International Journal of Scientific & Engineering Research, Volume 6, Issue 4, April-2015
ISSN 2229-5518

A Security Analysis of Amazon's Elastic Compute Cloud Service

Mr. N NIRANJAN RAO, Mr SREERAG P.S, Mr. SHRIHARI M.R

Abstract— Cloud services such as Amazon's Elastic Compute Cloud and IBM's SmartCloud are quickly changing the way organiza- tions are dealing with IT infrastructures and are providing online services. Today, if an organization needs computing power, it can simply buy it online by instantiating a virtual server image on the cloud. Servers can be quickly launched and shut down via application programming interfaces, of- fering the user a greater flexibility compared to traditional server rooms. A popular approach in cloud-based services is to allow users to create and share virtual images with other users. In addition to these user-shared images, the cloud providers also often provide virtual images that have been pre-configured with popular software such as open source databases and web servers.This paper explores the general security risks associated with using virtual server images from the public catalogs of cloud service providers. In particular, we investigate in detail the security problems of public images that are avail- able on the Amazon EC2 service. We describe the design and implementation of an automated system that we used to instantiate and analyze the security of public AMIs on the Amazon EC2 platform, and provide detailed descriptions of the security tests that we performed on each image. Our findings demonstrate that both the users and the providers of public AMIs may be vulnerable to security risks such as unauthorized access, malware infections, and loss of sensi- tive information. The Amazon Web Services Security Team has acknowledged our findings, and has already taken steps to properly address all the security risks we present in this paper.

Index Terms— Cloud Computing, Elastic Compute Cloud Service, Security, AMI, Amazon.

1 INTRODUCTION

Cloud computing has changed the view on IT as a pre-paid asset to a pay-as-you-go service. Several companies such as Amazon Elastic Compute Cloud [7] (EC2), Rackspace[9], IBM SmartCloud [12], Joyent Smart Data Center [14] or Ter- remark vCloud [10] are offering access to virtualized servers in their data centers on an hourly basis. Servers can be quickly launched and shut down via application program- ming interfaces, offering the user a greater flexibility com- pared to traditional server rooms. This paradigm shift is changing the existing IT infrastructures of organizations, al- lowing smaller companies that cannot afford a large infrastructure to create and maintain online services with ease.

A popular approach in cloud-based services is to allow users to create and share virtual images with other users. For example, a user who has created a legacy Linux De- bian Sarge image may decide to make this image public so that other users can easily reuse it. In addition to user- shared images, the cloud service provider may also provide customized public images based on common needs of their customers (e.g., an Ubuntu web server image that has been pre-configured with MySQL, PHP and an Apache). This allows the customers to simply instantiate and startnew servers, without the hassle of installing new software them- selves.Unfortunately, while the trust model between the cloud user and the cloud provider is well-defined (i.e., the user can assume that cloud providers such as Amazon and Mi- crosoft are not malicious), the trust relationship between the provider of the virtual image and the cloud user is not as clear. In this paper, we explore the general security risks asso- ciated with the use of virtual server images from the public catalogs of a cloud service provider. In particular, we fo- cus our investigation to the security problems of the public images available on the Amazon EC2 service. Over several months, we instantiated and analyzed over five thousands Linux and Windows images provided by the Amazon cata- log, checking for a wide-range of security problems such as the prevalence of malware, the quantity of sensitive data left on such images, and the privacy risks of sharing an image on the cloud. In particular, we identified three main threats related, re- spectively, to: 1) secure the image against external attacks, secure the image against a malicious image provider Sanitize the image to prevent users from extracting and abusing private information left on the disk by the image provider.

INTERNATIONAL JOURNAL OF SCIENTIFIC & ENGINEERING RESEARCH, VOLUME 6, ISSUE 4, APRIL-2015
ISSN 2229-5518

2) For example, in our experiments we identified many images in which a user can use standard tools to un- delete files from the filesystem, and recover important doc- uments including credentials and private keys.

Although public cloud server images are highly useful for organizations, if users are not properly trained, the risk as- sociated with using these images can be quite high. The fact that these machines come pre-installed and pre-configured may communicate the wrong message, i.e., that they can provide an easy-to-use "shortcut" for users that do not have the skills to configure and setup a complex server. The reality is quite different, and this paper demonstrates that many different security considerations must be taken into account to make sure that a virtual image can be operated securely. During our study we had continuous contact with the Amazon Web Services Security Team. Even though Amazon is not responsible of what users put into their images, the team has been prompt in addressing the security risks identified and described in this paper. Meanwhile, it has pub- lished public bulletins and tutorials to train users on how to use Amazon Machine Images (AMIs) in a secure way [29, 28]. A more detailed description of the Amazon feedback is provided in Section 6. In summary, this paper makes the following contributions:

- We describe the design and implementation of an au- tomated system that is able to instantiate and analyze the security of public AMIs on the Amazon EC2 plat- form.
- We provide detailed descriptions of the security tests that we performed on public AMIs.
- We describe the results of our security tests that demon- strate that both the users and the providers of public AMIs may be vulnerable to security risks such as unau- thorized access, malware infections, and loss of sensi- tive information.
- The Amazon Web Services Security Team has acknowl- edged and taken steps to address the issues we have identified. We discuss the countermeasures that they have taken, and report on the information campaigns that they have started.

1 Overview of Amazon EC2

The Amazon Elastic Compute Cloud (EC2) is an Infrastructure- as-a-Service cloud provider where users can rent virtualized

Servers (called instances) on an hourly base. In particular, each user is allowed to run any pre-installed virtual machine image (called Amazon Machine Image, or AMI) on this ser- vice. To simplify the setup of a server, Amazon offers an online catalog where users can choose between a large num- ber of AMIs that come pre-installed with common services such as web servers, web applications, and databases. An AMI can be created from a live system, a virtual machine image, or another AMI by copying the file system contents to the Amazon Simple Storage Service (S3) in a process called bundling. Public images may be available for free, or may be associated with a product code that allows companies to bill an additional usage cost via the Amazon DevPay payment service. Thus, some of these public machines are provided by companies, some are freely shared by single individuals, and some are created by the Amazon team itself.

In order to start an image, the user has to select a resource configuration (differing in processing, memory, and IO per- formance), a set of credentials that will be used for login, a firewall configuration for inbound connections (called a se- curity group), and the region of the data center in which the machine will be started.

When an AMI is instantiated, its public DNS address is announced via the Amazon API, and the machine is made accessible via SSH on port 22 (Linux) or Remote Desktop on port 3389 (Windows). An important aspect of this cloud computing service is that the instance's maintenance is com- pletely under the responsibility of the user. That is, she is the one who can be held responsible for any content provided by the machine, and she is the one who has to assure its se- curity. This includes, for example, the usual administration tasks of maintaining the configuration in a secure state (i.e., applying patches for vulnerable software, choosing the right passwords, and firewall configuration), and only allowing se- cure, encrypted communication protocols.

2 AMI Testing Methodology

To conduct our security evaluation, we developed an auto- mated system to instantiate and test the Amazon's AMIs. The architecture of our system is highlighted in Fig. 1, and consists of three main

- *N Niranjan Rao is currently pursuing BE degree program in Computer science & engineering in Visvesvaraya Technological University, India, PH-08023419099. E-mail: niranjan.rao.cse@gmail.com*
- *Sreerag P.S is currently pursuing BE degree program in Computer Science& engineering in Visvesvaraya Technological University, India, PH-+918050609779. E-mail: sreerag.sa@gmail.com*

INTERNATIONAL JOURNAL OF SCIENTIFIC & ENGINEERING RESEARCH, VOLUME 6, ISSUE 4, APRIL-2015
ISSN 2229-5518

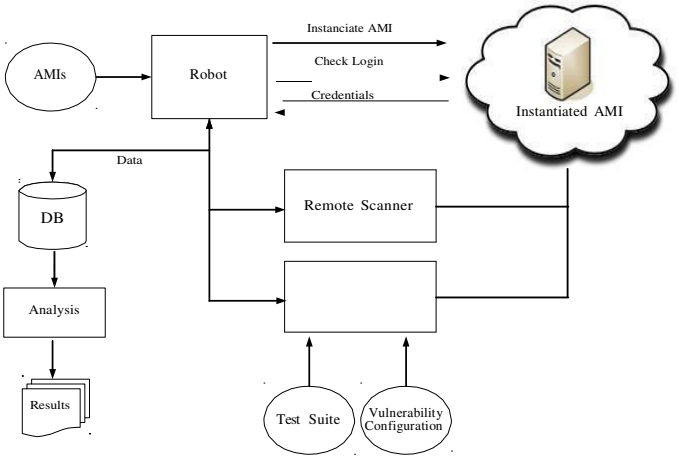

Figure 1: System Architecture

and bitnami for Linux). Despite these attempts, there are cases in which the robot may fail to retrieve the correct login information. This is the case, for example, for AMIs whose credentials are distributed only to the image provider's cus- tomers by companies that make business by renting AMIs. Hence, these type of images are outside the scope of our evaluation.

After an AMI has been successfully instantiated by the Robot, it is tested by two different scanners. The Remote Scanner collects the list of open ports[1] using the NMaptool [23], and downloads the index page of the installed web applica- tions. In Section 5, we explain how an attacker can use this information as a fingerprint to identify running images. The Local Scanner component is responsible for uploading and running a set of tests. The test suite to be executed is packaged together in a self-extracting archive, uploaded to the AMI, and run on the machine with administrative privileges. In addition, the Local Scanner also analyzes the system for known vulnerabilities using the Nessus tool [30]. For AMIs running Microsoft Windows, the scripting of au- tomated tasks is complicated by the limited remote adminis- tration functionalities offered by the Windows environment. In this case, we mounted the remote disk and transfered the data using the SMB/Netbios subsystem. We then used the psexec tool [27] to execute remote commands and invoke the tests.

The test suite uploaded by the Local Scanner includes 24 tests grouped in 4 categories: general, network, privacy, and security (for the complete list see Appendix A).

The general category contains tests that collect general information about the system (e.g. the Linux distribution name, or the Windows version), the list of running processes, the file-system status (e.g., the mounted partitions), the list of installed packages, and the list of loaded kernel mod- ules. In addition to these basic tests, the general category also contains scripts that save a copy of interesting data, such as emails (e.g., /var/mail), log files (e.g., /var/log and %USER\Local Settings), and installed web applications (e.g., /var/www and HKEY_LOCAL_MACHINE\SOFTWARE).

The network test suite focuses on network-related infor- mation, such as shared directories and the list of open sock- published the AMI. This includes, for example, unprotected private keys, application history files, shell history logs, and the content of the directory saved by the general test cases. Another important task of this test suite is to scan the

suspicious connections.

Finally, the security test suite consists of a number of well-known audit tools for Windows and Linux. Some of these tools look for the evidence of known rootkits, Tro-jans and backdoors (e.g. Chkrootkit, RootkitHunter and RootkitRevealer), while others specifically check for processes and sockets that have been hidden from the user (PsTools/PsList and unhide). In this phase, we also run the ClamAV antivirus software (see Section 4.2) to scan for the presence of known malware samples. These security tests also contain checks for credentials that have been left or forgotten on the system (e.g., database passwords, login passwords, and SSH public keys). As al- ready mentioned in an Amazon report published in June 2011 [15], these credentials could potentially be used as back- doors to allows attackers to log into running AMIs.

3 Results of the AMIs Analysis

Over a period of five months, between November 2010 to May 2011, we used our automated system to instantiate and analyze all Amazon images available in the Europe, Asia, US East, and US West data centers. In total, the cata- log of these data centers contained 8,448 Linux AMIs and 1,202 Windows AMIs. Note that we were successfully able to analyze in depth a total of 5,303 AMIs. In the remaining cases, a number of technical problems prevented our tool to successfully complete the analysis.

[1] Since Amazon does not allow external portscans of EC2 machines, we first established a virtual private network con- nection to the AMI through SSH, and then scanned the ma- chine through this tunnel.

INTERNATIONAL JOURNAL OF SCIENTIFIC & ENGINEERING RESEARCH, VOLUME 6, ISSUE 4, APRIL-2015
ISSN 2229-5518

Average #/AMI	Windows	Linux
Audit duration	77 min	21 min
Installed packages	–	416
Running Processes	32	54
Shares	3.9	0
Established sockets	2.75	2.52
Listening sockets	22	6
Users	3.8	24.8
Used disk space	1.07 GB	2.67 GB

Table 1: General Statistics

3.1 Software Vulnerabilities

The goal of this first phase of testing is to confirm the that the software running on each AMIs is often out of d and, therefore, must be immediately updated by the u after the image is instantiated.

For this purpose, we decided to run Nessus [30], an tomated vulnerability scanner, on each AMI under test. order to improve the accuracy of the results, our tes system provided Nessus with the image login credentials, that the tool was able to perform a more precise local sc
. In addition, to further reduce the false positives, the vul ability scanner was automatically configured to run only tests corresponding to the actual software installed on machine. Nessus classifies each vulnerability with a se-
ity level ranging from 0 to 3. Since we were not interes in analyzing each single vulnerability, but just in assesthe general security level of the software that was instal , we only considered vulnerabilities with the highest seve (e.g., critical vulnerabilities such as remote code executio .

We also looked at the most common vulnerabilities t affect Windows and Linux AMIs. These results are deta in Appendix B.

From our analysis, 98% of Windows AMIs and 58% Linux AMIs contain software with critical vulnerabili This observation was not typically restricted to a single plication but often involved multiple services: an averag 46 for Windows and 11 for Linux images (the overall
tribution is reported in Figure 2). On a broader scale, we observed that a large number of images come with software that is more than two years old. Our findings empirically demonstrate that renting and using an AMI without any adequate security assessment poses a real security risk for users. To further prove this point, in Section 4.2, we describe how one of the machines we were testing was probably com- promised by an Internet malware in the short time that we were running our experiments.

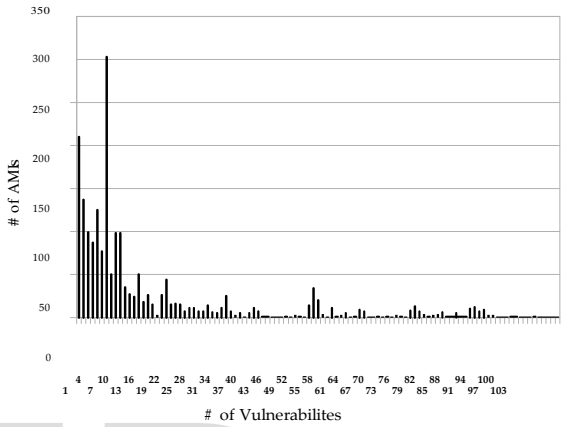

Figure 2: Distribution AMIs / Vulnerabilites (Win- dows and Linux)

3.2 Security Risks Malware

As part of our tests, we used ClamAV [8], an open
Windows installation, and less then a minute for a Linux one.In our malware analysis, we discovered two infected AMIs, both Windows-based. The first machine was infected with a Trojan-Spy malware (variant 50112). This trojan has a wide range of capabilities, including performing key logging, monitoring processes on the computer, and stealing data from files saved on the machine. By manually analyzing this machine, we found that it was hosting different types of suspicious content such as Trojan.Firepass, a tool to de- crypt and recover the passwords stored by Firefox. The sec- ond infected machine contained variant 173287 of the Trojan.Agent malware. This malware allows a malicious user to spy on the browsing habits of users.

- *Author name is currently pursuing masters degree program in electric power engineering in University, Country, PH-01123456789. E-mail: author_name@mail.com*
- *Co-Author name is currently pursuing masters degree program in electric power engineering in University, Country, PH-01123456789. E-mail: author_name@mail.com*
(*This information is optional; change it according to your need.*)

INTERNATIONAL JOURNAL OF SCIENTIFIC & ENGINEERING RESEARCH, VOLUME 6, ISSUE 4, APRIL-2015
ISSN 2229-5518

the evidence of some kind of backdoor, or the sign for a mal- ware infection. Outgoing connections that are more stealthy may also be used to gather information about the AMI's us- age, and collect IP target addresses that can then be used to attack the instance through another built-in backdoor.In our experiments, we observed several images that opened connections to various web applications within and outside of Amazon EC2. These connections were apparently check- ing for the availability of new versions of the installed soft- ware. Unfortunately, it is almost impossible to distinguish between a legitimate connection (e.g., a software update) and a connection that is used for malicious purposes.

Nevertheless, we noticed a number of suspicious connec- tions on several Linux images: The Linux operating system comes with a service called syslog [3] for recording various events generated by the system (e.g., the login and logout of users, the connection of hardware devices, or incoming requests toward the web server).Standard installations record these kinds of events in files
Usually stored under the /var/log directory and only users with administrative privileges are allowed to access the logs generated by the syslog service. In our tests, we discovered two AMIs in which the syslog daemon was configured to send the log messages to a remote host, out of the control of the user instantiating the image. It is clear that this setup constitutes a privacy breach, since confidential information, normally stored locally under a protected directory, were sent out to a third party machine.

Backdoors and Leftover Credentials

The primary mechanism to connect to a Linux machine re- motely is through the ssh service. When a user rents an
AMI, she is required to provide the public part of the her ssh key that it is then stored by Amazon in the autho- rized_keys in the home directory. The first problem with
this process is that a user who is malicious and does not remove her public key from the image before making it pub- lic could login into any running instance of the AMI. The existence of these kinds of potential backdoors is known by Amazon since the beginning of April 2011 [25].

A second problem is related to the fact that the ssh server may also permit password-based authentication, thus pro- viding a similar backdoor functionality if the AMI provider does not remove her passwords from the machine. In addi- tion, while leftover ssh keys only allow people with the corre- sponding private key (normally the AMI image creator), to obtain access to the instance, passwords provide a larger at- tack vector: Anybody can extract the password hashes from an AMI, and try to crack them using a password-cracking tool (e.g., John the Ripper [13]).

In other words, ssh keys were probably left on the images by mistake, and without a malicious intent. The same ap- plies to password, with the difference that passwords can also be exploited by third parties, transforming a mistake in a serious security problem.

During our tests, we gathered these leftover credentials, and performed an analysis to verify if a remote login would
be possible by checking the account information in /etc/passwd
and /etc/shadow, as well as the remote access configuration of OpenSSH.

The results, summarized in Table 2, show that the prob- lem of leftover credentials is significant: 21.8% of the scanned AMIs contain leftover credentials that would allow a third-

	East	West	EU	Asia	Total
AMIs (%)	34.8	8.4	9.8	6.3	21.8
With Passwd	67	10	22	2	101
With SSH keys	794	53	86	32	965
With Both	71	6	9	4	90
Superuser Priv.	783	57	105	26	971
User Priv.	149	12	12	12	185

party to remotely login into the machine. The table also reports the type of credentials, and lists how many of these would grant superuser privileges (either via root, sudo or su with a password).

3.3 Privacy Risks

The sharing of AMIs not only bears risks for the customers who rent them, but also for the user who creates and dis- tributes the image. In fact, if the image contains sensitive in- formation, this would be available to anybody who is renting the AMI. For example, an attacker can gather SSH private keys to break into other machines, or use forgotten Amazon Web Services (AWS) keys to start instances at the image provider's cost. In addition, other data sources such as the browser and shell histories, or the database of last login at- tempts can be used to identify and de-anonymize the AMI's creator.

Private keys

We developed a number of tests to search the AMIs' file- system for typical filenames used to store keys (e.g., id_dsa and id_rsa for SSH keys, and pk-[0-9A-Z]*.pem and cert-
[0-9A-Z]*.pem for AWS API keys). Our system was able to identify 67 Amazon API keys, and 56 private SSH keys that were forgotten. The API keys are not password pro-
tected and, therefore, can immediately be used to start im- ages on the cloud at the expense of the

INTERNATIONAL JOURNAL OF SCIENTIFIC & ENGINEERING RESEARCH, VOLUME 6, ISSUE 4, APRIL-2015
ISSN 2229-5518

key's owner. Even though it is good security practice to protect SSH keys with a passphrase, 54 out of 56 keys were not protected. Thus, these keys are easily reusable by anybody who has access to them. Although some of the keys may have been generated specifically to install and configure the AMI, it

Browser History

Nine AMIs contained a Firefox hif ethical concerns, we did not manually inspect the contents of the browser history. Rather, we used scripts to check which domains had been contacted. From the automated analysis of the history file, we discovered that one machine was used by a person to log into the portal of a Fortune 500 company. The same user then logged into his/her personal Google email account. Combining this kind of information, history files can easily be used to de-anonymize, and reveal information about the image's creator.

Shell History

When we tested the AMI using our test suite, we inspected common shell history files (e.g. ~/.history, ~/.bash_history, ~/.sh_history) that were left on the image when it was created. We discovered that 612 AMIs (i.e., 11.54% of the total) contained at least one single history file. We found a total of 869 files that stored interesting information (471 for root and 398 for generic users), and that contained 158,354 lines of command history. In these logs, we identified 74 dif- ferent authentication credentials that were specified in the command line, and consequently recorded on file (ref. Ta- ble 3).

For example, the standard MySQL client allows to spec- ify the password from the command line using the -p flag.
A similar scenario occurs when sensitive information, such as a password or a credit card number, is transferred to a web application using an HTTP GET request. GET re- quests, contrary to POST submissions, are stored on the web server's logs. The credentials we discovered belong to two categories: local and remote.

The credentials in the image group grant an attacker ac- cess to a service/resource that is hosted on the AMI. In contrast, remote credentials enable the access to a remote target. For example, we identified remote credentials that can be used to modify (and access) the domain name in- formation of a dynamic DNS account. A malicious user that obtains a DNS management password can easily change the DNS configuration, and redirect the traffic of the orig- inal host to his own machines. In addition, we discovered four credentials for the Amazon Relational Database Service (RDS) [32] – a web service to set up, operate, and scale a relational database in the Amazon cloud. We also found credentials for local and remote web applications for differ- ent uses (e.g. Evergreen, GlassFish, and Vertica) and for a database performance monitoring service. One machine was configured with VNC, and its password was specified from the command line. Finally, we were able to collect 13 credentials for MySQL that were used in the authentication of remote databases.

Recovery of deleted files

In the previous sections, we discussed the types of sensitive information that may be forgotten by the image provider. Unfortunately, the simple solution of deleting this informa- tion before making the image publicly available is not satis- factory from a security point of view.In many file systems, when a user deletes a file, the space occupied by the file is marked as free, but the content of the file physically remains on the media (e.g. the hard-disk). The contents of the deleted file are definitely lost only when this marked space is overwritten by another file. Utilities such as shred, wipe, sfill, scrub and zerofree make data recovery difficult either by overwriting the file's contents before the file is actually unlinked, or by overwriting all the corresponding empty blocks in the filesystem (i.e., secure deletion or wiping). When these security mechanisms are not used, it is possible to use tools (e.g., extundelete and Winundelete) to attempt to recover previously deleted files. In the context of Amazon EC2, in order to publish a custom image on the Amazon Cloud, a user has to prepare her image using a predefined procedure called bundling. This procedure involves three main steps: Create an image from a loopback device or a mounted filesystem, compress and en- crypt the image, and finally, split it into manageable parts so that it can be uploaded to the S3 storage. The first step of this procedure changes across different bundling methods adopted by the user (ref. Table 4). For example, the ec2-bundle-image method is used to bundle an image that was prepared in a loopback file. In this case, the tool transfers the data to the image using a block level operation (e.g. similar to the dd utility). In contrast, if the user wishes to bundle a running system, she can choose the ec2-bundle-vol tool that creates the image by recursively copying files from the live filesystem (e.g., using rsync). In this case, the bundle system works at the file level.

Any filesystem image created with a block-level tool will also contain blocks marked as free, and thus may contain parts of deleted files.

INTERNATIONAL JOURNAL OF SCIENTIFIC & ENGINEERING RESEARCH, VOLUME 6, ISSUE 4, APRIL-2015
ISSN 2229-5518

Type	#
Home files (/home, /root)	33,011
Images (min. 800x600)	1,085
Microsoft Office documents	336
Amazon AWS certificates and access keys	293
SSH private keys	232
PGP/GPG private keys	151
PDF documents	141
Password file (/etc/shadow)	106

Table 5: Recovered data from deleted files

Undelete tool [31], and were able to recover deleted files in all cases. Interestingly, we were also able to undeleted 8,996 files from an official image that was published by Amazon AWS itself.

4 Machine Fingerprinting

In the previous sections, we presented a number of experi- ments we conducted to assess the security and privacy issues involved in the release and use of public AMIs. The results of our experiments showed that a large number of factors must be considered when making sure that a virtual ma- chine image can be operated securely (e.g., services must be patched and information must be sanitized).

A number of the issues we described in the previous sec- tions could potentially be exploited by an attacker (or a ma- licious image provider) to obtain unauthorized remote access to any running machine that adopted a certain vulnerable AMI. However, finding the right target is not necessarily an easy task.

For example, suppose that a malicious provider distributes an image containing his own ssh key, so that he can later lo- gin into the virtual machines as root. Unfortunately, unless he also adds some kind of mechanism to "call back home" and notify him of the IP address of every new instance, he would have to brute force all the Amazon IP space to try to find a running machine on which he can use his creden- tials. To avoid this problem, in this section we explore the feasibility of automatically

In order to explore the feasibility, from an attacker point of view, of automatically matching a running instance back to the corresponding AMI, we started our experiment by querying different public IP registries (ARIN, RIPE, and LAPNIC) to obtained a list of all IPs belonging to the Ama- zon EC2 service for the regions US East/West, Europe and Asia. The result was a set of sub-networks that comprises 653,401 distinct IPs that are potentially associated with run- ning images.

For each IP, we queried the status of thirty commonly used ports (i.e., using the NMap tool), and compared the results with the information extracted from the AMI analysis. We only queried a limited number of ports because our aim was to be as non-intrusive as possible. (i.e., see Section 6 for a detailed discussion of ethical considerations, precautions,and collaboration with Amazon). For the same reason, we configured NMap to only send a few packets per second to prevent any flooding, or denial of service effect. Our scan detected 233,228 running instances. This num- ber may not reflect the exact number of instances there were indeed running. That is, there may have been virtual ma- chines that might have been blocking all ports.

We adopted three different approaches to match and map a running instance to a set of possible AMIs. The three methods are based on the comparison of the SSH keys, ver- sions of network services, and web-application signatures.

Table 6 depicts the results obtained by applying the three techniques. The first column shows the number of running instances to which a certain technique could be applied (e.g., the number of instances where we were able to grab the SSH banner). The last two columns report the number of running machines for which a certain matching approach was able to reduce the set of candidate AMIs to either 10 or 50 per matched instance. Since 50 possibilities is a number that is small enough to be easily brute-forced manually, we can conclude that it is possible to identify the AMI used in more than half of the running machines. SSH matching Every SSH server has a host key that is used to identify itself. The public part of this key is used to verify the authenticity of the server. Therefore, this key is disclosed to the clients. In the EC2, the host key of an image needs to be regenerated upon instantiation of an AMI for two reasons: First, a host key that is shared among sev- eral machines makes these servers vulnerable to man-in-the- middle attacks (i.e., especially when the private host key is freely accessible). Second, an unaltered host key can serve as an identifier for the AMI, and may thus convey sensitive information about the software that is used in the instance.

This key regeneration operation is normally performed by the cloud-init script provided by Amazon. The script

the user has modified the running services. However, since most services installed on the AMIs were old and out of date, it is very unlikely that new services (or updated ones) will match the same banners as the one extracted from the AMIs. Therefore, a service update will likely decrease the matching rate, but unlikely generate false positives. The fact that over 7,000 machines were identified using this method seems to support the hypothesis that a large number of users often forget to update the installed software after they rent an AMI.

Web matching For our last AMI matching approach, we first collected web information from all the instances that had ports 80 and 443 (i.e., web ports)

INTERNATIONAL JOURNAL OF SCIENTIFIC & ENGINEERING RESEARCH, VOLUME 6, ISSUE 4, APRIL-2015
ISSN 2229-5518

open. We then com- pared this information with the data we collected during the scan of the Amazon AMIs.

In the first phase, we used the WhatWeb tool [2] to ex-
tract the name and version of the installed web server, the configuration details (e.g., the OpenSSL version), and the installed interpreters (e.g., PHP, and JSP). In addition, we also attempted to detect the name and version of the web applications installed in the root document of the web server by using WhatWeb's plugins for the detection of over 900 popular web software.

In the second phase, we compared this information to the scanned AMIs, and checked for those machines that had the same web server, the same configuration, and the same ver- sions of the language interpreters. Since different installa- tions of the same operating system distribution likely share this information, we then further reduced the size of the can- didate set by checking the web application name detected by the WhatWeb tool.

The last row of Table 6 shows that we were able to identify more than 5,000 machines by using this technique.

5 Amazon's Feedback

Clearly, one question that arises is if it is ethically accept- able and justifiable to conduct experiments on a real cloud service. During all our experiments, we took into account the privacy of the users, the sensitivity of the data that was analyzed, and the availability of Amazon's services. In addi- tion, all our AMI tests were conducted by automated tools running inside virtual machines we rented explicitly for our study. We did not use any sensitive data extracted from the AMI, or interact with any other server during this test. In addition, we promptly notified Amazon of any problem we found during our experiments.

Amazon has a dedicated group dealing with the security issues of their cloud computing infrastructure: the AWS (Amazon Web Services) Security Team. We first contacted them on May 19th 2011, and provided information about the credentials that were inadvertently left on public AMIs. Amazon immediately verified and acknowledged the prob- lem, and contacted all the affected customers as summa- rized by a public bulletin released on June 4th [29]. In cases where the affected customer could not be reached immedi- ately, the security team acted on behalf of the user, and changed the status of the vulnerable AMI to private to pre- vent further exposure of the customer's personal credentials. We also communicated to the AWS Security team our con- cerns regarding the privacy issues related to publishing of public AMIs (e.g., history files, remote logging, and left- over private keys). The security team reacted quickly, and realeased a tutorial [28] within five days to help customers share public images in a secure manner. Finally, we con- tacted again Amazon on June 24th about the possibility of recovering deleted data from the public Amazon AMIs. To fix the problem, we provided them some of the countermea- sures we discussed in Section 4.3. Their team immediately reported the issue internally and was grateful of the issue we reported to attention. By the time of writing, Amazon has already verified all the public AMIs where we have been able to recover data, and has moved on to check the status of all other public AMIs. The AWS security team is also working on providing a solution to prevent the recovery of private documents by undeletion.

The second part of our experiments included the use of a port scanner to scan running images. Even though port scanning has not been considered to be illegal per se (e.g., such as in the legal ruling in [1]), this activity may be consid- ered an ethically sensitive issue. However, given the limited number of ports scanned (i.e, 30) and the very low volume of packets per second that we generated, we believe that our activity could not have caused any damage to the integrity and availability of Amazon's network, or the images running on it. As researchers, we believe that our experiments helped Amazon and some of its customers to improve their security and privacy. In addition, we hope other cloud providers will benefit from the results of this research to verify and improve their security just as Amazon has done.

6 Related Work

There are several organizations that released general secu- rity guidance on the usage of cloud computing, such as [4, 11]. Amazon Web Services, in addition to the security bul- letins already mentioned, released a paper describing the security processes put in place, focusing more specifically on the management of public images [6]. The problem statement of security on cloud computing infrastructures has been widely explored. Garfinkel and Rosenblum [19] studied the

implement or perform a wide range of tests on an existing cloud infrastructure.More specific to Amazon EC2, a novel approach was pro- posed by Bleikerts et al. [16]. The paper analyses the secu- rity of an infrastructure (a set of connected virtual machines) deployed on Amazon EC2 through graph theory techniques. Other works focused on the placement algorithm of Amazon EC2 instances [26], and showed how to exploit it in order to achieve co-residence with a targeted instance.

Finally, concurrently and in parallel to our work, Bugiel et al. [17] have recently conducted a study in which they per- form similar experiments on the Amazon's EC2 catalogue and have reached similar conclusions. Note, however, that our experiments are more comprehensive and have been con- ducted on a larger scale. While they have only considered 1255 AMIs, we selected and automatically analyzed over 5000 public images provided by Amazon in four distinct data centers. We also discovered and discussed a wider number of security issues by testing every image for known malware samples and vulnerabilities. Furthermore, we collaborated closely with Amazon's Security Team to have the identified problems acknowledged and fixed.Even though most of these papers highlighted trust and security problems associated to the use of third party im- ages, to the best of our knowledge we are the first to preset a large-scale, comprehensive study of the security and pri- vacy of existing images.

7 Conclusion

Cloud services such as Amazon's Elastic Compute Cloud and IBM's SmartCloud are quickly changing the way organiza- tions are dealing with IT infrastructures and are providing online services. It is easy to obtain computing power today. One can simply buy it online and use application program- ming interfaces provided by cloud companies to launch and shut down virtual images. A popular approach in cloud- based services is to allow users to create and share virtual images with other users. Cloud providers also often provide virtual images that have been pre-configured with popular software such as open source web servers. In this paper, we explored the general security risks associated with vir- tual server images from the public catalogs of cloud service providers. We investigated in detail the security problems of public images that are available on the Amazon EC2 service.

hope that the results of this study will be useful for other cloud service providers who offer similar services.

8 Acknowledgments

This research has been partially funded by the European Union Seventh Framework Programme (FP7/2007-2013) un- der grant agreement n. 257007, by the NSF (CNS-1116777) and by COMET K1, FFG - Austrian Research Promotion Agency and the Secure Business Austria. The authors would like to acknowledge Don Bailey and the Amazon Web Ser- vices Security Team for their commitment in addressing the security risks we identified.

9 References

[1] Security focus: Port scans legal, judge says. http://www.securityfocus.com/news/126.
[2] Whatweb webapp scanner. http: //www.morningstarsecurity.com/research/whatweb.
[3] Syslog-ng, 2000-2011. http: //www.balabit.com/network-security/syslog-ng/.
[4] Security guidance for critical areas of focus in cloud computing v2.1, Dec. 2009. https://cloudsecurityalliance.org/csaguide.pdf.
[5] Extundelete linux data recovery tool, 2010. http://extundelete.sourceforge.net/.
[6] Amazon elastic compute cloud, May 2011. http://aws.amazon.com/security.
[7] Amazon elastic compute cloud (amazon ec2), May 2011. http://aws.amazon.com/ec2/.
[8] Clamav, July 2011. http://www.clamav.net/.
[9] Cloud computing, cloud hosting and online storage by rackspace hosting, May 2011. http://www.rackspace.com/cloud/.
[10] Enterprise cloud computing from terremark, May 2011. http://www.terremark.com/services/ cloudcomputing.aspx.
[11] Guidelines on security and privacy in public cloud computing, draft special publication 800-144, 2011.
[12] Ibm smartcloud, May 2011. http: //www.ibm.com/cloud-computing/us/en/#!iaas.
[13] John the ripper unix password cracker, Apr. 2011. http://www.openwall.com/john/.
[14] Joyent smartdatacenter, May 2011. http://www. joyent.com/services/smartdatacenter-services/.
[15] Reminder about safely sharing and using public amis, Jun 2011. http: //aws.amazon.com/security/security-bulletins/ reminder-about-safely-sharing-and-using-public-amis/.
[16] S. Bleikertz, M. Schunter, C. Probst, D. Pendarakis, and K. Eriksson. Security audits of multi-tier virtual infrastructures in public infrastructure clouds. In Proceedings of the 2010 ACM workshop on Cloud computing security workshop.
[17] S. Bugiel, S. Nürnberger, T. Pöppelmann, A. Sadeghi, and T. Schneider. Amazonia: When elasticity snaps back. 2011.
[18] M. Christodorescu, R. Sailer, D. Schales, D. Sgandurra, and D. Zamboni. Cloud security is not (just) virtualization security: a short paper. In Proceedings of the 2009 ACM workshop on Cloud computing security.
[19] T. Garfinkel and M. Rosenblum. When virtual is harder than real: Security challenges in virtual machine based computing environments. In Proceedings of the 10th conference on Hot Topics in Operating Systems-Volume 10.
[20] R. Glott, E. Husmann, A. Sadeghi, and M. Schunter. Trustworthy clouds underpinning the future internet

INTERNATIONAL JOURNAL OF SCIENTIFIC & ENGINEERING RESEARCH, VOLUME 6, ISSUE 4, APRIL-2015
ISSN 2229-5518

APPENDIX

A Test Suite

Tests	Type	Details	OS
System Information	General	-	Windows + Linux
Logs/eMails/WWW Archive	General	-	Linux
Processes and File-System	General	-	Windows + Linux
Loaded Modules	General	lsmod	Linux
Installed Packages	General	-	Linux
General Network Information	Network	Interfaces, routes	Windows + Linux
Listening and Established Sockets	Network	-	Windows + Linux
Network Shares	Network	Enabled Shares	Windows + Linux
History Files	Privacy	Common Shells + Browsers	Windows + Linux
AWS/SSH Private Keys	Privacy	Loss of sensitive info (Only on X AMIs)	Linux
Undeleted Data	Privacy	-	Linux
Last logins	Privacy	MySQL and PostgresSQL	Linux
SQL Credentials	Privacy/Security	Enabled Logins	Linux
Password Credentials	Privacy/Security	Backdoor access	Windows + Linux
SSH Public Keys	Security	Rootkit	Linux
Chkrootkit	Security	Rootkit	Linux
RootkitHunter	Security	Rootkit	Linux
RootkitRevealer	Security	General Security Issues	Windows
Lynis Auditing Tool	Security	Antivirus	Linux
Clam AV	Security	Processes/Sockets Hiding	Windows + Linux
Unhide	Security	Processes Hiding	Linux
PsList	Security	-	Windows
Sudoers Configuration	Security		Linux

Table 7: Details of the tests included in the automated AMI test suite

B Vulnerabilities in AMIs

	Windows	Linux
Tested AMIs	253	3,432
Vulnerable AMIs	249	2,005
With Vuln. <=2 Years	145	1,197
With Vuln. <=3 Years	38	364
With Vuln. <=4 Years	2	106
Avg. # Vuln./AMI	46	11
TOP 10 Vuln.	MS10-037, MS10-049, MS10-051, MS10-073, MS10-076, MS10-083, MS10-090, MS10-091, MS10-098, MS11-05	CVE-2009-2730, CVE-2010-0296, CVE-2010-0428, CVE-2010-0830, CVE-2010-0997, CVE-2010-1205, CVE-2010-2527, CVE-2010-2808, CVE-2010-3847, CVE-2011-0997

Table 8: Nessus Results

Table 8 reports the most common vulnerabilities that affect Windows and Linux AMIs. For example, the vulnerabilities MS10-098 and MS10-051 affect around 92% and 80% of the tested Windows AMIs, and allows remote code execution if the user views a particular website using the Internet Explorer. Microsoft Office and the Windows' standard text editor Wordpad contained in 81% of the Windows AMIs allow an attacker to take control of the vulnerable machine by opening a single malicious document (i.e., vulnerability MS10-83). A similar vulnerability (i.e., CVE-2010-1205) affects Linux AMIs as well: A PNG image sent to a vulnerable host might allow a malicious user to run code remotely on the AMI. We also observed that 87 public Debian AMIs come with the now notorious SSH/OpenSSL vulnerability discovered in May 2008 (i.e., CVE-2008-0166) in which, since the seed of the random number generator used to generate SSH keys is predictable, any SSH key generated on the vulnerable systems needs to be considered as being compromised.

8.74% of Linux AMIs contain a DHCP client that is vulnerable to a remote code execution. In fact, it fails to properly escape certain shell meta characters contained in the DHCP server responses (vulnerability CVE-2011-0997). An attacker that setups a bastion host in the Amazon cloud, can send around DHCP custom packets that may exploit users' machines installed in the neighborhood. Finally, more than 26.5% of machines contained a 2-years old vulnerability (CVE-2009-2730) that may allows an attacker settled in the Amazon cloud to spoof arbitrary SSL servers via a crafted certificate.