Security Requirements Engineering for Socio-Technical Systems

Fabiano Dalpiaz
University of Toronto, Canada
University of Trento, Italy

www.cs.toronto.edu/~dalpiaz
dalpiaz@cs.toronto.edu

September 27th, 2012
Outline

Part 1. Security Requirements: the landscape
- Security Requirements Engineering (SRE)
- Socio-Technical Systems (STSSs)

Part 2. The Socio-Technical Security modeling language (STS-ml)
- Multi-view modeling
- Reasoning about security requirements
- Deriving the security requirements document
- Tool support: STS-Tool
1. Security Requirements: the landscape
Why Security Requirements Engineering (SRE)?

Oversimplifying...

... the system shall meet the goals of the stakeholders... and it has to be secure!

The system will use RSA-1024, SSL 3.0, RBAC, CBC-MAC, ...

What???

Requirements Engineer

Security Expert

Different perspectives, primitives, and vocabularies
Security requirements and solutions

**Security requirements**

- Needs concerning security that stakeholders express
- Define *why* security is needed and *what* security need
  - e.g., “confidentiality”, “integrity”, …
- Trade-offs with other types of requirements [Elahi 07]
- Security is a **socio-technical problem**
  - e.g., someone leaves confidential printouts in an open space

**Security solutions**

- Encryption algorithms, authentication protocols, …
Socio-Technical Systems

- Today's systems a complex interplay of different subsystems
  - Not only technical systems, but also **humans** and **organizations**
  - Each subsystem is **autonomous**
  - Subsystems **interact** to fulfill their purpose

- Such systems are **Socio-Technical Systems**
  - Examples include eHealth systems, air traffic management, smart cities, disaster response, etc.
  - The term was first introduced in social science [Emery 59]

[Dalpiaz 12]
An example of STS: smart homes
Another example of an STS

Fabiano

Cheap travels inc.

flight itinerary identified

amaDEUS.net

Itinerary provided

Air Canada

Porter Airlines

CityHotel

social dependence
The security problem in STSs

- **Interaction** is everywhere!
  - Technical Sys to Technical Sys
  - Technical Sys to Social Actor (and vice versa)
  - Social Actor to Social Actor

- Subsystems are autonomous!
  - **No control → how to guarantee security?**
  - Existing approaches are not well suited
    - Take a rather centralized viewpoint
    - Limited account of interaction
The security requirements engineer

... the socio-technical system shall ensure confidentiality of Fabiano's personal data, because...
Several **concepts** are to be considered

- Security goal
- Security policy
- Security requirements
- Security mechanism
- Security risk
- Threat
- Attack
- Asset
- Vulnerability

**SRE approaches are not comprehensive**

[Firesmith 04]
Analysis framework: SRE for STSs

- We review existing approaches by considering these factors
  - **Primitive concepts**
  - **Modeling language**
    - None, informal, formal
  - **Perspective**
    - Organizational, Attacker-oriented, System-oriented
  - **Existence of a method**
  - **Development phase**
    - Early requirements, late requirements, support for design?
  - **Automated analysis tools**
  - **Socio-Technical perspective**
SRE methods

- They describe how to conduct an SRE process
  - Focus on the elicitation **method and key concepts**
  - No social perspective
  - No modeling language nor formal analysis

Requirements types – guidelines to elicit each type [Firesmith 03]
- Threat-driven
- Types: identification, authentication, authorization, immunity, integrity, intrusion detection, nonrepudiation, privacy,...

SQUARE: Security Quality Requirements Engineering [Mead 05]
- 9-steps methodology
- Goal-based at the early stage
- Includes risk assessment
- Prioritization
- Expressly thought for practitioners
SRE: The “Abuse/Misuse” family

- **Crux:** elicit and model **attack scenarios**
  - Focus is mainly on system to-be usage, few early concerns
  - Informal modeling language, no analysis
  - No social perspective (only system-user)

Misuse cases [Sindre 00]  
Abuse cases [McDermott 99]
SRE: Anti-models

- Aim at identifying which are the goals of attackers
  - Higher level of abstraction, a goal comprises several scenarios
  - Formal modeling language
  - **Automated analysis** is possible: finding the “best” alternative
  - No social primitives

[van Lamsweerde 04]
SRE: Context and Argumentation

- Create a context for the system, using problem frames
- Develop satisfaction arguments for security requirements
  - Arguments fail if (1) the security requirements are not satisfiable in the context, or (2) the context is insufficient to develop the argument
- Formal language, automated reasoning is possible
  - System-oriented, no social perspective

[Haley 08]
Explicitly acknowledge that the system involves the interaction among a number of actors **socially depending** on one another

- Exploit variants of the *i* framework [Yu 96]
- **Organizational perspective**
- Capture security at the early requirements stage

**SI* [Zannone 07]**

- Formal modeling language based on delegation, permission on assets, and trust
- Automated reasoning
Secure Tropos [Mouratidis 07]
- Security constraints on dependencies
- Secure entities (goals, tasks, resources) to talk about security requirements
- From early reqs to design
  - Extends the Tropos methodology [Bresciani 2004]
  - Detailed design with AUML

Privacy [Liu 03]
- Puts together the organizational perspective and the attacker perspective
- Detect vulnerabilities → Identify countermeasures → Specify access control
2. The Socio-Technical Security modeling language (STS-ml)

Co-authors: Elda Paja, Paolo Giorgini
STS-ml

- Role- and goal-oriented requirements modeling language
  - Organizational perspective
- Models are built diagrammatically
  - Multiple views, each focusing on a specific perspective
  - Formal language that supports automated reasoning about the expressed security needs
- Security reqs as social contracts that constrain interactions
  - Social dependence
  - Documents exchange

[Dalpiaz 11]
Security Requirements Engineer

expresses security needs

STS-ml views: social, information, authorization

automated derivation via STS-Tool

Security Requirements Document
The STS-ml method

Modeling Activities

Phase 1. Model the Social View
Step 1.1 Identify stakeholders
Step 1.2 Identify assets and interactions
Step 1.3 Express security needs

Phase 2. Model the Information View
Step 2.1 Identify information and its owner
Step 2.2 Represent information structure

Phase 3. Model the Authorization View
Step 3.1 Model authorizations

Phase 4. Automated analysis
Step 4.1 Consistency analysis
Step 4.2 Security analysis

Phase 5. Derive Security Requirements
Step 5.1 Derive security requirements document

[refinement needed]
[analysis errors/warnings]
Phase 1. Modeling the social view

- **Step 1.1 Identify stakeholders**
  - Agents and roles

- **Step 1.2 Identify assets and interactions**
  - **Assets**: goals, documents
  - **Interactions**: goal delegations and document provisions

- **Step 1.3 Express security needs**
  - Express expectations concerning security over interactions
    - Elicited from the stakeholders
Social view: an example

- **Agent**: Bob, Tourist, Hotel
- **Role**: Play, Trip planned, Need
- **Goal**: Hotel booked, Tickets booked, Traveling Order
- **Document Provision**: Amadeus Service, Flight ticket booked, Itinerary details
- **Document Delegation**: Tickets booked, Need

- **Need**: Itinerary details, Produce
- **Verify**: Credit card verified
- **Generate**: Flight ticket generated

This diagram illustrates the social view of a travel booking scenario, highlighting the interactions between roles, goals, and documents.
Step 1.1 Identify stakeholders

- Elicit **roles** and **agents**
  - Role is an abstract characterization of the behavior of an active entity within some context
    - Most participants are unknown at design time
    - e.g., Tourist, Travel Agency Service, Hotel, …
  - Agents play (adopt) roles at runtime, and they can change the roles they play
    - e.g., Bob, Fabiano, CheapTravels Inc.
    - Some agents are known, e.g., Amadeus Flight Service
Step 1.2 Identify assets and interactions

- A **goal** is a state of affairs that an actor intends to achieve
  - e.g., trip planned, flight tickets booked
  - Used to capture motivations and responsibilities of actors
- Goal can be decomposed (refined)

- **Or-decomposition**
  - Tickets booked
    - Train ticket booked
    - Flight ticket booked

- **And-decomposition**
  - Trip planned
    - Tickets booked
    - Hotel booked
Step 1.2 Identify assets and interactions

- **Goal delegation**
  - An actor (delegator) delegates the fulfillment of a goal (delegatum) to a different actor (delegatee)
    - Lack of capability or transfer of responsibility
  - e.g., Tourist is not capable of booking the tickets on his own, he depends on a Travel Agency Service to achieve this goal
  - In STS-ml, only leaf goals can be delegated
Step 1.2 Identify assets and interactions

- A **document** represents an exchangeable entity which may contain some information
  - Actors possess or manipulate documents to achieve their goals

- Goal-document relationships
  - An actor may **need** one or more documents to fulfill a goal
  - An actor may **produce** documents while fulfilling a goal
  - An actor may **modify** a document while fulfilling a goal
Step 1.2 Identify assets and interactions

**Document provision**

- Captures an exchange of documents between a provider actor and a providee actor
- Provider: an actor that possesses the document
- Providee: an actor that might need documents to achieve its goals
Step 1.3 Express security needs

Non-delegation

The re-delegation of the fulfillment of a goal is forbidden

Non-repudiation

- The **delegator** cannot repudiate he delegated
- The **delegatee** cannot repudiate he accepted the delegation
Step 1.3 Expressing security needs

**Redundancy**
- Alternative ways of achieving a goal
- Different redundancy types
  - True and Fall-back
  - Single- and Multi-Actor

**Combine/Incompatible**
- Two goals shall be achieved by different (the same) actors
  - e.g., two people required to open a safebox
- Two roles are incompatible, i.e., cannot be played by the same agent
  - e.g., one cannot process his own loan application!
Step 1.3 Expressing security needs

**Integrity of transmission**

- The document shall not be altered during the transmission from the sender to the receiver.
Social view: expressing security needs

- non-delegation
- non-repudiation
- redundancy
- Integrity of transmission
- Incompatibility (separation of duties)
Phase 2. Modeling the information view

- Step 2.1 Identify information and its owner
  - Confidentiality requirements are concerned with protecting the disclosure and usage of information
  - Documents represent information
  - Represent the owners of different information

- Step 2.2 Represent information structure
  - **Tangible By**: information → document
  - **Part Of**: info (doc) → info (doc)
Information view: an example

ownership

Amadeus Service

Tangible By

Eticket

Tangible By

Personal data

Tangible By

Tourist

Tangible By

ID Doc Copy

PartOf

Traveling Order

Tangible By

TAS

Tangible By

Itinerary details

PartOf

Destination

PartOf

Schedule

Itinerary
Phase 3. Modeling the authorization view

- Step 3.1 Model **authorizations**
  - Transfer of rights/permission between actors
- Authorizations about information, specifying
  - **Scope of usage** (a set of goals)
    - The customer permits the travel agency to use her personal data only to book the tickets
  - Allowed **operations**: use (read), modify, produce, distribute
  - **Transferability**
    - Further propagate rights to other actors
Authorization view: an example

Allowed operations: **Use**, Modify, Produce, Distribute

- **Tourist**
  - **Personal data**
    - **Hotel booked**
  - **Itinerary**
    - **Tickets booked**
  - **TAS**
    - **Itinerary**
      - **Personal data**
    - **Personal data**
    - **Flight ticket booked**

- **Destination**
- **Schedule**
- **PartOf**
- **PartOf**
- **Itinerary**
- **Amadeus Service**

Scope of information:
Expressing security needs via authorizations

Security needs about **confidentiality** are expressed by allowing only certain operations and limiting the scope

- **Need-to-know** ← limiting the scope
- **Non-usage** ← not allowing usage
- **Non-modification** ← not allowing modification
- **Non-production** ← not allowing production
- **Non-disclosure** ← not allowing distribution
Security needs via authorizations

Need-to-know: can use personal data only in the scope of hotel booked

Non-production: cannot produce documents that represent personal data or itinerary
Security needs via authorizations

Non-disclosure: documents representing personal data or itinerary cannot be distributed

Non-modification: cannot modify documents representing personal data
Phase 4. Automated analysis

- **Step 4.1 Consistency analysis**
  - Does the model comply with the semantics of STS-ml?
  - E.g., part-of cycles, contribution cycles

- **Step 4.2 Security analysis**
  - Security requirements cannot be fulfilled in the modeled STS
  - E.g., Non-delegation, non-usage, non-disclosure
Security analysis

- It relies upon generating possible worlds
  - Identify and visualize possible problems
  - The engineer fixes the problem
  - Behind-the-scenes: formalization in disjunctive datalog

![Diagram of security analysis process]
Phase 5. Derive Security Requirements

- Requirements **models** are useful for **communication** purposes with the stakeholders

- Requirements **specifications** tell **designers** what the system has to implement
  - In STS-ml, security requirements specifications are automatically derived from requirements models
  - Output: security requirements document
Requirements specifications via commitments

- In STS-ml
  - Security requirements constrain interactions in contractual terms
  - These contracts are expressed as **social commitments**
- Social commitment: a promise with contractual validity
  - made by a *debtor* actor to a creditor *actor*
  - that a state of affairs will be brought about [*consequent*]
  - (optional) provided that another state of affairs holds [*antecedent*]

  e.g., C(Fabiano, RE12-Chairs, slot allocated, tutorial presented)

  [Singh 99]
Commitments as requirements

- Commitments can express requirements
  - Rather than specifying a single system, they specify the interaction among systems

- Security requirements via commitments
  - Debtor actor = Responsible
  - Creditor actor = Requester
  - Antecedent = Precondition
  - Consequent = Security requirement
Deriving security requirements: an example

1. non-repudiation

2. redundancy

3. non-delegation

4. Integrity of transmission

5. Incompatibility (separation of duties)
### Derived security requirements

<table>
<thead>
<tr>
<th>Responsible</th>
<th>Security Requirement</th>
<th>Requester</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAS</td>
<td>Non-repudiation-of-acceptance (delegated(Tourist,TAS,tickets booked))</td>
<td>Tourist</td>
</tr>
<tr>
<td>Tourist</td>
<td>Non-repudiation-of-delegation (delegated(Tourist,TAS,tickets booked))</td>
<td>TAS</td>
</tr>
<tr>
<td>TAS</td>
<td>True-redundancy-multiple-actor (tickets booked)</td>
<td>Tourist</td>
</tr>
<tr>
<td>Hotel</td>
<td>Non-delegation (hotel booked)</td>
<td>Tourist</td>
</tr>
<tr>
<td>Amadeus Service</td>
<td>Integrity-of-transmission (provided(TAS,Amadeus Service,Itinerary details))</td>
<td>TAS</td>
</tr>
<tr>
<td>any</td>
<td>Not-achieve-both (eticket generated,credit card verified)</td>
<td>Org</td>
</tr>
</tbody>
</table>
Deriving security requirements: an example

Non-modification, Non-production, Non-disclosure, Need-to-know
## Derived security requirements

<table>
<thead>
<tr>
<th>Responsible</th>
<th>Security Requirement</th>
<th>Requester</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAS</td>
<td>need-to-know(personal data (\land) itinerary, tickets booked)</td>
<td>Tourist</td>
</tr>
<tr>
<td>TAS</td>
<td>non-modification(personal data (\land) itinerary)</td>
<td>Tourist</td>
</tr>
<tr>
<td>TAS</td>
<td>non-production(personal data (\land) itinerary)</td>
<td>Tourist</td>
</tr>
<tr>
<td>TAS</td>
<td>non-disclosure(personal data (\land) itinerary)</td>
<td>Tourist</td>
</tr>
</tbody>
</table>
STS-Tool

- STS-Tool is the modeling and analysis support tool for STS-ml
  - Built on top of Eclipse
    - Standalone Eclipse RCP application
  - Freely available for download: http://www.sts-tool.eu
  - Derivation of sec. requirements
  - Report generation
  - Multi-platform (Win, Linux, Mac)
The STS-ml Security Requirements Document

- STS-Tool **automatically generates a security requirements document**, which includes
  - Social, Information, and Authorization views
  - Automatically generated textual description of the models
  - Tables listing the main concepts
  - Security requirements

- How to produce a high quality document?
  - Use descriptive fields in STS-Tool

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Travel Agency Service is a composite service that uses a service from Amadeus reservation</td>
</tr>
<tr>
<td>Mission</td>
<td>Allow the end user to read about various destinations and book flights’ tickets</td>
</tr>
<tr>
<td>Purpose</td>
<td></td>
</tr>
</tbody>
</table>
In this section are described for each role/agent, the authorisations it passes to others and what authorisations it receives from other roles/agents.

In the Report_example project (Figure 3) the authorisations for each role/agent are:

- **Agent Amadeus Service:**
  - is authorised by TAS to use and produce information Itinerary and Personal data, in the scope of goal Flight ticket booked, without having the right to further authorising other actors.

- **Role Tourist:**
  - authorises TAS to use information Personal data and Itinerary, in the scope of goal Hotel booked, passing the right to further authorising other actors, and authorises information Personal data, in the scope of goal Flight ticket booked, passing the right to further authorising other actors.

- **Role TAS:**
  - authorises Amadeus Service to use and produce information Itinerary and Personal data, in the scope of goal Flight ticket booked, without passing the right to further authorising other actors.

### Security Requirements

<table>
<thead>
<tr>
<th>Responsible</th>
<th>Security Requirement</th>
<th>Requester</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAS</td>
<td>no-delegation (Flight ticket booked)</td>
<td>TAS</td>
<td>Amadeus Service requires no-delegation for goal Flight ticket booked, when delegating Flight ticket booked to Amadeus Service.</td>
</tr>
<tr>
<td>Amadeus Service</td>
<td>dual-repudiation (delegation(TAS,Amadeus Service,Flight ticket booked))</td>
<td>TAS</td>
<td>Amadeus Service requires dual-repudiation for goal Flight ticket booked, when delegating Flight ticket booked to Amadeus Service.</td>
</tr>
<tr>
<td>TAS</td>
<td>non-modification (Itinerary,Personal data)</td>
<td>TAS</td>
<td>TAS requires Amadeus Service non-modification of Information Itinerary and Personal data.</td>
</tr>
<tr>
<td>TAS</td>
<td>non-disclosure (Itinerary,Personal data)</td>
<td>TAS</td>
<td>TAS requires Amadeus Service non-disclosure of Information Itinerary and Personal data.</td>
</tr>
<tr>
<td>TAS</td>
<td>need-to-know (Itinerary,Personal data)</td>
<td>TAS</td>
<td>TAS requires Amadeus Service need-to-know of Information Itinerary and Personal data, in the scope of goal Flight ticket booked.</td>
</tr>
<tr>
<td>Tourist</td>
<td>single-actor-time-redundancy (Tickets booked)</td>
<td>Tourist</td>
<td>TAS requires single-actor-time-redundancy for goal Tickets booked, when delegating Tickets booked to TAS.</td>
</tr>
<tr>
<td>Tourist</td>
<td>dual-repudiation (delegation(Tourist,TAS,Tickets booked))</td>
<td>Tourist</td>
<td>TAS requires dual-repudiation for goal Tickets booked, when delegating Tickets booked to TAS.</td>
</tr>
<tr>
<td>Tourist</td>
<td>non-modification (Personal data,Itinerary)</td>
<td>Tourist</td>
<td>Tourist requires TAS non-modification of Information Personal data and Itinerary.</td>
</tr>
<tr>
<td>Tourist</td>
<td>non-production (Personal data,Itinerary)</td>
<td>Tourist</td>
<td>Tourist requires TAS non-production of Information Personal data and Itinerary.</td>
</tr>
</tbody>
</table>
Beyond the STS-ml method

- Each participant is developed independently (due to autonomy)
- Each participant is concerned with the security requirements related to the role it plays
- Example: the implementation of a TAS has to ensure that
  - The delegation of tickets booked will not be repudiated
  - Personal data will not be modified
  - Personal data will be used only for booking the trip
  - The itinerary will not be further redistributed
Commitments at runtime

- Commitments are not only a primitive that captures the contractual nature of requirements at design-time.
- They **arise and evolve due to the exchanged messages** between debtor and creditor at runtime.
  - Messages are interpreted in terms of creation, fulfilment, and violation of commitments (i.e., here, security requirements).
  - Thus, they can be **monitored to check compliance**.
Conclusions

- On Security Requirements Engineering
  - The importance of security requirements is widely agreed
  - Existing approaches are not well suited for STSs

- SRE in the era of Socio-Technical Systems
  - **Security cannot be guaranteed in a centralized fashion**
    - Due to the autonomy of participants and their interactions
  - Security requirements constrain the interaction among participants
  - STS-ml is an SRE method expressly suited for STSs
Acknowledgements

The STS team @Univ. Trento

- Prof. Paolo Giorgini
- Elda Paja
- Dr. Pierluigi Roberti
- Mauro Poggianella

Valuable suggestions concerning this tutorial from John Mylopoulos, Eric Yu, and Arnon Sturm

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant no 257930 (Aniketos)
Thanks!

Contact: dalpiaz@cs.toronto.edu

More? See you at the demo track at 3.30pm
Bibliography


Bibliography


