Retningslinjer for datadeling i kunnskapssektoren

Tilgangsstyring og tilgangskontroll

Versjon 0.95

Arbeidsdokument for versjon 0.95

Dette er gjeldende versjon per 11.05.2024

INNHOLDSFORTEGNELSE

INTRODUKSJON

I 2022 ble "Referansearkitektur for deling av data i høyere utdanning og forskning" levert gjennom Datadelingsprosjektet hos Unit. Hensikten var å legge grunnlaget for samhandling i kunnskapssektoren gjennom datadeling. HK-dir forvalter referansearkitekturen.

Dette dokumentet konkretiserer elementer fra referansearkitekturen ved å beskrive en målarkitektur. Målarkitekturen består av et sett med retningslinjer for sikker datadeling.

Krav som stilles til målarkitekturen er at den skal

- Være basert på beste praksis fra norsk offentlig sektor og tilsvarende domener.
- Være basert på åpne standarder som er i bred bruk.
- Tilfredsstille sikkerhetskrav som stilles for deling av data.
- Være så konkret at den er nyttig for etablering av praktiske tekniske og ikke-tekniske tiltak.
- Være gjennomførbar på sektornivå. Dette innebærer at virksomheter og leverandører med forskjellig teknisk kompetanse og infrastruktur skal kunne forstå og etterleve målarkitekturen.
- Være mulig å etablere med eksisterende felleskomponenter.

De viktigste målgruppene for målarkitekturen er

- Eiere og leverandører av
  - Tjenester som konsumerer data.
  - Tjenester som deler data. En tjeneste kan levere data fra flere dataeiere[1].
- Tekniske arkitekter, prosjektledere og utviklere som jobber med
  - API-er.
  - Konsumering av API-er.
  - Fellestjenester og komponenter.
- Andre interessenter av sømløs og sikker datadeling i offentlig sektor.

For å avgrense omfanget til målarkitekturen, er det satt rammer for hvilke former for datadeling den gjelder. Kort oppsummert beskriver den tilgjengeliggjøring og sikring av HTTP-baserte API-er. Mer detaljert innebærer dette at

- Målarkitekturen omhandler primært tilgangsstyring og tilgangskontroll.

[1] Altså behandlingsansvarlige i personopplysningsloven.

- Målarkitekturen fokuserer på teknisk og semantisk samhandlingsevne relatert til tilgangskontroll.
- Juridisk og organisatorisk samhandlingsevne er viktig for tilgangsstyring, men diskuteres ikke i detalj.
- Målarkitekturen fokuserer på synkron datadeling over HTTP, men retningslinjene for tilgangsstyring vil også gjelde asynkrone tjenester.
- Integrasjonsplattformer er ikke omtalt i målarkitekturen, men de kan opptre som datakonsumenter og datatilbydere.

Deling av data innad i virksomheter, og andre tilhørende interne prosesser beskrevet i Orden i eget hus, er i utgangspunktet ikke omfattet av retningslinjene i arkitekturen. Det er likevel ikke noe i veien for å bruke retningslinjene internt der det er hensiktsmessig.

Ordbruk

Datadeling er et komplisert tema, og preges av en blanding av tekniske, juridiske og domenespesifikke termer på både norsk og engelsk. I dette dokumentet er det gjort noen valg basert på målgruppen. "Vedlegg - Ordliste" beskriver de viktigste begrepene som er brukt, og grupperer dem med overlappende – men sjelden synonyme – ord som brukes i samme kontekst.

Tilnærming

Kunnskapssektoren har allerede et komplekst økosystem for datadeling. Dette har vokst frem organisk over tid. For å kunne beskrive en målarkitektur innenfor rammene som er satt, har det vært viktig å kartlegge dagens situasjon godt.

Det har blitt gjennomført intervjuer med et bredt sjikt av interessenter. Dette inkluderer både datakonsumenter, datatilbydere og leverandører av tjenester og felleskomponenter. Personene som har blitt intervjuet har besittet ulike roller som produkteiere, utviklere, driftspersonell, IT-ansvarlige og annet.

Det er også gjennomført samtaler med ansvarlige for eksisterende tillitstjenester i offentlig sektor[2] for å sikre at målarkitekturen er i tråd med beste praksis.

Det har vært et mål å gjøre målarkitekturen så konsis som mulig. Utdypende informasjon er derfor inkludert i vedlegg.

[2] ID-porten / Maskinporten og HelseID

Generelle prinsipper for datadeling over HTTP

En stor del av datadeling på Internett skjer i dag over HTTP. Det er derfor nødvendig å forstå HTTP og hvordan denne protokollen benyttes.

"Vedlegg – Datadeling med HTTP" foreslår noen overordnede prinsipper for sikker datadeling over HTTP. Prinsippene er generelle, men er et godt utgangspunkt for å forstå retningslinjene i målarkitekturen. Vedlegget er ment å kunne leses frittstående, og inneholder derfor en del informasjon som diskuteres andre steder i dette dokumentet.

DAGENS SITUASJON

Felles komponenter, datakilder og tjenester

Under beskrives noen av de sentrale komponentene som benyttes for datadeling i kunnskapssektoren i dag, og hvordan de er tatt i bruk for å etablere «Felles IAM». Kommende tjenester som «UH:Sak» bygger på tilsvarende løsningsmønster.

Felles IAM benytter flere nasjonale datakilder og samsvarer med referansearkitekturen. Som vi vil se er det likevel et spenn i hvordan de forskjellige integrasjonene etableres og forvaltes, noe som fører til høy kompleksitet.

Felles arkitekturkomponenter

Feide og Feide Kundeportal

Feide er en nasjonal løsning for sikker identifisering og autentisering innen utdanningssektoren i Norge. Identiteten Feide tilbyr knytter en person til en virksomhet hvor personen har en rolle. Feide tilbyr funksjonalitet for sluttbrukers samtykke, altså at en person samtykker til å dele personlig informasjon med ulike tjenester.

En annen funksjon som tilbys i Feide er tilgjengeliggjøring av grensesnitt for datadeling og API-er, som tillater ulike tjenester å kommunisere og dele data på en strukturert og sikker måte. Dette inkluderer en tjenestekatalog og tilhørende funksjonalitet rundt tilgangsstyring.

Feide Kundeportal brukes av vertsorganisasjoner og tjenesteleverandører. Portalen er dedikert til konfigurering av felles tjenester og API-er. I tillegg til å tilby disse tjenestene, fungerer kundeportalen som et støtteverktøy for andre aspekter av datadeling som for eksempel å hjelpe med opprettelsen av databehandleravtaler.

Feide og Feide Kundeportal leveres av Sikt, nasjonal tjenesteleverandør for kunnskapssektoren.

Fellestjenesten for Datadeling (UH:IntArk)

UH:IntArk - Fellestjeneste for datadeling - er en felles tilnærming til datadeling i kunnskapssektoren. IntArk er basert på nasjonal referansearkitektur, og består av ett sett med prinsipper for datadeling.

IntArk tilbyr også en teknisk plattform for håndtering av API-er og tilhørende notifikasjoner til virksomheter innen høyere utdanning og forskning. Plattformen

består av en API Gateway og en meldingskø[3] med tilhørende verktøy for administrasjon og tilgangsstyring.

Felles datakilder

Felles Studentsystem og DFØ SAP er eksempler på sentrale datakilder som lagrer og leverer data på vegne av virksomheter. Leverandørene av datakildene opptrer altså som databehandler på vegne av de behandlingsansvarlige virksomhetene.

Mange virksomheter har valgt å etablere egne API-proxyer foran de sentrale datakildene. Årsaken til dette er behov for å kunne styre tilgang til data virksomhetene er behandlingsansvarlig for. I tillegg får man bedre mulighet for etterkontroll når data er utlevert. Dette er et mønster som brukes i «Felles IAM».

Felles Studentsystem

Felles Studentsystem (FS) er et studentinformasjonssystem som brukes av de fleste universiteter og høyskoler i Norge. Data fra FS tilgjengeliggjøres gjennom et sett med felles HTTP API-er.

Tilgang til disse er basert på brukernavn og passord som deles ut til virksomheter som ønsker å etablere system-til-system integrasjoner.

Neste versjon av FS er basert på GraphQL med bruk av OAuth2 for tilgangsstyring. Det antas at Feide skal brukes til tilgangsstyring og at FS kan kreve informasjon om pålogget sluttbruker.

DFØ SAP

Direktoratet for forvaltning og økonomistyring (DFØ) tilbyr lønns- og HR-tjenester til kunnskapssektoren og andre deler av offentlig sektor. Data tilgjengeliggjøres gjennom HTTP API-er. Tilgang til API-ene er styrt av Maskinporten og Samarbeidsportalen til Digdir, med OAuth2 som sikkerhetsprotokoll.

Felles tjeneste: Identitets- og tilgangsstyring

"Felles IAM" er en fellestjeneste for identitet- og tilgangsstyring i kunnskapssektoren. Den benytter eksisterende komponenter som IntArk og Feide, kombinert med andre produkter, for å håndtere forvaltning av brukere og tilganger innad i virksomheter.

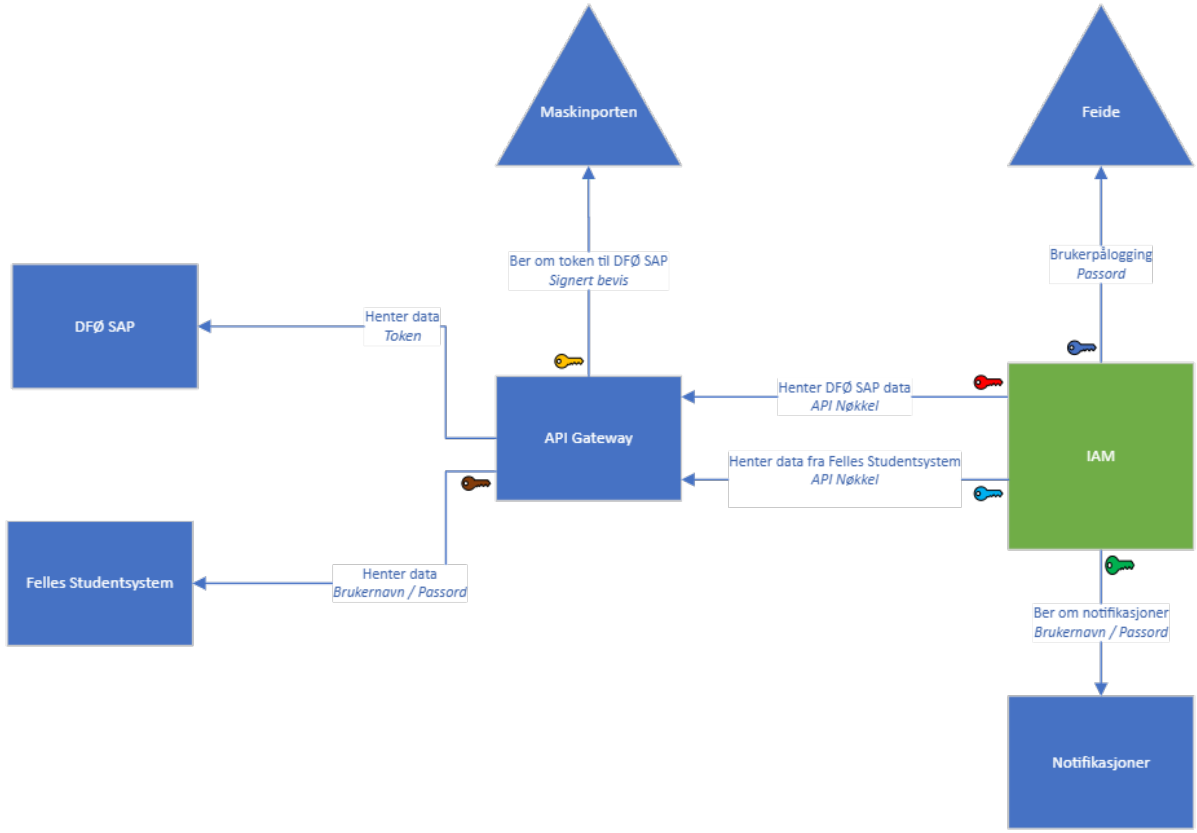
Felles IAM anskaffes og tilgjengeliggjøres per virksomhet, og krever egne integrasjoner mot API-er som DFØ-SAP og Felles Studentsystem. Hver instans av IAM krever separate oppsett per integrasjon med tilhørende forvaltning, inkludert notifikasjoner.

For å tilgjengeliggjøre felles datakilder for konsum av IAM, settes det opp egne integrasjoner fra API Gateway til datakilde. Deretter eksponeres disse til sentral IAM-

[3] Gravitee og RabbitMQ

løsning med egne sikringsmekanismer. Dette gjøres også per instans av API Gateway-en.

Figur og tabellen under oppsummerer hvordan de forskjellige komponentene og tjenestene sikres.



Figur 1 - Datadeling i Felles IAM. Hver nøkkel indikerer en unik hemmelighet.

	Autentiserings-mekanisme	**Sikkerhets-mekanisme**	**Tilgangsstyring**
Brukerpålogging	Passord	OAuth2	Feide Kundeportal
Felles Studentsystem API	Brukernavn og Passord	Basic Authentication	Manuelle prosesser
Maskinporten / DFØ SAP API	Privat nøkkel / Token	OAuth2	Samarbeidsportalen
API Gateway FS	API nøkkel	Produktspesifikk HTTP header	Manuelle prosesser og egen selvbetjeningsløsning.
API Gateway DFØ SAP	API Nøkkel	Produktspesifikk HTTP header	Manuelle prosesser og egen selvbetjeningsløsning
Notifikasjoner	Brukernavn og passord	AMQP / Basic Authentication	BROM Selvbetjening

Utfordringer

Autentisering, sikkerhet og tilgangsstyring håndteres på ulik måte i forskjellige løsninger i sektoren. Dette fører til flere utfordringer:

- Datakilder tilgjengeliggjøres i forskjellige brukergrensesnitt, og hos forskjellige aktører.
- Forskjellige sikkerhetsmekanismer er i spill, og samme mekanismer brukes på forskjellig måte.
- Hemmeligheter tildeles per datakilde en datakonsument benytter. Hver enkelt hemmelighet har gjerne forskjellige egenskaper, og må forvaltes forskjellig av datakonsumenten.
- Identifisering av datakonsument, spesielt person og virksomhet, håndteres forskjellig.
- Sporbarhet i lengre verdikjeder er problematisk.

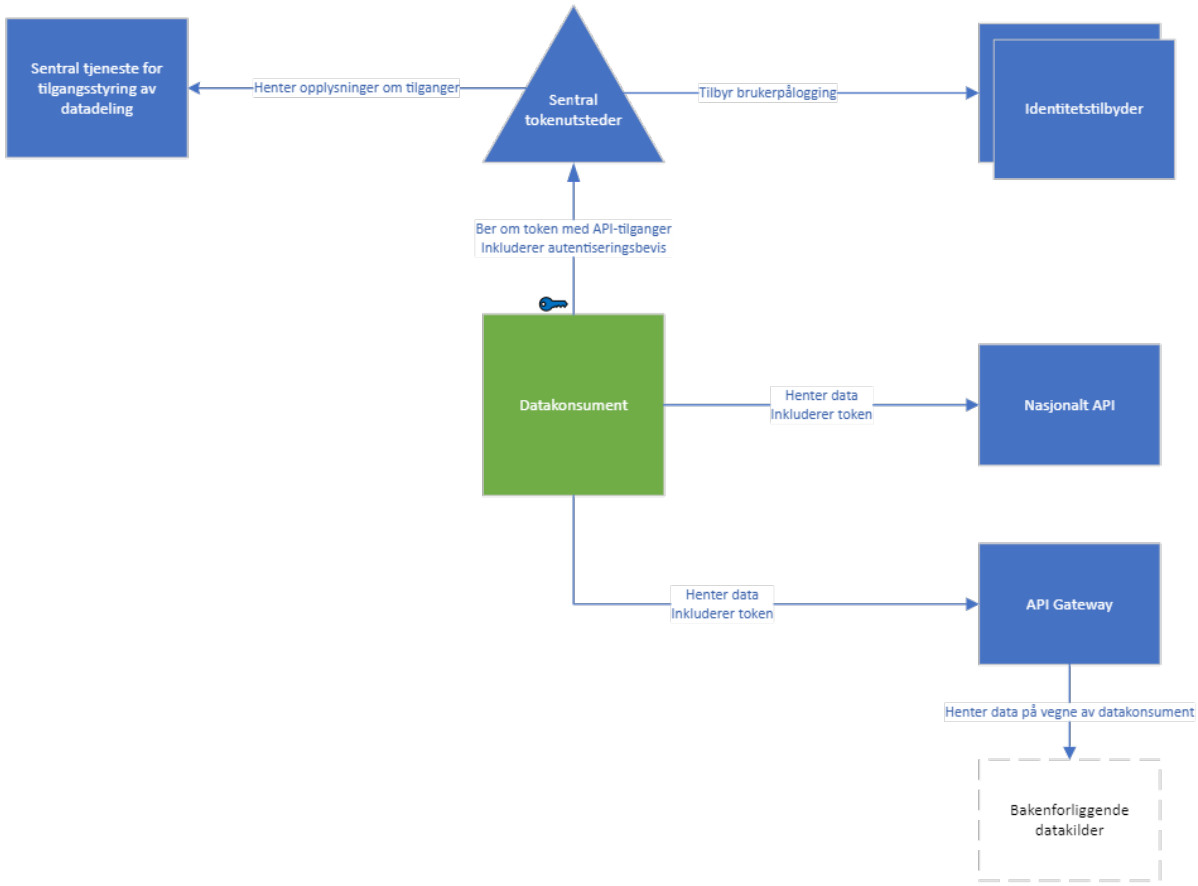
En datakonsument med flere datakilder må ofte håndtere en «hemmelighet» per datakilde. Policy for hemmeligheter (prosess for utdeling, levetid, type hemmelighet) varier avhengig av kilden som deler ut hemmeligheter og må håndteres individuelt.

Oppsummert fører dette til mer friksjon for datadeling. Spesielt datakonsumenter rammes av komplekse oppsett og tilhørende forvaltningsbehov, noe som gir høy risiko for sikkerhetsavvik og høyere kostnader.

Målarkitekturen adresserer disse utfordringene. «Vedlegg – Bruk av Målarkitektur i Felles IAM» viser en tenkt løsningsarkitektur for Felles IAM basert på målarkitekturen.

MÅLARKITEKTUR FOR DATADELING I KUNNSKAPSSEKTOREN

Målarkitekturen består av fem retningslinjer. Disse er beskrevet i de følgende kapitlene. Hvert kapittel gir et kort underlag og avsluttes med selve retningslinjen.



Figur 2 - Målarkitektur for datadeling i kunnskapssektoren

Figuren over viser hvordan målarkitekturen ser ut fra perspektivet til en datakonsument.

Retningslinje 1 - Bruk OAuth2 for datadeling over HTTP

OAuth2 skal benyttes for sikring av synkrone HTTP-baserte datakilder som deler data på tvers av virksomheter i kunnskapssektoren.

Bakgrunn

OAuth2 er en internettstandard for tilgangsstyring til HTTP-baserte API-er[4]. Protokollen har gjennomgått betydelig modning etter første versjon, og er utvidet med mekanismer som gjør den egnet for bruk i domener med deling av taushetsbelagte opplysninger eller som har høye krav til sikkerhet.

Det finnes alternativer for sikring av HTTP-baserte API som for eksempel API-nøkler og mTLS. Dette er punkt-til-punkt mekanismer som skalerer dårlig, og som i forskjellig grad er standardisert.

I Norge og internasjonalt er OAuth2 i utstrakt bruk for datadeling i både offentlig og privat sektor.

Standarden OpenID Connect er et påbygg på OAuth2 som gir mulighet for føderert brukerautentisering i applikasjoner. Ved å støtte begge protokollene i samme tjeneste kan man tilby

- føderert autentisering
- støtte for API-er som krever høy tillit til informasjon om virksomhet
- støtte for API-er som krever høy tillit til informasjon om sluttbruker
- støtte for API-er som krever høy tillit til informasjon om både virksomhet og sluttbruker

De to siste punktene omtales også som "brukerstyrt datadeling".

Utbredelsen av OAuth2 har ført til et bredt spekter av tilgjengelige programmeringsbibliotek for ulike programmeringsspråk og tekniske plattformer. Dette forenkler utviklingen av programvare som bruker OAuth 2.0 protokollen fordi man får tilgang til verktøy som sikrer smidige og konsistente integrasjoner.

[4] Andre protokoller som AMQP og SMTP har egne regimer for tilgangskontroll. OAuth2 støttes av enkelte AMQP-baserte produkter, blant annet RabbitMQ.

Retningslinje 2 - Harmoniser bruk av OAuth2

Sikt skal etablere og forvalte detaljerte profiler for bruk av OAuth2 i kunnskapssektoren.

Bakgrunn

OAuth2 er et åpent protokollrammeverk som er tilrettelagt for utvidelser basert på nye behov. Disse utvidelsene er egne spesifikasjoner. Disse spesifikasjonene har blitt introdusert for å tette kjente svakheter eller hull rundt sikkerhet, tilpasse protokollen til moderne plattformer, og for å møte krav fra aktører som ønsker å dele spesielt sensitiv informasjon.

Denne kompleksiteten gjør det utfordrende å anvende OAuth2 på "riktig" måte innenfor en spesifikk kontekst.

Med bakgrunn i dette anbefales det å detaljere bruk av OAuth2 i kunnskapssektoren. En slik detaljering, heretter kalt "profil," bør følge beste gjeldende praksis, samtidig som den er tilpasset behovene til sektoren.

Det er viktig at profilen unngår å henvise til protokollmekanismer som ikke er standardiserte eller utbredte. Dette vil sikre at sertifiserte tredjepartsbiblioteker kan benyttes, noe som øker kvaliteten og dermed sikkerheten til integrasjoner.

"Vedlegg - Forslag til profil for bruk av OAuth2" inneholder et detaljert forslag til tre profiler – "Lav", "Normal" og "Høy". Profilene er delt opp på denne måten for å støtte konfidensialitetsklassene for informasjon beskrevet i veilederen "Klassifisering av informasjon og informasjonssikkerhet".

Profilene anbefaler bruk av JSON Web Token (JWT) som token-format. Mer informasjon om JWT-er og konsekvensene av bruk av disse finnes i "Vedlegg - JSON Web Token som token-format".

Spesielt viktig er hvordan identitetene til datakonsumenter uttrykkes overfor datakilder. Dette inkluderer fysiske og juridiske personer. Det er skrevet mer om dette her.

Retningslinje 3 - Etabler en sentral tokenutsteder

Sikt skal etablere og forvalte en sentral tokenutsteder for å støtte HTTP-basert datadeling på tvers av virksomheter i kunnskapssektoren.

Bakgrunn

En sentral tokenutsteder for kunnskapssektoren vil være i tråd med gjeldende løsninger ellers i offentlig sektor som ID-porten, Maskinporten og HelseID i helsesektoren.

Sentralisering av en slik tjeneste reduserer antall tillitsforhold som datakonsumenter og datakilder i virksomhetene må etablere for datadeling. Dette reduserer den totale kompleksiteten for leverandører og utviklere, og bidrar til et mer robust økosystem.

Dette kan på sikt forenkle kommunikasjonen på tvers av ulike deler av offentlig sektor, som igjen kan gi bedre tjenester til innbyggerne.

En slik arkitektur legger grunnlaget for en felles plattform for API-sikkerhet, noe som gjør det enklere å implementere støtte for protokoller, raskt tette nye sikkerhetshull, og etablere en samlet oversikt over tilgjengelige datakilder.

Andre fordeler med arkitekturen er:

- Økt grad av standardisering, både teknisk og ikke-teknisk.
- Mulighet for bedre overvåkning og etterkontroll.
- Redusering av sårbarheter og kostnader forbundet med en fragmentert sikkerhetsinfrastruktur.

Det er viktig å merke seg at en sentral tokenutsteder også kan brukes for API-er som kun konsumeres internt i organisasjoner, men man må være klar over hvilke følger dette har. For eksempel vil det introdusere en avhengighet til den sentrale tokenutstederen, med de konsekvenser dette har for tilgjengeliggjøring av interne datakilder.

En sentral tokenutsteder må støtte protokollmekanismene som er beskrevet i profilene for bruk av OAuth2 i kunnskapssektoren.

Retningslinje 4 - Etabler en sentral tjeneste for tilgangsstyring av datadeling

Sikt skal etablere og forvalte en sentral tjeneste for å støtte tilgangsstyring for sentral tokenutsteder.

Tjenesten bør støtte automatisk synkronisering av informasjon nødvendig for tilgangskontroll i API Gateway-er hos den enkelte virksomhet.

Bakgrunn

En sentral tokenutsteder krever at det eksisterer konfigurasjonsdata for tilgangsstyring til datakilder. Denne informasjonen vil beskrive hvilke datakilder en datakonsument har tilgang til. Det anbefales å etablere en dedikert tjeneste som gir virksomheter og leverandører mulighet til å selv konfigurere slik informasjon. Tjenesten må ha et brukergrensesnitt, men bør også kunne automatiseres for enkelte behov.

Dette dreier seg om tilgangsstyring på funksjonelt nivå, det vil si hvilke datakonsumenter[5] som kan benytte hvilke datakilder. Hver datakilde har selv ansvaret for å minimere utleverte data. Datakilden skal basere en slik filtrering på informasjon som sendes som parametere, og eventuelt bekreftes av informasjon utledet fra token[6].

Tilgangsstyringstjenesten må som et minimum støtte:

- Sikker relasjon mellom innloggede brukere i tjenesten, og virksomheten(e) de representerer.
- Registrering og konfigurering av API-er.
- Registering av klienter av følgende type:
    - Med støtte for brukerpålogging.
    - Uten brukerpålogging[7].
- Registrering (offentlige nøkler) eller opprettelse (passord) av klienthemmeligheter. Disse skal brukes til autentisering av klienter hos tokenutstederen, og kan brukes for virksomhetsautentisering[8].
- Godkjenning av forespørsler om tilgang til API-er fra datakonsumenter.

---

[5] En «datakonsument» i denne konteksten kan inkludere både virksomheter og applikasjoner / tjenester / systemer / plattformer.

[6] Les Sikkerhetsmodellen for HTTP for mer informasjon om dette.

[7] "System-til-system"

[8] Enten kan hemmeligheten være knyttet opp imot et virksomhetssertifikat, eventuelt har det oppstått knytning mellom hemmeligheten og virksomheten i forbindelse med tilgangsstyring i den sentrale tjenesten.

- Synkronisering av teknisk informasjon nødvendig for tilgangskontroll i API Gateway-er. Dette inkluderer identifikator for API ("audience") og tilgangsnivåer ("scopes").

Når det gjelder klienthemmeligheter gjelder følgende:

- En klient bør bare ha én hemmelighet som brukes til autentisering mot sentral tokenutsteder. Dette vil være tilstrekkelig for å gi klienten tilgang til alle API-er den er godkjent for i sentral tokenutsteder.
    - Ved bytte av hemmelighet kan klienten ha to aktive hemmeligheter i en overgangsperiode.
- Hemmeligheten må ha en begrenset levetid, og ansvarlige for klienten må varsles før hemmeligheten utløper.
- Eier av klienten, datakonsumenten, er ansvarlig for å fornye hemmeligheten.
- Det bør være mulighet for automatisert fornyelse av hemmeligheter før utløpsdato.

En bieffekt av en slik tjeneste vil være at en felles API-katalog som https://data.norge.no/ kan berikes med informasjon om tilgangsstyring. Dette kan være informasjon om hvilke tekniske krav en datakilde setter for tilgang, hvilke tilgangsnivåer den støtter og mer. Nytteverdien av en slik katalog øker betraktelig dersom tilgangsstyringsinformasjon er tilgjengelig.

Om API Gateway-er

En konsekvens av den foreslåtte målarkitekturen er at en API Gateway (heretter "mellomtjener") må kunne stole på token utstedt av sentral tokenutsteder. En mellomtjener må altså være i stand til å validere gyldigheten til et token fra utstederen. Dette innebærer både kontroll av at et token i seg selv er gyldig, for eksempel ved å sjekke en kryptografisk signatur, samt validering av tilhørende informasjonselementer[9].

Slike informasjonselementer inkluderer både standardisert informasjon og domenespesifikk informasjon om tilgangsforespørselen. Bruk av sistnevnte krever at identifikatorer som identifiserer datakilden[10] er kjent av både tokenutstederen og mellomtjeneren.

[9] I et JWT-basert token kalles slike informasjonselementer "claims". Ett eksempel på et claim er ("organisasjonsnummer", 123456789). Denne informasjonen kan et API bruke for å kontrollere at virksomheten med organisasjonsnummer 123456789 skal få tilgang til data som API-et tilgjengeliggjør.

[10] For OAuth2 er standardiserte informasjonselementer "audience" for å identifisere API-et / datakilden, og "scope" for å identifisere tilgangsnivå. Et forenklet eksempel på verdi for "audience" er "sikt:organisasjonsstruktur". Enkle eksempler på verdier for "scope" er "les" og "skriv".

Det anbefales at sentral tjeneste for tilgangsstyring defineres som autorativ kilde for slike identifikatorer.

Mellomtjenere kan konfigureres manuelt for den enkelte datakilde, men støtte for automatisk synkronisering av informasjon nødvendig for tilgangskontroll fra sentral tjeneste til mellomtjener anbefales. Med dette unngår man å sette opp den samme informasjonen både i sentral tjeneste og i administrasjonsgrensesnittet til den enkelte mellomtjener. Dette gjør det enklere å etablere nye datakilder, samt at det blir mindre risiko for feil ved endringer.

Det bør være sporbart fra administrasjonsgrensesnittet i en mellomtjener at en konfigurasjon for en datakilde stammer fra sentral tjeneste. Det bør også være mulighet for å enkelt navigere mellom sentral tjeneste og administrasjonsgrensesnittet i mellomtjeneren for en datakilde.

Mellomtjenere vil eksponere datakilder som bare benyttes lokalt i en virksomhet og ikke er kjent av sentral tjeneste. Det må være et tydelig skille mellom behandling av datakilder som tilgangsstyres av sentral tjeneste og for datakilder som bare styres lokalt av virksomheten.

En mellomtjener kan kalle datakilder i en annen virksomhet på vegne av en datakonsument. Mellomtjeneren må da be om et nytt token med tilgang til denne datakilden. Dette token-et vil ikke inneholde informasjon om opprinnelig datakonsument. Dette kan være problematisk avhengig av kravene til datakilden. Det er skrevet mer om denne problematikken i “Vedlegg - Kjeding av API-kall”.

Retningslinje 5 - Datadeling på tvers av offentlig sektor

Det anbefales å etablere et langsiktig, men konkret, målbilde for datadeling på tvers av offentlig sektor.

Bakgrunn

Datadeling på tvers av offentlig sektor byr i dag på mange av de samme utfordringene man møter innad i kunnskapssektoren. Datakonsumenter må forholde seg til forskjellige modeller for tilgangsstyring, krav til sporbarhet og sikkerhetskrav per datakilde de konsumerer. Dette inkluderer policy-er, systemer og prosesser for tilgangsstyring, hemmelighetshåndtering og mer.

OAuth2 har blitt en viktig protokoll for tilgangsstyring av datadeling i offentlig sektor. Etablerte nasjonale tokenutstedere (med tilhørende tjenester for tilgangsstyring) som støtter OAuth2 inkluderer:

- ID-porten / Maskinporten for offentlig sektor generelt.
- HelseID for helsesektoren.
- Feide for enkelte tjenester i kunnskapssektoren.

Sømløs støtte for datadeling på tvers av offentlig sektor er komplekst. En omforent arkitektur for autentisering og tilgangskontroll gjør det enklere å nå et slikt mål. Målarkitekturen beskrevet i dette dokumentet er i tråd med hva som er etablert i andre deler av offentlig sektor.

Noen løsningsalternativer for å oppnå en mer sømløs dataflyt basert på eksisterende løsninger er beskrevet i "Vedlegg - Datadeling på tvers av offentlig sektor". Å bli enig om en konkret vei fremover krever et godt samarbeid på tvers av de forskjellige fagmiljøene, og må forankres hos respektive eiere og behovshavere.

OPPSUMMERING

Retningslinjene anbefaler at kunnskapssektoren omforenes om bruk av OAuth2 for datadeling på tvers av virksomheter. Gitt standardens kompleksitet skal det utarbeides spesifikke profiler for hvordan protokollen skal benyttes.

For å redusere kompleksiteten for datakonsumenter skal det etableres en sentral tokenutsteder for datadeling på tvers av virksomheter, med tilhørende tjeneste for tilgangsstyring.

Retningslinje er i tråd med tilsvarende tjenester fra Digdir og helsesektoren. Man bør tilstrebe at offentlig sektor jobber videre med en slik tilnærming i fremtiden, slik at datadeling i stort kan forenkles.

Gevinster som kan hentes ut er:

- Lavere kompleksitet for datakonsumenter.
- Standardisering forenkler bruk av open source komponenter og kommersiell programvare.
- Et høyere og mer forutsigbart nivå av sikkerhet.
- En felles informasjonsmodell for tilgangsstyring.
- Mer effektiv utvikling av løsninger for datatilbydere og datakonsumenter.
- En sentral kilde for oversikt og kontroll - både for datakonsumenter og datakilder / dataeiere.
- Forenkling av datadeling generelt i offentlig sektor.

Retningslinjene vil innebære videreutvikling av eksisterende felleskomponenter. Dette er bare gjennomførbart med en stegvis tilnærming og godt samarbeid mellom forskjellige aktører i kunnskapssektoren.

Vedlegg

VEDLEGG – FORSLAG TIL PROFIL FOR BRUK AV OAUTH2

Dette vedlegget inneholder forslag til en enhetlig bruk av OAuth2 i kunnskapssektoren – såkalte «profiler». Det presenteres her tre profiler som er i samsvar med nivåene for konfidensialitetsklassene som er beskrevet i veilederen «Klassifisering av informasjon og informasjonssikkerhet». Sammenhengen mellom profil og konfidensialitetsklasse er beskrevet sist i vedlegget.

Profilene er i stor grad i tråd med «OAuth 2.1» og «OAuth 2.0 Security Best Current Practice», samt «FAPI 2.0» for sensitiv informasjon.

Profilene må også inkludere aksepterte krypteringsalgoritmer for signering og eventuelt kryptering. Denne oversikten er ikke inkludert her.

Kravene gjelder API-er og klientene som konsumerer disse.

Eksplisitte krav til tokenutsteder er ikke inkludert, men den må nødvendigvis støtte de beskrevne mekanismene.

Profil «Offentlig»

Profilen «Offentlig» dekker API-er som tilbyr data med behov for en viss kontroll over konsumenter. Den er også velegnet for API-er som eksponeres innenfor en kontrollert infrastruktur hvor behovet for sikkerhet dekkes av andre mekanismer.

Profilen kan brukes av «public clients», altså systemer som ikke er i stand til å beskytte hemmeligheter. Et eksempel på dette er nettleserapplikasjoner (Single Page Application, «SPA») uten en tilhørende serverbasert tjeneste. Et annet eksempel er «native apps», altså applikasjoner for mobiltelefon eller pc.

Utviklere av nettleserapplikasjoner bør følge anbefalingene i gjeldende versjonen av «OAuth 2.0 for Browser-Based Apps». For «native apps» bør anbefalingene i gjeldende versjon av «OAuth 2.0 for Native Apps» følges.

Profilen «Offentlig» gir API-et mulighet til å identifisere klienter og datakonsumenter, men med lavere sikkerhet for at identifikasjonen av klienten er korrekt enn for profilen «Normal».

Generelle krav

- Dersom biblioteker sertifisert for OIDC og OAuth2 finnes for teknologiplattformen man benytter, SKAL disse benyttes av klienter og API-er.
- All kommunikasjon SKAL skje over HTTPS, minimum TLS 1.2
- Access Token SKAL ha en så kort levetid som praktisk mulig.
- Et Access Token SKAL kunne utstedes som en JWT.

Krav til klienter

- Følgende protokollflyt SKAL brukes av klienter for å få utstedt Access Token:
  - For brukerpålogging: “authorization code”.
- Ved bruk av “authorization code”
  - SKAL PKCE benyttes.

    - SKAL klienten sjekke at parameteret «iss» returnert fra tokenutsteder i ID-token er korrekt.
- Ved bruk av OpenID Connect SKAL klienten validere ID Token som beskrevet her.

Krav til API-er

- Dersom et Access Token er en JWT, SKAL det valideres som beskrevet her.
- API-er SKAL IKKE benytte informasjon fra token som parametere, bare for tilgangskontroll. Det SKAL være mulig å erstatte tilgangskontrollmekanismen uten at det påvirker funksjonaliteten til API-et.

Profil «Normal»

Disse kravene bygger på kravene fra profil «Lav», og kommer i tillegg. Profilen kan bare benyttes av «confidential clients», altså systemer som er i stand til å sikre hemmeligheter.

«Normal» er meget lik OAuth 2.1, neste versjon av OAuth2. Denne spesifikasjonen er ikke endelig ennå, men når den foreligger kan man i stor grad peke direkte på den.

Generelle krav

- Et Access Token SKAL bare brukes mot et enkelt API (slik det er registrert i tjeneste for tilgangsstyring). Dersom en klient skal konsumere flere API-er, må den be om et token per API. Dette gjelder bare dersom token IKKE er kryptografisk bundet til klient.

Krav til klienter

- En av følgende protokollflyter SKAL brukes av klienter for å få utstedt Access Token:
    - For brukerpålogging: “authorization code”.
    - For system-til-system: “client credentials”.
- Ved bruk av “client credentials”
    - SKAL klienten sjekke at parameteret «iss» returnert fra tokenutsteder som separat parameter er korrekt.
- Alle klienter SKAL autentisere seg mot tokenutsteder med en hemmelighet med en av følgende typer:
    - client_secret
    - private_key_jwt
- Klienthemmeligheter SKAL ha en begrenset varighet, og må byttes med gitte intervaller.
- Access Token SKAL IKKE være tilgjengelig i nettleser. Se OAuth 2.0 for Browser-Based Apps.

Profil «Høy»

Profil “Høy” bygger videre på profil “Normal”. Den er tenkt brukt i tilfeller hvor lovkrav eller andre faktorer fører til ekstra høye behov for sikkerhet. Disse kravene er på linje med krav for deling av finansielle data og helseinformasjon.

Krav til klienter

- Klientautentisering SKAL bare skje ved bruk av privat nøkkel - private_key_jwt.
- Se neste avsnitt for fremtidige krav.

Om forvaltning av profilene

OAuth2 er under stadig utvikling, både funksjonelt og sikkerhetsmessig. Profilene må forvaltes kontinuerlig og oppdateres som hensiktsmessig. Det er viktig at det etableres en tydelig veiplan for profilene av forvalter. Datakonsumenter og eiere av datakilder må være i stand til å planlegge i god tid for endringer.

To standardiserte protokollmekanismer som er aktuelle for en fremtidig versjon av profil «Høy» er:

- Access Token SKAL være kryptografisk bundet til klient, for eksempel ved bruk av DPoP.
- Ved bruk av "authorization code" SKAL parametere overføres via bak-kanal – Pushed Authorization Requests.

Om klassifisering av data som skal deles

En behandlingsansvarlig som ønsker å dele data må selv klassifisere dataene den har ansvar for. Ut ifra dette kan det settes krav til hvilken profil en datakonsument må benytte. Det er altså behandlingsansvarlig som må vurdere hvilke krav som stilles til klienter som skal få tilgang til datakilden.

En datakonsument som konsumerer datakilder med forskjellige krav til hvilken profil som skal benyttes, må benytte profilen med høyest nivå. En klient som for eksempel konsumerer ett API som aksepterer "Offentlig" og et annet API som krever "Normal", må benytte profilen "Normal".

Tabellen under knytter sammen profilene og konfidensialitetsklassene i veilederen «Klassifisering av informasjon og informasjonssikkerhet». Beskrivelsene i tabellen er hentet fra veilederen.

Konfidensialitetsklasse	**Tilsvarende OAuth2 Profil - Minimumsnivå**	**Beskrivelse av konfidensialitetsklasse**
Åpen ("Grønn")	Offentlig	Informasjon kan være tilgjengelig for alle uten særskilte tilgangsrettigheter. Eksempler på slik informasjon er en web-side som presenterer en avdeling eller enhet som legges åpent ut på internett eller studiemateriell for et emne eller kurs som ligger åpent, men som er merket med en gitt lisens eller opphavsrett.
Intern ("Gul")	Offentlig	Informasjonen må ha en viss beskyttelse og kan være tilgjengelig for både eksterne og interne, med kontrollerte tilgangsrettigheter. Benyttes dersom det vil kunne forårsake en viss skade for institusjonen, eller samarbeidspartner hvis informasjonen blir kjent for uvedkommende. Eksempler på slik informasjon er enkelte arbeidsdokumenter,

		informasjon som er unntatt offentlighet, personopplysninger, karakterer, store studentarbeider, eksamensbesvarelser, forskningsdata og -arbeider.
Fortrolig ("Rød")	Høy	Benyttes hvis det vil forårsake skade for offentlige interesser, institusjonen, enkeltperson eller samarbeidspartner hvis informasjonen blir kjent for uvedkommende. Informasjonen skal ha strenge tilgangsrettigheter. Eksempler på slik informasjon er enkelte strategidokumenter, store mengder av sensitive personopplysninger, helseopplysninger, eksamensoppgaver før de er gitt, enkelte typer forskningsdata og -arbeider.
Strengt fortrolig ("Svart")	Høy	Hvis man har behov for et fjerde og høyere nivå for konfidensialitet kan man bruke klassen og gjøre en avgrensning mellom den og Fortrolig. Strengt fortrolig benyttes dersom det vil kunne forårsake betydelig skade for offentlige interesser, institusjonen, enkeltperson eller samarbeidspartner at informasjonen blir kjent for uvedkommende. Informasjonen skal ha de strengeste tilgangsrettigheter. Eksempler på slik informasjon er informasjon om personer som har adressesperre kode 7 eller som har behov for annen særlig beskyttelse og svært

		konfidensielle forskningsdata og -arbeider.

VEDLEGG – JSON WEB TOKEN SOM TOKEN-FORMAT

- Spesifikasjonen til OAuth2 sier ikke noe spesifikt om formatet til et Access Token.
- JSON Web Token (JWT) er ett av de standardiserte formatene.
- Andre alternativer er CBOR Web Token og «ugjennomsiktige» token som valideres med Token Introspection.
- De ulike formatene har forskjellige fordeler og ulemper, og hver sine anvendelser.

Valg av format for Access Token er et omfattende tema, og bør være gjenstand for en egen vurdering.

Det skal likevel sies at JWT er utbredt og velegnet for et heterogent økosystem som kunnskapssektoren. En felles tokenutsteder kan støtte flere formater på token, men det er sannsynlig at JWT vil være ett av de anbefalte.

Under følger derfor en kortfattet oppsummering over hva som taler for og mot bruk av JWT.

Fordeler med JWT som format

- Selvbeskrivende - kan inneholde all nødvendig informasjon for å ta en tilgangsbeslutning.
- Tilgjengelighet. Datakilder som mottar en JWT trenger ikke kontakt med tokenutsteder for å behandle det.
- Relativt kompakt.
- Innebygd støtte for kryptografiske funksjoner som signering og kryptering.
- Kan konsumeres av en API Gateway uten tilbakekall til tokenutstederen.
- Gir mange praktiske fordeler ved utvikling og overvåkning.
- Velegnet til logging og etterkontroll.
- Krever en omforent informasjonsmodell.

Ulemper med JWT som format

- Kan ikke revokeres - krever kompenserende tiltak.
- JWT-er inneholder informasjon og overføres mellom klient og API. Dette kan føre til informasjonslekkasjer dersom token-et inneholder sensitiv informasjon.
- Er selvstendige tilgangselementer, et token på avveie kan misbrukes.
- API-er som mottar en JWT har selv ansvar for å validere gyldigheten.
- Selv om formatet er kompakt, kan JWT-er bli flere kilobytes. Dette kan gi utfordringer i en heterogen nettverksinfrastruktur ettersom Access Tokens overføres i HTTP headere.

Om informasjonselementer tilknyttet Access Token

Uavhengig av formatet til et Access Token vil det gi tilgang til informasjon fra tokenutstederen. En del av disse informasjonselementene[11] er standardiserte. Annen informasjon er lokal for økosystemet og må standardiseres der – både syntaktisk og semantisk.

Eksempler på det siste er identifikatorer av forskjellig art samt annen informasjon nødvendig for tilgangskontroll og etterkontroll. Av spesiell interesse er identifikatorer som identifiserer personer og virksomheter.

Om identifisering av personer og virksomheter

Før et API leverer ut informasjon, skal det som oftest utføres tilgangskontroll. Ved bruk av OAuth2 vil denne tilgangskontrollen være basert på validering av Access Token-et som fulgte med i forespørselen. Utover å sjekke at et token er gyldig, er det ofte et krav å kontrollere hvilken virksomhet, person eller system som ønsker tilgang. Denne informasjonen må da være tilgjengelig direkte eller indirekte gjennom token-et.

For å indikere hvor trygg man er på at denne informasjonen er korrekt, brukes det ofte «sikkerhetsnivåer». De meste kjente eksemplene på slike sikkerhetsnivåer er eIDAS sin bruk av «Levels of Assurance» og ID-porten sin bruk av «acr» for personautentisering.

Et sikkerhetsnivå er en kombinasjon av prosessene brukt for knytte opp en identifikator til en fysisk eller juridisk person og hvilke identifikasjonsmidler som har vært i spill. Mer informasjon om dette kan man finne i «Veileder for identifikasjon og sporbarhet i elektronisk kommunikasjon med og i offentlig sektor» og «Veileder for virksomhetsautentisering».

Det anbefales at det jobbes videre med å bli enig om, og eventuelt definere, en standard i kunnskapssektoren for identifikatorer og sikkerhetsnivåer som dekker:

- Personer.
- Virksomheter.
- Personer og virksomheter uten norsk tilhørighet.
- Det bør vurderes om det er behov for å identifisere systemer som konsumerer data.
- Det bør vurderes om det er behov for å identifisere andre konstellasjoner som er spesifikke for kunnskapssektoren, for eksempel forskingsprosjekter.

Arbeidet bør ta hensyn til hva som allerede gjøres i norsk offentlig sektor og EU.

[11] «claims» eller «påstander»

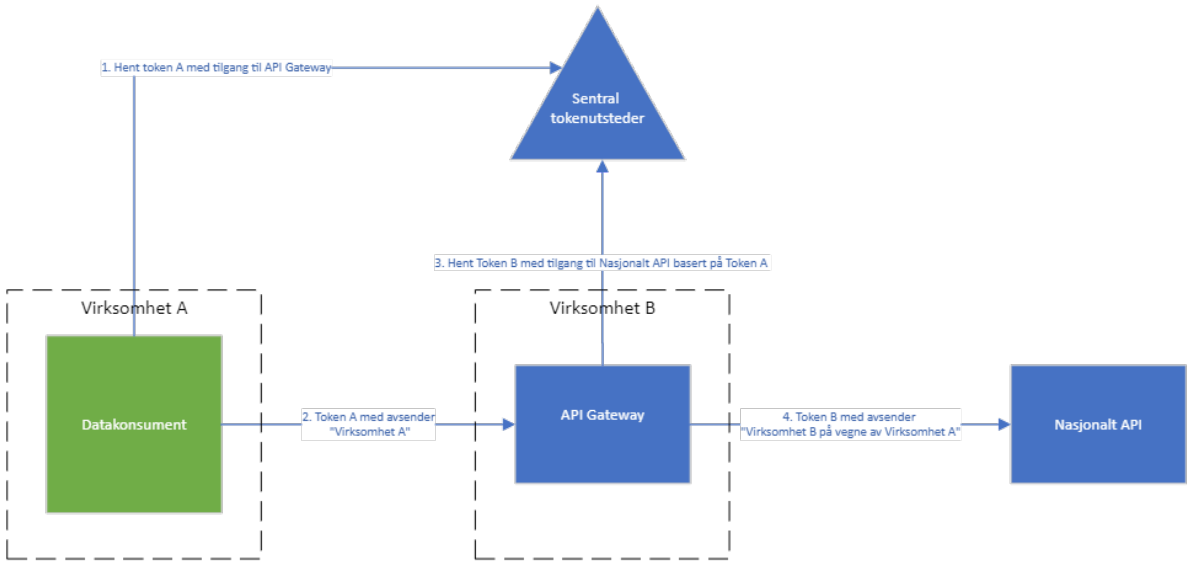
VEDLEGG – KJEDING AV API-KALL

Kunnskapssektoren er i bevegelse mot et økosystem som i større og større grad involverer HTTP-basert datadeling. Dette innebærer også en utvikling mot lengre verdikjeder hvor API-er vil kalle andre API-er på tvers av virksomheter.

En datatilbyder har ofte krav eller behov for å spore hvilke datakonsumenter, spesifikt virksomheter og/eller personer, man utleverer data til. En lengre kjede av API-kall kan føre til at slik sporing blir problematisk. Et økosystem med en sikkerhetsmodell som beskrevet i denne målarkitekturen muliggjør støtte for slik sporing.

Et eksempel vil være en nasjonal datakilde som inneholder data tilhørende flere virksomheter. En virksomhet kan ønske kontroll på hvordan data som tilhører virksomheten deles. Den nasjonale datakilden vil ha det samme behovet.

Virksomheten kan da sette opp et mellomliggende API i en API Gateway, og peke dette mot det nasjonale API-et. Uten tiltak vil da det nasjonale API-et ikke vite hvem det faktisk deler data med - bare virksomheten som eier API Gateway-en er identifisert.



Figur 3 - Et API kaller et annet API på vegne av en datakonsument

Et mulig alternativ er at sentral tokenutsteder støtter utveksling av Access Token mottatt av datakilde mot Access Token med nye tilganger. Ett nytt token kan da beholde informasjon om identiteten til første datakonsument i verdikjeden, i tillegg til identiteten til mellomledd.

Token Exchange er en standardisert måte å gjøre dette på, og er i dag i bruk i norsk offentlig sektor. Spesifikasjonen er relativt åpen, og må tilpasses behovene til kunnskapssektoren.

Det anbefales å vurdere støtte for en slik eller tilsvarende mekanisme i forbindelse med etablering av sentral tokenutsteder.

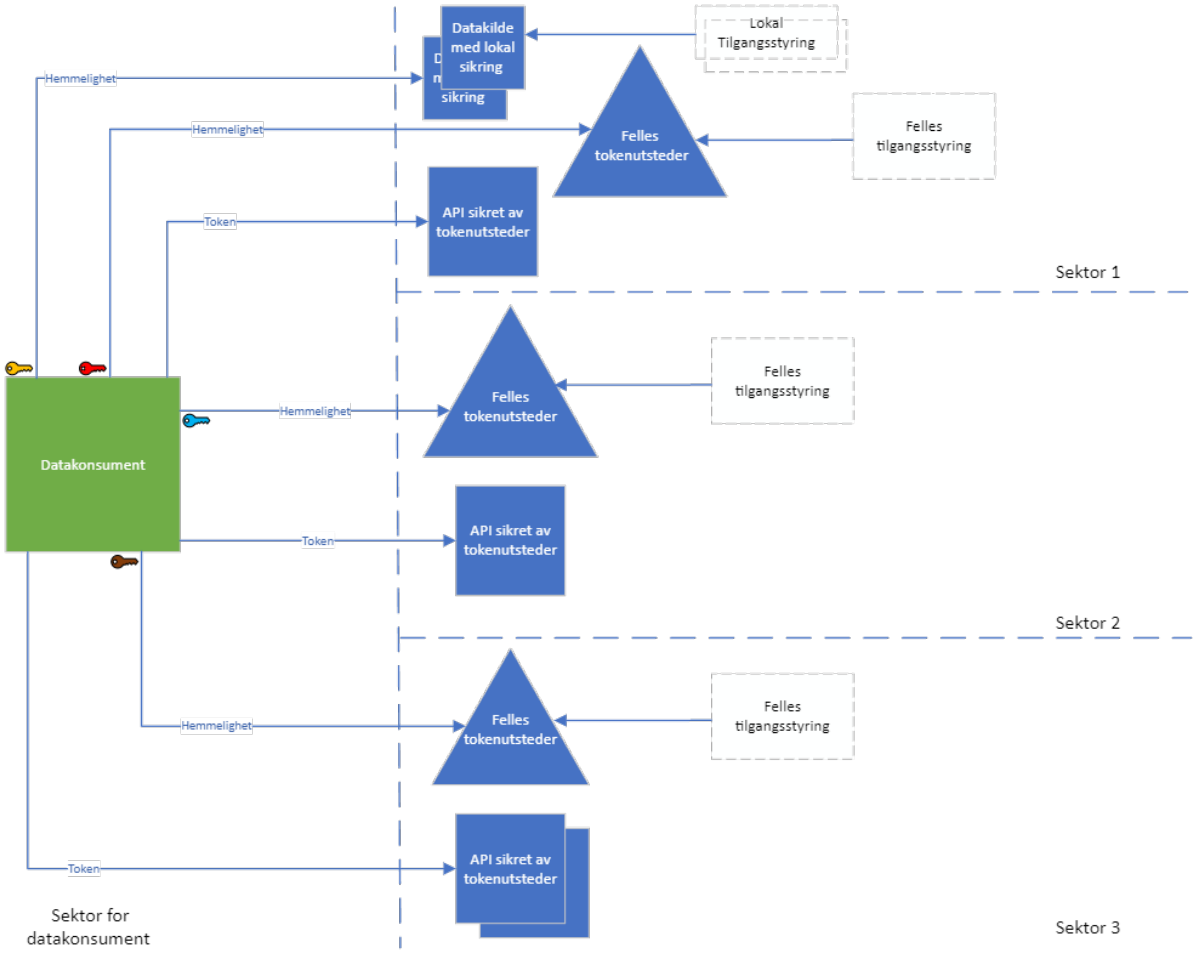
VEDLEGG – DATADELING PÅ TVERS AV OFFENTLIG SEKTOR

Dette vedlegget presenterer noen mønstre for å oppnå mer sømløs datadeling på tvers av forskjellige deler av offentlig sektor. Det er på ingen måte utfyllende, men er ment som et utgangspunkt for videre diskusjoner.

Alle mønstrene baserer seg på at det allerede eksisterer en sentral tokenutstedere innad i en sektor.

Alternativ 0 Lokale løsninger («as-is»)

I dag oppstår det gjerne lokale løsninger i scenarioer hvor en datakonsument eller en datakilde må forholde seg aktører fra forskjellige deler av offentlig sektor.



Figur 4 - Dagens situasjon

Et vanlig mønster er at datakonsumenter fra en sektor etablerer punkt-til-punkt integrasjoner når de har behov for å konsumere data fra en annen sektor. Dette betyr at de må forholde seg til flere regimer for tilgangsstyring, og forvalte integrasjonene separat med de utfordringer det medfører.

Et annet mønster innebærer at en datakilde etablerer multiple tillitsforhold, altså at de stoler på token levert av forskjellige tokenutstedere. På samme måte som for datakonsumenter, betyr det at

datakilden må forholde seg til flere regimer for tilgangsstyring. Datakilden må derfor forstå forskjellige tokenformat og informasjonsmodeller nødvendig for tilgangskontroll.

En tredje variant er å sette opp en mellomtjeneste, for eksempel med en API Gateway, mellom et API fra en annen sektor og mot egen virksomhet. Mellomtjenesten vil da måtte etablere eget regime for tilgangsstyring og tilgangskontroll.

Alternativ 1 Felles, nasjonal, tokenutsteder

En felles tokenutsteder for kunnskapssektoren er en sentral brikke i denne målarkitekturen. En *nasjonal* tokenutsteder for offentlig sektor vil kunne gi mange av de samme effektene, men på nasjonalt nivå.

En utfordring med et slikt mønster vil være å støtte pålogging med sektorspesfikke identiteter, for eksempel fra Feide. Dette innebærer at en nasjonal tokenutsteder må etablere integrasjoner mot sektorielle identitetstilbydere. Tjenesten må også være i stand til å videreføre domenespesifikk informasjon fra identitetstilbydere i token den utsteder.

En annen utfordring er å etablere en tilhørende tjeneste for tilgangsstyring som også støtter sektornære behov. En mulig løsning er å tilby API-er for automatisering til sektorspesifikke portaler.

Alternativ 2 Tette integrasjoner mellom tjenester for tilgangsstyring

De forskjellige tjenestene for tilgangsstyring i offentlig sektor har ikke noen form for integrasjon i dag. Det er mange årsaker til dette - både historiske og behovsdrevne.

Med en felles, overordnet arkitektur for datadeling over HTTP, kan det være mulig å integrere tilgangsstyringstjenestene. En datakonsument kan da bruke en sektorlokal tjeneste, og be om tilgang til datakilder beskyttet av tjenester i andre sektorer.

Datakonsumentene må fortsatt forholde seg til en tokenutsteder per sektor, men teknisk oppsett – inkludert klienthemmeligheter – vil kunne gjenbrukes.

Slike integrasjoner vil stille høye krav:

- Felles informasjonsmodell for tilgangsstyring på tvers av tjenestene.
- Høye krav til tillit og sikkerhet – kallende tjeneste vil kunne opptre på vegne av en hel sektor, og vil be om tilganger på vegne av aktører i denne sektoren.
- Omforente arbeidsflyter. Et eksempel er hvordan en eier av en datakilde skal godkjenne en tilgangsforespørsel fra en aktør i en annen sektor.
- Det må etableres et omforent regime for styring av tilgang til de forskjellige tjenestene. Man må for eksempel bli enig om hvordan en innlogget bruker i en tjeneste blir knyttet opp imot en virksomhet.

Det pågår arbeid for å standardisere slike mekanismer.

Eksempel - Alternativ 2

Under følger et konkret eksempel for å illustrere hva en slik integrasjon innebærer.

Maskinporten er en tokenutsteder for offentlig sektor som støtter OAuth2 for maskin-til-maskin API-er.

Maskinporten benytter **Altinn delegering** og sin egen tjeneste, **Samarbeidsportalen**, for tilgangsstyring.

HelseID er en tokenutsteder for helsesektoren som støtter OAuth2 for maskin-til-maskin API-er.

HelseID benytter **Altinn delegering** og sin egen tjeneste, **HelseID Selvbetjening**, for tilgangsstyring.

Tillitsnivåene for tilgangsstyring til Maskinporten og HelseID er relativt like, og man kan se for seg at det etableres avtaler og tekniske mekanismer som innebærer at Maskinporten kan stole på "attester" fra HelseID i form av signerte JWT-er. Datakonsumenter kan få utstedt en attest av HelseID, og sende den til Maskinporten i bytte mot et Access Token. Denne vil gi tilgang til API-er beskyttet av Maskinporten. Se her for en illustrasjon av denne tenkte tekniske flyten.

Konsekvensen blir da at datakonsumenter i helsesektoren kun trenger å forholde seg til en tjeneste for tilgangsstyring, HelseID Selvbetjening, for å kunne konsumere datakilder i annen offentlig sektor.

En felles, nasjonal, API-katalog som data.norge.no vil fortsatt være viktig for semantisk og syntaktisk interoperabilitet for dataene som deles av API-et.

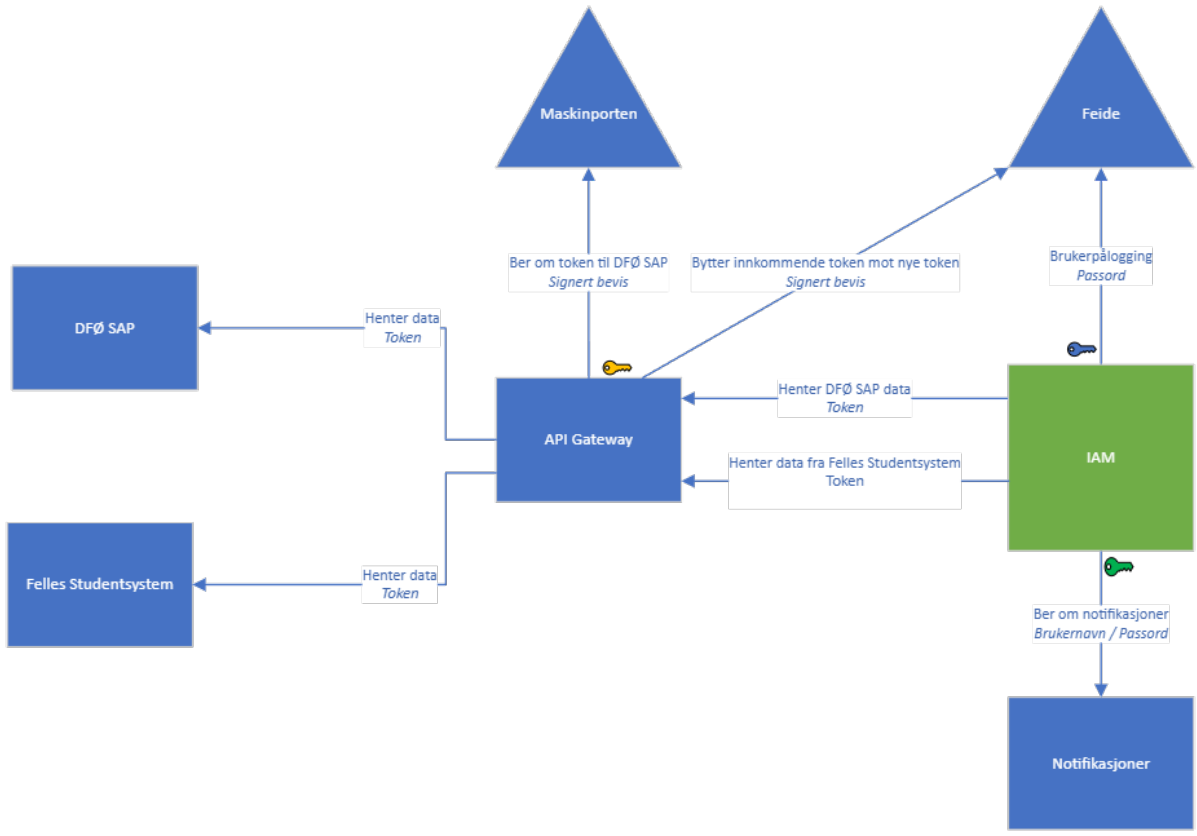
Eksemplet over krever teknisk og ikke-teknisk samspill mellom tokentjenestene, men kan forenkle prosessarbeid og tekniske implementasjon for datakonsumenter.

VEDLEGG – BRUK AV MÅLARKITEKTUR I FELLES IAM

For å illustrere effekten av målarkitekturen, beskriver dette vedlegget en versjon av Felles IAM hvor arkitekturen er fulgt. Dette er ett av flere mulige løsningsalternativer. Følgende antagelser er gjort i dette eksemplet:

- Sentral tokenutsteder er Feide
- Sentral tjeneste for tilgangsstyring er Feide Kundeportal
- Token Exchange støttes for utveksling av token, og benyttes av API Gateway
- Privat nøkkel som benyttes av API Gateway mot Maskinporten kan også benyttes mot Feide.
- Feide Kundeportal støtter IKKE tilgangsstyring for Notifikasjoner
- Notifikasjoner støtter IKKE token / OAuth2 som sikkerhetsmekanisme
- Det er ingen form for integrasjon mellom Maskinporten og Feide.

Figuren og tabellen under viser hvordan de forskjellige komponentene og tjenestene vil sikres med disse antagelsene



Figur 5 - Datadeling i Felles IAM med bruk av målarkitekur

	Autentiserings-mekanisme	**Sikkerhets-mekanisme**	**Tilgangsstyring**
Brukerpålogging	Passord	OAuth2	Feide Kundeportal

Felles Studentsystem API	Token	OAuth2	Feide Kundeportal
Maskinporten / DFØ SAP API	Privat nøkkel / Token	OAuth2	Samarbeidsportalen
API Gateway FS	Token	OAuth2	Feide Kundeportal
API Gateway DFØ SAP	Token	OAuth2	Feide Kundeportal
Notifikasjoner	Brukernavn og passord	AMQP / Basic Authentication	BROM Selvbetjening[12]

Tilgangsstyring av notifikasjoner støttes ikke av Feide Kundeportal i dette eksemplet, og vil fortsatt kreve separate hemmeligheter for klientautentisering.

Effekter for Felles IAM

Eksemplet over vil gi følgende effekter sammenlignet med dagens situasjon.

- All tilgangsstyring, med unntak av notifikasjoner og Maskinporten, skjer i Feide Kundeportal.
- OAuth2 benyttes for sikring av alle API-er.
- Hver klient (f.eks IAM) har bare en hemmelighet. Denne brukes mot Feide som utleverer token som gir tilgang til API-er.
- Identifisering av person og virksomhet håndteres av Feide og Feide Kundeportal.
- Sporbarhet sikres gjennom bruk av Token Exchange.

[12] Feide Kundeportal kan i teorien støtte tilgangsstyring for Notifikasjoner, men det ansees som utenfor skop for dette dokumentet.

VEDLEGG – DATADELING MED HTTP

Hva kjennetegner tjenester som bruker HTTP?

HTTP er en tilstandsløs internettprotokoll på applikasjonslaget som definerer hvordan selvbeskrivende meldinger sendes mellom en klient og en server. En melding er enten en forespørsel eller et svar (request og response).

Tjenester som sender meldinger via HTTP kjennetegnes ved at de har "HTTP" eller "HTTPs" i URI, benytter TCP-port 80 eller 443, og at de identifiserer HTTP-versjon i meldingene (for eksempel HTTP/1.1, h2, h3).

HTTP definerer et sett av metoder som brukes når man sender meldinger fra klient til server, hvor GET, POST, PUT og DELETE er mest kjent. Det er bare metodene GET og HEAD som må støttes i HTTP servere i henhold til spesifikasjon.

Den vanligste bruken av HTTP er ved visning av nettsider, hvor innholdet i svaret fra HTTP-serveren er HTML, CSS og Javascript. Men, det er også vanlig å benytte HTTP som meldingsprotokoll for API.

Design av HTTP-baserte API

Et HTTP-basert API bør fungere uavhengig av mellomtjenere og sikkerhetsmekanismer, og skal være i stand til å behandle forespørsler basert på innholdet i HTTP-meldingen det mottar. En HTTP-melding skal være selvbeskrivende, og data i meldingen skal være tilstrekkelig for punkt-til-punkt kommunikasjon.

Et HTTP-basert API bør f.eks. kunne bytte ut eventuelle sikringsmekanismer og mellomtjenere uten at det påvirker hvordan det brukes av sine klienter. Med andre ord: informasjon i sertifikater og/eller token (f.eks. Access Token) skal ikke brukes til å endre resultatet i svaret fra API-et, men de vil kunne benyttes for å avgjøre hvorvidt klienten får tilgang til den forespurte ressursen på serveren.

Forskjellige designstiler

Det er mange tilnærminger til design av HTTP-baserte API. Noen vanlige stiler er REST, SOAP og GraphQL. Hvilken designstil man velger avhenger i stor grad av smak og behag, samtidig kan en gitt designstil ha praktiske fordeler sammenlignet med andre. Stilen som velges gjenspeiler i mange tilfeller den grunnleggende informasjonsmodellen for dataene som skal deles.

Serialisering av innhold i HTTP-meldinger

HTTP stiller ingen krav til hvordan data i HTTP-meldinger skal serialiseres/formatteres. For API er det er vanlig å benytte serialiseringsformatene JSON og XML.

Hvordan overføre parametere til HTTP baserte API?

HTTP tillater overføring av parametere i forskjellige deler av en melding: ved bruk av query-parametere, i egendefinerte headere, som en del av stien eller i meldingskroppen.

Vurder nøye bruk av query-parametere i URL i startlinjen («Request Line»)

Det kan være gode grunner til å unngå bruk av query-parametere i URL i startlinja, dette er beskrevet i større detalj i spesifikasjonen.

Unngå bruk av meldingskropp i GET, HEAD eller DELETE metoder

Selv om HTTP ikke forbyr bruk av meldingskropp i en GET, HEAD eller DELETE melding, bør bruk av meldingskropp i GET, HEAD eller DELETE vurderes nøye av flere årsaker. HTTP-spesifikasjonen definerer at innholdet i meldingskroppen ved en GET-forespørsel ikke skal påvirke svaret. I tillegg er det en risiko for at meldingskroppen fjernes av mellomliggende tjenere (proxy-er).

Spesifikasjonen anbefaler bruk av POST metoden i stedet for GET når det overføres sensitiv informasjon i parametere i forespørsler:

> *"When information retrieval is performed with a mechanism that constructs a target URI from user-provided information, such as the query fields of a form using GET, potentially sensitive data might be provided that would not be appropriate for disclosure within a URI (see Section 17.9). In some cases, the data can be filtered or transformed such that it would not reveal such information. In others, particularly when there is no benefit from caching a response,* ***using the POST method (Section 9.3.3) instead of GET can transmit such information in the request content rather than within the target URI.***"

Begrens bruk av proprietære HTTP-headere

Spesifikasjonen anbefaler å begrense bruken av proprietære headere for HTTP-meldinger. Det er en risiko for at mellomliggende programvare håndterer ukjente HTTP headere på ulik måte ved at de for eksempel fjerner eller logger informasjonen.

Spesifikasjonen anbefaler derimot å definere proprietære HTTP-headere når:

- En proprietær header er nyttig for mellomliggende programvare (f.eks. API Gateway-er) som ikke skal lese innholdet i meldingskroppen til HTTP-meldingen.
- Når innholdet ikke kan inkluderes i meldingskroppen fordi formatet ikke tillater det. Kjente format for HTTP-content er registrert i IANA. Formatet på innholdet (verdien) i HTTP-headere er udefinert i spesifikasjonen.
- Når feltet er nyttig for generiske HTTP-programvarekomponenter

Sikkerhetsmodellen for HTTP

HTTP-baserte API er avhengig av sikkerhetsegenskapene til den underliggende transporten for å sørge for at konfidensialitet og integritet blir ivaretatt. Derfor skal et HTTP-basert API som eksponerer informasjon alltid bruke TLS og ha "HTTPs" i sin URL.

I tillegg skal alle som utvikler API som eksponerer taushetsbelagt informasjon gjennomføre tiltak mot kjente sikkerhetsrisikoer. En god kilde til de mest aktuelle sikkerhetsrisikoene er OWASP Top 10 API Security Risks.

I OWASP sin topp 10 liste for 2023 er 4 av 10 risikoer knyttet til autentisering og autorisasjon, og punkt 2 og 3 i lista er direkte knyttet til tilgangsstyringsmekanismene OAuth 2.0 og OIDC. Det følgende tar for seg autentisering og autorisasjon i HTTP.

Autentisering i HTTP

HTTP tilbyr et generelt rammeverk for autentisering, som lar en HTTP-server velge hvilken autentiseringsmekanisme den ønsker å bruke. For HTTP baserte API er *Bearer* token ved bruk av OAuth 2.0 vanlig, og har flere fordeler sammenlignet med andre metoder. IETF publiserte nylig standarden OAuth DPoP, som er en måte å kryptografisk binde Access Token til HTTP-klienten. DPoP er en videreutvikling av Bearer token som reduserer sannsynligheten for tyveri og misbruk av Access Token.

Tilgangsstyring ved bruk av OAuth 2.0

OAuth 2.0 er et protokollrammeverk som beskriver hvordan en tilgang til et API kan delegeres fra sluttbruker til programvare, slik at programvare kan opptre på vegne av sluttbrukeren. I sin enkleste form beskriver protokollen teknisk meldingsflyt mellom en HTTP-klient, en autorisasjonsserver og en ressursserver (API). Protokollen tilbyr et grunnleggende sett av tilgangsstyringsmekanismer, men er laget slik at den kan utvides og tilpasses forskjellige forretningskrav og funksjonelle behov.

Tilganger i autorisasjonsserveren: grant og scope

Når en HTTP-klient sender en vellykket tilgangsforespørsel til en OAuth 2.0 autorisasjonsserver vil autorisasjonsserveren opprette og lagre en representasjon av tilgangen ("grant") som ble gitt. En tilgang er knyttet til en spesifikk HTTP ressurs ("scope") og/eller en spesifikk ressursserver ("API").

HTTP-klienten kan deretter be om å få utstedt et Access Token som representerer denne tilgangen. Når HTTP-klienten skal sende en HTTP-melding til et API-et som forventer et Access Token må den legge ved Access Token-et i headeren «Authorization». Denne headeren er definert i det generiske rammeverket for autentisering i HTTP.

Detaljert autorisasjonsinformasjon

Et scope i OAuth 2.0 er en enkel tilgangsstyringsmekanisme, og er i mange sammenhenger ikke tilstrekkelig for å styre tilganger i dagens HTTP-servere. IETF har derfor utvidet spesifikasjonen av OAuth 2.0 med «Rich Authorization Requests» som lar HTTP-klienten berike tilgangsforespørselen med mer detaljert autorisasjonsinformasjon om sluttbrukeren.

Access Token

Et Access Token er en representasjon av en tilgang som er gitt i en autorisasjonsserver.

Et Access Token kan enten være en referanse til en tilgang gitt i en autorisasjonsserver, eller selvbeskrivende («self contained») ved at det inneholder informasjon om tilgangen. Det mest vanlige formatet for selvbeskrivende Access Token er JWT (JSON Web Token). JWT formatet ble utviklet spesielt for OAuth 2.0, men er designet til å kunne brukes uavhengig av protokollrammeverket.

Bruk av Access Token i et HTTP-basert API

Internettprotokollen OAuth 2.0 definerer ikke hvilket format et Access Token skal ha, men det finnes flere standardiserte formater som blir brukt i dag, som f.eks. JWT og CWT (CBOR Web Token).

Et Access Token med eventuelle påstander skal brukes til å validere hvorvidt avsender skal få tilgang til etterspurt ressurs. På grunn av at JWT og CWT er kryptografisk signert, kan det også være velegnet til sporbarhetsformål og kan benyttes til revisjonslogging dersom den kryptografiske nøkkelen er sterkt knyttet til en juridisk eller fysisk person.

Når en HTTP-server (API) mottar en melding fra en klient vil Access Token overføres som en verdi i HTTP-header feltet "Authorization".

Dersom et token er selvbeskrivende må HTTP-serveren sørge for

- validering
    - kontrollere at utstederen av token-et er kjent
    - validere signaturen
    - kontrollere at HTTP-serveren er riktig mottaker
    - kontrollere at token-et ikke er utløpt
- å kontrollere at riktige tilganger er gitt
- å kontrollere at eventuelle parametere i stemmer overens med påstander i token-et

Oppsummering

- BRUK TLS og HTTPS i URI
- BRUK «OWASP Top 10» for å sette søkelys på sikkerhetstiltak mot kjente sikkerhetsrisiko
- Bruk POST og HTTP-content (også kjent som *body*) for overføring av parametere fra klient til HTTP-basert API.
- SØRG FOR at det er mulig å bytte ut eller endre sikkerhetsmekanismer uten at det bryter integrasjonen mellom klient og API
- BRUK OAuth2 i tråd med gjeldende retningslinjer for tilgangsstyring og autentisering
- BRUK OAuth2 i henhold til beste praksis når informasjonen som utveksles er sensitiv eller taushetsbelagt
- BRUK informasjon om klienten og sluttbrukerens tilgang som ligger i Access Token for validering av parameter i HTTP-API
- IKKE BRUK Access Token til overføring av parameter som er nødvendige for at API-et skal fungere
- BRUK Access Token til revisjonslogging

VEDLEGG – ORDLISTE

Begrep	Overlappende, ikke nødvendigvis synonyme, begrep	Beskrivelse
Tilgangsstyring	Access Management	Tilgangsstyring handler om å administrere og organisere hvordan ulike virksomheter, brukere eller systemer får tillatelse til å få tilgang til data, tjenester og andre ressurser. Dette innebærer ofte å tildele rettigheter og privilegier til juridiske eller fysiske personer, og å sikre at tilgangen er i samsvar med gjeldende lover, retningslinjer og sikkerhetskrav.
Tilgangskontroll	Autorisasjon, Access Control	Tilgangskontroll refererer til den tekniske utøvelsen av tilgangsstyring. Dette innebærer bruk av teknologier og mekanismer som begrenser eller tillater forespurt tilgang til spesifikke ressurser eller data, basert på rettigheter gitt gjennom tilgangsstyring.
Klient	OAuth klient, HTTP klient, Client, Relying party, Applikasjon, Datakonsument, Tjeneste	
API	HTTP API, HTTP server, HTTP ressurs, Datagrensesnitt, REST API, Tjener	Et teknisk grensenitt som kalles maskinelt. I dette dokumentet brukes begrepet i kontekst av protokollen HTTP, og kan derfor konsumeres over en internett-basert infrastruktur.
API Gateway, Mellomtjener	HTTP mellomtjener, API Proxy	En mellomtjener som står mellom en datakonsument og en datakilde. Tilbyr berikende tjenester som tilgangskontroll, monitorering, trafikkontroll og mer.
Token	JWT, Sikkerhetsbillett	Et stykke data som gir en potensielt tidsbegrenset tilgang til en eller flere datakilder.
JSON Web Token	JWT, Token	Et JSON Web Token er en sikker og kompakt måte å representere påstander på mellom to parter.
Datakonsument	Klient	Rollen til en «aktør» som leser eller skriver data til en datatilbyder / datakilde. Mindre teknisk rettet enn «klient», samsvarer ofte med en juridisk eller fysisk person.
Datatilbyder	Behandlingsansvarlig	Den juridisk ansvarlige aktøren for en datakilde.
Datakilde	API, HTTP Server, Server	Rollen til en «aktør» som tilbyr en datakonsument mulighet til å lese eller

		skrive ett sett med data som aktøren har tilgang til.
OAuth2		En sikker, token-basert internettprotokoll som gir en applikasjon tilgang til eksterne datakilder uten å benytte passord.
OpenID Connect	OIDC	En utvidelse av OAuth2 som standardiserer autentisering av brukere.
Tokenutsteder	Authorization Server, Token Service, OpenID Connect Provider,	En tjeneste som er ansvarlig for å utstede token som gir tilgang til datakilder.
Klienthemmelighet	Hemmelighet, Secret, Passord, Privat nøkkel, API nøkkel.	En hemmelighet som benyttes av en klient for å bevise hvem den er, for eksempel opp imot en tokenutsteder.
IDP	Identitetstilbyder, Identity Provider	En tjeneste som tilbyr autentisering av personer eller andre entiteter.
Føderert autentisering	IDP, tiltrodd tredjepart, OpenID Connect, SAML	En autentiseringsprosess der en bruker kan få tilgang til flere uavhengige systemer eller tjenester ved å autentisere seg én gang.
Claim	Påstand	Et claim er et attributt i innholdsdelen av en JWT. I kontekst av OAuth2 er dette informasjon som tokenutsteder utleverer i forbindelse med at det er gitt en tilgang i form av et token. Det kan være informasjon om en sluttbruker, en virksomhet eller mer teknisk informasjon om tilgangen.
Scope	Tilgang	Et scope i kontekst av OAuth beskriver en tilgang som har blitt gitt til en klient. I en JWT uttrykkes scopes som ett claim.
Tjeneste	Klient, Datakonsument	I kontekst av Feide er en tjeneste en applikasjon som tilbyr brukerpålogging med Feide.
Tjenesteleverandør	Databehandler	En leverandør som tilbyr utvikling, forvaltning og/eller drift av tjenester.
Vertsorganisasjon		En vertsorganisasjon er en utdanningsinstitusjon som bruker Feide som innloggingsløsning til systemene og tjenestene de benytter

VEDLEGG - REFERANSELISTE

Rammeverk, profiler og arkitektur

Referansearkitektur for deling av data i høyere utdanning og forsking	https://unit-norge.github.io/unit-ra/main/index.html
Profil for FAPI 2	https://openid.bitbucket.io/fapi/fapi-2_0-security-profile.html
Profil for bruk av HelseID	https://helseid.atlassian.net/wiki/spaces/HELSEID/pages/128352260
Referansearkitektur for KUDAF	https://kunnskapsdata.no/aktuelt/referansearkitektur-for-kudaf
Profil for HelseID	https://helseid.atlassian.net/wiki/spaces/HELSEID/pages/128352260/Se clients+using+HelseID
Sertifiserte OpenID Connect implementasjoner. Dekker implisitt også OAuth 2.0	https://openid.net/developers/certified-openid-connect-implementatio

Spesifikasjoner

OAuth 2.0	https://datatracker.ietf.org/doc/html/rfc6749
OAuth 2.1	https://datatracker.ietf.org/doc/html/draft-ietf-oauth-v2-1-09
OAuth 2.0 Security Best Current Practices	https://datatracker.ietf.org/doc/html/draft-ietf-oauth-security-topics-24
OAuth 2.0 for Browser-Based Apps	https://datatracker.ietf.org/doc/html/draft-ietf-oauth-browser-based-apps
OpenID Connect	https://openid.net/specs/openid-connect-core-1_0.html
Json Web Tokens	https://datatracker.ietf.org/doc/html/rfc9068
Demonstrating Proof of Possession (DPoP)	https://datatracker.ietf.org/doc/html/rfc9449
Pushed Authorization Requests (PAR)	https://datatracker.ietf.org/doc/html/rfc9126
Resource Indicators	https://datatracker.ietf.org/doc/html/rfc8707

Viktige Tjenester

Feide Kundeportal	https://www.feide.no/kundeportalen
IntArk	https://sikt.no/tjenester/intark
Felles IAM	https://sikt.no/tjenester/felles-iam
Kudaf	https://hkdir.no/hoyere-utdanning-og-forskning/digital-omstilling/kudaf
Maskinporten	https://samarbeid.digdir.no/maskinporten/maskinporten/25
ID-porten	https://samarbeid.digdir.no/id-porten/id-porten/18
Samarbeidsportalen	https://samarbeid.digdir.no/eformidling/om-samarbeidsportalen/1023
HelseID	https://www.nhn.no/tjenester/helseid/hva-er-helseid

Veiledere fra Digdir

Nasjonal verktøykasse for deling av data	https://www.digdir.no/datadeling/deling-av-data/2243
Veileder for orden i eget hus	https://www.digdir.no/informasjonsforvaltning/veileder-orden-i-eget-hus/2716

Veileder for identifikasjon og sporbarhet i elektronisk kommunikasjon med og i offentlig sektor	https://www.digdir.no/digital-samhandling/veileder-identifikasjon-og-sporbarhet-i-elektronisk-kommunikasjon-med-og-i-offentlig-sektor/2992
Veileder for virksomhetsautentisering	https://www.digdir.no/datadeling/veileder-virksomhetsautentisering/2435