

# Documentatie - Redundante stations

IDP Waterkering

Gemaakt door:  
Sijmen Jaarsma  
Jeroen van der Burgt

# Inhoudsopgave

<b>Documentatie - Redundante stations</b>	<b>1</b>
Inhoudsopgave	2
Inleiding	3
Benodigdheden	3
Hardware	3
Raspberry Pi 3 Model B - 2x	3
UTP netwerkkabel	3
Software	4
Raspbian	4
Django	4
Nano	4
Python libraries	4
os	4
time	4
import subprocess	4
Installatie stapsgewijs	5
Werking	8
<b>Bronvermelding</b>	<b>9</b>

## Inleiding

Dit is de documentatie over de failover van de proof of concept waterkering. Als de raspberry pi die de waterkering runt crashed kan een tweede raspberry pi de draad oppakken zonder veel data te verliezen. Op deze manier voorkom je ook dat de waterkering niet werkt op het moment waarop je waterkering in actie moet komen.

## Benodigdheden

hieronder volgt een lijst van keuzes die we gemaakt hebben over het gebruik van hardware en software communicatie bij het opzetten van een redundant systeem. Erbij staat vermeld waarom we deze systemen gebruiken.

### Hardware

Alle fysieke objecten die gebruikt zijn voor het maken van het systeem.

#### Raspberry Pi 3 Model B - 2x

We gebruiken twee Raspberry Pi's waarvan we van eerste (de main Raspberry Pi) gebruiken om de waterkering aan te sturen. De bedoeling is dat de tweede (backup) Pi het werk van hem overneemt, als de eerste Raspberry Pi uitvalt.

Raspberry Pi's zijn goedkoper dan de meeste minicomputers op de markt. Ze zijn zeer makkelijk in gebruik, vanwege het toestaan van de programmeertaal Python als basistaal voor de hardware, waardoor het opstellen van een proof of concept goed mogelijk is.

#### UTP netwerkkabel

Voor de verbinding tussen de twee Raspberry Pi's gebruiken wij een UTP netwerkkabel. Dit is alleen zo in het proof of concept, voor een officiële implementatie zouden we meerdere UTP kabels gebruiken om te voorkomen dat communicatie uitvalt als een van de UTP kabels breekt

UTP kabels zijn beter om te gebruiken dan een lokale router met een wireless netwerk, ze zijn snel, betrouwbaar en relatief goedkoop. Glasvezel zou ook een goede optie zijn, maar dat is moeilijker om aan te leggen

## Software

Een lijst van software toepassingen die we gebruikt hebben:

### Raspbian

Het standaard besturingssysteem op een Raspberry Pi, perfect om onze python code te draaien. Linux staat al een aantal jaren bekend als het besturingssysteem dat het meest continue kan draaien zonder dat er fouten ontstaan in de software.

We hebben voor Raspbian gekozen in plaats van Noobs, omdat er bredere ondersteuning bestaat voor Raspbian (het lijkt meer op debian, een operating system dat op veel meer systemen wordt gebruikt dan alleen Raspberry Pi's).

### Django

De keuze voor Django is niet direct van relevantie voor het redundant systeem. We hebben ervoor gekozen omdat django geschikt is om een goed werkende visuele layout te maken. We hebben uitgezocht of Django goed kon werken met redundantie en het leverde geen problemen op.

### Nano

Een standaard applicatie van linux die perfect is voor het aanmaken en bewerken van Python bestanden. We kunnen deze applicatie ook gebruiken om snelle aanpassingen aan de Django code aan te brengen.

## Python libraries

Er zijn een aantal libraries (extensions) voor Python die we gebruikten om redundantie mogelijk te maken.

### os

Deze import zorgt ervoor dat de Python code direct commando's aan het Raspbian operating system kan geven. Zo laten we het operating system de stappen maken die we nodig hebben om redundantie te laten optreden.

### time

Zorgt ervoor dat je een proces voor een bepaalde tijd kan laten rusten waardoor je Raspberry Pi niet overbelast raakt. Je kunt hiermee ook achter de leeftijden van aanpassingen komen.

### import subprocess

Zorgt ervoor dat een Python applicatie een andere applicatie kan opstarten. Op deze manier kan de backup Raspberry het werk van de main Raspberry opstarten in zijn eigen code.

## Installatie stapsgewijs

1. Als eerste beginnen we met Raspbian, deze moet geconfigureerd worden via een SD-Card. Deze kan geformatteerd worden via Win32DiskImager. Dat is een programma dat automatisch de SD-Card formatteert en Raspbian installeert. Het bestand dat je kan downloaden van de website is een .iso bestand. Met behulp van Win32DiskImager kan je de SD-Card selecteren en deze laten formatteren en daarna de juiste bestanden erop laten zetten.
2. Daarna moet er, voor het Technische Informatica gedeelte, Django en Django channels geïnstalleerd worden. Dit hebben we gedaan door middel van pip. Pip is een package manager, voor softwarepakketten voor Python, die we met vorige projecten ook nodig hebben gehad. Het voordeel van pip is dat meerdere bestanden tegelijkertijd kan installeren. Op het moment dat je Django en Django channels wil installeren gebruik je de volgende commando's:

```
sudo pip3 install Django
```

```
sudo pip3 install -U channels
```

3. Op het moment dat Django en Channels geïnstalleerd is gaan we het IP aanpassen van de Pi. Dit doen we zodat hij later gemakkelijk verbinding kan maken met ssh (Secure Shell). Navigeer naar de interfaces tab met:

```
sudo nano /etc/network/interfaces
```

Pas daarna de standaardinstelling van Eth0 aan, zodat deze automatisch verbinding probeert te maken met een pi op een static IP adres.

### Configuratie Raspberry Pi 1 (Main Raspberry Pi)

```
auto eth0
iface eth0 inet static
address 192.137.0.1
netmask 255.255.255.0
```

### Configuratie Raspberry Pi 2 (Backup Raspberry Pi)

```
auto eth0
iface eth0 inet static
address 192.137.0.2
netmask 255.255.255.0
```

Sla nu op met ctrl+o, dan de enter toets en daarna ctrl+x.

De netwerkinstellingen zijn hiermee voltooid.

- Op het moment dat beide netwerkverbindingen ingesteld staan als bovenstaande, wordt het u aangeraden om de Raspberry Pi opnieuw te laten opstarten. Dit wordt gedaan met het volgende commando:

```
sudo reboot
```

Als de instellingen nu juist zijn ingesteld dan kan er een ping plaatsvinden tussen de beide Raspberry Pi's. Dit doen we om te bekijken of er verbinding kan plaatsvinden tussen beide Raspberry Pi's.

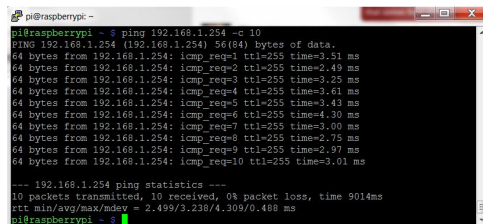
Raspberry Pi 1 (Main Raspberry Pi)

```
ping 192.168.137.2
```

Raspberry Pi 2 (Backup Raspberry Pi)

```
ping 192.168.137.1
```

Als er een verbinding gemaakt wordt dan ziet dit er ongeveer zo uit:



```
pi@raspberrypi:~$ ping 192.168.1.254 -c 10
PING 192.168.1.254 (192.168.1.254): 56(84) bytes of data:
64 bytes from 192.168.1.254: icmp_req=1 ttl=255 time=3.51 ms
64 bytes from 192.168.1.254: icmp_req=2 ttl=255 time=2.49 ms
64 bytes from 192.168.1.254: icmp_req=3 ttl=255 time=3.25 ms
64 bytes from 192.168.1.254: icmp_req=4 ttl=255 time=3.43 ms
64 bytes from 192.168.1.254: icmp_req=5 ttl=255 time=3.43 ms
64 bytes from 192.168.1.254: icmp_req=6 ttl=255 time=4.30 ms
64 bytes from 192.168.1.254: icmp_req=7 ttl=255 time=3.00 ms
64 bytes from 192.168.1.254: icmp_req=8 ttl=255 time=2.75 ms
64 bytes from 192.168.1.254: icmp_req=9 ttl=255 time=2.97 ms
64 bytes from 192.168.1.254: icmp_req=10 ttl=255 time=3.01 ms
--- 192.168.1.254 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9014ms
rtt min/avg/max/mdev = 2.499/3.238/4.309/0.488 ms
pi@raspberrypi:~$
```

Op dat moment kan de volgende stap gemaakt worden.

- SSH instellen voor de raspberry pi connectie. Er zal op dit moment de mogelijkheid zijn om ssh in te stellen voor de Raspberry Pi. Dit is nodig, zodat het backup systeem feilloos gaat werken. Op dit moment zijn er nog geen keys gegenereerd en daarom gaan we dit als eerst doen:

```
sudo ssh-keygen
```

Daarna kunnen de standaardinstellingen hetzelfde blijven, druk op enter, totdat de keygen aangemaakt is. Op dit moment is er een ssh key beschikbaar en opgeslagen op de Raspberry Pi. Nu is het de zaak om dit ook op de andere Raspberry Pi aan te maken. Mocht het zo zijn dat ssh uitgeschakeld staat op uw Raspberry Pi, dan kunt u deze inschakelen op deze manier:

```
sudo raspi-config
```

Daarna kan er via Advanced Options, ssh aangezet worden.

6. Nu gaan we de virtuele handdruk maken door het wachtwoord van de andere Raspberry Pi in te vullen op de eerste Raspberry Pi. Dan kunnen met de eerste Raspberry Pi inloggen zonder een wachtwoord in te moeten vullen, omdat de key wordt opgeslagen in een database.

Configuratie Raspberry Pi 1 (Main Raspberry Pi)

```
ssh-copy-id -i ~/.ssh/id_rsa.pub 192.137.0.2
```

Configuratie Raspberry Pi 2 (Backup Raspberry Pi)

```
ssh-copy-id -i ~/.ssh/id_rsa.pub 192.137.0.1
```

Als dit is ingesteld kan er een altijd verbinding worden gemaakt met de andere Pi, zonder een wachtwoord in te moeten vullen. Dit kan je doen door middel van het volgende commando:

Op Raspberry Pi 1 (Main Raspberry Pi) voor het verbinding maken naar Raspberry Pi 2 (Backup Raspberry Pi).

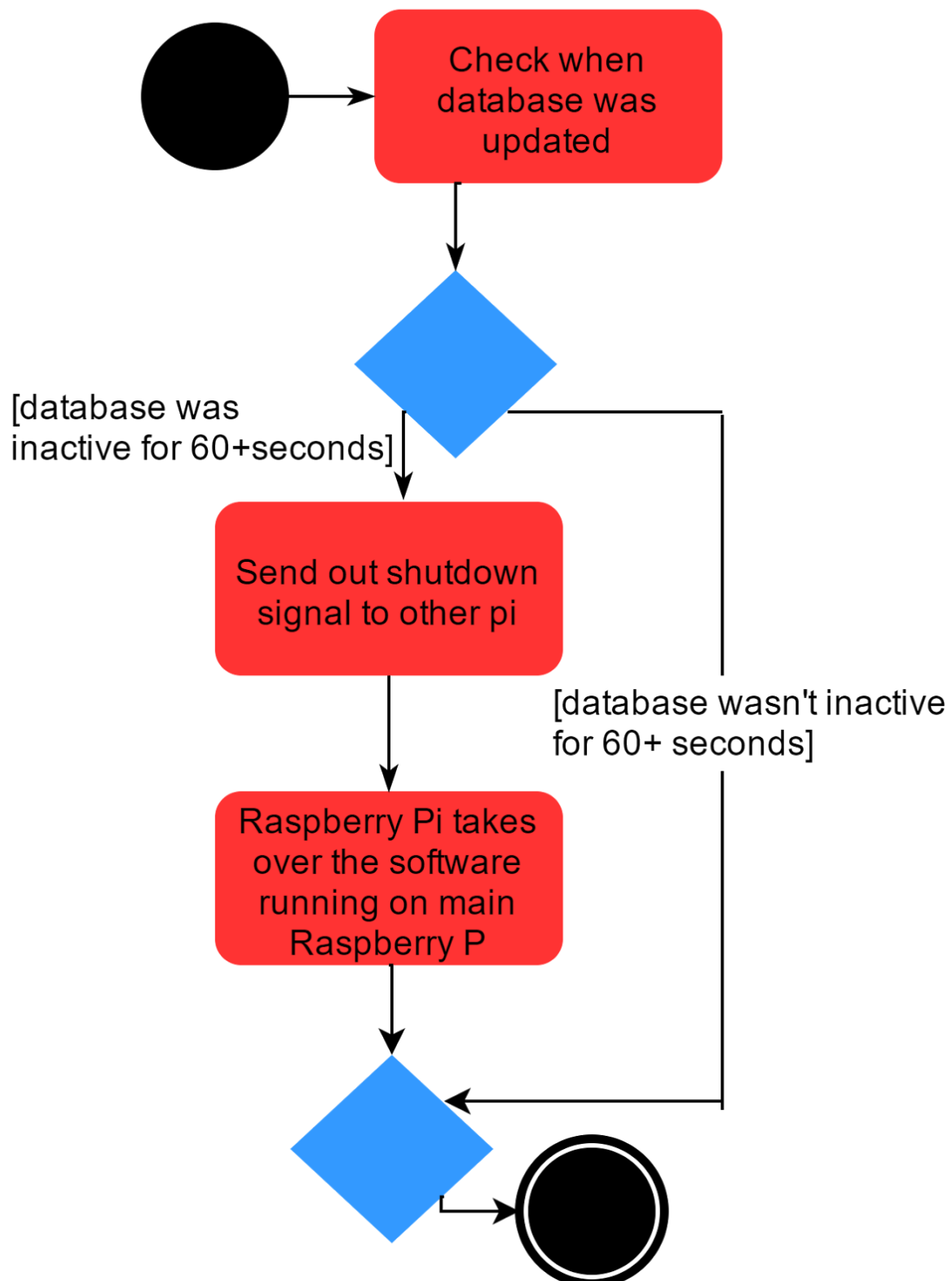
```
ssh 192.137.0.2
```

Op Raspberry Pi 2 (Backup Raspberry Pi) voor het verbinding maken naar Raspberry Pi 1 (Main Raspberry Pi).

```
ssh 192.137.0.1
```

Na deze acties kan de code worden uitgevoerd op de Raspberry Pi.

## Werking





De failover werkt in een paar stappen:

1. Raspberry 1 voert zijn standaard metingen uit en slaat ze op in de database;
2. Elke 20 metingen worden de metingen verstuurd naar de backup raspberry om daar de database bij te werken;
3. Elke 20 seconden checkt de backup pi of deze updates heeft gekregen van het centrale systeem;
  - a. Als de laatste update meer dan 60 seconden geleden was dan geeft hij de eerste raspberry pi een shutdown commando
  - b. Als de laatste update minder dan 60 seconden geleden was dan herstart hij vanaf stap 1

## Bronvermelding

Natarajan, R. (2008, 20 november). 3 Steps to Perform SSH Login Without Password Using ssh-keygen & ssh-copy-id. Geraadpleegd van <http://www.thegeekstuff.com/2008/11/3-steps-to-perform-ssh-login-without-password-using-ssh-keygen-ssh-copy-id>

Raspberrypi.org. (z.j.). 3 Steps to Perform SSH Login Without Password Using ssh-keygen & ssh-copy-id. Geraadpleegd op 20 januari, 2017, van <https://www.raspberrypi.org/documentation/remote-access/ssh/scp.md>

RPI ADMIN (2015, 19 mei). Django + Raspberry Pi Tutorial (PART I). Geraadpleegd van <http://rasberrypituts.com/django-raspberry-pi-tutorial/>