# iRules Recipe 2: Transparent Request Target Rewriting

**Vernon, 2016-03-10**

## The Problem

You want to show a maintenance page for pages related to a specific application (called *borneo*), but don't want to change the URL that appears in the browser location bar.

## The Code

```
when HTTP_REQUEST {
    switch -glob [HTTP::path] {
        "/apps/borneo/*" -
        "/apps/cgi-bin/borneo/*" {
            HTTP::path "/general/hubble-maintenance.html"
        }

        "/apps/images/borneo/*.gif" {
            HTTP::host "images.internal.example.com"
            HTTP::path "/general/images/maintenance.gif"
        }

        "/apps/images/borneo/*.png" {
            HTTP::host "images.internal.example.com"
            HTTP::path "/general/images/maintenance.png"
        }
    }
}
```

## Analysis

Regardless of the hostname in the URL requested, this code analyzes the Request Target path of the request. Generically speaking, the parameters for the `switch` command are:

```
  switch <flags> <string> {
      <match-case1> {
          <case1-code>
      }

      <match-case2> {
          <case2-code>
      }

      < ... etc ... >
  }
```

`switch` compares the <string> against possible matches. When it finds a match, it executes the corresponding code, then the switch exits (and, importantly, does not match more than one case). In the code above, the switch looks at the Request Target path element (to understand how `HTTP::path` differs from `HTTP::uri` and why I chose `HTTP::path`, rather than `HTTP::uri`, see the "**Elaboration**" section below). Normally, `switch` treats each case as an exact matcher, and will only match if the <string> is identical to the case. However, the `-glob` flag tells switch to use "glob-style matching". In glob matching, an asterisk (*) means "match zero or more characters", a question-mark (?) means "match any one character", and characters in square brackets ([...]) match any single character in the brackets.

The first case in the code above is "*/apps/borneo/**". This case will match if `HTTP::path` starts with "*/apps/borneo/*". I say "starts with" because the globbing asterisk will match any number of characters (including no characters!) after "*/apps/borneo/*". Notice, however, that the "code" for this case is just a dash (-). This is a "fall-through" for switch. It tells the switch command that, if this case matches, then execute the code for the next case. This allows you to use the same code for more than one case without having to copy-and-paste the code for each condition.

The code for the first two conditions changes the incoming HTTP Request Target to "*/general/hubble-maintenance.html*". BIG-IP is a full-proxy. When the `HTTP::uri` command is passed a parameter, it changes the Request Target in an HTTP Request message to the new value before passing the request to the pool member. This change is not visible on the client-side, so the user-agent never knows the change is made. Moreover, the pool member never knows that the Request Target was changed.

The next case -- "*/apps/images/borneo/*.gif*" -- matches any file under */apps/images/borneo* that ends with **.gif*. In this case, it transparently changes the **Host** header value to "*images.internal.example.com*" and the Request Target to "*/general/images/maintenance.gif*". The last case does the same thing, but for PNG files.

## Elaboration

If you type in http://www.example.com/hubble/index.html, the HTTP Request message will have a Start-Line and headers something like this:

```
GET /apps/images/borneo/logo.png HTTP/1.1
Host: www.example.com
  User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko)
  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
  Accept-Encoding: gzip, deflate, sdch
  Accept-Language: en-US,en;q=0.8,da;q=0.6
  Connection: keep-alive
```

Because BIG-IP normally operates as a full-proxy, when data arrives on the client-side, the BIG-IP creates a copy, and sends it over an independent connection to the pool member on the server-side. It is because of this that the BIG-IP can change the Request Target (or, for example, the **Host** header) before forwarding the request to the server. The client is "unaware" of this change, and from the perspective of the server, the BIG-IP is the client. This is arguably one of the most powerful aspects of the BIG-IP, allowing data to not only be inspected but altered as it transits through the proxy.

In the code above, `HTTP::uri` changes the Start-Line and `HTTP::host` changes the **Host** header. For the message provided immediately above, the server-side copy (that is, the HTTP Request message sent from the BIG-IP to the server) would look like this, based on the iRule:

```
  GET /general/images/maintenance.png HTTP/1.1
  Host: images.internal.example.com
  User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko)
  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
  Accept-Encoding: gzip, deflate, sdch
  Accept-Language: en-US,en;q=0.8,da;q=0.6
  Connection: keep-alive
```

```
Connection: keep-alive
```

Notice that I matched `HTTP::path` and not `HTTP::uri.` As you can see in the example message above, the Start-Line of an HTTP Request message is:

```
<METHOD> <Request Target> HTTP/<version>
```

The Request Target consists of a path, and optionally, query parameters. If those are included, it might look like this:

```
GET /cgi-bin/reflect.cgi?userid=wells&source=10.11.1.5
```

As you might have guessed, `HTTP::path` returns only the path portion (*/cgi-bin/reflect.cgi,* in this example), while `HTTP::uri` returns the entire Request Target. In my code above, I'm interested in matching on the path part only.

## How Else Could I Have Solved This?

Starting in BIG-IP version 11.4, Local Traffic Policies were introduced. Describing this feature is -- as they say -- outside the scope of this article, but in brief: it's a really cool way to do common HTTP transforms, including transparent rewrites. It has the advantage of not being code, and it is part of the built-in feature set. *Local Traffic Policies* can be modified using the BIG-IP web UI or tmsh. However, to give you a taste, here is what the same logic would be as a *Local Traffic Policy* on 12.1:

```
ltm policy borneo-maintenance {
    requires { http }
    rules {
        borneo-html {
            actions {
                0 {
                    http-uri
                    replace
                    value /general/hubble-maintenance.html
                }
            }
            conditions {
                0 {
                    http-uri
                    path
                    starts-with
                    values { /apps/borneo/ /apps/cgi-bin/borneo/ }
                }
            }
        }
        borneo-images-gif {
            actions {
                0 {
                    http-uri
                    replace
                    value /general/images/maintenance.gif
                }
                1 {
                    http-host
                    replace
                    value images.internal.example.com
                }
            }
```

```
            }
            conditions {
                0 {
                    http-uri
                    path
                    starts-with
                    values { /apps/images/borneo/ }
                }
                1 {
                    http-uri
                    path
                    ends-with
                    values { .gif }
                }
            }
            ordinal 1
        }
        borneo-images-png {
            actions {
                0 {
                    http-uri
                    replace
                    value /general/images/maintenance.png
                }
                1 {
                    http-host
                    replace
                    value images.internal.example.com
                }
            }
            conditions {
                0 {
                    http-uri
                    path
                    starts-with
                    values { /apps/images/borneo/ }
                }
                1 {
                    http-uri
                    path
                    ends-with
                    values { .png }
                }
            }
            ordinal 1
        }
    }
    status published
    strategy first-match
}
```

Now you're probably thinking "Wow! That's a whole lot more complicated than the iRule above. Why would I ever use this method?" If you were to actually load this policy, then view it in the Web UI (Local Traffic >> Policies >> Policy List) you would, I think, have a much clearer idea of how easy it can be to build policies.

F5 Networks, Inc. | 401 Elliot Avenue West, Seattle, WA 98119 | 888-882-4447 | f5.com

F5 Networks, Inc.
Corporate Headquarters
info@f5.com

F5 Networks
Asia-Pacific
apacinfo@f5.com

F5 Networks Ltd.
Europe/Middle-East/Africa
emeainfo@f5.com

F5 Networks
Japan K.K.
f5j-info@f5.com