MSOA2Authenticator
Users Guide

MSOA2Authenticator is a subclass of net.ae5pl.emailgate.OA2Authenticator. It is targeted to MS 365 Exchange Online but should be able to be used with any service provider that uses OAuth2 to access IMAP, POP3, and SMTP. MSOA2Authenticator is a Java service provider subclass of OA2Authenticator which is a subclass of javax.mail.Authenticator or jakarata.mail.Authenticator (both are the exact same signature so it can be used interchangeably). OA2Authenticator assumes the Client Flow of OAuth2 is being used (application accessing independent of user).

The support library from Microsoft, msal4j, requires slf4j-api, slf4j-jdk14, and azure.json jars.

MSOA2Authenticator Users Guide

MSOA2Authenticator

```
package net.ae5pl.msauth;

import java.util.*;
import java.util.logging.*;
import net.ae5pl.emailgate.OA2Authenticator;
import com.microsoft.aad.msal4j.*;

/**
 * Microsoft OAuth2 Authenticator for daemon access
 *
 * For MS 365,
 * App must be registered single tenant with (from My Organizations APIs)<br>
 * Office 365 Exchange Online permissions
 * <code>
 * Imap.AccessAsApp
 * Mail.ReadWrite
 * Mail.Send
 * SMTP.SendAsApp
 * </code>
 * <code>
 * tenantId property required for MS 365
 * </code>
 * Add as Service Principal in Exchange (PowerShell).
 * Add as new permission to desired mailbox (-user is service principal id).
 *
 * @author Pete Loveall AE5PL
 */
public class MSOA2Authenticator extends net.ae5pl.emailgate.OA2Authenticator
{
    private final static Logger classlogger =
Logger.getLogger(MSOA2Authenticator.class.getName());

    private final ConfidentialClientApplication app;
    private final ClientCredentialParameters clientCredentialParam;

    /**
     * Dummy instance for ServiceLoader
     */
    public MSOA2Authenticator()
    {
        app = null;
        clientCredentialParam = null;
    }

    private MSOA2Authenticator(String clientID, String clientSecret, String authEndpoint,
Set<String> scope) throws Exception
    {
        app = ConfidentialClientApplication.builder(clientID,
ClientCredentialFactory.createFromSecret(clientSecret))
                .authority(authEndpoint).executorService(execsvc).build();
        clientCredentialParam = ClientCredentialParameters.builder(scope).build();
    }

    @Override
    protected OA2Authenticator initialize(Properties providerspec, String clientID, String
clientSecret, String authEndpoint, String scope)
    {
        String tauthendpoint;
        Set<String> tscope;
        String ttenantid = providerspec.getProperty("tenantId", "").trim();
        if (ttenantid.isEmpty())
        {
            // Non-Microsoft 365 Exchange Online
            if (authEndpoint.isEmpty() || scope.isEmpty())
                return null;
            tauthendpoint = authEndpoint;
            tscope = new HashSet<>(Arrays.asList(scope.split(" ")));
        }
        else
        {
            tauthendpoint = "https://login.microsoftonline.com/" + ttenantid +
"/oauth2/v2.0/token";
            tscope = Collections.singleton("https://outlook.office365.com/.default");
        }
        try
        {
            return new MSOA2Authenticator(clientID, clientSecret, tauthendpoint, tscope);
        }
        catch (Exception e)
        {
            classlogger.log(Level.WARNING, "Exception creating ConfidentinalClientApplication",
e);
        }
        return null;
    }

    @Override
    protected String acquireTokenConfidential() throws Exception
    {
        return app.acquireToken(clientCredentialParam).join().accessToken();
    }
}
```

MSOA2Authenticator
Users Guide

Exchange Online Setup

To use OAUTH2 with Microsoft 365 Exchange Online (business/enterprise, not personal), the following must be done:

If you wish to send using SMTP, you ***must*** disable *Security Defaults* in the Tenant. This is to ensure "legacy protocols" are not disabled. This does not enable Basic authentication but it does allow the client flow used in OAuth2 to work with SMTP. IMAP and POP3 do not require this.

Register the app in the Entra ID admin center. Add a secret to the app (remember to copy and save the secret as it will not be visible when you leave the page). Now add API permissions (left side navigation) by clicking on *Add Permission* link at the top of the permissions. Click on the *APIs my organization's uses* tab and search for office. Select *Office 365 Exchange Online* and then select *Application permissions*. Then select the following permissions and save:

```
IMAP.AccessAsApp (IMAP access)
POP.AccessAsApp (POP3 access)
Mail.ReadWrite
Mail.Send
SMTP.SendAsAp
```

You will need 4 GUIDs from Entra ID at this point:

```
tenantId can be found on the Tenant Overview page.
clientId is the Application ID in the app page under Enterprise Apps.
clientSecret is the secret you created above.
Object ID is the Object ID in the app page under Enterprise Apps (to be used in setting up
Exchange).
```

Now start a PowerShell Exchange Online session.

Create a Service Principle using the Object ID previously copied from the Entra ID *Enterprise Apps* page using the *New-ServicePrincipal* cmdlet. Be sure to give it a Display Name since that will help you identify it later.

Now, register the app with each mailbox that will be either accessed and/or used to send SMTP (you cannot send from a Shared Mailbox). This is done in the PowerShell session started above and using the *Add-MailboxPermission* on the individual mailboxes and setting -User to the Security Principal ID that was created in the *New-ServicePrincipal* (if you forget, try *Get-ServicePrincipal | fl*). Finally, for the mailbox you wish to send with, run the *Set-CASMailbox* cmdlet with the parameter -*SmtpClientAuthenticationDisabled $false* (the organization should be set to $true using the *Set-TransportConfig* cmdlet).

If you are sending for a Shared Mailbox, set *SendAs* permission for the SMTP user mailbox you are using as your sending mailbox. You can make sure you have set up the application for a mailbox by going to the Delegation page for the mailbox and you should see the display name of the app in the *Read and Full Access* page.

Now, add clientId, clientSecret, and tenantId to your properties file.

NOTE on IMAP: javax.mail and angus.mail IMAP implementations do not currently give you the ability to reset IDLE after a period of silence on the socket. This is being addressed in angus.mail but, for the moment, you must use polling instead. I have opened a case with Angus to add the IDLE reset ability based on socket receive timeout per the IMAP RFC. When addressed, set mail.imaps.timeout=600000 so a socket timeout will occur every 10 minutes causing the “keep-alive” to occur. Until then, the Inbox must be polled so I recommend using POP3 instead.

MSOA2Authenticator
Users Guide

Exchange Online Properties

Properties (IMAP):

```
mail.store.protocol=imaps
mail.imaps.timeout=600000
mail.imaps.host=outlook.office365.com
mail.imaps.port=993
mail.imaps.user=mailbox email address
mail.imaps.ssl.enable=true
mail.imaps.auth=true
mail.imaps.auth.mechanisms=XOAUTH2
mail.imaps.partialfetch=false
mail.imaps.ignorebodystructuresize=true
```

Properties (POP3):

```
mail.store.protocol=pop3s
mail.pop3s.host=outlook.office365.com
mail.pop3s.port=995
mail.pop3s.user=mailbox email address
mail.pop3s.ssl.enable=true
mail.pop3s.auth=true
mail.pop3s.auth.mechanisms=XOAUTH2
mail.pop3s.auth.xoauth2.two.line.authentication.format=true
```

Properties (SMTP):

```
mail.transport.protocol=smtp
mail.smtp.port=587
mail.smtp.host=smtp.office365.com
mail.smtp.user=SMTP-authorized email address
mail.smtp.starttls.enable=true
mail.smtp.auth=true
mail.smtp.auth.mechanisms=XOAUTH2
```