**Cross-VM Network Channel Attacks and Countermeasures within Cloud Computing Environments**

**1. INTRODUCTION**

Cloud computing has risen in prominence due to its service model enabling elastic on-demand access to computing resources and now underpins modern business operations. Cloud computing security is an important concern for enterprises when they shift critical information to geographically distributed cloud platforms that are directly not under their jurisdiction of control. There are many security concerns for cloud computing as it utilizes different technologies spanning networks, databases, operating systems, virtualization, resource scheduling, transaction management, load balancing, concurrency control and memory management, which are potentially vulnerable to attacks. Cloud computing security researchers work to introduce new security exploitation events or attacks that may affect the providers and users. Cloud computing heavily leverage virtualization methodologies. Virtualization facilitates multiple operating systems (different or same) to co-reside on the same physical server concurrently. Virtualization technologies such as Hyper, Xen, KVM, and VMWare are key enablers of cloud computing systems. Their key benefit is in cost saving. Same physical hardware is offered among multiple VMs along with the ability to provide strong isolation between co-residence VMs, i.e. a guest VM cannot interfere in the operation of other guest VMs running on the same machine. Such isolation is a key support for major public providers such as Microsoft Azure, Amazon Elastic Compute Cloud (EC2), Google Compute Engine (GCE), and Rackspace. However, such logical isolation is not impenetrable. A myriad of previous studies has demonstrated how coresident VMs could be vulnerable to attacks through shared file systems, cache side-channels or through compromising hypervisor layer using rootkits. Thus, the threat of cross-VM attacks remains where an attacker uses one VM to control or access other VMs on the same hypervisor. As such, various methods have been devised for strategic VM placement in order to exploit co-residency. Hypervisors virtualization attempt to realize this assumption by implementing logical isolation between VMs using traditional access-control approaches. However, it is possible for attackers to circumvent them via side-channel attacks. Moreover the existence and threat of ROP (Return- Oriented Programming) attack in real time systems has been discussed in variety of settings. Researchers have demonstrated that if an attacker can successfully exploit the earlier version of Adobe reader and Acrobat by launching ROP, they may be able to are at high risk of bugs, due to which an attacker manages to launch an ROP attack on them. Despite the clear potential of cross-VM attacks for exploiting shared memory and disk, exhibition of cross-VM network-channel and privilege escalation that uses the ROP in conjunction with the network-channel attack has not been demonstrated. Current network-based attacks exploit existing vulnerabilities such as ARP spoofing and DNS poisoning that are difficult to use for VM-targeted attacks. The most commonly discussed network-based challenges focus on the fact that cloud providers place more layers of isolation between co-resided VMs than in non-virtualized settings because the attacker and victim are often assigned to separate segmentations of virtual networks and domains. Cloud providers mitigate cross-VM network channel attacks by introducing the concept of isolation through an internal virtual network. Logical isolation of hardware resources can give protection against poor access-control policies, preventing VMs running on the same hardware from interfering with each other’s execution or data ex-filtration. However, such logical isolation may not be appropriate if an attacker can bypass them by launching a different type of attack. This paper proposes novel methodologies in which the attacker VM can redirect network traffic of the victim VM and set the unobtrusive destination point to receive the network traffic of the victim. It also launches a privilege escalation attack by exploiting ROP in conjunction with network-channel in cross-VM settings. The aim of these investigations is to explore whether the isolation of cloud systems, i.e., virtual machines and hypervisors, can be circumvented by the proposed attack (and if so, how?). We demonstrate that these two attacks result in successful violation of isolation properties of virtualization and escalate privilege level of non-root VMs. For responsible disclosure, all vulnerabilities found in our research have been reported to the OpenStack and Ravello security teams, we have also provided solutions for fixing the identified issues. This paper is an extension of our conference paper, which proposed a novel zero-day network channel attack for redirecting the traffic of other co-located VMs. In this attack, the created dummy interface impersonates a TAP (Test Access Point) device. Impersonated TAP device in combination with the network mirror exploits the network channel. The network mirror then redirects the network traffic of other co-located VMs to the desired destination point. This approach is further extended through another novel zero-day privilege escalation attack in a cross VM cloud environment. It works by escalating the privilege level of non-root VMs. The exploitation of Return Oriented Programming (ROP) in conjunction with the network channel is used to launch this attack. Non-Root VM will hijack and control the Tool Stack of the hypervisor, from this it can control all other co-located VMs. This paper also discusses the countermeasures for these attacks. compromise and control the victimized system. Furthermore, developers have developed ROP-based rootkits to compromise Window operating Systems . After the execution of these rootkits, the attackers are successful in hiding a malicious process through which these rootkits manage to bypass the integrity protection system of the OS. So far, all ROP based attacks target either the applications or operating systems and do not target hypervisors directly. Hypervisors having large set of code.

* 1. **Objective of the Project**

Cloud providers attempt to maintain the highest levels of isolation between Virtual Machines (VMs) and inter-user processes to keep co-located VMs and processes separate. This logical isolation creates an internal virtual network to separate VMs co-residing within a shared physical network. However, as co-residing VMs share their underlying VMM (Virtual Machine Monitor), virtual network, and hardware are susceptible to cross VM attacks. It is possible for a malicious VM to potentially access or control other VMs through network connections, shared memory, other shared resources, or by gaining the privilege level of its non-root machine. This research presents a two novel zero-day cross-VM network channel attacks. In the first attack, a malicious VM can redirect the network traffic of target VMs to a specific destination by impersonating the Virtual Network Interface Controller (VNIC). The malicious VM can extract the decrypted information from target VMs by using open source decryption tools such as Aircrack. The second contribution of this research is a privilege escalation attack in a cross VM cloud environment with Xen hypervisor. An adversary having limited privileges rights may execute Return-Oriented Programming (ROP), establish a connection with the root domain by exploiting the network channel, and acquiring the tool stack (root domain) which it is not authorized to access directly. Countermeasures against this attacks are also presented.

**2.LITERATURE SURVEY**

**"SecVisor: A tiny hypervisor to provide lifetime kernel code integrity for commodity OSes",**

A. Seshadri et al.,2007

We propose SecVisor, a tiny hypervisor that ensures code integrity for commodity OS kernels. In particular, SecVisor ensures that only user-approved code can execute in kernel mode over the entire system lifetime. This protects the kernel against code injection attacks, such as kernel rootkits. SecVisor can achieve this propertyeven against an attacker who controls everything but the CPU, the memory controller, and system memory chips. Further, SecVisor can even defend against attackers with knowledge of zero-day kernel exploits.Our goal is to make SecVisor amenable to formal verificationand manual audit, thereby making it possible to rule out known classes of vulnerabilities. To this end, SecVisor offers small code size and small external interface. We rely on memory virtualization to build SecVisor and implement two versions, one using software memory virtualization and the other using CPU-supported memory virtualization. The code sizes of the runtime portions of these versions are 1739 and 1112 lines, respectively. The size of the external interface for both versions of SecVisor is 2 hypercalls. It is easy to port OS kernels to SecVisor.

**"Hey you get off of my cloud: Exploring information leakage in third-party compute clouds",**

T. Ristenpart, E. Tromer, H. Shacham,2009

Third-party cloud computing represents the promise of outsourcing as applied to computation. Services, such as Microsoft's Azure and Amazon's EC2, allow users to instantiate virtual machines (VMs) on demand and thus purchase precisely the capacity they require when they require it. In turn, the use of virtualization allows third-party cloud providers to maximize the utilization of their sunk capital costs by multiplexing many customer VMs across a shared physical infrastructure. However, in this paper, we show that this approach can also introduce new vulnerabilities. Using the Amazon EC2 service as a case study, we show that it is possible to map the internal cloud infrastructure, identify where a particular target VM is likely to reside, and then instantiate new VMs until one is placed co-resident with the target. We explore how such placement can then be used to mount cross-VM side-channel attacks to extract information from a target VM on the same machine.

**"Whispers in the hyper-space: High-speed covert channel attacks in the cloud",**

Z. Wu, Z. Xu and H. Wang,2012

Information security and privacy in general are major concerns that impede enterprise adaptation of shared or public cloud computing. Specifically, the concern of virtual machine (VM) physical co-residency stems from the threat that hostile tenants can leverage various forms of side channels (such as cache covert channels) to exfiltrate sensitive information of victims on the same physical system. However, on virtualized x86 systems, covert channel attacks have not yet proven to be practical, and thus the threat is widely considered a "potential risk". In this paper, we present a novel covert channel attack that is capable of high-bandwidth and reliable data trans-mission in the cloud. We first study the application of existing cache channel techniques in a virtualized environment, and uncover their major insufficiency and difficulties. We then overcome these obstacles by (1) re-designing a pure timing-based data transmission scheme, and (2) exploiting the memory bus as a high-bandwidth covert channel medium. We further design and implement a robust communication protocol, and demonstrate realistic covert channel attacks on various virtualized x86 systems. Our experiments show that covert channels do pose serious threats to information security in the cloud. Finally, we discuss our insights on covert channel mitigation in virtualized environments.

**"Subverting VistaTM kernel for fun and profit",**

J. Rutkowska,2006

Many defensive approaches have been proposed to protect the integrity of the operating system kernel stack. However, some types of attacks, such as the return-to-schedule rootkit, pose a serious threat to these approaches. In this paper, we present a kernel stack protection model to protect the integrity of the kernel stack. It adopts a synchronous design strategy to bind the execution unit with its kernel stack using virtualization technology, and allows the execution unit to write its own current kernel stack with legal kernel codes. To test the model, we propose three kinds of potential attacks which extend the return-to-schedule rootkit. The experimental results show that the prototype of the model can be effective against all attack methods, and introduces a performance cost of only 2%. Therefore, it effectively protects all types of data on the kernel stack with a small performance overhead.

**"A survey on the security of virtual machines",**

D. Hyde,2009

Virtual machines (VM) are rapidly replacing physical machine infrastructures for their abilities to emulate hardware environments, share hardware resources, and utilize a variety of operating systems (OS). VMs provide a better security model than traditional machines by providing an additional layer of hardware abstraction and isolation, effective external monitoring and recording, and on-demand access. However, this new model requires adaptation of existing security methods, which cannot currently keep up with the ease of creating new VMs with a variety of configurations and lifecycles. Attackers have successfully compromised VM infrastructures, allowing them to access other VMs on the same system and even the host. Fortunately, these security concerns are being addressed and users can prevent most intrusions by applying traditional security measures to each VM.

**"Virtual environment security-considerations & practices",**

S. R. Kumari and V. Kathiresan,2011

Virtualization is playing a vital role in various organizations by providing high flexibility to their environment. The users are getting tremendous benefits in terms of cost, utilization and efficiency. It promises to overcome the pitfalls in the traditional computing & provides the best solutions. Though Virtualization has many advantages over traditional computing, many organizations still have the dilemma to implement virtualization because of the security concerns. This paper provides the literature survey of virtualization, types of virtualization technologies and possible threats in virtual environment. We are providing a list of best practices and considerations which any organization must consider before implementing virtualization in their environment.

**"Deep-diving into an easily-overlooked threat: Inter-VM attacks",**

S. Zhang,2012

As cybersecurity threats evolve, cloud computing defenses must adapt to face new challenges. Unfortunately, due to resource sharing, cloud computing platforms open the door for insider attacks, which consist of malicious actions from cloud authorized users (e.g., clients of an Infrastructure-as-a-Service (IaaS) cloud) targeting the co-hosted users or the underlying provider environment. Virtual machine (VM) migration is a Moving Target Defense (MTD) technique to mitigate insider attacks effects, as it provides VMs positioning manageability. However, there is a clear demand for studies quantifying the security benefits of VM migration-based MTD considering different system architecture configurations. This paper tries to fill such a gap by presenting a Stochastic Reward Net model for the security evaluation of a VM migration-based MTD. The security metric of interest is the probability of attack success. We consider multiple architectures, ranging from one physical machine pool (without MTD) up to four physical machine pools. The evaluation also considers the unavailability due to VM migration. The key contributions are a set of results highlighting the probability of insider attacks success over time in different architectures and VM migration schedules, and suggestions for selecting VMs as candidates for MTD deployment based on the tolerance levels of the attack success probability. The results are validated against simulation results to confirm the accuracy of the model.

**"On detecting co-resident cloud instances using network flow watermarking techniques",**

A. Bates et al.,2014

Virtualization is the cornerstone of the developing third-party compute industry, allowing cloud providers to instantiate multiple virtual machines (VMs) on a single set of physical resources. Customers utilize cloud resources alongside unknown and untrusted parties, creating the co-resident threat—unless perfect isolation is provided by the virtual hypervisor, there exists the possibility for unauthorized access to sensitive customer information through the exploitation of covert side channels. This paper presents co-resident watermarking, a traffic analysis attack that allows a malicious co-resident VM to inject a watermark signature into the network flow of a target instance. This watermark can be used to exfiltrate and broadcast co-residency data from the physical machine, compromising isolation without reliance on internal side channels. As a result, our approach is difficult to defend against without costly underutilization of the physical machine. We evaluate co-resident watermarkingunder a large variety of conditions, system loads and hardware configurations, from a local laboratory environment to production cloud environments (Futuregrid and the University of Oregon’s ACISS). We demonstrate the ability to initiate a covert channel of 4 bits per second, and we can confirm co-residency with a target VM instance in <10 s. We also show that passive load measurement of the target and subsequent behavior profiling is possible with this attack. We go on to consider the detectability of co-resident watermarking, extending our scheme to create a subtler watermarking attack by imitating legitimate cloud customer behavior. Our investigation demonstrates the need for the careful design of hardware to be used in the cloud.

**3. SYSTEM ANALYSIS**

**3.1 Existing System**

Cloud servers provides heavy computation resources and storage at cheaper cost so all users migrating their social and business data to 3rd party cloud servers. Data stored at cloud servers will be away from user control and can be misuse by cloud servers in various ways such as data tamper by internal employees or attackers who can hack VM (virtual machine) to divert traffic to different IP or send huge request to other VM to crash VM resources. In all this attack scenarios user data will be at risk.

**Disadvantages:**

1. Less Accuracy
2. More time taking process

**3.2 Proposed System**

In proposed system, introducing Monitor Node which will monitor all VM’s and if any VM sending huge packet or diverting request to other VM then that monitor will detect and drop such attack request to save user data.

Following are the resources used by author to monitor VM

VM-Monitor/Controller: Responsible for executing the services of management software that are needed for functioning of cloud platform.

Compute: Compute nodes execute virtual machine instances in cloud. KVM is used as a hypervisor in this node. This node is also responsible for providing firewall services. One can deploy more than one compute node in a setup.

Network: The responsibilities of network nodes ensure the creation of virtual networks needed by the customers to create public or private networks. It connects their virtual machines with the external networks, i.e. the Internet.

**Advantages:**

1. High Accuracy
2. Takes less time

**3.3 Modules Description**

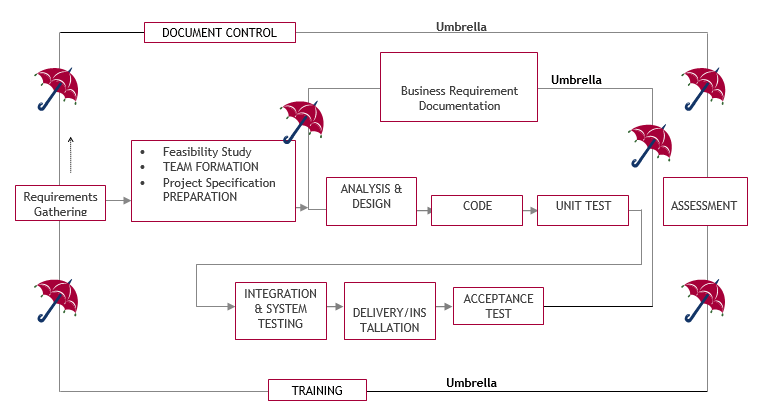
To implement this project we have designed following modules

1. Cloud Server: use to receive and store data from servers and for each request cloud will create and destroy VM as THREADS.
2. VM-Monitor Node: this is a controller node which will monitor each VM behaviour and if request diverting or sending huge packet data then VM will be detected as attack. Here there is no external attacker so we will upload huge file size which will be detected by monitor

User/simulation node: here user will upload or download files from cloud.

**3.4. Process Model Used With Justification**

**SDLC (Umbrella Model):**

****

SDLC is nothing but Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

**Stages in SDLC:**

1. Requirement Gathering
2. Analysis
3. Designing
4. Coding
5. Testing
6. Maintenance
7. **Requirements Gatheringstage:**

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.



These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are not included in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term requirements traceability.

The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

* Feasibility study is all about identification of problems in a project.
* No. of staff required to handle a project is represented as Team Formation, in this case only modules are individual tasks will be assigned to employees who are working for that project.
* Project Specifications are all about representing of various possible inputs submitting to the server and corresponding outputs along with reports maintained by administrator.

1. **Analysis Stage:**

The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.



The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high level estimates of effort for the out stages.

1. **Designing Stage:**

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.



When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

1. **Development (Coding) Stage:**

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artefacts will be produced. Software artefacts include but are not limited to menus, dialogs, and data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artefacts, and an online help system will be developed to guide users in their interactions with the software.



The RTM will be updated to show that each developed artefact is linked to a specific design element, and that each developed artefact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

1. **Integration & Test Stage:**

During the integration and test stage, the software artefacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability. During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.



The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

**Installation & Acceptance Test:**

During the installation and acceptance stage, the software artefacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer. After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.



The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labour data into the project schedule and locks the project as a permanent project record. At this point the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

1. **Maintenance:**

Outer rectangle represents maintenance of a project, Maintenance team will start with requirement study, understanding of documentation later employees will be assigned work and they will undergo training on that particular assigned category. For this life cycle there is no end, it will be continued so on like an umbrella (no ending point to umbrella sticks).

**3.5. Software Requirement Specification**

**3.5. 1. Overall Description**

A Software Requirements Specification (SRS) – a [requirements specification](http://en.wikipedia.org/wiki/Requirements_specification) for a [software system](http://en.wikipedia.org/wiki/Software_system) is a complete description of the behaviour of a system to be developed. It includes a set of [use cases](http://en.wikipedia.org/wiki/Use_case) that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. [Non-functional requirements](http://en.wikipedia.org/wiki/Non-functional_requirements) are requirements which impose constraints on the design or implementation (such as [performance engineering](http://en.wikipedia.org/wiki/Performance_engineering) requirements, [quality](http://en.wikipedia.org/wiki/Quality_%28business%29) standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A [business analyst](http://en.wikipedia.org/wiki/Business_analyst), sometimes titled [system analyst](http://en.wikipedia.org/wiki/System_analyst), is responsible for analysing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the [systems development lifecycle](http://en.wikipedia.org/wiki/Systems_development_life_cycle) domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

1. [Business requirements](http://en.wikipedia.org/wiki/Business_requirements) describe in business terms what must be delivered or accomplished to provide value.
2. Product requirements describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements.)
3. Process requirements describe activities performed by the developing organization. For instance, process requirements could specify .Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

* **ECONOMIC FEASIBILITY**

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economic feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs. The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, there is nominal expenditure and economical feasibility for certain.

* **Operational Feasibility**

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization’s operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

* **TECHNICAL FEASIBILITY**

Earlier no system existed to cater to the needs of ‘Secure Infrastructure Implementation System’. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to .the users. The database’s purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security.

**3.5. 2. External Interface Requirements**

1. **User Interface**

The user interface of this system is a user friendly python Graphical User Interface.

1. **Hardware Interfaces**

The interaction between the user and the console is achieved through python capabilities.

1. **Software Interfaces**

The required software is python.

1. **Operating Environment**

Windows XP.

1. **Hardware Requirements:**

# Processor - Intel i3(min)

* Speed - 1.1 GHz
* RAM - 256 MB(min)
* Hard Disk - 20 GB
* Key Board - Standard Windows Keyboard
* Mouse - Two or Three Button Mouse

1. **Software Requirements:**

* Operating System - Windows 10(min)
* Programming Language - Python

**4. SYSTEM DESIGN**

**4.1 UML Diagram:**

The Unified Modelling Language allows the software engineer to express an analysis model using the modelling notation that is governed by a set of syntactic semantic and pragmatic rules.

A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagram, which is as follows.

* + **User Model View**

1. This view represents the system from the user’s perspective.
2. The analysis representation describes a usage scenario from the end-user’s perspective.
   * **Structural Model view**
3. In this model the data and functionality are arrived from inside the system.
4. This model view models the static structures.

* **Behavioural Model View**

It represents the dynamic of behavioural as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

* **Implementation Model View**

In this the structural and behavioural as parts of the system are represented as they are to be built.

* **Environmental Model View**

In this the structural and behavioural aspects of the environment in which the system is to be implemented are represented.

1. **Class Diagram:**

The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. In the diagram, classes are represented with boxes which contain three parts:

* The upper part holds the name of the class
* The middle part contains the attributes of the class
* The bottom part gives the methods or operations the class can take or undertake

****

1. **Use case Diagram:**

A **use case diagram** at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.



1. **Sequence diagram:**

A sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



1. **Collaboration diagram:**

A collaboration diagram describes interactions among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behaviour of a system.



1. **Component Diagram:**

In the Unified Modelling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another component. This illustrates the service consumer - service provider relationship between the two components.



1. **Deployment Diagram:**

A **deployment diagram** in the Unified Modeling Language models the physical deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected (e.g. JDBC, REST, RMI).

The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have sub nodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.



1. **Activity Diagram:**

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. It is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent.

Send File Request to Cloud

Get Files From Cloud

Download File

Attack Simulation Graph

1. **Data Flow Diagram:**

Data flow diagrams illustrate how data is processed by a system in terms of inputs and outputs. Data flow diagrams can be used to provide a clear representation of any business function. The technique starts with an overall picture of the business and continues by analyzing each of the functional areas of interest. This analysis can be carried out in precisely the level of detail required. The technique exploits a method called top-down expansion to conduct the analysis in a targeted way.

As the name suggests, Data Flow Diagram (DFD) is an illustration that explicates the passage of information in a process. A DFD can be easily drawn using simple symbols. Additionally, complicated processes can be easily automated by creating DFDs using easy-to-use, free downloadable diagramming tools. A DFD is a model for constructing and analyzing information processes. DFD illustrates the flow of information in a process depending upon the inputs and outputs. A DFD can also be referred to as a Process Model. A DFD demonstrates business or technical process with the support of the outside data saved, plus the data flowing from the process to another and the end results.

User

1. Send File Request to Cloud 2. File request sent to clous successfully

3. Get Files From Cloud 4. Get files from cloud Successfully

5. Download File 6. File downloaded successfully

7. Attack Simulation Graph 8. Attack Simulation Graph generated successfully.

**5. IMPLEMETATION**

**5.1 Python**

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

**5.1.1 History of Python:**

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, [C](https://www.tutorialspoint.com/cprogramming/index.htm), [C++](https://www.tutorialspoint.com/cplusplus/index.htm), Algol-68, Smalltalk, and [Unix](https://www.tutorialspoint.com/unix/index.htm) shell and other [scripting languages](https://www.tutorialspoint.com/scripting_language_tutorials.htm).

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

For many uninitiated people, the word Python is related to a species of snake. Rossum though attributes the choice of the name Python to a popular comedy series **Monty Python's Flying Circus** on BBC.

Being the principal architect of Python, the developer community conferred upon him the title of **Benevolent Dictator for Life** (BDFL). However, in 2018, Rossum relinquished the title. Thereafter, the development and distribution of the reference implementation of Python is handled by a nonprofit organization **Python Software Foundation**.

* + 1. **Features of Python:**

1. **Easy to Learn**

This is one of the most important reasons for the popularity of Python. Python has a limited set of keywords. Its features such as simple [syntax](https://www.tutorialspoint.com/python/python_basic_syntax.htm), usage of indentation to avoid clutter of curly brackets and dynamic typing that doesn't necessitate prior declaration of variable help a beginner to learn Python quickly and easily.

1. **Dynamically Typed**

Python is a dynamically typed programming language. In Python, you don't need to specify the variable time at the time of the variable declaration. The types are specified at the runtime based on the assigned value due to its dynamically typed feature.

1. **Interpreter Based**

Instructions in any programming languages must be translated into machine code for the processor to execute them. Programming languages are either compiler based or interpreter based.

In case of a compiler, a [machine language](https://www.tutorialspoint.com/machine_learning/index.htm) version of the entire source program is generated. The conversion fails even if there is a single erroneous statement. Hence, the development process is tedious for the beginners. The C family languages (including [C](https://www.tutorialspoint.com/cprogramming/index.htm), [C++](https://www.tutorialspoint.com/cplusplus/index.htm), [Java](https://www.tutorialspoint.com/java/index.htm), [C#](https://www.tutorialspoint.com/csharp/index.htm) etc) are compiler based.

Python is an interpreter based language. The interpreter takes one instruction from the source code at a time, translates it into machine code and executes it. Instructions before the first occurrence of error are executed. With this feature, it is easier to debug the program and thus proves useful for the beginner level programmer to gain confidence gradually. Python therefore is a beginner-friendly language.

1. **Interactive**

Standard Python distribution comes with an interactive shell that works on the principle of REPL (Read – Evaluate – Print – Loop). The shell presents a Python prompt >>>. You can type any valid Python expression and press Enter. Python interpreter immediately returns the response and the prompt comes back to read the next expression.

>>> 2\*3+1

7

>>> print ("Hello World")

Hello World

The interactive mode is especially useful to get familiar with a library and test out its functionality. You can try out small code snippets in interactive mode before writing a program.

1. **Multi-paradigm**

Python is a completely [object-oriented](https://www.tutorialspoint.com/python/python_oops_concepts.htm) language. Everything in a Python program is an [object](https://www.tutorialspoint.com/python/python_object_classes.htm). However, Python conveniently encapsulates its object orientation to be used as an imperative or procedural language – such as C. Python also provides certain functionality that resembles functional programming. Moreover, certain third-party tools have been developed to support other programming paradigms such as aspect-oriented and logic programming.

1. **Standard Library**

Even though it has a very few keywords (only Thirty Five), Python software is distributed with a standard library made of large number of modules and packages. Thus Python has out of box support for programming needs such as serialization, data compression, internet data handling, and many more. Python is known for its batteries included approach.

Some of the Python's popular modules are:

* [NumPy](https://www.tutorialspoint.com/numpy/index.htm)
* [Pandas](https://www.tutorialspoint.com/python_pandas/index.htm)
* [Matplotlib](https://www.tutorialspoint.com/matplotlib/index.htm)
* Tkinter
* [Math](https://www.tutorialspoint.com/python/python_maths.htm)

## **Open Source and Cross Platform**

Python's standard distribution can be downloaded from <https://www.python.org/downloads/> without any restrictions. You can download pre-compiled binaries for various operating system platforms. In addition, the source code is also freely available, which is why it comes under open source category.

Python software (along with the documentation) is distributed under Python Software Foundation License. It is a BSD style permissive software license and compatible to GNU GPL (General Public License).

Python is a cross-platform language. Pre-compiled binaries are available for use on various operating system platforms such as [Windows](https://www.tutorialspoint.com/windows10/index.htm), [Linux](https://www.tutorialspoint.com/unix/index.htm), Mac OS, [Android OS](https://www.tutorialspoint.com/android/index.htm). The reference implementation of Python is called CPython and is written in C. You can download the source code and compile it for your OS platform.

A Python program is first compiled to an intermediate platform independent byte code. The virtual machine inside the interpreter then executes the byte code. This behaviour makes Python a cross-platform language, and thus a Python program can be easily ported from one OS platform to other.

## **GUI Applications**

Python's standard distribution has an excellent graphics library called TKinter. It is a Python port for the vastly popular GUI toolkit called TCL/Tk. You can build attractive user-friendly GUI applications in Python. GUI toolkits are generally written in C/C++. Many of them have been ported to Python. Examples are [PyQt](https://www.tutorialspoint.com/pyqt/index.htm), [WxWidgets](https://www.tutorialspoint.com/wxpython/index.htm), [PySimpleGUI](https://www.tutorialspoint.com/pysimplegui/index.htm) etc.

## **Database Connectivity**

Almost any type of database can be used as a backend with the Python application. DB-API is a set of specifications for database driver software to let Python communicate with a relational database. With many third party libraries, Python can also work with NoSQL databases such as [MongoDB](https://www.tutorialspoint.com/mongodb/index.htm).

## **Extensible**

The term extensibility implies the ability to add new features or modify existing features. As stated earlier, CPython (which is Python's reference implementation) is written in C. Hence one can easily write modules/libraries in C and incorporate them in the standard library. There are other implementations of Python such as Jython (written in Java) and [IPython](https://www.tutorialspoint.com/jupyter/ipython_introduction.htm) (written in C#). Hence, it is possible to write and merge new functionality in these implementations with Java and C# respectively.

## **Active Developer Community**

As a result of Python's popularity and open-source nature, a large number of Python developers often interact with online forums and conferences. Python Software Foundation also has a significant member base, involved in the organization's mission to "**Promote, Protect, and Advance the Python Programming Language**"

Python also enjoys a significant institutional support. Major IT companies Google, Microsoft, and Meta contribute immensely by preparing documentation and other resources.

**5.1.3 Applications of Python:**

[Python](https://www.tutorialspoint.com/python/python_overview.htm) is a general-purpose programming language. It is suitable for the development of a wide range of software applications. Over the last few years Python has been the preferred language of choice for developers in the following application areas −

* [Data Science](https://www.tutorialspoint.com/python/python_application_areas.htm#data_science)
* [Machine Learning](https://www.tutorialspoint.com/python/python_application_areas.htm#machine_learning)
* [Web Development](https://www.tutorialspoint.com/python/python_application_areas.htm#web_development)
* [Computer Vision and Image processing](https://www.tutorialspoint.com/python/python_application_areas.htm#computer_vision_and_image_processing)
* [Embedded Systems and IoT](https://www.tutorialspoint.com/python/python_application_areas.htm#embedded_systems_and_iot)
* [Job Scheduling and Automation](https://www.tutorialspoint.com/python/python_application_areas.htm#job_scheduling_and_automation)
* [Desktop GUI Applications](https://www.tutorialspoint.com/python/python_application_areas.htm#desktop_gui_applications)
* [Console-based Applications](https://www.tutorialspoint.com/python/python_application_areas.htm#console_based_applications)
* [CAD Applications](https://www.tutorialspoint.com/python/python_application_areas.htm#cad_applications)
* [Game Development](https://www.tutorialspoint.com/python/python_application_areas.htm#game_development)

## **Data Science**

Python's recent meteoric rise in the popularity charts is largely due to its Data science libraries. Python has become an essential skill for data scientists. Today, real time web applications, mobile applications and other devices generate huge amount of data. Python's data science libraries help companies generate business insights from this data.

Libraries like [NumPy](https://www.tutorialspoint.com/numpy/index.htm), [Pandas](https://www.tutorialspoint.com/python_pandas/index.htm), and [Matplotlib](https://www.tutorialspoint.com/matplotlib/index.htm) are extensively used to apply mathematical algorithms to the data and generate [visualizations](https://www.tutorialspoint.com/python_pandas/python_pandas_visualization.htm). Commercial and community Python distributions like Anaconda and ActiveState bundle all the essential libraries required for data science.

1. **Machine Learning**

Python libraries such as [Scikit-learn](https://www.tutorialspoint.com/scikit_learn/index.htm) and [TensorFlow](https://www.tutorialspoint.com/tensorflow/index.htm) help in building models for prediction of trends like customer satisfaction, projected values of stocks etc. based upon the past data. [Machine learning](https://www.tutorialspoint.com/machine_learning/index.htm) applications include (but not restricted to) medical diagnosis, statistical arbitrage, basket analysis, sales prediction etc.

1. **Web Development**

Python's web frameworks facilitate rapid web application development. [Django](https://www.tutorialspoint.com/django/index.htm), [Pyramid](https://www.tutorialspoint.com/python_pyramid/index.htm), [Flask](https://www.tutorialspoint.com/flask/index.htm) are very popular among the web developer community. etc. make it very easy to develop and deploy simple as well as complex web applications.

Latest versions of Python provide asynchronous programming support. Modern web frameworks leverage this feature to develop fast and high performance web apps and APIs.

1. **Computer Vision and Image processing**

[OpenCV](https://www.tutorialspoint.com/opencv_python/index.htm) is a widely popular library for capturing and processing images. Image processing algorithms extract information from images, reconstruct image and video data. Computer Vision uses image processing for face detection and pattern recognition. OpenCV is a C++ library. Its Python port is extensively used because of its rapid development feature.

Some of the application areas of computer vision are [robotics](https://www.tutorialspoint.com/artificial_intelligence/artificial_intelligence_robotics.htm), industrial surveillance, automation, and [biometrics](https://www.tutorialspoint.com/biometrics/index.htm) etc

* 1. **Python Libraries**

1. **NumPy**

**NumPy**, which stands for **Numerical Python**, is an open-source [Python](https://www.tutorialspoint.com/python/index.htm) library consisting of multidimensional and single-dimensional array elements. It's a standard that computes numerical data in Python. NumPy is most widely used in almost every domain where numerical computation is required, like science and engineering; hence, the NumPy API functionalities are highly utilized in data science and scientific [Python packages](https://www.tutorialspoint.com/python/python_packages.htm), including [Pandas](https://www.tutorialspoint.com/python_pandas/index.htm), [SciPy](https://www.tutorialspoint.com/scipy/index.htm), [Matplotlib](https://www.tutorialspoint.com/matplotlib/index.htm), [scikit-learn](https://www.tutorialspoint.com/scikit_learn/index.htm), scikit-image, and many more.

**Why NumPy - Need of NumPy**

NumPy is a fundamental package for numerical computation in Python. It provides mathematical functions to compute data as well as functions to operate multi-dimensional arrays and matrices efficiently. Here are some reasons why NumPy is essential:

* NumPy includes a wide range of mathematical functions for basic arithmetic, linear algebra, Fourier analysis, and more.
* NumPy performs numerical operations on large datasets efficiently.
* NumPy supports multi-dimensional arrays, allowing for the representation of complex data structures such as images, sound waves, and tensors in [machine learning](https://www.tutorialspoint.com/machine_learning/index.htm) models.
* It supports the writing of concise and readable code for complex mathematical computations.
* NumPy integrates with other libraries to do scientific computation; these are SciPy (for scientific computing), Pandas (for data manipulation and analysis), and scikit-learn (for machine learning).
* Many scientific and numerical computing libraries and tools are built on top of NumPy.
* Its widespread adoption and stability make it a standard choice for numerical computing tasks.

NumPy plays a crucial role in the Python ecosystem for scientific computing, data analysis, machine learning, and more. Its efficient array operations and extensive mathematical functions make it an indispensable tool for working with numerical data in Python.

**Installation command:**

**pip install numpy==1.19.2**

**Example:**

import numpy as np

a = np.array([0, 30, 45, 60, 90])

print('Array containing sine values:')

sin = np.sin(a \* np.pi / 180)

print(sin)

print('\n')

print('Compute sine inverse of angles. Returned values are in radians.')

inv = np.arcsin(sin)

print(inv)

print('\n')

print('Check result by converting to degrees:')

print(np.degrees(inv))

print('\n')

print('arccos and arctan functions behave similarly:')

cos = np.cos(a \* np.pi / 180)

print(cos)

print('\n')

print('Inverse of cos:')

inv = np.arccos(cos)

print(inv)

print('\n')

print('In degrees:')

print(np.degrees(inv))

print('\n')

print('Tan function:')

tan = np.tan(a \* np.pi / 180)

print(tan)

print('\n')

print('Inverse of tan:')

inv = np.arctan(tan)

print(inv)

print('\n')

print('In degrees:')

print(np.degrees(inv))

**Its output is as follows −**

Array containing sine values:

[ 0. 0.5 0.70710678 0.8660254 1. ]

Compute sine inverse of angles. Returned values are in radians.

[ 0. 0.52359878 0.78539816 1.04719755 1.57079633]

Check result by converting to degrees:

[ 0. 30. 45. 60. 90.]

arccos and arctan functions behave similarly:

[ 1.00000000e+00 8.66025404e-01 7.07106781e-01 5.00000000e-01

6.12323400e-17]

Inverse of cos:

[ 0. 0.52359878 0.78539816 1.04719755 1.57079633]

In degrees:

[ 0. 30. 45. 60. 90.]

Tan function:

[ 0.00000000e+00 5.77350269e-01 1.00000000e+00 1.73205081e+00

1.63312394e+16]

Inverse of tan:

[ 0. 0.52359878 0.78539816 1.04719755 1.57079633]

In degrees:

[ 0. 30. 45. 60. 90.]

1. **Matplotlib**

What Is Matplotlib?

**Matplotlib** is one of the most popular Python packages used for data visualization. It is a cross-platform library for making 2D plots from data in arrays. It provides an object-oriented API that helps in embedding plots in applications using Python GUI toolkits such as [PyQt](https://www.tutorialspoint.com/pyqt/index.htm), [WxPython](https://www.tutorialspoint.com/wxpython/index.htm), or [Tkinter](https://www.tutorialspoint.com/python/python_gui_programming.htm). It can be used in Python and IPython shells, [Jupyter notebook](https://www.tutorialspoint.com/matplotlib/matplotlib_jupyter_notebook.htm) and web application servers also.

Matplotlib is a [Python](https://www.tutorialspoint.com/python/index.htm) library that is specifically designed to do effective data visualization. It's a cornerstone of plotting libraries in Python which empowers beginners to dive into the world of attractive data visualization. Matplotlib is an open-source Python library that offers various data visualization (like Line plots, histograms, scatter plots, bar charts, Scatter plots, Pie Charts, and Area Plot etc). A beauty of the Python matplotlib library is its Python code. Its script is structured which denotes that a few lines of code are all that are required in most instances to generate a visual data plot.

**Matplotlib and Pyplot**

**Matplotlib** is a versatile toolkit that allows for the creation of static, animated, and interactive visualizations in the Python programming language.

Generally, matplotlib overlays two APIs:

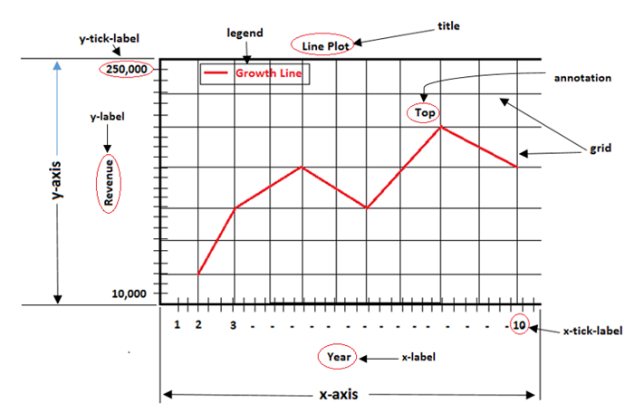
* **The pyplot API**: to make plot using **matplotlib.pyplot**.
* **Object-Oriented API**: A group of objects assembled with greater flexibility than [pyplot](https://www.tutorialspoint.com/matplotlib/matplotlib_pyplot_api.htm). It provides direct access to Matplotlib’s backend layers.

Matplotlib simplifies simple tasks and enables complex tasks to be accomplished. Following are the key aspects of matplotlib:

* Matplotlib offers to create quality plots.
* Matplotlib offers interactive figures and customizes their visual style that can be manipulated as per need.
* Matplotlib offers export to many file formats.

The most common way to use Matplotlib is through its pyplot module.

**Components of Matplotlib**



**Installation command:**

**pip install matplotlib==3.1.1**

**Example**

import matplotlib.pyplot as plt

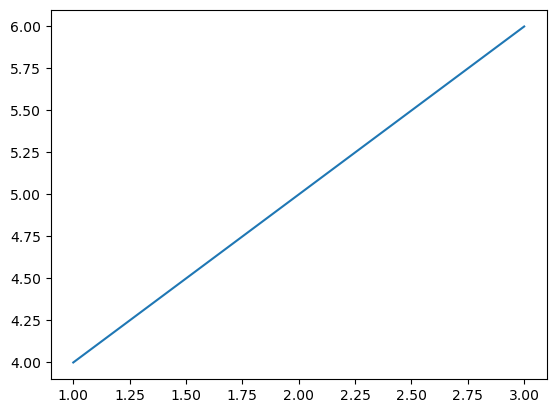
# Create a new figure

fig = plt.figure()

# Add a plot or subplot to the figure

plt.plot([1, 2, 3], [4, 5, 6])

plt.show()



1. **Socket Server**

Socket programming is a way of connecting two nodes on a network to communicate with each other. One socket(node) listens on a particular port at an IP, while the other socket reaches out to the other to form a connection. The server forms the listener socket while the client reaches out to the server.

They are the real backbones behind web browsing. In simpler terms, there is a server and a client.   
Socket programming is started by importing the socket library and making a simple socket.

import socket

s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

Here we made a socket instance and passed it two parameters. The first parameter is **AF\_INET** and the second one is **SOCK\_STREAM**. AF\_INET refers to the address-family ipv4. The SOCK\_STREAM means connection-oriented TCP protocol.

Now we can connect to a server using this socket.

**Connecting to a server:**

Note that if any error occurs during the creation of a socket then a socket. error is thrown and we can only connect to a server by knowing its IP. You can find the IP of the server by using this :

$ ping www.google.com

You can also find the IP using python:

import socket

ip = socket.gethostbyname('www.google.com')

print ip

Here is an example of a script for connecting to Google.

* Python3

|  |
| --- |
| # An example script to connect to Google using socket  # programming in Python  **import** socket # for socket  **import** sys    **try**:      s **=**socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)  **print** ("Socket successfully created")  **except**socket.error as err:      print ("socket creation failed with error %s" **%**(err))    # default port for socket  port **=** 80    **try**:      host\_ip**=**socket.gethostbyname('www.google.com')  **except**socket.gaierror:        # this means could not resolve the host      print ("there was an error resolving the host")      sys.exit()    # connecting to the server  s.connect((host\_ip, port))    print ("the socket has successfully connected to google") |

**Output :**

Socket successfully created

there was an error resolving the host

Here when we will be successfully connected the output will be:

Socket successfully created

the socket has successfully connected to google

* First of all, we made a socket.
* Then we resolved google’s IP and lastly, we connected to google.
* Now we need to know how can we send some data through a socket.
* For sending data the socket library has a *sendall* function. This function allows you to send data to a server to which the socket is connected and the server can also send data to the client using this function

**5.3 Sample Code:**

**Main.py**

**from tkinter import messagebox**

**from tkinter import \***

**from tkinter import simpledialog**

**import tkinter**

**import matplotlib.pyplot as plt**

**import numpy as np**

**import pyaes, pbkdf2, binascii, os, secrets**

**import base64**

**from tkinter import ttk**

**from tkinter import filedialog**

**import os**

**import json**

**from hashlib import sha256**

**import socket**

**main = Tk()**

**main.title("Cross-VM Network Channel Attacks and Countermeasures within Cloud Computing Environments")**

**main.geometry("1300x1200")**

**global files, tf1**

**def getKey(): #generating key with PBKDF2 for AES**

**password = "s3cr3t\*c0d3"**

**passwordSalt = '76895'**

**key = pbkdf2.PBKDF2(password, passwordSalt).read(32)**

**return key**

**def encrypt(plaintext): #AES data encryption**

**aes = pyaes.AESModeOfOperationCTR(getKey(), pyaes.Counter(31129547035000047302952433967654195398124239844566322884172163637846056248223))**

**ciphertext = aes.encrypt(plaintext)**

**return ciphertext**

**def decrypt(enc): #AES data decryption**

**aes = pyaes.AESModeOfOperationCTR(getKey(), pyaes.Counter(31129547035000047302952433967654195398124239844566322884172163637846056248223))**

**decrypted = aes.decrypt(enc)**

**return decrypted**

**def uploadFile():**

**text.delete('1.0', END)**

**filename = filedialog.askopenfilename(initialdir=".")**

**with open(filename, 'rb') as file:**

**data = file.read()**

**file.close()**

**name = os.path.basename(filename)**

**data = encrypt(data)**

**print(len(data))**

**hashcode = sha256(data).hexdigest()**

**data = base64.b64encode(data)**

**client = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)**

**client.connect(('localhost', 3333))**

**jsondata = json.dumps({"request": 'upload', "filename": name, "filedata": data.decode(), "hash": hashcode})**

**client.send(jsondata.encode())**

**data = client.recv(100)**

**data = data.decode()**

**text.insert(END,"Server Response : "+data+"\n\n")**

**def downloadFile():**

**text.delete('1.0', END)**

**name = tf1.get()**

**jsondata = json.dumps({"request": 'download', "filename": name})**

**client = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)**

**client.connect(('localhost', 2222))**

**client.send(jsondata.encode())**

**data = client.recv(100000)**

**decrypted = decrypt(data)**

**with open('Received/'+name, 'wb') as file:**

**file.write(decrypted)**

**file.close()**

**text.insert(END,"File saved as "+name+" inside Received folder\n")**

**def getFiles():**

**global files**

**client = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)**

**client.connect(('localhost', 2222))**

**jsondata = json.dumps({"request": 'files'})**

**client.send(jsondata.encode())**

**data = client.recv(1000)**

**data = data.decode()**

**data = data.strip()**

**files = data.split(",")**

**tf1['values'] = files**

**if len(files) > 0:**

**tf1.current(0)**

**def graph():**

**client = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)**

**client.connect(('localhost', 3333))**

**jsondata = json.dumps({"request": 'graph'})**

**message = client.send(jsondata.encode())**

**data = client.recv(100)**

**data = data.decode()**

**data = data.strip()**

**data = data.split(" ")**

**height = [int(data[0]), int(data[1]), int(data[2])]**

**bars = ('Total Request', 'Attack Request', 'Normal Request')**

**y\_pos = np.arange(len(bars))**

**plt.bar(y\_pos, height)**

**plt.xticks(y\_pos, bars)**

**plt.title("Attack Simulation Graph")**

**plt.show()**

**def close():**

**main.destroy()**

**def runGUI():**

**global text, tf1, files**

**font = ('times', 15, 'bold')**

**title = Label(main, text='Cross-VM Network Channel Attacks and Countermeasures within Cloud Computing Environments')**

**title.config(bg='mint cream', fg='olive drab')**

**title.config(font=font)**

**title.config(height=3, width=120)**

**title.place(x=0,y=5)**

**font1 = ('times', 14, 'bold')**

**ff = ('times', 12, 'bold')**

**l1 = Label(main, text='Available Files')**

**l1.config(font=font1)**

**l1.place(x=50,y=100)**

**files = []**

**files.append("Available Files")**

**tf1 = ttk.Combobox(main,values=files,postcommand=lambda: tf1.configure(values=files))**

**tf1.place(x=200,y=100)**

**tf1.config(font=font1)**

**uploadButton = Button(main, text="Send File Request to Cloud", command=uploadFile)**

**uploadButton.place(x=50,y=150)**

**uploadButton.config(font=ff)**

**uploadButton = Button(main, text="Get Files From Cloud", command=getFiles)**

**uploadButton.place(x=280,y=150)**

**uploadButton.config(font=ff)**

**downloadButton = Button(main, text="Download File", command=downloadFile)**

**downloadButton.place(x=480,y=150)**

**downloadButton.config(font=ff)**

**graphButton = Button(main, text="Attack Simulation Graph", command=graph)**

**graphButton.place(x=650,y=150)**

**graphButton.config(font=ff)**

**closeButton = Button(main, text="Exit", command=close)**

**closeButton.place(x=50,y=200)**

**closeButton.config(font=ff)**

**font1 = ('times', 13, 'bold')**

**text=Text(main,height=22,width=100)**

**scroll=Scrollbar(text)**

**text.configure(yscrollcommand=scroll.set)**

**text.place(x=10,y=250)**

**text.config(font=font1)**

**main.config(bg='gainsboro')**

**main.mainloop()**

**if \_\_name\_\_ == '\_\_main\_\_':**

**runGUI()**

**6. TESTING**

**Implementation and Testing:**

Implementation is one of the most important tasks in project is the phase in which one has to be cautions because all the efforts undertaken during the project will be very interactive. Implementation is the most crucial stage in achieving successful system and giving the users confidence that the new system is workable and effective. Each program is tested individually at the time of development using the sample data and has verified that these programs link together in the way specified in the program specification. The computer system and its environment are tested to the satisfaction of the user.

## **A. Implementation**

## The implementation phase is less creative than system design. It is primarily concerned with user training, and file conversion. The system may be requiring extensive user training. The initial parameters of the system should be modifies as a result of a programming. A simple operating procedure is provided so that the user can understand the different functions clearly and quickly. The different reports can be obtained either on the inkjet or dot matrix printer, which is available at the disposal of the user. The proposed system is very easy to implement. In general implementation is used to mean the process of converting a new or revised system design into an operational one.

## **Testing**

Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system property functions as a unit. The test data should be chosen such that it passed through all possible condition. Actually testing is the state of implementation which aimed at ensuring that the system works accurately and efficiently before the actual operation commence. The following is the description of the testing strategies, which were carried out during the testing period.

### **System Testing**

Testing has become an integral part of any system or project especially in the field of information technology. The importance of testing is a method of justifying, if one is ready to move further, be it to be check if one is capable to with stand the rigors of a particular situation cannot be underplayed and that is why testing before development is so critical. When the software is developed before it is given to user to use the software must be tested whether it is solving the purpose for which it is developed. This testing involves various types through which one can ensure the software is reliable. The program was tested logically and pattern of execution of the program for a set of data are repeated. Thus the code was exhaustively checked for all possible correct data and the outcomes were also checked.

1. **Module Testing**

To locate errors, each module is tested individually. This enables us to detect error and correct it without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level. Each module in the system is tested separately. For example the job classification module is tested separately. This module is tested with different job and its approximate execution time and the result of the test is compared with the results that are prepared manually. The comparison shows that the results proposed system works efficiently than the existing system. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

1. **Integration Testing**

After the module testing, the integration testing is applied. When linking the modules there may be chance for errors to occur, these errors are corrected by using this testing. In this system all modules are connected and tested. The testing results are very correct. Thus the mapping of jobs with resources is done correctly by the system.

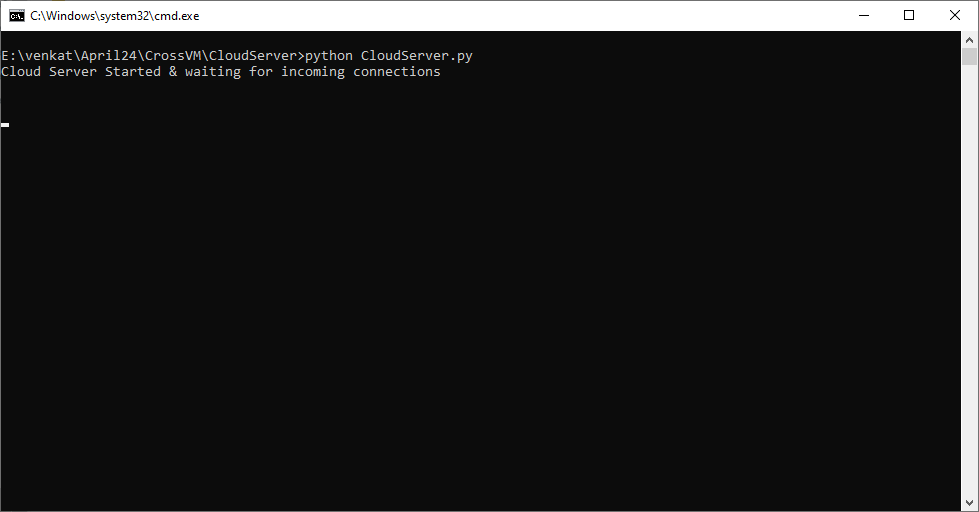
1. **Acceptance Testing**

When that user fined no major problems with its accuracy, the system passers through a final acceptance test. This test confirms that the system needs the original goals, objectives and requirements established during analysis without actual execution which elimination wastage of time and money acceptance tests on the shoulders of users and management, it is finally acceptable and ready for the operation.

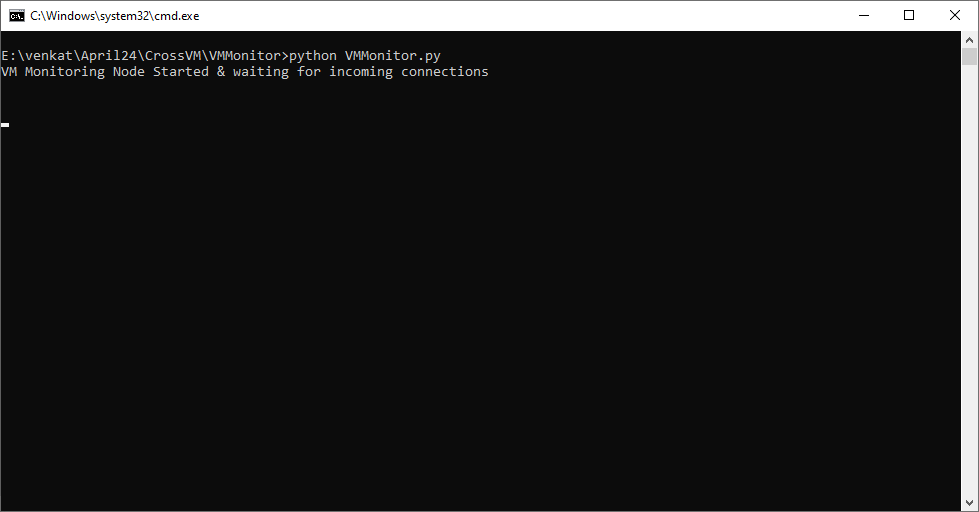
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test  Case  ID | Test  case  Name | Test Case  Desc. | Step | Expected | Actual | Test  Case  Status | Test  Case  Priority |
| 01 | Send File Request to Cloud | Verify the  Whether File Request to Cloud send or not | If  File Request to Cloud May Not be send | We cannot do the further operations | Test dataset is Uploaded | High | High |
| 02 | Get Files From Cloud | Verify  Whether Files Get From Cloud not | If The  Files From Cloud may  not got | We cannot do the further operations | We can do the further operations | High | High |
| 03 | Download File | Verify  Download Filedone or not | If theDownload File may not done | We cannot do the further operations | We can do the further operations | High | High |
| 04 | Attack Simulation Graph | Verify Attack Simulation GraphShow or not | If the Attack Simulation Graph is not shown | We cannot do the further operation | We can do the further operations | High | High |

**7. SCREENSHOTS**

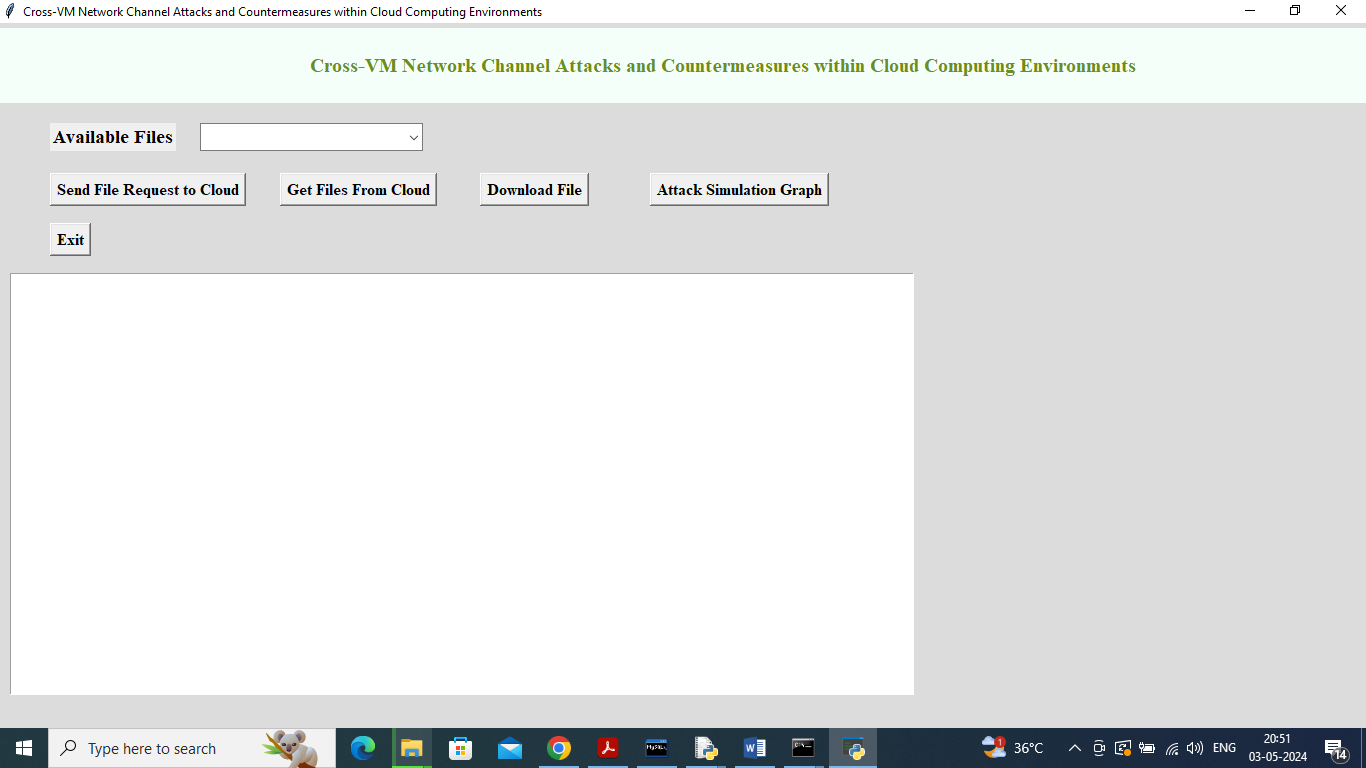
To run project first double click on ‘runServer.bat’ file from ‘Cloud Server’ folder to get below screen



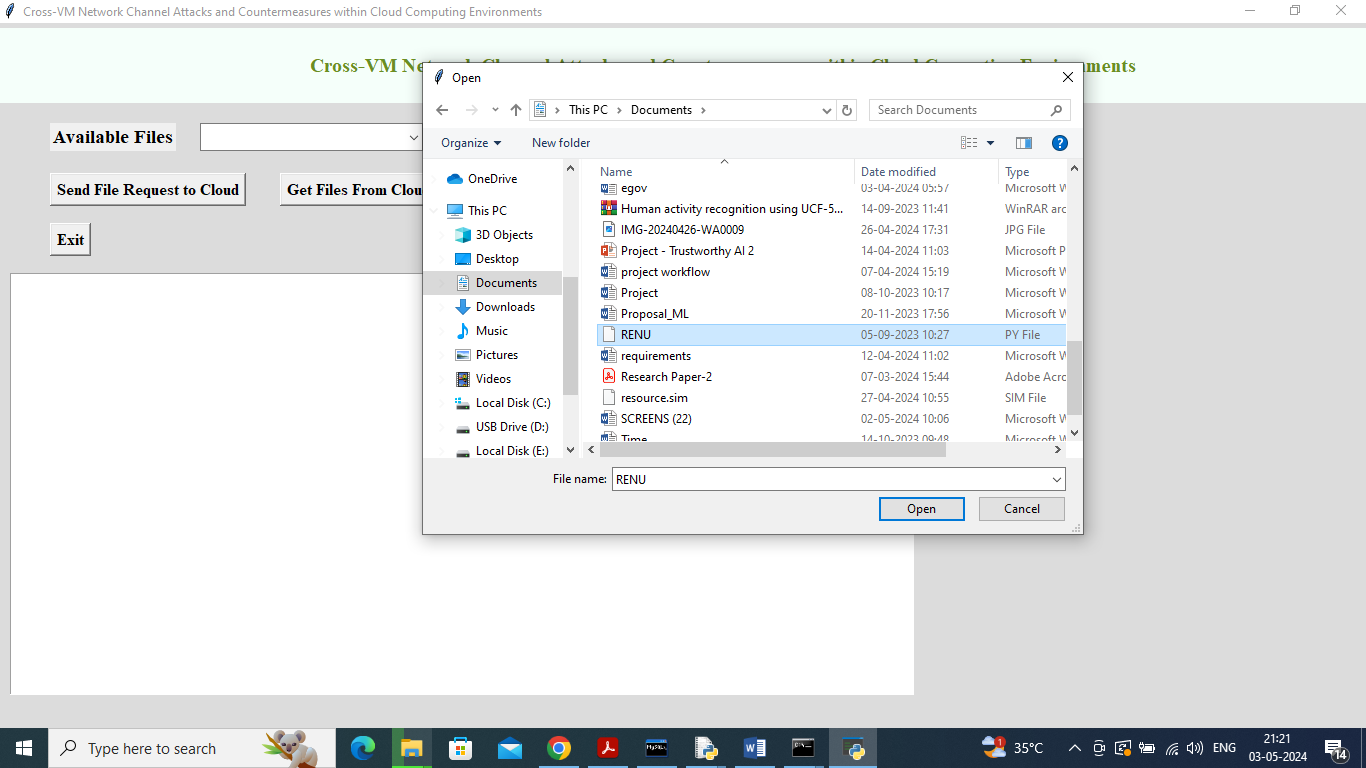
In above screen Cloud Server started and let it running and now double click on ‘runMonitor.bat’ file from ‘VM Monitor’ folder to start monitor and get below screen



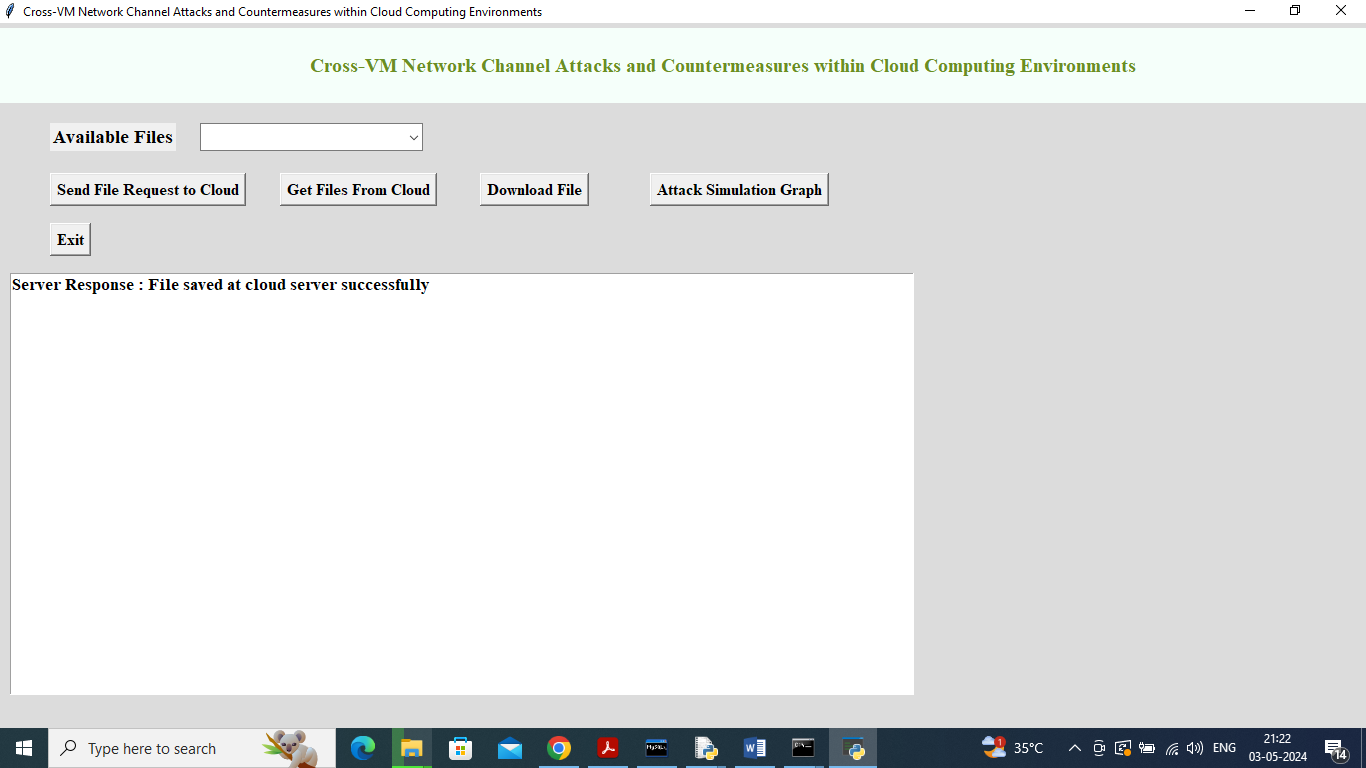
In above screen Monitor started and let it running and now double click on ‘run.bat’ file from ‘Simulation Node’ folder to start simulation and get below screen



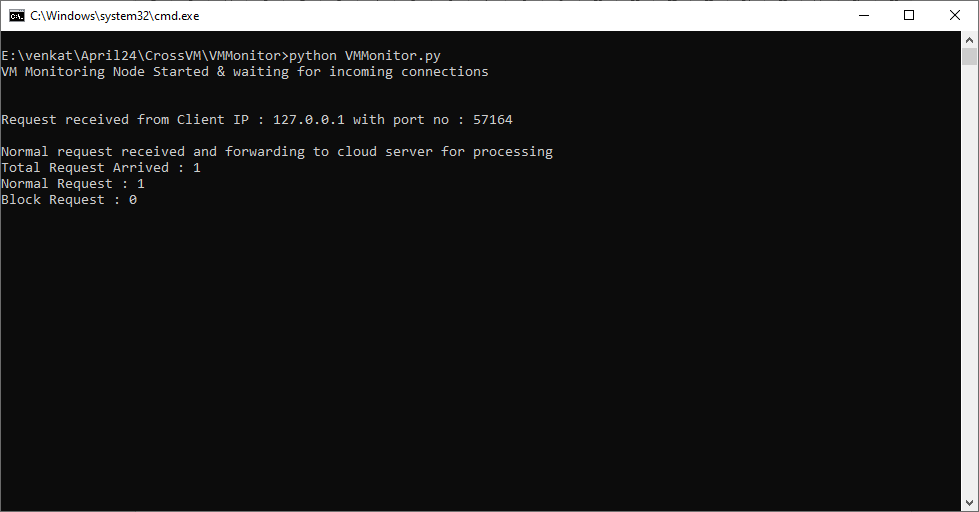
In above screen cloud user as simulation screen started and now click on ‘Send Request to Cloud’ button to send file to cloud and get below page



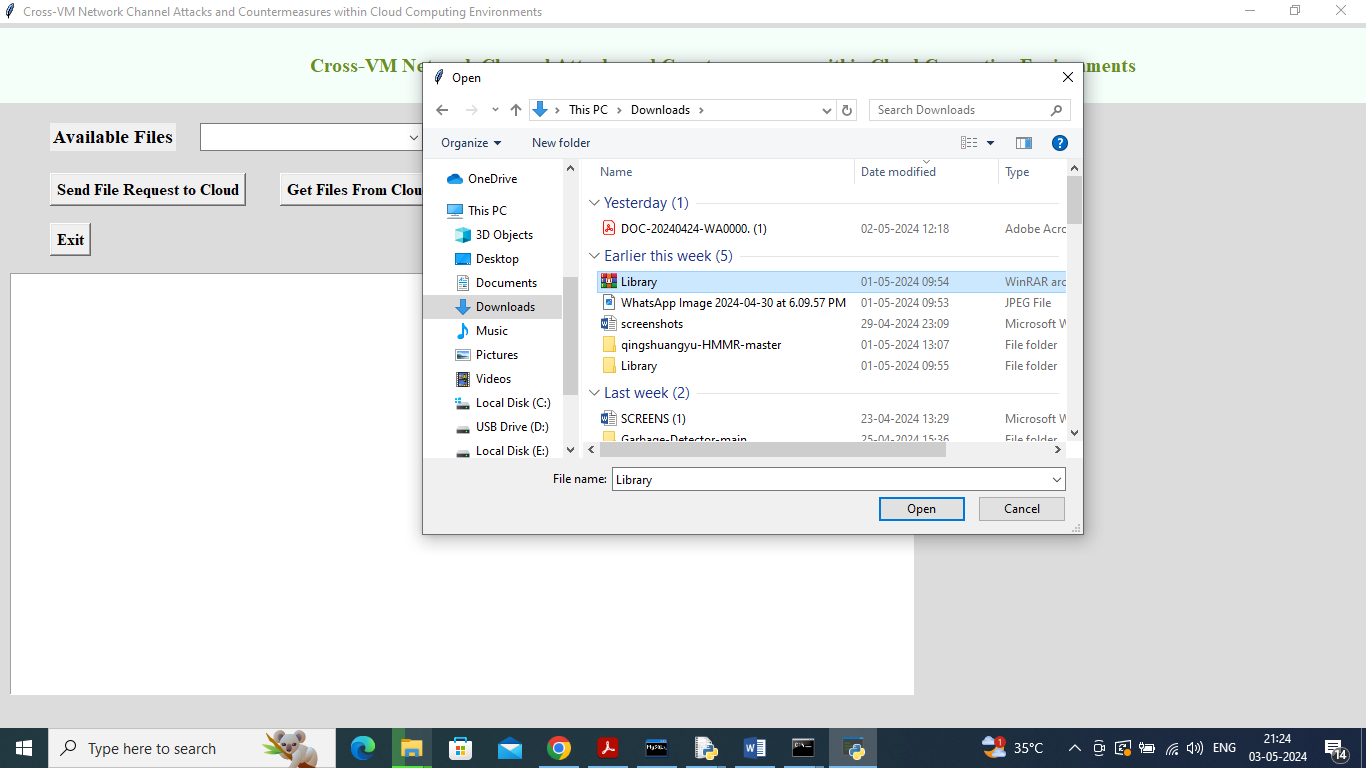
In above screen selecting and uploading a file and then click on ‘Open’ button send file to cloud which will monitor by VM and then will get below output



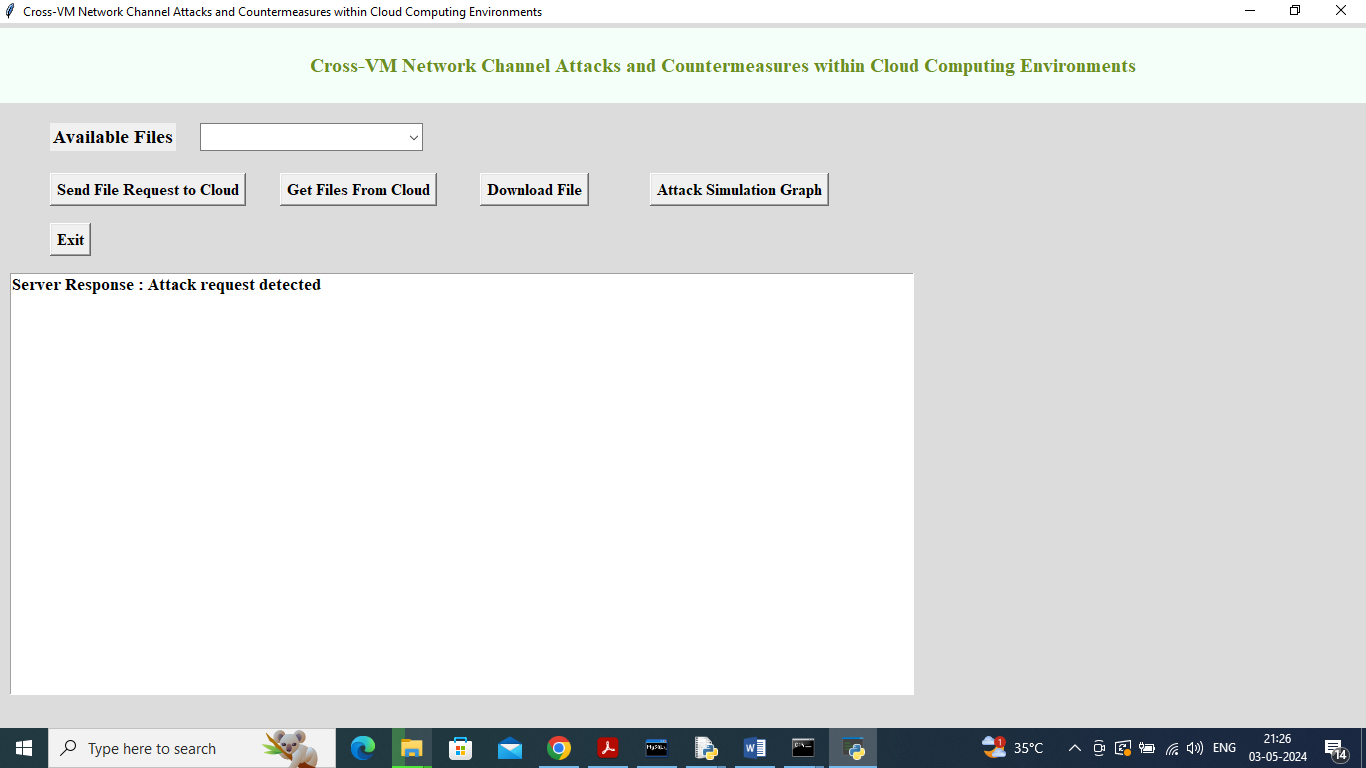
In above screen we got response from cloud as file saved successfully and below is the monitor response



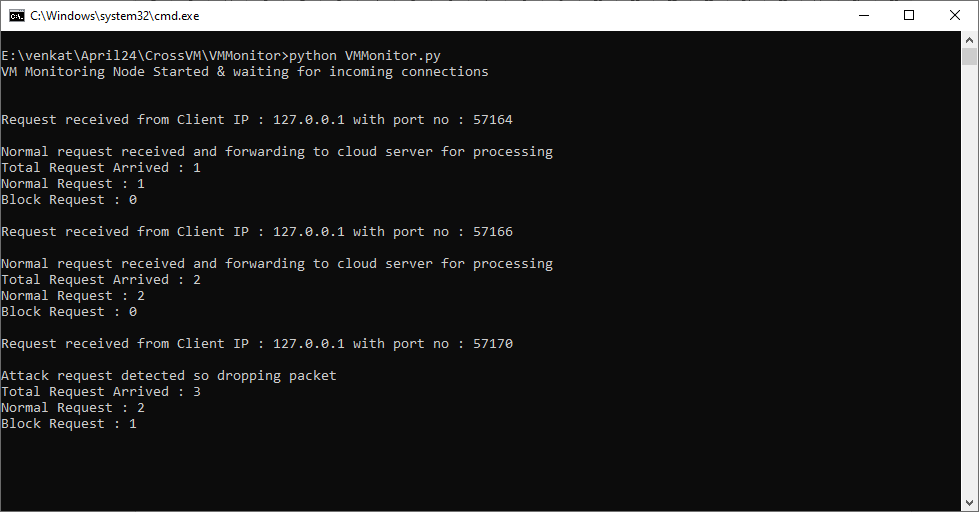
In above screen Monitor consider request as normal and Block count is 0 and now upload another file



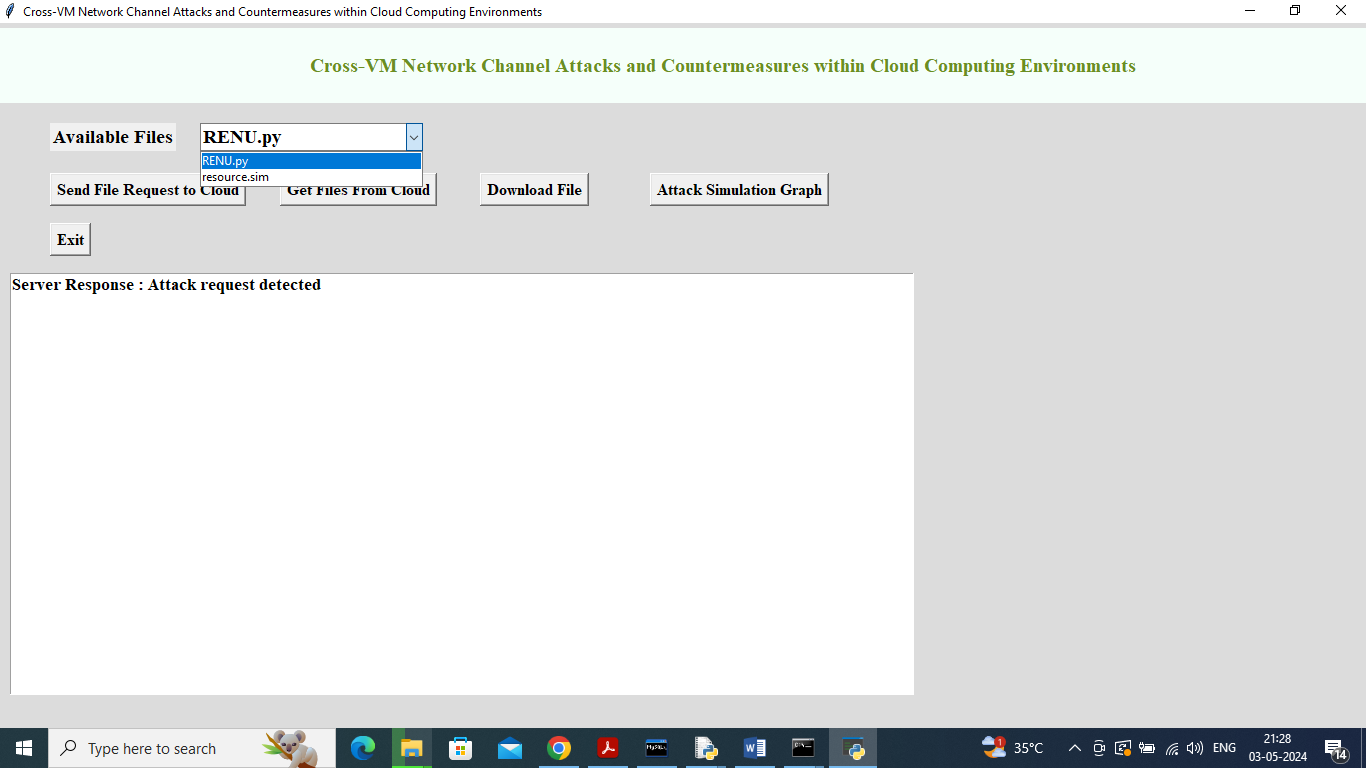
In above screen uploading huge zip file intentionally to crash cloud server and this will monitor by VM monitor and drop the request and get below output



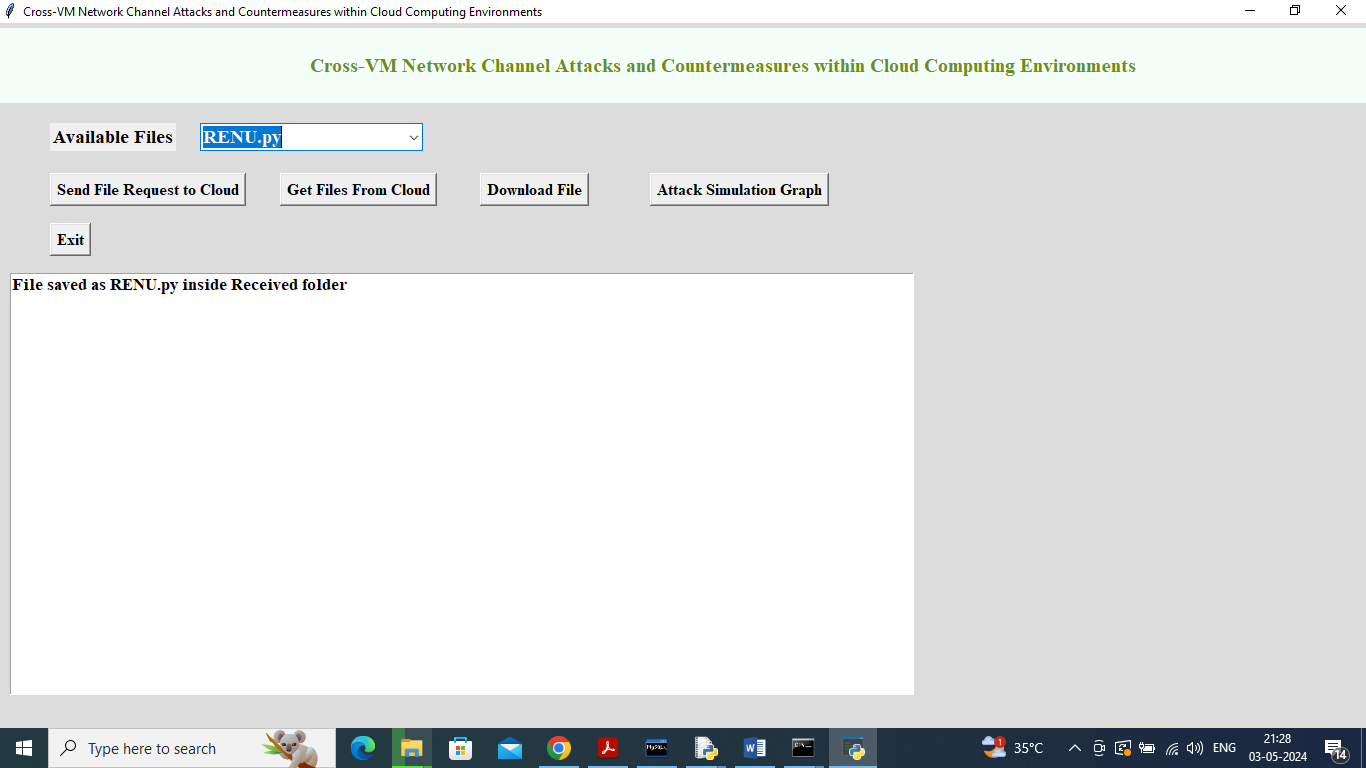
In above screen cloud monitor detected request as ATTAACK and then drop that request to save user data and below is the monitor response



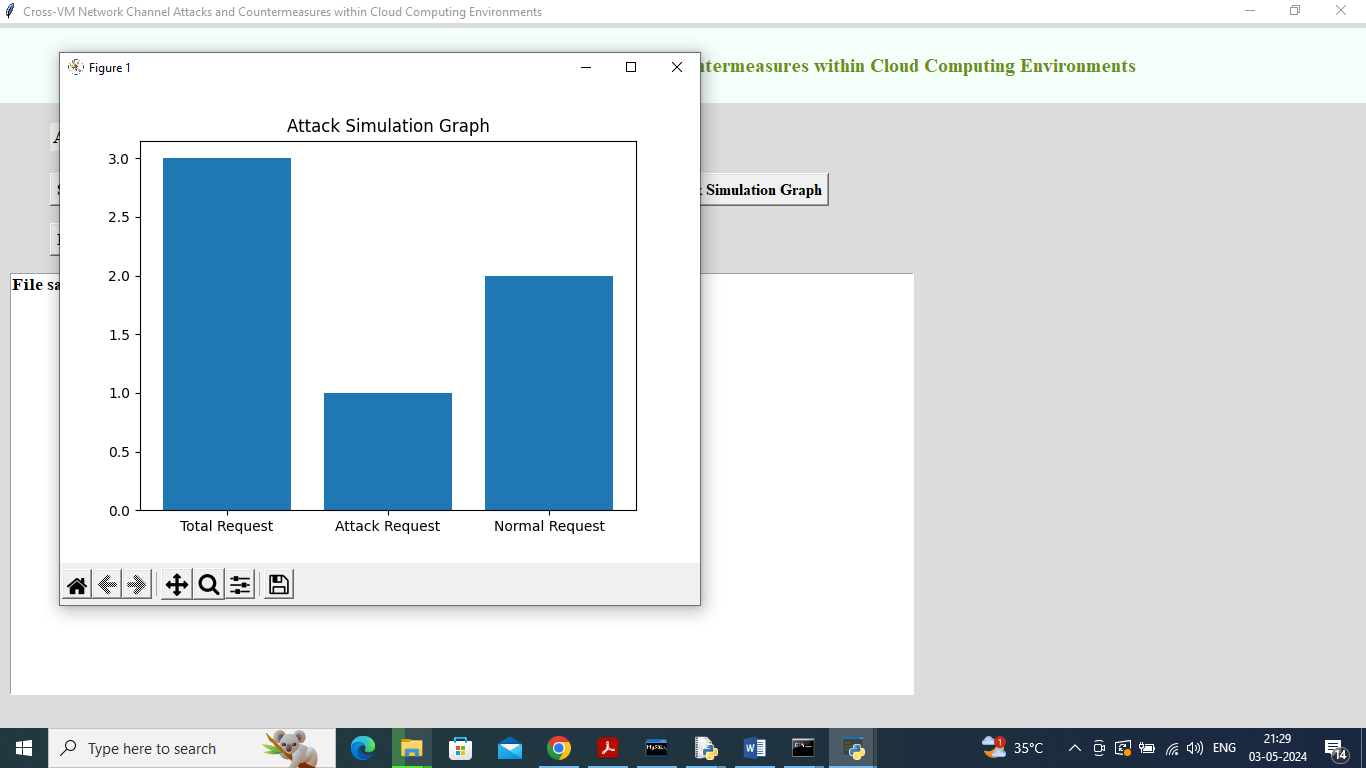
In above screen monitor saying 2 Normal request arrived and 1 attack request arrived and in below screen click on ‘Get Files from Cloud’ button to get below page



In above screen can get all file list from cloud and user can selected desired file and then click on ‘Download File’ button to get below output



In above screen can see file downloaded and saved inside Received folder and now click on ‘Attack Simulation Graph’ button to get below graph



In above graph x-axis represents total, attack and normal request type and y-axis represents count and in above graph can see how many request MONITOR received and how many are normal and attacks.

**8. CONCLUSION**

This research demonstrates two successful zero-day crossVM network attacks within a leading cloud platform i.e OpenStack. The first attack applies a combination of impersonating a TAP interface and a network mirror in the bridge interface. The consequence of this combination allows attackers to successfully redirect and monitor target VM network traffic within the same physical machine unbeknownst to customers. The second attack exploits ROP in conjunction with a network channel for the escalation of the privilege level of non-root VMs. This exploitation allows a non-root VM to make a connection with the root VM and control Tool Stack from where it can manage other co-located VMs. The countermeasure solutions of these two zero-day attacks have also been presented. They block penetration of any external device into the system and properly scrutinize the connection request before granting the root connection. We have highlighted the challenge for cloud providers to observe and detect such attacks due to an attacking VM which neither violates assigned VM resource capacity nor it makes any illegal connection with the root user for privilege escalation. Future work will focus on improving the current heuristics that prevent penetration of external devices into the network and also observe the request for root-connection. Furthermore, we will investigate how to overcome the challenges of distinguishing between normal resource patterns and target attacks

**9.REFERENCES**

[1] A. Seshadri et al., “Secvisor: A tiny hypervisor to provide lifetime kernel code integrity for commodity oses,” in ACM SIGOPS Operating Systems Review, vol. 41, pp. 335–350, ACM, 2007.

[2] <http://cve.mitre.org/cgibin/cvename.cgi?name=CVE-2008-0923>.

[3] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, “Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds,” in Proceedings of the 16th ACM conference on Computer and communications security, pp. 199–212, ACM, 2009.

[4] Z. Wu, Z. Xu, and H. Wang, “Whispers in the hyper-space: High speed covert channel attacks in the cloud.,” in USENIX Security symposium, pp. 159–173, 2012.

[5] J. Rutkowska, “Subverting vistatm kernel for fun and profit,” Black Hat Briefings, 2006.

[6] D. Hyde, “A survey on the security of virtual machines,” Dept. of Comp. Science, Washington Univ. in St. Louis, Tech. Rep, 2009. [7] S. R. Kumari and V. Kathiresan, “Virtual environment security considerations & practices,” Networking and Communication Engineering, vol. 3, no. 2, pp. 87–92, 2011.

[8] S. Zhang, “Deep-diving into an easily-overlooked threat: Inter-vm attacks,” tech. rep., Technical Report). Manhattan, Kansas: Kansas State University, 2012.

[9] A. Bates, Mood, et al., “On detecting co-resident cloud instances using network flow watermarking techniques,” International Journal of Information Security, vol. 13, no. 2, pp. 171–189, 2014.

[10] V. Varadarajan et al., “A placement vulnerability study in multitenant public clouds,”

[11] Adobe, “Adobe systems. security advisory for flash player, adobe reader and acrobat: Cve-2010-1297..” http://www.adobe.com/support/security/advisories/apsa10 -01.html, 2010.

[12] S. Ragan. http://www.thetechherald.com/articles/Adobeconfirms-Zero-Day-ROP-used-to-bypass-Windowsdefenses/11273/.

[13] R. Hund, T. Holz, and F. C. Freiling, “Return-oriented rootkits: Bypassing kernel code integrity protection mechanisms,” in USENIX Security Symposium, pp. 383–398, 2009.

[14] C. Modi, D. Patel, B. Borisaniya, A. Patel, and M. Rajarajan, “A survey on security issues and solutions at different layers of cloud computing,” The journal of supercomputing, vol. 63, no. 2, pp. 561– 592, 2013.

[15] A. Saeed et al., “A cross-virtual machine network channel attack via mirroring and tap impersonation,” in IEEE International Conference on Cloud Computing (CLOUD), pp. 606–613, IEEE, 2018.

[16] H. Wu, Y. Ding, C. Winer, and L. Yao, “Network security for virtual machine in cloud computing,” in Computer Sciences and Convergence Information Technology (ICCIT), 2010 5th International Conference on, pp. 18–21, IEEE, 2010.

[17] Q. Zhang, L. Cheng, and R. Boutaba, “Cloud computing: stateof-the-art and research challenges,” Journal of internet services and applications, vol. 1, no. 1, pp. 7–18, 2010.

[18] S. J. Murdoch and S. Lewis, “Embedding covert channels into tcp/ip,” in International Workshop on Information Hiding, pp. 247– 261, Springer, 2005.

[19] P. Ranjith, C. Priya, and K. Shalini, “On covert channels between virtual machines,” Journal in Computer Virology, vol. 8, no. 3, pp. 85–97, 2012.

[20] N. Q. Z. Z. A. D. V. G. Jonathan M. McCune, Yanlin Li and A. Perrig., “Trustvisor: Efficient tcb reduction and attestation. in ieee symposium on security and privacy, 2010..”

[21] T. Shinagawa et al., “Bitvisor: a thin hypervisor for enforcing i/o device security,” in Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments, pp. 121– 130, ACM, 2009.

[22] U. Steinberg and B. Kauer, “Nova: a microhypervisor-based secure virtualization architecture,” in Proceedings of the 5th European conference on Computer systems, pp. 209–222, ACM, 2010.

[23] A. M. Azab, P. Ning, Z. Wang, X. Jiang, X. Zhang, and N. C. Skalsky, “Hypersentry: enabling stealthy in-context measurement of hypervisor integrity,” in Proceedings of the 17th ACM conference on Computer and communications security, pp. 38–49, ACM, 2010.

[24] Z. Wang and X. Jiang, “Hypersafe: A lightweight approach to provide lifetime hypervisor control-flow integrity,” in Security and Privacy (SP), 2010 IEEE Symposium on, pp. 380–395, IEEE, 2010.

[25] J. Wang, A. Stavrou, and A. Ghosh, “Hypercheck: A hardware assisted integrity monitor,” in International Workshop on Recent Advances in Intrusion Detection, pp. 158–177, Springer, 2010.

[26] DEP, “Microsoft. data execution prevention (dep). http://support.microsoft.com/kb/875352/en-us/, 2006.”

[27] J. Zhan, X. Fan, J. Han, Y. Gao, X. Xia, and Q. Zhang, “Ciadl: cloud insider attack detector and locator on multi-tenant network isolation: an openstack case study,” Journal of Ambient Intelligence and Humanized Computing, pp. 1–23, 2019.

[28] T. Madi, Y. Jarraya, A. Alimohammadifar, S. Majumdar, Y. Wang, M. Pourzandi, L. Wang, and M. Debbabi, “Isotop: auditing virtual networks isolation across cloud layers in openstack,” ACM Transactions on Privacy and Security (TOPS), vol. 22, no. 1, pp. 1–35, 2018.

[29] B. Ding, Y. Wu, Y. He, S. Tian, B. Guan, and G. Wu, “Return oriented programming attack on the xen hypervisor,” in 2012 Seventh International Conference on Availability, Reliability and Security, pp. 479–484, 2012.

[30] https://www.wireshark.org/.

[31] H.-C. Li et al., “Analysis on cloud-based security vulnerability assessment,” in e-Business Engineering (ICEBE), 2010 IEEE 7th International Conference on, pp. 490–494, IEEE, 2010.

[32] V. Iozzo, “Ralf-philippweinmann&vincenzoiozzo own the iphone at pwn2own,” 2010.

[33] B. Ding et al., “Return-oriented programming attack on the xen hypervisor,” in Availability, Reliability and Security (ARES), 2012 Seventh International Conference on, pp. 479–484, IEEE, 2012.

[34] M. Schwarz et al., “Zombieload: Cross-privilege-boundary data sampling,” arXiv preprint arXiv:1905.05726, 2019.

[35] A. Moghimi et al., “Memjam: A false dependency attack against constant-time crypto implementations,” International Journal of Parallel Programming, vol. 47, no. 4, pp. 538–570, 2019.

[36] <https://www.openstack.org/>.

[37] https://azure.microsoft.com/.

[38] B. Ding, F. Yao, Y. Wu, and Y. He, “Improving flask implementation using hardware assisted in-vm isolation,” in IFIP International Information Security Conference, pp. 115–125, Springer, 2012.

[39] OpenStack, “OpenStack Networking (”Neutron”).” [Accessed: Jan-18].

[40] https://www.xenproject.org.

[41] https://wiki.openstack.org/wiki/XenServer/XenAndXenServer.

[42] L. Davi and others, “Privilege escalation attacks on android,” in international conference on Information security, pp. 346–360, Springer, 2010.

[43] <https://www.paessler.com/prtg>.

[44] A. Kozlowski, “Comparative analysis of cyberattacks on estonia, georgia and kyrgyzstan,” European Scientific Journal, ESJ, vol. 10, no. 7, 2014.

[45] B. R. Raghunath et al., “Network intrusion detection system (nids),” in International Conference on Emerging Trends in Engineering and Technology. ICETET’08., pp. 1272–1277, IEEE, 2008.