

Race conditions in Flask

Bente Schopman

Januari 2019

Abstract

komt nog

1 Introductie

De hoofdvraag is *Wat moet je doen als er race conditions in Flask zijn?*. De deelvragen zijn:

1. *Wat is een race condition in Flask?*
2. *Hoe ontstaan race conditions in Flask?*
3. *Hoe kan je race conditions voorkomen in Flask?*
4. *Hoe kan je beschermen tegen race conditions in Flask?*

In dit paper wordt beschreven wat een race condition is, waarom het een probleem is, wat het veroorzaakt en hoe je het kan voorkomen en ook hoe je je er tegen kan beschermen. Het doel van dit onderzoek is om er voor te zorgen dat bedrijven en personen geen last meer hebben van race conditions bij Flask.

1.1 Aanpak

Het onderzoek is als volgt aangepakt.

1. Wat is een race condition in Flask?

*Aanpak: Gezocht op **race conditions in Python**. De bronnen zijn gekozen op basis van de Google zoekresultaten en of het een betrouwbare bron is. Dat betekent dat het een autoriteit moet zijn. Dus dat het vaak wordt gebruikt als referentie. Daarvoor is op Google gezocht op **best websites for python**. Daar kan CodeConquest met 50 websites uit. Die link daarvoor kan je hier vinden: [1]*

2. Hoe ontstaan race conditions in Flask?

*Aanpak: Gezocht op **race conditions in Python**. De bronnen zijn gekozen op basis van de Google zoekresultaten en of het een betrouwbare bron is. Dat betekent dat het een autoriteit moet zijn. Dus dat het vaak wordt gebruikt als referentie. Daarvoor is op Google gezocht op **best websites for python**. Daar kan CodeConquest met 50 websites uit. Realpython kwam daar uit als een van de 50 websites. Daarna een praktijkonderzoek gedaan door 2 functies te maken in Flask op ba-*

sis van de RealPython[2] die er uit kwam in Python en veranderd voor de bruikbaarheid van bij het framework Flask.

3. Hoe kan je race conditions voorkomen in Flask?

*Door op de InHolland bibliotheek te zoeken op **how to prevent race conditions in** Daarvoor is op Google gezocht op **best websites for python**. Daar kan CodeConquest met 50 websites uit. Realpython kwam daar*

uit als een van de 50 websites. Daarna een praktijkonderzoek gedaan door 2 functies te maken in Flask op basis van de RealPython[2] die er uit kwam in Python en veranderd voor de bruikbaarheid van bij het framework Flask.

4. Hoe kan je beschermen tegen race conditions in Flask?
*Door op de InHolland bibliotheek en op Google te zoeken op **race conditions in python***

2 Begrippen

komt nog

3 Wat is een race condition

Race Conditions kunnen voorkomen als meerdere threads dezelfde bron gebruiken.[2]. Dit kan gebeuren als twee threads een *sleep*time hebben en een gedeelde resource hebben. Dat kan je zien in het volgende voorbeeld van [3]:

```
Thread 123145539575808 started...
Thread 123145539575808 read value 0
Thread 123145544830976 started...
Thread 123145544830976 read value 0
Thread 123145539575808 slept for 1
Thread 123145539575808 write 1
Thread 123145544830976 slept for 3
Thread 123145544830976 write 1
```

Eigenlijk verwacht je het resultaat 2 maar het resultaat is 1. Dat is dus een goed voorbeeld van een race condition.

4 Oorzaak voor race conditions in flask

De oorzaak voor race conditions in Flask is dat meerdere threads de dezelfde source willen gebruiken. De meeste race conditions zijn lastig om te ontdekken. [2] Dit wordt ook onder-

zocht door de bron [2] te gebruiken om twee voorbeelden van race conditions te laten zien. De code daarvoor is te vinden op https://github.com/SchopmanBente/race_conditions_flask. Het eerste voorbeeld gaat over een gedeelde variabele die door twee threads wordt geüpdatet. De uitkomst

is te zien in afbeelding 1. Het tweede voorbeeld gaat over een gedeelde variabele die door drie threads wordt geüpdatet. De uitkomst is te zien in afbeelding 2.

5 Race conditions voorkomen in Flask

Om een race condition te voorkomen in Flask, kan je het beste een *lock* gebruiken. Een *lock* is een mechanisme die ervoor zorgt verschillende threads gesynchroniseerd worden waarbij er limieten zijn om ervoor te zorgen dat er niet ongelimiteerd gebruik van een bepaalde gedeelde *source* in een *computing environment*. Het is een methode die is bedoeld om de toegang te regelen door simultaneous control policies toe te passen. [4]

Dat is toegepast in de derde functie van de applicatie flask_race_conditions. De derde functie is een applicatie met drie threads daardoor heeft de database.update

functie een lock
.Dat is te zien in afbeelding 3.

6 Beschermen tegen race conditions

7 Conclusie

References

- [1] Code Conquest. The 50 best websites to learn python, 2016. Last accessed 5 januari 2020.
- [2] J. Anderson. An intro to threading in python, 2019. Last accessed 5 januari 2020.
- [3] M. M. Tahrioui. *asyncio Recipes*. Mohamed Mustapha Tahrioui, Darmstadt, Hessen, Germany, 2019.
- [4] Techopedia. An intro to threading in python, z.d. Laast bezocht op 17 januari 2020.