motionEyeOS mobile phone presence detection for auto on/off based on network IP or MAC address

---

I wanted my motionEyeOS system to turn on and off automatically if our mobile phones are present on the home network.

I recommend having a router which you can SSH into for this purpose, but in theory, the script could work without that. This script can run as a cron job; for that, run `crontab -e` on the motionEyeOS[^1] Raspberry.

bash

```
#!/bin/bash

declare -a debugmsgs

SSHCMD="/usr/bin/ssh -i /data/etc/ssh_key"
ROUTER=$(/sbin/route -n | awk '/^0.0.0.0/ { print $2}')
ROUTERUSER="root"
LOGFILE="/data/log/status.log"
LASTDETECTED="/data/log/lastdetected.log"

MAILHOST="a.mail.capable.server.which.you.can.log.into"
MAILHOSTUSER="the.mail.host.user"

MOTION_KEY="@motion_detection"
MOTION_CONFIG="/data/etc/thread-1.conf"

GRACETIME=230

# this is inverse: off means device detected and home and
motion detection
# should be off
```

```
DETECTED="on"
DETECTEDIP=""

declare -a DEVICES
DEVICES=( "00:11:22:33:44:55" )
declare -A IPS


function init() {
    if [ ! -f "${LOGFILE}" ]; then
        touch "${LOGFILE}"
    fi
}

function debug() {
    local msg="DEBUG $(date -Iseconds) $1"
    #debugmsgs+=("$msg")
    echo "$msg"
}

function get_motion() {
    local current="$(cat "${MOTION_CONFIG}" | grep "$
{MOTION_KEY}" | awk '{print $NF}')"
    # unpassed to-be-set value handling
    if [ "${current}" != "on" ] && [ "${current}" != "off" ];
then
        current="off"
    fi

    echo "${current}"
}

function set_motion() {
    local set=$1

    # unpassed to-be-set value handling
    if [ "${set}" != "on" ] && [ "${set}" != "off" ]; then
        set="off"
    fi

    /etc/init.d/S85motioneye stop
```

```
    sed -i "s/\(#\s*$MOTION_KEY\s\s*\).*/\1$set/"
$MOTION_CONFIG
    /etc/init.d/S85motioneye start

    ${SSHCMD} ${MAILHOSTUSER}@${MAILHOST} -- "echo \"$(date -
Iseconds) $DETECTED $DETECTEDIP\" | mail -s \"motion status
change: ${set}\" -a\"Date:$(date -R)\" alerts@petermolnar.eu"
}

function collect_ips() {
    for dev in "${DEVICES[@]}"; do
        local ip="$(${SSHCMD} ${ROUTERUSER}@${ROUTER} -- "cat /
tmp/dnsmasq.leases | awk '/$dev/ {print \$3}'" 2>/dev/null)"
        IPS[$ip]=1
    done
}

function ping_ips() {
    for k in "${!IPS[@]}"; do
        ping -c3 -w3 -W1 "${k}" >& /dev/null
    done
}

function test_ips() {
    for k in "${!IPS[@]}"; do
        if ping -c1 -w1 -W1 "${k}" >& /dev/null; then
            # turning off motion
            DETECTEDIP="${k}"
            DETECTED="off"
            echo "$(date +%s) $DETECTEDIP" > "${LASTDETECTED}"
            break
        fi
    done
}

function test_lastrun() {
    # return 0: change status
    # return 1: don't change status

    # logline should be: "epoch ISO-8601 DETECTED IP"
    local lastrun="$(tail -n1 "${LOGFILE}")"
```

```
    #debug "lastrun: $lastrun"

    # no need to do anything in case the previous detection
status is the same
    # as now
    local laststatus="$(echo "${lastrun}" | cut -d" " -f3)"
    if [ "${laststatus}" == "${DETECTED}" ]; then
        return 1
    fi

    # no need for restarts if the current value is the same
    local current="$(get_motion)"
    if [ "${current}" == "${DETECTED}" ]; then
        return 1
    fi

    # check previous timestamp to prevent flapping
    local lasttime="$(cat "${LASTDETECTED}" | cut -d" " -f1)"
    #local lasttime="$(echo "${lastrun}" | cut -d" " -f1)"
    if [ -z ${lasttime} ]; then
        lasttime=0
    fi

    local minepoch=$(($(date +%s)-${GRACETIME}))
    if [ ${lasttime} -gt ${minepoch} ]; then
        #debug "timeout not reached"
        return 1
    fi

    debug "all clear, change status"
    # detect status is different and the last entry is
$GRACETIME+ old: act
    echo "$(date +%s) $(date -Iseconds) $DETECTED $DETECTEDIP"
>> "${LOGFILE}"

    return 0
}


init
collect_ips
```

```
ping_ips
test_ips

if test_lastrun; then
    set_motion "${DETECTED}"
fi
```

Links

1. https://github.com/ccrisan/motioneyeos