rule sets are relatively easy for humans to understand, and can easily be incorporated into (existing) general rule execution engines. One solution is to learn a model, e.g. a decision tree, and then translate the model into an equivalent set of rules, e.g. one rule for each leaf node in the tree. A better solution is to directly apply a rule learning algorithm. Rule learners can outperform decision tree learners and other algorithms on many problems, and can easily incorporate certain types of prior knowledge into the learning process.

RIPPER is the rule learning algorithm and introduced in W.W.Cohen (1995). Each RIPPER rule consists of a conjunction of conditions, i.e. attribute-value tests, and a consequence as given by R.Venkatesan et al. (2013), counted as label of the class. A condition is of the form $A_d = v$, $A_n \leq \theta$ or $A_n \geq \theta$, where $A_d$ is a discrete attribute and $v$ is legal value of $A_d$, or $A_n$ is a numerical attribute and $\theta$ is some value of $A_n$. A data item is covered by a rule if its attribute values satisfy the conditions of the rule. To a rule’s target class, i.e. their consequences, data items that belong to this class are called the positive examples, and those in other classes are called the negative examples.

A RIPPER rule is further classified and trained in two phases, a growing phase and a pruning phase. The training data set $D$ is also partitioned into a growing set and a pruning set for these two phases. A rule is ‘grown’, from an empty conjunction, by repeatedly adding a condition that maximizes the FOIL information gain criterion as per W.W.Cohen (1994) until the rule covers no negative data items in the growing data set. The FOIL information gain is measured by solving the number of examples covered before and after adding the condition to the rule. It can be interpreted as the reduction in the total number of bits required to encode the classification of all positive examples covered by the rule. After the growing phase, a rule is immediately pruned, i.e. simplified, by repeatedly deleting a condition (or a conjunction of conditions) that can lead to a more accurate rule on the pruning data set as concluded in Editorial survey (2011). The pruning stage is thus a means to improve both the generalization accuracy and the simplicity of the rule.
RIPPER can handle multiple classes. Let training data set $D$ is with $n$ classes, and then this algorithm first orders the classes in increasing order as per the condition i.e., among the ordered classes $c_1, c_2, \ldots, c_n$, $c_1$ is the least prevalent class and $c_n$ is the most prevalent. In RIPPER, it first learns a rule set that separates $c_1$ from the rest of the classes, removes from $D$ the data items covered by the learned rules, and then learns another rule set that separates $c_2$ from $c_3, c_4, \ldots, c_n$. This process continues until there is a single class $c_n$, which is called the default class is obtained.
Use of Classification Rules as Intrusion Detection Models

For any IDS, the use of classification rules, \textit{i.e.}, RIPPER rules is useful to do intrusion detection. Since, these learned rules have the standard if-then format, with minimum processing; they can be used in many rule-based IDSs in the same manner as hand-coded rules. Further, the RIPPER rules are compact and instinctive, and can be examined and altered by security experts when required and needed to be reuse \textit{i.e.}, they can be checked for rationality before being employed in the actual systems as stated in P. Devanbu \textit{et al.} (1998). Additionally, RIPPER rules have two very desirable properties for intrusion detection \textit{i.e.} good generalization accuracy and clear conditional information. There will always be ‘new’ attacks, in the forms of slight variations of ‘known’ attacks or completely new breed of attacks, after the intrusion detection models are installed and deployed for the intrusion detection. The ability to detect these new intrusions, the generalization accuracy of the rules on the unseen data, is thus critical for IDS. Real-time IDSs require simple models for the sake of efficiency. Rule execution involves computing and testing the attribute values for rule condition checking. Having concise conditions therefore contributes to the real-time performance of IDSs.

\textbf{Why not Alternative Models?}

In this work, we did not test the feasibility and effectiveness of using alternative learning techniques, \textit{e.g.} Hidden Markov Models (HMMs), neural networks, \textit{etc.} to construct intrusion detection models. We believe that, at this stage of our research, the essential task is to develop an automatic approach for constructing dataset I from the unstructured raw audit data stream and intrusion detection within. Regardless of the specific machine learning algorithm to be used, we need to supply it with data items that bear ‘learnable’ characteristics expressed in a set of meaningful attributes. Here, we need to first deal with the feature construction problem. Therefore, using a ‘sensible’ model, \textit{i.e.} classification rules, suffices to evaluate the concepts and algorithms in the research work. The brief about background algorithms studied is given in next section.

\textbf{4.3.2.2 Algorithms used to build System Model}

Recently, significant study has done on intrusion detection algorithms and their usages. Though effective algorithms are designed, implemented and evaluated for IDS performance
improvement, but mostly these algorithms are found with specific detection technique only *i.e.* either misuse-based or anomaly-based intrusion detection. In this research work, the main aim of constructing such algorithms was to utilize the data mining techniques and their relevant algorithms to improve the IDS performance with speedy, accurate data analysis and correct intrusion detection. So, this section illustrates such data mining algorithms with their basic concepts involved in algorithm construction. Since the main component of any IDS is its incoming input data form, its analysis and evaluation. So, the first basic algorithm studied for the understanding of item set, frequent item set, support and confidence followed by association rule mining algorithm used to construct rules for intrusion detection based on Apriori Algorithm theme.

**Basic Algorithm(s)**

As given in R. Agrawal, *et al.* (1993). Let $A$ is collection of traits or attributes, and $I$ be corresponding values on $A$, called things. Any subset of $I$ is called an thing situated or item set. The quantity of things in an item set is called its length. Let $D$ be a database with $n$ properties (segments). Definition of $support(X)$ as Meaning of as the rate of exchanges (records) in $D$ that contain thing in set $X$. At that point an affiliation guideline is the statement:

$$X \rightarrow Y_{r, [Confidence, support]}$$

Here $X$ and $Y$ are item sets, and $X \cap Y = \emptyset$.

$support(X \cup Y)$ is the known as rule support, and $support(X \cup Y)/support(X)$ is considered as rule confidence.

For illustration, an affiliation principle from the shell order history record (which is a stream of summons and their contentions) of a client is

$$true \rightarrow rec.humor, [0.3, 0.1]$$

Indicates that 30% of the time when the client summons trn, he or she is perusing the news in rec.humor, and perusing this newsgroup represents 10% of the exercises recorded in his or her summon history record. The effective rule generation for proposed system is implemented based on the association principles calculation emulating the principle thoughts
of Apriori algorithm as mentioned earlier. Wu Liu (2005) considered, X is frequent item set if \( \text{support}(X) \geq \text{minimum support} \).

Wu Liu (2005) observed that any subset of a continuous thing set must likewise be a successive thing set. The Apriori calculation begins with discovering the continuous thing sets of length 1, at that point iteratively processes visit thing sets of length \( k + 1 \) from those of length \( k \). This procedure ends when there are no new successive thing sets reduced. It then returns to figure decides that fulfill the base certainty prerequisite. The Apriori algorithm is outlined in Figure 4.1.

Since, we search for relationship among estimations of distinctive traits, and the (preprocessed) review information normally has numerous traits, each with a substantial number of conceivable qualities, we don't change over the information into a double database as recommended by R. Agrawal and R. Srikant (1995). W Lee (2005) claimed that in invented system execution performed memory exchange for rate. The information structure for a visit thing set has a line (bit) vector that records the exchanges in which the thing set is contained. The database is examined just once to create the rundown of incessant thing sets of length 1. At the point when a length \( k \) applicant thing set \( ck \) is created by joining two length \( k -1 \) visit thing set \( lk1-l \) and \( lk2-l \), the line vector of \( ck \) is basically the bitwise AND result of the line vectors of \( lk1-l \) and \( lk2-l \). The backing of \( ck \) can be computed effortlessly by including the 1s its line vector, as opposed to examining the database. There is also no compelling reason to perform the prune venture in the applicant era capacity. The column vectors of length \( k - 1 \) thing sets are authorized to spare memory after they are utilized to produce the length \( k \) thing sets. Since generally (preprocessed) review information records are little enough, this execution functions admirably in our application area. Briefly, given an occasion database \( D \) where every exchange is connected with a timestamp, an interim \([t_1, t_2]\) is the arrangement of exchanges that begins from timestamp \( t_1 \) and closes at \( t_2 \). The width of the interim is characterized as \( t_2 - t_1 \). Given a thing set \( An \) in \( D \), an interim is a negligible event of \( An \) on the off chance that it contains \( An \) and none of its legitimate subintervals contains \( A \). Characterize \( \text{support}(x) \) as the degree between the quantity of least events that contain thing set \( X \) and the number of records in \( D \). A successive scene principle is:

\[
X, Y \rightarrow Z, [\text{confidence, support, window}]
\]

Here \( X, Y \) and \( Z \) are item sets in \( D \). \( \text{support}(X \cup Y \cup Z) \) is the support of the rule and the confidence is represented as
Here the width of each of the events must be short of what window. A serial scene standard has the extra requirement that $X$, $Y$ and $Z$ must happen in exchanges in halfway time request, i.e. $Z$ takes after $Y$ and $Y$ takes after $X$. As said in H. Manila and H. Toivonen (1996) the depiction here contrasts as given by Martin REH 'AK (2008), we don't consider a different window imperative on the LHS (left hand side) of the rule. The regular scene calculation discovers designs in a solitary arrangement of occasion stream information. The issue of discovering continuous consecutive examples that show up in various information arrangements was presented in R. Agrawal and R.Srikant (1995). This related calculation is not utilized as a part of our study following the incessant system or framework action examples must be found in the single review information stream from the system or the working framework. Here any subset of a successive thing set must likewise be an incessant thing set following each one interim that contains the thing set additionally contains every last bit of its subsets. We can accordingly additionally begin with discovering the successive scenes of length 2, then length 3, and so forth. As opposed to discovering relationships crosswise over traits, we are searching for connections crosswise over records.

**Extensions Added in Basic Algorithm(s)**

It is found that these essential calculations don't consider any area learning and as a result they can create numerous irrelevant tenets.
Step 1.
Scan database $D$ to form $L_1 = \{\text{frequent 1-itemsets}\}$;

Step 2.
Initialize $k = 2$; /* $k$ is the length of the item sets */
While $L_{k-1} \neq \emptyset$
Do /* association generation */
For each pair of $l_{k1} - 1, l_{k2} - 1 \in L_{k-1}$ and $l_{k1} - 1 = l_{k2} - 1$, where, their first $k - 2$ items are the same.
Do
Construct candidate item set $c_R$ such that it's first $k - 2$ items are the same as $l_{k1} - 1$, and the last two items are the last item of $l_{k1} - 1$ and the last item of $l_{k2} - 1$;

Step 3.
If there is a length $k - 1$ subset $s_{k-1} - 1 \subseteq c_R$ and $s_{k-1} - 1 \in L_{k-1}$ then
Remove $c_R$; /* the prune step */
Else
Add $c_R$ to $C_R$;
End
Scan $D$ and count the support of $\exists c_R \in C_R$;
$L_k = \{c_R | \text{support}(c_R) \geq \text{minimum support}\}$;
$k = k + 1$; $L_j$
End.

Step 4.
For all $l_k, k > 2$ do begin /* rule generation */
For $\forall a_m \subseteq l_k$
Do
\[ \text{conf} = \text{support}(l_k) / \text{support}(a_m); \]
5. If $\text{conf} \geq \text{minimum confidence}$ then
\{ Output rule $a_m \Rightarrow (l_k - a_m)$, With $\text{confidence} = \text{conf}$ and $\text{support} = \text{support}(l_k); \}
\}
\}

Figure 4.1 Apriori Association Rule Algorithm
R. Venkatesan et al. (2013) and Lee (1999)

R. Srikant, Q. Vu, and R. Agrawal (1997) introduced the rule templates to specify the allowable attribute to post-process the discovered rules, these values are used. Additionally, Padmanabhan and A. Tuzhilin (1998) used the boolean representations over the trait values as
thing demands amid principle revelation and a belief driven structure is utilized to find the unexpected' (i.e. uninteresting') examples. A downside of all these methodologies watched is that one needs to realize what governs or examples are intriguing or are now in the belief framework. Due to multidimensionality and large volume of network traffic data, here, on altogether audit data, we cannot adopt this kind of strong preceding knowledge. So identifying the interesting patterns require the interestingness correlation among the network traffic attributes to get interestingness measures. So, the following section describes about the extensions required while using this basic algorithm to detect intrusion through the attributes available in the network traffic data. These extensions can be provided by one of the following ways.

**Interestingness Measures Based on Attribute Relevancy**

The data available for the IDS experimentation and evaluation, is the network connection records captured within certain time frame window. It is clear that, this audit data can be normalized and can be represented in the tabular representation as like relational databases for further processing. The same theme is used while performing data preprocessing for the proposed research work and explained in more depth in chapter 5.

Generally, Relational database is generated using schema-level information and utilize the schema level data about review records to run the example mining procedure i.e., Though we can't know ahead of time what designs, which examples include genuine characteristic qualities, are intriguing, we regularly realize what properties are more critical or helpful given an information investigation errand.

Here, the studied basic algorithm help to identify the interestingness measures for this audit data by using the base backing and certainty qualities to yield just the statistically critical' designs. Since the fundamental calculations verifiably measure the interestingness (i.e. importance) of examples by their backing and certainty values, without any accessible former space learning. So, these interestingness measures can be derived from this main property of basic algorithm. So assume IM is the interestingness measure and it can be represented as:

$$IM(p) = RF(support(p), confidence(p))$$

Where, $RF$ is used to denote ranking function.
Here, the main aim is to include schema level data into the like-wise actions to get certain intrusion detection patterns from the audit data as specified in Hua Bing Yang (2003). Assume $IMA$ is an extended interestingness measure on pattern $p$ contains the specified interesting attributes, IMA is measured on

$$IMA(p) = RF \left( RF \left( IMA(p), \text{support}(p), \text{confidence}(p) \right) \right)$$

$$= RF \left( IMA(p), IM(p) \right)$$

Where, $RF$ is a positioning capacity that first considers the traits in the example, then the backing and certainty values. In the accompanying segment, we speak to a few schema level qualities of review information; in the types of what characteristics must be considered that can be utilized to distinguish the mining of important characteristics.

We don't utilize these IMA measures as a part of post preparing to channel out superfluous governs by rank requesting. Rather than it, for productivity, we utilize them as thing demands, i.e. conditions, amid competitor thing set era and recognizing intrusion detection attack types. The actual use of this measure is done during the evaluation of KDD Cup 1999 Dataset analysis details are explained in the Chapter 6.

Additionally, like this relevancy measurement among the attributes of network audit data, even one can easily find out interestingness measures and their respective patterns from audit data reference line attributes. The details of these interestingness measure pattern generation from referenced line attributes is illustrated in next section.
### Table 4.1 Network Connection Records Lee (1999)

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>service</th>
<th>src_host</th>
<th>dst_host</th>
<th>src_bytes</th>
<th>dst_bytes</th>
<th>flag</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>telnet</td>
<td>A</td>
<td>B</td>
<td>100</td>
<td>2000</td>
<td>SF</td>
<td>...</td>
</tr>
<tr>
<td>2.0</td>
<td>ftp</td>
<td>C</td>
<td>B</td>
<td>200</td>
<td>300</td>
<td>SF</td>
<td>...</td>
</tr>
<tr>
<td>2.3</td>
<td>Smtp</td>
<td>B</td>
<td>D</td>
<td>250</td>
<td>300</td>
<td>SF</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

### Interestingness Measures Based on Reference Line Attributes

In audit data, it is found that some attributes available in data are important while describing the data, while others attributes provides auxiliary information only. Consider the audit data of network connections as shown in Table 4.1.

In Table 4.1, network linking attribute values describe each record. In these records, it contains the continuous attribute values other than the timestamps, which are discrete in nature. Very often, the network connection is easily distinguished by the following rule from each other as given below:

\[
<\text{timestamp},\text{src_host},\text{src_port},\text{dst_host},\text{service}> 
\]

These are the important and useful attributes contains combination of its initial time, source host, source port, target host, and service presented at endpoint port. Also, seems useful attributes to visualize the network traffic data. During this data analysis, it is observed that relevant rules generated from this network traffic data should represent patterns related to useful attributes only and meanwhile, unwanted attributes lead towards the irrelevant patterns. With the following example, it is clarified properly. Suppose the basic association rules algorithm generates rules as:

\[
\text{src_bytes} \rightarrow 300 \rightarrow \text{flag} \rightarrow \text{SF}, [0.6, 0.2]. 
\]

It is found that these guidelines are not valuable and are deluding at certain degree. There is no any importance for the relationship between the quantities of bytes from the source, src_bytes, and the ordinary status (Flag = SF) of the association. From this, we can conclude that useful attribute(s) are known as reference line attribute(s) when they are utilized as a manifestation of paradigm thing in the affiliation guidelines calculation. Amid
competitor set era, a thing set must contain value(s) of this reference line attribute(s) and we can presume that relationships among non-axis attributes are mostly irrelevant and not important to form association.

So, mathematically, we can consider as

\[ \text{IMA}(P) = \begin{cases} 
1, & \text{if } P \text{ contains reference line attribute(s)} \\
0, & \text{Otherwise} 
\end{cases} \]

Normally, not all useful attributes can be used as reference line attributes at a time and even it is not required in practice. Since during the network analysis, sometime analyst may focused on network services and at the same time other analysts may look for network payload attributes and so on. So, by using service attributes as reference line attributes, we can form the association rules related to service pattern and others can use payload attributes to form their patterns based on these attributes association rules. Additionally, the main benefit with reference line attribute is that item generation for frequent episodes can be constrained and only useful frequent item set can be generated. It generally reduces the computational time and speed up the system. The studied basic algorithm generates so many frequent scene guidelines which contain just the unimportant' and irrelevant' trait values. It is illustrated with following example,

\[ \text{src_bytes} = 400, \text{src_bytes} = 400 \rightarrow \text{dst_bytes} = 600, \text{src_bytes} = 400, [0.5, 0.3, 2s] \]

It is observed that in these rules, each attribute value is different, for example, \( \text{src_bytes} = 400 \), is different for each connection record.

The situation can be more panic when the support of an affiliation control on Non-reference line traits, is high then there will be colossal sum of meaningless' serial scene principles of the structure will be produced like \( (P|Q)(P|Q) \to (P|Q)(P|Q) \to \) due to the accompanying hypothesis.

**Theorem 1**

Let \( S \) is support of the association rule \( P \to Q \), and let \( N \) is the total sum of episode rules on \( (P|Q) \), therefore, rule will be \( (P|Q)(P|Q) \to (P|Q)(P|Q) \to \) and \( N \) converts then at least an exponential factor of \( S \).

**Proof**
We are aware about, that, any subset of an incessant thing set is likewise visit according to the continuous episodes algorithm, for example, if $P, P \rightarrow P, Q$ is a frequent episode rule, then $P \rightarrow P, \ an \ P \rightarrow Q, \ etc.,$ number of item set are increased and generated from its subsets at each iteration of pattern generation. The numbers of iterations which are generating the frequent item sets are usually bounded with the maximum length of an item set, $L$ (which is always large in number), instead of attributes associated in association rule.

Meanwhile, at iteration, we can take $\mathcal{E}_n$ as count, the number of episode patterns on $(P|Q)$, where $n$ is length of generated item set in the current emphasis. Subsequently, the aggregate number of scenes produced on $(P|Q)$ is

$$N = \sum_{n=1}^{L} N_n$$

Now consider the database of size ‘$M$’ number of records and ‘$T$’ seconds is the time difference between the first and last record. Then $(S \ast M)$ are the records which contain $P \cup Q$ as the same transaction. Then the quantity of insignificant and non-overlapping interims that have $n$ records with this $PUQ$ is $\frac{S \ast M}{n}$. It is watched that each of these interims in this manner contains $2^n$ length $n$ scenes on $(P|Q)$. Then $N_n = \frac{S \ast M}{n} \ast 2^n$ and therefore, the total number of episodes generated on $(P|Q)$ are

$$N = \sum_{n=2}^{L} S \ast M \ast 2^n$$

Therefore, $N$ can be considered as exponential factor of $S$ at least level. With this proof, we can defined another theorem as

**Theorem 2**

The value of $L$ i.e. maximum length of item set is monotonically increased with support value $S$ but it is restricted by database size $M$.

**Proof**

Firstly, it is assumed that there is even distribution of database records based on time and hence, do the connections with $PUQ$. So, now we can compute the width of interval, at iteration as $width = \frac{mT}{E \cdot M}$ as per given width requirement $W$, $width \leq W$ must have, then we have the maximum value of $n$, is $n \leq \frac{W \cdot S \cdot M}{T}$ and $L$ can be defined as
\[ L = \min \left\{ m, \frac{W \ast S \ast M}{T} \right\} \]

W Lee (2005) concluded it is not difficult to see that if the records are not equally disseminated, then \( W \) is \( \frac{NT}{S \cdot M} \) a variable and subsequently value of \( L \) is monotonically increased in this case too. To reduce the irrelevant and unwanted frequent episode rules, it is necessary that we should extend the basic frequent episodes algorithm to compute frequent sequential patterns as like the following two phases.

**Extension Phase 1**

Find the frequent association rules using the reference line attribute(s);

It means that as per the requirement of generation of frequent episode rules, we must be able to select certain reference line attributes and from these selected reference line attributes, generate the frequent association rules. The main advantage of such flexible selection of reference line attribute is to cut down the computational time requirement as well as to avoid the useless and unwanted reference line attributes association rule generation.

**Extension Phase 2**

Generate the frequent sequential patterns from these association rules.

It means that for the second augmentation, the things from which scene thing sets are produced are the relationship about the reference line attribute(s), and the reference line property estimations (i.e. unit length affiliations just). Consider the illustration of such produced affiliation guideline:

\[(service = smtp, src_bytes = 400, dst_bytes = 500, flag = SF),
(service = telnet, flag = SF) \rightarrow (service = http, src_bytes = 400), [0.3, 0.2, 2s]\]

Note that everything set is a relationship of the scene principle:

\[(service = smtp, src_bytes = 400, dst_bytes = 500, flag = SF).\]

Here, it can possible to combine the associations among traits and the consecutive examples produced utilizing the affiliation guideline mining calculation among the records of database into a solitary guideline. The profit of such manage era not just wipes out unessential and futile examples, additionally gives rich and valuable data about system information for the interruption discovery.

**Utilization of Next Relevant Attributes by Reference**
The information considered for this examination work assessment likewise hold an alternate intriguing trademark is that a few qualities can be the references of other important characteristics. These reference qualities ordinarily convey data about some _target traits, and different characteristics depict the execution' ascribes that alludes to the same _target. Consider the web log records, as demonstrated in Table 4.2.

**Table 4.2 Web Log Records (Lee (1999))**

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Remote Host</th>
<th>Action</th>
<th>Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>his.moc.kw</td>
<td>GET</td>
<td>/image</td>
</tr>
<tr>
<td>1.1</td>
<td>his.moc.kw</td>
<td>GET</td>
<td>/images</td>
</tr>
<tr>
<td>1.3</td>
<td>his.moc.kw</td>
<td>GET</td>
<td>/shuttle/missions/sts-71</td>
</tr>
<tr>
<td>3.1</td>
<td>taka10.taka.is.uec.ac.jp</td>
<td>GET</td>
<td>/images</td>
</tr>
<tr>
<td>3.2</td>
<td>taka10.taka.is.uec.ac.jp</td>
<td>GET</td>
<td>/images</td>
</tr>
<tr>
<td>3.5</td>
<td>taka10.taka.is.uec.ac.jp</td>
<td>GET</td>
<td>/shuttle/missions/sts-71</td>
</tr>
</tbody>
</table>

Martin REH ´AK (2008) described that; here activity and appeal are the execution' done by the target'. We can watch that for various focuses on, each of them makes the same arrangement of executions like —/images||, —/images and —//shuttle/missions/sts71. It is vital to utilize the target' as a kind of perspective at whatever point one needs to discover execution' designs since the _executions' from diverse _targets' are typically not identified with one another. In this way, this sort of consecutive examples can be composed and spoke to as:

\[(Target = X, Execution = a), (Target = X, Execution = b) \rightarrow (Target = X, Execution = c), [confidence, support, window]\]

It is clear that, within every event of the example, the execution qualities allude to the same target and the real target quality may not be given in the standard here on the grounds that any specific target worth may not be visit regarding the records of entirety dataset. Along these lines, it implies that, target is essentially a reference or a variable. In the same sense, the essential successive scenes calculation delineated prior can be enhanced and advanced to consider the use of next relevant attributes by reference. More specifically, when episode
rules are formed, an additional condition can be put as given in US 8805839 B2 (2014),
within its insignificant events in affiliation, the records secured by its important and relating
thing sets have the same quality. So interestingness measure \( IM_a(p) \) can be derived as
given below.

\[
IM_a(p) = \begin{cases} 
1, & \text{if the item set of } p \text{ indicates same next reference attribute value} \\
0, & \text{Otherwise}
\end{cases}
\]

One major thing observed during the analysis of audit data, along with potentially
frequent episode, it is additionally important to incorporate the low recurrence designs in the
system during its evaluation. The main reason behind this is in the audit data itself. In daily
network traffic, it is found that some services like gopher and account are occurred at very
least time. Still we require including such low occurrence patterns into the system activity
profile with the goal that we can speak to examples for each one underpinned administration
and their attributes in the collected network traffic data.

**Use of Level wise Mining Algorithm**

This consideration of very less occurrence of pattern may lead to generation of unnecessary
and unwanted huge number of patterns related to frequently occurred services. This can be
illustrated with the next example. On internet, user frequently sends emails with the help of
e-mail web services and in this case, the emails are getting transferred via SMTP protocol at
each time, means they frequently uses this SMTP protocol and it has high occurrence in this
scenario.

Consider the following example, where the frequently used service is SMTP.

\[(service = smtp,sr c\_bytes = 400), (service = smtp,sr c\_bytes = 400)\]

\[\rightarrow (service = smtp, dst\_bytes = 500), [0.4, 0.4, 3a].\]

In such cases, we can utilize the idea of the Martin REH ´AK(2008), level savvy
rough mining method portrayed in Figure 4.2 for finding frequent sequential patterns from
given data input by identifying the patterns related to at least one frequently used attribute in
the given data and then reduce the support threshold value at each next step so that along with
high frequency attributes, we are able to get low frequency axis values by avoiding the
already identified and output frequent attributes in the previous iteration. So, old frequent
values will not participate in next iteration to find low frequency patterns.
Generally, rule is created; it must contain no less than one new (low recurrence) reference line characteristic worth. As portrayed in prior case, in the second emphasis, where SMTP now is old reference line property estimation, so we get pattern as

\[(\text{service} = \text{smtp}, \text{src_bytes} = 400), (\text{service} = \text{http}, \text{src_bytes} = 400)\]
\[\Rightarrow (\text{service} = \text{smtp}, \text{src_bytes} = 500), [0.3, 0.5, 32].\]

This procedure ends when a low help quality is arrived at. By and by, this can be the most minimal recurrence of all reference line property estimations. It is watched that for a high recurrence reference line property estimation, we tosses its low recurrence examples (produced amid this execution with low help values) since they are discovered superfluous and not intriguing in our case.

So, the Interestingness Measure on pattern \(p\) \(i.e.\ (IM_\alpha (p))\) based on reference line attributes can be computed as given in the following equation at the time of iteration.

\[
IM_\alpha (p) = \begin{cases} 
1, & \text{if } p \text{ contains at least one new reference line attribute value} \\
0, & \text{Otherwise}
\end{cases}
\]

In this case, we are dealing with multiple levels of a single concept, we are not considering multiple levels of concepts for this approximate leveling and hence it is different approach than traditional one.
**Input:**
Let, $d$ as database,
$S_f$ is the terminating minimum support value,
$S_i$ is the initial minimum support, and the list of reference line attribute(s).

**Output:** Frequent patterns identified
1. Start
2. Let initially, $\text{Limited}_{\text{attribute}} = \emptyset$
3. Start Scanning of database $d$ to get $L = \{\text{frequent 1-itemsets } \exists S_i\}$;
4. Consider $s = S_i$;
5. do {
6. Compute frequent pattern from $L$ where each pattern must contain at least one reference line attribute value that is not in $\text{Limited}_{\text{attribute}}$.
7. Add this new reference line attribute values to $\text{Limited}_{\text{attribute}}$.
8. Add this new frequent pattern to the output rule set as Rules.
9. Reduce the value of $s$ to its $s/2$ reduction which will be smaller support value for the next iteration.
10. } while ($s \geq S_i$).

**Figure 4.2 Level wise Approximate Mining of Frequent Pattern Lee (1999)**

Relative Support Mining Algorithm

In comparison with previously described approaches and algorithms, even more useful and flexible approach can be used to deal with attribute values with skewed data distribution to get relevant patterns is the use of relative support mining algorithm. In this case, instead of focusing on size of the network database, D.M. Farid et al. (2010) identified; number of events of remarkable trait esteem in the database can be considered and used as the reference while calculating the support value of certain pattern. The main theme is if a relative support value $S_i$ is considered for attribute $A_l$, and it has unique value as $A_l = V_{ij}$, has a total of $n_{ij}$ occurrences in the database, then a pattern can be said as frequent based on $A_l = V_{ij}$ frequent if it has at least $S_i \times n_{ij}$ attribute occurrences. Different values of relative support can be defined and utilized for different attributes. So, relative support mining algorithm is specified in the following Figure 4.3.
**Input:** Let database $D$, the overall minimum support $s$, and the relative support $S_i$ for each attribute $A_i$

**Output:** frequent patterns or association rule

Start

1. Scan database $D$ to get $L = \{\text{unique attribute values}\}$, the count $C$ of each attribute value is obtained and, the number of records in $D$ as $DB\_Size$, is also recorded.

2. For ($v_{ij} \in L$) {

3. If the value of $S_i$ is given then {compute $v_{ij}\_rel\_support = S_i \times v_{ij}\_C / DB\_Size$}

4. Else {if $S_i < v_{ij}\_C < DB\_Size$ then {remove $v_{ij}$ from $L$}

5. Else { Compute $v_{ij}\_rel\_support = s \times DB\_Size$}

6. } /* $L$ is becoming the set of frequent 1 item sets; */

7. Start to Compute frequent patterns from $L$ by using step 8 to step 10

8. When a candidate item set $l_R$ is constructed from $l_{R_1} = 1$ and $l_{R_2} = 1$ then go for step 9

9. Calculate $l_{R}\_rel\_support = \min(l_{R_1}\_rel\_support, l_{R_2}\_rel\_support)$, and

10. $l_R$ is frequent if $l_R\_C > l_R\_rel\_support$.

Stop

---

**Figure 4.3 Relative Support Mining Algorithms for Frequent Patterns**

The detailed illustration of this algorithm is described here for the understanding of this extension. The general steps of this algorithm are:

**Step 1:** Find all remarkable attribute values in given database.

**Step 2** Set the $rel\_support$ requirements for frequent patterns contain these attribute values, which specifies the ‘overall’ support.

**Step 3** Modify the candidate generation process for frequent pattern generation, when such candidate item set contain values from different that attributes than the $rel\_support$ requirements. The smallest value will be considered for this frequent pattern generation.

Based on this the Interestingness Measure on pattern $p$ i.e. $(IM_A(p))$ can be modified as shown in the following equation.

$$IM_A(p) = \begin{cases} 1, & \text{if the count of } p > \text{smallest } rel\_support \text{ of its attribute values} \\ 0, & \text{otherwise} \end{cases}$$

This task is vital because when we required computing patterns related to ‘rare’ events, which are based on skewed value distribution of the multiple attributes. The Next example simplifies this concept. It is found that, in network data traffic, most connection records raised the value of attribute $\text{flag} = SF$ which means that normal connection
establishment and termination is done according to the network protocols. Sometime we may require finding the cases when the flag value is not $SF$, it may be $REJ$ which is usually used to show connection rejected.

We may have the different relative support requirements for attributes like $flag$, $service$, and $dst\_bytes\ host$ get the patterns of the anomalous traffic. Consider if $flag = REJ$ may appear in only 1% of the connections, then the overall support $s = 0.05$ will filter out such patterns having the $flag = REJ$. On other hand, using relative support for $flag$ may produce $flag = REJ$ patterns because they are measured relatively based on to the number of occurrences of $flag = REJ$. 
Findings of Extension added in Basic Mining Algorithm(s)

Here, initially, K.W. Mok (1999) found that in request to distinguish and build the gimmicks to build effective classification models for intrusion detection, we require mining and analyzing the frequent patterns based on association tenets and successive examples, from review information.

The focus of our research efforts here is on the issue of how to compute the useful and associated frequent patterns effectively from this huge audit data collection and from multiple attributes of the audit data. Here, the solution applied is to that incorporates data on the qualities, rather than simply help and certainty values in the finding of regular and productive examples for the recognition. These measures are fused in our augmented calculations as different types of thing imperatives in the mining methodology as various forms of item constraints in the mining process. These expansions are shown top to bottom for the reasonable understanding of their functionalities, specifically, utilizing reference line attributes to register pertinent affiliations, utilizing next of reference credits to process legitimate frequent patterns, Lee’s level wise approximate mining is used to include low frequency but important patterns in the detection, and relative frequency is used. The top frequent patterns which are relative to each unique value of the specified attributes of the audit data are computed.

4.3.2.3 Generation of Mathematical Model based on Set Theory

This section describes the details about the mathematical modeling based on set theory used for the system computation. In computer system computation, the problem definitions are mainly categorized into the following types of the problems i.e. P-class, NP-complete and NP-hard problems. Thus, the designed and proposed system is on the basis of producing the attack classification either known or unknown attacks on the provided input dataset of captured network traffic as online or offline contents. It clearly states that proposed system is result oriented and based on these result, further, network administrator’s prevention strategies as well as attack analysis task is carried out; so it define the selected problem definition leads towards P-class type problem; where P-class problem is polynomial, time-stipulated and depends on fixed algorithms execution and solution oriented as compared to other types of problems. The mathematical modeling of proposed system includes representation of system as input, processes, rules and output sets of the system. The details
about each set are illustrated in the following section, as well as the relationship between all of these set is mapped using Venn diagram as shown in Figure 4.5.

In set theory representation, Let $S$ represents the system, $I$ represent the input set of the system, $P$ is the processes set of the system, $R$ is the rule set applied on the system during its processing and $O$ is the output set of expected outcomes of the system. Therefore, the system representation in set form is

$$S \in \{ I, P, R, O \}$$

**Input**

Initially, the implemented system model does the intrusion detection by analyzing the given attack input dataset. As described earlier, data input for the proposed system can be online captured traffic packet data. For experimental analysis, standard intrusion attack dataset is obtained online which are collections of captured network traffic mixed with TCP dump traced file. The input for this proposed system is considered as online as well as offline network traffic captured data. The online network traffic data is captured through wire shark protocol analyzing tool which was further preprocessed and then utilized for intrusion detection. The offline data is obtained from KDD cup 1999 intrusion attack dataset from the MIT/LL which one also preprocessed further for the system evaluation. Therefore, the input set $I$ represented as

$$I = \{ I_1, I_2 \}$$

Where,

$I_1$ is online captured input network traffic data,

$I_2$ is KDD cup 1999 standard intrusion dataset.

<table>
<thead>
<tr>
<th>duration, protocol_type, service, flag, src_bytes, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, logged_in, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files, num_outbound_cmds, is_host_login, ..</th>
<th>dst_host_srvr_error_rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>0, tcp, ftp_data, SF, 491, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0</td>
<td>..........0</td>
</tr>
<tr>
<td>0, udp, other, SF, 146, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0</td>
<td>..........0</td>
</tr>
<tr>
<td>0, tcp, private, S0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0</td>
<td>..........0</td>
</tr>
<tr>
<td>0, tcp, http, SF, 232.8153, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0</td>
<td>..........0</td>
</tr>
<tr>
<td>0, tcp, http, SF, 199.420, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0</td>
<td>..........0</td>
</tr>
</tbody>
</table>

Figure 4.4 KDD Cup 1999 Test Dataset Sample (Ntw_doc.html (2009)),

NSL –KDD Data Set (2010)

As introduced earlier in chapter 3, in KDD 1999 dataset, it contains various classes of intrusion attacks like Denial of Service (DoS), User to Remote (U2R), Probe, R2L (Remote
to Local) and their classes of attacks like for DOS attack, it has Apache, land, mail-bomb, etc., for R2L attack, it contains Send-mail, Xclock, etc., then for U2R attack, subtype of attack like Buffer Overflow, and for Probe, it consists of Exploitation as subtypes of attack as shown in Figure 4.4 as sample of original KDD Cup 1999 dataset represented with its 41 attributes and their values of network traffic packet data before the preprocessing.

**Processes**

The processes are the main elements of system computation. These are the functions of the system to represent the complete flow of execution from initial process to final step of execution. Each process does certain task as part of system computation.

For this proposed system, the main processes are described below.

- **P1** is Data preprocessing process used to make data compatible for further processing,
- **P2** is known attack detection process based on misuse detection
- **P3** is Unknown attack detection process based on anomaly detection
- **P4** is Data correlation process used to perform known and unknown attack correlation
- **P5** is Attack alert generation process for known as well as unknown attacks
- **P6** is Prevention policies application process to provide layer of protection and
- **P7** is the management console process used to administer the whole system.

Therefore, the set P is represented as

\[
P = \{P1, P2, P3, P4, P5, P6, P7\}
\]

**Output**

Output of the proposed system are the final outcomes of the system i.e. type of known and unknown attacks detection i.e. detection of classes of attacks which are mentioned earlier in this section, applied prevention policies. Generally, this set represents only the final expected outcomes and does not provide the intermediate outcome details. The output set O can be written as

\[
O = \{O1, O2, O3, O4, O5, O6, O7\}
\]

Where,
- **O1** represents DOS attack and its subtype’s detection
- **O2** represents R2L attack and its subtype’s detection
- **O3** represents U2R attack and its subtype’s detection
- **O4** represents Probe attack and its subtype’s detection
- **O5** represents unknown or anomalous attack detection
- **O6** represents generated alerts once known or unknown attack is detected
$O7$ represents applied prevention policies.

Based on the above set explanation for the proposed system, the functional dependencies of the system can be traced as shown in following Table 4.3. The functional dependencies provide the help to identify the mapping of elements of system set $S$. Basically; it depicts the functional dependencies of Processes $P$ of the system during their execution where value ‘1’ in the table 4.3 represents the presence of functional dependency whereas value ‘0’ indicates process independency.

Table 4.3 Functional Dependency of $S$ based on Processes $P$

<table>
<thead>
<tr>
<th></th>
<th>$f_0$</th>
<th>$f_1$</th>
<th>$f_2$</th>
<th>$f_3$</th>
<th>$f_4$</th>
<th>$f_5$</th>
<th>$f_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_0$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$f_1$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$f_2$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$f_3$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$f_4$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$f_5$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$f_6$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

The mapping of proposed system input, process and output set can be represented with the help of Venn diagram as shown in Figure 4.5.

![Figure 4.5 Venn diagram for the System S with \{I, P, O\} Mapping](image-url)
More precisely, the mapping shown in Venn diagram gives another more useful and vital representation of the system for the development in the form of process state diagram, where the computation and representation of intermediate and final outcome can be possible. Even, it makes the model of the system more expressive and explanatory. For the proposed system, to bring out completeness in the model presentation, the process state diagram is shown in Figure 4.6.

![Figure 4.6 Process State Diagram](image)

**4.4 Modeling of Proposed Research Work as System**

In this section, we first present the information digging idea utilized for cutting edge inconsistency discovery. At that point we portray the AIDS structural planning, the AIDS plan, and the association gimmicks utilized as a part of AIDS and mechanized mark era and then prevention phase is discussed to avoid further similar attacks in future.

**4.4.1 Network Anomaly Detection based on Data Mining**

It is identified that open networks are always threaten. In Transmission Control Protocol (TCP), User Datagram Protocol (UDP), or from both system intrusions and anomalies. So, the major components of our network anomaly detection process are shown in Figure 4.7, the
normal profile database generation from trained data is done along with the construction of
the oddity identification motor. The location motor is equipped for distinguishing odd activity
that is created by movement irregularities.

The information records are separated from inspected Internet movement information. After this, the rule mining engine is defined where it contributes for the two phases of
development. First phase, \textit{i.e.} to generate the normal traffic database \textit{training phase} is
required. In the second phase, attacks may appear \textit{i.e.} detection phase in captured real time
traffic data.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{data_mining_strategy}
\caption{Data Mining Strategy for Anomaly Detection in Network Data Kai Hwang \textit{et al.} (2007)}
\end{figure}

System abnormality is discovered once then standard depicting the genuine activity
associations can't discover any match with typical database prepared information in the
database. Along these lines, for the network anomaly detection, we have been used normal
profile use detection model.

\subsection*{4.4.2 Proposed System}

No single strategy or engineering is the — magic bullet to ensure assurance against present or
future machine assaults.

In order to robustly protect enterprise and government networks against the complete
spectrum of threats and vulnerabilities, IDS must accomplish more than discover assaults *i.e.* it ought to empower exact identification to keep assaults from arriving at and harming discriminating system assets and information of the association. Without extensive variety of assault discovery and the execution to precisely avoid assaults, numerous IDS are much the same as an advanced Maginot Line. So this research work is intended to enhance the effectiveness of intrusion detection system with prevention policies provision based on Data mining techniques and performance improvement scope with the use of sentiment analysis in accurate attack identification.

This work introduces Advanced intrusion Detection System (AIDS) construction modeling and demonstrates its viability through recreation tests followed by prevention phase of the system by giving different prevention options to the network administrator over an internet.

The research design architecture can be divided into three phases of development namely,

1. Use of Data Mining to perform intrusion detection
2. Known and unknown attack detection
3. Prevention

Before proceeding for the main workflow of the proposed research system, the primary requirement of this system is input data availability and it’s preprocessing as shown in Figure 4.8. So, details about Data preparation and its processing are given in next session and then main architecture details are illustrated further.

**4.4.2.1 Data Preparation and Preprocessing**

The data obtained from the standard dataset is preprocessed to get the data in the format which is acceptable for the detection. The outcome of this step will give us formatted dataset which will be the input for detection phase of the system.
Network intrusion detection system is mainly targeted at network data and the quality of data extraction determines system work efficiency. Under normal circumstances, large amounts of data flow through the network, and capturing long data segment will put pressure on storage space. The network data captured by system are binary data, and cannot be directly used, it is necessary to extract the information we are interested in and process large numbers of data packets into TCP connection records which reflect session information. Then the system will complete the works such as data filtering, information statistic based on the window and format conversion, ultimately, generate the property list which can reflect the network data. The following section gives brief about the data collected for system evaluation and its preprocessing.

We have been used KDD Cup 1999 IDS evaluation data set as input for proposed system which is Massachusetts Institute of Technology/Lincoln Laboratory traffic files collection and is available on web portal of kdd.ics.uci.edu.

It is found that Frequent Episode Rule (FER) based approach is having constraint on attack detection. Since, numerous assaults are produced by a solitary association and May not ready to produce any irregular FERs.

With a specific end goal to tackle this issue, we, we have been considered from ICSTC (2008) as rare qualities of single associations. Case in point, associations with the same source and same terminus locations are frequently assaults. An alternate issue is that a solitary assault may keep going for a drawn out stretch of time on the machine. In this way,
we can use association succession numbers, rather than time stamps, to gather associations heading to the same objective and to detect such attacks. The data collected from the MIT/LL web portal, it is available in the following original form in before preprocessing for few captured network traffic packets as shown in Figure 4.9.

Figure 4.9 KDD Cup 1999 Dataset in its Original Captured Form before Preprocessing (Ntw_doc.html (2009))

The original KDD Cup 1999 dataset contains 41 features as shown in Figure 4.9 as. Each row contribute for these 41 features of the network traffic packet and the each packet feature attribute value is separated with comma (,) operator for the understanding. The detail of each feature attribute and the type of value hold by each attribute is given in the following Table 4.4.

In the collected data set of KDD Cup 1999, it is found that it contains about five million connection records which are trained. In this case, connection is defined as sequence of TCP packets collected for specific duration of time. Abhi R. Varma et.al (2012) clarified as with 41 continuous and nominal features, each record is identified unique and distinct in the dataset and nominal or symbolic feature attributes. In the dataset, total nominal or symbolic feature attributes values are 07 like, Protocol_type, Flag, Land, etc. and total continuous feature attributes are 34 like, Duration, Src_Bytes, Dst_bytes, etc.
The symbolic or nominal attribute represents the string values of the object whereas continuous values provide the numerical values of the object. Therefore, as shown in Figure 4.9, it contains symbolic as well as numerical values of the feature attributes of captured packet data. Additionally, these feature attributes are categorized into further four classification based on the features are extracted from the parts of packet i.e. Packet Header, Packet Payload Contents and Packet Tail. Each classification along with feature attribute classification is shown in Table 4.5.

On the source portal of KDD Cup 1999 dataset which is stated earlier, the testing data collection files, the feature attribute definition file, and the training dataset collection file are
given separately, So to work with this KDD Cup dataset, one needs to merge together first the actual dataset collection file contents as shown in Figure 4.9 and the feature attribute definition file properly to get exact values of each attribute value for further processing.

Table 4.5 Feature Attribute Classification

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>Classification Type</th>
<th>Contributing Feature Attributes for the Classification (As given in Table 4.3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Individual TCP Connection Features</td>
<td>Feature Attribute No. 1-9</td>
</tr>
<tr>
<td>2</td>
<td>Content Features</td>
<td>Feature Attribute No. 10-21</td>
</tr>
<tr>
<td>3</td>
<td>Traffic Features Computed Using 2s Time Window</td>
<td>Feature Attribute No. 22-31</td>
</tr>
<tr>
<td>4</td>
<td>Traffic Features Computed Using 2s Time Window from Destination to host</td>
<td>Feature Attribute No. 32-41</td>
</tr>
</tbody>
</table>

This merging requires the data preprocessing of KDD Cup 1999 Dataset Collection to get data in compatible form. The file after the merging of uncompressed, unprocessed KDD Cup 1999 Dataset file and its feature attribute definition file as specified by Zoubir Mammeri, Pascal Lorenz (2004) and Mahbod Tavallaee (2009), the contents of merged file are shown in Figure 4.10.

<table>
<thead>
<tr>
<th>duration, protocol_type, service, flag, src_bytes, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, logged_in, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_access_files, num_outbound_cmds, is_host_login, is_guest_login, count, srv_count, serror_rate, srv_serror_rate, rerror_rate, srv_rerror_rate, same_srv_rate, diff_srv_rate, srv_diff_host_rate, dst_host_count, dst_host_srv_count, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>0, tcp, smtp, SF, 829, 327, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 8, 113, 0, 88, 0, 25, 0, 12, 0, 2, 0, 0, 0, 0, 0</td>
</tr>
<tr>
<td>0, udp, private, SF, 105, 146, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 255, 253, 0, 99, 0, 0, 0, 0, 0, 0, 0, 0</td>
</tr>
<tr>
<td>0, udp, private, SF, 105, 146, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 254, 1, 0, 0, 0, 0, 0, 0, 0, 0</td>
</tr>
</tbody>
</table>

Figure 4.10 Merged KDD Cup 1999 Original Dataset File with Feature Definition File

Once this KDD Cup 1999 dataset with feature definition file is obtained, it is possible to process this complete file for the further modules of the proposed system. This file is processed with the initial module simulation through program and its Comma Separated Value (.CSV) file is obtained which is opened in Microsoft Excel Sheet as shown in Figure 4.11. It gives more clearly the details about each feature attribute of packet data for the KDD Cup 1999 dataset. In the research work, the further data preprocessing is illustrated in the next section.
4.4.2.2 Intrusion Detection Phase

In the intrusion detection phase as per details described by D.M. Farid et al. (2010), the integration of signature based IDS and anomaly based IDS is done which is very rarely found in the literature review study and Abdulla Amin Aburomman, Mamun Bin Ibne Reaz (2013) observed that existing IDSs are designed and implemented from misuse based or deviation based intrusion detection methodology called as signature or anomaly based detections respectively with specific attack detection capabilities. However, their performances found good with certain attack detection spectrum but failed with other attack type detection. Even, they are generally developed to do detection rather than prevention at the same time. The attacker are changing their ways to havoc the computer system and networks of the organization with different attacks, so relying on certain attack detection IDSs is risky to protect and hence need the solution by which maximum coverage of attack types is possible.

Additionally, the IPS is developed to perform attack prevention once the intrusions are reported by any deployed IDS in the organization information infrastructure. But it is surprised, many business organizations pays less attention to install and configure separate IPS once IDS is installed along with firewall as another layer of protection. This is time consuming and may damage all the vital resources of the organization though intrusion detection and prevention is employed. Hence quick response and action for detected intrusion is expected by the today’s organization with cost effective solution. Again, the signature or misuse based intrusion detection techniques has limited performance since the signature required to detect intrusion must be properly defined and analyzed by the human experts. On other hand, anomaly based intrusion detection techniques may increase the false positives during the intrusion detection.
The proposed system is aimed to integrate the signature-based IDS with the anomaly-based IDS with improvement in intrusion detection rate accurately with minimum false positive rate. Also, prevention policy is developed along with intrusion detection integration to provide quick action on the detected intrusion with the management console provision. The details of this integrated intrusion detection are further mentioned in the following section.

**Signature and Anomaly Intrusion Detection**

In the proposed system architecture, the intrusion detection phase is further divided into three steps as known attacks identification is performed by signature intrusion detection and unknown attacks are found by anomaly-based intrusion detection; and new signature generation from the detected unknown attack detection to reduce further system computation for the similar intrusion detection. The details of each step are given below.

**Signature Based Intrusion Detection**

After the data preprocessing, in this first step, the aim is to screen ready the recognized outbreak traffic by Snort similar signature detection based on signature matching with the existing or provided signature database prepared by human experts at initial stage. The input traffic data is considered for the proposed system is KDD Cup 1999 Dataset and or the captured network traffic data for the local network infrastructure with compatible form. The processing of this signature-based intrusion detection consists of signature matching algorithm, signature database retrieval and response for the detected attack and forwarding of remaining traffic to next step of the system. Finally, the outcome of this step is detected known attack.

**Anomaly Based Intrusion Detection**

Once the known attacks are detected from the provided input data which is preprocessed; the residual traffic covering unidentified or eruption outbreaks is advanced to the outline configuration withdrawal engine to produce recurrent outline rules with supported diverse support threshold levels. The provision is given to provide threshold level as user defined to detect unknown attacks. The discovery of uncommon infrequent attacks permits such smoothing endowment which will be later declared as anomalies or unknown intrusions in the given input data. The frequent patterns are then compared with precompiled successive examples from ordinary activity database. The profile designs that don't match the ordinary
profiles or match them with strangely high recurrence are marked as irregular and obscure interruptions. The located obscure interruptions are utilized to produce new assault marks focused around caught odd conduct utilizing a weighted continuous thing set mining plan calculation. These recently produced marks are then added to the signature database for future discovery of comparable assaults.

Generally, this step detects unknown, burst, or multi-connection attacks from the given network traffic data. So, the input for this step contains remaining traffic fed by signature based intrusion detection step. The processing is based on deviation in the input from its normal pattern profiles, normal profile database prepared based on data mining scheme illustrated in section 4.4.1 and the forwarding of detected unknown attack profile pattern to signature generation module for the further same or similar intrusion detection as known attack. Lastly, the output of this step is detected unknown attack detection along with input to new signature generation module.

**New signature Generation from detected Unknown Attack**

The Figure 4.8 shows that signature generation unit combines together with the above mentioned two detection subsystems. It characterizes detected unidentified attacks along with extraction of their signs and stores them in existing signature database automatically. It reduces further requirement of human expert to add new signatures form detected unknown attack data. Here, the signature generation is done on frequent signature generation mining mechanism is described in next section based on modified version of A priori algorithm with the weight addition to each generated signature called as weighted A priori Algorithm.

It is found that, rare connection outlines are excluded by usual data mining and it affects directly on system anomaly intrusion detection performance. On other side, if we lower the support threshold value during the production of frequent pattern, there is possibility of generation of huge amount of unwanted patterns at the time of innovative signature construction. Therefore, there is need to have some pattern pruning techniques to eliminate redundant patterns and reduction in search time for the new signature patterns.

To address and handle this problem, newfangled base support procedure based on data mining is presented. The process is specified in the following algorithm. The main motivation is adopted from Lee’s level oriented procedure. This is called as base support
traffic records mining algorithm specified by Kai Hwang et al. (2007) as given in the following Figure 4.12.

User can reduce the level of support value up to its minimum value at each time of Lee’s algorithm execution. In this one, initially, lowest support value is considered as maximum to identify the outline with respect to maximum occurrence value of feature traits. Afterword, it lowers the value of support inception by 50 percent for next repetition. It produces every innovative contender sign outline containing at least one value of feature attribute. When lowest threshold level is stretched, this process gets ended. The following example illustrates the idea behind this algorithm.

**Input(s):** Base support threshold value \( f_0 \), all features attributes of input dataset and the set \( T \) of all network connections record set

**Output(s):** New generated signature pattern to add into existing signature database set \( L \)

1. Start
2. For feature attribute set \( X \in T \), Calculate \( \text{support}(X) \);
3. Start Scanning of \( T \) and compute the elements of \( L = \{ \text{Itemset} \mid f(y) \geq f_0 \} \);
4. Generate new pattern sets \( E = \{ e_0, e_1, \ldots, e_n \} \).
5. Where check \( \text{support}(e_0, e_1, \ldots, e_n) \geq f_0 \times \min\{ \text{base sup}(e_j) \} \);
6. If \( (E = \emptyset) \). { 
7. Generate signature patterns from \( E \) with confidence \( c > \min e_0 \) where \( e_0 \) is minimum confidence.
8. Add the generated signature pattern into signature database set \( L \); } 

**Figure 4.12 Base Support Data Mining Algorithm**

Let \( X \) is a thing situated and base_ help estimation of the peculiarity characteristic thing set. For instance, consider the administration as service and banner as Flag, the gimmick properties, and then the support base for item set \( X \) can be represented as

\[
X = (\text{service} = \text{FTP}, \text{flag} = \text{50}, \text{src host} = 127.1.1.1, \text{dest} = 127.1.1.1)
\]

And it is calculated as

\[
\text{base sup}(X) = \text{support}(\text{service} = \text{FTP}, \text{flag} = \text{50}).
\]

Hence, the fractional value \( f \) of \( \text{base sup}(X) \) for item set \( X \) can be computed as

\[
f(X) = \frac{\text{support}(X)}{\text{base sup}(X)}
\]
Similarly, this fractional value $f$ of $\text{bure\_sup}(X)$ is defined as the rate of the quantity of least signature design events to the aggregate number of records in $T$ data dataset, which contains the most unprecedented peculiarity characteristics accessible in information dataset. The bases help base estimation of pattern set $(e_2, e_3, ..., e_n)$ is denoted by $\min\{\text{bure\_sup}(e_i)\}$.

With the AIDS model, detection of many new and attacks covered up in like manner Internet administrations used at information infrastructure, like, Telnet, HTTP, SMTP, email, Verification, etc. is conceivable. The AIDS model arrangement requests especially to secure network based frameworks of machines, assets inside interior systems (intranets), and computational networks configured at organization side.

4.4.2.3 Discussion on Support for Denial of Service (DoS) Attack Detection

Today’s another concern in IDS intrusion detection is DoS attack detection, though DoS attacks are covered under the signature based and anomaly based intrusion detection. So, sophisticated Denial of Service security advances like self-learning and threshold-based identification is available to deal with specifically DoS attacks which are explained as follows.

The detection architecture used for DoS attack detection identified combination of threshold based identification and protected self-learning profile based identification systems that convey knowledge to Denial of Service identification. With edge based identification, system security chiefs can use prearranged breaking points on information movement to guarantee servers won’t get to be inaccessible because of over-burden imposed on them by intruders. Meanwhile, self-learning systems empower the location construction modeling to study the examples of system use and movement, understanding the wide mixture of legal, however surprising, obscure, use designs that may happen amid true blue system operations and can cause denial of service.

The combination of these two technologies provides the highest exactness of identification for a full range of Dos assault including dispersed Denial of Service assaults, when hundreds or even a large number of servers are assaulted and entered by a pernicious developer to strike against an undertaking or government system. Thus, these Dos identification strategies are imperative on the grounds that mainstream sites and systems do
experience genuine what’s more off and on again startling activity infuse for an especially new program, administration, or application

4.4.2.4 Prevention Phase

This is the last and most important phase of this system. It includes attack detection correlation, prevention phase application, database updating on supporting signature database and management console for network administrator. In this phase, attacks detected by detection phase as shown in Figure 4.8 are given as input to this phase (Shown with labels ‘A’ and ‘B’ in Figure 4.14) and after their detection correlation, the suitable prevention policy can be applied to avoid the intrusion on the system.

Detection correlation is mainly done for the newly generated signature and existing signature database used for the signature detection in earlier phase. It compares these two signature patterns and keeps only that signature which will be more suitable and will cover the attack classes which could be identified and detected by individual use of either newly generated signature or existing similar signature used. So, it prunes the unnecessary and duplicated entries of similar attacks and hence sustain with storage capacity of servers efficiently. The following section gives idea behind the construction of this step.

In earlier section, the outcome of A priori algorithm is generally an association rule formed based on potential feature attributes. An association rule is basically used to find interesting interrelationships inside a solitary association record. The frequent signature patterns generated using the specified algorithm mentioned in Figure 4.12, describes the interrelationship among various association records of information dataset. Let TC be a set of movement associations. By and large, visit mark example is defined by the accompanying representation: L1, L2… Ln → R1, R2 … Rm (c, s, w, f),

Where, Li (1≤i≤ n) and Rj (1≤ j≤m) are connection occasions in an activity record set TC in requested arrangement. We can say, L1, L2… Ln as the LHS (Left Hand Side) design and R1, R2… Rm as the (Right Hand side) of the mark design. The (c, s, w, f) were characterized as parameters of edge levels and exemplified in prior segment.

We consider a continuous mark successful on the off chance that it is all the more every now and again utilized as a part of the discovery process. The example is incapable on the off
chance that it is once in a while utilized as a part of catching assaults. Kai Hwang et al. 
(2007) likewise, it is observed that few continuous mark designs differ just either at the LHS 
or at the RHS of them. Actually, keeping all recently produced marks will develop the 
inquiry space and in this way expand the overhead. For this situation, the accompanying 
change laws will diminish the mark pursuit space altogether what's more will accelerate the 
framework execution too. Consider the accompanying two often created marks as guideline 
in H Wang (2005),

(Service=smtp, flag=sf) $\rightarrow$ (Service=http, flag=s0), (Service=http, flag=s0) 

And 

(Service=smtp, flag=sf), (Service=http, flag=s0) $\rightarrow$ (Service=http, flag=s0) 

The feature attribute pattern (Service =http, flag = S0) is inferred by (Service =smtp, flag=sf). Hence, the second example can be supplanted by the first example. We just need to 
keep the to begin with example here. The general tenet of the thumb is utilized to make the 
LHS as short as could be allowed. LHS pattern be moved to the RHS if the new example 
fulfills the necessity of negligible certainty condition.

During the location stage, we produce stand out signature design from a successive 
mark designs. A substantial number of excess example examinations could be maintained a 
strategic distance from if more mind boggling examples were uprooted.

As given by Kai Hwang et al. (2007) examples produced with shorter LHSs are more 
powerful and productive than examples with longer LHSs. The fundamental reason is, shorter 
examples are less demanding to look at and assess. In the accompanying sample,

(Service=http) (Service=authentication, flag=sf) $\rightarrow$ (Service=smtp) (0.5, 0.2) 

This generated pattern is ineffective and unwanted due to the existence of the 
following pattern in the database i.e. (Service=authentication) $\rightarrow$ (Service=smtp) (0.55, 0.2). Since, the authentication service is is related just to the SMTP convention operation and the 
HTTP does not have any impact on the other two provided elements of pattern and hence, 
(Service=http) can be ignored and dropped from the generated pattern. Longer pattern may
introduce and may contain the redundant information which will lead towards increasing false alarms in the detection.

Many frequently generated patterns found with transitive in nature the network traffic of given input dataset. Consider, the example, suppose we have two examples like, A→B and B→C in the produced examples, at that point, we can devised them into the longer pattern as A→B, C. Since, however we got this example from two shorter designs, the longer design A→ B, C gets to be undesirable. The recreation of examples makes a difference us to decay longer designs into shorter and this one reduces the false positive rate which is one of the aims for the proposed system. Generally, here, the main focus is on network traffic input dataset. For example, in the accompanying example, (Service = ftp, src_byte=2,000) → (Service = smtp) (Administration = confirmation) is inadequate and undesirable, in light of the fact that it can be remade from the accompanying two shorter designs (Service=ftp, src_byte=2,000) → (Service=smt) and (Service=smt) → (Service=authentication).

It is also studied that reconstruction of such signature pattern is all the more capable if the time window size is expanded following the littler time window sizes, may generate longer patterns than the increased window size and may result in less false cautions.

As stated earlier, traffic information set is put away into typical connection table comprising of N associations after the preprocessing. Every connection Ci has M peculiarity quality worth sets like, <Aj, Vi, j >, where 1≤ i ≤ N and 1≤ j ≤ M. Presently when examples are produced, the weights can be assigned to identify potential signature patterns to utilize them for further processing.

The application of weight on signature is performed on weighted principle with the fact that if any association contains a thing set X, then it likewise comprises of all subsets of item set X and hence, if the weighted backing of subsets of X is computed, it must be more prominent than that of X.

Following Figure 4.13 specifies modified and considered as weighted from the earlier calculation for producing weighted incessant mark designs. The weighted backing of a thing set can be adjusting utilizing association weight. Every association is relegated distinctive inconsistency edge qualities, and we expect to discover successive mark era designs concerning defined threshold values. The following step shows the signature weight calculation.
Let $w_i$ be the weight of association $tc_i$, the weighted backing of a thing set $X$ is characterized by

$$\text{wsup}(X) = \sum_{tc} \sum_{tc_t \supseteq X, tc_t \in TC} w_i = \frac{W_i}{\sum_{tc_t \in TC} W_i}$$

The main purpose is to find all examples whose weighted backings are over the least backing. We are defining these patterns as weighted signature pattern. The anomaly score is calculated for weighing a connection and the minimum support ($\text{min}_\text{wsup}$) is used to select desired signature patterns.

$$L = \{ \forall \text{itemset} \; | \; f(y) \geq f_0 \}$$

$$E = \{ a_0, a_1, \ldots, a_n \}$$

$$\text{support}(a_0, a_1, \ldots, a_n) \geq f_0 \times \text{min}\{ \text{base}_\text{sup}(a_i) \}$$

$$(E \neq \emptyset)$$

$$\& \; \gamma > \text{min} \; \gamma_0$$

**Figure 4.13 Weighted Apriori Algorithm Kai Hwang et al. (2007)**

The algorithm is defined for weighted aberrance signature generations from the detected unknown attack. In this one, if the backing of mark example surpasses $\text{min}_\text{wsup}$ esteem, then every last bit of its subsets must be backed. This rule is utilized by the from the earlier calculation to successfully prune competitor signature pattern set and we can select only those signature patterns where none of its prompt supersets has a backing over the $\text{min}_\text{wsup}$ edge to get desired signature patterns. In this sense, the data correlation phase is constructed and get the exact attack signature pattern to detect the future similar attacks posed on the network by intruder with label as known attacks.
The following Figure 4.14 shows the prevention phase of this research work and further section gives the details about the functionalities of each prevention phase policy and their usage choice for the network administrator.

![Figure 4.14 Prevention Phase Used in System Model](image)

**Figure 4.14 Prevention Phase Used in System Model**

As we have seen, the discovery construction modeling empowers multi modes of operation that permit the framework to catch noxious activity gives intensive assault investigation techniques and executes a complete set of astute Signature Discovery, Anomaly Detection.

The detection Correlation layer associate the framework's Signature, Anomaly, and at some degree, Denial of administration discovery usefulness and this relationship furthermore cross checking of suspicious activity gives more exact assault identification than utilizing individual interruption discovery. Here, the endeavor is, a solitary framework ought to give comprehensive assurance by checking open, private, and portions of the system with firewall can offer connection among these portions to give more precise picture of system assaults that were either obstructed by the firewall or made them conceivable to go into the private system. The portrayed prior framework structural planning gives the most precise assault recognition capacities alongside the establishment for the framework's assault reaction components inside the same framework building design.

IDS without sufficient reaction limit get to be restricted utility to system security supervisors, managers. Today's present day world, IDS items must distinguish assaults and must give a few helps to redirect and stop noxious movement once it is identified. So, to provide such solutions and to make IDS responsive for the detection, this architecture gives network security supervisors with range of manual and programmed reaction activities that can structure the help of a venture's or government office's data innovation. Figure 4.15 shows the prevention phase policies in details.
So, the proposed system during this research work gives the following prevention policy responses which can be used by network administrator during prevention phase of the system

![Figure 4.15 Prevention Policies Used by Network Administrator during Prevention Phase](image)

Upon detecting an attack and its nature of severity, the prevention phase of the system model enables the system to do:

**Drop Packets:**
The system permits the IDS to function in logged off and also online mode, in online mode, it has the capacity drop or square a solitary parcel, single session, or activity stream between the assault source and end progressively, hindering an assault in progress without influencing whatever other activity information.

**Terminate Session:**
The counteractive action stage takes into consideration the launch of TCP resets to focus on frameworks, assailants, or both. The system security architect can design reset bundles to be sent to the source as well as terminus IP address.

**Modify Firewall Policies:**
This permits clients to reconfigure system firewalls as an assault happens by briefly changing the client defined access control approach while alarming the security administrator.

**Create Alerts:**
This alternative enables an alarm channel that permits system security specialists to filter out alarms in light of the source or the goal of the security occasion. Case in point, if the IT office executes powerlessness examines from one of its own IP addresses, occasions beginning from that address can be sifted out. Also, it gives the subtle elements of that specific client confirmation to set the future credits to the same source or goal machine.

**Log Packets Details:**

Systems focused around this structural engineering catch and log parcels former, amid, or resulting to the assault and can divert movement to an extra framework port for natty gritty scientific investigation. This bundle data goes about as a record of the real stream of movement that set off the assault. At the point when the information is seen it can be changed over to libpcap position for presentation with the instruments like Ethereal, a system convention analyzer for UNIX and Windows can be utilized to inspect the bundle log information for more point by point investigation of the discovered occasion what's more an assault. The framework building design's reaction gives the premise to the item stages that security administrators need to create an arrangement of activities, alarms, and logs that give ideal security to complex systems. In the proposed system, this prevention policy is provided in such a way so that at the time of its execution for user’s every action it logs its record. As mentioned earlier by examining this log data; it is possible to find out intrusive actions of the user and to reset their access rights for the network access.

While applying all these above mentioned prevention policies it is observed that most of the time the performance of the system is get affected by various internet connection features which are described and that can be analyzed with the help of offline nature of the proposed system to get details about attack types and their detection.

### 4.5 Conclusion

In this chapter, design details of research work system are presented. Based on the review of literature, it is observed that, there is need of integration of two or more IDSs along with quick response for detected attacks and hence, it was set as main objective of the proposed research work.

The section 4.2 has given the reasons behind studying various existing IDS modeling architectures which provide the advantages like, Architectures make a critical difference in
terms of effectiveness that intrusion detection and intrusion prevention technology produces. At the same time, it is important to understand that “Rome was not built in a day.” If organization is not sufficed with financial resources, use of simple single-tiered architecture is best option to develop IDS / IPS architecture. As organization grows, more resources are available, functionalities of intrusion detection and intrusion prevention efforts become clearly visualize, then architecture change is possible. A peer-to-peer architecture is suitable for those organizations that have enough investment to obtain and deploy cooperating firewalls but cannot offer the deployment of IDS/IPS. Moreover, Architecture provides foundation and guidelines to design and implement next generation IDS/IPS architectures.

In section 4.3, being the modeling of proposed system, the aspects of architectural design are considered initially with the theoretical formulation and later with mathematical formulation for the proposed system simulation to do performance evaluation. Theoretical formulation gives the necessary requirements and their analysis for the architectural design of the proposed research work. Mathematical formulation is used to give the details about the basic algorithms used during system simulation to describe the proposed system execution with certain time complexity as well as it gives the emphasis on generation of mathematical model required to exhibit inputs, processes and outputs of the proposed system to provide better understanding for the reader. In mathematical formulation of the proposed work, all the basic algorithms including Apriori Algorithm, Classification Model for intrusion detection, Level-wise mining algorithm, Relative Support Mining Algorithm.

In section 4.4, actual modeling of proposed system is done with in-depth illustration of data collection and its preprocessing for the proposed system. In proposed model, the provision of data preprocessing is made available for online as well as offline input data used for the proposed system based on data mining preprocessing techniques as explained in section 4.4.2.1. The section 4.4.2.3 discusses about the intrusion detection phase where, integration of signature based IDS and anomaly based IDS is done which is very rarely found in the literature review study and it is observed that existing IDSs are either created utilizing mark based or oddity based intrusion detection methodology with specific attack detection capabilities.
Lastly, section 4.4.2.4 highlights system’s attack response mechanisms within the same system architecture. IDS without satisfactory reaction limit get to be constrained utility to system security chiefs, overseers.

Today's current world, IDS items must catch assaults and must give a few supports to redirect and stop noxious movement once it is identified. So, to provide such solutions and to make IDS responsive for the detection, this architecture gives network security chiefs with range of manual and programmed reaction activities that can structure the support of an endeavor's or government organization's data engineering.

Based on the system architecture introduced in this chapter, the implementation is carried out and it is illustrated in chapter 5 in-depth and detailed result analysis of the simulated system is given in chapter 6.