Secure Multi-Server Active Directory Synchronization

This project provides a robust framework for synchronizing Active Directory (AD) users, security groups, and identity attributes between disconnected or air-gapped environments. It leverages **OpenBao** as a centralized security vault to provide "Zero-Knowledge" transport of sensitive credentials.

1. System Architecture

The system follows a **Source-to-Target** synchronization pattern designed for high-security environments where servers cannot communicate directly over a network.

Primary Operational Modes

- **Export (Source Server):** Scans a designated Target OU, extracts user metadata, generates secure passwords (if missing), and encrypts all sensitive data into a portable payload.
- **Import (Target Server):** Restores the cryptographic environment, verifies the integrity of the payload, decrypts user credentials, and reconciles the local Active Directory to match the source state.

2. Component Directory

File	Role	Description
Initialize-SyncServer.ps1	**Infrastructure**	Prepares the Windows environment: creates directories, sets NTFS permissions, installs the OpenBao Windows Service, and configures local firewalls.
Invoke-BaoAutomation.ps1	**Security**	Automates the Vault lifecycle: initializes the master keys, unseals the vault, enables the Transit engine, and securely ingests AD admin credentials.

Sync-AD-Transport.ps1	**Engine**	The primary logic driver. Handles AD attribute mapping, group membership reconciliation, and the encryption/decryption workflow.

3. Credential Provisioning (ad_creds_temp.json)

To perform Active Directory operations, the system requires a Domain Admin or a delegated Service Account. These credentials must be ingested into the Vault using a temporary JSON file.

Sample File Structure

Create a file at C:\ADSync\Sync\ad_creds_temp.json with the following content:

```
{
   "username": "DOMAIN\\SyncServiceAccount",
   "password": "YourSecurePassword123!"
}
```

Security Lifecycle

1. **Creation:** The administrator manually creates this file on both the Source and Target servers.
2. **Ingestion:** Running Invoke-BaoAutomation.ps1 detects this file and moves the credentials into the Vault's internal KV-V2 encrypted storage.
3. **Automated Cleanup:** Once successfully stored in the Vault, the script **permanently deletes** ad_creds_temp.json to prevent plaintext passwords from remaining on the disk.

4. Account Access Requirements

The system interacts with two security layers: the **Local OS Context** (running the script) and the **AD Authentication Context** (performing the sync).

Local Execution Context (Task Scheduler)

The account used to run the Scheduled Task requires:

- **Log on as a batch job** rights.
- **Full Control** over the C:\ADSync directory tree.

- **Administrator Privileges** on the local member server (required to interact with the OpenBao service and write to the Windows Event Log).

AD Synchronization Context (Vaulted Credentials)

You have two options for the credentials stored inside OpenBao:

Option A: Domain Administrator (Default)

The simplest configuration. Provides full authority to create, delete, and modify users and groups across the domain.

Option B: Sync Service Account (Least Privilege)

To limit security exposure, you may use a dedicated Service Account. This account must be delegated the following permissions on the **Target OU**:

- **Create/Delete User Objects**
- **Write All Properties** (specifically for synchronization of attributes like Title, Department, etc.)
- **Reset Password**
- **Read/Write Group Membership** (to manage security group reconciliation)

5. Automation via Scheduled Task

To ensure the Target environment remains in sync with the Source, the Sync-AD-Transport.ps1 script should be scheduled to run automatically.

Configuration Steps

1. **Open Task Scheduler:** Run taskschd.msc as Administrator.
2. **Create New Task:** Name it AD-Sync-Import.
3. **Security Options:**
    - Select the Local Service Account or Admin.
    - Select **"Run whether user is logged on or not"**.
    - Check **"Run with highest privileges"**.
4. **Triggers:** Set to "Daily" or at a specific interval.
5. **Actions:**
    - **Program/script:** powershell.exe
    - **Add arguments:** -ExecutionPolicy Bypass -File "C:\ADSync\Sync-AD-Transport.ps1"
    - **Start in:** C:\ADSync

6. Security Model & OpenBao Integration

The core philosophy of this project is that **no sensitive data should exist in plain text outside of Active Directory or the Vault memory.**

OpenBao: The "Zero-Trust" Vault

OpenBao acts as the "Root of Trust." It is responsible for:

1. **Secret Storage (KV-V2):** Encrypting the AD Administrator credentials at rest.
2. **Transit Engine:** Providing "Encryption-as-a-Service," where data is encrypted/decrypted without the script seeing the underlying keys.

Cryptographic Implementation

- **Asymmetric Encryption (RSA-4096):** The Export side uses the Public Key to encrypt passwords, while the Import side uses the Private Key stored securely in the local Vault.
- **Integrity (HMAC-SHA256):** Every export is signed. The Import engine verifies the signature before processing to prevent tampering.

7. Setup & Operation Workflow

Phase 1: Preparation (Both Servers)

1. Copy scripts to C:\ADSync.
2. Place bao.exe in C:\ADSync\OpenBao.
3. Execute Initialize-SyncServer.ps1 as Administrator.

Phase 2: Credential Provisioning

1. Create the ad_creds_temp.json file as described in Section 3.
2. Run Invoke-BaoAutomation.ps1 to ingest credentials and delete the temporary file.

Phase 3: Daily Operation

1. **On Source:** Run Sync-AD-Transport.ps1.
2. **Transfer:** Move the contents of C:\ADSync\Export to the Target server's C:\ADSync\Import folder.
3. **On Target:** The Scheduled Task will automatically process the files.

8. AD Hardening & Safety

- **CannotChangePassword:** Prevents users from falling out of sync.
- **PasswordNeverExpires:** Prevents account lockouts due to sync intervals.
- **Group Reconciliation:** Automatically removes users from groups at the target if they are removed at the source.

9. Automated Unsealing on Reboot

By design, OpenBao starts in a **Sealed** state after a reboot. To automate unsealing in a disconnected environment:

Scripted Unseal (Local)

You can use the existing Invoke-BaoAutomation.ps1 to unseal the vault automatically at startup.

1. **Create a Startup Task:** In Task Scheduler, create a task named OpenBao-AutoUnseal.
2. **Trigger:** Set to **"At startup"**.
3. **Action:** - **Program:** powershell.exe
    - **Arguments:** -ExecutionPolicy Bypass -File "C:\ADSync\Invoke-BaoAutomation.ps1"
4. **Delay:** Add a 30-second delay to the trigger to ensure the Windows Service has fully initialized before the script attempts to unseal.

Security Note: This method relies on vault_keys.json being present on the disk. Ensure this file is protected with NTFS permissions so only the **SYSTEM** account can read it.