Single Sign-On for Cloud Services using Enterprise Active Directory

Written by

Prof. Suman Banerjee, CTO, StratusWorX & Professor of Computer Sciences, Univ. of Wisconsin-Madison

Locally Administered Active Directory

Assuming the customer is managing their own locally administered AD server, the organization may be willing to grant StratusWorX LDAP access to their server. In that case, it is possible to read the user and group directory using an LDAP client implementation. As it is a standardized protocol, LDAP libraries exist in many languages (Go, Java, Python). The customer will need to provide information about the server, its security settings, and how to pass through their firewall. It is recommended that the customer provide a set of credentials for StratusWorX with only the needed read access to the user and groups directories. Authenticating the customer's users can be performed by issuing a BIND command to the server with the user's supplied ID and password. Issuing a BIND command is recommended over querying the server for the user's password and comparing because of vagaries in implementations of password hashing.

Information needed to establish a connection to the customer AD server:

- AD server address (IP or hostname)
- AD server port
- LDAP version (2 or 3) and security method (SSL or TLS)
- Credentials for a designated user for StratusWorX with read-only access to user and groups directories.

Azure AD Connect

Customers may understandably prefer to keep their AD server private and not grant LDAP access to an outside entity. In that case, it is recommended to use Azure AD as a trusted intermediary between the customer and StratusWorX. If they are not already, customers can run the Azure AD Connect sync service to synchronize their locally administered AD server with the Azure AD service[1]. Once users and groups have been synchronized with Azure AD, they are available for SSO functionality across Microsoft and third-party offerings including StratusWorX. Many organizations may already be synchronizing with Azure AD or running off Azure AD entirely without an on-premises AD server.

Information needed to authenticate users through Azure AD:

- Customer's Azure tenant ID or domain, e.g. "6aeb4f94-7c2c-47d1-b3b2-f196cd404db3" or "example.onmicrosoft.com"

[1] An introduction to Azure AD and synchronizing from Windows Server AD: https://www.red-gate.com/simple-talk/cloud/security-and-compliance/azure-active-directory-part-1-an-introduction/

- Naming convention used for the customer's user IDs, e.g. "user" or "user@example.com".
- (Optional) Name or GUID of a user group(s) that the customer manages and has designated as users who should be permitted access to services on StratusWorX. A reasonable default for small organizations might be that all authenticated users from the customer organization gain access to the selected product offerings on StratusWorX, but larger organizations might prefer to permit different services based on user groups.

It is not strictly necessary for StratusWorX to enumerate all the users and groups from the customer organization if user authentication is delegated to Azure AD. After a user passes the authentication check and optionally a group membership check against the customer's own directory, a user may be permitted to access the StratusWorX service offerings as agreed between the customer and StratusWorX[2]. Treating Azure AD as the authoritative source of users and groups allows for easy onboarding of new users using the tools and processes the customer

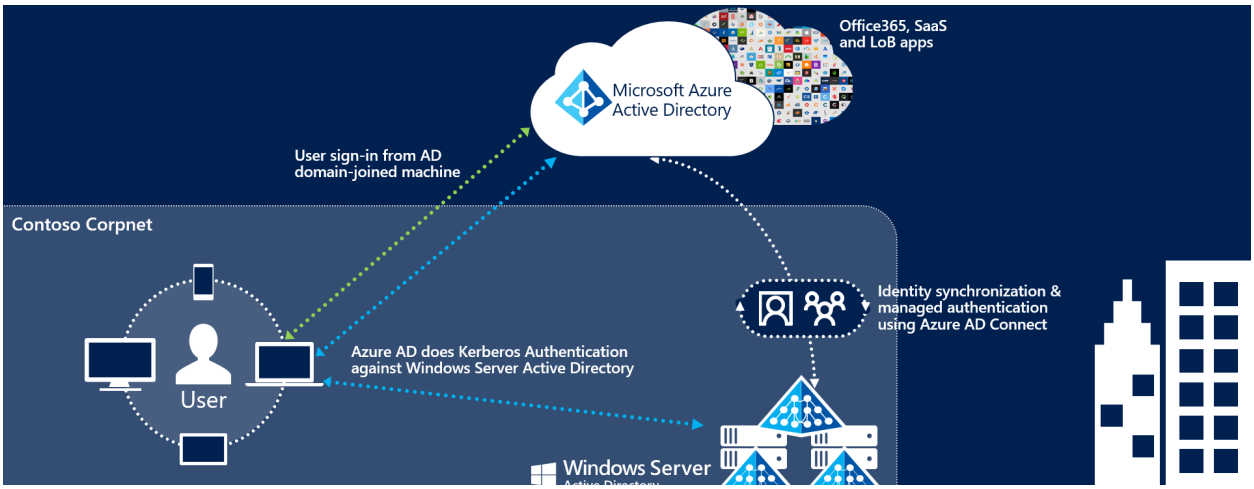

Figure 1: Synchronization with Azure AD Connect. Image source: https://docs.microsoft.com/en-us/azure/active-directory/hybrid/how-to-connect-sso

already has in place. Notably, it also avoids the problem of sticky user privileges when the customer revokes group membership or deletes the user account entirely. If a user is no longer able to authenticate against the customer's AD, the user inherently will not be permitted access to StratusWorX.

Azure AD Passthrough Authentication

In some cases, the customer organization may have a locally administered AD server and may agree on the benefits of using Azure AD to facilitate SSO for its users to cloud services but also be unwilling to synchronize their AD data to the cloud. Azure AD passthrough authentication offers

[2] This page presents some example Python code for authenticating a user against another tenant's AD records. Although the code is intended for a different use case it is a working example of the idea in practice. https://docs.microsoft.com/en-us/samples/azure-samples/data-lake-analytics-python-auth-options/authenticating-your-python-application-against-azure-active-directory/

a solution for customers in that situation[3]. Instead of synchronizing their AD data to the cloud, the customer runs a lightweight authentication agent on their on-premises AD server that responds to authentication challenges initiated through Azure AD. Whether the customer uses Azure AD Connect or passthrough authentication, the user authentication process through Azure AD remains the same from the perspective of StratusWorX.

One disadvantage of passthrough authentication is that it may be difficult to diagnose communication issues between Azure AD and the on-premises AD server, for example, if the on-premises server is offline. Since passthrough authentication relies on a private communications channel between the customer and Azure, it is difficult for StratusWorX to remedy issues even though those issues may impact the ability of StratusWorX to offer services to end users. As a practical example, users may become frustrated and blame StratusWorX if they are unable to log in. If the customer chooses passthrough authentication, it is recommended that StratusWorX take some proactive steps in order to be ready to respond to issues rapidly and create trouble tickets automatically in some cases.

1. Monitor user authentication failures, especially those that appear to fail after timeout or due to connection failure rather than invalid credentials. Error responses from Azure AD may provide some insight here.
2. Ask the customer to create a canary user with minimal permissions and periodically, such as every 15 minutes, perform a login attempt. Failure to log in with known working credentials would be a strong indicator of a problem somewhere along the authentication path.

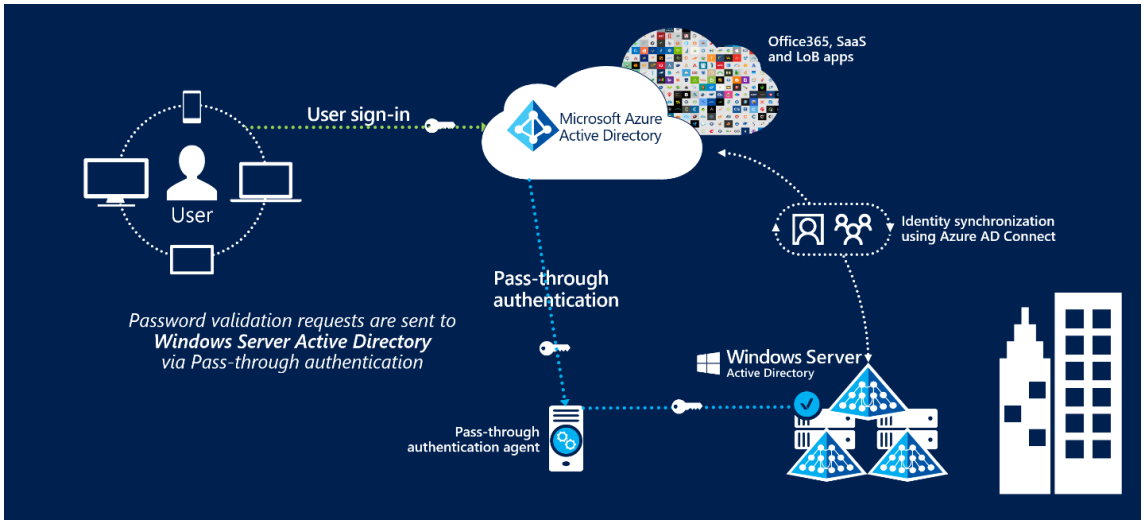

Figure 2: Pass-through authentication via an authentication agent vs. cloud authentication using Azure AD Connect. Image source: https://docs.microsoft.com/en-us/azure/active-directory/hybrid/how-to-connect-pta

[3] Information on the choice between AD Connect and passthrough authentication from the perspective of the customer: https://docs.microsoft.com/en-us/azure/active-directory/hybrid/choose-ad-authn.

Tracking Changes to Active Directory

While it is not necessary for StratusWorX to maintain a separate copy of the customer's user and group lists in order to authenticate users, it may still be useful for certain business purposes such as billing or to build useful interfaces that help the customer select from users and groups. For this purpose, as well as gaining information about a user's group membership[4] and roles, it may be possible to use the Microsoft Graph API. It is also possible to use delta queries to detect changes within a time range instead of querying for the entire list[5].

[4] Graph API documentation for getMemberGroups: https://docs.microsoft.com/en-us/graph/api/user-getmembergroups?view=graph-rest-1.0&tabs=http. After a user has authenticated, this can be used to list all the groups to which the user belongs.

[5] Documentation on delta queries: https://docs.microsoft.com/en-us/graph/delta-query-overview