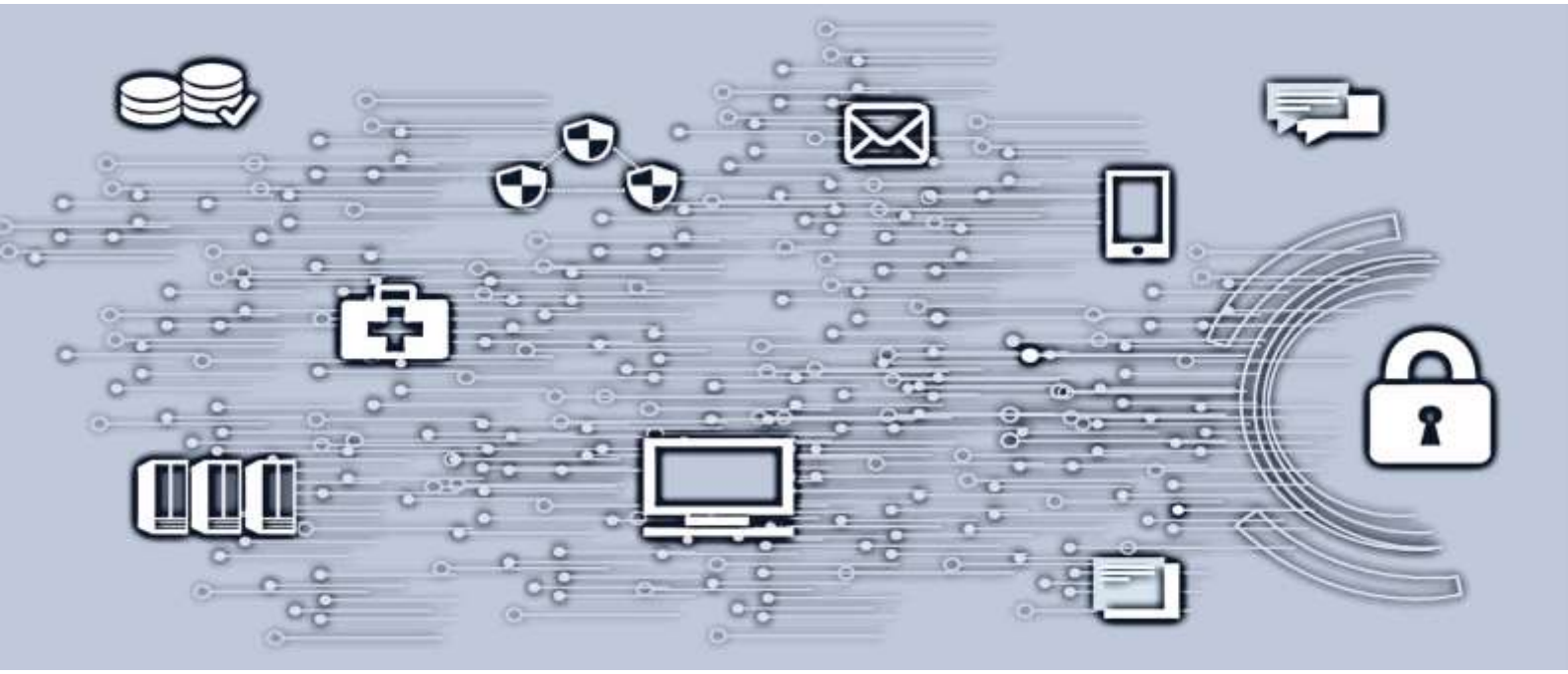
SYKEHUSPARTNER

NO-19 – Vedlegg Sikkerhetsprinsipper- og krav for IAM



Innhold

Versjon	Dato	Godkjent av
1.1	2019-09-04	RSV
2.0	2020-08-01	Christian Corneliussen
2.1	2021-08-12	Christian Corneliussen
2.2	2022-07-03	Christian Corneliussen

1. Innledning

1.1 Hvordan bruke dokumentet

Sikkerhetsprinsippene for IAM kan brukes som sjekkliste sammen med et løsningsdesign i forbindelse med etablering av tjeneste, endring av tjenester, eller i forbindelse med revisjoner og internkontroll av tjenester som skal benytte IAM. Dokumentet brukes også som grunnlag ved anskaffelser av tjenester som skal benytte IAM. Dette dokumentet er støttedokument til NO-19 - Sikkerhetsprinsipper og -krav for IKT-infrastruktur og applikasjoner.

1.2 Begreper

Innledende forklaringer av noen av begrepene brukt i kravene:

- Med «Application» menes et IT-system som må ha kjennskap til brukere (kan være en integrasjonsbruker eller en person) som benytter applikasjon av hensyn til å:
- Differensiere tilganger til funksjoner og data på en sikker måte, med andre ord må applikasjonen håndheve tilgangskontroll
- Spore tilganger via auditlogg. Hvis behandlingsrettet helseregister, innsynslogg.
- «SAML token» er brukt som synonym for «SAML assertion».
- «Application role» er brukt som synonym for «System role»: definerer en rolle som kan tilegnes en bruker inne i selve applikasjonen

2. IAM-krav

Nr	Requirement	Rational / Remarks
	Authentication	
1	The application must act as service provider integrated with Helse Sør-Øst's authentication service.	Synonyms for service provider are • relying party (RP) • "claims aware application" • "claims-based application" The requirement implicitly means that authentication to the application must NOT be based on local user database, Kerberos or LDAP. Credential management and authentication is totally externalized. Passwords or other end user's credentials are never processed by the application. This requirement enables, but is not limited to, central management of authentication policy, adaptive and risk-based authentication.
2	Integration with Helse Sør-Øst's authentication service must be based on standard identity federation protocols.	Identity federation protocols currently supported by HELSE SØR-ØST are OpenID Connect, SAML, OpenToken and WS-FED. Integration might be either based on built-in functionality ("native" support of identity federation) or through a third party access management tool. If built-in, describe whether and which third party security libraries are used or not. In case of support via third party access management tool describe which mechanism and third party services/products are currently supported. HELSE SØR-ØST is currently using Ping Access and Ping Federate for Access Management.
3	If SAML is used as authentication mechanism, the application should support "Service Provider Initiated" federation process.	This is to ensure that the application will support redirect functionality to the IdP in case no SAML assertion is available, as "Identity Provider Initiated" as alternative is difficult to implement in a regional/many AD context.
4	Security tokens must be validated before using claims.	Validation of the security token must be done according to best practices and recommendations for the used standard (e.g. JWT, SAML), preferably using an Helse Sør-Øst's security token service or locally if external validation is not appropriate. A justification of selected validation method needs to be supplied.

5	APIs provided by the application should be stateless and use claims from the security token for identification and audit trail.	The requirement assumes that the application is acting as an API provider, e.g. FIHR. Identification of both the end user and the organization the end user is acting on behalf of must be logged in the audit trail. In addition to identification, the claims can also be used for authorization. Describe whether the application can leverage claims in the security token to perform other controls than identification and audit trail. In case the application offer many type of API's (like operational integration, deployment / administration) describes which API fulfill the requirement.
6	Call from the application to external APIs should include a security token with relevant claims that identify at a minimum the end user and the organization the user is acting on behalf of.	The requirement assume that the application is acting as a service consumer of APIs. The security token must be derived from the current end user security context to include information like organization, role. If it is a patient related call, the security token should also include the patient id. Interaction with an external security token service might therefore be necessary to fulfil this requirement.
7	How security related to "security tokens" is implemented, and how to install and configure the application to follow good security practices must be documented in a security guide delivered with the application.	Examples of relevant information are • which security libraries are used for validation of certificate and the security token, • which security control are performed • how keys and certificates are stored, used and managed by the application (e.g. pull of keys from JWKS end point, caching, CRL check).
8	When using SOAP protocol, SAML token must be used according to WS-standards. When using REST protocol, JWT token must be used.	
9	At user's login time, the application must authorize access, establish a security session and log user activities based at a minimum on the user id, organization unit and role.	Due to the "multi tenancy" operation model, this information triplet must always be used for both access control decision and audit log.

10	The application must create, enforce and maintain a security session for the end user.	Security session is typically created at login time. Enforcement includes user inactivity, start and end time of the access token if used. Maintenance is e.g. refreshing the access token, enforcing re-authentication.
11	Helse Sør-Øst's authentication service based on standard identity federation protocols should be used if the application needs to re-authenticate the user, or for authentication of a second user within the same security session.	Some common use cases are re-authentication due to inactivity or additional control when doing a specific operation in the application. Additional control could require a step-up authentication (stronger authentication than the one used at login time). Another use case could be the need for "two-man" rule control.
12	Upon unsuccessful authentication of a user, user must be denied access and informed accordingly if he/she is not authorized to use the application.	The application must not fall back to an alternative authentication mechanism.
	Authorization	
13	The application must be able to use the attributes based access control (ABAC) and Policy Based Access Control (PBAC) paradigm.	
14	Access to functionality and patient data in the application must be based on a combination of user's role and organizational affiliation.	Essentially, the role gives access to functions while the organization controls access to patient data. This must be supported by the use of either internal access control mechanisms or/and external authorization services.
15	The application should ask Helse Sør-Øst's authorization service, an external decision point, if internal access control mechanism cannot determine access.	In other words, the application should only enforce access control that is to implement a Policy Enforcement Point (PEP) according to XACML reference architecture, the decision being done by the external Policy Decision Point (PDP) implemented in Helse Sør-Øst's authorization service. One example of external decision which is already in use is whether the practitioner, like a physician or nurse, has an active treatment relationship with the patient.
16	If the application supports external authorization, a standardized request / response interface should be used between the application enforcement point and the external authorization server for policy decision.	This requirement is relevant if the application support external PDP. Such a standard is XACML, e.g. Request / Response Interface based on JSON and HTTP for XACML 3.0.

17	If the application does not support external authorization, it should use an internal policy decision point (PDP) where access control rules are defined in a structured declarative policy language. The more "human readable" the language is, the better.	Implicitly it is a requirement that the policy can be read from, or pushed by an external source, so that the management of the access control policy is externalized (i.e. "Policy Administration Point" (PAP) can be externalized from the application). Examples of policy languages are XACML or ALFA. A natural but structured language format fulfils as well the requirement.
18	Application's APIs acting as a "resource server" (according to OAUTH2 standard) must enforce a first level of authorization at access time and is subject to the same access control rules (as for interactive usage of the application) at processing time.	Access time" is the first line of defense based on international standard like OAUTH2 (Access Token, scopes, claims). "Processing time" is when the API internally process the call to build the response.
19	If access is denied, the application should return a friendly error message, machine readable JSON/XML when acting as a "resource server", on which the cause of denied access is stated using a language understandable by the largest possible audience.	The communication between PEP and PDP is not end user communication, so it should be machine readable, while the response from Client to end user should be a user friendly response.
	Identity Management	
20	Persons has to be identified by one single and unique user identifier, even in a multitenancy scenario (e.g. regionally consolidated application serving different health trusts).	
21	Application's rights and permissions should be defined as roles. It should be possible, within the provisioning process, to associate a user with an application role to allow a proper usage of the application.	
22	The application must support that there may be multiple roles and / or organizations associated with the same unique user ID (multitenancy).	
23	The application must support a unique identifier for unambiguous identification of organizational units.	
24	Access can be differentiated depending on where the employee works at all times.	

25	If the application uses its own user store, the application must expose Identity Management (IdM) APIs through which it is possible to perform CRUD operations (Create, Read, Update, Delete, List) so that user management, granting and removing of role based access can be automated. The API must: - Be well documented, so that there is no need for additional training and/or particular effort for the development team to integrate - Have standardized authentication mechanism - Communicate using HTTP, LDAP or SQL over an encrypted channel Ideally APIs adhere to REST architectural constraints and can be consumed by an IDM connector based on the SCIM specification and that works over HTTPS.	
26	A user provisioned via IdM APIs provided by the application must be fully enabled with the permissions and grants related to the assigned role. Additional manual operation must not be required in order to enable the user to use the application.	
27	The API should allow the allocation of individual rights in addition to standard access based on role.	
28	If the application uses its own user store, it should offer a graphical interface to manage users locally.	
29	The API must enable the retrieval of existing data related to the users, roles, organizations and access from the application.	
	PAM	

30	The application must offer technical means to integrate with a Privileged Access Management (PAM) solution, so secrets and objects that give high privileged access can be under the control and protection of Helse Sør-Øst's PAM solution.	An example of "control and protection", when using shared admin account/password is the ability to automate password rotation, scheduled based or after each individual usage, as well as password verification to monitor that the password is not being changed outside the PAM solution. In term of technical requirement the application must then provide secured APIs or CLI (command line) to allow such integration with the PAM solution. Another feature could be the ability for the application to read password or other secrets from the password safe of the PAM solution. In that case the application must leverage API's provided by the PAM solution. Currently, HSØ is using Cyberark as a PAM solution.

3. Definisjoner

Se eget dokument: SP-S-ISMS-02 – Kilder og definisjoner

4. Avvik eller dissens

Avvik på denne instruks meldes i virksomhetens avvikssystem. Informasjonssikkerhetsleder og/eller personvernombud skal varsles

5. Referanser

https://services.fisp.no/sites/kvalitetsportalen/kvalitet/Sider/informasjonssikkerhet.aspx