International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. #, No.# (Mon-20..)

Failure Predictive and Remediation System for Windows Infrastructure

Deepali Arun Bhanage[1], Dr Ambika Vishal Pawar[2], Dr Aparna Joshi and[1] Dr. Rajendra G. Pawar[3]

[1]PCET's, Pimpri Chinchwad College of Engineering, Pune 411044, India
[2]Persistent University, Persistent Systems, Pune, India
[3]Department of Computer Science & Engineering MIT Art, Design and Technology University, Pune, India

E-mail address: deepali.naik@pccoepune.org, ambikap.imc@gmail.com, aparna.joshi@pccoepune.org, rgpawar13@gmail.com

Received ## Mon. 20##, Revised ## Mon. 20##, Accepted ## Mon. 20##, Published ## Mon. 20##

Abstract: The demand for IT infrastructures has grown due to their importance in business and everyday life. Downtime due to the unavailability of any IT infrastructure components is undesirable. Ensuring IT infrastructure's continuous availability and stability is crucial for organizations to prevent downtime and its associated consequences. Thus, prompt failure detection, analysis of underlying causes, and corrective measures are vital. IT infrastructure logs register every detail of the executed operation and provide a lot of dimensional information about it. Therefore, the research field of IT infrastructure failure detection and prediction using log analysis techniques is gaining prominence. The proposed method uses a BERT pre-trained model-based semantic analysis framework and an attention-based mechanism OLSTM classification model. Furthermore, the remediation model offers failure notifications to the system administrator on the dashboard and registered email ID, along with potential solutions to address the issue and mitigate the failure of IT Infrastructure components. The effectiveness of the developed prediction and remedial system was evaluated on a real-time Windows infrastructure by implementing a proof of concept. In this process, the trained model was utilized to analyse newly generated log entries and forecast potential failure situations. Consequently, a remediation strategy was applied in order to address the problem and prevent downtime effectively.
The integration of automatic failure detection and prediction using IT infrastructure logs has the potential to become a routine practice in IT infrastructure monitoring. The suggested remediation approach shows promise in being widely adopted for timely failure mitigation, resulting in reduced downtime.

Keywords: Log analysis, System log, IT Infrastructure, Deep Learning, BERT, POC

1. INTRODUCTION

The modern era is growing reliance on digital technology, and the continuous operation of IT infrastructure has emerged as a fundamental pillar for individuals and businesses. The dependency of organisations and institutions on IT infrastructure is evident in its role of delivering services to clients, handling crucial data, and facilitating internal communication. The widespread incorporation of technology into our everyday routines has emphasised the utmost significance of upholding the dependability and accessibility of IT infrastructure [1]. The topic of failure prediction and remediation includes the integration of advanced technology, data analytics, and proactive maintenance practices [2]. The concept centres on the proactive detection of possible failures, typically prior to their manifestation as tangible disturbances, and the prompt implementation of corrective measures to prevent or mitigate their consequences. This strategy surpasses conventional reactive methods, wherein system administrators react to failures only after they have already transpired.

In order to address the challenges associated with unavailability and enhance the dependability of IT infrastructures, a research study has been undertaken to develop a predictive and remediation system. The findings of this study have provided valuable insights and made

notable contributions to the respective domain. The robustness of derived results offers the potential to revolutionise IT infrastructure management to maintain its health.

The Windows Operating System as the infrastructure was selected in order to collect real-time log records and monitor the IT infrastructure during runtime. To gather a dataset in real time, the necessary preparations were made on personal computer, including installing several services, assets, software and activating an event manager to monitor the executing actions. Event viewer can be utilised to gather and retain event details, thereby creating a dataset that can be employed for further activities. When comparing log records and event records, it can be observed that event records contain more comprehensive information regarding the activity that was executed. The provision of detailed information pertaining to a specific event occurrence will benefit the system administrator in comprehending the circumstances around the failure and devising a strategy for resolving the issue.

In addition to the implementation of the predictive system, remediation has also been implemented for the Windows Infrastructure. The solution dataset was explicitly created for the purpose of remediation, focusing on events that signify anomalous levels, including critical, failure, and warning. In the solution dataset, the combination of source and event ID is unique. The predicted failure can be addressed by retrieving a viable remedy based on this unique combination. Additionally, the notification regarding the prediction of failure was sent to the system administrator through both email and a dashboard, together with the potential solution. This failure notification will prompt the system administrator to implement corrective measures to prevent downtime in the infrastructure components.

The email notification sent to system administrator contains both the log–event details and the possible solution. The provided event details will be helpful to understand the problem as well as gather further details such as hardware involved, assets connected, dependencies, channel, provider, etc. Therefore, the system administrator can study the specific information and implement appropriate measures as recommended by the remediation system.

The IT Industry necessitates such IT infrastructure monitoring, failure detection, and troubleshooting solutions to increase the stability, availability and reliability of IT infrastructure components due to the heightened demand and utilisation of intricate IT infrastructure.

Rest of the paper is arranged in the six sections. Section 2 elaborate on the work related to IT infrastructure failure detection, prediction and handling followed by background in section 3. Discussion on the dataset preparation and collection conducted in section 4. Experiment results are demonstrated in the section 5 and paper is concluded with section 6 conclusion and future scope.

2. RELATED WORK

In recent times, there has been an introduction of NLP-based analysis techniques for the purpose of comprehending the semantic content of logs in intricate IT infrastructures [3] [4]. Researchers have explained that the utilization of semantic analysis in the examination of textual logs may offer greater advantages compared to conventional analysis approaches [5]. In the realm of document processing, Natural Language Processing (NLP) approaches are frequently employed to discern the thoughts expressed by writers. The sentiment analysis was applied on tweeter data to identify instances of cyber-attacks. However, the use of NLP approaches for log analysis was not employed. The authors converted log data into vectors with several dimensions, representing various attributes. Subsequently, the researchers employed various classifiers, including the random forest, multilayer perceptron (MLP), and Gaussian Naive Bayes (NB), to identify instances of abnormal behaviour in the Vending Machine. These classifiers demonstrated an approximate accuracy of 90%. For embedding process on the specified dataset commonly employs popular algorithms are Bag of Words (BoW) [6], Term Frequency - Inverse Document Frequency (TF-IDF), Global Vectors for Word Representation (GloVe), and the feature matrix algorithm.

In their research, Bertero et al. [7] explored the unconventional application of NLP techniques. They utilized Google's word2vec algorithm for log mining and employed the Binary classifier, random forest, and Gaussian NB to identify abnormal behaviour within a Virtual Network (VN). DeepLog [8] utilized an LSTM neural network model to convert log records into plain language sequences and purportedly achieved a 100% accuracy in detecting anomalies. In the study, Wang, Xu, and Guo [9] applied the TF-IDF and Word2vec methodologies for feature extraction, converting words into vector representations. Researchers asserted that Word2vec exhibited superior performance in capturing semantic information inside log data compared to TF-IDF, hence enhancing the efficacy of anomaly detection. Researchers then utilized the generated vectors as input to a Long Short-Term Memory (LSTM) model for the purpose of detecting anomalies and reducing the occurrence of false alarms. The authors, Meng et al. [10] introduced a model called template2Vec, which drew inspiration from word2vec. This model was the first to use both semantic (synonyms and antonyms) and syntax information of log templates in order to identify anomalous

log sequences. Zhang et al. [11] introduced a series of semantic vectors into the Bi-LSTM (Bidirectional Long Short-Term Memory) model. LogRobust [11] framework does not rely solely on the basic occurrence information of log templates instead, it transforms each log template into a semantic vector of a predetermined dimension. This vector was designed to effectively capture the semantic information included inside the log template. Borghesi et al. [12] employed a semi-supervised approach with an autoencoder-based strategy in order to mitigate challenges associated with data tagging. In the research of Xie et al. [13] employed a confidence-guided anomaly identification model that integrates numerous methods in order to address the issue of idea drift. In the study, Wang et al. [14] introduced the LogEvent2vec offline feature extraction approach, which aims to extract the relationship between log events and vectored log events. The combination of LogEvent2vec with TF-IDF and Naïve Bayes demonstrates a notable reduction in computational time, with a duration of 30 minutes.

3. BACKGROUND

A. *Study of IT Infrastructure Logs and Events*

Logs containing messages from various operations that are generated by all components in the IT infrastructure. To generate IT infrastructure logs during the execution of the operations, a software developer writes predefined logging statements in the source code of the software. According to Zhang et al. [11], every 58th line in the source code is dedicated to logging. The logs record the activities and events of the assets and components in the IT infrastructure. These logs provide information about the status of each component and document operational changes in the components of IT infrastructure, such as starting or stopping services, software configuration modifications, software execution errors, and hardware faults. Therefore, they contain valuable details that can be used to understand and maintain the state of the IT infrastructure. Logs are widely used to analyse the behaviour of IT infrastructure and monitor their health. They are considered the primary data source as they record runtime information of the software. Each computer system generates logs for every event execution, resulting in a large number of records. These logs capture every detail of executed operations and provide extensive dimensional information. Additionally, logs document the causes of problems in IT infrastructure components. By analysing IT infrastructure logs, IT teams can monitor for anomalies or irregularities that may indicate potential failures. Log analysis is an effective and comprehensive method for managing, monitoring, intervening, predicting failures, and diagnosing root causes in IT infrastructure. As a result, IT infrastructure logs are widely used in anomaly and failure detection or prediction due to their usefulness [15][16][17][18]. However, the analysis of massive log collections from complex IT infrastructure has challenges. Each operating system has a specific logging technique in place to record and monitor users' activities through event logs [19]. These logs store details of important events that support the IT infrastructure and its applications. The recorded information is valuable for detecting software and hardware issues. Thus, logs contain crucial information about the operations and activities of the IT infrastructure components. Logs are a subset of Event records and provide more detailed information about activities in the IT Infrastructure components.

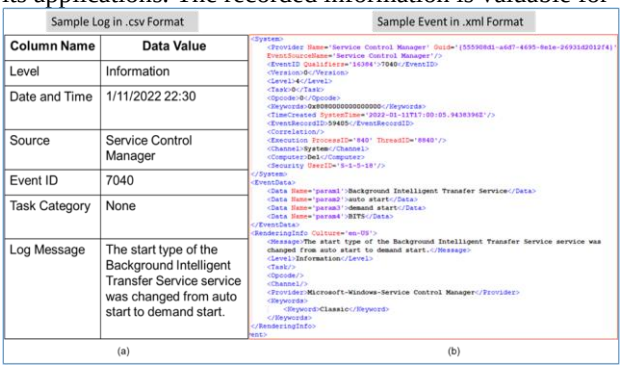

Figure 1. Sample Windows Log and Event Parameters

TABLE I. TYPES OF LOG OR EVENT LEVELS

Log/Event Level	Description
Fatal/ Critical (Always check)	The fatal level indicates the most severe issues in the business application, this log level presents a critical failure that prevents the whole system from satisfying the business functionalities. Example: In an e-commerce application users cannot connect to the payment portal or check the cart.
Error/ Failure (Always check)	Error level is recorded on the issue due to which one or more application functionalities get hampered but the application execution continues. Example: In the e-commerce application, the user cannot connect to the payment portal but can check the cart.
Warn (Check for possible future problems)	In warn level application behaviour changes and application becomes unpredictable. However, application continues to work and the primary functionalise execute as anticipated.
Info (Check to track specific events)	Info level is about the specific event which need to be track. An event took place, and the activity is absolutely informative and perhaps overlooked in the course of normal operations.
Debug (Check to track specific events)	A log-level allocation for events reviewed will be helpful during software debugging when more information is needed. Example: Records to check the installation of an application.
Trace (Check to track specific events)	This level portrays events screening step-by-step execution of the code that may be overlooked during the regular operation but may be helpful during extended debugging terms.

Figure 1 illustrates sample Windows log record presented in 1.2 (a) and sample event record presented in 1.2 (b). It is evident that event records contain more details compared to log records. Logs provide only six parameter values, while event records offer 21 parameter values. These additional parameters provide more information about the events that occurred, the system administrator can determine the appropriate actions to address and investigate the error by carefully analysing these event elements. Therefore, event logs are the preferred choice for troubleshooting network and device-related problems. When a failure occurs, the system administrator must identify the cause of the failure, try to recover any missing records, and prevent similar issues in the future. The system administrator can use it to determine the reasons for the failure, understand the circumstances in which it occurred, and suggest solutions to prevent potential failures by considering the various elements and valuable information present in the event data. The most important aspect of a log or event record is the level assigned to log, this level indicates the severity of the log statement. Depending on the level, the executed activity or operation can be classified as normal or abnormal. Table I provides a list of log/event levels along with examples and descriptions.

B. Log-Event Fusion Data

Every operating system records a distinct logging technique to document and monitor users' actions using event logs [19]. The Operating System stored details of meaningful events and applications running on the infrastructure component. The captured data has the potential to be beneficial in identifying and diagnosing software and hardware-related problems. Logs play a crucial role in documenting the operations and activities of the IT Infrastructure. Event logs can serve as a valuable tool for troubleshooting purposes. However, the question arises as to the source from which logs and related events data might be gathered for the purpose of analysis. The solution to the aforementioned issue lies in utilising the Windows event viewer, an advanced feature inside the Windows Operating System. The Windows Event Viewer offers a significant event log service that facilitates the examination and analysis of important messages inside the log records. The Event Viewer tool allows system administrators to monitor and analyse the comprehensive information regarding important events that have occurred within Windows-based Operating Systems. The Event Viewer application facilitates the recording and subsequent extraction and analysis of many types of logs. Table II presents the several categories of Windows logs that are retained within the Windows event viewer, namely application, security, system, and setup [20]. In the context of such occurrences, the system administrator can examine and address concerns about hardware and software components. Each logging statement is assigned a level. The allocation of logging levels is contingent upon the severity of the occurrence. The level is an essential parameter that plays a vital role in the analysis of logs and events.

Identifying the relationship between logs and events is of utmost importance. To get several supplementary parameters available in event records, which are formatted as .xml files. The inclusion of these further characteristics will prove advantageous for system administrators in identifying the underlying cause of the problem, resolving the issue promptly, and mitigating the negative impact on work productivity resulting from downtime. The data obtained in .xml format exhibits a higher granularity level and a durable structure.

TABLE II. TYPES OF WINDOWS LOGS PRESENT UNDER EVENT VIEWER

Log Name	Description
Application	It records the events logged by applications present in the system. For example, the Apache server is inactive.
Security	This type of event triggers when any authentication issue occurs. For example, Valid or invalid login attempts, and unauthorised access to files
System	It records system components related events. For example, driver failure
Setup	It preserves additional events that are configured as domain controllers.
Forwarded Events	It records the events that are forwarded from another system

Element	Type	Description
Provider	System Properties Type	Identifies the provider that logged the event.
EventID		The identifier that the provider used to identify the event.
Version		Contains the version number of the event's definition.
Level		Contains the severity level of the event.
Task		The task defined in the event.
Opcode		The opcode defined in the event.
Keywords		A bitmask of the keywords defined in the event.
TimeCreated		The time stamp that identifies when the event was logged.
EventRecordID		The record number assigned to the event when it was logged.
Execution		Contains information about the process and thread that logged the event.
Channel		The channel to which the event was logged.
Computer		The name of the computer on which the event occurred.
Security		Identifies the user that logged the event.
EventData	Event Type	Contains the event data.
Message	Rendering Info Type	Contains the event message that is rendered for the event.
Level		The rendered message string of the level specified in the event.
Task		The rendered message string of the task specified in the event.
Opcode		The rendered message string of the opcode specified in the event.
Channel		The rendered message string of the channel specified in the event.
Provider		The rendered message string for the provider.
Keywords		A list of rendered keywords.

Figure 2. Event Schema Elements (Source: https://docs.microsoft.com/en-us/windows/win32/wes/eventschema-elements)

Figure 2 displays a comprehensive compilation of event items together with their corresponding descriptions. The administrator can assess this abundant information to establish connections between relevant components. Based on a comprehensive examination of the many elements or characteristics associated with the incident, the system administrator can make decisions regarding the appropriate actions to be taken in order to address and thoroughly

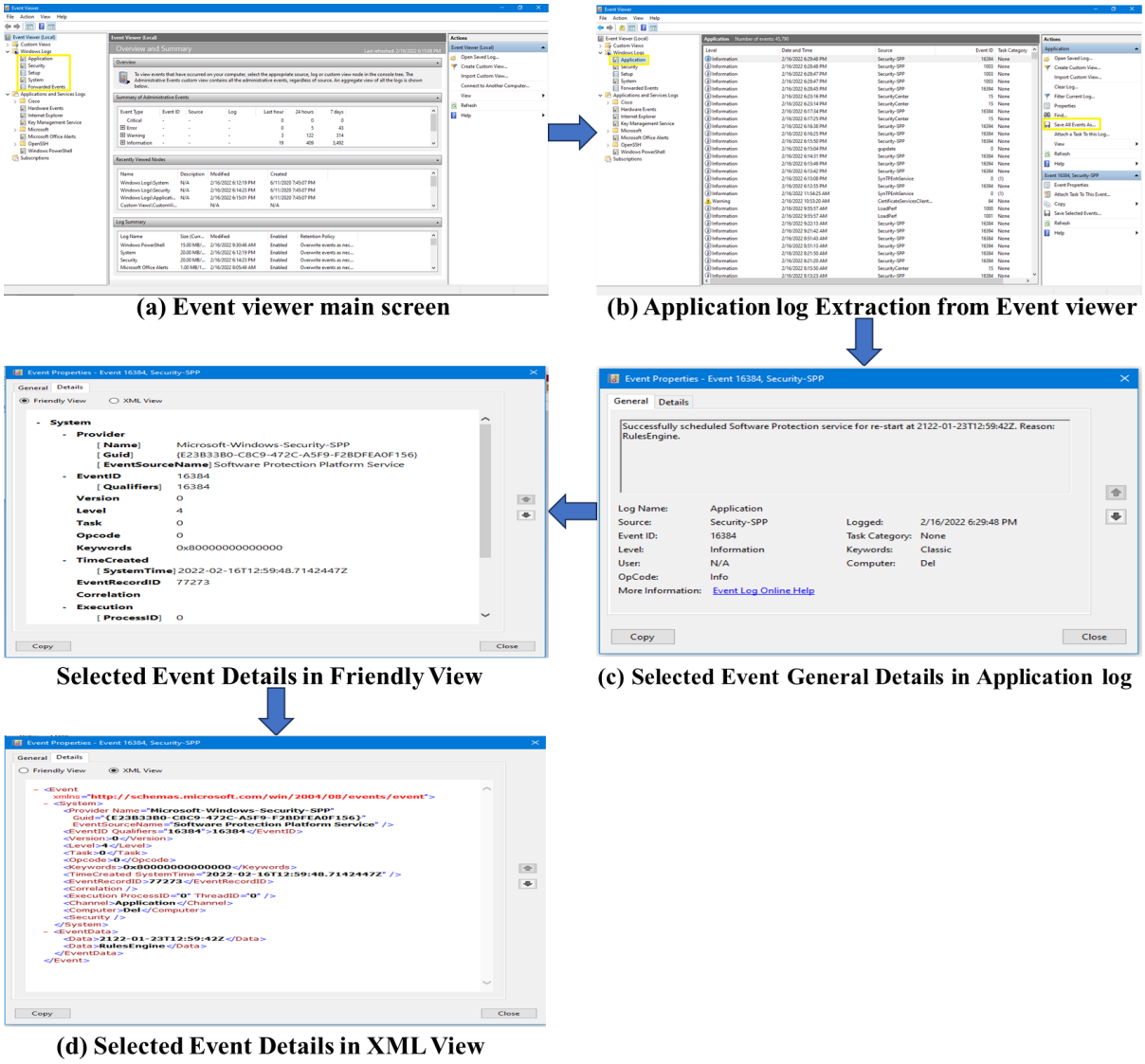

Figure 3. Steps to Extract and Visualize General and Detailed Information of Application Log in the Event Viewer

investigate the issue effectively. Therefore, event logs are suitable for addressing network and device-related issues.

In the event of a failure, it is imperative for the system administrator to ascertain the underlying cause of the problem, make efforts to recover any lost data and implement measures to prevent its recurrence in subsequent instances. With the help of many components and significant data contained within the events data, the system administrator can leverage it to discern the causes behind the failure, determine the specific conditions under which it occurred, and propose appropriate measures for addressing foreseeable failures.

4. DATA ACQUISITION

A. Log-Event Data Acquisition Techniques

To evaluate the proposed predictive system on Windows Operating System Infrastructure, runtime logs and event records were collected to construct the dataset. The Windows application was set to obtain different types of logs and related events. Windows system generates an ample number of events that must be collected, stored,

normalized and analysed to use it for further operations. Accordingly, we can collect generated Windows logs by two ways of extraction of necessary logs and events 1) using PowerShell (provides records on the console) and 2) using Windows Event Viewer Tool (present as a feature of Windows operating system). In the subsequent points, the process of logs and event collection is described with the help of two techniques.

- Using PowerShell

1) Start the Windows PowerShell app by going to start-> search box -> type PowerShell
2) Within PowerShell, execute the "Get-EventLog" command to get a log from the local computer
3) Need to specify “ComputerName” to reach to the logs of remote computer
4) Extracted logs can be stored by copying them in the required format file.

- Using Windows Event Viewer

1) Start Windows Event Viewer by going to Start -> search box -> type eventvwr.
2) Within Event Viewer, expand Windows Logs.
3) Click the type of logs you need to export.
4) Click Action -> Save All Events As
5) Select the file type in which records need to be stored (.evtx, .xml, .txt, .csv)
    a. If the file is stored with .txt or .csv, log records will be stored with Level, Date and Time, Source, Event ID, Task Category, and log message parameters.
    b. If the file is stored with .evtx or .xml, evet records will be kept with various elements of the system, EventData and RenderingInfo parameters.
6) Click the radio button to select display information in the English language.
7) Files will be available at stored locations.

The Windows Event Viewer is commonly utilised as the primary method for retrieving Windows logs and events from the underlying Operating System. Therefore, the Event Viewer tool was utilised to collect the necessary data in this study. Figure 3 illustrates the sequential processes entailed in extracting and visualising comprehensive and specific data from the application log within the event viewer. In Figure 3 (a), the primary interface of the event viewer is depicted, showcasing the different types of logs that are accessible. These log categories are visually emphasised by a yellow box positioned on the left side of the interface. In Figure 3 (b), the process of extracting application logs is demonstrated through the action of clicking on the "Save All Events as" option located on the right side of the screen. When data is stored in the .csv format, it offers a restricted amount of information, mainly consisting of log records. On the other hand, data saved in .xml format provides more comprehensive details, referred to as event details. The subsequent sections, 3 (c), (d), and (e), elucidate the varying perspectives about the presentation of specific information about the selected event, namely General details, Friendly view, and XML view, respectively.

B. Dataset Preparation

The logs were extracted in a .csv file, and the events were extracted in an .xml file, as observed through the event viewer. Ensuring that these two records were presented in a consistent style was crucial in order to establish their relationship. Consequently, the events file in XML format was transformed into a CSV file by means of a plugin integrated into Microsoft Excel. The data that has been transformed exhibits noise as a result of the presence of multilevel XML elements. Therefore, it is necessary to do data cleansing operations such as eliminating duplicate rows, consolidating columns created for sub-elements, and aligning columns with each element of the .xml file.

Figure 4 illustrates the method to build the fusion dataset comprising Windows logs and associated events. The timestamp serves as the conventional parameter within log and event records. In event file, timestamp recorded in Greenwich Mean Time (GMT) format, but the log file timestamp was present in Indian Standard Time (IST) format. The log timestamp was changed from Indian Standard Time (IST) to Greenwich Mean Time (GMT) to obtain an equivalent timestamp value. When two files were in the same format and match common parameters, they were merged.

The prediction model was trained using a log dataset, while the correlated event dataset is employed to enhance the information provided in the failure notification.

C. Remediation Dataset

The remediation system is activated in response to any anomalous occurrences within the operational aspects of IT infrastructure components. In the proposed system, the notifications regarding predicted failures were transmitted to the system administrator. The analysis and comprehension of the anomalous scenario necessitates an examination of the log and event particulars shared within the failure notification email. The aforementioned analysis necessitates a significant investment of time and specialised knowledge to comprehend the topic at hand and effectively address issues that may arise. The suggested remediation system aims to assist system administrators in effectively addressing failure conditions by offering feasible and practical solutions.

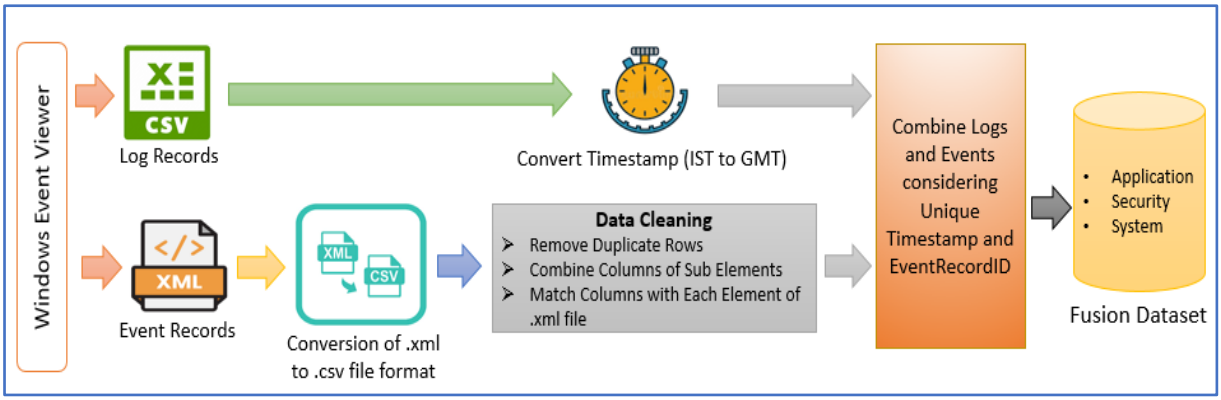

Figure 4. Methodology to Generate Fusion Dataset

Source	Event Id	Description	Action
MS Defender	5	Microsoft Defender for Endpoint service failed to connect to the server at variable.	Check the connection to the URL. See Configure proxy and Internet connectivity.
Microsoft-Windows-FailoverClustering	1193	Cluster network name resource '%1' failed to create its associated computer object in domain '%2' for the following reason: %3.	Check permissions and quota related to creating computer objects
Microsoft-Windows-CertificationAuthority	91	A connection to Active Directory Directory Services could not be established. Active Directory Certificate Services will try to connect again when it needs Active Directory access.	Enable AD CS to connect to Active Directory Domain Services
Microsoft-Windows-OnlineResponderRevocationProvider	16	For configuration %1, the Online Responder revocation provider failed to update CRL information: %2.	Enable access to current certificate revocation lists
Microsoft-Windows-OnlineResponder	33	Online Responder Service: For configuration %1, failed to create an enrollment request for the signing certificate template %2.(%3)	Submit an enrollment request for a properly configured signing certificate
Microsoft-Windows-ADFS	677	The AD FS auditing subsystem failed to write an audit event. An unexpected error occurred Additional Data The data field contains a Win32 error code	Configure the component to run as an appropriate principal
Active Directory Rights Management Services	183	Active Directory Rights Management Services (AD RMS) requires that you install ASP.NET before you install AD RMS.	Ensure that ASP.NET is listed as a required role service
Microsoft-Windows-Complus	4773	An unauthenticated message was received by an application that accepts only authenticated messages.%1%0	Check the calling application, and change its security properties
Microsoft-Windows-EventSystem	4359	The type library "%2" specified in EventClass %3 ("%4") could not be loaded, or is not correct for this EventClass. The HRESULT was: %1.%0	Report the event class error to the application developer
Microsoft-Windows-DistributedCOM	10000	Unable to start a DCOM server: %3. The error: "%2" happened while starting this command: %1	Correct the low-resource condition or report the error to Microsoft

Figure 5. Sample Failure Solution Database for Windows Infrastructure

The dataset about potential failure conditions in the Windows infrastructure has been compiled through the utilisation of documents provided under Windows documentation [21], discussions in the Windows communities [22], Netsurion- event tracker [23], and input from domain experts.

Figure 5 displays a subset of the solution dataset. The structure of the log entry includes the source, which denotes the location where the service/asset/component event took place. Additionally, an event ID was assigned uniquely to the logging statement within a given source. The log entry also contains a description of the event and provides guidance on the recommended remedial action to be taken. In this dataset, the source and the event ID combination are unique. The same combination is available in the log dataset also. The combination of a source and event ID is employed to establish the relationship between an abnormal log entry and the corresponding solution in the solutions dataset. The potential solution associated with the given problem was retrieved from the dataset of solutions. This solution was then shown on the dashboard and also specified in the notification email.

5. EXPERIMENTATION

A. Data Pre-processing

The Windows logs were generated on the execution of logging statements scripted during the software development. The Windows log extracted in the .csv format provides the information of Level, Date and time, Source, Event ID, Task Category, and log message. In this research "Log Content, EventTemplate and ParameterList" these parameters were used to perform the semantic analysis and further provided as features to the OLSTM classifier. "Log Content, EventTemplate and ParameterList" these features can be extracted from the log message using the parser. Various log parsers are available and discussed in the existing literature. Drain [24] parser has been adapted in this research work to parse log datasets considering execution availability, accuracy, and flexibility parameters of the parser. The Drain parser employed the fixed-depth tree structure to perform and retrieve log templates. Drain stipulates acceptable accuracy for different types of logs.

TABLE III. WINDOWS LOG DATASET STATISTICS AFTER PARSING WITH DRAIN PARSER

Infrastructure	Time Span	Number of log Messages	Number of Unique Template	Template Max Length
Windows Operating System	Dec 2020 – April 2023	2,81,366	2,676	538

The statistical details of the Windows dataset following the parsing procedure are presented in Table III. A total of 2,676 distinct log templates have been recovered, with the longest template having a length of 538. The number of distinct templates are considerably more in comparison to the dataset used in earlier experiments. The Windows dataset was obtained from personal computers, where deliberate actions were executed to capture diverse log entries.

The statistical details of the Windows dataset following the parsing procedure are presented in Table III. A total of 2,676 distinct log templates have been recovered, with the longest template having a length of 538. The number of distinct templates are considerably more in comparison to the dataset used in earlier experiments. The Windows dataset was obtained from personal computers, where deliberate actions were executed to capture diverse log entries.

B. Dataset Description

Table IV provides the details of the Windows real-time dataset that has been extracted from the personal computer for more than two years (Dec 2020 – April 2023). The event manager was activated to monitor all the activities occurring while using the computer system to collect the log and event data. To gather the different types of log records, the machine was prepared with the installation of various services and software. Also, some unexpected actions were intentionally performed so that the count of anomalous log entries will increase.

In the dataset, a total number of 2, 2,366 logs, whereas 59,893 events records were extracted. Here, the count of logs and events vary, although they were recorded on the execution of the same logging statement. More log statements can be recorded at different time stamps for the same event. Thus, there was a change in the count of logs and evens. Also, change in the count was one of the challenges while performing the fusion of the log-event dataset.

TABLE IV. WINDOWS LOG DATASET STATISTICS USED FOR EXPERIMENTATION

Infrastructure	Data set	Source Type	Time Span	Number of logs	Number of Events	Number of Unique Template	No of Anomalous Records
Windows Operating System	Windows event log	Operating System	Dec 2020 – April 2023	2,81,366	59,893	2,676	45,208

The unique templates were extracted after the parsing operation (performed using a Drain parser). In this dataset, a total of 2,676 unique templates are available. This means different types of events have occurred on the computer system.

The last column of Table IV states the number of anomalous records in the dataset. A total of 45,208 anomalous log entries were recorded, approximately 16% of the whole dataset. Compared to the dataset available in the literature, this ratio of normal to anomalous records is more. Thus, with the help of adding error conditions at the time of usage of the computer delivers the balanced dataset in some proportion.

C. Evaluation on Unseen Windows Logs

In this section the semantic based embedding technique and attention-based classifier was applied on the real-time Windows operating system's log dataset and the results were recorded in terms of Accuracy, Precision, Recall, F1-Score, and Specificity matrices.

This section presents an evaluation of the proposed system's robustness using a dataset of unstable entries from the Windows Operating System event log. The presence of many logging statement formats, the absence of a standardized approach to designing logging statements, and the introduction of noise during the parsing process all contribute to the inherent instability of log data. Windows unstable dataset was synthesized, with the injection ratio of 10%, 20%, 40%, 60% and 80% as training data. Based on the results collected during the experimentation, observation says that the BERT pre-trained model leveraged semantic information to identify "Log Content, LogTemplate and ParameterList". At the same time, most of the state-of-the-art tools work only with Event/Log Template.

The proposed system was trained using five distinct IT Infrastructure logs available in the literature for experimentation purposes. More details regarding these experimentations are available in our paper [25][26]. Subsequently, the system was evaluated on a real-time, unseen dataset consisting of Windows event logs. The experimental results of the BERT + OLSTM model applied to the real-time Windows event log with different training ratios are displayed in Table V and Figure 6. The proposed system exhibits enhanced robustness while handling unstable, unseen, and newly occurring log entries, particularly under conditions of heavy injection.

TABLE V. EXPERIMENTAL RESULT OF BERT PLUS OLSTM MODEL EXECUTED ON THE REAL-TIME WINDOWS EVENT LOG WITH VARYING TRAINING RATIO

Training Ratio	Accuracy	Precision	Recall	F1-Score	Specificity
10%	94.57	95.37	99.37	96.73	94.40
20%	95.40	95.20	96.20	94.56	94.23
40%	95.97	96.77	97.77	96.13	95.80
60%	97.11	96.61	97.61	95.97	95.64
80%	98.05	97.85	98.85	97.21	96.88

In order to evaluate the performance of the OLSTM model, several vital metrics were employed, including Accuracy, Precision, Recall, F1-score, and Specificity. Although Accuracy may not be the sole comprehensive statistic in failure detection or prediction study, it is vital to contemplate a blend of metrics customized to the particular application at hand. The evaluation of failure detection or prediction systems frequently relies on metrics such as Accuracy, Precision, Recall, F1-score, and Specificity, as they provide more meaningful and relevant insights into system performance. These metrics provide a more comprehensive view of the model's behaviour, particularly in scenarios where class imbalance and the cost of false positives and false negatives are significant factors. The attribute of specificity offers valuable insights into the rates of false positives, assists in the selection of appropriate thresholds, and plays a significant role in the comprehensive evaluation and optimization of proposed systems.

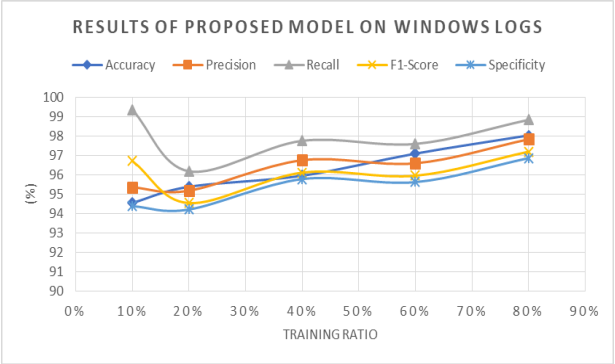

Figure 6. Experimental Result of BERT plus OLSTM Model Executed on the Real-Time Windows Event Log with varying Training Ratio

D. Testing of the Model/ Case Study

The architecture presented in Figure 7 illustrates the comprehensive design of a simulator specifically developed to carry out a predictive and remedial system. This simulator is utilized to showcase the execution of a research model on a real-time Windows infrastructure that has been trained on historical data.

Step 1: The initial step involves preparing a Windows PC by installing services such as "Windows Audio Service, Windows Biometric Service, Windows Updater Service, Security Centre, Disk, and Windows Service Simulator." The configured services "Windows Audio Service, Windows Biometric Service, Windows Updater Service, and Security Center" are standard services in the Windows operating system, whereas the "Windows Service Simulator" service was designed specifically for POC. The Windows Service Simulator service allows for the execution of events under specified requirements while monitoring state changes accordingly.

Step 2: A controlled environment was established to systematically execute various events, ensuring that logs were accurately documented.

Step 3: In the third step, the logs that were produced during the execution of events can be accessed to analyse them using a research model that has been developed. During the log processing step, the log entry was subjected to parsing, semantic embedding, and attention-based classification processes. If the results of the model execution were supplied in the form of an abnormal record, more procedures will be undertaken. When a normal log record occurs, it will be displayed on the

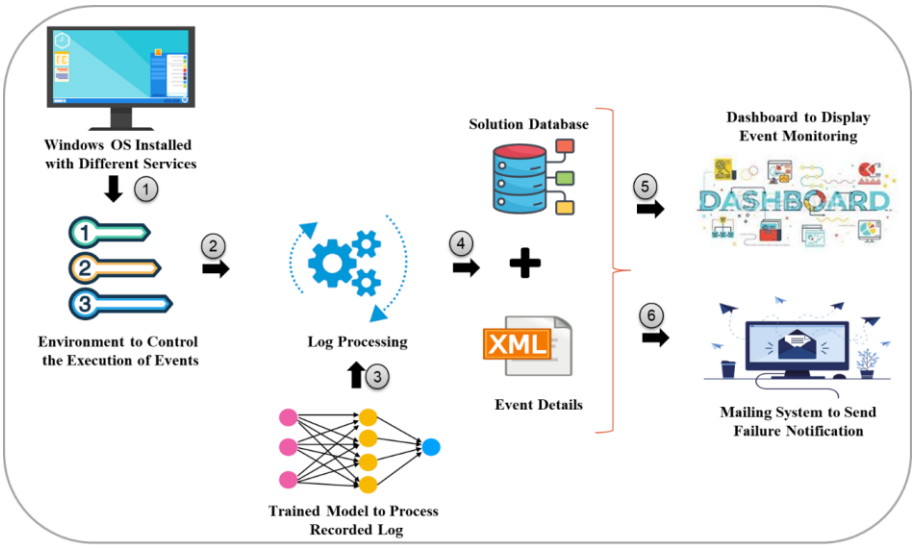

Figure 7. Architecture of Predictive and Remediation Simulator

dashboard with a green color and a message indicating that no action is necessary.

Step 4: On the log record predicted as an abnormal condition, it was necessary to execute the remediation module. In order to effectively communicate failure notifications, transmitted them through both the dashboard and email channels. These notifications includes the failure message, suggested actions, and event details for comprehensive information sharing. Therefore, the recommended course of action was extracted from the solution database using a unique combination of the source and event ID. The specifics of the event, in the .xml format, were communicated by email as an attachment. These details were taken from the dataset of the event.

Step 5: In Step 5, the dashboard presents the anomalous logs, which were shown as either orange or red based on the severity of the log. Additionally, the recommended course of action was displayed.

Step 6: In Step 6, the system sends a mail notification to the registered system administrator, providing additional information about the event. This email includes suggested actions to be taken and specific facts about the event.

Notifications shared to the system administrators are presented in the figure 8 as dashboard and figure 9 as sample mail.

Figure 8. Dashboard to Show Predicted Notification

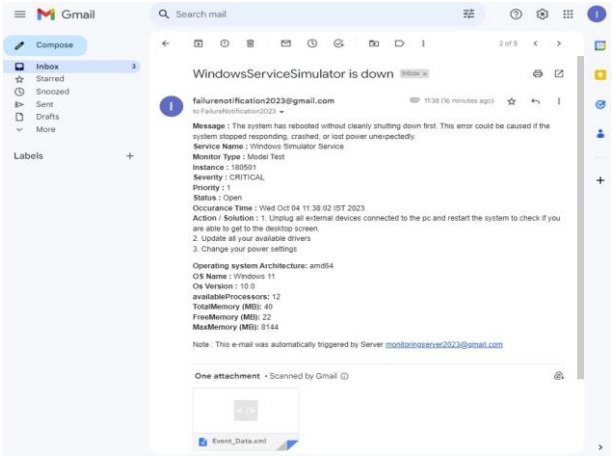

Figure 9. Sample Failure Alert Mail

6. CONCLUSION AND FUTURE SCOPE

The proposed model underwent training using five distinct datasets of IT Infrastructure logs, as available in the existing literature, to perform the log analysis, anomaly detection, and failure prediction operations. However, it should be noted that the historical datasets exhibit an imbalance, with a limited number of failure conditions due to overall stability observed in the components of the IT Infrastructure.

Furthermore, event details were utilized in the proposed research to understand the failure situation comprehensively. Access to log and event records of any private IT infrastructure is not feasible due to the sensitivity of the data. To obtain the necessary dataset, the personal computer was equipped with the installation of several services and software applications. Also, the event viewer was enabled to monitor all activities taking place within the computer system. With the help of event viewer logs, event records were extracted, and the fusion dataset was prepared to establish the association between logs and related events. Further extracted logs were used for experimentation by applying a log parsing process using Drain parser followed by semantic embedding using BERT pre-trained model and then attention based OLSTM classifier for prediction.

In case prediction result was a failure condition, alert notifications and potential solutions were sent to the registered system administrator. The solution dataset was compiled by consulting many sources, including the official documentation provided by Windows, discussions on the Windows forum, the Netsurion-Event Tracker, and input from domain experts. The right solution can be determined by utilizing a unique combination of the Event ID and Source in both the log and solution datasets. The suggested remediation system aims to assist system administrators in effectively addressing failure conditions by offering feasible and practical solutions.

A user interface with a simplistic design has been developed for the aim of system testing. This interface allows for the input of log data and then predicts the system's behaviour, classifying it as anomalous or normal. When the results were predicted as anomalous, an alter notification and the potential solution were sent to the system administrator. The trained system was evaluated on unstable and previously unseen Windows logs, and it was noted to exhibit superior performance, specifically in the context of new records.

REFERENCES

[1] D. A. Bhanage and A. V. Pawar, “Bibliometric survey of IT Infrastructure Management to Avoid Failure Conditions,” Inf. Discov. Deliv., no. September, 2020, doi: 10.1108/IDD-06-2020-0060.

[2] D. A. Bhanage, “DigitalCommons @ University of Nebraska - Lincoln Review and Analysis of Failure Detection and Prevention Techniques in IT Infrastructure Monitoring,” 2021.

[3] H. Ott, J. Bogatinovski, A. Acker, S. Nedelkoski, and O. Kao, “Robust and Transferable Anomaly Detection in Log Data using Pre-Trained Language Models,” 2021, [Online]. Available: http://arxiv.org/abs/2102.11570

[4] D. A. Bhanage, A. V. Pawar, and K. Kotecha, “IT Infrastructure Anomaly Detection and Failure Handling: A Systematic Literature Review Focusing on Datasets, Log Preprocessing, Machine & Deep Learning Approaches and Automated Tool,” IEEE Access, vol. 9, pp. 156392–156421, 2021, doi: 10.1109/access.2021.3128283.

[5] D. A. Bhanage and A. V. Pawar, “Improving Classification-Based Log Analysis Using Vectorization Techniques,” pp. 271–282, 2023, doi: 10.1007/978-981-19-9228-5_24.

[6] T. Kimura, A. Watanabe, T. Toyono, and K. Ishibashi, “Proactive failure detection learning generation patterns of large-scale network logs,” IEICE Trans. Commun., no. 2, pp. 306–316, 2019, doi: 10.1587/transcom.2018EBP3103.

[7] C. Bertero, M. Roy, C. Sauvanaud, and G. Tredan, “Experience Report: Log Mining Using Natural Language Processing and Application to Anomaly Detection,” Proc. - Int. Symp. Softw. Reliab. Eng. ISSRE, vol. 2017-Octob, pp. 351–360, 2017, doi: 10.1109/ISSRE.2017.43.

[8] M. Du, F. Li, G. Zheng, and V. Srikumar, “DeepLog: Anomaly detection and diagnosis from system logs through deep learning,” Proc. ACM Conf. Comput. Commun. Secur., pp. 1285–1298, 2017, doi: 10.1145/3133956.3134015.

[9] M. Wang, L. Xu, and L. Guo, “Anomaly detection of system logs based on natural language processing and deep learning,” 2018 4th Int. Conf. Front. Signal Process. ICFSP 2018, pp. 140–144, 2018, doi: 10.1109/ICFSP.2018.8552075.

[10] W. Meng et al., “Loganomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs,” IJCAI Int. Jt. Conf. Artif. Intell., vol. 2019-Augus, pp. 4739–4745, 2019, doi: 10.24963/ijcai.2019/658.

[11] X. Zhang et al., “Robust log-based anomaly detection on unstable log data,” ESEC/FSE 2019 - Proc. 2019 27th ACM Jt. Meet. Eur. Softw. Eng. Conf. Symp. Found. Softw. Eng., pp. 807–817, 2019, doi: 10.1145/3338906.3338931.

[12] A. Borghesi, A. Bartolini, M. Lombardi, M. Milano, and L. Benini, “A semisupervised autoencoder-based approach for anomaly detection in high performance computing systems,” Eng. Appl. Artif. Intell., vol. 85, no.

June, pp. 634–644, 2019, doi: 10.1016/j.engappai.2019.07.008.
[13] X. Xie, Z. Jin, J. Wang, L. Yang, Y. Lu, and T. Li, "Confidence guided anomaly detection model for anti-concept drift in dynamic logs," J. Netw. Comput. Appl., vol. 162, no. February, pp. 1–10, 2020, doi: 10.1016/j.jnca.2020.102659.
[14] J. Wang et al., "LogEvent2vec: LogEvent-to-vector based anomaly detection for large-scale logs in internet of things," Sensors (Switzerland), vol. 20, no. 9, pp. 1–19, 2020, doi: 10.3390/s20092451.
[15] S. Khatuya, N. Ganguly, J. Basak, M. Bharde, and B. Mitra, "ADELE: Anomaly Detection from Event Log Empiricism," Proc. - IEEE INFOCOM, vol. 2018-April, no. April, pp. 2114–2122, 2018, doi: 10.1109/INFOCOM.2018.8486257.
[16] W. Meng et al., "Device-Agnostic Log Anomaly Classification with Partial Labels," 2018 IEEE/ACM 26th Int. Symp. Qual. Serv. IWQoS 2018, no. 1, pp. 1–6, 2019, doi: 10.1109/IWQoS.2018.8624141.
[17] S. Zhang, "PreFix : Switch Failure Prediction in Datacenter Networks PreFix : Switch Failure Prediction in Datacenter Networks," no. June, 2018, doi: 10.1145/3219617.3219643.
[18] S. Satpathi, S. Deb, R. Srikant, and H. Yan, "Learning Latent Events from Network Message Logs," no. i, pp. 1–21.
[19] J. Talebi, A. Dehghantanha, and R. Mahmoud, "Introducing and analysis of the Windows 8 event log for forensic purposes," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 8915, pp. 145–162, 2015, doi: 10.1007/978-3-319-20125-2_13.
[20] E. Logging et al., "No Title".
[21] "Technical documentation | Microsoft Learn." https://learn.microsoft.com/en-us/docs/ (accessed Oct. 03, 2023).
[22] "event ID 88 - Microsoft Community." https://answers.microsoft.com/en-us/windows/forum/all/event-id-88/726c885e-fc54-4ac3-885e-df8a33544531 (accessed Oct. 03, 2023).
[23] "EventTracker Knowledgebase." https://kb.eventtracker.com/ (accessed Oct. 03, 2023).
[24] P. He, J. Zhu, Z. Zheng, and M. R. Lyu, "Drain: An Online Log Parsing Approach with Fixed Depth Tree," Proc. - 2017 IEEE 24th Int. Conf. Web Serv. ICWS 2017, pp. 33–40, 2017, doi: 10.1109/ICWS.2017.13.
[25] D. A. Bhanage and A. V. Pawar, "Robust Analysis of IT Infrastructure ' s Log Data with BERT Language Model," vol. 14, no. 6, pp. 705–714, 2023.
[26] D. A. Bhanage, A. V. Pawar, K. Kotecha, and A. Abrahim, "Failure Detection Using Semantic Analysis and Attention-Based Classifier Model for IT Infrastructure Log Data," IEEE Access, vol. 11, no. September, pp. 108178–108197, 2023, doi: 10.1109/ACCESS.2023.3319438.

DEEPALI BHANAGE received master's in computer engineering from Sinhgad Institute of Technology, University of Pune. She is currently pursuing a PhD from the Symbiosis Institute of Technology, Symbiosis International (Deemed University), Pune. She is employed as an Assistant Professor with PCET's Primpri Chinchwad College of Engineering, Pune. Her research interests include IT Infrastructure Monitoring, Machine Learning, Deep Learning, and Natural Language Processing

DR AMBIKA PAWAR is currently Senior Manager, Learning & Development. Heading Higher Education and University Tie-up, Persistent University, Persistent Systems, Pune, India. She has 20+ years of experience as an academician and 11+ years as a researcher. She has received a PhD degree from Symbiosis International (Deemed University), Pune, India. She has published 50 research paper publications in international journals/conferences and one book published by Taylor & Francis, CRC Press. According to Google Scholar, her articles have 140 citations, with an H-index of Six and an i10-index of four. Her research interests include security & privacy solutions using Blockchain & AIMLDL Technologies.

DR APARNA SHASHIKANT received a PhD degree from Department of Computer Science & Engineering, Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology Chennai, India. Author working as an Assistant Professor at Department of computer engineering department at PCET's Primpri Chinchwad College of Engineering, Pune. Authors area of interests is cloud computing.

DR RAJENDRA PAWAR working as an Associate Professor, School of Computing, MIT Art, Design and Technology University Pune. Authors area of interests is AI, ML.