ISSN (ONLINE): 2395-695X
ISSN (PRINT): 2395-695X

International Journal of Advanced Research in Basic Engineering Sciences and Technology (IJARBEST)
Volume 2, Special Issue 19, October 2016

Cloud Scheduling of Security Implications in Data Mining

MRS. S. VASANTHI[1], MS. R. VAISHALI[2]

Assistant Professor, Dept. of Computer Science., P.S.G College of Arts and Science, India[1]
vasanthinandhu@gmail.com

Research Scholar, Dept. of Computer Science, P.S.G College of Arts and Science, India[2]
vaishalirajapsg@gmail.com

Abstract — Cloud computing has revolutionized the way computing and software services are delivered to the clients on demand. It offers users the ability to connect to computing resources and access IT managed services with a previously unknown level of ease. Due to this greater level of flexibility, the cloud has become the breeding ground of a new generation of products and services. Cloud providers host an increasing number of popular applications, on the premise of resource flexibility and cost efficiency. Most of these systems expose virtualized resources of different types and sizes. As instances share the same physical host to increase utilization, they contend on hardware resources, e.g., last-level cache, making them vulnerable to side-channel attacks from co-scheduled applications. In this work we show that using data mining techniques can help an adversarial user of the cloud determine the nature and characteristics of co-scheduled applications and negatively impact their performance through targeted contention injections. We design Bolt, a simple runtime that extracts the sensitivity of co-scheduled applications to various types of interference and uses this signal to determine the type of these applications by applying a set of data mining techniques. We validate the accuracy of Bolt on a 39-server cluster. Bolt correctly identifies the type and characteristics of 81% out of 108 victim applications, and constructs specialized contention signals that degrade their performance. We also use Bolt to find the most commonly-run applications on EC2. We hope that underlining such security vulnerabilities in modern cloud facilities will encourage cloud providers to introduce stronger resource isolation primitives in their systems. This is a big concern for many clients of cloud. In this paper, we first identify the data mining based privacy risks on cloud data and propose a distributed architecture to eliminate the risks.

Index Terms— Super (very large) computers, Security and privacy protection, Scheduling and task partitioning, Application studies resulting in better multiple-processor systems.

I. INTRODUCTION

Cloud computing hosts an increasing number of applications in private and public clouds. It offers two main premises to end users and datacenter operators: flexibility and cost efficiency. Public clouds expose resources as virtual machines and more recently containers of different types and sizes. Public cloud schedulers typically collocate several VMs on the same physical machine to increase system utilization. As VMs get co-scheduled they share certain hardware resources, such as the last level cache, or the network switch. This hides security and privacy pitfalls, as resource isolation is not strictly enforced. For example, while memory capacity is partitioned, contention in the last level cache can still leak information about a co-scheduled program. There has

been significant related work on side-channel attacks [10], [17], VM detection [14], [21], distributed denial-of-service attacks (DDoS) [1], [9], [17] and data leakage vulnerabilities [10], [20] in cloud providers. In this paper we present Bolt, a practical system that leverages data mining techniques to quickly determine the type and characteristics of any applications scheduled on the same machine in a public cloud. We evaluate Bolt on a local cluster and demonstrate that it correctly detects the type of co-scheduled applications for 81% out of 108 diverse workloads. Additionally, it is able to degrade the performance of those workloads, by 32% on average and more than 60% in some cases, by injecting interference in the resources a victim application is sensitive to. It does so without saturating compute and/or memory utilization, which could trigger migration and auto scaling in cloud providers that offer such solutions to remedy performance unpredictability.

II. DATA MINING ON CLOUD

Data mining is one of the fastest growing fields in computer industry [14] that deals with discovering patterns from large data sets [22]. It is a part of knowledge discovery process and is used to extract human understandable information [8]. Mining is preferably used for a large amount of data [25][26] and related algorithms often require large data sets to create quality models [1]. The relationship between data mining and cloud is worth to discuss. Cloud providers use data mining to provide clients a better service [27]. According to the survey done by Rexer Analytics, 7% data miners use cloud to analyze data [16]. As cloud is a massive source of centralized data, data mining gives attackers a great advantage in extracting valuable information and thus violating clients' data privacy. A. The Importance of Client Privacy Client privacy is a tentative issue as all clients do not have the same demands regarding privacy. Some are satisfied with the current policy while others are quite concerned about their privacy. The proposed system is designed preferably for the clients belonging to the second category for whom privacy is a great concern. These clients may not afford the luxury of maintaining private storage while they are interested in spending a little more money on maintaining their privacy on the cloud. If the client itself is a company providing services to others, the violation of privacy of the client affects the privacy of its customers. Sometimes leaking information regarding a particular company leads to a national catastrophe. The events of TIA (Total Information Awareness) gathering financial, educational, health and other information about people in 2002 and NSA obtaining customer records from phone companies and analyzing them to identify potential terrorists in May 2006 can be considered as examples.

A. The Importance of Client Privacy
B. Data Mining: A Potential Threat to Privacy.

III. ELIMINATING CLOUD-MINES

In this section, we first discuss the data mining based privacy threats to the single provider cloud architecture. Then give an overview of the state-of-the-art distributed approach to prevent data mining based privacy attacks on the cloud.

A. Existing System Threats The current cloud storage system is a vulnerable one because data remain under a single cloud provider. This can lead to data loss in case of events like network outage, the cloud provider going out of business, malware attack etc. The current system also gives a great advantage to the attackers as they have fixed targets in the forms of cloud providers. If an attacker chooses to attack a specific client, then he can aim at a fixed cloud provider, try to have access to the client's data and analyze it. This eases the job of the attackers. As long as the entire data belonging to a client remain under a single cloud provider, both inside and outside attackers get the benefit of using data mining to a great extent. Inside attackers in this context refers to malicious employees at a cloud provider. Data mining models often require large number of observations and single provider architecture is a great advantage suiting the case as all the samples remain under the provider.

Thus single provider architecture is the biggest security threat concerning data mining on cloud.

B. A Distributed Approach to the Cloud To eliminate the disadvantage of storing all data of a client to the same provider, data can be split into chunks and distributed among multiple cloud providers. The advantage of this distributed system can be visualized when an attacker chooses a specific client but the distribution of data obliges him to target multiple cloud providers, making his job increasingly difficult. Mining based attacks on cloud involves attackers of two categories: malicious employees inside provider and outside attackers. Distribution of data chunks among multiple providers restricts a cloud provider from accessing all chunks of a client.

IV. SYSTEM ARCHITECTURE

In this section we discuss our proposed system architecture that prevents data mining based privacy attacks on the cloud. Our system consists of two major components: Cloud Data Distributor and Cloud Providers. The Cloud Data Distributor receives data in the form of files from clients, splits each file into chunks and distributes these chunks among cloud providers. Cloud Providers store chunks and responds to chunk requests by providing the chunks.

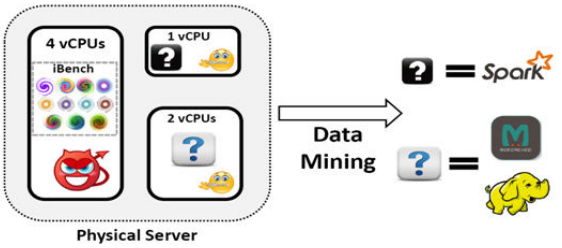
V. VM placement detection:

Risten part et al. [14] show how leveraging the IP naming conventions of machines in cloud providers can help an adversarial user pinpoint where a victim VM is residing in a large-scale cluster. Subsequently, the adversarial user can launch VMs until one is co-scheduled on the same physical machine as the target VM. Home Alone [21] tracks the utilization of the L2 cache during periods of low traffic from "friendly" VMs to detect whether the physical machine is shared across VMs.

A. DDoS attacks:

Distributed Denial of Service attacks [8], [12], [15] in the cloud have increased significantly in number and impact over the past few years. This has generated a lot of interest in detection and prevention techniques [13]. Gupta et al. [9] outline the characteristics of cloud facilities that make DDoS attacks more likely, discuss the challenges that current DDoS prevention schemes face, and propose a scheme based on VM profiling to detect network DDoS attacks. Bakshi et al. [1] develop a system that detects abnormally high network traffic in cloud machines that would signal an upcoming DDoS attack. Finally, Darwish et al. [3] explore different DDoS attack types in cloud resources, and propose practical defense mechanisms.

Fig. 1. Overview of the system's operation.



The adversarial VM uses iBench to measure the pressure the two victim VMs place on shared resources. Bolt then uses data mining to determine the type and characteristics of applications running in the victim VMs.

B. Side-channel attacks in public clouds:

Such systems attempt to extract information about co-scheduled applications, including confidential data, such as private keys [2], [11], [18], [19]. Zhang et al. [20] describe a system that launches side-channel attacks in a virtualized environment. The system overcomes three main challenges: the frequent re-scheduling of VMs by a hypervisor or cluster scheduler, the noise in shared resource usage and the implications introduced by core migrations. They demonstrate that the system can extract an ElGamal decryption key from a victim VM. Wang et al. [17] specifically target intrusion detection in cloud settings, while Liu et al. [10] design a scheduling system that protects against covert channels in resources such as the memory bus in a cloud environment. The system controls the overlapping execution of different VMs and injects noise on the

memory bus to prevent the extraction of confidential information by an adversarial user.

VI. BOLT

Threat Model We consider an IaaS provider that operates a public cloud. Multiple VMs can be co-scheduled on the same physical server. Each VM has no control over where it is placed in the cluster, and has no a priori information on any other VMs scheduled on the same host. We assume that the cloud provider is neutral with respect to VM detection by an adversarial VM, i.e., it does not employ any additional resource isolation techniques than what is available by default to hinder such attacks.

Adversarial VM: An adversarial VM has the goal of determining the nature and characteristics of jobs co-scheduled on the same physical host, and negatively impacting their performance.

Friendly VM: This is a VM scheduled on a physical host that runs one or more applications. Friendly VMs do not attempt to determine the existence and characteristics of other co-scheduled VMs. They also do not employ any schemes to prevent detection.

Application Detection: The operation of Bolt at a high level is shown in Figure 1. The adversarial VM runs a set of micro benchmarks that have tunable intensity and put progressively more pressure in a specific shared resource each [4]. Once a micro benchmark starts running it will increase its intensity until it finds pressure from co-scheduled workloads, i.e., until its performance is lower than its expected value, when it is running in isolation. The intensity of the micro benchmark at that point is the pressure the co-scheduled applications put in shared resource i and is denoted by ci, where i ∈ [1, N], N = 10. Large ci values mean that the co-scheduled applications put a lot of pressure in resource i. The same operation is performed for 2-3 micro benchmarks, and requires 5-10 seconds.

There are cases where a victim VM may not have a clear application type it resembles, or may change behavior during its execution. To address this issue Bolt repeats the classification periodically (every 5 minutes in our experiments), until the confidence scores of similarity converge. For the majority of examined applications, convergence occurs after 2 iterations.

Multiple co-scheduled jobs: A challenge with the previous approach is the case where more than one victim VMs are coscheduled on the same physical server. The adversarial VM has no access to the hypervisor, and hence only sees the aggregate interference from all co-scheduled applications. To decouple different jobs sharing a physical server, Bolt examines the combinations of interference profiles of different application types, and compares them against the aggregate interference observed by the adversarial VM. For example, if we have only seen three types of applications, A1, A2 and A3, with U vectors: [u11, u12, ..., u1N], [u21, u22, ..., u2N] and, [u31, u32, ..., u3N] and the victim VM has interference profile: [x1, x2, ..., xN], Bolt will examine all combinations of the three application types, and return the one that most closely resembles the victim VM. We plan to investigate solutions that scale better and detect nonlinear interference relations in future work.

Internal Denial of Service (DoS) Attack Once an adversarial VM knows the type of applications that coexist on the same physical host, it tries to degrade their performance through a targeted interference injection. Constructing an interference signal that will affect the victim VM relies on the knowledge of the resources the victim VM is sensitive to. This corresponds to the level of interference the application can tolerate in the different shared resources. The injected contention is then simply an interference signal that slightly exceeds that level. For example, in the case of memcached, the detection scheme determines that the victim VM puts a lot of pressure in the LLC, followed by lower pressure in the CPU, memory and network subsystems. This means that the application tolerates a lot of contention

in storage, followed by lower tolerance in the memory, network and CPU. Note that tolerated interference is not simply the complement of the generated contention of a workload.

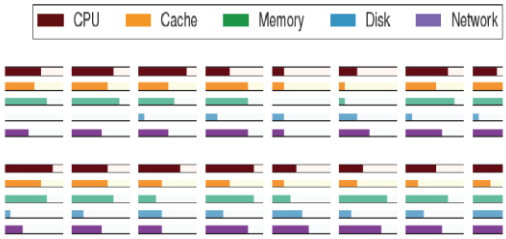


Fig. 2. Per-server resource pressure in the controlled experiment.

TABLE 1
Estimation accuracy for the controlled experiment.

Applications	correct app type	Correct 1st incorrect estimation	2nd incorrect estimation
Aggregate	81%	- -	- -
memcached	76%	Spark	speccpu2006
Hadoop	84%	Cassandra	Spark
Spark	85%	speccpu2006	Memcached
Cassandra	87%	Hadoop	Spark
speccpu2006	81%	Spark	Memcached

Injecting this contention signal degrades the performance of the victim VM with high probability. Furthermore, because the injection does not simply saturate shared resources, e.g., by scanning the entire cache/memory or saturating the CPU, it will not trigger the migration/auto scaling schemes that cloud providers have in place to avoid machine oversubscription when application load increases. As a result, performance degrades without the user being able to ameliorate the problem, except by terminating and restarting the VM.

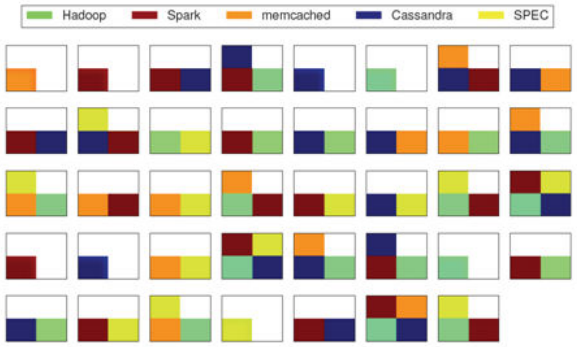


Fig. 3. Per-server application mapping in the controlled experiment.

In the following section we evaluate the accuracy of Bolt in detecting the type and characteristics of victim VMs, and the performance degradation it can induce to them.

VII. EVALUATION

Controlled Experiment

We first perform a controlled experiment, in which all victim applications are known in advance. This allows us to validate the accuracy of Bolt in detecting application types and degrading their performance through targeted contention injection.

We use a cluster of 39 dedicated machines, with 8 2-way hyperthreaded physical cores and 64GB of RAM each. In each server we launch a 2 physical core (of 2 vCPU each) VM running Ubuntu 14.04 as the adversarial VM. The remainder of the machine is allocated to one or more victim VMs, running over Ubuntu 14.04 or Debian 8.0. Victim applications are scheduled using the Quasar cluster manager [6]. Quasar minimizes interference between co-scheduled workloads by only collocating applications that do not contend in the same resources. This helps quantify the impact of the adversarial VM on performance, and decouple it from any existing interference between co-scheduled applications.

We will explore how different schedulers affect the detection accuracy as part of future work. The adversarial applications have no a priori information on the number and type of co-scheduled applications. We schedule a total of 108 applications, from five classes (Table 1). Within each

class there are several individual workloads, including the Mahout library for Hadoop and machine learning applications for Spark. Each machine has at least one victim VM, and at most four victim VMs. Each VM can use one or more physical cores. When exceeding four applications the interference between the victim applications alone does not permit them to meet their QoS constraints.

Figure 3 shows the type of applications running in each physical machine, and Figure 2 shows the level of interference they induce in each of the main shared resources (from 0 to 99%), as detected by Bolt. Note that different mixes of applications produce different interference profiles in Figure 2. Based on this signal Bolt tries to identify the type of co-scheduled applications. Table 1 shows Bolt's detection accuracy, per application type, and aggregate. Bolt correctly identifies the majority of jobs, 81%, and for certain application types like databases and analytics, the accuracy exceeds 85%. 86% of correctly identified applications required one profiling run for identification. For an extra 9%, a second profiling run was necessary. We also show the most frequent misconceptions for each job type. Most incorrectly-identified applications occur in servers hosting more than three workloads.

If interference translates to CPU or memory saturation, there is a high probability that at least one of the co-scheduled VMs will be migrated to a new physical machine, for cloud providers that support live migration, e.g., Google Compute Engine, or scaled out to additional machines. Figure 5 compares the tail latency and CPU utilization Bolt causes to that of a naive system that simply saturates the CPU through a compute-intensive kernel. We focus on a single victim VM running memcached. Performance degradation is similar for both systems as time progresses. On the other hand utilization is quite different, with Bolt keeping CPU utilization fairly low, hence not triggering migration or auto scaling, while the naive scheme quickly saturates the core.

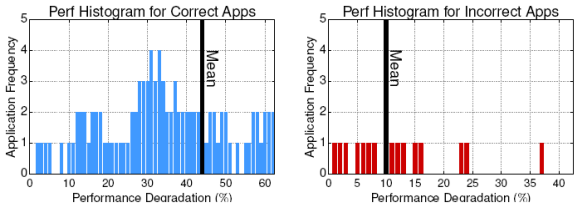


Fig. 4. Performance degradation in the controlled experiment.

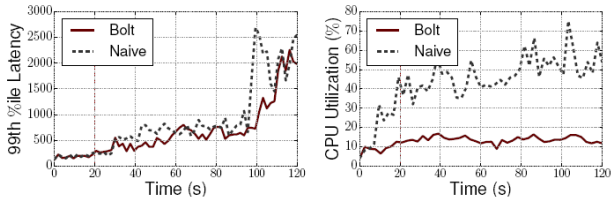


Fig. 5. Comparison of latency and utilization against a naïve scheme.

Bolt in the Wild :

We now use Bolt in a real (non-controlled) setting on a large EC2 cluster to detect the type of applications submitted by external users. This experiment is limited to detecting, but not negatively impacting the performance of the co-scheduled applications, to prevent service interruptions for other users. We request 400 4 vCPU on-demand instances, and verify that they are not on the same physical machine [16]. We inject iBench to determine the interference profile of the co-scheduled applications and use the classification engine to map interference profiles to specific application types. Table 2 shows how many co-scheduled applications were found across the 400 physical machines. For a large fraction of physical machines the adversarial VM is the only application occupying the server, despite the fact that it only requires 4 vCPU. Most of the remaining machines have 1-2 co-scheduled applications, while a small number of servers host more than 3 victim applications. We keep requesting 4 vCPU instances until we have 400 instances with at least one co-scheduled VM. Figure 6 shows the probability distribution function (PDF) of detected application types. Interestingly there is a very small number of application types that dominates the utilization of the examined cluster. As expected, these applications primarily include analytics, webservers and databases. Apart from the main five applications, there is a long tail of less common applications,

which were encountered less than 10 times across all VMs.

VIII. FUTURE WORK

We have presented Bolt, a system that highlights the performance implications of using data mining in cloud settings. Bolt projects the contention an adversarial VM experiences in shared resources from co-scheduled VMs, against a dataset of application profiles to determine the type and characteristics of co-scheduled VMs. Furthermore, it negatively impacts victim applications, through targeted contention injections. Such attacks are facilitated by the lack of strict resource isolation guarantees between workloads sharing a physical machine. Introducing isolation techniques, primarily in the memory hierarchy which is a strong indicator of the type of co-scheduled applications could alleviate some of these security concerns. Anecdote: By verifying the timestamps, instance configuration and zone, and benchmark order, we were able to identify one victim VM running several SPECCPU2006 benchmarks which belonged to a student group at Stanford.

TABLE 2

Breakdown of the number of co-scheduled VMs found by Bolt in the EC2 experiment.

	no VM	1 VM	2 VMs	>2 VMs
% of machines	41%	32%	16%	11%

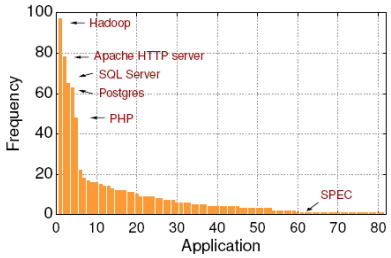


Fig. 6. Probability distribution function of application types on EC2.

Although the proposed system provides an effective way to protect privacy from mining based attacks, it introduces performance overhead when client needs to access all data frequently, e.g. client needs to perform a global data analysis on all data. The analysis may have to access data from multiple locations, with a degraded performance. In future, we look forward to improve our system by reducing such overhead.

x. CONCLUSION

Ensuring security of cloud data is still a challenging problem. Cloud service providers as well as other third parties use different data mining techniques to acquire valuable information from user data hosted on the cloud. In this paper, we have discussed the impact of data mining on cloud and have proposed a distributed structure to eliminate mining based privacy threat on cloud data. Our approach combining categorization, fragmentation and distribution, prevents data mining by maintaining privacy levels in cloud providers.

REFERENCES

[1] A. Bakshi and B. Yogesh, “Securing cloud from ddos attacks using intrusion detection system in virtual machine,” in ICCSN. 2010.

[2] N. Benger, J. van de Pol, N. P. Smart, and Y. Yarom, “Ooh aah... just a little bit. : A small amount of side channel can go a long way,” in Proc. of CHES. 2014.

[3] M. Darwish, A. Ouda, and L. Capretz, “Cloud-based ddos attacks and defenses,” in Proc. of i-Society. Toronto, ON, 2013.

[4] C. Delimitrou and C. Kozyrakis, “iBench: Quantifying Interference for Datacenter Workloads,” in Proc. of IISWC. 2013.

[5] C. Delimitrou and C. Kozyrakis, “Paragon: QoS-Aware Scheduling for Heterogeneous Datacenters,” in Proc. of ASPLOS. 2013.

[6] C. Delimitrou and C. Kozyrakis, “Quasar: Resource-Efficient and QoS-Aware Cluster Management,” in Proc. of ASPLOS. 2014.
[7] C. Delimitrou and C. Kozyrakis, “The Netflix Challenge: Datacenter Edition.” IEEE Computer Society, July 2013.
[8] S. Gupta and P. Kumar, “Vm profile based optimized network attack pattern detection scheme for ddos attacks in cloud,” in Proc. of SSCC. 2013.
[9] F. Liu, L. Ren, and H. Bai, “Mitigating cross-vm side channel attack on multiple tenants cloud platform,” in Journal of Computers, 2014.
[10] F. Liu, Y. Yarom, Q. Ge, G. Heiser, and R. B. Lee, “Last-level cache side-channel attacks are practical,” in Proc. of IEEE S&P. 2015.
[11] J. Mirkovic and P. Reiher, “A taxonomy of ddos attack and ddos defense mechanisms,” SIGCOMM CCR, Apr. 2004.
[12] T. Peng, C. Leckie, and K. Ramamohanarao, “Survey of networkbased defense mechanisms countering the dos and ddos problems,” ACM Comput. Surv., vol. 39, no. 1, Apr. 2007.
[13] 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 Proc. of CCS. Chicago, IL, 2009.
[14] V. Varadarajan, T. Kooburat, and et al., “Resource-freeing attacks: Improve your cloud performance (at your neighbor’s expense),” in Proc. of CCS. 2012.
[15] V. Varadarajan, Y. Zhai, and et al., “Scheduler-based defenses against cross-vm side-channels,” in Proceedings of the 23rd Usenix Security Symposium. San Diego, CA, 2014.
[16] H. Wang, H. Zhou, and C. Wang, “Virtual machine-based intrusion detection system framework in cloud computing environment,” in Journal of Computers, October 2012.
[17] Y. Zhang and M. K. Reiter, “Duppel: retrofitting commodity operating systems to mitigate cache side channels in the cloud,” in Proc. of CCS. 2013.
[18] Y. Zhang, A. Juels, and et al., “Cross-vm side channels and their use to extract private keys,” in Proc. of CCS. 2012.
[19] Y. Zhang, A. Juels, A. Oprea, and M. K. Reiter, “Homealone: Coresidency detection in the cloud via side-channel analysis,” in Proc. of the IEEE Symposium on Security and Privacy. 2011.
[20] M. J. Shaw, C. Subramaniam, G. W. Tan, and M. E. Welge. Knowledge management and data mining for marketing. Decis. Support Syst., 31(1):127–137, 2001.