IBM Resilient

Incident Response Platform Integrations

LDAP Multidomain Utilities V1.1.1

Release Date: November 2020

Resilient Functions simplify development of integrations by wrapping each activity into an individual workflow component. These components can be easily installed, then used and combined in Resilient workflows. The Resilient platform sends data to the function component that performs an activity then returns the results to the workflow. The results can be acted upon by scripts, rules, and workflow decision points to dynamically orchestrate the security incident response activities.

This guide describes the LDAP Multidomain Utility Functions.

Overview

The Lightweight Directory Access Protocol, or LDAP, is an open, vendor-neutral, industry standard application protocol for accessing and maintaining distributed directory information services over an IP network. It is used to connect to, search, and modify internet directories.

These LDAP Multidomain Utility integrations with the IBM Resilient platform allow multiple LDAP tasks to be initiated from the Resilient platform to an external LDAP server. The returned results can be used to make customized updates to the Resilient platform such as updating incidents, artifacts, data tables and so on.

These LDAP Multidomain Utility Functions integration package contains several useful workflow functions for common automation and integration activities in the Resilient platform.

These LDAP Multidomain Utility Functions integration package is strongly based on IBM's fn_ldap_utilities. Capability to interact with any number of domains is provided through app.config settings. In addition, set_password function is improved with an option to auto generate passwords instead of manually input.

Key features:

- Mainly based on IBM's fn_ldap_utilities.
- Improved to interact with any number of ldap/AD domains.
- Added an input field in each function to set a destination domain.
- New feature in ldap_md_set_password, allowing auto generated password, and length setting. In addition, for security reason it is possible to do not return the password.

This document describes each utility function, how to configure it in custom workflows, and any additional customization options.

Installation

Before installing, verify that your environment meets the following prerequisites:

- Resilient platform is version 30 or later.
- You have a Resilient account to use for the integrations. This can be any account that has the permission to view and modify administrator and customization settings, and read and update incidents. You need to know the account username and password.
- You have access to the command line of the Resilient appliance, which hosts the Resilient platform; or to a separate integration server where you will deploy and run the functions code. If using a separate integration server, you must install Python version 3.6 or later, and "pip". (The Resilient appliance is preconfigured with a suitable version of Python.)

Install the Python components

The functions package contains Python components that are called by the Resilient platform to execute the functions during your workflows. These components run in the Resilient Circuits integration framework.

The package also includes Resilient customizations that will be imported into the platform later.

Complete the following steps to install the Python components:

1. Ensure that the environment is up-to-date, as follows:

```
sudo pip install --upgrade pip
sudo pip install --upgrade setuptools
sudo pip install --upgrade resilient-circuits
```

2. Run the following command to install the package:

```
sudo pip install --upgrade fn_ldap_utilities-1.1.0.zip
```

Configure the Python components

The Resilient Circuits components run as an unprivileged user, typically named integration. If you do not already have an integration user configured on your appliance, create it now.

Complete the following steps to configure and run the integration:

1. Using sudo, switch to the integration user, as follows:

   ```
   sudo su - integration
   ```

2. Use one of the following commands to create or update the resilient-circuits configuration file. Use `-c` for new environments or `-u` for existing environments.

   ```
   resilient-circuits config -c
   ```

   or

   ```
   resilient-circuits config -u
   ```

3. Edit the resilient-circuits configuration file, as follows:

   a. In the [resilient] section, ensure that you provide all the information required to connect to the Resilient platform.

   b. In the [fn_ldap_multidomain_utilities] section, configure group of settings for each ldap/AD domain as needed. Each attribute group for a domain is identified with a label prepended to each attribute. There is no limit on how many domains you can add.

      When a function is called, the function input "ldap_md_domain_name" has to be completed with the corresponding label (without "**-**")

      For example, to configure connection to two domains "centraloffice" and "southbranch", where "centraloffice" is a **non-encrypted connection** to either **Active Directory or OpenLDAP,** whereas **"**southbranch" is an **encrypted connection over SSL to an Active Directory Server**, edit the settings as follows:

      ```
      [fn_ldap_multidomain_utilities]
      # Config to ldap_md_domain_name = centraloffice
      centraloffice-ldap_server = [ip address of your LDAP Server]
      centraloffice-ldap_port = 389
      centraloffice-ldap_use_ssl = false
      centraloffice-ldap_auth = SIMPLE [Can be ANONYMOUS, SIMPLE or NTLM]
      centraloffice-ldap_user_dn = cn=Username1,cn=Users,dc=example,dc=com,dc=ar
      # centraloffice-ldap_user_ntlm = domain\\user
      centraloffice-ldap_password = password
      centraloffice-ldap_is_active_directory = false
      centraloffice-ldap_connect_timeout = 10

      # Config to ldap_md_domain_name = southbranch
      southbranch-ldap_server = [ip address of your LDAP Server]
      southbranch-ldap_port = 636
      southbranch-ldap_use_ssl = True
      southbranch-ldap_auth = simple
      southbranch-ldap_user_dn = cn=Username,cn=Users,dc=example,dc=com,dc=ar
      # southbranch-ldap_user_ntlm = domain\\user
      ```

```
southbranch-ldap_password = password
southbranch-ldap_is_active_directory = True
southbranch-ldap_connect_timeout = 10
```

Deploy customizations to the Resilient platform

The package contains function definitions that you can use in workflows, and includes example workflows and rules that show how to use these functions.

1. Use the following command to deploy these customizations to the Resilient platform:

```
resilient-circuits customize
```

2. Respond to the prompts to deploy functions, message destinations, workflows and rules.

Run the integration framework

To test the integration package before running it in a production environment, you must run the integration manually with the following command:

```
resilient-circuits run
```

The resilient-circuits command starts, loads its components, and continues to run until interrupted. If it stops immediately with an error message, check your configuration values and retry.

Configure Resilient Circuits for restart

For normal operation, Resilient Circuits must run continuously. The recommended way to do this is to configure it to automatically run at startup. On a Red Hat appliance, this is done using a systemd unit file such as the one below. You may need to change the paths to your working directory and app.config.

1. The unit file must be named `resilient_circuits.service` To create the file, enter the following command:

```
sudo vi /etc/systemd/system/resilient_circuits.service
```

2. Add the following contents to the file and change as necessary:

```
[Unit]
Description=Resilient-Circuits Service
After=resilient.service
Requires=resilient.service
```

```
[Service]
Type=simple
User=integration
WorkingDirectory=/home/integration
ExecStart=/usr/local/bin/resilient-circuits run
Restart=always
TimeoutSec=10
Environment=APP_CONFIG_FILE=/home/integration/.resilient/app.config
Environment=APP_LOCK_FILE=/home/integration/.resilient/resilient_circuits.
lock

[Install]
WantedBy=multi-user.target
```

3. Ensure that the service unit file is correctly permissioned, as follows:

```
sudo chmod 664 /etc/systemd/system/resilient_circuits.service
```

4. Use the systemctl command to manually start, stop, restart and return status on the service:

```
sudo systemctl resilient_circuits [start|stop|restart|status]
```

Function Descriptions

Once the function package deploys the functions, you can view them in the Resilient platform Functions tab, as shown below. The package also includes example workflows and rules that show how the functions can be used. You can copy and modify these workflows and rules for your own needs.

LDAP Utilities: Add to Groups

This function allows adding any number of accounts to multiple groups.

Supports: Active Directory only

Inputs:

Name	Type	Required	Example
`ldap_md_domain_name`	String	Yes	`centraloffice`
`ldap_md_multiple_user_dn`	String representation of a List	Yes	`"['dn=tom smith,dc=example,dc=com', 'dn=ted smith,dc=example,dc=com']"`
`ldap_md_multiple_group_dn`	String representation of a List	Yes	`"['dn=Group1,dc=example,dc=com', 'dn=Group1,dc=example,dc=com']"`

Output:

Name	Type	Description
`results.success`	Boolean	True if users successfully added to groups; otherwise, False.
`results.domain_name`	String	Domain name in which the function was executed.
`results.users_dn`	List	List of users added to the groups.
`results.groups_dn`	List	List of groups where users were added.

Condition: The user and group DNs must be valid.

LDAP Utilities: Remove from Groups

This function allows removing any number of accounts from multiple groups.

Supports: Active Directory only

Inputs:

Name	Type	Required	Example
ldap_md_domain_name	String	Yes	centraloffice
ldap_md_multiple_user_dn	String representation of a List	Yes	"['dn=tom smith,dc=example,dc=com', 'dn=ted smith,dc=example,dc=com']"
ldap_md_multiple_group_dn	String representation of a List	Yes	"['dn=Group1,dc=example,dc=com', 'dn=Group2,dc=example,dc=com']"

Output:

Name	Type	Description
results.success	Boolean	True if users successfully removed from the groups; otherwise, False.
results.domain_name	String	Domain name in which the function was executed.
results.users_dn	List	List of users removed from the groups.
results.groups_dn	list	List of groups that had users removed.

Conditions:

- The group DNs must be valid.
- Only valid user DNs are removed from the groups. Any user DN that is not a member of the group or is invalid is ignored.

LDAP Utilities: Search

The function runs a search query against an LDAP server.

Supports: Active Directory and OpenLDAP

Inputs:

Name	Type	Required	Example
ldap_md_domain_name	String	Yes	centraloffice
ldap_md_search_base	String	Yes	"dc=example,dc=com"
ldap_md_search_filter	String	Yes	"(&(objectClass=person)(mail=%ldap_param%))"
ldap_md_search_attributes	String	No	"uid,cn,sn,mail,telephoneNumber"
ldap_md_search_param	String	No	"Einstein"

Output:

Name	Type	Description
`results.success`	Boolean	True if at least one entry was found.
`results.domain_name`	String	Domain name in which the function was executed.
`results.entries`	List	List of entries returned from LDAP. Each entry will always contain the DN Attribute. Each entry is a Dictionary with the Key being the Attribute and the Value being the Attributes Value. Note: Some Attribute Values can be a List.

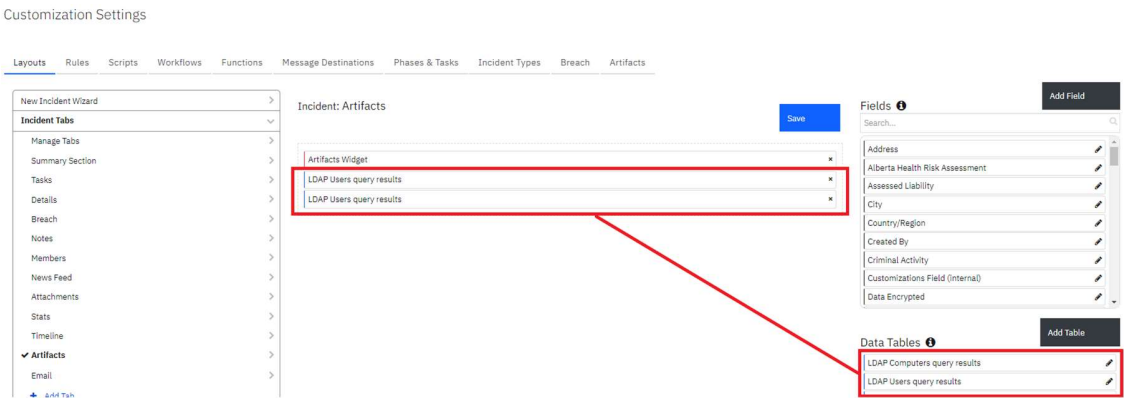
Conditions:

- The search_base and search_filter must be valid.
- If the %ldap_param% wildcard is used in the search_filter, the search_param input is required.

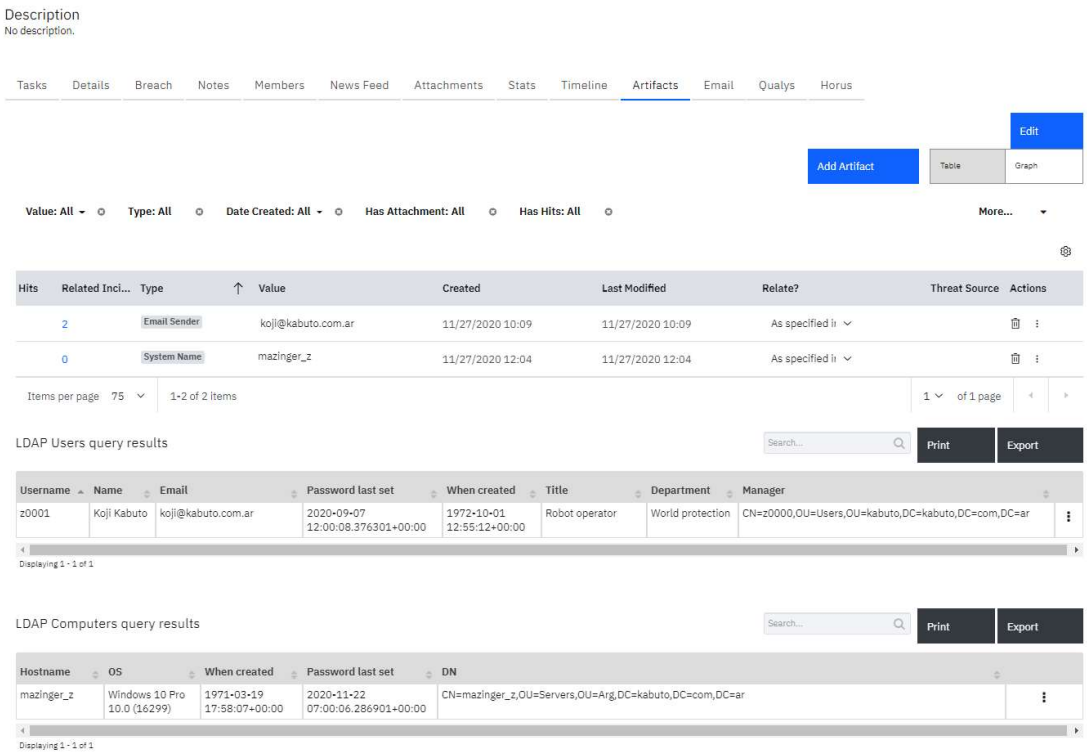
Additional Configuration (only for Workflows – "Example: LDAP MultiDomain Utilities: Search" and "Example: LDAP MultiDomain Utilities: Search Computer"):

To display query results, users need to manually add the "LDAP Users query results" and "LDAP Computers query results" data tables to the Artifacts tab.

1. Navigate to the Customization Settings and select the **Layouts** tab.
2. Select **Artifacts**.
3. Drag the "LDAP Users query results" and "LDAP Computers query results" and data tables to your Artifacts tab.
4. Click **Save**.



Example showing results in Artifacts tab:



LDAP Utilities: Set Password

This function allows you to set a new password for a given user.

Alternatively, you can let the function to auto generate a random password to set to the user by setting the Length param: `ldap_md_new_auto_password_len`. The resulting auto generated password will have the complexity: at least one upper case, at least one lower case, at least one number and at least one special character.

Supports: Active Directory and OpenLDAP

Inputs:

Name	Type	Required	Example
`ldap_md_domain_name`	String	Yes	`centraloffice`
`ldap_md_dn`	String	Yes	`"dn=tom smith,dc=example,dc=com"`
`ldap_md_new_password`	String	No	`"newpassword123"` Note: The new password you want to set for the entry. If empty will auto generate password with lenght as in ldap_md_new_auto_password_len (8 default)
`ldap_md_new_auto_password_len`	String	No	`"12"` Note: Length for auto-generated password
`ldap_md_return_new_password`	Boolean	Yes	Choose if new password is returned in results

Output:

Name	Type	Description
results.success	Boolean	True if user passwords were successfully changed.
results.domain_name	String	Domain name in which the function was executed.
results.user.dn	String	DN of the user whose password was changed.
results.new_password	String	If ldap_md_return_new_password was setted True, shows the password in clear text, otherwise, shows '********'

Condition: User DN must be valid.

LDAP Utilities: Toggle Access

This function allows enabling and disabling of an Active Directory account.

Supports: Active Directory only

Inputs:

Name	Type	Required	Example
ldap_md_domain_name	String	Yes	centraloffice
ldap_md_dn	String	Yes	"dn=tom smith,dc=example,dc=com"
ldap_md_toggle_access	Select	Yes	Enable/Disable [Pick from Dropdown]

Output:

Name	Type	Description
results.success	Boolean	True if user was successfully Enabled or Disabled.
results.domain_name	String	Domain name in which the function was executed.
results.user_dn	String	DN of the user whose access was changed.
results.user_status	String	Will be Enabled or Disabled.

Condition: User DN must be valid.

LDAP Utilities: Update

This function updates the attribute of a DN with a new value.

Supports: Active Directory and OpenLDAP

Inputs:

Name	Type	Required	Example
ldap_md_domain_name	String	Yes	centraloffice
ldap_md_dn	String	Yes	"dn=tom smith,dc=example,dc=com"
ldap_md_attribute_name	String	Yes	"mail"
ldap_md_attribute_values	String representation of a List	Yes	"['email1@example.com', 'email2@emample.com']"

Output:

Name	Type	Description
results.success	Boolean	True if user attributes were successfully updated.
results.domain_name	String	Domain name in which the function was executed.
results.user_dn	String	DN of the user whose attribute was updated.
results.attribute_name	String	Name of the attribute updated.
results.attribute_values	String representation of a List	List of the attribute's updated values.

Conditions:

- User DN must be valid.
- Attribute Name must be valid.
- The Attribute Values must meet the Custom Constraints set on your LDAP server.

Troubleshooting

There are several ways to verify the successful operation of a function.

- Resilient Action Status

  When viewing an incident, use the Actions menu to view Action Status. By default, pending and errors are displayed. Modify the filter for actions to also show Completed actions. Clicking on an action displays additional information on the progress made or what error occurred.

- Resilient Scripting Log

  A separate log file is available to review scripting errors. This is useful when issues occur in the pre-processing or post-processing scripts. The default location for this log file is: `/var/log/resilient-scripting/resilient-scripting.log`.

- Resilient Logs

  By default, Resilient logs are retained at `/usr/share/co3/logs`. The `client.log` may contain additional information regarding the execution of functions.

- Resilient-Circuits

  The log is controlled in the `.resilient/app.config` file under the section `[resilient]` and the property `logdir`. The default file name is `app.log`. Each function will create progress information. Failures will show up as errors and may contain python trace statements.

Support

For additional support, contact support@resilientsystems.com.

Including relevant information from the log files will help us resolve your issue.