Write-up for

NC3 Jule-CTF 2022

opgave "sparenissen"

Indledning
Opgaven består er et lille digt og en binær fil.

(a) Sparenissen skriver kort og trangt,
(b) hilser ikke først,
(c) og siger ej farvel,
(d) modsætter sig størst,
(e) at gentage sig selv,
(f) men stadig bliver brevet stort og langt.

```
------------------------------------------------------------------------
     0   6E 63 33 7B 73 70 61 72  65 81 A6 49 28 80 B6 47   nc3{spare..I(..G
    10   98 28 92 A6 4A 1D 2A A2  7A A8 81 BE 76 DB 6A 68   .(..J.*.z...v.jh
    20   AD 9F 1A 1B 29 C6 DA AE  C6 AC A9 B2 DA A4 6A B9   ....).........j.
    30   DA DA AF A2 CA A8 B6 AC  A6 9A 69 A0 C6 DA 99 B2   ..........i.....
    40   9B 1A A9 AA 88 2A CA 9C  B2 A2 A6 BB A7 B6 AD 69   .....*.........i
    50   C9 F4                                             ..
------------------------------------------------------------------------
```

Hints til at gennemskue opgaven
Det lille digt giver nogle forskellige spor:

- filen indeholder ikke mere end allerhøjst nødvendigt (a)
- der er ingen header (b)
- ej heller en footer (c)
- der er gjort noget for at undgå gentagelser (d+e)
- beskeden er formentlig længere end antalle af bytes i filen (f)

Ud fra ovenstående hints kunne man formode at de første 9 karakterer "nc3{spare" rent faktisk er den første del af beskeden og at den næste karakter (index 9) ikke skrives fordi det er en gentagelse.

Nedenfor ses de første 12 bytes gengivet i binær:

```
idx   hex  binary        idx   hex  binary        idx   hex  binary
-------------------------------------------------------------------
 0    6E   01101110       4    73   01110011       8    65   01100101
 1    63   01100011       5    70   01110000       9    81   10000001
 2    33   00110011       6    61   01100001      10    A6   10100110
 3    7B   01111011       7    72   01110010      11    49   01001001
-------------------------------------------------------------------
```

En fin teori kunne være, at det mest betydende bit i den 10. byte (index 9) indikerer at der er tale om en "gentagelse" – man kunne måske formode at teksten starter med "sparenisse". Hvis det er tilfældet, og der rent faktisk skal spares nogle bit, bør der bruge mindre en 7 bit på at angive hvilken karakter som skal gentages.

Teorien er altså at de første 5 eller 6 bit (som er 0) henviser tilbage til det første tegn (n), så der kan stå "sparenissen". I så fald skal den næste karakter være et "i", hvilket kommer til at passe, hvis der er præcis 5 bit til et tilbagevisende index. Nedenfor er de 4 bytes ved index 8 -11 del op i binær:

```
--------------------------------------
   01100101       -> ascii 'e'
   100000         -> index 0 -> 'n'
   01101001       -> ascii 'i'
   100100         -> index 4 -> 's'
   1001...
--------------------------------------
```

Opsummeret er der altså 7 bit til hver karakter (fordi det første 0 angiver at karakteren er 'ny' og ikke set før), og hvis den første bit er 1, er de følgende 5 bit en index for en tidligere karakter (talt fra begyndelsen).

Den komplette løsning

Følgende (lidt grove) python-script parser de 82 bytes som en bitstream og dekoder beskeden som ender på 102 tegn.

```
#!/usr/bin/python3

with open("sparenissen", "br") as wfd:
	data = wfd.read()

index = 0
byte = 0
def nextbit(count):
	global index, byte
	bits = 0
	while count > 0 and index < len(data) * 8:
		if index % 8  == 0:
			byte = data[index//8]
		bits = (bits << 1) | (1 if byte & 0x80 else 0)
		byte = (byte << 1) & 0xFF
		index += 1
		count -= 1
	return bits

history = []
while index < len(data) * 8:
	if(nextbit(1) == 0):
		# character found in bitstream
		char = chr(nextbit(7))
		history.append(char)
	else:
		# character found in history
		idx = nextbit(5)
		char = history[idx]
	print(char, end='')
print()
```

Hermed ses flaget:

```
	nc3{alle-ved-at-sparenissen-synes-det-er-enormt-hyggeligt-og-lidt-sjovt-med-et-
	laaaangt-flag-i-en-lille-kort-fil}
```