WAN Connectivity Test and Failover

6310-DX

WAN Connectivity Test and Failover

Background

Our 63xx-series line of routers utilize the *Active Recovery* feature to perform connectivity tests on live Internet connections, both wired and cellular, to determine failover conditions. However, for wired WAN connections, these tests only restart the interface or reboot the device, instead of performing a true failover. This is problematic for WAN connections where the upstream connection is down (e.g. the coax of the WAN modem gets cut).

This script, when setup in the configuration profile as shown below, will perform a connectivity test once every minute on the WAN interface. If those connectivity tests fail four times in a row, then the script will automatically failover to the cellular modem inside the 63xx-series router, and start its connection process over again.

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 ***Interval*** to the desired interval you would like this script to run.

```
# Test the WAN connection with a ping test.  Keep a count of failed tests.
# If this test fails for three times in a row, then bring down the WAN link.

# Adjustable settings
test_server='128.136.167.120'  # where we perform the ping tests to
fail_count_file='/tmp/custom_wan_test_fail_count.txt'
fail_count_limit='3'  # number of concurrent failures that can occur

test_failed() {
  try=$((try+1))
  echo "$try" > "$fail_count_file"
  accns_log w config "custom: wan test failed ($1 - try $try)"
}

test_passed() {
  rm -f "$fail_count_file"
  config set network.interface.wan.ipv4.metric 1 # raise WAN priority
  try=0
  # Note: uncomment the following line if you want to log successful tests
  accns_log w config "custom: wan test passed"
}

ping_test() {
  # 3-attempts, 5-second timeout, 1-byte packet size
  /etc/netmon/netmon_test.sh wan "$(runt get network.mark.ipv4.interface_wan)" 3 5 ipv4
```

```
ping "$test_server" 1
}

### MAIN ###
try=$(cat "$fail_count_file" 2> /dev/null)
try=${try:-0}

# make sure $try is an integer. set to zero if not
case $try in
  ''|*[!0-9]*)
    try=0
    ;;
esac

# do the connectivity test
if ! ip link show dev wan | grep -q 'state UP'; then
  ip link set dev wan up
  sleep 5
fi

if ! ip route | grep -q "dev wan"; then
  test_failed "No WAN connection"
elif ping_test; then
  test_passed
else
  test_failed "ping failure to IP $test_server"
fi

# lower WAN priority if failed test count is greater than specified limit
if [ "$try" -ge "$fail_count_limit" ]; then
  # note, that we don't reset the fail count. If we fail next attempt, lower the WAN
metric again.
  config set network.interface.wan.ipv4.metric 5 # modem metric is typically 3, but put
it really low just in case
fi
```