Security Best Practices for Academic Cloud Service Providers

Version 1.0

http://hdl.handle.net/2022/22123

About this Document

This document is the product of the following projects and was supported by the National Science Foundation under the grants listed. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

- The Agave Platform: NSF - OCA - SS2-SSI - 1450437
- Cornell University Center for Advanced Computing, CI-1541215 CC*DNI DIBBs: Data Analysis and Management Building Blocks for Multi-Campus Cyberinfrastructure through Cloud Federation and ACI-1548562 Extreme Science and Engineering Discovery Environment (XSEDE)
- CyVerse: NSF DBI (Division of Biological Infrastructure), DBI-0735191 and DBI-1265383
- Jetstream: NSF 1445604
- Trusted CI: The NSF Cybersecurity Center of Excellence - ACI-1547272

To check for later versions of this document, please visit:
https://trustedci.org/cloud-service-provider-security-best-practices/

Citing this Document

Please cite this document as follows:

> *Security Best Practices for Academic Cloud Service Providers. Rion Dooley, Andy Edmonds, David Y. Hancock, Richard Knepper, John Michael Lowe, Edwin Skidmore, Andrew K. Adams, Ryan Kiser, Mark Krenz, Von Welch. May, 2018. http://hdl.handle.net/2022/22123*

License

Executive Summary

Operating a cloud resource involves addressing security requirements of multiple stakeholders: primarily those of the resource operator and those using the resource. These parties may have different incentives related to security as well as different levels of acumen. Operators may at times run images whose trustworthiness is not established and provide users with privileged access within a running virtual machine or container that would be uncommon on normal computing resources. These factors combine to form an environment that, by its nature, is difficult to secure.

This document, authored as a collaborative effort between academic cloud service providers and security professionals, puts forth a set of Security Best Practices for developing and operating an academic cloud resource. Nine use cases deemed important by the authors are explored along with their security concerns. For each use case, one or more security best practices is given that balances the needs of the stakeholders and mitigates risk. The nine use cases are:

1. Disseminate Localized Best Practices
2. Ensure Image Trustworthiness
3. Provide Method to Manage User Secrets
4. Support Privileged Access within Images
5. Empower Users with Self-service DNS Management
6. Provide Method to Manage User Configurations
7. Provide Service Accounts
8. Offer Monitoring Services
9. Offer Identity and Access Management-aware Continuous Integration / Continuous Delivery Services

Table of Contents

Introduction

A “cloud resource” provides a means for users to run virtual machines or containers such that they can have a custom software stack and isolation from other users. Virtual machines or container images (henceforth “images”) can be curated and provided by the cloud resource operator, provided by the user, or provided by third parties.

Operating a cloud resource involves addressing security requirements of multiple stakeholders: those of the resource operator and those of the resource user. These stakeholders may have different prioritization of security requirements and different levels of security acumen. Operators may at times run images whose trustworthiness is not established and grant users privileged access within their running image that would be uncommon on non-virtualized computing resources. Moreover, users, with their elevated privileges, can misconfigure services, expose sensitive data or choose protocols/solutions that offer less security for the sake of installation or operating costs. These factors can lead to an environment that, by its nature, is difficult to secure.

Guiding our effort in tackling the unique security risks to academic cloud services are three basic principles, specifically: security is a shared concern between a cloud service provider and a cloud service user, neither can expect the other to fully address security; a clean delineation between cloud service provider and cloud service user of security responsibilities is critical to ensure all responsibilities are met; and the cloud service provider has the responsibility to ensure all security responsibilities are articulated and the cloud service user is educated about how to fulfill their responsibilities. These principles manifest themselves throughout the document.

Scope

The term Cloud Service Provider can imply a broad range of organizations who are not intended as the audience of the output of this document. This document focuses only on those academic organizations and projects that host and develop the system infrastructure for providing access to host virtualization technologies at the hardware or operating system level, including, Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Science-as-a-Service (SaaS) services. The focal point of this document also encompasses the organizational processes and policies in managing and securing these technologies[1]. It is a list of recommendations for security best practices within an academic cloud service provider, but the document is not intended to provide an exhaustive list, rather to identify and address those practices unique to cloud services provided in an academic or research environment.

[1] This document does not address concerns with network hardware virtualization technologies such as Software Defined Networking (SDN).

Stakeholders

As alluded to above, the security of a cloud resource affects multiple stakeholders. In an academic environment, we have identified three types, Academic Cloud Resource/Service Providers, Academic CI (Cyberinfrastructure) Providers, and Users. We list examples of each and then identify general security concerns that affect each type. Although the security concerns we include are general and not all-inclusive, our goal is to provide a basis of why our best practices are relevant to each of the various stakeholders:

- **Academic Cloud Resource/Service Providers**: providers of cloud services to the community. Atmosphere (http://www.cyverse.org/atmosphere), HUBzero (https://hubzero.org/), Jetstream (https://jetstream-cloud.org/), RedCloud (https://www.cac.cornell.edu/services/cloudservices.aspx), and science gateways are examples of Academic Cloud Resouce/Service Providers. Examples of general security concerns that may affect them, include *1. Loss/failure of operational dependencies*, *2. Increased exposure to vulnerabilities*, *3. Ineffective or untimely incident response/analysis*, *4. Loss/exposure of user data within images due to factors within provider's purview*, and *5. Loss of reputation* (see Appendix A).
- **Academic CI (Cyberinfrastructure) Providers**: developers of infrastructure (typically software) which is integrated by Resource/Service Providers into their services. Agave (https://agaveapi.co/), Cyverse (http://www.cyverse.org/), Galaxy (https://galaxyproject.org/cloud/), Globus (https://www.globus.org/), an SciGaP (https://scigap.org/) are examples of Academic CI Providers. Example security concerns that affect these types are *2. Increased exposure to vulnerabilities*, *3. Ineffective or untimely incident response/analysis*, *4. Loss/exposure of user data within images due to factors within provider's purview*, and *5. Loss of reputation* (see Appendix A).
- **Users**: researchers and others using cloud infrastructure, including, researchers, image providers, external users who may be impacted by activities within an image. Example security concerns that impact Users are *1. Loss or unpredictability of services/performance*, *2. Loss/exposure of secrets & sensitive/embargoed data*, and *3. Loss of reputation* (see Appendix B).

Prerequisites

Our assumption is that the first two stakeholders above, i.e., academic cloud resource/service and academic CI providers, have certain common IT and security practices already in place in their environment. These practices are not unique to the operations of a cloud service, and thus, are outside of the scope of this document, including: system logging and monitoring, network logging and monitoring, control of privileged access, configuration management, Identity and access management, effective physical and wireless security, hardware lifecycle practices, and an active information security program.

Recommended resources describing these practices and their implementation can be found from a variety of sources. Some examples of these are:

- CIS Benchmarks and Security Controls (https://www.cisecurity.org/cis-benchmarks/, https://www.cisecurity.org/controls/)
- Guide to Developing Cybersecurity Programs for NSF Science and Engineering Projects (https://trustedci.org/guide/)
- NIST 800 Series Special Publications 800-92, 800-95, 800-125, 800-125A-rev1, and 800-190 (https://csrc.nist.gov/publications/sp800)
- OpenStack Security Guide (https://docs.openstack.org/security-guide/)

Related Work

Several recent projects attempt to address similar issues as to our goal in this document, but we note that the cloud best practice documents we list in Appendix C are for *general* cloud service providers, not necessarily intended for the uniqueness of the academic cloud. However, there is indeed overlap between general cloud best practices and academic cloud best practices. In the cases where our best practices are similar to recommended solutions suggested by others, we note that in our practices (see References).

Terminology

Terminology and definitions used in this document:

- **Cloud Service:** An internet accessible compute service with the intended purpose of allowing users to instantiate and manage system images.
- **Image:** A data file containing the contents of a virtual machine or container.
- **Resource/Service Provider**: A operator of a cloud service**.**
- **Running image:** An instantiated image that is actively processing.
- **Security concern**: Any issue that increases security risk to the cloud service.
- **Use case**: "In software and systems engineering, a use case is a list of actions or event steps typically defining the interactions between a role (known in the Unified Modeling Language as an actor) and a system to achieve a goal." - https://en.wikipedia.org/wiki/Use_case
- **User**: A user of a cloud service. Someone instantiating and managing running images.

Use Cases, Security Concerns, and Best Practices

This section explores nine cloud use cases that the working group identifies as desirable to support, but also may increase risk to the service provider's security. For each, the use case is described and the security issues or concerns that arise are analyzed. The working group then introduces its recommended, best practices to mitigate against the risk imposed by the offering the use-case. Finally, where best practices only address certain aspects of the risk, open challenges are itemized.

Note, the list below is ordered by importance that this community bestows upon each use-case, where the lowest itemized use-cases are deemed most important. This ordering may not necessarily reflect cost, both in expertise or ease of implementation, or the potential impact reaped by the service provider enabling any of the use-cases. It is conceivable that all use-cases could be deemed “first class services”, depending on the service provider.

1. Disseminate Localized Best Practices

Use Case: Users of a cloud service will have a number of tasks that are common across cloud services, but vary in details whose implementation can lead to errors with security consequences. Some examples include:

- How to carry out non-interactive tasks (e.g., deploying code, patching images, rotating hosts, refreshing tokens, backing up data)?
- Identifying images that are ready for production versus those that are not.
- The configuration of IT automation tools (Ansible, chef, puppet, terraform, etc) for the cloud service’s specific environment.
- What minimal custom configuration of images (e.g., security groups, password setting, key generation, etc.) is needed to make them secure?
- How should development environments securely interact with the cloud service to deploy, debug, and manage service?

Security Concern: Unless users are aware of the specifics of the service providers implementation and given clear guidance in utilizing resources the service provider has made available for them, users are more likely to make mistakes in their tasks.

Example mistakes include:

- exposing credentials by hard coding secrets in code or configurations
- leaving running image unpatched
- failing to notice anomalous behavior of their applications in a timely manner by not utilizing appropriate monitoring and logging functions

Recommended Security Best Practices

- **Provide best practice documents and support services regarding the issues identified in this use case:** Service providers should provide users with documentation that specifies the provider’s implementation and key services that are available for the user to leverage. This document should be made apparent to the user when they are granted access, as well at key times during configuration when it is relevant.
- **Detect common patterns of misbehavior so that users can be alerted and educated**: Service providers not only need to detect misbehavior, but they should

attempt to educate the user through documentation as well. Ideally, this process should be automated.

2. Ensure Image Trustworthiness

Use Case: In the process of their research, users may execute images of unknown provenance on a cloud resource.

Security concerns: Executing images of unknown provenance presents several problems, including: how an operator establishes a curated image with a acceptable degree of trust, how an operator manages security updates for the operating systems and applications contained within images, and how operators manage the security updates of software which may be installed post-boot of an image (e.g. cloud-init or through user-defined metadata)?

Recommended Security Best Practice

Image registries: Image registries offer an acceptable mechanism to address the security concerns listed above. The minimum capabilities a third-party or home-grown registry solution should include, consist of (i) the ability to generate digital signatures over each image, (ii) the ability to scan for both vulnerabilities and stored sensitive information within images (detection should automatically invalidate images and alert users/admins), and (iii) robust (i.e., fine grained user and group) access control measures for managing images. Additional, desirable features include, (iv) Single Sign-on (SSO) authentication, (v) an API for automation, (vi) an interactive web application for management, including: diffs and history of image (e.g., build process, configurations, subscriptions, updates), and (vii) the ability to tag images based on provenance (e.g., system provided, user provided) or flagged as problematic (e.g., malicious) [1][2][3][5][6][7][8][9].

Example Virtual Machines:

- OpenStack's Image Service (https://docs.openstack.org/ocata/config-reference/image.html)
- VMware vCenter Server (https://www.vmware.com/products/vcenter-server.html)

Example Container Registries:

- Docker Trusted Registry (https://docs.docker.com/datacenter/dtr/2.4/guides/) - suitable for both Docker & Singularity
- Twistlock (https://www.twistlock.com/)
- RedHat/CoreOS (https://coreos.com/products/container-linux-subscription/)
- Black Duck OpsSight (https://www.blackducksoftware.com/products/opssight)
- Amazon Elastic Container Service (https://aws.amazon.com/ecs/) - suitable for Docker

Open Challenges

- Managed registries do not ensure images will always be safe, it only accredits, depending on the features of the registry, particular aspects regarding the registered images, e.g., absence of known vulnerabilities, provence, or integrity.

3. Provide Method to Manage User Secrets

Use Case: Users need a reliable solution for managing (i.e., management, provisioning, storage, and retrieval of) secrets such as API keys, signing certificates, passphrases, and other sensitive information used to manage their environment.

Security concerns: Cloud environments provide the ability to quickly and easily scale services, often in an automated fashion. Users of the environment will frequently need to store secrets such as API keys or passphrases for later use. If no method of managing secrets is provided by the service provider, the user is forced to resort to implementing their own, or worse, using none at all. Thus, basic security threats involving poor management of passphrases, keys, or other secrets could be avoided if the provider included strong and tractable secret management within their infrastructure.

Recommended Security Best Practice

Secret managers: Service providers should offer secret managers. Desirable features of a secret manager, include: API driven, mountable to the file system, support webhooks and event notifications, and make an access history available to query per stored secret. Additionally, contents should be bidirectionally encrypted with the option to store a hash and provide a challenge interface for simple authentication checks rather than retrieval of secrets [1][5][9][10].

Example Secret Managers:

- Hashicorp Vault (https://www.hashicorp.com/products/vault)
- Keywhiz (https://square.github.io/keywhiz/)
- Barbican (https://github.com/openstack/barbican)
- AWS key management service (https://aws.amazon.com/kms/)
- AWS Secrets Manager (https://aws.amazon.com/secrets-manager/)
- Stache (https://stache.utexas.edu/)

Other Solutions / Anti-patterns

- There are a variety of user-focused secret management solutions which work well, but there are barriers to using them effectively to solve the problem for cloud providers. These barriers may include issues such as lack of integration, cost, and/or missing functionality.

Example Alternate Solutions:

- Lastpass (https://www.lastpass.com/)
- Keybase (https://keybase.io/)

Open Challenges

- There is currently no consistency between service providers in what they supply, making each experience different for the user.

4. Support Privileged Access within Images

Use Case: Users of cloud services need privileged access within their running image to perform their desired tasks, e.g., running services that require registered network ports.

Security concern(s): Privileged access for a user means there is no mechanism within that virtualized environment for an operator to constrain the user. Any processes put into place by the operator can be circumvented or disabled by a privileged user. Hence, a privileged user increases risk to the service operator, other users and third parties on the Internet.

Recommended Security Best Practices

- **Limit image functionality via the network**: Firewalls or equivalent network configuration (e.g. access control lists in a router), can be used to limit the IP address ranges a running image can communicate with and the port range they can use [2][5][8][9].
- **Network monitoring, passive or active**: Network monitoring systems, e.g. Bro, Suricata, can be used to monitor what a virtual image doing via its network traffic. If unusual or bad activity is detected, they can be configured to alert a service provider, suspend a running image, or reconfigure the network to block the activity [2][5][8][9].

Other Solutions / Anti-patterns

- **Encourage users to run security-enhanced O/S**: If a user's image was implemented with, e.g., SELinux, the risk incurred by a privileged user could be mitigated. However, few users have the expertise to configure and run an enhanced O/S.
- **Disallow the granting of privileged access**: This is certainly possible, but limits the utility of the cloud service.
- **Allow privileged access and accept risk**: This is not advised since a running image is on the service provider's network address space and any consequences from misbehavior of the image are likely to be felt by the service provider.

Open Challenges

- Limiting or detecting activity that requires no network traffic within a running image with a privileged user is a challenge.

- Network traffic from a running image can use encryption or other means of obfuscation to circumvent network detection.

5. Empower Users with Self-service DNS Management

Use Case: Users need the ability to enable secure communications across their applications, infrastructure, and third-party services.

Security concerns: The canonical method for enabling secure communication is through SSL/TLS, which requires possession of a certificate for the host or service. If the process of obtaining a certificate is thorny, users may resort to less secure methods of communication. Additionally, a common variant of the canonical method has the service provider acting as a Certificate Authority (CA). However, the burden imposed in operating a CA with an acceptable level of trust is significant, especially considering that entities already exist whose primary function is just that, e.g., Comodo (https://www.comodo.com/), and Let's Encrypt (https://letsencrypt.org/).

Recommended Security Best Practice

Automatic DNS record generation & DNSaaS: To make the process in obtaining a certificate seamless for the users as well as the provider, the service provider should (i) automatically generate DNS resource records for users categorized over several fields (e.g., project, account, resource), and (ii) provide a programmable, API-driven, self-service DNS management or DNS-as-a-Service (DNSaaS) which allows users to build upon the resource records they received in (i) in order to generate new resource records describing their services. Using these latter resource records, users can then request certificates through, e.g., Let's Encrypt[2].

Example DNS Management Components:

- AtomiaDNS (http://atomiadns.com/)
- PowerDNS (https://www.powerdns.com/)

Example DNSaaService Components:

- OpenStack's Designate (https://docs.openstack.org/designate/latest/)

Other Solutions / Anti-patterns

- **User-centric process**: The canonical process for users to obtain a certificate is as follows: (i) users obtains a domain or subdomain name, after spinning up a VM and getting its hostname (ii) users publish PTR records mapping their domain to A records, (iii) request a certificate for their host or service from a Certificate Authority (CA), and (iv) possibly prove that they actually own the host and ip. However, this human-in-the-loop

[2] https://github.com/deardooley/dns-as-a-service-recommendations

process is cumbersome due to time, effort and resources lost in waiting on humans to request/issue certs, installing, monitoring, renewing, and rotating certificates every 1-2 years, and leasing a domain name and possibly purchasing a certificate authorities services.

6. Provide Method to Manage User Configurations

Use case: Users with more than one similar application to manage across images may lack consistency in the implementation of security controls and state between applications.

Security concern: As a user needs to manage more applications, the difficulty of managing the security controls can lead to security controls being incorrectly configured or unimplemented. This leaves systems open to a wider range of attacks and increases the security risk of the overall environment. Additionally, if a host's configuration is corrupted (e.g., through compromise), the user may not know how to restore that host's configuration to a known good configuration quickly and thus may avoid properly re-implementing security controls.

Recommended Security Best Practice

Provide configuration managers: Providers should offer solutions that provide some level of management over the applications they run within their images. This can be passed to the image on startup, e.g., atmosphere, or pulled in from the application or other system software running within the image [1].

Example Configuration Managers:

- Ansible (https://www.ansible.com/)
- Puppet (https://puppet.com/)
- Chef (https://www.chef.io/chef/)
- Salt Stack (https://saltstack.com/)
- Atmosphere's Ansible Instance Deployment Setup[3]
- AWS Config (https://aws.amazon.com/config/)

Open Challenges

- Securing access and storage of configuration

7. Provide Service Accounts

Use Case: Users should be able to carry out non-interactive tasks, e.g., deploying code, patching images, rotating hosts, refreshing tokens, and backing up data without losing full account access if the service providers identifies an issue during those task and thus needs to

[3] https://github.com/cyverse/atmosphere-ansible

shutdown the account in response to a potential security event.

Security concern: Users with a single account for managing all aspects of their cloud utilization increase their risk when any aspect of their account is compromised requiring the service provider is to disable all access to the user. Thus the user faces significant risk in adopting a single cloud provider for multiple projects and/or use cases as a compromise in one context will result in suspension in all contexts.

Recommended Security Best Practice

> **Service accounts**: Users should carrying out non-interactive tasks, e.g., deploying code, patching images, rotating hosts, refreshing tokens, backing up data, etc., through unique service accounts -- a user account that is created with the sole purpose of providing a specific security context to a service when the service is running.
>
> Compartmentalizing a user's account to an authorized scope mitigates the issue of a user being completely locked-down during a security incident. That is, the authorized scope, or service account, can instantly be invalidated instead, without requiring the need to take the entire user account offline. Moreover, by managing service accounts at the provider level, the provider can offer secure management of the service account credentials, thus enabling automated key rotation on images, repositories, etc.
>
> The ability to create and associate accounts is found in almost all IAM solutions in use today. For example, WSO2's Identity Server (https://docs.wso2.com/display/IS550) has a feature called Federated Authentication that fulfills this utility. Distinguished Names (DN) with attribute assignment in LDAP (https://ldap.com/ldap-dns-and-rdns/) also provides methods to create service accounts.

Other Solutions / Anti-patterns

- XSEDE community accounts – multiple, related users run jobs under one account (warehouse approach)
- Unix service accounts (insufficient scope)
- OAuth2 scopes (insufficient accessibility and customization)

8. Offer Monitoring Services

Use Case: Users need to be aware of the current state of their systems including hosts, firewalls, networks, storage services, and infrastructure.

Security concerns: Comprehensivity is a core principle of good security[4]. Users who are not aware of the current state of system operations, dependent systems, vulnerable systems and

[4] https://cacr.iu.edu/principles/

software, available resources, and activity in their environment are at at increased risk. They may also be less aware of a compromise when it occurs.

Recommended Security Best Practices

- **Provide an interface to existing monitoring solutions for users to access**: The service provider should leverage existing monitoring solutions in place by enabling users to automatically retrieve pertinent events regarding the users' account, image, applications, etc.
- **Provide notification of infrastructure service outages:** The service provider should provide a user accessible page providing infrastructure monitoring information to alert users when a planned or unplanned outage is occurring that potentially affects their services.

  Example Monitoring Interface Solutions:

  - Nagios (https://www.nagios.com/)
  - Icinga (https://www.icinga.com/products/icinga-2/)
  - AWS CloudWatch (https://aws.amazon.com/cloudwatch)
  - Vulnerability scanners such as OpenVAS (http://www.openvas.org/)
  - Cachet (https://github.com/CachetHQ/Cachet)
  - Public facing service status pages
  - Provide user API to check status of operational attributes.
  - Rackspace Monitoring as a Service for OpenStack
  - Notifying affected users/stakeholders of service outages

Open Challenges

- Providing status information to users in secure manner.
- Not all providers offer access to system status information via an API

9. Offer IAM-aware CI/CD Services

Use Case: Users will utilize CI/CD (Continuous Integration / Continuous Delivery) services, e.g., Ansible[5], GitLab Continuous Integration & Deployment[6], Drone[7], and Jenkins[8] in order to ease Development and Operations (DevOps) of their applications.

Security concern: Although users can easily setup-up CI/CD solutions, it is extremely easy to leave sensitive information exposed, thereby exposing the infrastructure on which it is running as well as the infrastructure with which the CI/CD service interacts with.

[5] https://www.ansible.com/overview/it-automation

[6] https://about.gitlab.com/features/gitlab-ci-cd/

[7] https://drone.io/

[8] https://jenkins.io/

Recommended Security Best Practice

Offer hosted, multi-tenant CI/CD services: Providers should offer hosted, multi-tenant CI/CD services that can integrate with the providers' existing AAA and monitoring solutions (see 3. Provide Methods to Manage User Secrets, and 8. Offer Monitor Services). This will reduce the accessibility of a user's sensitive information, while adding value to their clouds and increasing their accessibility.

Example CI/CD Services (w/Integrating IAM) Solutions:

- Jenkins (https://jenkins.io/)
- Tower (https://github.com/ansible/awx)
- TravisCI (https://travis-ci.org/)
- Rundeck (https://www.rundeck.com/open-source)
- CircleCI (https://circleci.com/)
- AWS Code Deploy (https://aws.amazon.com/codedeploy/)
- CodeShip (https://codeship.com/)
- TeamCity (https://www.jetbrains.com/teamcity/)

References

[1] "Application Container Security Guide - Nvlpubs.nist.gov….", https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-190.pdf. Accessed 10 May. 2018.

[2] "Ten layers of container security - Red Hat." 21 Dec. 2017, https://www.redhat.com/en/resources/container-security-openshift-cloud-devops-whitepaper. Accessed 10 May. 2018.

[3] "Docker : Whitepaper - Introduction To Container Security ...." https://confluence.cornell.edu/display/CLOUD/Docker+%3A+Whitepaper+-+Introduction+To+Container+Security. Accessed 10 May. 2018.

[4] "Docker security | Docker Documentation." https://docs.docker.com/engine/security/security/. Accessed 10 May. 2018.

[5] "Best Practices for Mitigating Risks in Virtualized Environments - Cloud ...." https://cloudsecurityalliance.org/download/best-practices-for-mitigating-risks-in-virtualized-environments/. Accessed 10 May. 2018.

[6] "The Ultimate Guide to Container Security | Twistlock." 6 Jul. 2017, https://www.twistlock.com/2017/07/06/ultimate-guide-container-security/. Accessed 10 May. 2018.

[7] "SP 800-125A, Security Recommendations for Hypervisor Deployment ...." https://csrc.nist.gov/publications/detail/sp/800-125a/final. Accessed 10 May. 2018.

[8] "Virtualization Security Checklist - isaca." http://www.isaca.org/Knowledge-Center/Research/Documents/Virtualization-Security-Checklist_res_Eng_1010.pdf. Accessed 10 May. 2018.

[9] "OpenStack Docs: OpenStack Security Guide." 24 Apr. 2018, https://docs.openstack.org/security-guide/. Accessed 10 May. 2018.

[10] "Best Practices for Cloud Security - SEI Insights - Carnegie Mellon ...." 12 Mar. 2018, https://insights.sei.cmu.edu/sei_blog/2018/03/best-practices-for-cloud-security.html. Accessed 10 May. 2018.

Appendix A: General Security Concerns For Providers

Below are a list of security concerns that could affect cloud resource and service providers when exposed to the use-cases above without implementing the best practices for mitigating the risk the use-cases introduce. It is not meant to be an all inclusive list, but merely a highlight of some of the common and impactful security concerns.

1. Loss/failure of operational dependencies

This is a high-level concern facing resource and service providers. It encompasses the loss or failure of hardware, software (e.g., hypervisor and images) or data (e.g., configuration, accounting, policies, and user filestores) that the provider's service is dependent on. The loss can be resultant from human error (e.g., mis-configuration), malicious actors (e.g, compromise), or natural phenomenon. Specific instances of the general follow:

1.1 Denial of Service (DoS, DDoS)

A malicious actor may subvert the network, or in some cases host applications, in order to inhibit access to, or operation of, provider services. Similarly, a user due to either an ambitious/overzealous workload or misconfiguration could consume more than their fair share of resources.

> **Mitigating Use-cases**: Disseminate Localized Best Practices, Ensure Image Trustworthiness, Offer Monitor Services, Provide Method to Manage User Secrets, Provide Method to Manage User Configurations, Provide IAM-aware CI/CD Services, Support Privileged Access within Images,

1.2 Execution of malicious applications within images

A malicious actor may gain control of an user application within an image, and although the actor may not be able to compromise hypervisor/management tools, they could initiate DoS attacks against the provider's CI, as well as attempt to attack other user applications.

> **Mitigating Use-cases**: Disseminate Localized Best Practices, Ensure Image Trustworthiness, Offer Monitor Services, Provide Method to Manage User Secrets, Provide Method to Manage User Configurations, Provide IAM-aware CI/CD Services, Support Privileged Access within Images,

2. Increased exposure to vulnerabilities

Users applications, whether natively insecure or unpatched, can increase the attack surface of an image, raising the risk of lost science to the users within the image, and potentially loss/failure of operational dependencies for the provider.

Mitigating Use-cases: Disseminate Localized Best Practices, Ensure Image Trustworthiness, Offer Monitor Services, Provide Method to Manage User Secrets, Provide Method to Manage User Configurations, Provide IAM-aware CI/CD Services, Support Privileged Access within Images,

3. Ineffective or untimely incident response/analysis

Incident response and analysis are essential components within a provider's security program. However, implementing and ensuring the efficacy of response procedures is challenging, for it is difficult to measure their performance outside of a security event.

Mitigating Use-cases: Ensure Image Trustworthiness, Offer Monitor Services

4. Loss/exposure of user data within images due to factors within provider's purview

Images, either through faulty software/hardware, misconfiguration, or a malicious actor (e.g., a side-channel attack) could manipulate the data within another image.

Mitigating Use-cases: Ensure Image Trustworthiness, Provide Method to Manage User Secrets, Provide Method to Manage User Configurations, Support Privileged Access within Images,

5. Loss of reputation

Loss of reputation is, usually, a side-effect inherited by a tangible security concern. Note, the service provider could suffer reputation not only from an incident within their system, but an incident that affects users' images.

Mitigating Use-cases: Disseminate Localized Best Practices, Ensure Image Trustworthiness, Offer Monitor Services, Provide Method to Manage User Secrets, Provide Method to Manage User Configurations, Provide IAM-aware CI/CD Services, Support Privileged Access within Images,

Appendix B: General Security Concerns For Users

Below are a list of security concerns that could affect users of cloud service providers when exposed to the use-cases above without implementing the best practices for mitigating the risk the use-cases introduce. It is not meant to be an all inclusive list, but merely a highlight of some of the common and impactful security concerns users may experience.

1. Loss or unpredictability of services/performance

Users expect resources to be available and operate at a certain level of performance, either through contract or simple familiarity. Deviations from those expectations become major impediments to users.

> **Mitigating Use-case**: Disseminate Localized Best Practices, Ensure Image Trustworthiness, Offer Monitor Services, Provide Method to Manage User Secrets, Provide Method to Manage User Configurations, Provide IAM-aware CI/CD Services, Provide Service Accounts, Support Privileged Access within Images,

2. Loss/exposure of secrets & sensitive/embargoed data

Through the use of faulty applications (i.e., unpatched, misconfigured, unencrypted) within their images, users may expose secrets and/or data that was not public.

> **Mitigating Use-case:** Disseminate Localized Best Practices, Ensure Image Trustworthiness, Offer Monitor Services, Provide Method to Manage User Secrets, Provide Method to Manage User Configurations, Provide IAM-aware CI/CD Services, Support Privileged Access within Images,

3. Loss of reputation

Loss of reputation is, usually, a side-effect inherited by a tangible security concern.

> **Mitigating Use-cases**: Disseminate Localized Best Practices, Empower Users with Self-service DNS Management, Ensure Image Trustworthiness, Offer Monitor Services, Provide Method to Manage User Secrets, Provide Method to Manage User Configurations, Provide IAM-aware CI/CD Services, Support Privileged Access within Images,

Appendix C: Artifacts Inventory

The following table lists documents considered by the working group in their discussions.

Name	URL
Application Container Security Guide - Draft NIST SP 800-190	https://csrc.nist.gov/CSRC/media/Publications/sp/800-190/draft/documents/sp800-190-draft.pdf
10 Layers of Container Security	https://www.redhat.com/cms/managed-files/cl-container-security-openshift-cloud-devops-tech-detail-f7530kc-201705-en.pdf
Introduction to Container Security	https://www.docker.com/sites/default/files/WP_IntrotoContainerSecurity_08.19.2016.pdf
Docker Security	https://docs.docker.com/engine/security/security/
Best practices for Azure VM security	https://docs.microsoft.com/en-us/azure/security/azure-security-best-practices-vms
Best Practices for Mitigating Risks in Virtualized Environments	https://downloads.cloudsecurityalliance.org/whitepapers/Best_Practices_for%20_Mitigating_Risks_Virtual_Environments_April2015_4-1-15_GLM5.pdf
The Ultimate Guide to Container Security	https://www.twistlock.com/2017/07/06/ultimate-guide-container-security/?ads_cmpid=751910345&ads_adid=55078119828&ads_matchtype=b&ads_network=g&ads_creative=236166466021&utm_term=docker%20container%20security%20best%20practices&ads_targetid=kwd-360439846536&utm_campaign=&utm_source=adwords&utm_medium=ppc&ttv=2&gclid=EAIaIQobChMIvYS4zILs2AIVWlcNCh3KmwQJEAAYASABEgKPwPD_BwE
Security Recommendations for Hypervisor Deployment on Servers - NIST SP 800-125A	http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-125A.pdf
AppArmor	https://en.wikipedia.org/wiki/AppArmor
SELinux	https://en.wikipedia.org/wiki/Security-Enhanced_Linux
Virtualization Security Checklist	http://m.isaca.org/Knowledge-Center/Research/Documents/Virtualization-Security-Checklist_res_Eng_1010.pdf

Security of the VMware vSphere Hypervisor	https://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/whitepaper/techpaper/vmw-white-paper-secrty-vsphr-hyprvsr-uslet-101.pdf
Security Recommendations When Deploying Citrix XenServer	https://www.citrix.com/content/dam/citrix/en_us/documents/white-paper/security-recommendations-when-deploying-citrix-xenserver.pdf
Amazon Web Services: Overview of Security Processes	https://d1.awsstatic.com/whitepapers/Security/AWS_Security_Whitepaper.pdf
Security At Linode	https://www.linode.com/security
Rackspace System Status	https://rackspace.service-now.com/system_status/
Openstack Security Guide	https://docs.openstack.org/security-guide/
Amazon AWS Secrets Manager	https://aws.amazon.com/secrets-manager/
SEI Best Practices for Cloud Security	https://insights.sei.cmu.edu/sei_blog/2018/03/best-practices-for-cloud-security.html

Appendix D: Working Group and Process

This document is a product of the working group consisting of:

- Agave Platform:□
    - Rion Dooley, Principal Investigator / Lead Developer□
- Cornell University Center for Advanced Computing:□
    - Richard Knepper, Deputy Director, CAC □
    - Resa Reynolds, Assistant Director, Systems□NSF Aristotle Cloud Federation Infrastructure Lead □
- CyVerse: □
    - Edwin Skidmore, Director of Infrastructure□
    - Andy Edmonds, Senior Systems Administrator □
- Jetstream: □
    - David Hancock, Principal Investigator, Program Director for Advanced Cyberinfrastructure□
    - Mike Lowe, Senior Cloud Engineer□
    - IU University Information Security Office. The IU UISO has agreed to review the final report if desired. Depending on the size and scope may also be willing to participate in the security review process as well. □
- Trusted CI:
    - Andrew Adams, PSC
    - Ryan Kiser, IU/CACR
    - Mark Krenz, IU/CACR
    - Von Welch, IU/CACR

This working group was formed as a result of an Engagement Application request to Trusted CI[9] . The group met over a period of six months from January, 2018 through June, 2018. It used the following process for authoring this document:

1. Determining and documenting the relevant cloud use cases.□
    a. Use cases identified by the participants in their applications (e.g. Jetstream's secure enclave for untrusted VM images) would be incorporated at this stage.□
2. Determining the relevant stakeholders from the use cases and their security concerns.□
3. Surveying existing Best Practices documents that cover container and VM security.□
4. Identify those solutions that address any of our concerns.□
5. Weighting the security concerns and determining and documenting reasonable mitigations for each.□
    a. Those identified in other Best Practices may just need to be weighted.□
6. Selecting a reasonable subset of the mitigations and using them as the basis for the Best Practices.□

[9] https://trustedci.org/application

7. Determining and documenting a set of principles for secure operation of a cloud computing resource.□
8. Documenting relevant resources for implementing the Best Practices.