Health Registries for Research
Norway

Report 3/2017

Logging and tracing in health registries

Work package 3: Data security implementation and support service

1 EXECUTIVE SUMMARY

This report describes the solutions for logging and tracing of data access implemented in health registers at FHI to fulfil governmental regulatory requirements. The logging solutions record when and who accessed personally identifiable data of sensitive nature, which data was accessed, and for which reason. The report also discusses prospective solutions to be implemented in the modernization projects for the health registers.

The report covers the following registers with sensitive personal data:

a) Medisinsk fødselsregister (MFR)
b) Modernisert Medisinsk fødselsregister (mMFR)
c) Dødsårsaksregisteret utleveringsdatabase (DÅR)
d) Hjerte- og karregisteret (HKR)

Table of Content

Document information:
Date published: 22. February 2017
Written by: Alexander Kholosha and edited by Kjell Jørgen Hole
Doc-ID: Report - Logging and tracing in health registers ver.1
Classification: Enterprise confidential

2 INTRODUCTION

2.1 Objective

The objective of this report is to describe the logging solutions implemented on FHI's Oracle platform to fulfil regulatory requirements. The logging solutions record who accessed personally identifiable sensitive data, which data was accessed, when the data was accessed, and for which reason. The report also suggests improvements to the described logging system.

2.2 Scope

The report covers the following registers with sensitive personal data:

a) Medisinsk fødselsregister (http://www.fhi.no/helseregistre/medisinsk-fodselsregister)
b) Modernisert Medisinsk fødselsregister
c) Dødsårsaksregisteret (utleveringsdatabase) (http://www.fhi.no/helseregistre/dodsaarsaksregisteret)
d) Hjerte- og karregistert (basisdelen) (http://www.fhi.no/helseregistre/hjerte-og-karregisteret)

2.3 Regulatory documents

The term "regulatory documents" denotes laws, regulations, standards, supervisory documents from FHI that either contain requirements for information security or outline satisfactory solutions that meet the requirements of laws and regulations.

The requirements for logging and tracing follow the Health Register Law enforced on 1.1.2015. The law's § 24 states:

"*Rett til informasjon og innsyn følger av personopplysningsloven §§ 18 flg. Innsynsretten gjelder også der helseopplysningene behandles for historiske, statistiske eller vitenskapelige formål, og behandlingen ikke får noen direkte betydning for den registrerte.*

Den registrerte har rett til innsyn i hvem som har hatt tilgang til eller fått utlevert helseopplysninger som er knyttet til den registrertes navn eller fødselsnummer, fra helseregistre etter §§ 8 til 11. Departementet kan i særlige tilfeller gi den databehandlingsansvarlige en tidsbegrenset dispensasjon fra plikten til å gi innsyn etter dette leddet.

Når det er nødvendig for å vurdere innsyn kan den databehandlingsansvarlige innhente personopplysninger fra Det sentrale folkeregisteret. Dette gjelder uten hensyn til taushetsplikt."

This report is based on the following laws and regulations:

1. [HRL] (2014): *Lov om helseregistre og behandling av helseopplysninger (helseregisterloven)* https://lovdata.no/dokument/NL/lov/2014-06-20-43
2. [MFR] (2001): *Forskrift om innsamling og behandling av helseopplysninger i Medisinsk fødselsregister (Medisinsk fødselsregisterforskriften)*, https://lovdata.no/dokument/SF/forskrift/2001-12-21-1483
3. [DÅR] (2001): *Forskrift om innsamling og behandling av helseopplysninger i Dødsårsaksregisteret (Dødsårsaksregisterforskriften)*, https://lovdata.no/dokument/SF/forskrift/2001-12-21-1476?q=D%C3%B8ds%C3%A5rsaksregisteret

4. [HJK] (2011): *Forskrift om innsamling og behandling av helseopplysninger i Nasjonalt register over hjerte- og karlidelser (Hjerte- og karregisterforskriften)*, https://lovdata.no/dokument/SF/forskrift/2011-12-16-1250?q=Hjerte-+og+karregister

It is not always evident which operational measures should be implemented to fulfil the requirements stated by relevant laws and regulations. Whenever there is doubt, FHI is committed to follow the norms for information security in healthcare (http://www.normen.no). The norm is chosen as the basis for the security requirements.

The following regulations are taken as basic:

Document and references	Date
[Normen] Norm for informasjonssikkerhet, helse- og omsorgstjenesten, 5. utgave (versjon 5.1) www.normen.no	June 2015
[27002] Norsk Standard: NS-ISO/IEC 27002:2013 - Informasjonsteknologi - Sikringsteknikker - Tiltak for informasjonssikring http://en.wikipedia.org/wiki/ISO/IEC_27002#Outline_for_ISO27002:2013 og http://www.standard.no	January 2015

We also refer to FHI's internal security audit of the logging process, and to the response and solution to the vulnerabilities found during the audit.

1. Sikkerhetsrevisjon av logging i helseregistrene – Folkehelseinstituttet (FHI), november 2015, rev. 30.11.2015.
2. Comments and response to the above audit report.
3. Løsningsforslag MFR ver. 1.0.

2.4 Logging requirements

The logging system is developed to satisfy the following two requirements:

- *Security logging* to detect attempts at unauthorized access to data and to monitor the stability of the system's operation.
- *Technical logging* to provide information needed to respond to requests for personal insight (innsyn) in health data as required by the Health Register Law § 24 (see Sec. 2.3).

The security logging is mostly based on functionality for system logging provided by a database system (e.g. Oracle's FGA technology). Examples of events monitored by the security logging are user logons/logoffs, deletion/changes of critical data objects, and the granting of privileges or roles to users. The best technical logging is achieved when it is implemented inside the applications that access sensitive data. (Here, it is assumed that the data is only available through applications.) However, in legacy systems (like MFR) where it is very costly or hard to modify applications, logging in the database itself can be used to realize technical logging (see use of FGA in MFR in Sec. 3.2).

In the following we distinguish between a *production database* and a *datawarehouse* (utleveringsdatabase). Operations in a production database are strictly defined and are limited to operations such as insertion of new data, quality control, and simple analysis. Knowing the available functionality in a database, appropriate applications should be developed with built-in technical logging. Application developers have the best opportunity to judge if an event should be logged and how to do smart collection of logged information because—unlike database

developers—the application developers have information about the contexts in which data are used.

The situation for a datawarehouse is quite different. Authorized personnel have full access to the data (e.g. they can issue any SQL query on the data). Therefore, encryption of Direct Person Identification (DPI) data should be a standard for all datawarehouses. All calls to the encryption/decryption function should be logged inside the function itself or at the system level using the system's own logging functionality. Depending on the logging solution, we face the problem that logging is performed for events that need not be logged. Examples are

- Automatic processes that do not produce DPI data visible to humans.
- Bulk encryption/decryption when, say, all (a few million) table entries are decrypted and the security context (whether it was an authorized action) was checked.

Avoiding logging in the above cases will drastically reduce the volume of logs and simplify the log analysis. To reduce log sizes, it is necessary to answer questions like, when an authorized user runs an SQL query requesting decryption of 100 000 data entries should it be logged? What about a query involving 6 million entries?

Logging becomes useless when a system records an enormous number of events because it is impossible to analyse all of them in a meaningful way. The goal should therefore be to ***minimize the need for decryption in the datawarehouse***. To achieve this goal, common data analysis routines, in particular data linkage, should be performed without decrypting DPI data but instead using encrypted values.

The issues of substituting important session information (e.g. user name, process name, object name, IP address etc.) should be considered. Note that data for logging is largely collected from environment variables that can be replaced in some contexts.

An application for internal control of logs should be built on top of the logging system. The task of internal control cannot be managed manually because of the large data volumes. Log data should also be stored in a structured and indexed format to facilitate access and analysis. Logs should be monitored continuously to detect attempts at unauthorized access to data, especially unauthorized change.

3 LOGGING OF DATABASE ACCESS

This section describes the logging features implemented in Medisinsk fødselsregister (MFR), Hjerte- og karregister (HKR), and Dødsårsaksregister (DÅR) utleveringsdatabase. It also describes solutions being implemented in mMFR at the time of this writing.

3.1 Logging solutions for Oracle platform

The section first addresses logging and encryption approaches which are common to the register data components hosted on FHI's Oracle platform. The register data components addressed below are found in MFR, HKR, and DÅR.

We describe the logging features implemented for each of these Oracle-based register data components. Here, we consider separately the MFR production database and a Data Warehouse (DW). The logging features for these databases differ. For purposes of supporting statistical production as well as the delivery of data for research purposes, the DÅR register also includes a component called the DÅR utleveringsdatabase. This data component is hosted on FHI's Oracle platform.[1]

Logging features in the MFR and HKR registers and DÅR utleveringsdatabase make use of two basic logging solutions provided by Oracle:

- *System Logging* **(**SL**)** provides means for imposing logging at the object level. The objects that logging can be imposed on are data objects (e.g. tables and views), user activities (e.g. logons and logoffs), and execution of procedures.
- *Fine Grained Audit* **(**FGA**)** provides means for imposing logging at the column level of selected data objects. Log event occurs every time someone issues the SELECT command on a column being under logging control.

Every event for which SL was set up leaves a record in the System Log Table. In the case when a logged function is called, the record contains the following information:

- When and who executed the function.
- The SQL text and the associated bind variables.

In the case when user activity is logged, the record contains the information:

- When and who executed the activity.
- What activity was executed and what privilege was used.
- Return code showing whether the activity was completed.

FGA logging allows us to define what information we want to have in the log file. Below, we provide more details on what information we log using FGA.

3.2 MFR production database

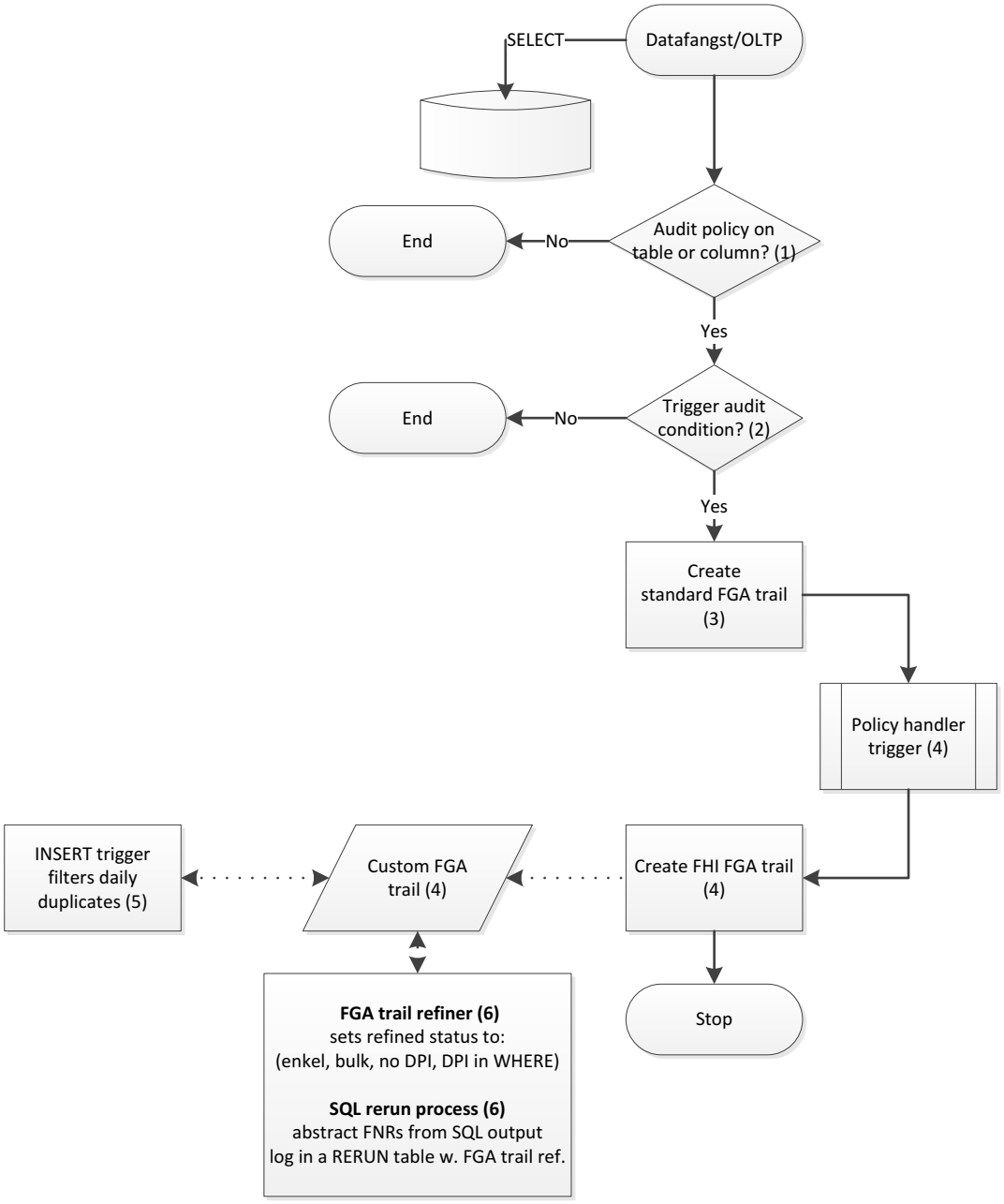
For the MFR production database, the following procedures are invoked:

- All calls of the encryption/decryption function are recorded by SL.
- Logons and logoffs (both successful and unsuccessful) of all users are recorded by SL.

[1] For specific details explaining the division of DÅRs data servers across Microsoft SQL Server vs. Oracle platforms, see section 5.2 of the document specifically concerning DÅR.

- Columns that contain personal data are identified in all the tables in the database. Every SELECT command on an identified column is logged by FGA.

The following diagram depicts the implementation of FGA.



1. An FGA policy is defined for every column that contains personal data. The policy is triggered each time a SELECT command is issued on the column protected by the policy.
2. Each time the policy is triggered, the custom-defined audit condition is being checked first. In the audit condition, we define instances when the policy is triggered due to system processes running that do not involve humans. Such instances are not logged.
3. If the audit condition is fulfilled, the record is created by system in the *standard* FGA trail.

4. After that, the audit handler procedure is called. In this custom procedure, we define what data to record in the *custom* FGA trail. The custom FGA trail is a table created in advance to contain all the data we want to log when the FGA policy is triggered. This data is collected in the handler using system environment and FGA variables. In particular, this table contains:
    - When and who and executed the command that triggered the FGA policy.
    - Which column with personal data was accessed.
    - The SQL text and connected bind variables that triggered the policy.
    - The System Change Number that identifies exactly the state of the database when the action was executed. The number allows us to rerun the action, if needed, in the same state the database was at the time of the original execution.
5. INSERT triggers on the custom FGA trail. It will check and eliminate obvious duplicate entries to reduce the log size considerably.
6. An hourly scheduled process will take new entries from the custom FGA trail and analyse the recorded SQL queries. In the case when personal data is found in the SQL, this data is abstracted, encrypted, and recorded in the separate table that contains a reference to the relevant row in the custom FGA trail. In the remaining cases, personal data returned by the query is obtained by rerunning the SQL using the actual System Change Number and flashback technology.

Applications used by the production staff at MFR have their own logging features. Firstly, the IK1002 application for processing "fodselsmelding" and "barnemelding" on paper encrypts all the fields with personal data in the TIF file for every paper form scanned. Every lookup of personal data in a TIF file is logged. Also, logged are the cases when personal data is uncovered by the application. The log contains a MELDINGID field that allows identification of a person whose data was accessed. Secondly, similar logging features are provided by the application for processing electronic "fodselsmelding" and "barnemelding" (XML files).

Note that in the modernization project, the current MFR Prod will be replaced. In the new database, the users will be given access to the data only by the WEB applications that will also implement all the logging of data access. For this reason, the need for FGA and SL monitoring of encryption/decryption will be eliminated on MFR Prod.

3.3 MFR DW, HKR and DÅR Utleveringsdatabase

All databases are encrypted using shell encryption technology (for Oracle databases known as Transparent Data Encryption (TDE)). In addition, all sensitive DPI data is encrypted in MFR DW, HKR, and DÅR utleveringsdatabase using AES. Therefore, to provide complete monitoring of all instances when the plain personal data is accessed in MFR DW, HKR and DÅR utleveringsdatabase, it is sufficient to log all the calls to the encryption/decryption function. The following logging features are empowered:

- All calls for encryption/decryption function are recorded by SL.
- Logons and logoffs (both successful and unsuccessful) of all users are recorded by SL.

3.4 mMFR

Currently, every register has its own logging solution. Most of the implemented technical logging requires technical expertise to make lookups in the logs. Furthermore, it is very time consuming to do lookups across registries. Therefore, an important feature to be included in

every register modernization project is the use of a common solution**,** or *component*, for logging and tracing in all registers. The common component should satisfy the following requirements:

- Only register owners should be able to specify for how long logs are to be kept.
- Log data should be used to monitor the security and stability of the system. Criteria of abnormal behaviour should be developed that include both indicators of unauthorized access and threats to system stability.
- Logs must handle and store huge amounts of data. In some cases, the data volumes equal to the whole register will be logged.
- Logs have to be protected since they contain sensitive data and since they can be misused.
- Parts of the logs should be accessible for the personal insight.
- A log file must contain the following minimum of information:
    - Who ran a database lookup disclosing personal data, decryption or encryption (identity or process).
    - When the disclosure occurred.
    - Whose data was disclosed (identity).
    - In which register the disclosure occurred.
    - Why (context, e.g., coding, case processing, data delivery, quality assurance).
    - Assignment/project number for data linking or delivery of data.
- It should be possible to exclude database procedures from being logged, including automatic procedures and batch jobs that do not provide an output which can be accessed by humans.
- Various reporting solutions have to be developed.

Every call to an encryption function need to be logged to protect against attempts of linking personal and health data by the use of encrypted identities.

Currently, the same person has access to both the key file and anonymized data sets. This means that every instance of data linking will be logged despite the fact that data from FHI is delivered anonymized. Therefore, there should be established a strict distinction between the user roles that have access to the key file (are authorized to run decryption) and those that have access to anonymized data sets. Implementing this feature requires major changes in the work processes in the production of research data.

4 SUGGESTED IMPROVEMENTS

This section suggests how to improve existing logging solutions.

4.1 Log analysis system

Firstly, we should stress that the most important new functionality needed is a set of routines for internal control of logs. Because of the huge amounts of log data, it is not possible to manually control logs. Therefore, a separate system has to be developed to meet this need. Here are two aspects of log data analysis that need to be considered.

1. Log data should be regularly and automatically analysed for attempts at unauthorized data access. It is important to develop common policies for FHI that define what events are considered an indication of unauthorized access. Furthermore, routines for investigating such cases should also be developed. These criteria and routines will define the functionality required from the log analysis system.
2. Log data will be used when responding to requests for personal insight in health registers. This functionality should also be part of the log analysis system.

Since logging and tracing must handle large amounts of data, other database technologies than traditional relational databases should be considered. Queuing technologies and a document database could be used. "Big data" products should be considered for log analysis. The large data volumes and the need for powerful analytic tools make it preferable to host future solutions in computing clouds. Since the log will contain a field with personal identifiers, this field must be encrypted or replaced with a surrogate key before the data leaves FHI.

4.2 Encryption of person-identifiable data

All register data that identifies a person should be encrypted and calls to the encryption/ decryption function should be logged. As stated earlier, we have a problem when bulk decryption applied to (nearly) a complete table is logged and generates a large amount of log data. In many cases, the value of the log data produced by bulk decryption is close to zero since it may seem that health data for very many people is accessed several times a day. Therefore, suggested measures are:

1. Sensitive data from third parties should be encrypted before the data is stored in FHI's databases (like DSF, quality registers, non-FHI health registers, etc.).
2. Develop routines for joining the data from different tables using encrypted identities.

4.3 Application logging

It was argued earlier that it is best to implement logging at the application level. Each application should have internal mechanisms to decide the activities to be logged and the data to be placed in the audit log. This approach to logging allows the application developers to minimize the amount of logged information. Note that the developers must update the logging when new functionality is added to existing applications.

Most users should only be allowed to access data via applications and not be allowed to run free SQL queries. If application logging is implemented, logging in the database system can be used exclusively to audit user activities that involve custom SQL queries. There are almost no such queries run in a production system, which means that the system logging will just record suspicious/abnormal activities and will not significantly increase the system's work load.