Trust Evaluation Based on a Risk Assessment and Game Theory for Virtual Machines in a Cloud Environment

Victor Morales

CSCE-5585 Advanced Network Security
Denton, TX United States
victormoralesavalos@unt.edu

***Abstract*—Abstract This research focuses on the trust evaluation of Virtual Machines (VMs) within cloud environments to enhance secure access to critical resources. It explores cloud computing fundamentals and addresses the challenges related to trust and security in these dynamic ecosystems. While trust evaluation has been a topic of extensive study across various technological contexts, this work applies a comprehensive trust model tailored for virtualized cloud infrastructures. Vulnerability assessments, risk analysis, and simulations were performed to derive trust metrics and enable the identification of reliable VMs. The findings highlight the impact of specific metrics on trust evaluation, with results indicating that historical behavior patterns and risk factors are critical in determining VM reliability. The proposed trust model effectively supports the selection of VMs that maintain higher security standards, improving overall cloud system reliability.**

***Index Terms*—**

I. Introduction

Cloud Computing has significantly changed how organizations store and process their data, providing them with scalable resources and flexibility to manage their IT infrastructure cost-effectively. Organizations from different industries have migrated large amounts of data into the cloud. However, it has introduced significant security concerns related to accessibility, integrity, privacy, and trust. Therefore, trust evaluation within cloud environments has become critical in securing cloud-based resources and identifying reliable resources. Despite various security measures, new challenges continue in ensuring the trustworthiness of Virtual Machines (VMs), which are the main components in these dynamic and complex environments [1].

According to Markets and Markets, the global cloud computing market is projected to expand by 15.1% annually, increasing from $626.4 billion in 2023 to $1,266.4 billion by 2028 [2]. This growth highlights the increasing reliance on cloud environments to store, process, and manage critical data and applications. Also, new services and architectures are being introduced continuously by Cloud providers [3]. In addition, according to a report by Goldman Sachs, global Cloud profits are projected to reach $2 trillion by 2030, highlighting this sector's continued expansion. This growth is also attributed to adopting AI across different industries, which requires more computational power and scalable infrastructure, which are capabilities provided by cloud services. Moreover, the continuous provisioning, migration, and decommissioning of VMs in cloud platforms presents a unique challenge: maintaining a consistent level of trust in an environment marked by dynamic changes. As organizations frequently add or remove VMs, they introduce potential vulnerabilities and points of failure, increasing the likelihood of security breaches. The dynamic nature of cloud computing further complicates trust management as cyber threats evolve quickly alongside the technology. A single breach could compromise sensitive information and trust as a secure and reliable platform.

The need for a comprehensive trust management framework becomes evident as cloud environments become increasingly susceptible to attacks. Trustworthiness in a cloud setting is essential to safeguarding the confidentiality and integrity of information and ensuring that cloud computing remains a dependable and secure platform for businesses. Consequently, trust evaluation mechanisms that account for risk assessments, dynamic vulnerabilities, and adaptable defense strategies are critical for maintaining security and trustworthiness in cloud infrastructure.

This paper focuses on developing a trust evaluation model for VMs in a cloud environment. Trust evaluation has been extensively explored in various computing contexts, but its application to cloud infrastructures remains an evolving study area. This research utilizes vulnerability assessments and risk analyses to determine trust metrics for VMs, distinguishing more reliable machines from potentially compromised ones. By employing a risk-based approach and game theory principles, this study contributes to the field by proposing a dynamic and adaptive trust model that can effectively respond to changing security conditions within a cloud infrastructure. To address these challenges, this research proposes to implement risk assessment with game theory to develop a dynamic and adaptive trust evaluation model for virtual machines in a cloud environment. The proposed model will simulate interactions between virtual machines and cloud administrators, effectively capturing the evolving nature of threats and providing a more robust mechanism for identifying and managing malicious behaviors, thereby enhancing cloud environments' overall trustworthiness and security.

II. Related Work

Trust evaluation, vulnerability assessment, and game-theoretic approaches have been studied to address security challenges in dynamic and distributed systems, including cloud environments. Therefore, this section reviews relevant works and highlights their contributions to the proposed model. Trust evaluation models have played a crucial role in enhancing the reliability of cloud services. Huang and Nicol [4] proposed a trust management system that utilizes evidence-based evaluations to assess the trustworthiness of cloud nodes dynamically. Their approach integrates multiple metrics, including service performance and historical behavior, to create adaptive trust models. Similarly, Wu and Zhang in [5] introduced a hierarchical trust framework for virtualized infrastructures, emphasizing the need for trust chains between physical servers, hypervisors, and virtual machines. Their work demonstrated the importance of trust propagation in ensuring secure resource allocation [5].

Data breaches have become inevitable for many organizations, making it imperative for corporate security teams to adopt a proactive approach to vulnerability management. By identifying and addressing potential weaknesses, companies can reduce the likelihood of cyberattacks exploiting systems and gaining unauthorized access. As cyber threats become more sophisticated, attackers leverage advanced techniques, tools, and strategies to infiltrate networks. This increasing complexity highlights the importance of preventive measures to safeguard critical assets. Proactively managing vulnerabilities reduces the chances of exploitation and helps protect the organization's overall stability and operations. Implementing a comprehensive vulnerability management program is one of the most effective ways to protect systems and sensitive information by prioritizing and resolving security issues before they are exploited, significantly reducing the risk and enhancing their ability to avoid potential attacks. Vulnerability assessment is critical in evaluating the security posture of virtual machines. Li et al. in [6] explored risk-based trust management systems incorporating real-time vulnerability assessments to adjust trust scores dynamically. Their use of system logs and security events to derive risk metrics provided a robust basis for trust evaluation. Moreover, Balasubramaniam and Byrappa in [7] presented a risk quantification model for cloud environments, focusing on identifying and mitigating vulnerabilities in real time. Their methodology included advanced monitoring tools to detect threats at multiple layers of the cloud architecture.

Game theory can be implemented as a robust framework for analyzing trust and decision-making in uncertain environments. Chiregi and Navimipour applied non-cooperative game theory to model interactions between cloud service providers and consumers, focusing on the dynamic nature of trust evolution. Their findings highlighted the utility of Nash equilibrium in predicting optimal trust strategies [8]. Expanding on this, Pawlick et al.in [9] proposed a multi-layered game-theoretic trust model for IoT-enabled cloud systems, emphasizing the interplay between risk management and trust evaluation. Their approach incorporated Perfect Bayesian Equilibrium to adapt trust calculations based on real-time observations.

Combining trust evaluation, vulnerability assessment, and game theory has produced promising results. For example, Sun et al. introduced a hybrid trust framework integrating risk assessment with game-theoretic decision-making. Their model dynamically updates trust scores based on observed behaviors and identified vulnerabilities, providing a comprehensive approach to trust management in cloud environments [10]. Similarly, Alam et al. developed a trust-aware resource allocation model that utilizes game theory to balance security and efficiency in multi-tenant cloud systems [11].

III. Methodology

This section describes the dynamic model for trust evaluation of VMs in a cloud environment, performing risk assessment, game theory, and Bayesian updates. By analyzing the trustworthiness of VMs over time and adapting strategies accordingly, the model would improve the security of a cloud environment, ensuring only trustworthy VMs are maintained while malicious VMs are identified and penalized or replaced.

A. Architecture

The architecture of the proposed trust evaluation model for Virtual Machines (VMs) in a cloud environment integrates vulnerability assessment, trust evaluation, and game-theoretic principles to enhance security and reliability. This section describes the key components and their interactions. This architecture is the cloud environment that comprises physical servers, hypervisors, and virtual machines. Physical servers are the backbone, hosting multiple Type-1 and Type-2 hypervisors as shown in Fig. 1. These hypervisors manage the allocation and operation of VMs. Each VM is assigned a unique profile, including its operational parameters, historical behavior, and vulnerability attributes. This infrastructure supports dynamic resource allocation and scalability while serving as the basis for trust evaluation [5].

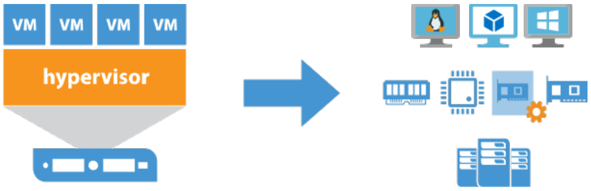


Fig. 1. Virtual Machines overview

The risk assessment module identifies and quantifies vulnerabilities associated with each VM. This process involves analyzing system logs, monitoring traffic patterns, and evaluating application-level security. Metrics such as the number of unresolved vulnerabilities, exposure to attacks, and frequency of updates are used to compute a risk score for each VM. This score is dynamically updated to reflect changes in the security posture of the VM, ensuring real-time adaptability [6].

The trust evaluation engine assigns and updates trust scores for each VM based on its risk score and observed behavior. Trust values range from 0.1 (trusted) to 1.0 (not trusted). The engine employs Bayesian inference to iteratively refine trust scores as new data is collected. Reward and penalty mechanisms are integrated to account for consistent or abnormal behavior. For instance, VMs exhibiting consistent, secure behavior receive incremental trust rewards, while those flagged for security incidents incur penalties [4]. The trust evaluation process incorporates a game-theoretic approach to model interactions between the cloud administrator and VMs. Each VM is treated as a player in a repeated game, where the administrator observes and adjusts strategies based on the VMs' behavior. Bayesian updates are used to manage uncertainty and adapt to dynamic conditions. This approach ensures that trust evaluations are context-aware and reflect the evolving state of the system [8].

The decision-making module utilizes trust scores to classify VMs into trusted and malicious groups. Trusted VMs are allocated critical tasks and resources, while untrusted ones are isolated or subjected to stricter monitoring. This module gives administrators actionable insights to guide security interventions, such as vulnerability patching or reconfiguration. To validate the architecture, simulations mimic real-world scenarios, including the dynamic addition and repair of vulnerabilities. Visualization tools display trust scores, risk levels, and system states over time, enabling administrators to assess the model's effectiveness and make data-driven decisions. This architecture combines a robust risk assessment framework with adaptive trust evaluation techniques to provide a secure and scalable solution for managing VMs in cloud environments. Its modular design allows for future integration of advanced technologies such as blockchain and machine learning, further enhancing its capabilities.

B. Trust Model

The trust model we propose is designed to evaluate the trustworthiness of Virtual Machines (VMs) in a cloud environment. The approach begins with a vulnerability assessment in VMs using tools such as Nessus, Rapid 7, and OpenVAS to identify vulnerabilities from the Common Vulnerabilities and Exposures (CVE) database, which allows us to elaborate a risk analysis based on the assessment results. The scenario includes a cloud environment with multiple VMs installed across various physical servers. The VMs can behave as trusted, showing regular operations or malicious, demonstrating an abnormal behavior. Therefore, the goal of the model is to evaluate and classify the VMs based on their behavior over time using a combination of vulnerability assessment, game theory, and Perfect Bayesian Equilibrium (PBE), and dynamically assess trustworthiness and identify malicious VMs. In addition, every VM has attributes such as initial trust, risk value, vulnerabilities, utility, and strategies. Trust values are defined by sending queries to the private cloud, considering the security risk values and resources. Every new virtual machine will start with a trust value of 0.5, also determined as a threshold. The trust will be measured from $0 \leq T \leq 1$, defined as follows:

$$T = \{t_1, t_2, t_3, ..., t_k\} \tag{1}$$

Assuming that will have N virtual machines VM as part of the cloud environment, where they are defined as players:

$$Players = \{player_1, player_2, player_3, ..., player_k\} \tag{2}$$

The game theory in this paper shows that every VM starts with a predetermined trust value of 0.5. Also, each selects a strategy value from 0.1 to 1 based on the data obtained [12]. The selected strategies by players are sent to the administrator as a strategy set Eq. 3:

$$S = \{s_1, s_2, s_3, ..., s_k\} \tag{3}$$

If a virtual machine needs to calculate a trust value, it will verify the information from prior knowledge stored in the trust information. Then, the administrator will assign a trust value to the virtual machines analyzed according to the vulnerability assessment and trust analysis. So, it will be assumed that virtual machines would be assigned a greater value and penalized if the required attributes are missing.

Each player's primary objective is to minimize their utility, which reflects the degree of deviation in their chosen strategies, thereby maximizing their overall reward. The utility function integrates two distinct deviation components to compute the utility values for all players, focusing on the current strategies and beliefs of each VM. The first component, the strategy deviation, evaluates how a VM's strategy S_i diverges from the average strategy or probability distribution of the other VMs in the cloud, excluding VM_i and considering the $k1$ remaining players. This deviation highlights how much a VM's behavior varies from its peers. The second component, the prior deviation function, uses Bayesian principles to incorporate prior beliefs. It quantifies the variance between a VM's observed behavior and the expected conditional probability based on predefined conditions and the consistency of its peers' actions [12].

C. Game Description

The game theory approach involves multiple players (VMs and cloud administrators) participating in the dynamic game over time. The trust evaluation uses strategies, utility functions, and perfect Bayesian equilibrium concepts. The players are described below:

1) VMs - Every VM can use two strategies, ”Normal operations” or ”Malicious Operation”.
2) Cloud administrator - they monitor and adjust the trust levels based on observed actions. The strategy is ”Defend” or ”Not Defend” based on identifying a probability of attack.

As mentioned above, at the start of the game, each VM begins with a default trust level of 0.50; then, they are assigned multiple vulnerabilities, establishing a matrix to assign and track the vulnerabilities across all VMs; each cell in the matrix represents a VM to represent a diverse risk profile. During the simulation, VMs should share their strategies with the cloud administrator. VMs within the accepted range of utility values would be considered trusted, while VMs out of the range would be classified as malicious [12]. Each VM should decide on a strategy for malicious actions or expected operations during every iteration. VMs' decisions will be influenced by the VM's trust level, the risk assigned to them, and their previous behavior; when a VM's utility is low, it suggests reliability so that the trust level will increase on a small scale. If the utility value is high, the trust will be reduced if it shows untrustworthy behavior.

$$\text{Utility} = W_{\text{Risk}} \cdot (I_{\text{Risk}} + 0.003 \times \text{V}) + W_{\text{B}} \cdot (1 - \text{B}) \tag{4}$$

Where W_{Risk} and W_B are weights assigned to risk and behavior consistency, respectively.

In addition, a VM's trust level is updated based on its utility value and behavior. If the utility is low, close to 0, the trust is increased, but if the utility is high (close to 1), the trust is decreased. Penalties are applied for consistent malicious actions.

$$\text{Trust} = \begin{cases} \min(\text{T}_{\text{MAX}}, \text{T} + r) & \text{if Utility} \leq 0.01 \\ \max(\text{T}_{\text{MIN}}, \text{T} - pn) & \text{if Utility} \geq 1.0 \end{cases} \tag{5}$$

Where r is the reward increment for trustworthy behavior, and pn is the penalty decrement for untrustworthy behavior.

The cloud administrator uses Bayesian update analysis to calculate and adjust beliefs about the probability of VM attacks based on their observed actions. The belief update is calculated with the following equation:

$$\text{Beliefs}_{\text{Attack}} = \frac{\text{Beliefs}_{\text{Attack}} + \text{Obs}_{\text{attack\_prob}}}{2} \tag{6}$$

The beliefs about the VM's behavior are updated after every iteration, and the probability of malicious behavior is adjusted based on the observed actions. VMs with decreasing trust levels below a specific threshold will be replaced with new trusted VMs.

The perfect Bayesian equilibrium would be accomplished when the cloud administrator strategy of ”Defend” or ”Not Defend” is optimal given the updated beliefs. Also, every VM's strategy, whether trusted or malicious, reaches its expected utility, given that the administrator strategies and trust levels are updated.

In our case, the outcome over time (100 iterations) shows that VMs with consistent malicious behavior are penalized and replaced if their trust falls below the required threshold. Additionally, the trust levels of VMs fluctuate according to their action and the cloud admin belief updates, allowing the trust model to adjust to behaviors dynamically.

The trust model should immediately identify this compromised VM based on its initial behavior to handle a compromised VM with high risk or a VM that starts performing malicious actions. Since the VM starts performing malicious actions, its utility value will likely exceed the established thresholds. The trust level should decrease until the untrusted VM is replaced with a new one after a few iterations.

It also includes levels of compromise VMs, such as Low compromise, Moderate compromise, and high compromise. This organization allows a better understanding of the VMs' behavior.

1) Low compromise - minimal malicious actions or low risk for vulnerabilities.
2) Moderate compromise - VMs with normal and malicious actions or medium vulnerabilities.
3) High compromise - VMs showing malicious actions and critical vulnerabilities.

D. Vulnerability Assessment in Virtual Machines Environments

Virtual Machines (VMs) are essential to modern data center and cloud infrastructures due to their scalability, flexibility, and resource efficiency, but they also face risks that can compromise data integrity and system performance. A comprehensive risk and threat assessment process is crucial to safeguarding these environments. Risk identification involves addressing vulnerabilities across all infrastructure layers, including operating systems, by applying timely patches, implementing antivirus tools to prevent malware infections, managing access permissions to prevent unauthorized changes, and ensuring robust backup strategies to mitigate data loss. Threat assessment evaluates potential exploits, such as mitigating denial-of-service (DDoS) attacks through traffic management, enforcing strong password policies to prevent brute force attacks, securing applications against code injection, and applying least-privilege principles to curb privilege escalation risks. To mitigate these risks, businesses can adopt strategies such as network segregation to contain malicious activities, enforcing comprehensive security policies, and deploying advanced monitoring tools to detect and respond to threats. Organizations can secure their virtual environments and ensure uninterrupted operations by systematically identifying vulnerabilities, analyzing threats, and implementing mitigation measures.

The vulnerability scores are obtained from an online CVE database. For every vulnerability (CVE IDs), we gather three scores: the base score, which measures the overall severity of a vulnerability; the impact score, which represents the potential damage if the vulnerability is exploited; and the exploitability score, which shows how easy it is to exploit a vulnerability.

TABLE I
CVSS SCORE AND RISK MAPPING

Rate	CVSS Score	Risk = Score /10
None	0	0
Low	0.1 – 3.9	0.01 – 0.39
Medium	4.0 – 6.9	0.4 – 0.69
High	7.0 – 8.9	0.7 – 0.89
Critical	9.0 - 10	0.9 - 1

In addition, the table I the risk analysis follows the Common Vulnerability Scoring System (CVSS), a framework designed to score known vulnerabilities to determine how severe a security vulnerability could be [13]. After those values are obtained, the risk level is calculated using the following Eq. 7:

$$\text{Risk Level} = \sqrt{\text{Impact Score} \times \text{Exploitability Score}} \tag{7}$$

The Eq. 7 combines the impact and exploitability scores to calculate a single value for a risk level, reducing the effect of high values and normalizing the risk level. Then, the risks are categorized into several levels based on the impact and the probability to determine how critical a vulnerability could be and how it could affect the trust level of a VM. This risk level is then adjusted over time using exponential smoothing to reduce the fluctuation in the risk levels over time, as described in the equation 8.

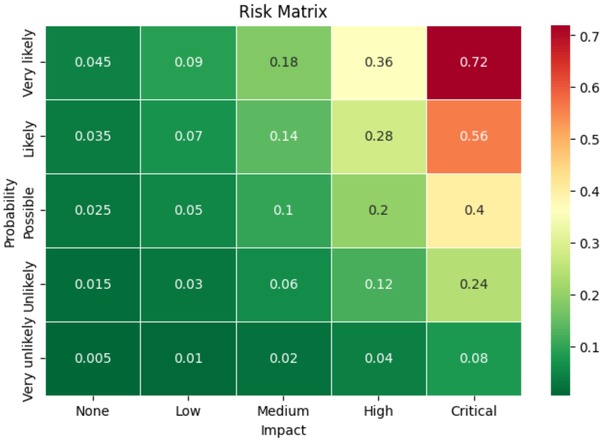


Fig. 2. Risk Matrix

Exponential smoothing equation:

$$S_t = \alpha \cdot R_t + (1 - \alpha) \cdot S_{t-1} \tag{8}$$

S_t is the smoothed risk value at time t, R_t. Also, $alpha$ is the smoothing factor (between 0 and 1), and S_{t-1}, is the smoothed risk value from a previous time. This analysis will help prioritize recent risk levels while still considering past data. $alpha$ determines how quickly the model reacts to changes in the risk level.

E. Scenario Simulation

Traditional Data Centers with a hierarchical architecture of three layers started to present problems as they grew, such as latency, performance, and inadequate response to applications that required scalability and performance. Nowadays, data center infrastructure has been modernized to a hierarchical two-layer spine and leaf model, which brings improved speed and simplicity and supports new communication protocols like VxLAN that are suitable for multi-tenant infrastructure. Spine and Leaf improve the communications performance of servers that hyper-convergence (propagation latency and processing latency), allowing the segmentation of the network and improving the scalation limitation of VLANs. This technology allows the creation of 16 million VxLANs in one administrative domain. On the contrary, it is only possible to create 4094 VLANs, supporting extensive networks and many tenants. In addition, the migration of VMs can be tunneled over a layer three network, allowing for the allocation of resources dynamically in the data center or through a wide area, avoiding the limitations of layer two networks [14].

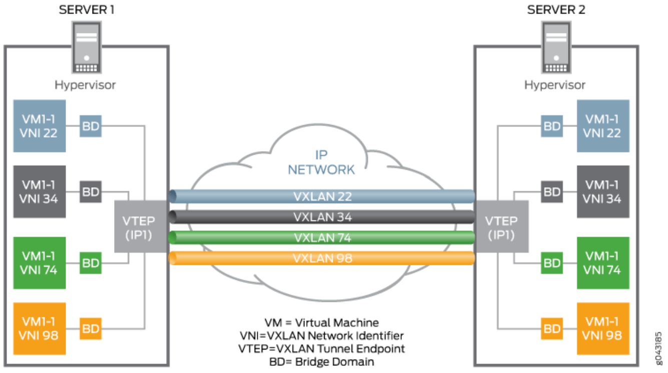


Fig. 3. VxLAN overview, Juniper Networks

Ethernet Virtual Private Network (EVPN) is a modern, standards-based approach to establish virtual multipoint connectivity across Layer 2 domains over IP or IP/MPLS backbone infrastructures. Like traditional VPN solutions, EVPN instances (EVIs) are deployed on Provider Edge (PE) routers to ensure logical separation of customer services. The PE routers interface with Customer Edge (CE) devices, including routers, switches, or VMs. Connectivity information is exchanged between PE routers using Multi-Protocol Border Gateway Protocol (MP-BGP), and traffic is encapsulated and transmitted across the backbone network. Attributable to its architectural similarities with other VPN technologies, EVPN offers straightforward integration into existing service ecosystems and solutions. Cloud providers also use this technology to provide multi-tenancy and flexibility that can be extended, so EVPN and BGP provide dynamic migrations of VMs, and is also known as VM motion. EVPN, uses control plane-based MAC address learning via MP-BGP, which distinguishes it from traditional solutions like VPLS that rely on data plane flooding. This approach enables advanced features, flexibility in data plane encapsulations, and seamless integration into enterprise networks. EVPN is widely adopted for Data Center Interconnect (DCI), extending Layer 2 connectivity across data centers for improved performance and disaster recovery [14].

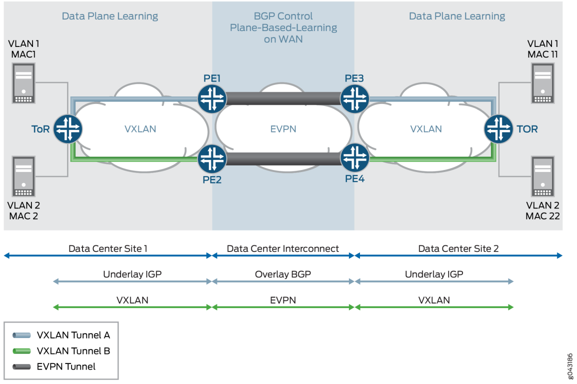


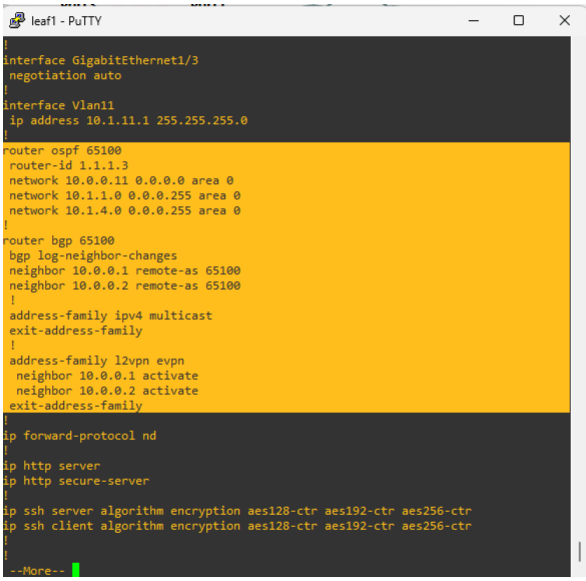
Fig. 4. VxLAN-EVPN integration overview, Juniper Networks

The integration of VXLAN with EVPN combines the benefits of both technologies. VXLAN overlays are mapped to bridge domains at PE devices, with MAC addresses distributed through EVPN over MP-BGP. The network packets are encapsulated with VXLAN headers, ensuring robust and flexible connectivity. Therefore, EVPN-VXLAN is an ideal solution for modern data center and virtualization challenges [14].

The following tools have been used to perform the simulation:

- GNS3.
- Cisco IOS CSR100v that supports EVPN and VxLAN implemetation.
- Fortinet Firewall
- VMWare Workstation.
- Virtual Box.
- Rapid7.
- Greenbone.
- Nessus.
- Pfsense Firewall, and Suricata Intrusion Prevention System.
- Cumulus

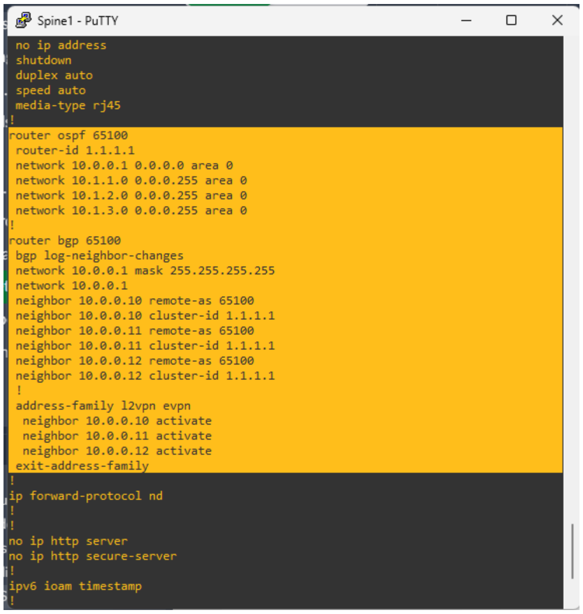
The cloud environment consists of hosts in GNS3; each hosting VM is configured to be part of Vlan 11 and Vlan 12 includes the Web servers and database servers described above. Vlan 10 consists of the VMs, tools, and systems for vulnerability assessment. The architecture is defined as shown in Fig. 5, where all the nodes and networks are labeled to be able to develop the emulation in an orderly and correct way. The topology shows one data center configured with two spines multilayer switches and three leaf multilayer switches. Every leaf switch is connected to a host representing a physical server containing virtualized environments; all this infrastructure represents the ”LAN,” which is connected through Port 2 with the Fortinet Firewall. Port 1 is connected to Cloud 1 and represents the Internet or external networks. In addition, the underlay network is configured using the OSPF routing protocol, and the overlay network is configured with MP-BGP routing protocol. The Fig. 6 shows the configurations performed in the Leaf multilayer switches.



```
!
interface GigabitEthernet1/3
 negotiation auto
!
interface Vlan11
 ip address 10.1.11.1 255.255.255.0
!
router ospf 65100
 router-id 1.1.1.3
 network 10.0.0.11 0.0.0.0 area 0
 network 10.1.1.0 0.0.0.255 area 0
 network 10.1.4.0 0.0.0.255 area 0
!
router bgp 65100
 bgp log-neighbor-changes
 neighbor 10.0.0.1 remote-as 65100
 neighbor 10.0.0.2 remote-as 65100
 !
 address-family ipv4 multicast
 exit-address-family
 !
 address-family l2vpn evpn
  neighbor 10.0.0.1 activate
  neighbor 10.0.0.2 activate
 exit-address-family
!
ip forward-protocol nd
!
ip http server
ip http secure-server
!
ip ssh server algorithm encryption aes128-ctr aes192-ctr aes256-ctr
ip ssh client algorithm encryption aes128-ctr aes192-ctr aes256-ctr
!
!
 --More--
```

Fig. 6. Leaf, Multilayer Switches configuration

Fig. 7 shows the configurations performed in the Spine multilayer switches and the configuration where the EVPN protocol is activated.



```
 no ip address
 shutdown
 duplex auto
 speed auto
 media-type rj45
!
router ospf 65100
 router-id 1.1.1.1
 network 10.0.0.1 0.0.0.0 area 0
 network 10.1.1.0 0.0.0.255 area 0
 network 10.1.2.0 0.0.0.255 area 0
 network 10.1.3.0 0.0.0.255 area 0
!
router bgp 65100
 bgp log-neighbor-changes
 network 10.0.0.1 mask 255.255.255.255
 network 10.0.0.1
 neighbor 10.0.0.10 remote-as 65100
 neighbor 10.0.0.10 cluster-id 1.1.1.1
 neighbor 10.0.0.11 remote-as 65100
 neighbor 10.0.0.11 cluster-id 1.1.1.1
 neighbor 10.0.0.12 remote-as 65100
 neighbor 10.0.0.12 cluster-id 1.1.1.1
 !
 address-family l2vpn evpn
  neighbor 10.0.0.10 activate
  neighbor 10.0.0.11 activate
  neighbor 10.0.0.12 activate
 exit-address-family
!
ip forward-protocol nd
!
!
no ip http server
no ip http secure-server
!
ipv6 ioam timestamp
!
```

Fig. 7. Spine, Multilayer Switches configuration

The figure below Fig. 11 shows the ICMP (ping) tests performed between the spine and leaf switches to validate the communication across all the nodes.

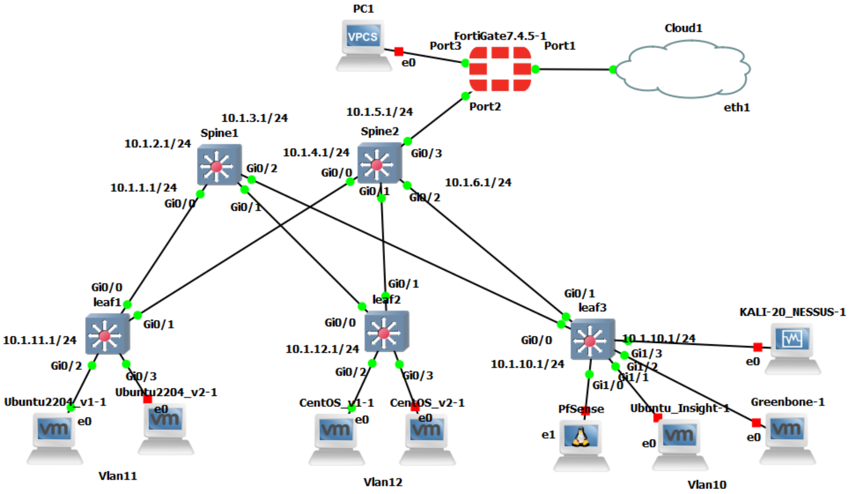
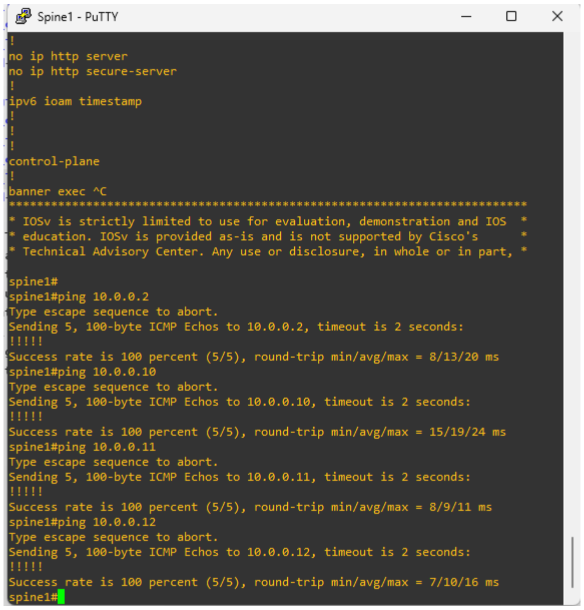
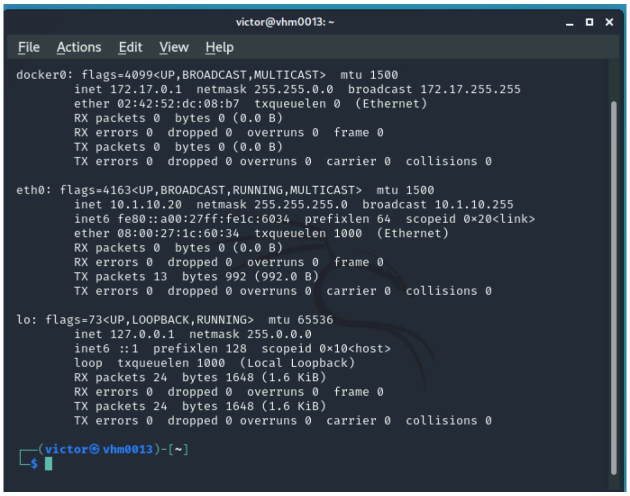


Fig. 5. Spine and Leaf Arquitecture designed in GNS3



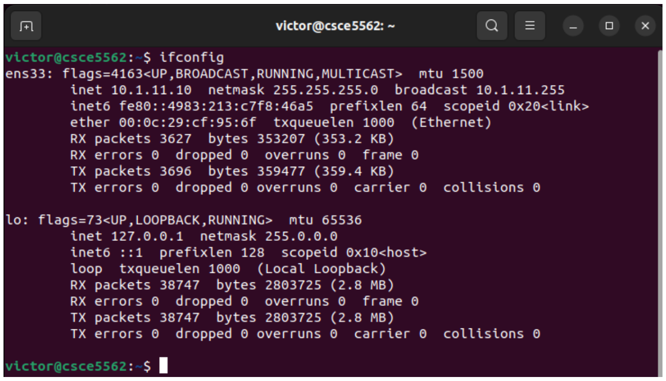
```
!
no ip http server
no ip http secure-server
!
ipv6 ioam timestamp
!
!
!
control-plane
!
banner exec ^C
**************************************************************************
* IOSv is strictly limited to use for evaluation, demonstration and IOS  *
* education. IOSv is provided as-is and is not supported by Cisco's      *
* Technical Advisory Center. Any use or disclosure, in whole or in part, *

spine1#
spine1#ping 10.0.0.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.0.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 8/13/20 ms
spine1#ping 10.0.0.10
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.0.10, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 15/19/24 ms
spine1#ping 10.0.0.11
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.0.11, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 8/9/11 ms
spine1#ping 10.0.0.12
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.0.12, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 7/10/16 ms
spine1#
```

Fig. 8. Spine1 multilayer switch ICMP test, Spine and Leaf switches.



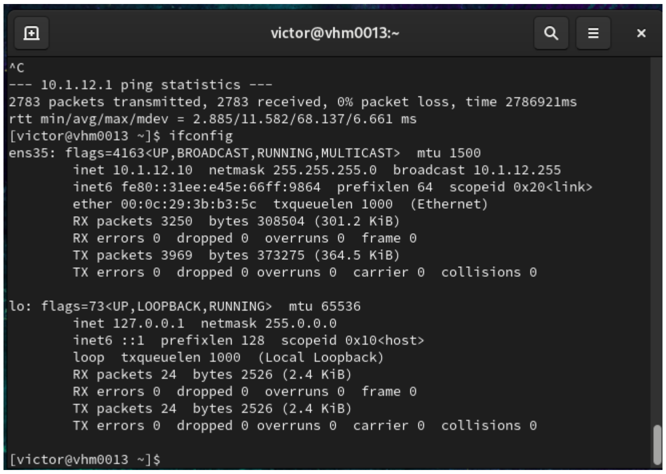
```
docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:52:dc:08:b7  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.1.10.20  netmask 255.255.255.0  broadcast 10.1.10.255
        inet6 fe80::a00:27ff:fe1c:6034  prefixlen 64  scopeid 0×20<link>
        ether 08:00:27:1c:60:34  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 13  bytes 992 (992.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0×10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 24  bytes 1648 (1.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 24  bytes 1648 (1.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

┌──(victor㉿vhm0013)-[~]
└─$ 
```

Fig. 9. Vlan 10 Kali Nessus IP settings.



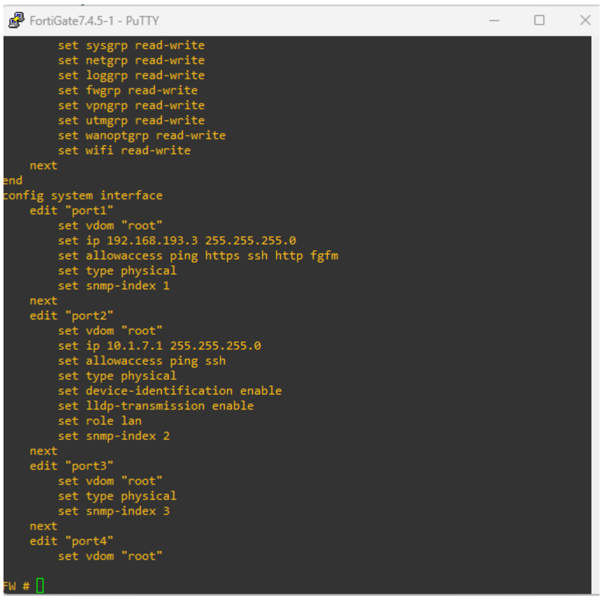
```
victor@csce5562:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.1.11.10  netmask 255.255.255.0  broadcast 10.1.11.255
        inet6 fe80::4983:213:c7f8:46a5  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:cf:95:6f  txqueuelen 1000  (Ethernet)
        RX packets 3627  bytes 353207 (353.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3696  bytes 359477 (359.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 38747  bytes 2803725 (2.8 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 38747  bytes 2803725 (2.8 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

victor@csce5562:~$ 
```

Fig. 10. Vlan 11 Ubuntu Webserver IP settings.



```
^C
--- 10.1.12.1 ping statistics ---
2783 packets transmitted, 2783 received, 0% packet loss, time 2786921ms
rtt min/avg/max/mdev = 2.885/11.582/68.137/6.661 ms
[victor@vhm0013 ~]$ ifconfig
ens35: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.1.12.10  netmask 255.255.255.0  broadcast 10.1.12.255
        inet6 fe80::31ee:e45e:66ff:9864  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:3b:b3:5c  txqueuelen 1000  (Ethernet)
        RX packets 3250  bytes 308504 (301.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3969  bytes 373275 (364.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 24  bytes 2526 (2.4 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 24  bytes 2526 (2.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[victor@vhm0013 ~]$
```

Fig. 11. Vlan 12 CentOS Webserver IP settings.



```
        set sysgrp read-write
        set netgrp read-write
        set loggrp read-write
        set fwgrp read-write
        set vpngrp read-write
        set utmgrp read-write
        set wanoptgrp read-write
        set wifi read-write
    next
nd
onfig system interface
    edit "port1"
        set vdom "root"
        set ip 192.168.193.3 255.255.255.0
        set allowaccess ping https ssh http fgfm
        set type physical
        set snmp-index 1
    next
    edit "port2"
        set vdom "root"
        set ip 10.1.7.1 255.255.255.0
        set allowaccess ping ssh
        set type physical
        set device-identification enable
        set lldp-transmission enable
        set role lan
        set snmp-index 2
    next
    edit "port3"
        set vdom "root"
        set type physical
        set snmp-index 3
    next
    edit "port4"
        set vdom "root"

FW #
```

Fig. 12. Fortinet Firewall settings

The following figure of the architecture of Spine and Leaf shows the traffic between Vlan 10 and Vlan 11

The Trust model tracks the behavior of these VMs over time to determine their trustworthiness. VMs are regularly assigned sets of vulnerabilities and monitored to identify whether they act trusted or malicious (untrusted) over a series of determined interactions. Through the dynamic evaluation process, our trust model will continuously update the trust level of each VM to maintain a secure environment by rewarding trustworthy behavior and penalizing by replacing VMs that behave abnormally. To achieve the trust evaluation of the VMs in the cloud environment, we use game theory principles to simulate the interaction between VMs and the cloud administrator, constantly monitoring the VM behavior to update the trust levels based on the observed actions. This process will guarantee that trusted VMs remain in the system.

IV. Experimental Results

A Multi-tenant Cloud Environment was built using the tools described above, where multiple users share the same physical resources, so it is crucial to establish trust between Virtual Machines. The VMs will consist of the following services:

- Web Servers
- DB Server
- Obtain VMs Vulnerabilities / tools

To collect the data, it was configured the following environment that consists of four main servers (VMs):

- Ubuntu Webserver
- CentOS Webserver
- Ubuntu DB MySQL
- CentOS DB Posrgresql

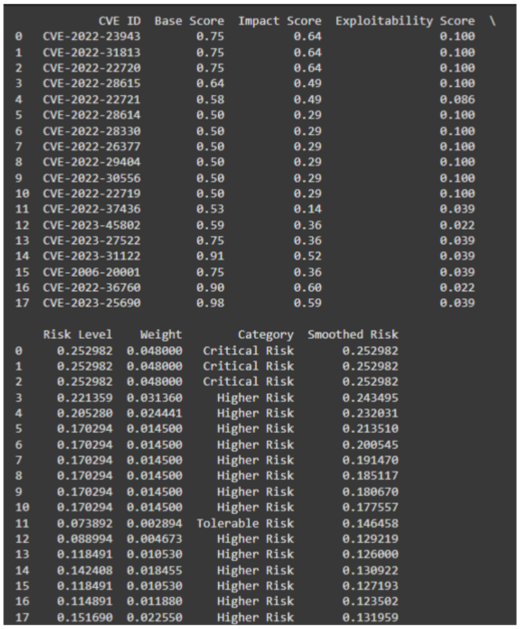
In addition, it was used the following tools to collect the data:

- Deploy IDS to monitor network, assess, and VM behavior (pfsense + suricata)
- Rapid7, Nessus, OpenVAS, Sparta Nikto
- VMware Workstation17
- VirtualBox
- Pfsense IDS/IPS Suricata

The risk assessment was completed using four different tools, and different results were obtained. The following table compares the vulnerabilities identified in the analysis. The vulnerability assessment was performed using four different vulnerability assessment tools to scan the infrastructure to identify vulnerabilities for risk identification.

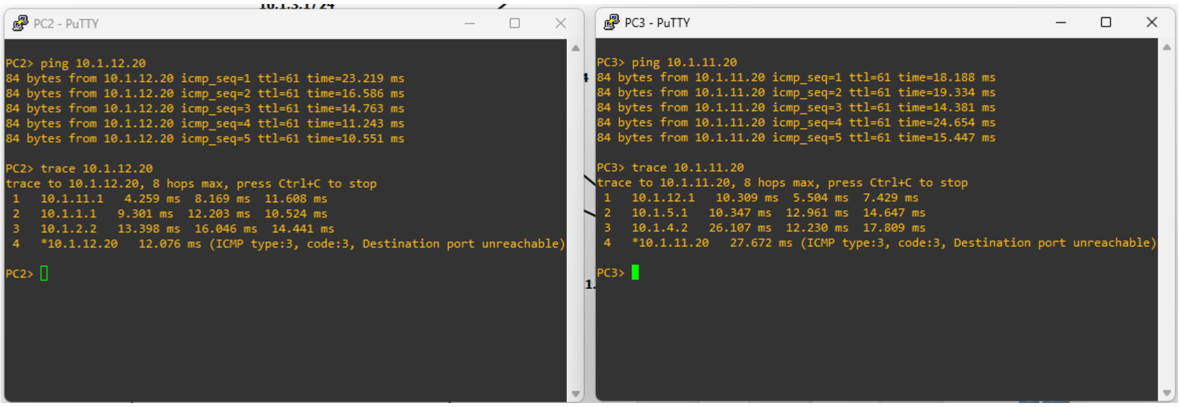
The simulation was conducted over 100 iterations, reflecting the ongoing interactions between the cloud administrator and VMs. The initial trust values were set based on the risk assessments, with VMs' behaviors monitored throughout the simulation period. Key metrics, including historical behavior, vulnerability profiles, and risk exposure, were used to update trust values dynamically.

Results indicate that VMs with lower risk scores consistently maintained higher trust values, confirming the model's effectiveness in distinguishing reliable VMs from potentially compromised ones. Integrating dynamic risk assessments and Bayesian updates enabled the model to adapt to changing conditions within the cloud environment, offering a nuanced understanding of VM reliability over time. The figure below shows how it measured the model's effectiveness using the Gaussian distribution method.



```
          CVE ID  Base Score  Impact Score  Exploitability Score  \
0   CVE-2022-23943        0.75          0.64                 0.100
1   CVE-2022-31813        0.75          0.64                 0.100
2   CVE-2022-22720        0.75          0.64                 0.100
3   CVE-2022-28615        0.64          0.49                 0.100
4   CVE-2022-22721        0.58          0.49                 0.086
5   CVE-2022-28614        0.50          0.29                 0.100
6   CVE-2022-28330        0.50          0.29                 0.100
7   CVE-2022-26377        0.50          0.29                 0.100
8   CVE-2022-29404        0.50          0.29                 0.100
9   CVE-2022-30556        0.50          0.29                 0.100
10  CVE-2022-22719        0.50          0.29                 0.100
11  CVE-2022-37436        0.53          0.14                 0.039
12  CVE-2023-45802        0.59          0.36                 0.022
13  CVE-2023-27522        0.75          0.36                 0.039
14  CVE-2023-31122        0.91          0.52                 0.039
15  CVE-2006-20001        0.75          0.36                 0.039
16  CVE-2022-36760        0.90          0.60                 0.022
17  CVE-2023-25690        0.98          0.59                 0.039

    Risk Level    Weight        Category  Smoothed Risk
0     0.252982  0.048000   Critical Risk       0.252982
1     0.252982  0.048000   Critical Risk       0.252982
2     0.252982  0.048000   Critical Risk       0.252982
3     0.221359  0.031360     Higher Risk       0.243495
4     0.205280  0.024441     Higher Risk       0.232031
5     0.170294  0.014500     Higher Risk       0.213510
6     0.170294  0.014500     Higher Risk       0.200545
7     0.170294  0.014500     Higher Risk       0.191470
8     0.170294  0.014500     Higher Risk       0.185117
9     0.170294  0.014500     Higher Risk       0.180670
10    0.170294  0.014500     Higher Risk       0.177557
11    0.073892  0.002894  Tolerable Risk       0.146458
12    0.088994  0.004673     Higher Risk       0.129219
13    0.118491  0.010530     Higher Risk       0.126000
14    0.142408  0.018455     Higher Risk       0.130922
15    0.118491  0.010530     Higher Risk       0.127193
16    0.114891  0.011880     Higher Risk       0.123502
17    0.151690  0.022550     Higher Risk       0.131959
```

Fig. 14. Risk Analysis, of the vulnerabilities identified in Table 2



PC2 - PuTTY

```
PC2> ping 10.1.12.20
84 bytes from 10.1.12.20 icmp_seq=1 ttl=61 time=23.219 ms
84 bytes from 10.1.12.20 icmp_seq=2 ttl=61 time=16.586 ms
84 bytes from 10.1.12.20 icmp_seq=3 ttl=61 time=14.763 ms
84 bytes from 10.1.12.20 icmp_seq=4 ttl=61 time=11.243 ms
84 bytes from 10.1.12.20 icmp_seq=5 ttl=61 time=10.551 ms

PC2> trace 10.1.12.20
trace to 10.1.12.20, 8 hops max, press Ctrl+C to stop
 1   10.1.11.1   4.259 ms  8.169 ms  11.608 ms
 2   10.1.1.1   9.301 ms  12.203 ms  10.524 ms
 3   10.1.2.2   13.398 ms  16.046 ms  14.441 ms
 4   *10.1.12.20   12.076 ms (ICMP type:3, code:3, Destination port unreachable)

PC2>
```

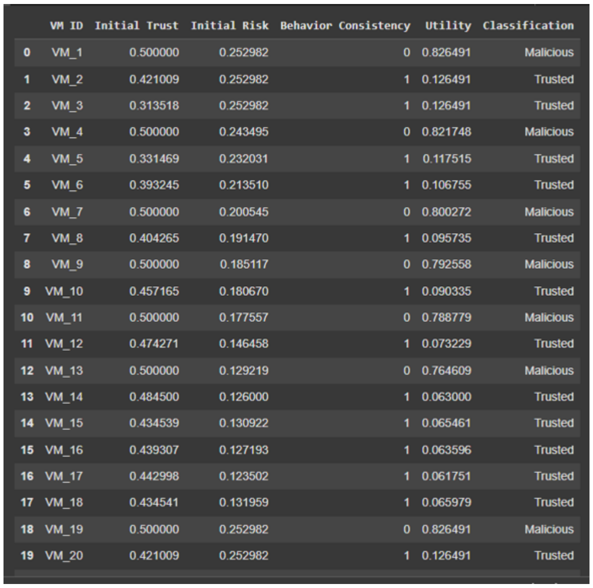
PC3 - PuTTY

```
PC3> ping 10.1.11.20
84 bytes from 10.1.11.20 icmp_seq=1 ttl=61 time=18.188 ms
84 bytes from 10.1.11.20 icmp_seq=2 ttl=61 time=19.334 ms
84 bytes from 10.1.11.20 icmp_seq=3 ttl=61 time=14.381 ms
84 bytes from 10.1.11.20 icmp_seq=4 ttl=61 time=24.654 ms
84 bytes from 10.1.11.20 icmp_seq=5 ttl=61 time=15.447 ms

PC3> trace 10.1.11.20
trace to 10.1.11.20, 8 hops max, press Ctrl+C to stop
 1   10.1.12.1   10.309 ms  5.504 ms  7.429 ms
 2   10.1.5.1   10.347 ms  12.961 ms  14.647 ms
 3   10.1.4.2   26.107 ms  12.230 ms  17.809 ms
 4   *10.1.11.20   27.672 ms (ICMP type:3, code:3, Destination port unreachable)

PC3>
```

Fig. 13. Verification of communication between Vlan 11 and Vlan 12, through VxLAN

TABLE II
COMPARISON TABLE VUNERABILITY ASSESSMENT

CVE-ID	Vulnerability Type	Vulnerability	Nessus	GreenBone	Rapid7	Sparta/Nikto
CVE-1999-0524	Ubuntu Web Server	ICMP TimestampReply Information Disclosure	X	X	X	
		Web Server Transmits Cleartext Credentials	X			
		Web Application Potentially Vulnerable to Clickjacking	X			X
		Weak MAC Algorithm(s) Supported (SSH)	X	X	X	
		TCP Timestamps Information Disclosure	X	X	X	
CVE-2022-23943		Apache HTTPD: mod_sed: Read/write beyond bounds			X	
CVE-2022-31813		Apache HTTPD: mod_proxy X-Forwarded-For dropped by hop-by-hop mechanism			X	
CVE-2022-22720		Apache HTTPD: HTTP request smuggling vulnerability in Apache HTTP Server 2.4.52 and earlier			X	X
CVE-1999-0524	CentOS Web Server	ICMP Timestamp Request Remote Date Disclosure	X	X	X	
		PHP Unsupported Version Detection	X			
		Web Application Potentially Vulnerable to Clickjacking	X			X
CVE-1999-0524	CentOS Web PostgreSQL	ICMP Timestamp Request Remote Date Disclosure	X	X	X	
		TCP Timestamps Information Disclosure	X	X		
		SSH Weak Message Authentication Code Algorithms	X		X	
		OpenSSH Detection	X			



	VM ID	Initial Trust	Initial Risk	Behavior Consistency	Utility	Classification
0	VM_1	0.500000	0.252982	0	0.826491	Malicious
1	VM_2	0.421009	0.252982	1	0.126491	Trusted
2	VM_3	0.313518	0.252982	1	0.126491	Trusted
3	VM_4	0.500000	0.243495	0	0.821748	Malicious
4	VM_5	0.331469	0.232031	1	0.117515	Trusted
5	VM_6	0.393245	0.213510	1	0.106755	Trusted
6	VM_7	0.500000	0.200545	0	0.800272	Malicious
7	VM_8	0.404265	0.191470	1	0.095735	Trusted
8	VM_9	0.500000	0.185117	0	0.792558	Malicious
9	VM_10	0.457165	0.180670	1	0.090335	Trusted
10	VM_11	0.500000	0.177557	0	0.788779	Malicious
11	VM_12	0.474271	0.146458	1	0.073229	Trusted
12	VM_13	0.500000	0.129219	0	0.764609	Malicious
13	VM_14	0.484500	0.126000	1	0.063000	Trusted
14	VM_15	0.434539	0.130922	1	0.065461	Trusted
15	VM_16	0.439307	0.127193	1	0.063596	Trusted
16	VM_17	0.442998	0.123502	1	0.061751	Trusted
17	VM_18	0.434541	0.131959	1	0.065979	Trusted
18	VM_19	0.500000	0.252982	0	0.826491	Malicious
19	VM_20	0.421009	0.252982	1	0.126491	Trusted

Fig. 15. Trust model output

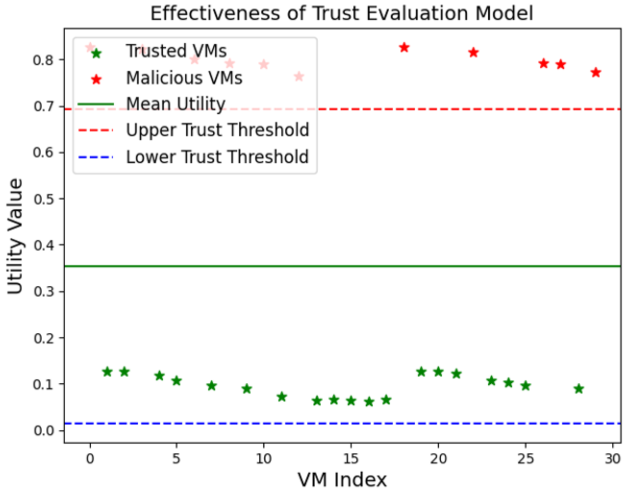


Fig. 16. Effectiveness of the VM trust model, where the trusted nodes are shown with green stars

Furthermore, the simulations revealed that using rewards and penalties based on VM strategies significantly influenced the evolution of trust. VMs demonstrating consistent, secure behavior were rewarded, thus enhancing their trust levels, while those exhibiting risky behavior faced penalties, leading to decreased trust values. This dynamic adjustment process supports the selection of VMs that align with the cloud administrator's security objectives.

V. Conclusion

The emulation performed in GNS3 demonstrates that the Spine and Leaf topology is optimized explicitly for the traffic, ensuring high availability and reliability for organizational applications. In modern data centers, where applications must accommodate high user demand, this topology enables seamless hyperconvergence, providing users with fast, secure, and efficient access to resources.

Moreover, it demonstrates the numerous advantages of employing VXLAN-EVPN and showcases, through multilayer switches, the potential to deliver robust and modern solutions. This approach enables seamless interconnection between dispersed data centers using the same VLANs, optimizing IP address utilization, enhancing application performance, and improving traffic control.

This research introduces a dynamic trust evaluation model for VMs in cloud environments, emphasizing the importance of integrating risk assessment and game theory in enhancing cloud security. The results demonstrate that using real-time risk data and Bayesian updates in trust calculations can effectively identify reliable VMs, supporting secure access to cloud resources. By addressing the limitations of traditional trust models, our approach offers a more adaptive mechanism for cloud administrators to manage VM trustworthiness.

Future work will explore extending this model to accommodate more complex cloud environments, considering various types of attacks and mitigation strategies. Additionally, integrating machine learning techniques could further enhance the model's ability to predict and adapt to evolving security threats within the cloud ecosystem.

References

[1] J. John and K. John Singh, "Trust value evaluation of cloud service providers using fuzzy inference based analytical process," *Scientific Reports*, vol. 14, no. 1, p. 18028, 2024.

[2] Markets and Markets, "Cloud computing market by service model (iaas, paas, and saas), deployment model (public, private, and hybrid), organization size (smes and large enterprises), vertical and region - global forecast to 2027," 2024, accessed: 2024-11-07. [Online]. Available: https://www.marketsandmarkets.com/Market-Reports/cloud-computing-market-234.html

[3] A. Oliveira, "Modelling trust and risk for cloud services," *Journal of cloud computing*, vol. 7, no. 4, pp. 1–16, 2018.

[4] J. Huang and D. M. Nicol, "A survey of trust mechanisms in cloud computing," 2013, accessed: 2024-11-27. [Online]. Available: https://journalofcloudcomputing.springeropen.com/articles/10.1186/2192-113X-2-9

[5] Y. Wu and C. Zhang, "Trust evaluation model in virtualized cloud environments," 2011, accessed: 2024-11-27. [Online]. Available: https://link.springer.com/content/pdf/10.1007/978-3-642-10549-4_22.pdf

[6] X. Li and Y. Wang, "Blockchain-based trust management for cloud systems: A survey," 2021, accessed: 2024-11-27. [Online]. Available: https://journalofcloudcomputing.springeropen.com/articles/10.1186/s13677-021-00247-5

[7] R. Balasubramaniam and S. Byrappa, "Risk assessment framework for cloud-based systems: A quantitative approach," 2020, accessed: 2024-11-27. [Online]. Available: https://doi.org/10.4018/IJCAC.2020040103

[8] M. Chiregi and N. J. Navimipour, "Trust evaluation mechanisms in cloud computing: A systematic review," 2017, accessed: 2024-11-27. [Online]. Available: https://doi.org/10.1016/j.jnca.2017.03.004

[9] J. Pawlick, Q. Zhu, and R. Poovendran, "istrict: Interdependent strategic trust mechanisms in cloud systems," 2018, accessed: 2024-11-27. [Online]. Available: https://doi.org/10.1145/3180760

[10] W. Sun, J. Li, and X. Wang, "A hybrid trust framework for cloud computing environments," 2020, accessed: 2024-11-27. [Online]. Available: https://doi.org/10.1109/TCC.2020.2966154

[11] M. Alam, A. Khan, and H. Raza, "Trust-aware resource allocation in multi-tenant cloud systems using game theory," 2019, accessed: 2024-11-27. [Online]. Available: https://doi.org/10.1016/j.future.2018.11.015

[12] H. Namdari, C. Tunc, and R. Dantu, "Phoenix: Iot trust evaluation using game theory with second chance protocol," in *2023 10th International Conference on Internet of Things: Systems, Management and Security (IOTSMS)*. IEEE, 2023, pp. 117–124.

[13] Balbix, "Understanding cvss scores: What they are and how to use them," 2024, accessed: 2024-09-17. [Online]. Available: https://www.balbix.com/insights/understanding-cvss-scores/

[14] J. Networks, "Vxlan evpn integration overview," n.d., accessed: 2024-11-30. [Online]. Available: https://www.juniper.net/documentation/us/en/software/junos/evpn/topics/concept/vxlan-evpn-integration-overview.html