iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 6 **Issue:** IV **Month of publication:** April 2018

DOI: http://doi.org/10.22214/ijraset.2018.4275

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887
Volume 6 Issue IV, April 2018- Available at www.ijraset.com

Securing KVM Based Virtual Machine Monitoring System for Handling and Detection of Rootkit Attack

Vijay Pardeshi[1], Gauri Kasat[2], Vishesh Neve[3], Ms. Pallavi Baviskar[4]
[1,2,3] *Department of Computer Engineering, P.E.S Modern College of Engineering, Pune*
[4] *Assistant Professor, Department of Computer Engineering, P.E.S Modern College of Engineering, Pune*

Abstract: ***Virtual machine introspection (VMI) is the mechanism which allows indirect inspection and management of the status of virtual machines. Indirection of this approach offers attractive isolation properties which have resulted in a variety of VMI-based applications for security, performance, and debugging in virtual machine environments. This gives the access to the virtual machine monitor, Virtual Machine Introspection functionality is unfortunately not available to cloud users on public cloud platforms.***
We present our work on the CloudVMI architecture to address about this concern. CloudVMI virtualizes VMI interface and makes this available as-a-service in cloud environment. Because this allows introspection of users' VMs running on arbitrary physical machines in cloud environment, VMI-as-a-service abstraction allows a new class of cloud-centric VMI applications to be developed.
A rootkit recognition scheme that takes cloud computing environments into contemplation. This method utilizes vIPS platform to gain many useful traits including hypervisor-independency, agent less virtual security appliance structure, and most importantly usability. Our system will monitor VM statically and dynamically with the help of SHA-512 and Feed forward neural network respectively. This runtime status of the VM will be monitored with the help of QEMU hypervisor. Thus the method provides effective protection against rootkit in cloud computing environment.
Keywords : Rootkit Attack, Virtual Machine Introspection (VMI), CloudVMI, QEMU, Feed Forward Neural Network algorithm, Secure Hash Algorithm-512.

I. INTRODUCTION

Virtual machine introspection is the technique used to inspect and analyze code running on a virtual machine. Virtual machine introspection has got significant attention in computer security research area. Virtual Machines are the prime target for adversary to take control by exploiting the identified vulnerability present in Virtual Machine Environment. This increasing number of advanced attacks like malware, rootkit, etc., virtual machine protection is great challenging task. The key element of this advanced threat is rootkit that provides hidden control of underlining Operating System kernel. Protecting individual operating system by antivirus and commercial security defense mechanism is cost-effective and ineffective in a virtualized environment.
Virtual Machine Introspection has been emerged as one of the promising approaches to a secure state of the virtual machine. Virtual Machine Introspection inspects state of the multiple virtual machines by operating outside the virtual machine i.e. at hypervisor level. It is a fine-grained approach. Isolation property of the VMI allows inspecting Memory, Disk, CPU registers, Network connections and other hardware events of virtual machine during run state at hypervisor level. Securing virtual machine by using VMI is an active research area.

II. RELATED WORK

The paper “Virtual Machine Introspection”, lets users have "one machine, multiple operating systems, multiple applications" and switch between them at will. This not only lets developers easily test their programs on multiple OSs and enterprise users more effectively utilize hardware through server consolidation, it's also useful to computer users in general. When virtual machines are distributed with a set of preconfigured applications, users can easily utilize complex applications. Further, the isolation offered by VMs provides some security benefit, such as allowing general Web browsing while reducing the risk of compromise to the underlying physical system. The recent development of x86 virtualization products has revived interest in the virtualization market. This has led to the evolution of virtual machine introspection (VMI) techniques and tools to monitor VM behavior. VMI tools inspect a VM from the outside to assess what's happening on the inside. 1 This makes it possible for security tools—such as virus

scanners and intrusion detection systems—to observe and respond to VM events from a "safe" location outside the monitored machine. Here, we survey and categorize the current crop of VMI technologies, then offer a detailed description of the Virtual Introspection for Xen (VIX) tool suite, which addresses key VMI requirements.[1]

The paper "Nitro Hardware-based System Call Tracing for Virtual Machines", gives VMI framework, Nitro, for hardware-based system call tracing and monitoring. Nitro is the first VMI-based system that supports all three system call mechanisms provided by the Intel x86-architecture and has been proven to work for Windows, Linux, 32-bit, and 64-bit guests. This framework is flexible enough to feasibly support almost any OS built upon the x86-architecture. [2]

Paper "Leveraging Derivative Virtual Machine Introspection Methods for Security Applications", describes malware detection that makes use of a support vector machine (SVM) to classify system call traces. It uses system call traces for malware detection, a string kernel to make better use of the sequential information inherent in a system call trace. By classifying system call traces in small sections and keeping a moving average over the probability estimates produced by the SVM, This is capable of detecting malicious behaviour online and achieves a very high accuracy. [3]

This paper "Preventing Cache-Based Side-Channel Attacks in a Cloud Environment", investigate the usage of CPU-cache based side-channels in the Cloud and how they compare to traditional side-channel attacks. The Cache Flushing Technique is necessary to mitigate these sorts of attacks in a Cloud environment and implementing them in a state-of-the-art Cloud system, and testing them against traditional Cloud technology. [4]

The paper "Detection of Malware and Kernel-level Rootkits in Cloud Computing Environments", presents a narrative malware and rookit detection system which protects the guests against different attacks. It combines system call monitoring and system call hashing on the guest kernel together with Support Vector Machines (SVM)-based external monitoring on the host. It gives solution by evaluating attacks against well-known user-level malware as well as kernel-level rootkit attacks. [5]

"CloudMon: Monitoring Virtual Machines in Clouds", gives dynamic framework CloudMon to detect kernel rootkits and guarantee the runtime security of guest VMs. CloudMon is transparent to a guest VM, neither requires its specific system information, nor has to one-on-one run with it. Meanwhile, CloudMon detects kernel rootkits through self-adjusting monitoring on memory with an acceptable overhead. CloudMon is implemented based on Xen. CloudMon is effective to detect kernel rootkits in guest VMs, while the performance experiments demonstrate that it brings a low performance overhead. [6]

"An Intelligent Virtual Machine Monitoring System Using KVM for Reliable and Secure Environment in Cloud", monitors the virtual machines under KVM hypervisor. The KVM is a type II hypervisor that is Linux kernel based virtual machine manager which comes with Linux OS. The VMs are the target for the malicious or abnormal attacks. To protect the VMs from the attack system uses the VM monitoring script to get the status of the VMs. [7]

III.SYSTEM ARCHITECTURE

A. Problem Statement

Securing KVM based virtual machine monitoring system for handling and detection of rootkit attack.

1) To protect virtual machine from attacks like rootkit.
2) To add intelligence in virtual introspection through pattern recognition and neural processing of data.
3) To enhance the security and efficiency of hypervisor and virtual machine in cloud environment.

The architecture demonstrates the virtual machine introspection system running in cloud environment. The hypervisor (QeMU), the virtual machine monitor, provides all resources of the host machine to all the VMs through the virtualized environment. Type-1 hypervisor runs directly on the system hardware. Type-1 hypervisor is provides efficient integration of it into firmware. The fig. 1 shows the overall system architecture of the virtual machine introspection system. The proposed framework has following parts:

B. Virtual Machine

Virtual machine are a reflection of the actual computer architecture which provides all the functionality of the physical system. It runs in a cloud environment. The role of hypervisor is to protect the virtual machines from attacks that can cause real damage to the cloud environment. Hypervisor should provide a security mechanism to the VM users, by detecting whether the VM was attacked or not. Since the use of KVM, it allows to run multiple heterogeneous VM. The attack on VM cloud environment can take the following ways:

1) Attack on the VM from another VM outside the cloud environment.
2) Attack on the VM from the VM in same cloud environment.

3) Attack on the Hypervisor itself.

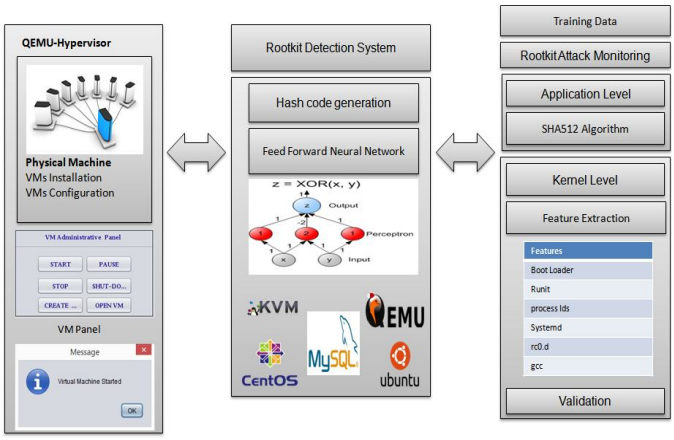


Fig 1:- Architecture diagram of proposed system

C. Hypervisor

The cloud environment consist of several different VMs running. A hypervisor manages all the VMs. It allows us to host several different VMs on single hardware environment. QeMU stands for Quick Emulator. It is a free and open-source hypervisor which provides hardware virtualization. It can be used with KVM to provide additional functionality. KVM provides emulation of hardware. KVM is a Linux kernel module that enables mapping of physical CPU to virtual CPU. This mapping provides hardware acceleration and boosts the performance of the VM. KVM is integrated in QeMU for providing full hypervisor functionality. When QeMU uses KVM, this combination becomes a Type-1 Hypervisor. KVM is a loadable kernel module (kvm.ko), which provides processor specific module.

The proposed system has following parts at hypervisor level:

1) VM monitoring open source tool (QeMU): Continuously monitors the status of VMs form the hypervisor level.
2) Host OS: The host OS is underlying physical OS on which the hypervisor is running.
3) Host Hardware: It is the physical resources attached to particular machine in cloud infrastructure.

IV. ATTACK MODEL

A. Rootkit Attack

Rootkit is collection of malicious software's that allows to access computer that is not otherwise allowed (Example, unauthorized user). The word rootkit attack is combination of word "root" (the account on unix-like os) and the word "kit" (software component that implements the tool). In this type of attack there is direct attack on the system. Once the rootkit is installed it becomes possible to hide the intrusion on the system and manipulate the system without the user's acknowledgement.

The proposed system has two types of rootkit attack:

1) *User Mode:* In this the rootkit works with other applications of user. It modifies the behavior of the application performance.
2) *Kernel Mode:* Kernel-mode runs with the highest privileges of operating system. It modifies the code of the operating system by adding code or replacing code of the operating system. In this type of attack the system cannot be trusted in any view.

V. MATHEMATICAL MODEL

A. Input

VM= {VM_1, VM_2, VM_3, …….. ,VM_n} No. of Virtual Machine in Cloud.

B. Processing

VMI tool= {VM status, System-level process tracing, user-level application monitoring}
SHA-512 algorithm= {VM application status monitoring}
Feed Forward Neural Network algorithm= {VM system process status monitoring}

C. Output

VMTargeted= {VM_1, VM_2, VM_3, …….. ,VM_n} No. of targeted Virtual Machine in Cloud.

VI. EXPERIMENTAL RESULT

1) The fig.2 shows the GUI for the VM information.
2) The GUI will show all the VM list along with all the details such as VM name, whether its running or paused, its IP address and its status whether its attacked or normal.
3) Static checking will monitor system statically. It will check whether the user application are secure or not. It will match the SHA signature and show whether the system is attacked or not.
4) Dynamic checking will monitor the system dynamically. It will check all the system-level processes. It monitors at kernel level. It will show if any of the module of process is changed or not.
5) System logs will show all the system log details.
6) VM status information will show all the detail of the VM.

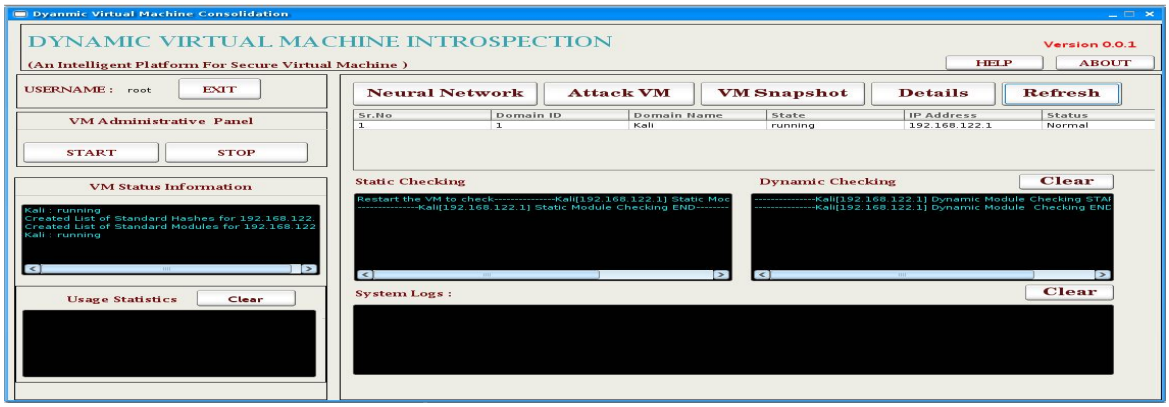


Fig. 2:- Virtual machine details

7) Fig. 3 shows the neural network details.
8) Its shows the list of VMs, along with their state and IP address.
9) The input layer of neural network consist of 5 inputs, those are change in SHA, change in WTMP, change in UTMP, changes in modules, and change in boot process.
10) The neural network has been trained by providing different training data set.
11) Using the training data set the neural network is trained using back-propagation the neural network reduces the error rate of the output.
12) Depending on the parameters value provided the neural network will tell whether there is attack or not.

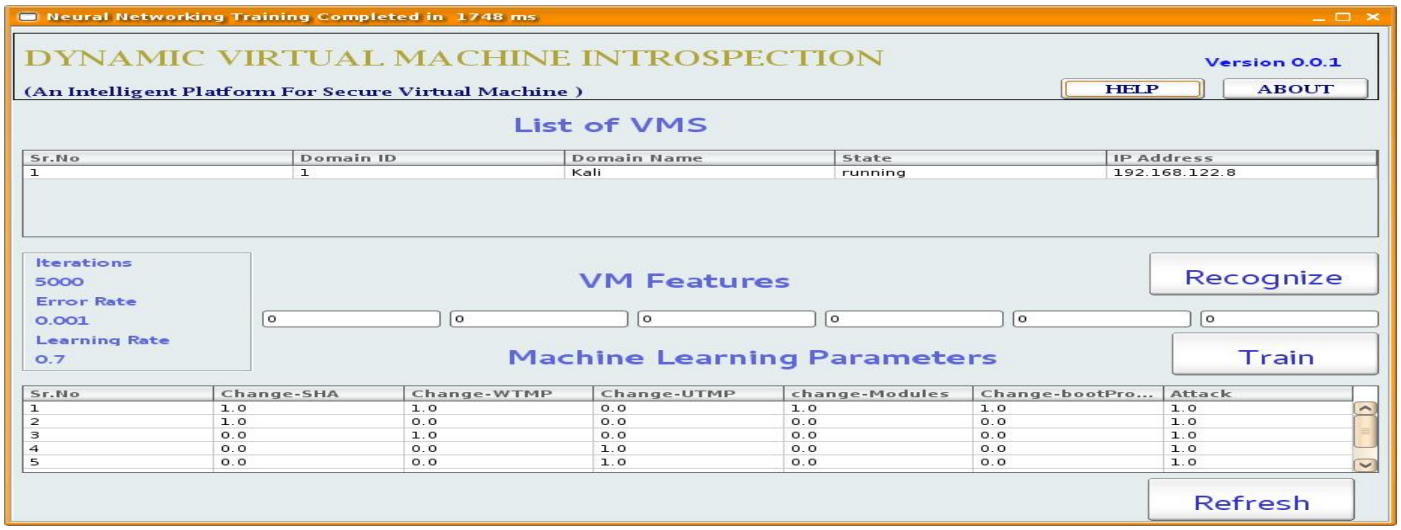


Fig. 3:- Neural Network

VII. CONCLUSION

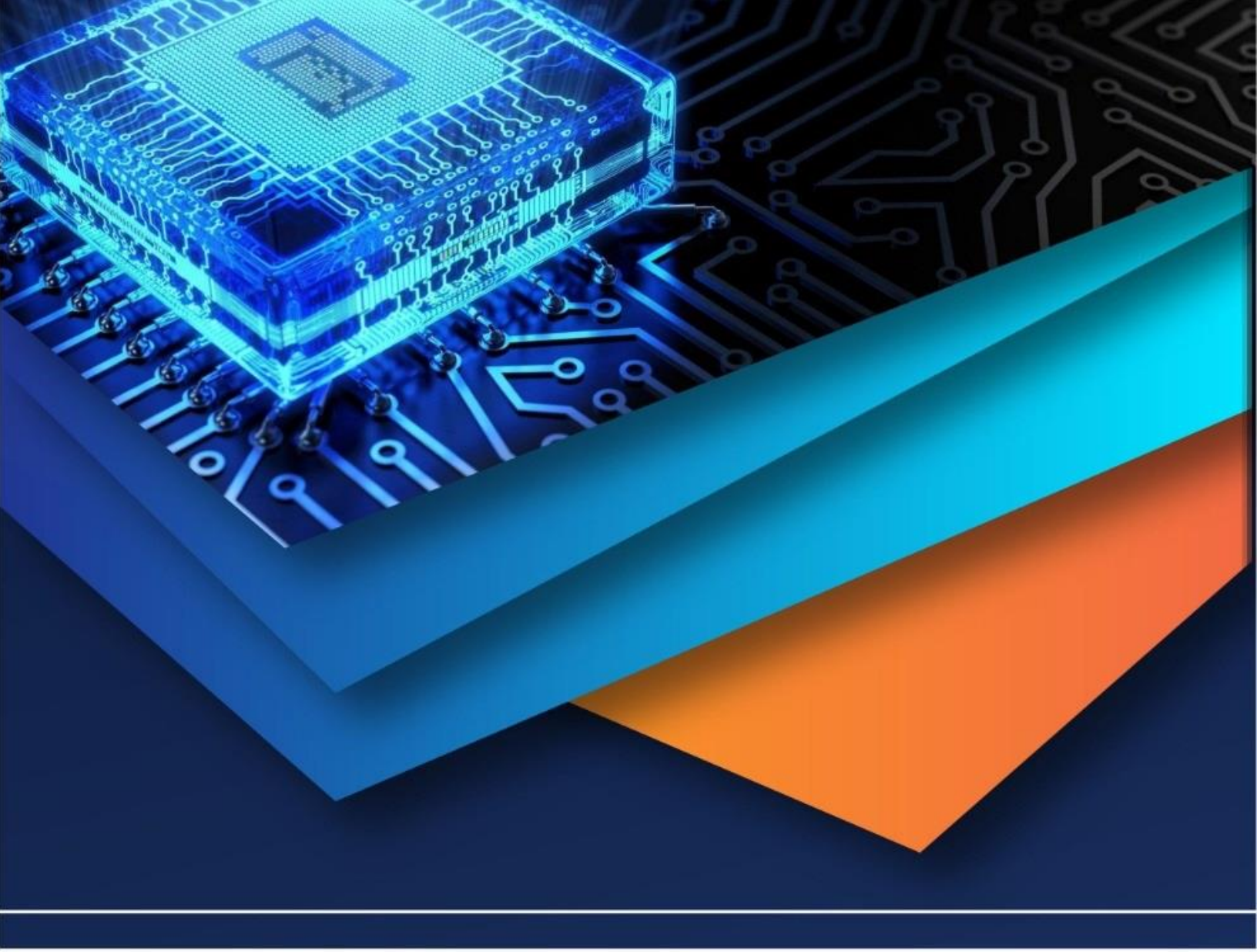
We propose a paper name for cloud platform. The system will monitor static and dynamic virtual machine status using qemu (kvm) tool. It identifies the system and user level processes by applying the sha and neural network algorithm. This system provides protection based on the trained neural network. In this the number of vm can be increased and more attacks can be added in the future. The attacked system can be stopped or migrated or pause the vm.

VIII. ACKNOWLEDGMENT

We are thankful to Computer Engineering department, P.E.S Modern College of Engineering, Pune for the support , facilities and guidance provided for the work undertaken.

REFERENCES

[1] Kara, Nance, Matt Bishop and Brian Hay, “Virtual Machine Introspection: Observation or interference?”, IEEE Security & Privacy, 5: 32-37, 2008.
[2] Pfoh, Jonas, Christian Schneider and Claudia Eckert, “Nitro: Hardware-based system call tracing for virtual machine. Advances in Information and Computer Security” , Springer Berlin Heidelberg: 96-112, 2011.
[3] Jonas Pfoh, “Leveraging Derivative Virtual Machine Introspection Methods for Security Applications”, IEEE, 2013.
[4] Michael Godfrey, Mohammad Zulkernine, “Preventing Cache-Based Side-Channel Attacks in a Cloud Environment”, IEEE Transactions on Cloud Computing, 2013.
[5] Thu Yein Win, Huaglory Tianfield, Quentin Mair, “Detection of Malware and Kernel-level Rootkits in Cloud Computing Environments”, IEEE 2nd International Conference on Cyber Security and Cloud Computing, 2015.
[6] Chuliang Weng, Qian Liu, Kenli Li, and Deqing Zou, “CloudMon: Monitoring Virtual Machines in Clouds”, IEEE Transaction, 2016.
[7] Mrs. Swarupa Mahesh Deshpande, Prof. Mrs. Bharati Ainapure, “An Intelligent Virtual Machine Monitoring System Using KVM for Reliable and Secure Environment in Cloud”, IEEE International Conference on Advances in Electronics, Communication and Computer Technology (ICAECCT) Rajarshi Shahu College of Engineering, Pune India. Dec 2-3, 2016.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY