

## TI-Nspire CX II

## Achtergrondinformatie

- Wachtwoorden zijn beveiligingsmaatregelen om ongeautoriseerde toegang tot accounts, sloten en apparaten te voorkomen.
- Een wachtwoord kan op een computer of apparaat op een manier worden opgeslagen die niet leesbaar is voor mensen. Als hackers je computer binnendringen, zullen ze je wachtwoord in platte tekst niet vinden. In plaats daarvan zien ze een versleutelde versie van het wachtwoord. Wanneer een account wordt aangemaakt, wordt het wachtwoord in **platte tekst** door een wiskundige hashingfunctie gescrambled die niet ongedaan kan worden gemaakt, zoals **SHA-256**. De output van een hashingfunctie is de **hash** en heeft een lengte van 256 bits. De hash wordt opgeslagen op de micro:bit als een tekenreeks van 64 tekens in een bestand genaamd "wachtwoord.txt".
- Tijdens het inloggen wordt een **authenticatieproces** gebruikt om te verifiëren of het wachtwoord van de gebruiker correct is. Tijdens de authenticatie wordt de hash die in het bestand "wachtwoord.txt" is opgeslagen, gelezen vanaf de micro:bit en vergeleken met de uitvoer van de hashingfunctie van het zojuist ingevoerde wachtwoord. Als de twee gelijk zijn, krijgt de gebruiker toegang.
- Een hacker kan de hash van een wachtwoord niet gebruiken zoals het wachtwoord in platte tekst, omdat tijdens de authenticatie de hash opnieuw door het hashing proces zal gaan en dus niet de hash, maar de hash van de hash gevalideerd wordt.

## Wat is jouw opdracht?

1. Oefenen op hashing:
  - a. Ga naar de pagina met "**Oefenen\_hash.py**", voer het programma uit en voer een wachtwoord in om de hash-tekenreeks weer te geven. Let op – De tekenreeks is 64 tekens of 32 paren; elk paar vertegenwoordigt een tweecijferig hexadecimaal getal; bijvoorbeeld, "e4" vertegenwoordigt het enkele byte 0xe4.  $32 \text{ bytes} \times 8 \text{ bits/byte} = 256 \text{ bits}$ .
2. Stel het wachtwoord in op de micro:bit:
  - a. Ga naar de pagina met "**stel\_wachtwoord\_in.py**" en voer het programma uit om je wachtwoord te hashen en de hash-tekenreeks op te slaan in het bestand "wachtwoord.txt" op je micro:bit. Let op – de hash is een tekenreeks van 64 tekens; elk teken vereist één byte geheugen wanneer het op de micro:bit wordt opgeslagen.
3. Lees de hash op de micro:bit:
  - a. Ga naar de pagina met "**lees\_wachtwoord\_hash.py**" en voer het programma uit. Het bestand genaamd "wachtwoord.txt" wordt van de micro:bit gelezen en weergegeven. De hash-tekenreeks die in 2a is opgeslagen, wordt gelezen en op de rekenmachine weergegeven. Let op – de tekenreeks is **niet het wachtwoord in platte tekst**.
4. Oefen op authenticatie:
  - a. Ga naar de pagina met "**authenticatie.py**" en voer het programma uit om je wachtwoord en de authenticatieroutine te testen. Dit programma vergelijkt de hash die op de micro:bit is opgeslagen met de hash van het ingevoerde wachtwoord. Als de twee gelijk zijn, krijgt de gebruiker toegang.
5. Remote login:
  - De **ontvanger**:
    - **Fluister het wachtwoord van JOUW micro:bit naar de zender.** Ga naar '**student\_ontvanger.py**', wijzig de groep naar je toegewezen nummer en voer het programma vervolgens uit **voordat** de zender zijn programma heeft uitgevoerd.

## TI-Nspire CX II

- De zender:
  - Ga naar '**student\_zender.py**', wijzig de groep naar je toegewezen nummer en voer je programma uit **nadat** de ontvanger en hacker hun programma gestart hebben. Voer het **wachtwoord van de ontvanger's micro:bit** in. Je zult proberen in te loggen op de micro:bit van de ontvanger.
- De hacker:
  - Ga naar '**student\_hacker.py**', wijzig de groep naar je toegewezen nummer en voer het programma uit **voordat** de zender hun programma heeft uitgevoerd.
- Bespreek binnen de groep welke boodschap er via de radio is verzonden. Kan de hacker de boodschap die ze ontvangen hebben gebruiken om de micro:bit van de ontvanger te authenticeren? Wissel van rol en probeer het opnieuw.

### De code

#### Zender

```
student_zender.py 11/12
from microbit_radio import *
from hashing import *
# Gebruik het wachtwoord van de ontvanger
kanaal = 1
groep = 1
clear_history()
wachtwoord = input("Voer wachtwoord in:")
wachtwoord_hash = sha_hash(wachtwoord)
tx(wachtwoord_hash,kanaal,groep)
```

#### Ontvanger

```
*student Ontvanger.py 1/15
from microbit_radio import *
from hashing import *
# Deel je wachtwoord met de zender
kanaal = 1
groep = 1
clear_history()
test_hash = rx(kanaal,groep)
authentic_hash = read_file("wachtwoord.txt")
if test_hash == authentic_hash:
    display.show(Image.YES)
    music.play(music.BA_DING)
```

#### Hacker

```
student_hacker.py 11/11
from microbit_radio import *
from hashing import *
# wachtwoord mag niet doorgegeven zijn aan
# hacker
kanaal = 1
groep = 1
clear_history()
print("Man-in-the-middle aanval!")
hacked_hash = rx(kanaal,groep)
print ("Hacked hash:",hacked_hash)
```

### Extra uitdagingen

- Voer het programma "authentication.py" opnieuw uit en voer drie keer achter elkaar een verkeerd wachtwoord in.
- Probeer opnieuw in te loggen op afstand, maar stuur het hash-bericht dat de hacker heeft ontvangen in plaats van het wachtwoord in platte tekst.
- Probeer een andere rol binnen je team.

### Samengevat

- De wachtwoordhash, niet het wachtwoord in platte tekst, wordt opgeslagen op de micro:bit in het bestand genaamd "wachtwoord.txt."
- Een hacker kan de hash niet gebruiken als wachtwoord tijdens het inloggen; het zou opnieuw gehasht worden en de authenticatie zou mislukken.
- Een hashingfunctie is een wiskundige versleuteling. Het is een eenrichtingsproces; met andere woorden, het kan niet worden omgekeerd.
- Een uniek wachtwoord heeft één, en alleen één, hash.

### Tips voor als het misgaat

- Controleer of iedereen in het team hun toegewezen groepsnummer gebruikt.
- Zorg ervoor dat de ontvanger en hacker hun programma's uitvoeren en wachten voordat de zender het bericht verzendt.
- Zorg ervoor dat de wachtwoorden succesvol zijn ingesteld op elke micro:bit.