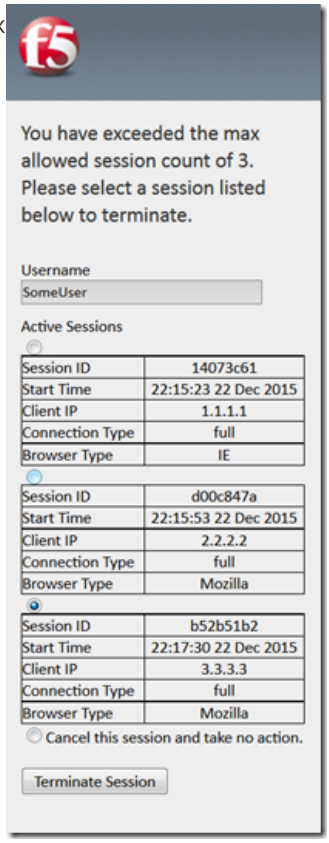
BIG-IP APM: Max Sessions Per User – Enable users to terminate a specified session

Robert Teller, 2015-22-12

Technical Challenge

Recently I was speaking with a customer and they mentioned that they leveraged the "Max Sessions Per User" setting within BIG-IP APM Access Profile to limit the number of concurrent connections that a single user could have at any point in time. The customer mentioned that this works but often their users would complain that the "wrong" session was terminated or that a session they were actively using was closed.



After reproducing the scenario in a lab environment I observed that the BIG-IP APM would terminate sessions based on FIFO (First In, First Out). Meaning that the oldest session was always terminated first regardless of which session the user was actively interacting with. Since this was confusing for the customer I figured others experienced this problem and it would be worth sharing my solution with the world.

So how do you enforce "Max Sessions Per User" and enable your users to intelligently select which session to terminate?

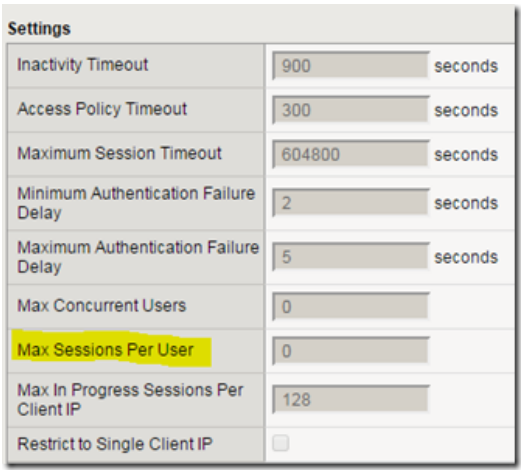
The Solution

If we break down the problem statement above we can see that it it is really two issues. First we need to identify if a user has exceeded the maximum number of allocated sessions, then if they have we need to provide them a way to select which session should be terminated.

Enforce Max Sessions Per User

BIG-IP APM Access Profiles natively provide a way to limit the Maximum number of Sessions a user can establish but it doesn't provide a way to interact with pre-existing sessions.

For more information on Access Profile settings see => https://support.f5.com/kb/en-us/products/big-ip_apm/manuals/product/apm-network-access-11-6-0/9.html#taskid



If the built-in functionality won't achieve what we want lets build our own using APM iRule Events.

iRule Session Enforcement

The ACCESS::uuid iRule function will allow us to identify all active sessions for a specified Access Profile and Username.

The iRule below will prevent a user from establishing more than 3 sessions

```
when CLIENT_ACCEPTED { ACCESS::restrict_irule_events disable }
when ACCESS_POLICY_COMPLETED {
 set max_sessions  3
 set apm_username  [ACCESS::session data get session.logon.last.username]
 set apm_cookie_list  [ ACCESS::uuid getsid "[PROFILE::access name].$apm_username" ]
 log local0. "[PROFILE::access name].$apm_username => session number [llength $apm_cookie_list]"
 if {[llength $apm_cookie_list] >= $max_sessions}
 {
  ACCESS::session remove
  ACCESS::respond 302 location "/vdesk/hangup.php3"
 }
}
```

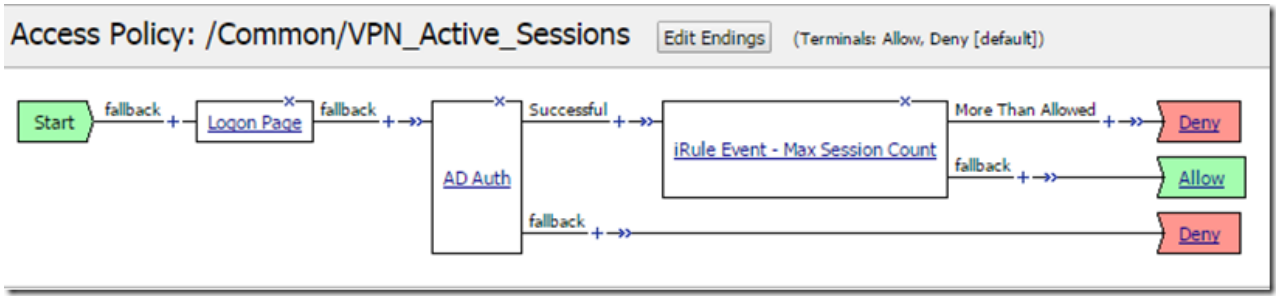
This will allow us to limit concurrent connections for a user but is too late in the authentication process to enable the user to select a session to terminate.

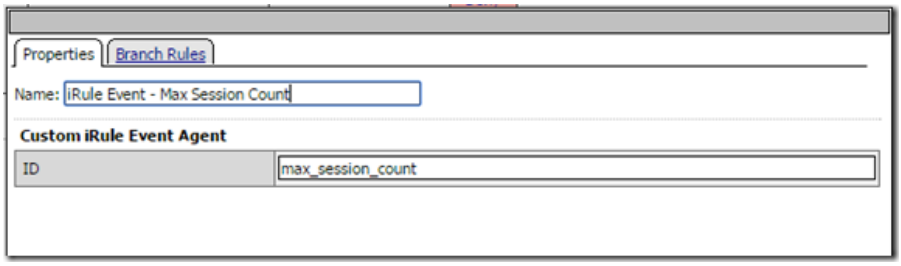
Instead of having the logic execute in the ACCESS_POLICY_COMPLETED section let's try using an VPE iRule event.

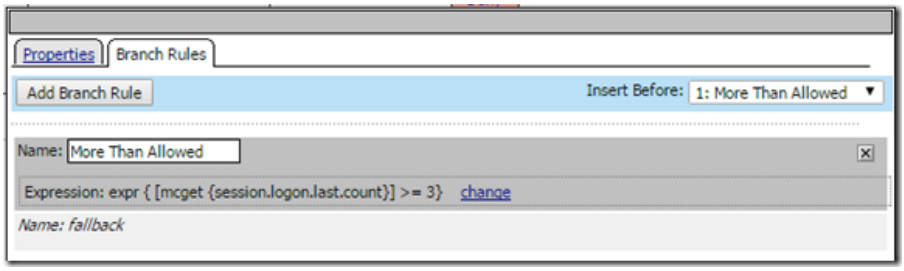
APM VPE iRule Event Session Enforcement

First update your Access Policy to look similar to the images below with an iRule Event placed after the user authentication event.

- The iRule event id will be referenced in your iRule
  - max_session_count
- The branch logic will be used to identify if the user has more than 3 concurrent sessions
  - expr { [mcget {session.logon.last.count}] >= 3}







Update the iRule you created earlier with the code below, this will allow the VPE policy to pause and execute events within the iRule.

```
when ACCESS_POLICY_AGENT_EVENT {
 switch  [ACCESS::policy agent_id] {
  "max_session_count"
  {
   set apm_username [ACCESS::session data get session.logon.last.username]
   set apm_cookie_list [ ACCESS::uuid getsid "[PROFILE::access name].$apm_username" ]

   ACCESS::session data set session.logon.last.count [llength $apm_cookie_list]
  }
 }
}
```

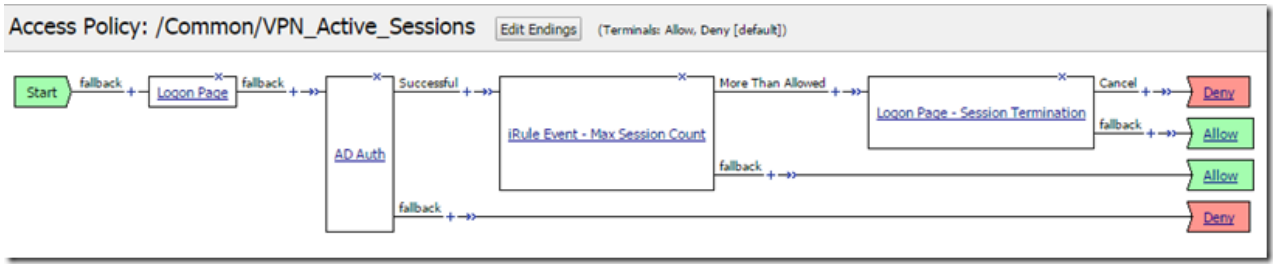
Now as long as the user is below the defined maximum allowed connections they will be allowed to connect as normal.

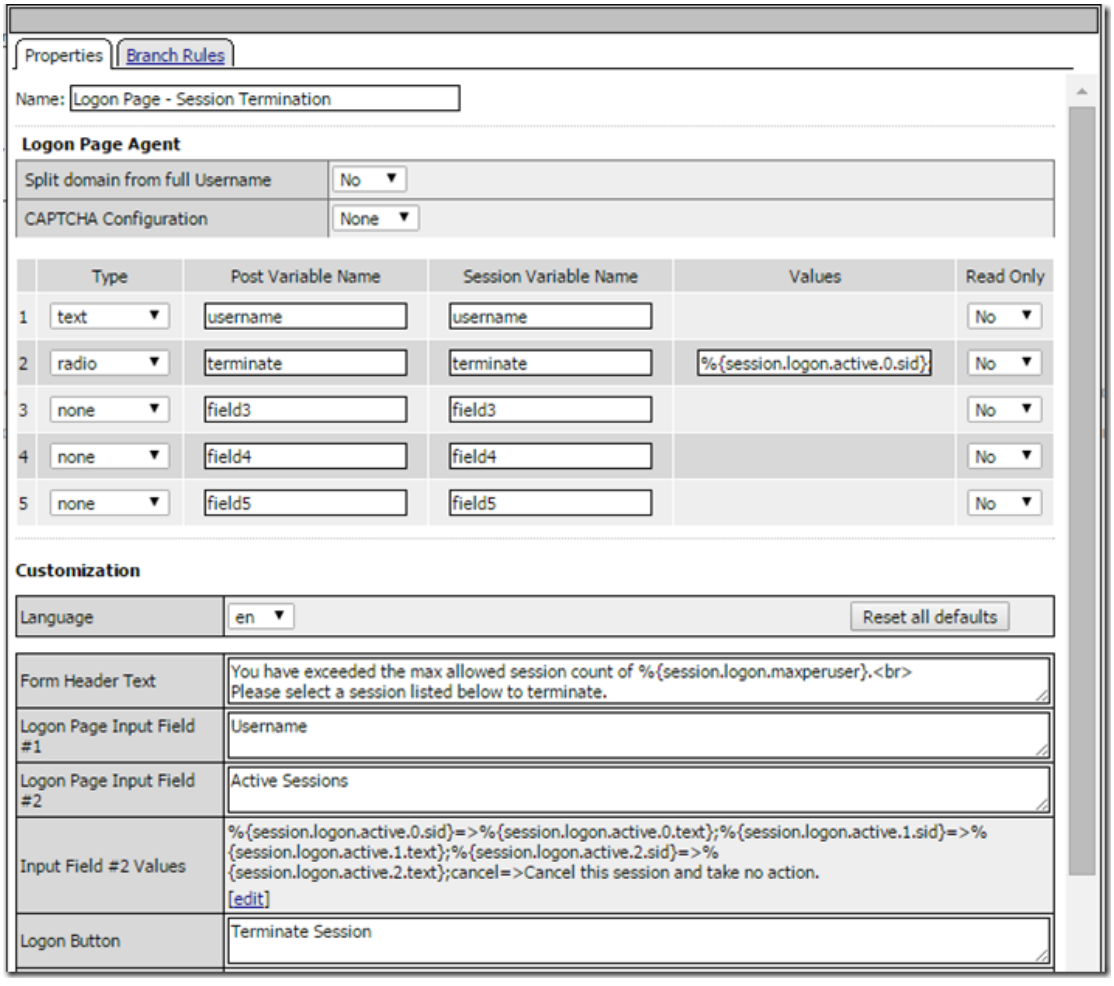
Enabling users to select a Session to be Terminated

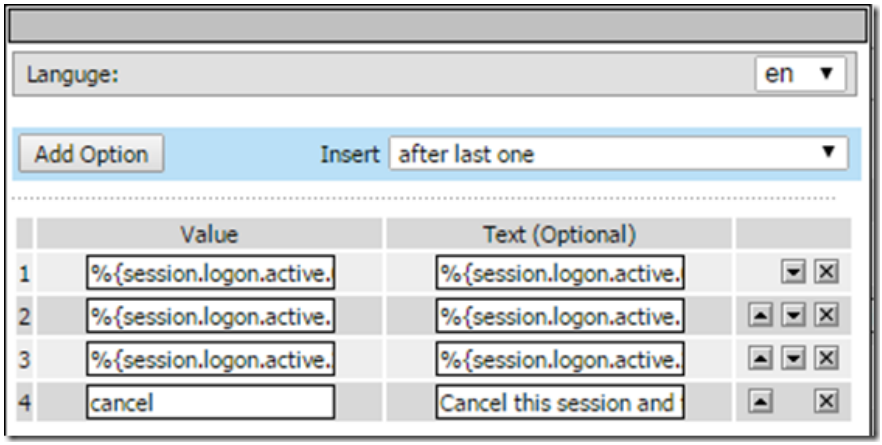
Now this is the tricky part, we need to provide an interface to the user that will enable them to select a session to be terminated. We could spend a bunch of time creating a custom web interface using javascript or we could re-purpose the Logon Page object built into the APM and display the information to the user with minimal customization.

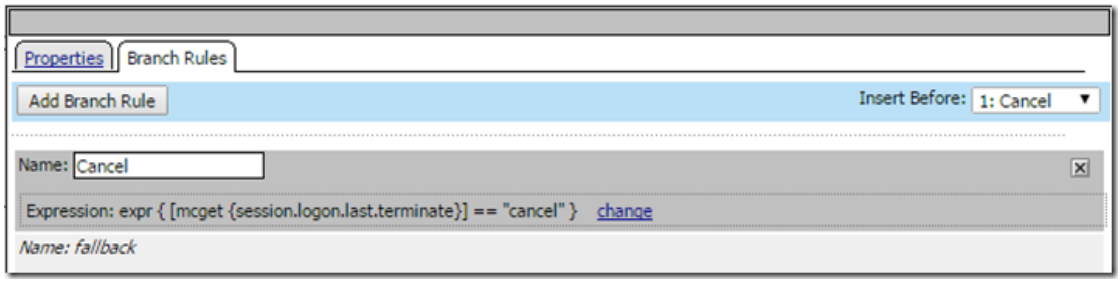
- Remove the Password field from the Logon Page object and replace it with Radio and set the variable name to terminate
  - Click on the textbox in the Values column of the terminate row and add one entry for each session the user is allowed to have. (If the max session count is set to 3 then add 3 options) The contents for the radio buttons will be dynamically generated within the iRule Event and stored as APM Session Variables
    - The Value field should be %{session.logon.active.#.sid} (Session ID's will be stored in a list variable that starts at 0, replace # with the appropriate index number starting at 0)
      - %{session.logon.active.0.sid}

        - %{session.logon.active.1.sid}
        - %{session.logon.active.2.sid}
    - The Text field should be %{session.logon.active.#.text} (The # should be replaced with the corresponding list index id)
        - %{session.logon.active.0.text}
        - %{session.logon.active.1.text}
        - %{session.logon.active.2.text}
  - After adding the appropriate number of options the final option should be cancel with text that will indicate that the current session will be terminated if the user selects cancel
- Click on the Branch Rules tab and add a new Branch Rule to handle logic that will allow the user to cancel Session Termination
  - expr { [mcget {session.logon.last.terminate}] == "cancel" }









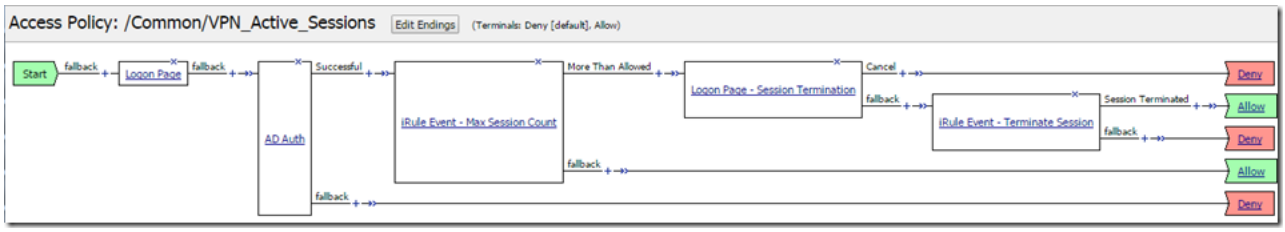
Next update the iRule created earlier with the snippet listed below. The updated iRule will populate the session variables that will be used to display session information to the user.

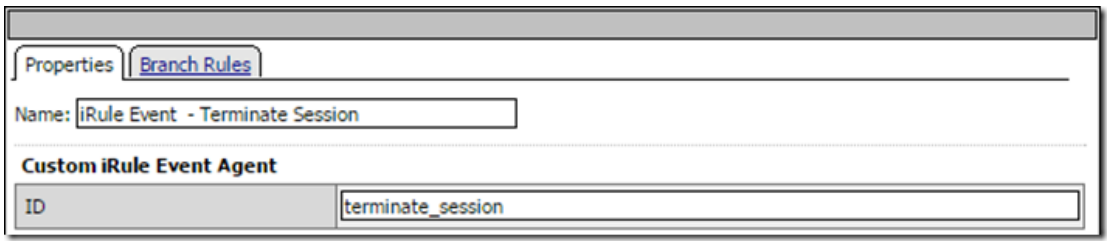
```
when ACCESS_POLICY_AGENT_EVENT {
 switch  [ACCESS::policy agent_id] {
  "max_session_count"
  {
   set apm_username [ACCESS::session data get session.logon.last.username]
   set apm_cookie_list [ ACCESS::uuid getsid "[PROFILE::access name].$apm_username" ]

   for {set i 0} {$i < [llength $apm_cookie_list]} {incr i} {
    set _clientip  [ACCESS::session data get -sid [lindex $apm_cookie_list $i] session.user.clientip]
    set _starttime  [ACCESS::session data get -sid [lindex $apm_cookie_list $i] session.user.starttim
    set _timeformat  [clock format $_starttime -format "%H:%M:%S %d %b %Y "]
    set _connectiontype [ACCESS::session data get -sid [lindex $apm_cookie_list $i] session.user.sess
    set _browsertype [ACCESS::session data get -sid [lindex $apm_cookie_list $i] session.client.type]
    set _sessionid  [ACCESS::session data get -sid [lindex $apm_cookie_list $i] session.user.sessioni

    set _sessioninfo "<table style='border-collapse: collapse' width='100%'>"
    append _sessioninfo "<tr><td style='border: 1px solid black'>Session ID</td><td style='border: 1p
    append _sessioninfo "<tr><td style='border: 1px solid black'>Start Time</td><td style='border: 1p
    append _sessioninfo "<tr><td style='border: 1px solid black'>Client IP</td><td style='border: 1px
    append _sessioninfo "<tr><td style='border: 1px solid black'>Connection Type</td><td style='borde
    append _sessioninfo "<tr><td style='border: 1px solid black'>Browser Type</td><td style='border:
    append _sessioninfo "</table>"

    ACCESS::session data set session.logon.active.$i.sid $_sessionid
    ACCESS::session data set session.logon.active.$i.text $_sessioninfo
   }

   ACCESS::session data set session.logon.last.count [llength $apm_cookie_list]
  }
 }
}
```

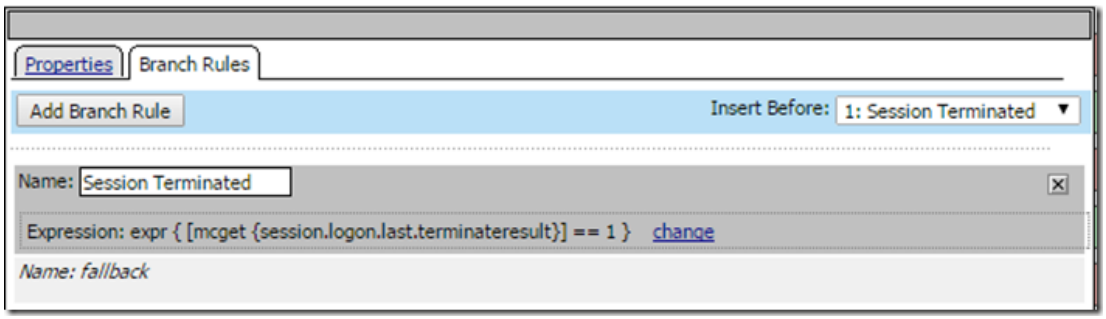
Terminate the selected Session

Now that we have a way to select a session to terminate add a second VPE iRule Event to handle the Session Termination

- The iRule event id will be referenced in your iRule
  - terminate_session
- The branch logic will be used to verify that the session was terminated successfully, if it fails to terminate the users current session will be terminated instead.
  - expr { [mcget {session.logon.last.terminateresult}] == 1 }







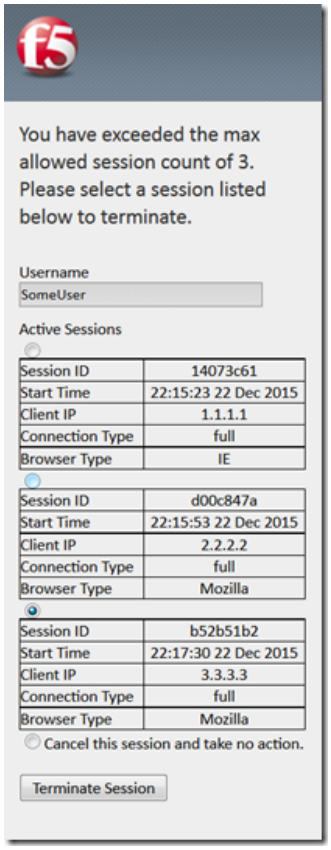
Next update the iRule created earlier with the snippet listed below. The updates will add a second iRule Event that will handle the session termination

```
when ACCESS_POLICY_AGENT_EVENT {
 switch  [ACCESS::policy agent_id] {
  "max_session_count"
  {
   set apm_username [ACCESS::session data get session.logon.last.username]
   set apm_cookie_list [ ACCESS::uuid getsid "[PROFILE::access name].$apm_username" ]

   for {set i 0} {$i < [llength $apm_cookie_list]} {incr i} {
    set _clientip  [ACCESS::session data get -sid [lindex $apm_cookie_list $i] session.user.clientip]
    set _starttime  [ACCESS::session data get -sid [lindex $apm_cookie_list $i] session.user.starttim
    set _timeformat  [clock format $_starttime -format "%H:%M:%S %d %b %Y "]
    set _connectiontype [ACCESS::session data get -sid [lindex $apm_cookie_list $i] session.user.sess
    set _browsertype [ACCESS::session data get -sid [lindex $apm_cookie_list $i] session.client.type]
    set _sessionid  [ACCESS::session data get -sid [lindex $apm_cookie_list $i] session.user.sessioni

    set _sessioninfo "<table style='border-collapse: collapse' width='100%'>"
    append _sessioninfo "<tr><td style='border: 1px solid black'>Session ID</td><td style='border: 1p
    append _sessioninfo "<tr><td style='border: 1px solid black'>Start Time</td><td style='border: 1p
    append _sessioninfo "<tr><td style='border: 1px solid black'>Client IP</td><td style='border: 1px
```

```
    append _sessioninfo "<tr><td style='border: 1px solid black'>Client IP</td><td style='border: 1px
    append _sessioninfo "<tr><td style='border: 1px solid black'>Connection Type</td><td style='borde
    append _sessioninfo "<tr><td style='border: 1px solid black'>Browser Type</td><td style='border:
    append _sessioninfo "</table>"

    ACCESS::session data set session.logon.active.$i.sid $_sessionid
    ACCESS::session data set session.logon.active.$i.text $_sessioninfo
   }

   ACCESS::session data set session.logon.last.count [llength $apm_cookie_list]
  }
  "terminate_session"
  {
   set removed  0
   set apm_username [ACCESS::session data get session.logon.last.username]
   set apm_cookie_list  [ ACCESS::uuid getsid "[PROFILE::access name].$apm_username" ]

   for {set i 0} {$i < [llength $apm_cookie_list]} {incr i} {

    set _terminateid [ACCESS::session data get session.logon.last.terminate]
    set _sessionid  [ACCESS::session data get -sid [lindex $apm_cookie_list $i] session.user.sessioni

    if {$_sessionid eq $_terminateid}
    {
     set removed 1
     ACCESS::session remove -sid [lindex $apm_cookie_list $i]
     break
    }
   }

   ACCESS::session data set session.logon.last.terminateresult $removed
  }
 }
}
```

Time to Test

Now establish enough sessions to exceed the maximum concurrent user count and you should see receive a logon page prompting you to select a session to terminate.



Putting Everything Together

Step 1 – Edit your Access Policy

When this step is complete your Access Policy should look similar to the attached imaged

1. The first iRule Event should have the following information populated
    - The iRule event id will be referenced in your iRule
        - max_session_count
    - The branch logic will be used to identify if the user has more than 3 concurrent sessions
        - expr { [mcget {session.logon.last.count}] >= 3}
2. The “Logon Page – Session Termination” should have the following information populated
    - Remove the Password field from the Logon Page object and replace it with Radio and set the variable name to terminate

        - Click on the textbox in the Values column of the terminate row and add one entry for each session the user is allowed to have. (If the max session count is set to 3 then add 3 options) The contents for the radio buttons will be dynamically generated within the iRule Event and stored as APM Session Variables
            - The Value field should be %{session.logon.active.#.sid} (Session ID’s will be stored in a list variable that starts at 0, replace # with the appropriate index number starting at 0)
                - %{session.logon.active.0.sid}
                - %{session.logon.active.1.sid}
                - %{session.logon.active.2.sid}
            - The Text field should be %{session.logon.active.#.text} (The # should be replaced with the corresponding list index id)
                - %{session.logon.active.0.text}
                - %{session.logon.active.1.text}

        - %{session.logon.active.1.text}
        - %{session.logon.active.2.text}
    - After adding the appropriate number of options the final option should be cancel with text that will indicate that the current session will be terminated if the user selects cancel
    - Click on the Branch Rules tab and add a new Branch Rule to handle logic that will allow the user to cancel Session Termination
        - expr { [mcget {session.logon.last.terminate}] == "cancel" }
3. The second iRule Event should have the following information populated
    - The iRule event id will be referenced in your iRule
        - terminate_session
    - The branch logic will be used to verify that the session was terminated successfully, if it fails to terminate the users current session will be terminated instead.
        - expr { [mcget {session.logon.last.terminateresult}] == 1 }

Step 2 – Create and Apply the Custom iRule

```
when ACCESS_POLICY_AGENT_EVENT {
 switch  [ACCESS::policy agent_id] {
  "max_session_count"
  {
   set apm_username [ACCESS::session data get session.logon.last.username]
   set apm_cookie_list [ ACCESS::uuid getsid "[PROFILE::access name].$apm_username" ]

   for {set i 0} {$i < [llength $apm_cookie_list]} {incr i} {
    set _clientip  [ACCESS::session data get -sid [lindex $apm_cookie_list $i] session.user.clientip]
    set _starttime  [ACCESS::session data get -sid [lindex $apm_cookie_list $i] session.user.starttim
    set _timeformat  [clock format $_starttime -format "%H:%M:%S %d %b %Y "]
    set _connectiontype [ACCESS::session data get -sid [lindex $apm_cookie_list $i] session.user.sess
    set _browsertype [ACCESS::session data get -sid [lindex $apm_cookie_list $i] session.client.type]
    set _sessionid  [ACCESS::session data get -sid [lindex $apm_cookie_list $i] session.user.sessioni

    set _sessioninfo "<table style='border-collapse: collapse' width='100%'>"
    append _sessioninfo "<tr><td style='border: 1px solid black'>Session ID</td><td style='border: 1p
    append _sessioninfo "<tr><td style='border: 1px solid black'>Start Time</td><td style='border: 1p
    append _sessioninfo "<tr><td style='border: 1px solid black'>Client IP</td><td style='border: 1px
    append _sessioninfo "<tr><td style='border: 1px solid black'>Connection Type</td><td style='borde
    append _sessioninfo "<tr><td style='border: 1px solid black'>Browser Type</td><td style='border:
    append _sessioninfo "</table>"

    ACCESS::session data set session.logon.active.$i.sid $_sessionid
    ACCESS::session data set session.logon.active.$i.text $_sessioninfo
   }

   ACCESS::session data set session.logon.last.count [llength $apm_cookie_list]
  }
  "terminate_session"
  {
   set removed  0
   set apm_username [ACCESS::session data get session.logon.last.username]
   set apm_cookie_list  [ ACCESS::uuid getsid "[PROFILE::access name].$apm_username" ]

   for {set i 0} {$i < [llength $apm_cookie_list]} {incr i} {

    set _terminateid [ACCESS::session data get session.logon.last.terminate]
    set _sessionid  [ACCESS::session data get -sid [lindex $apm_cookie_list $i] session.user.sessioni

    if {$ sessionid eq $ terminateid}
```

```
    {
     set removed 1
     ACCESS::session remove -sid [lindex $apm_cookie_list $i]
     break
    }
   }

   ACCESS::session data set session.logon.last.terminateresult $removed
  }
 }
}
```

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