Distributing and Trusting Images between Cloud Providers

An Honors Thesis
Presented to
the Honors College of
Clemson University

In Partial Fulfillment
of the Requirements for
Departmental Honors
Bachelors of Science
Computer Science

by
Loren Klingman
May 2011

Accepted by:
Dr. Sebastien Goasguen, Committee Chair

Abstract

The computing industry continues to shift towards the cloud computing paradigm. Companies use cloud computing daily to serve their websites and offer services in scalable and cost effective environment while researchers use it to process massive amounts of data at great speeds. One of the challenges identified for using cloud computing in computational science is that each user wants to provision their own customized environment. Users also want to be able to provision this environment on different cloud providers. Virtualization is being used to establish this type of environment and rich APIs are being developed to expose these services to the users.

However, the cloud providers want to know that the user is not creating a security risk on the machines by using the virtual image to access the machine hardware, thus enabling the user to access data or communications that should have been protected. The provider may also wish to ensure that the user complies with policy guidelines.

The work presented investigates building a chain of trust between image producers (users) and cloud providers. An endorser role is defined as the middle agent who checks the integrity of the images. We implement a solution using a virtual machine image catalogue (VMIC) originally developed at CERN which allows a provider or endorser to provide a list of the images they have approved for distribution. We extend it to provide a way to sign the list of images and the images themselves in order to provide trust. Finally, we add the distribution framework to allow catalogue owners to import the approved images from other catalogues.

Acknowledgments

I would first like to thank my advisor, Dr. Sebastien Goasguen, for his patience, support, ideas, and guidance throughout the process of my finding and completing a project. If it were not for him, I would not have been able to complete my honors thesis.

I also owe my appreciation to the team at CERN in Geneva, Switzerland who laid the foundation for the Virtual Machine Image Catalogue and worked with me to merge my work into their source code.

Many thanks are also due to Dr. Nancy Meehan and Dr. Roy Pargas who helped me begin my research career at Clemson University. Thanks to you for introducing me to the field and providing guidance and eduction to me on how to write my research.

A special thanks to my roommates Trent and Patrick who have listened to me talk about my work and provided encouragement to me along the way.

I would also like to thank my mother, Dr. Nancy Phillips, for her invaluable help proof-reading the manuscript.

Table of Contents

List of Figures

List of Listings

Chapter 1

Introduction

Cloud computing has become a buzzword in the era of Web 2.0. According to Foster, Cloud computing is ”A large-scale distributed computing paradigm that is driven by economies of scale...” [11] By this, he means that it allows a central pool of computers to be drawn from as needed such that companies do not have to purchase excess equipment to handle peak or intermittent loads. People have already begun to optimize Amazon’s EC2 cloud for both kinds of work. For example, recent uses include mass processing of data for research projects [17] and scaling systems as demand requires [25].

Because customers want to operate in a custom environment (to choose the operating system, software packages, and software configuration), this work extends existing work by investigating how a company can trust that specific images meet their policies. (Running a rogue or untrusted image can result in many security issues and policy violations.) Starting with a virtual machine image catalog (VMIC) developed at CERN which allows for the listing of approved images and distribution within a company [27], this project expands on that work by implementing a system that allows one company to trust another company to validate images. Our code then works to prevent anyone from falsely appearing to have been validated by that company and to prevent image files that have been modified from being approved to run.

1.1 Grid Computing vs Cloud Computing

While many people do not distinguish between cloud computing and grid computing, the differences are important.

1.1.1 Grid Computing

Grid computing is based on a project-oriented business model meaning the user organizations have to write a proposal and are then granted a certain number of computing hours on the grid. Grids define and provide certain protocols to enable interoperability and security. These items are both needed so that the people at the various administrative domains can ensure that the global and local resource usage policies are met and so that groups with different hardware and software configurations will be able to interact with each other and share resources. The resource usage for a grid is generally controlled by batch-scheduling, which creates inherit latency as the scheduling tool waits for the requested resources to be available. Grids may be connected via fast networking and shared data drives to allow for maximum throughput in shared applications. Grid sites natively build in the assumption that the equipment may change within one site, so applications are built to be movable and security is engineered into the fundamental infrastructure. [11]

1.1.2 Cloud Computing

In contrast, the model for cloud computing is based on a consumption basis where the user pays for the amount of resources used. This rewards resource conservation since users are only billed for what they use and requires little forward planning due to the appearance of unlimited resources which can quickly be brought online. [1] The architecture of a cloud is usually simpler than that of a grid with the resources being abstracted or virtualized for the user. The nature of a cloud reduces the wait time of a job since many jobs are run at the same time on a single system, but as a result it will decrease the speed. Also, cloud computing does not work as well as grid computing for data intensive applications since the infrastructure is not usually in place to allow fast data sharing. Virtualization is a common technique to handle the need to give each user on a system in the cloud the appearance that they are using it alone and helps to protect each users private data. In the past, the use of virtualization created a significant loss of speed, but now, through hardware support, the performance gap has been narrowed. [11] Clouds are defined to be, "A large-scale distributed computing paradigm that is driven by enconomies of scale, in which a pool of abstracted, virtualized, dynamically-scalable, managed computing power, storage, platforms, and services are delivered on demand to external cusomers over the Internet" [11]. A lack of common protocols in the cloud industry has caused integration and interoperability problems which prevent users from being able to use the same setup on different cloud providers. [1]

1.2 VMIC Principles

Virtual machine image catalogs must have the following sets of principles. First, they introduce a basic policy for trusting and distributing images within an organization. Second, they make machine provisioning more flexible since there is a pre-existing list of available operating system and software configurations (virtual machine images) which can be quickly brought online using a hypervisor. Third, they are self managing in terms of ensuring that the distributed images stay current with new versions of images and changes to the approved image list. Fourth, they allow image producers and endorsers to endorse all the images produced or endorsed by another person. Fifth, they release enough information through the use of metadata to allow for image retrieval, verification, and for users to know which image they need. (The information for users could include: version, architecture, software, and purpose.) [27]

1.3 Motivation for VMIC Sharing

Our project applies to both grid and cloud computing since it is possible for either to allow the use of virtual machine images. The benefits afforded by the ability to use the same image set across a variety of sites are as large as what the web provided when it became a universal information protocol [10] and prevents users' data from falling prey to data lock-in [1]. We enhance the benefits of virtual machine image catalog sharing by adding the ability to share the approval of images throughout various sites. However, the risks incurred by the incorrect use of image sharing are also very large. As a result, we are working to eliminate the possibility of attacks caused by being able to implement the kernel and adminstrative changes in the virtual machine image when sharing virtual machine image catalogs by strengthening the validation process.

1.4 Contributions

This thesis provides conributions in three areas. First, we investigate the trust and identity verification issues involved in allowing external users and groups to submit their images to a cloud. Second, we propose a solution which works with an existing image catalog to allow a catalog to be endorsed and exported to the web. Finally, we display the prototype system implementation.

1.5 Thesis Organization

Chapter 2 surveys some of the closely related work in cloud computing, virtual machines, and the use of virtual machines in the grid. Chapter 3 provides a brief overview of each of the technologies that we used to create our prototype (Django, Google App Engine, and Shibboleth). Chapter 4 details our work with the virtual machine image catalog to allow for the endorsing and use of external machine images. Finally, Chapter 5 presents a summary of our work and our conclusions.

Chapter 2

Related Work

Much work has already been done in the domain of clouds, grids, and virtual machines, and some work has been done on placing virtual machines in the grid and cloud. In this chapter, we present some of the most relevant work. The next section presents uses for grid computing. In section 2.2, we review related work on cloud computing. Section 2.3 investigates related work on virtual machines. Section 2.4 looks at virtual machines as they have moved into the grid. Section 2.5 discusses the use of virtual machines in the cloud. Finally, Section 2.6 examines security concerns with virtual machines.

2.1 Grid Computing

The grid is a combination of technologies designed to allow for sophisticated data processing and research computing, but it also requires social aspects to induce people to use the system. [12] By considering other distributed frameworks in history, such as electricity grids and the grid formed by Chicago as a gateway city to the West [5], we get an idea about how the grid should work. Because of the large-scale resource sharing, innovative applications, and, in some cases, high performance orientation, the grid makes quick work of changing society by allowing for fast processing of studies on ozone depletion, global warming, pollution, biology, and many other areas of research. Research in the grid looks at cooridinating resource sharing and user authentication between various virtual organizations through the use of protocols, services, application programming interfaces, and software developement kits. [13]

2.2 Cloud Computing

Computers may eventually be viewed as a fifth utility (after water, electricity, gas, and phone). This would be provided via cloud computing where the user could lease processing power and storage in the cloud rather than purchasing his own hardware. Cloud computing frequently leverages virtual machines, but that will be investigated more in section 2.5. Using computers in the cloud adds a layer of complexity via Service Level Agreements (SLAs) and risk management strategies in case the cloud becomes inaccessible. Several different cloud platforms have been developed such as Google AppEngine, Microsoft Azure, and Amazon EC2 which allow for application scalability and high performance computing. [3]

2.3 Virtual Machines

Virtual machines were developed by IBM in the late 1960's to provide concurrent, interactive access to mainframe computers. [24] This allowed the virtual machines to be a replica of the underlying physical machine so users had the illusion of running directly on the physical machine. More benefits such as the ability to isolate programs and run multiple flavors/configurations of operating systems came later in the process. There have also been many improvements in the ways that the virtual OS interacts with hardware devices on the host system. Some of the major areas of study for this pertain to the networking interface, since the virtual systems should not be able to detect the network traffic that is targeted to the host OS or to other virtual machines that are running. [24] Much work has also been done by Intel and Advanced Micro Devices to improve the performance of virtual machines by implementing hardware optimizations that reduce the amount of overhead from the virtual machine hypervisor.

2.4 Use of Virtual Machines in the Grid

Virtual machines in the grid address three issues that arise from using a standard grid environment: support for legacy applications, a layer of security between untrusted code and users, and computation deployment that is independent of site administration. They also allow complete environment customisation for the user, administrator privileges to execute restricted tasks, resource control at the virtual machine manager, and site-independence for implementation. Virtual machines do create some overhead but hardware integration with virtual machine management has greatly reduced this overhead. [11] According to tests, the difference is less than 10% for micro and macro tasks. [9] Virtual machines also take some time to bring up, but that is all one time and can be reduced with fast disc speeds and resumes instead of full boots. [9] Benefits are also gained with virtual machines because multiple jobs can be run simultaneously without any need to worry about conflict. This decreases the job wait time and allows resources that would have otherwise been left idle to be used. [18] Since virtual machines can be integrated with the current grid structure, it is an easy change to make to gain all of these benefits.

2.5 Use of Virtual Machines in the Cloud

The benefits of virtual machines in the cloud are very similar to the benefits of virtual machines in the grid. Cloud providers such as Amazon EC2 and Rackspace actively use virtual machines in the cloud. Since virtual machines in the cloud are the same as the virtual machine anyone can run on their desktop, research in this area pertains to the management of the virtual machines. One such paper looks at various techniques for provisioning virtual machine instances [22]. Other researchers look at how to manage and trust the virtual machine images [27] and [28]. Trusting the images is important, because the image is the starting point, a bad image will result in virtual machines that are compromised from the beginning. Still others investigate security weaknesses of virtual machines in the cloud. These security weaknesses exist in virtual machines in general but are more prevalent in the cloud where machines are shared between several different users. We look at these security concerns in section 2.6.

2.6 Security Concerns with Virtual Machines

A major concern with virtual machines is whether the various virtual machines on one host system are properly isolated from each other. Since users would have root and kernel access to the virtual system, it is important that the host only give the virtual machine access to its own data. Schiffman et al. investigate using hardware to verify that isolation is properly happening [20]. Christodorescu et al. research methods for securing virtual machines without need to make assumptions about the state of the virtual machine [4]. For more information on security concerns with virtual machines and security concerns for cloud computing see the ACM workshop on Cloud computing security. [21]

Chapter 3

Background and Technology Used

Before we explain the design of our virtual machine image catalog, it is important to give a brief overview of each of the technologies used in the development of the catalog so that the reader can better understand our design choices. The first technology used is the Django framework for Python which helps to automate the creation of administration pages and assists with code reusability. The second is WSGI which enhances the Apache web server by adding the ability to run Python scripts like Django and send the result to web browers. Third, we show how to add the ablilty to run Python and Django on Google AppEngine which could enable the catalog to live in the cloud. The fourth and final piece is Shibboleth which is a single sign on platform used to automate logins with Clemson University's (and many other companies and institutions) login framework.

3.1 Django

According to the project website, "Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design." This means that it focuses on high performance and fast development. Django also adheres to the DRY - (Don't Repeat Yourself) principle so the code tries to ensure that the same code segment never needs to be entered in two places.

The Django tutorial [7] shows the simplicity of creating a script with an administration interface. Simply fill in a model (see Listing 3.1) and create an admin.py file to register the model as part of the administration interface (see Listing 3.2), and your site will be ready to publish

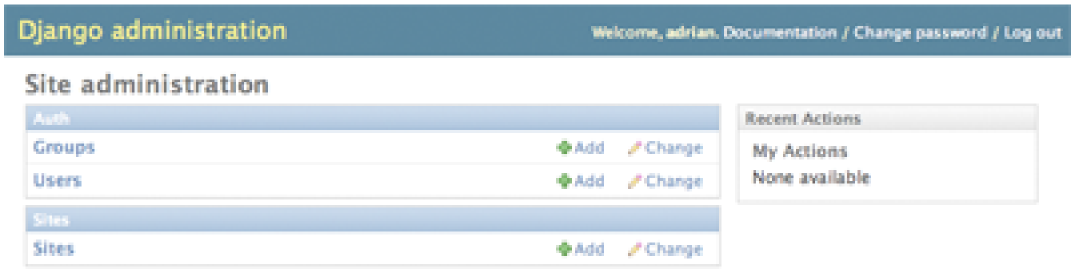


Figure 3.1: Django Admin Interface

with an administration interface already built for you. The default interface provided by Django would be similar to the one in Figure 3.1. This ease of use extends into the creation of the main pages of websites and into template design. For more information, see the Django website and documentation [6].

```
class Reporter(models.Model):
    full_name = models.CharField(max_length=70)

    def __unicode__(self):
        return self.full_name

class Article(models.Model):
    pub_date = models.DateTimeField()
    headline = models.CharField(max_length=200)
    content = models.TextField()
    reporter = models.ForeignKey(Reporter)

    def __unicode__(self):
        return self.headline
```

Listing 3.1: Sample Django Model

```
import models
from django.contrib import admin

admin.site.register(models.Article)
```

Listing 3.2: Sample Django Admin File

3.2 Django with Apache

WSGI must be used in order to run Python scripts through the Apache web-server. WSGI stands for Web Server Gateway Interface. It is a specification developed by Python for web servers and application servers to communicate with web applications (though it also has additional functions). Using it is a fairly simple process, install mod_wsgi into Apache, add a WSGI configuration file (see Listing 3.3), and a lines of code to the Apache configuration file to load the WSGI file (see Listing 3.4). More help can be obtained from the websites of WSGI [29] and modwsgi [8].

```
import os
import sys

sys.path.append('/home/loren/workspace/VMIC/src/vmic')
sys.path.append('/home/loren/workspace/VMIC/src/vmic/vmic')

os.environ['DJANGO_SETTINGS_MODULE'] = 'vmic.settings'

import django.core.handlers.wsgi
application = django.core.handlers.wsgi.WSGIHandler()
```

Listing 3.3: WSGI Configuration for Django

```
WSGIScriptAlias /vmic /home/loren/workspace/VMIC/src/vmic/vmic/apache/vmic.wsgi
```

Listing 3.4: Apache Configuration for WSGI

3.3 Django with Google AppEngine

Google AppEngine is a platform as a service which as Armbrust says in his paper, ”is targeted exclusively at traditional Web applications” [1]. The scalability allows for users applications to scaled and for the user to pay only for the services used. Other services provide scalability, with Google, the user never has to worry about how many servers to bring online at each point during the day because it automatically scales for you.

To implement this technology, first, the user needs a Google AppEngine account with a space for this application, and the Google AppEngine SDK installed on their system. After configuration (see Listing 3.5, the SDK allows the user to execute `appcfg.py update ./` to upload their application to the AppEngine servers and deploy it. For more information on Google AppEngine or to get an account, visit the website [15].

Getting a Django application online in Google AppEngine requires a bit of finesse because the standard sqlite or MySQL database backend will not work on Google AppEngine. Also, a few extra modules which allow for the framework to communicate properly in the cloud must be added to Django. Adding the modules, simply requires a few downloads and added configuration lines to load the new modules. The database changes require switching the database over to nonrel. If the fields are all compatible, this is as simple as changing a line in the settings and migrating your data over to the new database, but if the Django implementation is using ManyToManyFields or a few other non-supported features, then work arounds will have to be written in the code. A full tutorial on this topic is available from All Buttons Pressed [19].

```
application: loren-django
version: 1
runtime: python
api_version: 1

builtins:
- remote_api: on

inbound_services:
- warmup

handlers:
- url: /_ah/queue/deferred
  script: djangoappengine/deferred/handler.py
  login: admin

- url: /_ah/stats/.*
  script: djangoappengine/appstats/ui.py

- url: /media/admin
  static_dir: django/contrib/admin/media
  expiration: '0'

- url: /.*
  script: djangoappengine/main/main.py
```
Listing 3.5: Google AppEngine Configuration

3.4 Shibboleth

"The Shibboleth System is a standards based, open source software package for web single sign-on across or within organizational boundaries. It allows sites to make informed authorization decisions for individual access of protected online resources in a privacy-preserving manner." [16] This means that websites can release only the amount of information needed about a user to each service while also allowing the user to sign-on only one time to access all of the services.

An application, for example, could be told only that the user attempting to access it was a member of Clemson University which would be enough information for a digital library to know that the user could access articles under the Clemson license, but for other services, the identity provider service could send information such as the users name and email or more detailed membership information such as whether the user is faculty, staff, or a graduate or undergraduate level student.

When Shibboleth is installed as a client system, the user sets up each of the identity providers that the system should accept information from and in what order the providers should chain together. The user also selects other security settings and the language that the client and provider will use to talk to each other.

Once Shibboleth is setup and integrated with Apache the user can choose to setup secure directories in either the Apache (see Listing 3.6) or Shibboleth (see Listing 3.7) configuration file. For more information, see the Shibboleth website and documentation [16].

```
<Directory /var/www/html/secure>
        AuthType shibboleth
        ShibRequireSession On
        require valid-user
        Order deny,allow
        Allow from all
</Directory>
```

Listing 3.6: Apache Configuration for Security with Shibboleth

```
<Host name="130.127.49.8">
    <Path name="secure" authType="shibboleth" requireSession="true"/>
</Host>
```

Listing 3.7: Shibboleth Configuration to Secure a Folder

Chapter 4

VMIC

4.1 Functionality

The Virtual Machine Image Catalog (VMIC) helps to create a chain of trust between image producers, endorsers, and cloud providers. Each endorser can have their own VMIC where they publish the images that they have endorsed. (These images have been created/produced by either the endorser or another party who has sent the image to the endorser to be approved.) The cloud provider can then import the images from the VMICs of the endorsers that they trust and can create certain images for local use. Finally, the user of the cloud has access to run the images that have been loaded in the VMIC of the cloud provider since the images have been approved to run. See Figure 4.1 for a graphical representation.

4.2 Internal VMIC

The Virtual Machine Image Catalog as implemented by CERN already implements local (internal) functionality. This includes the ability to manage what machine images are able to be provisioned locally and who endorsed them (the local image repository). The catalog can also periodically push image updates out to all the servers in the cloud which are launching virtual machines so that the images are already locally distributed. The catalog also includes an RDF export to allow people to view the available images and some metadata about them or to allow some users read-only access to the system.

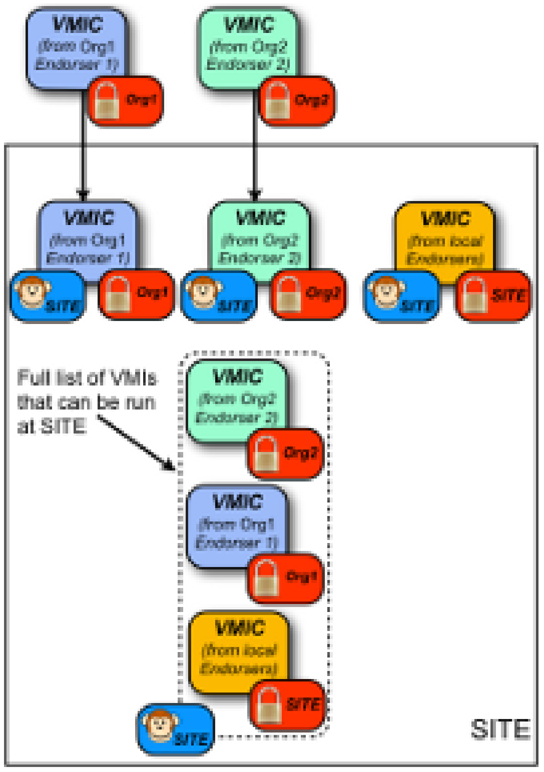


Figure 4.1: VMIC Diagram

4.3 VMI Staging

4.3.1 Internal

The virtual machine images can be staged internally in any way the site desires. Research by CERN shows that the BitTorrent distribution system works better than SCP-Wave [27]. Using several master nodes also helps to eliminate a single point of failure which could occur if only one system were to contain the VMIs from the catalog that were approved to run. Initialization of staging is caused by the VMIC, which will export the images to the hosts whenever the catalog is endorsed.

4.3.2 External

Images are cached at each VMIC so that the internal distribution can be done without having to contact an outside server. Images are directly transferred from the endorser VMIC to the client VMIC which trusts that endorser. Future work might look at using Bittorrent to transfer the images from the collection of VMICs that have already downloaded them to increase speed and reduce load on the endorser catalog.

4.4 External VMIC Issues

4.4.1 Image Validation

We must be able to verify that the received image file is valid and has not been tampered with on the endorser system or during transport. Tampering could include things like adding malicious software, back doors, or incorrect passwords.

4.4.2 Catalog Validation

We also need to verify that the server virtual machine catalog page is valid, and the client virtual machine catalog is allowed to access the images. This is a part of a process called trust negotiation. [2] First, we want to be sure that the client VMIC asks the correct server for information. Second, the server needs to ensure that the client has permission to access this information. Third, the client needs to check that the information has not been changed by a man in the middle.

4.4.3 Keeping Everything Up To Date

As security updates are released or vulnerabilities are discovered, the endorser must react by updating the virtual machine image (VMI) or removing their endorsement of the image. After which, the catalogs which subscribe to that endorser will need to update their catalog. Causing this update to happen quickly will result in a more secure system, but causing it to happen too often will waste bandwidth and cause unneeded server load for the endorser's VMIC. If images and catalog metadata were pulled together, the potentially massive size of images could cause the catalog update to take hours per endorser, which would delay the process and could cause other catalogs to fall out of date.

4.4.4 VMI Export Chaining

If one endorser endorses another endorser or if a cloud provider trusts an endorser, should those images be exported when they export their images? This question is complicated and pulls in two different needs, one, the need for image updates to quickly propagate throughout the entire system and the two, being able to have a full web of trust develop between endorsers and providers.

4.5 External VMIC Proposed Solution

4.5.1 Image Validation

Many would want to use MD5 to verify images, but since MD5 was found to have collisions in 2004 [26] and has had attacks spread as far as rogue CA certificate creation in 2009 [23], we believe that is not the answer and instead opted for the stronger SHA algorithms. Support was implemented for SHA-1, SHA-224, SHA-256, and SHA-512. Even though SHA-1 is weaker than the other algorithms, no attacks have compromised the full algorithm as of the writing of this paper, and so we have enabled its use.

4.5.2 Catalog Validation

The use of public and private keys with encryption is an easy solution to the catalog validation problem. To distribute and verify public keys, we considered three algorithms, PGP, PKI, and SHA-2. PGP "web of trust" [14] allows sites to certify each other, but, in a small network of sites, this would almost always result in directly contacting the other sites to acquire their public key. PKI (Public Key Infrastructure) [30] would allow us to build off of the existing setup to encrypt communication already online, but with this option each company would need to purchase a certificate. We have chosen to implement the SHA-2 hash algorithm, which is still strong, and recommend moving to SHA-3 when it becomes available in order to further strengthen image verification. In an effort to use what many companies already have available for credit card authentication and other secure needs, we decided to use public key infrastructure for public key distribution and verification.

To ensure that the public keys are valid, the client and server VMIC must have certain trusted Certification Authorities (CA) and check with them at runtime for expired or revoked credentials. It will also be necessary to check a local database of approved users to ensure that the public key is valid to receive information.

4.5.3 Keeping Everything Up To Date

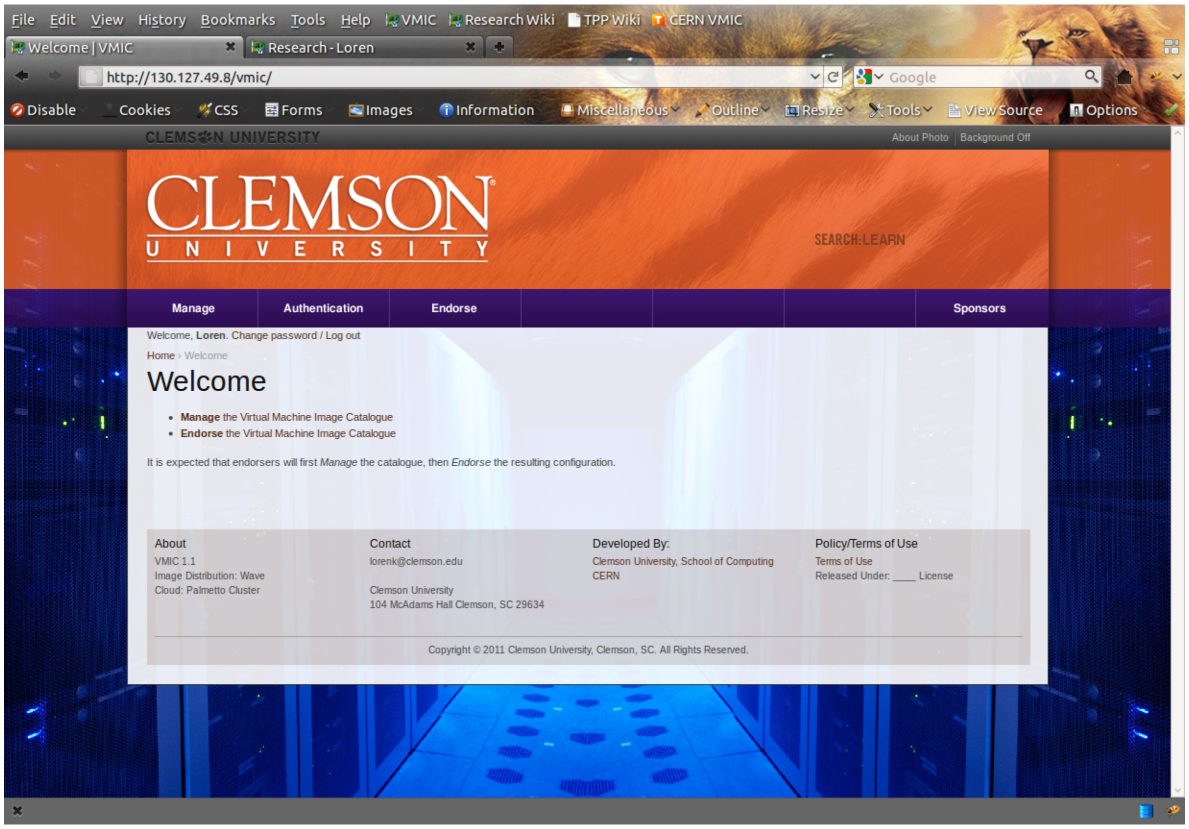
We used a cron job to periodically query and update the VMI listings from each of the endorsers. For our system, we choose to update the listing cache once an hour though the images could be updated more or less frequently depending on the needs of the system. We also implemented a second cron job that would fetch any image files that had been detected as new or updated by the cron job that fetched the VMI listings. This second cron job would fetch a maximum of one image per run and was written to allow simultaneous instances of it to run so that multiple images could be fetched at the same time. We choose to run this job every five minutes since it will simply die and not cause any internet traffic if there are no images that need to be fetched.

4.5.4 VMI Export Chaining

Because the VMIC is cached at each host, a chain of exports could cause it to take a long time for a change to make it from the top to the bottom of the chain. (For example, a chain of length five which is cached at each host for five hours could take over a day to update.) Since the VMICs do not have any limit to the chain size (e.g. master, submaster, client), we decided not to have images re-exported automatically. Future work might add the ability to export the list of trusted endorsers so that the trusted endorsers could chain without the trusted images becoming out of date. This would still have propagation issues if the top catalog added or removed a trusted endorser but would not have image synchronization issues.

4.6 Trust Negotiation

The user of any system should ensure a usable and reliable trust system, which is defined by Bertino, et al. to mean that the system has a solid theory for trust languages and algorithms which can support the needs of the system, and that the system is strong enough to limit damages caused by an intruder. [2] Our system meets the usability requirement since public/private keys and SSL are tested and often used in trust systems. The reliability requirement is met by using CAs which can reject the credentials of a compromised system to quickly prevent it from sharing and receiving data. Public/private keys prevent an attacker from gaining access to the private keys of partner systems, which prevents the attacker from being able to impersonate partner systems.


File Edit View History Bookmarks Tools Help VMIC Research Wiki TPP Wiki CERN VMIC
Welcome | VMIC
Research - Loren
http://130.127.49.8/vmic/
Google
Disable Cookies CSS Forms Images Information Miscellaneous Outline Resize Tools View Source Options
CLEMSON UNIVERSITY
About Photo | Background Off
CLEMSON UNIVERSITY
SEARCH:LEARN
Manage
Authentication
Endorse
Sponsors

Welcome, **Loren**. Change password / Log out

Home › Welcome

Welcome

- **Manage** the Virtual Machine Image Catalogue
- **Endorse** the Virtual Machine Image Catalogue

It is expected that endorsers will first *Manage* the catalogue, then *Endorse* the resulting configuration.

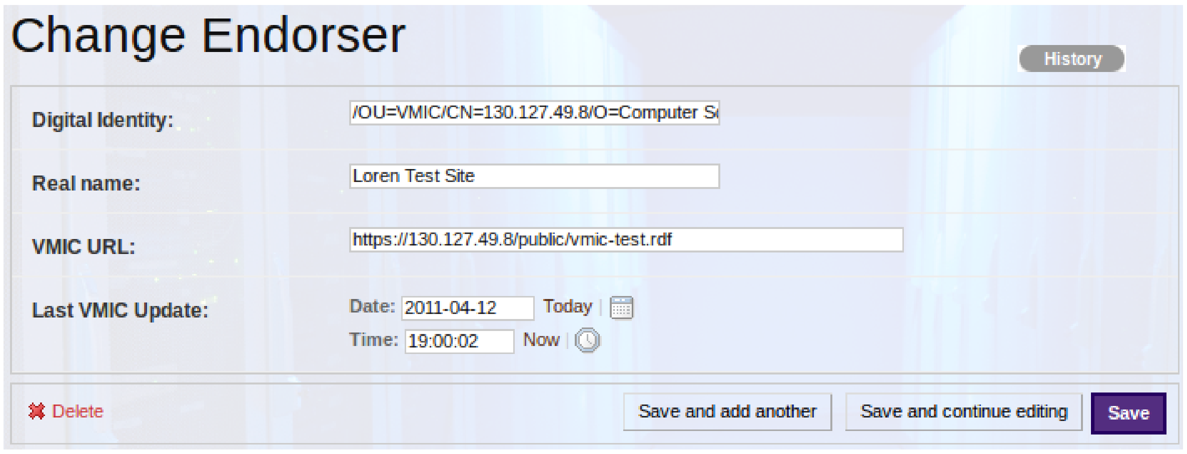
About	Contact	Developed By:	Policy/Terms of Use
VMIC 1.1 Image Distribution: Wave Cloud: Palmetto Cluster	lorenk@clemson.edu Clemson University 104 McAdams Hall Clemson, SC 29634	Clemson University, School of Computing CERN	Terms of Use Released Under: ____ License

Copyright © 2011 Clemson University, Clemson, SC. All Rights Reserved.

Figure 4.2: VMIC Home Page

4.7 VMIC Prototype Implementation

As previously mentioned, our work builds off of previous work already completed at CERN. The screenshots below will show all the features that pertain to our system. When users first login, they are brought to the home page (Figure 4.2). Administrative users can add external endorsers using the endorser management page. To add a new endorser, they simply enter the endorsers name, digital identity, and the URL for the VMIC export file (Figure 4.3). Once endorsers have been added, catalog imports can be done on the command line (or with a cron job) (Figure 4.4) or via the web interface (Figure 4.5). After the endorser catalog import is done, the images can be imported via command line or web (Figure 4.6). Adding local VMIs can also be done by filling in a few fields (Figure 4.7). The screenshot also shows custom local metadata from CERN. Once endorsed, the VMIC will export an RDF file which lists all the endorsed images (Figure 4.8) which can then be used by other VMICs to import the catalog.


Change Endorser

Digital Identity: /OU=VMIC/CN=130.127.49.8/O=Computer S

Real name: Loren Test Site

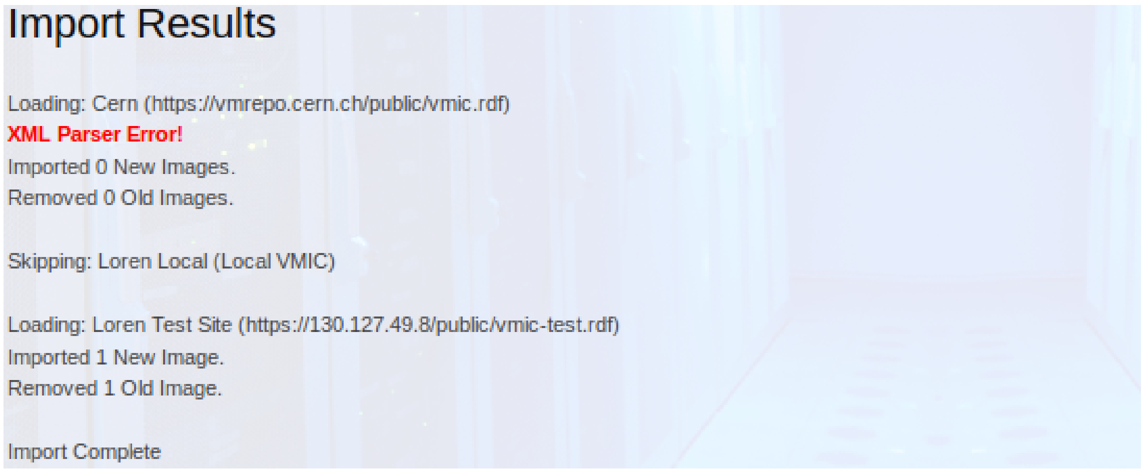
VMIC URL: https://130.127.49.8/public/vmic-test.rdf

Last VMIC Update: Date: 2011-04-12 Time: 19:00:02

Figure 4.3: VMIC Edit Endorser

```
loren@loren-research:~/workspace/VMIC/src/vmic/vmic$ ./manage.py importVMIC
Output:
Loading: Cern (https://vmrepo.cern.ch/public/vmic.rdf)
<span style='color:red;font-weight:bold;'>XML Parser Error!</span>
Imported 0 New Images.
Removed 0 Old Images.
Skipping: Loren Local (Local VMIC)
Loading: Loren Test Site (https://130.127.49.8/public/vmic-test.rdf)
Imported 0 New Images.
Removed 0 Old Images.
Import Complete
```

Figure 4.4: VMIC Import via Command Line


Import Results

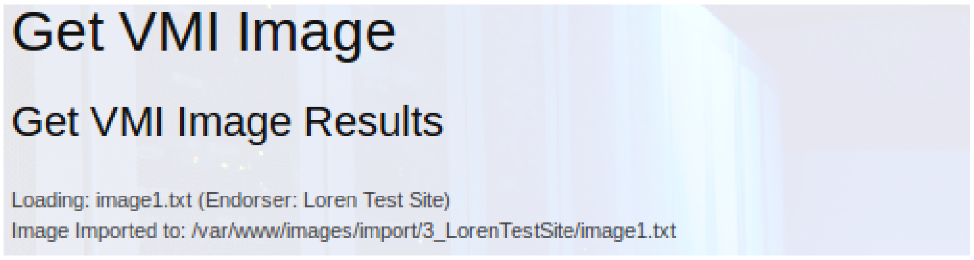
Loading: Cern (https://vmrepo.cern.ch/public/vmic.rdf)
XML Parser Error!
Imported 0 New Images.
Removed 0 Old Images.

Skipping: Loren Local (Local VMIC)

Loading: Loren Test Site (https://130.127.49.8/public/vmic-test.rdf)
Imported 1 New Image.
Removed 1 Old Image.

Import Complete

Figure 4.5: VMIC Import via Web Interface

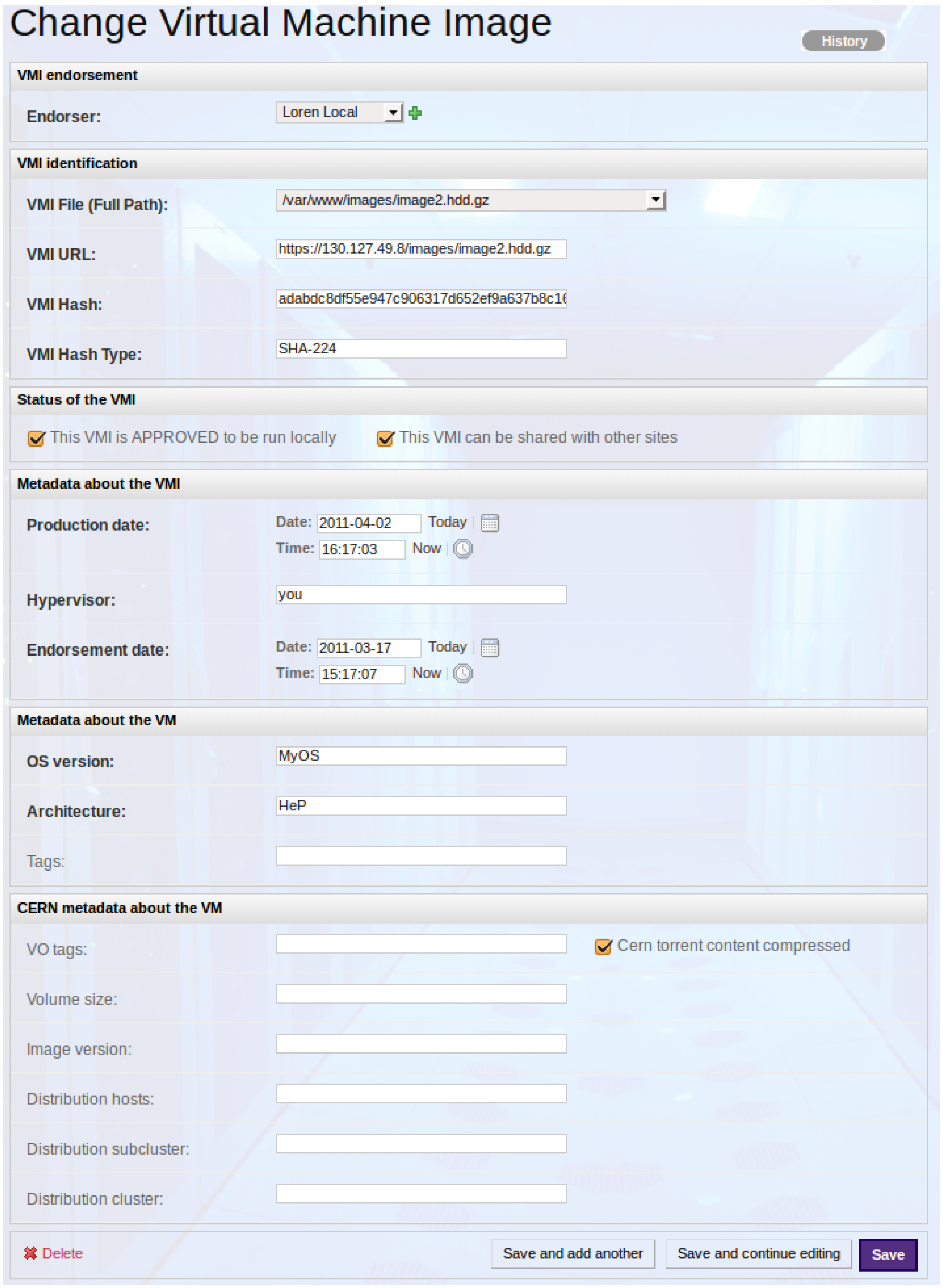


Get VMI Image

Get VMI Image Results

Loading: image1.txt (Endorser: Loren Test Site)
Image Imported to: /var/www/images/import/3_LorenTestSite/image1.txt

Figure 4.6: VMIC Get image via Web Interface



Change Virtual Machine Image

VMI endorsement

Endorser:	Loren Local

VMI identification

VMI File (Full Path):	/var/www/images/image2.hdd.gz
VMI URL:	https://130.127.49.8/images/image2.hdd.gz
VMI Hash:	adabdc8df55e947c906317d652ef9a637b8c16
VMI Hash Type:	SHA-224

Status of the VMI

✓ This VMI is APPROVED to be run locally ✓ This VMI can be shared with other sites

Metadata about the VMI

Production date:	Date: 2011-04-02 Today Time: 16:17:03 Now
Hypervisor:	you
Endorsement date:	Date: 2011-03-17 Today Time: 15:17:07 Now

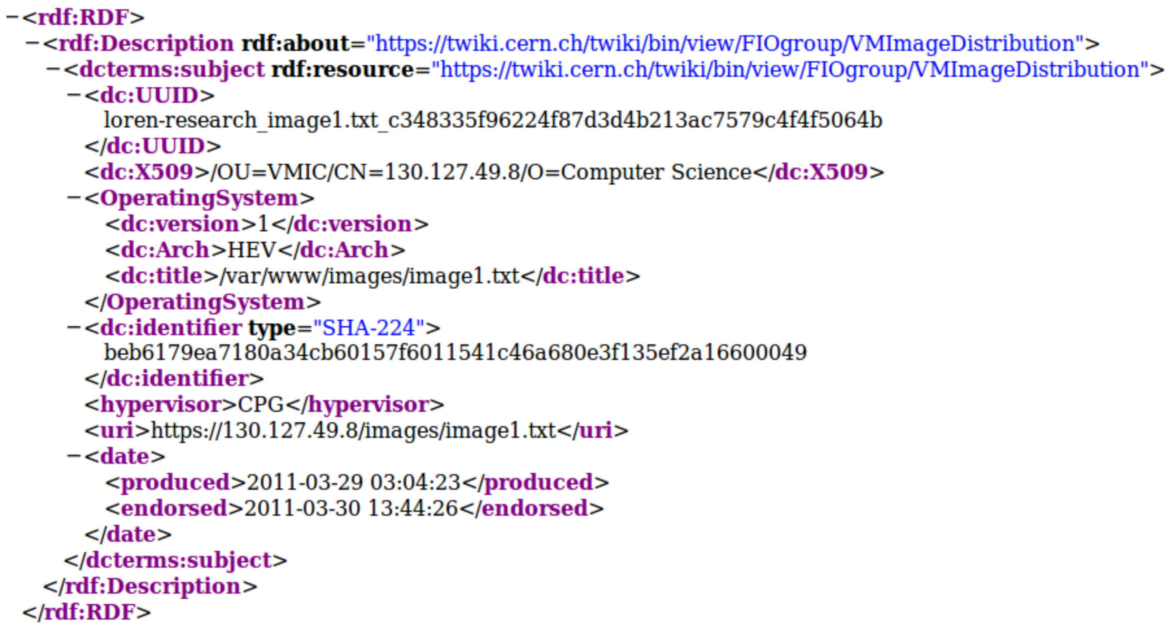
Metadata about the VM

OS version:	MyOS
Architecture:	HeP
Tags:	

CERN metadata about the VM

VO tags:		✓ Cern torrent content compressed
Volume size:		
Image version:		
Distribution hosts:		
Distribution subcluster:		
Distribution cluster:		

Figure 4.7: VMIC Edit VMI



```
-<rdf:RDF>
  -<rdf:Description rdf:about="https://twiki.cern.ch/twiki/bin/view/FIOgroup/VMImageDistribution">
    -<dcterms:subject rdf:resource="https://twiki.cern.ch/twiki/bin/view/FIOgroup/VMImageDistribution">
      -<dc:UUID>
          loren-research_image1.txt_c348335f96224f87d3d4b213ac7579c4f4f5064b
        </dc:UUID>
        <dc:X509>/OU=VMIC/CN=130.127.49.8/O=Computer Science</dc:X509>
      -<OperatingSystem>
          <dc:version>1</dc:version>
          <dc:Arch>HEV</dc:Arch>
          <dc:title>/var/www/images/image1.txt</dc:title>
        </OperatingSystem>
      -<dc:identifier type="SHA-224">
          beb6179ea7180a34cb60157f6011541c46a680e3f135ef2a16600049
        </dc:identifier>
        <hypervisor>CPG</hypervisor>
        <uri>https://130.127.49.8/images/image1.txt</uri>
      -<date>
          <produced>2011-03-29 03:04:23</produced>
          <endorsed>2011-03-30 13:44:26</endorsed>
        </date>
      </dcterms:subject>
    </rdf:Description>
  </rdf:RDF>
```

Figure 4.8: VMIC Export RDF File

Chapter 5

Conclusions and Discussion

5.1 Answering the Research Questions

In conclusion, this research shows how to distribute and trust images between cloud providers by building chains of trust between image producers, endorsers, and cloud providers. Our new changes have been added to the Virtual Machine Image Catalog in place at CERN and Clemson University and will continue to be developed. We hope to see more cloud providers implementing VMIC to encourage expansion of this research.

5.2 Recommendations for Further Research

This research may be directly furthered by the investigation of exporting the list of trusted endorsers and implications that can be caused by long chains. Research in the general area of Virtual Machine Image Catalogs may also wish to look at faster internal and external distribution methods.

Bibliography

[1] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. A view of cloud computing. *Commun. ACM*, 53:50–58, April 2010.

[2] E. Bertino, E. Ferrari, and A. Squicciarini. Trust negotiations: concepts, systems, and languages. *Computing in Science Engineering*, 06(4):27 – 34, 2004.

[3] Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Brandic. Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Gener. Comput. Syst.*, 25:599–616, June 2009.

[4] Mihai Christodorescu, Reiner Sailer, Douglas Lee Schales, Daniele Sgandurra, and Diego Zamboni. Cloud security is not (just) virtualization security: a short paper. In *Proceedings of the 2009 ACM workshop on Cloud computing security*, CCSW '09, pages 97–102, New York, NY, USA, 2009. ACM.

[5] W. Cronon. *Nature's metropolis: Chicago and the Great West.* W.W. Norton, 1992.

[6] Django. Django documentation. http://docs.djangoproject.com/en/1.2/, 2011.

[7] Django. Django tutorial. http://docs.djangoproject.com/en/1.2/intro/tutorial01/, 2011.

[8] Graham Dumpleton. modwsgi. http://www.modwsgi.org/, 2011.

[9] R.J. Figueiredo, P.A. Dinda, and J.A.B. Fortes. A case for grid computing on virtual machines. In *Distributed Computing Systems, 2003. Proceedings. 23rd International Conference on*, pages 550 – 559, May 2003.

[10] I. Foster. The anatomy of the grid: enabling scalable virtual organizations. In *Cluster Computing and the Grid, 2001. Proceedings. First IEEE/ACM International Symposium on*, pages 6 –7, 2001.

[11] I. Foster, Yong Zhao, I. Raicu, and S. Lu. Cloud computing and grid computing 360-degree compared. In *Grid Computing Environments Workshop, 2008. GCE '08*, pages 1 –10, 2008.

[12] Ian Foster and Carl Kesselman. *The Grid: Blueprint for a New Computing Infrastructure. The Grid2; 2nd ed.* Morgan and Kaufmann, San Francisco, CA, 2004.

[13] Ian Foster, Carl Kesselman, and Steven Tuecke. The anatomy of the grid: Enabling scalable virtual organizations. *Int. J. High Perform. Comput. Appl.*, 15:200–222, August 2001.

[14] Simson Garfinkel. Pretty good privacy (pgp). In *Encyclopedia of Computer Science*, pages 1421–1422. John Wiley and Sons Ltd., Chichester, UK, 2003.

[15] Google. Google app engine. http://code.google.com/appengine/, 2011.

[16] Internet2. Shibboleth. http://shibboleth.internet2.edu/, 2011.

[17] Huan Liu and D. Orban. Gridbatch: Cloud computing for large-scale data-intensive batch applications. In *Cluster Computing and the Grid, 2008. CCGRID '08. 8th IEEE International Symposium on*, pages 295 –305, May 2008.

[18] Michael Murphy, Linton Abraham, Michael Fenn, and Sebastien Goasguen. Autonomic clouds on the grid. *Journal of Grid Computing*, 8:1–18, 2010. 10.1007/s10723-009-9142-3.

[19] All Buttons Pressed. djangoappengine. http://www.allbuttonspressed.com/projects/djangoappengine, 2011.

[20] Joshua Schiffman, Thomas Moyer, Hayawardh Vijayakumar, Trent Jaeger, and Patrick McDaniel. Seeding clouds with trust anchors. In *Proceedings of the 2010 ACM workshop on Cloud computing security workshop*, CCSW '10, pages 43–46, New York, NY, USA, 2010. ACM.

[21] SIGSAC. Ccsw 2011: The acm cloud computing security workshop. http://crypto.cs.stonybrook.edu/ccsw11/, 2011.

[22] Borja Sotomayor, Kate Keahey, and Ian Foster. Combining batch execution and leasing using virtual machines. In *Proceedings of the 17th international symposium on High performance distributed computing*, HPDC '08, pages 87–96, New York, NY, USA, 2008. ACM.

[23] Marc Stevens, Alexander Sotirov, Jacob Appelbaum, Arjen Lenstra, David Molnar, Dag Osvik, and Benne de Weger. Short chosen-prefix collisions for md5 and the creation of a rogue ca certificate. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 55–69. Springer Berlin / Heidelberg, 2009. 10.1007/978-3-642-03356-8_4.

[24] Jeremy Sugerman, Ganesh Venkitachalam, and Beng-Hong Lim. Virtualizing i/o devices on vmware workstation's hosted virtual machine monitor. In *Proceedings of the General Track: 2002 USENIX Annual Technical Conference*, pages 1–14, Berkeley, CA, USA, 2001. USENIX Association.

[25] S. Toyoshima, S. Yamaguchi, and M. Oguchi. Storage access optimization with virtual machine migration and basic performance analysis of amazon ec2. In *Advanced Information Networking and Applications Workshops (WAINA), 2010 IEEE 24th International Conference on*, pages 905 –910, 2010.

[26] Xiaoyun Wang, Dengguo Feng, Xuejia Lai, and Hongbo Yu. Collisions for hash functions md4, md5, haval-128 and ripemd. Cryptology ePrint Archive, Report 2004/199, 2004. `http://eprint.iacr.org/`.

[27] Romain Wartel, Tony Cass, Belmiro Moreira, Ewan Roche, Manuel Guijarro, Sebastien Goasguen, and Ulrich Schwickerath. Image distribution mechanisms in large scale cloud providers. *Cloud Computing Technology and Science, IEEE International Conference on*, 0:112–117, 2010.

[28] Jinpeng Wei, Xiaolan Zhang, Glenn Ammons, Vasanth Bala, and Peng Ning. Managing security of virtual machine images in a cloud environment. In *Proceedings of the 2009 ACM workshop on Cloud computing security*, CCSW '09, pages 91–96, New York, NY, USA, 2009. ACM.

[29] WSGI. Wsgi. http://wsgi.org/wsgi/, 2011.

[30] R.W. Younglove. Public key infrastructure. how it works. *Computing Control Engineering Journal*, 12(2):99 –102, April 2001.