

Raspberry Pi Intelligent Surveillance

Si vuole installare un sistema di sorveglianza domestico che si attivi soltanto in caso di nostra assenza, evitando quindi inutili falsi allarme generati da noi stessi.

DISCLAIMER: Questa guida è a solo scopo didattico, non mi assumo alcuna responsabilità sull'uso che ne farete.

Trattandosi di un progetto ancora sperimentale, non è garantita una affidabilità totale.

Configurazione

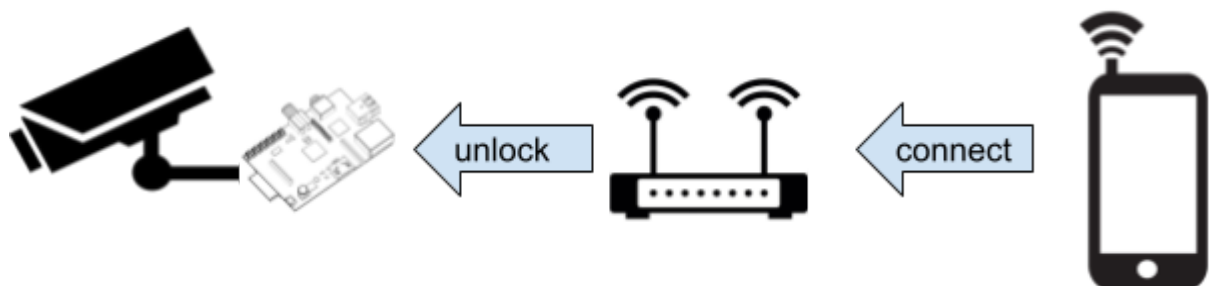
Per raggiungere tale scopo e ottimizzare i risparmi energetici, farò uso di un **Roseapple Pi** che funziona dignitosamente, fino a 2 punti camera.

Ad ogni modo, un Raspberry Pi 2 o successivi, dovrebbero adempiere senza problemi allo scopo.

L'idea è quella di sorvegliare una o più stanze in maniera del tutto automatica, senza doverci preoccupare di attivare e disattivare la rilevazione movimento quando usciamo e quando rientriamo. adotteremo alcune precauzioni (in seguito) per ridurre al minimo i falsi allarme.

Per il raggiungimento di tale scopo, faremo uso di un cellulare (il nostro) e della rete locale a cui si aggancia. è generalmente una buona idea usare il bluetooth, piuttosto che la rete locale per tracciare la nostra presenza nei paraggi, tuttavia non ho un adattatore bluetooth a portata di mano, e ho pensato agli utenti che (come me) lo disattivano per risparmiare energia.

L'utilizzo della rete comporta qualche svantaggio (vedi: standby del wifi su android) a cui porremo rimedio con qualche "stratagemma"



Occorrente

- 1 Raspberry Pi 2 (o equivalente piattaforma linux)
- Una IP Camera e/o una webcam
- Configurazione dei software (motion, script alarm.sh)

Motion detection

Naturalmente non abbiamo alcun interesse a registrare ore di vuoto.

Per questo useremo **motion** un software Open Source nato proprio allo scopo di registrare brevi video (e screenshot) solo in presenza di movimento.

1) Installiamo motion:

```
apt-get update && apt-get install libav-tools motion
```

2) Post installazione:

```
a) touch /var/run/motion/motion.pid
```

```
b) chmod 777 !$
```

```
c) cp /etc/motion/motion.conf /etc/motion/motion.conf.bak
```

```
d) mkdir /mnt/sorveglianza/motion
```

*assumendo che in /mnt/sorveglianza sia montato un dispositivo di archiviazione di massa

3) Configurazione di motion:

```
vim /etc/motion/motion.conf
```

e modifichiamo alcuni dei parametri principali:

```
daemon on
```

```
output_pictures best
```

```
process_id_file /var/run/motion/motion.pid
```

```
width 640
```

```
height 480      (in base alla vostra webcam/telecamera IP)
```

```
framerate 2
```

```
target_dir /mnt/sorveglianza/motion
```

4) Configurazione della/delle telecamere:

Nel caso di una sola telecamera, sarebbe sufficiente inserire l'apposita riga nel file di configurazione motion.conf

Mentre, per più punti camera, è necessario creare dei file (thread1.conf, thread2.conf) in cui inserire per ciascuno i parametri di configurazione di ogni telecamera in uso

NB: anche in presenza di una sola telecamera, possiamo comunque usare i file threadX.conf, e così faremo.

nel file **motion.conf** decommentiamo le righe

```
thread /etc/motion/thread1.conf
```

```
thread /etc/motion/thread2.conf
```

per attivare due punti camera (o soltanto la prima riga per un solo punto camera)

i file threadX.conf sono molto semplici:

Per una webcam usb:

```
videodevice /dev/video0  
text_left USBWebcam-1  
target_dir /mnt/sorveglianza/motion  
stream_port 8081
```

Per una IP Camera:

```
text_left Foscam IP  
netcam_url value http://192.168.1.121/videostream.cgi?user=admin&pwd=password  
target_dir /mnt/sorveglianza/motion  
stream_port 8082
```

NB: Il parametro `stream_port` definisce su quale porta, viene visualizzato lo streaming in tempo reale della camera in questione. Di default, lo stream è abilitato alla visualizzazione solo da localhost (quindi inaccessibile da un pc). Per abilitarlo, è necessario modificare questa voce all'interno di **motion.conf**:

stream_localhost on → stream_localhost off

inoltre, più in basso nel file di configurazione, c'è la possibilità di impostare una password per proteggere il flusso video in streaming, altrimenti non protetto.

stream_auth_method 2

è in genere una buona scelta.

NB2: Ogni IP camera ha un indirizzo diverso per lo streaming, a seconda del produttore. in questo caso, google è vostro amico.

A questo, dovremmo avere motion funzionante e già in grado di catturare schermate e video in presenza di movimento. può essere necessario un pò di fine tuning all'interno del file di configurazione per evitare eccessivi falsi allarmi, a questo scopo vi allegherò il mio file di configurazione per avere un confronto.

5) Un pò di scripting (1):

Naturalmente non vogliamo ricevere allarmi e registrare video di noi che gironzoliamo per la stanza.

Arriva il momento di scrivere uno script che rilevi periodicamente la nostra presenza e si comporti di conseguenza abilitando o disabilitando il motion detection

```
#!/bin/bash

#variables initialization here
macaddress1="aa:bb:d0:ff:2a:e2"
macaddress2="zz:cc:e6:a0:a7:c0"
scriptname=$(basename $BASH_SOURCE)
cycle=0

function check {
for i in {1..$2}
do
    if [ "$(arp-scan 192.168.1.0/24|awk {'print $2'}|tail -n+3|head -n -3 |sort|uniq|grep -i $1)" = $1 ]
    then
        echo 1
        break
    else
        echo 0
    fi
sleep 10
done
exit
}

for (( ; ))
do
    echo $scriptname
    echo "EXEC N. $cycle"
    let "cycle++"

    if [ "$(check $macaddress1 5)" = 1 ]
    then
        logger "$scriptname : disable trigger"
        lwp-request http://localhost:8080/0/detection/pause
    else
        if [ "$(check $macaddress2 1)" != 1 ]
        then
            logger "$scriptname : enable trigger"
            lwp-request http://localhost:8080/0/detection/start
        else
            echo "phone needed in order to unlock"
            #logger "$scriptname : phone needed in order to unlock"
        fi
    fi

sleep 300
done
```

Lo script in questione controlla periodicamente (300s = 5 minuti) la presenza del mac address del nostro cellulare all'interno della rete e sfrutta le API di motion per attivare o disattivare la rilevazione del movimento.

è risaputo che molti cellulari android, per risparmiare energia, disattivano periodicamente il wifi durante lo standby, perciò ho implementato una funzione che controlli più di un paio di volte a distanza di alcuni secondi.

è anche possibile che sia in casa, lavorando al computer, e il mio telefono sia spento.

Ragion per cui, in caso di mancato appello del telefono, viene controllata la presenza del mac address (riportato fittizio in questo documento) del mio computer.

Notare che la presenza del mac address del computer in rete, non provoca la disattivazione della sorveglianza, ma la sua assenza può provocarne l'attivazione nel caso manchi anche il cellulare.

Possiamo scrivere la regola come segue:

CELLULARE	COMPUTER	Disabilita sorveglianza
CELLULARE	COMPUTER	Disabilita sorveglianza
CELLULARE	COMPUTER	Non disabilita sorveglianza
CELLULARE	COMPUTER	Abilita sorveglianza

In questa configurazione, c'è un'unica possibilità in cui la sorveglianza possa rimanere disattivata in mia assenza:

Se esco di casa lasciando aperto il computer

Che può sembrare poco saggio, e infatti è alquanto improbabile nel mio caso.

CONSIDERAZIONI FINALI

Ho scritto questo script su misura che combacia le **mie** abitudini, ma sono fiducioso che possiate trarne vantaggio anche voi, magari *rimuovendo* il controllo sul secondo mac address se questo comportamento non corrisponde alle vostre abitudini (es: lasciate sempre il pc acceso e aperto uscendo di casa **(e non siete affatto green!)**)

o preferite intensificare le ricerche del mac address del vostro telefono soltanto.

Se poi disponete di un dongle bluetooth, si tratta di modificare solo un paio di righe per poter eseguire il controllo sul **bluetooth**. che volete più di così? ;-)

LINKS

Installing motion and basic configuration:

<https://www.linux.com/learn/tutorials/780055-how-to-operate-linux-spycams-with-motion>

Multi camera configuration with threads:

<http://ubuntuforums.org/showthread.php?t=1897786>

Remote control via APIs:

<http://www.lavrsen.dk/foswiki/bin/view/Motion/RemoteControlHttp>

UPDATE:

Mi sono procurato una chiavetta **Bluetooth**.

Allego script per effettuare il polling sul dispositivo Bluetooth di nostra scelta

Potete programmare l'esecuzione di questo script ogni 1, 2, o 5 minuti da crontab

```
#!/bin/bash
```

```
##### READ ME FIRST!! #####
```

```
# In order to correctly ping your device, you need to pair it first.
```

```
# 1) bluetoothctl -a #opens a shell and register the agent
```

```
# 2) agent on
```

```
# 3) default-agent
```

```
# 4) discoverable yes
```

```
# 5) on your mobile device, you should now see the bluetooth to pair
```

```
# 6) enter a pincode, which will be asked on the shell side as well.
```

```
# 7) you're done! now you can try to ping your device with l2ping $mac_address
```

```
#####
```

```
# Simple function to re-issue the command if an error exit code is caught.
```

```
# Usefull for buggy bluetooth adapters.
```

```
unfreeze (){
```

```
    "$@"
```

```
    local cmdstatus=$?
```

```
    if [ $cmdstatus -ne 0 ]; then
```

```
        echo "an error occurred. re-trying in 10 seconds..." && sleep 10
```

```
    "$@"
```

```
    fi
```

```
}
```

```
# Scan function to be called
```

```
scan (){
```

```
l2ping $MACADDR -c 1 2>/dev/null
```

```
local cmdstatus=$?
```

```
echo $cmdstatus
```

```
return $cmdstatus
```

```
}
```

```
#Variables initialization
```

```
exitcode=99
```

```
count=0
```

```
#MACADDR="34:FC:EF:0B:9E:22"
```

```
MACADDR="C4:43:8F:A0:2B:9D"
```

```
# uncomment the following if your adapter is really buggy
```

```
echo "restarting bluetooth interface..."
```

```
#service bluetooth restart && sleep 5
```

```
unfreeze hciconfig hci0 up && sleep 5
```

```
#unfreeze hciconfig hci0 iscan
```

```
while [[ $exitcode != 0 && count -lt 3 ]]
```

```
do
```

```
    echo "iteration n. $count"
```

```
    exitcode=$(scan)
```

```
    (( count++ ))
```

```
done
```

```
echo "the exit code is: $exitcode"
```

```

#READ CURRENT CAMERA STATUS
laststatus=$(curl http://localhost:8080/1/detection/status -u user:password|awk '{print $5}')
echo "last status: $laststatus"

echo "new status: "

if [[ $exitcode == 0 && $laststatus != "PAUSE" ]]; then
    curl http://localhost:8080/1/detection/pause -u user:password
    echo "detection paused"
else if [[ $exitcode == 1 && $laststatus == "PAUSE" ]]; then
    curl http://localhost:8080/1/detection/start -u user:password
    echo "detection resumed"
else
    echo "unchanged"
fi
fi

#uncomment this if your adapter can't handle long uptimes
#unfreeze hciconfig hci0 down && sleep 5

```

UPDATE 2:

è anche possibile abilitare e disabilitare la sorveglianza da remoto con **telegram-cli** ed un [semplice script](#):

e successivamente modificare motion.conf come segue:

```
on_picture_save echo "send_photo Morrolinux %f" |nc -U /tmp/tg.sock
```