Introduction

iLAPS implementation based off of PowerShell – Intune Local Administrator Password Solution (iLAPS)

Objectives

1. Deploy Automatically
2. Create Admin UI with Ability to Limit access via Azure AD
3. Automatically Reset passwords which have been viewed by an admin to verify passwords are always random

Getting Started Overview

1. Installation process
    1. Create Customer Specific Secrets File (Steps below)
    2. Build Software dependencies (Steps below)
    3. Deploy Software
2. Software dependencies
    1. GIT Commandline
    2. Dotnet Core 3.1 SDK
    3. Powershell 3+

Local Configuration

1. Clone solution into `c:\dev`
2. Navigate to `c:\dev\iLAPs`
3. Copy `settings.template.json` with name `settings.production.local.json`
    1. You can add as many `settings.environment-name.local.json` as you want if you want to be able to build for many environments. To use the new environments modify step `13.2` to look like `.\Build.ps1 -BuildEnvironment 'environment-name'`
    2. If `BuildEnvironment` is not specified it assumes `Production`

Portal Configuration

1. Create Resouce Group Named `iLaps-RG`

2. Create General Purpose V2 Storage Account (Example name `ilapscustomername`)

    1. Save storage account name into `settings.production.local.json` field named

        ```
        "Storage-Account-Name": "PasteValueHere"
        ```

3. Create Shared Access Signature for `Update Admin Script`

    1. Allowed Services: Table

    2. Allowed Resource Types: Object
    3. Allowed Permissions: Add, Create
    4. Set Start and End Expiration dates
    5. Allowed Protocols: Https only
    6. Generate SAS and Connection String
    7. Save into `settings.production.local.json` field named

       ```
       "Table-Object-Add-Create-SAS-Token": "PasteValueHere"
       ```

4. Create Shared Access Signature for `Request Password Reset Script`
    1. Allowed Services: Table
    2. Allowed Resource Types: Object
    3. Allowed Permissions: Read, Update
    4. Set Start and End Expiration dates
    5. Allowed Protocols: Https only
    6. Generate SAS and Connection String
    7. Save into `settings.production.local.json` field shown below

       ```
       "Table-Object-Read-Update-SAS-Token": "PasteValueHere"
       ```

5. Create Shared Access Signature for `View Admin Passwords Legacy Powershell Script`
    1. Allowed Services: Table
    2. Allowed Resource Types: Object
    3. Allowed Permissions: Read, List
    4. Set Start and End Expiration dates
    5. Allowed Protocols: Https only
    6. Generate SAS and Connection String
    7. Save into `settings.production.local.json` field shown below

       ```
       "Table-Object-Read-List-SAS-Token": "PasteValueHere"
       ```

6. Create Shared Access Signature for `Installation Script`

    1. Allowed Services: File

    2. Allowed Resource Types: Object

    3. Allowed Permissions: Read

    4. Set Start and End Expiration dates

    5. Allowed Protocols: Https only

    6. Generate SAS and Connection String

    7. Save into `settings.production.local.json` field shown below

        ```
        "File-Object-Read-Installer-SAS-Token": "PasteValueHere"
        ```

7. Create Shared Access Signature for `Admin UI`

    1. Allowed Services: Table

    2. Allowed Resource Types: Object

    3. Allowed Permissions: Read, Write, List, Add, Create, Update

    4. Set Start and End Expiration dates

    5. Allowed Protocols: Https only

    6. Generate SAS and Connection String

    7. Save into `settings.production.local.json` field shown below

        ```
        "File-Object-Read-Installer-SAS-Token": "PasteValueHere"
        ```

8. Scroll down to `Tables` on side navigation bar:

    1. Create Table called `AdminPasswords`

    2. Create Table called `ResetPasswords`

    3. Create Table called `Logs`

    4. Save into `settings.production.local.json` field shown below

        ```
        "Admin-Table-Name": "AdminPasswords",
        "Reset-Table-Name": "ResetPasswords"
        "Log-Table-Name": "Logs"
        ```

9. Scroll down to `File Shares` on side navigation bar:

    1. Create File Share named `installation`

    2. Save into `settings.production.local.json` field shown below

        ```
          "Installer-Container-Name": "installation"
        ```

10. Navigate back to `iLaps-RG` Resource Group

    1. Click `Add` then search for WebApp
    2. Name WebApp `iLaps-customername` where customername is your customer's name
    3. Runtime Stack: `.Net Core 3.1 (LTS)`
    4. App Service Plan:
        1. Create New and name it
        2. Change Size to `S1`
    5. Click `Review + Create`
    6. Click `Create`
    7. Once created naviagte to `TLS/SSL` in new resource
        1. Turn on the `HTTPS Only` setting

11. Navigate to `Azure Active Directory`

    1. Click `App Registrations`

    2. Click `New Registration`

    3. Name Application `ILAPS`

    4. Click `Authentication` and click `Add a Platform`

        1. Click `Web`
        2. Specify the url as
            1. US Gov Cloud
                1. `https://ilaps-customername.azurewebsites.us/signin-oidc`
            2. US Commercial Cloud
                1. `https://ilaps-customername.azurewebsites.com/signin-oidc`
        3. Add another url for development purposes `https://localhost:5001/signin-oidc`
        4. Click `Configure`

    5. Navigate to Manifest and replace line 8 which contains `appRoles` with the following

        ```
            "appRoles": [
            {
              "allowedMemberTypes": [
                "User"
              ],
              "description": "SuperUser user can view passwords but they DO NOT reset automatically",
        ```

```
         "displayName": "SuperUser",
         "id": "b4a94c3f-bdd5-4c86-8749-d7f110195a56",
         "isEnabled": true,
         "lang": null,
         "origin": "Application",
         "value": "SuperUser"
      },
      {
         "allowedMemberTypes": [
            "User"
         ],
         "description": "Helpdesk user can view passwords but they reset automatically",
         "displayName": "User",
         "id": "574cd779-fece-4f33-aa31-d1374e8ea5ca",
         "isEnabled": true,
         "lang": null,
         "origin": "Application",
         "value": "User"
      }
   ],
```

6. Navigate to Manifest and find `publisherDomain` and remember the value for step 7

7. Navigate to Overview tab to save the fields below into `settings.production.local.json`

```
   "Admin-UI-Domain": "Type publisherDomain here"
   "Admin-UI-TenantId": "Tenant-GUID",
   "Admin-UI-ClientId": "App-Registration-Client-GUID",
```

8. Click `Overview` and in the top header click the link next to `Managed application in local directory`

    1. Click `Properties`
    2. Toggle `User Assignment required` to `Yes`
    3. Click `Users and Groups` and add users who should have access to this application.
        1. Add Role based on if the User is a `User` or a `Super User`. `Super Users` have the ability to view passwords without forcing a reset automatically and view access logs

12. Open `settings.production.local.json` and change set the following settings based on if you are targeting `US Gov Cloud` or `US Commercial Cloud` and your `Customers Name`

    1. US Gov Cloud

```
   "Company-Name": "My Gov Customer Name",
   "Storage-Account-Suffix":"core.usgovcloudapi.net",
   "Admin-UI-Instance":"https://login.microsoftonline.us/",
```

   2. US Commercial Cloud

      ```
      "Company-Name": "My Commercial Customer Name",
      "Storage-Account-Suffix":"core.windows.net",
      "Admin-UI-Instance":"https://login.microsoftonline.com/"
      ```

13. Ensure you have .NET Core 3.1 SDK installed

    1. Open Powershell window and navigate to `c:\dev\iLAPs`
    2. run `.\Build.ps1`

14. Open `c:\dev\iLAPs\Output\app-service-advanced-editor-script.json`

    1. Select all text and copy

15. Navigate to `iLaps-RG` in the portal

    1. Click `ilaps` App Service
    2. Click `Configuration`
       1. Click `Advanced Edit`
       2. Paste the copied value right before the last `]`
       3. Click `OK`
    3. Scroll down to `Advanced Tools` in the side navigation
       1. Click `Go`
       2. Hover over `Tools` then click `Zip Push Deploy`
       3. Open `c:\dev\iLAPs\Output\` in File Explorer
       4. Drag `AdminUI.zip` to Zip Deploy Interface (You will see it turn blue)

16. Navigate back to `iLaps-RG` Resource Group

    1. Click `ilapscustomername` storage account
    2. Click `File Shares`
    3. Click `installation`
    4. Click `Upload`
       1. Navigate to `c:\dev\iLAPs\Output`
       2. Click both `Reset-LocalAdministratorPassword.ps1` and `Check-Reset-LocalAdministratorPassword.ps1`
       3. Click `Upload`

17. Navigate to Use PowerShell scripts on Windows 10 devices in Intune

    1. Deploy the code found in `c:\dev\iLAPs\Output\Install-iLaps.ps1` using the guide linked above

18. ENJOY