SIX BEST PRACTICES FOR SECURING AWS ENVIRONMENTS
# Contents

1. Introduction 2
2. Basic AWS Concepts 3
3. Security is a Shared Responsibility 4
4. Elephant in the Room 5
5. Use Cases 6
   5.1 Secure Access for AWS Service Management 6
      5.1.1 Vaulting AWS Root Account and IAM Users 6
      5.1.2 Federated Login to AWS Services 7
      5.1.3 Control AWS Role Assignment, and Grant the Right Amount of Privilege 8
   5.2 Secure Privileged Access on EC2 Instances 8
      5.2.1 Extend Enterprise Authentication to EC2 Instances 8
      5.2.2 Secure Shared Accounts and Remote Access, Constrain Rights, Audit Everywhere 9
      5.2.3 Enforce MFA 10
   5.3 Enterprise Access to Hosted Apps 10
      5.3.1 Extending Enterprise Identities to Hosted Apps 10
      5.3.2 Enforcing MFA and Smart Card Authentication 11
6. Conclusion 11
7. Appendix A 12
This white paper is aimed at two broad audiences. As a company still exploring privileged access security (PAS) solutions for Infrastructure-as-a-Service (IaaS), maybe you have an existing on-premises Identity and Access Management (IAM) product that won’t naturally extend to the cloud without holes, fragmentation, and the increased risk of shoe-horning it into place. Maybe you believe AWS will provide all the security you need, securing the infrastructure, services, platform, OS, and data.

The second audience is existing customers who have successfully implemented Centrify Infrastructure Services to manage end users and privileged users and their access to applications and infrastructure. The challenge, however, is expanding that security model to new workloads in the cloud without increasing the risk of a data breach or impacting your audits.

Either way, this white paper will help. It complements a Centrify best practices paper1 that goes in-depth on security best practices for privileged identity management. This paper, however, looks at a specific use-case - expanding your data center workloads to the cloud. It leans heavily on AWS’ Security Best Practices document as a backdrop, showing how Centrify meets (and in many cases, exceeds) their recommendations, while providing a single privileged access security solution for public cloud, private cloud and on-premises environments.

1. Introduction

Your company operates against a well-defined set of business processes that strongly influence user behavior and technology decisions. They’ve been carefully crafted and honed. Business stability and performance is critically dependent on them. As a best practice, business processes should always drive technology decisions, not the other way around.

Now you’re looking to expand your business to the cloud. Your big concern is whether new IaaS technology and the security required to support it will end up driving wholesale changes to your business processes. Is it possible to contain disruption? Is it possible to leverage the bulk of existing processes and security technology while still realizing the substantial benefits of the cloud?

AWS broadly recommends applying security controls in the cloud as you would on-premises. How realistic is this?

Although sentiment about security in the cloud has largely evolved from “whether it’s possible” to “how to do it effectively and efficiently”, it’s not a flick of a switch. There are many moving parts and nuances of a hybrid cloud infrastructure that warrant specific attention.

The good news is that the tools exist. They’re here now. The better news is that if you’re starting fresh, you can leverage Centrify Infrastructure Services to get the job done properly. Maybe the best news is that if you’re already invested in Centrify in your data center, you already have most of what you need.

The main goal of this paper, then, is to show how Centrify is naturally designed to help you better manage privileged access security (PAS) in hybrid IT infrastructures. So, the benefits you get from running workloads in the cloud are not offset by the overheads and potential business impact involved in securing and managing it.
2. Basic AWS Concepts

The remainder of this white paper assumes you’re familiar with AWS concepts. In case you’re not, here’s a quick primer on some of the key concepts we reference.

Account Types

For our purposes, we can distinguish two types of user accounts:

AWS service users

AWS service users log into AWS through one or more portals (such as Management Console, IAM, and EC2) and interact with the underlying services. The most sensitive is the AWS root account, created automatically and granted absolute privileges (e.g., access to billing functions, management of IAM users, full control over all EC2 instances.) This account is managed through the AWS Management Console.

Other AWS service accounts can be created and granted a subset of rights to perform more restrictive tasks (e.g., managing S3 buckets for an enterprise application.) The AWS IAM service is used to manage these accounts, roles, and rights.

EC2 Instance users

EC2 instance users log into Windows and Linux instances to perform activities at the operating system level. These users exist outside the AWS IAM service (e.g., local Linux users defined in /etc/passwd; Active Directory domain accounts; federated users managed in an external IDP).

AWS Security Credentials

Not surprisingly, these are used to verify who you are and determine your access rights to AWS resources. They consist of a public and private key pair and are managed in AWS. For our purposes, they come in two flavors: Access Keys and EC2 Key Pairs:

Access Key

The private key is used to digitally sign programmatic API calls and CLI commands that access AWS services. AWS validates with the corresponding public key to approve or deny. These are long-lived keys that persist until deleted or replaced.

EC2 Key Pair:

These are used to validate access to AWS EC2 instances (i.e., log in). You create a Key Pair through the AWS IAM portal and assigned it to a new EC2 instance during its creation. For Linux instances, AWS installs the public key on the instance itself. You then supply it to your SSH client when logging into the instance. For Windows instances, the private key is used to decrypt the default local administrator account password, so it can be used via RDP to login.
3. Security is a Shared Responsibility

Don’t be mistaken in thinking that AWS will take care of all your security requirements. They won’t.

Their Shared Responsibility Model clearly demarcates where responsibility begins and ends. AWS secures the lower layers of the infrastructure stack but the you’re accountable for the remainder up to the application layer.

To help guide you, AWS not only provides tools and security bootstrapping (e.g., automatically creating an administrator account with encrypted password when you create a new Windows instance) but also a Security Best Practices document.

In this paper, we isolate the AWS best practices that are relevant to the world of PAS and describe how to implement with Centrify Infrastructure Services. While AWS recommendations are extensive, we’ve expanded several of them with recommendations for stronger security and streamlined management where it’s warranted. Finally, there are a few nuances of security in a hybrid model that don’t map 1:1 with a pure on-premises (or pure cloud) approach. We highlight our solution to those as well.
4. Elephant in the Room

Let’s start by addressing the elephant in the room. Can you reasonably leverage what you have?

Security controls for the cloud are not that different to those on-premises. Amazon tells us “Conventional security and compliance concepts still apply in the cloud.” They go on to say, “…you can architect and build a cloud infrastructure using technologies similar to and largely compatible with on-premises solutions.”

In other words, you don’t have to start from scratch. Let’s remind ourselves of the core security best practices — the ones you’ve already applied on-premises — and at a very high level how they’re equally applicable in the cloud:

1. **Common security model** - Conventional security and compliance concepts still apply in the cloud. Leverage what you’ve already got for a common security infrastructure spanning on-premises and cloud resources.

2. **Consolidate identities** - Avoid additional silos of identity that expand your attack surface, increase overhead, and lead to identity sprawl. Instead of local AWS IAM accounts and Access Keys, use centralized identities (e.g., Active Directory) and enable federated login.

3. **Ensure accountability** – shared privileged accounts (e.g., ec2-user and administrator) are anonymous. Ensure 100% accountability by having users log in with their individual accounts and elevate privilege as required. Manage entitlements centrally from Active Directory, mapping roles and groups to AWS roles.

4. **Least privilege access** - Grant users Just Enough Privilege to complete the task at hand in the AWS Management Console, AWS services, on EC2 instances and for access to hosted apps. Implement cross platform privilege management for AWS Management Console, Windows and Linux instances.

5. **Audit everything** - Log and monitor both authorized and unauthorized user sessions to EC2 instances. Associate all activity to an individual, and report on both privileged activity and access rights. Leverage Centrify Auditing & Reporting Service for session recording as well as AWS CloudTrails and CloudWatch.

6. **MFA everywhere** - Thwart in-progress attacks and get higher levels of user assurance. Consistently implement multi-factor authentication (MFA) for AWS service management, on login and privilege elevation for EC2 instances, when checking out vaulted passwords, and when accessing enterprise apps.

The above best practices will all apply to various degrees in each of the major hybrid cloud use cases described in the next section.
5. Use Cases

Let’s decompose the AWS universe into 3 main areas/use cases our customers care about:
1. Securing access to AWS management services
2. Securing access to EC2 instances
3. Enterprise access to hosted applications (applications running on the EC2 instances)

5.1 Secure Access for AWS Service Management

Control IT staff access to the AWS console to manage AWS services.

For this use case, our main concerns are around:
• Vaulting the AWS root account password and enabling access for break-glass emergencies only with MFA for identity assurance
• Leveraging Centrify Application Services for federated single sign-on (SSO) to the AWS services, obviating the need for weak passwords or long-lived access keys
• Leveraging your choice of directory service to manage user identities, control AWS role assignment, and grant the right amount of privilege

5.1.1 Vaulting AWS Root Account and IAM Users

Amazon advocates strong security, but it doesn’t preclude the use of passwords. When you sign up for AWS, you specify an email address and password that gets tied to a “root” AWS billing account. Bad guys, malware, or well-meaning admins making simple mistakes can bring your entire AWS production environment to its knees with this account.

Amazon recommends not using the root account for day-to-day interactions due to its absolute power.

This account should be vaulted in the Centrify Shared Password Management Service, to be checked out only in emergency situations. When such a situation arises, password checkout from the Centrify vault should be controlled. Centrify’s Access Request Service ensures only explicitly-approved access is granted, for a time-limited period. This aligns with the best practice of Just Enough Privilege, Just in Time. With approval, the password can be checked out for manual login.

Gartner advocates, “Minimally, use a multi-factor authentication tool supported by the cloud provider.” So, as an added layer of security, you can optionally enforce submission of a 2nd factor at the point of checkout via Centrify Adaptive MFA for Privileged Access Service.

 Optionally, or in addition, you can configure AWS MFA to prompt for a 2nd factor of authentication after entering the checked out password on the AWS login page. This is a valuable defense against a brute force attack directly at the AWS portal.

One final note about the AWS root account. Since it’s so powerful with full rights across all AWS services, AWS recommends (and Centrify supports) creating separate AWS root accounts for each environment you need to manage. This allows you to separate administration of each environment (such as the typical dev, test and prod.) Once created, you can use Consolidated Billing to further minimize usage of these accounts.

---

2Gartner publication G00311111 “Mitigate Common IaaS Security Risks With Identity and Access Management” 24 August 2016 by Felix Gaethtgens and Steve Riley
3https://docs.aws.amazon.com/awsaccountbilling/latest/aboutv2/consolidated-billing.html
Another data point around the benefits of a federated approach is from Black Hat. A 2016 session described how attackers are using Lambda scripts to automatically create a second Access Key for every IAM user (AWS permits up to two such keys to be enabled simultaneously.) Users (and administrators) are unaware of the second key unless they explicitly check. This allowed the attacker to login from anywhere at any time, re-using the same long-lived Access Key time and time again.

Ultimately, Amazon recommends⁴ that you never use the AWS root account for every-day access. Instead of a single shared account, you should delegate a subset of privileges to accounts tied to individuals. Let’s discuss that in the next section.

### 5.1.2 Federated Login to AWS Services

Centrify advocates using AWS IAM federation vs. creating local AWS IAM user accounts. Federation enables you to grant an existing user identity within your enterprise directory the appropriate rights to access any AWS service, including billing information, via AWS IAM role assignment.

AWS IAM supports SAML-based identity federation for delegated access to the AWS Management Console or AWS APIs. With identity federation, you avoid identity sprawl and having to provision and manage yet another identity silo with issues of identity duplication or synchronization.

The SAML token can carry instructions for limiting scope in AWS. i.e., the user’s Centrify roles (which may be inherited from Active Directory). These are mapped to AWS roles, effectively limiting what a user can do (e.g. zero access to billing in the AWS Management Console.) So, with federation, the attack surface is massively reduced with no local AWS account to hack into.

Federation is easily enabled with Centrify Application Services. It gives users a convenient dashboard of tiles for each target service and the added benefit of 1-click single sign on (SSO.) Note that this same solution extends beyond federated SSO to AWS services, to other SaaS applications hosted in AWS, Azure, or elsewhere as well as to custom apps. If a user hits the AWS portal directly, federation will redirect the browser to the Centrify Application Services Portal (IDP) for authentication.

For programmatic access to AWS services, Amazon provides several APIs and CLI tools. Just as with interactive login, you need to secure API access and ensure full accountability. API calls are signed using an AWS Access Key. But as noted, these long-lived keys increase your attack surface. Instead, Centrify can obtain a short-lived access key from the AWS Security Token Service (STS.) Since it has a limited life-span, your attack surface is greatly reduced and obtaining a new key upon expiration requires re-authentication by the IDP.

⁴https://docs.aws.amazon.com/awssummarybilling/latest/aboutv2/grantaccess.html
Another data point around the benefits of a federated approach is from Black Hat. A 2016 session described how attackers are using Lambda scripts to automatically create a second Access Key for every IAM user (AWS permits up to two such keys to be enabled simultaneously.) Users (and administrators) are unaware of the second key unless they explicitly check. This allowed the attacker to login from anywhere at any time, re-using the same long-lived Access Key time and time again.

5.1.3 Control AWS Role Assignment, and Grant the Right Amount of Privilege

Access governance involves managing and auditing the roles and rights that give each user access at the AWS service level and at the EC2 Instance level (i.e., once you’re logged into your EC2 server — see next section.) The goal here is to leverage your existing Centrify and Active Directory role-based access management infrastructure for your AWS resources.

In this case, Centrify can add any IT user or group from any directory service as a member of a Centrify role. Then the SAML claim will contain the Centrify role which is mapped to an AWS role for granular rights in the AWS interface. Thus, you’re able to leverage your existing Centrify infrastructure (Centrify Zones and role management) to centrally manage these rights.

5.2 Secure Privileged Access on EC2 Instances

Secure privileged access for IT staff at the OS-level.

For this use-case, our main concerns are around:

- Extending enterprise authentication to EC2 instances
- Minimizing attack surface by securing shared accounts and remote access, granting just enough privilege, and auditing everything
- Enforce MFA at session initiation, server login, and privilege elevation to stop in-progress attacks

5.2.1 Extend Enterprise Authentication to EC2 Instances

AWS instances are the work horses that drive the business. Just as they do on-premises, these virtual servers need to be properly managed and protected. Organizations need to consider both authentication to the instance, and authorization to perform specific actions during a login session (at the OS-level.)

Best practice is to leverage your existing enterprise authentication; extending it to your hosted servers without replicating or moving credentials. There’s no need for another silo to manage. As with your on-premises servers, use Centrify Identity Broker Service to broker Active Directory identities to AWS. Then users log in with their existing credentials for full accountability. All activities can be audited and tied back to a unique user.

Some customers have multiple sources of identity (e.g., Active Directory, Google Compute Cloud, Centrify Directory, generic LDAP) typically to separate customer, contractor, or outsourced IT identities from employee identities. Centrify Identity Broker Service hides this complexity making it easy to use credentials in any of those sources in place without exposing or moving them to the cloud and without requiring the Linux instance have direct connectivity to the various directories (which also strengthens security of the directory).

With Active Directory, however, you get the added benefit of leveraging your Centrify Zone model to govern access as well. The perceived challenge is how to extend your corporate Active Directory to AWS. There are multiple ways to achieve this, each with varying complexities and security implications. Centrify has published a separate blog reviewing the common options ("How to Keep Active Directory Active in a Hybrid World").

The Identity Broker Service is designed to strengthen the security of your hosted infrastructure. It eliminates the need for direct external connectivity; all outbound traffic is tunneled through HTTPS to the Centrify Identity Service platform. There’s no need to open inbound firewall ports or establish a site-to-site VPN for access to Active Directory. Additionally, Centrify’s architecture accommodates more complex scenarios across multiple AWS VPCs in different AWS Regions and Availability Zones. Centrify obviates having to establish a VPN connecting back to the enterprise for each VPC.

As with Secure Access for AWS Service Management, governance and role management is managed centrally through Centrify Infrastructure Services. Through the least-privilege model, admins log into EC2 instances as themselves and based on their Centrify roles, can elevate privileges to perform actions consistent with their job function.

5.2.2 Secure Shared Accounts and Remote Access, Constrain Rights, Audit Everywhere

For authentication, AWS bootstraps a default admin account for your new instance (e.g. ec2-user on Linux and administrator on Windows.) The EC2 Key Pair specified during instance creation is used to access the instance, but the method is different for each platform.

For Linux, AWS installs the public key of the EC2 Key Pair as an “authorized key” for the ec2-user account. Thus, you can present the private key of the pair in an SSH connection to the Linux instance using your SSH client. E.g., `ssh –i myprivatekeyfile ec2-user@publicipaddress`.

For Windows, AWS encrypts the administrator password with the public key. Through the AWS EC2 portal you request the password and you’re prompted for the corresponding private key, which AWS uses to decrypt the password and display in plaintext. Going forward, you RDP to the system with the administrator account and plaintext password.

Once again, these are highly privileged accounts. It’s not best practice to share them or use them routinely. Best practice is to vault them for emergency access and rotate the password frequently. Instead, use individual enterprise identities to log into EC2 instances. Admins can login directly via SSH with their individual, low-privileged accounts. Alternatively, they can log into the Shared Password Management portal and (based on their role) log in remotely to an instance using their enterprise credentials. Their activities can then be audited via session-recording either at the proxy or host level. All this minimizes your attack surface with vaulted shared accounts, remote VPN-less login, least-privilege sessions, and auditing all activity across the entire hybrid Windows and Linux infrastructure.

Note that standard AWS DevOps automation processes can be leveraged to automate this. At instance creation, Centrify can be installed on the instance; the instance joined to Active Directory and enrolled in the Centrify Identity Service platform; the bootstrapped account vaulted; and its password rotated as required.
Now you’re logged into the EC2 instance, we need to constrain your rights. The AWS Shared Responsibility model makes it clear that granting or denying access is the customer’s responsibility. As with on-premises servers, Centrify Infrastructure Services centrally manages your users, computer resources, roles, and rights. Users login as themselves with minimal privileges (e.g., with their individual Active Directory account.) Granular privileges are assigned via Centrify roles and applied when a user explicitly elevates privilege. This is another critical best-practice where additionally Gartner advises “Create an IAM policy model for IaaS that respects the principal of least privilege.”

Authorization is not only for users during interactive login sessions. Admins often create batch files and scripts that run automatically (e.g., cron jobs and PowerShell scripts) to perform administrative tasks. Plaintext passwords embedded in these scripts are a prime target for hackers and malware trawling for elevated privileges and ways to move laterally within the network. Best practice is to replace a password with an API call to the Centrify Infrastructure Services to request the password in real-time. Once used, the password is rotated and any potential exposure mitigated.

5.2.3 Enforce Multi-Factor Authentication

Some actions may be considered more sensitive than others, requiring additional user validation for a higher level of assurance. E.g., even with the appropriate role, you can only perform the action if you assure your identity using an out of band factor such as a mobile push notification to a pre-enrolled device. This can significantly reduce your attack surface, preventing account misuse common in the latest cyberattacks.

Centrify Adaptive MFA is a capability of the Centrify Identity Services platform. As such, it can be applied across Centrify Infrastructure Services in multiple places based on your specific security and risk requirements: login to Centrify administrative UI portals; vaulted password checkout; remote session initiation; SaaS application login; login to on-premises servers and EC2 instances; and privilege elevation on those servers. For even greater assurance, Adaptive MFA for Privileged Access can establish behavioral profiles. If the Centrify analytics engine considers an activity anomalous, a 2nd factor can be requested. The analytics approach is adaptive; it does not rely on static policies and avoids situations where MFA is either always on which can be a pain for users who are forced into an MFA process for every access attempt.

5.3 Enterprise Access to Hosted Apps

Secure access for users (internal, business partners, customers), IT staff and DevOps.

For this use-case, our main concerns are around:

- Extending Enterprise identities to hosted applications through simplified federated authentication for employees, business partners and customers
- Enforcing MFA and Smart Card authentication to satisfy stringent security requirements and prove compliance
5.3.1 Extending Enterprise Identities to Hosted Apps

This use case leverages many of the Centrify capabilities already discussed.

As a company developing hosted applications in AWS, you can SAML-enable them for federated SSO by using one of the many SAML toolkits (e.g., C#, Ruby on Rails, Python (Bottle); PHP; and Java SAML toolkits.) Again, you reap the benefits of stronger security (vs. passwords), short-lived tokens, and not having to move or replicate identities within your application (or AWS.)

Once your apps have been SAML-enabled, you can establish a trust relationship between the app and Centrify Identity Services as the IDP. Centrify enables you to integrate new applications and represent them as “tiles” on a user portal to provide 1-click SSO for the end user. A user would simply login to the Centrify Application Services Portal with their enterprise identity (e.g., in Active Directory) and simply click on the appropriate tile to be logged into the hosted app. Like previous examples, Privileged Access Request can be used to obtain approval to access a specific app.

5.3.2 Enforcing MFA and Smart Card Authentication

Like our previous examples, you can apply MFA at various places such as login to the Centrify user portal as well as upon login to an individual hosted app (clicking on the app tile in the portal.) This gives you extra identity assurance and helps thwart in-progress attacks.

Organizations that require stronger authentication can also take advantage of Centrify’s ability to support PKI-based authentication which can be used to enforce smart card or derived credentials as well as corporate device-only authentication requirements.

6. Conclusion

Enterprises are evolving. Some day in the future, all production workloads may exist purely in the cloud. However, until then we’re all faced with the prospect of a hybrid world where our applications and infrastructure is spread across our data center and the cloud.

Existing applications being migrated to the cloud or brand new ones built in the cloud must be secured. To avoid significant additional investments in time, resources, and cost you should leverage processes and technologies you already have. This means consolidating identities, centralizing identity and privilege management, controlling access, and auditing all privileged activities as you do today on-premises.

With Centrify PAS services and a set of security best practices from AWS and Centrify designed to support such hybrid models, you can achieve success quickly, cost-effectively, and with minimum disruption.

Your IaaS can truly be a natural extension of your data center, leveraging the investments you’ve already made to secure your applications and infrastructure wherever it may be.
Seven Best Practices for Securing AWS Environments

7. Appendix A

This appendix documents a subset of best practices from Amazon’s AWS Security Best Practices document (August 2016) that are relevant in the context of Centrify Privileged Access Security. The “Centrify’s Recommendations” column documents how each can be addressed or expanded with Centrify Infrastructure Services.

### AWS Shared Responsibility Model

<table>
<thead>
<tr>
<th>AWS Best Practice</th>
<th>Centrify’s Recommendations</th>
</tr>
</thead>
<tbody>
<tr>
<td>You as the customer are responsible for the security of... Amazon Machine Images (AMIs), Operating systems, Applications, Credentials, Policies and configurations.</td>
<td>Centrify provides the means necessary to secure your layers of the AWS stack. Centrify solutions integrate naturally with AWS security capabilities for a consistent management experience.</td>
</tr>
<tr>
<td>If you have higher security requirements, you can implement alternative authentication mechanisms, including LDAP or Active Directory authentication, and disable EC2 key pair authentication.</td>
<td>Centrify Identity Broker enables this higher security by extending your Active Directory, generic LDAP, and Centrify Cloud directories to AWS transparently.</td>
</tr>
<tr>
<td>You are responsible for managing your data (including classifying your assets) and for using IAM tools to apply ACL-type permissions to individual resources at the platform level, or permissions based on user identity or user responsibility at the IAM user/group level.</td>
<td>Centrify RBAC model complements AWS’ IAM model. Centrify roles and rights map to AWS roles so that privileges can be centrally managed through Centrify (or through Active Directory).</td>
</tr>
<tr>
<td>IAM is configured to help ensure secure access control of AWS resources.</td>
<td>Centrify centralizes your IAM-related functions for most AWS use cases. Centrify complements AWS IAM in situations where AWS IAM is required (e.g., local role and policy definitions.)</td>
</tr>
<tr>
<td>IAM is configured to help ensure secure access control of AWS resources.</td>
<td>As above, Centrify allows you to centralize your IAM-related across the hybrid enterprise, working seamlessly with AWS IAM where necessary.</td>
</tr>
</tbody>
</table>

### Define and Categorize Assets on AWS

<table>
<thead>
<tr>
<th>AWS Best Practice</th>
<th>Centrify’s Recommendations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-factor authentication (MFA) token is enabled to provide two-factor authentication for the root AWS account.</td>
<td>Centrify provides MFA in multiple layers across the hybrid infrastructure. Combining this with AWS MFA for the AWS root account results in comprehensive MFA coverage.</td>
</tr>
</tbody>
</table>

### Secure Your Infrastructure

<table>
<thead>
<tr>
<th>AWS Best Practice</th>
<th>Centrify’s Recommendations</th>
</tr>
</thead>
<tbody>
<tr>
<td>The best protection against resource compromise is to follow the security best practices outlined in this document. While AWS provides certain security tools to help you establish strong defenses for your cloud environment, you must follow security best practices as you would for servers within your own data center. Consistently adopt simple defense practices, such as applying the latest software patches, restricting network traffic via a firewall and/or Amazon EC2 security groups, and providing least privilege access to users.</td>
<td>Centrify allows you to seamlessly extend your on-premises Centrify security model to AWS at the AWS Service level and the EC2 in-stance level. We recommend you continue to lead with a least-privilege/privilege elevation approach. For emergency situations requiring the use of a shared account, vault this account and strictly control its use through access request/approval combined with MFA. Auditing and session recording should occur at the host level to ensure full visibility in the event a bastion/proxy is bypassed.</td>
</tr>
<tr>
<td>Malicious, illegal, or harmful activities that use your AWS resources violate the AWS Acceptable Use Policy and can lead to account suspension.</td>
<td>To avoid this situation, you need a comprehensive security solution purposefully build for hybrid cloud. You use Centrify on-premises for the strongest security. Extend this to AWS for blanket coverage of your expanded enterprise including end-end auditing and session recording of all privileged activity.</td>
</tr>
</tbody>
</table>
### Manage AWS Accounts, IAM Users, Groups, and Roles

<table>
<thead>
<tr>
<th>AWS Best Practice</th>
<th>Centrify’s Recommendations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Best practice is to create individual IAM users for each individual that needs to access services and resources in your AWS account.</strong></td>
<td>Depending on situation, Centrify advocates stronger security using SAML-based federation and Active Directory (Kerberos) or LDAP credentials, obviating the need to duplicate identities in AWS IAM and maintaining a centralized identity management model.</td>
</tr>
<tr>
<td><strong>Do not use root account credentials for day-to-day interactions with AWS.</strong></td>
<td>Vault the AWS root account in the Centrify Shared Password Management Service and control its access aggressively for break-glass situations only. Have admins login as themselves and elevate privilege as required. For emergency situations, enable Centrify’s self-service access request service and additional identity assurance via Centrify MFA and AWS MFA.</td>
</tr>
<tr>
<td><strong>We strongly discourage the use of shared user identities, where multiple entities share the same credentials.</strong></td>
<td>Centrify recommends eliminating the AWS-bootstrapped ec2-user and local administrator shared accounts. With both Linux and Windows instances joined to Active Directory, admins can log in with their individual low-privileged accounts and elevate privilege as-required. The Linux root account can be vaulted in Centrify Infrastructure Services for emergency-break glass with the additional security of using a low-privileged local Linux account as a proxy to log in over SSH.</td>
</tr>
<tr>
<td><strong>As a best practice, users should rotate their access keys on a regular basis.</strong></td>
<td>Centrify’s best practice is to avoid the use of long-lived AWS access keys, using short-lived tokens such as SAML as a more secure alternative. Such keys do not require rotation since they have a limited time to live. Note that any shared account passwords stored in the Centrify vault can be automatically rotated on a customer-defined schedule.</td>
</tr>
<tr>
<td><strong>...distributing long-term credentials to each instance is challenging to manage and a potential security risk.</strong></td>
<td>Access Keys and passwords are long-lived. They should not be used (whether vaulted or not.) Instead, Centrify supports the use of stronger short-lived tokens such as SAML.</td>
</tr>
<tr>
<td><strong>Managing credentials for multiple accounts makes identity management difficult.</strong></td>
<td>Avoid Access Key sprawl. Instead of creating users and Access Keys in AWS IAM, have users log in with existing corporate credentials (Active Directory, LDAP, or SAML.)</td>
</tr>
<tr>
<td><strong>Using IAM roles and temporary security credentials means you don’t always have to manage long-term credentials and IAM users for each entity that requires access to a resource.</strong></td>
<td>Leverage Centrify Identity Broker to support your existing identity stores (Active Directory, LDAP, Centrify Directory) instead of creating a new silo in AWS IAM. AWS IAM roles can still be leveraged to ensure entitlements (Centrify and/or Active Directory) are mapped to AWS entitlements.</td>
</tr>
<tr>
<td><strong>We recommend you activate MFA for your AWS account and your IAM users to prevent unauthorized access to your AWS environment.</strong></td>
<td>Centrify recommends vaulting the AWS root account password for break-glass emergency use only. If required, enforce access request/approval to check out the password. For additional identity assurance, prompt the user for a second factor to check the password out. Finally, on login at the AWS Management Console, use AWS MFA in case (e.g.,) of a direct brute force attack on the AWS web site.</td>
</tr>
<tr>
<td><strong>Access keys are used to digitally sign API calls made to AWS services.</strong></td>
<td>Centrify does not recommend long-lived access keys (vaulted or otherwise.) When making an API call, Centrify can request a short-lived access key from AWS’ STS to sign the API call.</td>
</tr>
<tr>
<td><strong>You can use IAM roles by creating an identity broker that sits between your corporate users and your AWS resources to manage the authentication and authorization process without needing to re-create all your users as IAM users in AWS.</strong></td>
<td>Centrify provides an Identity Broker that supports user login to EC2 Linux instances using Active Directory, LDAP, or Centrify Cloud credentials. This obviates the need to duplicate identities in AWS IAM.</td>
</tr>
</tbody>
</table>

Authorization at the EC2 instance OS level is managed on the host via Centrify Privilege Elevation Service.
### Manage AWS Accounts, IAM Users, Groups, and Roles

<table>
<thead>
<tr>
<th>AWS Best Practice</th>
<th>Centrify’s Recommendations</th>
</tr>
</thead>
<tbody>
<tr>
<td>IAM groups are a powerful tool for managing access to AWS resources.</td>
<td>Rather than AWS becoming another silo to be managed, Centrify enables a single common source (Active Directory) of identity and policy for centralized management and simpler attestation.</td>
</tr>
<tr>
<td>Using IAM roles and temporary security credentials means you don’t always have to manage long-term credentials and IAM users for each entity that requires access to a resource.</td>
<td>Centrify enables use of Active Directory Groups. These groups map to Centrify roles that in turn map to AWS roles. Thus, entitlement can be centrally managed across the hybrid enterprise from Active Directory.</td>
</tr>
<tr>
<td>You can use IAM roles by enabling IAM users from another AWS account to access resources within your AWS account. This process is referred to as cross-account access.</td>
<td>Centrify enables the use of identity federation to solve this problem. Simply add the user to a role in the Identity Service platform. This has the net effect of centralizing identity management at the IDP instead of being split between the IDP and SP.</td>
</tr>
<tr>
<td>As a best practice, users should rotate their access keys on a regular basis.</td>
<td>Avoid Access Key sprawl. Instead of creating users and Access Keys in AWS IAM, have users login with existing in-place credentials (Active Directory, LDAP, or SAML.)</td>
</tr>
<tr>
<td>Ensuring that users have appropriate levels of permissions to access the resources they need, but no more than that, is an important part of every ISMS.</td>
<td>Centrify Privilege Elevation Service enforces a least-privilege access control model at the host level, avoiding situations where the vault and its proxy-based command filtering is bypassed by someone attacking the instance directly.</td>
</tr>
<tr>
<td>There are two primary types of credentials associated with these identities: (1) those used for sign-in to the AWS Management Console and AWS portal pages, and (2) those used for programmatic access to the AWS APIs.</td>
<td>(1) Centrify recommends: vaulting the AWS root account for emergency checkout only; managing delegated admin accounts centrally in your existing enterprise directory (e.g., Active Directory); and having users log in via SAML-based federation. (2) Centrify can request a short-term access key from AWS STS for API requests instead of a long-lived access key.</td>
</tr>
</tbody>
</table>

### Managing OS-level Access to Amazon EC2 Instances

<table>
<thead>
<tr>
<th>AWS Best Practice</th>
<th>Centrify’s Recommendations</th>
</tr>
</thead>
<tbody>
<tr>
<td>In the shared responsibility model, you own the operating system credentials but AWS helps you bootstrap the initial access to the operating system.</td>
<td>Vault the bootstrapped accounts (e.g., ec2-user and administrator) and allow access for break-glass emergencies only, under access request and MFA constraints. Log in using enterprise (e.g., Active Directory) accounts.</td>
</tr>
<tr>
<td>Even with resource permissions, accidental deletion by a privileged user is still a threat (including a potential attack by a Trojan using the privileged user’s credentials), which illustrates the importance of the principle of least privilege.</td>
<td>To reduce the risk of such issues, users should login as themselves (for full accountability) with least-privilege and elevate privilege via roles as required. Centrify Privilege Elevation Service also helps prevent Pass the Hash attacks by avoiding the use of administrator hashes.</td>
</tr>
</tbody>
</table>

### Protect Data in Transit

<table>
<thead>
<tr>
<th>AWS Best Practice</th>
<th>Centrify’s Recommendations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use SSH version 2 using non-privileged user accounts.</td>
<td>Infrastructure Services allow users to login to AWS instances using individual, non-privileged accounts and then use privilege elevation at the OS level. Traditional vault solutions encourage day-to-day login with full-privileged shared accounts.</td>
</tr>
</tbody>
</table>
## Secure Your Operating Systems and Applications

<table>
<thead>
<tr>
<th>AWS Best Practice</th>
<th>Centrify’s Recommendations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Password protect the .pem files on user machines.</td>
<td>Centrify’s best practice is to disable use of EC2 Key Pairs, using Active Directory or LDAP credentials instead to log into EC2 instances.</td>
</tr>
<tr>
<td>Disable root API access keys and secret key.</td>
<td>Long-lived access keys are not recommended (vaulted or otherwise.) When making an API call, Centrify can request a short-lived access key from AWS’ STS to sign the API call.</td>
</tr>
<tr>
<td>Give users the minimum privileges they need to carry out their tasks. That way, even if a user accidentally launches an infected executable, the impact on the instance and the wider cloud system is minimized.</td>
<td>Vaulting highly privileged shared accounts (such as ec2-user) does not prevent such mistakes. These accounts should be eliminated; users should login with individual (for full accountability) low-privilege accounts and elevate privilege via roles as required.</td>
</tr>
<tr>
<td>Use bastion hosts to enforce control and visibility.</td>
<td>Although Centrify can provide one, our best practice is to enforce control with an agent and audit on every EC2 instance. This can’t be bypassed whereas a central bastion host/proxy can.</td>
</tr>
<tr>
<td>Regularly run least privilege checks using IAM user Access Advisor and IAM user Last Used Access Keys.</td>
<td>User privileges at the host level are managed centrally in your enterprise Active Directory infrastructure. Centrify best practice is to eliminate Access Keys and leverage more secure forms of authentication such as Active Directory-based login or SAML.</td>
</tr>
</tbody>
</table>

## Manage Security Monitoring, Alerting, Audit Trail, and Incident Response

<table>
<thead>
<tr>
<th>AWS Best Practice</th>
<th>Centrify’s Recommendations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manage commands that can be used during sessions. For interactive sessions like SSH or appliance management, or AWS XLI, such solutions can enforce policies by limiting the range of available commands and actions.</td>
<td>Many solutions filter commands at a (bastion) host level. This is insufficient; it can be bypassed by directly compromising an instance and with malware on the instance. Centrify enforces access controls at the host level and so is not affected when the bastion is bypassed.</td>
</tr>
<tr>
<td>Restrict leapfrogging or remote desktop hopping by allowing access only to target systems.</td>
<td>Remote login to instances through the Centrify Secure Remote Access Service surgically places the user on the target server. It does not require use of a VPN that could expose the wider network. Centrify Infrastructure Services can restrict leapfrogging to target servers through RBAC at its portal and can prevent leapfrogging at the host/OS level including IPSec-based controls to isolate critical server-to-server communication.</td>
</tr>
<tr>
<td>...consider using a privilege escalation gateway to manage access control logs and authorization.</td>
<td>Infrastructure Services manage authorization at the host level and audit logs and recorded sessions. Gateway or proxy-based command filters can be bypassed and do not control access for (e.g.) malware running on the instance itself. If bypassed, without host-based controls, you cannot control access or record sessions.</td>
</tr>
<tr>
<td>We recommend configuring the following areas for logging and analysis: Actions taken by any individual with root or administrative privileges; access to all audit trails; invalid logical access attempts; use of identification and authentication mechanisms; initialization of audit logs; creation and deletion of system level objects.</td>
<td>Centrify audits all user activities across the hybrid enterprise and ties all activity back to a unique individual user for 100% accountability. Privileged login sessions can be video recorded either at the bastion host/proxy level or at the host level to avoid being bypassed by a direct login to the instance. All attempts to login to Centrify UIs, AWS portals, and EC2 instances as well as all privilege elevation attempts can be audited and recorded.</td>
</tr>
<tr>
<td>The shared responsibility model requires you to monitor and manage your environment at the operating system and higher layers.</td>
<td>Centrify auditing and session recording provides end-end visibility of “who has access to what” and “who did what”. It can record sessions at a proxy and OS-level. Centrify also integrates with leading commercial SIEM solutions such as Splunk, IBM QRadar, and HP ArcSight.</td>
</tr>
</tbody>
</table>
As the only industry recognized leader in both Privileged Identity Management and Identity-as-a-Service, Centrify provides a single platform to secure every user’s access to apps and infrastructure in today’s boundaryless hybrid enterprise through the power of identity services. This is the Next Dimension of Security in the Age of Access.

Founded in 2004, Centrify is enabling over 5,000 customers, including over half the Fortune 50, to defend their organizations. Centrify is a privately held company based in Santa Clara, California. To learn more visit www.centrify.com. The Breach Stops Here.

Centrify is a registered trademark and The Breach Stops Here and Next Dimension Security is a trademark of Centrify Corporation in the United States and other countries. Other trademarks mentioned herein are the property of their respective owners.