Norsk Regnesentral
NORWEGIAN COMPUTING CENTER

Report

Wireless Health and Care Security Architecture

Report no	1006
Authors	Ragni R. Arnesen, Jerker Danielsson, Jørn I. Vestgården and Jon Ølnes
Date	January 2005
ISBN	82-539-0514-9

Norsk Regnesentral

Norsk Regnesentral (Norwegian Computing Center, NR) is a private, independent, non-profit foundation established in 1952. NR carries out contract research and development projects in the areas of information and communication technology and applied statistical modeling. The clients are a broad range of industrial, commercial and public service organizations in the national as well as the international market. Our scientific and technical capabilities are further developed in co-operation with The Research Council of Norway and key customers. The results of our projects may take the form of reports, software, prototypes, and short courses. A proof of the confidence and appreciation our clients have for us is given by the fact that most of our new contracts are signed with previous customers.

Title	**Wireless Health and Care Security Architecture**
Authors	**Ragni R. Arnesen, Jerker Danielsson, Jørn I. Vestgården and Jon Ølnes**
Date	January 2005
Year	2005
ISBN	82-539-0514-9
Publication number	1006

Abstract

This document describes the general security architecture of the Wireless Health and Care project (WsHC). First a system model that encompasses all the demonstrators in the project is described; next, a corresponding security architecture is proposed. Finally, the security requirements on the elements of the system model, listed in the document "WsHC Security Requirements" ([1]), are discussed with respect to the proposed security architecture.

Keywords	Security Architecture
Target group	
Availability	public
Project number	320302
Research field	Computer Security
Number of pages	29
© Copyright	Norsk Regnesentral

Contents

List of figures

1 Definitions

User	Human user, typically medical personnel
Physical component	A physical component contains a number of logical system components.
Logical system component	A logical system component implements system functionality. A logical component can be seen as a module or a system component that may be deployed in one or several physical components.
Logical security component	A logical system component implements security functionality. A logical security component can be part of one or several logical system components and consequently a part of the corresponding physical components. A logical security component may also implement a "stand-alone" service used by logical system components or other logical security components.
Security infrastructure	A security infrastructure consists of a number of shared services used by the system components in the enforcement of security.
Demonstrator	A demonstrator is physically made up of physical components.

In the following, *system component* refers to logical system components and *security component* refers to logical security components.

2 Limitations to the Scope of the Security Architecture

The security architecture should as far as possible be neutral with respect to the security mechanisms used. For example, it should not be restricted to only one particular method for user authentication. However, parts of the security architecture may need to be targeted at particular mechanisms. Use of cryptography will impose requirements on an architecture, and the architecture for a system based on public key cryptography may be different from an architecture based on use of only symmetric cryptography.

Furthermore, the security architecture does not address initialization of the security functionalities, e.g., how authentication credentials and signature keys are issued and distributed to Users.

Security administration is only treated in as far as it has implications for the security architecture. Hence, we address the issues of coupling components to each other (e.g., Source to Patient, or Source to Patient Data Collector), but not administration of the infrastructure (management of users and roles, issuing certificates, etc.).

3 System model

3.1 Overview

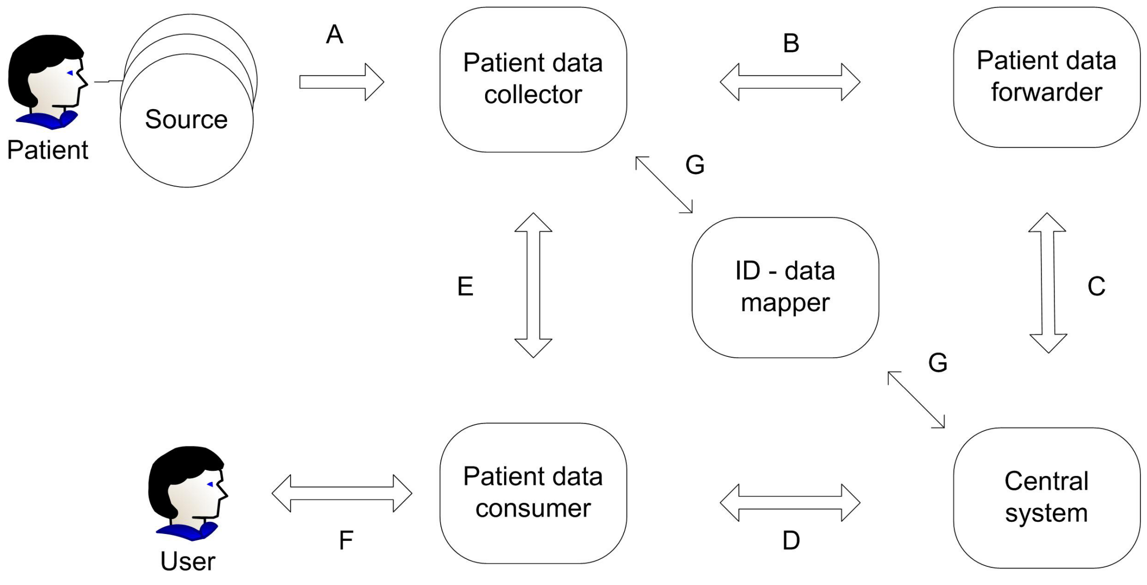

Figure 1. System Model

Figure 1 shows logical components. In a concrete instance (an implementation), their functionality can be distributed over several physical components, or several of them can be collocated on the same physical component. The mapping of the logical components to the physical components of a concrete instance can therefore imply additional security requirements than those that apply to the components and channels depicted in Figure 1.

3.2 Logical System Components

The logical components in the system model are listed in the table below.

Table 1. Logical System Components

Source	Source of patient data, e.g., a sensor, storage unit, or user input through some interface. Patient data is sent to a Patient Data Collector (channel A). Simple data forwarders placed between the actual Source and the Patient Data Collector (e.g., a sensor concentrator) are also treated as Sources in our model. Each Source is connected (physically or logically) to only one patient (at a time). The Source is assumed to have very limited capabilities of protecting the communication, e.g., no proper

	authentication of receiver.
Patient Data Collector	Collects patient data from one or more Sources (channel A) and sends it to a Patient Data Forwarder (channel B) and/or directly to a Patient Data Consumer (channel E). The Patient Data Collector is a *trusted* entity that can handle unencrypted personally identifiable patient data.
Patient Data Forwarder	Receives patient data (channel B) and forwards it to the Central System (channel C). The Patient Data Forwarder is *not* a trusted entity and should not handle unencrypted personally identifiable patient data.
Central System	Receives patient data (channel C) for processing or storage, and may send it to Patient Data Consumers (channel D).
Patient Data Consumer	Receives patient data (channel D or E) and displays it to users (channel F).
ID – data mapper	Determines the identity of the patient to whom patient data pertains and sends the identity to Patient Data Collector or Central System (channel G), depending on where it is implemented.

3.3 Channels

Properties of the channels in Figure 1 are listed below:

- Channel A is a short-range wired/wireless communication link.
- Channel B may be an internal interface or a wired/wireless communication link.
- Channel C is a long-range wired/wireless communication link, possibly over a public network.
- Channel D may be any type of communication link, possibly over a public network.
- Channel E may be an internal interface or a wired/wireless short-range communication link. (Long-range communication between Patient Data Collector and Patient Data Consumer should be provided via the Central System.)
- Channel F is a visual display.
- Channel G is an internal interface in one of the components used to retrieve patient identity.
- Channels A through C transport patient data from Source to Central System. System management data may be sent in the same, and in the opposite, direction.

4 Component Distribution and Security Infrastructure

A number of logical security components have been identified. Some of these logical security components must be distributed over the logical components of the system model. For example, running a security protocol between two components can only be done if protocol support is implemented at both ends.

The other logical security components constitute a central security infrastructure. While it is possible to base security purely on distributed functionality, most recommendations point at the need for infrastructure support for security. The reasons are scalability, easy administration, and security. If security-related information can be managed and stored centrally, both administration and security of the information can be greatly improved. A security infrastructure can provide uniform security services across the system(s) that it supports. In the following, the term *infrastructure* will always refer to the central security infrastructure.

The complete security architecture, consisting of distributed components and central infrastructure, must fulfill the identified security requirements (see [1]).

4.1 Overview of Logical Security Components

The following table describes briefly the security components, listed in order of appearance in chapter 6, where they are described in more detail.

Table 2. Security Components

Name	Description
Entity name database	Used to verify names and for storing relationships between names
Authentication front-end	Front-end towards Users used to relay authentication credentials to an Authentication service
Authentication service	Evaluates authentication credentials and verifies the correct identity of the User in question
Authentication database	Stores authentication credentials (e.g., passwords) used by the Authentication service in the authentication of users or components
Authentication protocols	Used in authentication of system components (e.g., challenge-response protocols)
Credentials and key storage	Local storage of credentials and keys
Credential management service	Controls the life cycle of credentials (creation, distribution, suspension, renewal, revocation)
Key generation service	Generates session keys used to ensure confidentiality and integrity of communication channels
Key agreement protocols	Used when two components agree on a shared secret key

Communication security protocols	Protocols for ensuring confidentiality and/or integrity of data on a communication channel
Cryptographic algorithms	Provides cryptographic algorithms for confidentiality, integrity, key generation, and signature generation and verification
Authorization database	Stores information about roles and their assigned privileges, contextual information and Audit policy
Role database	Links User identities and system component identities to zero or more roles
Relationship database	Links entities to each other (e.g., Patient to Sources, User to Patient Data Consumer)
User Access Decision service	Determines whether a User is allowed access or not (gives "yes" or "no" answer)
User Access Enforcement service	Ensures that the decision made by the User Access Decision service is enforced
Component Access Decision service	Determines whether a component is allowed access or not (gives "yes" or "no" answer).
Component Access Enforcement	Ensures that the decision made by the Component Access Decision service is enforced
Log Collector	Collects metadata about events (e.g., time of event, requesting user, location of event, etc) according to an Audit Policy and sends the log records to the Log Aggregator
Log Aggregator	Receives log records from Log Collectors
Signature Generation service	Generates signatures on messages or documents for traceability and non-repudiation purposes
Signature Verification service	Verifies signatures attached to messages or documents (the signature serves as a proof of origin of the message or document)
Vocabulary	Defines the domain language, i.e., what the meaning of the words used is (e.g., what exactly a "consultation" is)
Security Administration service	Provides a management interface to the Authorization, Role, Relationship, and Entity name databases.
Security Administration front-end	Used by Users to communicate with the Security Administration service

4.2 Distribution of Security Components

The boxes in Figure 2 are explained briefly in Section 4.1 and elaborated on further in chapter 6. In the figure, white boxes are basic security functionality that must be implemented in all components, including the infrastructure described in the following section. How this functionality is implemented in practice may vary between components and for different settings.

From Figure 2 it can be noted that all logical components except the Source have several logical security components in common. These logical security components are necessary to communicate securely (authentication, and communication confidentiality and integrity), for authenticating Users, and for controlling User and component access.

Additionally, the Patient Data Collector has a *Signature Generation* service and the Central System has a S*ignature Verification* service. This is motivated by the need for traceability of Patient data in the system. The Patient Data Collector signs collected Patient data so that its origin can be proved.

The Patient Data Collector also has *Security Administration* front-end, which is used to establish a relationship between Source and Patient Data Collector (see Section 6.7.1).

Finally, the Central System and Patient Data Collectors have *Log Collectors*, which collect metadata about events (e.g., time of event, requesting user, location of event, etc) according to an Audit Policy and sends the log records to the *Log Aggregator*. The Patient Data Consumer does not need any *LogCcollector* since all access to Patient data goes through the Central System or a Patient Data Collector.

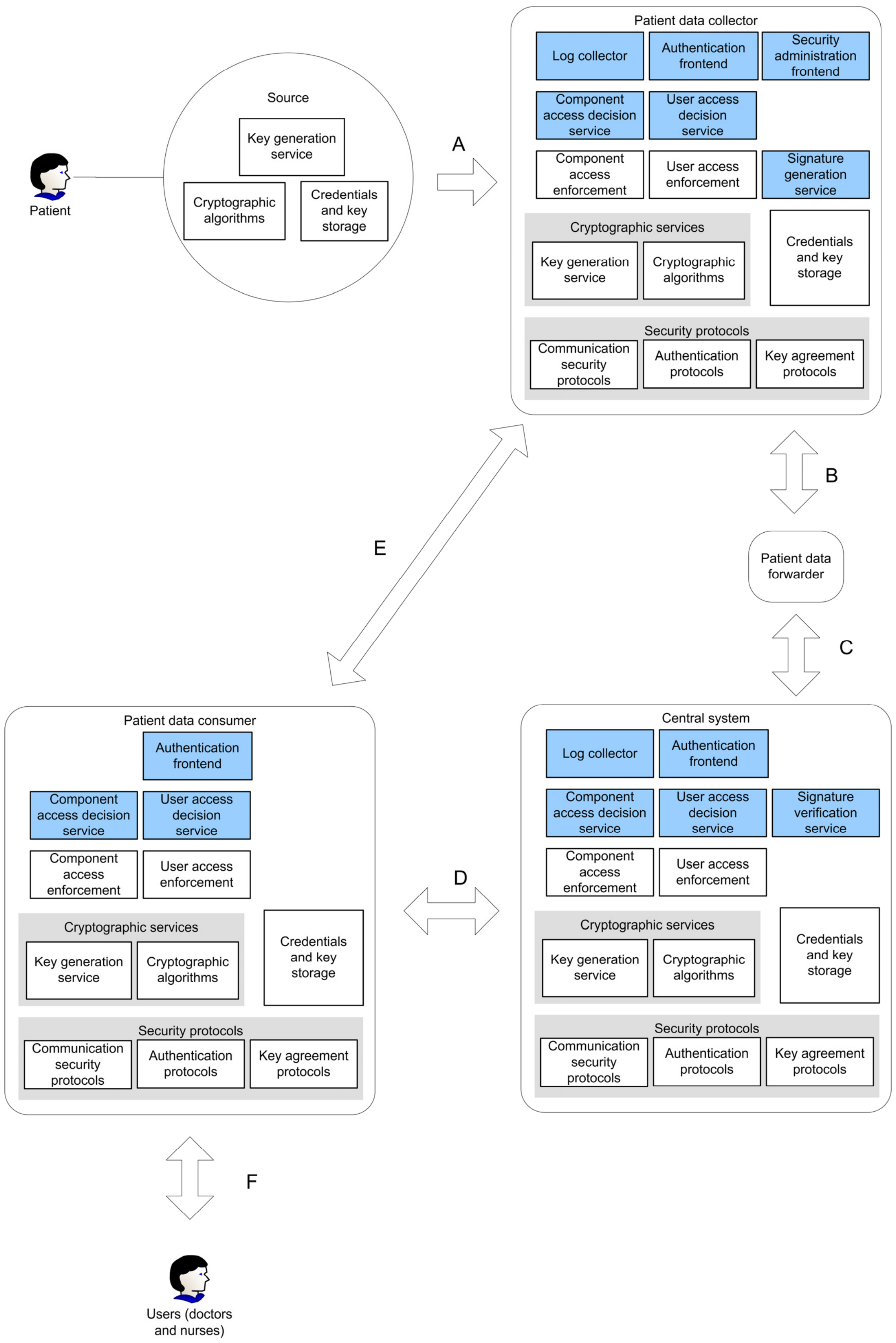

Figure 2. Distribution of Security Components

4.3 Central Security Infrastructure

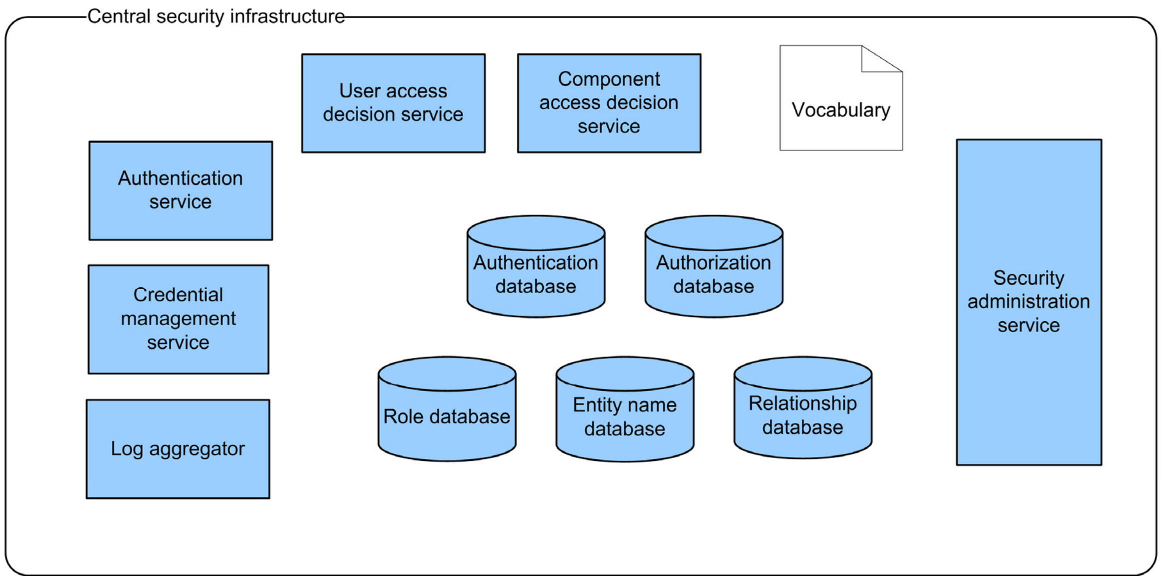
Central security infrastructure
User access decision service
Component access decision service
Vocabulary
Authentication service
Credential management service
Log aggregator
Authentication database
Authorization database
Role database
Entity name database
Relationship database
Security administration service

Figure 3. Central Security Infrastructure

The logical security components of the infrastructure may be distributed over several physical components. However, in the following, the central security infrastructure is seen as one physical domain to facilitate the discussion. Communication between the components, and security of this communication, is thus out of the scope.

The distribution of logical security components shown in Figure 2 and Figure 3 is one possible distribution, others may also be feasible. The application of the security architecture to the demonstrators developed in the project may reveal problems with the selected distribution, leading to its modification.

The rest of the document will discuss the rational behind the chosen mapping.

4.4 Relationship Between Logical System Components

Figure 4 shows how the logical security components relate to each other, i.e., which components that use the services offered by other components and which components communicate with each other.

It is not specified which specific components inside the boundary of Figure 4 are used by the components outside the boundary. Furthermore, the relationship between the centralized databases (the Authentication database, the Authorization database, the Role database, etc.) and the other components are not specified.

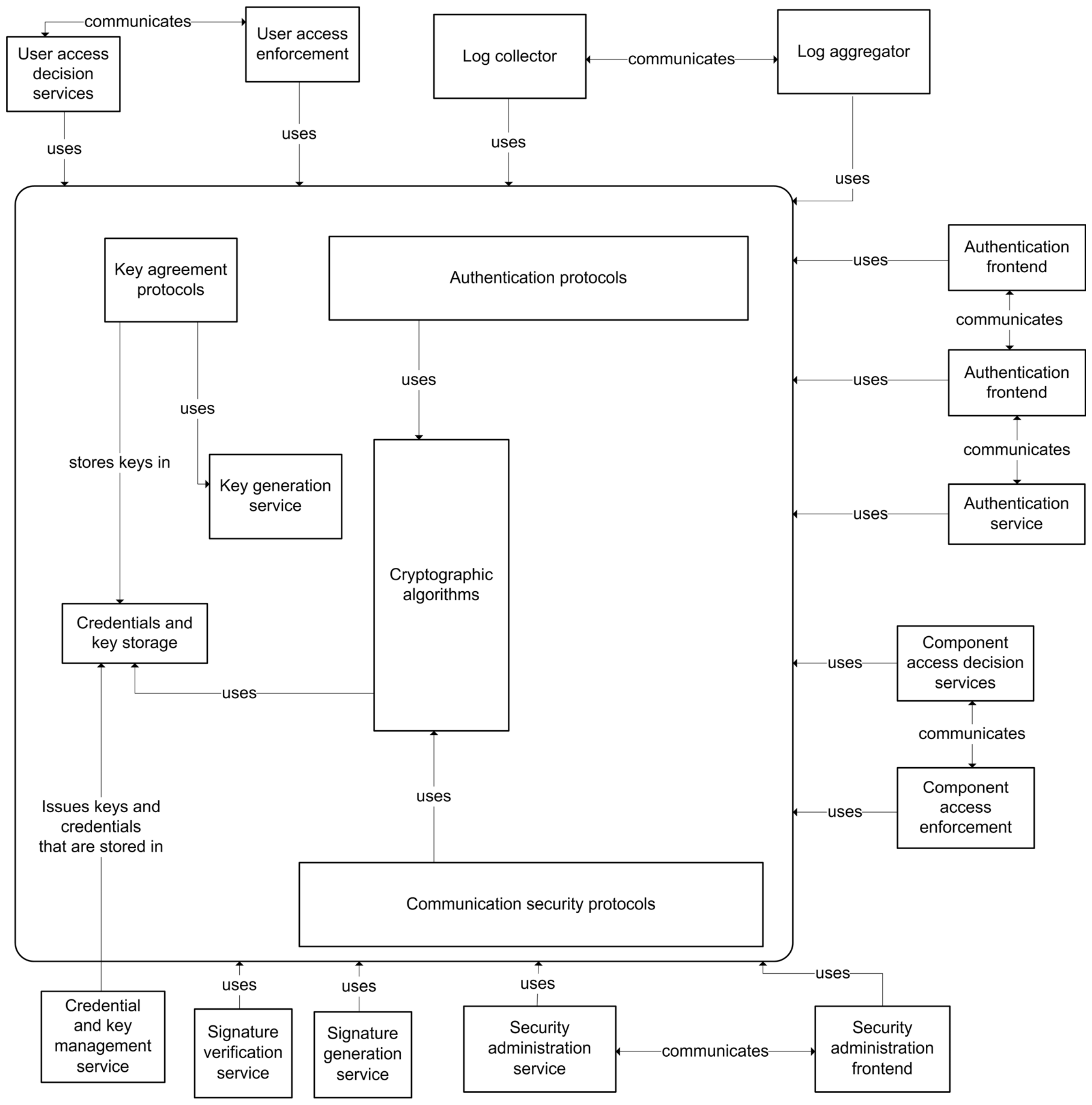

Figure 4. 4 Relationship Between Logical Security Components

5 Credentials and Keys Needed in the System

Credentials and keys are needed by system components for several purposes:

- Authentication credentials and keys are needed for authentication of system components and Users, and for exchange of session keys (key management).
- Credentials and keys are also needed for signature generation and signature verification.
- Session keys must be established to secure (i.e., protect confidentiality and integrity of) communication between system components.

Since session keys are used more (i.e., to encrypt larger amount of data per time unit), they are only used for a short time period to protect against cryptanalysis and only within one session to protect against replay attacks. However, long-term keys used for authentication, signatures, and key management may also have to be revoked and renewed, e.g., if one or several keys are compromised.

Different keys should be used for authentication, signature generation, and key management. There is a risk that someone claims that data exchanged during authentication is signed. For example, a nonce that is not a random bit string, but instead codifies a document that someone wants signed, may be generated and signed during authentication. One may want a backup copy of key management keys to ensure that data is not accidentally lost because the key is lost and decryption is impossible. However, commercial products seldom support the use of three different keys for these purposes. The most common scenarios are either one key for all three purposes, or two keys with a separate key dedicated to the key management functionality.

The need for establishment of session keys between communicating parties are discussed in Section 6.4.

5.1 Component Authentication

Instances of logical system components do not have to authenticate other instances of logical system components directly. The same is true for logical security components. It is more efficient if physical components authenticate each other and that the logical system components and the logical security components use the authenticated relationship established by their host physical component (see section 1) and the mechanism for secure communication that they provide.

Physical components may provide two types of communication mechanisms: (i) an authenticated and secured channel (i.e., by establishing a symmetric session key) or (ii) signed and encrypt messages (i.e., messages signed by the sending component's private key and encrypted with the receiver's public key).

Patient Data Collectors, Patient Data Consumers, the Central System, and the central security infrastructure need to be able to mutually authenticate each other.

The rationale for this is the following:

- Patient Data Collectors and the Central System need to be able to mutually authenticate each other to ensure authenticity of patient data towards the Central System and that patient data is not transmitted to other systems than the appropriate Central System.
- Patient Data Consumers need to authenticate the Central System and Patient Data Collectors they receive patient data from to ensure authenticity of the data.
- The authentication of the Patient Data Consumers by the Central System and Patient Data Collectors is not always necessary, since the Patient Data Consumer only acts on behalf of an authenticated User and receives whatever data the User requests and is allowed to receive. But authentication of the Patient Data Consumer is supported for symmetry and simplicity, since the Patient Data Consumer in any case must be able to authenticate itself to the infrastructure (see below). Also, if restrictions are imposed with respect to the equipment used (e.g., of a particular type or within a particular network), the Patient Data Consumer (equipment) may need to be authenticated.
- Patient Data Collectors, Patient Data Consumers, and the Central System need to be able to authenticate the infrastructure, to ensure the authenticity of security data (e.g., authorization policy).

- The infrastructure needs to authenticate Patient Data Collectors, Patient Data Consumers, and the Central System, to ensure that possibly sensitive information is not leaked (e.g., authorization policy) and to control that only authorized modifications to configuration is allowed (e.g., security administration).

In summary, all communication in the system must use authenticated communication channels. Authentication will usually be mutual, but one-way authentication may be sufficient in some cases. The communication channels also need to be integrity and confidentiality protected (unless a channel is entirely within a physically protected area).

The recommended setup for such communication is to use a cryptographic authentication mechanism (PKI-based, or perhaps Kerberos or similar schemes). During such an authentication process, session keys for integrity and confidentiality may also be established. We recommend use of PKI-based mechanisms, as argued in 5.2.

5.2 Channels and Components Relationships

Tthe relationships between system components are depicted in Figure 5.

Channel A (see Figure 1) forms a many-to-one relationship. Several Sources can be connected to the same Patient Data Collector (see Figure 5).

Channels B and C form a many-to-one relationship. Several Patient Data Collectors can be connected to the same Patient Data Forwarder, and many Patient Data Forwarders can be connected to the same Central System. It is presumed that each Patient Data Collector only communicates with one Central System. If necessary, several Central Systems can exchange data with each other but this exchange is not part of the system analyzed here.

Channel E forms a many-to-many relationship between Patient Data Collectors and Patient Data Consumers.

Channel F forms either a many-to-many relationship between Patient Data Consumers and Users, or alternatively (in the case of personal/private equipment) a one-to-one relationship.

Channel D forms a one-to-many relationship between one Central System and several Patient Data Consumers.

The one-to-many and many-to-many relationships rule out authentication methods based on separate, secret keys per pair of communicating parties, as these will not be manageable. Further, solutions based on symmetric group keys shared between many components will not be sufficiently secure.

Centralized key management systems based solely on symmetric cryptography (like Kerberos) must rely on an on-line service accessible at all times to all components. In the system model shown in Figure 1, this may not be appropriate for channels that may be implemented by asynchronous communication, e.g., channels B and C. Also, the scheme may not be appropriate for communication between components in different security domains, which may be the case for Patient Data Collectors and possibly also Patient Data Consumers.

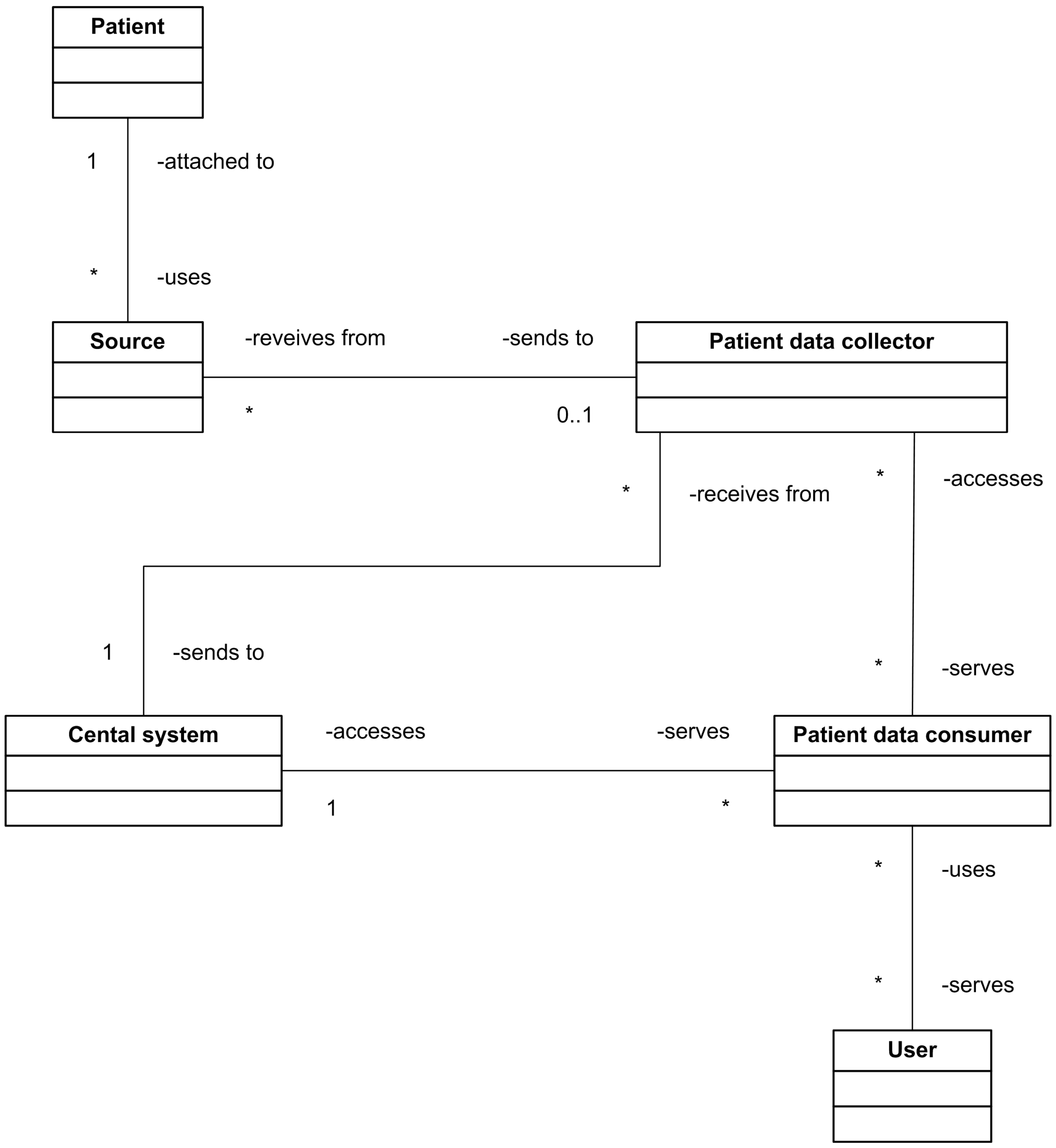

Figure 5. Relationships

Therefore, we require that the Patient Data Collector, Patient Data Consumer, Central System, and infrastructure are equipped with certificates and public/private key pairs. In addition, each component must have access to the necessary Certificate Authority's (CA) public keys and certificates either by local storage or through a certificate validation service in the infrastructure.

User authentication (channel F) may also be by PKI-based mechanisms, but the security architecture is open in this respect. Username/password or other mechanisms may be supported. The choice depends on the desired security level, the mechanisms supported by the components and the infrastructure, and – with respect to selection of PKI-based mechanisms – on the need to sign information.

5.3 Signature Generation and Verification

Patient Data Collectors need to be able to sign data for traceability and non-repudiation purposes, and the Central System must be able to verify the signatures. Signature verification (possibly also signature generation) may be a service provided by the infrastructure, or may be performed locally.

Users (through channel F or by means of the Patient Data Consumer) may need to sign data. This applies in particular when data is altered or inserted by the user, but may also be pertinent to acknowledge receipt of data. The Central System must be able to verify the signatures.

5.4 Summary of Credentials and Keys Needed

The summary of credentials and keys needed by the various logical components are listed in Table 3.

Note that, if several logical components are collocated in one physical component they may possibly share credentials and keys. That is, the physical component has authentication, key management, and signature key pairs and certificates that are used by all logical security components of the physical component.

Table 3. Credentials and Keys Needed by Logical Components

Logical system component	**Need to be issued with**	**Need to know and trust**
Source	-	-
Patient Data Collector	Public/private authentication key pair + certificate Public/private signature key pair + certificate Possibly also public/private key management key pair + certificate	Certificate of CA or validation service
Patient Data Forwarder	-	-
Central System	Public/private authentication key pair + certificate Public/private key management key pair + certificate Possibly also public/private signature key pair + certificate	Certificate of CA or validation service
Central security infrastructure	Public/private authentication key pair + certificate Public/private key management key pair + certificate Possibly also public/private signature key pair + certificate	Certificate of CA or validation service

Patient Data Consumer	Public/private authentication key pair + certificate Possibly also public/private key management key pair + certificate Possibly also public/private signature key pair + certificate	Certificate of CA or validation service
User	Public/private authentication key pair + certificate (or other credentials like username and password) Possibly also public/private signature key pair + certificate	Patient Data Consumer wrt. authentication of system components

6 Security Functionality

6.1 Naming

The *Entity name database* is used to verify names and for storing relationships between names. Users and (logical) physical components are entities. Entities have names that identify them, and one entity can have several names. Given one valid name of an entity it should be possible to access the other valid names of the same entity. Names are used for two different purposes: identification (of data source or communicating party), or data routing. Not all types of names are suitable for both purposes.

The Entity name database and associated services must be trusted to the same degree as the authentication mechanism used. If for example the certificate name (DN) of a user is authenticated, and then later translated into another name for the user, the translation needs to preserve the strength of the (PKI-based) authentication mechanism since no explicit authentication procedure is carried out for the new name. Consequently, communication towards the Entity name database and associated services needs to be sufficiently secure.

Each entity should have a “main name”. This “main name” should be unique in the domain where the system is deployed and not used for other purposes than to identify the entity within the system. For example, the main purpose of an IP-address is to facilitate routing of IP-packets to the destination and IP-address is therefore not suitable as a “main name” since conflicts may arise between its different uses

6.2 Authentication

The discussion in this section is related to the security requirements Sec18-Sec21 of [1]. There is a need for authentication of both users and physical components, and it is desirable to support different types of user authentication mechanisms.

6.2.1 User Authentication

Users accessing patient data through a Patient Data Consumer, and Users creating patient data on the Patient Data Collector user interface, must be authenticated.

The *Authentication front-end* interacts with Users when they authenticate themselves, or act as a mediator between another *Authentication front-end* and an A*uthentication* *service*. It relays the authentication credentials (e.g., username and password, or a signed challenge/response for a PKI-based mechanism) supplied by the User to an *Authentication service,* which evaluates the authentication credentials.

The credentials of Users must be stored in an *Authentication database* when authentication is based on a shared secret (like a password). In the case of PKI-based authentication, user certificates must be available either through a certificate database or supplied by the User. CA certificates must also be available.

Selection of User authentication mechanism is not restricted by the architecture. The system architecture shall be able to integrate any method used by the organization deploying the system. Authentication of Users is carried out by the *Authentication service*, which can be implemented in different ways. The only thing the rest of the system needs to do is to pass user credentials (like username and password) to the Authentication service and validate the response from the service.

6.2.2 Component Authentication

As described in Section 5.1, it is usually sufficient that physical components authenticate each other. Physical components authenticate each other through *Authentication protocols*, either as a set of request/response messages for a synchronous communication protocol, or by construction of a secure message in a defined format. Credentials and related (long-term) keys needed are stored (or at least accessible) locally to each component in the C*redentials and key storage*.

Section 5.4 summarizes the certificates and key pairs that may need to be issued to system components.

The architecture does not specify how the certificates and key-pairs listed in Section 5.4 are distributed. However, one feasible solution is that the CA's certificate(s) is pre-installed in every component's *Credential and key storage* and that the certificates of the communicating parties are exchanged through the *Authentication protocol*.

6.3 Credential Management

A *Credential management service* controls the life cycle of credentials. This includes functions such as creation, distribution, suspension, renewal, and revocation. Credential management is considered out of the scope of the security architecture.

In the case of PKI-based credentials, the service is a CA, which issues certificates to entities according to a well-defined policy. The policy includes requirements for key management of private and public keys, e.g., whether key pairs are generated by an entity that requests credentials or by the CA itself.

Several Credential management services may be used by the same system. In the case of PKI-based credentials, different entities may obtain certificates from different CAs. However, an Authentication service must be able to handle all these CAs and the certificates issued by them.

6.4 Key Management

Session keys are established based on long-term keys stored in the C*redentials and key storage* and/or public keys obtained from the certificates of authenticated counterparts.

Session keys are used for integrity protection and encryption of communication channels, or for encryption of secured messages (which are usually signed by means of public key cryptography rather than being integrity protected by a session key).

Session keys are usually generated by a local *Key generation service* or established through a *Key agreement protocol*, but this service may also be provided by a component in the infrastructure. Session keys are held by the local *Cryptographic service provider*, which also runs the cryptographic algorithms applied. The *Key generation service* also generates any nonces[1] required.

6.5 Authorization

Scalable authorization schemes must be role-based, not based on authorizations assigned to individual entities. An authorization process consists of three (four) steps:

- Definition of roles, typically guided by descriptions of the processes that are to be carried out in (or by means of) the system.
- Assignment of access rights (privileges) to roles.
- Coupling between entities (like Users) and roles.
- Possibly also definition of contextual information that may be applied in an access decision, like time of day, location etc.

The *Authorization database* stores and provides access to information about roles and assigned privileges, and typically constrained by contextual information. For example, nurses may be allowed access to some patient data while they are at work at the hospital but not from their homes.

The *Role database* links User identities (possibly also System component identities) to zero or more roles. Note that the Role database thus is strongly related to the Entity name database (see 6.1), although the two should be (at least logically) separate. The Role database and Authorization database will usually be parts of one common database.

The *Relationship database* links entities to each other, where an entity may be a User, a System component, or a Role (this may be made completely role-based by defining “dummy” roles for all entities.). The list below gives a few examples:

- Patient to Patient Data Collector(s),
- Patient to Source(s),
- Patient Data Collector to Source(s),
- User to Patient Data Consumer(s),
- Patient to Role(s)/User(s) (e.g., treating doctor),
- Patient Data Consumer to User(s).

All information in these databases needs to be established and maintained through administrative interfaces. This is considered out of the scope of this architecture.

[1] Nonce = “number used once.” The generation of such numbers is a common part of many security protocols.

6.6 Access Control

The discussion in this section is related to the security requirements Sec9, Sec15 and Sec23 of [1]. Access control mechanisms are used to enforce the authorization policy of the organization. All physical components on which patient data may be accessed (i.e., Patient Data Collector, Central System, and Patient Data Consumer) must ensure that no unauthorized actors (human Users or system components) gain access to patient data.

All access control presupposes that the entity requesting access has been authenticated. That is, we already know who the actor is, but we do not yet know whether or not this actor is allowed the requested access.

Access control is divided in two functions:

- An *Access Decision* service, which determines whether the entity is allowed access or not (gives "yes" or "no" answer). The access decision is taken based on the information in the Authorization, Role and Relationship databases.
- An *Access Enforcement* service, which ensures that the decision made by the Access Decision service is enforced.

The Access Enforcement service must be distributed (placed close to the resource it protects), whereas the Access Decision service may be either central or distributed (see below).

6.6.1 Control of User Access

The *User Access Decision* service evaluates access requests from users. The service may be local, acting on locally stored information, or it may be (partially) provided by the infrastructure, acting on information in the Authorization database. This is at least partly dependent on the granularity of the access decision. Access to (whole of) systems or services will typically be managed and decided centrally, while access to individual data elements or functions will typically be decided locally to a database or service (but perhaps based on centrally managed role information).

The *User Access Enforcement* component enforces access based on the decision of the *User Access Decision* service. Access enforcement must be done "close to" the resource to which access is controlled. It must not be possible to bypass the access enforcement component that protects the resource.

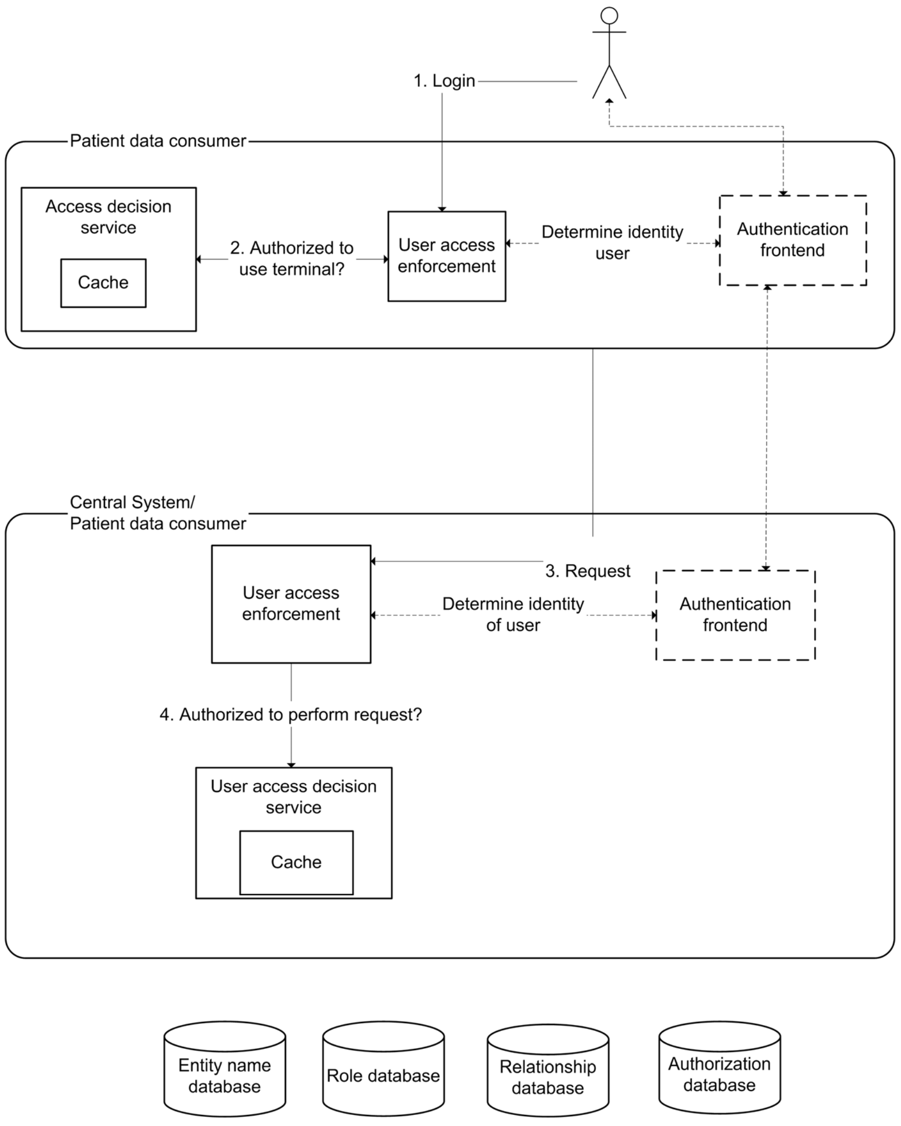

Figure 6. User Access Control

6.6.2 Control of Component Access

The *Component Access Decision* service evaluates access requests of components. The service may be local, acting on locally stored information, or it may be (partially) provided by the infrastructure, acting on information in the Authorization database.

The *Component Access Enforcement* component enforces decisions made, on access requests from components, by the *Component Access Decision* service. Access enforcement must be done “close to” the resource to which access is controlled. It must not be possible to bypass the Access Enforcement component that protects the resource.

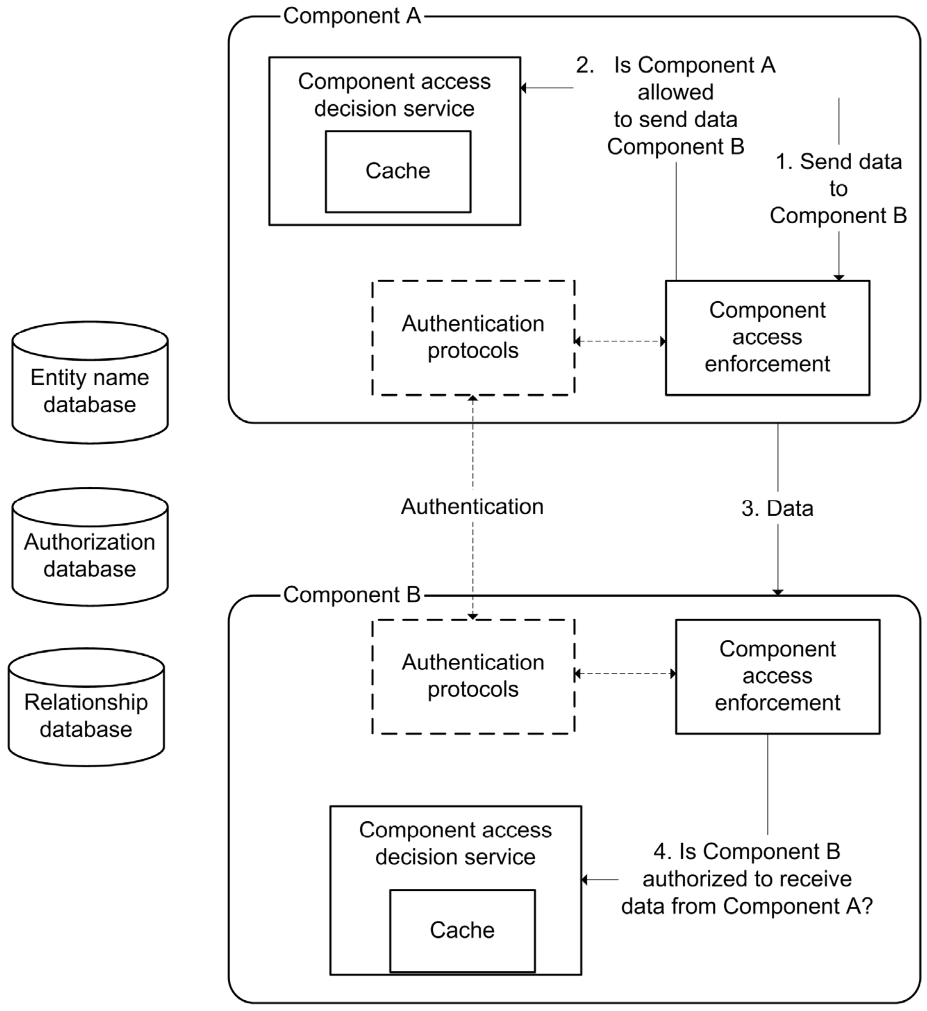
Component A
Component access decision service
Cache
2. Is Component A allowed to send data Component B
1. Send data to Component B
Authentication protocols
Component access enforcement
Entity name database
Authorization database
Relationship database
Authentication
3. Data
Component B
Authentication protocols
Component access enforcement
Component access decision service
Cache
4. Is Component B authorized to receive data from Component A?

Figure 7. Component Access Control

6.7 Communications Confidentiality and Integrity

The discussion in this section is related to the security requirements Sec11-Sec13 and Sec22 of [1]. Channels A, B, C, D and E (see Figure 1) should be confidentiality and integrity protected. Note that Channels B and C must be treated as a single channel with respect to security since the secure channel is established between the Patient Data Collector and the Central System, and since the Patient Data Forwarder is not a trusted component. Channel A is treated separately in Section 6.7.1.

The end points of a channel need the following logical security components:

- Communication security protocols
- Cryptographic service provider
- Key generation service
- *Key agreement protocols* (depending on implementation)
- *Credentials and key storage* – software, hardware components in computers, or smartcards and similar devices

Session key establishment protocols should be connected to the authentication (see Section 6.2) of communicating parties in such a way that each communicating party knows who the other party is, and can be certain that no man-in-the-middle or other third party can intercept the communication.

The data transmitted on the channel should be encrypted for confidentiality protection, and a message authentication code (MAC) attached for integrity protection. If data is digitally signed, the signature provides (additional) integrity protection. The recipient of data should verify the integrity of data. If an error is detected, the recipient has the following choices (one or more of the below):

- Drop data (e.g., in real-time transmission),
- Request re-transmission,
- Log error, and/or
- Send alert (e.g., to User or Central System).

Which actions are appropriate must be determined in each case.

6.7.1 Securing the Communication Between Source and Patient Data Collector

This section discusses the security requirements Sec2-Sec7 of [1]. The communication between a Source and a Patient Data Collector (channel A in Figure 1) must be treated somewhat differently from the other communication channels in the model, since both the possibility and the need for implementing strong security on this channel is limited. This is mainly due to the fact that we cannot assume that a Source is capable of authenticating itself to the Patient Data Collector, and, on the positive side, that the communication is short range (see Section 6.11).

The Source may be:

- A sensor transmitting patient data through some form of short-range wired or wireless connection
- A simple storage device communicating through some form of wired connection, e.g., an iButton communicating through an iButton reader.
- An interface where Users can enter patient data directly.

In the last two cases, we assume there is no need for communication security because the communication is sufficiently secured physically. This also applies to the first case, if the communication between Source and Patient Data Collector is *wired*. That leaves the case where the Source and Patient Data Collector are separate devices communicating via a short-range wireless connection. For simplicity, we will assume this is a Bluetooth connection.

Confidentiality protection of the Bluetooth channel may be provided by Bluetooth link-layer encryption functionality. This encryption functionality has some weaknesses, but with short-range communication, it is deemed sufficient. Establishment of a shared, secret key between the Source and the Patient Data Collector should also be sufficient to ensure authentication with respect to the traffic between them.

Integrity protection is not provided by the Bluetooth link-layer encryption, and must thus be provided at the application layer, if deemed necessary. Errors caused by noise are probably handled well enough by the built-in error handling in Bluetooth, but errors inserted on purpose by a malicious adversary is more difficult to protect against unless cryptographic mechanisms for integrity protection is implemented. Whether the latter is a relevant risk must be determined in each demonstrator.

The main security issue is ensuring that Sources only talk to the correct Patient Data Collector, and, correspondingly, that a Patient Data Collector knows which Sources it should listen to. To ensure this, we cannot allow automatic roaming of Sources. That is, a Source and a Patient Data Collector cannot initiate a link between them without some manual coupling procedure.

The manual coupling procedure can for example be to place the Source and Patient Data Collector close to each other and initiating the coupling through a *Security Administration* interface on the Patient Data Collector. This could be made as easy as one click of a button. In the initialization phase the Patient Data Collector will receive and store the identity of the Source, and a shared secret key (a Bluetooth link key) is established and stored on both the Source and the Patient Data Collector.

After the initial coupling, the Source should be able to initiate a connection to the Patient Data Collector whenever the Source has data to transmit (without any manual procedure). In each such connection, the Source and Patient Data Collector authenticates each other by proving possession of the shared secret key established in the initialization phase, and then they establish an encryption session key for securing the communication.

The Central System should know where patient data it receives was created, but the Central System cannot authenticate the Source. The solution is to authenticate the Source "by proxy," i.e., the Patient Data Collector identifies the Source and guarantees its authenticity towards the Central System.

6.8 Storage Confidentiality and Integrity

functions. Additionally, cryptographic protection may be used.

The *Cryptographic service provider* can be used to provide confidentiality and integrity protection of stored data. However, we assume storage is limited to a minimum, as explained in Section 6.11, except for the Central System.

If stored data are to be encrypted, the *Credentials and key storage* must store the long-term encryption keys (e.g., private keys for key management). Session keys may be stored encrypted under the long-term key.

If stored data are to be signed, public keys (certificates) for verification must be available.

6.9 Traceability and Non-repudiation

The discussion in this section is related to the security requirements Sec14, Sec16, Sec17 (Traceability), Sec 24 and Sec25 (Emergency access) of [1]. The proposed solutions also support non-repudiation (strong proofs of "transactions" carried out).

Audit is an important second-line security measure to detect and deter security breaches that are not stopped by the first-line security measures (authentication, access control, etc.) Traceability and accountability of events must be ensured to facilitate effective audit.

Additionally, a requirement imposed on systems that store and process medical data is that events shall be traceable to the individual User (or System component for automatic processes) responsible for the event. Note that this requirement on the traceability must be met even if access control is role-based.

The *Log Collector* collects metadata about events (e.g., time of event, requesting user, location of event, etc) according to an *Audit policy* and sends the log records to the *Log Aggregator*. The Audit policy is stored in the *Authorization database*.

The origin of patient data received by the Central System for processing or storage must be known. The *Signature Generation* service is used to attach a signature (of a User or a Patient Data Collector) to the data, and the corresponding *Signature Verification* service is used by the Central System to verify the validity of the signature.

6.10 Security Administration

The Security Administration service provides a management interface to the Authorization, Role, Relationship, and Entity name databases.

Users communicate with the Security Administration service through a Security Administration front-end.

Security management must be performed subject to a security level that is at least as strong as that required by the most security critical functionality in the system and security data must be protected to at least the same level as the most sensitive (medical) data in the system. Else, the security management system may become a back door to compromise of the (in principle) stronger security of the medical data.

A vocabulary is used to define the domain language. That is, the *Vocabulary* lists the definitions of terms used in the domain, such as metadata used to describe the contents of, e.g., patient journals, or definitions of roles and how they are arranged in hierarchies.

6.11 Other Security Relevant Properties

The discussion in this section is related to the security requirements Sec1, Sec2, Sec8 and Sec10 of [1]. There are some requirements that are important for security, but that cannot be fulfilled through the security architecture.

- The communication between Sources and Patient Data Collectors should be short-range. If it is long-range, the security provided by the security architecture described in this document will not be sufficient.
- The security architecture is not designed with distributed data storage in mind. Hence, storage of patient data in other components than the Central System should be limited as much as possible. Ideally, the Source and Patient Data Collector should only store data as long as necessary to ensure successful transmission, and the Patient Data Consumer should never store data when not in use.
- To ensure integrity of patient data, the Patient Data Collector should not be allowed to make changes in data, except for defined transformations and aggregations. The recipients of patient data must know which transformations have been made.

Some, normally necessary, security functions are considered out of the scope of this security architecture, although it is clear that the functions may be supported within the architecture:

- Security administration is not fully addressed.

- Alarms are usually used to alert personnel about possible security breaches. Alarms may be raised by the *Access Decision* and the *Access Enforcement* services, the *Log Collector* and/or by external systems like firewalls and IDSs.
- Security recovery functions are needed to restore a compromised system to a secure state.

7 References

[1] A rnesen, R. R., Danielsson, J., Vestgården, J. I. and Ølnes, J. *Wireless Health and Care Security Requirements*. Technical note **DART/01/04**, Norsk Regnesentral. December 2004.