Intelligent Cellular Access Tech Switching

6310-DX

Intelligent Cellular Access Tech Switching

Design

Our 63xx-series line of routers provide a configuration settings called *Access technology*, which can be used to set a cellular modem to connect on 4G-only, 3G-only, 2G-only, or all networks.

https://kb.accelerated.com/m/67492/l/819886-lte-troubleshooting-tree#yes_1

However, some roaming sites or locations with intermittent LTE connectivity have run into issues where the modem gets stuck on a bad radio access technology (rat) and won't bump to a different rat unless you set the modem to 3G-only or 4G-only.

This is a smarter process for setting a modem to 3G-only or 4G-only. Typically, doing so would lock the modem to only connect on that particular radio access technology (rat). This could be bad for sites with intermittent coverage of a particular CNTI or rat, causing the site to lose connectivity until that particular network is available again.

What this script does is it will set the device to 4G-only; if we get a connection, then life is good. If we don't connect within 10 minutes (adjustable), then switch to 3G-only. If we connect on 3G, then stick with that until this script gets executed again. If we still can't connect after 10 minutes, then switch down to 2G. If we still cannot connect after 10 minutes, try all-technologies and reset the modem.

Config Setup

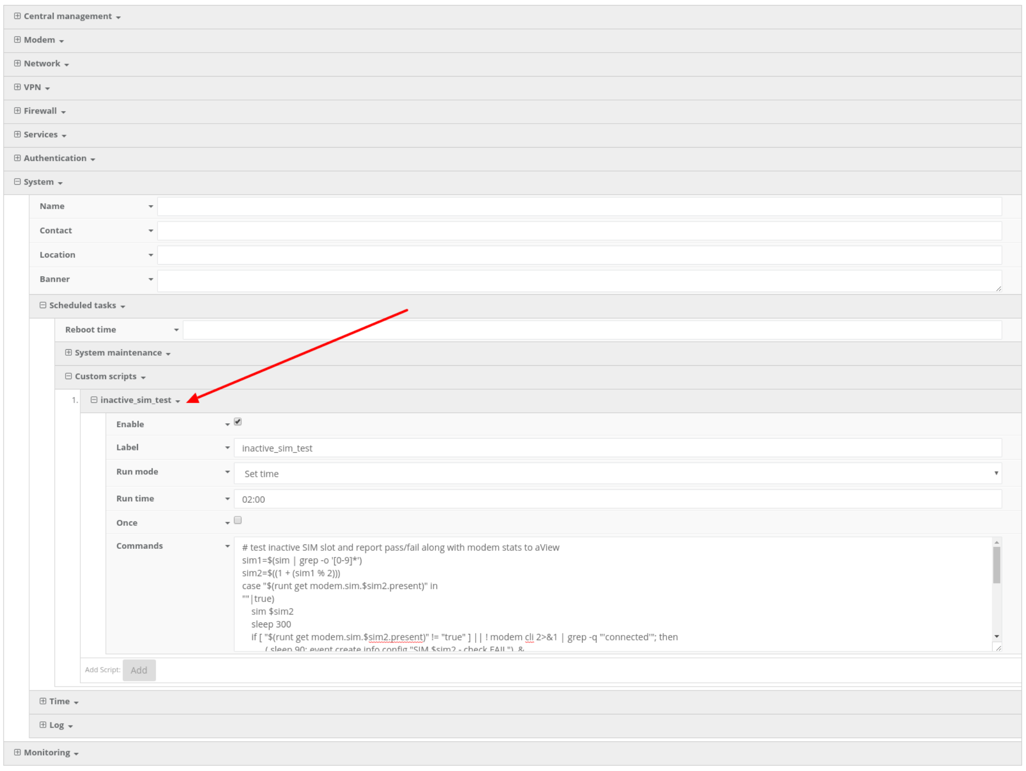
Minimum firmware: 18.4.54+

Create a new custom script under ***System -> Scheduled tasks -> custom scripts***, and enter in the following.   Adjust the ***Run time*** to the desired time of day you would like to test the inactive SIM. The suggestion is to run this script once per day during off hours to minimize customer impact/connectivity.

```
logexit() {
  echo "custom: cellular $1"
  accns_log w config "custom: cellular $1"
  exit
}



modem_index() {
  idx=$(modem idx)
  if ! [ "$idx" ]; then
    sleep 30
    idx=$(modem idx)
    [ "$idx" ] || logexit "modem not present"
  fi
  echo "$idx"
}
```

```
modem_is_online() {
  # wait up to two minutes for the modem to get a cellular connection
  i=0
  ret=0
  cellular_connection=0
  while [ "$i" -lt 5 ]; do
    if [ "$(runt get mm.modem.$(modem_index).status.state)" = 'connected' ]; then
      cellular_connection=1
      break
    fi
    sleep 30
    i=$((i+1))
  done
  if [ "$cellular_connection" = 0 ]; then
    ret=1
  fi
  return $ret
}


connect_to_rat() {
  [ "$1" ] || return 1
  sleep "$wait_time"
  if [ "$(runt get mm.modem.$(modem_index).status.gtech | tr '[a-z]' '[A-Z]')" = "$1" ]
&& modem_is_online; then
    logexit "$1 connection active"
  else
    return 1
  fi
}




wait_time='600' # 10 minutes


connect_to_rat '4G' || config set network.modem.modem.access_tech '4G'
connect_to_rat '4G' || config set network.modem.modem.access_tech '3G'
connect_to_rat '3G' || config set network.modem.modem.access_tech '2G'
connect_to_rat '2G' || config set network.modem.modem.access_tech 'all'


modem_is_online && logexit 'connection active after setting access_tech to ALL'
```

```
modem reset
logexit 'rat script failed to establish connection, resetting modem'
```



Name	
Contact	
Location	
Banner	

Reboot time	

Enable	✓
Label	inactive_sim_test
Run mode	Set time
Run time	02:00
Once	☐

Commands

```
# test inactive SIM slot and report pass/fail along with modem stats to aView
sim1=$(sim | grep -o '[0-9]*')
sim2=$((1 + (sim1 % 2)))
case "$(runt get modem.sim.$sim2.present)" in
""|true)
  sim $sim2
  sleep 300
  if [ "$(runt get modem.sim.$sim2.present)" != "true" ] || ! modem cli 2>&1 | grep -q '"connected"'; then
    ( sleep 90; event create info config "SIM $sim2 - check FAIL") &
```