sales@candelatech.com
support@candelatech.com
+1 (360) 380-1618 [PST, GMT -8]

Multiplexed REST Access via Nginx Proxy

Goal: Configure an NGINX proxy to allow REST traffic to a variety of isolated LANforge machines

It is possible to configure a Nginx proxy in a manner to allow remote REST clients access to multiple isolated LANforge systems. This leverages the proxy_pass feature in Nginx. There are multiple ways to configure proxy access.

For the example below, we will assume these values:

- public proxy hostname is bizproxy, 10.39.0.44
- bizproxy is running Nginx
- Isolated LAN with LF machines: 192.168.92.0/24
- Example LANforge machines:
  - 192.168.92.10 ct523-jedway1
  - 192.168.92.11 ct522-jedway3
- the LANforge machines need to have GUIs configured to start automatically

LANforge GUI HTTP Processing

The HTTP library that the LANforge GUI incorporates is very simple. It is not configured to parse Host: headers. There is no need to rewrite the Host header when proxying to port 8080.

Proxying to Apache on LANforge (mgt_ip, port 80) **is different.** If you want to proxy requests to a LF Apache instance on port 80, you should incorporate Host header rewriting. (No examples below, sorry.)

Proxy Request Rewriting

Three ways of making proxy requests include:

- **Port Rewriting**. Works best with our python libraries.
- Hostname Rewriting, more difficult, but still works with python libraries.
- URL (path-name) Rewriting: this does NOT work well with our python libraries.

Port Rewriting

This manner of proxying just translates different server listening ports to the target machines. It is another easy transformation, but it opens up quite a number of high-numbered ports on bizproxy.
Nginx config:

```
server {
   listen 1910;
   server name  ;
   root /usr/share/nginx/html;

    location / {
       rewrite            /(.*) /$1 break;
```

```
       proxy pass         http://192.168.92.10:8080;
       proxy redirect     off;
       proxy set header   Host $host;
       proxy_set_header   X-Real-Ip $remote_addr;
       proxy_set_header   X-Forwarded-For $remote_addr;
    }
}
server {
   listen 1911;
   server name  ;
   root /usr/share/nginx/html;

    location / {
       rewrite            /(.*) /$1 break;
       proxy pass         http://192.168.92.11:8080;
       proxy redirect     off;
       proxy set header   Host $host;
       proxy_set_header   X-Real-Ip $remote_addr;
       proxy_set_header   X-Forwarded-For $remote_addr;
    }
}
```

Use curl to test access:

```
curl -sqv -H 'Accept: application/html' http://bizproxy:1910/port/1/1/list
```

Example script usage:

```
./scenario.py --mgr bizproxy --mgr_port 1910 \
    --load BLANK --action overwrite
```

Hostname Rewriting

It is possible to rewrite hostnames and host headers to isolated LF systems. This is **complicated** rewrite because the DNS names need to be present at the developer's workstation. (It is unlikely that the the headers in the HTTP request can be manipulated to add the Host header.) Ideally, the non-isolated LAN DNS can be configured to return the return the IP of bizproxy.corp.me when hostnames like ct523-jedway1.bizproxy.corp.me are requested.

On the developer workstation, this is possible with extra effort on the user side by manipulating the `/etc/hosts` file on a workstation:

```
# etc/hosts
10.39.0.44    ct523-jedway1.bizproxy.corp.me    ct523-jedway1
```

Nginx config:

```
server {
    listen 80;
    server name ct523-jedway1;
    root /usr/share/nginx/html;

    location / {
        rewrite            /(.*) /$1 break;
        proxy pass         http://192.168.92.10:8080;
        proxy redirect     off;
        proxy set header   Host $host;
        proxy_set_header   X-Real-Ip $remote addr;
        proxy_set_header   X-Forwarded-For $remote_addr;
    }
}
```

Check the URL access using curl:

```
# check by IP:
$ curl -sqv \
    -H 'Host: ct523-jedway1' \
    -H 'Accept: application/json' \
    http://10.39.0.44/port/1/1/list

# check by hostname
$ curl -sqv \
    -H 'Accept: application/json' \
    http://ct523-jedway1.bizproxy.corp.me/port/1/1/list
```

Example script usage:

```
./scenario.py --mgr ct523-jedway1 --mgr_port 80 \
   --load BLANK --action overwrite
```

Logging HTTP Access

The bizproxy logs should be located in `/var/log/nginx`. In LF 5.4.6, the GUI can send messages to syslog. Messages from the GUI would look like:

```
1685573102952:  ip[192.168.92.1] sess[] GET url[/port/1/1/list]
```

Appendix

URL Rewriting is mentioned here so the reader can understand what not to configure.

URL Rewriting

Below is an example permitting REST access to LF hosts by way of a URL prefix. For example, the URL http://bizproxy/92.11/port/1/1/list becomes the URL http://192.168.92.11:8080/port/1/1/list . This is not the best kind of proxy rewriting, but it is the easiest. Using a URL prefix is less ideal because it inherently conflicts with the LANforge python libraries provided.
Nginx config:

```
server {
    listen       80;
    server_name   ;
    root         /usr/share/nginx/html;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location /92.10 {
        rewrite            /92.10/(.*) /$1 break;
        proxy_pass         http://192.168.92.10:8080;
        proxy_redirect     off;
        proxy_set_header   Host biz lflab5 9210;
        proxy_set_header   X-Real-Ip $remote_addr;
        proxy_set_header   X-Forwarded-For $remote_addr;
    }
    location /92.11 {
        rewrite            /92.11/(.*) /$1 break;
        proxy_pass         http://192.168.92.11:8080;
```

```
        proxy redirect     off;
        proxy set header   Host $host;
        proxy set header   X-Real-Ip $remote addr;
        proxy_set_header   X-Forwarded-For $remote_addr;
    }
}
```

Use curl to query the REST endpoint:

```
$ curl -sqv -H 'Accept: application/json' http://bizproxy/92.10/port/1/1/list
```

This is not compatible with the py-scripts library.