Computer and Information Science; Vol. 17, No. 1; 2024
ISSN 1913-8989 E-ISSN 1913-8997
Published by Canadian Center of Science and Education

On Preventing and Mitigating Cache Based Side-Channel Attacks on AES System in Virtualized Environments

Abdullah Albalawi[1]

[1] Department of Computer Science, Shaqra University, Kingdom of Saudi Arabia

Correspondence: Abdullah Albalawi, Department of Computer Science, Shaqra University, Kingdom of Saudi Arabia. E-mail: aalbalawi@su.edu.sa

Received: January 10, 2024 Accepted: February 19, 2024 Online Published: February 27, 2024

doi:10.5539/cis.v17n1p9 URL: https://doi.org/10.5539/cis.v17n1p9

Abstract

Cloud computing aims to cut costs through a reduction in spending on equipment, infrastructure, and software by applying the multi-tenancy feature. Despite all the benefits of multi-tenancy, it is still a source of risk in cloud computing. Cloud adoption may be hampered by security concerns if suitable cloud-based security solutions are not available. Moreover, virtualization that enables multi-tenancy, considered one of the main components of a cloud, introduces major security risks and does not offer appropriate isolation between different instances running on the same physical machine. In this paper, we present a preliminary idea that may support the development of new countermeasures for a particular type of threat, namely cache-based side-channel attacks that target cache memories in virtualized environments. Attackers specifically target virtual machines in this type of attack to create many side channels and gather sensitive data. Additionally, this research offers preliminary concepts to aid in developing of solutions or defenses that enable us to identify unusual activity that could point to attacks associated with multi-tenancy, as well as security measures that preserve the benefits of multi-tenancy while lowering security concerns.

Keywords: Side-channel attacks, memory deduplication, flush+reload, flush+flush

1. Introduction

Cloud computing is a promising technology that may optimize operations and enhance economic and operational efficiency, resulting in considerable cost reductions (Takabi et al., 2010). Essential services like social networking and commercial apps may also be operated in the cloud. Access to a shared pool of reconfigurable computing resources (including networks, servers, storage, apps, and services) that may be provided and released with little administration labour or service provider contact is made possible by cloud computing (Garrison et al., 2012).

One essential component of clouds is multi-tenancy. By dividing up a virtualized, shared infrastructure across multiple users, it enables cloud providers to maximize utilization of resources (Takabi et al., 2010). The automatic resource allocation algorithms used by cloud computing providers might result in two or more virtual machines representing distinct users residing on the same physical machine and sharing the same resources in certain situations (Aljahdali et al., 2013). However, there might be a confidentiality breach if the cloud's resources are shared with an adversarial user who uses the placement policy or allocation procedures to co-locate their virtual machine with the designated target virtual machine (Saxena et al., 2017) due to co-resident attacks being achieved. This allows and facilitates the exploitation of side channels for attacks.

According to Anwar et al. (2017), in the context of cryptography, side-channel attacks are physical attacks in which the confidential information of cryptographic methods, such as an encryption key, is illegally retrieved using a physical technique.. The side-channel attack uses the shared hardware resources as a secret path via which the attacker obtains crucial data, such as the cryptographic key or any other sensitive information about the victim. Side-channel attacks are not new, but their impact is increasingly being felt in cloud computing due to the sharing of resources to downsize the cost of computing. Side-channel attacks are one of the most efficient methods and they have successfully broken almost all cryptographic algorithms of today; indeed, they are a serious threat to the security of cryptographic modules (Saxena et al., 2017). Different forms of side-channel attacks can be distinguished according to virtualization, implementation, and module access.

This paper focuses on the Flush+Reload cache attack found on the AES cryptosystem. We have also created a suitable environment for the attack using a type-two hypervisor whilst accomplishing the attack. Appropriate solutions are reviewed and how to solve the problem and mitigate its effects are discussed.

The rest of the paper is organized as follows. Section 2 reviews the background. Section 3 presents related work. Section 4 discusses our research objectives and illustrates the proposed solutions. Finally, section 5 provides a brief conclusion.

2. Background

This section discusses some important topics and information related to the project to better understand and recognize the paper's objectives.

2.1 Multi-Tenancy

The main goal of multi-tenancy is the sharing of resources, which is the fundamental aim of cloud computing. It enables many users to be served and save costs, alongside other benefits. For example, when we run a single instance of a software application on one instance of a database and provide this application to multiple users, all users' data are hidden and isolated from other users, and each user is called a tenant of the multi-tenancy (AlJahdali et al., 2014). However, multi-tenancy presents severe security threats and opens doors for possible privacy leaks (Ren et al., 2007). Moreover, it presents the hazard of sharing a server with an adversary VM, thus risking the extraction of secret information via side-channels (Varadarajan et al. 2015).

Multi-tenancy is a security concern in cloud computing because it gives malicious parties the ability to access a shared server (Keller et al., 2010). In the realm of cloud computing, multi-tenancy has given rise to a variety of discussions. Software developers view it as an opportunity, while security professionals view it as a weakness that may result in breaches of confidentiality and, as a result, a problem that has to be resolved (AlJahdali et al., 2014).

2.2 Virtualization

Virtualization is making one physical machine act as many machines using a piece of software called Hypervisor (ab- straction layer), which can create a virtualization layer that makes server virtualization possible and contains a virtual machine manager (VMM) which enables the simultaneous operation of many operating systems on a host machine while monitoring how guests that share virtualized resources are being operated (Jasti et al., 2010), such as VMware ESX. Thus, we can run many machines in one physical host with lower cost, higher performance and faster maintenance.

Virtualization's primary objective is to manage workload by increasing the computing power's scalability, economy, and efficiency. Virtualization may be implemented at the hardware and software levels (Malhotra et al., 2014). Although virtualization has positive effects on a cloud computing environment, Keller et al. (2010) propose the removal of the virtualization layer as a preventive measure against multi-tenancy, a practice that is regarded as a security risk and vulnerability.

The architecture of virtualized technology consists of cloud users, service models, virtualized models, and host software and their hardware. Virtualization enables the running of multiple operating systems and multiple applications on the same physical host at the same time (Malhotra et al., 2014), which could be exploited by an attacker to achieve a co-resident attack of other victims on the same server (Keller et al., 2010).

2.3 Memory Deduplication

Many hypervisors use a method called memory deduplication, including VMWare ESX, Xen, and Linux KVM. Memory deduplication increases memory utilization by allowing the data to be shared between virtual machines and removing multiple copies from memory (Albalawi et al., 2021). The fundamental idea is that in the case where multiple memory pages contain identical content, the hypervisor is solely required to retain a single copy. While memory deduplication improves memory efficiency, it has a major impact on system security. It is possible to obtain confidential information about the encryption processes running in the target virtual machine by taking advantage of the memory deduplication feature. It is possible to obtain confidential information about the processes running in the target virtual machine by using the same data of shared encryption libraries. Additionally, an attacker may prime data to the cache and wait for a victim to access it. The victim's behavior may be inferred from the victim's access to the cached data (Xiao et al., 2013; Irazoqui et al., 2014). For widely used hypervisors, a variety of memory deduplication techniques are used, such as the KVM hypervisor, which is implemented on the Linux kernel and uses the Kernel Same page Merging (KSM) technique. Virtual machine memory pages are automatically searched by KSM for identical pages, and signatures are generated for these

pages. In the deduplication table, these signatures are stored. The pages are then deduplicated if two or more are discovered to have the same signature (Lindemann and Fischer, 2018).

As shown in Figure 1, within a virtualized environment, when the virtual machine of the victim and the virtual machine of the adversary are co-located on the same host computer, the adversary has the ability to access memory information from the virtual machine that is co-located with the victim. By loading the page into its own memory, waiting for a little until memory deduplication takes effect, and then writing to that page, for instance, the adversary may be able to detect whether or not a certain page is present in the memory of the co-located virtual machine (VM). Writing to a page that has been deduplicated will need more time than writing to a page that does not have any duplicates. With this knowledge, the attacker will be able to determine whether or not the page already exists on a virtual machine that is located nearby, which will result in the disclosure of sensitive information (Xiao et al., 2013).

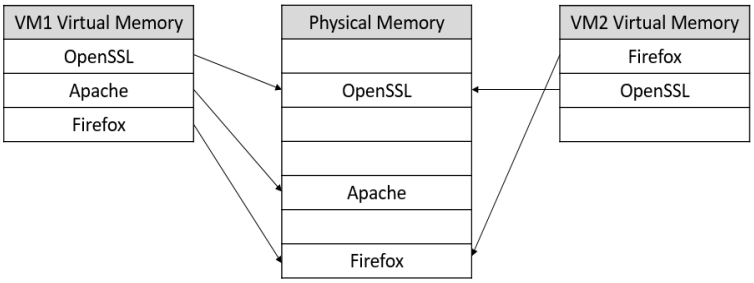

Figure 1. Memory Deduplication Feature

2.4 Cloud Side-channel Attacks

Cloud computing comes with its security problems; in this work, we are concerned with co-residency data leakage prob- lems. Because the same physical host is shared and only partitioned by a virtualization layer, there is a risk of data leakage, rather than assigning a dedicated machine to every user that leads to breaking of VM isolation and recovery of cryptosystem keys. Past studies have shown that cache side-channel attacks may be carried out in cloud computing, for instance, in Cross-VM Attack on AES implementations by Irazoqui et al. (2014), the attack takes advantage of using Transparent Page Sharing that is applied on virtualization environments. Additionally, Suzaki et al. (2011) use memory deduplication to identify the processes that are active in a target virtual machine. Moreover, Bernstein (2005) executes his attack in a non-virtualized server-client environment, he offers a cache-timing attack that is implemented in a client-server setup and aims to retrieve the AES secret key that the server uses to encrypt the incoming UDP packets (Bernstein, 2005). Also, Irazoqui et al. (2014) use Ubuntu 12.04 and Bernstein's attack on OpenSSL 1.0.1 to recover a portion of an AES key through a cross-VM attack that was executing in XEN and VMware. In terms of Bernstein’s attack, using an identical target machine to implement the profiling phase makes the attack unworkable in many situations (Atici et al., 2018).

Cache attacks and information leaks have been discussed in publications addressing security issues. For instance, Bernstein uses a cache-timing attack on the lookup table of the OpenSSL implementation of AES. Two servers are used in Bernstein's attack. One is an exact copy of the victim's server, while the other is the victim's actual server. The attack performs two primary stages: the profiling and the attack stage. During the profiling stage, the attacker sends a large volume of packets to the same server, which encrypts them using a known key and replies to the attacker with the encryption's timing details. Using an unidentified encryption key, a similar process is carried out once again on the target server during the attack stage. The time profiles from these two stages are then linked, and the most likely key candidate is the key value that has the greatest correlation (Bernstein 2005; Jayasinghe et al., 2010; Atici et al., 2018).

Yarom et al. present the Flush+Reload technique to extract the components of the private key from the GnuPG 1.4.13 implementation of RSA and recover the secret key from an ECDSA signature algorithm (Elliptic Curve Digital Signature Algorithm) (Yarom and Falkner 2014; Yarom and Benger, 2014). Gullasch et al. (2011) apply a cache timing attack on the Advanced Encryption Standard (AES) using the Flush+Reload technique to find out memory accesses. The attack does not need to know any information about the ciphertext or the plaintext. The attack retrieves the key with about 100 encryptions and it was against the 128-bit AES implementation of OpenSSL 0.9.8n on Linux kernel 2.6.23.

Irazoqui et al. (2014) implement Bernstein’s attack in a virtualized environment using Xen and VMware with

Cross VM setting. This attack setting is able to recover the AES secret key from the co-located virtual machine that is running AES encryption. The attack takes a long time to achieve on a core i5 processor. Irazoqui et al. (2014) also implement a cache-based side-channel attack on AES by employing the Flush+Reload technique to recover the full AES key across a virtualized environment. This attack needs to enable a deduplication mechanism called Transparent Page Sharing in VMware. The attack uses VMware ESXI 5.5.0 running Ubuntu 12.04 64-bits guest OSs.

2.5 Cache Attacks

This type of attack uses the shared cache to get data by monitoring the time needed to access certain cache lines. A cache side-channel attack helps an attacker to gain accurate and useful information. As caches have a higher rate of computing interactions between processes, it is considered one of the most important attacks in a cloud. The main approach used to carry out this kind of attack is Prime + Probe, which means that the attacker in the prime phase fills the whole cache set with his or her own data. In the probe phase, the attacker tracks how long it takes to access their own data from the same memory location during the probing phase. If there are significant variations in the execution time, it indicates that another process has removed the memory location and is fetching the data from RAM rather than the cache (Saxena et al., 2017).

The co-residency feature of cloud computing makes Cross-VM cache-based side-channel attacks powerful and active in cloud computing environments because of the shared resources. In order to retrieve the cryptographic key, these attacks employ a monitoring procedure to gather data about the accessed cache line. Because there is a lot of common interaction between virtual machines (VMs), CPU cache is seen to be the most exposed hardware in a cloud (Anwar et al., 2017).

In this kind of attack, the attacker places the malicious VM co-resident with the target VM so that they share the same resources and the attacker can monitor cache timing information by measuring the execution of different operations on the victim's VM. When data is accessed from the cache, it consists of lines containing data that may be written or retrieved. The cache memory is the first place the CPU looks for data when it needs data from memory. When information is cached, it takes less time to put it together. However, when necessary information is not cached, the CPU must obtain it from main memory, which takes more time; this is known as cache miss. The secret key of the encryption algorithm can be extracted by exploiting this timing information. Consequently, the encryption time of a cryptographic procedure is related to the positions of the accessible tables. The cache exposes information about memory accesses to an attacker who may track cache hits and misses by using this timing information to obtain information about the secret key used in the encryption. We recognize that processing a cache miss requires more time than processing a cache hit. Exploiting this information results in a full key recovery. An example is using this timing information in the T table implementation of AES that includes SubBytes, ShiftRows, and MixColumns operations into one table look-up and XORing operation to make the encryption time faster. Information about table values loaded into the cache can show information about the secret state of AES (Yan et al., 2015; Irazoqui et al., 2014).

3. Related Work

In order to mitigate the risk of co-resident attacks, Amazon EC2 offers a service known as Virtual Private Cloud (VPC). VPC is a networking environment that is logically isolated, meaning it has its own private IP space and routing configuration. Customers have the ability to establish a Virtual Private Cloud (VPC) and deploy instances inside the VPC instead of using the extensive network pool of Amazon EC2. Private addresses in VPC instances are only accessible to the owner and cannot be known by any other entity. Consequently, the attacker is unable to make educated guesses about the precise physical whereabouts of a target instance, hence diminishing the risk of co-residence (Xu et al., 2015), However, an attacker may still accomplish co-residence by exploiting a latency-based probing technique and taking advantage of virtual machine placement policies. Essentially, VPC simply reduces the risk of co-residence rather than completely eliminating it (Xu et al., 2015; Varadarajan et al., 2015).

Zhang and Reiter (2013) developed a technology called Duppel, which alters cache timing signals to protect a virtual machine against cache attacks in cloud computing environments (Zhang and Reiter, 2013). On the other hand, Duppel may not be feasible in situations involving Simultaneous Multithreading. Enabling SMT on processors might cause some time-shared caches to function as concurrently shared caches. As a result, Dupple is not a viable option for defending against these attacks (Zhang and Reiter, 2013; Shahzad and A. Litchfield, 2015).

In Han et al. (2014), a modified virtual machine allocation policy is proposed to lower the likelihood of certain

VMs co-locating with the target VMs. This policy is called the previous-selected-server-first (PSSF) policy. In order to reduce the vulnerability of the target virtual machines to an attacker, the key concept is to restrict the total quantity of hosts that each account will utilize. This increases the number of virtual machines that are co-located with the same account. This reduces the vulnerability of the victims to attackers and makes it more difficult for them to distribute their virtual machines. However, this policy only concentrates on security issues. It has obvious drawbacks regarding workload balance and power consumption. Cloud computing brings with it new risks related to the environment of shared resources such as shared caches. To deal with this risk, we can write code to eliminate the signal to make the malicious virtual machines unable to receive the signal by eliminating fine-grained timers using fuzzy timers instead of high-resolution clocks. However, this solution is not feasible for applications that require fine-grained timing information (Vattikonda et al., 2011; Liu et al., 2015).

For limiting cache-based side-channel in a multi-tenant cloud, Shi et al. (2011) proposed an approach that leverages dynamic cache coloring; this approach may provide effective cache isolation between different processes of security-critical operations (Yarom, 2015). On the other hand, the approach requires significant modification for the current cloud model to be correct for deployment (Using virtual machine allocation policies to defend against co-resident attacks in cloud computing, 2015). Moreover, a HomeAlone system is a co-residency direction approach using a side-channel (in the L2 memory cache) as a detection tool. The fundamental concept of HomeAlone is for the tenant to arrange for its virtual machines (VMs), often referred to as friendly VMs, to halt their activities in a designated caching zone temporarily. After that, the tenant monitors how much data is being cached over the next idle time to ensure no unexpected activity has occurred (Zhang et al., 2011). However, as it particularly focuses on the cache-based timing channel (Wu et al., 2012), some other side-channels could be exploited (Xiao and Xiao, 2012).

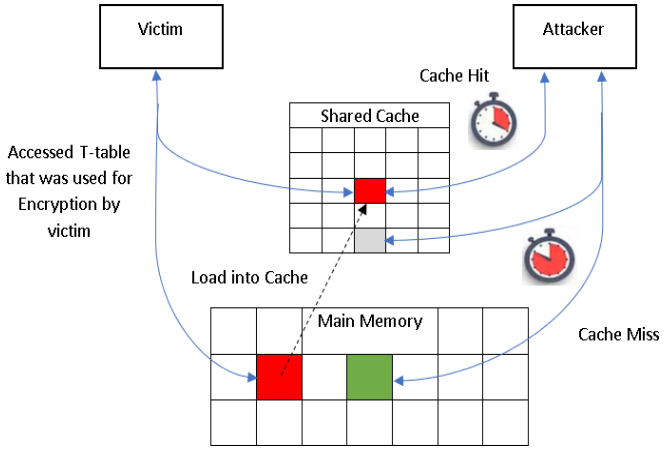

Figure 2. Cache Based Side-Channel Attacks

Despite the research so far, the solutions and mitigations available have multiple limitations, including performance degradation in some cases, concentration on a certain level of cache memory, and the need for significant modification to current cloud models. Regarding these limitations, this paper introduces initial solutions for addressing performance to reduce the risks of cache-based side-channel attacks in a seamlessly applicable manner.

4. Preliminary Solutions

This paper focuses on Flush+Reload cache attacks that target OpenSSL's AES-128's T-table implementation. In this type of attack, the victim's virtual machine and the attacker's virtual machine are on the same host and share the same cache memory between them. This leads to exploitation of the cache memory as a side-channel by using the advantage of the memory deduplication that enables the attacker to reach the same address that the victim uses in the cache. Since all the victim's operations are loaded in the cache memory, the attacker can monitor them if the cache memory is shared and the memory deduplication is enabled. This makes it even more harmful when table-based encryption systems are attacked in this way, and the secret key is broken.

Figure 2 shows the operations that take place inside the cache. Firstly, the attacker must know the offset of the T tables used for encryption. The attacker uses the 'clflush' command to flush the chosen memory lines from the cache to make sure that the flushed memory lines have to be retrieved from the main memory next time when

they need to be accessed. After that, the attacker requests encryptions and receives the ciphertext, then implements Flush+Reload between every encryption process to check whether the T-table values have been used and accessed during the execution of the encryption, by measuring the difference time between a cache hit and a cache miss. We can thwart the attack and make it more difficult for the attacker if we follow the cause of the threat or vulnerability exploited by the attacker, so we must track what the attacker does. We note that the attacker exploits the presence of data used during the encryption process inside the cache memory that is shared between the attacker and the victim. The attacker also checks for data by measuring the time difference between the cache hit and the cache miss; this is an important point to take into account, as this type of attack depends heavily on the process of measuring the time of encryption operations and thus breaking the secret key. Also, we note that the attacker requests several encryptions and applies Flush+Reload between each encryption; this may lead to the detection of the possibility of an attack, as this is considered abnormal behavior. After analyzing and identifying the vulnerabilities exploited by the attacker, we have proposed initial ideas for solutions to reduce the threat.

In this section, we study some suggested solutions that will mitigate these attacks in simple and important ways to protect data in shared environments. Specifically, the project aims to develop security controls that maintain the advantages of multi-tenancy while reducing security risks arising from side-channel attacks with acceptable degradation in performance. This can increase the difficulty for attackers to extract sensitive information. Also, it aims to detect abnormal behavior in the virtualized environment that may indicate multi-tenancy related attacks.

The solutions proposed will now be discussed. First, if we flush the lookup T-tables out of the cache after the encryption process then the attacker will not recognize what lookup T-table value has been accessed because all lookup T-tables are in the main memory. Therefore, the attacker will not detect any time differences when he or she measures the cache hit and miss. Also, if we load the lookup T-tables into the cache memory before the encryption process then the attacker will not recognize which lookup T table value has been accessed because all look up T-tables are in the cache memory and therefore the attacker will not detect any time differences when he measures the cache hit and miss. In addition, the encryption process will be faster. Moreover, if we monitor the activities performed on the cache and monitor the encryption requests then we will be able to detect the possibility of an attack because the attacker needs many encryption requests to mount the attack. Finally, if we control the Time-Stamp Counter ('rdtscp') that is used to read the processor's time stamp counter for the cache hit and miss (Albalawi et al, 2022; Albalawi et al., 2022) then the attacker will never recognize any time differences when he or she measures the cache hit and miss because cache side-channel attacks rely on the measurement of data retrieval time either from the cache memory or from the memory. The below table of proposed solutions shows the benefits of these solutions and the impact of their application.

Table 1. Overview of Proposed Solutions

Solution	Impact	Function
Flushing the AES lookup T-tables	Constant Retrieval Time	Making the attack on AES difficult
Preloading the AES lookup T-tables	Constant Retrieval Time	Making the attack on AES difficult
Monitoring the Activities Performed	Detecting Suspicious Behaviours	Defining and classifying processes
Controlling on the Time-Stamp Counter	Constant Retrieval Time	Making the attack on AES difficult

5. Conclusion

Since the architecture of cloud computing relies on multi-tenancy, which offers a shared environment for multiple users to access and share resources on a single physical machine, attackers can use side-channel attacks to take advantage of the co-residence issue in a multi-tenancy environment and extract sensitive data. This type of attack uses the shared hardware resources as a gateway to obtain sensitive data. There are many types of side-channel attacks that are launched on shared virtualized environments. This paper concentrated on the shared cache memory attack in a virtual- ized environment. The paper proposed preliminary solutions that preserve the benefits of a shared environment while preventing cache attacks.

Acknowledgments

We would like to extend our sincere gratitude to Shaqra University for their unwavering support throughout the research and preparation of this publication. The resources and academic environment provided by the university have played an integral role in shaping the outcome of this work. We are thankful for the opportunity to contribute to the scholarly community, and we recognize the invaluable contribution of Shaqra University in making this endeavor possible.

Authors contributions

Not applicable.

Funding

Not applicable.

Competing interests

Not applicable.

Informed consent

Obtained.

Ethics approval

The Publication Ethics Committee of the Canadian Center of Science and Education.

The journal's policies adhere to the Core Practices established by the Committee on Publication Ethics (COPE).

Provenance and peer review

Not commissioned; externally double-blind peer reviewed.

Data availability statement

The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.

Data sharing statement

No additional data are available.

References

Albalawi, A., Vassilakis, V. G., & Calinescu, R. (2022). Protecting Shared Virtualized Environments against Cache Side-channel Attacks. In *ICISSP* (pp. 507-514). https://doi.org/10.5220/0010897800003120

Albalawi, A., Vassilakis, V., & Calinescu, R. (2021). Memory deduplication as a protective factor in virtualized systems. In *Applied Cryptography and Network Security Workshops: ACNS 2021 Satellite Workshops, AIBlock, AIHWS, AIoTS, CIMSS, Cloud S&P, SCI, SecMT, and SiMLA, Kamakura, Japan, June 21–24, 2021, Proceedings* (pp. 301-317). Springer International Publishing. https://doi.org/10.1007/978-3-030-81645-2_17

Albalawi, A., Vassilakis, V., & Calinescu, R. (2022, April). Side-channel Attacks and Countermeasures in Cloud Services and Infrastructures. In NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium (pp. 1-4). IEEE. https://doi.org/10.1109/NOMS54207.2022.9789783

AlJahdali, H., Albatli, A., Garraghan, P., Townend, P., Lau, L., & Xu, J. (2014, April). Multi-tenancy in cloud computing. In *2014 IEEE 8th international symposium on service oriented system engineering* (pp. 344-351). IEEE. https://doi.org/10.1109/SOSE.2014.50

Aljahdali, H., Townend, P., & Xu, J. (2013, March). Enhancing multi-tenancy security in the cloud IaaS model over public deployment. In *2013 IEEE seventh international symposium on service-oriented system engineering* (pp. 385-390). IEEE. https://doi.org/10.1109/SOSE.2013.50

Anwar, S., Inayat, Z., Zolkipli, M. F., Zain, J. M., Gani, A., Anuar, N. B., ... Chang, V. (2017). Cross-VM cache-based side channel attacks and proposed prevention mechanisms: A survey. *Journal of Network and Computer Applications*, *93*, 259-279. https://doi.org/10.1016/j.jnca.2017.06.001

Atici, A. C., Yilmaz, C., & Savaş, E. (2018). Cache-timing attacks without a profiling phase. *Turkish Journal of Electrical Engineering and Computer Sciences*, *26*(4), 1953-1966. https://doi.org/10.3906/elk-1711-359

Bernstein, D. J. (2005). *Cache-timing attacks on AES.*

Garrison, G., Kim, S., & Wakefield, R. L. (2012). Success factors for deploying cloud computing. *Communications of the ACM*, *55*(9), 62-68. https://doi.org/10.1145/2330667.2330685

Gullasch, D., Bangerter, E., & Krenn, S. (2011, May). Cache games--bringing access-based cache attacks on AES to practice. In *2011 IEEE Symposium on Security and Privacy* (pp. 490-505). IEEE. https://doi.org/10.1109/SP.2011.22

Han, Y., Chan, J., Alpcan, T., & Leckie, C. (2014, June). Virtual machine allocation policies against co-resident attacks in cloud computing. In *2014 IEEE International Conference on Communications (ICC)* (pp. 786-792). IEEE. https://doi.org/10.1109/ICC.2014.6883415

Han, Y., Chan, J., Alpcan, T., & Leckie, C. (2015). Using virtual machine allocation policies to defend against co-resident attacks in cloud computing. *IEEE Transactions on Dependable and Secure Computing*, *14*(1), 95-108. https://doi.org/10.1109/TDSC.2015.2429132

Irazoqui, G., Inci, M. S., Eisenbarth, T., & Sunar, B. (2014). Wait a minute! A fast, Cross-VM attack on AES. In *Research in Attacks, Intrusions and Defenses: 17th International Symposium, RAID 2014, Gothenburg, Sweden, September 17-19, 2014. Proceedings 17* (pp. 299-319). Springer International Publishing. https://doi.org/10.1007/978-3-319-11379-1_15

Irazoqui, G., Inci, M. S., Eisenbarth, T., & Sunar, B. (2014, December). Fine grain cross-vm attacks on xen and vmware. In *2014 IEEE Fourth International Conference on Big Data and Cloud Computing* (pp. 737-744). IEEE. https://doi.org/10.1109/BDCloud.2014.102

Jayasinghe, D., Fernando, J., Herath, R., & Ragel, R. (2010, December). Remote cache timing attack on advanced encryption standard and countermeasures. In *2010 Fifth International Conference on Information and Automation for Sustainability* (pp. 177-182). IEEE.

Keller, E., Szefer, J., Rexford, J., & Lee, R. B. (2010, June). Nohype: virtualized cloud infrastructure without the virtualization. In *Proceedings of the 37th annual international symposium on Computer architecture* (pp. 350-361). Jasti, A., Shah, P., Nagaraj, R., & Pendse, R. (2010, October). Security in multi-tenancy cloud. In *44th Annual 2010 IEEE International Carnahan Conference on Security Technology* (pp. 35-41). IEEE.

Lindemann, J., & Fischer, M. (2018, April). A memory-deduplication side-channel attack to detect applications in co-resident virtual machines. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing* (pp. 183-192). https://doi.org/10.1145/3167132.3167151

Liu, F., Yarom, Y., Ge, Q., Heiser, G., & Lee, R. B. (2015, May). Last-level cache side-channel attacks are practical. In *2015 IEEE symposium on security and privacy* (pp. 605-622). IEEE. https://doi.org/10.1109/SP.2015.43

Malhotra, L., Agarwal, D., & Jaiswal, A. (2014). Virtualization in cloud computing. *J. Inform. Tech. Softw. Eng*, *4*(2), 1-3.

Ren, K., Wang, C., & Wang, Q. (2012). Security challenges for the public cloud. *IEEE Internet computing*, *16*(1), 69-73. https://doi.org/10.1109/MIC.2012.14

Saxena, S., Sanyal, G., Srivastava, S., & Amin, R. (2017). Preventing from cross-VM side-channel attack using new replacement method. *Wireless Personal Communications*, *97*, 4827-4854. https://doi.org/10.1007/s11277-017-4753-7

Shahzad, A., & Litchfield, A. (2015). Virtualization technology: Cross-VM cache side channel attacks make it vulnerable.

Shi, J., Song, X., Chen, H., & Zang, B. (2011, June). Limiting cache-based side-channel in multi-tenant cloud using dynamic page coloring. In *2011 IEEE/IFIP 41st International Conference on Dependable Systems and Networks Workshops (DSN-W)* (pp. 194-199). IEEE. https://doi.org/10.1109/DSNW.2011.5958812

Suzaki, K., Iijima, K., Yagi, T., & Artho, C. (2011, April). Memory deduplication as a threat to the guest OS. In *Proceedings of the Fourth European Workshop on System Security* (pp. 1-6). https://doi.org/10.1145/1972551.1972552

Takabi, H., Joshi, J. B., & Ahn, G. J. (2010). Security and privacy challenges in cloud computing environments. *IEEE Security & Privacy*, *8*(6), 24-31. https://doi.org/10.1109/MSP.2010.186

Varadarajan, V., Zhang, Y., Ristenpart, T., & Swift, M. (2015). A Placement Vulnerability Study in {Multi-Tenant} Public Clouds. In *24th USENIX Security Symposium (USENIX Security 15)* (pp. 913-928).

Vattikonda, B. C., Das, S., & Shacham, H. (2011, October). Eliminating fine grained timers in Xen. In *Proceedings of the 3rd ACM workshop on Cloud computing security workshop* (pp. 41-46).

https://doi.org/10.1145/2046660.2046671

Wu, J., Ding, L., Lin, Y., Min-Allah, N., & Wang, Y. (2012, June). Xenpump: a new method to mitigate timing channel in cloud computing. In *2012 IEEE Fifth International Conference on Cloud Computing* (pp. 678-685). IEEE. https://doi.org/10.1109/CLOUD.2012.28

Xiao, J., Xu, Z., Huang, H., & Wang, H. (2013, June). Security implications of memory deduplication in a virtualized environment. In *2013 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)* (pp. 1-12). IEEE. https://doi.org/10.1109/DSN.2013.6575349

Xiao, Z., & Xiao, Y. (2012). Security and privacy in cloud computing. *IEEE communications surveys & tutorials*, *15*(2), 843-859. https://doi.org/10.1109/SURV.2012.060912.00182

Xu, Z., Wang, H., & Wu, Z. (2015). A measurement study on co-residence threat inside the cloud. In *24th USENIX Security Symposium (USENIX Security 15)* (pp. 929-944).

Yan, L., Guo, Y., Chen, X., & Mei, H. (2015, November). A study on power side channels on mobile devices. In *Proceedings of the 7th Asia-Pacific Symposium on Internetware* (pp. 30-38). https://doi.org/10.1145/2875913.2875934

Yarom, Y., & Benger, N. (2014). Recovering OpenSSL ECDSA nonces using the FLUSH+ RELOAD cache side-channel attack. *Cryptology ePrint Archive*.

Yarom, Y., & Falkner, K. (2014). {FLUSH+ RELOAD}: A high resolution, low noise, l3 cache {Side-Channel} attack. In *23rd USENIX security symposium (USENIX security 14)* (pp. 719-732).

Yarom, Y., Ge, Q., Liu, F., Lee, R. B., & Heiser, G. (2015). Mapping the Intel last-level cache. *Cryptology ePrint Archive*.

Zhang, Y., & Reiter, M. K. (2013, November). Düppel: Retrofitting commodity operating systems to mitigate cache side channels in the cloud. In *Proceedings of the 2013 ACM SIGSA*. https://doi.org/10.1145/2508859.2516741

Zhang, Y., Juels, A., Oprea, A., & Reiter, M. K. (2011, May). Homealone: Co-residency detection in the cloud via side-channel analysis. In *2011 IEEE symposium on security and privacy* (pp. 313-328). IEEE. https://doi.org/10.1109/SP.2011.31