Kryptering

Dag 4 : Symetrisk/asymetrisk kryptering.

Indhold:

Faglige mål:

Dagens øvelse dækker følgende målpinde:

1. *Eleven kan udvikle serverside webapplikationer, der kan levere HTML-kode til browseren, samt Web API eller webservices, som kan udveksle data med en client-application, f.eks. en browser eller en mobil App.*
2. *Eleven kan redegøre for forskellige arkitekturer for web Applikationer og web API (web Services), med fordele og ulemper.*
3. ***Eleven kan opbygge og konfigurere en web Application og web API (web service) vha. et framework.***
4. *Eleven kan benytte validering af brugerinput i en web Applikation.*
5. *Eleven kan implementere passende ViewModels eller DTO klasser.*
6. *Eleven kan anvende Unit Test og mocking af objekter.*
7. *Eleven kan konfigurere routing i en applikation.*
8. ***Eleven kan udvide en applikation med en database, evt. med et ORM-framework.***
9. *Eleven kan programmere services til brug for en applikation, f.eks. data- og logging-services. r*
10. *Eleven kan benytte en hensigtsmæssig strategi for Exception handling.*
11. ***Eleven kan implementere sikkerhed og brugeradministration i en web applikation.***
12. *Eleven kan udrulle (deploy) en applikation, både On-Premises og Cloud baseret.*
13. *Eleven kan udføre Parallel Programmering.*
14. **Eleven kan redegøre for fordele/ulemper ved forskellige teknikker inden for Cryptography.**
15. **Eleven kan anvende Hashing, Symmetric og Asymmetric Encryption**

Øvelse:

Baggrund:

(se afsnit "**Symetrisk/asymet risk kryptering difinition"** på side 5)

Krav:

1. **Opret symetrisk kryptering**:
   - Opret en back-end **klasse** i din web app til håntering af symetriskkryptering. Din klasse skal indeholde:
     - PrivateKey felt
       - som **SKAL** være genereret ved brug af: **System.Security.Cryptography.RSA.**
     - En kryptering metode.
     - En dekryptering metode.

     Denne back-end klasse **SKAL** bruge **dependency injection** teknikken til kommunikation med din view.

     Hvis alle punkter foroven ikke er opfyldt, vil dette tælle som en væsenlig mangel.

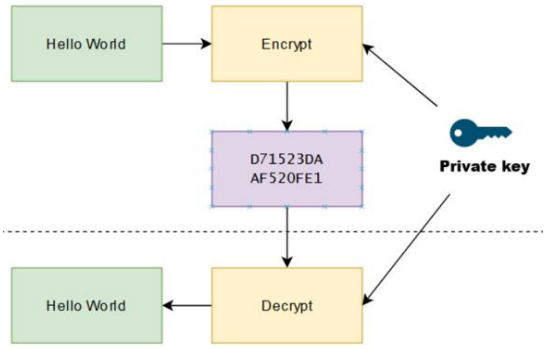
2. **Opret asymetrisk kryptering**:
   - Opret en back-end **klasse** i din web app til håntering af symetriskkryptering. Din klasse skal indeholde:
     - Denne back-end klasse **SKAL** implementer:
       - En privateKey felt
         - som **SKAL** være genereret ved brug af: **System.Security.Cryptography.RSA** og med RSA's **ExportRSAPrivateKey()** metode.
       - En publicKey felt
         - som **SKAL** være genereret ved brug af: **System.Security.Cryptography.RSA** og med RSA's **ExportSubjectPublicKeyInfo()** metode**.**
       - En dekryptering metode.
       - En kryptering metode, **men metoden skal ikke selv implementer logik til kryptering, men kalder i stedet en service på en Web API som fortager kryptering,** og returner det krypteret tekst tilbage til din krypterings metode som kalder den. Derfor skal du:
         - Opret en **WebAPI** som en sekundær og uafhængige applikation. Din API skal indeholde:
           1. **Kryptering metode.**
              - Denne API methode skal kunne modtag en PublicKey som parameter fra din serverside web app, samt to-do item tekst som den skal krypteres.
              - Metoden skal returner det krypteret tekst tilbage til din serverside web app som gemmer det i din database.
              - Din serverside web app skal kunne dekrypter det krypteret data med sin **PrivateKey** når data vises i dens view.
   - Denne back-end klasse **SKAL** anvend **dependency injection** teknikken til kommunikation med view.
   - Hvis alle punkter foroven ikke er opfyldt, vil dette tælle som en væsenlig mangel.

3. Af de 2 oprettet symetrisk og asymetrisk kryptering klasser du har implementeret, **SKAL** du brug **asymetrisk** kryptering til kryptering/dekryptering af din to-do items.
    - Din web app **SKAL** kunne krypter/dekrypter også efter du genstarter din editor (Visual Studio, eller det du bruger).
        - I skal her hjælpe hinanden internt hvis nogen ikke kan krypter/dekrypter når editor genstartes.
        - I skal også internt diskuter og find frem til, hvornår i skal sende data til kryptering og hvornår i sender data til dekryptering ud fra følgende best practis: **Data skal krypteres så tidligt som muligt, og dekrypteres så sent som muligt.**

Hvordan du bliver bedømmet

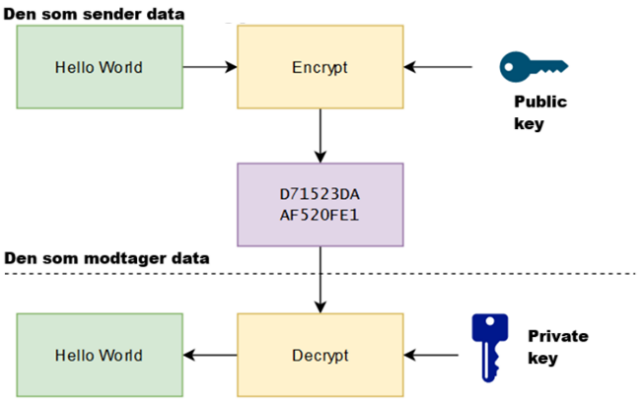
- Du skal kunne køre/fremvis din applikation fejlfrit på din browser med https:// prefix, men nu skal du kunne vise, at to-do items er krypteret med asymetrisk kryptering i den database, men stadig vises som ren tekst i din view.
- Du skal kunne genstart din Visual Studio (eller det editor du bruger) og vise at det krypteret data i databasen stadig vises som ren tekst i din view når applikation køres.
- Brug punkterne under krav som check list, at din kode opfylder de stillede krav.

Symetrisk/asymetrisk kryptering difinition



Symetrisk kryptering: Der anvendes kun en nøgler

Private key anvendes til kryptering og de-kryptering



Asymetrisk kryptering: Der anvendes to nøgler. Public key anvendes til kryptering, mens private key anvendes til de-kryptering