Secured Program Evaluation System (SPES)

Harshal Chepey[1], Rohit Deodhar[2], Sachin Kolhe[3], Shailesh Patil[4], Mamta Bhamare[5]
Pune, India

[1]harshalchepey@gmail.com
[2]r.deo91@gmail.com
[3]sachukolhe@gmail.com
[4]92patilshailesh@gmail.com
[5]mamta.bhamare@mitpune.edu.in

Abstract—**This paper focuses on a system which evaluates the untrusted third party source code submitted online by user ensuring security of server. The need for this kind of system arises because of some flaws in the existing systems (e.g. Reliscore.com). In the existing system there is no provision of reporting output/error back to the user and hence we aim at extending the functionality of reporting the output back to the user. SPES accepts program code written by the user/client and executes program in a controlled environment. We are going to use 'Virtualization' approach, creating virtual machine and virtual box for securing the server. If the program code is malicious, then it is flagged as malicious and error message is sent to user. If any of the source code contains any malicious code then it can only alter with the settings of the virtual machine and hence our server will remain secure in a way. A malicious code can be code that demands for huge amount of memory, accesses/deletes/modifies system files, try to connect to internet. Such kind of codes is not allowed to execute on system.**

Keywords—**Virtualization, SPES, hypervisor, Virtual Machine Monitor, Mount Point, Checkpoint**

I. Introduction

Internet has evolved into more interactive, vivid, productive, and has a potential to influence our lives tremendously. Many software development tools are available online, which can be used in various stages in software development. One of them is compiler, which allows users to compile and run programs written in different languages online. The advantage of using online compilers[1] is that users need not have compilers installed on their computers]. But if program containing some malicious code gets executed on server, then it could harm server. Hackers can use malicious programs to attack server by running them on server. Attacks include denial of service (DOS), backdoors, root kits, stealing of important information. Hence it is necessary to execute programs on server in a controlled manner such that programs have restricted privileges. Our system executes programs in a controlled and monitored environment. Program codes submitted by users are compiled and executed on our server in a virtual machine. Virtual machines are isolated from their environment, so that if a program is malicious, its effect is seen only on the virtual machine that executes it. Thereby protecting server from attacks.

II. Proposed Solutions

1) FIREWALL

A firewall can either be software-based or hardware-based and is used to help keep a network secure. Its primary objective is to control the incoming and outgoing network traffic by analysing the data packets and determining whether it should be allowed through or not, based on a predetermined rule-set. A network's firewall builds a bridge between an internal network that is assumed to be secure and trusted, and another network, usually an external (inter)network, such as the Internet, that is not assumed to be secure and trusted. Many personal computer operating systems include software-based firewalls to protect against threats from the public Internet. Many routers that pass data between networks contain firewall components and, conversely, many firewalls can perform basic routing functions. There are generations of firewall such as packet filters, stateful filter, and application layer firewall.

Application firewalls function by determining whether a process should accept any given connection. Application firewalls accomplish their function by hooking into socket calls to filter the connections between the application layer and the lower layers of the OSI model. Also, application firewalls further filter connections by examining the process ID of data packets against a rule-set for the local process involved in the data transmission. The extent of the filtering that occurs is defined by the provided rule-set. Given the variety of software that exists, application firewalls only have more complex rule-sets for the standard services, such as sharing services. These per process rule-sets have limited efficacy in filtering every possible association that may occur with other processes. . Because of these limitations, application firewalls are beginning to be supplanted by a new generation of application firewalls that rely on mandatory access control (MAC), also referred to as sandboxing, to protect vulnerable services. Firewall was considered one of the proposed solutions to implement SPES, but, due to some disadvantages of firewall it is discarded. Disadvantages of firewall are that they cannot handle e-mail viruses and phishing scams and also they cost much more than other options. Also, firewall has single point of failure in the system.

2) SANDBOX

A sandbox is a testing environment that isolates untested code changes and outright experimentation from the production environment or repository, in the context of software development. Sandboxing protects "live" servers and their data, vetted source code distributions from changes that could be damaging regardless of the intent of the author of those changes to system or which could simply be difficult to revert.

In computer security, a sandbox is a security mechanism for safely running programs. It is often used to execute untested code, or programs from unverified third-parties, suppliers and untrusted users. The sandbox typically provides a tightly-controlled set of resources for guest programs to run in, such as scratch space on disk and memory. Network access, the ability to inspect the host system or read from input devices is usually disallowed or heavily restricted. In this sense, sandboxes are a specific example of virtualization. For SPES, we need sandbox software which can handle the testing of untrusted third party source code efficiently. Unfortunately, some recent malwares have anti analysis mechanisms for examining the environment in which they are being executed [3]. Therefore, such malwares cannot be efficiently analyzed in isolated sandboxes and virtualized environments because they are capable of detecting analyzing environments. Also, sandbox softwares are not freely available and also they were not compatible to our system, hence we ruled it out.

3) CHROOT

Systems exposed to the internet are heavily challenged to keep the bad guys out, and keeping up with the latest security patches is not always easy. So, the wise admin will attempt to institute systemic steps to limit the damage should, and one excellent method is the use of a chroot() jail. The chroot system call changes the root directory of the current and all child processes to the given path, and this is nearly always some restricted subdirectory below the real root of the file system. This new path is seen entirely as "/" by the process, and we refer to this restricted environment as the "jail". The chroot system call is found in all versions of UNIX that we know of, and it serves to create a temporary root directory for a running process, and it's a way of taking a limited hierarchy of a file system (say, /chroot/named) and making this the top of the directory tree as seen by the application.

A chroot is basically a special directory on your computer which prevents applications, if run from inside that directory, from accessing files outside the directory. In many ways, a chroot is like installing another operating system inside your existing operating system. In other words, chroot temporarily changes the root directory (which is normally /) to the chroot directory (for example, /var/chroot). As the root directory is the top of the file system hierarchy, applications are unable to access directories higher up than the root directory, and so are isolated from the rest of the system. This prevents applications inside the chroot from interfering with files elsewhere on your computer. For SPES system, chroot was quite a good solution due to advantages such as it is an open source and hence freely available and was able to fulfil almost everything what our system was expected to do. But, there are some serious disadvantages of chroot that we came to know when we went into the details of chroot. And, one of the disadvantages of chroot is that it is susceptible to “jail break” and through that, any user can access parent directory or any other partition. Due to this, it was potential to compromise the system. Hence, chroot was also discarded.

4) VIRTUALIZATION

A virtualized infrastructure can benefit companies and organizations of all sizes. Virtualization greatly simplifies a physical IT infrastructure to provide greater centralized management over your technology assets and better flexibility over the allocation of your computing resources. This enables your business to focus resources when and where they're needed most, without the limitations imposed by the traditional "one computer per box" model.

What does it all mean? In the computing realm, the term virtualization refers to presenting a single physical resource as many individual logical resources (such as platform virtualization), as well as making many physical resources appear to function as a singular logical unit (such as resource virtualization). A virtualized environment may include servers and storage units, network connectivity and appliances, virtualization software, management software, and user applications.

Basically, a virtual server, or VM, is an instance of some operating system platform running on any given configuration of server hardware, centrally managed by a virtual machine manager, or hypervisor, and consolidated management tools.

Note: The software providing the virtualization is called the VMM (virtual machine monitor)[2] or hypervisor. A hypervisor can run on bare hardware (native VM) or on top of an operating system (hosted VM).

A single instance may operate in isolation or share resources with several other instances of the same (or separate) server platforms.

In computing, a hypervisor or virtual machine manager (VMM) is a piece of computer software, firmware or hardware that creates and runs virtual machines.

A computer on which a hypervisor is running one or more virtual machines is a host machine. Each of those virtual machines is called a guest machine. The hypervisor presents to the guest operating systems a virtual operating platform and manages the execution of the guest operating systems. Multiple instances of a variety of operating systems may share the virtualized hardware resources.

III. Problem Description

Secure Program Evaluation system (SPES) is an application running on remote server. SPES accepts program code written by the user/client. SPES executes program in a controlled environment and returns the output of program to the user. If the program code is malicious, then it is flagged as malicious and error message is sent to user. Every time client submits a malicious program code, an entry is made by the system in its internal database.
A malicious code can be code that demands for huge amount of memory, accesses/deletes/modifies system files, try to connect to internet. Such kinds of codes are not allowed to execute on system. The system is expected to have a Web user interface for users to submit programs.

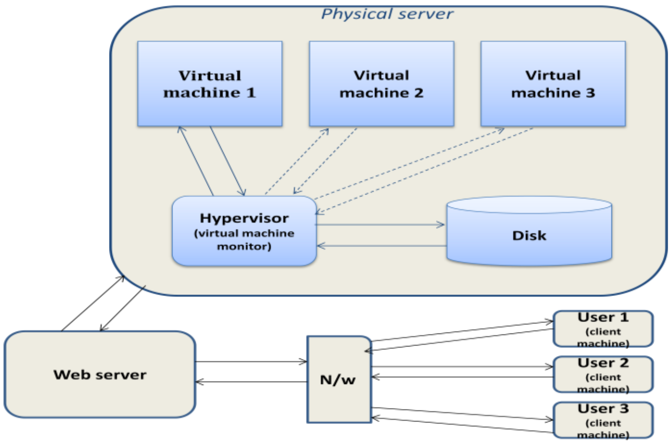

Fig 1. High Level Design

IV. CONTRIBUTION

This thesis contributes with implementing SPES, incorporating evaluating the source code with automated system for number of users to increase efficiency, time reduction and accuracy. It addresses the problems stated in previous section, i.e. to evaluate the code submitted by user in a controlled manner and by testing with sample set of programs which consists of simple code or potentially malicious code.

In order to reach its goal and gain acceptance, SPES has to prove that it can be worked in controlled manner and for different languages without breaking. The evaluation efficiency improvement must come without extensive overhead on the development time and budget. It is also vital that other aspects of SPES stay unaffected, such as scalability, maintainability, compatibility etc.

Another important goal of this work is to serve as a guide for implementing accessibility using SPES so that we can promote implementation by communities of developers working on similar frameworks, thus making users independent of the need of compiler on their own system.

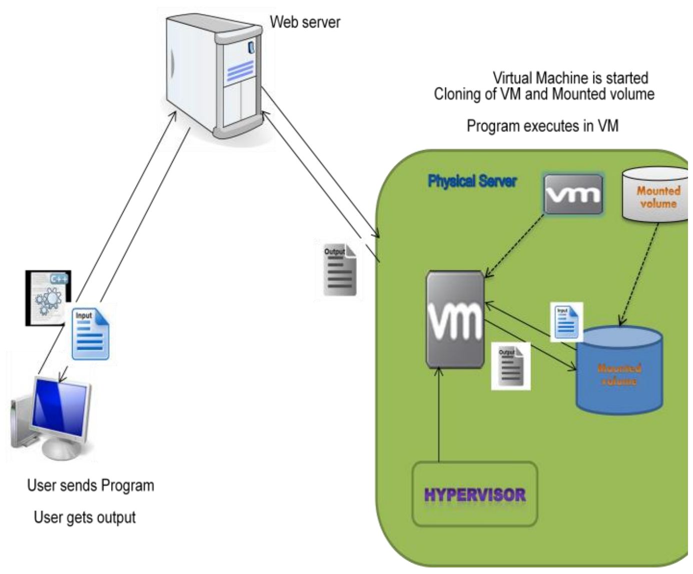

Fig. 2. Working of System

V. CONCLUSION

This paper proposed a new developed secure server system which will enable the evaluation of an untrusted third party program source code submitted online by user and provide the output or error information about program back to the user without any harm to the system or server by either crashing of server or data loss or data corruption.

ACKNOWLEDGEMENT

We are thankful to our college professors, family and friends for supporting and guiding us for the successful completion of this paper.

REFERENCES

[1] Aamir Nizam Ansari, Siddharth Patil, Arundhati Navada, Aditya Peshave, Venkatesh Borole “Online C/C++ compiler using cloud computing”- 978-1-61284-774-0 ,2011.

[2] Paul A. Karger IBM Corporation, Thomas J. Watson Research Center, “Is Your Virtual Machine Monitor Secure? “,978-0-7695-3363-6/08 2008 IEEE.

[3] Shinsuke Miwa, Oshiyuki Miyachi, Masashi Eto, Masashi Yoshizymi, Yoichi shinoda “Design and implementation of an isolated sandbox with mimetic intenet used to analyze malwares” USENIX Association Berkeley,2007