Research Proposal

Timely Virtual Machine Failure Prediction using Log Analysis for Proactive Fault Tolerance

Pratheek Senevirathne
2019cs162@stu.ucsc.lk
Index number: 19001622

Supervisor: Dr. Dinuni Fernando
Co-Supervisor: Dr. Jerome Dinal Herath

May 2023

University of Colombo School of Computing
Colombo, Sri Lanka.

Declaration

The project proposal is my original work and has not been submitted previously for any examination/evaluation at this or any other university/institute. To the best of my knowledge, it does not contain any material published or written by another person, except as acknowledged in the text.

Student Name : P. L. W. Senevirathne
Registration Number : 2019/CS/162
Index Number : 19001622


05/29/2023

Signature & Date

This is to certify that this project proposal is based on the work of Mr. P. L. W. Senevirathne under my supervision. The project proposal has been prepared according to the format stipulated and is of acceptable standard.

Supervisor Name : Dr. Dinuni K. Fernando

Signature & Date

Co-supervisor Name : Dr. Jerome Dinal Herath

Signature & Date

Contents

List of Acronyms

VM Virtual Machine

OS Operating System

VMM Virtual Machine Monitor

PM Physical Machine

AWS Amazon Web Services

GCP Google Cloud Platform

CDC Cloud Data Center

FT Fault Tolerance

SLA Service Level Agreement

ML Machine Learning

CI Continuous Integration

CD Continuous Delivery

CNN Convolutional Neural Network

LSTM Long Short-Term Memory

FNN Feed-forward Neural Network

SVM Support Vector Machine

LR Linear Regression

VNF Virtual Network Function

NFV Network Functions Virtualization

vEPC Virtual Evolved Packet Core

IDS Intrusion Detection System

NLP Natural Language Processing

BERT Bidirectional Encoder Representations from Transformers

1 Introduction

A Virtual Machine (VM) can be thought of as a software emulation of a physical computer. This technology allows several Operating Systems (OSs) to run on a single computer/server, each with its own virtualized hardware, and each VM functions separately from the other VMs in the same Physical Machine (PM). VMs are used for a variety of purposes, including testing and development, application deployment, build Continuous Integration (CI)/Continuous Delivery (CD) pipelines, server consolidation, running legacy applications, etc. They are managed by the hypervisor which is also known as a Virtual Machine Monitor (VMM), a piece of software that creates and manages VMs.

As with any system, VMs are failure prone. VM failures may occur at any moment of its execution or during migration from one host (PM) to another. VM failure can be seen as a failure in the end-user application/service running in the VM. An end-user application/service may experience a failure when there is a failure or an error in the PM (server) components, Network, or in software components such as the host OS, hypervisor, VM instance, the guest OS running in the VM, or due to a failure in the end-user application itself (Jhawar and Piuri 2012).

VM failure can cause downtime and disrupt service availability, leading to negative impacts on users and business operations. By utilizing failure prediction and proactive VM migration techniques, we can greatly reduce service downtime due to VM failures. One effective approach for proactive VM migration is live migration, which enables VMs to be migrated to a healthy PM while maintaining service continuity (Engelmann et al. 2009; Polze, Tröger, and Salfner 2011).

The software components related to VMs generate logs that contain valuable information about systems' states and events, including fault and failure data. Using Machine Learning (ML) techniques to analyze these logs, it is possible to predict VM failures and proactively migrate VMs from faulty PMs to healthy ones before the failure occurs (Nam, J. Hong, et al. 2021).

1.1 Motivation

The software industry has widely adopted deploying applications and services on large-scale cloud platforms like Amazon Web Services (AWS), Google Cloud Platform (GCP), and Microsoft Azure in recent years (Bulao 2023). These cloud service systems are required to offer a range of services to millions of users worldwide every day, which makes high service availability crucial, even minor issues can have significant consequences, and many service providers have put in considerable effort to maintain it (Khalili 2020).

Cloud VM is one such service provided by the cloud vendors, and they use several fault tolerance methodologies to keep the VMs up and running most of the time with minimal service downtime. For example, AWS claims to have "five nines" (Reynolds 2020), which means a service availability of 99.999%, allowing at most 26 seconds of downtime per month per VM.

Despite significant effort devoted to quality assurance, cloud service systems continue to face many problems and frequently fail to meet user requests. These problems are often due to computing node (physical server) failures and VM failures within cloud service systems(Lin et al. 2018; Lawrence 2022). Such systems typically comprise a vast number of computing nodes that provide processing, network, and storage resources for VM instances.

Accurate failure prediction will reduce VM downtime significantly, but as identified by Lin et al. (2018), the failure prediction of cloud service systems is extremely challenging due to the following reasons,

Complex causes of node failure: Because of the complexity of the cloud architecture, the node (PM) failure may be caused by different software or hardware issues.

Complicated failure indication: Detection of failure of the node is hard and could be indicated by many signs, and one node failure may affect other nodes to fail due to explicit/implicit dependencies among them. We need a proper way of defining node failure.

Highly imbalanced failure data: Node fault/failure data are extremely unbalanced, meaning, most of the data collected by the nodes will be allocated to the healthy class, and failure data are very rare because of the low failure rate of the nodes.

Researchers have attempted to achieve high VM availability by building a node failure prediction algorithm to proactively move the failure-prone VMs to a healthy node using live migration. They have used several strategies to achieve this while tackling the issues mentioned earlier. Most of them have used physical server (node) resource usage history data to train an ML model to predict the possible future node failure. Even though these papers claim to have achieved successful node failure prediction, they have left out the most crucial part of any digital system that keeps track of the system state and the events, the logs.

VM live migration takes some time, say x minutes, to move a VM from one node to another, so we need a way to reliably predict the VM failure before the time it takes to move the VM to a healthy node. That is, the failure should be predicted x minutes to VM failure. Most of the papers in this area have ignored this basic fact, and even though the failure prediction may be accurate, the VM may fail during migration due to late failure prediction.

Overall, utilizing log analysis and machine learning to predict VM failures before the time it takes to migrate the VM and proactively migrating the VM using the live migration technique is an effective strategy for reducing service downtime and ensuring a seamless user experience.

1.2 Background

1.2.1 Virtual Machine Live Migration

The live migration technique tries to migrate the VM while it is switched on, and the applications are still running in them with minimal interruptions to the running applications (Clark et al. 2005). The main objectives of this approach are to

optimize the end-user application performance and to minimize downtime. There are three sub-categories, Pre-copy migration, Post-copy migration, and hybrid approach. Figure 1 illustrates the timeline of pre-copy vs post-copy methods.

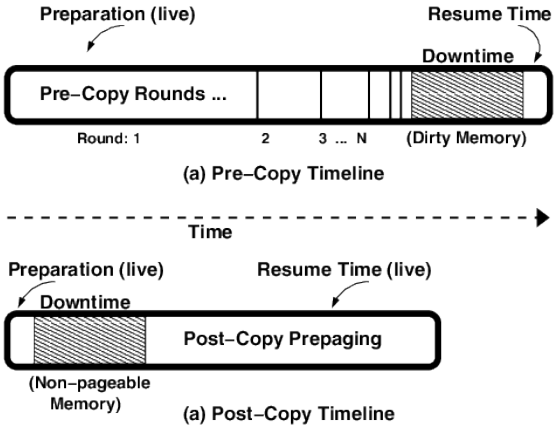


Figure 1: Pre-copy vs Post-copy timeline (M. Hines and Gopalan 2009)

1. **Pre-copy:**

   A pre-copy migration scheme will migrate the VM from a probable failing node to a target node by iteratively copying all of its memory content before stopping it in the source node and activating it in the target node. This migration scheme is resource intensive and the total migration time is also comparatively high (Shribman and Hudzia 2013).

2. **Post-copy:**

   This scheme will migrate the VM from the host node to the target node by immediately suspending the VM and capturing the minimum possible state to migrate the VM. This is then moved to the target, and the VM is started in the target, and if the guest OS requires a page to read/write which is not already copied, the migration system will copy it to the VMs memory on the fly, over the network. When all the pages are copied from the source server, the connection to the server will be terminated and the source VM can then be terminated (M. R. Hines, Deshpande, and Gopalan 2009).

3. **Hybrid approach:**

   These frameworks integrate the properties of both the previous methods to reduce migration time and improve system performance. It includes a finite pre-copy stage before moving on to migrating the VM and running the post-copy stage. This greatly reduces the page faults that may occur in the future as a large amount of the memory is already copied, so this method reduces the workload on the network and improves the application performance(Ahmad et al. 2015).

All the existing VM migration schemes can be classified mainly into two groups, namely, proactive migration and reactive migration (Ahmad et al. 2015; Polze,

Tröger, and Salfner 2011). Proactive migration frameworks continuously monitor the system and predict fault occurrence using an ML or a similar approach and if a VM or the PM is predicted to fail, it will migrate the VM(s) to a chosen healthy PM. This work is focused on proactive migration, where the failure is predicted using an ML approach by analyzing VM/PM logs.

1.2.2 Reactive Virtual Machine Migration

The reactive VM migration method is the most widely adopted and used method in cloud VM Fault Tolerance (FT). Reactive migration deal with faults after they have happened, they migrate the VM to a suitable destination node after a fault/failure has been detected. Using this method will take some time to get the failed VM properly running back again, which may lead to Service Level Agreement (SLA) violation. Even though this method is undesirable, one advantage of this method is that the overhead associated with running a machine learning algorithm to predict fault occurrence is completely avoided in this mechanism (Ahmad et al. 2015).

1.2.3 Log Analysis for Failure Prediction

VM logs and the physical server logs can be used to predict the failure of the VM and the physical server. We can utilize the system, kernel, and application logs available in the VM guest OS and the host OS for the prediction task. These logs are in different formats and contain unnecessary details which will affect the prediction accuracy of an ML model, so the raw logs should be pre-processed to convert them to a uniform format with unnecessary data removed (Nam, J. Hong, et al. 2021).

The pre-processed logs can be used in supervised ML model training, but most of the logs will be assigned to the "normal" class because the VM will not fail under regular operation. Waiting for the VM to fail is not feasible because a VM may not fail in its usual operation and may take months or years to fail under normal conditions. To tackle this issue, we need a way to simulate VM and server-overloaded situations; fault injection is an effective way of simulating the VM and server overload. During the overloaded operation of VMs and the server, we can capture near VM failure logs and failure VM logs. However, a similar approach has been followed by Nam, Yoo, and J. W.-K. Hong (2022), and from their test setup, they have only observed 11 failures over two months, even though they have heavily overloaded the memory and the CPU throughout the experiment period.

1.2.4 Fault Injection

Fault injection techniques can be utilized to deliberately overload the CPU and memory of virtual machines (VMs) and servers, enabling us to evaluate their performance and resilience under extreme conditions. By simulating excessive workload and resource demands, fault injection allows for identifying and collecting near-failure and failure logs of VMs and servers. The near-failure logs will contain

warning messages, fault logs, and other valuable logs for VM failure prediction. Techniques like stressing the processors with heavy computational tasks or running multiple resource-intensive applications can be utilized for CPU overloading. Similarly, memory overloading can be achieved by allocating excessive memory to applications or generating many memory-intensive operations.

1.2.5 Machine Learning for Log Analysis

We can utilize several supervised, semi-supervised, and unsupervised ML techniques for log analysis. Recent studies on the subject (Nam, Yoo, and J. W.-K. Hong 2022; Nam, J. Hong, et al. 2021; Jeong et al. 2021), have primarily utilized supervised ML models for failure prediction by analyzing the logs, and they have shown that several ML models prove to be effective, and the authors have shown that Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) models have shown superior performance over other models in the reviewed papers.

Some ML models like CNN require the logs to be in a numerical format such as vectors. There are several approaches to achieve this conversion. One common method is to employ word embedding, which involve using a language model to convert log words into tokens and then represent them as vectors. This technique captures the semantic meaning of the words and their relationships within the log data (Nam, J. Hong, et al. 2021). Alternatively, we can also consider converting the logs to a log event count representation, where each log event is treated as a feature, and the count of occurrences of each event is recorded (Jeong et al. 2021). This approach simplifies the log data into a numerical format that can be directly used by ML models. In the next section, I will discuss how the authors of the papers I have read have approached this conversion step, and which ML algorithms they have used, providing further insights into their methodologies.

2 Preliminary Literature Review

2.1 Physical Machine Failure Prediction

In large-scale cloud computing environments with thousands of physical servers, it is inevitable to encounter frequent server failures. According to the studies by Vishwanath and Nagappan (2010), and Birke et al. (2014), approximately 6-8% of all servers experienced at least one hardware issue during a year. Therefore, accurately predicting PM failures is crucial for ensuring cloud system FT.

Numerous studies have been conducted on this topic, and most researchers have utilized PMs' resource usage history data in combination with ML approaches to predict failure. The system administrators label the collected data to train a suitable supervised ML model to predict the failures. For example, the framework proposed by Guan, Z. Zhang, and Fu (2012) uses an unsupervised failure detection model using an ensemble of Bayesian models, and the found anomalous data get verified and labeled by system administrators. This labeled data is then used to

train a random forest model to classify the server resource usage data as failure prone or healthy. The framework proposed by Sun et al. (2019) uses a CNN model to predict the PM failures. A similar framework proposed by Gao, Wang, and Shen (2020) uses a Bi-directional LSTM model.

One notable framework that stands out from the rest is the MING framework, developed by Lin et al. (2018) at Microsoft Research. Its successful deployment in a production cloud environment sets MING apart, demonstrating its practicality and effectiveness. MING uses temporal data, such as performance data, log rate, and OS events, and spatial data, such as the server rack location and load-balancer data, to train a LSTM and Random forest models, respectively, to predict the server failure. MING also has a server ranking system, which will rank all the servers from their failure-proneness. Top k servers can then be selected as the faulty servers.

2.2 Virtual Machine Failure Prediction

When it comes to a VM failures, it can be due to either the physical server it is running on or any of the software components involved, such as the host OS, hypervisor, or guest OS. So, physical server failure prediction is a subset of VM failure prediction.

An analysis conducted by Birke et al. (2014) revealed that 60% of the collected VM failure cases were attributed to physical server failures, while the remaining 40% were caused by other factors. One approach to addressing VM failures is the utilization of redundant VMs. For instance, Scales, Nelson, and Venkitachalam (2010) discuss the VMWare VSphere 4.0 VM FT architecture, which employs a redundant VM pattern to replicate the entire execution state of the primary VM through a backup VM on another physical server. Nevertheless, this redundancy strategy can lead to increased costs for cloud service providers, which is not ideal.

An alternative approach involves predicting VM failures and proactively migrating the failure-prone VMs to other physical servers. Similar to the prediction of physical server failures, several studies have focused on VM failure prediction using ML models based on the VM's resource usage history. For instance, Saxena and Singh (2022) conducted a study utilizing an ensemble of predictors using Feed-forward Neural Network (FNN), Support Vector Machine (SVM), and Linear Regression (LR) ML models to identify failure-prone VMs and proactively migrate them to a different host.

Although there is a significant body of research on PM/VM failure prediction utilizing resource usage data, the literature on VM failure prediction using log analysis remains relatively scarce. I will discuss the few papers I found that specifically address this topic, in the related works section.

2.3 Virtual Machine Live Migration Time Estimation

VM live migration time estimation/prediction is another aspect of successful VM failure prediction because the failure should be predicted before the time it takes to migrate the VM. If not, there will be migration failures or downtime. In this

study, I focus on QEMU-KVM live migration, where the VMs' disk images are located on a network file system, and hence, disk image migration is not required.

There are several studies on KVM live migration time estimation/prediction, where most use a statistical method to calculate the estimated migration time. In this work, a statistical approach is preferable over an ML-based prediction technique because of low resource utilization.

Elsaid, Abbas, and Meinel (2022) have conducted a survey on VM live migration cost-modeling, and under section 7 of their paper, they have compared several models for VM migration time estimation. The simplest model they found was t = as+b, where s is the VM memory size, and a and b are constants. This shows that the VM migration time directly depends on the VM memory. However, the study by Nathan, Bellur, and Kulkarni (2015) shows several additional parameters affecting pre-copy migration time, namely, VM memory size, page transfer rate, number of unique pages dirtied during each iteration, and the number of skipped pages. The memory size and the transfer rate can be predetermined. The other two parameters depend on the application running on the VM. Thus, to accurately predict the migration time of the VM, we may need to use VM resource usage data.

The discussed literature focuses on the pre-copy live migration approach. In most cases, the pre-copy takes longer to migrate a VM when compared to other methods (M. R. Hines, Deshpande, and Gopalan 2009; Shribman and Hudzia 2013). Thus, the above-mentioned migration time prediction method is sufficient for this research.

2.4 Related Works

The framework proposed by Nam, J. Hong, et al. (2021) focuses on predicting the future failure of Virtual Network Functions (VNFs) in a Network Functions Virtualization (NFV) environment built on OpenStack cloud management software (OpenStack 2023). VNF is a VM that runs a network function application such as a firewall or an Intrusion Detection System (IDS). They leverage log data generated by the VNF application and the VM to predict failures. To convert the log data into word embeddings, they employ a Natural Language Processing (NLP) technique using the Google Word2Vec library (Mikolov et al. 2013). A CNN model is then used for failure prediction. The Word2Vec-CNN model achieved an overall F1 score of 0.67, predicting VM failure before 5 minutes of the actual failure.

The authors collected the training log data by using a fault injection method to simulate VM failures. Log data is collected at intervals (m minutes), and labels indicating whether the VM failed are collected after a gap time (n minutes). The trained CNN model predicts VM failure before the gap time. However, the authors observed some unexpected failures with no corresponding failure logs. Therefore, VM failure prediction through log analysis may not cover all possible VM failure scenarios.

In their subsequent work (Nam, Yoo, and J. W.-K. Hong 2022), the same authors improved their approach to predicting VM and PM failures in a similar NFV environment. Instead of Word2Vec, they employed the Google Bidirectional

Encoder Representations from Transformers (BERT) (Devlin et al. 2019) for word embedding and a CNN model for prediction. The BERT-CNN model achieved an F1 score of 0.74, predicting server failure 30 minutes before the actual failure. However, the impact of the prediction models on VMs' performance is not mentioned in either paper.

Jeong et al. (2021) proposed a framework to predict paging failure of Virtual Evolved Packet Core (vEPC) in 4G networking. They used VM and server logs and resource usage information to predict VM failure. Unlike the NLP approach in the previous papers, they used the log count (number of occurrences) of specific log types and resource usage data as inputs to an LSTM model for VNF failure prediction. They evaluated the total system throughput with and without the proposed proactive migration system and demonstrated that the framework successfully prevents long-term vEPC failures leading to depleted throughput.

It is important to note that there is a scarcity of research in the area of virtual machine failure prediction using log analysis, as indicated by the limited number of papers identified during the search. This highlights the need for further research and exploration in this specific field to bridge the existing knowledge gap.

3 Research Gap

The existing studies in the field of VM failure prediction have primarily focused on utilizing physical machine resource usage history and physical component health data to predict failures. However, an important aspect overlooked in these studies is the potential insights provided by VM and PM logs. These logs contain valuable information about VM behavior, performance, and potential failure indicators, which can significantly enhance failure prediction accuracy.

Furthermore, existing research in this specific area has been limited to studying specific VMs, such as VNFs, without generalizing the prediction for generic Linux VMs. This limits the failure prediction technique for a broader range of VMs commonly used in cloud computing environments.

Another crucial factor that has been largely disregarded by previous authors is the consideration of the time required for VM migration when predicting failures. For successful VM migration, it is imperative to predict the failure before the time it takes to migrate the VM to another physical machine. Unfortunately, most studies have neglected this aspect, leading to sub-optimal migration strategies and potential downtime.

Additionally, if the entire physical server is predicted to fail, all the VMs must be migrated. However, this process can be time-consuming. Therefore, it is crucial to predict the failure of the physical server in advance to ensure sufficient time for the successful migration of all the VMs running on it.

By addressing these research gaps, this study aims to leverage VM and PM logs to accurately predict the failure of any generic Linux-based VM and the PM. Furthermore, This research will consider the migration time factor to ensure timely and efficient VM migration in the event of failure.

4 Research Questions

1. How to effectively utilize VM and PM logs for machine learning-based VM failure prediction?
2. How to develop a generalized VM failure prediction approach using log analysis, enabling its applicability to a wide range of generic Linux-based VMs?
3. How can the timing of VM and PM failure prediction be optimized to ensure successful VM migration to a healthy PM, considering the total time required for the VM migration?

5 Aims and Objectives

The main aim of this study is to advance the field of VM failure prediction using log analysis for VM fault tolerance, contributing to improved reliability, and performance of VMs in server environments such as Cloud Data Centers (CDCs).

These are the main objectives of this study:

- To develop a generalized ML-based prediction approach that leverages key events and indicators present in VM and PM logs to predict failures in a variety of Linux-based VMs.
- To make the prediction time-aware so that it considers the time required for migration to ensure successful VM migration to another physical machine.
- To evaluate the proposed prediction approach and compare its performance against existing techniques using VM and PM log data.

6 Scope

6.1 In Scope

- **VM and PM failure prediction:** The study will focus on developing and evaluating techniques for predicting failures in both VMs and PMs running Ubuntu Server as the host OS, using the logs from VMs, PMs, and hypervisor.
- **Log analysis:** The research will explore the utilization of logs generated by the VMs, PM, and QEMU-KVM hypervisor to extract log events, and system states for failure prediction using an ML approach.
- **Real-time prediction:** The project will emphasize the development of real-time or near-real-time prediction models that can continuously monitor VM and PM logs to predict failures on time while considering VM migration time.

- **VM live migration:** The study will consider the use of QEMU-KVM live migration or the adaptive live migration technique that will be developed by my research colleague for efficient VM migration during failure scenarios.
- **Implementation of a working prototype:** The research will involve the development of a functional prototype that demonstrates the proposed failure prediction techniques using QEMU-KVM as the hypervisor.

6.2 Out of Scope

- **Non-Ubuntu host OSs:** The research will focus exclusively on Ubuntu Server as the host OS running on the physical machines.
- **Non-Linux guest OSs:** The research will not extensively cover VMs based on guest OSs other than Linux, such as Windows, or macOS.
- **Hypervisors other than QEMU-KVM:** This study will specifically utilize the QEMU-KVM based hypervisor for the implementation and evaluation of the prototype. Other Cloud Infrastructure software such as OpenStack (OpenStack 2023) will not be considered.
- **Handling unexpected VM failure situations:** This research will focus on predicting failures based on available log data, but pre-failure log data may not exist for some VM failures. Thus, the failure prediction of VM using log analysis is not effective in this situation.
- **Network-related failures:** The research will not specifically address failures related to network components or network infrastructure. It is important to note that in cases where network failure occurs, migration of the VM may not be feasible or effective, as the migration process itself may be impacted by the network failure.

7 Significance of the Research

This research project aims to enhance existing VM failure prediction approaches by incorporating VM, PM, and hypervisor logs. This contributes to the field by expanding the knowledge and understanding of effective failure prediction strategies in VMs. By accurately predicting VM and PM failures, this research project enables proactive measures to be taken, reducing system downtime and ensuring continuous availability of services. This is particularly significant for critical applications and services that rely on cloud computing infrastructure, such as healthcare systems, financial services, and emergency response systems, and this will help to provide improved user experience and service quality for end-users.

8 Planned Research Approach

Data Collection: Since there are no real-world Linux VM failure log data available, the log data needs to be collected using a tested, and the VM/PM failures must be simulated using fault injection methodologies because it is infeasible to wait for an actual failure to occur.

Testbed Setup: The set-up testbed contains two physical servers, each equipped with 12-core Intel Xenon processor with 16 GB of RAM. The two servers are interconnected with Gigabit Ethernet. The host OS is Ubuntu Server, and the chosen hypervisor is QEMU-KVM, running variety of VMs utilizing Linux-based OSs (guest OS). Figure 2 presents the tentative high-level architecture of the testbed that will be used for log data collection.

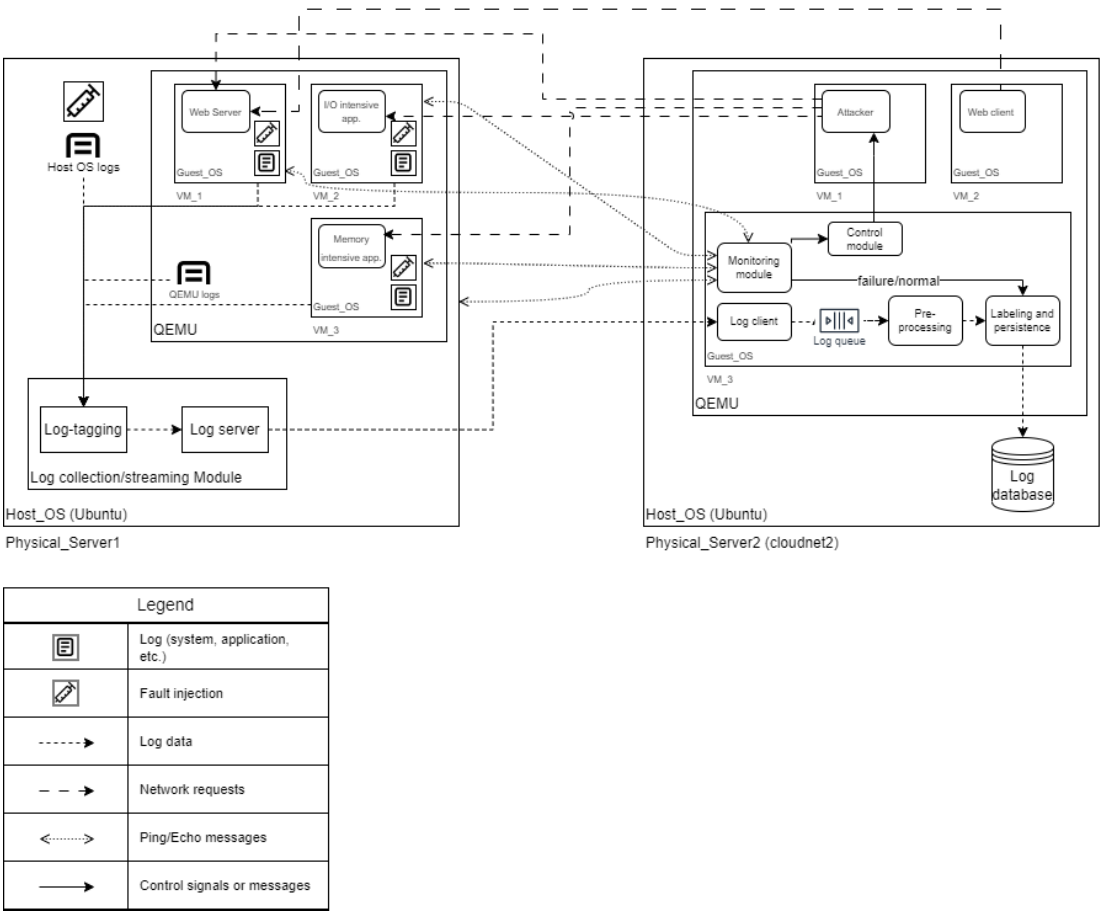


Figure 2: High-level architecture of the testbed

Data Preprocessing: The collected log corpus should be preprocessed in order to use them in a ML model. The preprocessing step may involve removal of unnecessary logs, duplicates, process numbers, and other unnecessary sentences/words/numbers from the log data. The raw log data are of different for-

mats, so they should also be converted to a uniform format by extracting the timestamp from the different types of logs.

ML Model Development: There are several ML models that prove to be effective for log analysis and failure prediction, such as CNNs and LSTM models. Each model can be individually developed and evaluated to select the best model, but first, for some models, the pre-processed log data must be converted to a numeric format. There are several approaches to achieve this conversion, such as word embedding, or using a language model like BERT for tokenization.

Evaluation of ML model: The developed models can be evaluated using appropriate evaluation metrics such as accuracy, precision, recall, and F1-score. The performance of the proposed approach can be compared with existing approaches, taking into account the accuracy of both VM and PM failure prediction. To validate the models' generalizability, additional log data from Linux-based VMs can be used for testing purposes, going beyond the specific logs utilized in the model's development.

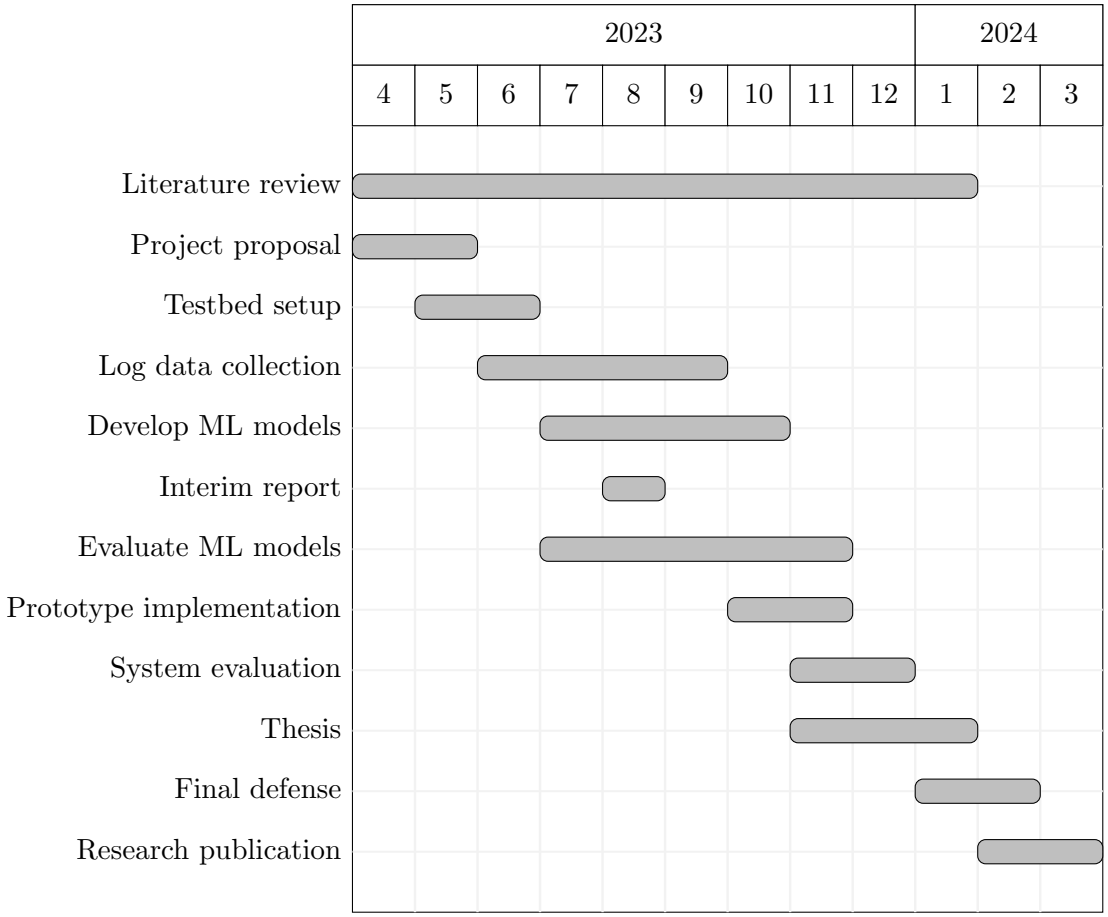
Timing and Migration Analysis: The timing aspect of the prediction in relation to VM migration should also be thoroughly analyzed to check for timely prediction of failures.

Prototype Implementation: A working prototype will be implemented to integrate the developed VM failure prediction technique with QEMU live migration, in order to demonstrate proactive fault tolerance for VMs.

Whole system evaluation: The implemented prototype system can be evaluated through experiments conducted on the test-bed to assess the efficiency, and reliability in simulated failure scenarios.

9 Project Timeline

Tentative timeline is as follows:



References

Ahmad, Raja Wasim et al. (2015). "Virtual machine migration in cloud data centers: a review, taxonomy, and open research issues". In: *The Journal of Supercomputing* 71.7, pp. 2473–2515.

Birke, Robert et al. (2014). "Failure Analysis of Virtual and Physical Machines: Patterns, Causes and Characteristics". In: *2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pp. 1–12. DOI: 10.1109/DSN.2014.18.

Bulao, Jacquelyn (2023). *How Many Companies Use Cloud Computing in 2023? All You Need To Know.* URL: https://techjury.net/blog/how-many-companies-use-cloud-computing/#gref (visited on 01/22/2023).

Clark, Christopher et al. (2005). "Live migration of virtual machines". In: *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume 2*, pp. 273–286.

Devlin, Jacob et al. (2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* arXiv: 1810.04805 [cs.CL].

Elsaid, Mohamed Esam, Hazem M Abbas, and Christoph Meinel (2022). "Virtual machines pre-copy live migration cost modeling and prediction: a survey". In: *Distributed and Parallel Databases* 40.2-3, pp. 441–474.

Engelmann, Christian et al. (2009). "Proactive Fault Tolerance Using Preemptive Migration". In: *2009 17th Euromicro International Conference on Parallel, Distributed and Network-based Processing*, pp. 252–257. DOI: 10.1109/PDP.2009.31.

Gao, Jiechao, Haoyu Wang, and Haiying Shen (2020). "Task failure prediction in cloud data centers using deep learning". In: *IEEE transactions on services computing* 15.3, pp. 1411–1422.

Guan, Qiang, Ziming Zhang, and Song Fu (2012). "Ensemble of Bayesian predictors and decision trees for proactive failure management in cloud computing systems." In: *J. Commun.* 7.1, pp. 52–61.

Hines, Michael and Kartik Gopalan (Mar. 2009). "Post-copy based live virtual machine migration using pre-paging and dynamic self-ballooning". In: pp. 51–60. DOI: 10.1145/1508293.1508301.

Hines, Michael R, Umesh Deshpande, and Kartik Gopalan (2009). "Post-copy live migration of virtual machines". In: *ACM SIGOPS operating systems review* 43.3, pp. 14–26.

Jeong, Seyeon et al. (2021). "Proactive live migration for virtual network functions using machine learning". In: *2021 17th International Conference on Network and Service Management (CNSM)*. IEEE, pp. 335–339.

Jhawar, Ravi and Vincenzo Piuri (2012). "Fault tolerance management in IaaS clouds". In: *2012 IEEE First AESS European Conference on Satellite Telecommunications (ESTEL)*. IEEE, pp. 1–6.

Khalili, Joel (2020). *Here's how much cash Google lost due to last week's outage.* URL: https://www.techradar.com/news/google-blackout-saw-millions-in-revenue-vanish-into-thin-air (visited on 01/28/2023).

Lawrence, Andy (2022). *2022 Outage Analysis Finds Downtime Costs and Consequences Worsening as Industry Efforts to Curb Outage Frequency Fall Short*. URL: https://uptimeinstitute.com/about-ui/press-releases/2022-outage-analysis-finds-downtime-costs-and-consequences-worsening (visited on 01/28/2023).

Lin, Qingwei et al. (2018). "Predicting node failure in cloud service systems". In: *Proceedings of the 2018 26th ACM joint meeting on European software engineering conference and symposium on the foundations of software engineering*, pp. 480–490.

Mikolov, Tomas et al. (2013). *Efficient Estimation of Word Representations in Vector Space*. arXiv: 1301.3781 [cs.CL].

Nam, Sukhyun, Jibum Hong, et al. (2021). "Virtual machine failure prediction using log analysis". In: *2021 22nd Asia-Pacific Network Operations and Management Symposium (APNOMS)*. IEEE, pp. 279–284.

Nam, Sukhyun, Jae-Hyoung Yoo, and James Won-Ki Hong (2022). "VM Failure Prediction with Log Analysis using BERT-CNN Model". In: *2022 18th International Conference on Network and Service Management (CNSM)*. IEEE, pp. 331–337.

Nathan, Senthil, Umesh Bellur, and Purushottam Kulkarni (2015). "Towards a comprehensive performance model of virtual machine live migration". In: *Proceedings of the Sixth ACM Symposium on Cloud Computing*, pp. 288–301.

OpenStack (2023). *What is OpenStack?* URL: https://www.openstack.org/software/ (visited on 05/17/2023).

Polze, Andreas, Peter Tröger, and Felix Salfner (2011). "Timely Virtual Machine Migration for Pro-active Fault Tolerance". In: *2011 14th IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops*, pp. 234–243. DOI: 10.1109/ISORCW.2011.42.

Reynolds, Ryan (2020). *Achieving "five nines" in the cloud for justice and public safety*. URL: https://aws.amazon.com/blogs/publicsector/achieving-five-nines-cloud-justice-public-safety/ (visited on 05/12/2023).

Saxena, Deepika and Ashutosh Kumar Singh (2022). "OFP-TM: an online VM failure prediction and tolerance model towards high availability of cloud computing environments". In: *The Journal of Supercomputing* 78.6, pp. 8003–8024.

Scales, Daniel J, Mike Nelson, and Ganesh Venkitachalam (2010). "The design of a practical system for fault-tolerant virtual machines". In: *ACM SIGOPS Operating Systems Review* 44.4, pp. 30–39.

Shribman, Aidan and Benoit Hudzia (2013). "Pre-copy and post-copy vm live migration for memory intensive applications". In: *Euro-Par 2012: Parallel Processing Workshops: BDMC, CGWS, HeteroPar, HiBB, OMHI, Paraphrase, PROPER, Resilience, UCHPC, VHPC, Rhodes Islands, Greece, August 27-31, 2012. Revised Selected Papers 18*. Springer, pp. 539–547.

Sun, Xiaoyi et al. (2019). "System-Level Hardware Failure Prediction Using Deep Learning". In: DAC '19. Las Vegas, NV, USA: Association for Computing Machinery. ISBN: 9781450367257. DOI: 10.1145/3316781.3317918. URL: https://doi.org/10.1145/3316781.3317918.

Vishwanath, Kashi Venkatesh and Nachiappan Nagappan (2010). "Characterizing cloud computing hardware reliability". In: *Proceedings of the 1st ACM symposium on Cloud computing*, pp. 193–204.