HARDENING GUIDE

3D Graphics Rendering Workflows

Microsoft Azure

Revision 3

August 2020

Executive Summary

Microsoft has engaged Independent Security Evaluators (ISE) to review 3D graphics rendering architectures and workflows using the Microsoft Azure cloud computing environment and to establish an Azure-specific hardening guide for the media and entertainment (M&E) industry.

The M&E industry wishes to use Azure with both its internal and vendors' software systems to increase the throughput, security, scalability, and cost-efficiency of its film production activities while improving the deployment's security posture. This document is intended for administrators who may be implementing rendering workflows and describes hardening steps specific to that workflow as well as general guidance that can be applied to any Azure deployment.

These security controls were developed after hands-on evaluations of relevant Azure services and access to all publicly available documentation. This guide is current as of August 2020. Changes to Azure after this date may invalidate certain recommendations or introduce new concerns. Furthermore, users are responsible for understanding their cloud deployments and associated security risks.

ISE recommends that studios consider the security controls described in this document and perform independent deployment assessments of individual asset management systems in the future.

Table of Contents

Introduction

The Microsoft Azure cloud platform provides its customers with an enterprise-grade, scalable, a high-reliability computing environment that operates more cost-effectively than hardware deployed in a traditional data center. ISE's engagement sought to identify effective security enhancing capabilities and provide guidance on the configuration of these features so that media and entertainment enterprises can securely use the Azure cloud platform for visual effects rendering of pre-release theatrical content and other high-value assets.

Microsoft Azure allows customers access to computing power without a substantial investment in on-premises hardware. Among the workflows that may benefit from cloud computing is the rendering of three-dimensional graphics. Rendering is a broad field across the media and entertainment industry encompassing modeling products such as Blender, NUKE, Autodesk Maya and 3ds Max; and rendering engines such as V-Ray RT, Octane, Redshift, and Blender.

Broadly, a general CPU or GPU-accelerated render job may be deployed in an interactive or batch mode of operation. Cloud rendering nodes may render locally-available modeling data stored in the cloud (e.g., in batch mode, or on behalf of interactive modeling software running in the cloud), or alternatively, they may receive network commands from remote interactive rendering software running outside of the cloud environment. Our concerns and recommendations aim to address batch mode processing in this revision.

Network speed and business requirements mean that the deployments we have shown here may not be appropriate or applicable for certain use cases; however, the security recommendations should be considered and applied, if possible. In addition, though we used Blender as our reference rendering tool, our concerns and recommendations are focused on Azure services and mostly rendering tool agonistic.

Securing Azure VFX Rendering and related Workflows

ISE reviewed 6 VFX rendering related reference deployments to establish a set of security controls. 4 out of the 6 reference workflows were created by ISE while the others were provided as "turn-key" setup by vendors or Microsoft engineering. The reference implementations provided hands-on experience with 3D graphic rendering software packages, rendering schedulers, virtual desktop infrastructure, hybrid cloud caching solutions, and Azure cloud services. ISE used the reference implementations to develop security controls for Azure cloud services. The security recommendations in this guide aim to address 4 different common rendering scenarios: on-premises, hybrid, hybrid with Azure Batch, and cloud only.

On-Premises VFX Rendering with Blender Network Render

Blender is free, and open-sourced, professional 3D computer graphics software product used for creating animated films, VFX, art, 3D printed models, interactive 3D applications and video games. Blender is used by individuals and small studios for photorealistic rendering, fast rigging, realistic simulations, and game creation. Blender includes a Python shell for custom extensions to the tool for creating custom workflows, such as network rendering or plug-ins to scale up/out using cloud services. We created this setup for reference and network traffic analysis.

Blender may execute on a single machine provisioned with a CPU and/or GPU, or using an arbitrary large cluster of machines using the network rendering feature through a local or WAN. We planned on running rendering software on local machines only for this implementation. For more information, see https://wiki.blender.org/index.php.

Figure ***1*** depicts a possible deployment.

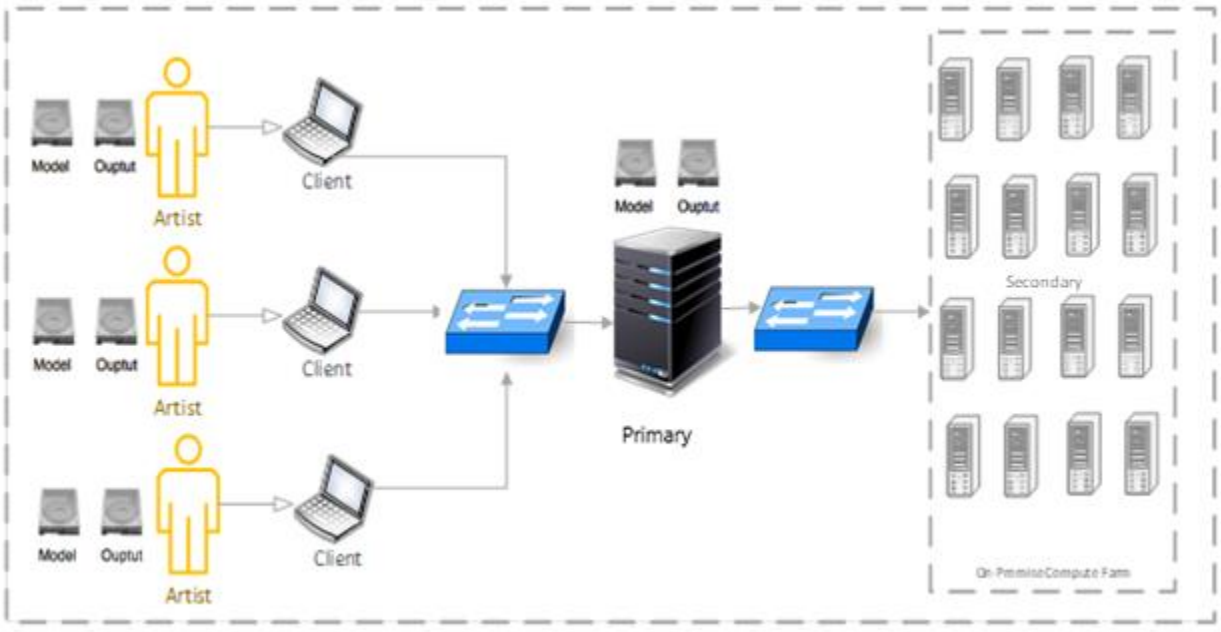

Figure 1. Sketch of on-premises local networked rendering deployment.

Security-relevant challenges pertaining to a local Blender network rendering system involve secure file transfer, local storage on compute nodes, operating system images, management of the master portal, and compute machine farm network.

Recommendations

We recommend considering the following when deploying an on-premises networked rendering deployment:

- Establish a separate network for each production.
    - Establish a new master node for each production.
    - Use network segmentation to control access to the network farm for each production.
    - Use inbound and outbound firewall rules to restrict data traffic between network segments.
- Enable Transport Layer Security for communication between clients, master and compute nodes.
    - If the deployment *must* employ non-secure communications, then use VPN technologies to encapsulate this traffic as a defense-in-depth mechanism, both to protect the traffic from eavesdropping as it travels over the Internet, and avoid exposing the server software to incoming connections from the Internet.
- Isolate the render master and nodes from the Internet as much as possible, to reduce the environment's attack surface.
    - Do not expose administrative ports (e.g., Windows Remote Desktop, or Secure Shell) to the Internet directly. Instead, use a bastion host or VPN to administer the render master and nodes. This avoids the need to expose each rendering node to incoming Internet traffic, and reduces the urgency in applying patches to administrative software in the event of vulnerability disclosure.
    - Consider using a managed or unmanaged controlled server (e.g. Microsoft WSUS) to distribute updates to render nodes, or host a local repository mirror for Linux-based nodes. This eliminates the need to allow direct outbound Internet connections from these nodes.
    - Whitelist outbound traffic from render nodes to needed destinations.
    - Utilize security services such as Azure Security Center to monitor deployed resources.
- Implement a central management scheme for the render nodes.
    - Consider Azure Active Directory, On-Premises Active Directory (Windows) or LDAP+Kerberos (Linux) to ease user provisioning and removal from a central location.
    - Consider automated deployment and configuration management such as Group Policy (Windows) or Ansible, Salt, Chef, and Puppet.
- Set up monitoring and alarm systems to monitor nodes.
    - Consider logging system activity, load, access and failure.
    - Consider setting up monitoring alarms to monitor all nodes.

Hybrid VFX Rendering with Blender Network Render

Blender may be executed on a single machine provisioned with a GPU or by using an arbitrary large cluster of machines using the network rendering feature. A cluster of machines can be composed using on-premises and cloud compute resources. Blender network rendering is designed to operate over multiple computers connected through a local network or WAN, see https://wiki.blender.org/index.php/Dev:Source/Render/Cycles/Network_Render.

Compute resources within Microsoft Azure and the local network are accessed using Windows Remote Desktop or similar remote access software. The rendering nodes connect to the Blender render master using

Blender's network rendering protocol. In this setup, we configured the rendering virtual machines in the cloud using the Azure portal.**Error! Reference source not found.** Figure 2 depicts a sketch of this deployment.

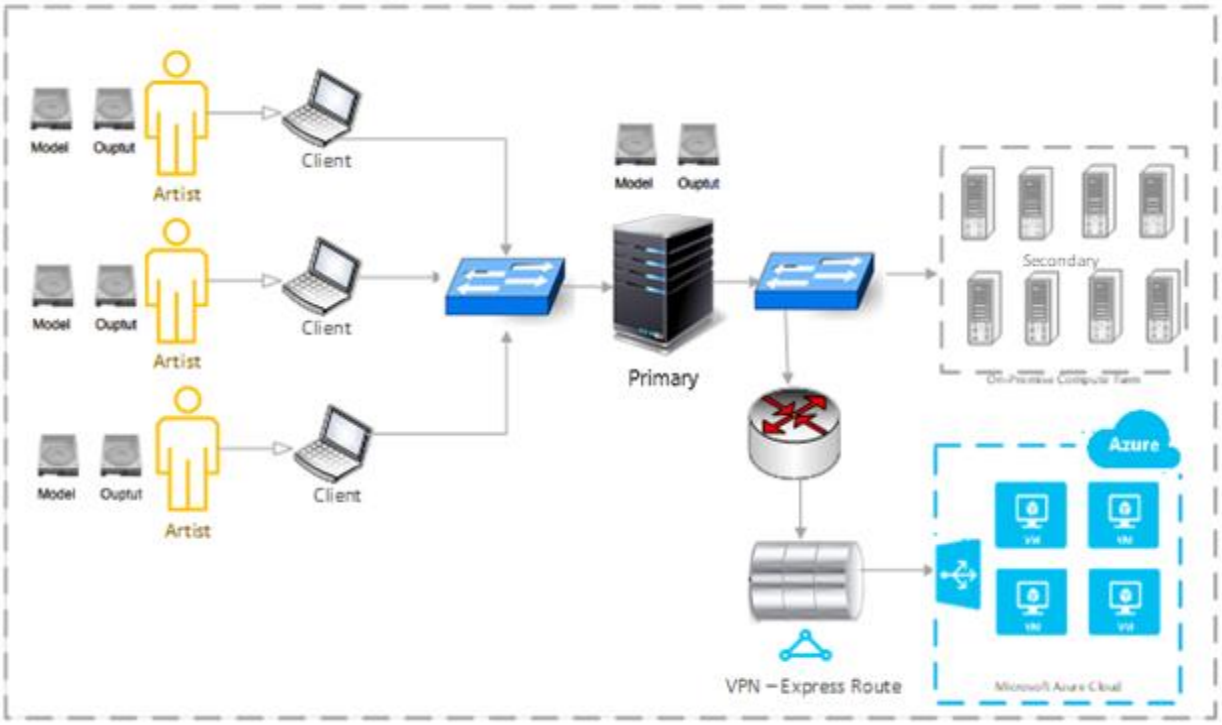

Figure 2. Sketch of hybrid rendering deployment.

Recommendations

We recommend considering the following when deploying such a "hybrid" rendering system. These recommendations supplement the ones provided in the first implementation above.

- Ensure cloud rendering nodes are configured with a hardened image.
- Ensure access to the data transfer between on-premises nodes and the cloud is secure.
    - File transfer protocols should be run over TLS version 1.2 and above for secure file transfers (i.e. SFTP).
    - Harden the protocol or product as appropriate for an environment hosting pre-release media content.
        - Use an SSH Certificate Authority either directly or through a privileged access management solution to prevent the need for Trust on First Use (TOFU) keys.
        - Apply normal SSH hardening practices for SFTP/SCP-only users, such as disabling shell access and port forwarding.
        - Use SSH asymmetric key authentication in place of password authentication when practical.
        - For TLS-based protocols (such as FTP over TLS) ensure that the client software correctly and effectively verifies the server's hostname and certificate.

        - Where possible, use Multi-factor authentication such as using time-based codes or cryptographic authenticators in addition to username and password combinations.
    - Restrict connections to file transfer software by source IP ranges, if possible.
- Consider grouping cloud resources as a production render resource element for abstracted management.
- Group together cloud storage, virtual machines, VNet, virtual appliances and IAM as a single resource for each production.
- Ensure a secure connection between on-premises and cloud resources.
    - Azure ExpressRoute is the preferred option for providing the most consistent performance and reliability
    - Commodity point-to-point or VPN link technologies may also be used
- Ensure cloud rendering nodes are part of a uniform network security group.
    - Network security groups provide control over network traffic flow at layer 4.
- Ensure cloud rendering nodes are segmented into frontend and backend subnets. Frontend subnets are public internet-facing, while backend subnets are available to only internal nodes and are not public internet-facing.
    - Consider instantiating a layer 7 virtual networking appliance (e.g. firewall) to protect nodes on the external subnets.
    - Consider bridging front-end and backend network segments using a virtual networking appliance (e.g. firewall).
- Ensure isolation in time and space between various concurrent productions.
- Ensure isolation between content rendering network zones and data input and output nodes.
    - Create content input and output network zones on the on-premises network. These network zones should not be accessible to internet-facing services or cloud rendering resources.
    - Local firewalls should be configured to restrict traffic on the network zone where original and final content is stored.
- Sanitize all cloud and local resources at the completion of network production.

Cloud VFX Rendering using Azure Batch

For our example workflow, we used Blender, a 3D creation suite that encompasses the entire 3D pipeline. Blender may be executed on a single machine provisioned with a CPU and/or GPU or by using an arbitrary large cluster of machines using the network rendering feature. However, the principles demonstrated in our example apply to any cloud VFX workflow and are not limited to Blender-based ones. In this deployment, we use the Microsoft-provided Azure plug-in for Blender to create a complete on cloud rendering system. The Batch Apps Blender plug-in sample is designed to allow users to submit rendering jobs to their own cloud-based rendering service using the Azure Batch Apps Service.

Azure Batch is a platform service for running large-scale parallel and high-performance computing (HPC) applications efficiently in the cloud. Azure Batch schedules compute-intensive work to run on a managed collection of virtual machines and can automatically scale compute resources to meet the needs of your jobs. A client provides an application to the Batch service and the Batch service provides job management,

resourcing, scheduling, disaster recovery, data movement, dependency management and all the other plumbing required to run jobs in the cloud.

For the rendering of images for 3D scenes the Azure Batch APIs can be used to interact with the Batch service which schedules intrinsically parallel work on a pool of compute nodes. A pool of compute nodes is the "render farm" that provides tens, hundreds, or even thousands of cores. In this reference example, we setup a pool using a few virtual machines. The following diagram shows a common Batch workflow, with a client application or hosted service using Batch to run a parallel workload.

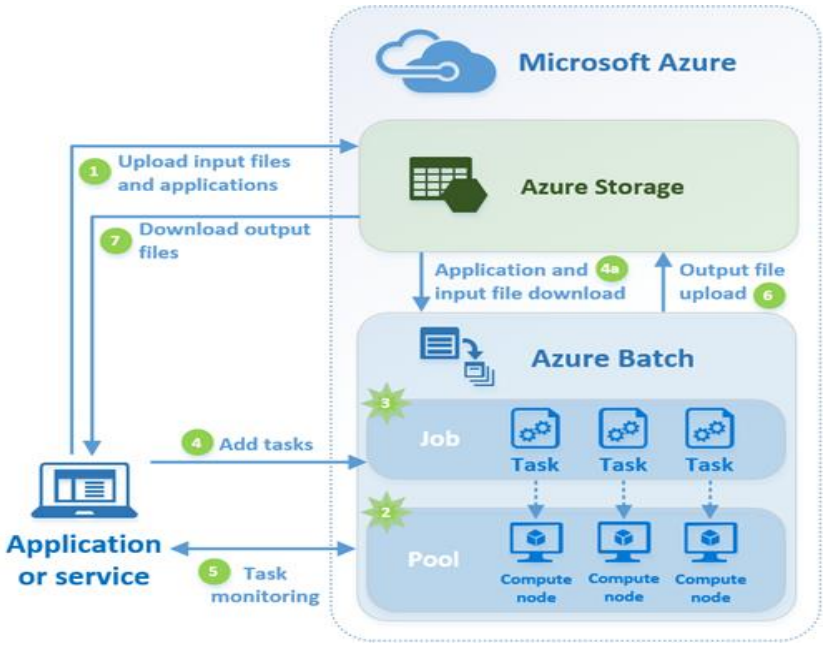

Figure 3. Sketch of hybrid rendering deployment.

The following is a list of steps required to use Azure batch for 3D rendering:

1. The artist uploads the input files and the Blender application to the Azure storage. The input files in this scenario are the 3D scenes to be rendered.

2. Create a Batch pool of compute nodes using the Azure portal. These nodes are the virtual machines that will execute our tasks. The pool is configured to automatically scale in response to the workload. Auto-scaling dynamically adjusts the number of compute nodes in the pool.

3. Create a Batch job to run the workload on the pool of compute nodes.

4. Add tasks to the job. The Batch service automatically schedules the tasks for execution on the to compute nodes in the pool. Each task uses the Blender application to render the frame.

5. Before a task executes, it can download an input file needed for rendering from Azure storage (see step 1).

6. As the tasks run, Azure Batch is used to monitor the progress of the job and its tasks. The client application or service communicates with the Batch service over HTTPS.

7. As the tasks, complete, they upload their result data to Azure Storage.

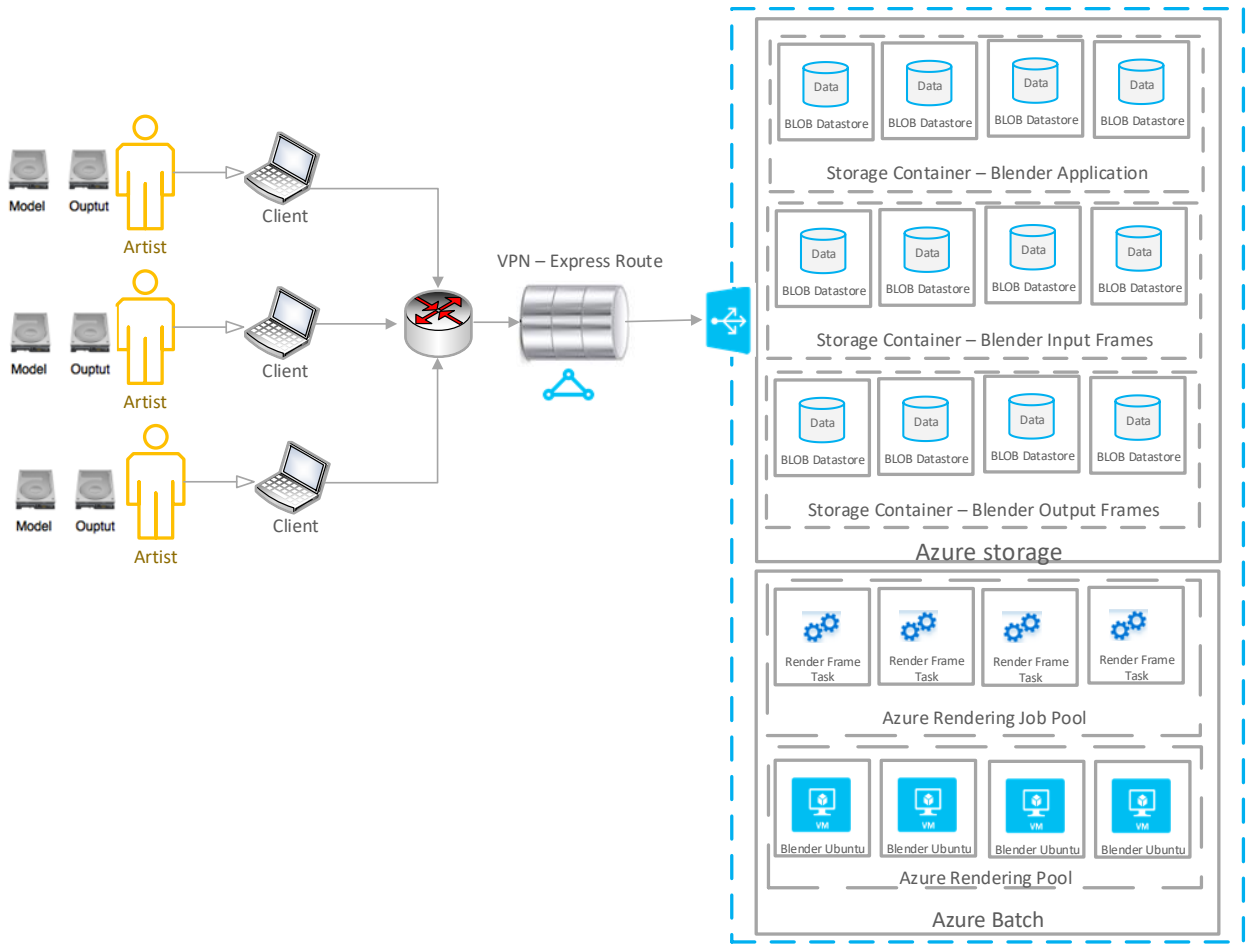

Figure 4. Blender Azure Batch Workflow

Recommendations

We recommend the following when deploying cloud rendering as a "batch cloud rendering farm" deployment. These recommendations supplement the ones provided in the previous sections.

- Instantiate separate storage containers for input, output and application files.
- Use separate Batch Pools to contain each independent tenant or production; Batch Pools are the security boundary for Azure Batch.
- Instantiate batch application services with a trusted software package whose security has been thoroughly vetted.
    - Deploy these services using custom OS images backed by the Shared Image Gallery[1].

[1] https://docs.microsoft.com/en-us/azure/batch/batch-custom-images

- Ensure the application is up to date and patched.
- Implement a centralized monitoring management scheme for the batch processing.
- Consider monitoring application performance, access, users, and data traffic using an automated resource manager.
- Restrict data sharing between storage containers using Azure PrivateLink where appropriate.
- Using a network security group or resource handler, setup IAM roles to control access to storage from associated rendering nodes (e.g. input storage should only be accessible to associated rendering nodes).
    - Use "`BatchNodeManagement`" service tag within NSG rules to control inbound and outbound management traffic[2]. Avoid using Batch Service IP addresses in NSG rules.
- Implement a central management scheme for the monitoring.
- Consider using a 3rd party or Azure tool to monitor performance and data flow of Batch services.
- Use Azure Private Link[3] and Azure VNets to connect Azure Batch pools to external resources.
- Use Azure AD with MFA to authenticate to and from Batch services[4]

Cloud VFX Rendering with RenderMan, Tractor, and Blender

In this scenario, we demonstrate a secure deployment of Pixar RenderMan. Pixar RenderMan is capable of operating as a rendering engine within industry standard animation and modeling tools such as Autodesk Maya, Houdini, Katana, and Blender. RenderMan may execute on a single machine provisioned with a GPU or using an arbitrarily large cluster of machines using the network rendering feature.

RenderMan network rendering is designed to operate "over multiple computers connected through a fast, local area network," through the use of Pixar Tractor render management tool. Pixar Tractor is a modern and robust solution for network rendering, capable of scaling up any render compute farms.

As a possible deployment, we envisioned running modeling software on a single machine within Azure Platform Cloud and accessed remotely using Windows Remote Desktop or similar remote access software, connected to an RenderMan workstation and a set of rendering workers located in the same Azure subscription as the interactive machine.

As part of this assessment, ISE deployed Pixar Renderman and Pixar Tractor within Blender to build a set of security recommendations that enforces data protection in full cloud environment. Content enters and leaves the cloud environment only over transient file transfer connections; it remains in the environment for processing and rendering from the interactive computer system. Figure 5 depicts a sketch of a possible deployment.

Security-relevant challenges pertaining to RenderMan involve the licensing model: RenderMan is designed to connect to a license server, either run on-premises (or inside a cloud deployment) or by Pixar themselves. If a license server is run on-premises, the server itself must have the ability to contact the master Pixar

[2] https://docs.microsoft.com/en-us/azure/batch/best-practices#network-security-groups-nsgs-and-user-defined-routes-udrs
[3] https://azure.microsoft.com/en-us/updates/azure-private-link-for-azure-batch-is-now-generally-available-in-select-regions/
[4] https://docs.microsoft.com/en-us/azure/batch/batch-aad-auth

license servers. For this reason, managing Internet access for Pixar RenderMan can be more complicated than an average deployment.

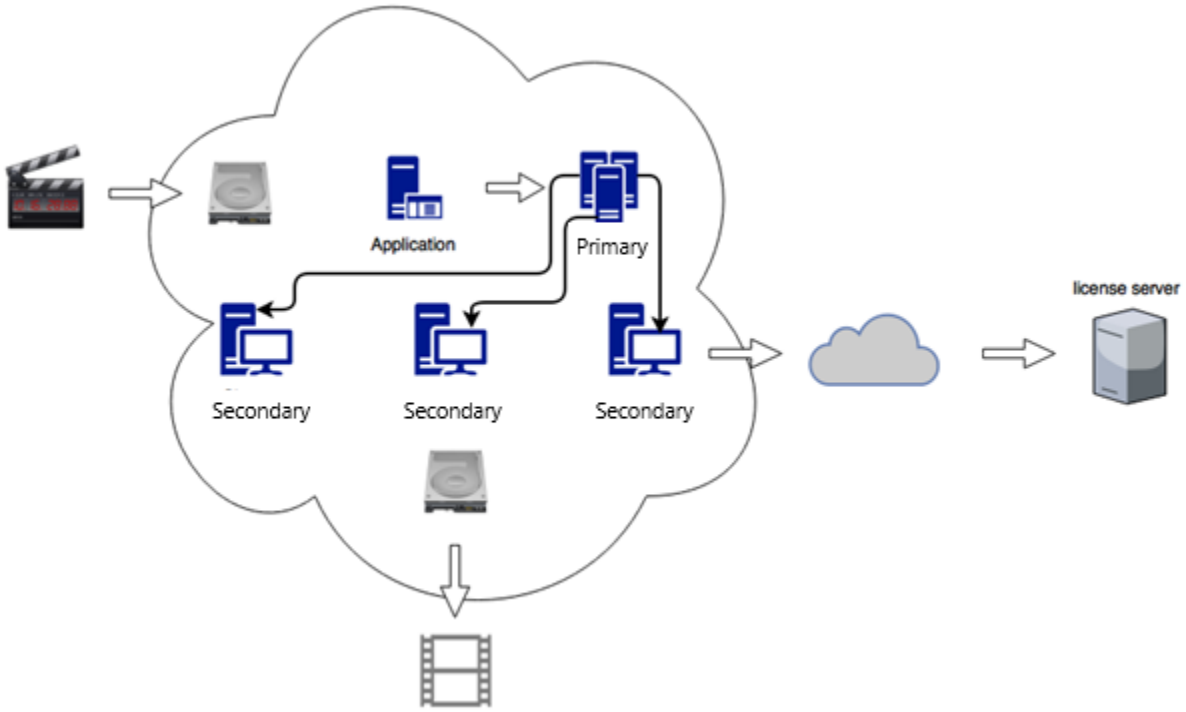

Figure 5. Sketch of Pixar RenderMan and tractor in the Azure cloud deployment

Recommendations

We recommend considering the following when deploying cloud rendering as an "interactive cloud render farm" deployment (render nodes and transient data contained within the cloud):

- All Internet-bound rendering licensing communications to and from render clients/servers must occur through a reverse proxy service or bastion host
- Network access controls for traffic traversing the network should be applied on a least-privilege basis.
- Configure a dedicated cloud subnet to host machines used by interactive users running the rendering client and modeling software with RenderMan plugin (e.g., Maya or Blender).
- Only open the specific ports that are required by the render server.
- Because there is no authentication on distributed render servers, presence on this network subnet, alone, regulates authorization to perform distributed render jobs.
- Apply the least privilege principle to restrict unauthorized rendering; place no machines other than those performing rendering on the render client subnet.
- Restrict traffic from the cloud render servers to render client subnet.
- Use access controls on the cloud VPN device and host-based firewalls on the render client machines to restrict traffic. This way, even a compromised Azure portal account may not be used to evade access controls.

- Restrict traffic from the render client subnet to the cloud render servers.
- Only traffic to the necessary ports should be permitted.
- Use Azure NSGs configuration and host-based firewalls on the render servers to restrict traffic.
- If using the newer online license method, then each host (cloud render or on-premises render client) must be set up with unrestricted access to license server and network scheduler ports.
- Restrict traffic from the license server subnet to the cloud render servers.
- No new connections should be allowed.
- Use Azure NSG configuration and/or host-based firewalls on the render servers to restrict traffic.
- Isolate the render servers from the Internet as much as possible, to reduce the environment's attack surface.
- Do not expose administrative ports (e.g., Windows Remote Desktop, or Secure Shell) to the Internet directly. Instead, use a bastion host or Azure Bastion service to bridge connection to the rendering client and master nodes.
- Using Azure NSG to configure egress firewall rules whitelist outbound traffic from render servers to needed destinations only, e.g., the license server.
- Implement a central management scheme for the render servers.
- Consider Azure Directory or local VM based Active Directory (Windows) or LDAP+Kerberos (Linux) to ease user provisioning and removal from a central location.
- Consider automated deployment and configuration management such as Group Policy (Windows) or Chef, Ansible, or Puppet.
- Ensure that the means of loading modeling input data and retrieving outputs from the cloud is secure.
- Use SFTP, SCP, FTP over TLS, or a commercial product specializing in providing secure file transfers in an Internet-facing environment.
- Harden the protocol or product as appropriate for an environment hosting pre-release media content.
- Use an SSH Certificate Authority, either directly or through a privileged access management solution to handle key provisioning for host and client keys. This avoids the need to manually distribute key fingerprints and prevents exploits taking advantage of SSH's Trust on First Use (TOFU) nature.
- Apply normal SSH hardening practices for SFTP/SCP-only users, such as disabling shell access and port forwarding.
- Use SSH asymmetric key authentication in place of password authentication when practical.
- For TLS-based protocols (such as FTP over TLS) ensure that the client software correctly and effectively verifies the server's hostname and certificate and uses at least protocol version 1.2.
- For commercial file transfer products, harden the configuration as recommended by the vendor.
- Deploy threat intelligence services as Azure Advance Threat protection to detect any user activity anomalies.
- Utilize Azure Security Center to monitor the overall security posture of the deployment.
- Use secure remote desktop techniques such as secure virtual desktop infrastructure (VDI) to access machines in the Azure environment.

- Restrict connections to file transfer software by source IP ranges, if possible. Use the Azure NSG option to whitelist specific source and destination IP addresses and destination ports and name the firewall rules appropriately so that it is easy to identify their role in the rendering environment.
- If the deployment *must* employ a less secure or robust file transfer protocol for compatibility reasons (e.g., unencrypted FTP, NFS, or SMB), then use VPN to encapsulate this traffic as a defense-in-depth mechanism, both to protect the traffic from eavesdropping as it travels over the Internet, and avoid exposing the server software to incoming connections from the Internet.
- Review the recommendations provided in this hardening guide for configuring Azure based VPN.
- Isolate the render workstation(s) and worker(s) from the Internet as much as possible, to reduce the environment's attack surface.
- Do not expose administrative ports (e.g., Windows Remote Desktop, or Secure Shell) to the Internet directly. Instead, use a bastion host to administer the render workstation and workers. This avoids the need to expose each rendering node to incoming Internet traffic and reduces the urgency in applying patches to administrative software in the event of vulnerability disclosure.
- Consider using Microsoft WSUS to distribute updates to Windows-based render nodes or host a local repository mirror for Linux-based nodes. This eliminates the need to allow direct outbound Internet connections from these nodes.
- Using Azure Egress firewall rules whitelist outbound traffic from render nodes to needed destinations only, e.g., the Internet-facing license server.

Filesystem Caching Solution — Avere vFXT

For this hardening guide, Microsoft provided access to a deployment of Avere vFXT. Avere vFXT provides a cloud-based cache layer for data-intensive high-performance computing (HPC) tasks. By caching files close to the compute nodes, Avere vFXT speeds read times and lets you work more smoothly even at peak load.

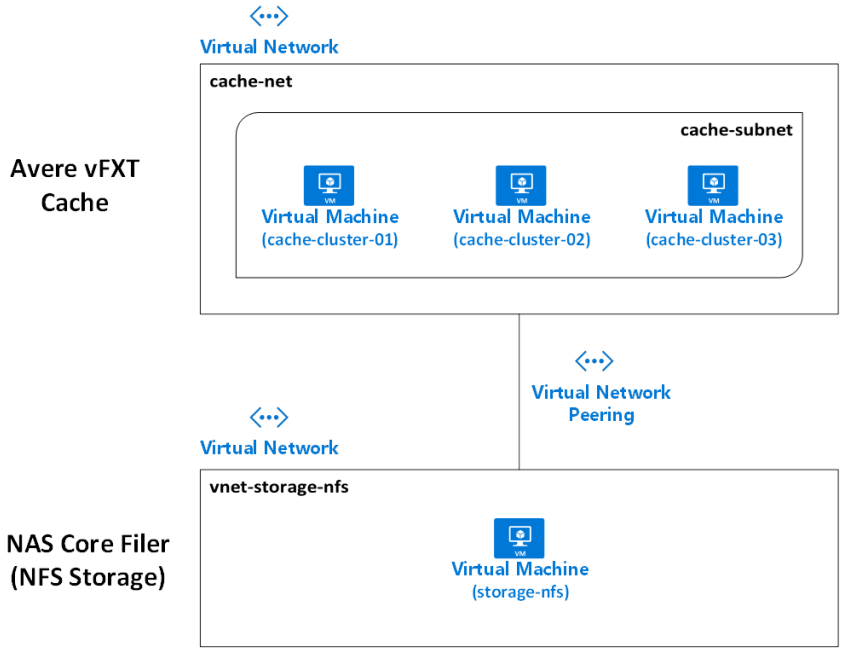

Figure 4. Sketch of Avere vFXT in the Azure cloud deployment

Two common Avere vFXT related computing scenarios:

- Hybrid cloud architecture: Avere vFXT for Azure can work with a hardware storage system, which provides the benefit of cloud computing without having to move files. This allows the data to available close to the compute nodes as possible.
- Cloud bursting: Avere vFXT for Azure can help you move your data to the cloud for a single project, or "lift and shift" the entire workflow permanently.

The evaluated Avere vFXT deployment was a proof of concept (POC) deployment that provided Azure-based computing resources local access to active files that are stored long-term in an on-premises datacenter. In this instance, the on-premises environment was simulated using Azure resources in a separate VNET, Resource Group and subnet.

Azure offers an additional mechanism to handle persistent data caching and clusrtering known as Azure HPC Cache[5] which serves a similar purpose — providing a high-bandwidth, low latency caching layer for large amounts of persistent data — but uses an entirely integrated, managed solution. ISE did not evaluate it as part of this guide but based on specific workload requirements it may be a more suitable caching solution.

[5] https://azure.microsoft.com/en-us/services/hpc-cache/

Recommendations

We recommend considering the following when deploying Avere vFXT:

- All Avere vFXT resources should be deployed in their own resource group.
- The Avere vFXT cluster and the cluster controller VM nodes should have a separate administration user account outside of the scope of any workflow.
- The Avere vFXT cluster and the cluster controller VM nodes should only be accessible through a Bastion host. Do not use Avere vFXT controller node as the jump host; instead use a dedicated bastion host.
- Avere vFXT cluster nodes should not be assigned a public IP address.
- Use a point-to-site or site-to-site VPN from a private network to Azure vFXT nodes for all data communication unless using dedicated lease line connection such as ExpressRoute.
- Use a separate Blob Storage account for Avere vFXT deployment.
- Configure Network Security Groups (NSGs) for the Service Endpoint for backend storage — the NSGs should be configured to deny all outbound internet traffic by default unless needed. It should be noted that default NSGs allow outbound Internet traffic, so this default configuration should be examined and reviewed prior to deployment.
- If needed use service tags in the NSG to allow only traffic to specific Azure services. Use a separate Blob Storage account for Avere vFXT deployment.
    - Use Azure PrivateLink to secure connections to Storage accounts.
- Log and monitor all resource activity associated with Avere vFXT resource group including monitoring Avere service level run-time issues, Avere VM nodes, Avere subnet traffic logs, and Avere storage account.
- Always manually check the Avere deployment configuration after installing Avere vFXT from the Azure Marketplace. Check the deployment logs, created resources and security center for any associated security risks.
- For Avere node authentication, use SSH public key authentication for connecting to the controller.
- The Avere controller password should be created as per local administrator policy or as recommended by NIST 800-63. Do not reuse any other password for the Avere controller password.

Recommendation-TPN-Service Mappings

The following table maps ISE's recommendations to a draft version (current as of June 2020) of the Trusted Partner Network's (TPN) control categories and a suggested Azure service to fulfil that recommendation.

ISE Recommendation	TPN Category	Azure Control
Employ custom Role-Based Access roles to manage user access	Access Control	Azure Access Control
Extend on-premises identity management for access control	Access Control	Azure Access Control
Utilize Secure Workstations to access Azure	Access Control	Azure Active Directory
Access Control	Utilize Secure Workstations to Access Azure	Azure Active Directory

ISE Recommendation	TPN Category	Azure Control
Extend Federated Identity Management for Access Control	Identity & Authentication	Azure Active Directory
Utilize the Custom Banned Password List	Access Control	Azure Active Directory
Isolate Bastion Subnet	Network Security	Azure Bastion
Log All Activity on Bastion Subnet	Auditing & Logging	Azure Bastion
Use RBAC to Control Access to VM Hosts	Access Control	Azure Bastion
Use obfuscated URLs for Batch account URLs	Data Protection	Azure Batch
Ensure Updated Node Agents are Used	System Integrity	Azure Batch
Isolate Jobs in Separate Batch Pools	Production Specific Controls	Azure Batch
Periodically update access keys	Access Control	Azure Batch
Sanitize and destroy Batch accounts when no longer needed	Data Protection	Azure Batch
Create storage exclusively for specific batch workflows	Data Protection	Azure Batch
Use security-validated application packages for batch workflow	System Integrity	Azure Batch
Use integrity checks on application packages for batch workflow	System Integrity	Azure Batch
Hold workflow if Batch applications fail	System Integrity	Azure Batch
Log Batch events for monitoring and diagnostics	Auditing & Logging	Azure Batch
Validate default configuration	Change & Config Management	Azure Cloud Services
Configure monitoring	Auditing & Logging	Azure Cloud Services
Regularly apply patches	System Integrity	Azure Cloud Services
Configure restrictive Traffic Rules	Network Security	Azure Cloud Services
Avoid Caching Session Information	Access Control	Azure Command Line Interface
Use public key authentication for virtual machines	Identity & Authentication	Azure Compute
Use only hardened OS images for VM instantiation	Change & Config Management	Azure Compute
Ensure Virtual Machines are Patched Regularly	System Integrity	Azure Compute
Ensure Disk Storage is Encrypted at Rest	Cryptographic Controls	Azure Compute
Use Azure Managed Identities	Identity & Authentication	Azure Compute
Restrict Access to Azure Container Registries	Access Control	Azure Container Registry
Scan images for security vulnerabilities	System Integrity	Azure Container Registry
Use Azure Managed Identities In Container Tasks	Identity & Authentication	Azure Container Registry
Use HTTPS for Origin Host Protocol	Production Specific Controls	Azure Content Delivery Network
Use Token Authentication to Protect CDN Content	Production Specific Controls	Azure Content Delivery Network

ISE Recommendation	TPN Category	Azure Control
Define Custom Access Policies for CDN-Hosted Content	Production Specific Controls	Azure Content Delivery Network
Do not Use Direct Access Tokens	Access Control	Azure Event Hub
Use Separate Keys for Access to Event Hub	Access Control	Azure Event Hub
Create Separate Event Hubs for Each Consumer Group	Access Control	Azure Event Hub
Avoid Transferring Sensitive Data over the Public Internet	Data Protection	Azure Express Route
Use Separate Azure Key Vaults for Productions	Access Control	Azure Key Vault
Use Key Vault permissions to manage access	Access Control	Azure Key Vault
Segregate data and key/secret owners	Access Control	Azure Key Vault
Manage access to keys and secrets on a per key/secret case	Access Control	Azure Key Vault
Audit all key management activity	Auditing & Logging	Azure Key Vault
Periodically Rotate Keys	Access Control	Azure Key Vault
Use Separate Azure Storage Accounts for Media Service Accounts	Access Control	Azure Media Services
Encrypt assets	Data Protection	Azure Media Services
Configure Live Media Archiving Policy	Production Specific Controls	Azure Media Services
Limit and Control Data Sources	Data Protection	Azure Monitor
Use RBAC-Based Access Control for Access to Log Analytics Workspaces	Access Control	Azure Monitor
Segregate Log Analytics Workspaces	Data Protection	Azure Monitor
Use Network Security Groups (NSG)	Network Security	Azure Networking
Isolate virtual appliances in their own subnet	Network Security	Azure Networking
Deploy workload using a multi-tiered architecture	Network Security	Azure Networking
Create separate VNets for production	Network Security	Azure Networking
Limit default Network Security Group VNet communications	Network Security	Azure Networking
Tightly configure endpoints	Network Security	Azure Networking
Do not use deprecated cryptography for IPSec VPNs	Cryptographic Controls	Azure Networking
Use Network Watcher	System Integrity	Azure Networking
Define Policies that Ensure Compliance	Network Security	Azure Policy
Limit use of personal Microsoft accounts for administration	Identity & Authentication	Azure Portal
Use a separate Azure subscription for each production, client and render workflow	Production Specific Controls	Azure Portal
Logically relate and manage deployments with resource groups	Security Planning	Azure Portal
Avoid concurrent access from multiple locations	Access Control	Azure Portal
Enforce custom session inactivity termination	Access Control	Azure Portal

ISE Recommendation	TPN Category	Azure Control
Enforce two-factor authentication for portal access	Access Control	Azure Portal
Deploy topologies and workloads consistently	Change & Config Management	Azure Portal
Disable non-TLS connections	Cryptographic Controls	Azure Redis Cache
Monitor cache performance	System Integrity	Azure Redis Cache
Define Custom Security Policy	Security Planning	Azure Security Center
Limit Security Center Data Collection	Data Protection	Azure Security Center
Define a Security Response Plan	Security Awareness	Azure Security Center
Perform Validation of Scheduled Actions	System Integrity	Azure Scheduler
Use separate database instances for production and clients	Production Specific Controls	Azure SQL Database
Ensure Updated Container Images are Used	System Integrity	Azure Container Registry
Enable Logging for Azure Container Registries	Auditing & Logging	Azure Container Registry
Create Separate Container Registries for Productions	Production Specific Controls	Azure Container Registry
Use Shared Access Signatures to Access Storage Account Resources	Access Control	Azure Storage
Periodically update access keys	Access Control	Azure Storage
Enable Advanced Threat Protection	System Integrity	Azure Storage
Protect Assets with Digital Rights Management (DRM)	Data Protection	Azure Storage
Require Secure Transfers	Data Protection	Azure Storage
Implement Strict Access Controls	Access Control	Azure Storage

Azure Security Controls

Security controls, as outlined in the executive summary, are presented here for rendering workflows deployed in Azure. They are grouped according to each specific Azure component and apply to both rendering-specific concerns and general Azure workflows.

These Azure-specific security controls consider the high-level architecture of the system within its operational context and offer recommendations for the hardening of that system. A description of each security control, recommendation, and where to find further documentation for each follows in this section.

Azure Access Control

Azure subscriptions are associated with one Azure Active Directory (AD) instance; that is, users, groups, and applications from that directory can manage resources in the Azure subscription. Administrators can grant access by assigning the appropriate RBAC role to users, groups, and applications within a certain scope.

Azure Portal Secure by Default Recommendations

The following is a list of secure by default recommendations for Microsoft Azure

- Do not save passwords in the Web Browser
- Validate Roles and Users in IAM
- View and understand role scope
- Assign role-based user as an administrator of a subscription

Employ custom Role-Based Access roles to manage user access

In many cases, studios have multiple parallel workflows for various productions and vendors involving a myriad of artists and resources. To manage user access, Azure offers customizable Role-Base Access Control (RBAC).

RBAC functions enable fine-grained access management for Azure resources. For example, an administrator can grant a minimal level of controlled access to user accounts to perform their jobs.

Recommendation: Use Azure RBAC With Custom Roles to Manage User Access

Users must be granted the lowest level of required access with minimal operational privileges. Additionally, the Azure built-in role definitions are constantly evolving, and the administrator must define and use custom roles rather than using the built-in roles.

Another possible solution is to use an Azure service principal as an identity for user-created apps, services, command-line access, and automation tools to access specific Azure resources. The service principal is a security identity similar to a user with a specific role, with tightly controlled permissions to access only designated resources.

Recommendation: Use Custom RBAC Based Permissions

For RBAC control to be effective, resources must be granted access based on user role and job function. Do not assign the RBAC based access to all resources – this reduces the overall effectiveness of RBAC roles and user's permission.

Based on our evaluation, we have seen deployment with well-defined RBAC roles and permission but at every resource across has templatized access level. For example – we have observed environment with the same user sets to access to compute, storage, network, monitoring, and application firewall. This type of broad uniform access grants can lead to some users having access to a resource that should be restricted.

Documentation: Azure RBAC built-in roles only support the management operations of the Azure resources in the Azure portal and Azure Resource Manager APIs. Thus, it is imperative that custom roles are defined for Rendering VFX workflow operations, using attributes of a custom role.

Azure role-based access control functionality is described here: https://docs.microsoft.com/en-us/azure/role-based-access-control/role-assignments-portal.

Service principal identity is described here: https://docs.microsoft.com/en-us/azure/active-directory/develop/howto-create-service-principal-portal .

Extend on-premises identity management for access control

Managing identity is as important in the Azure as it is for traditional on-premises computing environments. Studios use on-premises Active Directory (AD) systems to store directory data and manage communication between users and resources, including user logon processes, authentication, and directory searches. When scaling out rendering workflows to Azure in a hybrid deployment or a cloud only environment, the cloud resources are being used as an extension of the on-premises datacenter — in this scenario, there are applications that require a domain controller to handle authentication and authorization.

Recommendation: Use the Azure Active Directory Service

Use an Active Directory service in the cloud. The Windows Server AD is running in VMs created using Azure Virtual Machines, and the AD VMs should be grouped into a virtual network connected to an on-premises datacenter using the Azure Virtual Network.

Documentation: Azure Active Directory services can be implemented in multiple architectural patterns, including Software as a Service (SaaS), platform's IaaS technologies and global enterprise. Information about Azure Active Directory is described here: https://docs.microsoft.com/en-us/azure/active-directory/active-directory-whatis.

Do not use the deprecated Azure Access Control service

Though the Azure Access Control service (ACS) can provide a straightforward way of authenticating and authorizing users to gain access to web applications and services, its functionality has been merged with more advanced and centralized Azure Active Directory services. Starting June 30th, 2017, Microsoft has restricted the creation of new ACS namespaces.

Recommendation: Do Not Use the Azure Access Control Service

Use the Azure Active Directory instead of ACS. Azure Active Directory natively supports many of the scenarios enabled by ACS.

Documentation: Deprecation of Azure Access Control Service (ACS) is discussed here: https://azure.microsoft.com/en-us/blog/acs-access-control-service-namespace-creation-restriction.

Azure Active Directory

Extend Federated Identity Management for Access Control

Managing identity is as essential in Azure as it is on-premises. Studios use on-premises Active Directory (AD) systems to store directory data and manage communication between users and resources, including user logon processes, authentication, and directory searches. When scaling out virtual desktop infrastructure workflows to Azure in a hybrid deployment, the cloud resources are being used as an extension of the on-premises datacenter—in this scenario. Some applications require a domain controller to handle authentication and authorization.

Recommendation: Use the Azure Active Directory Service

Use an Active Directory service in the cloud. The Windows Server AD is running in VMs created using Azure Virtual Machines, and the AD VMs should be grouped into a virtual network connected to an on-premises datacenter using the Azure Virtual Network.

The virtual network carves out a group of cloud virtual machines that interact with the on-premises network via a virtual private network (VPN) connection, which allows the AD Azure virtual machines to look like just another subnet to the on-premises datacenter.

Documentation: Information about Azure Active Directory is described here: https://docs.microsoft.com/en-us/azure/active-directory/active-directory-whatis.

Utilize a Custom Banned Password List

Compromised user accounts are a common cause of breaches and security incidents. Often, the source can be traced to an account using a weak password. To help mitigate this risk, administrators can set password policies and restrictions that prevent the usage of insecure passwords. Azure supports two forms of banned password lists that can aid this goal: a global banned password list that uses a Microsoft-curated compilation of exposed passwords, and a custom banned password feature. The custom list allows administrators to ban passwords containing words and phrases that may be easy to guess, such as the company's name.

Recommendation: Use a Custom Banned Password list

Azure administrators should configure a custom banned password list that blacklists easily guessable words or phrases. Consider the following suggestions:

- Company name and/or initials
- City, state/province, and country where the company is located
- Product and project names
- Names of software used in Azure environment

Documentation: Information about banned password lists is available in Azure's documentation: https://docs.microsoft.com/en-us/azure/active-directory/authentication/concept-password-ban-bad.

Utilize Secure Workstations to access Azure

Secure workstations are hardened computers that are used for a single purpose, such as accessing Azure resources. These workstations should not be used to perform any other tasks such as general web browsing.

Azure facilitates the creation and management of secure workstations using services provided by Azure Active Directory.

Recommendation: Configure Secure Workstations

Managed workstations dedicated to accessing Azure resources should be used by administrators and employees as possible. These machines should be restricted as much as possible to prevent using them for non-Azure purposes. Training should be given to employees on the use of these secure workstations.

Documentation: Azure provides detailed guidance on the enrollment and configuration of managed secure workstations here: https://docs.microsoft.com/en-us/azure/active-directory/devices/howto-azure-managed-workstation.

Azure Bastion

Azure Bastion offers a trusted, HTTP-based privileged access management solution that tunnels remote access protocols such as SSH and RDP over an HTTPS connection with MFA and RBAC built-in. Using the Bastion service allows administrators to apply restrictive security policies, avoid exposing management planes directly to the public internet, and provides extensive auditing and logging facilities. All administration and direct interaction with VM resources should be performed using this service.

Isolate Bastion Subnet

Azure Bastions are deployed inside an Azure virtual subnet. Bastions should be deployed in their own, isolated subnet with restrictive NSGs that allow traffic only on port 443 and 80. There is no specific configuration in the portal or via CLI to limit resources from using this subnet, so it must be enforced administratively.

Recommendation: Lock Down Bastion Subnet

After deploying Bastion in a specific subnet, lock down the subnet and do not deploy any other VM or service in this subnet.

Documentation: An overview of the Bastion service is discussed here: https://docs.microsoft.com/en-us/azure/bastion/bastion-overview.

Log All Activity on Bastion Subnet

Because an Azure Bastion is designed to act as a secure point of entry into an Azure tenancy, it should be treated with the highest possible scrutiny for logging and auditing purposes. This includes logging all network traffic in the subnet to allow the detection of malicious activity.

Recommendation: Log All Subnet Network Traffic

Using the Azure Monitor service, log all network traffic inside the Bastion subnet for later incident response and detection purposes.

Recommendation: Enable Diagnostic and Resource Logs

The Bastion service supports diagnostic logging for highly granular access and activity logs on a Bastion instance. These logs can be used for troubleshooting as well as incident response and detection, and should be enabled, then stored, and processed in accordance with the other recommendations in this document.

Documentation: Information about Bastion diagnostic logging is available here: https://docs.microsoft.com/en-us/azure/bastion/diagnostic-logs.

Documentation: An overview of the Bastion service is discussed here: https://docs.microsoft.com/en-us/azure/bastion/bastion-overview.

Use RBAC to Control Access to VM Hosts

In order to connect to a VM using a Bastion, a user must have the following roles:

- Reader role on the virtual machine
- Reader role on the NIC with private IP of the virtual machine
- Reader role on the Azure Bastion resource

Azure AD can be used to assign access and roles to users based on their AD attributes. This can in practice be used to provide an additional layer of access control to VM resources, as users will not be allowed to even attempt to login to hosts that they should not have access to.

Recommendation: Use Azure AD to Enable RBAC

Use Azure AD to assign the necessary roles to only users that need to be able to interact with a given VM in order to prevent unauthorized users from even attempting to connect to VMs they do not need access to.

Documentation: An overview of how to use RBAC in Azure is available here: https://docs.microsoft.com/en-us/azure/role-based-access-control/overview.

Azure Batch

This section deals with recommendations for securing and hardening the use of the Azure Batch Service, which provides scaling, distributed job management and execution.

Use obfuscated URLs for Batch account URLs

To develop an application with Azure Batch APIs, an account URL is needed to access the Batch resources. A Batch account URL is specified by the user at Batch service initialization.

Recommendation: Anonymize URL

We recommend that a business policy be created making it mandatory for users to choose a URL that does not provide any underlying production name and purpose.

Documentation: Azure batch can be used to efficiently run large-scale parallel and high-performance computing applications such as graphic rendering. Azure batch is described here: https://docs.microsoft.com/en-us/azure/batch/batch-technical-overview.

Periodically update access keys

To authenticate access to an Azure Batch account from the client application, an account access key is required.

Recommendation: Rotate Batch Account Keys

We recommend that users be required to rotate all Batch account access keys periodically to account for workflow, personnel, and production environment changes.

Documentation: Azure batch keys are required to setup and execute jobs. Azure batch configurations are described here https://docs.microsoft.com/en-us/azure/batch/batch-api-basics#pool. Specifically, Azure batch key retrieval and regeneration are described here: https://docs.microsoft.com/en-us/azure/batch/batch-management-dotnet.

Sanitize and destroy Batch accounts when no longer needed

Azure Batch service is not a billable item; the underlying resource pools are billable as they are used. Batch resource pools should be destroyed at the end of their lifetime to prevent reuse across vendors or productions.

Recommendation: Sanitize Batch Account

We recommend that business policies specify that it is mandatory for the user to sanitize and destroy batch accounts and underlying job resources once their associated production has wrapped up.

Documentation: Azure batch initialization, operation and management should be automated using scripts. The processing overhead of sanitizing batch resources by re-instantiating resources pool can be limited using automation. Azure batch scripting using CLI 2.0 is described here: https://docs.microsoft.com/en-us/azure/batch/batch-cli-get-started.

Create storage exclusively for specific batch workflows

The Batch service uses the associated Storage account for the storage and retrieval of application packages. A batch can automatically deploy the packages stored in the linked Storage account to your compute nodes.

Recommendation: Separate Storage Accounts

We recommend using separate storage accounts for each Azure Batch workflow to limit data comingling between productions and rendering setups.

Documentation: Azure batch requires a storage account. The storage account is used to store intermediate, resource files and application input and output files. As mentioned earlier, we recommend segregated Azure Batch resources for each rendering workflow including storage accounts. Azure storage accounts are described here: https://docs.microsoft.com/en-us/azure/storage/storage-create-storage-account.

Use security-validated application packages for batch workflow

Each batch workflow requires application binaries and supporting files that are required to execute the application. As part of the file upload operation, a user defines collections of application and input file paths as they exist on the local machine. Those files are uploaded to the associated storage containers.

Recommendation: Validate Applications

We recommend users be required to upload only validated and trusted application files. Before use, application packages should be assessed for security risks, preferably through a defined security evaluation process such as the Microsoft Security Development Lifecycle practices (SDL).

Documentation: Azure batch compute pool executes user defined applications to perform batch jobs. User defined applications are stored in the Azure storage. Azure application store and add-ons are described here: https://docs.microsoft.com/en-us/azure/storage/storage-create-storage-account. The Microsoft SDL is described here: https://www.microsoft.com/en-us/securityengineering/sdl/practices

Use integrity checks on application packages for batch workflow

Each batch workflow requires application binaries and supporting files that are required to execute the application. Users upload application and input files to associated storage containers. The pool `StartTask` downloads these files to nodes as they join the pool.

Recommendation: Perform Integrity Checks

We recommend users be required perform a predefined integrity check on the downloaded application files as part of the pool `StartTask` execution.

Related Documentation: Azure batch compute pool executes user defined applications to perform batch jobs. User defined applications are stored in Azure storage. Some level of static and run-time integrity validation of these applications is imperative. Run-time validation of these applications can be performed as part of the initial batch start task function. The optional start task executes on each node as that node joins the pool, and each time a node is restarted or reimaged. The initial task is described here: https://docs.microsoft.com/en-us/azure/batch/batch-api-basics.

Hold workflow if Batch applications fail

If an application package deployment fails for any reason, the Batch service marks the node unusable, and no tasks will be scheduled for execution on that node.

Recommendation: Investigate All Failures

We recommend that in case of Batch service marks a node unusable, a user must investigate the cause prior to restarting the package deployment

Documentation: Azure sample workflow processes are described here: https://docs.microsoft.com/en-us/azure/batch/batch-technical-overview.

Log Batch events for monitoring and diagnostics

The Batch service emits log events for certain resources during the lifetime of the resource. Events like pool creation, pool deletion, task start, task completion, and others are included in Batch diagnostic logs. Diagnostic logging is not enabled by default and must be explicitly enabled for each Batch account.

Recommendation: Check Logs

We recommend enabling Azure Batch diagnostic logs to record events for resources like pools and tasks, and then use the logs for diagnostic evaluation and monitoring. This information will allow rendering farm administrators to monitor health status and anomalies.

Documentation: Graphics rendering is a repetitive task, where artists continuously update rendering scenes until they achieve desired results and quality. Storage of rendering results and logging is important for workflow continuity and consistency. Azure batch output storage and job logging are described here: https://docs.microsoft.com/en-us/azure/batch/batch-task-output.

Isolate jobs in separate Batch pools

Azure Batch jobs within different pools are unable to view or communicate with one another. This may be useful if processing must occur in a manner that requires isolation.

Recommendation: Place Jobs Requiring Isolation in Separate Pools

Jobs performing processing that could benefit from isolation should be placed in separate batch jobs. For example, processing on media assets can be separated by production.

Ensure updated node agents are used

The Batch node agent is a software installed on virtual machines used in Batch jobs that provides an interface for controlling the virtual machine. Periodically, new releases of the node agent are released that contain new features and bugfixes.

Recommendation: Update Node Agents

Node agent updates can be applied by periodically reducing the size of pools to zero compute units. Administrators should back up any log files before performing this operation. Companies should document and implement a policy to perform this action on a regular cadence.

Documentation: Microsoft provides documentation on this procedure and others in their best practices guide for Azure Batch: https://docs.microsoft.com/en-us/azure/batch/best-practices.

Azure Cloud Services

Azure Cloud Services offers a convenient way for developers to combine Azure services such as Storage, Compute, and managed SQL servers. Each service is described by a service bundle, which contains an XML description of the various roles such as web services or backend service workers, as well as a description of other cloud services involved. Applications deployed using Cloud Services can leverage streamlined monitoring and scaling workflows that make long-term management and service build-out significantly easier than traditionally deployed applications.

Because using Azure Cloud Services inherently involves instantiating other Azure resources, the relevant hardening steps for those resources in this guide apply. For instance, follow the “Azure Compute” section of this guide for information on hardening the compute resources instantiated through Cloud Services.

Validate default configuration

Using Azure Cloud Services involves defining configuration steps for virtual machine instances that are dynamically constructed and destructed based on demand. It is important to include configuration steps that change any insecure or sub-optimal default settings to more secure versions, as manually doing so is an error prone, time consuming process.

Recommendation: Ensure VMs Are Created Using Secure Settings

As part of the development process, developers should ensure that the hosts and Azure resources that their service instantiates are created using secure, hardened configurations. Any settings that cannot be configured directly using the application bundle should be configured using the hosts’ startup tasks. For host and hardening info, see the portions of this guide.

Documentation: Microsoft provides documentation on host startup tasks here: https://docs.microsoft.com/en-us/azure/cloud-services/cloud-services-startup-tasks.

Configure monitoring

Azure Cloud Services collects general performance metrics for each service and host automatically, and supports custom application metrics as defined by the application developer. Developers should configure custom monitoring metrics as appropriate for their application and ensure that all metrics are collected and monitored per the monitoring guidelines found later in this document.

Recommendation: Configure Monitoring

As part of the development process, developers should configure relevant metrics to be collected by the Azure Cloud Services service, and ensure those metrics are analyzed and monitored as appropriate. See the Azure Monitor section later in this document for additional recommendations for monitoring applications in a secure, hardened manner.

Documentation: Microsoft provides documentation on monitoring Azure Cloud Services here: https://docs.microsoft.com/en-us/azure/cloud-services/cloud-services-how-to-monitor.

Regularly apply patches

Azure Cloud Services-based applications involve the developer directly specifying the underlying OS version as well as implicitly specifying the version of any third-party software or runtimes (e.g., web frameworks) that are included in the application bundle. It is therefore the developer's responsibility to regularly update both of these to ensure that applications are not vulnerable to known or unknown security issues.

Recommendation: Regularly Patch Applications

Developers should regularly apply patches to third-party software included in their application bundles and upload those patches to the Azure Cloud Services instance.

Recommendation: Regularly Update OS

Azure Cloud Services allows developers to specify that the Azure Cloud Services service should automatically upgrade the OS running on individual hosts, or to manually specify a patch level for each host. Automatic updates should be employed if possible. If not, developers should regularly perform manual updates to ensure that hosts are patched against the latest known security issues.

Documentation: Microsoft provides documentation on updating a cloud service here: https://docs.microsoft.com/en-us/azure/cloud-services/cloud-services-how-to-manage-portal. Information on how to configure host OS upgrades is available here: https://docs.microsoft.com/en-us/azure/cloud-services/cloud-services-how-to-configure-portal.

Configure restrictive Traffic Rules

Azure Cloud Services offers the ability to specify Traffic Rules in an application bundle. These rules define whether a host in a given role is allowed to accept connections on a given port from either other internal hosts or external clients. These rules are analogous to Network Security Groups and should be configured equally restrictively, using the principle of Least Privilege.

Recommendation: Configure Restrictive Traffic Rules

As part of the development process, developers should identify the minimum subset of traffic required for each service role to function and ensure that Traffic Rules are created to reflect that subset and allow no other traffic.

Documentation: Microsoft provides documentation on configuring Traffic Rules in Azure Cloud Services here: https://docs.microsoft.com/en-us/azure/cloud-services/cloud-services-enable-communication-role-instances.

Azure Command Line

This section deals with the Azure Command Line Interface (CLI), which is the primary ways for developers and administrators to interact with Azure APIs and services.

Avoid caching session information

One way that users authenticate to Azure is by using a Microsoft account email address and password. To prevent a user from needing to retype credentials upon every invocation of the command line utility, the utility persistently caches the user's OAuth session information (excluding passwords) on the local system.

Azure Caveat: CLI Caches OAUTH Session Data

Caching session information presents a security threat should the local system used to log in become compromised, even at a later date. Storing the OAuth access token, client secret, and other session information on the file system means that if an adversary gains access to the credentials file or a copy of it, the attacker may then use the session information to gain unauthorized access to the user's account.

Recommendation: Mandate Deletion of Credentials at Session

Create a policy that mandates deleting credentials when logging off.

Recommendation: Use A RAM disk For Credential Storage

Using a RAM disk ensures that credentials are never written to disk. To do so, create a RAM disk and then create a symbolic link from the credentials file to a file located on the RAM disk. This can be done using the Linux tmpfs file system, or using analogous techniques on Windows or Mac OS X. This ensures that the credentials are stored only in volatile memory and lost upon unmounting the RAM disk or rebooting the machine.

Recommendation: Use Active Directory for User Management

Use Azure Active Directory to manage user accounts, including auditing and logging capabilities.

Documentation: Azure CLI 2.0 and Azure PowerShell can be used to manage and administer Azure resources from the command line and for the creation of automation scripts that work with the Azure Resource Manager. Both interfaces have built-in methods for developer authentication. Azure CLI and PowerShell are described here respectively: https://docs.microsoft.com/en-us/cli/azure/overview.

Azure Compute

This section deals with controls designed to secure virtual machines used within the Azure environment.

Use public key authentication for virtual machines

Azure Virtual machines allow authentication using a username and password or using public keys. Using public key-based authentication is a good practice since it eliminates weak passwords on the system or hardcoded default passwords. Using public key-based authentication also makes brute force attacks infeasible and dictionary attacks impossible. Disabling login via passwords and requiring login keys is a configuration option when instantiating a VM or VM scale set in Azure.

Recommendation: Use Public Key-Based Authentication for Virtual Machines

Configure the SSH processes on Azure VMs to use public key authentication to protect against weak and hardcoded default passwords.

In addition to replacing passwords and their associated problems, with SSH the login key option can be set up to provide the following functions:

1. Only allow access from a specific IP; and

2. Only allow the accessing user/script to execute a certain command.

Documentation: Public-key cryptography provides a more secure way to log in to the virtual machine. The use of SSH keys with Windows for Linux VMs is described here: https://docs.microsoft.com/en-us/azure/virtualmachines/linux/ssh-from-windows.

Use only hardened OS images for VM instantiation

Within the Azure platform, there are a number of pre-built operating system images available to users for rapid deployment. The images provided include a range of Linux distributions (including CentOS and Ubuntu) as well as Windows 2012 R2 and Server 2016 R2. In the current state, the OS images are supplied with default configurations. Some default configurations of OS images are not configured for maximum security and instead are designed to provide greater convenience or compatibility. For example, the images may have their TLS configurations set up for maximum compatibility, rather than security.

Recommendation: Create and Use Customized, Hardened OS Images

Harden all supplied operating system images using current best practices. Microsoft-provided configuration scripts for Azure Managed Secure Workstations can be of use as starting points for the creation of these images.

Hardening steps should include closing unused ports, uninstalling unnecessary applications, updating software, configuring web servers using current best practices, and setting firewall rules. When completed, the use of these hardened images will support the secure by default principle.

Documentation: Editing workloads require a specific set of resources to optimally function. These resources include a base operating system, application specific software site, and other supporting software applications. Azure allows uploading customized/hardened virtual machine images using the Azure Resource Manager service to realize a consistent operating environment

Descriptions of how to create Windows virtual machines using an Azure Resource Manager template are described here: https://docs.microsoft.com/en-us/azure/virtual-machines/windows/app-frameworks.

Descriptions of how to create Linux virtual machines using template images are defined here: https://docs.microsoft.com/en-us/azure/virtual-machines/linux/tutorial-manage-vm.

It should be noted that these images should not be considered secure or hardened for VFX rendering workflows. It is suggested that the administrator creates a secure golden image and imports the VM image into Azure using the process described here: https://docs.microsoft.com/en-us/azure/virtual-machines/linux/tutorial-custom-images. It maybe helpful to use Azure Managed Secure Workstation configuration scripts as a starting point for creating Windows-based hardened OS images. Information about Managed Workstations and the configuration scripts used to create them can be found here: https://docs.microsoft.com/en-us/azure/active-directory/devices/concept-azure-managed-workstation.

Ensure Virtual Machines are Patched Regularly

The applications and operating systems in virtual machines hosted in Azure must be kept up to date to ensure they are protected against known security issues. To better accomplish this, organizations should develop patch management strategies that ensure all software hosted in Azure is updated frequently.

Recommendation: Implement a Patch Management Policy

Administrators should develop and implement a patch management process that regularly updates operating systems and applications hosted in Azure Virtual Machines. A common update cadence is 30 days; however, updates may need to be applied more frequently depending on the software used and the sensitivity of the machines' workflows. Administrators should also develop a strategy to deploy updates that patch critical vulnerabilities in a timely manner, which may need to occur between regular update cadences.

Ensure Disk Storage is Encrypted at Rest

Encryption at rest protects data stored on the physical medium from unauthorized access if an attacker is able to interact with the storage device. Azure Virtual Machines support encryption at rest via the Azure Disk Encryption feature, which uses BitLocker and dm-crypt on Windows and Linux systems, respectively, to encrypt data.

Recommendation: Enable Disk Encryption Features

Azure Disk Encryption should be enabled on all disks used by virtual machines, especially if they store sensitive assets. The additional protection offered by this feature provides a layer of defense-in-depth if attackers are able to compromise the physical storage devices used by the virtual disk. Azure Key Vault should be used to manage secrets such as disk encryption keys.

Documentation: Azure provides additional information about disk encryption here: https://docs.microsoft.com/en-us/azure/security/fundamentals/azure-disk-encryption-vms-vmss.
Documentation about using Azure Key Vault to manage disk encryption keys can be found here: https://docs.microsoft.com/en-us/azure/virtual-machines/linux/disk-encryption-key-vault.

Use Azure Managed Identities

Managed Identities is an Azure feature that allows supported services to be automatically provisioned a secure identity that allows it to access appropriate resources, subject to customer-managed controls. This reduces operational complexity and prevents configuration errors. These identities are automatically integrated with Azure AD and can be used to access any Azure AD managed resource.

Recommendation: Use Azure Managed Identities

Where appropriate, Azure Managed Identities should be used to authenticate Azure services such as virtual machines against other Azure AD managed resources.

Documentation: Azure provides additional information about managed identities here: https://docs.microsoft.com/en-us/azure/active-directory/managed-identities-azure-resources/qs-configure-portal-windows-vm.

Azure Container Registry

Restrict access to Azure container registries

Azure Container Registries (ACR) store container images that can then be deployed using technologies such as Docker. As the Registries may house images containing proprietary or sensitive data, regulating access to ACRs is an important aspect of securing container-based workflows. To support this effort, ACR supports the use of firewall rules to control access to registries.

Recommendation: Use Firewall Rules to Control Access to ACR

Firewall rules should be created to restrict access to Azure Container Registries. When possible, access should be restricted to whitelisted IP addresses or networks.

Recommendation: Use Azure PrivateLink To Control Access to ACR

Azure PrivateLink can also be used to provide a private, high performance connection to Azure Container Registries.

Documentation: Microsoft provides documentation on ACR firewalls at https://docs.microsoft.com/en-us/azure/container-registry/container-registry-firewall-access-rules and https://docs.microsoft.com/en-us/azure/container-registry/container-registry-vnet. Information on using Azure PrivateLink with ACRs can be found here: https://docs.microsoft.com/en-us/azure/container-registry/container-registry-private-link.

Scan images for security vulnerabilities

Containers typically bundle an operating system and applications and may contain vulnerabilities from their software and configurations. Using an automated scanner can help detect security flaws before the containers are used in production environments. Azure Security Center supports integration with ACR to scan images as they are uploaded to the registry.

Recommendation: Use the Security Center to Scan Images

Azure Security Center should be configured to scan images stored in Azure Container Registry. This automated testing may be used to detect common vulnerabilities affecting container images. It is important to note that automated scans do not replace manual audits and may not find all security vulnerabilities in an image.

Documentation: Information about scanning ACR images can be found at https://docs.microsoft.com/en-us/azure/security-center/azure-container-registry-integration.

Use Azure Managed Identities In Container Tasks

Container tasks often need to access other Azure resources such as storage services. Azure offers Managed Identities, which are automatically provisioned strong identities that authenticate against any Azure AD managed resource and require no manual secrets management. These identities can be attached to Container Tasks (as opposed to customer managed identities such as a service username and password). Managed identities should be used wherever possible to reduce complexity and increase security and resilience.

Recommendation: Use Azure Managed Identities Where Appropriate

Azure Managed Identities should be used where appropriate (e.g., to connect to other Azure AD managed services).

Documentation: Information about using Managed Identities with Azure Container Registry can be found here: https://docs.microsoft.com/en-us/azure/container-registry/container-registry-tasks-authentication-managed-identity.

Azure Content Delivery Network

Azure offers a Content Delivery Network (CDN) to allow customers to optimize the delivery of media and other assets to their users. The CDN service uses geographically distributed cache servers to reduce user latency, application load, and bandwidth costs. These cache servers are configured to connect back to a customer controlled origin host for any data that is either not cachable (such as dynamic content) or that is not already in the server's cache.

Use HTTPS for Origin Host Protocol

When instantiating a new CDN service, the user must select a protocol and origin port that will be used to connect to the origin host. Origin protocols can be either cleartext HTTP or secure HTTPS.

Recommendation: Use HTTPS for Origin Connections

Use HTTPS for origin connections to prevent cleartext content from being transmitted between backend servers and CDN cache nodes. Using HTTPS requires a user to use an SSL certificate provided by the CDN, and use a CDN-provided domain to access HTTPS content.

Documentation: Azure CDN creation is described here: https://docs.microsoft.com/en-us/azure/cdn/cdn-create-new-endpoint.

Use Token Authentication to Protect CDN Content

Content requests should be authenticated by CDN edge points-of-presence (POPs) before delivering the asset. Lack of authentication can lead to the Azure CDN serving assets to unauthorized clients. For example, links to assets can be shared on different websites and with different users.

Recommendation: Use Token Authentication

Use token authentication to prevent the Azure CDN from serving assets to unauthorized clients. Token authentication verifies requests generated by a trusted site through a token value containing encoded information about the requester.

Documentation: Azure CDN token authentication is discussed here: https://docs.microsoft.com/en-us/azure/cdn/cdn-token-auth.

Define Custom Access Policies for CDN-Hosted Content

In media production workflows, it is imperative to protect digital assets against theft, misuse, or piracy. When servicing content through CDN, it is important to setup specific policies and rules for blocking the delivery of certain types of content, defining a caching policy, and modifying HTTP headers.

Recommendation: Define Explicit CDN Access Policies

Define explicit CDN rules to limit asset delivery to a select set of users and requests. A robust caching policy must be defined that limits the content's "time to live;" for example, define an aggressive cache purge policy for sensitive or protected content. These access policies are created using the CDN rules engine, which allows developers to specify complex rules defining access policies as well as server behaviors such as adding or removing specific headers.

Documentation: The CDN rules engine is discussed here: https://docs.microsoft.com/en-us/azure/cdn/cdn-rules-engine

Azure Event Hub

The Event Hub is designed to the internet facing "front door" for an event pipeline. The Event Hub uses an event "ingestor" service that exists between event publishers and event consumers to decouple the production of an event stream from the consumption of those events. Event Hubs provides message stream handling capability but has characteristics that are different from traditional enterprise messaging. The following figure depicts this architecture.

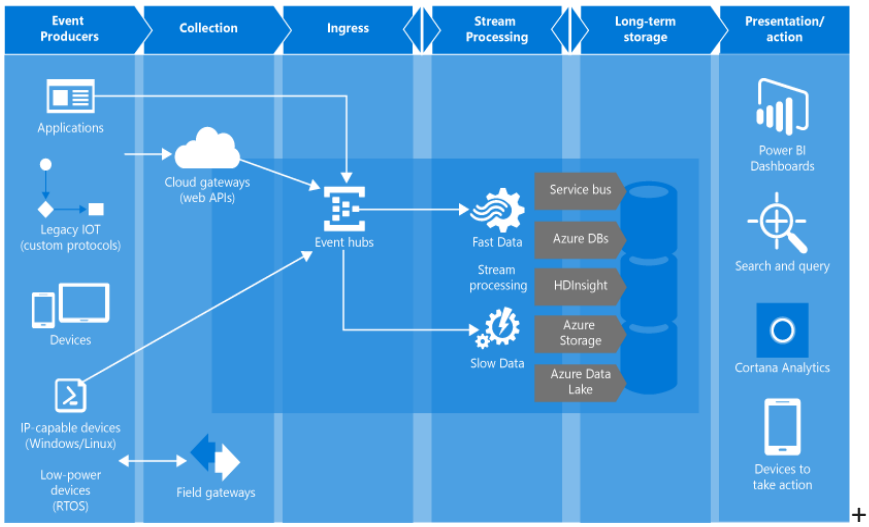

Figure 6 Sketch of Event Hub Reference Implementation

Do not Use Direct Access Tokens

The Event Hub's security model is based on a combination of Shared Access Signature (SAS) tokens and event publishers. Each Event Hubs client is assigned a unique token, which is uploaded to the client. Clients

claim access to Event Hub resources by presenting a SAS token. The tokens are produced such that each unique token grants access to a different unique publisher. A client that possesses a token can only send to one publisher. If multiple clients share the same token, then each of them shares a publisher. It is also possible to equip devices with tokens that grant direct access to an event hub. Any device that holds this token can send messages directly into that event hub. Such a device will not be subject to throttling and cannot be blacklisted from sending events to that event hub. These tokens therefore present a security risk and should not be used.

All tokens are signed with a SAS key.

Recommendation: Use SAS Tokens

Use SAS tokens to grant access to event hub endpoints for publishing. Do not create tokens that grant direct access to the event hub itself.

Documentation: The Azure Event Hub security model is described here: https://docs.microsoft.com/en-us/azure/event-hubs/event-hubs-authentication-and-security-model-overview.

Use Separate Keys for Access to Event Hub

Each Event Hubs namespace has a specific 256-bit SAS key called "`RootManageSharedAccessKey`". This key grants send, listen, and manage rights to the namespace. Single or separate keys can be used to grant varying levels of access.

Recommendation: Use Separate Keys

Create a separate key that grants send permissions to the specific event hub. For example, generate a key that grants only "Send" permission and is different from a key that has listen and manage rights.

Documentation: The Azure Event Hub security model is described here: https://docs.microsoft.com/en-us/azure/event-hubs/event-hubs-authentication-and-security-model-overview.

Create Separate Event Hubs for Each Consumer Group

In a multi-tenant application where Azure Event Hub is shared among different tenants, tenants can possibly send messages on different partitions. In such scenarios, a user might want to isolate tenants using event partitions. however, partitions are not intended to be a security boundary. Instead, use separate Event Hub instances to separate data between tenants.

Recommendation: Use Separate Event Hubs for Each Tenant

Use multiple Event Hubs to isolate data between tenants in a multi-tenant application.

Documentation: Azure Event Hub event publishers and partitions are discussed here: https://docs.microsoft.com/en-us/azure/event-hubs/event-hubs-features.

Azure ExpressRoute

Avoid transferring sensitive data over the public Internet

While modern protocols with transport-layer encryption provide robust security that ensures the confidentiality and integrity of data in transit, for an additional level of security, consider transferring sensitive assets solely over ExpressRoute, a dedicated connection to Azure that does not use the public Internet.

Recommendation: Use ExpressRoute to Transfer Sensitive Data

Instead of using connections to Azure resources that travel over the public Internet and may be exposed to adversaries with privileged network access, utilize an ExpressRoute connection between on-premises data centers and Azure to transfer sensitive data.

Azure Key Vault

Azure provides the Key Vault service for controlling cryptographic keys, digital certificates, and passwords and access to these items. The use of Key Vault is integrated into all of Azure's other services.

Use a separate Key Vault for each production

The Key Vault service allows the creation of multiple segregated key vault instances per Azure subscription. This should be used to segregate keys, passwords, and other secrets between productions; a single key vault should not span across multiple disjoint productions. Each production has a defined lifecycle with varying priority, protection, and collaboration requirements. Production-specific key vaults should be used to limit access to protected assets associated with the production lifecycle.

Recommendation: Use Separate Key Vaults for Productions

A separate instance of the Azure Key Vault service should be used for each production to enable individualized resource monitoring and fine-grained lifecycle control.

Documentation: Understanding of the Azure Key Vault model is imperative to securing Azure instances. An Azure subscription grants the user access to the Azure services and management portal. A subscription can have multiple vaults in multiple geographic areas. Azure subscriptions are described here: https://account.windowsazure.com/Subscriptions. Key Vault information can be found here: https://docs.microsoft.com/en-us/azure/key-vault.

Use Key Vault permissions to manage access

During production, various parties will require access to the production Key Vault. Azure has built-in permissions for access to this. In conjunction with Active Directory, these permissions enable fine-grained access control to a specific Azure Key Vault. It is feasible that in workflows the applications or users will only need limited read-only, encrypt or decrypt access to a specific Key Vault in order to be able to perform their functions.

Recommendation: Use Least Privilege in Assigning Access To The Key Vault

Services and users accessing the Azure Key Vault must be granted the lowest level of required access with minimal operational privileges.

As mentioned in the RBAC security control previously, a possible solution is to use a service principal as an identity for user-created apps, services, command-line access, and automation tools to access specific Azure resources.

Documentation: Use Azure RBAC built-in Key Vault permissions to limit access to management operations of an Azure Key Vault. Resources and permissions are described here: https://docs.microsoft.com/en-us/azure/key-vault/key-vault-secure-your-key-vault.

Segregate data and key/secret owners

In many cases, studios will employ a security administrator or team to manage Key Vaults. The security administrator or team will be responsible for initializing the vault, furnishing and maintaining keys, and controlling access to the vault, keys, and secrets. The data owners will use an assigned Key Vault to encrypt, decrypt, or wrap keys and manipulate digital assets. It is imperative that key and data owners have mutually exclusive permissions on a key pair.

Recommendation: Assign Users, Service Principle, Key Owner or Manager Rights

Assign these key level functions:

- Create
- Delete
- Update
- Import
- List versions
- Get
- Backup
- Restore
- Key Wrap and Unwrap

Recommendation: Assign Data Owners and Services Rights

Assign these key level functions:

- Encrypt and Decrypt
- Sign and Verify

Documentation: Key Vault allows an organization to securely manage and protect cryptographic keys and secrets which can be used by cloud-enabled applications and services. Key access policies are different from secrets access policies.

Azure Key operations are described here: https://docs.microsoft.com/en-us/rest/api/keyvault.

Azure Secrets operations are described here: https://docs.microsoft.com/en-us/rest/api/keyvault.

Manage access to keys and secrets on a per key/secret case

A production Key Vault will contain keys to support various independent workflows. These workflows are usually executed in parallel in a compartmentalized manner. Azure offers key and secret operations to support this. Access to keys and secrets can be managed either on a per key/secret case or on the whole of Key Vault.

Recommendation: Customize Access Control Lists for Keys and Secrets

Provision keys and secrets of having a defined customized access control list. Keys and secrets should only be accessible to users or applications with a functional need.

Documentation: Key access policies are different from secrets access policies.

Azure Key operations are described here: https://docs.microsoft.com/en-us/rest/api/keyvault.

Azure Secrets operations are described here: https://docs.microsoft.com/en-us/rest/api/keyvault.

Audit all key management activity

In media and entertainment workflows, the Azure Key Vault will be used to control access to digital assets. The content owners should track how, when and by whom the Key Vault is accessed to manipulate a protected asset. Key Vault logs will indicate who accessed specific assets, and what operations were performed in near real-time. These logs can be piped into a higher-level event management system to build a higher-level data use model.

Recommendation: Log All Key Vault Activity

Enable logging for all Key Vaults for the subscription in "Log specific Azure storage account." Access to the log storage account should be restricted as per storage security controls. It is recommended to setup specific alerts and monitor the events using Azure Security Center and Monitor.

Documentation: Azure Key Vault logging can be configured in a deeply customizable manner. The Azure Key Vault logging is described here: https://docs.microsoft.com/en-us/azure/key-vault/key-vault-logging.

Furthermore, the Azure Key Vault event log data can be used with the Azure Log Analytics application. Logging data can then be used for data analytics. Key Vault logging data is described here: https://docs.microsoft.com/en-us/azure/azure-monitor/insights/azure-key-vault.

Periodically rotate keys

Rotating keys is as important in the Azure as it is for non-cloud applications. Applications can offload the storage of keys and secrets to Key Vault, and applications request keys from the Key Vault as needed. This centralized approach allows the administrator to update keys and secrets without affecting the behavior or structure of applications. Though rotation of keys doesn't decrease the risk of keys being breached, it does limit the amount of data encrypted under a certain key, so for example, if a future key gets breached, past encrypted data is safe. Also, rotation of keys limits the time an adversary has to break the current key.

Recommendation: Rotate Key Vault Keys Every 90 Days

There are various options for implementing a rotation strategy for values you store as Azure Key Vault secrets. Though secrets can be rotated as part of a manual process, they should be rotated programmatically by using API calls or automation scripts.

System owners are encouraged to consult the guidance in NIST SP 800-57 to create a baseline key management policy. The recommendations in SP 800-57 instruct data owners to consider the sensitivity and lifetime over which data needs to be protected and to tailor cryptographic algorithms and key sizes chosen to fit the required protection needed.

Documentation: Azure Key Vault key rotation using automation is described here: https://docs.microsoft.com/en-us/azure/key-vault/key-vault-key-rotation-log-monitoring.

For NIST recommendations see Special Publication 800-57 Part 1 Revision 4, Recommendations for Key Management. The document is located here: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r4.pdf.

Azure Media Services

Azure Media Services was created for video workflows. Microsoft outlines this service here: https://azure.microsoft.com/en-us/services/media-services/.

Use separate Azure storage accounts for media service accounts

Storage accounts must be located in the same geographic region as the Media Services account. When a user creates a Media Services account, they either use an existing storage account in the same region or create a new storage account in the same region.

Recommendation: Use Separate Storage Accounts for Media Accounts

Use separate storage accounts for media accounts since, logically, media accounts associated with the Media service are separate entities. Assets in the storage account associated with the Media service will likely be shared with an audience or processed for a specific purpose. The content of the storage account associated with the media should only share a limited set of data that the user has explicitly marked for processing or sharing. The storage should be protected with guidance provided earlier in this document.

Documentation: Azure Media account setup is described here: https://docs.microsoft.com/en-us/azure/mediaservices/media-services-portal-create-account.

Encrypt assets

Media production workflows rely heavily on ad-hoc and near real-time asset sharing and collaboration. Azure Media service video delivery can be used to deliver video assets within a content production team across the globe. The video assets should be protected when shared in this manner.

Recommendation: Use Key Delivery Service With AES

Use the Key Delivery service with Advanced Encryption Standard (AES) using 128-bit encryption keys for delivery of HTTP-Live-Streaming (HLS) and Smooth Stream video assets.

The Azure Key Delivery service can be used to deliver encryption keys to authorized users. A user can encrypt an asset and associate an encryption key with the asset, then configure authorization policies for the key. When a stream is requested by a player, Media Services uses the specified key to dynamically encrypt the content using AES encryption.

To decrypt the stream, the player will request the key from the key delivery service. To decide whether the user is authorized to get the key, the service evaluates the authorization policies that were specified for the key. This method allows a content publisher to control content delivery by revoking the key when access is no longer needed.

Documentation: Azure Media encryption is defined here: https://docs.microsoft.com/en-us/azure/mediaservices/media-services-protect-with-aes128.

Configure Live Media archiving policy

Media Services offers live streaming of events directly from a camera device. While streaming the event, the service can be configured to archive content in the storage account as it is encoded and streamed live.

Recommendation: Define Content-Specific Archive Policy

Define a content-specific archiving policy for streamed content. In media production environments, each asset has its own lifecycle which should be mirrored in its archive policy.

Documentation: Azure Media live streaming is defined here: https://docs.microsoft.com/en-us/azure/mediaservices/media-services-manage-channels-overview.

Azure Monitor

The Azure Monitor service collects log data from multiple sources including but limited to applications, operating systems, Azure resources, Azure subscriptions, and Azure tenants. The collected log data is used to create metrics about the platform — administrators can use the data to proactively identify issues affecting the environment, applications, and users. Azure Monitor feeds, collected logs and metrics should be used following purposes:

- Information Building — Build useful data driven insights about applications, containers, VMs
- Visualize — Build custom dashboards, views, Power BI, workbooks
- Analyze — Analyze metrics and logs
- Respond — Alert and auto-scale to deal with performance issues
- Integrate — Integrate the data into apps and export it using API services

Limit and Control Data Sources

Azure Monitor is capable of collecting massive amounts of data from a large set of data sources. It is possible some of the data collected could include personal data or data about the production, workflows, and clients. The administrator should configure the data sources carefully and manage the data that is pushed out to the Azure Monitor.

Recommendation: Limit Data Sources

Data from applications, operating systems, Azure resources, and Azure tenants should be carefully reviewed before being pushed to Azure Monitor. It is recommended that data from guest operating systems and Azure resources not include any specific references to clients, workflow or production. If possible, the data should be anonymized to prevent any accidental data leakage.

Documentation: Azure Media Monitor data sources setup are described here: https://docs.microsoft.com/en-us/azure/azure-monitor/platform/data-sources.

Use RBAC-Based Access Control for Access to Log Analytics Workspaces

A Log Analytics workspace is where data is collected, aggregated, analyzed, and presented. A workspace is primarily used as a means to partition data, and each workspace is unique. The administrator should control user access to the data through role-based access controls for each workspace. Each workspace can have multiple user accounts associated with it, and each user account can access multiple Log Analytics workspaces.

Recommendation: Define RBAC Based Access to Log Analytics Workspace

Use RBAC to restrict access to log analytics workspaces. It is recommended that no one user have access to all workspaces concurrently. Instead, distribute the privilege of accessing log analytics to multiple users.

Documentation: Azure Monitor data security is discussed here: https://docs.microsoft.com/en-us/azure/azure-monitor/platform/data-security.

Segregate Log Analytics Workspaces

Each Azure subscription can have multiple log analytics workspaces. Rather than using a single workspace to view all log information, segregated workspaces should be used to limit the exposure of log information and prevent cross-contamination between separate workflows, productions, clients, etc.

Recommendation: Segregate Log Analytics Workspace based on Workflow or production

Define and use separate Log Analytics Workspaces based on the needs of individual VFX rendering workflows.

Documentation: Azure Media encryption is defined here: https://docs.microsoft.com/en-us/azure/azure-monitor/platform/data-security.

Azure Networking

Azure provides a number of networking products and capabilities allowing users to tailor the virtual networking environment to the services they would like to provide. Much of the work of securing Azure networking involves the separation of data. This is largely accomplished by using the Azure Network Security Group (NSG) feature.

Use Network Security Groups (NSGs)

While Azure completely restricts incoming traffic from the Internet, it is more permissive about internal traffic — essentially allowing open communication between all VM instances within the virtual network (VNET) similar to a physical LAN network. While the default endpoint security features are a useful mechanism for securing Azure VMs, they have limited functionality. Network Security Groups (NSG) secure both inbound and outbound access to both Azure VMs and Azure VNets, analogous to a traditional layer 4 firewall. NSG rules are defined with a standard five-tuple definition (source network, source port, the destination network, destination port, protocol) as well as a name, type, priority, protocol, and access (allow or deny).

Azure also offers the Azure Application Gateway service, a web load balancer which includes Web Application Firewall (WAF) functionality. As a Layer 7 firewall with specific understanding of Web technologies, it provides additional security benefits over an NSG-based configuration, and should therefore be employed for web applications instead of or in addition to NSG-based security configurations.

Finally, it should be noted NSGs do not ensure the confidentiality, authenticity, or integrity of traffic; as a result, they should be augmented using the Azure VPN Gateway service, which provides a Virtual Private Network (VPN) in order to allow on-premises resources to securely communicate with Azure-based cloud resources.

Recommendation: Secure Traffic Flow with Azure Network Security Groups

Network Security Groups must be employed for all Azure resources.

Recommendation: Use the Azure Application Gateway for Web Applications

Among other advantages, the Azure Application Gateway includes a web application firewall to protect web applications from malicious attacks.

Recommendation: Leverage Security Tools When Connecting Data Centers

Azure allows connecting on-premises data centers to the Azure with the VPN Gateway. Protocols such as SSTP (Secure Sockets Tunneling Protocol), OpenVPN, and IPsec are available to allow a secure point-to-site connection.

Documentation: Filtering of network traffic is a security and workflow function. Traffic filters can be deployed to direct or restrict access to compute farm subnets based on a production or a source. A network security group is a list of security rules that allow or deny network traffic to resources connected to Azure Virtual Networks (VNet). Azure Network Security Groups are described here: https://docs.microsoft.com/en-us/azure/virtual-network/security-overview.

More information about the Azure Web Application Gateway can be found at the following address: https://azure.microsoft.com/en-us/services/application-gateway/. For information about VPN, gateways see https://azure.microsoft.com/en-us/services/vpn-gateway/.

Isolate virtual appliances in their own subnet

When using virtual appliances, such as a firewall, WAN accelerator, Active directory server, or VPN gateway in Azure, isolate them in their own gateway subnet.

Recommendation: Use a Gateway Subnet with User-Defined Routing

Use a gateway subnet with a user-defined routing mechanism to isolate networking appliances in their own dedicated private network subnets. Specifically, to secure these services and appliances, prevent direct internet connectivity by placing them in a separate subnet with an NSG acting as a firewall. Additionally, close all ports on the appliance or service servers except those necessary for authentication, authorization, and server synchronization.

Documentation: VFX rendering workflows require implementing a secure hybrid network that extends the on-premises network and datacenter to Azure. The user-defined routing mechanism in the gateway subnet filters or blocks all user requests other than those received from the on-premises network.

The Azure network DMZ architecture is described here: https://docs.microsoft.com/en-us/azure/architecture/reference-architectures/dmz/secure-vnet-dmz.

Deploy workload using a multi-tiered architecture

A three-tiered virtual network has front, mid and back-end network segments to create isolation between various types of rendering workflows and digital assets.The front end, which contains web servers in its own subnet, directly faces the internet (e.g., CDN, load balancer). The mid-tier, which contains business logic, does not have direct internet access, either inbound or outbound, and can only be reached from the front-end subnet. The back-end tier, again in its own isolated subnet, contains persistent data such as a database system or storage and can communicate only with the middle-tier. In addition to these subnets, a bastion server is set up for the management of these subnets by administrators.

In VFX rendering, place compute or data storage resources in the backend, while placing authentication and traffic shaping software (e.g., load balancing) servers in the front-end.

Recommendation: Apply a Three-Tiered Network Architecture with VNets

Place workload computes machines in the backend while placing authentication and scheduler software servers in the front-end. Apply different Network Security Groups (NSGs) for each subnet.

Documentation: An Azure VNet is a logical isolation of the Azure dedicated to your subscription. As mentioned before, the VFX rendering workflow should be deployed on separate subscriptions to segregate productions, vendors, or artists.

Azure virtual networks are described here: https://docs.microsoft.com/en-us/azure/virtual-network/virtual-networks-overview. Furthermore, a reference architecture for deployment of N-tier applications is described here: https://docs.microsoft.com/en-us/azure/architecture/reference-architectures/virtual-machines-windows/n-tier.

Create separate VNets for production

Create separate VNets for each production to isolate different workloads from one another. It is acceptable, though discouraged, to use overlapping IP address ranges for these VNets.

Recommendation: Use Azure Network Security Groups

Create network segmentation for various VFX Rendering workflows with NSGs and network appliances.

Documentation: As mentioned before, VFX rendering workflows should be deployed on separate subscriptions to segregate productions, vendors, or artists. Azure virtual networks are described here: https://docs.microsoft.com/enus/azure/virtual-network/virtual-networks-overview; network security concepts and guidance can be found here: https://docs.microsoft.com/en-us/azure/virtual-network/security-overview.

Limit default Network Security Group VNet communications

While having the default network security group and firewall in place when setting up Azure heavily restricts incoming traffic from the Internet, it is more permissive with regard to internal traffic — essentially allowing open communication between all VM instances within the same zone. It is important that customers realize the need to harden the default firewall before allowing an environment to shift into production.

Azure Caveat: Default VM Firewall Too Permissive with Internal Traffic

The firewall settings for the default VMs that Azure offers set to allow all traffic within the internal network.

Recommendation: Use Azure Network Security Groups

Administrators should remove the default firewall rules which allow all inter-VM communication, and replace them with a deny-all policy with specific IP/protocol/port-based exceptions.

Documentation: Azure Network Security groups are the primary method of filtering and restricting traffic within the Azure platform. Azure Network Security Groups are described here: https://docs.microsoft.com/en-us/azure/virtual-network/virtual-networksoverview.

Tightly configure endpoints

Endpoints are a key feature of Azure VMs deployed using the Azure Portal, similar in functionality to Network Address Translation (NAT). An endpoint is configured with a public port (TCP or UDP) and a private port (TCP or UDP); the public port is the port open to the internet, while the private port is the port open on the Azure VM connected to the VNet for a configured application or service.

Recommendation: Configure Endpoints with Access Control Lists for Azure

Configure the endpoints securely through the use of Access Control Lists (ACLs), which restrict access to an endpoint based on a series of allow/deny rules.

Documentation: Though Azure Network Security Groups (NSG) provide traffic filtering at the Azure subnet layer, it is important to apply defense in depth principles, and apply access control at VM endpoints. Endpoint security and port configuration are described here: https://docs.microsoft.com/en-us/azure/virtual-machines/windows/endpoints-in-resource-manager.

Do not use deprecated cryptography for IPSec VPNs

In contrast to SSL/TLS, configuring the set of permitted cipher suites at each end of an IPsec connection can be a manual and time-consuming process. The following concerns affect the configuration of Azure IPsec VPN Gateways:

Azure Caveat: Deprecated Cryptography Supported in IPSec VPNs

The following deprecated ciphers and protocols are supported in Azure IPSec VPN Gateways and should not be used:

- *IKEv1 protocol supported*. Azure VPN Gateways, for compatibility, support both IKE version 1 (introduced in 1998) and IKE version 2 (introduced in 2005). One of the goals of IKEv2 was to improve security over IKEv1, including cryptographic weaknesses[6]. Specifically, the IKEv1 supports 3DES and SHA1 (SHA128) as the encryption and hashing algorithms, respectively.
- *HMAC-MD5, supported (IKEv2).* Azure VPN Gateways allow *HMAC-MD5* to be used for integrity checking. HMAC-MD5 is deprecated due to weaknesses in the underlying MD5 algorithm[7].
- *SHA1 supported (IKEv2).* Azure VPN Gateways allow SHA1 to be used for integrity checking. SHA-1 has been practically broken and is considered insecure and ineffective[8].
- *DES/3DES, supported (IKEv2)*. Azure VPN Gateways allow DES, 3_DES to be used for data encryption. DES is inherently insecure, while Triple-DES has much better security characteristics but is still considered problematic.

Recommendation: Configure Endpoints with Access Control Lists for Azure

Configure IPsec VPN Gateways to avoid the use of deprecated or inherently insecure protocols and modes of operations. Administrators should configure their IPsec clients in accordance with their security policies and avoid using ciphers and security protocols that have been deemed by broken or weak by the industry including but not limited to DES, 3DES, SHA1, and MD5.

This recommendation supports cryptographic security principles.

Documentation: Azure VPNs support three cross-premises and VNet-to-VNet configurations:

1. Site-to-site
2. Point-to-site
3. ExpressRoute

---

[6] RFC 4306 Appendix A, https://tools.ietf.org/html/rfc4306#page-96
[7] http://tools.ietf.org/html/rfc6151
[8] https://security.googleblog.com/2017/02/announcing-first-sha1-collision.html

VFX rendering workflows will require site-to-site VPN gateways or an ExpressRoute connection for data interface between the cloud and on-premises. Planning and design for Azure VPN gateway devices are described here: https://docs.microsoft.com/en-us/azure/vpn-gateway/vpn-gateway-plan-design. Specifically, Azure VPN IPsec and IKE parameters are described here: https://docs.microsoft.com/en-us/azure/vpn-gateway/vpn-gateway-about-vpn-devices.

Use Network Watcher

VFX rendering deployments may be accessed by users across the globe, and it is imperative that network traffic is monitored and logged at some level. Azure Network Watcher is a built-in service that can be used to monitor, diagnose, view metrics, and enable or disable logs for resources in an Azure virtual network. The network watcher service can be used to monitor unstable or inactive endpoints, troubleshoot connections, and other system level failure issues. Though Network Watcher is enabled automatically, it is still necessary to enable NSG flow logs or connection monitor to view traffic flows.

Recommendation: Create and Configure Connection Monitor

Monitor the connection between VM or resources when needed using a connection monitor

Documentation: Logging and monitoring of network traffic is a security and workflow function. Azure Network Watcher described here: https://docs.microsoft.com/en-us/azure/network-watcher.

Azure Policy

Define policies to ensure compliance

Azure Policy allows systems administrators to create compliance checks to ensure resources in their Azure environment follow those rules. To maximize benefits from this service, policies should be designed to provide company and industry guidelines, and best practices are developed.

Recommendation: Create Policies and Audit Compliance

Administrators should configure compliance policies that are tailored to their organization's virtual desktop infrastructure architecture and workflows. In addition, policies should be reviewed regularly to ensure that systems are in compliance with all policies.

Azure Portal

Limit use of personal Microsoft accounts for administration

Access to Azure is possible through two types of Microsoft account (formerly known as Microsoft Live ID) and a work or school account, which is an account stored in Azure Active Directory (AD). Any Azure account can be used to set up and administer the Azure portal. If an employee uses a personal account to set up the Microsoft Cloud infrastructure, then the employer may have difficulty gaining control of the account if the employee later resigns or is terminated. Further, employers are unable to maintain security policies on personal accounts or otherwise audit the accounts' security.

Recommendation: Prohibit Use of Personal Accounts for Administration

It is mandatory for users to sign up for Azure as an organization and use an enterprise account to manage resources in Azure. Enterprise accounts are preferred because they can be centrally managed by the

organization that issued them; they have more features than Microsoft accounts, and are directly authenticated by the Azure AD service. This measure directly supports the identification, authorization and authentication principles.

Documentation: Azure accounts are managed via two separate web interfaces, 1) Azure portal ("Developer Portal") and 2) Usage and Billing. The Azure portal allows a user to configure and use Azure services and are described in detail here: https://docs.microsoft.com/en-us/azure/azure-portal/azure-portal-overview. The usage and billing interface is designed for tracking usage and Azure management actions, including Azure subscriptions.

Use a separate Azure subscription for each production, client and render workflow

Though Azure subscriptions are primarily a container for billing, they can also be used as a security boundary. Subscriptions can be used to limit who can access Azure services associated with that subscription. Each Azure subscription is associated with a single Azure Active Directory domain. The subscription itself governs access to and use of the Azure services that are subscribed to. The subscription administrator manages the subscription service through the Azure Portal.

Recommendation: Use Separate Azure Subscriptions to Segregate Work

Documentation: An understanding of the Azure subscription model is imperative for securing Azure instances. The fundamental unit of work for a developer is a subscription. An Azure subscription grants user access to the Azure services and management portal. Azure subscriptions are described in detail here: https://account.windowsazure.com/Subscriptions.

Logically relate and manage deployments with resource groups

In Azure, most things can be considered a resource. For instance, a virtual machine (VM) is a resource, the network adapter interface (NIC) used by a VM is a resource, the public IP address used by a NIC is a resource, the VNet that the NIC is connected to is a resource, etc. Logically group related resources such as storage accounts, virtual networks, and virtual machines (VMs) to deploy, manage, and maintain them as a single entity.

Azure provides a logical resource group as a container that can hold related resources for an Azure solution. A resource group is a container for resources that share a common lifecycle. It is beneficial to logically group resources that are inter-related or share Rendering workflow components. For example, a virtual machine that depends on a specific storage account should be in the same Batch resource group. The resource group can include all the resources for the VFX rendering workflow, or only those resources that you want to manage as a group.

Recommendation: Use Azure Resource Groups

Use Azure resource groups as logical containers for resources associated with production. Rendering system resources grouped together in resource groups make it easier to manage them as a unit. A resource group can contain resources from multiple regions if the resources belong to the same subscription. When deploying resources, assign them to a logical resource group. It is suggested that resource groups be created for each workflow deployment which will combine storage accounts, batch services, virtual networks, and their subnets, VMs, load balancers, etc.

During the lifecycle of the resource, manage the resources as a function of the related resource group. Specifically, the Azure Resource Manager can be used to manage resource groups.

Documentation: Resource group implementation is described here: https://docs.microsoft.com/en-us/azure/azure-resource-manager/manage-resource-groups-portal.

Avoid concurrent access from multiple locations

A user can be logged into the Azure platform concurrently from multiple computer systems — highly security-conscious enterprises often desire such a feature to reduce the possibility that a user may inadvertently leave an active session unattended in a non-secure location (e.g., conference room computer) and log in again elsewhere.

Recommendation: Enforce Two Factor Authentication

Enable two-factor authentication to limit cases where a user can be logged onto the portal from multiple locations or devices at the same time.

Recommendation: Mandate Manual User Session Termination with Policy

Provide user policy that mandates users immediately terminate their session when access to Azure is no longer needed.

Recommendation: Use Active Directory Reporting

Use AD Reporting to identify multiple logins of the same user account and alert an administrator. Active directory supports risk events — any suspicious action is stored in a record. An administrator should continuously monitor risk event records.

Recommendation: Restrict Access by Locations or IP Addresses

Used in conjunction with Active Directory, restricting access will reduce the potential that AD auditing would miss repeated logins.

Enforce custom session inactivity termination

Users sessions should not be kept alive after a defined timeout period, even if they have not logged out manually.

Recommendation: Mandate Manual User Session Termination Policy

Provide user policy that mandates users immediately terminate their session when access to Azure is no longer needed.

Recommendation: Specify Token Lifetime with Azure Active Directory

Set token lifetimes for all applications in the organization. Use as short a token lifetime is reasonably feasible.

This security control supports robust session management and authorization. Note that this is the same recommendation as to the previous one.

Documentation: Information for AD token lifetimes can be found here: https://docs.microsoft.com/en-us/azure/active-directory/active-directory-configurable-token-lifetimes.

Enforce two-factor authentication for portal access

Multi-factor authentication (MFA) is a method of computer access control in which a user is granted access only after successfully presenting several separate pieces of evidence to an authentication mechanism.

Recommendation: Enable Two-Factor Authentication

Administrators must enforce two-factor (2FA) authentication for management of the Azure environment. If possible, SMS-based 2FA should be avoided in favor of 2FA based on hardware authenticator dongles or Time-Based One Time Password (TOTP) codes.

This recommendation supports robust identification and authentication.

Documentation: The Azure administrator should enable 2FA on all accounts which access the enterprise Azure account. Individual users can enable 2FA by accessing account information here: https://account.microsoft.com.

Deploy topologies and workloads consistently

Free form deploying of resources can lead to configuration errors and open security vulnerabilities. To avoid such issues, use resource templates to instantiate new resources or workloads. Resource templates take the form of JavaScript Object Notation (JSON) files containing descriptions of each resource the template instantiates, with configuration information and other applicable Azure settings.

For resources that are repeatedly deployed in similar configuration but in disjoint rendering workflows, the resource templates will ensure a consistent and known state.

Recommendation: Automate Deployments

Use Microsoft Azure resource manager and Azure policy to deploy resources across rendering workflows.

Documentation: Rendering workflows for an organization will be similar for a vendor or across multiple shows. Deploying a validated setup using a template lowers the risk of service misconfiguration, human error and incomplete setup.

Azure reference architectures are described here: https://docs.microsoft.com/en-us/azure/#pivot=architecture.

Azure Resource Manage templates are described here: https://docs.microsoft.com/en-us/azure/azure-resource-manager. Deploying templates are described here: https://docs.microsoft.com/en-us/azure/azure-resource-manager/resource-group-overview#template-deployment.

Azure Redis Cache

Azure Redis Cache is based on the open source Redis cache. It has its own secure setup items which are outlined below.

Disable non-TLS connections

Redis is designed to be accessed by trusted clients inside trusted environments and natively does not support encryption for data in-transit. The Azure-based Redis database cache service provides access via TLS proxy for an additional layer of protection. This option should always be used.

Recommendation: Disable Non-TLS Connections on Redis Cache Servers

Disable non-TLS ports/connections on all Redis caches to force uses to use TLS-based connections only.

Documentation: Information about the Azure Redis Cache service is located here: https://docs.microsoft.com/en-us/azure/azure-cache-for-redis/cache-how-to-premium-vnet.

Monitor cache performance

Redis is susceptible to attacks triggered by carefully selected inputs from external clients.

Redis Caveat: DoS Attack

An attacker could supply, via a web form, a set of strings that is known to hash to the same bucket into a hash table in order to turn the O(1) expected time (the average time) to the O(N) worst case, consuming more CPU than expected, and ultimately causing a denial of service. This attack is rare, and usually requires an advanced adversary and/or a poorly developed application to be triggered.

Recommendation: Monitor Redis Cache Instances

Monitor the Azure Redis Cache instances using the Azure Monitor service. Azure Monitor can provide cache metrics, e.g., cache hits, misses, number connected clients, used memory, CPU, and cache reads/writes in megabytes per second (MB/s). In addition, it can set alerts when certain conditions are met. Use Azure Monitor to view alerts and setup triggers.

Documentation: Azure Redis monitoring is discussed here: https://docs.microsoft.com/en-us/azure/redis-cache/cache-how-to-monitor.

Azure Security Center

Azure Security Center provides a dashboard with a security score based on policy and compliance, subscription coverage, and regulatory compliance to quantitively measure the security posture of an Azure subscription.

Define Custom Security Policy

Using built-in policies, the security center can continuously assess the configuration of the resources to identify security issues and vulnerabilities. The built-in policies cover basic security concerns around resource health, malware, access, and availability; in order to provide targeted coverage of concerns custom policies should be defined. These custom security policies should focus specific VFX rendering architectures — for example, for hybrid VFX rendering deployment for cloud or hybrid cloud VFX rendering workflows, the security policies should focus on site-to-site VPN status, network traffic rules, protection of data caching compute and storage, license server reverse proxy, storage account key rotations, and key vault use.

Recommendation: Define Custom Security Policy

Use workflow-specific security policies to measure security compliance. General security compliance is a good start in gauging security posture, but it can lead to a false sense of security if not augmented with complete workflow specific security policies.

Documentation: Information about Azure Security Policies is available here: https://docs.microsoft.com/en-us/azure/security-center/security-center-info-protection-policy.

Limit Security Center Data Collection

The security center collects and processes security-related data, including configuration information, metadata, event logs, crash dump files, and more to prevent, detect, and respond to threats. Azure security center sources data from Azure services, network traffic, partner solutions, and virtual machines/servers. Though the data is kept logically separate from data access controls, there is a large amount of contextual data that can provide sensitive information about the workflow or production or even possibly the content being produced. The user should understand and limit the data being provided security center, and configure destruction policies for data that is no longer needed.

Recommendation: Define Separate Data Location

Define a separate workspace where the data collected from the Azure virtual machine including crash dumps, and some types of alert data are stored.

Recommendation: Ensure Diagnostic Data is Regularly Purged

According to Azure documentation, the Azure Security Center collects ephemeral copies of your crash dump files and analyzes them for evidence of exploit attempts and successful compromises — subscription administrators should validate that this data is purged after it is no longer needed.

Documentation: Azure Security Policy is defined here: https://docs.microsoft.com/en-us/azure/security-center/security-center-info-protection-policy.

Define a Security Response Plan

Security center provides great insight on possible vulnerabilities, threats and in near real-time actual incidents or attacks — unfortunately, this data is not very useful if it is not used or applied towards deriving business and security insight. Most organizations define an incident plan after an actual attack has occurred, which is too late to be of use.

Recommendation: Define Security Incident Response Plan

Based on the threat model, deployment architecture and workflow type define a custom incident response plan. The plan should include evidence collection, data process, analysis, assessment, updates, and conclusions.

Documentation: Information about responding to security incidents using Azure resources is available here: https://docs.microsoft.com/en-us/azure/security-center/security-center-incident-response.

Azure Scheduler

The Azure Scheduler service allows developers to declaratively define schedules and jobs, and automatically manages the execution of those jobs based on the schedule. Jobs can involve making HTTP(S) requests, interacting with other Azure services, or publishing messages onto Azure Event Hubs.

Perform Validation of Scheduled Actions

Azure Scheduler allows developers to specify different one-time and recurring schedules for a job. Scheduler creates, maintains, and invokes scheduled work by calling job action services. Jobs can involve interacting with Azure or third-party services.

Recommendation: Validate Services

Validate external job services prior to setting up a scheduled service and use only trusted, vetted third-party services.

Documentation: The Azure Scheduler service is discussed here in detail: https://docs.microsoft.com/en-us/azure/scheduler/scheduler-intro.

Azure SQL Database

Microsoft has an extensive set of articles that discuss various aspects of securing the SQL database service. These articles are located at https://docs.microsoft.com/en-us/azure/sql-database/sql-database-security-overview. These articles discuss creation of server-level and database firewall rules, use of Azure SQL Database transparent data encryption (TDE) to automatically encrypt data at rest, auditing, data masking, backups, low-level security, and SQL database threat detection.

Use separate database instances for production and clients

An Azure subscription can be used to create an unlimited number of database instances. Each Azure database should be assigned to a high-level business unit such as a production, client, or workflow project.

Recommendation: Instantiate Databases in Separate Azure Instances

Instantiate databases in a separate Azure subscription for each VFX rendering workflow or media production, or end clients to limit access. Logically grouping databases through subscriptions will provide administrative autonomy and individualized resource monitoring.

Documentation: The Azure SQL database security model is described here: https://docs.microsoft.com/en-us/azure/sqldatabase/sql-database-security-overview.

Azure Storage

The section pertains to all storage services that Azure offers: blobs, tables, queues, disks, and files. Storage Service encryption (SSE) is turned on by default for all storage services. Data is encrypted before being written to storage and decrypted after the data is read. Microsoft manages the keys by default, but a user can provide their own keys as an option. The Azure Key Vault can be used to manage the keys, or the Azure Key Vault APIs can be used by applications to support user supplied keys. This is documented at: https://docs.microsoft.com/en-us/azure/storage/common/storage-service-encryption-customer-managed-keys.

Use Shared Access Signatures to access storage account resources

An Azure Storage Account is a logical container used to store and access Azure Storage data objects. Each storage account has two (2) Azure generated 512-bit Storage Access Keys (SAK), which are used for authentication when the storage account is accessed. SAKs are similar to a root password in that users with the key have unfettered access to all the storage account's services. To authenticate access to an Azure Storage Account from a client application, an account access key is required. Most client applications should not require access to the entire storage environment, which includes all storage services including Tables, Queues, Files, Blobs and Azure virtual machine disks.

Recommendation: Use Shared Access Signatures (SAS)

A SAS provides granular access to services within a storage account. The goal is to avoid distributing the SAK to other users or applications, hardcoding it, or saving it anywhere in plaintext that is accessible to others. VFX rendering workflows should only require access to a subset of services within a storage account; access via SAS should be sufficient. Additionally, we recommend a policy be put in place to explicitly define controlled SAS expiration times, tokens and source IP address ranges.

Documentation: Azure SAS has multiple use-cases and deployment models. Our recommendation focuses on the use of Shared Access Signature to control access to services within a storage account. Azure SAS is defined here: https://docs.microsoft.com/en-us/azure/storage/storage-dotnet-shared-access-signature-part-1.

Periodically update access keys

To authenticate to an Azure Storage Account from a client application, an account access key is required. Regenerating SAKs can affect associated Azure services (e.g., Batch) that are dependent on the storage account.

Recommendation: Regenerate Storage Access Keys Every 90 Days

The administrator must regenerate storage account access keys periodically. All storage account client services that use the access keys to access the storage account must be updated to use the regenerated key. Key rotation can be handled automatically through the use of Azure Key Vault. Azure Active Directory can also be used to delegate access to individual identities.

Documentation: Azure compute virtual machines, and Batch computes processes rely on storage accounts. Each storage account has a set of access keys. Storage account keys are 512-bit strings created by Azure that, along with the storage account name, can be used to access the data objects in storage. In the context of VFX rendering workflows, these storage account keys should be rotated after a pre-defined period (e.g., 90 days) or after completion of production.

Azure storage security processes are defined here: https://docs.microsoft.com/en-us/azure/storage/storage-securityguide. Documentation on the use of Azure Key Vault to manage Storage Access Keys can be found here: https://docs.microsoft.com/en-us/azure/key-vault/secrets/overview-storage-keys.

Protect disks

All Virtual machines have an operating system disk and possibly multiple attached data disk devices. These devices should be encrypted using a key managed by the Key Vault. Encrypting disks protects against the case of a user account compromise or unauthorized access to physical disks; in these cases, the adversary will not be able to access the disks' contents without additional access to the encryption keys.

Recommendation: Encrypt Disks

Disk encryption should be enabled when Virtual machines or data disks are instantiated.

Documentation: Azure compute virtual machines rely on disk storage for operations system and user data. These storage account disks should be protected at-rest using disk encryption since these disks hold sensitive media content, metadata and possibly PII.

Azure Disk encryption is described here: https://docs.microsoft.com/en-us/azure/security/azure-security-disk-encryption-faq.

Enable Advanced Threat Protection

Advanced Threat Protection attempts to detect anomalous or potentially malicious interactions with a storage account. This automated protection can help detect attacks on storage without human intervention.

Recommendation: Enable Advanced Threat Protection

Administrators should ensure that advanced threat protection is enabled on any applicable storage accounts. Alerts should be sent to an email address that can be reviewed in a timely manner.

Documentation: Information on advanced threat protection is available at: https://docs.microsoft.com/en-us/azure/storage/common/storage-advanced-threat-protection.

Protect Assets with Digital Rights Management (DRM)

Media production workflows rely heavily on ad-hoc and near real-time asset sharing and collaboration. Azure Media service video delivery can be used to deliver video assets within a content production team across the globe. The video assets should be protected when shared in this manner.

Recommendation: Use DRM on Media Assets

Media assets should be delivered using a robust DRM solution designed to prevent unauthorized copying or sharing. Azure Media Services supports popular DRM solutions, including Microsoft PlayReady, Apple FairPlay, and Google Widevine.

Documentation: A guide describing how to enable DRM is located here: https://docs.microsoft.com/en-us/azure/media-services/latest/protect-with-drm.

Require Secure Transfers

Azure storage accounts can support both HTTP and HTTPS protocols for transfers. The former is a plaintext connection that can expose assets to adversaries in privileged network locations, while the latter uses robust encryption to ensure the confidentiality and integrity of network traffic. Depending on the method used to create a storage account, the secure transfer required property may not be enabled.

Recommendation: Enable Secure Transfer Required Property

Administrators should ensure that the secure transfer required property is enabled on any applicable storage accounts.

Documentation: Information on secure transfer is available at: https://docs.microsoft.com/en-us/azure/storage/common/storage-require-secure-transfer.

Implement Strict Access Controls

Like most Azure resources, Azure Managed Disks support role-based access controls that allow administrators to limit which users can read or modify them. Managed disks contain sensitive information like operating system configuration and potentially sensitive media content and should therefore have strict access controls applied to them according to the principle of least privilege.

Recommendation: Use RBAC to Enforce the Principle of Least Privilege

All Azure Managed Disks should be configured to use RBAC to ensure that users with access have the lowest level of access possible in order to perform their job. For instance, users who are tasked with monitoring an Azure cloud deployment should be assigned a role that does not include the ability to modify disks.

Documentation: Microsoft provides a number of different security recommendations for securely deploying Azure storage. These recommendations are listed here: https://docs.microsoft.com/en-us/azure/storage/blobs/security-recommendations.

Azure Cloud Rendering Reference Deployment

ISE developed a deployable reference architecture that incorporates controls from this guide along with Microsoft's best practices for extending an on-premises rendering setup to Azure. This architecture is an extension of the Microsoft developed generic “Implementing a secure hybrid network” reference architecture. The deployment scripts are located in the following gihub directoty: https://github.com/Azure/MediaEntertainment

Architecture

The reference architecture includes provisions to extend an on-premises network and Active Directory (AD) environment to the Azure cloud. Furthermore, the architecture includes a perimeter network between an on-premises network and an Azure virtual network for the protection of network appliances. All outgoing traffic from the VNet is force-tunneled to the external network through the on-premises network, so that it can be audited. The architecture requires a connection to the on-premises datacenter, which can be either a VPN gateway or an ExpressRoute connection.

This architecture depicts both virtual machine and batch processing rendering farm, but in practice, either setup is sufficient. Lastly, the architecture includes a management subnet with a bastion server to monitor the setup or a jump box to traverse to other VMs.

Key Elements of the Architecture:

- On-premises network: a private local-area network implemented in an organization.
- Azure virtual network (VNet): The VNet hosts the application and other resources running in Azure.
- Gateway: the gateway provides connectivity between the routers in the on-premises network and the VNet. User-defined routes handle routing for on-premises traffic that passes to Azure.
- Network appliance: the network appliances are computing device which performs tasks such as allowing or denying access as a firewall, optimizing wide area network (WAN) operations (including network compression), custom routing, or other network functionality.
- Management subnet: this subnet contains VMs that implement management and monitoring capabilities for the components running in the VNet.
- User defined routes define the flow of IP traffic within Azure VNets.
- Active Directory servers: these are domain controllers implementing directory services (AD DS) running as VMs in the cloud. These servers can provide authentication of components running in your Azure virtual network
- Active Directory subnet: The AD DS servers are hosted in a separate subnet. Network security group (NSG) rules protect the AD DS servers and provide a firewall against traffic from unexpected sources.
- Management bastion host: A VM on the network that administrators can use to connect to the other VMs. The bastion host has an NSG that allows remote SSH traffic only from public IP addresses on a white list. The Azure Bastion service can also be used for this purpose.
- Use Azure Advance Threat Protection to monitor and detect any user related threats.

The following figure depicts the important components of an on-premises and cloud-based VFX rendering architecture.

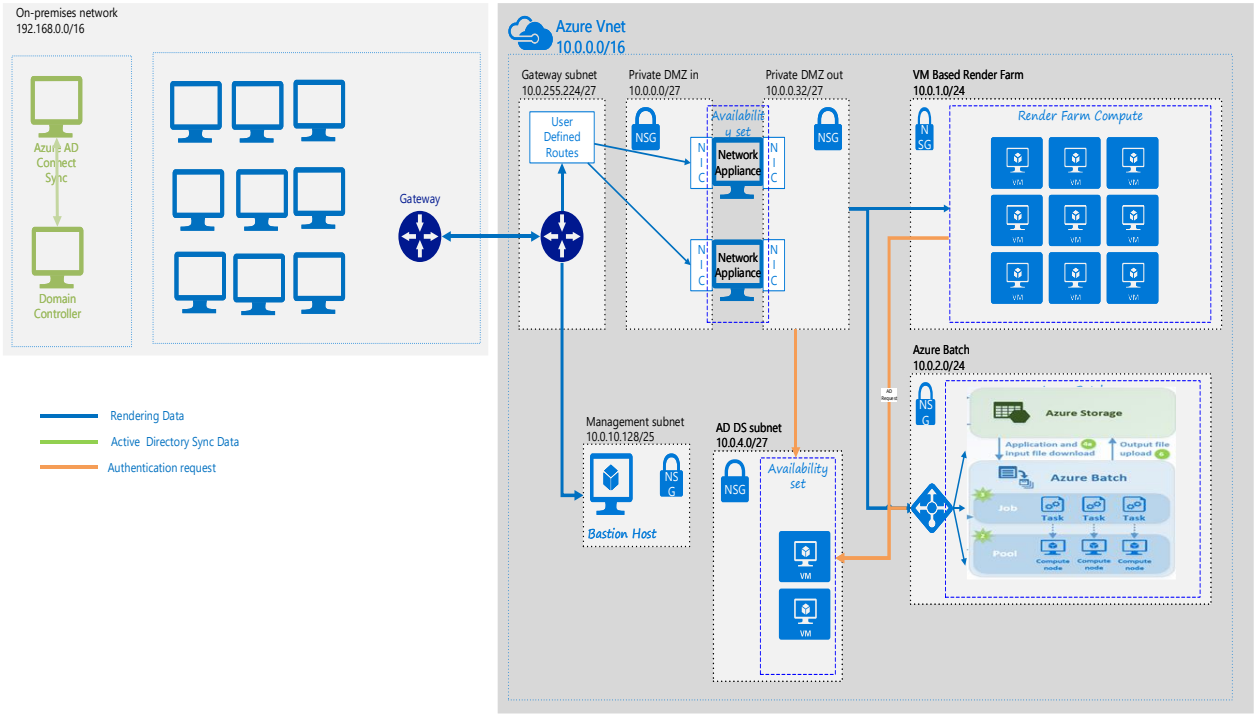

Figure 7. Reference Deployment Architecture

Deployment Scripts

Utilize templates for deployments: The reference implementation depicts an Azure-based native rendering solution. Unfortunately, the free-form deployment of resources can lead to configuration errors and may result in security vulnerabilities. Resource templates or automation can be used to instantiate new resources or workloads to avoid such issues. For resources that are repeatedly deployed in similar configurations but in disjoint workflows across multiple projects, the resource templates will ensure a consistent and known state. Automated deployment can provide easy configuration and, therefore, security by default. It is recommended the cloud deployment manager or similar automated deployment technologies to deploy resources consistently, securely, and robustly.

Deploy pre-configured environments with Azure Blueprints: Azure Blueprints offers several pre-defined sample configurations that adhere to various compliance and regulatory standards. VDI architects should consider using a secure foundation provided by Azure Blueprints when deploying their environment.

Azure offers a Blueprint designed for Motion Pictures Association (MPA) compliance called “MPAA Audit” that ISE reviewed during the creation of this hardening guide. The MPAA Audit Blueprint has several artifacts that require security-enhancing features of other Azure services to be enabled. Some artifacts are focused on Windows virtual machines and SQL Server databases and customers using other platforms will need to add additional artifacts that correspond to their technology stack. MPAA Audit also features a number of parameterized artifacts that require system administrators to manually configure secure values.

The MPAA Audit Blueprint, and other premade Blueprints, can be found under Azure Policy → Blueprints.

About ISE

ISE is an independent security firm headquartered in Baltimore, Maryland. We are dedicated to providing clients proven scientific strategies for defense. On every assessment, our team of analysts and developers use adversary-centric approaches to protect digital assets, harden existing technologies, secure infrastructures, and work with development teams to improve our clients' overall security.

Assessing the client through the eyes of potential attackers allows us to understand possible threats and how to protect against those attacks. Our security analysts are experienced in information technology and product development, which allows them to understand the security problems the client faces at every level. In addition, we conduct independent security research, which allows us to stay at the forefront of the ever-changing world of information security.

Attacks on information systems cannot be stopped, however with robust security services provided by an experienced team, the effects of these attacks can often be mitigated or even prevented. We appreciate the confidence placed in us as a trusted security advisor. Please don't hesitate to get in touch for additional assistance with your security needs.

Independent Security Evaluators, LLC

4901 Springarden Drive

Suite 200

Baltimore, MD 21209

(443) 270-2296

contact@ise.io