Security Working Group

Sub-Resource Access in FHIR – Security and Privacy Requirements

Mohammad Jafari,
Kathleen Connor, John M. Davis, Christopher Shawn

Version 1.1

(Draft for Comments)

March 9, 2020

Table of Contents

List of Figures

List of Tables

1 INTRODUCTION

This report presents and discusses security and privacy requirements for a range of technologies that enable granular clients' access, at the level of resource attributes in the context of the Health Level 7 (HL7) Fast Healthcare Interoperability Resources (FHIR).

FHIR resources are well-defined units of healthcare (or healthcare system) information, such as Patient, Encounter, or Observation. FHIR defines a standard interface based on Representational State Transfer (REST) to enable clients to read, create, update, delete, or search for, such resources. FHIR's REST interface allows a client access to individual resources (or a collection of resources); in other words, the smallest unit of data in FHIR REST interface is a FHIR Resource.

A number of emerging technologies and specifications, which will be referred to as Sub-Resource Access (SRA) in this report, take this a step further in terms of granularity and enable clients to request access to individual attributes of a resource and composing new data objects based on the attributes taken from different resources.

There are various efforts in FHIR that are aimed at, or will lead to, providing SRA. The most noteworthy ones are discussed in Section 3. While we consider SRA services in broad terms, for the purpose of this report, our primary focus is on GraphQL [1] which is the most common technology in the developer community, with growing interest and relatively higher maturity, and is also widely used in other application domains outside of FHIR and the healthcare sector.

1.1 Scope of This Report

The focus of this report is on requirements specific to SRA technologies and their use. Every application using SRA ultimately has to accommodate further security and privacy requirements pertinent to its specific application area, or other technologies used which are often discussed in application-specific implementation guides. In other words, the requirements in this report must not be construed as comprehensive for any specific application and must be considered alongside all other applicable security and privacy requirements.

2 APPLICATIONS OF SRA

Different reasons have led to the design and implementation of services to provide sub-resource access. This section discusses some of main reasons for developing and supporting SRA.

2.1 Fine-Grained Authorization

Some applications depend on access to non-sensitive attributes from resources, in cases where access to the entire resource could be either unauthorized for that application. For example, a client requesting access to data for research purposes may not be authorized to access the entirety of an immunization resource, but could be granted access to see the date or the location of the immunization in order to make some statistical analyses. Or, a client for a public health monitoring application may require access to immunization date, location, and type, perhaps in aggregate from, without seeing the attribute that links the immunization resource to an individual patient.

These types of queries which are often in the form of aggregation (e.g., sum, average, or other similar functions) are common in applications like data mining, population-level health, and different types of research.

The SRA requirement for many of these applications is currently addressed by using de-identification algorithms that mask or transform the resource into a data object that only includes what the client is authorized to see. This is by definition a form of sub-resource access which will be discussed further in Section 6.2.3.

2.2 Need-to-Know Access

Need-to-know is a fundamental principle in access control which stipulates that users must be granted access within the minimum extent required for performing their duties and tasks. This is often determined by assigning users to organizational roles with well-defined sets of permissions, and/or tasks within organizational workflows which define how business operations must be conducted.

At the inter-organizational level, access to health information is often granted at the granularity of the entire FHIR resources or bundles of resources, or even an entire patient record document. For example, when an organization is authorized to receive the record of all or some specific immunizations for a patient, this means that organization will receive a bundle of FHIR resources. At the intra-organizational level, however, permission to access is often granted at a more fine-grained level on a need-to-know basis. For example, a third party admin desk to which scheduling immunization appointments is outsourced can see the patient name, contact information, and the dates of most recent immunizations (attribute of the immunization resource), but may not be authorized to see the type of immunization administered.

2.3 Application-Specific Sensitive Information

Sensitive data is usually identified by overarching laws, regulations, and polices based on common-sense expectations of privacy and social norms. For example, in the US, federal regulations (42 CFR Part 2) stipulates the general rule that history of substance abuse must be treated as confidential information.

In some particular applications areas, however, certain resource attributes, which are not considered sensitive or confidential in the general sense, could be considered sensitive in specific application contexts or for a particular resource instances. For example, a patient's address could be sensitive for particular patients in the context of facilities hosting victims of spousal abuse [2]. This means that even though patient address is not considered sensitive in general, in this specific

application domain, some patient's address could be considered sensitive and require additional protections. Accommodating this type of privacy control requires enabling granular access for clients to avoid unnecessary blocking of information.

2.4 Access Efficiency

One of the major motivations for SRA is efficient retrieval of information. In the REST model, it is common for a client to fetch some resources from a service, pick and choose certain attributes from each resource to ultimately compose the data object it needs. The top part of Figure 1 depicts this process.

For example, consider the user interface for a receptionist at an immunization facility which includes the name, address, and phone number of a patient as well as the dates and locations for their most recent three immunizations. The name and phone number of the patient are part of the Patient resource in FHIR which can be fetches with a REST call to get the Patient resource. The dates for each immunization is an attribute within the immunization resource, so, fetching the last three immunization dates requires a search query that returns the most recent immunization resources from which the date attribute is, then, extracted.

In the above scenario, the client needs to make two requests for each patient, then parse and process the returned resources to extract a tiny fraction of the data returned by each of those queries. For more complicated user interfaces, this could lead to receiving an excessively large volume of data over several network requests which could lead to significant network and parsing overheads. The idea of SRA (in particular GraphQL) was proposed to tackle this issue. Instead of requesting large units of data, most of which will be unused, the client can request exactly the attributes it needs it needs to fill the user interface and, thus, avoid the overhead of unnecessary network requests and receiving and parsing unneeded data. The bottom part of Figure 1 depicts this model and provides a comparison between the two approaches.

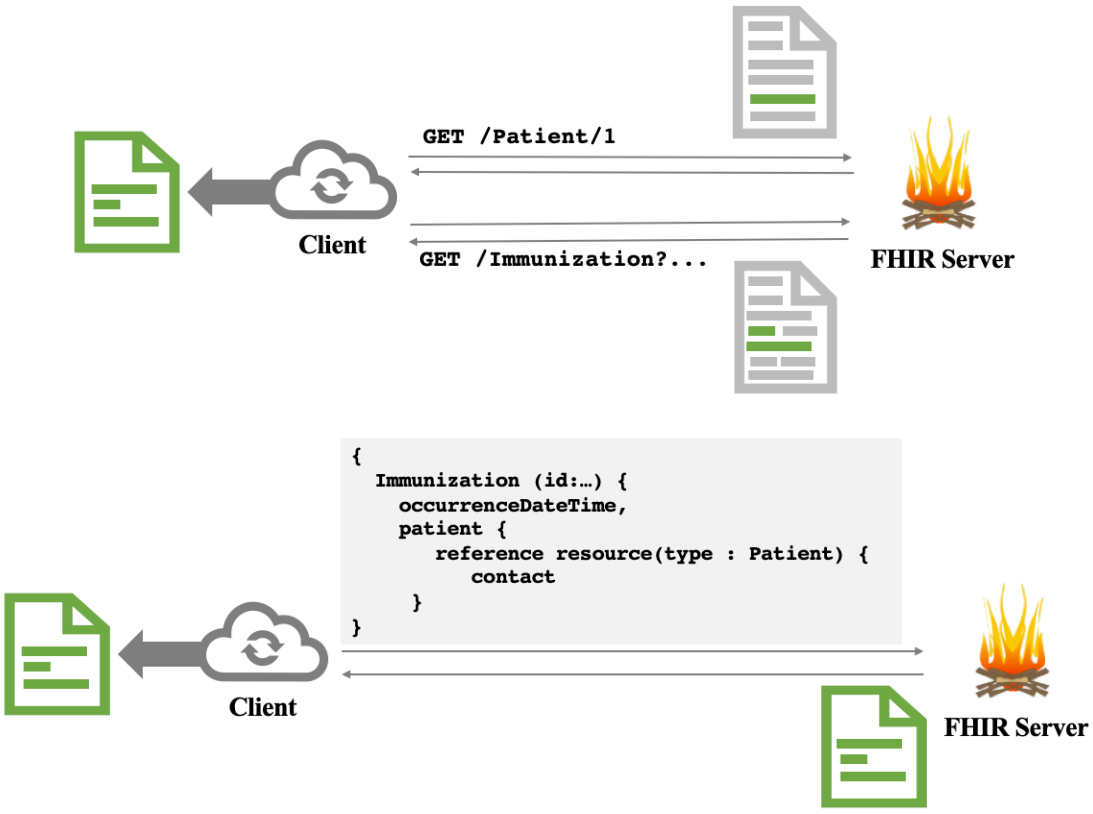

Figure 1: Comparison between access at resource level (top) such in the REST model vs. access at sub-resource level (bottom) such as the GraphQL model

3 SRA TECHNOLOGIES AND SPECIFICATIONS

3.1 FHIRPath

FHIRPath is a set of specifications that define a language for extracting attributes from FHIR resources using path-based queries and based on operations such as traversal, selection, and filtering of the hierarchical data contained in a resource [3]. For example, the following FHIRPath expression selects the display name for the site of immunization administration (e.g., "left arm"):

```
Immunization.site.coding.display
```

Or the following FHIRPath expression checks whether any contact information of type mobile phone exists for a patient resource:

```
Patient.telecom.exists(system='phone' and use='mobile')
```

FHIRPath provides a powerful language for sub-resource access queries which is also used in other sub-resource access technologies such as the current draft for FHIR GraphQL [4].

3.2 SQL on FHIR

Structured Query Language (SQL) is a well-established standard query language for the data stored in relational databases. SQL on FHIR is an effort to define a language inspired by SQL to query the information stored in FHIR resources [5]. For example, the following query selects the display names for the site of immunization administration for all the immunizations taking place after 2019:

```
SELECT site.coding.display FROM immunization
WHERE occuranceDataTime > '2019';
```

Compared to FHIRPath, SQL on FHIR has been much less widely used but some commercial FHIR products support it for analyses and aggregation applications.

3.3 GraphQL

GraphQL is a data query language that enables clients to query data from a web service at the attribute level and by defining the precise shape of the data objects needed. Driven by the needs of front-end user interface development (as discussed in Section 2.4), the goal of GraphQL is to enable the client to define its own precise data model and as a result, minimize parsing, processing, and network communication overheads and the volume of data transferred between the server and the client. In other words, the burden of processing and curating data and turning it into the right shape for use by the client is shifted from the client to the server.

This contrasts with the REST model in which a client usually has to query the data at the granularity of resources as pre-defined by the server, receive a potentially large volume of data, and parse and process this data to turn it into the required shape. A comparison of these two paradigms is presented in Figure 1.

GraphQL was initially developed and designed by Facebook in 2012 and was later turned into an open-source project owned by the GraphQL Foundation. It has been widely used in many modern web applications and is supported by most emerging technologies, new programming language, and web platforms, as well as major industry stakeholders. A draft specification for FHIR GraphQL is currently available [4] and is already supported at an experimental level by some of the major reference implementations.

Figure 2 shows a simple example of a GraphQL query and response in FHIR based on the current draft specifications. The query defines a simple data object by finding a patient with the id `231`. This object is composed of the patient ID, the patient's official given and family names, and the patient's phone number. The returned object only includes the attributes requested by the client in the composition defined by the query and no other attributes are included.

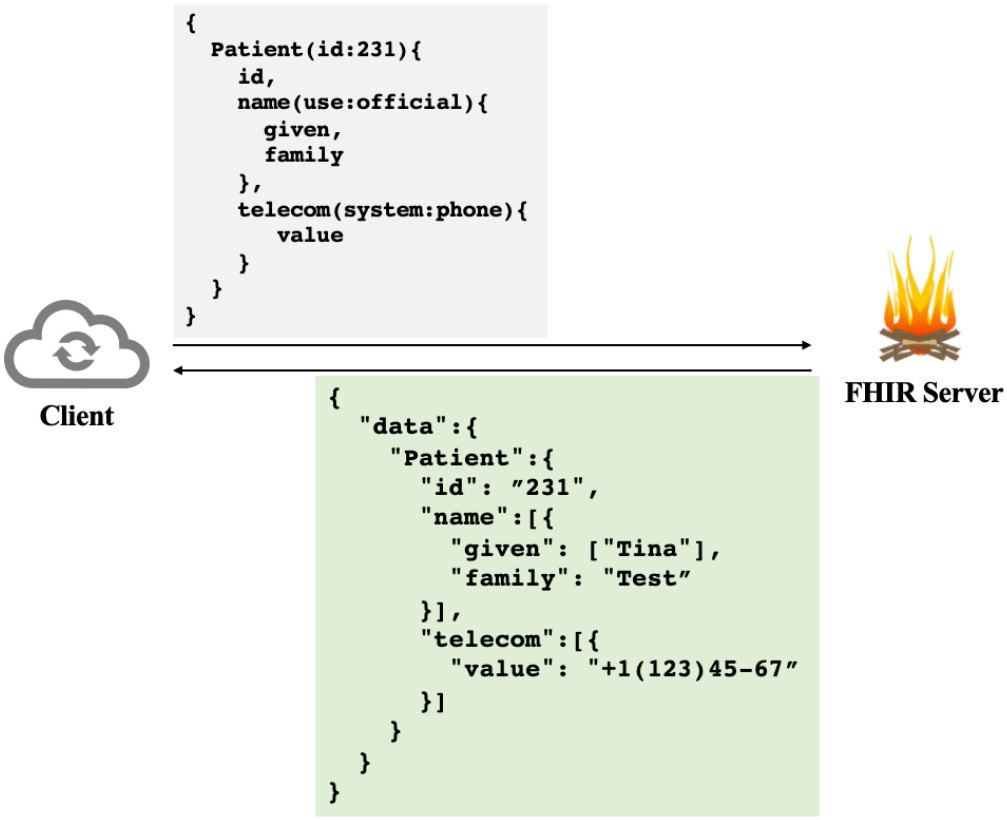

Figure 2: An example GraphQL query and response in FHIR based on the current draft specifications

4 ANATOMY OF AN SRA SERVICE

Since the SRA service follows a different query model from that of typical RESTful web services, a brief discussion of the processing model is provided in this section. As shown in Figure 3, this discussion is focused on the placement of fulfilling privacy and security requirements and the interplay with standard security and privacy services, namely:

- Access Control Service (ACS),
- patient Consent Service,
- Security Labeling Service (SLS),
- Privacy-Preserving Services (PPS), and
- Provenance Service.

The two major components of an SRA system are: Query Analyzer and Query Processor. The Query Analyzer is in charge of parsing and validating the SRA query from a client and ensuring its syntactic and semantic validity. This is where the SRA *understands* the meaning of the query and what the client is asking for. At this stage, since the SRA can identify the precise meaning of the client's query, it is able to consult with the ACS to make authorization decisions as to whether or not the client is authorized to make this query based on its shape. In other words, whether the client is permitted to ask *the kind of* question it is asking.

The Query Processor fulfills the client's query by collecting, curating, and adjusting the data from the FHIR server, or directly from the database where FHIR resources are stored. This is where the SRA prepares the response to the client's query. At this stage, the Query Processor can make the necessary consultations with other security and privacy services to determine:

- whether there is anything in the client's response to which the client is not authorized access (by consulting with ACS),
- the security labels applicable to the composite data object in the client's response (by consulting with SLS),
- any fine-grained filtering or redaction required from any collection of data requested by the client based on the client's clearance and the security labels associated with those data items (by consulting with the PPS), and
- the provenance for the composite data object in the client's response (by consulting with Provenance Service).

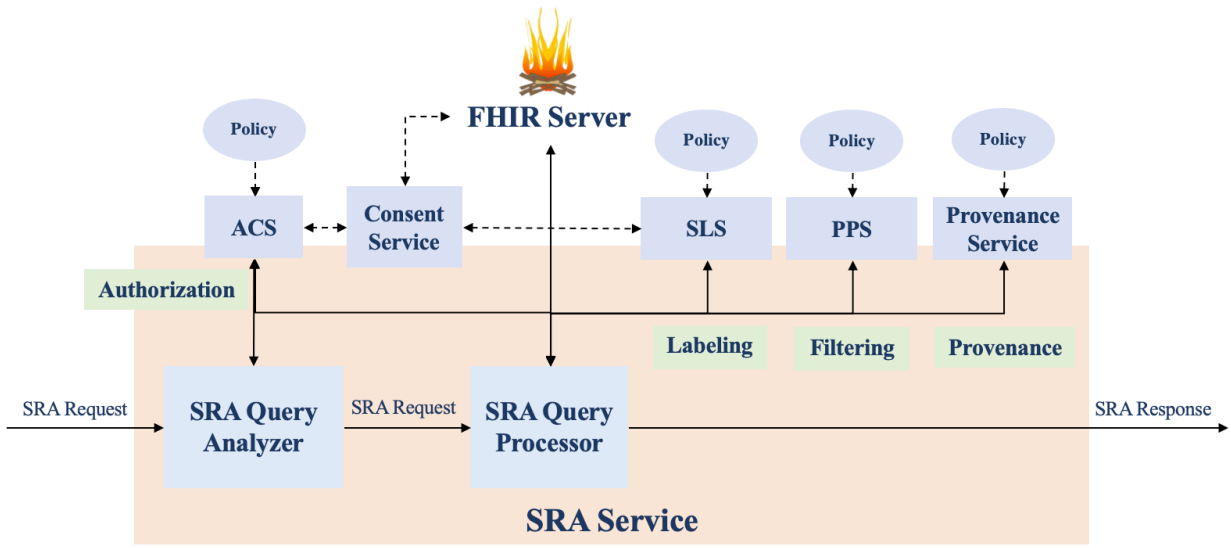

Figure 3: Anatomy of an SRA Service and its interactions with other security and privacy services to fulfill the security and privacy requirements

4.1 Walk-Through

Consider a simple case where the client sends the following query to the SRA:

```
{
  Patient(id:231){
    id,
    name(use:official){
      given,
      family
    },
    telecom(system:phone){
       value
    }
  }
}
```

In this request, the client provides an access token previously acquired from an Authorization Server (e.g., OAuth 2.0). Upon receiving this query, the following steps take place within the SRA:

- The SRA ACS module extracts the client's access token and determines the client's granted permissions and clearances (e.g., using OAuth Token Introspection [6]).
- The Query Analyzer receives and parses this query and determines the data types and the validity of the request.
- By consulting with the ACS, the Query Analyzer determines whether the client is authorized to make the query based on the query type and shape. This decision will be based on the query, the client's granted permissions, and the access control policies. In this case, the client is requesting access to patient names and phone numbers, so, at this stage the ACS must determine whether the client is authorized (in principle) to access that information for patients (regardless of any individual patient's preferences).
- If the request is deemed authorized, the Query Processor starts determining how to fulfill the query and compose a response. In this case, all the information is coming from a single

FHIR resource, a patient with ID 231. After fetching this resource, the Query Processor extracts the requested attributes and composes the data object requested by the client. This object looks like the following:

```
{
  "data":{
    "Patient":{
      "id": "231",
      "name":[{
        "given": ["Tina"],
        "family": "Test"
      }],
      "telecom":[{
        "value": "+1(123)45-67"
      }]
    }
  }
}
```

- By consulting with the ACS again, the Query Processor determines whether the client is authorized to receive the requested data. For example, in this case, the ACS is consulted to make sure that the client is authorized to receive the contact information of patient with the ID 231. The answer could be negative if access to this information is unauthorized for this individual patient (e.g., based on patient consent). Note how this is different from the ACS check occurring at the Query Analyzer; at that stage, the question is whether the client is authorized to ask about *any* patient's contact information, regardless of the particular patient in question.
- By consulting with the SLS, the Query Processor determines the security labels that must be assigned to this data object. This is based on labeling rules and policies governing the SLS but can be conducted using different techniques and methods as will be discussed later in this report.
- By consulting with the PPS and based on the client's clearances, the Query Processor determines whether any of the data items within a collection must be filtered out (i.e., redacted). This decision can be made in cases where the client is in principle authorized to make the query and receive the results but there are individual data items that are not authorized to be included in the results, for example, based on the patient consent, or the level confidentiality that exceeds that client's clearance. In this particular example, no filtering or redaction is applicable.
- Optionally, and in use-cases where this matters, the Query Processor also consults with the Provenance Service to establish the provenance for the composed data object. In this case, this could simply be a link to the provenance of the one FHIR resource used in composing the response (i.e., the Patient resource). In cases where more resources are involved and the response data object is composed based on the attributes from many FHIR resources, the provenance will be more complex, for example, in the form of a bundle. Tracing the provenance for each attribute (i.e., the ability to track where each attribute value has come from) is a far more complex case which is applicable if the use-cases in question call for it.

5 REQUIREMENTS

As shown in Figure 3, there are four groups of security and privacy requirements for sub-resource access:

1. *Authorization* requirements specify whether or not an SRA service should permit a request *at all*, based on policies, request context, and the client's credentials and granted clearances.
2. *Labeling* requirements cover the requirements for assigning security labels to outgoing data objects resulting from fulfilling an SRA request.
3. *Filtering and Redaction* requirements specify the capabilities of an SRA system to optionally remove certain values from collections based on the client's clearances.
4. *Provenance* requirements specify the capabilities for tracking and recording the provenance for the data objects resulting from fulfilling an SRA request.

These requirements are summarized in Table 1 and are discussed in the rest of this section.

Table 1: Summary of Requirements

Title	Description	Involved Services
1. Authorization		
1.1. Distinct Client Permission for SRA	The SRA service SHALL require a distinct permission by the client for making SRA requests.	ACS
1.2. Authorization Based on the Implied FHIR Resource Types	The SRA service SHALL make authorization decisions based on the FHIR resource types implied by the client's query.	ACS
1.3. Authorization Based on the Implied FHIR Resource Attributes	The SRA service SHOULD make authorization decisions based on the FHIR resource attributes implied by the client's query.	ACS
1.4. Authorization Based on the Applied Aggregation-Level	The SRA service SHOULD make authorization decisions based on the level of granularity and aggregation applied to FHIR resource attributes implied by the client's query.	ACS
1.5. Authorization Based on Patient Consent at the FHIR Resource Level	The SRA service SHALL enforce patient consents when a FHIR resource related to the patient is included in an SRA response.	ACS, Consent Service
1.6. Authorization Based on Patient Consent at the Attribute Level	The SRA service SHOULD enforce patient consents when attributes of a FHIR resource related to the patient are included in an SRA response.	ACS, Consent Service
2. Labeling		
2.1. Object-Level Labeling	The SRA service SHALL determine and assign security labels to the composite data object created as the SRA response.	SLS
2.2. Attribute-Level Labeling	The SRA service MAY determine and assign security labels to the individual attributes and data objects created as the SRA response.	SLS

Title	Description	Involved Services
3. Filtering and Redaction		
3.1. Clearance-Based Filtering at Resource Level	The SRA service SHOULD redact values from object collections included in the SRA response based on client's clearances.	SLS, PPS
3.2. Consent-Based Filtering at Attribute Level	The SRA service SHOULD enforce patient consent by redacting values from object collections included in the SRA response.	SLS, PPS
4. Provenance		
4.1. Object-Level Provenance	The SRA service SHALL establish the provenance of the composite object created for the SRA response.	Provenance Service
4.2. Attribute-Level Provenance	The SRA service MAY establish the provenance of the individual attributes in the object created for the SRA response.	Provenance Service

5.1 Authorization

These requirements articulate the level and extent of control the SRA service needs to assert over whether or not a request is permitted. Such decisions are based on policies, the client's credentials and granted clearances, and the content of the SRA request and response.

Note that authorization decisions are also made by considering the applicable jurisdictional and organizational policies, client's granted permissions, and other contextual information about the request such as purpose of use. So, when a requirement in this section stipulated that the SRA should make authorization decisions based on some factor, it is in addition to these standard authorization factors.

Some of these requirements are purely based on the analysis of the SRA request, so the decision can be made before the response is created or known, while others depend on the information that will appear in the response, so, they will have to be made after the SRA response is created.

5.1.1 Distinct Client Permission for SRA

The SRA service SHALL require a distinct permission by the client for making SRA requests.

To mitigate the risk of unauthorized access, this requirement ensures that the availability of an SRA service does not automatically open it up to every client with access to the FHIR server, and only clients that are intentionally and explicitly granted access to the SRA service can make SRA requests.

5.1.2 Authorization Based on the Implied FHIR Resource Types

The SRA service SHALL make authorization decisions based on the FHIR resource types implied by the client's query.

This requirement ensures that after determining the types of the FHIR Resources used in creating the SRA response, the SRA must ensure that the client is authorized to access those resource types. For example, if a client does not possess the permission to access immunizations, any SRA request by that client which requires information from immunization resources will be rejected.

5.1.3 Authorization Based on the Implied FHIR Resource Attributes

The SRA service SHOULD make authorization decisions based on the FHIR resource attributes implied by the client's query.

This is a more fine-grained version of the previous requirement which requires decision-making based on FHIR Resource attributes, rather than just FHIR Resource types. For example, if a client does not possess the permission to access immunizations types, an SRA request by that client which requires specific access to the attributes carrying immunization type will be rejected. The same client however can proceed with a request that includes access other attributes of immunization such as the immunization date. Note that if the decision is made at the level of FHIR resource type (as required by the previous requirement), both of those requests would have been rejected, so the finer level of granularity enables more data access and less blocking.

5.1.4 Authorization Based on the Applied Aggregation-Level

The SRA service SHOULD make authorization decisions based on the level of granularity and aggregation applied to FHIR resource attributes implied by the client's query.

This requirement ensures that the SRA is able to distinguish between access to individual attribute values versus aggregate-level access. For example, a client may not be authorized to access the individual records of immunization type for individual patients, but it may be deemed authorized to access the total number of immunizations per type within a (large enough) group of patients. This requirement ensures that the SRA is able to factor in the level of granularity and aggregation when analyzing the FHIR Resources and attributes implied by a client's query.

Note that enabling access to aggregate results is only safe when the number of data items included in the aggregation is large enough, otherwise, it can compromise the privacy of individual values incorporated in the aggregate. Thus, determining the safety of access to aggregate results often requires determining the response to ensure that the data set targeted by the query is large enough.

5.1.5 Authorization Based on Patient Consent at the FHIR Resource Level

The SRA service SHALL enforce patient consents when a FHIR resource related to the patient is included in an SRA response.

This requirement ensures that, where required by policy, access to FHIR Resources related to a patient is subject to the patient's consent. So, if a patient consent does not allow sharing of a resource with the client (e.g., because of the client's identity or the purpose of use), an SRA request targeting such resources is rejected. In other words, the SRA service must not create a backdoor access to FHIR Resource where it was otherwise unauthorized.

5.1.6 Authorization Based on Patient Consent at the Attribute Level

The SRA service SHOULD enforce patient consents when attributes of a FHIR resource related to the patient are included in an SRA response.

This is a more fine-grained version of the above requirement. The previous requirement allows the SRA to deny access if a resource is deemed unauthorized for the client based on the patient consent, without really delving into, and distinguishing between the attributes of that resource. This requirement ensures that this analysis is more granular and at the attribute level. For example, overarching policies, or a fine-grained patient consent may allow sharing the immunization dates

and types for population health purposes but not the link to the patient who is the subject of the immunization.

5.2 Labeling

This section covers the requirements for assigning security labels to outgoing data objects resulting from fulfilling an SRA request.

5.2.1 Object-Level Labeling

The SRA service SHALL determine and assign security labels to the composite data object created as the SRA response.

Some security labels applicable to the outgoing data object in the SRA response depend simply on the client and context of access (e.g., purpose of use). But other security labels, such as confidentiality or sensitivity labels usually depend on the content of the outgoing data (i.e., the attributes included) and the security labels of the FHIR Resources from which they originate. Thus, the SRA service must often compute the security labels applicable to the composite object based on its data ingredients. This requires the ability to segment the data at the sub-resource level which is discussed in Section 6.2.

Note that currently there are no specifications for communicating the security labels on an outgoing SRA response (e.g., in FHIR GraphQL draft). Since in SRA services like GraphQL, the shape of the outgoing object is defined and determined by the client, it is not possible for the server to include arbitrary extensions in the outgoing response object to communicate security labels. Thus, either a separate metadata container object must be defined and included in the SRA response, or the security labels must be recorded using the response headers.

5.2.2 Attribute-Level Labeling

The SRA service MAY determine and assign security labels to the individual attributes and data objects created as the SRA response.

This is an optional requirement which stipulates a more granular level of the previous requirement. Instead of determining the security labels for the entirety of the outgoing object, this requirement ensures that the SRA is capable of determining the security labels applicable to each attribute of the outgoing composite object. There are not many use-cases where communicating security labels at the attribute level to the client is called for. However, determining security labels at the attribute level, is a key to accurate determination of the object-level labels based on the high watermark principle as discussed in Section 6.2.

5.3 Filtering and Redaction

This section covers the requirements for optionally removing certain values from collections based on the client's clearance or patient consent. This is a compromise decision to apply minor adjustments to the response object instead of rejecting the client's request altogether. It is also less compromising of privacy since it does not reveal the existence of information that is not authorized to be shared with the client.

Note that since some SRA technologies such as GraphQL rely on strict adherence to the data structure defined by the client, the SRA service is not free to remove arbitrary attributes from the response object in a way that compromises the data model defined by the client. However, in places where collections of values are included in the response object, the SRA service may be able to safely remove values from arrays without compromising the structural integrity of the outgoing

data. If this is not possible, the SRA service must reject the client's request to avoid sharing any information that the client is not authorized to access.

Since it gives a lot of flexibility to the client in inquiring about the data at a granular level, redaction and filtering in SRA may particularly increase the risks of revealing sensitive information by inference as discussed in Section 6.1.

5.3.1 Clearance-Based Filtering

The SRA service SHOULD redact values from object collections included in the SRA response based on client's clearances.

This ensures that the SRA compares the clearances of the client with the labels of all the FHIR Resources used in composing the response object, and if there is any mismatch, attempts to adjust the response by removing values from collections to ensure all resources used to create the response bear labels that match the client's clearance. In other words, adjust collections with redaction until the labels of the data object match the clearances of the client.

For example, consider the following case: a client with normal confidentiality clearance sends an SRA request and the response includes an array of immunization dates. There are a total of seven immunization records but two of them are labeled as restricted and therefore not authorized to access by the client. The SRA service excludes these two immunization dates from the array in the response object.

5.3.2 Consent-Based Filtering

The SRA service SHOULD enforce patient consent by redacting values from object collections included in the SRA response.

Similar to the previous requirement, this ensures that values are removed from data object collections included in the SRA response if they are coming from FHIR Resources to which client's access is prohibited based on the patient consent.

5.4 Provenance

This section discussed the requirements for tracking and recording provenance for the data objects resulting from fulfilling an SRA request.

5.4.1 Object-Level Provenance

The SRA service SHALL establish the provenance of the composite object created for the SRA response.

This ensures that the SRA service can keep track of the FHIR resources used in creating the composite object in the SRA response and potentially any provenance records associated with those resources, so that the client can access the provenance of the received data object.

Note that as discussed before, since in common SRA technologies the shape of the response object is defined and determined by the client, including provenance information as an extension in this object, is not usually possible. However, other mechanisms, such as a separate metadata object, or a pointer to the provenance in response headers could be used to communicate the provenance to the client in use-cases where it matters.

5.4.2 Attribute-Level Provenance

The SRA service MAY establish the provenance of the individual attributes in the object created for the SRA response.

This is a more granular version of the previous requirement which ensures that the SRA can trace individual attributes included in the response back to their FHIR resource of origin and the corresponding provenance. While this is not particularly complicated from a technological perspective, it requires a lot of processing and tracking which is only reasonable if this level of provenance tracking is called for by a specific use-case.

6 DISCUSSION

Some further considerations for fulfilling the requirements laid out in this report are discussed in this section.

6.1 Risk of Inference

A major risk in enabling access at the attribute level (especially when coupled with redaction and filtering capabilities) is various types of inference risks resulting from statistical analyses and query comparisons.

For example, consider the case where access to immunization codes is restricted for certain immunizations such as Hepatitis B, but immunization dates are considered normal. Now consider a patient who has seven immunizations, two of which are restricted. In such a case, when a client with normal clearance requests the list of all immunization dates, it will receive a list of seven dates. When the same client requests the list of all immunization codes, if the SRA enables filtering and redaction, it will return a list with five codes –after redacting the two restricted codes from the list. By comparing these results, the client can infer that there are two restricted immunizations for this patient. With more aggressive and fine-grained queries, the client can also narrow down the dates of the restricted immunizations.

Note that if the SRA does not conduct any redactions or filtering, the second query will be declined as unauthorized which will simply indicate to the client that there are *some* restricted immunizations; so, there is still some inference.

While the risk of inference always exists in any information system and these risks are only mitigated by a careful analysis of specific applications and use-cases, enabling more fine-grained access to information always increases the risks for more creative and complex inference attacks. There is ultimately a balance between the increased risk of inference and enabling more flexible and fine-grained access. For example, if instead of FHIR Resource (which are comparingly more fine-grained) the entire patient's immunization record was stored as a single document, the risk of inference would be much smaller, but ultimately a client without the restricted clearance would not be able to see any of the patient's immunizations. Therefore, a lot of non-sensitive information would be unnecessarily blocked from clients with legitimate needs.

Mitigating the risk of inference requires careful policies based on a careful analysis of the system, query mechanisms, and data structures. For example, prohibiting certain types of queries for clients with limited access, limiting queries on certain attributes to aggregates, or prohibiting queries based on a client's previous queries (i.e., maintaining the client's history of queries) can curb some of the ways a client can organize inference attacks.

6.2 Sub-Resource Segmentation and Labeling

Similar to the broad concept of Data Segmentation for Privacy (DS4P), Sub-Resource Segmentation refers to the capabilities that enable a system to distinguish among individual attributes of a FHIR resource (or portions comprised of a group of attributes) from a security and privacy perspective; this is a key concept in enforcing the privacy and security requirements and is at the at the core of a system's ability to make decisions at a finer level of granularity about authorizing access to the information, or making judgements about its privacy metadata (e.g., sensitivity, confidentiality, integrity, compartment, handling instructions, and provenance).

This section discusses some of techniques to go from a resource-level to sub-resource-level segmentation. Some of these methods can be used in conjunction with each other in order to achieve more accurate results.

6.2.1 Top-Down Segmentation

The simplest way to determine labeling or authorization at sub-resource level is to apply at the attribute level the same decision that is applicable at the resource level. For example, a system can rely on the standard Security Labeling Service (SLS) for labeling at the resource level, and then, apply all the security metadata assigned to a FHIR resource to all of its components, including every individual attribute. In other words, the decisions and metadata about any sub-resource portion or attribute is deduced directly from the decisions and metadata applicable to the entire resource.

For example, using this strategy, the confidentiality label assigned to an immunization resource is propagated to all of its components and attributes including the immunization code, or immunization date and time.

Note that this approach is inherently prone to overestimating the labels since the resource-level labels are determined based on the most sensitive data included in the resource which is then propagated to any non-sensitive (or less sensitive) attributes as well. For instance, in the above example, the date of an immunization is clearly not as sensitive as the immunization code, but in this approach, these two attributes are assigned the same confidentiality label.

6.2.2 Bottom-Up Segmentation

The opposite approach is to assign individual labels to each resource attribute and use the high-watermark rule to determine the labels of each portion of a resource in a bottom-up fashion. For example, consider the following attributes in an immunization resource:

```
<vaccineCode>
    <coding>
      <system value="urn:oid:1.2.36.1.2001.1005.17"/>
      <code value="Hepatitis B"/>
    </coding>
 </vaccineCode>
```

In this block, the attribute `vaccineCode.coding.system` represents the code system used for recording the vaccination type (in this case SNOMED™) and the `vaccineCode.coding.code` attribute represents the code for the Hepatitis B vaccine.

The first attribute is clearly not sensitive or confidential since it does not refer to any particular information about the patient. For example, if a client wants to conduct an analysis on the popularity of different code system for recording immunizations in health information systems, it can access this value without compromising the patient's privacy. The second attribute, however, records the actual immunization type which could be sensitive. As shown in **Figure 4**, by applying the high watermark rule, the overall confidentiality label for this portion is R. This process can be applied incrementally to determine the labels for any portion of resource data that appears in an SRA response.

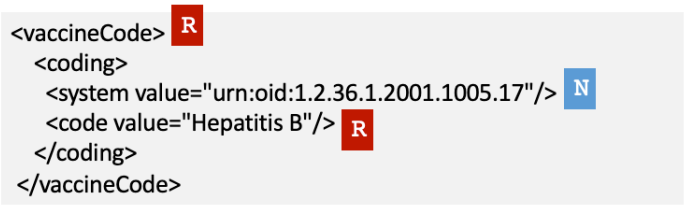

Figure 4: Bottom-up Segmentation based on the High Watermark Rule

6.2.3 Template-Based Segmentation

A simple mechanism to achieve sub-resource segmentation is to rely on pre-defined templates for FHIR Resources which determine the distinct labeling requirements for each attribute in a FHIR Resource. This is very similar to the mechanisms used in de-identification which transforms a resource into a new data object by removing or changing some attributes.

These templates can reside as rules in the Security Labeling System (SLS). For example, a template for the immunization resource can provide a generic labeling as shown in Figure 4 indicating that the `vaccineCode.coding.system` attribute is normal confidentiality even if the `vaccineCode.coding.code` attribute is restricted. Using such templates, the SLS can apply the high-watermark rule to determine the labels for any arbitrary composite SRA data object based on the labels for the underlying attributes (as described in the previous section).

7 REFERENCES

[1] GraphQL Specifications. http://spec.graphql.org/draft/

[2] HL7 FHIR Validated Healthcare Directory Implementation Guide STU 1, Example Bundle for Women's Shelter (fetched on 03/03/2020). http://build.fhir.org/ig/HL7/VhDir/Bundle-womens-shelter.xml.html

[3] HL7 FHIR Path Specifications, ANSI/HL7 NMN R1-2020, Jan. 2020. http://hl7.org/fhirpath/

[4] FHIR GraphQL Draft Specifications. https://www.hl7.org/fhir/graphql.html

[5] SQL on FHIR (fetched on 03/03/2020). https://github.com/FHIR/sql-on-fhir/blob/master/sql-on-fhir.md

[6] OAuth Token Introspection, Request for Comments: 7662, October 2015. https://tools.ietf.org/html/rfc7662

8 ACRONYMS

ACS	Access Control Service
CFR	Code of Federal Regulations
DS4P	Data Segmentation for Privacy
FHIR	Fast Healthcare Interoperability Resources
HL7	Health Level 7
OAuth	Open Authorization
PPS	Privacy-Preserving Services
REST	Representational State Transfer
SLS	Security Labeling Service
SNOMED	Systematized Nomenclature of Medicine
SQL	Structured Query Language
SRA	Sub-Resource Access