Automating Root Credential Management Across AWS Organization

Context and purpose:

In AWS environments, securing root user credentials is crucial due to their extensive access privileges. AWS recommends minimizing or tightly controlling root access keys, signing certificates, and MFA devices to mitigate security risks. This script automates the process of removing these sensitive credentials across AWS member account within an organization, enhancing security, and significantly reducing the time and potential for manually managing these sensitive credentials.

Short Description:

The provided script enables Root Credential Management and automates the detection and deletion of root user credentials such as access keys, signing certificates, and MFA devices across the organization, enhancing security by reducing potential attack vectors.

Pre-requisites:

1. You must use credentials from an AWS Organizations management account for IAM to call AssumeRoot. You cannot use root user credentials to make this call.

2. Ensure you have the necessary permissions to perform these actions across the organization. (iam:EnableOrganizationsRootCredentialsManagement, iam:EnableOrganizationsRootSessions, sts:AssumeRoot, organizations:EnableAwsServiceAccess, "organizations:DescribeAccount", "organizations:DescribeOrganization", "organizations:ListAccounts")

Script:

```
#!/bin/bash

# Set your AWS Organizations ID
ORGANIZATION_ID="YOUR_ORGANIZATION_ID "

# Function to process a single member account
process_account() {
    local MEMBER_ACCOUNT_ID=$1
    echo "Processing account: $MEMBER_ACCOUNT_ID"
    echo "------------------------------------------------------------"

    ACCOUNT_STATUS=$(aws organizations describe-account --account-id
$MEMBER_ACCOUNT_ID --query 'Account.Status' --output text)
    if [ "$ACCOUNT_STATUS" == "SUSPENDED" ]; then
```

```
        echo "Account $MEMBER_ACCOUNT_ID is suspended. Skipping..."
        echo "------------------------------------------------------------"
        return
    fi

    # Attempt to assume a temporary root session in the member account
    echo "Attempting to assume root..."
    ASSUME_ROOT_OUTPUT=$(aws sts assume-root --target-principal $MEMBER_ACCOUNT_ID --
task-policy-arn arn=arn:aws:iam::aws:policy/root-task/IAMDeleteRootUserCredentials
2>&1)
    if [[ $? -ne 0 ]]; then
        echo "Failed to assume root: $ASSUME_ROOT_OUTPUT"
        echo "Skipping further operations for this account."
        echo "------------------------------------------------------------"
        return
    fi

    export AWS_ACCESS_KEY_ID=$(echo $ASSUME_ROOT_OUTPUT | jq -r
'.Credentials.AccessKeyId')
    export AWS_SECRET_ACCESS_KEY=$(echo $ASSUME_ROOT_OUTPUT | jq -r
'.Credentials.SecretAccessKey')
    export AWS_SESSION_TOKEN=$(echo $ASSUME_ROOT_OUTPUT | jq -r
'.Credentials.SessionToken')
    echo "AssumeRoot has been successfully performed."
    echo ""

    # Verify IAM entity
    echo "Below is get-caller-identity output for AssumeRoot"
    aws sts get-caller-identity
    echo "------------------------------------------------------------"

    # Verify the root user credentials
    echo "Below is get-user output"
    aws iam get-user
    echo "------------------------------------------------------------"

    # Check and handle Login Profile
    echo "Checking for root user's login profile..."
    echo ""
    LOGIN_PROFILE_OUTPUT=$(aws iam get-login-profile 2>&1)
    if echo "$LOGIN_PROFILE_OUTPUT" | grep -q "NoSuchEntity"; then
        echo "No login profile found for root user."
    else
        echo "$LOGIN_PROFILE_OUTPUT"
        echo "Deleting existing login profile..."
        aws iam delete-login-profile
        echo "Login profile has been deleted successfully."
    fi
```

```
    echo "------------------------------------------------------------"

    # List and delete access keys if present
    echo "Listing Access Keys for root user:"
    echo ""
    ACCESS_KEYS=$(aws iam list-access-keys | jq -r
'.AccessKeyMetadata[].AccessKeyId')
    if [ -z "$ACCESS_KEYS" ]; then
        echo "No access keys found for root user."
    else
        for ACCESS_KEY in $ACCESS_KEYS; do
            echo "Deleting Access Key: $ACCESS_KEY"
            aws iam delete-access-key --access-key-id $ACCESS_KEY
            echo "Access Key $ACCESS_KEY has been deleted successfully."
        done
    fi
    echo "------------------------------------------------------------"

    # List and delete signing certificates if present
    echo "Listing Signing Certificates for root user:"
    echo
    SIGNING_CERTS=$(aws iam list-signing-certificates | jq -r
'.Certificates[].CertificateId')
    if [ -z "$SIGNING_CERTS" ]; then
        echo "No signing certificates found for root user."
    else
        for CERT_ID in $SIGNING_CERTS; do
            echo "Deleting Signing Certificate: $CERT_ID"
            aws iam delete-signing-certificate --certificate-id $CERT_ID
            echo "Signing Certificate $CERT_ID has been deleted successfully."
        done
    fi
    echo "------------------------------------------------------------"

    # List and deactivate MFA devices if present
    echo "Listing MFA Devices for root user:"
    echo
    MFA_DEVICES=$(aws iam list-mfa-devices | jq -r '.MFADevices[].SerialNumber')
    if [ -z "$MFA_DEVICES" ]; then
        echo "No MFA devices found for root user."
    else
        for MFA_DEVICE in $MFA_DEVICES; do
            echo "Deactivating MFA Device: $MFA_DEVICE"
            aws iam deactivate-mfa-device --serial-number $MFA_DEVICE
            echo "MFA Device $MFA_DEVICE has been deactivated successfully."
        done
    fi
    echo "------------------------------------------------------------"
```

```
    # Unset temporary credentials
    unset AWS_ACCESS_KEY_ID AWS_SECRET_ACCESS_KEY AWS_SESSION_TOKEN
}

# Enable service access for IAM in AWS Organizations
aws organizations enable-aws-service-access --service-principal iam.amazonaws.com
echo "Service access for IAM has been enabled."
echo "------------------------------------------------------------"

# Enable "Root Credentials Management" in IAM
aws iam enable-organizations-root-credentials-management
echo "Root Credentials Management has been enabled."
echo "------------------------------------------------------------"

# Enable "Root Sessions" in IAM
aws iam enable-organizations-root-sessions
echo "Root Sessions has been enabled."
echo "------------------------------------------------------------"

# Get management account ID
MANAGEMENT_ACCOUNT_ID=$(aws organizations describe-organization --query
'Organization.MasterAccountId' --output text)

# Get all member accounts excluding the management account
MEMBER_ACCOUNTS=$(aws organizations list-accounts --query
'Accounts[?Id!=`'$MANAGEMENT_ACCOUNT_ID'`].Id' --output text)

for ACCOUNT_ID in $MEMBER_ACCOUNTS
do
    if [ "$ACCOUNT_ID" != "$MANAGEMENT_ACCOUNT_ID" ]; then
        process_account $ACCOUNT_ID
    else
        echo "Skipping Management Account: $ACCOUNT_ID"
        echo "------------------------------------------------------------"
    fi
done

echo "All member accounts have been processed."
```

The script automatically includes all active member accounts within the organization. There's no need to manually input individual account IDs. It also excludes the management account and any suspended accounts.

Notes:

1. Kindly, update variable “YOUR_ORGANIZATION_ID” with actual Organization ID in above script.

2. Save it as a file (e.g., delete_root_credentials.sh).
   Make it executable: chmod +x delete_root_credentials.sh

3. Run the script: ./delete_root_credentials.sh

References:

[1] https://docs.aws.amazon.com/IAM/latest/UserGuide/id_root-enable-root-access.html

[2] https://docs.aws.amazon.com/IAM/latest/UserGuide/id_root-user-privileged-task.html

[3] https://docs.aws.amazon.com/STS/latest/APIReference/API_AssumeRoot.html