International Journal of Science and Research (IJSR)
ISSN: 2319-7064
SJIF (2022): 7.942

Active Directory Service Interfaces (ADSI): Provisioning / De-Provisioning Access and Ensuring Accurate User Identity and Access across All the Forest Using IAM

Sampath Talluri

Department of Computer Science, Western Michigan University, 1903 W Michigan Ave, Kalamazoo, MI 49008, USA

Abstract: ***Creating user identities and access rights in organizations is a parliament activity to ensure data security and operational efficacy. ADSI is a set of COM(Component Object Model) interfaces provided by Microsoft for working with directory services, specifically the Windows Active Directory(AD) service. AD is a directory service used by Windows networks to store information about resources and users hierarchically. ADSI allows developers and administrators to programmatically interact and manage objects in the Active Directory, such as users, groups, computers, and organizational units (OU). It provides a consistent way to perform tasks like querying for objects, creating or deleting objects, and modifying object attributes within the directory of different forests or child forests. We must use the ADSI interface to connect to multiple AD forests and Domain Controllers (DC). ADSI is used as a Global catalog to query the Directory services. Despite these advantages, it also identifies challenges, including configuration complexity, trust relationships, and the need for security and maintenance efforts. Organizations must carefully address these challenges to harness the full potential of ADSI integration in modern identity and access management strategies.***

Keywords: Provisioning, de-provisioning, Active Directory Service Interface (ADSI), AD, Access, Users, Accounts, COM, Forest and DN

1. Introduction

In a securely driven world, managing access to applications, data, and services is critical and crucial for every organization. The efficient way of managing the user accounts and necessary permission is paramount. Active Directory Service Interfaces (ADSI) help to connect to all the directory services without any gaps in the connectivity. ADSI is a critical component in identity and access management, offering a standardized and robust means of communicating with directory services, particularly within Microsoft's Active Directory environment when an organization has multiple Forest and DC to manage. Provisioning access confines creating user accounts, configuring permissions, and ensuring users can seamlessly interact with the resources essential to their roles without interruption.

The incongruity of provisioning and de-provisioning access is a fundamental challenge faced by organizations of all sizes from an Audit standpoint, demanding a balance between enabling access for authorized users while safeguarding against unauthorized access. ADSI helps to manage all the users from different environments into a global catalog. Identity and Access Management (IAM) tool makes things easy from an implementation standpoint. This paper delves into the multifaceted landscape of provisioning and de-provisioning access, focusing on how ADSI empowers organizations to streamline these critical IAM processes.

We will explore the core principles of ADSI, its services in provisioning and de-provisioning access, and the implications of this technology on the broader spectrum of IAM. Examining the principles and methodologies of ADSI in this context, this paper seeks to provide valuable insights into how modern organizations can optimize their access management strategies, aligning them with the ever-evolving growth of technology and security. ADSI can help organizations to make their process more secure and connected. Additionally, I will explore the evolving terrain of access management, considering the influence of emerging technologies and the evolving threat in cybersecurity.

2. Literature Review

The field of information technology and security has got the most attention on user access and permission management. I am integrating ADSI with IAM tools to provision and de-provision the access to users' accounts efficiently.

Challenges in access management are multifaceted, with elaborateness arising from evolving technologies, regulatory requirements, and the persistent threat of unauthorized access. Research highlights the growing concern over access management challenges, quoting the need for automated and standardized solutions [1]. ADSI replaces the older domain controller (DC) concept. As such, it combines both the directory and the control functionality [2]. I will also discuss the ports used for this ADSI integration. ADSI is one component of the Windows Open Services Architecture (WOSA) Open Directory Service Interfaces (ODSI) [3]. ADSI's adaptability to the ever-changing threat landscape is a characteristic that deserves further investigation, as it holds the potential to offer advanced access management solutions.

3. Methodology

Determine the research design suitable for assessing the integration of ADSI with Saviynt. An exploratory or experimental research design may be appropriate to evaluate this integration's technical aspects and outcomes. ADSI is a single set of directory service interfaces for managing networks & resources; developers and Engineers use ADSI services to manage directory resources from any network or the forest [4]. Looking at Figure 1, we will clearly understand how the ADSI agent is deployed on the IIS Server in a Windows virtual machine (VM). This server helps with .NET SDK communicating to multiple forests in AD and creates a centralized target application.

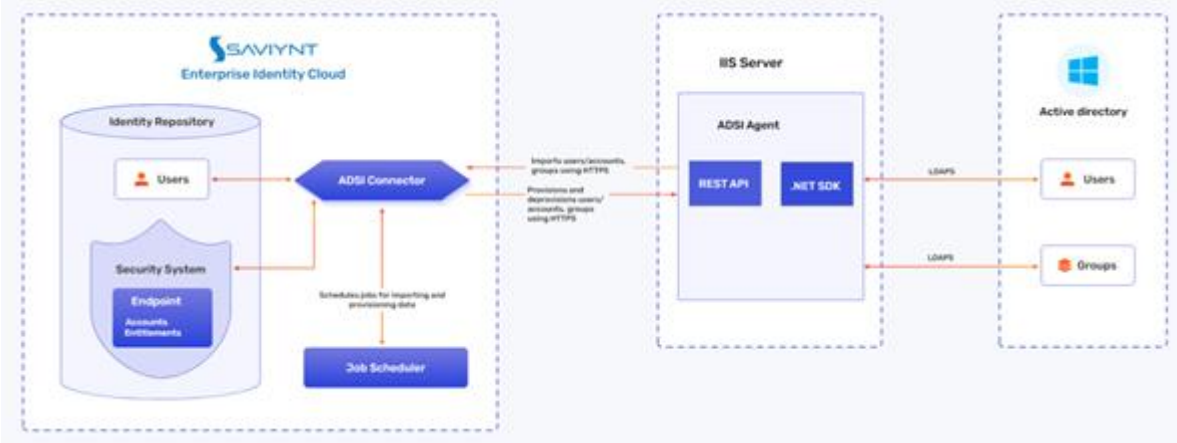


Figure 1: ADSI application is cross-domain or multi-domain architecture

ADSI can be developed or interacted with in many languages, like Java, Visual Basic Scripting (VBScript), Microsoft Visual Basic, and Powershell [4]. Specify the tools and techniques for collecting data, focusing on integrating ADSI and Saviynt. Languages may include PowerShell scripts, Saviynt's APIs, Internet Information Services (IIS Server), and Active Directory management tools. Outline the procedures for conducting integration testing. Include configuring integration settings between ADSI and Saviynt, provisioning/De-provisioning test users, and monitoring the integration's functionality.We must establish trust between each forest in the target application to achieve this. If they have no trust relationship, we can provide access to cross-domain groups/entitlements.

Import Access and Accounts from Target Application:
We need to configure the connection in the IAM application. Import accounts and access from the target AD application using the ADSI agent.

Table 1: Connection parameters

Parameter	Description	Example Configurations
Connection Name	Specify a unique name of the connection.	ADSI
Connection Description	Specify the description for the connection.	Active Directory Connection
Connection Type	Specify the connection type as ADSI.	ADSI
Email Template	Specify this parameter to select an email template for sending notifications.	Create a Email template
URL	Specify the Primary/root domain URL	LDAP://sam.com:636
USERNAME	User Principal name (UPN) of the user with privileged service account	svc_saviynt@sam.com
PASSWORD	Password for the user with privileged service account	**********
Connection_URL	ADSI agent is deployed in the following format: <IIS server URL>+/api/v1/discovery	http://chsaviynt.sam.com:8080/api/v1/discovery http://00.000.000.0:8080/api/v1/discovery
FORESTLIST	Specify the list of all forests in a comma-separated value	sam.com, sam.uk.com, sam.in.com

FOREST_DETAILS	Details of domains and domain controllers for a forest after the connection is successfully established.	{
		"forestDetails": {
		"sam.com": [
		{
		"sam.com": [
		"DC01.sam.com"
		]},
		{
		"sam.uk.com": [
		"DC02.sam.uk.com"
		]},{
		"sam.in.com": [
		"DC03.sam.in.com"
		]}]}}

We also need to specify the search criteria where the agent has to communicate and fetch the data from the target application using the *searchfilter*. In Figure 2, We have three forests, "sam.com, sam.uk.com, sam.in.com" Below is the syntax to fetch all the data from the forests.

```
DC=sam,DC=com###DC=sam,DC=uk,DC=com###DC=sam,DC=in,DC=com
```

Figure 2: Active Directory Search Filter

When importing the account, we must specify the mapping and populate the necessary information from the target application. Figure 3 shows the mappings between the target and the IAM application. The left side is the IAM attributes, and the Right is the Target AD attributes.

```
[CUSTOMPROPERTY1::samaccountname#String,
CUSTOMPROPERTY2::userprincipalname#String,
LASTLOGONDATE::lastLogon#millisec,
DISPLAYNAME::cn#String,
NAME::distinguishedname#String,
CUSTOMPROPERTY21::manager#String,
CUSTOMPROPERTY23::mobile#String,
ACCOUNTCLASS::objectclass#String,
ACCOUNTID::distinguishedname#String,
CUSTOMPROPERTY24::useraccountcontrol#String,
CUSTOMPROPERTY26::objectguid#String,
CUSTOMPROPERTY28::forest#String,
CUSTOMPROPERTY29::domain#string,
CUSTOMPROPERTY30::objectclass#String,
```

The status value mapping is based on the status of the target application [6][5] [site my AD paper].

```
{
"statusAndThresholdConfig":
    {
        "activeStatus":["512","544","66048"],
    }
}
```

Figure 4: Status of account in the target application

The groups/entitlements parameters are stored in AD's "*memberOf*" attribute. This attribute only accepts lowercase characters, which is typical for all the domains [7]. The group search is similar to the *searchfilter* in Figure 2.

```
{
    "importGroupHierarchy": "true",
    "entitlementTypeName": "memberOf",
    "performGroupAccountLinking": "true",
    "groupObjectClass": "(objectclass=group)",
    "mapping": "memberHash:member_char,customProperty1:samaccounttype_char,customProperty2:instancetype_char,customProperty3:usncreated_char,
    customProperty4:grouptype_char,customProperty5:dscorepropagationdata_char,customProperty12:dn_char,customProperty13:cn_char,lastscandate:whencreated_date,
    customProperty15:managedby_char,entitlement_glossary:description_char,description:description_char,displayname:name_char,customProperty9:name_char,
    customProperty10:objectcategory_char,customProperty11:samaccounttype_char,entitlement_value:distinguishedname_char,entitlementid:distinguishedname_char,
    customProperty14:objectclass_char,updatedate:whenchanged_date,customProperty17:distinguishedname_char,customProperty18:objectguid_char,
    RECONCILATION_FIELD:customProperty18",
    "entitlementOwnerAttribute": "managedby",
    "tableFieldAttribute": "accountID"
}
```

Figure 5: Groups/Entitlements Owner Mapping

Import Users from Target Application:
Importing users from the target application, we must focus on "*searchfilter, objectfilter, mapping, status.*" The search criteria can be used to specify all the forests that we need to import from. Figure 2 is the sample example of all the distinguished names (DN) it is searching form. Object filter is used to specify the data it needs to return from the "*(&(objectCategory=person)(objectClass=user))"* target.
Figure 6: User import Mapping

We are using the attribute as per Figure 6, the syntax to map the fields via import into the IAM tool. The mapping will help to populate the values in Saviynt from the target AD application.

We can also get the user status populated from the target AD application along with the import. If you look at Figures 4 & 7, we are using these mapping to update the status of the users [8].

```
{
"STATUS_ACTIVE": ["512","544","66048"],
"STATUS_INACTIVE": ["546","514"]
}
```

Figure 7: Status Value Mapping

Provision and De-provision Accounts and access from Target application:
The connection to the target AD application using ADSI will also help us automate the provisioning and de-provisioning to multiple domains using a single connectivity. If you look at Figures 8 & 9, base DN filters the provisioning and de-provisioning to the respective OUs [9]. This will also help us create users based on the organizations' criteria.

```
{
"objects": [
{
"objectClasses": [
    "user",
    "Person"
    ],
"attributes": {
    "sn": "${user.lastname}",
    "sAMAccountName": "${task.accountName}",
    "cn": "${cn.replace(',', '\\,')}",
    "userAccountControl": 512,
    "co": "${user.country}",
    "department": "${user.departmentname}",
    "displayName": "${user.displayname}",
    "employeeID": "${user.employeeid}",
    "employeeType": "${user.employeeType}",
    "givenName": "${user.firstname}",
    "l": "${user.city}",
    "mail": "${user.email}"
    },
"baseDn": "${if(user?.country.equals('US') && user?.customproperty29.equals('USA'))'OU=Users,DC=sam,DC=com' else if (user?.street.equals('UK') && user?.
customproperty29.equals('United Kingdom')) 'OU=Users,DC=uk,DC=sam,DC=com' else if (user?.street.equals('IN') && user?.customproperty29.equals('India')) 'OU=Users,DC=in,
DC=sam,DC=com' else 'CN=Test Users,DC=sam,DC=com'}",
"password": "${password}"
}
]
}
```

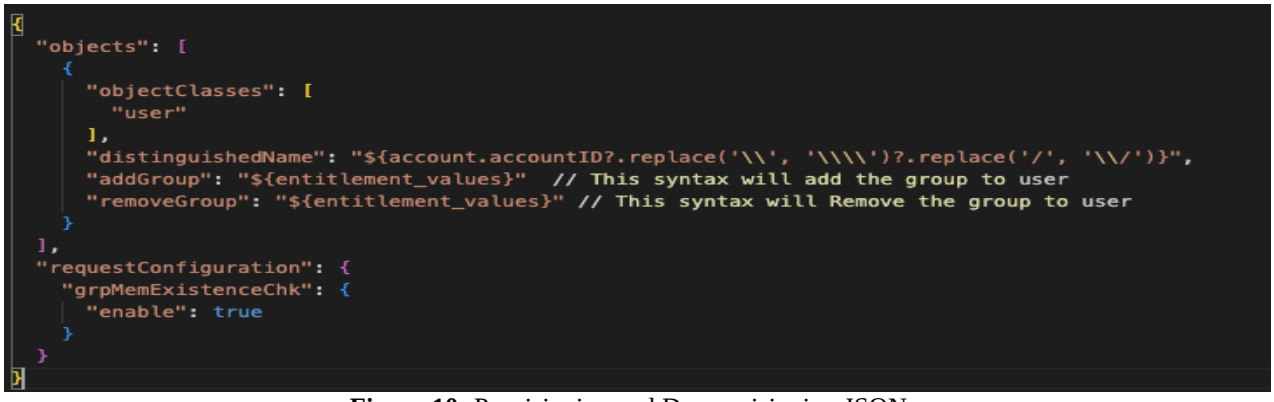

Figure 8: Create user JSON

Updating users will also play a critical role in the organization, as people change positions and regions; Figure 9 is used to update the user OU based on the criteria.

```
{
    "objects": [
    {
    "objectClasses": ["user"],
    "distinguishedName": "${account.accountID?.replace('\\', '\\\\')?.replace('/', '\\/')}",
    "moveObjectToOU": "${if(user?.country.equals('US') && user?.customproperty29.equals('USA'))'OU=Users,DC=sam,DC=com' else if (user?.street.equals('UK') && user?.
    customproperty29.equals('United Kingdom')) 'OU=Users,DC=uk,DC=sam,DC=com' else if (user?.street.equals('IN') && user?.customproperty29.equals('India')) 'OU=Users,
    DC=in,DC=sam,DC=com' else 'CN=Test Users,DC=sam,DC=com'}",
    "attributes": {
        "displayName": "${user.displayname}",
        "streetAddress": "${user.street}",
        "additionalAttributes": {
            "departmentName": "PM",
            "companyName": "Saviynt"
            }
        }
    }
    ]
}
```


Figure 9: Update user JSON

Automating access provisioning and de-provisioning plays a crucial role in an organization. Figure 10 will show the add access and remove access JSON for assigning and revoking user access [9]. The *object Classes* restrict the provisioning or de-provisioning to a specific set of objects defined. This will first find the user from the global catalog and perform the relevant action based on the task. This automation will help the organization save many human errors, efficiency, and operational costs.



```
{
  "objects": [
    {
      "objectClasses": [
        "user"
      ],
      "distinguishedName": "${account.accountID?.replace('\\', '\\\\')?.replace('/', '\\/')}",
      "addGroup": "${entitlement_values}"  // This syntax will add the group to user
      "removeGroup": "${entitlement_values}" // This syntax will Remove the group to user
    }
  ],
  "requestConfiguration": {
    "grpMemExistenceChk": {
      "enable": true
    }
  }
}
```


Figure 10: Provisioning and De-provisioning JSON

Along with access management, deactivating and activating the account on re-hire or termination plays a critical role in the organization. Figure 11 will give us a clear idea of how to take action on account management. *User Account Control* deactivates or activates the account in the target applications per the Microsoft Active Directory syntax. We can also delete all user access using the *delete AllGroups* on termination; we can also restrict the removal of specific groups using the *group Exclusion ListOnRemoval.* We will

set a default password on termination so users can't access the application anymore.

```
{
  "objects": [
    {
      "objectClasses": [
        "user"
      ],
      "distinguishedName": "${account.accountID?.replace('\\', '\\\\')?.replace('/', '\\/')}",
      "moveObjectToOU": "CN=Users,DC=sam,DC=com",
      "password": "${password}", //This syntax is used while disableing the account to set a random passsword
      "deleteAllGroups": true,
      "groupExclusionListOnRemoval": [
        "CN=UniversalDistGroupUS,CN=Users,DC=sam,DC=com",
        "CN=UniversalDistGroupUK,CN=Users,DC=uk,DC=sam,DC=com",
        "CN=UniversalDistGroupIN,CN=Users,DC=in,DC=sam,DC=com"
      ],
      "attributes": {
        "userAccountControl": 512 //This syntax is used to enable the account
        "userAccountControl": 514 //This syntax is used to disable the account
      }
    }
  ]
}
```

Figure 11: Disable and Enable JSON

4. Result

Implementing this robust multi-domain application has helped the organization to establish real-time connectivity to AD. This also helped in the synchronization process to sync to all the domains and reduced the efforts to manage multiple connections to individual forests in AD.

This Integration will help to manage almost 90 percent of the AD services without any operational work and reduce human errors. This connectivity has helped us to manage all the users in one integration, which also helped the Audit and compliance teams review all the data from one connection instead of different connections. ADSI offers a granular approach to access control, allowing administrators to assign and manage permissions at a detailed level, tailoring access to specific resources and services. Moreover, ADSI reinforces synchronization mechanisms to keep imported data up-to-date, whether incremental data imports capture changes promptly or scheduled imports to ensure data remains accurate.

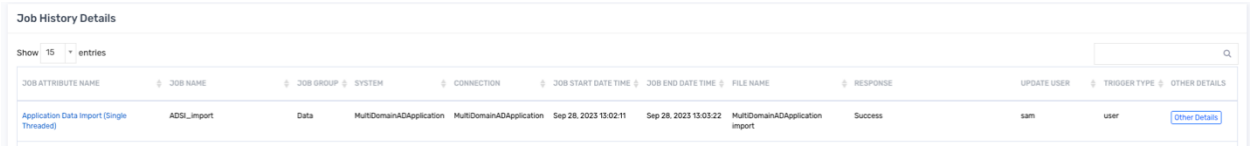


Figure 12: Access Import Results from Target Application.

It facilitates the seamless import of user accounts, access entitlements, and account information from Active Directory domains and forests.The efficiency of these data import and synchronization processes simplifies identity management and minimizes administrative overhead by automating these critical tasks.

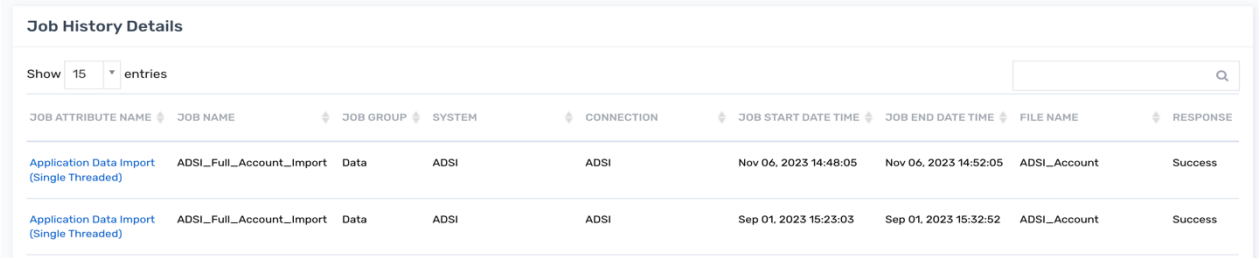
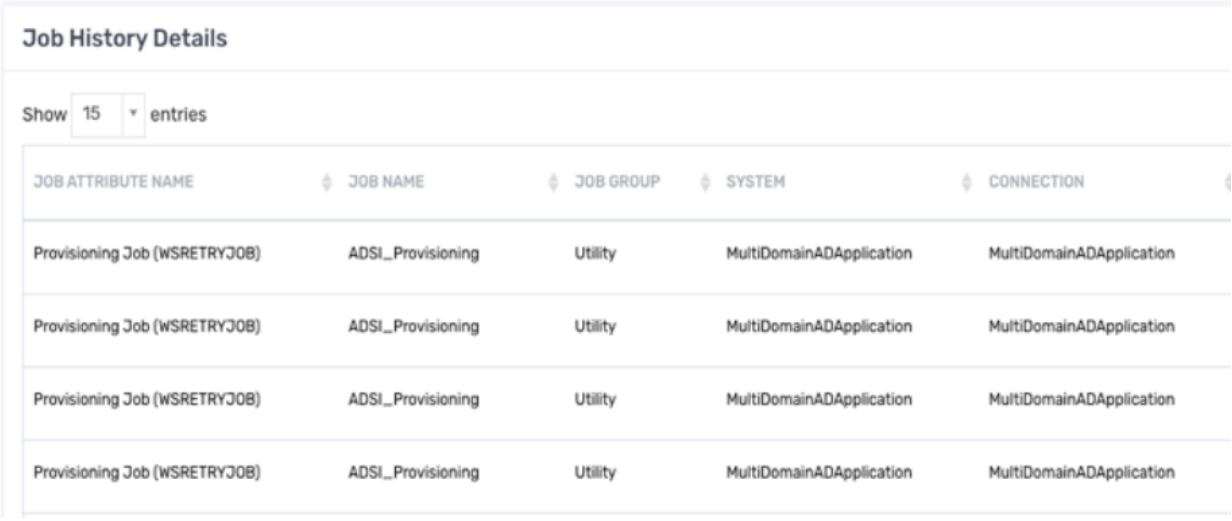


Figure 13: Accounts Import Results from Target Application

Figures 12 & 13 show the import results of accounts, users, and access from the target application. Figure 14 & 15 streamlines granting and revoking access, ensuring that users are provided with the appropriate privileges when needed and that access is promptly revoked when no longer required.



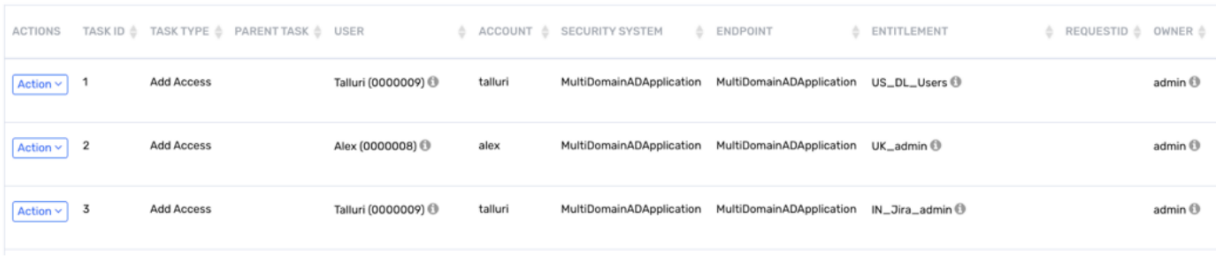
Job History Details

Show 15 entries

JOB ATTRIBUTE NAME	JOB NAME	JOB GROUP	SYSTEM	CONNECTION
Provisioning Job (WSRETRYJOB)	ADSI_Provisioning	Utility	MultiDomainADApplication	MultiDomainADApplication
Provisioning Job (WSRETRYJOB)	ADSI_Provisioning	Utility	MultiDomainADApplication	MultiDomainADApplication
Provisioning Job (WSRETRYJOB)	ADSI_Provisioning	Utility	MultiDomainADApplication	MultiDomainADApplication
Provisioning Job (WSRETRYJOB)	ADSI_Provisioning	Utility	MultiDomainADApplication	MultiDomainADApplication

Figure 14: Provisioning/De-provisioning Job Results

This fine-grained control results in a more secure and efficient access provisioning process. ADSI integration enhances efficiency, synchronization, and control over access provisioning and de-provisioning while ensuring a seamless and accurate import process for users, access, and accounts. All contribute to a more secure and streamlined approach and are crucial in maintaining data integrity.



ACTIONS	TASK ID	TASK TYPE	PARENT TASK	USER	ACCOUNT	SECURITY SYSTEM	ENDPOINT	ENTITLEMENT	REQUESTID	OWNER
Action	1	Add Access		Talluri (0000009)	talluri	MultiDomainADApplication	MultiDomainADApplication	US_DL_Users		admin
Action	2	Add Access		Alex (0000008)	alex	MultiDomainADApplication	MultiDomainADApplication	UK_admin		admin
Action	3	Add Access		Talluri (0000009)	talluri	MultiDomainADApplication	MultiDomainADApplication	IN_Jira_admin		admin

Figure 15: Provisioning Access Tasks

5. Challenges and Lessons Learned

Indeed, while ADSI integration provides multiple benefits for access management, there can be challenges and issues that organizations may encounter. Some typical troubles and topics associated with ADSI integration might arise in the implementation.

Trust Relationship: The ADSI can manage multiple DCs and Forests simultaneously. If the DC and forests don't trust each other, cross-domain provisioning or De-provisioning is impossible using ADSI.

Security: ADSI is a powerful tool for managing access and identity; it's vital to ensure its safety. Misconfigurations or vulnerabilities in the integration can expose sensitive data and compromise the security of the Active Directory [11].

Version Compatibility: Compatibility issues can arise when working with outdated software. Keeping the ADSI integration up to date with the latest AD versions and any updates from the IAM solution provider is critical.

Integration Issues: ADSI requires Directory replication permissions and Enterprise Admin group permission to manage access, provisioning, and de-provisioning. Configuring the necessary components, including connector and agent, and establishing trust relationships between domains and forests requires careful planning and technical expertise. Minor misconfiguration can lead to substantial operational issues. ADSI cannot create or manage local system accounts.

6. Future Work

Managing the service and local accounts using ADSI, managing the standard version capability to ensure successful connectivity, maintaining trust relationships with new and existing domains, and updating the configuration per the new requirements. Ensure the password synchronization is also established in the IAM tool.

7. Conclusion

The ADSI integration is a versatile solution for managing and integrating with Active Directory forests and domains. It offers a range of capabilities, including importing accounts, access, and users, provisioning and de-provisioning, and managing Active Directory groups using the ADSI agent. Whether you need to administer a single domain, multiple domains, or multiple forests with trust relationships, the ADSI connector provides a flexible and efficient way to automate tasks and govern access within your enterprise.

To set up an ADSI integration, carefully consider the organization-specific use case and the required features. It's

essential to ensure that prerequisites are met, such as creating a service account with appropriate privileges, establishing domain and forest trusts, and deploying the ADSI agent. Configuring a high-availability connection and identifying a service account with domain administrator permissions are crucial for successful integration.

The integration can meet your organization's needs, whether you need to manage user lifecycles, gain visibility into membership details, or automate everyday administrative tasks within Active Directory environments. The ADSI integration's flexibility and robust features make it a valuable tool for identity and access management within complex enterprise setups compared to a regular AD setup.

Overall, the ADSI integration provides a powerful mechanism to enhance your Active Directory infrastructure's security, efficiency, and manageability, helping you achieve better governance and control over user identities and access privileges over the entire organization.

References

[1] Alexander Puchta, Fabian Böhm, Günther Pernul. Contributing to Current Challenges in Identity and Access Management with Visual Analytics. 33th IFIP Annual Conference on Data and Applications Security and Privacy (DBSec), Jul 2019, Charleston, SC, United States. pp.221-239, ff10.1007/978-3- 030-22479-0_12ff. ffhal-02384584f

[2] Gibbons, P. (2002). Using Active Directory Service Interface. In: .NET Development for Java Programmers. Apress, Berkeley, CA. https://doi.org/10.1007/978-1-4302-1106-8_13

[3] "Windows 2000 Server - ADSI Open Interfaces for Managing and Using," www.scrigroup.com. Available: https://www.scrigroup.com/calculatoare/tutorials/414/Windows-Server-ADSI-Open-Inter74795.php.

[4] alvinashcraft, "Active Directory Service Interfaces - Win32 apps," learn.microsoft.com, Aug. 21, 2020. Available: https://learn.microsoft.com/en-us/windows/win32/adsi/active-directory-service-interfaces-adsi.

[5] Saviynt, "Saviynt Documentation," docs.saviyntcloud.com. Available: https://docs.saviyntcloud.com/bundle/AD-v2022x/page/Content/Configuring-the-Integration-for-Importing-Users.htm

[6] Deland-Han, "UserAccountControl property flags - Windows Server," learn.microsoft.com. Available: https://learn.microsoft.com/en-us/troubleshoot/windows-server/identity/useraccountcontrol-manipulate-account-properties

[7] Saviynt, "Saviynt Documentation," docs.saviyntcloud.com. Available: https://docs.saviyntcloud.com/bundle/ADSI-v2022x/page/Content/Configuring-the-Integration-for-Importing-Accounts-and-Access.htm.

[8] Saviynt, "Saviynt Documentation," docs.saviyntcloud.com. Available: https://docs.saviyntcloud.com/bundle/ADSI-v2022x/page/Content/Configuring-the-Integration-for-Importing-Users.htm.

[9] Saviynt, "Saviynt Documentation," docs.saviyntcloud.com. Available: https://docs.saviyntcloud.com/bundle/ADSI-v2022x/page/Content/Configuring-the-Integration-for-Provisioning-and-Deprovisioning.htm.

[10] Lissoir, "Understanding Microsoft Windows Script Host and Active Directory Service Interfaces in Windows 2000," 2000. Available: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=de3173bfdf869ed77f6e69b1db549c3f6eb2a084.