SECURITY IN CLOUD VIRTUALIZATION LAYER

A Synopsis submitted to Gujarat Technological University

for the Award of

Doctor of Philosophy

in

Computer Engineering

by

Dhara H. Buch

[139997107002]

under the supervision of

Prof. (Dr.) Haresh S. Bhatt

GUJARAT TECHNOLOGICAL UNIVERSITY

AHMEDABAD

August – 2020

Index

List of Abbreviations

ABEA	Asynchronous BTB Eviction Attack
AES	Advanced Encryption Standard
API	Application Programming Interface
BPA	Branch Prediction Analysis
BPU	Branch Prediction Unit
BTB	Branch Target Buffer
CPU	Central Processing Unit
CRT	Chinese Remainder Theorem
CSP	Cloud Service Provider
DES	Data Encryption Standard
DoS	Denial of Service
DTA	Direct Timing Attack
ECC	Elliptical Curve Cryptography
HTTPS	Hyper Text Transfer Protocol Secure
IaaS	Infrastructure as a Service
IDS	Intrusion Detection System
IPS	Intrusion Prevention System
MITM	Man-In-The-Middle
OS	Operating System
PaaS	Platform as a Service
PAK	Private Asymmetric Key
PSK	Private Symmetric Key
RSA	Rivest-Shamir-Adleman
SaaS	Software as a Service
SBEA	Synchronous BTB Eviction Attack
SCA	Side Channel Attack
SFTP	Secure File Transfer Protocol
SLA	Service Level Agreement
SQL	Structured Query Language
TDA	Trace Driven Attack
TLB	Translation Look-aside Buffer
VM	Virtual Machine
VMM	Virtual Machine Monitor
XSS	Cross-site Scripting

Abstract

Branch Prediction Analysis attack is one of the most significant Side-Channel Attack (SCA), which causes severe issues on a machine hosting multiple services by exploiting shared resources. The current state of the art cloud technology provides a level of isolation by hosting processes on different VMs (Virtual Machines). Still, the scope of exploitation does not get eliminated even in the virtualization environment. The severity of the BPA attack and its normal-looking attack detecting mechanism makes its study very interesting and challenging. With the main research focus on security issues in the virtual environment, handling of Cross-VM BPA attack is the core of the present research work. The applicability of four BPA attack launching methods has been assessed on different types of VM configurations. Simulation of two important types of BPA (Branch Prediction Analysis) attacks; DTA (Direct Timing Attack) and TDA (Trace-Driven Attack) was also done on the most common VM configuration. With an in-depth study of attack launching methods and their behavior analysis, a four-eyed model *Chaturdrashta* is proposed. *Chaturdrashta* is comprised of two solutions: *Trilochan* to detect Cross-VM Direct Timing Attack (DTA) and *Trinetra* to detect Cross-VM Trace-Driven Attack (TDA). Solutions can successfully detect the attack by the time when just a few bits are predicted. The processing overhead of the proposed approach is hardly 1%. Additionally, *Trilochan* and *Trinetra* in their original form were also found capable of detecting the presence of the BPA attack launched with the Asynchronous and Synchronous BTB Eviction methods. A testbed comprising of various types of genuine processes was simulated to check the efficiency of solutions. With high accuracy in attack detection, the solutions do not have any false positives. The proposed solutions neither depend on any cryptographic algorithm nor manipulate any architectural components.

Chaturdrashta is a host-based solution, where one of the components is embedded in the kernel. The other three components are implemented as Linux services. Such an implementation requires a system reboot to bring their manipulations into effect. In turn, it reduces the scope of *Chaturdrashta* of getting exploited.

1. State of the Art

Cloud technology has become a defacto standard for service provisioning due to its resource optimization capabilities. Its feature of providing virtual machines (VMs) to different users for different purposes is being used very commonly by cloud administrators. This multi-tenant environment of the cloud technology opens up a new dimension of the security threats due to its intrinsic characteristics. Most of the users and administrators consider virtual machines as independent machines. Configuring full proof isolated virtual machine is not possible in available tools and technology directly. Very few tools tried to provide this facility but at the cost of resource optimization and compromising useful features like load balancing and fault tolerance. Furthermore, it requires high-level expertise and in-depth knowledge to implement such configuration. The common out of the box, standard and adopted configuration model does provide isolation of memory, disk space, OS, Applications, etc. but shares CPU cores across virtual machines. Thus, hardware resources like Cache Memory, Memory Bus, Network Queue, and Branch Prediction Unit (BPU) are also shared among co-hosted VMs. The sharing of resources opens the scope of Side-Channel Attacks, which are very common in machines used to host multiple services. We have studied one of the SCA, called Branch Prediction Analysis attack for our research work on "Security in Cloud Virtualization Layer". The study revealed the necessity of working out solutions to address BPA attacks in the virtual environment.

1.1 Introduction

Subsection 1.2 gives brief information on the literature survey, which leads us to the conclusion of selecting BPA attack for further research. Subsection 1.3 briefs about the branch prediction attack and Subsection 1.4 deals with the available solutions for handling BPA attack on a traditional model of hosting services on standalone servers.

Section 2 defines the problem statement, followed by the listing of the objectives and scope of our work in Section 3. Section 4 focuses on related work, which discusses the scope of BPA attack in virtualization and applicability of existing solutions in Subsections 4.1 and 4.2, respectively. Further, attack launching methods are discussed in Subsection 4.3. The original contribution of our work is discussed in Section 5, where Subsection 5.1 discusses attack simulation for BPA attack methods. Subsection 5.2 briefly highlights the behavior-based characteristic of the proposed approach, followed by a discussion of the proposed solution in Subsection 5.3.

Experimental analysis is covered in Section 6, which includes a simulated operation environment and observed results of implementation in Subsection 6.1 and 6.2, respectively. Achievements and conclusions are presented in Sections 7 and 8, respectively. Finally, the list of publications, as well as the papers under review, are provided in Section 9.

1.2 Literature Survey

A large surface area of cloud architecture requires security consideration from different perspectives. In our research work, we classified various surveys on cloud security into two categories considering virtualization as one of the essential aspects of cloud environment : (1) Survey of generic cloud security issues [1][2][3][4][5][6][7]. (2) Survey of cloud security with virtualization as the main focus [8]–[11].

Among generic cloud security surveys falling in category 1, Khan et al. [1], classifies attacks as network, VM, storage, and application-based attacks. They have also discussed the application of the Intrusion Prevention System (IPS) and Intrusion Detection System (IDS) to handle them. The paper gives an excellent discussion on cloud security attacks. A slightly different type of classification was suggested by Gonzalez et al. [2] that focuses on network security, interfaces, data security, virtualization, governance, and compliance. They categorize the taxonomy of cloud security into Privacy, Architecture, and Compliance. Without diving deep into the details, they provide a graphical representation of existing contributions in various issues. In an absolutely different form of classification, invoking, using, and the managing cloud was identified as the primary functions of cloud by Gruschka et al. [3]. Six attack surfaces shown in the presented taxonomy can be applied to handle Amazon EC2 Hack, Cloud War, and Cloud Malware Injection [3]. Srinivasan et al. [4], on the other hand, focuses on architectural, technological, and process-regulatory related aspects. Logical storage segregation, identity management, insider attacks, cryptography, governance, Insecure API, CSP migration, and SLA trust gap are the main categories covered in [4]. In another survey, Coppolino et al. [5] have discussed open issues in a cloud environment and listed security challenges like shared technologies vulnerabilities, data breach, Service Traffic Hijacking, Denial of Service (DoS) and malicious insiders. They identify need to tackle malicious insiders by studying existing countermeasures. Singh et al. in [6] discuss issues in the areas like the embedded system, application, trust, client management, clustering, data management and Operating System (OS). The study also includes discussion on issues like digital forensics, virtualization, fault tolerance and risk analysis which measures the cloud security

in an utterly generic yet efficient way. Probable security issues among four domains, namely, an infrastructure layer, a platform layer, application layer, and administration of cloud computing environment, were the parts of the work of Seyyed et al. in [7]. All the above approaches give a good insight into generic cloud security issues where details regarding the virtualization security are not considered.

Among the virtualization-centric survey approaches mapping to category 2, Pek [8] has carried out an in-depth analysis of all hardware virtualization attacks that provides excellent support to researchers working in that direction. Virtualization-based attacks and related handling approaches are also discussed by C. Modi et al. in [9]. They provide an insight into firewall and IDS/IPS based approaches and identify a need for a cloud IDS to overcome the existing limitations. In another analytical survey by D. Sgandurra et al. [10], probable attack paths are presented. With an objective to provide a common framework for security, a common base of cloud assumptions and methodologies is also employed by them. Riddle and Chung in [11], focuses on the primary study of hypervisor security, where they identify VM Escape as a very difficult one to cope with. With virtualization security as the primary parameter, Ali et al. [12] give a detailed survey on cloud security types, namely, communication, architectural, and legal aspect. They have carried out a comparative analysis of contemporary approaches to test features like privacy, integrity, access control, and scalability for analyzing the effectiveness of the existing solutions.

The above discussion reveals that none of the existing surveys covers generic as well as virtualization security issues with a brief review of present countermeasures handling them. Accordingly, we planned to define a taxonomy that includes generic cloud security issues with the main focus on virtualization security.

The study of cloud security cannot be ended without considering SaaS (Software as a Service), PaaS (Platform as a Service) and IaaS (Infrastructure as a Service) as they are the primary elements in providing cloud services [13]. Additionally, as per Almorsy et al. [14], as SaaS is built on the top, i.e., on PaaS and IaaS, security issues of these models are inherited by SaaS also. Iqbal et al. [15] have recognized the importance of these service delivery models and discussed issues pertaining to them like DoS, SQL Injection, Phishing, XSS, VM Escape, VM rollback and Man-in-the-Middle. They also analyze how IDS/IPS can be applicable for handling them. According to W. Dawoud et al. [16], Service Level Agreement, Utility Computing, Platform Virtualization,

Cloud Software, Network & Internet Connectivity, and Computer Hardware are the main components of IaaS, which we need to consider while imposing IaaS security.

After studying the surveys and related papers, we find it interesting to present the study in the form of taxonomy with a special focus on virtualization with an aim to identify the research gap. Our presented taxonomy maps at the intersection of categories 1 and 2, which is shown in Fig. 1.

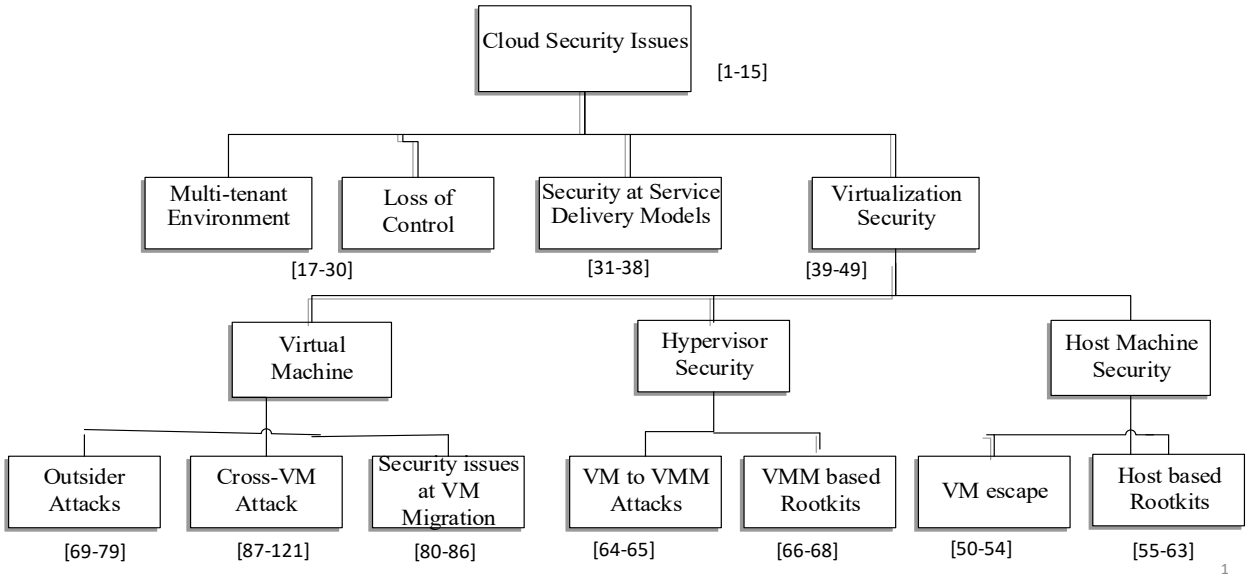

Fig. 1 Taxonomy of Cloud Virtualization Security

As shown in the diagram, our taxonomy gives more weightage to virtualization layer security issues where other essential security issues raised because of the multi-tenant environment, service delivery models, and third party CSP are also included.

Study of the first category, i.e., security issues at a multi-tenant environment, was carried out by Takahashi et al. [17], where they discuss security challenges and issues caused due to the simultaneous access to the web, application, operating system, hypervisor, and hardware-software layers. While talking on security issues in the multi-tenant environment, data security plays a very important role. Many researchers have contributed to handling data security issues, which include a five-model proposal by Zhao et al. [18] for data security, and Identity Management based framework by Aishwarya et al. [19] for assuring correctness of cloud data. Data authenticity, along with data encryption schemes, is the main point of discussion for Sudha et al. [20] and Gobi et al. [21]. Some approaches have worked on dynamic data supporting a distributed scheme. Approaches

suggested by Wang et al. [22] works in that direction to provide localization of data error. Similar approaches are also proposed by Purushothaman et al. [23] and Reddy et al. [24]. A distributed scheme is proposed by them to verify erasure-coded data and handles the colluding attack with a homomorphism token. Among the other solutions for data security, Sangroya et al. [25] analyze data security with risk analysis for some trust parameters, G. R. Reddy et al. [26] suggests scattering of data among multiple CSPs and Juels et al. [27] provides a data auditing framework for ensuring data integrity. Smith et al. [28] handled data security from a different angle by presenting an anomaly detection IDS for automatic management in the cloud system. The above analysis reveals that data security is analyzed from many different aspects by researchers along with possible handling mechanisms which increases the data assurance in cloud users. However, the control of CSP on confidential data is also an important point to be discussed. Hamlen et al. [29] have contributed to that area by proposing a scheme of secure third-party publication of documents and secure query processing with Map Reduce and Hadoop. A similar track is followed by Almorsy et al. [30] where a security framework for increasing collaboration among cloud providers, service providers, and service consumers is suggested.

Where researchers are working on various data security aspects on one end, contribution in handling issues at service delivery models SaaS, PaaS, and IaaS is also an area of interest of many researchers. Such contribution includes an IaaS level IDS by Tupakula et al. [31] and critical analysis on IaaS layer issues raised due to multi-tenancy by Vaquero et al. [32]. In another policy-based approach presented by Yildiz et al. [33], security for the network, server, storage, system management domains, and infrastructure scope is covered. Arora et al. [34] and Alharkan et al. [35] also focus on IaaS security, where a cubic model defining the relationship between IaaS components and related security requirements and an elastic signature-based IDS framework are proposed respectively. SaaS and PaaS were also focused by Gustavo et al. [36], Subashini et al. [37], and Jensen et al. [38] where the Gustavo proposes an anomaly-based IDS for data protection on SaaS web application. The other two discuss security issues at different service delivery models.

Analysis of present work carried out in the direction of SaaS, PaaS, and IaaS security ensures authentic and authorized service provisioning. However, secure service delivery models and Data Centers without a secure virtualization layer bring down the entire means of providing abstract services. Hence, the study of virtualization security becomes essential while considering cloud security.

There are several researchers who discuss various virtualization layer security issues in a general form. Among them, the concept of virtualization as well as hardware and software tools to handle the related issues are discussed by Kirch et al. in [39]. A similar kind of discussion is carried out by Reuben et al. in [40] and Anand et al. in [41]. Vulnerabilities in virtualized environments are analyzed by comparing features of hosted, OS, and Bare-Metal types by Brooks et al. in [42]. Mobility, Diversity, and Identity are also the points to be considered for security at the virtual layer as per T. Garfinkel et al. [43]. Potential of virtualization security risks are suggested by Nan et al. [44], while Li et al. [45] proposed architecture for virtual machine security, virtual network security, and policy-based trust management services. A set of security considerations like Role-based Access Control, Data Isolation, and Customer Privacy Protection for cloud computing virtualization environments are identified by S. Yeo et al. [46], while security requirements, attacks and security solutions for virtualization are presented by Kazim et al. [47]. Cleef et al. [48] have given a detailed literature study of the virtualization implications. They have suggested to remove or to tightly manage non-essential features like introspection, rollback and transfer to ensure data security. Cloud virtualization security, with a primary focus on hypervisor security, is discussed in detail by Orman in [49].

Approaches discussed above are providing a base for studying security attacks specific to each of the virtualization components. Accordingly, we need to consider the security issues at the Host level, Hypervisor (VMM) level, and VM level.

We have classified the host-based security under the category of virtualization security as the host machine is the base of Hypervisor and VMs. Host security issues have a serious impact on virtualization.

The exploitation of host is very challenging in the virtualization environment as it is not directly accessible from the end-users. Still, there are some cases like VM Escape (Guest-to-Host attack) and host level rootkits, that can damage the host. Although here are some proposals for VM Escape Attack like Cloudburst [50] for VMWare and Virtunoid [51] for KVM, a useful real-time implementation is yet not devised. However, approaches for maintaining hypervisor integrity are suggested by J. Wang et al. [52] and Z. Wang [53]. The mitigation method of VM Escape was suggested by Szefer et al.[54] also suggests eliminating hypervisor itself to make VM communicate to the hardware directly. The other way of exploiting the host machine is via kernel-level rootkits. Kernel level rootkits reside and operate from ring 0 and thus affect the operation of

system resources directly from the root level. Approaches mainly observing and handling control flow to identify such rootkits are proposed, which operate either from kernel-level or hypervisor level. Mitigation approaches to handle kernel-level rootkit includes State-Based Control-Flow Integrity (SBCFI) based kernel integrity monitoring system by Petroni et al. [55], VMM based memory shadowing approach NICKLE to identify unauthorized kernel code by Riley et al. [56], designing of tiny hypervisor ensuring kernel integrity by Seshadri et al. [57] and VMM based Patagonix to detect binaries' code modifications by Litty et al. [58]. Among the other approaches handling kernel-level rootkit, Srivastava et al. [59] propose to observe OS behavior by intercepting applications. Lycosid is another detecting tool proposed by Jones et al. [60] that uses guest OS information to prevent guest-initiated attacks. In another similar approach by Ficco et al. [61], guest OS kernel space was suggested to be observed with the RootkitDel tool to identify the affected area. The effect of the approaches preventing the execution of kernel rootkit was bypassed when Return-Oriented Rootkits [62] came into existence. It bypasses the mechanisms set for kernel code integrity protection. However, Hooksafe [63] was proposed to mitigate this malware by providing a thin hook indirection to trap each attempt trying to modify the code.

Attack to the Hypervisor is launched mainly by trapping and modifying the hyper/system calls. In the approach suggested by Bharadwaja et al. [64], an IDS Collabra is suggested to filter the malicious hyper calls. The validity of system calls is tested with the behavior analysis method by Y. Du et al. [65]. Accordingly, attack to hypervisor can be handled by filtering the system calls, but the things are different where hypervisor level rootkits are launched. Hypervisor-level rootkits like SubVirt [66] and Blue Pill [67] keep their identity hidden, and so, it is very difficult to be detected. However, as per [68], the hypervisor does not have full control over architectural components like Translation Look-aside Buffer (TLB), Branch Prediction, and Counter-based Clock using which the rootkits can be detected.

Analysis of the last category, i.e., VM security, includes Outsiders, VM Migration, and Cross-VM attacks. An Outsider attack is typically launched by one of the compromised services to other services or VMs. Although the concept of captive account may be introduced to restrict user rights [69], it is not that difficult to exploit the service running on the VMs to launch the Outsider attacks. Various VMM-based or dedicated-VM based IDSs/IPSs are suggested for detecting the Outsider attacks. The dedicated IDS/IPS was suggested to reside either on a separate privileged VM or on Dom 0 machine in Xen hypervisor system as per F. Zhao et al. [70] and K. Kourai et al. [71]. Among

the other approaches, hypervisor-based IDS/IPS were suggested by U. Tupakala et al. [72], Y. Zhang et al. [73], H. Jin et al. [74], A. Joshi et al. [75], and Jiang X. et al.[76]. On the other hand, a host-based VM logging system was suggested by G. Dunlap et al. [77]. A distributed IDS performs better because of the increased scope, and accordingly, M. Noura et al. [78] and Bryan D. Payne et al. [79] propose a distributed IDS for preventing the Outsider attacks.

Outsider attacks explicitly affect the performance of the system, where security during VM migration is also an important parameter to be considered for VM security. Live VM Migration requires mechanisms such that even attacked during the state transition is carried out, the state information must be protected. Parameters such as Virtual Trusted Platform, Live Migration Defense framework, and Role-based policies were suggested by Aiash et al. [80] for secure migration. A survey of VM migration security issues and solutions were presented by Kadam et al. [81]. A policy-based approach by Wang et al. [82] and an energy-aware cloud resource provisioning approach by Sammy et al. [83] is also efficient to support secure migration. A different context considering control, data, and migration planes to study issues raised during live migration was suggested by Oberheide et al. [84]. Apart from the above approaches, migration after data compression to lower downtime and migration of overloaded VM to prevent saturation were suggested by Jeincy [85] and Reeba et al. [86], respectively.

The third type of attack on the VM is a Cross-VM attack, which is launched from one VM to the other VM operating on the same host. There are basically two ways of launching the attack: the first is by hijacking/replaying the packets, and the other is by exploiting the resources shared among the co-resident VMs. Security loss during Cross-VM communication can be prevented by providing a dedicated path or imposing packet encryption. Various Inter-VM communication schemes are suggested in [87][88][89][90][91][92] with different level of support for user and kernel transparency, standard protocol support, isolation, and complete CM discovery, etc. Where they increase security with a dedicated path between pair of VMs, they do not eliminate the scope of attack completely. Approaches suggested in [93] and [94] discuss the features of each of the Inter-VM attacks in detail. Approaches are also proposed in the direction of increasing the efficiency and security of inter-VM data transfer operations in [95][96][97][98][99][100].

The above discussion reveals that there are a number of approaches leading to secure packet exchange mechanisms, but there are other Cross-VM security issues also that need proper handling. One such type of attack is Side-Channel Attack (SCA). The major vulnerability in the

Cross-VM case lies in simultaneous access to architectural resources like CPU cache, network queue, memory bus, and BPU (Branch Prediction Unit). Analysis of performance parameters of these shared resources to extract private key bits is the main idea behind the SCA.

The majority of the attack procedures employ explicitly looking malicious actions for launching the attack. They can be detected by imposing effective encryption mechanisms or by positioning IDS/IPS or firewall. On the opposite front, attack launching merely by profiling makes SCA very difficult to be detected. Thus, a unique attack launching method and challenge in detection make the SCA very interesting from the research point of view.

In an approach to launching SCA, Zhang et al. [101] claim of implementing Side-Channel Attack on a multi-core system. Analysis of the work done for handling SCA reveals that Side-Channel Attack on AES (Advanced Encryption Standard) loaded in the cache memory is focused in the majority of the research work. For removing AES specific issues, Sevak et al. [102] suggested following a confusion-diffusion approach for randomly selecting an encryption algorithm among DES (Data Encryption Standard), AES, and DES3. SCA was launched through the covert channel, the main media which provides a path between pair of isolated VMs. Yunjing Xu et al. [103] has discussed approach to improve the efficiency of L2 cache covert channel for preventing private key leakage. Similar approaches include mitigation methods of the cache timing attack proposed by Percival [104] and Bernstein [105]. Page [106] proposed a partitioned cache architecture, and Yu et al. [107] proposed a cache-based SCA detection approach that employs observation of resource utilization. For preventing attackers from retrieving confidential information, approaches like the addition of noise to cause delay and randomization while accessing memory are suggested by B. Mughal et al. [108]. One more approach in the area of cache attacks on AES has been explored very efficiently by D. Osvik et al. [109] that can extract the entire key. This is unlike other approaches where part of the key is known, and remaining key bits are extracted with time measurement, here the entire key can be extracted without any knowledge about the corresponding plaintext or ciphertext. With Evict + Time and Prime + Probe methods, it efficiently detects the attacks and also discusses possible countermeasures. Another approach detecting the presence of cache-based SCA on AES was proposed by Y. Kulah et al. that includes the detection of prime-and-probe attack, flush-and-reload attack, ECDSA and Flush+Flush attack [110].

The impact of cache memory in the isolated memory of VMs and the applicability of available solutions is interesting. However, we found the impact of BPU sharing more interesting and essential to be explored more as very limited work is carried out in handling the Branch Prediction Analysis (BPA) attack. It motivated us to explore the same in our research work.

1.3 Branch Prediction Analysis Attack

BPA attack is one such type of SCA where parts of Branch Prediction Unit (BPU) like Branch Target Buffer (BTB) and Branch Predictors are exploited to extract the private key. Asymmetric cryptographic algorithms like RSA (Rivest Shamir Adleman) and ECC (Elliptical Curve Cryptography) are common targets of this attack. Onur et al. [111] have suggested a BPA attack where there are four different ways to launch the attack. The first method, Direct Timing Attack (DTA), extracts the private key bits by simulating the behavior of the branch predictor. The other three methods, (Asynchronous, Synchronous BTB Eviction Attack and Trace-Driven Attack (TDA)) manipulate (either fill or clear) the BTB to predict the status of key bits. The reading of the hardware performance counter is carried out by all the four methods to perform the bit prediction. A discussion of existing solutions is carried out in Subsection 1.4.

1.4 Existing Solutions

Execution of conditional branch instruction in the Square and Multiply algorithm depends on the respective secret key bits [111] [112]. Further, a replacement of the Square & Multiply algorithm by Montgomery Ladder Algorithm [113] was suggested for the OpenSSL library, where the balanced branch feature eliminates the scope of BPA attack [114]. We need to modify each vulnerable library for the effective implementation of this solution. However, even if the Ladder algorithm is used for RSA implementation, a BPA attack is still possible as per S. Bhattacharya et al. in [115] and [116] where observation of a number of missed branches is carried out in place of CPU clock cycles. Additionally, S. Bhattacharya et al. [117] have also shown that even for RSA with Chinese Remainder Theorem (CRT) implementation, the BPA attack is possible.

Many researchers have proposed approaches to handle BPA attack. Among them, Agosta et al. [118] suggested to either eliminate or to replace the conditional branch instruction from the vulnerable cryptographic algorithms with indirect branch instructions. In another solution, Ya Tan et al. [119] suggested a mechanism for locking some of the BTB entries of the processes, which prevents the spy process from filling the entire BTB with branch instructions of the spy process. The

spy process fails to predict the bits as it cannot keep pace with execution flow by filling all the entries of the BTB. They also state that the attack is possible even with RSA blinding. Julien et al. [120] have proposed a blacklisting approach that allows only white-listed processes to access the hardware counters. Prohibiting the access of performance parameters prevents the launching of a BPA attack as it needs to read CPU time or missed branch instructions. In a mitigation technique for BPA with DTA, S. Bhattacharya et al. [121] have proposed an approach to manipulate dynamic predictors. They have suggested executing a randomization module in concurrence to the compromised process alters the state of BPU because of which the compromised process lose its correlation with the key. Suggested approach prevents of BPA attack launched with the DTA method.

Analysis of the approaches handling BPA attack reveals that the majority of the approaches manipulate the functioning of the components like BTB, Branch Predictors or Performance Counters. Manipulation of the architectural components affects normal system functioning.

Existing work done in the area of BPA attack and their limitations has led us to define the problem statement of our work, as presented in Section 2.

Our study also revealed that Onur et al. [111] have just shown the theoretical possibility of four types of BPA attacks. Neither they nor other researchers [112], [113], [115]–[121] have shown any result of the successful launching of all types of BPA attacks on non virtualized or virtualized environment. Hence, we decided to explore the possibility of simulating BPA attacks in the virtualization environment.

2. Problem Definition

BPA attack suggested by Onur et al. [111] considers a non-virtualization environment where both the spy and victim processes reside on the same system. They have just suggested that the attack is possible even in the presence of virtualization and sandboxing but did not provide any other information. Solution approaches [118]–[121] also lacks in giving useful information about their effectiveness in the virtual environment.

Thus, we found that a detailed study is needed to explore the scope of the BPA attack and the effectiveness of existing solutions in a virtual environment. We also found the necessity to overcome the limitations of existing solutions and work out robust, accurate, and independent solutions to handle the BPA attack in virtual environments.

3. Objective and Scope of the work

We define the objective and scope of our work as:

1 Assessing the scope of BPA attack in a virtualization environment
2 Identifying the applicability of existing approaches to handle the Cross-VM BPA attack
3 Carrying out a detailed study of BPA attack launching methods
4 Simulating BPA attack in virtual environments
5 Working out a new solution to overcome the limitations of existing solutions
6 Working out testbed to examine the effectiveness and accuracy of our solution
7 Carrying out various experiments for proving effectiveness and accuracy.

4. Related Work

We have carried out a detailed study on the scope of BPA attack in the virtualization and applicability of existing solutions in a virtualized environment.

4.1. Scope Assessment of BPA attack in Virtualization

Virtualization hides the internal details of the underlying resources from the end-users. However, an attacker can manage to place a VM co-resident to a target VM [122]. Such attacker VM or a compromised VM can launch a BPA attack on the other co-resident VM, giving rise to the Cross-VM BPA attack. However, the probability of a Cross-VM BPA attack and possible attack mechanisms depend on how VMs are sharing the hardware (CPU core) and software (Cryptographic Library) resources.

VMs host multiple services for robust management and effective utilization of the resources. Each machine is treated as an individual machine having the required resource. VM technology can optimize between sharing and isolation of resources. The security of this environment depends on the configuration adopted by the system administrator for hosting multiple services. Higher sharing enables better resource utilization, whereas strict isolation provides better security.

If a VM is isolated by allocating dedicated hardware and software resources, then the possibility of a Cross-VM BPA attack is completely eliminated. However, from the experience of experts working on Vmware or Citrix, as well as from the personal experience of KVM, it has been found very difficult to configure a VM with dedicated resources. Additionally, even if such

VM is configured, the primary features of cloud technology like load balancing and fault tolerance would not be supported. Hence, practically this option is rarely preferred by VM administrators. In most prevailing options, the virtualization administrator allocates dedicated memory size, CPU frequency, and disk space size as security best practices. In such a scenario, CPU cores are generally shared by multiple VMs. The BPA attack is possible even if either CPU core or Cryptographic Library is shared. The underlying attack method plays a significant role in such a case to decide applicability.

Among the four attack launching methods, the first method, DTA, extracts unknown key bits by simulating the behavior of the branch predictor. It requires a common cryptographic library between the spy and victim processes. The other three methods launching BPA attack manipulate BTB for tracking the behavior of the victim cryptographic process.

Our analysis suggests that the Cross-VM BPA attack can be launched by the three methods other than DTA, only if the CPU core is shared. The Asynchronous BTB Eviction method also needs a shared cryptographic library. Additionally, if the two VMs have a separate core but shared library, then the Cross-VM BPA attack is possible with only the DTA method. Table 1 presents a summary of the discussion. The scope of existing solutions to handle the Cross-VM BPA attack is discussed in Subsection 4.2.

Table 1 Scope of BPA Attack Methods for Cross-VM BPA

BPA Attack Mechanism	**Required Sharing Configuration**	
	Cryptographic Library	**CPU Core and BTB**
Direct Timing Attack	Shared	Shared / Separate
Asynchronous BTB Eviction Attack	Shared	Shared
Synchronous BTB Eviction Attack	Shared / Separate	Shared
Trace-Driven Attack	Shared / Separate	Shared

4.2. Applicability of Existing Solutions

Scope analysis of present solutions revealed that except Julien et al.[120], the majority of the above solutions are suggested for traditional server environments. Although other solutions can also be applied in virtualization in their direct form, each of them has its own limitations.

The approach suggested by Julien et al. [120] can be directly applied to handle the Cross-VM platform irrespective of the underlying attack mechanism and sharing configuration. However, the

proposed approach may fail if white-listed processes like HTTPS (Hypertext Transfer Protocol Secure) and SFTP (Secure File Transfer Protocol) get compromised. Suggested algorithmic changes by Agosta et al. [118] can address both common and separate core Cross-VM BPA attack irrespective of the employed attack mechanisms. However, suggested preventive steps are required to be implemented in each of the vulnerable algorithms. A direct application of a solution by Ya Tan et al. [119] is possible only for a Trace-Driven attack on a common core Cross-VM platform. Additionally, the suggested approach may lead to a false positive. Mitigation technique suggested by S. Bhattacharya et al. [121] can work for common as well as separate core Cross-VM BPA attacks launched with the DTA method. However, it affects the performance of the legitimate processes that have a large number of conditional instructions. The summary of the behavior of existing approaches is presented in Table 2.

Table 2 Performance Comparison of Existing Solutions

Approach	Targets Virtualization?	Applicable to handle Cross-VM BPA?	Focused BPA mechanism	Affects Legitimate Process?	Dependent on Cryptographic Algorithm?
Agosta et al.[118]	No	Yes	Independent of Attack Mechanism	No	Yes. requires modification in each vulnerable algorithm
Tan et al. [119]	No	Only for VMs with common core	Trace-Driven Attack	Yes	No
Julien et al. [120]	Yes	Yes	Independent of Attack Mechanism	Yes	No
S. Bhattacharya [121]	No	Yes	Direct Timing Attack	Yes	No

The above analysis reveals that we need a solution to handle a Cross-VM BPA attack such that irrespective of the targeted public key cryptographic algorithm, BPA attack must be prevented without affecting the performance of legitimate activities. The handling of the BPA attack requires consideration of all the four methods. In that regard, a brief overview of the underlying attack methodologies is given the next subsection.

4.3. BPA Attack Launching Methods

Description of the attack procedures considers the virtualization environment, and accordingly, two co-resident VMs are assumed to run the victim and spy processes. We have considered RSA as the target process for our work. BPA attack procedures are explained briefly in the following subsections.

4.3.1. Direct Timing Attack

DTA performs simulations over known data sets to monitor the behavior of the target process. The output of the branch predictor is compared with the actual behavior of the conditional branch instruction for some known bits of the secret key for a large number of encrypted messages. The large message set M is divided into four sets, namely M1, M2, M3, and M4, based on the results of the comparison. Further, the total number of missed branches are observed during the decryption of messages in each of the message set. Finally, the next bit is predicted from the following equation, where BM represents the total number of missed branches.

$$if\ (avg(BM(M_1)) > avg(BM(M_2)) \text{ and } avg\big(BM(M_3)\big) < avg(BM(M_4))) \quad \text{then} \quad di = 1 \\ else\ \ di = 0 \tag{1}$$

The above process is repeated until all the unknown bits are extracted [111].

4.3.2. Asynchronous BTB Eviction Attack

Prediction of secret key bits in Asynchronous BTB Eviction Attack is carried out by executing a dummy process that frequently evicts all / a set of BTB entries where RSA process maps. Frequent clearance of the BTB entries leads to a BTB miss event on every call of that target instruction. Similar to the DTA method, simulation of exponentiations is carried out to predict the secret bits where the partitioning is done based on whether the target branch is taken during the computation of m^2 (mod N) or not.

4.3.3. Synchronous BTB Eviction Attack

Synchronous BTB Eviction Attack executes a dummy process similar to the Asynchronous Attack with only one difference that the dummy process runs in synchronism to the RSA process as the name specifies. The dummy process clears BTB entries just before the i^{th} step of the RSA

process. In turn, the next bit (bit i) is predicted based on the observed execution time of the dummy process.

4.3.4. Trace-Driven Attack

In Trace-Driven Attack (Time-Driven Attack) the spy process is comprised of a large number of conditional branch instructions. The total number of conditional instructions are selected to fill the entire BTB or to fill the BTB set where the victim crypto process is executing. The spy process starts executing before the victim crypto process executes. The spy process continuously executes and fills the target BTB entries. Hence, when the conditional branch instruction of the RSA algorithm executes, the corresponding target address would not be available in BTB and would be required to be fetched from memory. The entry of the target address of the target branch in BTB results in the eviction of one of the BTB entries of the spy process. During this time, the spy process continuously measures its execution time, so when one of its branch target addresses is evicted, it finds a considerable time difference. Hence, when conditional branch instruction of the RSA algorithm gets executed, it is reflected in the execution time of the spy process, and accordingly, the presence of bit 1 in the secret key is predicted by the attacker [111].

5 Original Contribution

5.1 Attack Simulation

We started our practical research work by simulating the BPA attack. The simulation was done on KVM VMs operating on the Ubuntu operating system.

5.1.1 Direct Timing Attack

The attack environment for DTA simulation is shown in Fig. 2. As shown in the figure, VM2 launches DTA on VM1, where secure communication is going on between VM1 and Machine A. As per the procedure of Direct Timing Attack, simulation is carried out on a set of around 1000 ciphertexts. Results obtained from the number of missed branches during the extraction of the 50^{th} bit are represented in Fig. 3. The bit is correctly predicted from the observed results as per equation 1.

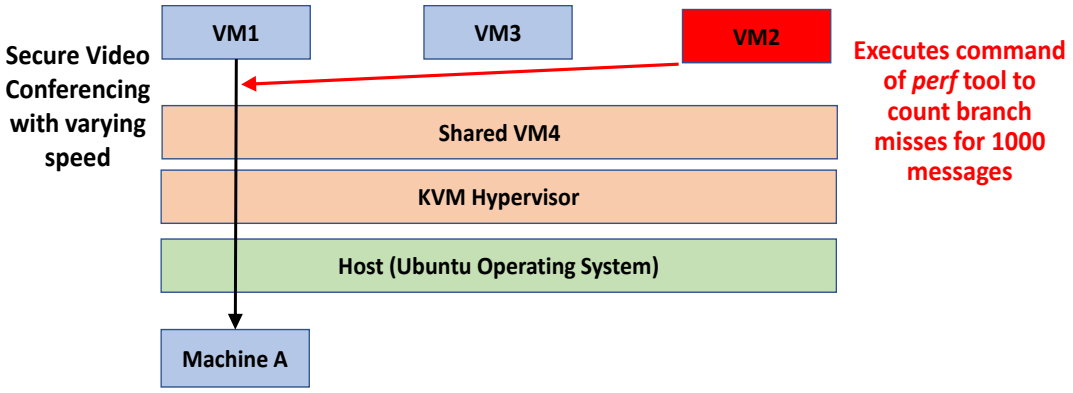

Fig. 2 Attack Environment: Direct Timing Attack

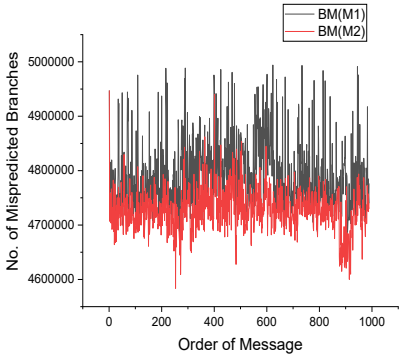

Fig. 3(a) Distribution of BM (M1) and BM (M2)

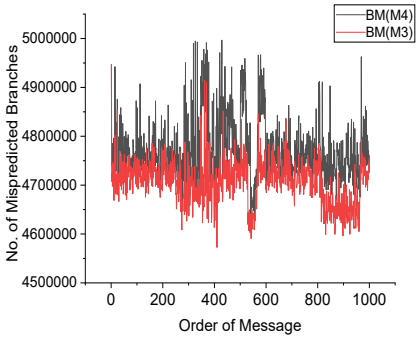

Fig. 3(b) Distribution of BM (M3) and BM (M4)

We have counted an average number of missed branches instructions by executing the following command with *perf* [123] tool.

$perf stat -e branch-misses java RSA_File_to_Monitor msg_to_decrypt

Although our simulation was in the line of [111] and [116], which claimed 100% success, our prediction accuracy was found to 97-98%. However, our objective is to simulate the attack rather than stealing the data, and hence 100% accuracy is not needed.

5.1.2 Trace-Driven Attack

We have used the Gem5 simulator [124] for simulating the TDA attack. Gem5 is an open-source multithreaded simulation platform. The attack environment is shown in Fig. 4. Spy process on VM2 continuously measures the execution time of the RSA process as per the attack procedure.

Obtained results plotted in Fig. 5 represents the number of clock cycles (execution time) of the spy process in accordance with the execution of the conditional branch instruction of the RSA algorithm. The plot represents that the spy process takes more time in execution for the positions where the RSA algorithm encounters bit 1 in the private key. However, time taken for the rest of the places, corresponding to bit 0, is considerably low. As shown in the diagram, a bit sequence is predicted based on the observed execution time.

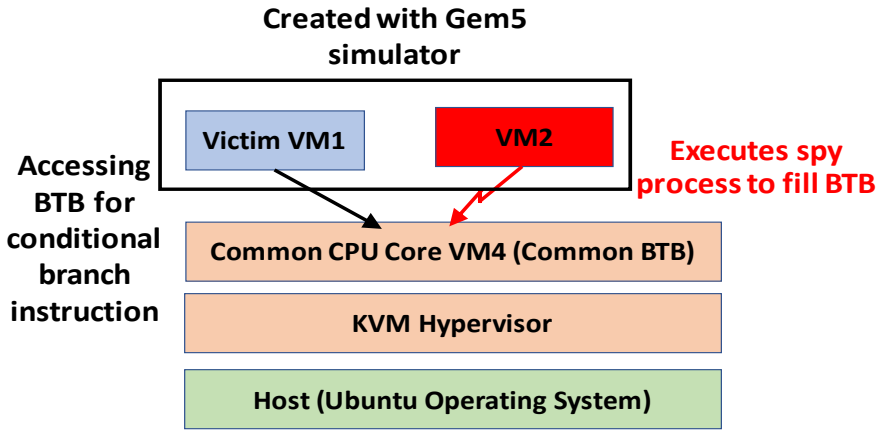

Fig. 4 Attack Environment of TDA

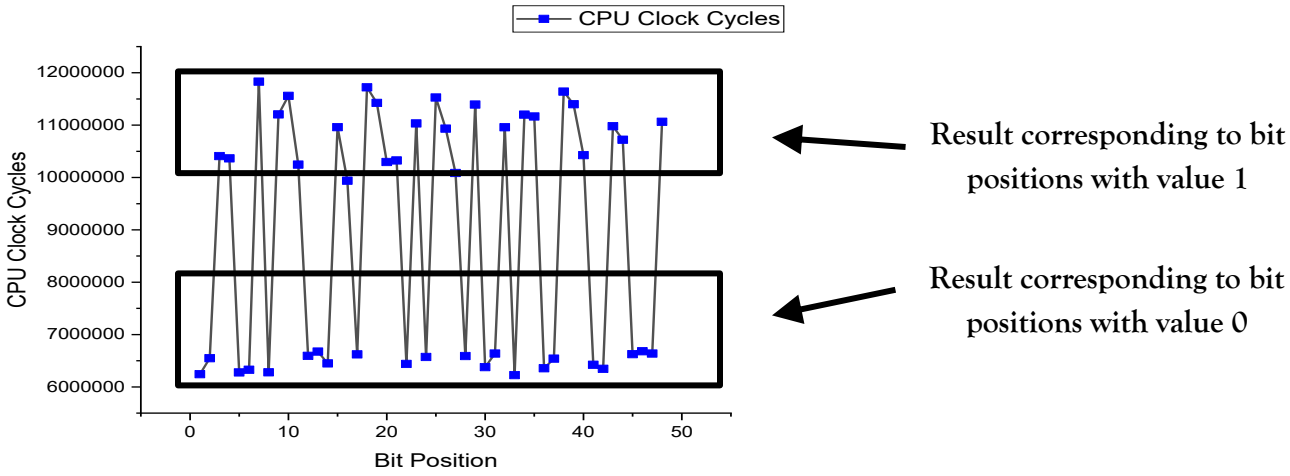

Fig. 5 Results of Simulation for Trace-Driven Attack for total 50 bits of key

5.1.3 Asynchronous and Synchronous BTB Eviction Attacks

Method to launch Asynchronous BTB Eviction Attack (ABEA) is similar to that of DTA with only one difference. As discussed in Section 4, the large number of messages taken for simulation are distributed among four sets based on the result of branch misprediction in the case of DTA. Instead, the actual status of branch instruction execution is considered while distributing the messages. Additionally, one dummy process is executed simultaneously to evict the BTB entries so that the execution time gets affected. Further, it helps in correctly predicting the private bits.

Although we have partially simulated ABEA in the virtualization environment, it is also essential to analyze whether the attack is possible in the presence of the Ladder algorithm. The simulation of Cross-VM Synchronous BTB Eviction Arrack (SBEA) is very difficult as it needs perfect synchronism between two processes running on different VMs. However, even if the BTB eviction arracks are launched successfully, it is shown further in Section 6 that the proposed approach can detect the BPA attack irrespective of the underlying attack method.

Obtained results reveal that it is possible to launch both DTA and TDA on the Cross-VM platform. The study of the BPA attack procedures has provided a base to identify the primary actions performed by the attacker. Our solution is a behavior-based approach that detects the presence of the BPA attack by monitoring and validating the identified actions employed by the attacker. A brief discussion of the fundamentals details of the suggested approach is provided in Subsection 5.2.

5.2 Behavior-Based Approach

As an initial task of proposing a behavior-based approach, careful observation of the attack procedures was done. During the observation, the spy process of DTA was found to read the total number of missed branches during the decryption of a large set of messages. This action requires frequent access to cryptographic process RSA as well as needs a frequent reading of the Hardware Performance Counters (HPCnt). On the other hand, the spy process of TDA was found to carry out frequent access to BTB for continuous filling. Moreover, observation of the execution time carried out during TDA also needs to read the HPCnt values like DTA. Observation of primary actions

performed during the other two attack methods revealed that the behavior of the SBEA method is similar to the TDA method. Additionally, the ABEA method exhibits the combined primary actions of DTA and TDA methods.

The above actions can be employed by legitimate processes like application profiler or secure video conversation also. However, these actions are performed with a considerably high frequency by the DTA/TDA process, and so we need to assess the action frequency to identify the legitimacy of the process. At the same time, detection of malicious VM just by observing the access frequency of the RSA algorithm or BTB, and HPCnts may lead to false positives also. Hence, one more essential point to be considered here is the usage of the extracted key.

As a result of DTA or TDA, the Private Asymmetric Key (PAK) of the target crypto algorithm, i.e., RSA, becomes available to the attacker. Once PAK is available, the attacker can get Private Symmetric Key (PSK). This is with reference to the process of key exchange where PAK is used to encrypt PSK. All the confidential data is encrypted by this PSK. Hence, an obvious step that an attacker takes after stealing PSK is to trap the data packets for meaningful attacks by launching a MITM (Man-In-The-Middle) attack ***[125][126]***.

Summarizing the above discussion, frequent access of RSA (during DTA and ABEA), frequent filling of BTB during (TDA, ABEA, and SBEA) and frequent reading of performance counters followed by packet trapping by all the four methods are the four basic actions performed by the attacker. By keeping these actions into consideration, a four-eyed solution *Chaturdrashta* is presented, which is discussed in Subsection 5.3. The solution is designed to detect the two most common types of the BPA attack, DTA and TDA. However, it would be shown further that the *Chaturdrashta* can also detect the presence of the ABEA and SBEA in its original form.

5.3 *Chaturdrashta*: The Proposed Approach

The *Chatursrashta* is designed to detect the presence of a BPA attack launched either by DTA or TDA on the Cross-VM platform. It is comprised of four monitors, namely, CryptoLibrary Access Monitor (CAM), BTBAccess Monitor (BM), Interrupt Monitor (IM), and Network Monitor (NM). The suggested four monitors observe the four primary actions discussed in Subsection 5.1. CAM monitors the access frequency of the target cryptographic library (RSA), while BM keeps track of BTB occupancy ratio of running processes. IM is configured to count the total number of times a

process reads the value of performance counters, while packet trapping activities are observed by the NM.

These four monitors are grouped to form two separate logical units, namely, *Trilochan* and *Trinetra*. *Trilochan* and *Trinetra* are designed to detect the presence of DTA and TDA respectively which are discussed in the Subsections 5.3.1 and 5.3.2.

5.3.1 *Trilochan*: Solution to Detect Cross-VM DTA

As per the primary actions performed during the DTA procedure, *Trilochan* is comprised of three of the four monitors: CAM, IM, and NM. CAM and IM are initiated concurrently to observe RSA access frequency and performance counter access frequency of all the running *trustworthy* (initial state of VMs) VMs. On observing a very high frequency compared to the rest of the VMs, CAM and IM independently declare a VM as *suspicious*. A VM is declared as *spy* provided it is declared as *suspicious* by both CAM and IM. The NM is turned ON only for the *spy* VM, and if it finds the VM trapping data packets, then the VM is declared as *malicious*. Finally, the *malicious* VM is blocked. Experimental analysis on *Trilochan* is discussed in Subsection 6.2.

5.3.2 *Trinetra*: Solution to Detect Cross-VM TDA

Comparison of the actions taken while launching DTA and TDA identifies that only one of the three primary actions is different in both the procedures. Where frequent access of RSA is carried out during DTA, frequent access of BTB is a pioneer task employed by TDA. Hence, the place of CAM is taken by BM in *Trinetra* to observe BTB access frequency. The other two actions observed by IM and NM are common in both cases. Accordingly, *Trinetra* is comprised of BM, IM, and NM. The way of working of *Trinetra* for identifying a *malicious* VM is similar to *Trilochan*. *Trinetra* can also successfully detect the presence of TDA like *Trilochan,* which is presented in Subsection 6.2.

5.3.3 *Trilochan and Trinetra*: Extended Scope to Detect of BTB Eviction Methods

Primary actions of ABEA and SBEA discussed in Subsection 5.2 represent their coincidence with the actions of DTA and TDA, respectively. Accordingly, *Trilochan* and *Trinetra*, which are designed to detect the presence of DTA and TDA, can also detect the presence of ABEA and SBEA in their original form.

A combined model of both *Trilochan* and *Trinetra,* as well as the local flow of underlying procedure, is shown in Fig. 6(a) and Fig. 6(b), respectively.

Both *Trilochan* and *Trinetra* are host-based systems. As users do not have direct access to host in the virtualization environment, there are rare chances of these systems of getting compromised. Moreover, kernel embedded implementation of CAM and implementation as a Linux service of the other three monitors reduces the scope of *Chaturdrashta* getting compromised. Attack simulation and experimental analysis of proposed approaches are discussed in Section 6.

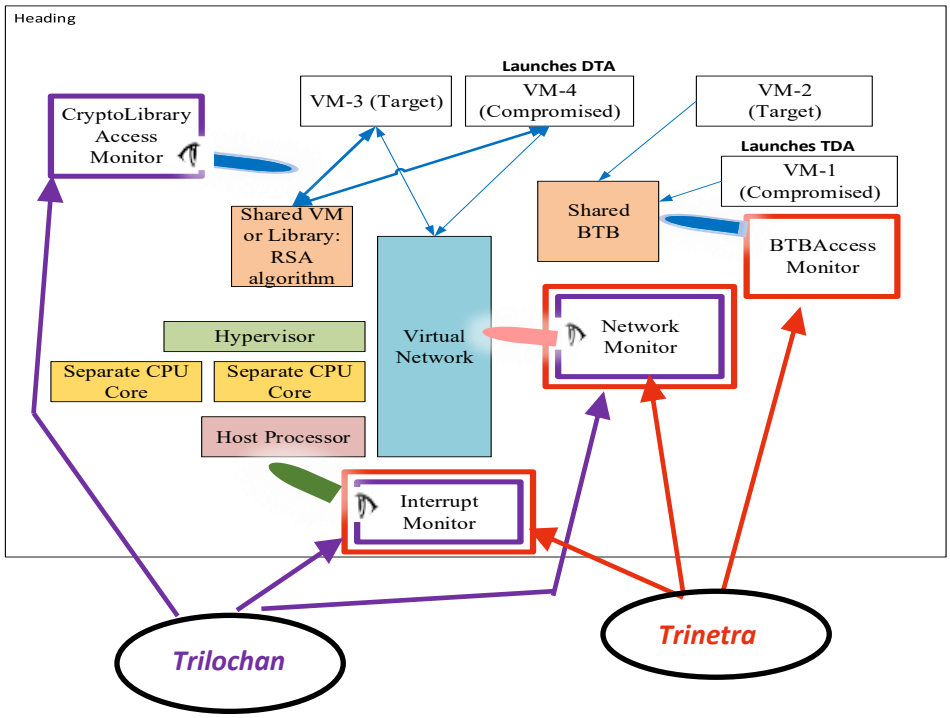

Fig. 6(a) Model of *Trilochan* and *Trinetra*

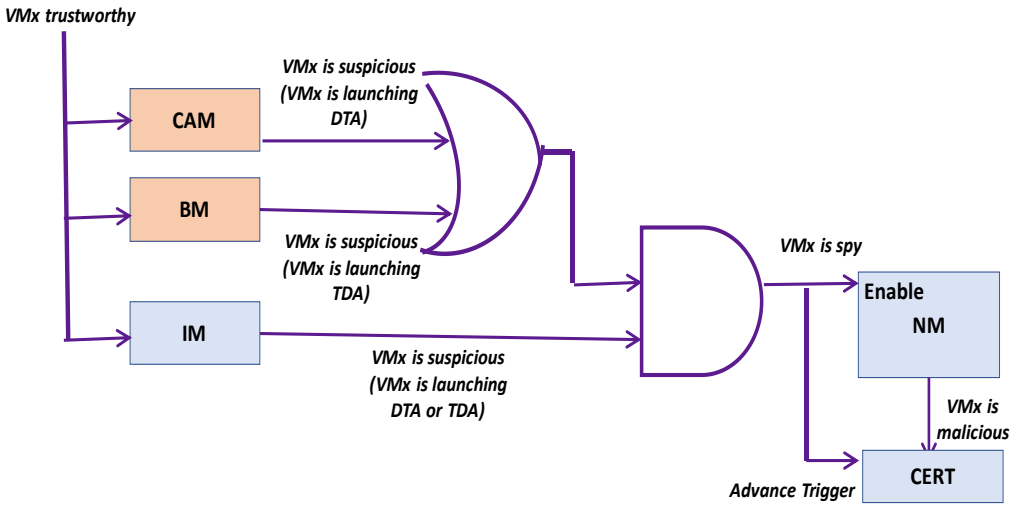

Fig. 6(b) Logical Flow of *Chaturdrashta* (combined *Trilochan* and *Trinetra)*

6 Testbed and Results

As a pre-requisite of implementing the proposed solution, we have carried out attack simulation of Cross-VM DTA and TDA, as discussed in Subsection 5.1. This section discusses the results obtained from the experimental analysis. The simulation environment and experimental analysis are discussed in Subsections 6.1 and 6.2, respectively.

6.1 Simulated Operation Environment

We have deployed KVM VMs on the Ubuntu operating system for the attack simulation of DTA and TDA. VM1 is set to initiate secure communication with another machine A. VM2 is a compromised machine that is launching DTA or TDA for extracting the private key. Launching of DTA requires a shared VM holding cryptographic library for which we have created VM3. The simulation environment is similar to that of attack simulation, which is represented in Fig. 2.

6.2 Experiments and Results

We have implemented *Trilochan* and *Trinetra* on a Linux based system. Implementation of each of the four monitors, i.e., CryptoAccess Library Monitor (CAM), BTBAccess Monitor (BM), Interrupt Monitor (IM), and Network Monitor (NM), is carried out as a separate entity, where their results are combined as per Fig. 6(b), to take the final decision. The following discussion provides the implementation details along with the obtained results of each of the monitors.

As per the functional requirement of CAM, we implemented it with *tcpdump* tool to count the packet traversal frequency between the shared VM holding cryptographic library and all the other VMs. Application Profiler, Secure VC, and DTA attack procedures were initiated on different VMs to observe and compare packet traversal frequencies of different processes. The experiment was also performed in the presence of the ABEA method. The combined results are shown in Fig. 7(a), which clearly represents that the packet traversal frequency between the VM launching DTA/ABEA is much more than any other VM. We also initiated secure VC by taking different transmission speeds using *trueconf*, where results in Fig. 7(b) show that the observed frequency of the DTA process is much more than the frequency observed for the highest transmission speed.

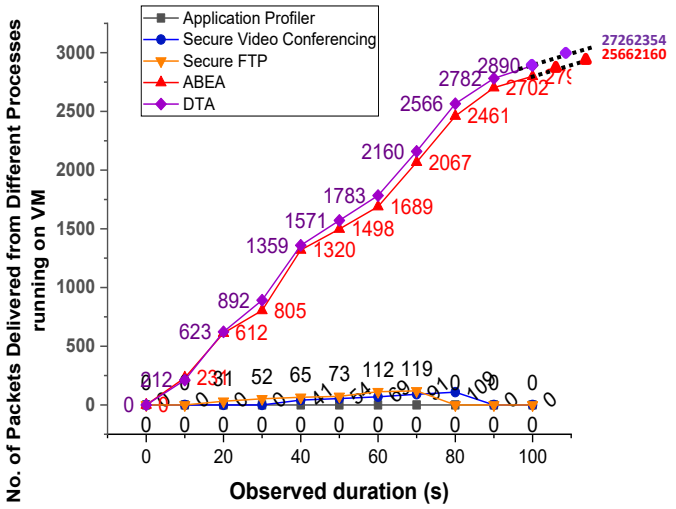

Fig. 7(a) Results for Packet Monitoring

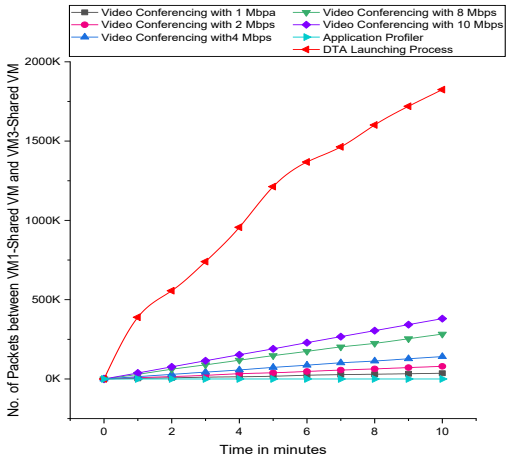

Fig. 7(b) Results for Packet Monitoring for Video Conferencing with varying speed

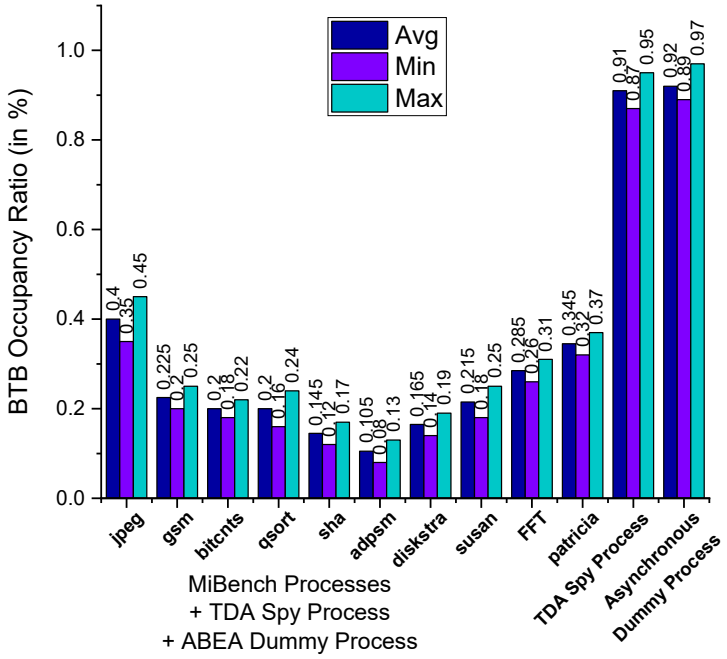

Fig. 8 BTB Occupancy Ratio observed by BM

The second monitor, BM, is implemented with the Gem5 simulator that measures the occupancy of different processes. Sample algorithms of MiBench [127] are simulated for multiple times results are plotted for the maximum, minimum, and average values in Fig. 8. The plot reveals that the occupancy ratio of TDA and ABEA is considerably more than the other processes. The average value obtained for the spy process of TDA and ABEA is taken as a threshold by the BM.

Hence, a VM with a process with an occupancy ratio higher than the obtained threshold is recommended as *suspicious* by the BM.

Implementation of IM is done by applying a patch in the kernel level. We have modified the code of kernel function *native_read_pmc* to count the total number of times the reading activity of the hardware performance counter is carried out by a process. When the counter value hits a specific threshold value, the corresponding process ID is written as a kernel log. We measure the time taken by different running processes to hit the counter threshold. The experiment was carried out by setting different threshold values like 10, 20, 30, and 40. The obtained result is shown in Fig. 9. The results reveal that compared to other running processes of Linux, processes initiating DTA and TDA hit the counter threshold within considerably less time. Additionally, the number of processes hitting the counter threshold decreases as we increase the threshold value. We find a log of only DTA, TDA, and ABEA processes for counter threshold 40. Hence, 40 can be chosen as the final threshold for detecting processes accessing performance counters with high frequency.

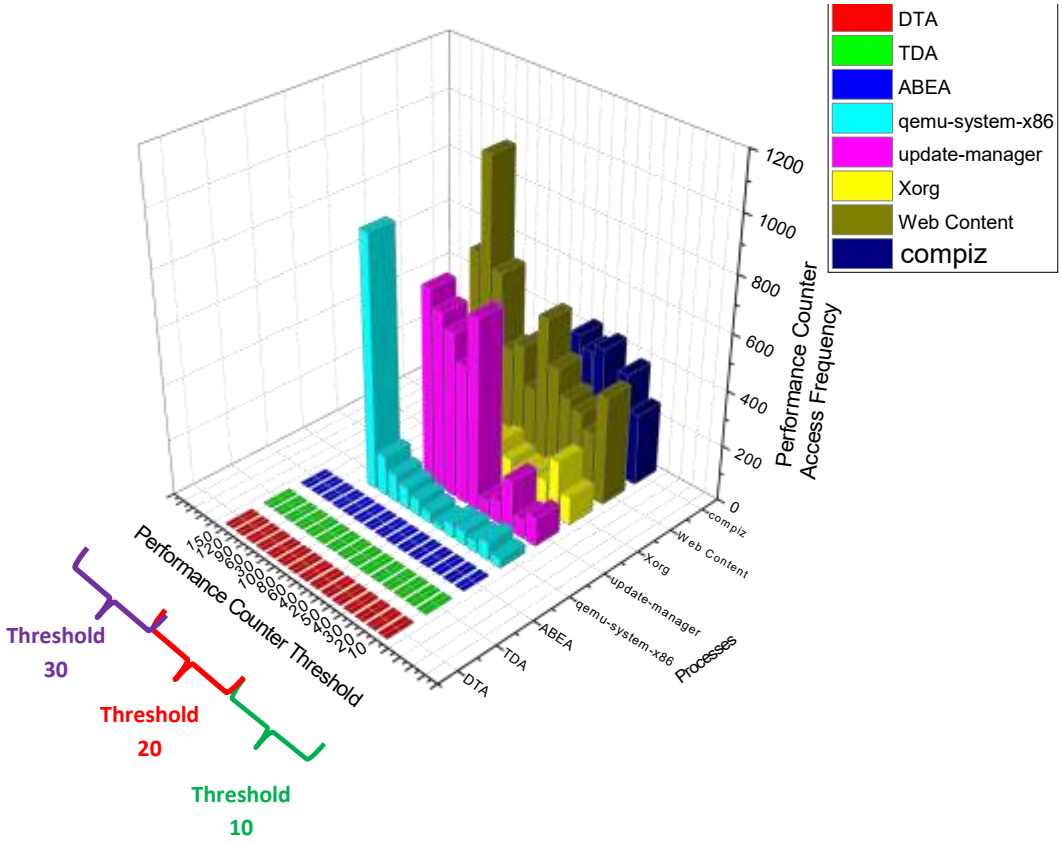

Fig. 9 Observations of Interrupt Monitor

NM is meant to track packet trapping activity for which there are multiple ways of implementation. We have employed ARPSpoofing to launch a MITM attack as per which, if NM observes duplicate entries of the MAC address of *spy* VM, then the status of the *suspicious* VM is changed to *malicious*.

We initiated CAM, BM and IM followed by launching the DTA, TDA, and ABEA process where it was found that CAM, BM, and IM generate the alert for the VM launching the BPA attack (DTA/TDA/ABEA) by the time when hardly 8-10 bits are detected. Additionally, none of the monitors among CAM, BM, or IM catches legitimate Linux processes like Application Profiler, Secure VC, Xorg, update manager, or Web Content. However, even if either of the CAM, BM or IM declares a legitimate process as *suspicious*, and then as *spy*, the process will never initiate malicious action of packet trapping. Hence, the process will not get caught by NM and would be prevented from being wrongly declared as *malicious*. From the above discussion, it can be stated that the false positive percentage for both *Trilochan* and *Trinetra* is extremely low (almost zero).

We evaluated the performance overhead of *Trilochan* as well as *Trinetra* by monitoring the free memory, CPU usage, and interrupts encountered. The obtained results are shown in Fig. 10, which is found within 1% for both the algorithms.

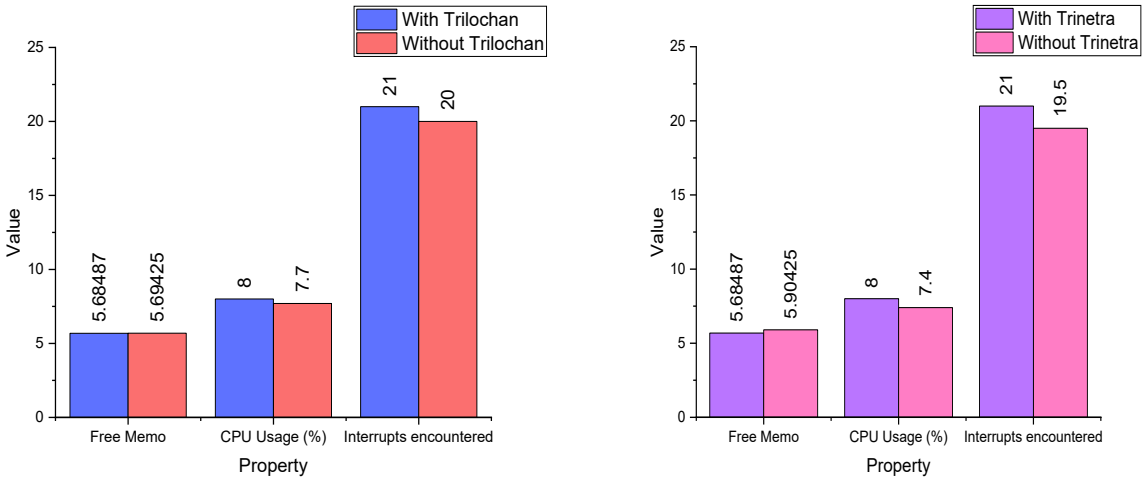

Fig. 10 Performance Evaluation

7 Achievements

Identification of problems in the area of cloud security itself is the biggest challenge looking to the quantity and quality of work done targeting various security issues. Our literature survey began with the study and analysis of generic as well as virtualization security issues, and ended

with the summary showing the research gap in the area of Cross-VM Branch Prediction Analysis attack. We found that none has studied BPA attack as a Cross-VM attack where even scope of the attack is also an essential parameter to be considered. We analyzed the Cross-VM BPA attack from an absolutely different angle by taking virtualization features and resource sharing configuration into account. Our study also revealed that Onur et al. [111] had shown the possibility of four types of BPA attacks in a non-virtualized environment. Neither they nor other researchers [112], [113], [115]–[121] have shown any result of the successful launching of all types of BPA attacks on a virtualized environment. We carried out experiments in successfully simulating two types of BPA attacks in the virtualization environment. During the simulation, we found that the third type of BPA attack, i.e., ABEA, requires more analysis regarding its applicability for the Ladder algorithm. The simulation of the fourth type of BPA attack, SBEA, is theoretically possible but extremely difficult to execute in practical conditions.

Our analysis resulted in mapping between virtualization resource sharing configuration and the applicable BPA attack launching method. Further, we also found from the study of existing solutions handling BPA attack that there is a need to propose a novel solution such that the normal functioning of the system does not get affected as well as false positives do not rise.

We have proposed a solution, *Chatudrashta,* to detect the presence of a BPA attack. We have considered the most probable two out of the four attack methods, DTA and TDA, for our work. *Chaturdrashta* is logically divided into *Trilochan* and *Trinetra* to detect the presence of DTA and TDA, respectively, in the Cross-VM environment. We implemented both the approaches at the host level to reduce the chances of getting compromised. We had also studied kernel source code to understand the logical flow of the system call reading HPCnts. Experimental analysis of both *Trilochan* and *Trinetra* reveals that they can successfully detect the presence of DTA and TDA respectively by the time when a very few bits are predicted. The detection does not result in any false positive and takes overhead lower than 1%. Additionally, *Chaturdrashta* was also found to be able to detect the presence of a BPA attack even when it is launched with ABEA or SBEA method.

8 Conclusion and Future Work

The virtualization layer is the backbone of the Cloud Computing environment. Hence, we found it very essential and interesting to study security issues in the virtualization environment. A detailed literature survey on various cloud security issues with a primary focus on virtualization revealed that there is a research scope in the area of the Cross-VM Branch Prediction Analysis attack, a type of SCA. An in-depth study of the BPA attack had thrown light on a very important aspect that the attack launching method and the resource sharing configuration are very crucial parameters to be considered for assessing the scope of Cross-VM BPA attack. We carried out scope assessment of the Cross-VM BPA attack and identified the applicability of different BPA attack methods.

Study of existing solutions handling BPA attack and their analysis to check their applicability in the Cross-VM platform presented that there is a need to propose a new approach to handle the Cross-VM BPA attack. We also analyzed that the new solution must not affect the normal system functioning as well as it should be independent of the cryptographic algorithm. Our proposed solutions *Trilochan* and *Trinetra* (Combinedly known as *Chaturdrashta*) can successfully detect the presence of DTA and TDA attacks respectively, with a very negligible overhead of 1%. The approach can detect the presence of the BPA attack by the time when a very few bits are extracted without resulting in any false-positive.

We have targeted most-commonly employed DTA and TDA methods while designing *Chaturdrashta*. However, the behavior analysis of the attack methods revealed that the primary actions of the Asynchronous and Synchronous BTB Eviction methods coincide with those of DTA and TDA. The resemblance in the actions reveals that the *Chaturdrashta* can detect the presence of all the four types of BPA attacks in its original form. Moreover, its applicability can be further extended for a non-virtualization environment also just by a small change in the design. Simulation of BTB Eviction methods and revision of *Chaturdrashta* for its compatibility in the non-virtualization environment is the future scope of our work.

9 Our Publications

1 Published

a. D. H. Buch and H. S. Bhatt, “Taxonomy on Cloud Computing Security Issues at Virtualization Layer,” International Journal of Advanced Research in Engineering and Technology. (IJARET), vol. 9, no. 4, pp. 50–76, 2018.

b. D. H. Buch and H. S. Bhatt, “Cross-VM Branch Prediction Analysis Attack : Scope Assessment and Simulation,” International Journal of Recent Technology and Engineering (IJRTE), vol. 8, no. 2, pp. 4868–4873, 2019.

c. D. H. Buch and H. S. Bhatt. "*Trinetra*: a solution to handle cross-VM time-driven attack." SN Applied Sciences, vol. 2, no. 524, pp. 1-12, 2020.

2 Under Review

D. H. Buch and H. S. Bhatt, “*Trilochan*: A Solution to detect the presence of Cross-VM Direct Timing Attack” in Journal of Software: Practice and Experience.

References

[1] Khan Minhaj Ahmed, M. A. Khan, and Khan Minhaj Ahmed, "A survey of security issues for cloud computing," *Journal of Network and Computer Applications*, vol. 71, pp. 11–29, 2016.

[2] N. Gonzalez *et al.*, "A quantitative analysis of current security concerns and solutions for cloud computing," *Journal of Cloud Computing: Advances, Systems and Applications*, vol. 1, no. 1, p. 11, 2012, doi: 10.1186/2192-113X-1-11.

[3] N. Gruschka and M. Jensen, "Attack surfaces: A taxonomy for attacks on cloud services," in *Proceedings - 2010 IEEE 3rd International Conference on Cloud Computing, CLOUD 2010*, 2010, pp. 276–279, doi: 10.1109/CLOUD.2010.23.

[4] M. K. Srinivasan, K. Sarukesi, P. Rodrigues, M. S. Manoj, and P. Revathy, "State-of-the-art cloud computing security taxonomies - A classification of security challenges in the present cloud computing environment," *Proceedings of the International Conference on Advances in Computing, Communications and Informatics - ICACCI '12*, p. 470, 2012, doi: 10.1145/2345396.2345474.

[5] L. Coppolino, S. D'Antonio, G. Mazzeo, and L. Romano, "Cloud security: Emerging threats and current solutions," *Computers and Electrical Engineering*, 2015.

[6] S. Singh *et al.*, "A survey on cloud computing security: Issues, threats, and solutions," *Journal of Network and Computer Applications*, vol. 75, pp. 200–222, 2016, doi: 10.1016/j.jnca.2016.09.002.

[7] S. M. S. Hashemi and M. Ardakani, "Taxonomy of the Security Aspects of Cloud Computing Systems-A Survey," *Networks*, vol. 4, no. 1, pp. 21–28, 2012.

[8] G. Pék, L. Buttyán, and B. Bencsáth, "A Survey of Security Issues in Hardware Virtualization," *ACM Computing Surveys (CSUR)*, vol. 45, no. 3, pp. 40:1-40:34, 2013, doi: 10.1145/2480741.2480757.

[9] C. N. Modi and K. Acha, "Virtualization layer security challenges and intrusion detection/prevention systems in cloud computing: a comprehensive review," *The Journal of Supercomputing*, pp. 1–43, 2016, doi: 10.1007/s11227-016-1805-9.

[10] D. Sgandurra and E. Lupu, "Evolution of Attacks, Threat Models, and Solutions for Virtualized Systems," *ACM Computing Surveys (CSUR)*, vol. 48, no. 3, pp. 1–38, 2016, doi: 10.1145/2856126.

[11] A. R. Riddle and S. M. Chung, "A survey on the security of hypervisors in cloud computing," in *Proceedings - 2015 IEEE 35th International Conference on Distributed Computing Systems Workshops, ICDCSW 2015*, 2015, pp. 100–104, doi: 10.1109/ICDCSW.2015.28.

[12] M. Ali, S. U. Khan, and A. V. Vasilakos, "Security in cloud computing: Opportunities and challenges," *Information Sciences*, vol. 305, pp. 357–383, 2015, doi: 10.1016/j.ins.2015.01.025.

[13] S. Qinggang, W. Fu, and H. Qiangwei, "Study of Cloud Computing Security Service Model," *Engineering and Technology (S-CET), 2012 Spring Congress on*, pp. 1–4, 2012, doi: 10.1109/SCET.2012.6341967.

[14] M. Almorsy, J. Grundy, and I. Müller, "An analysis of the cloud computing security problem," *17th Asia-Pacific Software Engineering Conference (APSEC 2010) Cloud Workshop,Sydney, Australia*, no. December, p. 7, 2010.

[15] S. Iqbal *et al.*, "On cloud security attacks: A taxonomy and intrusion detection and prevention as a service," *Journal of Network and Computer Applications*, vol. 74. pp. 98–120, 2016, doi: 10.1016/j.jnca.2016.08.016.

[16] W. Dawoud, I. Takouna, and C. Meinel, "Infrastructure as a Service Security : Challenges and Solutions," *Security*, pp. 1–8, 2010.

[17] T. Takahashi, G. Blanc, Y. Kadobayashi, D. Fall, H. Hazeyama, and S. Matsuo, "Enabling secure multitenancy in cloud computing: Challenges and approaches," in *2012 2nd Baltic Congress on Future Internet Communications, BCFIC 2012*, 2012, pp. 72–79, doi: 10.1109/BCFIC.2012.6217983.

[18] G. Zhao, C. Rong, M. G. Jaatun, and F. E. Sandnes, "Deployment models: Towards eliminating security concerns from cloud computing," *Proceedings of the 2010 International Conference on High Performance Computing and Simulation, HPCS 2010*, pp. 189–195, 2010, doi: 10.1109/HPCS.2010.5547137.

[19] C. S. Aishwarya, "Insight into Cloud Security Issues," *UACEE International Journal of Computer Science and its Applications*, pp. 30–33, 2011.

[20] M. Sudha, D. Rao, and M. Monica, "A Comprehensive approach to ensure secure data communication in cloud environment," *International Journal of Computer*, 2010.

[21] M. Gobi and R. Sridevi, "An Approach for Secure Data Storage in Cloud Environment," *International Journal of Computer and*, 2013.

[22] C. Wang, Q. Wang, K. Ren, and W. Lou, "Ensuring data storage security in Cloud Computing," *2009 17th International Workshop on Quality of Service*, pp. 1–9, 2009, doi: 10.1109/IWQoS.2009.5201385.

[23] D. Purushothaman and S. Abburu, "An approach for data storage security in cloud computing," *International Journal of Computer Science*, vol. 9, no. 2, pp. 100–106, 2012.

[24] K. S. Reddy and M. BALARAJU, "An Integrated Approach of Data storage and Security in Cloud Computing," *ijaiem.org*.

[25] A. Sangroya, S. Kumar, J. Dhok, and V. Varma, "Towards analyzing data security risks in cloud computing environments," *Communications in Computer and Information Science*, vol. 54, pp. 255–265, 2010, doi: 10.1007/978-3-642-12035-0_25.

[26] G. R. Reddy, "A Novel Approach for Multi-Cloud Storage security in Cloud Computing," *Citeseer*, vol. 3, no. 5, pp. 5043–5045, 2012.

[27] A. Juels, Ari and Oprea, "New Approaches to Security and Availability for Cloud Data," *Commun. ACM*, vol. 56, no. 2, pp. 64–73, 2013.

[28] D. Smith, Q. Guan, and S. Fu, "An Anomaly Detection Framework for Autonomic Management of Compute Cloud Systems," in *2010 IEEE 34th Annual Computer Software and Applications Conference Workshops*, 2010, pp. 376–381, doi: 10.1109/COMPSACW.2010.72.

[29] K. Hamlen, M. Kantarcioglu, and L. Khan, "Security issues for cloud computing," *Optimizing*, 2012.

[30] M. Almorsy, J. Grundy, and A. S. Ibrahim, "Collaboration-based cloud computing security management framework," in *Proceedings - 2011 IEEE 4th International Conference on Cloud Computing, CLOUD 2011*, 2011, pp. 364–371, doi: 10.1109/CLOUD.2011.9.

[31] U. Tupakula, V. Varadharajan, and N. Akku, "Intrusion Detection Techniques for Infrastructure as a Service Cloud," in *2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing*, 2011, pp. 744–751, doi: 10.1109/DASC.2011.128.

[32] L. M. Vaquero, L. Rodero-Merino, and D. Morán, "Locking the sky: A survey on IaaS cloud security," *Computing (Vienna/New York)*, vol. 91, no. 1, pp. 93–118, 2011, doi: 10.1007/s00607-010-0140-x.

[33] M. Yildiz, J. Abawajy, T. Ercan, and A. Bernoth, "A layered security approach for cloud computing infrastructure," in *I-SPAN 2009 - The 10th International Symposium on Pervasive Systems, Algorithms, and Networks*, 2009, pp. 763–767, doi: 10.1109/I-SPAN.2009.157.

[34] P. Arora, W. Rubal Chaudhry, and S. P. Ahuja, "Cloud computing security issues in infrastructure as a service," *ijartet.com*, vol. 2, no. 1, pp. 1–7, 2012.

[35] T. Alharkan and P. Martin, "IDSaaS: Intrusion detection system as a service in public clouds," in *Proceedings - 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, CCGrid 2012*, 2012, pp. 686–687, doi: 10.1109/CCGrid.2012.81.

[36] G. Nascimento and M. Correia, "Anomaly-based intrusion detection in software as a service," *IEEE/IFIP 41st International Conference on Dependable Systems and Networks Workshop (DSN-W), 2011*, 2011, [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5958858.

[37] S. Subashini and V. Kavitha, "A survey on security issues in service delivery models of cloud computing," *Journal of Network and Computer Applications*, vol. 34, no. 1. pp. 1–11, 2011, doi: 10.1016/j.jnca.2010.07.006.
[38] M. Jensen, J. Schwenk, N. Gruschka, and L. L. Iacono, "On Technical Security Issues in Cloud Computing," in *Cloud Computing, 2009. CLOUD '09. IEEE International Conference on*, 2009, pp. 109–116, doi: 10.1109/CLOUD.2009.60.
[39] W. Consulting, "Virtual Machine Security Guidelines Version 1.0," *tp-seginf.googlecode.com*, no. September, 2007.
[40] J. Reuben, "A survey on virtual machine security," *Helsinki University of Technology*, p. 5, 2007.
[41] R. Anand, S. Sarswathi, and R. Regan, "Security issues in virtualization environment," in *2012 International Conference on Radar, Communication and Computing, ICRCC 2012*, 2012, pp. 254–256, doi: 10.1109/ICRCC.2012.6450589.
[42] T. T. Brooks, C. Caicedo, and J. S. Park, "Security Vulnerability Analysis in Virtualized Computing Environments," *International Journal of Intelligent Computing Research (IJICR)*, vol. 3, no. Issues1/2, pp. 277–291, 2012.
[43] T. Garfinkel and M. Rosenblum, "A Virtual Machine Introspection Based Architecture for Intrusion Detection," *Proc. Network and Distributed Systems Security …*, vol. 1, pp. 253–285, 2003, doi: 10.1109/SP.2011.11.
[44] Z. Nan, "Virtualization safety problem analysis," in *2011 IEEE 3rd International Conference on Communication Software and Networks, ICCSN 2011*, 2011, pp. 195–197, doi: 10.1109/ICCSN.2011.6013693.
[45] J. Li *et al.*, "CyberGuarder: A virtualization security assurance architecture for green cloud computing," *Future Generation Computer Systems*, vol. 28, no. 2, pp. 379–390, 2012, doi: 10.1016/j.future.2011.04.012.
[46] S. S. Yeo and J. H. Park, "Security considerations in cloud computing virtualization environment," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2013, vol. 7861 LNCS, pp. 208–215, doi: 10.1007/978-3-642-38027-3_22.
[47] M. Kazim, R. Masood, M. A. Shibli, and A. G. Abbasi, "Security aspects of virtualization in cloud computing," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2013, vol. 8104 LNCS, pp. 229–240, doi: 10.1007/978-3-642-40925-7_22.
[48] A. Cleeff, W. Pieters, and R. Wieringa, "Security implications of virtualization: A literature study," in *Proceedings - 12th IEEE International Conference on Computational Science and Engineering, CSE 2009*, 2009, vol. 3, pp. 353–358, doi: 10.1109/CSE.2009.267.
[49] H. Orman, "Both Sides Now: Thinking about Cloud Security," *IEEE Internet Computing*, vol. 20, no. 1, pp. 83–87, Jan. 2016, doi: 10.1109/MIC.2016.17.
[50] K. Kortchinsky, "Cloudburst: Hacking 3D (and Breaking Out of VMware) for Black Hat USA 2009."
[51] N. Elhage, "Virtunoid: A KVM Guest → Host privilege escalation exploit," 2011.
[52] J. Wang, A. Stavrou, and A. Ghosh, "Hyper Check: A Hardware-Assisted Integrity Monitor," in *RECENT ADVANCES IN INTRUSION DETECTION*, 2010, vol. 6307, pp. 158–177, doi: 10.1007/978-3-642-15512-3_9.
[53] Z. Wang and X. Jiang, "HyperSafe: A lightweight approach to provide lifetime hypervisor control-flow integrity," in *Proceedings - IEEE Symposium on Security and Privacy*, 2010, pp. 380–395, doi: 10.1109/SP.2010.30.
[54] J. Szefer, E. Keller, R. B. Lee, and J. Rexford, "Eliminating the Hypervisor Attack Surface for a More Secure Cloud Categories and Subject Descriptors," *Proceedings of the 18th ACM conference on*

Computer and Communications Security (CCS'11), pp. 401–412, 2011, doi: 10.1145/2046707.2046754.

[55] N. L. Petroni and M. Hicks, "Automated detection of persistent kernel control-flow attacks," *ACM conference on Computer and communications security (CCS) (2007)*, pp. 103–115, 2007, doi: 10.1145/1315245.1315260.

[56] R. Riley, X. Jiang, and D. Xu, "Guest-transparent prevention of kernel rootkits with VMM-based memory shadowing," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2008, vol. 5230 LNCS, pp. 1–20, doi: 10.1007/978-3-540-87403-4_1.

[57] A. Seshadri, M. Luk, N. Qu, and A. Perrig, "SecVisor: A Tiny Hypervisor to Provide Lifetime Kernel Code Integrity for Commodity OSes," in *Proceedings of the ACM SIGOPS Symposium on Operating Systems Principles*, 2006, vol. 41, pp. 335–350, doi: 10.1145/1323293.1294294.

[58] L. Litty, H. A. Lagar-Cavilla, and D. Lie, "Hypervisor support for identifying covertly executing binaries," *USENIX Security*, pp. 243–258, 2008.

[59] A. Srivastava, K. Singh, and J. Giffin, "Secure Observation of Kernel Behavior."

[60] S. T. Jones, A. C. Arpaci-Dusseau, and R. H. Arpaci-Dusseau, "VMM-based hidden process detection and identification using Lycosid," *Proceedings of the fourth ACM SIGPLAN/SIGOPS international conference on Virtual execution environments - VEE '08*, p. 91, 2008, doi: 10.1145/1346256.1346269.

[61] M. Ficco *et al.*, "RootkitDet: Practical end-to-end defense against kernel rootkits in a cloud environment," *Proceedings - International Conference on Cloud Computing Research and Innovation 2016, ICCCRI 2016*, vol. 67, no. 11, pp. 295–300, 2014, doi: 10.1016/j.jnca.2016.01.001.

[62] R. Hund and F. C. Freiling, "Return-Oriented Rootkits : Bypassing Kernel Code Integrity Protection Mechanisms," *Usenix Sec*, pp. 383–398, 2009.

[63] Z. Wang, X. Jiang, W. Cui, and P. Ning, "Countering kernel rootkits with lightweight hook protection," *Proceedings of the 16th ACM conference on Computer and communications security - CCS '09*, p. 545, 2009, doi: 10.1145/1653662.1653728.

[64] S. Bharadwaja, W. Sun, M. Niamat, and F. Shen, "Collabra: A xen hypervisor based collaborative intrusion detection system," in *Proceedings - 2011 8th International Conference on Information Technology: New Generations, ITNG 2011*, 2010, pp. 695–700, doi: 10.1109/ITNG.2011.123.

[65] Y. Du, R. Zhang, and M. Li, "Research on a security mechanism for cloud computing based on virtualization," *Telecommunication Systems*, vol. 53, no. 1, pp. 19–24, 2013, doi: 10.1007/s11235-013-9672-7.

[66] S. T. King, P. M. Chen, Y. M. Wang, C. Verbowski, H. J. Wang, and J. R. Lorch, "SubVirt: Implementing malware with virtual machines," in *Proceedings - IEEE Symposium on Security and Privacy*, 2006, vol. 2006, pp. 314–327, doi: 10.1109/SP.2006.38.

[67] J. Rutkowska, "Subverting Vista Kernel For Fun And Profit," 2006.

[68] E. Barbosa, "Detection of Hardware Virtualization Rootkits." 2007.

[69] F. Hao, T. V Lakshman, S. Mukherjee, and H. Song, "Secure Cloud Computing with a Virtualized Network Infrastructure," *Integration The Vlsi Journal*, pp. 16–16, 2010, doi: 10.1234/12345678.

[70] F. Zhao, W. Yang, H. Jin, and S. Wu, "VNIDS: A virtual machine-based network intrusion detection system," in *2nd International Conference on Anti-counterfeiting, Security and Identification, ASID 2008*, 2008, pp. 254–259, doi: 10.1109/IWASID.2008.4688384.

[71] K. Kourai and S. Chiba, "HyperSpector: virtual distributed monitoring environments for secure intrusion detection," *in Proceedings of 1st ACM/USENIX International conference on Virtual execution environments*, pp. 197–207, 2005, doi: 10.1145/1064979.1065006.

[72] U. Tupakula and V. Varadharajan, “On the design of Virtual machine Intrusion detection system,” *12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011) and Workshops*, pp. 682–685, 2011, doi: 10.1109/INM.2011.5990655.

[73] Y. Zhang, Y. Gu, H. Wang, and D. Wang, “Virtual-machine-based intrusion detection on file-aware block level storage,” in *Proceedings - Symposium on Computer Architecture and High Performance Computing*, 2006, pp. 185–192, doi: 10.1109/SBAC-PAD.2006.32.

[74] H. Jin *et al.*, “A VMM-based intrusion prevention system in cloud computing environment,” *The Journal of Supercomputing*, vol. 66, no. 3, pp. 1133–1151, 2013, doi: 10.1007/s11227-011-0608-2.

[75] A. Joshi, S. T. King, G. W. Dunlap, and P. M. Chen, “Detecting past and present intrusions through vulnerability-specific predicates,” *ACM SIGOPS Operating Systems Review*, vol. 39, no. 5, p. 91, 2005, doi: 10.1145/1095809.1095820.

[76] X. Jiang, X. Wang, and D. Xu, “Stealthy malware detection and monitoring through VMM-based ‘out-of-the-box’ semantic view reconstruction,” *ACM Transactions on Information and System Security*, vol. 13, no. 2, pp. 1–28, 2010, doi: 10.1145/1698750.1698752.

[77] G. Dunlap, S. King, and S. Cinar, “ReVirt: Enabling intrusion analysis through virtual-machine logging and replay,” *ACM SIGOPS Operating Systems Review*, vol. 36, no. SI, pp. 211–224, 2002, doi: 10.1145/844128.844148.

[78] M. Noura, S. Mohammadalian, L. Fathi, and M. Torabi, “Secure Virtualization for Cloud Environment Using Guest OS and VMM-based Technology,” *International Journal of Computer Networks & Communications Secu*, vol. 1, no. 2, p. 61, 2013.

[79] B. D. Payne, M. Carbone, M. Sharif, and W. Lee, “Lares: An architecture for secure active monitoring using virtualization,” in *Proceedings - IEEE Symposium on Security and Privacy*, 2008, pp. 233–247, doi: 10.1109/SP.2008.24.

[80] M. Aiash, G. Mapp, and O. Gemikonakli, “Secure live virtual machines migration: Issues and solutions,” in *Proceedings - 2014 IEEE 28th International Conference on Advanced Information Networking and Applications Workshops, IEEE WAINA 2014*, 2014, pp. 160–165, doi: 10.1109/WAINA.2014.35.

[81] M. Kadam, Rajesaheb R and Bangare, “A Survey on Security Issues and Solutions in Live Virtual Machine Migration,” *International Journal of Advance Foundation and Research in Computer*, vol. 1, no. 12, 2014.

[82] W. Wang, Y. Zhang, B. Lin, X. Wu, and K. Miao, “Secured and reliable VM migration in personal cloud,” in *ICCET 2010 - 2010 International Conference on Computer Engineering and Technology, Proceedings*, 2010, vol. 1, doi: 10.1109/ICCET.2010.5485376.

[83] C. For, C. Computing, K. Sammy, R. Shengbing, and C. Wilson, “Energy Efficient Security Preserving VM Live Migration In Data,” *Journal of Computer Science*, vol. 9, no. 2, pp. 33–39, 2012.

[84] J. Oberheide, E. Cooke, and F. Jahanian, “Empirical exploitation of live virtual machine migration,” *Proc. of BlackHat Security Conference*, no. Vmm, 2008, [Online]. Available: http://63.236.103.240/presentations/bh-dc-08/Oberheide/Whitepaper/bh-dc-08-oberheide-WP.pdf.

[85] G. J. Jeincy, R. S. Shaji, and J. P. Jayan, “A secure virtual machine migration using memory space prediction for cloud computing,” 2016, doi: 10.1109/ICCPCT.2016.7530379.

[86] P. J. Reeba, “Using Processor Workload Prediction Method for Cloud Environment,” pp. 0–5, 2016.

[87] B. Guan, Y. Wu, L. Ding, and Y. Wang, “CIVSched: Communication-aware Inter-VM Scheduling in Virtual Machine Monitor Based on the Process,” in *Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on*, 2013, pp. 597–604, doi: 10.1109/CCGrid.2013.105.

[88] W. Huang, M. J. Koop, Q. Gao, and D. K. Panda, "Virtual machine aware communication libraries for high performance computing," in *Proceedings of the 2007 ACM/IEEE conference on Supercomputing*, 2007, p. 9, doi: 10.1145/1362622.1362635.

[89] K. Kim, C. Kim, S.-I. Jung, H.-S. Shin, and J.-S. Kim, "Inter-domain socket communications supporting high performance and full binary compatibility on Xen," in *Proceedings of the fourth ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*, 2008, pp. 11–20, doi: 10.1145/1346256.1346259.

[90] J. Wang, K.-L. L. Wright, and K. Gopalan, "XenLoop: a transparent high performance inter-vm network loopback," in *Proceedings of the 17th international symposium on High performance distributed computing*, 2008, vol. 12, no. 2 SPEC. ISS., pp. 109–118, doi: 10.1007/s10586-009-0079-x.

[91] L. Youseff, D. Zagorodnov, and R. Wolski, "Inter-OS communication on highly parallel multi-core architectures," Technical Report, University of California, Santa Barbara, 2008.

[92] X. Zhang, S. McIntosh, P. Rohatgi, and J. L. Griffin, "XenSocket: A high-throughput interdomain transport for virtual machines," in *Middleware 2007*, vol. 4834, Springer, 2007, pp. 184–203.

[93] C. Gebhardt and A. Tomlinson, "Challenges for inter virtual machine communication," Citeseer, 2010.

[94] J. Wang, "Survey of State-of-the-art in Inter-VM Communication Mechanisms," *Research Proficiency Report*, 2009.

[95] K. Benzidane, S. Khoudali, F. Leila, and A. Sekkaki, "Toward inter-VM visibility in a Cloud environment using packet inspection," in *Telecommunications (ICT), 2013 20th International Conference on*, 2013, pp. 1–5, doi: 10.1109/ICTEL.2013.6632122.

[96] K. Benzidane, S. S. S. Khoudali, and A. Sekkaki, "Secured architecture for inter-VM traffic in a Cloud environment," in *2nd IEEE Latin American Conference on Cloud Computing and Communications, LatinCloud 2013*, 2013, pp. 23–28, doi: 10.1109/LatinCloud.2013.6842218.

[97] F. Diakhaté *et al.*, "Efficient shared memory message passing for inter-VM communications," in *Euro-Par 2008 Workshops-Parallel Processing*, 2009, vol. 5415 LNCS, pp. 53–62, doi: 10.1007/978-3-642-00955-6_7.

[98] S. Khoudali, K. Benzidane, and A. Sekkaki, "Inter-vm packet inspection in cloud computing," in *Communications, Computers and Applications (MIC-CCA), 2012 Mosharaka International Conference on*, 2012, pp. 84–89.

[99] D. Li, H. Jin, Y. Shao, X. Liao, Z. Han, and K. Chen, "A high-performance inter-domain data transferring system for virtual machines," *Journal of Software*, vol. 5, no. 2, pp. 206–213, 2010.

[100] S.-F. F. Yang, W.-Y. Y. Chen, and Y.-T. T. Wang, "ICAS: An inter-VM IDS log cloud analysis system," in *Cloud Computing and Intelligence Systems (CCIS), 2011 IEEE International Conference on*, 2011, pp. 285–289, doi: 10.1109/CCIS.2011.6045076.

[101] Y. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Cross-VM side channels and their use to extract private keys," in *Proceedings of the 2012 ACM conference on Computer and communications security - CCS '12*, 2012, p. 305, doi: 10.1145/2382196.2382230.

[102] B. Sevak, "Security against side channel attack in cloud computing," *International Journal of Engineering and Advanced Technology (IJEAT)*, vol. 2, no. 2, p. 183, 2013.

[103] Y. Xu, M. Bailey, F. Jahanian, K. Joshi, M. Hiltunen, and R. Schlichting, "An exploration of L2 cache covert channels in virtualized environments," in *Proceedings of the 3rd ACM workshop on Cloud computing security workshop - CCSW '11*, 2011, pp. 29–40, doi: 10.1145/2046660.2046670.

[104] C. Percival, "Cache missing for fun and profit." BSDCan, 2005.

[105] D. J. Bernstein, "Cache-timing attacks on AES." 2005.

[106] D. Page, "Partitioned Cache Architecture as a Side-Channel Defence Mechanism," *IACR Cryptology ePrint Archive*, vol. 2005, p. 280, 2005.

[107] S. Yu, X. Gui, and J. Lin, “An approach with two-stage mode to detect cache-based side channel attacks,” in *Information Networking (ICOIN), 2013 International Conference on*, 2013, pp. 186–191.
[108] B. K. Mughal, A. Syed, and A. Khan, “Information leakage in the cloud,” in *Open Source Systems and Technologies (ICOSST), 2014 International Conference on*, 2014, pp. 56–61.
[109] D. A. Osvik, A. Shamir, and E. Tromer, “Cache attacks and countermeasures: the case of AES,” in *Topics in Cryptology–CT-RSA 2006*, Springer, 2006, pp. 1–20.
[110] Y. Kulah, B. Dincer, C. Yilmaz, and E. Savas, “SpyDetector: An approach for detecting side-channel attacks at runtime,” *International Journal of Information Security*, vol. 18, no. 4, pp. 393–422, 2019, doi: 10.1007/s10207-018-0411-7.
[111] J. Seifert, Ç. Koç, and O. Aciiçmez, “Predicting Secret Keys Via Branch Prediction,” *Ct-Rsa*, pp. 225–242, 2007, doi: 10.1007/11967668_15.
[112] O. Aciiçmez, Ç. K. Koç, and J.-P. Seifert, “On the power of simple branch prediction analysis,” in *Proceedings of the 2nd ACM symposium on Information, computer and communications security*, 2007, pp. 312–320, doi: 10.1145/1229285.1266999.
[113] O. Acıiçmez, S. Gueron, and J. Seifert, “New branch prediction vulnerabilities in OpenSSL and necessary software countermeasures,” *Cryptography and Coding*. pp. 1–16, 2007, doi: 10.1007/978-3-540-77272-9_12.
[114] M. Joye and S.-M. Yen, “The Montgomery Powering Ladder,” in *International Workshop on Cryptographic Hardware and Embedded Systems (CHES 2002)*, 2002, pp. 291–302, doi: 10.1007/3-540-36400-5_22.
[115] S. Bhattacharya, D. M.-I. C. ePrint Archive, and U. 2014, “Fault Attack revealing Secret Keys of Exponentiation Algorithms from Branch Prediction Misses.,” *IACE cryptography ePrint*, p. 790, 2014.
[116] S. Bhattacharya and D. Mukhopadhyay, “Who Watches the Watchmen?: Utilizing Performance Monitors for Compromising Keys of RSA on Intel Platforms,” Springer, Berlin, Heidelberg, 2015, pp. 248–266.
[117] S. Bhattacharya and D. Mukhopadhyay, “Formal fault analysis of branch predictors: attacking countermeasures of asymmetric key ciphers,” *Journal of Cryptographic Engineering*, vol. 7, no. 4, pp. 299–310, 2017, doi: 10.1007/s13389-017-0165-6.
[118] G. Agosta, L. Breveglieri, G. Pelosi, and I. Koren, “Countermeasures against branch target buffer attacks,” in *Fault Diagnosis and Tolerance in Cryptography, 2007. FDTC 2007. Workshop on*, 2007, pp. 75–79.
[119] Y. Tan, J. Wei, and W. Guo, “The micro-architectural support countermeasures against the branch prediction analysis attack,” *Proceedings - 2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2014*, pp. 276–283, 2015, doi: 10.1109/TrustCom.2014.38.
[120] J. Sebot and S. Gueron, “Mitigating branch prediction and other timing based side channel attacks,” *US Patent 8,869,294*, 2014.
[121] S. Bhattacharya, S. Bhasin, and D. Mukhopadhyay, “Online Detection and Reactive Countermeasure for Leakage from BPU Using TVLA,” in *2018 31st International Conference on VLSI Design and 2018 17th International Conference on Embedded Systems (VLSID)*, Jan. 2018, pp. 155–160, doi: 10.1109/VLSID.2018.54.
[122] 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*, 2009, pp. 199–212, doi: 10.1145/1653662.1653687.
[123] A. C. de Melo, “The New Linux ’perf’ Tools,” p. 11.
[124] N. Binkert *et al.*, “The gem5 simulator,” *ACM SIGARCH Computer Architecture News*, vol. 39, no. 2, p. 1, 2011, doi: 10.1145/2024716.2024718.

[125] A. R. C. ; S. M. ; A. Y. Javaid, “Man-in-the-Middle (MITM) Attack Based Hijacking of HTTP Traffic Using Open Source Tools,” *IEEE International Conference on Electro/Information Technology*, 2018.

[126] B. I. M.M.S. KUMAR, “A STUDY ON WEB HIJACKING TECHNIQUES AND BROWSER ATTACKS,” *INTERNATIONAL JOURNAL OF APPLIED ENGINEERING RESEARCH*\, vol. 13, 2018.

[127] M. R. Guthaus, J. S. Ringenberg, D. Ernst, T. M. Austin, T. Mudge, and R. B. Brown, “MiBench : A free , commercially representative embedded benchmark suite The University of Michigan Electrical Engineering and Computer Science,” *Proceedings of the Workload Characterization*, pp. 3–14, 2001.