Random Letter Generator

Use Case:

There are several use cases where you may want a random letter generator utility to use it with different kind of content changes we do while parsing. This example will demonstrate how to generate a string of pseudo-random letters.

iRule:

```
when RULE_INIT {

   # Number of random letters to generate
   set count 100

   # Create a list of the letters indexed 0 through 25
   set letters [ list a b c d e f g h i j k l m n o p q r s t u v w x y
z ]

   # Initialize a variable to store the random letters in
   set random ""

   # Loop through X times where X is the number of random letters to ge
nerate
   for { set i 1 } { $i < $count } { incr i } {

      # Generate a random number between 0 and 1, using rand()
      # Multiply that by 26 to get a number representing a letter
      # Use int to trim off the decimal value

      # set rand [expr { int (rand() * 26) }]
      # append random [lindex $letters $rand]
      # Or in one command:
      append random [lindex $letters [expr { int (rand() * 26) }]]
   }
  log local0. "Random letters: $random"
```

```
}
```

Netscaler Solution:

We are using NetScaler Policy Extensions (custom written LUA function) to achieve this

LUA Script:

```
function NSNUM:rand_letters() : NSTEXT
    local input = self
    local out = ""
    local s =  ""
    for i = 1, input do
        s = s .. string.char(math.random(97, 122)) -- Generate random number from 97 to
122(a-z), turn it into character and add to string
    end
    return s -- Return string
end
```

The above script should be named as rand_letters.lua and placed in /var/tmp directory in Netscaler for the below configs to work.

NS Configs:

```
add responder action act1 respondwith "\"HTTP/1.1 200 OK\r\nHEX:
\"+http.req.header(\"String_Input\").typecast_num_at.rand_letters+\"\r\n\r\n\""
add responder policy pol1 true act1
```

NetScaler has the ability to use LUA based scripts and functions with feature processing. These are called policy extensions as they create extended routine which can be used in NetScaler advance policy infrastructure.