# AVAILABILITY ANALYSIS OF PREDICTING WEB SERVER FOR SOFTWARE AGING

Dr.Ohnmar Nhway
Faculty of Computer Systems and Technologies
University of Computer Studies (Pyay)
ohnmarnhway77@gmail.com

**Abstract**: Nowadays, software faults are the major cause of computer system failure. Many researchers have reported the phenomenon of 'software aging', characterized by progressive performance degradation due to the exhaustion of operating system resources. Software rejuvenation should be planned and initiated in the face of the actual system behaviour which requires the measurement, analysis and prediction of system resource usage. In this paper, we show availability analysis and predicting web server for software aging in virtualized environment. We focus on prediction method for software aging using Linear regression, DecisionStump and M5P from machine learning algorithms. Firstly, we collect data for response time and memory usage from a web server based on Apache Server using Linux. Secondly, we present load and time dependent software rejuvenation policy that can be applied in virtualized environment. The behaviour of the system is represented through a Stochastic Petri Net (SPN) model. Numerical analysis of the system availability is carried out the SHARPE tool simulation.

**Keywords**: Apache Server 2.2.10; Prediction; Stochastic Rewarded Net; Software Rejuvenation; Virtualization

## 1. INTRODUCTION

In long running client-server, software faults are the major cause of computer system failure. Several recent studies have established that most system outages are due to software faults.

Software rejuvenation is a technique for software fault tolerance which consists of stopping the executing software, 'cleaning' the 'internal state' and restarting. So, we compose virtualization technology for solving to protect software aging.

Virtualization technology, which allows multiple operating systems to run different applications on a single computer, has caught the attention of IT managers for its promise to let them better manage and utilize corporate IT resources. Moreover, availability is one of the key characteristics of virtualized data center.

In figure (1) and (2) show architectures of two hosts systems. Figure (1) presents non-virtualized system whereas Figure (2) shows virtualized two hosts system. The virtualized system consists of two physical virtualized servers in which each host has one Virtual Machine running on the Virtual Machine Monitor in the host.

In this paper, we describe to solve software aging using predicting method based on software rejuvenation and virtualization technology. We mainly emphasize the healing of software aging of web server in single physical machine and hardware failure is not considered.

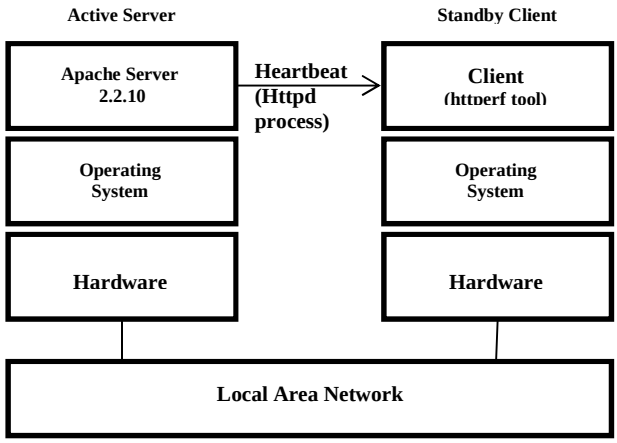


**Figure 1. Architecture of Non-virtualized System**

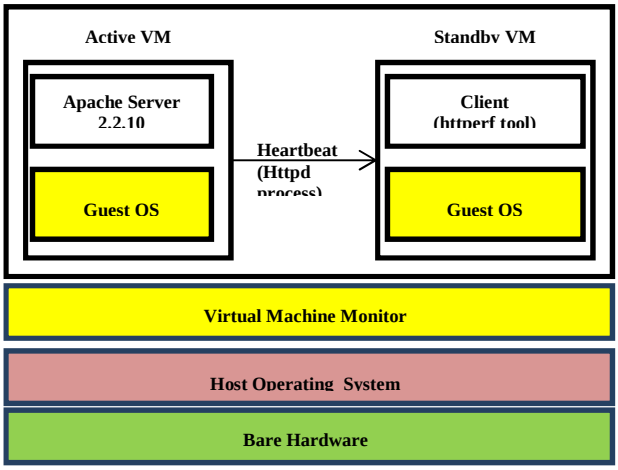


**Figure 2. Architecture of Virtualized System**

Rest of the paper is organized as follows. Related research is highlighted in Section II. We explain the proposed system using Apache Server on Linux in Section III. We present the experimental results using Httperf and Weka 3.5.8 in Section IV. We conclude this work and discuss future avenues of research in Section V.

## 2. RELATED WORKS

In other VMs, we will setup a SRA (Software Rejuvenation Agent) that is responsible for rejuvenation operation. Using measurement based modeling; we predict when two physical machines are used to four virtual machines.

Y. Huang et. al. [5] has suggested a complimentary technique which is preventive in nature. It involves periodic maintenance of the software so as to prevent crash failures. They call it Software Rejuvenation, and define it as the periodic preemptive rollback of continuously running applications to prevent failures. While monitoring real applications, it was observed that software typically "ages" as it is run. Potential fault conditions are thus slowly accumulated since the beginning of the software activity.

D. Seong Kim et. al. [2] construct non-virtualized and virtualized two hosts system models using a two-level hierarchical approach in which fault trees are used in the upper level and homogeneous Continuous Time Markov chains (CTMC) are used to represent sub models in lower level. Then, they incorporate not only hardware failures (e.g., CPU, memory, power, etc) but also software failures including VMM, VM, and application failures. To improve high availability (HA) service and VM live migration in the virtualized system and use metrics according to system steady state availability, downtime in minutes per year and capacity oriented availability.

Grottke et. al.[4] studies the development of resource usage in a web server while subjecting it to an artificial workload. They only focus on software rejuvenation using Apache based on Linux. They did not consider virtualization technology. Moreover, they show how the exploitation of the seasonal variation can help in adequately predicting the future resource usage. They apply such as proactive management techniques like software rejuvenation triggered by actual measurements can be built.

In this paper, we promote virtualization technology upon the operating system. We also focus on software rejuvenation for software aging using Apache web server based on Linux. We consider not only software rejuvenation but also virtualization technology.

## 3. PROPOSED SYSTEMS

In this section, our proposed system consists of two main parts. Firstly, we created the Apache Server and applied httpd process using Httperf tool for Software Aging. Moreover, we focus on three Machine Learning algorithms predicting Web Server for Software Aging in Section (3.1). Secondly, we presented a SRN model for load and time base Software rejuvenation policy in a single physical server as well as Apache Server with a preventive maintenance in Section (3.2)

### 3.1. Time-based Software Aging Prediction Model based on Machine Learning

In this section, we predicted the Time to Crash (TTC) due to resource exhaustion causing the software aging based on limited set of system metrics (CPU, Memory, etc).

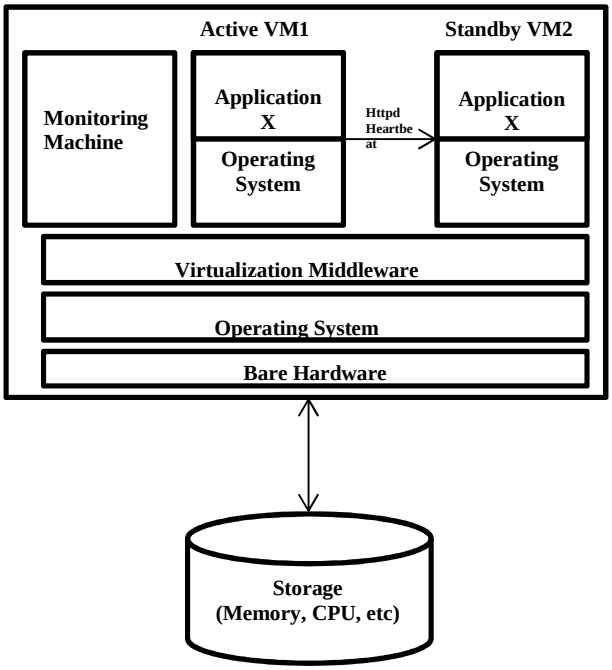


**Figure 3. Proposed System Architecture**

Our proposed system consists of server and client running Apache on a Linux platform on the top of Virtualization Middle ware in Figure 3. On the top of the virtualization middle layer, we create virtualized two hosts system. The virtualized system consists of two physical machines in which each host has two virtual machines running on the Virtual Machine Monitor

in the host. Two virtualized hosts share a common storage (CPU, Memory, etc).

Firstly, we have collected data from Apache server to predict for software aging caused by resource exhaustion in Figure (3). Apache server is one of the most popular used web server systems, and discovered that the Apache server shows evidence of aging after inserting memory leak into the web site running in it. Moreover, httperf is a tool for measuring web server performance. It provides a flexible facility for generating various HTTP workloads and for measuring server performance [8].

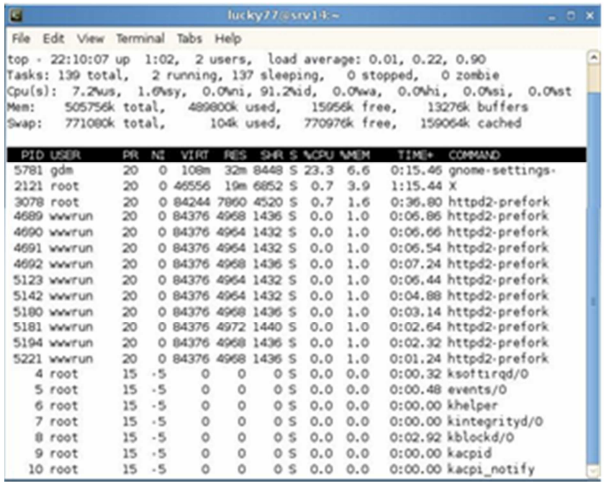


**Figure 4. Httpd processes due to Software Aging**

According to Figure (4), software aging has occurred 3078 PID and connection rates are used about 27000.We have used our prediction algorithm because it follows our prediction presented earlier: while a global behavior of software aging may be highly non-linear, it may be composed of a number of linear patches. Specially, we have used an implementation of Linear Regression, Decision Stump and M5P extracted from the WEKA distribution.

*3.1.1 Linear regression*

A Linear regression of variables $x_1$, …..,$x_n$ is a function of the form

$$\alpha_0 + \sum_{i=1}^{n} \alpha_i . x_i \qquad (1)$$

For some set of coefficients $\alpha_0$ , ... , $\alpha_n$

It will work well to predict some variable y when (and only when) y depends linearly on the $x_i$ variable. The α coefficients are calculated from the training data, trying to minimize the sum of the squares of these differences over all the training instances.

It is an efficient, simple method for numeric prediction and widely used today to help for capacity planning analysis or detecting anomalies on the system. Disadvantage is relying too much on linearity.

*3.1.2 DecisionStump*

A decision stump is a machine learning model consisting of a one-level decision tree. That is, it is a decision tree with one internal node (the root) which is immediately connected to the terminal nodes (its leaves). A decision stump makes a prediction based on the value of just a single input feature. Sometimes they are also called 1-rules.

Depending on the type of the input feature, several variations are possible. For nominal features, one may build a stump which contains a leaf for each possible feature value or a stump with the two leaves, one of which corresponds to some chosen category, and the other leaf to all the other categories. For binary features these two schemes are identical. A missing value may be treated as a yet another category.

*3.1.3 M5P*

The M5P or M5Prime algorithm [Wang & Witten, 1997] is a regression-based decision tree algorithm, based on the M5 algorithm by Quinlan [1992]. M5P is developed using M5 with some additions made to it. We will first describe the M5 algorithm and then the features added to it in M5P [1].

M5P builds a tree to predict numeric values for a given instance. The algorithm requires the output attribute to be numeric while the input attributes can be either discrete or continuous. Each leaf has a linear regression model associated with it in Equation (1). As the leaf nodes contain a linear regression model to obtain the predicted output, the tree is called a model tree.

## 3.2 Load and time based Software Rejuvenation Policy using Stochastic Petri Net Model

In Figure (5), we present a SRN model for rejuvenation policy in a single physical server with a preventive maintenance.

In our case, it also models the degraded service suffered by the server due to software aging using virtualization technology. It consists of active-standby virtual machine servers. Regarding the server model, the free capacity of

the server is modeled with the place $P_k$ and $P_{k1}$. The transition $T_{arr}$ and $T_{arr1}$ model the arrival of jobs. The place $P_{L2}$ and $P_{L1}$ model the number of jobs enqueued in the server. The transition $T_{serv}$ and $T_{serv1}$ represent the service event. The service rate of the server depends on the age of the system since the last rejuvenation, which is modeled with a degradation factor related to the place $P_{fp2}$ and $P_{fp1}$. The transition Tempty and Tempty1: which are enabled when the place $P_{R1\_2}$, $P_{R2\_2}$, $P_{R1\_1}$ and $P_{R2\_1}$ or the place $P_{Down}$ contain a token.

Both VMs are "healthy" working states, indicated by a token in place $P_{up2}$. As time progresses, active VM eventually transits to failure probably states in place $P_{fp2}$ and $P_{fp1}$ through the transition $T_{fp2}$ and $T_{fp1}$. The VM is still operation in this state. At that time, VM can be entering the failure state (Place $P_{D2}$) through the firing transition $T_{down..}$ But VM can be switch over to another standby VM. When transition $T_{sw}$ is enabled, the operation of active VM is switched to standby VM and a token is moved to a place $P_{up1}$. After that operation will be restarted on standby VM. On the other hand, the VM can be rejuvenation.

**System notations**

| | |
|---|---|
| Pupi | : VMs are in 'robust' state |
| PRj | : Rejuvenation State |
| Pfprob | : Failure Probably State |
| PD2 | : Failure Probably State |
| Pfpi | : Failure Probably State |
| PDown | : Failure State |

Where i=1,2, j = 1-2,2-2,1-1,2-1

## 4. EXPERIMENTAL RESULTS

In this section, we express the two experimental results. One is prediction using Linear Regression. Decision Stump and M5P from machine learning algorithms to predict the system crash due to software aging caused by resource exhaustion. Another is availability analysis using Stochastic Petri net model to solve software aging.

### 4.1 Prediction's Result

In Table (1), we calculated prediction for software aging using Linear Regression, DecisionStump and M5P algorithms from Machine Learning. Linear Regression will work well to predict some variable y when (and only when) y depends linearly on the $x_i$ variable. The α coefficients are calculated from the training data, this process got the correlation coefficient (-1) and Mean Absolute Error (2.4889 second) from Linear Regression and M5P algorithms.

According to experimental results, Linear Regression and M5P's MAE is less than DecisionStump's MAE.

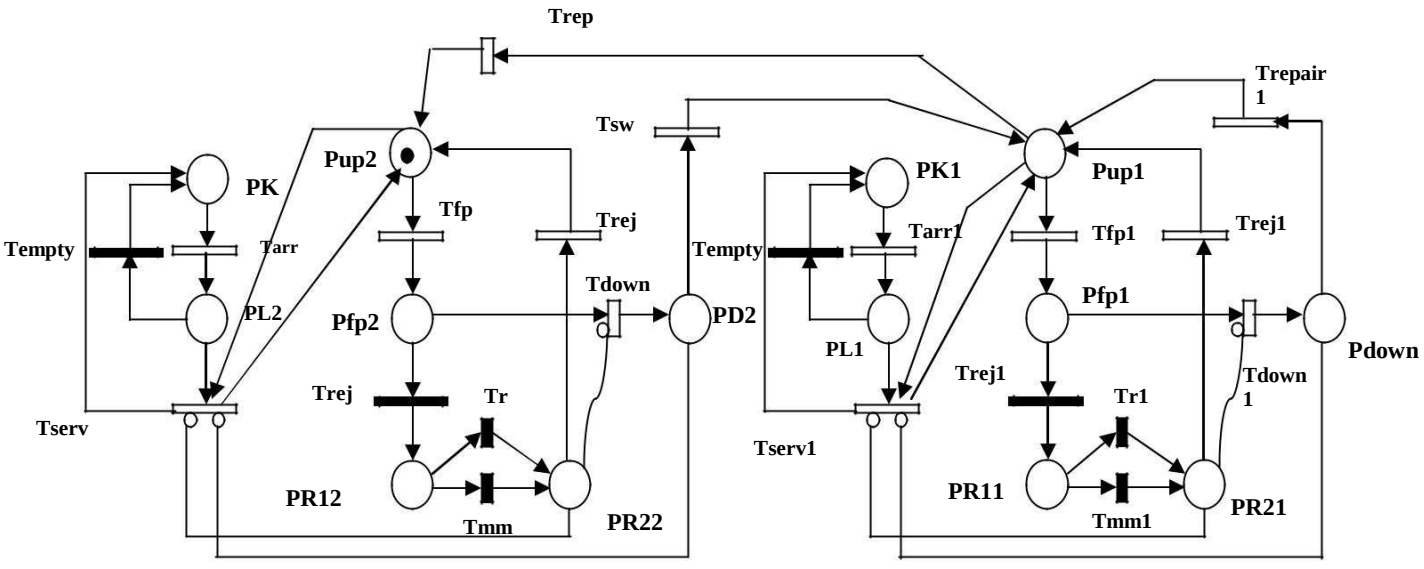


**Figure 5. Load and time based software rejuvenation policy using Stochastic Petri Net Model**

**Table 1. Results of Prediction Algorithm**

| Prediction Algorithm | Correlation Coefficient | Mean Absolute Error |
|---|---|---|
| Linear Regression | -1 | 2.4889 second |
| DecisionStump | 0 | 4966.8 second |
| M5P | -1 | 2.4889 second |

In this paper, we focused on prediction for software aging due to resource exhaustion such as memory and response time. A set of Machine Learning Algorithms has been analyzed for predicting system crashes due to the resource exhaustion caused by software aging [3]. So, we measured Apache web server performance in such a system it is necessary to run a tool on the clients that generates a specific HTTP workload. To convey a concrete feeling of how httperf is used, this section presents a brief example of how to measure the request throughput of a web server.

### 4.2 Availability's Result

The availability of the proposed model is defined as

Availability = 1-Unavailability (2)

Availability = $1 - P_{DOWN}$ (3)

The expected total downtime with preventive maintenance in an interval of T time units is

Downtime (T) = $P_{DOWN}$ x T (4)

**Table 2. System Operating Parameters**

| Transition Parameter | Firing Rates (per hour) |
|---|---|
| Tserv | 1 time/1 hour |
| Tfail | 3 times/1 month |
| Tfprob | 1 time/a day |
| Thw | 1 time/$10^6$ |
| Trepair | 2 times/a week |
| 1/Trej1, 1/Trej2 | 5 mins |
| Tclock1, Tclock2, Tmm, Tmm1, Tarr, Tarr1 | Variable |

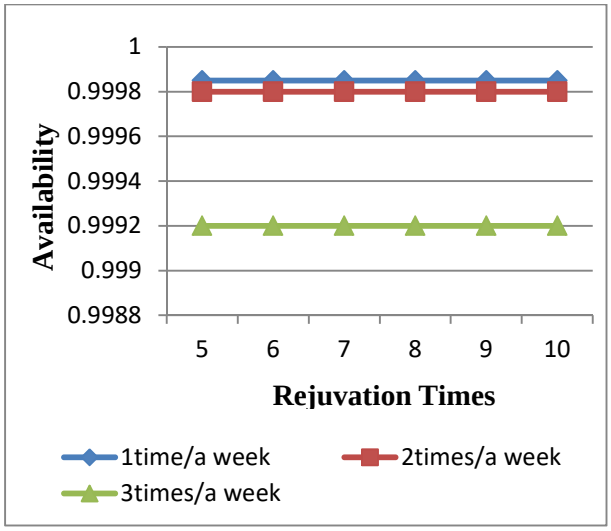


**Figure 6. Availability Vs Different rejuvenation interval and repair rates**

The availability and different rejuvenation interval and repair rates are resulted in Figure (6). We focus on different rejuvenation interval and repair rates. As a matter of fact, 1 time/a week are more availability than 2 times/ a week and 3 times/ a week.

## 5. CONCLUSIONS

In this paper, we show availability analysis and predicting web server for software aging using virtualization technology. We focus on prediction to solve software aging using a set of Machine Learning Algorithms such as Linear regression, DecisionStump and M5P. Firstly, we create Apache Server using Linux. And then data have collected as well as response time and memory usage from this web server. Secondly, we present load and time dependent software rejuvenation policy that can be applied in virtualized environment. The behavior of the system is represented through a Stochastic Petri Net (SPN) model. Numerical analysis of the system availability is carried out the SHARPE tool simulation.

## 6. ACKNOWLEDGEMENTS

I would like to thank my teachers Rector, Dr.Mie Mie Thet Thwin, University of Computer Studies (Yangon), Rector, Dr. Thandar Thein, University of Computer Studies (Maubin), other teachers and my parents during my research for their encouragement, knowledge sharing and supporting.

## 7. REFERENCES


[1] A. Polumetla, " Machine Learning Methods for the Detection of RWIS Sensor Malfunctions", Master Thesis, Pages 30-33, July 2006.

[2] D. Seong Kim, F. Machida and K. S. Trivedi, "Availability Modeling and Analysis of a Virtualized System", 15th IEEE Pacific Rim International Symposium on Dependable Computing, Pages 365-371, 2009.

[3] J.Alonso, L.Belanche and D.R.Avresky, "Predicting Software Anomalies using Machine Learning Techniques",2011.

[4] M.Grottke, L. Li, K. Vaidyanathan and K. S. Trivedi, "Analysis of Software Aging in a Web Server", IEEE Transactions on Reliability, 55:441-420, 2006.

[5] Y. Haung, C. Kintala, N. Kolettis, and N. Fulton, "Software rejuvenation : Analysis, module and applications", In Proc . Twenty-fifth International Symposium on Fault-Tolerant Computing, Pages 381-390, 1995.

[6] DecisionStump http://en.wikipedia.org/wiki/Decision_stump

[7] WEKA 3.5.8. WEKA http://www.cs.waikato.ac.nz/ml/weka/

[8] Httperf tool http://http-performance-testing-with-httperf.html ftp://ftp.hpl.hp.com/pub/httperf