Azure Infrastructure Configuration Request

Service: USABC Document Upload Service
Date: February 12, 2026
Requestor: John Bouchard
Subscription: e108977f-44ed-4400-9580-f7a0bc1d3630
Resource Group: rg-rfpo-e108977f

Executive Summary

This document outlines required Azure infrastructure configurations for the USABC Document Upload Service. The service is currently operational but requires administrative-level permissions to complete production-grade security and usability enhancements.

Current Status: ☑ Service deployed and functional at https://usabc-upload.livelyforest-d06a98a0.eastus.azurecontainerapps.io/
Goal: Enable managed identity, malware scanning, custom domain, and monitoring

---

Request 1: Enable Managed Identity for Storage Account Access

Why This is Needed:

Currently, the container app uses storage account keys (similar to passwords) to access Azure Blob Storage. Managed Identity provides keyless, credential-free authentication that is more secure and eliminates the risk of key exposure or rotation issues.

Resources Involved:

- **Container App:** usabc-upload (in rg-rfpo-e108977f)
- **Storage Account:** strfpo5kn5bsg47vvac (in rg-rfpo-e108977f)

Step-by-Step Instructions:

Step 1: Enable System-Assigned Managed Identity on Container App

```
# Enable managed identity
az containerapp identity assign \
  --name usabc-upload \
  --resource-group rg-rfpo-e108977f \
  --system-assigned

# Capture the principal ID (needed for next step)
$principalId = az containerapp identity show \
  --name usabc-upload \
  --resource-group rg-rfpo-e108977f \
  --query principalId -o tsv
```

```
Write-Host "Managed Identity Principal ID: $principalId"
```

Step 2: Grant Storage Permissions to Managed Identity

```
# Get storage account resource ID
$storageId = az storage account show \
  --name strfpo5kn5bsg47vvac \
  --resource-group rg-rfpo-e108977f \
  --query id -o tsv

# Assign "Storage Blob Data Contributor" role
az role assignment create \
  --assignee $principalId \
  --role "Storage Blob Data Contributor" \
  --scope $storageId

# Verify role assignment
az role assignment list \
  --assignee $principalId \
  --scope $storageId \
  --query "[].{Role:roleDefinitionName,Scope:scope}" -o table
```

Step 3: Remove Storage Key from Container App Environment

```
# Update container app to remove AZURE_STORAGE_ACCOUNT_KEY
# (App will automatically use managed identity when no key is present)
az containerapp update \
  --name usabc-upload \
  --resource-group rg-rfpo-e108977f \
  --remove-env-vars AZURE_STORAGE_ACCOUNT_KEY

# Verify the change
az containerapp show \
  --name usabc-upload \
  --resource-group rg-rfpo-e108977f \
  --query "properties.template.containers[0].env" -o table
```

Verification:

After completing these steps, test file upload at: https://usabc-upload.livelyforest-d06a98a0.eastus.azurecontainerapps.io/

Files should upload successfully without any authentication errors in the Container App logs.

Expected Outcome:

- [x] Container app authenticates to storage using managed identity
- [x] No storage keys stored in environment variables
- [x] Improved security posture with credential-free authentication

---

Request 2: Enable Microsoft Defender for Storage - Malware Scanning

Why This is Needed:

The application code includes virus scanning capabilities that integrate with Microsoft Defender for Storage. When enabled, all uploaded files are automatically scanned for malware within 10-30 seconds. Malicious files are quarantined automatically.

Resources Involved:

- **Subscription:** e108977f-44ed-4400-9580-f7a0bc1d3630
- **Storage Account:** strfpo5kn5bsg47vvac (in rg-rfpo-e108977f)

Step-by-Step Instructions:

Step 1: Verify Defender for Storage is Enabled at Subscription Level

```
# Check current Defender status
az security pricing show --name StorageAccounts

# If not enabled, enable it (requires Security Admin role)
az security pricing create \
  --name StorageAccounts \
  --tier Standard \
  --subplan DefenderForStorageV2
```

Expected Output:

```
{
  "pricingTier": "Standard",
  "subPlan": "DefenderForStorageV2"
}
```

Step 2: Enable Malware Scanning on Storage Account

```
# Enable malware scanning via REST API
az rest --method PUT \
  --url "https://management.azure.com/subscriptions/e108977f-44ed-4400-9580-
f7a0bc1d3630/resourceGroups/rg-rfpo-
e108977f/providers/Microsoft.Storage/storageAccounts/strfpo5kn5bsg47vvac/providers
/Microsoft.Security/defenderForStorageSettings/current?api-version=2022-12-01-
```

```
preview" \
  --body '{
    "properties": {
      "isEnabled": true,
      "malwareScanning": {
        "onUpload": {
          "isEnabled": true,
          "capGBPerMonth": 5000
        }
      },
      "sensitiveDataDiscovery": {
        "isEnabled": false
      },
      "overrideSubscriptionLevelSettings": true
    }
  }'
```

Step 3: Create Quarantine Container (for infected files)

```
# Create quarantine container
az storage container create \
  --name quarantine \
  --account-name strfpo5kn5bsg47vvac \
  --auth-mode login

# Set appropriate access policy (private)
az storage container set-permission \
  --name quarantine \
  --account-name strfpo5kn5bsg47vvac \
  --public-access off
```

Alternative: Enable via Azure Portal (if CLI fails)

If the REST API command fails due to permissions, use the Azure Portal:

1. Navigate to: https://portal.azure.com
2. Search for storage account: **strfpo5kn5bsg47vvac**
3. In the left sidebar under **Security + networking** section
4. Click **Microsoft Defender for Cloud**
5. Locate **Malware Scanning** section
6. Toggle **Malware Scanning** to **ON**
7. Set **Capping (GB per month)**: 5000
8. Click **Save**
9. Wait 5-10 minutes for provisioning

Verification:

```
# Test with a file upload
# Upload a file via the web form
# Then check blob tags for scan results:

az storage blob tag list \
  --account-name strfpo5kn5bsg47vvac \
  --container-name usabc-uploads-stage \
  --name "uploads/2026/02/13/<submission-id>.zip" \
  --auth-mode login
```

Look for tags:

- `Malware Scanning scan result`: "No threats found" or "Malicious"
- `Malware Scanning scan time UTC`: Timestamp

Expected Outcome:

- ☑ Malware scanning enabled on storage account
- ☑ Uploaded files automatically scanned within 30 seconds
- ☑ Malicious files quarantined to quarantine container
- ☑ Application returns 403 error for infected files

Cost Impact:

- Base: $10/month
- Scanning: $0.15 per GB scanned
- Monthly cap: 5000 GB
- Estimated: $10-30/month depending on upload volume

Request 3: DNS Configuration for Custom Domain

Why This is Needed:

Currently, the service uses the auto-generated Azure domain. A custom domain (uploads.uscar.org) provides:

- Professional, branded URL
- Easier to remember for users
- Consistent with organizational domain
- Automatic HTTPS certificate from Azure

DNS Provider: (Whoever manages uscar.org DNS)

Step-by-Step Instructions:

Step 1: Add TXT Record for Domain Verification

```
Name:  asuid.uploads.uscar.org
Type:  TXT
```

```
Value: 92AD77A30DEAB51492DBE7D424BC6B8026AA51D21C1856F7498C4D009F697494
TTL:   3600 seconds (1 hour)
```

Why: Azure requires this TXT record to verify ownership of the domain before allowing custom domain binding.

Step 2: Add CNAME Record for Traffic Routing

```
Name:  uploads.uscar.org
Type:  CNAME
Value: usabc-upload.livelyforest-d06a98a0.eastus.azurecontainerapps.io
TTL:   3600 seconds (1 hour)
```

Alternative if CNAME not supported:

```
Name:  uploads.uscar.org
Type:  A
Value: 172.212.19.84
TTL:   3600 seconds (1 hour)
```

Note: CNAME is preferred as it follows Azure infrastructure changes automatically. Use A record only if DNS provider doesn't support CNAME for subdomains.

Step 3: Wait for DNS Propagation

Wait 5-15 minutes, then verify:

```
# Verify TXT record
nslookup -type=TXT asuid.uploads.uscar.org

# Verify CNAME or A record
nslookup uploads.uscar.org
```

Step 4: Bind Custom Domain to Container App

```
# Add custom domain
az containerapp hostname add \
  --hostname uploads.uscar.org \
  --name usabc-upload \
  --resource-group rg-rfpo-e108977f

# Enable managed TLS certificate (automatic HTTPS)
az containerapp hostname bind \
  --hostname uploads.uscar.org \
```

```
  --name usabc-upload \
  --resource-group rg-rfpo-e108977f \
  --environment-certificate-issuer "Managed" \
  --validation-method CNAME
```

Step 5: Verify SSL Certificate

```
# Check certificate status
az containerapp hostname list \
  --name usabc-upload \
  --resource-group rg-rfpo-e108977f \
  -o table
```

Verification:

- Open browser and navigate to: https://uploads.uscar.org
- Verify HTTPS padlock icon (valid TLS certificate)
- Test file upload functionality

Expected Outcome:

- ☑ Service accessible at https://uploads.uscar.org
- ☑ Automatic HTTPS with managed certificate
- ☑ Certificate auto-renews before expiration
- ☑ Old URL still works for backward compatibility

Request 4: Email Configuration for uploads@uscar.org

Why This is Needed:

The service now includes automated email confirmation functionality that sends receipt notifications to file submitters. This requires Azure Communication Services to be configured with a verified email domain (uploads@uscar.org). Additionally, a monitored mailbox is helpful for support inquiries.

Part A: Azure Communication Services for Automated Emails (REQUIRED)

The application sends automated confirmation emails to submitters after they upload files. This requires:

1. Azure Communication Services Email resource
2. DNS verification of the uscar.org domain
3. Configuration of sender address (uploads@uscar.org)
4. Connection string added to container app environment

Step 1: Create Azure Communication Services Email Resource

```
# Create Email Communication Service
az communication email create \
  --name usabc-email-communication \
  --resource-group rg-rfpo-e108977f \
  --data-location "United States"

# Note the resource ID for next steps
$emailServiceId = az communication email show \
  --name usabc-email-communication \
  --resource-group rg-rfpo-e108977f \
  --query id -o tsv
```

Step 2: Add Custom Domain (uscar.org) to Email Service

```
# Add custom domain - this will provide DNS records for verification
az communication email domain create \
  --name uscar-org-domain \
  --email-service-name usabc-email-communication \
  --resource-group rg-rfpo-e108977f \
  --domain-management-type CustomerManaged \
  --custom-domain uscar.org

# Get DNS verification records (you'll need to add these to DNS)
az communication email domain show \
  --name uscar-org-domain \
  --email-service-name usabc-email-communication \
  --resource-group rg-rfpo-e108977f \
  --query verificationRecords -o json
```

Step 3: Add Required DNS Records

Add the following records to uscar.org DNS (records will be provided by previous command):

- **TXT record** for domain verification: `_azuremail-validation.uscar.org`
- **TXT record** for SPF: `v=spf1 include:spf.protection.outlook.com -all`
- **CNAME record** for DKIM: `selector1._domainkey.uscar.org`
- **CNAME record** for DKIM: `selector2._domainkey.uscar.org`

Step 4: Verify Domain and Add Sender Address

```
# After DNS records are propagated, verify domain
az communication email domain initiate-verification \
  --name uscar-org-domain \
  --email-service-name usabc-email-communication \
  --resource-group rg-rfpo-e108977f \
  --verification-type Domain
```

```
# Check verification status
az communication email domain show \
  --name uscar-org-domain \
  --email-service-name usabc-email-communication \
  --resource-group rg-rfpo-e108977f \
  --query domainVerificationStatus

# Once verified, add sender username (uploads@uscar.org)
az communication email domain sender-username create \
  --domain-name uscar-org-domain \
  --email-service-name usabc-email-communication \
  --resource-group rg-rfpo-e108977f \
  --sender-username uploads \
  --username uploads \
  --display-name "USABC Document Uploads"
```

Step 5: Create Communication Services Resource (for connection)

```
# Create Communication Services resource to get connection string
az communication create \
  --name usabc-communication \
  --resource-group rg-rfpo-e108977f \
  --data-location "United States"

# Link email service
az communication email domain association create \
  --connection-string "$(az communication list-key --name usabc-communication --
resource-group rg-rfpo-e108977f --query primaryConnectionString -o tsv)" \
  --email-service-resource-id $emailServiceId

# Get connection string (needed for app config)
az communication list-key \
  --name usabc-communication \
  --resource-group rg-rfpo-e108977f \
  --query primaryConnectionString -o tsv
```

Step 6: Configure Container App with Connection String

```
# Add environment variables for email functionality
az containerapp update \
  --name usabc-upload \
  --resource-group rg-rfpo-e108977f \
  --set-env-vars \
    AZURE_COMMUNICATION_CONNECTION_STRING="<connection-string-from-step-5>" \
    AZURE_COMMUNICATION_SENDER_ADDRESS="uploads@uscar.org"
```

Part B: Email Monitoring Mailbox (OPTIONAL)

While not required for automated confirmations, a monitored mailbox allows support for user inquiries.

Option B1: Microsoft 365 Shared Mailbox

```
# Create shared mailbox (no license needed, up to 50GB)
New-Mailbox -Shared -Name "USABC Document Uploads" -PrimarySmtpAddress
uploads@uscar.org

# Grant permissions to team members
Add-MailboxPermission -Identity uploads@uscar.org -User "admin@uscar.org" -
AccessRights FullAccess
```

Option B2: Email Alias/Forwarding

```
# Create alias on existing mailbox to receive replies
Set-Mailbox -Identity "support@uscar.org" -EmailAddresses
@{add="uploads@uscar.org"}
```

Verification:

1. Upload a file through the web form with a valid email address
2. Check that confirmation email is received
3. Verify email shows correct sender (uploads@uscar.org)
4. Check container app logs for `EMAIL_SENT` messages

Expected Outcome:

- [x] Azure Communication Services configured and verified
- [x] uploads@uscar.org can send automated confirmation emails
- [x] Submitters receive professional email confirmations with file details
- [x] Email delivery tracked in application logs

Cost Impact:

- Azure Communication Services Email: $0.0000625 per email (negligible for expected volume)
- Shared mailbox (if created): Free with Exchange Online

Request 5: Configure Azure Monitor Alerts

Why This is Needed:

Production services require monitoring and alerting for operational issues, security events, and capacity management.

Resources Involved:

- **Container App:** usabc-upload
- **Storage Account:** strfpo5kn5bsg47vvac

Step-by-Step Instructions:

Step 1: Create Action Group for Alert Notifications

```
# Create action group for email notifications
az monitor action-group create \
  --name usabc-upload-alerts \
  --resource-group rg-rfpo-e108977f \
  --short-name usabc-alerts \
  --email-receiver name="Admin" email-address="admin@uscar.org" \
  --email-receiver name="Uploads" email-address="uploads@uscar.org"
```

Step 2: Create Container App Health Alert

```
# Alert when container app is not running
az monitor metrics alert create \
  --name usabc-upload-down \
  --resource-group rg-rfpo-e108977f \
  --scopes "/subscriptions/e108977f-44ed-4400-9580-f7a0bc1d3630/resourceGroups/rg-
rfpo-e108977f/providers/Microsoft.App/containerapps/usabc-upload" \
  --condition "count Replicas < 1" \
  --window-size 5m \
  --evaluation-frequency 1m \
  --action usabc-upload-alerts \
  --description "Container app has no running replicas"
```

Step 3: Create High Error Rate Alert

```
# Alert when HTTP 5xx errors exceed threshold
az monitor metrics alert create \
  --name usabc-upload-errors \
  --resource-group rg-rfpo-e108977f \
  --scopes "/subscriptions/e108977f-44ed-4400-9580-f7a0bc1d3630/resourceGroups/rg-
rfpo-e108977f/providers/Microsoft.App/containerapps/usabc-upload" \
  --condition "total Requests count > 10 where ResultType includes '5'" \
  --window-size 15m \
  --evaluation-frequency 5m \
  --action usabc-upload-alerts \
  --description "High rate of server errors (5xx)"
```

Step 4: Create Malware Detection Alert

```
# Alert when malware is detected in uploaded files
# Note: This monitors blob tags via Log Analytics
# Requires storage account diagnostic settings to be enabled first

# Enable diagnostic logging
az monitor diagnostic-settings create \
  --name storage-logs \
  --resource "/subscriptions/e108977f-44ed-4400-9580-
f7a0bc1d3630/resourceGroups/rg-rfpo-
e108977f/providers/Microsoft.Storage/storageAccounts/strfpo5kn5bsg47vvac" \
  --logs '[{"category": "StorageRead", "enabled": true}, {"category":
"StorageWrite", "enabled": true}]' \
  --workspace "/subscriptions/e108977f-44ed-4400-9580-
f7a0bc1d3630/resourceGroups/rg-rfpo-
e108977f/providers/Microsoft.OperationalInsights/workspaces/<workspace-name>"
```

Note: Replace `<workspace-name>` with your Log Analytics workspace name, or create one if it doesn't exist.

Step 5: Create Storage Capacity Alert

```
# Alert when storage usage exceeds 80%
az monitor metrics alert create \
  --name usabc-storage-capacity \
  --resource-group rg-rfpo-e108977f \
  --scopes "/subscriptions/e108977f-44ed-4400-9580-f7a0bc1d3630/resourceGroups/rg-
rfpo-e108977f/providers/Microsoft.Storage/storageAccounts/strfpo5kn5bsg47vvac" \
  --condition "total UsedCapacity > 80000000000" \
  --window-size 1h \
  --evaluation-frequency 15m \
  --action usabc-upload-alerts \
  --description "Storage usage exceeds 80GB"
```

Expected Outcome:

- ☑ Email notifications for service health issues
- ☑ Alerts for malware detections
- ☑ Proactive capacity management
- ☑ Reduced mean time to detection (MTTD)

Request 6: Enable Container App Continuous Deployment

Why This is Needed:

Currently, deployments are manual. Setting up continuous deployment from the container registry enables automatic updates when new versions are pushed.

Step-by-Step Instructions:

Enable Continuous Deployment

```
# Configure container app to poll registry for updates
az containerapp registry set \
  --name usabc-upload \
  --resource-group rg-rfpo-e108977f \
  --server acrrfpoe108977f.azurecr.io \
  --username acrrfpoe108977f \
  --password "<registry-password>"

# Note: Password can be retrieved with:
# az acr credential show --name acrrfpoe108977f --query "passwords[0].value" -o
tsv
```

Expected Outcome:

- [x] New container versions automatically deployed
- [x] Zero-downtime rolling updates
- [x] Simplified deployment process

Request 7: Configure Backup and Disaster Recovery

Why This is Needed:

Protect against accidental deletion and enable point-in-time recovery for uploaded documents.

Step-by-Step Instructions:

Step 1: Enable Blob Soft Delete

```
# Enable soft delete with 30-day retention
az storage account blob-service-properties update \
  --account-name strfpo5kn5bsg47vvac \
  --resource-group rg-rfpo-e108977f \
  --enable-delete-retention true \
  --delete-retention-days 30

# Enable container soft delete
az storage account blob-service-properties update \
  --account-name strfpo5kn5bsg47vvac \
  --resource-group rg-rfpo-e108977f \
  --enable-container-delete-retention true \
  --container-delete-retention-days 30
```

Step 2: Enable Blob Versioning

```
# Enable versioning for point-in-time recovery
az storage account blob-service-properties update \
  --account-name strfpo5kn5bsg47vvac \
  --resource-group rg-rfpo-e108977f \
  --enable-versioning true
```

Step 3: Configure Lifecycle Management (Optional Cost Optimization)

```
# Create lifecycle policy to move old data to Cool tier
az storage account management-policy create \
  --account-name strfpo5kn5bsg47vvac \
  --resource-group rg-rfpo-e108977f \
  --policy '{
    "rules": [
      {
        "enabled": true,
        "name": "move-old-to-cool",
        "type": "Lifecycle",
        "definition": {
          "actions": {
            "baseBlob": {
              "tierToCool": {
                "daysAfterModificationGreaterThan": 90
              },
              "tierToArchive": {
                "daysAfterModificationGreaterThan": 365
              }
            }
          },
          "filters": {
            "blobTypes": ["blockBlob"],
            "prefixMatch": ["uploads/"]
          }
        }
      }
    ]
  }'
```

Expected Outcome:

- ☑ 30-day recovery window for deleted files
- ☑ Point-in-time restore capability
- ☑ Protection against accidental deletion
- ☑ Cost optimization through tiering

Summary of Required Actions

#	Task	Estimated Time	Impact	Priority
1	Enable Managed Identity	15 minutes	High security improvement	High
2	Enable Malware Scanning	20 minutes	Critical security feature	High
3	Configure Custom Domain DNS	30 minutes	User experience improvement	Medium
4	Set up uploads@uscar.org	20 minutes	Professional communication	Medium
5	Configure Monitoring Alerts	30 minutes	Operational visibility	High
6	Enable Continuous Deployment	10 minutes	Simplified operations	Low
7	Configure Backup/DR	15 minutes	Data protection	High

Total Estimated Time: 2.5 - 3 hours

Testing and Validation Checklist

After completing all requests, verify:

- ☐ File uploads work with managed identity (check container logs)
- ☐ Malware scan results appear in blob tags
- ☐ Custom domain redirects to HTTPS correctly
- ☐ Email to uploads@uscar.org is received
- ☐ Alert emails are received when test condition triggered
- ☐ Soft-deleted blob can be recovered from Azure Portal
- ☐ Health endpoint returns 200: https://uploads.uscar.org/health

---

Support Contacts

Requestor: John Bouchard (johnbouchard@icloud.com)
Service Documentation: See repository /docs folder
Container App Logs: Azure Portal → rg-rfpo-e108977f → usabc-upload → Log stream

---

Additional Notes

Cost Impact Summary:

- Managed Identity: No additional cost
- Defender for Storage: +$10-30/month
- Custom Domain: No additional cost (included)
- Email (Shared Mailbox): No additional cost
- Monitoring: Included in Container Apps pricing
- Continuous Deployment: No additional cost
- Backup/Versioning: ~$0.01/GB/month for versioning storage

Total Additional Monthly Cost: ~$10-35/month (primarily Defender scanning)

Security Compliance: This configuration achieves:

- ☑ Zero stored credentials (managed identity)
- ☑ Automated malware protection
- ☑ TLS 1.2+ encrypted traffic
- ☑ 30-day backup retention
- ☑ Operational monitoring and alerting
- ☑ Professional branding and communication

Questions or Issues?

If any of these steps fail or require clarification, please contact the requestor or refer to the service repository documentation at:

- Virus Scanning: `VIRUS_SCANNING.md`
- Third-Party Integration: `THIRD_PARTY_INTEGRATION.md`
- Deployment Guide: `deploy-guide.md`