SECURE Framework Architecture (Beta)

Jean-Marc Seigneur, Vinny Cahill, Christian D. Jensen,
Elizabeth Gray, Yong Chen

Distributed Systems Group, Trinity College Dublin, Ireland
Jean-Marc.Seigneur@trustcomp.org, secure-tcd@cs.tcd.ie

Abstract.

The SECURE project is investigating the applicability of autonomous trust/risk-based security in the context of global computing, characterized by heterogeneity, uncertainty and a large number of previously unknown roaming entities. In this document, we present the SECURE framework that can be instantiated under different operating assumptions to implement trust/risk management for different application scenarios in a way that ensures compliance with the SECURE formal model of trust. We start by identifying required, recommended and optional functionalities of such a framework. We then describe the architecture of the framework, which is composed of a set of components where each component is responsible for a key aspect of the framework, and the interactions between these components using UML sequence diagrams.

Contents

1 Introduction

One of the goals of the SECURE project is to study the feasibility of autonomous trust/risk-based security in the context of global computing, where (software, hardware and communication) heterogeneity, uncertainty and a large number of previously unknown roaming entities are found. In this document, we present the SECURE framework that can be instantiated under different operating assumptions to implement trust/risk management for different application scenarios in a way that ensures compliance with the formal model developed in SECURE (please refer to D1.1[1] [6] and D1.2 [7]).
We discuss what functionalities are required, recommended and optional in such a trust/risk-based security framework based on iterative feedback from early prototyping of an application programming interface (API) and application scenarios (see D5.1 [11], D5.2 and its demonstration). The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in IETF RFC 2119 [1]. We explain how the framework specifications can be applied in alternative implementations.
In this document, we assume that components of the SECURE framework notify other components of the SECURE framework when needed. However, the system of notification MAY be replaced by a system of periodical triggering: in this case, the components notified in the former approach MUST periodically trigger themselves in the latter approach to carry out similar operations.
We first describe the high-level view of the framework. Then, the framework is decomposed in a set of components. Each component is responsible for a key aspect of the framework. These components collaborate, as different Principals[2] (i.e., entities) interact. In Section 4, the main types of Requests exchanged during these interactions are detailed. Each of these Requests triggers different UML [9] sequence diagrams presented in Section 5. Finally, we draw conclusions and present future work.

2 Framework Main Components and Activities

In the SECURE framework, there are two high-level processes (as depicted in Figure 1), which are run in parallel: the decision-making process and evidence processing. Activities related to these processes occur because a Principal can be: a Requester (of an Action), a Decision-Maker, an Observer (of first-hand pieces of Evidence, e.g., on its own interactions or captured from its own sensors), a Receiver (of indirect pieces of Evidence), the Subject (of a piece of Evidence), a Recommender (of a piece of Evidence), a Referer and a Referee (when the level of trust of the Referee in a third Principal is requested by the Referer with a trust Reference Request).

[1] Throughout the document, Dx.y means the SECURE deliverable Dx.y.
[2] Some words in this document starting with an upper-case letter are defined in the Glossary document of the SECURE project.

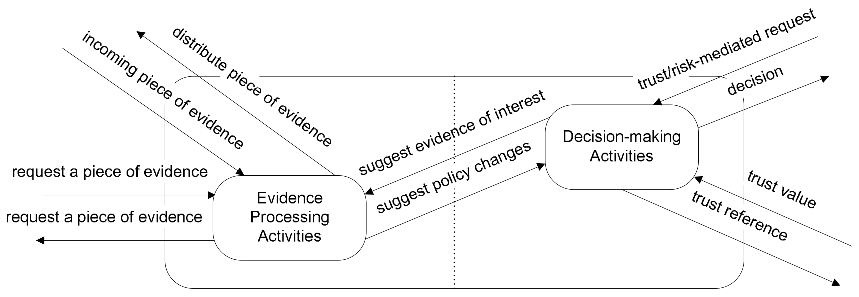

incoming piece of evidence
distribute piece of evidence
request a piece of evidence
request a piece of evidence
Evidence Processing Activities
suggest evidence of interest
suggest policy changes
Decision-making Activities
trust/risk-mediated request
decision
trust value
trust reference

Figure 1. SECURE Framework High-level Activities

The Requester (i.e., the requesting Principal) of an Action and the Decision-Maker (i.e., the Principal to whom the Request is made) MAY collaborate during the decision-making. The Requester MAY specify Authorisation Hints to facilitate the decision-making process. The Decision-Maker MAY return a Decision. In the case of negative decisions, the Decision MAY suggest Authorisation Hints to be specified in subsequent requests. The different Request/Decision exchanges form the Trust Negotiation. The decision-making process might be faster[3] but care SHOULD be taken when taking into account Authorization Hints[4] given by the Requester because it creates opportunities for attacks on the decision-making process (e.g., if Authorisation Hints consist of a list of Observers and a restricted number of Observers are queried during trust formation [12], the Requester can force the Decision-Maker to query only colluding Observers).
The decision-making process SHOULD be able to suggest what kind of Evidence is of interest to improve scalability, performance and context adaptation (e.g., Authorisation Hints specify evidence of interest to the Evidence Gatherer). However, it depends on the implementation of the evidence process, e.g., which may be a distributed hash table of recommendations and observations or a local database of observations only. In return, the evidence process SHOULD be able to suggest changes in the policies driving the decision-making process. If the framework persistently provides poor security protection (e.g., external users manage to use 90% of the resources even though it should not happen), it makes sense to revise the policies (e.g., change the access control policy to try to modify the trend).
Context plays an important role in both processes. Context SHOULD affect the way in which Principals behave. There are different granularities of Context: Environmental Context concerns the context in which the Decision-Maker seems to be (e.g., under-attack) and is built from various pieces of Evidence; an Action request inherently carries further Context.
In the literature, there is no real consensus regarding the digital representation of trust, e.g., the format of Trust Values of an entity, and pieces of Evidence exchanged between interacting parties. At the beginning of research on computational trust, a

[3] Further optimisation MAY consist of attaching the Recommendations themselves in Authorisation Hints.
[4] The SECURE deliverable D1.2 describes particular hints, called Proof Carrying Requests: ideally a small easily verifiable piece of Evidence is provided along with the Request in order to speed decision-making by verifying that the trust least fixed point is above a certain value.

Trust Value was a value on a scale [0,1] or was composed of discrete levels: full distrust, distrust, neutral, middle trust, blind trust. Then, more complex structures appeared: Jøsang's (belief, uncertainty, disbelief) (b,u,d)-triple [4] or the Trust Information Structure of SECURE (see D2.1, D2.2 and D2.3 [12]). Our initial investigations on a range of applications scenarios and early-prototypes (see D5.1 [11], D1.2 [6] and D5.2, which provides an instantiation of the SECURE framework in Java) advocate that the choice of a representation of a Trust Value is possible as long as Trust and Information orderings (see D1.1 [3] and D1.2 [2]) are defined on Trust Values. The SECURE deliverable D1.2 [7] explains how to come up with Trust Value representations compliant with the formal trust model. The finalised version of a Trust Value has yet to be finalised but a (s,i,c)-triple (where s is the number of events that supports a proposition f, i is the number of events that have no information or are inconclusive about f and c is the number of events that contradict f) seems a promising format to be usable in diverse applications (D1.3 [8] develops further the use of (s,i,c)-triples). For example, a (b,u,d)-triple can be computed from (s,i,c)-triples.

The format can be contrasted further from the point of view of privacy protection, interoperability, ease and accuracy of trust calculation, performance and scalability. Trust values can be more or less expressive (i.e., they contain more or less information): the level of expressiveness seems to depend on the application. However, due to the broad range of applications that can be found in global computing, there SHOULD be a Context Mapping mechanism to adjust Trust Values calculated in one application to different applications or more generally different Contexts. Such a mechanism increases interoperability. More expressive representation is likely to help this mapping. A Trust Value may be the aggregation of Trust Values in specific Contexts: this helps to exclude trust irrelevant to the Context of interest. For example, if there are two applications: one for allowing the Requestor to drive a car and another one to ride a motorcycle, the trust value is the aggregation of a trust value for cars and a trust value for motorcycles. It makes sense that the Trust Value for motorcycles can be extrapolated from the Trust Value for cars, because the same traffic laws apply and the ability to position yourself in traffic is similar for cars and motorcycles. A Trust Value MAY simply consist of the inexpressive result of trust calculation due to privacy reasons but it is harder for mapping. In our example, knowing that the Trust Value for car is 0.6 (which can be the result of many pieces of Evidence) is less useful that knowing that the Trust Value contains the success rate of the driving exam questions also found in the motorcycle exam. A Trust Value MAY include these pieces of Evidence to facilitate mapping but this may violate privacy (see the latter example). At the other extreme, a Trust Value MAY only consist of the pieces of Evidence without the trust calculation result, because the Trust Value calculation can reveal more than the value (e.g., how trust is calculated). Another reason may be that the Observer or Recommender are willing to provide objective Evidence without wanting to disclose the subjective feeling represented by some Trust Value calculation. From a performance and scalability point of view, the more trust Contexts are aggregated and pieces of Evidence are stored in the Trust Value, the larger the Trust Value becomes. In fact, performance and scalability are of great concern in global computing where severely resource constrained devices may be found. Large Trust Values mean that fewer can be stored. Past history of one specific

Principal may be longer with Trust Values with more Evidence though. Following (in Table 1) is a qualitative example trade-off summary between Trust Value format choices. The top half of the table describes the Trust Values and the bottom half describes the impact of each type of Trust Value on each of the criteria examined, e.g., the Trust Value format in column 4 contains an inexpressive result of trust calculation and additional pieces of Evidence and has a medium negative effect on privacy risk, a medium positive effect on interoperability and a small negative effect on scalability.

Trust Value format	contains					
Inexpressive result of trust calculation	x		x	x		x
Pieces of Evidence		x	x		x	x
Agreggation according to Context				x	x	x
	estimation					
Privacy risk	=	=	--	-	-	---
Context Mapping / interoperability	-	+	++	+	++	+++
Scalability (based on Trust Value size)	+	-	-	-	--	--

x: the Trust Value contains …
Negative effect compared to other formats: -: small; --: medium; ---: strong
Positive effect compared to other formats: =: similar;+: small; ++:medium; +++:strong

Table 1. Qualitative Assessment of Different Trust Value Formats

An outstanding choice related to the format of Trust Values has still to be made. In SECURE, it is possible to query a Referee to obtain the Trust Value of a Requestor with a Reference Request. If the Trust Value contains an aggregation of Trust Values related to different Contexts/applications, the Referer can choose to request either the full Trust Value or the part of the Trust Value of interest. For example, if the Request for driving a car is made, the Trust Value related to driving a motorcycle is not sent in the Reference Trust Value. Again, there is a privacy issue. Asking for the full Trust Value is a bigger privacy threat for the Referee than sending a specific part of the Trust Value. There is less privacy risk for the Referer because it discloses less about what the Requestor has asked for than if a specific part of the Trust Value is asked. An advantage of getting the full Trust Value is to allow for the best Context Mapping possible without several exchanges between entities involved in the Reference. Due to the notion of reciprocity in privacy concerns, the final choice seems to be in favour of asking for part of Trust Values. In doing so, the Referee knows more about what the Requestor asked the Referer for (to compensate the disclosure of its part of Trust Value) and the Referer is still able to carry out the decision making. The most appropriate way of referencing MAY depend on the type of application though.
A high-level view of the SECURE framework (depicted in Figure 2) exposes a decision-making component that is called when a Decision-Maker has to decide what Decision should be taken due to a Request made by the Requester.

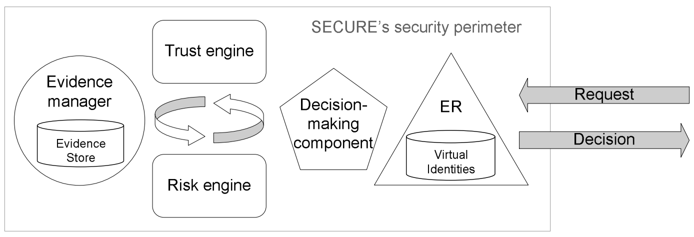


Figure 2: High-level View of the SECURE Framework

The Entity Recognition (ER [10]) module deals with virtual identities (mapping to Principals in SECURE) and is in charge of recognising them.
In order to take this Decision, two sub-components are used:

- a trust engine that dynamically assesses the trustworthiness (i.e., the Trust Value) of the Requester based on pieces of Evidence (e.g., Observation or Recommendation)
- a risk engine that dynamically evaluates the Risk (a combination of Cost and a likelihood of occurrence) involved in the interaction and make the Decision that maintains the appropriate cost/benefit

In the background, another component is in charge of Evidence processing (e.g., gathering Recommendations, comparisons between expected Outcomes of the chosen Actions and real Outcomes…) This Evidence is used to update risk and trust information. Thus, trust and risk follow a managed life-cycle.
In order to improve the design and to benefit from separation of concerns (i.e., reuse of some components and adaptation of others for different implementations), the responsibilities of the components of this high-level view are spread into the following main SECURE components (depicted in Figure 3).

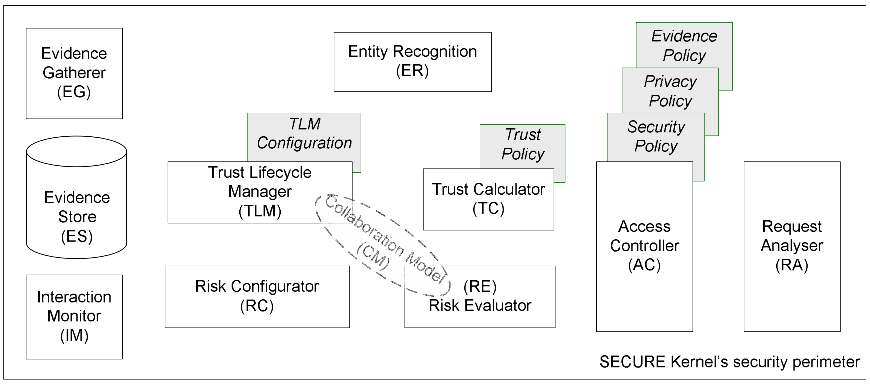


Figure 3. SECURE Framework Components

This decomposition is based on the use of Class-Responsibility-Collaboration (CRC) cards [9] detailed in Section 3 and iterative feedback from early prototyping of an application programming interface (API) and application scenarios (see D5.1 [11], D5.2 and its demonstration prototype). The end of this section briefly describes these main components:

- The difference between shaded and unshaded boxes in Figure 3 is that shaded boxes depend on the application and should be tailored to this application. The changes can be seen as configuration and policy writing tasks.
- The Collaboration Model (CM) is not a component per se but defines how the intrinsic relation between trust and risk is taken into account in SECURE.
- The Request Analyser (RA) is the front-end of the SECURE framework, which receives/sends Requests/Decisions and keeps track of them. This is where the Request's Context is analysed (e.g., what are the parameters for the Action).
- Entity Recognition (ER) is in charge of recognising who made the Request given the ER schemes plugged into the framework and the given Principal (which contains information for the recognition). For message-based ER schemes, the information for the recognition is mainly embedded into the Request. ER is part of the contextualisation of the Request Analyser, follows the ER process (see D4.1 [10]) and provides a level of confidence in recognition.
- The Access Controller (AC) is where policies related to security are enforced. This includes evidence and privacy policies. D3.2 [2] specifically addresses Trust-Based Access Control (TBAC) policies.
- The trust engine is split into two subcomponents:
    - The Trust Calculator (TC) provides the local Trust Values about Principals according to the Trust Policy (where References are specified) with compliance to the formal model of trust (developed in D1.1 [6] and D1.2 [7]).
    - The Trust Lifecycle Manager (TLM) is more related to the Evidence processing in charge of formation and evolution of Trust Values based on new Evidence.
- Similarly, the risk engine is split into two subcomponents:
    - The Risk Evaluator (RE) provides the result of the risk analysis, i.e., the Risk Profiles (see D2.1, D2.3 and D2.3 [12]). These Risk Profiles can be considered as risk probability density functions (RiskPDFs a.k.a. RiskMetrics) introduced in D3.1 [3] that are built from Cost and likelihood of occurrence of Outcomes.
    - The Risk Configurator (RC) is more related to the Evidence process and the notion of feedback (depicted in D3.1 [3]), which is important for the autonomous aspect of the SECURE framework. It monitors and maintains Environmental Context and eventually SHOULD suggest policy changes.
- The Evidence Store (ES) is where information is kept and structured in three layers: the bottom layer with the list of pieces of Evidence; a middle layer with two types of Trust Values (the Trust Value due to Observations and the

Trust Value due to Recommendations) to avoid issues of the use of second-hand pieces of Evidence; the top layer with the combined Trust Values which are used as the local Trust Values.

- The Interaction Monitor (IM) is responsible for the detection of the real Outcome of a past Action given new Evidence for the creation of a new Observation.
- The Evidence Gatherer (EG) is in charge of retrieving and distributing pieces of Evidence. Retrieval and distribution MUST comply with the Evidence Policy due to the security sensitivity of these tasks.

3 SECURE CRC Cards

This section describes the Class-Responsibility-Collaboration (CRC) cards of the SECURE framework components. A responsibility is “a high-level description of the purpose of the class” [9].
We give an example of a CRC card below:

Class Name	
Responsibility 1 of this Class	This Class collaborates with such and such other Classes to fulfil Responsibility 1
Responsibility 2 of this Class	Collaborate with … to fulfil Responsibility 2
…	…
Responsibility[5] n of this Class	Collaborate with …

3.1 Request Analyser

The Request Analyser is used in the UML sequence diagrams of Section 5 dealing with Requests. For example, the contextualisation of the Request, which includes the recognition of the Requester Principal, is described in Figure 11.

Request Analyser (RA)	
Receive/Send Requests/Decisions	
Contextualise Request and Action[6]	
SHOULD maintain history of Requests[7]	
OPTIONAL report on Request workload[8]	Evidence Store

[5] The number of responsibilities n should be around 3 in order to have a good design. A responsibility MUST be implemented if not specified otherwise.
[6] Parameters of the Action and Request type (see the different Request types in Section 4).
[7] A history of the interactions allows the framework to take into account the sequencing of interactions. The history MAY also be used to bypass other checks for performance improvement (e.g., the reply to a trust Reference request is accepted without access control checks, Recommendations initiated and asked by the Decision-Maker MAY be accepted without access control checks).
[8] Such report can be given by the Evidence Store to the Risk Configurator, which uses it to suggest policy changes (e.g., in case of request flooding attacks).

3.2 Access Controller

The Access Controller component is associated with Security Policy (i.e., rules specifying how to make Trust and Risk-based security decision required by an Action from another Principal[9]), Privacy Policy (see Subsection 4.2 and 4.4.2) and Evidence Policy (see Subsection 3.8 and 4.4.1).

Access Controller (AC)	
Make Decision regarding an Action	Trust Calculator, Risk Evaluator, Security Policy
OPTIONAL Register Evidence	Evidence Policy
OPTIONAL Reply to Trust Reference or Recommendation Request	Privacy Policy
OPTIONAL Reply to Selection Request[10]	

3.3 Trust Calculator

The Trust Calculator is associated with the Trust Policy (D1.2 [7] explains how to create a Trust Policy). As an aside D1.2 deals with issues surrounding operationalisation of the formal trust model, e.g., overly long computation of trust least fixed points. Two techniques are proposed: Proof Carrying Requests[4] and a distributed approximation algorithm to compute part of the trust least fixed point in order to improve the responsiveness when needed. The sequence diagram in Figure 5 describes the main steps done in the Trust Calculator.

[9] The SECURE deliverable D3.2 provides a language for Security Policy. Investigations on different applications have suggested that many policies will be based on comparisons of trust and Cost Metrics.

[10] The Selection Request is presented in Subsection 4.2 and Figure 15.

Trust Calculator (TC)	
Evaluate the Trust Policy	Trust Policy
Get local current Trust Value or call the Referee for the Trust Value[11]	Referee, TLM
Contextualise Trust[12], OPTIONAL Evidence	Request Analyser, Risk Evaluator

3.4 Trust Lifecycle Manager

The Trust Lifecycle Manager (TLM) is an essential component of the Collaboration Model (CM) (described in D2.1, D2.2 and D2.3 [12]). The TLM Configuration specifies for all Principals: in case of no Evidence, an initial Trust Value (it MAY be a trust Reference) and trust dynamics parameters. The sequence diagram in Figure 10 is mainly related to the TLM.

Trust Lifecycle Manager (TLM)	
Form/update the Trust Value in a Principal	Evidence Gatherer, Evidence Store
Evolve the Trust Value due to Attraction[13]	
Evaluate the Attraction of pieces of Evidence	Evidence Gatherer, Evidence Store
Adjust Recommendation[14]	

[11] OPTIONAL, only the part of the Trust Value related to the Context of interest (e.g., for driving) is requested in the Reference Request (see Subsections 5.1 and 5.2).

[12] In D3.2, the requested Action MAY span different Contexts of trust (e.g., ability to drive a car or honesty). The Trust Value SHOULD be contextualised (i.e., only the parts of the Trust Value relevant to the Context of the Action should be taken into account) in the Risk Evaluator. The part of the Trust Value relevant to the Context of interest (e.g., the Action) is kept.

[13] Attraction is the measure of the impact the new pieces of Evidence has over two dimensions: Trust Ordering and Information Ordering (see SECURE deliverables on the Collaboration Model).

[14] The Collaboration Model (CM) details *recommendation adjustment*, which means that the level of trust in the Recommender (a.k.a. Recommender Accuracy) is used to raise or lower the impact of the piece of Evidence.

3.5 Entity Recognition

The Entity Recognition (ER) process is specified in D4.1 [10].

Entity Recognition (ER)	
Provide an Entity Recognition Confidence in the Recognised Principal	Entity Recognition Schemes, OPTIONAL Risk Configurator[15]
SHOULD report on ER exceptions[16]	Evidence Store

The outcome of ER [10] MAY be a set of Recognised Principals, i.e., n Principals p associated with a level of confidence in recognition lcr (a.k.a. Entity Recognition Confidence):

$$\sum_{i=1}^{n}(p_i, lcr_i), e.g.\{(PublicKey_1, APERLevel1), (PublicKey_2, APERLevel1)\}$$

The above example is when an APER [10] claim is signed by two keys[17] and both signatures are valid. It may be because both keys are indeed pseudonyms for the same entity or two entities decided to form a group and sign the claim as one entity. However, we envision that ER can be more proactive and uses evidence not directly provided by the Requestor to compute an OPTIONAL probability distribution of Recognised Principals instead of recognising a single entity. A range of methods can be used to compute this distribution (e.g., using fuzzy logic or Bayes). For example, a person among n persons enters a building which is equipped with a biometric vision-based ER scheme (the VER scheme, which is under development, uses vision techniques to recognise people). The outcome of recognition demonstrates hesitation between two persons: p_2 and p_3 are recognized at 45% and 55% respectively. So, all other Principals are given 0%. We have:

$$OutcomeOfRecognition = \sum_{i=1}^{n} lcr_i\, p_i = 0{*}p_1 + 0{,}45{*}\, p_2 + 0{,}55{*}\, p_3 + \ldots + 0{*}p_i + \ldots + 0{*}p_n$$

3.6 Risk Evaluator

The Collaboration Model (CM) also plays a role in the Risk Evaluator: it is where trust exploitation occurs (i.e., the *select* method [12], which is the *starFunction* of the SECURE deliverable D3.1 [3], is applied). The sequence diagram in Figure 4 describes parts of the steps done in the RE.

[15] The Risk Configurator MAY specify how much ER detective work is to be carried out based on Environmental Context.

[16] Different exceptions can occur during the ER process. These exceptions can be evidence of attacks (e.g., dictionary attack on password ER scheme or Denial-of-Service) and can be used by the RC to modify the Environment Context (e.g., the system is now under attack).

[17] We restrain from using other technical trust clues (e.g., key length and algorithm).

Risk Evaluator (RE)	
Lookup relevant Cost Metrics for Outcomes of an Action	Risk Configurator
Select[18] Risk Metrics for Outcomes of an Action based on trust in the Principal[19], Cost Metrics, RECOMMENDED Environmental Context	Risk Configurator

3.7 Risk Configurator

Since statistical analysis is needed for risk analysis, a large number of pieces of Evidence are needed to change (assuming initial values are specified) Cost Metrics, Risk Metrics and Outcomes associated with Actions. The sequence diagram in Figure 16 is mainly related to the Risk Configurator.

Risk Configurator (RC)	
Update Cost Metrics of Actions based on Evidence	Evidence Gatherer, Evidence Store, Risk Evaluator
SHOULD update Environmental Context and Outcomes associated with Actions based on (correlated) Evidence	Evidence Gatherer, Evidence Store, Risk Evaluator
SHOULD suggest policy changes [20]	Access Controller, ER

3.8 Evidence Gatherer

The Evidence Gatherer consults the Evidence Policy for the following tasks:

- To whom evidence MUST be distributed
- As said in the introduction, it SHOULD be checked that Authorisation Hints do not allow for new attacks based on driving what pieces of Evidence are retrieved

[18] Different operations of the Collaboration Model MAY be used (see D2.1, D2.2 and D2.3), e.g., *select*$^{-1}$*()* or calculation of the distance between two Risk Profiles.

[19] This is where trust is exploited.

[20] Environmental Context can be an indicator for the Risk Configurator to suggest a change in policies.

The sequence diagram in Figure 14 is mainly related to the Evidence Gatherer.

Evidence Gatherer (EG)	
Distribute Evidence	Evidence Policy
Gather Evidence[21]	Evidence Store, Request Analyser, Evidence Policy
OPTIONAL focus Gathering of Evidence (OPTIONAL on Principal; OPTIONAL on Authorisation Hints)	Access Controller

3.9 Evidence Store

Evidence is a very broad term. It may be represented as a Trust Value (e.g., the Trust Information Structure of the Collaboration Model is stored in the Evidence Store), an Observation, a Recommendation, another Observable…
The Evidence Store consults the Evidence Policy to know when the following components MUST be notified that new pieces of Evidence have been stored (e.g., it MAY be each time a new piece of Evidence or a batch of Evidence has been received): TLM, Risk Configurator, Interaction Monitor and Evidence Gatherer.

Evidence Store (ES)	
Store Evidence	
Notify new Evidence internally[22]	TLM, IM, RC, EG
Provide local Evidence of interest	
OPTIONAL garbage collect Evidence[23]	TLM, Risk Configurator, Access Controller

The basic mechanism is that each time a new piece/batch of Evidence is stored in the Evidence Store, the Evidence Store notifies components which should know when Evidence of a specific type is updated. This mechanism MAY be optimised as long as the components needing specific Evidence get it on a reasonable basis (i.e., Evidence of importance is not missed). Anyway, in the sequence diagrams of Section 5, all

[21] Gathering MAY imply that the Evidence Gatherer maintains states about how much Evidence of interest is needed and tries to fulfill this need as good as possible (e.g., Recommendation requests MAY be resubmitted or sent to other Observers if not enough Recommendations are obtained).

[22] A said in the introduction, if notification is not possible, an alternative MAY be that the components needing Evidence periodically polls the Evidence Store to check if there is new Evidence of interest.

[23] A piece of Evidence SHOULD be marked as already discarded for scalability and performance reasons when both TLM and RC agree that this piece of Evidence can be discarded or Environment Context suggests that this type of evidence is useless in this Context (and likely future Contexts) with high probability.

notifications are not represented for clarity reasons. For example, the Interaction Monitor stores other pieces of Evidence after being notified of new piece of Evidence demonstrating that a real Outcome has been observed. The sequence diagram depicted in Figure 12 describes the full sequence of notifications occurring when the Evidence Store is called.
The use of a Distributed Hash Table (DHT) in one of the applications [11] challenged the fact that the ES is inside the security perimeter of the SECURE framework. However, this is an implementation issue rather than a design flaw because we can define policies, which logically allows the DHT:

- In the Evidence Policy, any piece of evidence is distributed to all other Principals and Pushed Evidence Requests are unconditionally accepted.
- In the Privacy Policy, any Request is accepted.

Another implementation issue arises when pieces of Evidence are directly captured by the SECURE framework (e.g., sensors directly deliver Observables), this can also be logically allowed if in the Evidence Policy, Pushed Evidence Requests of direct pieces of Evidence are unconditionally stored in the Evidence Store.

3.10 Interaction Monitor

The sequence diagram in Figure 13 describes the main steps done in the Interaction Monitor.

Interaction Monitor (IM)	
Maintain history of Decisions and unresolved Outcomes	Access Controller
Detect an Outcome has occurred based on Evidence	Evidence Store, Evidence Gatherer
Store an Observation based on the Outcome	Evidence Store

4 Main Request Types

Depending on the type of Request, different sequence diagrams are followed. The Request Analyser checks the type of the Request and initiates the right sequence according to the type of the Request. There are four main types of Requests: Action Request, Selection Request, Reference Request and Evidence Request.

4.1 Action Request

This type of Request is the reason for the existence of SECURE. The Request contains an Action that the Requester would like to be carried out. The Decision-Maker has to make a security Decision regarding this Action. The Decision is made after going through the whole trust/risk decision-making process (please refer to the sequence diagrams in Figure 4, Figure 5 and Figure 6).

4.2 Selection Request

This type of Request (see the sequence diagram in Figure 15) is a special case of Action Request. This type is needed for specific applications, where the Decision consists of returning the Risk Metric for each of the different Outcomes of an Action for each Principal of a list of Principals of interest. Then, the best Principal can be chosen based on the Decision. For example, it may be to pick the best online shop (an online shop is a Principal) among a set of shops. Another example occurs in Mobile Ad-Hoc Networks (MANETs), one of the application scenarios (see [11]), when the best route (each route corresponds to a Principal) has to be chosen among a set of different routes. Because there is a list of different Principals, *getTrust* and *evaluateRisk* are simply processed several times, one time for each Principal. Obviously, it SHOULD be checked that the Requester of a Selection Request is allowed to obtain this type of Decision.

4.3 Reference Request

This type of Request occurs when the Trust Policy specifies that the Trust Value of another Principal MUST be used (see the sequence diagram in Figure 7). The Privacy Policy of the requested Principal (i.e., the Referee) determines if the Trust Value should be given out. This Privacy Policy MAY consider that such a Request is equivalent to a trust/risk-mediated Action Request. Information about the part of the Trust Value related to the Context of interest (e.g., the Action is to allow someone to drive a car) MAY be specified in the trust Reference Request due to the reasons discussed in Section 2.

4.4 Evidence Request

This type of Request is used when pieces of Evidence are exchanged between Principals. Any piece of Evidence MUST go through the Request Analyser. As a piece of Evidence can be either pushed by an Observer (i.e., the Principal who provides the piece of Evidence) or pulled by a Receiver (i.e., the Principal who receives the piece of Evidence). There are two subtypes of Evidence Request: Pushed Evidence Request and Pulled Evidence Request.

4.4.1 Pushed Evidence Request

The Evidence Policy determines if the piece of Evidence provided by the Observer should be taken into account by the Receiver (please refer to the sequence diagram in Figure 9). This Evidence Policy MAY consider that such Request is equivalent to a trust/risk-mediated Action Request.

4.4.2 Pulled Evidence Request

The Privacy Policy of the requested Principal (i.e., the Recommender) determines if the pieces of Evidence of interest should be given out (see the sequence diagram in Figure 8). This Privacy Policy MAY consider that such a Request is equivalent to a trust/risk-mediated Action Request.

5 Sequence Diagrams

Each type of Request follows a specific sequence diagram. As said in Section 2, there are two main processes. If trust Reference is needed, Reference Requests follow synchronously from Action Requests. Some Pulled Evidence Requests MAY also arise from Action Requests. Pushed Evidence Requests occur in an asynchronous manner.

5.1 Action Request Sequence Diagram

In step 1.2 of Figure 4, to contextualise an Action means that the Action is parameterised according to parameters embedded into the Request (e.g., this is an Action to accept a bank note and the parameter is the type of bank note). If the Action carries new Context (e.g., this is not a Request for the standard application, which is to allow a person to drive a car, but to allow a person to drive a motorcycle), the contextualisation MAY also consist of mapping the new Context in order to still be able to make a Decision in spite of this novel Context (e.g., part of the Trust Value in the ability to drive a car may be used to derive the Trust Value in the ability to drive a motorcycle). The process of mapping Trust Values to this part of Context is done in step 1.3.3.1 (e.g., the part of the Trust Value related to the Context of Recommenders, a.k.a. Meta-Trust or Recommendation Accuracy, is discarded). In fact, this mapping takes into account Entity Recognition Confidence, which is obtained from the Recognised Principal. The final mapping due to Context, summarized as Context Mapping, is done in step 1.3.3.3, when Environmental Context (e.g., the system is under attack) modifies the *select* function of the Collaboration Model (a.k.a the *starFunction* of the SECURE deliverable D3.1 [3]). The *applyOFunction* of step 1.3.4 of Figure 5, is detailed in D3.1.

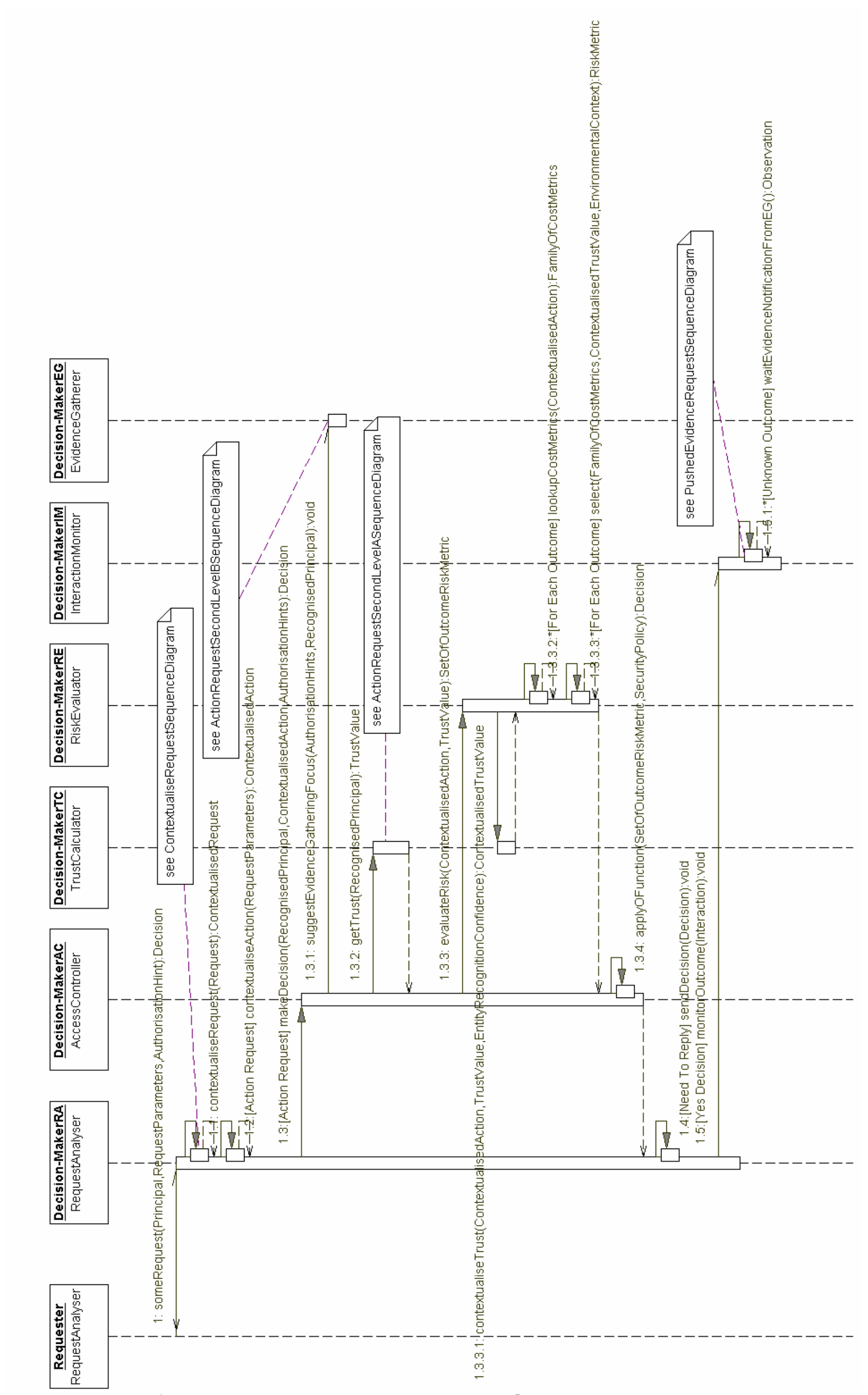
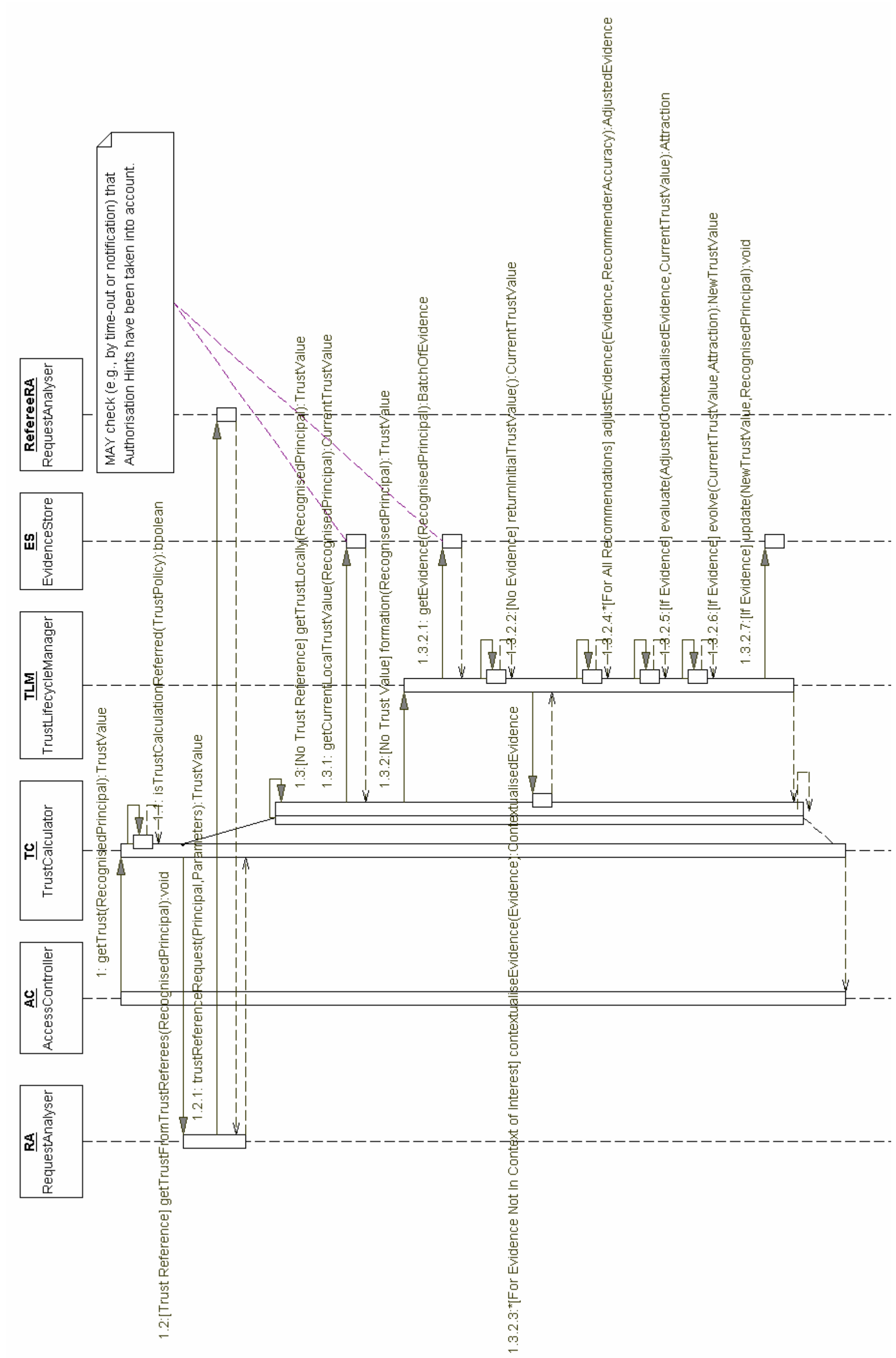

Requester
RequestAnalyser
Decision-MakerRA
RequestAnalyser
Decision-MakerAC
AccessController
Decision-MakerTC
TrustCalculator
Decision-MakerRE
RiskEvaluator
Decision-MakerIM
InteractionMonitor
Decision-MakerEG
EvidenceGatherer
1: someRequest(Principal,RequestParameters,AuthorisationHint):Decision
1.1: contextualiseRequest(Request):ContextualisedRequest
1.2:[Action Request] contextualiseAction(RequestParameters):ContextualisedAction
1.3:[Action Request] makeDecision(RecognisedPrincipal,ContextualisedAction,AuthorisationHints):Decision
1.3.1: suggestEvidenceGatheringFocus(AuthorisationHints,RecognisedPrincipal):void
1.3.2: getTrust(RecognisedPrincipal):TrustValue
1.3.3: evaluateRisk(ContextualisedAction,TrustValue):SetOfOutcomeRiskMetric
1.3.3.1: contextualiseTrust(ContextualisedAction,TrustValue,EntityRecognitionConfidence):ContextualisedTrustValue
1.3.3.2:*[For Each Outcome] lookupCostMetrics(ContextualisedAction):FamilyOfCostMetrics
1.3.3.3:*[For Each Outcome] select(FamilyOfCostMetrics,ContextualisedTrustValue,EnvironmentalContext):RiskMetric
1.3.4: applyOfFunction(SetOfOutcomeRiskMetric,SecurityPolicy):Decision
1.4:[Need To Reply] sendDecision(Decision):void
1.5:[Yes Decision] monitorOutcome(Interaction):void
1.6.1:*[Unknown Outcome] waitEvidenceNotificationFromEG():Observation
see ContextualiseRequestSequenceDiagram
see ActionRequestSecondLevelBSequenceDiagram
see ActionRequestSecondLevelASequenceDiagram
see PushedEvidenceRequestSequenceDiagram

Figure 4. Action Request Top Level Sequence Diagram



RA
RequestAnalyser
AC
AccessController
TC
TrustCalculator
TLM
TrustLifecycleManager
ES
EvidenceStore
RefereeRA
RequestAnalyser
MAY check (e.g., by time-out or notification) that Authorisation Hints have been taken into account.
1: getTrust(RecognisedPrincipal):TrustValue
1.1: isTrustCalculationReferred(TrustPolicy):boolean
1.2:[Trust Reference] getTrustFromTrustReferees(RecognisedPrincipal):void
1.2.1: trustReferenceRequest(Principal,Parameters):TrustValue
1.3:[No Trust Reference] getTrustLocally(RecognisedPrincipal):TrustValue
1.3.1: getCurrentLocalTrustValue(RecognisedPrincipal):CurrentTrustValue
1.3.2:[No Trust Value] formation(RecognisedPrincipal):TrustValue
1.3.2.1: getEvidence(RecognisedPrincipal):BatchOfEvidence
1.3.2.2:[No Evidence] returnInitialTrustValue():CurrentTrustValue
1.3.2.3:*[For Evidence Not In Context of Interest] contextualiseEvidence(Evidence):ContextualisedEvidence
1.3.2.4:*[For All Recommendations] adjustEvidence(Evidence,RecommenderAccuracy):AdjustedEvidence
1.3.2.5:[If Evidence] evaluate(AdjustedContextualisedEvidence,CurrentTrustValue):Attraction
1.3.2.6:[If Evidence] evolve(CurrentTrustValue,Attraction):NewTrustValue
1.3.2.7:[If Evidence] update(NewTrustValue,RecognisedPrincipal):void

Figure 5. Action Request Second Level A Sequence Diagram

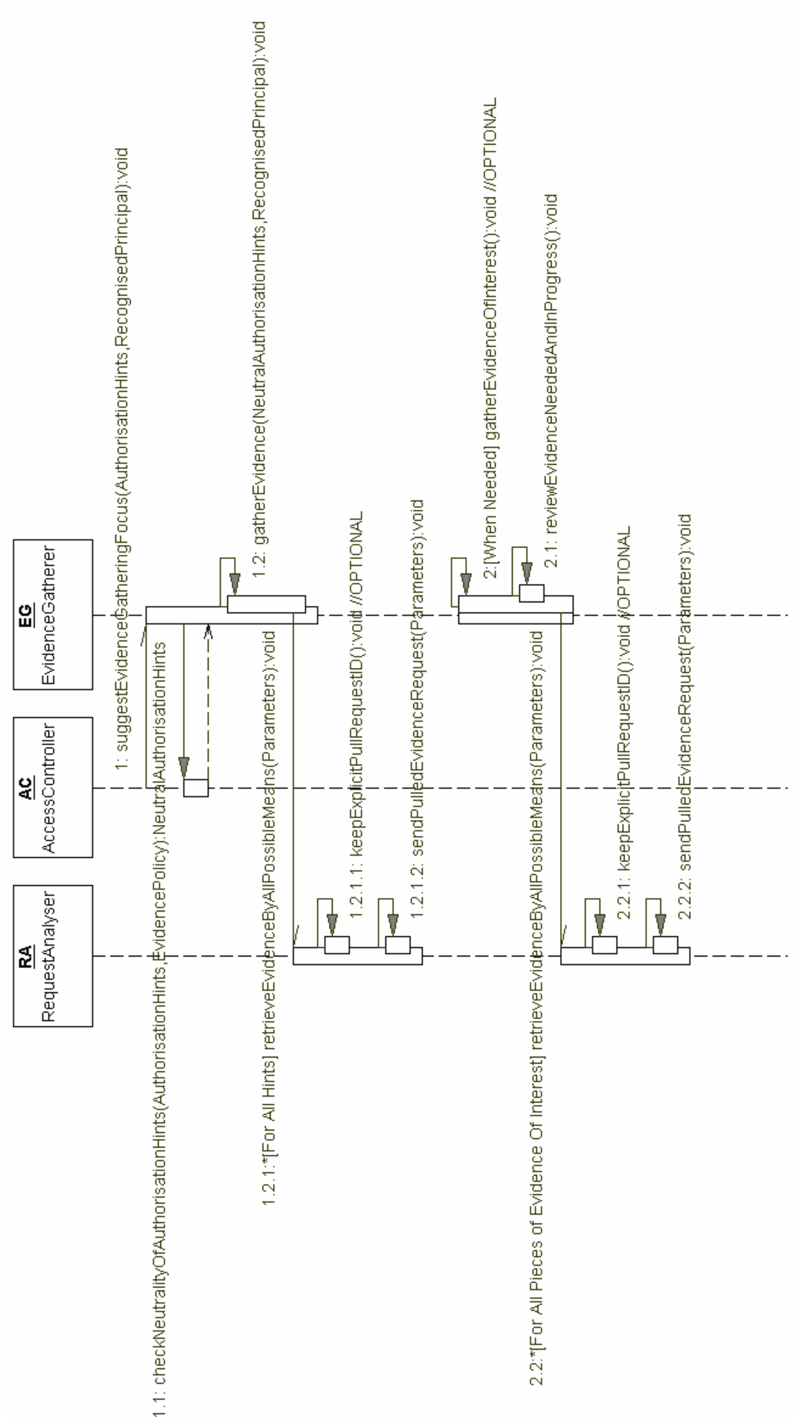

RA
RequestAnalyser
AC
AccessController
EG
EvidenceGatherer
1: suggestEvidenceGatheringFocus(AuthorisationHints,RecognisedPrincipal):void
1.1: checkNeutralityOfAuthorisationHints(AuthorisationHints,EvidencePolicy):NeutralAuthorisationHints
1.2: gatherEvidence(NeutralAuthorisationHints,RecognisedPrincipal):void
1.2.1:*[For All Hints] retrieveEvidenceByAllPossibleMeans(Parameters):void
1.2.1.1: keepExplicitPullRequestID():void //OPTIONAL
1.2.1.2: sendPulledEvidenceRequest(Parameters):void
2:[When Needed] gatherEvidenceOfInterest():void //OPTIONAL
2.1: reviewEvidenceNeededAndInProgress():void
2.2:*[For All Pieces of Evidence Of Interest] retrieveEvidenceByAllPossibleMeans(Parameters):void
2.2.1: keepExplicitPullRequestID():void //OPTIONAL
2.2.2: sendPulledEvidenceRequest(Parameters):void

Figure 6: Action Request Second Level B Sequence Diagram

In step 1.2.1 of Figure 5, information about the part of the Trust Value related to the Context of interest MAY be specified in the trust Reference Request due to the reasons discussed in Section 2. This requires that step 1.3.2 of Figure 4 has the Context of interest as a parameter as well. The Collaboration Model deliverable [12] gives the details of the *adjustment*, *evaluate* and *evolve* functions. In step 1.3.2.4, the Recommender Accuracy MAY be the result of a trust calculation and in this case the Trust Calculator is contacted to obtain the Trust Value of the Recommender regarding Recommendation Accuracy.

5.2 Reference Request Sequence Diagram

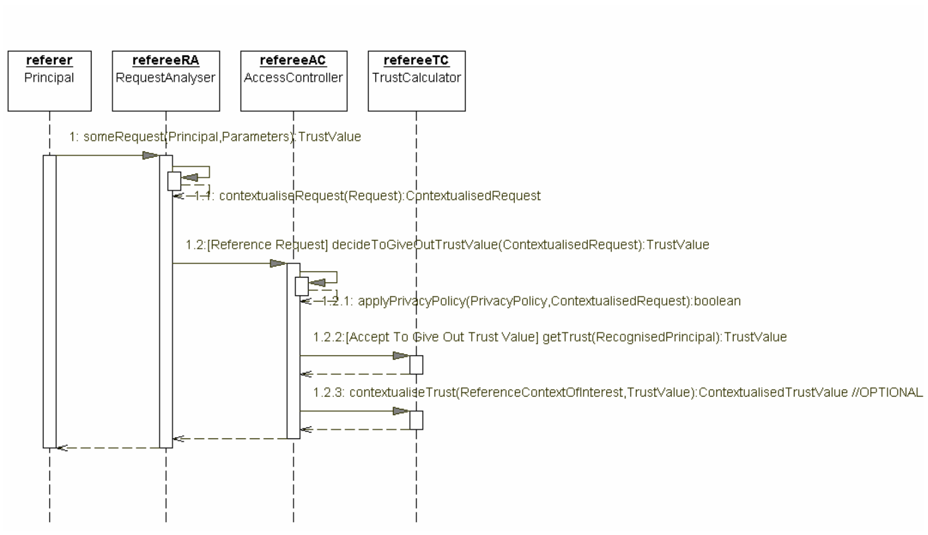

referer
Principal
refereeRA
RequestAnalyser
refereeAC
AccessController
refereeTC
TrustCalculator
1: someRequest(Principal,Parameters):TrustValue
1.1: contextualiseRequest(Request):ContextualisedRequest
1.2:[Reference Request] decideToGiveOutTrustValue(ContextualisedRequest):TrustValue
1.2.1: applyPrivacyPolicy(PrivacyPolicy,ContextualisedRequest):boolean
1.2.2:[Accept To Give Out Trust Value] getTrust(RecognisedPrincipal):TrustValue
1.2.3: contextualiseTrust(ReferenceContextOfInterest,TrustValue):ContextualisedTrustValue //OPTIONAL

Figure 7: Reference Request Sequence Diagram

Alternatively, this type of Request MAY be considered as an Action Request. In step 1.2.3 of Figure 7, due to the reasons discussed in Section 2 and if the trust Reference Request specifies what part of the Trust Value is of interest, the Trust Value MAY be contextualised. For example, if the Trust Value contains two parts, one for an application related to driving a car and another part for an application related to cooking, and the trust Reference concerns driving, only the part for driving is returned.

5.3 Pulled Evidence Request Sequence Diagram

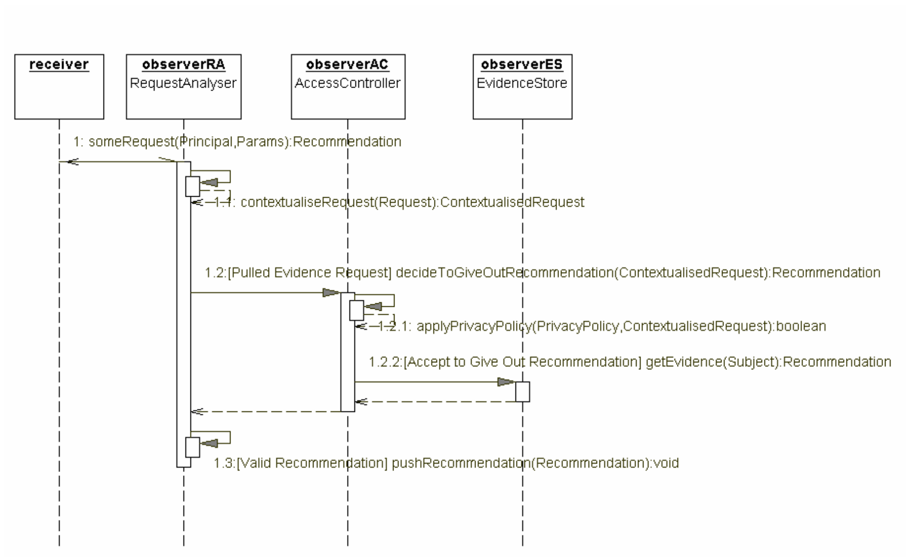

receiver
observerRA
RequestAnalyser
observerAC
AccessController
observerES
EvidenceStore
1: someRequest(Principal,Params):Recommendation
1.1: contextualiseRequest(Request):ContextualisedRequest
1.2:[Pulled Evidence Request] decideToGiveOutRecommendation(ContextualisedRequest):Recommendation
1.2.1: applyPrivacyPolicy(PrivacyPolicy,ContextualisedRequest):boolean
1.2.2:[Accept to Give Out Recommendation] getEvidence(Subject):Recommendation
1.3:[Valid Recommendation] pushRecommendation(Recommendation):void

Figure 8. Pulled Evidence Request Sequence Diagram

Alternatively, this type of request MAY be considered as an Action Request.

5.4 Pushed Evidence Request Sequence Diagram

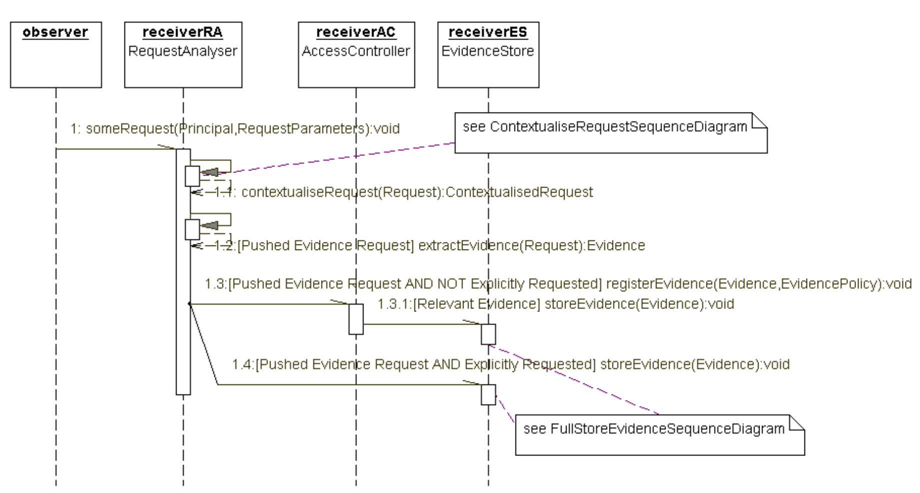

observer
receiverRA
RequestAnalyser
receiverAC
AccessController
receiverES
EvidenceStore
1: someRequest(Principal,RequestParameters):void
see ContextualiseRequestSequenceDiagram
1.1: contextualiseRequest(Request):ContextualisedRequest
1.2:[Pushed Evidence Request] extractEvidence(Request):Evidence
1.3:[Pushed Evidence Request AND NOT Explicitly Requested] registerEvidence(Evidence,EvidencePolicy):void
1.3.1:[Relevant Evidence] storeEvidence(Evidence):void
1.4:[Pushed Evidence Request AND Explicitly Requested] storeEvidence(Evidence):void
see FullStoreEvidenceSequenceDiagram

Figure 9. Pushed Evidence Request Diagram

Alternatively, this type of request MAY be considered as an Action Request. The choice of step 1.4, instead of step 1.3, is OPTIONAL and for performance improvement.

5.5 TLM Evidence Notification Sequence Diagram

In this diagram, the TLM is notified by the Evidence Store that new Evidence is available. Another alternative MAY be that the TLM periodically polls the Evidence Store. For optimisation's sake, in the former case, the Evidence Policy MAY specify when the Evidence Gatherer notifies and, in the latter case, the trust dynamics parameters MAY specify when it is considered that there is enough Evidence for trust formation or evolution. Another optimisation MAY be to specify in the notification what new pieces of Evidence are available and what Evidence of interest should be retrieved in the Evidence Store (e.g., by default a piece of Evidence which has already been used for trust update should not be retrieved).
The Collaboration Model deliverable [12] gives the details of the *adjustment*, *evaluate* and *evolve* functions. In step 1.4, the Recommender Accuracy MAY be the result of a trust calculation and in this case the Trust Calculator is contacted to obtain the Trust Value of the Recommender regarding Recommendation Accuracy.

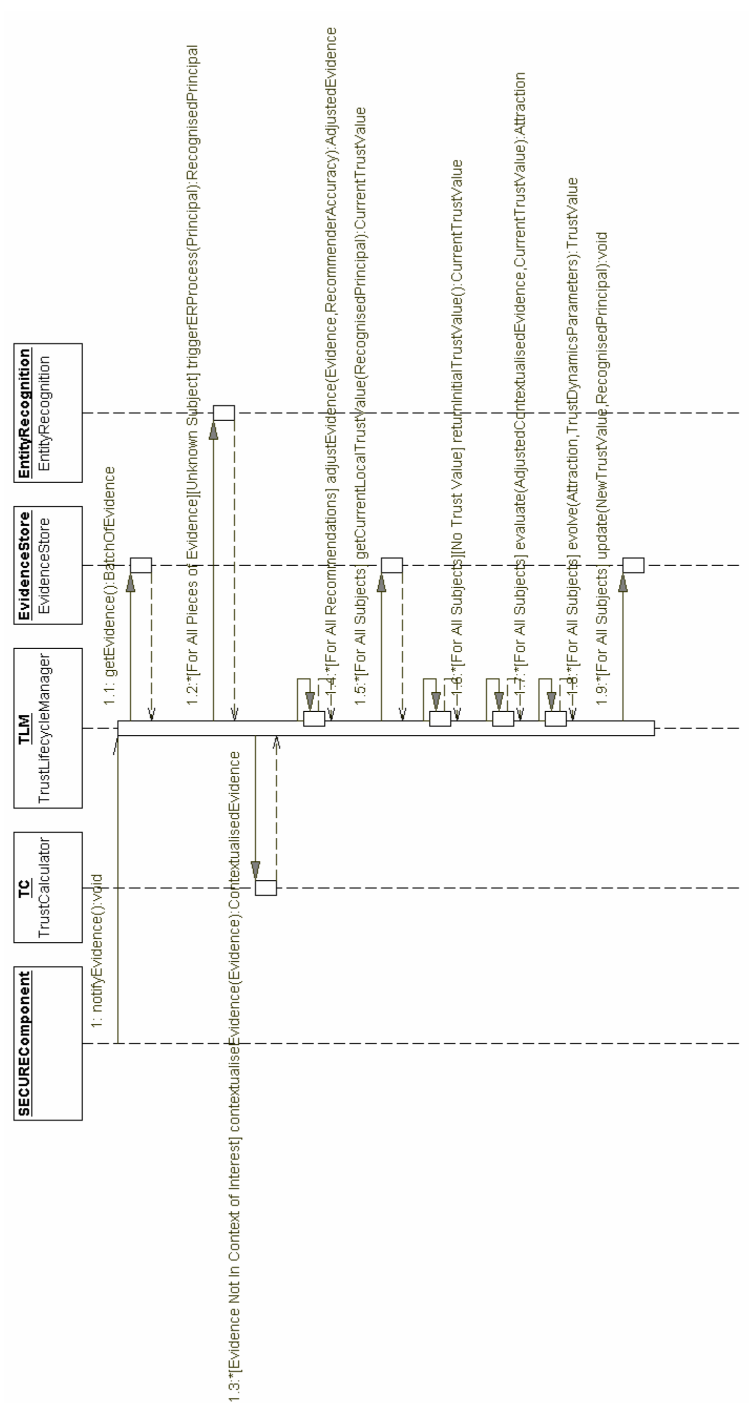

SECUREComponent
TC
TrustCalculator
TLM
TrustLifecycleManager
EvidenceStore
EvidenceStore
EntityRecognition
EntityRecognition
1: notifyEvidence():void
1.1: getEvidence():BatchOfEvidence
1.2:*[For All Pieces of Evidence][Unknown Subject] triggerERProcess(Principal):RecognisedPrincipal
1.3:*[Evidence Not in Context of Interest] contextualiseEvidence(Evidence):ContextualisedEvidence
1.4:*[For All Recommendations] adjustEvidence(Evidence,RecommenderAccuracy):AdjustedEvidence
1.5:*[For All Subjects] getCurrentLocalTrustValue(RecognisedPrincipal):CurrentTrustValue
1.6:*[For All Subjects][No Trust Value] returnInitialTrustValue():CurrentTrustValue
1.7:*[For All Subjects] evaluate(AdjustedContextualisedEvidence,CurrentTrustValue):Attraction
1.8:*[For All Subjects] evolve(Attraction,TrustDynamicsParameters):TrustValue
1.9:*[For All Subjects] update(NewTrustValue,RecognisedPrincipal):void

Figure 10: TLM Evidence Notification Sequence Diagram

5.6 Contextualise Request Sequence Diagram

This is worth noting that pieces of Evidence stored in this sequence diagram are particularly important for the Risk Configurator because they give Evidence on different attacks (e.g., dictionary attacks on ER and flooding attacks on the Request Analyser).

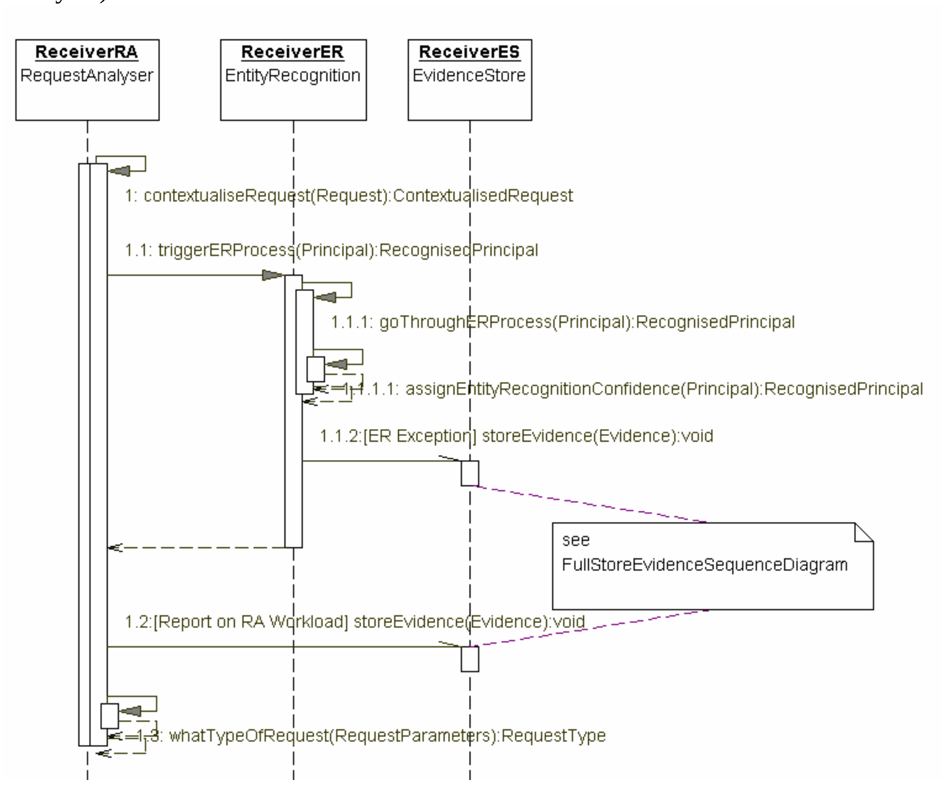

ReceiverRA
RequestAnalyser
ReceiverER
EntityRecognition
ReceiverES
EvidenceStore
1: contextualiseRequest(Request):ContextualisedRequest
1.1: triggerERProcess(Principal):RecognisedPrincipal
1.1.1: goThroughERProcess(Principal):RecognisedPrincipal
1.1.1.1: assignEntityRecognitionConfidence(Principal):RecognisedPrincipal
1.1.2:[ER Exception] storeEvidence(Evidence):void
see
FullStoreEvidenceSequenceDiagram
1.2:[Report on RA Workload] storeEvidence(Evidence):void
1.3: whatTypeOfRequest(RequestParameters):RequestType

Figure 11. Contextualise Request Sequence Diagram

5.7 Full Store Evidence Sequence Diagram

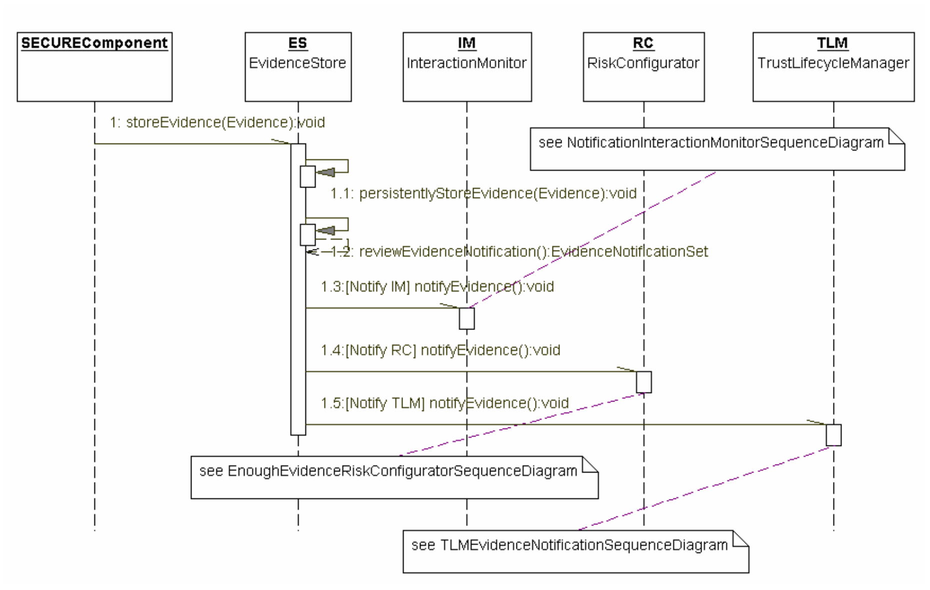

SECUREComponent
ES EvidenceStore
IM InteractionMonitor
RC RiskConfigurator
TLM TrustLifecycleManager
1: storeEvidence(Evidence):void
see NotificationInteractionMonitorSequenceDiagram
1.1: persistentlyStoreEvidence(Evidence):void
1.2: reviewEvidenceNotification():EvidenceNotificationSet
1.3:[Notify IM] notifyEvidence():void
1.4:[Notify RC] notifyEvidence():void
1.5:[Notify TLM] notifyEvidence():void
see EnoughEvidenceRiskConfiguratorSequenceDiagram
see TLMEvidenceNotificationSequenceDiagram

Figure 12. Full Store Evidence Sequence Diagram

In step 1 of Figure 12, the notification MAY also happen when *update(Trust Value, RecognisedPrincipal)* is called on the Evidence Store. Alternatively, the notified components MAY periodically poll the Evidence Store for new pieces of Evidence.

5.8 Notification Interaction Monitor Sequence Diagram

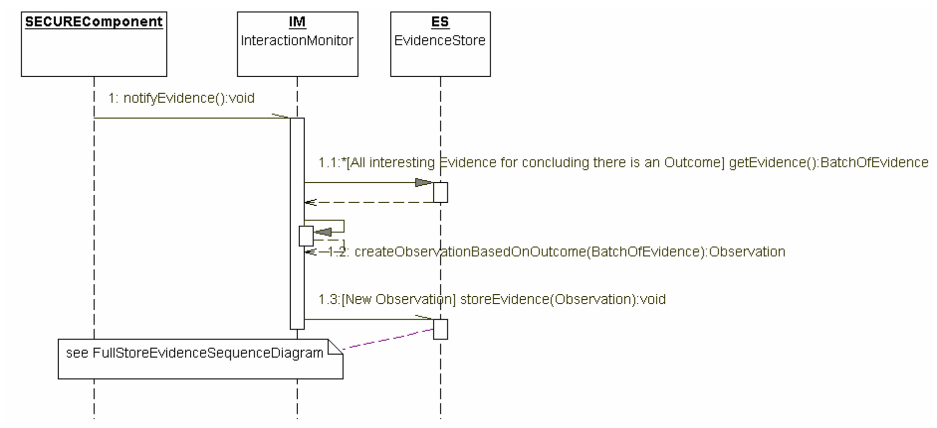

SECUREComponent
IM InteractionMonitor
ES EvidenceStore
1: notifyEvidence():void
1.1:*[All interesting Evidence for concluding there is an Outcome] getEvidence():BatchOfEvidence
1.2: createObservationBasedOnOutcome(BatchOfEvidence):Observation
1.3:[New Observation] storeEvidence(Observation):void
see FullStoreEvidenceSequenceDiagram

Figure 13. Notification Interaction Monitor Sequence Diagram

In step 1 of Figure 13, an alternative MAY be that the notifying SECURE component is indeed the Interaction Monitor itself: it triggers the sequence periodically or ideally when needed.

5.9 Notification Evidence Gatherer Sequence Diagram

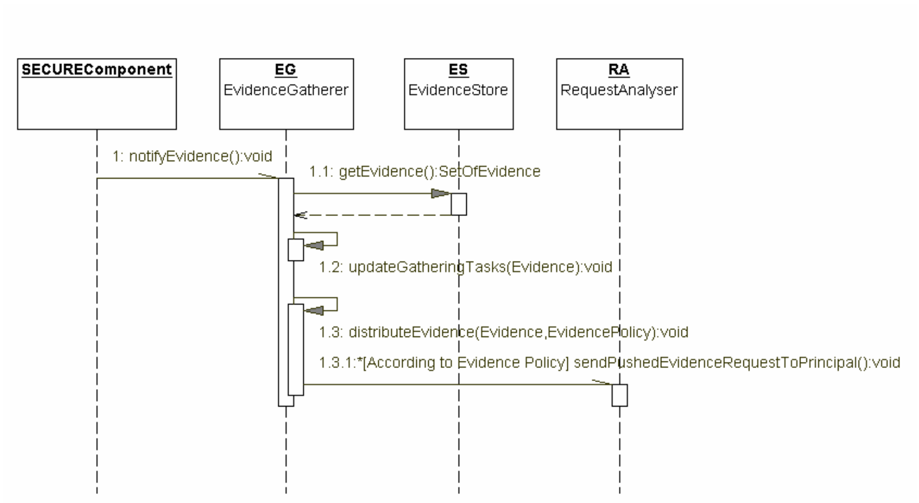

SECUREComponent
EG
EvidenceGatherer
ES
EvidenceStore
RA
RequestAnalyser
1: notifyEvidence():void
1.1: getEvidence():SetOfEvidence
1.2: updateGatheringTasks(Evidence):void
1.3: distributeEvidence(Evidence,EvidencePolicy):void
1.3.1:*[According to Evidence Policy] sendPushedEvidenceRequestToPrincipal():void

Figure 14. Notification Evidence Gatherer Sequence Diagram

In step 1 of Figure 14, an alternative MAY be that the notifying SECURE component is indeed the Evidence Gatherer itself: it triggers the sequence periodically or ideally when needed.

5.10 Selection Request Sequence Diagram

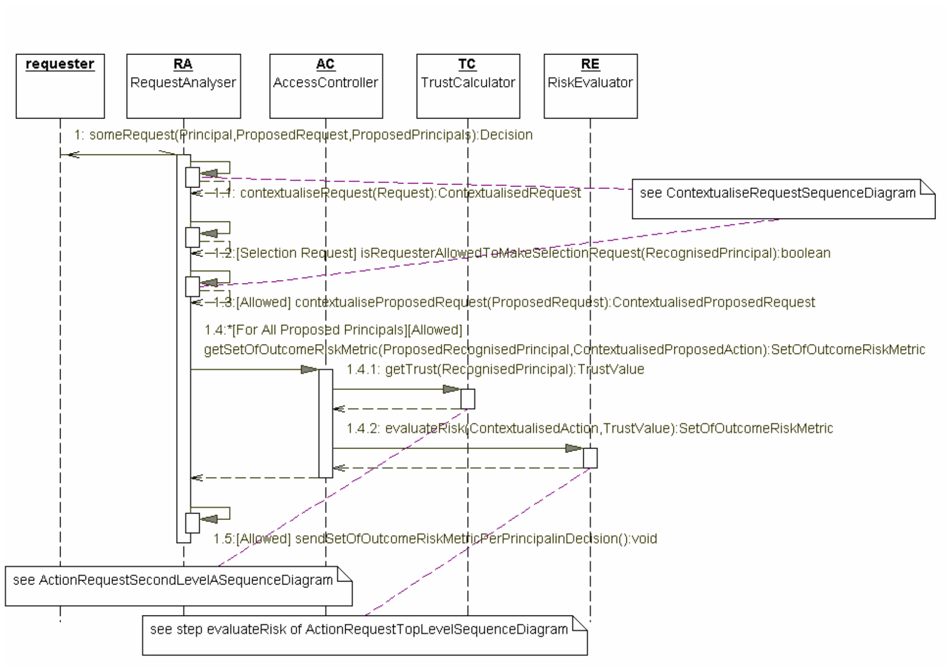

requester
RA
RequestAnalyser
AC
AccessController
TC
TrustCalculator
RE
RiskEvaluator
1: someRequest(Principal,ProposedRequest,ProposedPrincipals):Decision
1.1: contextualiseRequest(Request):ContextualisedRequest
see ContextualiseRequestSequenceDiagram
1.2:[Selection Request] isRequesterAllowedToMakeSelectionRequest(RecognisedPrincipal):boolean
1.3:[Allowed] contextualiseProposedRequest(ProposedRequest):ContextualisedProposedRequest
1.4:*[For All Proposed Principals][Allowed]
getSetOfOutcomeRiskMetric(ProposedRecognisedPrincipal,ContextualisedProposedAction):SetOfOutcomeRiskMetric
1.4.1: getTrust(RecognisedPrincipal):TrustValue
1.4.2: evaluateRisk(ContextualisedAction,TrustValue):SetOfOutcomeRiskMetric
1.5:[Allowed] sendSetOfOutcomeRiskMetricPerPrincipalinDecision():void
see ActionRequestSecondLevelASequenceDiagram
see step evaluateRisk of ActionRequestTopLevelSequenceDiagram

Figure 15. Selection Request Sequence Diagram

As explained in Subsection 4.2, Selection Request is a special case of Action Request, which is worth describing because it is useful in different scenarios (e.g., the selection of the most appropriate route in a MANET).

5.11 Enough Evidence Risk Configurator Sequence Diagram

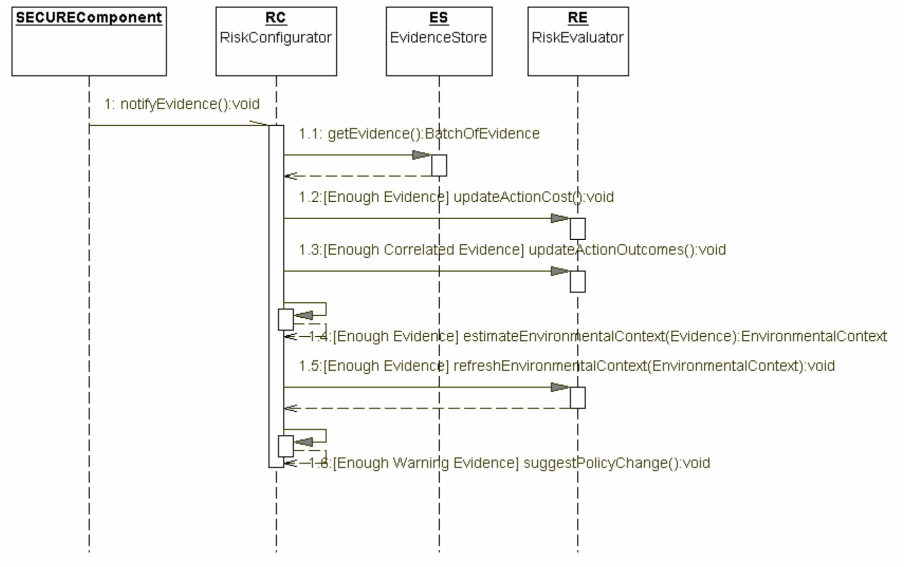

SECUREComponent
RC
RiskConfigurator
ES
EvidenceStore
RE
RiskEvaluator
1: notifyEvidence():void
1.1: getEvidence():BatchOfEvidence
1.2:[Enough Evidence] updateActionCost():void
1.3:[Enough Correlated Evidence] updateActionOutcomes():void
1.4:[Enough Evidence] estimateEnvironmentalContext(Evidence):EnvironmentalContext
1.5:[Enough Evidence] refreshEnvironmentalContext(EnvironmentalContext):void
1.6:[Enough Warning Evidence] suggestPolicyChange():void

Figure 16: Enough Evidence Risk Configurator Sequence Diagram

In step 1 of Figure 16, an alternative MAY be that the notifying SECURE component is indeed the Risk Configurator itself: it triggers the sequence periodically or ideally when needed.

6 Conclusion & Future Work

The SECURE project is investigating the feasibility of autonomous trust/risk-based security in the context of global computing, characterized by (software, hardware and communication) heterogeneity, uncertainty and a large number of previously unknown roaming entities. The goal is that the SECURE framework can be instantiated under different operating assumptions to implement trust/risk management for different application scenarios in a way that ensures compliance with the SECURE formal model of trust. Based on feedback from early prototyping of an application programming interface (API) and application scenarios, we have identified required, recommended and optional functionalities of such a framework. This identification task facilitates alternative implementations based on the same specifications. This is further facilitated because each component of the architecture is responsible for a specific concern, which allows for the reuse of some components and the tailoring of others for diverse implementations.
Refinements are needed in some areas. We are close to make the final choice regarding the Trust Value format since the revision of the formal model of trust (please refer to D1.3) indicates that (s,i,c)-triples bring the required properties and can be used in a broad panel of applications. Concerning the Collaboration Model, the impact of the sequencing of interactions should be investigated. Since Environmental

Context seems important for auto-configuration and the autonomous goal of the SECURE framework, the Risk Configurator may need better mechanisms for risk to evolve in a similar manner to trust and eventually to carry out policy changes.
We are pursuing our iterative feedback process between formal modelling, design, API implementation and prototyping of diverse applications to refine these outstanding issues. Most of the work is now shifted on prototyping and validation though. Future work is no less challenging since validation of trust/risk-based security frameworks has recently been argued to be difficult (according to [5]).

7 Acknowledgment

This work is sponsored by the European Union, which funds the IST-2001-32486 SECURE project, part of the EU Global Computing initiative. The authors would like to acknowledge the very helpful interaction we have had with all the members of the project consortium: the Object Systems Group at University of Geneva (Switzerland), the SmartLab at University of Strathclyde (Scotland), the Opera Group at University of Cambridge (England) and Basic Research in Computer Science (BRICS) at University of Aarhus (Denmark), with a special acknowledgment for their usable formal model of trust.

8 References

[1] S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels", RFC2119, IETF, 1997, http://www.ietf.org/rfc/rfc2119.txt.

[2] N. Dimmock, J. Bacon, A. Belokosztolszki, D. Eyers, D. Ingram, and K. Moody, "Preliminary Definition of a Trust-based Access Control Model", SECURE Deliverable 3.2, 2004, http://secure.dsg.cs.tcd.ie.

[3] D. Ingram, J. Bacon, N. Dimmock, K. Moody, B. Shand, and A. Twigg, "Definition of Risk Model", SECURE Deliverable 3.1, 2003, http://secure.dsg.cs.tcd.ie.

[4] A. Jøsang, "A Logic for Uncertain Probabilities", in *Fuzziness and Knowledge-Based Systems*, vol. 9(3), 2001.

[5] M. Langheinrich, "When Trust Does Not Compute – The Role of Trust in Ubiquitous Computing", in *Proceedings of Privacy Workshops of Ubicomp'03*, 2003, http://guir.berkeley.edu/pubs/ubicomp2003/privacyworkshop/papers/ubicomp-trust.pdf.

[6] N. Mogens, M. Carbone, O. Danvy, I. Damgaard, K. Krukow, A. Møller, and J. B. Nielsen, "A Model for Trust", SECURE Deliverable 1.1, 2003, http://secure.dsg.cs.tcd.ie.

[7] N. Mogens, M. Carbone, and K. Krukow, "An Operational Model", SECURE Deliverable 1.2, 2004, http://secure.dsg.cs.tcd.ie.

[8] N. Mogens, M. Carbone, and K. Krukow, "Revised Computational Trust Model", SECURE Deliverable 1.3, 2004, http://secure.dsg.cs.tcd.ie.

[9] F. Scott, "UML Distilled", Addison Wesley, 2000.

[10] J.-M. Seigneur, S. Farrell, C. D. Jensen, E. Gray, and Y. Chen, "End-to-end Trust Starts with Recognition", in *Proceedings of the First International Conference on Security in Pervasive Computing*, pp. 130-142, LNCS 2804, Springer-Verlag, 2003, http://www.informatik.uni-trier.de/~ley/db/conf/spc/spc2003.html.

[11] G. d. M. Serugendo, C. Bryce, V. Cahill, C. English, S. Farrell, E. Gray, C. D. Jensen, P. Nixon, J.-M. Seigneur, S. Terzis, W. Wagealla, and Y. Chen, "Application Scenarios", SECURE Deliverable 5.1, 2003, http://secure.dsg.cs.tcd.ie.

[12] S. Terzis, W. Wagealla, C. English, A. McGettrick, and P. Nixon, "The SECURE Collaboration Model", SECURE Deliverables D2.1, D.2.2 and D2.3, 2004, http://secure.dsg.cs.tcd.ie.