RaspberryPi - Tutorial für einen DIVERA 24/7 Wachenmonitor

Quelle und Skripte

Dieses Tutorial wurde uns freundlicherweise von Dustin1358 zur Verfügung gestellt. Die hier verwendeten Sources und Skripte sollten von GitHub geladen werden, da sie hier nicht fortlaufend von der Original-Quelle aktualisiert werden:

https://github.com/Dustin1358/Raspberry-Pi-Divera-Monitor

- DIVERA 24/7 Wachenmonitor mit Raspberry Pi
  - Monitor und Bildschirm Befehle hinzufügen
  - Automatisches Ein-/Ausschalten über Zeitplan od. Bewegungsmelder: (Optional)
    - Wachenmonitor ohne Bewegungsmelder
    - Wachenmonitor mit Bewegungsmelder
  - Optionale Einstellungen
    - Autostart einrichten (Optional)
    - Bildschirmschoner (Optional)
  - Problembehandlung

DIVERA 24/7 Wachenmonitor mit Raspberry Pi

In diesem Tutorial wird auf einem Raspberry Pi eine autommatische Steuerung des Divera Monitors eingerichtet. Der Raspberry schaltet bei Alarmen sowie zu Dienstzeiten den Monitor an und im Anschluss wieder aus. Hier werden 2 Versionen präsentiert. Die erste Version schaltet bei Alarmen und Dienstzeiten den Monitor an bzw. wieder aus. Die zweite Version integriert außerdem einen Bewegungsmelder. Hierbei wird zu Dienstzeiten, wenn sich niemand vor dem Monitor befindet, der Bildschirmschoner eingeschaltet. Außerhalb der Dienstzeiten wird bei der Detektion von Bewegung der Monitor eingeschaltet.

Das Folgende ist eine Schritt für Schritt Anleitung, die vom Kauf des Raspberrys bis zum fertigen Monitor geht.

Dieses Tutorial basiert auf den aktuellen Raspberry 3 B+. Falls noch keiner gekauft wurde, kann bspw. von ABOX ein Komplettpacket mit Ladegerät, SD-Karte und weiterem Zubehör gekauft werden. Als Betriebssystem Image wurde *Raspbian Stretch with desktop* verwendet.

```
sudo apt install jq unclutter cec-utils xscreensaver
```

q wird dabei für die Bash Variante des Skripts benötigt. Mittels unclutter wird der Mauszeiger ausgeblendet. Cec-utils wird verwendet um einige Fernsehr ein bzw. aus zu schalten. Xscreensaver wird für die Version mit Bewegungsmelder benötigt.

Monitor und Bildschirm Befehle hinzufügen

Als nächstes werden bash-Befehle die den Fernseher bzw. den Monitor an/aus schalten können hinzugefügt. Im Terminal wird nun der Editor *nano* verwendet um eine neue Datei mit folgendem Befehl anzulegen:

```
nano .divera_commands.sh
```

In diese Datei kommt der Inhalt der folgenden Datei (am besten copy-paste nutzen und nicht abtippen):

.divera_commands.sh

```
#!/bin/bash
# This file is an extension for the bash which contains the commands for
# turning the screen ond and off and also starts and stops the monitor.
# Source this file in your bash.rc to have the commands in your terminal.


#The url of your monitor (with Acesskey for autologin).
MONITOR="URL hier"

export DISPLAY=:0
```

```
# This function starts or stops the divera monitor.
function monitor(){
    if [ $1 = on ]; then
        # starts chromium in in kiosk mode
        chromium-browser --noerrdialogs --kiosk --incognito $MONITOR &>
/dev/null &
    elif [ $1 = off ]; then
        # just kill every chromium process
        pkill chromium >/dev/null
    else
        echo Unknown parameter
    fi
}


# Turns the screen on and off. Every screen is diffrent and many are bad
 programmed
# for that here are two diffrent ways for turning it on and off.
function screen(){
    if [ $1 = on ]; then
        # Version 1: enables hdmi port after the screen was in standby
        vcgencmd display_power 1 >/dev/null

        # Version 2: send cec-signal to the screen that he should wake up
        #echo on 0 | cec-client -s -d 1

        # Version 2b: if the screen turns on but at wrong input (e.
g. AV1) force it to
        # switch to HDMI1 port. This works only if the screen is on already
        # therefore send it a few times in a row. 4F:82:10:00 is HDMI1, 4F:
82:20:00
        # is HDMI2, 4F:82:30:
00 is HDMI3 and so on. Attention this is an unoffical
        # cec-signal which may not work with your screen.
        #echo tx 4F:82:10:00 | cec-client -s -d 1
        #echo tx 4F:82:10:00 | cec-client -s -d 1
        #echo tx 4F:82:10:00 | cec-client -s -d 1
        #echo tx 4F:82:10:00 | cec-client -s -d 1
        #echo tx 4F:82:10:00 | cec-client -s -d 1
        #echo tx 4F:82:10:00 | cec-client -s -d 1
        #echo tx 4F:82:10:00 | cec-client -s -d 1
    elif [ $1 = off ]; then
        # Version 1: disable hdmi port that the screen goes in standby
        vcgencmd display_power 0 >/dev/null

        # Version 2: send cec-
signal to the screen that he should go in standby
        #echo standby 0 | cec-client -s -d 1
    else
        echo Unknown parameter
    fi
}
```

Hierbei muss die Variable MONITOR die url des Monitores enthalten. Außerdem gibt es, abhängig vom Bildschirm den man ein/aus schalten möchte, mehr oder weniger Probleme. Deshalb wurden hier zwei verschiedene Optionen eingefügt wie sich ein Bildschirm ein bzw. aus schalten lässt. Falls das An-/Ausschalten des Bildschirms Probleme bereitet, kann in der Problembehandlung ein Lösungsansatz gefunden werden.

Mit STRG-o wird eine Datei nach einem weiteren Enter gespeichert und mit STRG-x wird nano verlassen.

Diese Datei enthält die Befehle um den Divera Monitor an

```
monitor on
```

und aus zu schalten

```
monitor off
```

. Sowie den Bildschirm an

```
screen on
```

und aus zu schalten

```
screen off
```

Damit die Befehle nun auch im Terminal verwendet werden können, muss die Datei in der bashrc gesourced werden. Dafür wird im Terminal die bashrc geöffnet

```
nano .bashrc
```

und am Ende der Datei wird in einer neuen Zeile Folgendes hinzugefügt:

```
source .divera_commands.sh
```

Nachdem die Zeile hinzugefügt wurde kann die bashrc im Terminal neu geladen werden mit dem Befehl:

```
. ~/.bashrc
```

Nun sollten die vier Kommandos in der bash ausführbar sein (Achtung: Beim Ausführen von `screen off` wird der Bildschirm ausgeschaltet und die Eingabe der Tastatur wird nicht mehr erkannt. Deswegen muss, wenn das Kommando aus der Kommandozeile ausgeführt wird, im Anschluss der Raspberry neu gestartet werden).

Automatisches Ein-/Ausschalten über Zeitplan od. Bewegungsmelder: (Optional)

Ohne Bewegungsmelder

Wachenmonitor ohne Bewegungsmelder

Skript einrichten

Zu Beginn muss die Skript-Datei angelegt und befüllt werden. Dies geschieht mit Folgenden Terminalbefehl:

```
nano .divera_script.py
```

Weiter ist folgendes Skript einzufügen:

In dem Skript muss noch der Accesskey hinzugefügt werden und die Dienstzeiten müssen angepasst werden.

Nachdem das Skript hinzugefügt wurde, muss es mit folgendem Befehl noch ausführbar gemacht werden:

```
chmod +x .divera_script.py
```

Das Skript kann nun im Terminal mit folgenden Befehl aufgerufen werden:

```
./.divera_script.py
```

Mit Bewegungsmelder

Wachenmonitor mit Bewegungsmelder

Im Folgenden kann nun entweder das Bash- oder das Python-Skript verwendet werden. Sie enthalten beide die gleiche Funktionalität, je nach Vorliebe kann eins ausgewählt werden. Generell wird empfohlen, das Python-Skript zu verwenden.

Python Skript:

Zu Beginn muss die Skript-Datei angelegt und befüllt werden. Dies geschieht mit Folgenden Terminalbefehl:

```
nano .divera_script.py
```

und fügen dieses Skript ein:

In dem Skript muss noch der Accesskey hinzugefügt werden und die Dienstzeiten müssen angepasst werden.

Nachdem das Skript hinzugefügt wurde muss es mit folgenden Befehl noch ausführbar gemacht werden:

```
chmod +x .divera_script.py
```

Das Skript kann nun im Terminal mit folgenden Befehl aufgerufen werden:

```
./.divera_script.py
```

Bash Skript:

Zu Beginn muss die Skript-Datei angelegt und befüllt werden. Dies geschieht mit Folgenden Terminalbefehl:

```
nano .divera_script.sh
```

und fügen dieses Skript ein:

In dem Skript muss noch der Accesskey hinzugefügt werden und die Dienstzeiten müssen angepasst werden.

Nachdem das Skript hinzugefügt wurde muss es mit folgenden Befehl noch ausführbar gemacht werden:

```
chmod +x .divera_script.sh
```

Das Skript kann nun im Terminal mit folgenden Befehl aufgerufen werden:

```
./.divera_script.sh
```

Damit das Skript funktioniert, muss noch der Bewegungsmelder angeschlossen werden und der Bildschirmschoner eingerichtet werden.

Bewegungsmelder

Zum einem wird der Bewegungsmelder benötigt. Dieser kann zum Beispiel hier oder hier erworben werden. Dazu werden noch Jumperkabel zum Verbinden benötigt. Der Anschluss der Kabel kann aus dem Abbild von diesem Tutorial entnommen werden. Eine detalierte Beschreibung des Sensors ist hier zu finden. Beim Sensor sollte der Jumper set auf High gesetzt werden. Die Time On und Sensitivität müssen abhängig vom Ort des Sensors eingestellt werden.

Nun kann mit dem Autostart fortgefahren werden.

Optionale Einstellungen

Autostart einrichten (Optional)

Damit das Skript nun automatisch startet, muss eine Autostartdatei hinzugefügt werden. Dies geschieht mit folgenden Befehlen:

```
cd .config
mkdir -p ./lxsession/LXDE-pi
touch ./lxsession/LXDE-pi/autostart
nano ./lxsession/LXDE-pi/autostart
```

In die Autostartdatei wird folgender Inhalt hinzugefügt:

```
# remove the next three diamonds to use the desktop again
#@lxpanel --profile LXDE-pi
#@pcmanfm --desktop --profile LXDE-pi
#point-rpi
# start screensaver
#@xscreensaver -no-splash
# stops displaying mouse after five seconds without moving
@unclutter -display :0 -noevents - grab
# does not allow the raspberry to go to sleep
@xset s off
@xset s noblank
@xset -dpms
#starts script
#./.divera_script.sh
./.divera_script.py
```

Mit dieser veränderten Autostartdatei wird zum einen der Desktop nicht mehr gestartet (dies kann Rückgängig gemacht werden indem die ersten drei Kommandos wieder einkommentiert werden) zum anderen wird der Mauszeiger nach 5 Sekunden ausgeblendet.

Abhängig davon, ob ein Python oder Bash-Skript verwendet wird, muss eine der letzten beiden Zeilen auskommentiert werden. Das heißt, wenn das Python-Skript verwendet wird, darf die Zeile *./.divera_script.py* keine Raute am Anfang enthalten und *#./.divera_script.sh* muss mit einer Raute anfangen. Im Falle des Bash-Skripts muss es *#./.divera_script.py* und *./.divera_script.sh* sein.

Wird das Script mit Bewegungsmelder verwendet muss die Zeile *#@xscreensaver -no-splash* auskommentiert werden!

Dieses Skript ließt, im Gegensatz zum normalen Skript, einen angeschlossenen Bewegungsmelder aus. Dies wird dazu genutzt, um in Dienstzeiten, wenn niemand vor dem Monitor steht, einen Bildschirmschoner anzuzeigen und außerhalb von Dienstzeiten bei einer Bewegung den Monitor einzuschalten. Außerdem werden in diesem Skript Alarme, die das Wort "Probealarm" enthalten, explizit nicht als Alarm behandelt.

Bildschirmschoner (Optional)

Außerdem muss noch der Bildschirmschoner eingerichtet werden. Als Bildschirmschoner kann bspw. xscreensaver verwendet und nach Wünschen eingerichtet werden. Die Einstellungen können unter Start->Einstellungen->Bildschirmschoner gesetzt werden. Beim ersten Start muss der xscreensaver-Prozess, wie im Pop-Up empfohlen, gestartet werden. Mittels

```
sudo apt install xscreensaver-gl-extra
```

wird unter anderem der Bildschirmschoner GLSlideshow installiert mit dem eine einfach Slideshow von Bildern angezeigt werden kann.

Problembehandlung

Screen-Befehl und Raspberry Pi 4

Der verwendete Befehl zum abschalten des Monitorausgangs ist für die Modellreihe des Raspberry Pi 4 derzeit seitens des Betriebssystems nicht implementiert und kann daher nicht verwendet werden.

Alternativ kann der Befehl "vcgencmd display_power x" verwendet werden.
Dabei ist x gegen 0 für Bilschirm = aus und 1 für Bildschirm = an zu verwenden.

Schwarze Ränder am Bildschirm

Falls am Rand des Bildschirms schwarze Ränder vorkommen, führe folgendes Kommando aus:

```
sudo nano /boot/config.txt
```

Entferne dort die # vor der folgenden Zeile:

```
#disable_overscan=1
```

Nach einem Neustarten des Raspberrys sollten keine schwarzen Ränder mehr vorhanden sein.

Falls dennoch welche vorhanden sein sollten oder zu wenig Rand vorhanden sein sollte, kann dieser manuell in der Datei angepasst werden. Dazu wird die config-Datei wieder geöffnet:

```
sudo nano /boot/config.txt
```

Die folgenden Zeilen müssen auskommentiert werden und die Werte manuell geändert werden.

```
#overscan_left=16
#overscan_right=16
#overscan_top=16
#overscan_bottom=16
```

Nach einem Neustarten werden die Änderungen sichtbar. Achtung: Nach jedem Ändern der Werte muss der Raspberry neu gestartet werden um die Änderung sichtbar zu machen.

Bildschirm geht nicht aus

Obwohl der Bildschirm ausgehen müsste, also kein Dienst ist (und keine Bewegung detektiert wurde), bleibt er an. Dies ist ein häufiges Problem bei älteren Fernsehern, die das Ein/Ausschalten des HDMI-Ports nicht erkennen. Um dies zu Beheben ist im `.divera_commands.sh` eine weitere Option, mit der es funktionieren kann. Hierbei werden cec-Signale an den Fernseher gesendet, allerdings unterstützen auch nicht alle Fernseher alle cec-Signale. Dies entspricht der Version 2 im Quellcode. Um dies zu verwenden muss die Version 1 einkommentiert und Version 2 auskommentiert werden (dies ist beim Ein/Ausschalten im Quellcode notwendig, also 2 Positionen im Quellcode).

Außerdem wurde hier noch ein manuelles Umschalten auf den HDMI-Port 1 hinzugefügt, falls der Fernseher standardmäßig auf einem anderen Eingang ist. Das heißt, dass der Monitor zwingend an HDMI-Port 1 angeschlossen werden muss oder der Port im Quellcode geändert werden muss. Das Umschalten des HDMI-Ports ist nicht in der cec-Spezifikation vorhanden, das heißt, dass dies ebenso nicht jeder Fernseher unterstützen muss. Dies entspricht der Version 2b und muss Zusätzlich zu Version 2 auskommentiert werden.

Danksagung

Dieses Tutorial wurde uns von Dustin Matzel aus dem THW OV Gütersloh zur Verfügung gestellt. Vielen Dank hierfür! Habt Ihr auch interessante Eigenbaulösungen erstellt und möchtet diese den Nutzern von DIVERA 24/7 zur Verfügung stellen? Meldet euch einfach bei unserem Support per E-Mail oder das Kontaktformular.