

# 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	<b>Infrastructure</b>	Prepares the Windows environment: creates directories, sets NTFS permissions, installs the OpenBao Windows Service, and configures local firewalls.
Invoke-BaoAutomation.ps1	<b>Security</b>	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	<b>Engine (v4.3)</b>	The primary logic driver. Handles AD attribute mapping, group membership reconciliation, and the asymmetric encryption/decryption workflow.
Update-VaultPassword.ps1	<b>Utility</b>	Interactive script for administrators to manually update a user's password in the Vault and generate a corresponding local log file.

### 3. Cryptographic Security Deep-Dive

The synchronization engine implements a high-entropy security model to ensure that sensitive data remains encrypted even if the physical transport media (USB drive, etc.) is intercepted.

#### Asymmetric Implementation (RSA-4096)

The system uses the OpenBao **Transit Engine** configured for rsa-4096.

- **The Public Key:** Is used by the **Source Server** to encrypt user passwords. Public keys can only encrypt; they cannot be used to reverse the process.
- **The Private Key:** Is stored strictly within the **Target Server's** OpenBao instance. This key is required for decryption.

#### Scenario: What if the Export files are stolen?

If an attacker intercepts the C:\ADSync\Export folder (containing the JSON payload, the HMAC signature, and the Key Backup), they face three massive cryptographic barriers:

1. **The RSA Barrier:** The AD\_State\_Export.json contains "Ciphertext" (e.g., vault:v1:base64...). Without the RSA Private Key, this data is mathematically impossible to decrypt.
2. **The Transit Key Encryption:** The transport-key.backup file is **not** a raw private key. It is an encrypted "blob" exported by OpenBao. To restore this key, an attacker would need a functioning OpenBao instance initialized with the **exact same Master Keys** used by the original vault.
3. **The Signature Barrier:** The AD\_State\_Export.hmac file ensures integrity. Any attempt to modify the user data in the JSON file will result in an HMAC mismatch, causing the Import script to immediately reject the payload.

## Requirements for Decryption

To successfully decrypt the passwords, an attacker would need:

- The **JSON Payload** (AD\_State\_Export.json).
- The **Key Backup** (transport-key.backup).
- The **Vault Unseal Keys** (vault\_keys.json) from either the Source or Target server to unseal a Vault instance and restore the key.

## 4. Administrative Utilities

### Manual Password Updates (Update-VaultPassword.ps1)

In scenarios where a specific user's password must be changed outside of the automated sync cycle, administrators can use this utility.

- **Functionality:** Prompts for a UserID and a new password (masked input), updates the KV-V2 store in OpenBao, and writes a log file to C:\ADSync\Users\.
- **Audit Trail:** Every manual update is logged to the ADSync Windows Event Log (Event ID 2000), capturing the administrator's identity and the timestamp.
- **Consistency:** Uses the same pathing and encryption logic as the main engine to ensure the updated password is included in the next Export cycle.

## 5. Credential Provisioning (ad\_creds\_temp.json)

To perform AD operations, the system requires a Domain Admin or delegated Service Account.

### Sample File Structure

Create C:\ADSync\Sync\ad\_creds\_temp.json:

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

### Security Lifecycle

1. **Creation:** Admin creates the file on Source and Target.
2. **Ingestion:** Invoke-BaoAutomation.ps1 moves credentials into the Vault's internal KV-V2 storage.
3. **Cleanup:** The script **permanently deletes** the JSON file once ingested.

## 6. Account Access Requirements

## Local Execution Context (Task Scheduler)

The account running the Scheduled Task requires:

- **Log on as a batch job** rights.
- **Full Control** over C:\ADSync.
- **Local Administrator Privileges**.

## AD Synchronization Context (Vaulted Credentials)

The account stored inside OpenBao requires the following delegated permissions on the Target OU:

- **Create/Delete User Objects**
- **Write All Properties** (for attribute sync)
- **Reset Password**
- **Read/Write Group Membership**

## 7. Automation via Scheduled Task

1. **Task Name:** AD-Sync-Import.
2. **Security:** "Run whether user is logged on or not" + "Run with highest privileges".
3. **Action: \* Program:** powershell.exe
  - **Arguments:** -ExecutionPolicy Bypass -File "C:\ADSync\Sync-AD-Transport.ps1"
  - **Start in:** C:\ADSync

## 8. Setup & Operation Workflow

### Phase 1: Preparation (Both Servers)

1. Copy scripts and bao.exe to C:\ADSync.
2. Run Initialize-SyncServer.ps1 as Administrator.

### Phase 2: Credential Provisioning

1. Create ad\_creds\_temp.json.
2. Run Invoke-BaoAutomation.ps1 to unseal and ingest.

### Phase 3: Daily Operation

1. **Source:** Run Sync-AD-Transport.ps1 (it will auto-detect Export mode).
2. **Transfer:** Move C:\ADSync\Export contents to Target's C:\ADSync\Import.
3. **Target:** The Scheduled Task processes the files (auto-detects Import mode).

## 9. AD Hardening & Safety

- **CannotChangePassword:** Set to True to maintain sync parity.
- **PasswordNeverExpires:** Set to True to prevent lockout.
- **Group Reconciliation (v4.2):** Implements a HashSet keep-list to ensure groups created

in the same session are not flagged as "stale" and deleted.