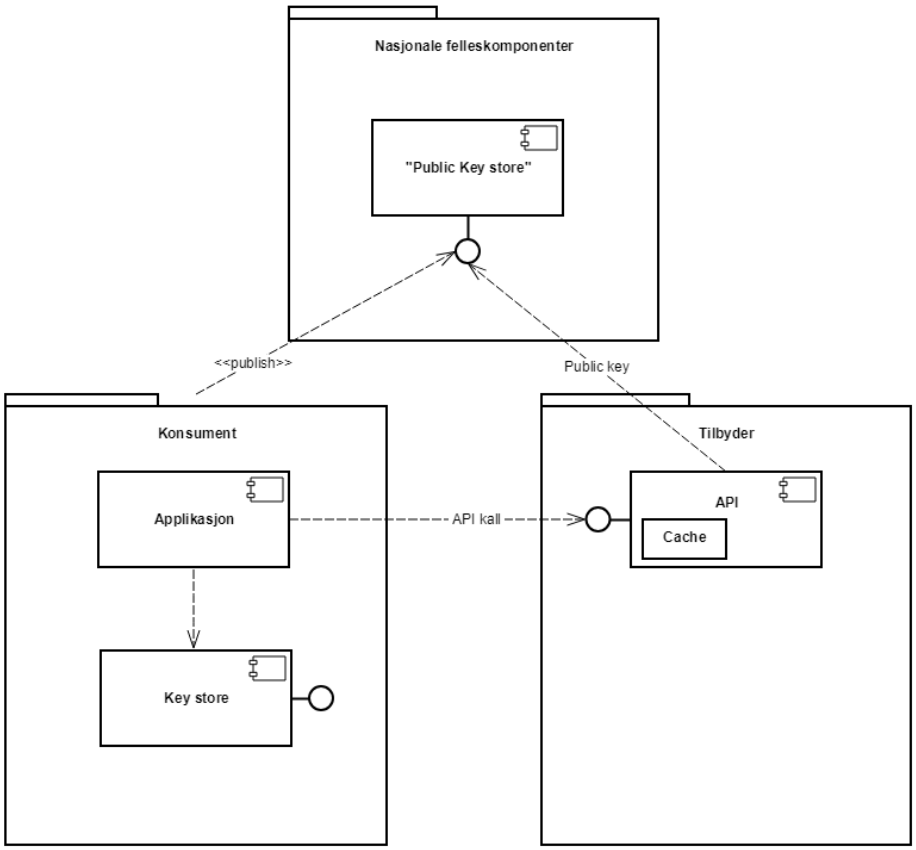
Kryptering (eOppslag) - underlag

Forutsetninger

- Eksemplene forutsetter at det kun er responsen som skal krypteres, som er mest sannsynlig scenario gitt eOppslags-mønster
- Asymmetriske nøkler benyttes for krypteringen

Alternativ 1 - Bruk av felles "sertifikattjeneste"/BCP



Forutsetning: Konsument har publisert sin offentlige nøkkel for kryptering i "Public key store".

1. Konsumenten gjør eOppslag-forespørsel til en API tilbyder hvor det er bestemt at responsen skal være kryptert*
2. Tilbyder henter ut den offentlige nøkkelen som skal brukes for kryptering fra fellestjenesten for offentlige nøkler.
    a. Det vil være naturlig at tilbyder holder en lokal cache over nøklene slik at det ikke blir behov for å slå opp for hver melding som skal krypteres, den må invalideres med jevne mellomrom.
3. Tilbyder bruker nøkkelen som er oversendt til å kryptere responsen, og besvarer forespørselen
    a. Det kan være hensiktsmessig at tilbyder oppgir hvilken nøkkel som har blitt benyttet for krypteringen, slik at konsument får vite om det er gjort endringer (støttes f.eks. i JWE - https://tools.ietf.org/html/rfc7516 4.1.6)
4. Konsumerende applikasjon benytter privat nøkkel til å dekryptere responsen

➕ Sentralisert håndtering av nøkler gjør det enklere å forholde seg til for konsumenter

➕ Kan totalt sett gi en sikrere løsning dersom kompleksitet sentraliseres til et miljø som er gode på dette

➕ En del av kontrollen av sertifikatene kan håndteres sentralt

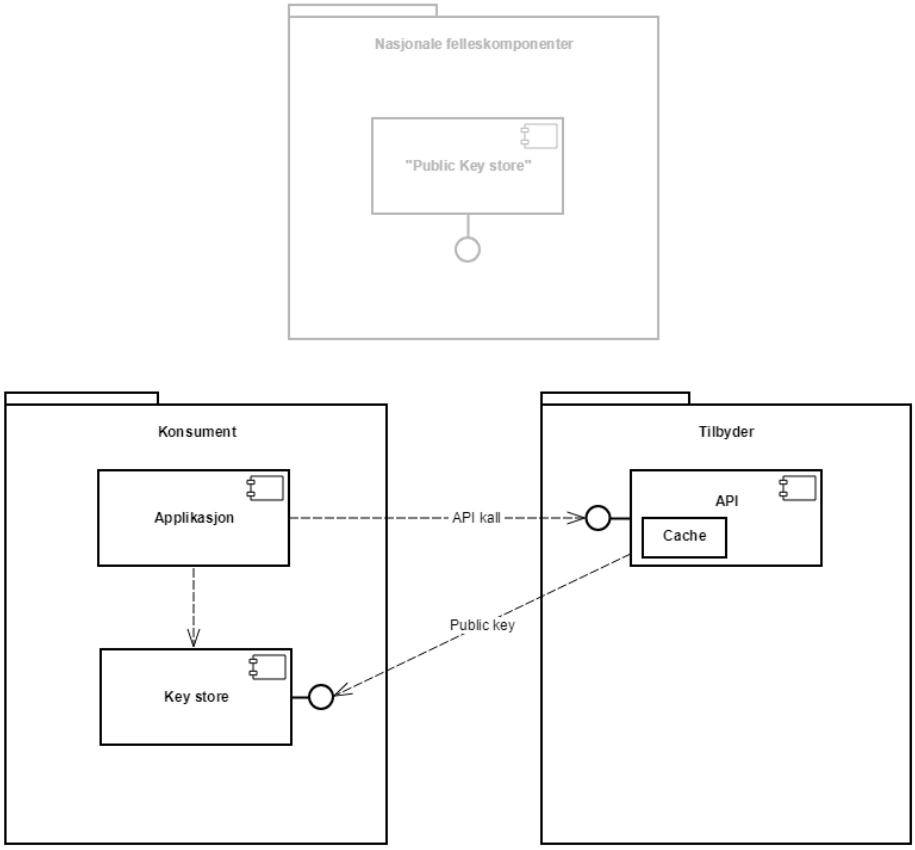
➕ Muliggjør caching hos tilbyder

➖ Krever at konsumentene vedlikeholder nøklene som skal benyttes, større sjanse for at informasjon er usynkronisert

➖ Flere parter i verdikjedene, med de problemstillingene det medfører mht. økt kompleksitet i løsningene, nedetid på felleskomponentene osv

➖ Konsument og tilbyder blir nødt til å stole på en 3. part, som også vil kunne være gjenstand for sikkerhetshull

Alternativ 2 - "Desentralisert" sertifikattjeneste



Forutsetning: Konsumenten gjør det kjent for tilbyderen hvor deres offentlige nøkler kan hentes ut, for eksempel gjennom at endepunktet publiseres i et sentralt adresseregister ("Capability lookup").

1. Konsumenten gjør eOppslag-forespørsel til en API tilbyder hvor det er bestemt at responsen skal være kryptert*
2. Tilbyder henter ut den offentlige nøkkelen til konsumenten som skal brukes for kryptering fra et endepunkt som konsumenten eksponerer
3. Tilbyder bruker nøkkelen som er oversendt til å kryptere responsen, og besvarer forespørselen
4. Konsumerende applikasjon benytter privat nøkkel til å dekryptere responsen

*) I REST kan dette også løses gjennom Content negotation ved at konsumerende applikasjon oppgir hvilket format som man ønsker på responsen, i dette tilfellet en kryptert respons - eks. application/jose+json.

➕ Kun konsument og tilbyder i verdikjeden, mindre kompleksitet i løsningen

➕ Kun et key store gjør risikoen mindre for at informasjonen er usynkronisert

➕ Muliggjør caching av nøkler hos tilbyder

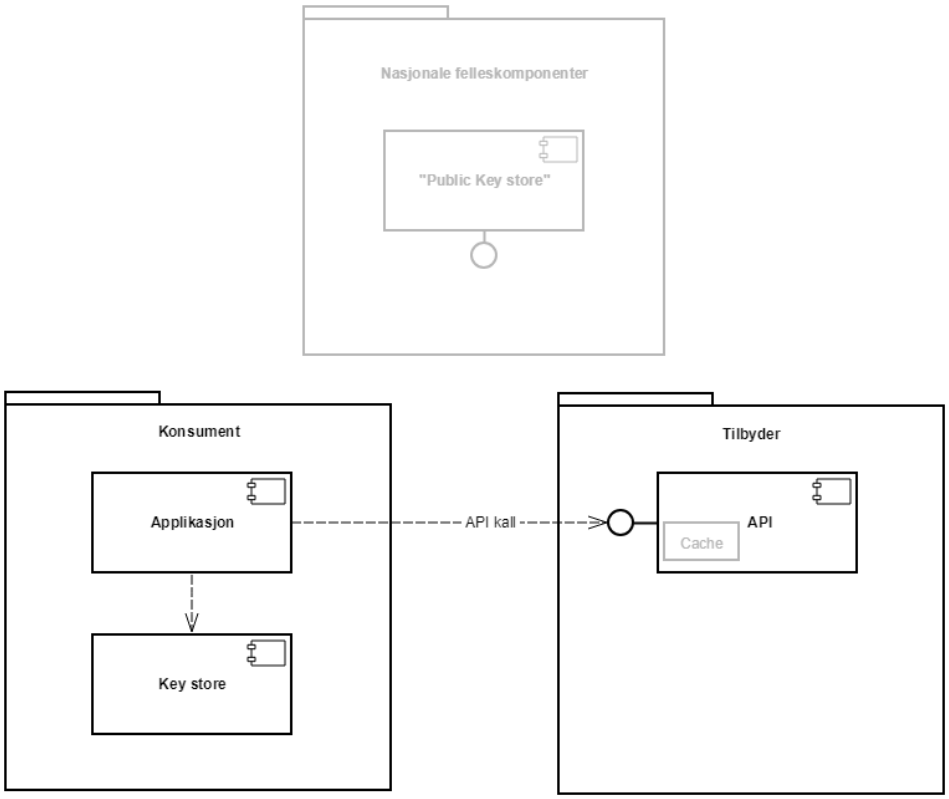
➕ Kan gjøres på en standardisert måte, f.eks. JWKS

➕ Forenkler hyppigere utskifting av nøkler

➖ Hver konsument må eksponere et eget endepunkt for nøkler,

➖ Hver tilbyder må forholde seg til at hver konsument sitter med eget sett av nøkler

Alternativ 3 - Oversendelse av key



1. Konsumenten oversender den offentlige nøkkelen som skal brukes for kryptering i hvert kall
2. API tilbyder bruker nøkkelen som er oversendt til å kryptere responsen
3. Konsumerende applikasjon benytter privat nøkkel til å dekryptere responsen

➕ Konsument og tilbyder er sikre på hvilken nøkkel som benyttes

➕ Forenkler hyppigere utskifting av sertfikater/rotering av nøkler

➖ Veldig ineffektivt å oversende nøkkelen i hvert kall, dette er ganske statisk informasjon som endres sjelden, og nøklene er også såpass store at de kan utgjøre en betydelig del av HTTP-responsen

➖ Ingen mulighet for caching av nøklene

➖ Det finnes ingen standardisert måte å overføre denne typen informasjon på i en forespørsel

➖ Sårbar for man-in-the-middle angrep gjennom at nøklene kan byttes ut av angriperen

Viser ellers til valgene som er gjort for kryptering i DSOP Kontroll (Konto)-prosjektet:

- Bruk av JWE (Json Web Encryption) - kryptering av innhold på en JSON-basert struktur
- Relevante algoritmer knyttet til krypteringen fra JWA (typisk 'Recommended')