iRules Challenge Results: Can Everyone Win?

Colin Walker, 2011-06-05

It would seem that, at least in this contest, everyone can indeed be a winner. I got the distinct pleasure to, once again, help contribute to the iRules ~~delinquency~~ education of our inbound FSE crew while they were here in Seattle for their boot camp. These folks are all technical, but very few have any iRules experience. Heck, most don't even have much scripting experience, so to lay down the gauntlet and say "Here, act like I'm a customer giving you these requirements, and go write me an iRule that'll make me happy." is a tall order. I'm very pleased to report, though, that they answered that challenge with vigor.

Amongst the entries I received there were a host of solid efforts and obvious understanding of the challenge, the inner workings of the LTM and a very solid showing of budding iRules knowledge. It was also obvious that, as I had hoped, the challengees made heavy use of DevCentral for ideas. Sourcing publicly available code is the first step for almost any coder these days, and iRules are no different. If they hadn't made use of the CodeShare, search functions, the forums, etc. I'd be highly disappointed.

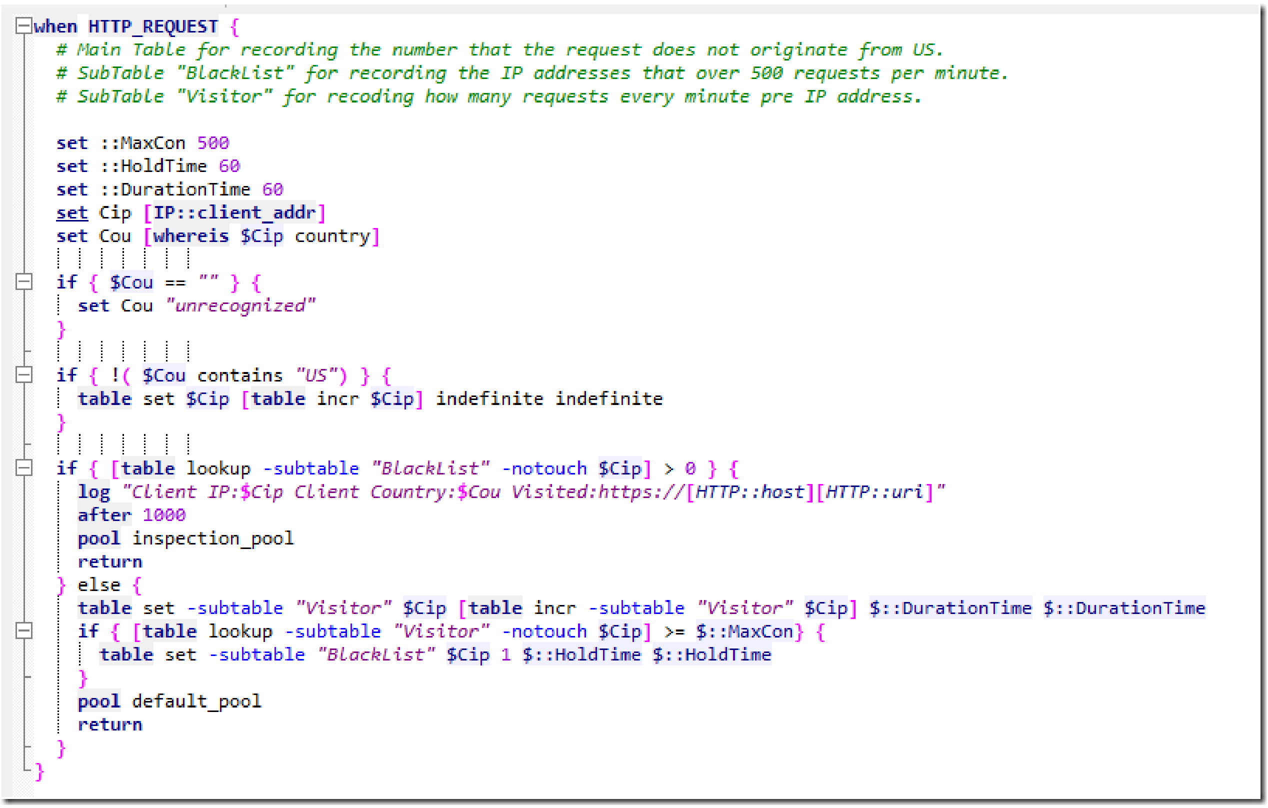
All of that being said, we have to declare a winner. Even with the overall quality of the entries being so high, there were a few that stood out as real gems. Before we get to the top entries, a quick reminder about the challenge. If you recall from my last post, here's what they were up against:

Requirements:

1. Identify the origin country of each incoming HTTPS request
2. If the request does not originate from within the US, track the number of requests being issued from that source.
3. If a single source makes more than 500 requests per minute, perform the following:
    1. Log the IP, country of origin, and the page requested that put them over the limit.
    2. Delay all requests from that source by 1 second for the next minute
    3. For the duration of that minute, forward all requests from that particular source to a separate pool for further inspection (in addition to the default pool, not in place of it)

And now, without further adieu, the Top3 FSE iRules Challenge Entries:

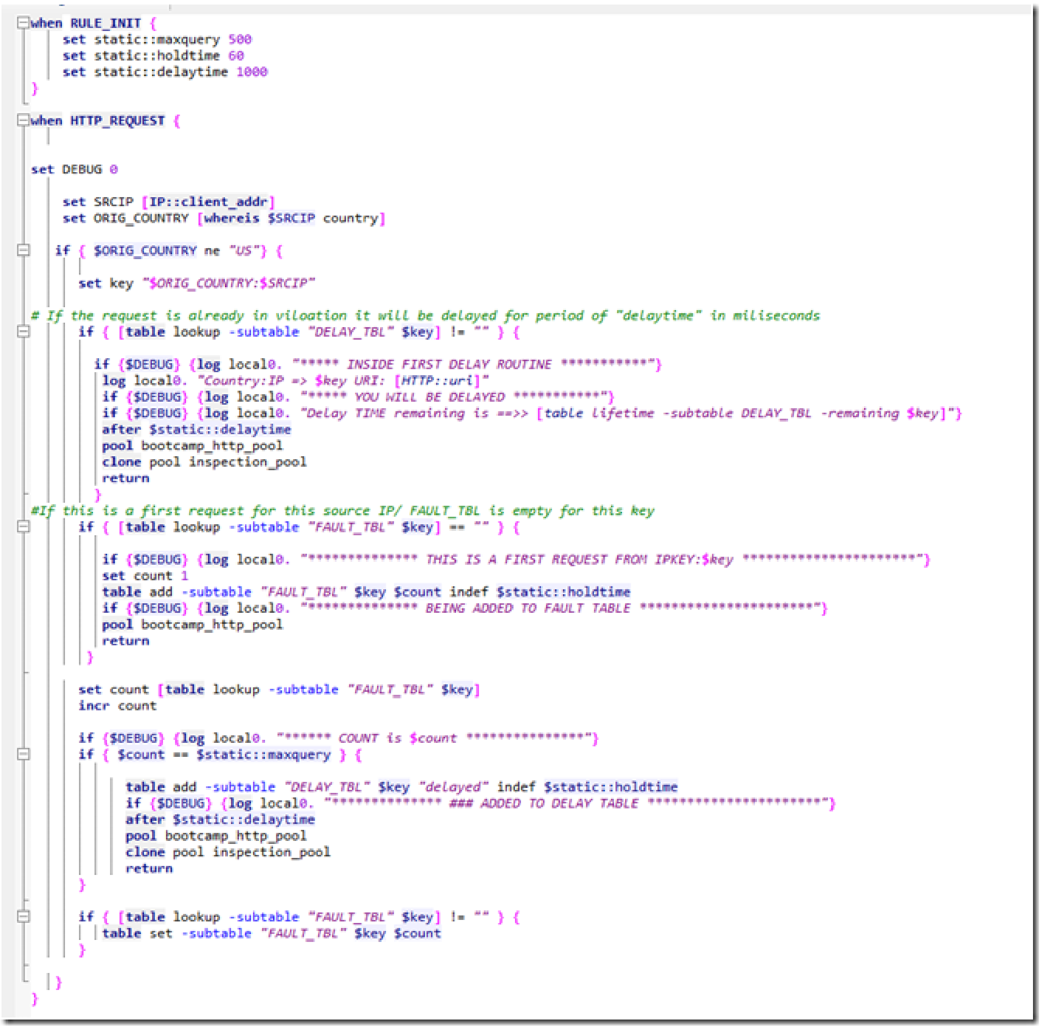
3.) First on the list, our Second Runner Up, Gavin Zhou:



```
when HTTP_REQUEST {
  # Main Table for recording the number that the request does not originate from US.
  # SubTable "BlackList" for recording the IP addresses that over 500 requests per minute.
  # SubTable "Visitor" for recoding how many requests every minute pre IP address.

  set ::MaxCon 500
  set ::HoldTime 60
  set ::DurationTime 60
  set Cip [IP::client_addr]
  set Cou [whereis $Cip country]

  if { $Cou == "" } {
    set Cou "unrecognized"
  }

  if { !( $Cou contains "US") } {
    table set $Cip [table incr $Cip] indefinite indefinite
  }

  if { [table lookup -subtable "BlackList" -notouch $Cip] > 0 } {
    log "Client IP:$Cip Client Country:$Cou Visited:https://[HTTP::host][HTTP::uri]"
    after 1000
    pool inspection_pool
    return
  } else {
    table set -subtable "Visitor" $Cip [table incr -subtable "Visitor" $Cip] $::DurationTime $::DurationTime
    if { [table lookup -subtable "Visitor" -notouch $Cip] >= $::MaxCon} {
      table set -subtable "BlackList" $Cip 1 $::HoldTime $::HoldTime
    }
    pool default_pool
    return
  }
}
```

Gavin showed a very solid grasp of what he needed to do, and made some wise choices in how he approached the task at hand. Multiple tables for sorting the blacklist vs. counts, very close to the entire list of required functionality, and some good looking code were among the strong points for Gavin's entry. Once he figures out how to use the static:: variables rather than demoting to CMP, he'll quickly be an iRuler to watch.

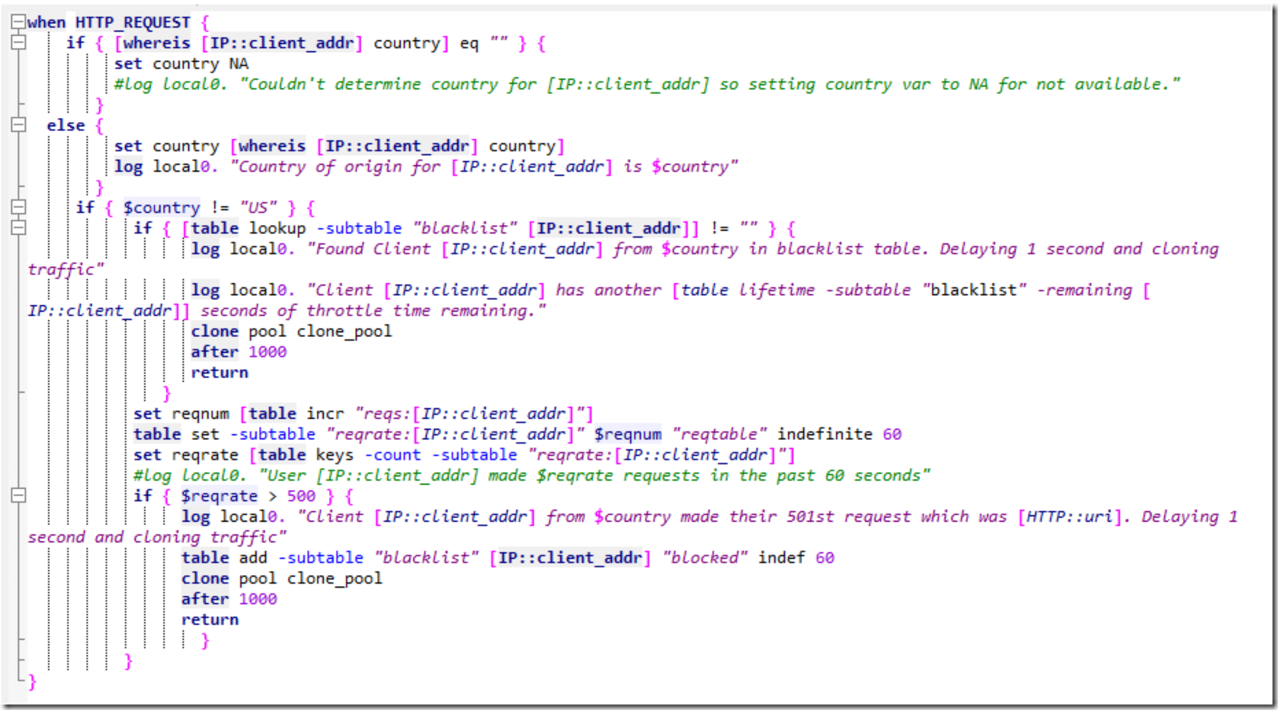
2.) Next up, our First Runner Up, Sachin Thatte:



```
when RULE_INIT {
    set static::maxquery 500
    set static::holdtime 60
    set static::delaytime 1000
}

when HTTP_REQUEST {


set DEBUG 0

    set SRCIP [IP::client_addr]
    set ORIG_COUNTRY [whereis $SRCIP country]

   if { $ORIG_COUNTRY ne "US"} {

      set key "$ORIG_COUNTRY:$SRCIP"

# If the request is already in viloation it will be delayed for period of "delaytime" in miliseconds
      if { [table lookup -subtable "DELAY_TBL" $key] != "" } {

        if {$DEBUG} {log local0. "***** INSIDE FIRST DELAY ROUTINE ***********"}
         log local0. "Country:IP => $key URI: [HTTP::uri]"
         if {$DEBUG} {log local0. "***** YOU WILL BE DELAYED ***********"}
         if {$DEBUG} {log local0. "Delay TIME remaining is ==>> [table lifetime -subtable DELAY_TBL -remaining $key]"}
         after $static::delaytime
         pool bootcamp_http_pool
         clone pool inspection_pool
         return
        }
#If this is a first request for this source IP/ FAULT_TBL is empty for this key
      if { [table lookup -subtable "FAULT_TBL" $key] == "" } {

         if {$DEBUG} {log local0. "************** THIS IS A FIRST REQUEST FROM IPKEY:$key **********************"}
         set count 1
         table add -subtable "FAULT_TBL" $key $count indef $static::holdtime
         if {$DEBUG} {log local0. "************** BEING ADDED TO FAULT TABLE **********************"}
         pool bootcamp_http_pool
         return
       }

      set count [table lookup -subtable "FAULT_TBL" $key]
      incr count

      if {$DEBUG} {log local0. "****** COUNT is $count ***************"}
      if { $count == $static::maxquery } {

            table add -subtable "DELAY_TBL" $key "delayed" indef $static::holdtime
            if {$DEBUG} {log local0. "************** ### ADDED TO DELAY TABLE **********************"}
            after $static::delaytime
            pool bootcamp_http_pool
            clone pool inspection_pool
            return
      }

      if { [table lookup -subtable "FAULT_TBL" $key] != "" } {
         table set -subtable "FAULT_TBL" $key $count
      }

    }
}
```

Sachin used many of the same concepts as Gavin, with the important optimization of the aforementioned static:: commands. He also tied up the clone pool requirement nicely. Overall a very solid rule, and one that's darn near production ready. This was a very, very solid attempt indeed.

3.) Last but not least, our Winner, Chris Miller!



```
when HTTP_REQUEST {
    if { [whereis [IP::client_addr] country] eq "" } {
        set country NA
        #log local0. "Couldn't determine country for [IP::client_addr] so setting country var to NA for not available."
      }
  else {
        set country [whereis [IP::client_addr] country]
        log local0. "Country of origin for [IP::client_addr] is $country"
      }
     if { $country != "US" } {
          if { [table lookup -subtable "blacklist" [IP::client_addr]] != "" } {
                log local0. "Found Client [IP::client_addr] from $country in blacklist table. Delaying 1 second and cloning
traffic"
                log local0. "Client [IP::client_addr] has another [table lifetime -subtable "blacklist" -remaining [
IP::client_addr]] seconds of throttle time remaining."
                clone pool clone_pool
                after 1000
                return
            }
          set reqnum [table incr "reqs:[IP::client_addr]"]
          table set -subtable "reqrate:[IP::client_addr]" $reqnum "reqtable" indefinite 60
          set reqrate [table keys -count -subtable "reqrate:[IP::client_addr]"]
          #log local0. "User [IP::client_addr] made $reqrate requests in the past 60 seconds"
          if { $reqrate > 500 } {
              log local0. "Client [IP::client_addr] from $country made their 501st request which was [HTTP::uri]. Delaying 1
second and cloning traffic"
              table add -subtable "blacklist" [IP::client_addr] "blocked" indef 60
              clone pool clone_pool
              after 1000
              return
                }
         }
}
```

Chris may possibly be the most seasoned iRuler in the bunch, and it showed in his entry. He used tables, the clone pool command, the after command...frankly he did just about everything I'd do. Be that good or bad, I'll leave it up to him to decide. With a few minor tweaks to the logic to save some cycles and doing away with a couple variables that might not be necessary, this would be ready to plug in and live in Prod for quite some time. An excellent, impressive entry from what's sure to be a rock star iRules contributor in the future.

A huge thanks to everyone that participated, and a big pat on the back to all of those that submitted their examples. They were all very strong and judging was tight. Keep your eye out for a special 20LoL in the near future to show my take on the first 3 iRules Challenges and the solutions I'd provide.

For now though, keep iRuling.

#Colin

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