Prevention of VM Side-Channel Timing Attack in Cloud Computing using Randomized Timing Approach in AES - 128

[1*]Animesh Kumar, [2]Sandip Dutta, [3]Prashant Pranav

[1]Research Scholar, Department of Computer Science and Engineering, Birla Institute of Technology, Mesra, Ranchi, 835215

[2]Professor, Department of Computer Science and Engineering, Birla Institute of Technology, Mesra, Ranchi, 835215

[3]Assistant Professor, Department of Computer Science and Engineering, Birla Institute of Technology, Mesra, Ranchi, 835215

*is the corresponding author

Abstract: Cloud computing is one of the most widely used technologies in the world. More and more services are offered to clients by cloud service providers (CSP). Today Cloud computing services are facing constant threats and vulnerabilities unique to such systems. One of the most important cloud-specific attacks is the VM side-channel timing attack. In this paper, we propose a new method to prevent VM side-channel timing attacks in AES-128.

Keywords: Cloud Computing, Virtualization, Cloud Security, VM side-channel attack, Cross VM timing side-channel attack, VM cache side-channel attack.

1. INTRODUCTION

Cloud computing provides an efficient business model for the deployment of IT infrastructure and software services on pay per basis. It enables cloud service providers to offer services, resources over the internet. It is effective, reliable, and also cost-effective as compared to maintaining its physical data centers. It uses the concept of distributed computing. Virtualization plays a key role in providing cloud computing services. The benefits of cloud computing are nice pricing, no prior commitments, on-demand access, efficient resource allocation, etc. Highly energy-efficient, use of third-party services are also permitted in cloud service. The backbone of cloud computing is distributed systems, virtualization, web 2.0, service-oriented computing, and utility-oriented computing. Some of the popular computing platforms and technologies are *Amazon Web Services*, *Google AppEngine*, *Microsoft Azure*, *Hadoop*, *Manjrasoft Aneka*, *Salesforce*, etc.

1.1.Security Issues in Cloud Computing

Cloud Computing Technology has many underlying security challenges. Some of the most prominent security threats to a cloud computing environment are discussed underneath.

Insecure Application Programming Interfaces (APIs): CSP offers services to their clients and gives them access to services through application programming Interfaces. Any unwanted

change in APIs creates a lot of security-related issues.

Denial of Service(DoS)[1]*:* Interruption of services to genuine customers comes under the category of Denial of service attack. Distributed DoS attacks create a much bigger problem for cloud service providers.

Abuse of Cloud Resources: In IaaS, CSPs provide large hardware resources to customers. Customers may exploit this feature to perform malicious activity on powerful hardware resources.

Shared Resources: CSPs use the concept of virtualization which shares the hardware resources to many customers such as CPU, High-level Cache, Storage devices, Network interface card, high level, which creates Co-residency attack issues.

VM Escape [2]*:* It is an exploratory method that enables the virtual machine to attack its Host. The attacker runs the malicious code on the VM, tries to take control of the operating system, and indirectly breaks the Hypervisor.

Hyper-Jacking [3]*:* This is a method where the attacker compromises the hypervisor. Their target area is the operating system of Virtual machines. when gaining access, they create a fake hypervisor over the top of the main hypervisor. Attacker direct access to the original hypervisor takes place in a Hyper-jacking attack.

VM Side-Channel Attack: The attacker tries to gain important information from the target VM by exploiting its hardware shared by its co-resident VM. Electromagnetic signals, timing, Electricity supply, etc are analyzed by the attacker to target the victim system.

Among many prevailing possible attacks in a cloud environment, we have concentrated on VM side-channel attacks with a focus on timing side-channel attacks.

2. VM Side-Channel Attacks

VM Side Channel Attack is one of the most popular attacks in cloud computing. In cloud computing, we share the physical resources over multiple clients. CPU Core and high-level Cache memory are compromised during this type of attack. Attackers try to get information or sensitive data from the other co-resident virtual machine. They try to decode cryptographic keys of the target virtual machine. They also try to analyze the whole network traffic data. Some of the variants of the VM side-channel attack are discussed below.

Cache Based Side-Channel Attack

Prime + Probe Method [4]: It is a type of side-channel attack used to gain Data access from the co-resident VM sharing the same cache memory. This type of attack was first detected in 2005.

Flush + Reload Technique [5]: Memory lines of shared pages are targeted in this type of

side-channel attack. Attacker hits the last level Cache memory of the VM. It is one of the most sophisticated types of attacks in the side-channel category.

Types of Cache Side-Channel Attacks

Time-Driven Method [6]: The attacker tries to break the cryptosystem using the time required to execute the processing of cryptographic algorithms. attacker compares the execution times of all processors to decode the patterns, which are ultimately used to break the encryption keys.

Access-Driven Method [7]: The attacker investigates the cache behavior of the target VM minutely. Analysis of the cache behavior of the victim's VM takes place in the access-driven method.

Trace-Driven Method [8]: Attackers control the cache lines accessed in the cache memory set of the victim's VM and try to obtain all cache memory activity done during the encryption technique. It is estimated that through cache memory set along with S-box misalignment, some 200 samples are required to decode to extract a complete 128-bit AES within the next few seconds.

2.1.VM side-channel Timing Attack

A timing attack enables the attacker to extract secret information by analyzing the time taken to respond to different queries. Execution time was properly inspected by the attacker. As the state of cache memory directly depends on its memory access timing. Different types of processes take different execution times; that time difference could be used for breaking cryptographic keys like AES, RSA, or ElGamal keys. Access to cache lines by decryption directly depends on key and ciphertext. Such reliance leads to side-channel timing attacks. S-Box lookup of AES algorithm relies on AES keys and plain text. The attacker performs encryption on a large number of known plain text and execution time differences of the S-Box lookup process and tries to decode the key after performing offline analysis.

2.2.VM Side-Channel Attack Countermeasure

Software-Based solutions

Huge size page Disability: Attacks can be avoided by not allowing the guest Virtual Machine to use a huge memory size page. Memory space resources should not be shared. Other methods of software-based solution are as follows: Private LLC Cache slices, Controlling Clflush Instruction, Preventing Page Sharing, Prefetching Cache Memory, Flushing Cache Memory, Hardware masking of Addresses, Address translation using shadow page tables, Dynamic Software Diversity, Cache Partition using Cache Memory.

Hardware-Based Solutions

Each CPU is assigned separate cache memory. Some of

The other methods are as follows: Concealing Correlation of cache-data, Customized Hardware, Cache-Disabling, latency and Timing Information, Cache State Normalization, Cache Warming, Cache Partitioning, Cache Flushing.

3. BACKGROUND AND MOTIVATION

AUTHORS	METHODOLOGY	ADVANTAGE	DISADVANTAGE	RESULT
Liming Wang, Xiaojie Tao et al. [9] 2021	A novel CSCAs mitigation system is proposed in the cloud.	SCAMS can exactly identify Inappropriate cache activity of co-resident VMs.	VM with minimum and maximum Workload is ignored in this paper.	Square and multiply algorithm is implemented. Protected VM Performance is shown in graph form. Cache Behavior of sharing-based CSCA and Conflict based CSCA is properly shown in graph form.
Yehonatan Lusky et al.[10] 2021	Sandboxes are detected using Hardware-based side-channel leakage.	Comparison of MAVMM, VMware Workstation, Virtual Box, Microsoft Hyper-V attack was shown.	Detailed analysis of Mitigation Techniques and Discussion is missing.	A full attack Algorithm is used. All hypervisors were tested according to his taken attack.
Hosein Mohammadi Makrani et al. [11] 2021	Cloak & Co-locate New attack was proposed by the author. It targets the Resource Provisional System to locate victim VMs.	It helps to provide better security and performance to develop RPSs Shown on Medium Size Cluster.	Higher and Lower Size Cluster is not Shown.	Increasing the number of applications on the host, reducing the probability of attack's success rate from 60% to 16% shown in this paper.

Prashant Pranav et al [12] 2021	In this paper, author compared the intermediate steps of RSA and AES-128 statistically.	The method works well in determining which step among many sub steps of RSA and AES-128 is most time consuming.	The proposed method is machine dependent.	Result shows that the power operation of RSA is more time consuming. Mix column sub steps of AES-128 is the most dominant sub step.
Zeyu Mi et al. [13] 2020	Author Proposed CREASE method to protect Cross -VM runtime Monitoring.	In the related work section, four different solutions are provided to solve side-channel attacks.	CREASE lacks large overheads for both principal and peer.	Error Rate Comparison of Frequency Scaling 1 to 30 ms Random quantum-based upon 8 independent experiments.
Jeroen Van Cleemput et al. [14] 2020	Offline and online approach for eliminating timing side-channel attacks.	7 different types of mitigation techniques were discussed in this paper.	Paper lacks in the graphical analysis part on calculating average HMAC execution time on the core 2.	Method Selection based on the Call Edge Count and Execution Counts algorithm is used in this paper.
Shun-Wen Hsiao et al. [15] 2020	MMU Redirection Mechanism is developed to counter VM cache side-channel attacks.	It provides better transparency, high performance, and comparable semantics.	Paper lacks in explaining the threat model.	EPT transversal mapping GPA to Given HPA properly shown in host kernel memory backed by proper figures.
Gregory Levitin et al. [16] 2020	The Threshold Voting-based N-version programming (NVP) is proposed to improve the reliability of services.	A probabilistic model is introduced to check the probability of data theft.	Service Models were explained in a very short way.	The optimal number of SCVs and corresponding AVMs is properly explained in graphical forms.

Ahmed Osama Fathy Atya et al. [17] 2019	Identification of a new set of side-channel attacks in co-residency. A set of guidelines are given in this paper, under what condition the victim VM is Targeted.	Prime +trigger +Probe and LLC, Prime + Probe, and Deduplication side-channel attacks were simulated with Average time and shared Resources.	Comparison of bandwidth cost with Nomad is properly not shown on the graph.	Authors' Guidelines can limit attackers' efficiency to about 1percent with very modest bandwidth and downtime costs.
Hisham Alhulayyil et al. [18] 2018	The author detailed analyzed the problem of Malicious VM Co-residency in Cloud Environment.	Detection Risk The function is properly shown in Graph. Attacker and Defender best Response properly explained by a graph.	The Game Model is not properly explained in it.	Attacks on multiple VM are missing in this Paper.
Berk Gulmezoglu et al. [19] 2016	The threat was detected in using insecure software implementations of AES on virtualized systems without AES-NI hardware Support.	T-table Searching algorithms are used in this paper.	Software-based countermeasures for the side-channel attack were not explained in a detailed way.	Flush Reload, Prime + probe Reload, EC2 Public Cloud setup, Amazon EC2 Results are properly discussed in this Paper.

Sandeep Saxena, et al. [20] 2017	It identifies the earlier shortcomings of defense techniques used in the cloud environment. The author also investigated CPU Cache Vulnerabilities.	Square-Multiply Algorithm, Proposed Prevention Algorithm, Attack simulation, Signature operation using Victim VM, Signature operation using Victim VM performing Signature is properly implemented.	A simple permutation function was applied to this Paper.	Confidence intervals of mean CPU cycles were properly shown using a Table format. A sequence of Square-Reduce and Multiply operations under is attack shown in the graph.
Fangfui Liu et al. [21] 2016	Author proposed CATalyst for cloud service providers and cloud customers to protect against LLC-based side-channel attacks by using Intel processors.	Square and Multiply exponentiation algorithms are properly used in this paper.	EPT mapping on VM launch and Mapping after assigning secure pages not shown the detailed way.	Time to encryption of a 5 MB file using AES and GnuPG ElGamal decryption times properly shown in Table format.
Abid Shahzad, et al. [22] 2015	The author presented the different techniques of Cross-VM cache-based side-channel attack.	DoS, Hypervisor attack, Hyper-jacking, VM side-channel attacks are properly explained in this paper.	Proper Experimental The result is not shown in this Paper.	Common vulnerability in VM side Channel attack With leakage on AES cryptographic keys properly surveyed by the author.

Mohamed A.Elshabka et al. [23] 2021	The author proposed a VM placement algorithm.	MRI with RITH VM Placement algorithm is shown.	Paper lacks in suggesting the strongest security assessment model that detects Intrusion detection and prevention.	RITH has improved from 2%to 5%.
K.E Narayana et al. [24] 2021	Types of cloud attacks with their solutions are discussed Using AES and DES.	ALIBABA Cloud and AWS shield were used in this paper to detect cloud-specific attacks.	Very little Literature Survey was done by the author.	Flush + Reload attack detection on RSA and Flush + flush attacks detection on AES is shown in the result section.
Adi Maheswara Reddy Gali et al. [25] 2020	The author proposed dynamic and scalable placement algorithms by using The author's SF policy. Limitation of NOMAD placement algorithm is also used.	The author used 20 different VM in different servers and showed better efficiency in parameters such as hit rate, loss rate, and loss of resources as compared to other VM placement policies.	Metric for side-channel attacks not explained properly.	A comparison of different Mitigation techniques for SVF and SNR is properly shown in the Result section.
jin B. Hong et al. [26] 2019	Attacks in the cloud are categorized using OWASP attack, STRIDE threat model, and its vulnerability are also discussed.	48 different types of cloud attacks are shown in this paper.	Paper lacks in providing the future work in a detained way.	Threat Identification method steps and actions are shown by proper diagrams.

Shahid Anwar et al. [27] 2017	CPU Cache memory-based side-channel the attack along with their solutions was memory-based discussed attacks High-level in this paper.	Prevention of cross VM side-channel attack shown adequately.	Proper Result analysis is missing in this paper.	The existing cache-based side-channel attack is properly shown in Table format.
Marco Chiappetta et al. [28] 2016	Introduction of three methods for detection of spy process performed on cache side-channel attack using FLUSH + RELOAD Technique.	Square Multiply algorithm, FLUSH RELOAD attack on RSA, ECDSA signature, Double and add point scalar multiplication, Montgomery ladder point scalar multiplication was properly discussed.	Hardware performance counters to be elaborated.	The relationship between total LLC accesses of the AES victim process and the edited version of the spy process graph is shown.
Muhammed, sadique et al. [29] 2016	Data Theft can be stopped using the PTP technique.	Statistical Analysis shows that algorithms are efficient.	More Literature Paper to be added.	Comparison of flush () and wait () is shown in graph form.

Zhenyu Wu et al. [30] 2014	A covert channel attack in the cloud is proposed. It is properly analyzed and mitigated.	Classic Cache Channel protocol, Timing-based cache channel protocol, Reliable Timing Based Memory Bus channel Protocol are ciphertext applied in the paper.	Improvement is possible in his proposed method.	Effects of non-participating workload on bandwidth and error rate are shown by the graph. Reliable transmission with adaptive rates on EC 2 micro instances is properly shown by graph form.
Johanna Ullrich et al [31] 2017	Survey on Network-based secret communication in the cloud.	Comparing different attacks shown in the paper.	Results and finding to be properly elaborated.	covert channels in cloud computing and Obfuscation in cloud computing are properly shown in the result.
By Prashant Pranav et al. [32] 2017	A credit-based scheduling algorithm is proposed.	Time wasted in allocating jobs to VM is saved reasonably.	SJF, FCFS, Round Robin Methods are ignored by the authors.	Comparison of waiting time of priority factor-based and the credit-based algorithm is properly shown in Table.

4. PROPOSED METHODOLOGY

To defy the VM timing side-channel attack, we have analyzed the time complexity of AES 128 both theoretically and empirically.

*AES Algorithm***:** Advanced Encryption Standard is an asymmetrical block cipher algorithm that takes uses the plain text of 128 bits and converts it into ciphertext using keys pattern of 128,192 and 256 bits.AES algorithm is accepted worldwide.US Government standardized the AES algorithm in the 1990s.AES is adjustable to modern processors. It has both symmetrical and parallel structures.AES also works on suitable smart cards. It is a much faster and more

secure algorithm as compared to DES.

AES has four rounds of the encryption process. Each consists of four sub-processes. Byte Substitution (Sub-byte):16 input bytes are inserted in the fixed table (S-Box). Four columns and four rows are created in the matrix terms.

Shift-Rows: Four rows of the matrix are shifted to the left side. The first row is not moved. the second row shifted to one byte in the left position. the third row shifted to two positions on the left. The fourth row is transferred three positions to the left side. A new matrix is generated of the same 16 bytes but positions are changed to each other.

Mix-Columns: Four-byte columns are now moved using mathematical functions. It takes as input the four bytes of one column and generates four new bytes. Thus replacing the original column which results in creating a new matrix of 16 bytes. This step was not carried out in the last round.

Add-Round key: The matrix of 16 bytes is now changed to 128 bits. XOR gate is added to 128 bits of the round key. if the step stops, it's the final ciphertext otherwise similar rounds were further repeated.

4.1.Theoretical Complexity of AES 128

The time of AES encryption/decryption in any of the standard modes like CBC or CTR or GCM is polynomial in the size of the message. One call to the AES encryption/decryption function takes some constant number of steps, which we can represent with the constant cc. For example, AES-128 makes one call to the key schedule to generate the round keys, one 'whitening' step with a round key, and then 10 calls to the round function, which itself takes a constant number of simpler steps - 16 parallel applications of the S-Box, one 'Shift Rows' transposition of bytes, 4 parallel applications of the Mix Columns linear function (absent in the last round), and one Add Round Key step. AES-192 and AES-256 similarly make constant numbers of calls to the round function. For any of the standard modes and a message of bit-length ℓ, the number of calls to the AES encryption/decryption function is some linear function, $f(\ell)$f(ℓ).

For example-"the cipher-call function for CTR is $fCTR(\ell)=[\ell 128]$fCTR(ℓ)=[l128]. Sometimes these modes have to do other things than make calls to the block cipher (e.g. GCM has to do one finite field multiplication per 128-bit message block), but in the standard modes, these are also always linear or constant in the length of the message. Let's say the function $g(\ell)$g(ℓ) returns the number of 'extraneous' (non-AES) steps as a function of the

message length. Therefore, the encryption/decryption time for AES in any of the standard modes is some linear function of the length of the message, c.f(ℓ)+g(ℓ)c.f(ℓ)+g(ℓ).”

4.2.Empirical Complexity of AES - 128

AES 128 has four sub-steps. We have measured the time complexity of each individual sub-step empirically. We have taken three different trials of input sizes to measure the complexity of each sub-step of varying and increasing input.

Tables 1, 2, 3, are cipher text and 4 below shows the mean execution time of the three trials for varying input sizes of substitution byte, shift row, add-round, and mix-column respectively. We plotted the fitted line plot of each sub-step as shown in fig 1, 2, 3, 4, respectively. As it can be seen from figure 4, the mix column sub-step is the most dominant sub-step among the sub-steps of AES 128. So, it can be said that the overall time complexity of AES-128 largely depends on the mix column sub-step. The mix column sub-step as can be seen from the fitted line plot has an empirical complexity of O (n).

The Theoretical time complexity of AES 128 is O (n) where n is the number of inputs. So, it can be said that the mix column sub-step solely determines its time complexity. The rest of the three sub-steps take very little time for execution so, in this paper, we have proposed to insert a random amount of time randomly in any of the three sub-steps. This random amount of time added to either of shift -row, sub-byte, and add-round key will be aligned with the execution time of these sub-steps for varying and increasing input sizes.

TABLE 1: EXECUTION TIME FOR SUBSTITUTE BYTES

INPUT (In Bytes)	**TRIAL 1**	**TRIAL 2**	**TRIAL 3**	**MEAN EXECUTION TIME IN SECOND**
16	0.000130	0.000013	0.000013	0.000052
32	0.000150	0.000014	0.000008	0.000057
48	0.000140	0.000015	0.000012	0.000056
64	0.000120	0.000016	0.000011	0.000049
80	0.000116	0.000018	0.000013	0.000049
96	0.000190	0.000019	0.000011	0.000073
112	0.000140	0.000013	0.000015	0.000056
128	0.000190	0.000016	0.000009	0.000072
144	0.000210	0.000018	0.000014	0.000081
160	0.000170	0.000022	0.000008	0.000067
176	0.000210	0.000021	0.000013	0.000081

192	0.000180	0.000020	0.000012	0.000071
208	0.000140	0.000018	0.000013	0.000057
224	0.000150	0.000021	0.000015	0.000062
240	0.000160	0.000023	0.000021	0.000068
256	0.000230	0.000020	0.000022	0.000091

TABLE 2: EXECUTION TIME FOR SHIFT ROWS

INPUT (In Bytes)	**TRIAL 1**	**TRIAL 2**	**TRIAL 3**	**MEAN EXECUTION TIME IN SECOND**
16	0.000025	0.000021	0.000019	0.0000217
32	0.000021	0.000016	0.000012	0.0000163
48	0.000027	0.000019	0.000022	0.0000227
64	0.000023	0.000017	0.000013	0.0000177
80	0.000024	0.000015	0.000013	0.0000173
96	0.000021	0.000021	0.000016	0.0000193
112	0.000029	0.000022	0.000027	0.0000260
128	0.000023	0.000028	0.000019	0.0000233
144	0.000031	0.000019	0.000024	0.0000247
160	0.000026	0.000021	0.000021	0.0000227
176	0.000022	0.000019	0.000019	0.0000200
192	0.000024	0.000031	0.000021	0.0000253
208	0.000032	0.000027	0.000031	0.0000300
224	0.000024	0.000019	0.000021	0.0000213
240	0.000029	0.000022	0.000018	0.0000230
256	0.000028	0.000024	0.000025	0.0000257

TABLE 3: EXECUTION TIME FOR ADD ROUND KEY

INPUT (In Bytes)	**TRIAL 1**	**TRIAL 2**	**TRIAL 3**	**MEAN EXECUTION TIME IN SECOND**
16	0.0000013	0.0000080	0.0000040	0.00000443
32	0.0000019	0.0000090	0.0000050	0.00000530
48	0.0000015	0.0000014	0.0000012	0.00000137
64	0.0000021	0.0000015	0.0000008	0.00000147
80	0.0000016	0.0000019	0.0000090	0.00000417
96	0.0000015	0.0000017	0.0000070	0.00000340

112	0.0000018	0.0000017	0.0000018	0.00000177
128	0.0000019	0.0000019	0.0000080	0.00000393
144	0.0000022	0.0000016	0.0000017	0.00000183
160	0.0000016	0.0000019	0.0000014	0.00000163
176	0.0000021	0.0000018	0.0000017	0.00000187
192	0.0000016	0.0000021	0.0000011	0.00000160
208	0.0000023	0.0000018	0.0000019	0.00000200
224	0.0000026	0.0000021	0.0000018	0.00000217
240	0.0000024	0.0000025	0.0000021	0.00000233
256	0.0000026	0.0000023	0.0000022	0.00000237

TABLE 4: EXECUTION TIME FOR MIX COLUMN

INPUT (In Bytes)	**TRIAL 1**	**TRIAL 2**	**TRIAL 3**	**MEAN EXECUTION TIME IN SECOND**
16	0.02280	0.02590	0.02180	0.023500
32	0.05330	0.05430	0.05550	0.054367
48	0.08640	0.08650	0.08530	0.086067
64	0.11430	0.11370	0.11630	0.114767
80	0.13520	0.14760	0.14530	0.142700
96	0.17550	0.15840	0.17850	0.170800
112	0.20560	0.20860	0.19570	0.203300
128	0.23530	0.22830	0.23500	0.232867
144	0.29500	0.24970	0.25320	0.265967
160	0.29600	0.30120	0.30740	0.301533
176	0.32350	0.30630	0.32380	0.317867
192	0.35900	0.33350	0.35560	0.349367
208	0.39800	0.47870	0.46500	0.447233
224	0.55320	0.40950	0.49380	0.485500
240	0.44680	0.51860	0.61860	0.528000
256	0.53380	0.59250	0.53540	0.553900

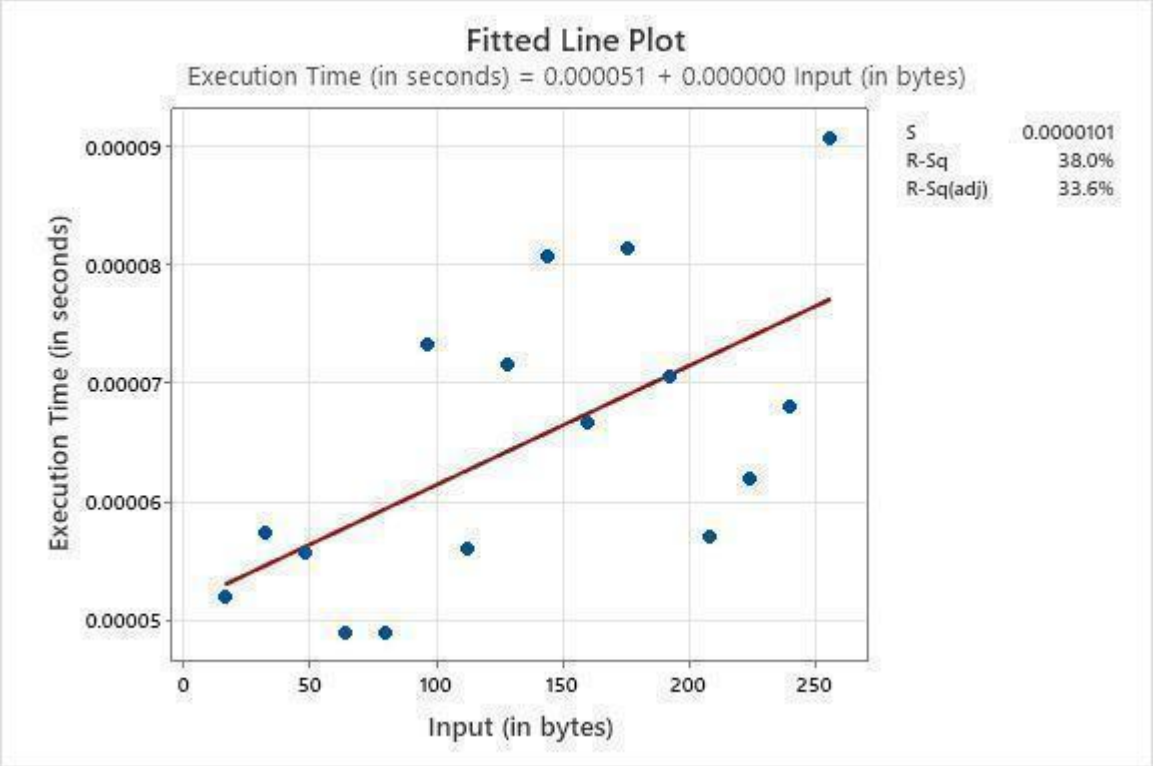

Figure 1: Fitted Line Plot of Substitute Bytes

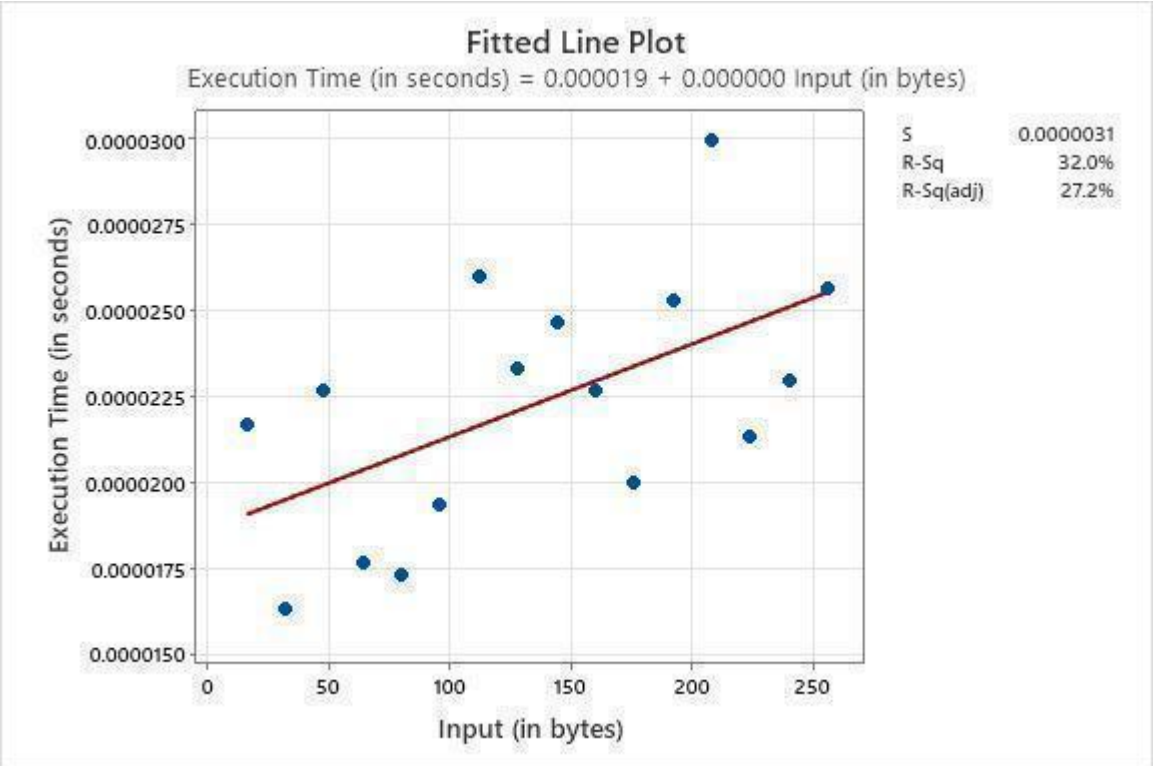

Figure 2: Fitted Line Plot of Shift Rows

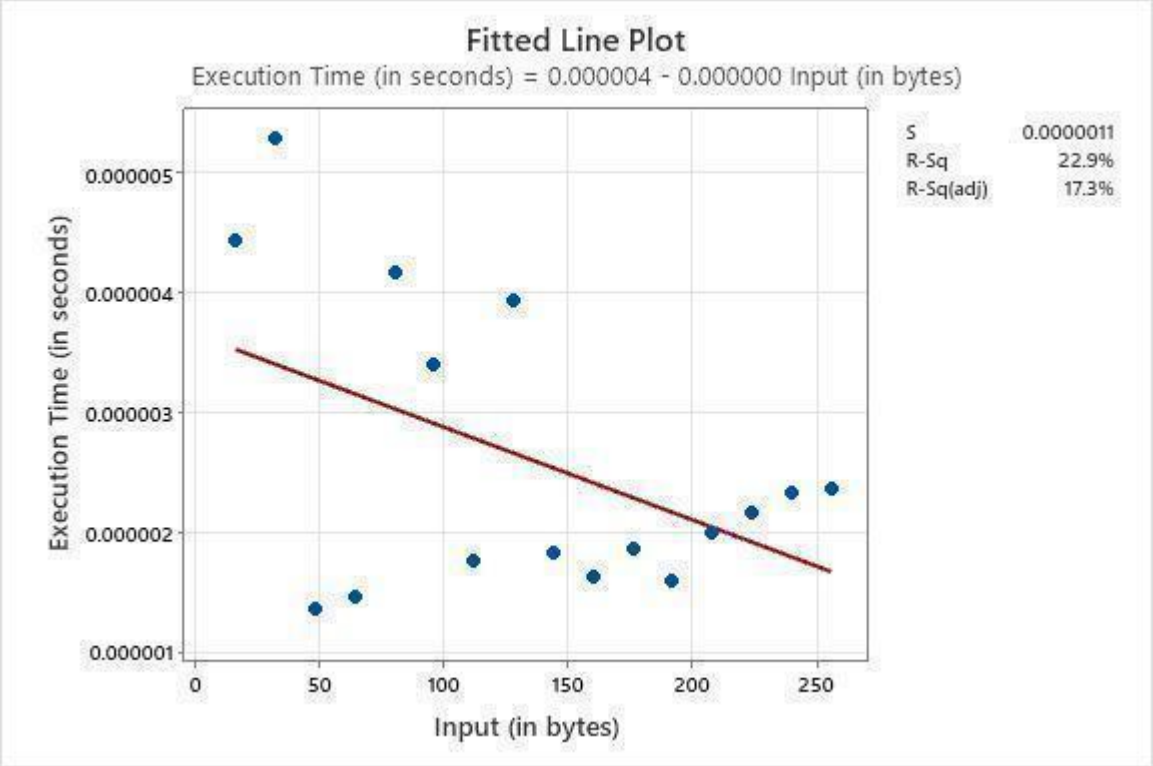

Figure 3: Fitted Line Plot of Add Round Key

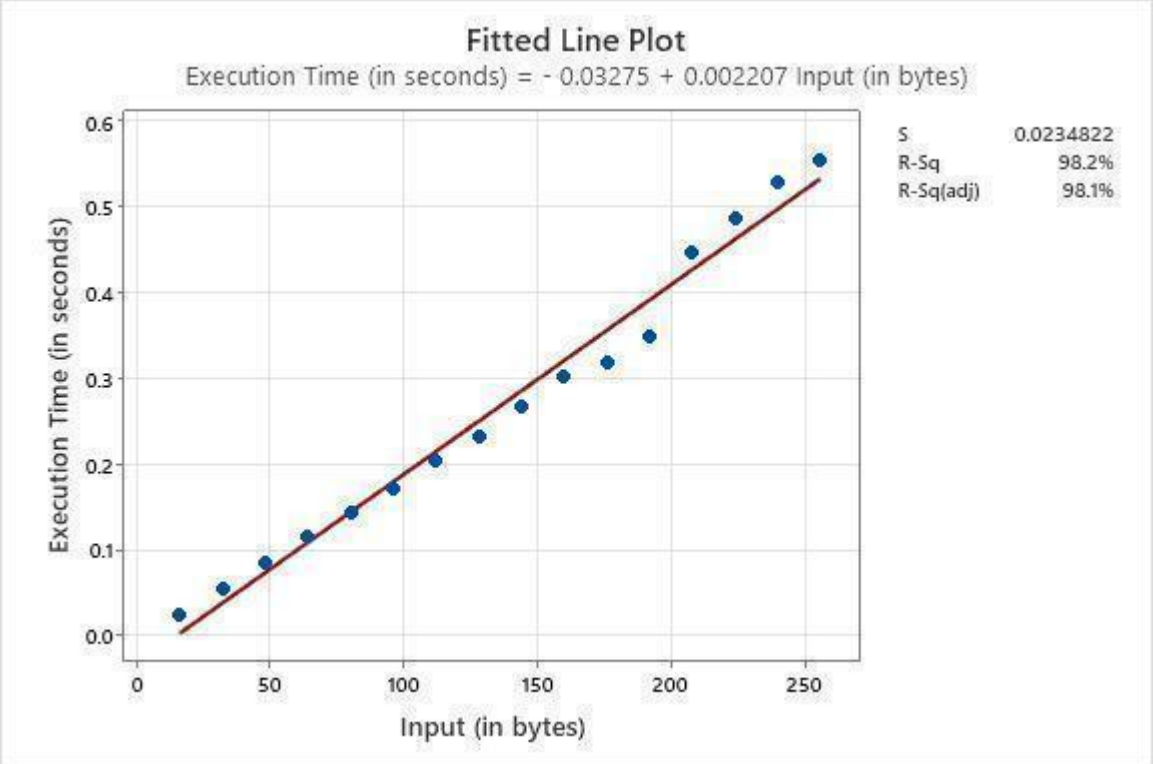

Figure 4: Fitted Line Plot of Mix Column

4.3.Proposed Algorithm

We have proposed the following algorithm to defy an intruder of guessing the time for the

processes of encryption and decryption of AES – 128.

ALGORITHM

Input: Plain Text Message in the form of bits/ bytes to be fed to AES -128

Output: Cipher Text

While (End of Plain Text)

do

Step 1: Compute the execution time for varying and increasing Plain Text size for 3 different trials of

- Step 1.1: Shift Row
- Step 1.2: Substitute Bytes
- Step 1.3: Mix Column
- Step 1.4: Add Round Key

Step 2: Take the mean of the three trials to get the final execution time of the above sub-steps for different input

Step 3: Fit a statistical model taking into consideration the two parameters viz. Input Size and Mean Execution Time

Step 4: Estimate an empirical run time for the following

- Step 4.1: Shift Row
- Step 4.2: Substitute Bytes
- Step 4.3: Mix Column
- Step 4.4: Add Round Key

Step 4: Take a sub-step randomly by generating a random number between [1 and 3] among

- Shift Row
- Substitute Bytes
- Add Round Key

Step 5: Add a random amount of time for a specific input size from Step 2 by generating a random variable between the 'least execution time and mean execution time' for the sub-step, selected in Step 4

Fig 5, 6, 7, 8 shows the residual plot for four sub-steps i.e. substitution– byte, shift row, add round key, and mix-column respectively.

As seen in fig 5, the residual plot for the execution time of sub-byte contains four subplots: normal probability plot, versus fits, histograms, versus order.

The normal probability plot of the sub-byte depicts that the data are normally distributed. The

verses fit the plot of the execution time of sub - byte shows there is no unusual data point in our fitted model. The histogram of the sub-byte sub-step does not show the presence of an outlier in our data and it also shows that residuals are not skewed. The versus order of the sub-byte step shows that the residuals are randomly distributed among the normal line. The same can be interpreted from the residual plot of shift rows, add-round key, and mix column sub-step, shown in fig 6, 7, 8 respectively.

The verses fit the plot of the mix column sub-step shown in fig 8, which suggests that the residual and the fitted value are co-related, which means that the empirical time complexity which is predicted for the mix column sub-step i.e. *O(n)* is the sub-steps time complexity of the mix column sub-steps.

Figure 5: Residual Plot of Substitute Byte

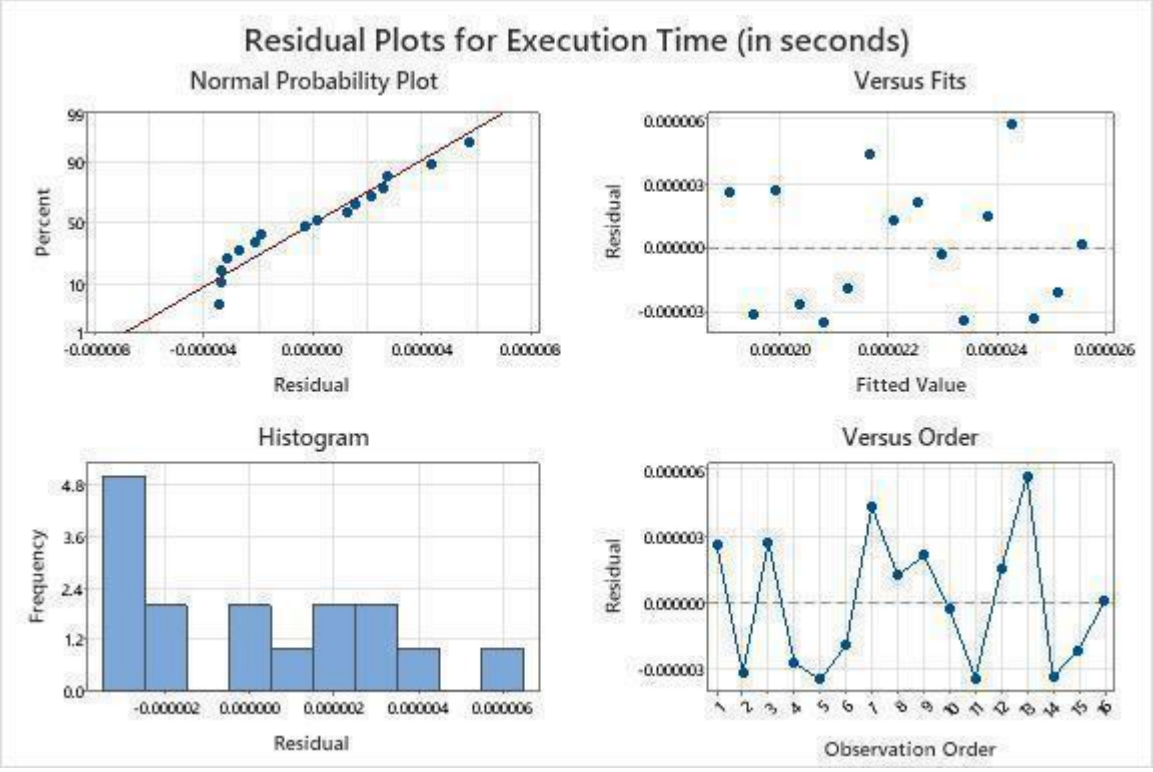

Figure 6: Residual Plot of Shift Rows

Figure 7: Residual Plot of Add Round Key

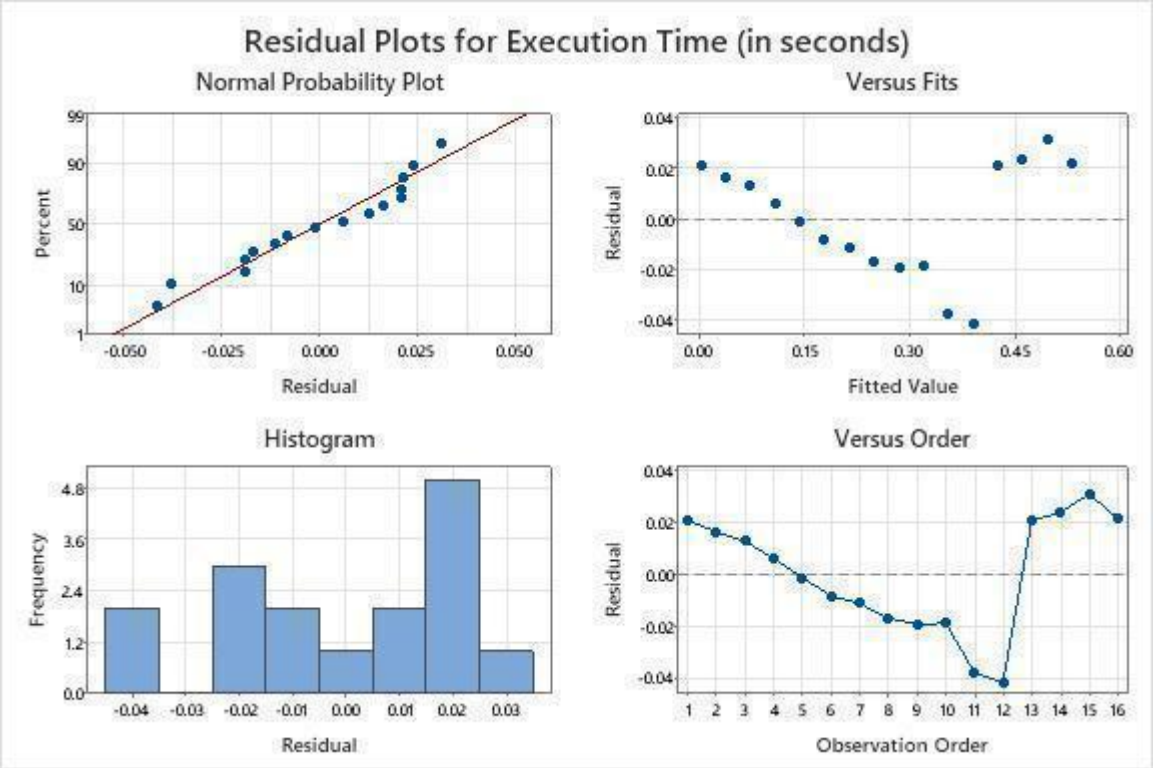

Figure 8: Residual Plot for Mix Column

5. CONCLUSION AND FUTURE WORK

In this paper, we have tried to defy the VM side-channel timing attack by adding some random timing to existing sub-steps of AES 128. WE have computed the time complexity of each sub step of AES - 128 empirically which shows that the mix column sub step is the most dominant sub step. So, we select any three of the sub steps randomly and insert some random amount of time comparable to the execution time of a specific sub step. This will defy the intruder of guessing the time required by the algorithm to perform any operation

In the future, we will try checking the proposed method in RSA and ElGammal cryptosystems

Compliance with Ethical Standard

Ethical Approval: We have adhered to the accepted ethical standards of a genuine research study

Funding Details

The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Conflict of Interest: The authors declare that they do not have any conflict of interest.

Informed Consent: The authors have consent to carry out the research

Author Contributions

All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Animesh Kumar, Sandip Dutta and Prashant Pranav. The first draft of the manuscript was written by Animesh Kumar and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

References:

1. Velliangiri, S., Karthikeyan, P. and Vinoth Kumar, V., 2021. Detection of distributed denial of service attack in cloud computing using the optimization-based deep networks. *Journal of Experimental & Theoretical Artificial Intelligence*, *33*(3), pp.405-424.
2. Shaikh, A.H., and Meshram, B.B., 2021. Security issues in cloud computing. In *Intelligent Computing and Networking* (pp. 63-77). Springer, Singapore.
3. Ahmad, W., Rasool, A., Javed, A.R., Baker, T. and Jalil, Z., 2022. Cyber Security in IoT-Based Cloud Computing: A Comprehensive Survey. *Electronics*, *11*(1), p.16.
4. Shusterman, A., Agarwal, A., O'Connell, S., Genkin, D., Oren, Y. and Yarom, Y., 2021. Prime+ Probe 1, JavaScript 0: Overcoming Browser-based Side-Channel Defenses. In *30th {USENIX} Security Symposium ({USENIX} Security 21)*.
5. Mata, P. and Rao, N., 2021, April. Flush-Reload Attack and its Mitigation on an FPGA Based Compressed Cache Design. In *2021 22nd International Symposium on Quality Electronic Design (ISQED)* (pp. 535-541). IEEE.
6. Sönmez, B., Sarıkaya, A.A. and Bahtiyar, Ş., 2019, September. Machine Learning based Side Channel Selection for Time-Driven Cache Attacks on AES. In *2019 4th International Conference on Computer Science and Engineering (UBMK)* (pp. 1-5). IEEE.
7. Xinliang, M., Liehui, J. and Rui, C., 2020. Survey of Access-Driven Cache-Based Side Channel Attack. *Journal of Computer Research and Development*, *57*(4), p.824.
8. Saxena, S. and Sanyal, G., 2018, October. Cache Based Side Channel Attack: A Survey. In *2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)* (pp. 278-284). IEEE.
9. Tao, X., Wang, L., Xu, Z. and Xie, R., SCAMS: A Novel Side-Channel Attack Mitigation System in IaaS Cloud. In *MILCOM 2021-2021 IEEE Military Communications Conference (MILCOM)* (pp. 329-334). IEEE.
10. Lusky, Y. and Mendelson, A., 2021, April. Sandbox Detection Using Hardware Side Channels. In *2021 22nd International Symposium on Quality Electronic Design (ISQED)* (pp. 192-197). IEEE.
11. Makrani, H.M., Sayadi, H., Nazari, N., Khasawneh, K.N., Sasan, A., Rafatirad, S. and Homayoun, H., 2021, September. Cloak & Co-locate: Adversarial Railroading of Resource Sharing-based Attacks on the Cloud. In *2021 International Symposium on Secure and Private Execution Environment Design (SEED)* (pp. 1-13). IEEE.
12. Pranav, P., Dutta, S. and Chakraborty, S., 2021. Empirical and statistical comparison of intermediate steps of AES-128 and RSA in terms of time consumption. *Soft Computing*, *25*(21), pp.13127-13145.
13. Mi, Z., Chen, H., Zhang, Y., Peng, S., Wang, X. and Reiter, M.K., 2018. Cpu elasticity to mitigate cross-VM runtime monitoring. *IEEE Transactions on Dependable and Secure Computing*, *17*(5), pp.1094-1108.
14. Van Cleemput, J., De Sutter, B. and De Bosschere, K., 2017. Adaptive compiler strategies for mitigating timing side-channel attacks. *IEEE Transactions on Dependable and Secure Computing*, *17*(1), pp.35-49.
15. Hsiao, S.W., Sun, Y.S. and Chen, M.C., 2020. Hardware-Assisted MMU Redirection for In-Guest Monitoring and API Profiling. *IEEE Transactions on Information Forensics and Security*, *15*, pp.2402-2416.
16. Levitin, G., Xing, L.,side-channel and Xiang, Y., 2020. Co-residence data theft attacks on N-Version programming-based cloud services with task cancelation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*.
17. Atya, A.O.F., Qian, Z., Krishnamurthy, S.V., La Porta, T., McDaniel, P. and Marvel, L.M., 2019. Catch me if you can: A closer look at malicious co-residency on the cloud. *IEEE/ACM Transactions on*

Networking, *27*(2), pp.560-576.
18. Alhulayyil, H., Khalil, K., Krishnamurthy, S.V., Cansever, D., La Porta, T. and Swami, A., 2018, December. On the Detection of Adaptive Side-Channel Attackers in Cloud Environments. In *2018 IEEE Global Communications Conference (GLOBECOM)* (pp. 1-6). IEEE.
19. Gulmezoglu, B., Inci, M.S., Irazoqui, G., Eisenbarth, T. and Sunar, B., 2016. Cross-VM cache attacks on AES. *IEEE Transactions on Multi-Scale Computing Systems*, *2*(3), pp.211-222.
20. Saxena, S., Sanyal, G., Srivastava, S. and Amin, R., 2017. Preventing cross-VM side-channel attacks using a new replacement method. *Wireless Personal Communications*, *97*(3), pp.4827-4854.
21. Liu, F., Ge, Q., Yarom, Y., Mckeen, F., Rozas, C., Heiser, G. and Lee, R.B., 2016, March. Catalyst: Defeating last-level cache side-channel attacks in cloud computing. In *2016 IEEE international symposium on high-performance computer architecture (HPCA)* (pp. 406-418). IEEE.
22. Litchfield, A. and Shahzad, A., 2016. Virtualization technology: Cross-VM cache side-channel attacks make it vulnerable. *arXiv preprint arXiv:1606.01356*.
23. Elshabka, M.A., Hassan, H.A., Sheta, W.M. and Harb, H.M., 2021. Security-aware dynamic VM consolidation. *Egyptian Informatics Journal*, *22*(3), pp.277-284.
24. Narayana, K.E., and Jayashree, K., 2021. Survey on cross virtual machine side-channel attack detection and properties of cloud computing as a sustainable material. *Materials Today: Proceedings*, *45*, pp.6465-6470.
25. Gali, A.M.R. and Koduganti, V.R., 2021. Dynamic and scalable virtual machine placement algorithm for mitigating side-channel attacks in cloud computing. *Materials Today: Proceedings*.
26. Hong, J.B., Nhlabatsi, A., Kim, D.S., Hussein, A., Fetais, N. and Khan, K.M., 2019. Systematic identification of threats in the cloud: A survey. *Computer Networks*, *150*, pp.46-69.
27. Anwar, S., Inayat, Z., Zolkipli, M.F., Zain, J.M., Gani, A., Anuar, N.B., Khan, M.K. and Chang, V., 2017. Cross-VM cache-based side-channel attacks and proposed prevention mechanisms: A survey. *Journal of Network and Computer Applications*, *93*, pp.259-279.
28. Chiappetta, M., Savas, E. and Yilmaz, C., 2016. Real-time detection of cache-based side-channel attacks using hardware performance counters. *Applied Soft Computing*, *49*, pp.1162-1174.
29. Sadique, U.M. and James, D., 2016. A novel approach to prevent the cache-based side-channel attacks in the cloud. *Procedia Technology*, *25*, pp.232-239.
30. Wu, Z., Xu, Z. and Wang, H., 2014. Whispers in the hyper-space: high-bandwidth and reliable covert channel attacks inside the cloud. *IEEE/ACM Transactions on Networking*, *23*(2), pp.603-615.
31. Ullrich, J., Zseby, T., Fabini, J. and Weippl, E., 2017. Network-based secret communication in clouds: A survey. *IEEE Communications Surveys & Tutorials*, *19*(2), pp.1112-1144.
32. Pranav, A.P., Rizvi, N. and Jha, B.R., 2017, February. QoS Aware VM Allocation Policy in Cloud using a Credit based Scheduling Algorithm. In *Communication and Computing Systems: Proceedings of the International Conference on Communication and Computing Systems (ICCCS 2016), Gurgaon, India, 9-11 September, 2016* (p. 445). CRC Press.

Statements & Declarations

Funding

The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Competing Interests

The authors have no relevant financial or non-financial interests to disclose

Author Contributions

All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Animesh Kumar, Sandip Dutta and Prashant

Pranav. The first draft of the manuscript was written by Animesh Kumar and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Data Availability

Data may be made available on request