Switch Gone Wild: Using Wildcards with the Tcl &quot;switch&quot; command

Deb Allen, 2008-24-06

In addition to the built-in Tcl relational operators, iRules includes a number of custom relational operators that were intended to be both intuitive and backward compatible with BIG-IP 4.x iRules. For most simple relational comparisons, the custom iRules operators are more efficient, but there are many cases for which a case construct such as the Tcl "switch" command is a better choice for testing multiple conditions.

In this article, I will demonstrate using the Tcl "switch" command to emulate the custom iRules operators "starts_with", "ends_with", "contains", and "equals".

switch

The switch command is built into Tcl. It provides a framework to execute one of several script bodies depending on a given value.

The basic switch construct looks like this:

```
switch (string) {
    "pattern1" { script body }
    "pattern2" { script body }
    "pattern3" { script body }
    default    { script body }
}
```

The switch command matches one string against a list of patterns in order. As soon as it finds a pattern that matches, it evaluates the associated script body. If the last pattern argument is ***default*** then it matches anything. If no pattern matches the string and no default is given, then the switch command returns an empty string.

equals

The equals operator tests if one string equals another string exactly. The default action of the switch command is to match the string exactly, which is equivalent to the iRules "equals" operator. The default exact match is useful in cases where you are comparing a whole string to see if it matches a specific pattern.

An example of the exact matching capability would be to test for specific URI requests, as in the LTM MaintenancePage codeshare entry:

```
# Return the requested page
switch $uri {
  "/"              -
  "/index.html"   { HTTP::respond 200 content [lindex $::maint_index_html 0]           "Content-Type"
  "/logo.png"     { HTTP::respond 200 content [b64decode [lindex $::maint_logo_png 0]] "Content-Type"
  default         { HTTP::respond 404 }
}
```

Notice in this case that the string we are trying to match is the exact value you wish to act upon.

starts_with

The starts_with operator tests if one string starts with another string. To create the equivalent comparison with the switch command, you can use the "*-glob*" option, which allows you to use wildcards to look for partial matches.
For the string to match a pattern, their contents must be identical except for except for any defined character sets or the following wildcard characters that appear in the pattern:

- ***** (asterisk) matches any sequence of characters in ***string***, including a null string.
- **?** (question mark) matches any single character in ***string***.

To simulate the "*starts_with*" matching behaviour using the switch command, you can use the * (asterisk) wildcard at the end of the pattern.

An example of the "*starts_with*" capability would be to test for specific content type categories, as in the first pattern in the Content Type Tracking codeshare entry:

```
switch -glob [HTTP::header "Content-Type"] {
   "image/*"         { STATS::incr "ContentType" "Images" }
   "text/html"       { STATS::incr "ContentType" "HTML" }
   "text/css"        { STATS::incr "ContentType" "Stylesheets" }
   "*javascript"     { STATS::incr "ContentType" "Scripts" }
   "text/vbscript"   { STATS::incr "ContentType" "Scripts" }
   "application/pdf" { STATS::incr "ContentType" "Documents" }
   "application/msword" { STATS::incr "ContentType" "Documents" }
}
```

The first pattern above will match on any Content-Type header that starts with "image/".

ends_with

The ends_with operator tests if one string ends with another string. To create the equivalent comparison with the switch command, you can again use the "*-glob*" option and the * wildcard.

To simulate the "*ends_with*" matching behaviour using the switch command, you can use the * (asterisk) wildcard at the *beginning* of the pattern.

An example of the "*ends_with*" capability would be to test for certain file types by extension, as demonstrated in the Pool Based On Extension codeshare entry:

```
switch [HTTP::path] {
  "*.jpg"        -
  "*.gif"        -
  "*.png"        { pool image_pool }
  "*.pdf"        { pool pdf_pool }
  default        { pool web_pool }
}
```

contains

The contains opearator tests if one string contains another string. To create the equivalent comparison with the switch command, you can again use the "*-glob*" option and two * wildcards.

To simulate the "*contains*" matching behaviour using the switch command, you can use the * (asterisk) wildcard at the *beginning and the end* of the pattern.

An example of the "contains" functionality is demonstrated in the ControllingBots codeshare entry:

```
switch -glob [string tolower [HTTP::header User-Agent]] {
   "*scooter*" -
   "*slurp*" -
```

```
    "*msnbot*" -
    "*fast-*" -
    "*teoma*" -
    "*googlebot*" {
      # Send bots to the bot pool
       pool slow_webbot_pool
    }
    default {
       # Send all other requests to a default pool
       pool default_pool
    }
}
```

Summary

These are not all the options offered, or even the only syntax you can use with the Tcl switch command but should be helpful in cleaning up some of those long ugly "*if / elseif*" chains for which you didn't realize there was a better option. If you need even more flexible case matching options, be sure to take a closer look at the man page for switch to familiarize yourself with the other capabilities of the "*-glob*" option, and other options such as "*-regexp*" and "*-nocase*".

Here are links to the full documentation of the operators we worked with in this article:

- switch - Evaluates one of several scripts, depending on a given value.
- equals - Tests if one string equals another string.
- starts_with - Tests if one string starts_with another string
- ends_with - Tests if one string ends with another string.
- contains - Tests if one string contains another string.

And to the codeshare entries:

- LTM MaintenancePage
- Content Type Tracking
- Pool Based On Extension
- ControllingBots

Get the Flash Player to see this player.
20080624-SwitchGoneWild.mp3

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

©2016 F5 Networks, Inc. All rights reserved. F5, F5 Networks, and the F5 logo are trademarks of F5 Networks, Inc. in the U.S. and in certain other countries. Other F5 trademarks are identified at f5.com. Any other products, services, or company names referenced herein may be trademarks of their respective owners with no endorsement or affiliation, express or implied, claimed by F5. CS04-00015 0113