

Password Cracker

Sie sind erfolgreich in ein Computersystem eingedrungen und haben einen Ordner mit verschlüsselten Passwörtern gefunden. Sie konnten weiters durch Zufall herausfinden, dass die originalen Passwörter mit Hilfe der Hash-Funktion SHA-256 verschlüsselt wurden.

Weiters haben Sie herausgefunden, dass für die gefundenen Passwörter folgende Bedingungen zutreffen:

- **password0** ... Das Passwort besteht aus vier Kleinbuchstaben.
- **password1** ... Das Passwort besteht aus sechs Großbuchstaben.
- **password2** ... Das Passwort besteht aus fünf Zeichen. Die Zeichen können Kleinbuchstaben, Großbuchstaben und Ziffern sein.
- **password3** ... Sie wissen nicht, wie lange das Passwort ist. Sie haben aber im Browserverlauf folgenden Link gefunden: https://de.wikipedia.org/wiki/Liste_von_Fabelwesen. Vielleicht kann dieser Link weiterhelfen ...

Aufgabe 1

Entwickeln Sie das Backend als **.NET 6 WebApi**. Die Kommunikation soll per **SignalR** erfolgen. Der Hub soll dabei eine Anfrage von einem Client zum Entschlüsseln der übergebenen Passwortdetails entgegen nehmen können. Diese Details bestehen aus:

- Hashcode
- Vermutetes Alphabet des originalen Passworts
- Vermutete Länge des originalen Passworts

Implementieren Sie zwei Strategien zum Passwort-Entschlüsseln:

- Brute-Force
- Durchprobieren der gefundenen Fabelwesen (falls nur Hashcode übergeben wurde)

Stellen Sie sicher, dass der **Server** durch das Entschlüsseln eines Passworts **nicht blockiert wird**.

Wurde eine Übereinstimmung gefunden, senden Sie das gefundene Passwort an den Client zurück.

Wurde keine Übereinstimmung gefunden, senden Sie den Text `* no match *` zurück.

Senden Sie zusätzlich jede Sekunden den aktuellen Fortschritt in Prozent an den Client zurück (verwenden Sie dazu am Server die Klasse Progress).

Brute-Force

Probieren Sie alle möglichen Kombination der im Alphabet übergebenen Zeichen durch, ob der SHA-256 Hashcode mit dem übergebenen Hashcode übereinstimmt. Verwenden Sie dazu die Klasse `System.Security.Cryptography.SHA256`.

HINWEIS:

Implementieren Sie zuerst Algorithmen für 4, 5 und 6 Zeichen. Wurde eine andere Zeichenanzahl vom Client geliefert, meldet der Hub den Text `* invalid length *` zurück.

Überlegen Sie sich erst dann eine Lösung für eine beliebige Anzahl an Zeichen, wenn Ihr Algorithmus für 4, 5 oder 6 Zeichen funktioniert (sie werden für die allgemeine Lösung vermutlich eine rekursive Funktion benötigen).

Versuchen Sie nun, Ihren Algorithmus so zu beschleunigen, dass Sie das Durchprobieren der verschiedenen möglichen Zeichenkombinationen in verschiedene Tasks aufteilen (z.B. `a[a-zA-Z]` übernimmt der erste Task, `g[a-zA-Z]` der zweite usw.). Verwenden Sie dazu die Methode `Task.WhenAny()`.

Durchprobieren der gefundenen Fabelwesen

Wurde nur der Hashcode bei der Anfrage übermittelt, wenden Sie diese Strategie zum Herausfinden des verschlüsselten Passworts an. Lesen Sie dazu die im oben angegebenen Link vorkommenden Fabelwesen aus dem Quelltext aus und testen sie den Hashcode von jedem einzelnen Fabelwesen auf Übereinstimmung mit dem übergebenen Hashcode.

Verwenden Sie zum Parsen der URL `Html Agility Pack` (<https://html-agility-pack.net>).

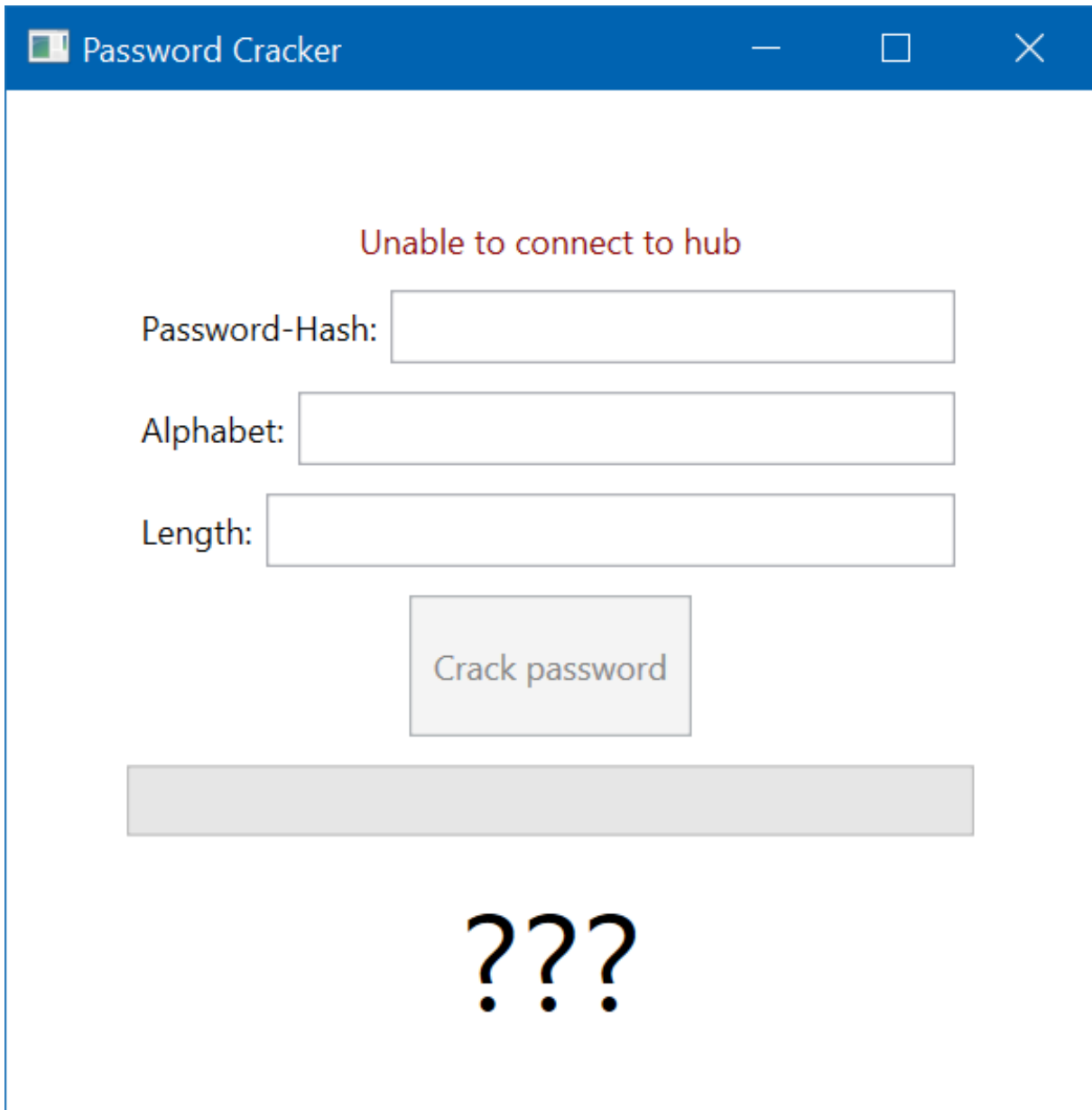
Beachten Sie, dass die `LoadHtml()`-Methode der Klasse `HtmlDocument` synchron arbeitet, der Server aber nicht durch das Auslesen Webseite blockiert werden soll ...

HINWEIS: Durchsuchen Sie das heruntergeladene HTML-Dokument nach allen Vorkommen eines `a`-Tags (XPath: `//a`), dessen `href`-Attribut mit `/wiki` beginnt. Das liefert zwar nicht exakt nur die Fabelwesen, aber das macht nichts :)

Aufgabe 2

Entwickeln Sie den **Client** als **.NET 6.0 WPF Application**. Beim Start des Programms soll sich sofort mit dem SignalR-Server verbunden werden (siehe [ASP.NET Core SignalR .NET Client](#))

Klappt das nicht, geben Sie einen Fehlertext aus:



Unable to connect to hub

Password-Hash:

Alphabet:

Length:

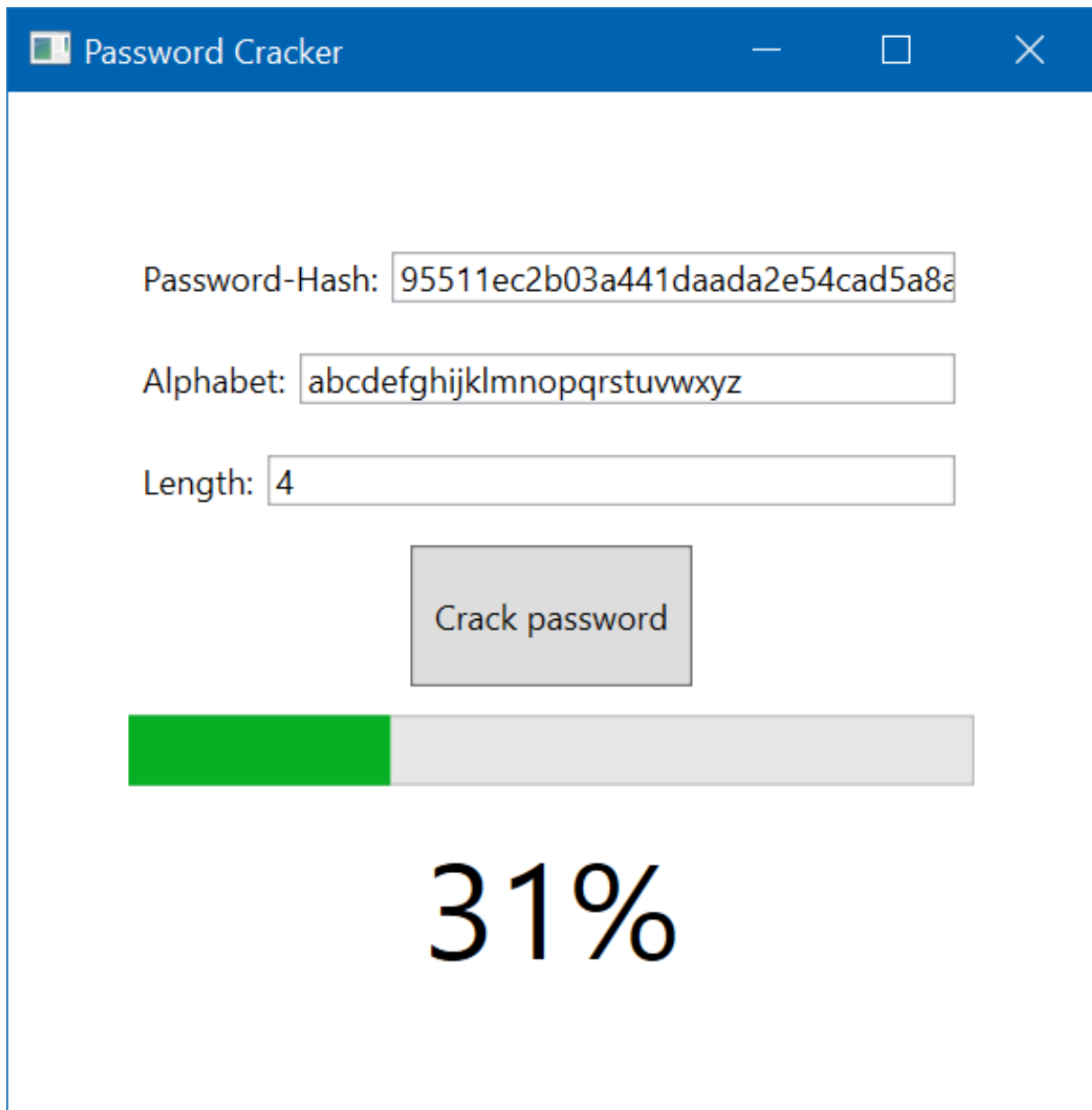
Crack password

???

HINWEIS: Sie können den Verbindungsausbau auch mit einem Button umsetzen bzw. eine Meldung aufpoppen lassen und das Programm wieder beenden.

Konnte eine Verbindung hergestellt werden, kann der Benutzer die gewünschten Daten eingeben und die Anfrage an den SignalR-Server senden.

Der SignalR-Server soll jede Sekunde den aktuellen Fortschritt in Prozent an den Client zurückliefern.



The screenshot shows a web application titled "Password Cracker". It features three input fields: "Password-Hash:" with the value "95511ec2b03a441daada2e54cad5a8a", "Alphabet:" with the value "abcdefghijklmnopqrstuvwxyz", and "Length:" with the value "4". Below these fields is a grey button labeled "Crack password". Under the button is a progress bar that is 31% full, indicated by a green segment. Below the progress bar, the text "31%" is displayed in a large font.

Wurde ein Ergebnis vom Server empfangen, setzen Sie die Progressbar auf 100% und zeigen Sie den empfangenen Text an (konnte das Passwort gefunden werden, wird dann entsprechend das gefundene Passwort angezeigt):

Password Cracker

Password-Hash: 95511ec2b03a441daada2e54cad5a8a

Alphabet: abcdefghijklmnopqrstuvwxyz

Length: 4

Crack password

* no match *