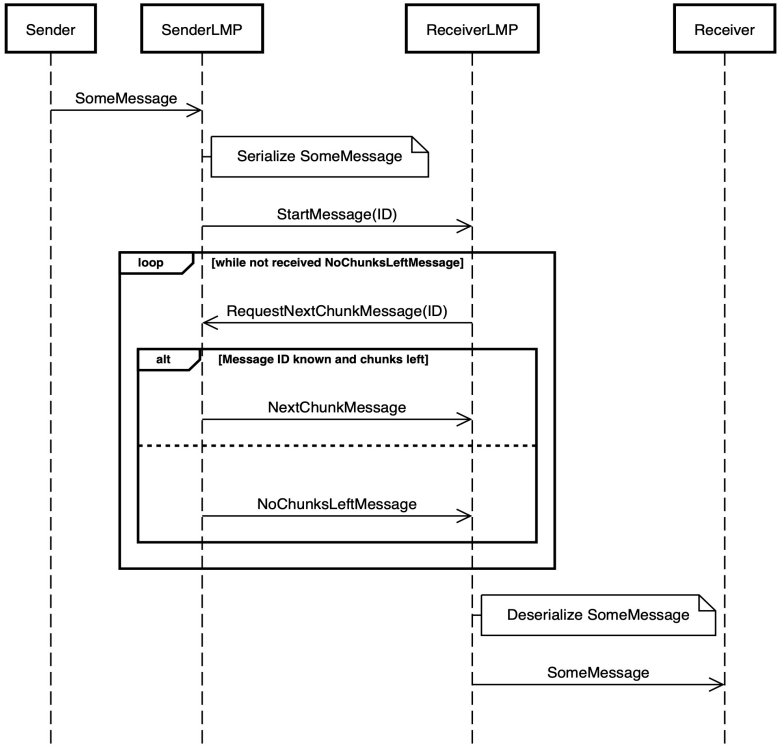
Akka-Handson

So LargeMessageProxy, much Password Crack…

Team: Chewbakka
Timofei Kornev
Felix Gohla

1. LargeMessageProxy

- Simple Pull-Pattern:
  - Sender LMP sends start message
  - Receiver LMP pulls chunks until all chunks received.
  - Advantages:
    - Inbox won't be flooded.
    - We could delete chunks already transmitted
  - Disadvantages:
    - How large should a chunk be? (MTU 1500 Ethernet)
    - Wireshark shows: many empty packets



DDM Exercise: Akka-Handson

Team: ChewbAKKA
Timofei Kornev
Felix Gohla
Chart **2**

2. Password Cracking (1/5)

- Hints are cool, but does it always make sense to crack them?
  - Task:
    - The more hints we have, the easier it is to find the password.
  - Each hint allows one more character to exclude:
    $$\#unqiueCharsInPassword = passwordLength - \#hints$$
    Example: password length of 11, 9 hints → password consists of 2 different characters
  - Difficulty (worst-case, max. number of hashes) of cracking a hint?
    $$D_{Hint} = (\#charsInAlphabet - \#crackedHints) \cdot (\#charsinAlphabet - 1)!$$

For each cracked hint, we can exclude the already known excluded characters.

Each hint is a permutation of one less character than the alphabet.

DDM Exercise: Akka-Handson

Team: ChewbAKKA
Timofei Kornev
Felix Gohla
Chart **3**

2. Password Cracking (2/5)

- Difficulty of cracking a password?

$$leftoverChars = (\#charsInAlphabet\ - \#crackedHints)$$

$$D_{Password} = \frac{(leftoverChars)!}{\#uniquePasswordChars! \cdot (\#hints\ - \#crackedHints)!} \cdot \#uniquePasswordChars^{passwordLength}$$

2. Password Cracking (3/5)

- Difficulty of cracking a password?

$$leftoverChars = (\#charsInAlphabet\ - \#crackedHints)$$

$$D_{Password} = \frac{(leftoverChars)!}{\#uniquePasswordChars! \cdot (\#hints\ - \#crackedHints)!} \cdot \#uniquePasswordChars^{passwordLength}$$

DDM Exercise: Akka-Handson

Team: ChewbAKKA
Timofei Kornev
Felix Gohla
Chart **5**

2. Password Cracking (4/5)

- Difficulty of cracking a password?

$$leftoverChars = (\#charsInAlphabet\ - \#crackedHints)$$

$$D_{Password} = \frac{(leftoverChars)!}{\#uniquePasswordChars! \cdot (\#hints\ - \#crackedHints)!} \cdot \#uniquePasswordChars^{passwordLength}$$

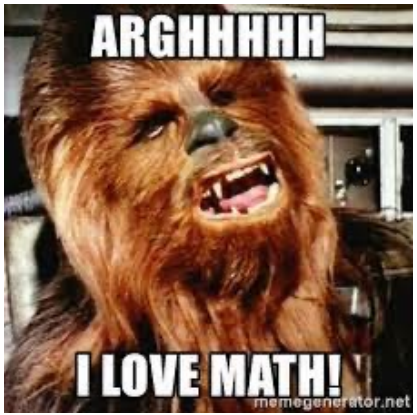
- Simple example:
  - 11 chars in alphabet, password length of 10, 9 hints (0 cracked) → 2 unique characters
  - $\frac{(11-0)!}{2! \cdot (9-0)!} \cdot 2^{10} = 56320$ combinations **vs.**
    $(11-0) \cdot (11-1)! = 39916800$ for cracking the first hint

DDM Exercise: Akka-Handson

Team: ChewbAKKA
Timofei Kornev
Felix Gohla

2. Password Cracking (5/5)

- Further improvements:
  - When there are less passwords than workers, assign already cracking passwords to the free workers.
  - They probe the combinations in a random order to not just waste energy.
- Cracks the given small dataset in ~2 seconds.



3. Tests

- Eeeeeeeehhhhh… wellllllll… :D