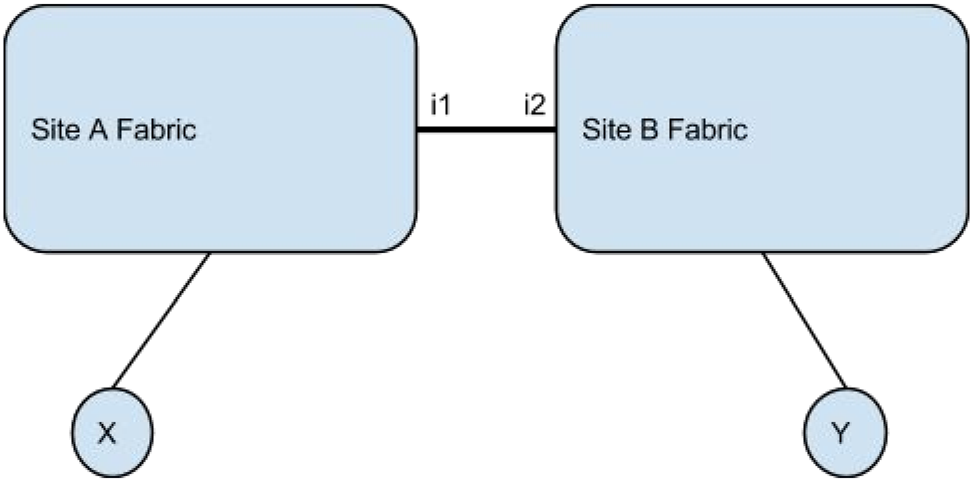
Multisite Application

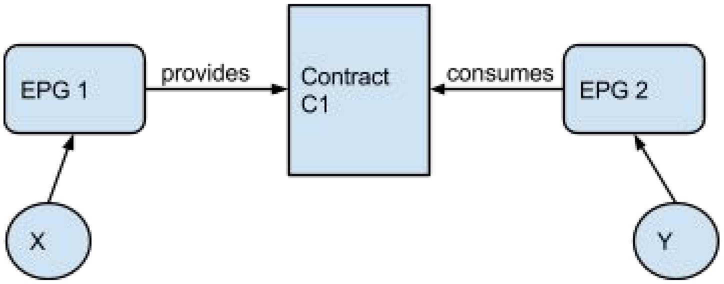
The Multisite application allows a Cisco Application Centric Infrastructure (ACI) fabric to be extended across multiple sites. These sites are independent ACI fabrics where each fabric has their own Application Policy Infrastructure Controller (APIC) cluster. The multisite application preserves the group based policy model of Cisco ACI by allowing a contract to be extended across multiple sites so that endpoint groups from different sites can communicate.

Example

An example of a multisite topology is described below using a 2 site datacenter fabric. The multisite application supports more than 2 sites, but we use 2 sites here for demonstration. The 2 sites are labeled as Site A and Site B. Both sites have their own APIC clusters. Within Site A, there is an endpoint labeled X and within Site B, there is an endpoint labeled Y. The 2 sites are connected via external routed network interfaces on interfaces labeled i1 and i2. There exists an L3 network between the 2 fabrics.



The desired application topology is defined as shown in the diagram below.



There are 2 endpoints, X and Y, and these endpoints are members of separate endpoint groups (EPG). Endpoint X belongs to EPG1 that provides contact C1 which is consumed by EPG2 that has endpoint Y as a member. Since X and Y are in different sites, the application topology must be “stretched” between them.

The Multisite application enables this topology by “exporting” the contract from 1 site to other sites. This allows the contract to be defined in a single site, exported to other sites, and then provided/consumed by the EPGs in the sites where the contract is present. It should be noted that not all contracts need to be exported to other sites and that the set of sites where a contract is exported could vary from contract to contract.

One issue that multisite deployments face is that EPG membership is not preserved as packets travel across generic IP networks between the sites. The multisite application alleviates this by communicating endpoint group membership between the sites. As endpoints connect to an EPG that is providing or consuming a contract that is being exported or imported between sites, the remote sites participating in the same contract will be automatically informed of the endpoint’s EPG membership.

Deployment Models

Exporting contracts between the sites enables 2 important deployment models. For the purposes of this discussion, we will refer to these 2 models as Remote Service Model and Stretched Service Model.

Remote Service Model

In the Remote Service Model, the EPG is largely isolated to a single site but the desire is that this service can be consumed by EPGs in other sites. In this scenario, the contract is defined where the service resides. This contract is provided by the EPG that wishes its services to be consumed by other sites. This contract is exported by the multisite application at the local site to the remote sites. The remote sites can then consume the contract as they wish. This is done through configuration on the APIC. For convenience, the imported contract name will be prefixed by the site name from where the contract was exported.

<pic remote service model>

Stretched Service Model

In the Stretched Service Model, the contract is provided by many or all of the sites participating in the contract. Similarly, the contract is consumed in the same set of sites. This allows forwarding to determine which endpoint will provide the service. This can be very useful in cases where the service is to be consumed locally but it is desirable to consume remote services in the case of failure. In this case, the contract can be defined in a single site for convenience and then exported to all of the other participating sites where it can then be consumed and provided as so desired.
<pic stretched service model>

Installation

The multisite application is included in the acitoolkit package. The acitoolkit package can be installed according to the instructions found at this link.

Application Usage

The multisite application can be found in the applications/multisite directory of the acitoolkit package. The multisite application runs as a standalone Python application. Each site will have an instance of the multisite application running. Each instance will be configured with credentials of the local site and all of the remote sites. There may be more than 1 instance of the application running per site for redundancy purposes. Each application can be run with or without the supplied GUI. These 2 usage modes are described below.

Command Line Usage

When run directly from the command line, the Multisite application loads its configuration from a file. This file contains the credentials for the APIC in the local site as well as each of the remote sites. It also contains the list of contacts that are to be exported from this site. When run, it will log in to each of the sites, export the contracts indicated in the configuration file, and continually monitor the endpoints connecting and disconnecting from the fabric. When an endpoint connects to an EPG that is consuming or providing a contract that is being imported or exported, the appropriate remote sites will be informed by installing a fully qualified /32 subnet on the remote site along with the set of contracts that are being provided or consumed by the endpoint.

It should be noted that this mode provides the best starting point for those wishing to include the multisite application within their own applications and/or portal. The file can be imported and the main() function can be examined to determine the touchpoints to include in the developer's application.

Command Line Options

The command line options are shown below.

```
python multisite.py -h
usage: multisite.py [-h] [--config CONFIG] [--generateconfig]

ACI Multisite Tool

optional arguments:
  -h, --help        show this help message and exit
  --config CONFIG   Configuration file
  --generateconfig  Generate an empty example configuration file
```

The generateconfig file option will generate an empty configuration file named `sample_config.json`. It will then print the following text and then exit.

```
python multisite.py --generateconfig
Sample configuration file written to sample_config.json
```

```
Replicate the site JSON for each site.
    Valid values for use_https and local are 'True' and 'False'
    One site must have local set to 'True'
Replicate the export JSON for each exported contract.
```

The content of the `sample_config.json` file is shown below:

```
{
    "config":[
        {
            "site":{
                "username":"",
                "name":"",
                "ip_address":"",
                "password":"",
                "local":"",
                "use_https":""
            }
        },
        {
            "export":{
                "tenant":"",
                "contract":"",
                "sites":[
                    {
                        "site":{
                            "name":""
                        }
                    }
                ]
            }
        }
    ]
}
```

A sample configuration file with filled in values is shown below:

```
{
    "config":[
        {
            "site":{
                "username":"admin",
                "name":"Site1",
                "ip_address":"1.2.3.4",
                "password":"mypassword",
                "local":"True",
                "use_https":"True"
            }
        },
        {
            "site":{
                "username":"admin",
                "name":"Site2",
                "ip_address":"3.4.5.6",
                "password":"mypassword",
                "local":"False",
                "use_https":"True"
            }
        },
        {
            "export":{
                "tenant":"production",
                "contract":"mysql",
                "sites":[
                    {
                        "site":{
                            "name":"Site2"
                        }
                    }
                ]
            }
        }
    ]
}
```

GUI Usage

The application can also run as a web based application and by default will run on TCP port 5000 and the loopback IP address (127.0.0.1) of the machine where it is run. It can be executed with the following command:

```
python multisite-gui.py
```

The following options are available:

```
python multisite-gui.py [-h] [--ip IP] [--port PORT] [--test]
optional arguments:
  -h, --help    show this help message and exit
  --ip IP       IP address to listen on.
  --port PORT  Port number to listen on.
  --test        Enable functions for lab testing.
```

Site Credentials

Once the application is running, the credentials must be entered for the various sites. The credentials consist of the following parameters:

- Site name
- IP address
- username
- password
- use https
- local

The site name is simply a string to represent the site and must be unique. The IP address is that of the APIC in the site corresponding to the site name. The username and password are used to login to that APIC and the 'use https' checkbox indicates whether the connection to the APIC should use http or https. The 'local' checkbox indicates that this site is the local site for this instance of the multisite application. There must be one local site configured in each multisite application.

The credentials are shown in a table format where the columns consist of site name, APIC IP address, username, an indicator showing whether the communication occurs via HTTP or HTTPS, and an indicator showing whether the site is local or remote. For security purposes, the password is not shown in a column. Each column is sortable by clicking on the column heading. When the number of sites goes beyond a certain number, table paging is provided at the bottom of the table to view additional site credentials. At the top right, there is a search text field that will limit the table rows to those matching the free-form text entered. Next to that, there are 2 pulldown menus labeled 'With selected' and 'Add Filter'. The 'With selected' provides a Delete option that will delete all of the credentials in the rows that the checkbox has been selected. The 'Add Filter' pulldown menu provides the ability to apply a specific search filter to certain columns within the table. To the right of the pulldown menus is a button labeled 'Create', this button will add a new site to the credentials list. Clicking on this button will pull up a new screen where the credentials can be entered.

Site Contracts

The site contracts menu item displays all of the contracts visible to the multisite application on the local site. Each contract is listed in a table with the columns of the table consisting of tenant name, contract name, export state, and remote sites. The export states are exported, imported, or local. Local contracts are those that are not participating in any other site but may be exported if desired. Imported contracts are those defined in other sites but have been exported to this site. Exported contracts are the locally defined contracts that are being exported from this site to other sites.For imported contracts, the Remote site column contains the site that exported the contract to this site. For exported contracts, the Remote site column contains the list of Remote site names where this contract has been exported.

Local contracts can be exported and 'unexported' by selecting the contract row checkbox and selecting the 'Change Export Settings' found under the 'With selected' pulldown menu.

EPGs

The EPGs menu item displays the EPG table. The EPG table contains columns consisting of tenant name, application profile name, EPG name, and contract name. This table is simply informative and shows the EPGs that are participating in contracts that are being imported or exported to or from this site.

Known Issues / Remaining Work Items / Restrictions

As of this writing (6/20/15), this application remains a work in progress. This section contains a list of the known issues and remaining work items as well as assumptions made when developing the application.

GUI related

- Although the application is relatively stateless, the GUI uses a local sqllite database to store the credentials and also the contract data. The contract data is thrown away and rebuilt from the tags placed in the APIC but the credentials is kept in the database. The database has a default filename which means that only one instance of the application can be currently run from a given directory. If you would like to run multiple copies of the application on the same system, it is recommended that you clone the application into separate directories. If inconsistencies arise in the database file, it can simply be deleted, the application restarted, and the credentials reentered. The database file will automatically be rebuilt. The database file is currently named multisite_db.sqlite.

Standalone related

- The standalone command line version of the application currently requires the application to be restarted to read in any changes to the configuration file.

General

- It is assumed that tenant names are consistent between the sites for the tenants that will have contracts imported and exported.
- It is assumed that the site names are consistent across all instances of the multisite application.
- Changes within a given contract are not currently synced on changes
- Connection to the APIC is currently reestablished when connection is lost
- Use same tenant name and consistent sitenames

Test coverage

- The automated test suite does not currently cover usage of tenant common and multiple l3extOuts.

Architecture / Design Details

- tags
  - inserted per tenant to store state for the exported/imported contracts
  - inserted in each l3extInstP that the application installs to determine EPG membership and installing site when the endpoint is deleted
- one app per site
- exports contract with the prepended site name
  - indicates where the contract was defined & where it should be updated
- Endpoint connectivity
  - install /32 in remote sites that are participating in the contract