Trust4All: trustworthy embedded systems

Dr.ir. Henk Eertink
Telematica Instituut
Henk.Eertink@telin.nl

IPA Herfstdagen; 2005/11/23

Thanks to: Gabriele Lenzini, Hans Zandbelt, Andrew Tokmakoff, Cristian Hesselman
Agenda

- Trust4All objectives
- Example scenario
- Trust management architecture
- Trust modelling
Trust4All motivation

- Update & tailor products during their lifetime
  - Extends the lifetime
  - Increased business agility (changing requirements)

→ embedded systems become open systems

This enables
- Integration of 3rd party software components
- a new market for embedded applications
Trust4all objective

The Trust4All aims to develop a middleware software architecture for embedded systems, to maintain correct operation while the software is being upgraded and extended.

Belief: correct operation of a system can be guaranteed if the system is built from trusted components.
Agenda

- Trust4All objectives
- Example scenario
- Trust management architecture
- Trust modelling
Scenario- overview

- Medical Application requires device re-configuration when attempting to join a “secure” networked environment.
- Medical Applications can be “pushed” onto devices by medical practitioner.
Scenario-2

• Bob wears a number of medical sensing instruments that sense and report some of his vital signs to his mobile device.
• The mobile device is able to receive data from the sensors and present Bob with both aggregate and statistical information regarding his heart rate, blood sugar levels etc.
Scenario-3

- When Bob visits his medical specialist (located at the hospital) for a check-up, there is sometimes a need for a software update to the Mobile Device.
- In other situations, Bob’s medical specialist wishes to access some of the Raw Data that Bob’s sensors have accumulated, so that he can send it off for further analysis.
- In both situations, Bob needs to “connect” to the trusted network of the hospital.
- In order to do so, Bob’s mobile device is required to conform to a number of requirements that have been set in place to ensure the “trustworthiness” of the network.
Scenario-4

The first requirement is that Bob’s mobile device operates with a Trust4All environment. This is to enable configuration of the runtime components on his device. In particular, before being granted access to the hospital network, Bob’s device must have the following:

- A TCP/IP stack component which has been “certified” by a medical systems auditor.
- A set of certified device drivers (Services) for Bob’s medical sensors.
- A certified medical application which can:
  - Make use of the TCP/IP stack to communicate with a medical systems back-end server to perform component upgrades and data downloads.
  - Make use of the device drivers to interact with Bob’s medical sensors.
Scenario-5

- Additionally, Bob’s device must ONLY have these components operational at the time that he wishes to join the hospital network.
- The Trust4All runtime takes care of this, ensuring that all other components are disabled at that time.
- It also ensures that the certified medical application can store its data in a way that is both confidential and integral, potentially using a “secure” storage component.
Scenario-new functionality

• Unfortunately for Bob, his specialist believes that he needs to add a new sensor to Bob’s medical sensor network. This sensor is responsible for monitoring and reporting on Bob’s cholesterol level.

• The specialist installs the sensor on Bob and activates it. At this point, Bob’s mobile device “discovers” the new sensor and realizes that it requires a new Driver Service for the sensor.

• This Driver Service is acquired (potentially from the sensor itself using technology such as Jini) and is installed on the mobile device, as a Trust4All service.
Scenario-7

- Once the data transfer (which is application-level interaction) and component upgrade of the application has occurred, Bob can disconnect from the hospital network.

- At that point, the Trust 4All environment is free to re-start other components as it sees fit, as the constraints imposed by the hospital network are no longer in force.
Agenda

- Trust4All objectives
- Example scenario
  Trust management architecture
- Trust modeling
  - Informal trust definition
  - Functions for defining trust
Trust4All challenge

Open system
Arbitrary Feature addition
Failed system

Trusted
Open system

Trusted
Feature addition

Failed system

Trusted system
High-level (device) architecture

- applications
- resource framework
- fault mgmt framework
- trust framework
- middleware layer
- platform layer (operating systems + device drivers)
- hardware
- runtime
- infrastructure
Trust Architecture

- Extension of the middleware layer
- Trust Framework
  - Trust Calculations for single components and compositions
  - Decision Making analyzing risks and trustworthiness guide by policies and decide on logical actions
- Runtime level
  - Sensors, meters -> monitoring -> events
  - Implementation of control actions
Functional View of Trust Framework

Decision Maker

Actuator

vector of logical actions

T4A runtime and operating system

requests

events

application component

framework component

actuations

events
Agenda

- Trust4All objectives
- Example scenario
- Trust management architecture
  - Informal trust definition
  - Trust reasoning framework
Trust - Properties

- Trust is relative to a given context
- Trust is directed
- Trust is a measurable belief
- Trust exists in time
- Trust evolves over time
- Trust is transferable
Trust -- definition

Trust is the extent to which a **trustor** is willing to depend on a **trustee** to act **dependably** and **securely** in a given situation, with a feeling of relative security even though negative consequences are possible.

Abstract Trust Framework

- A system $S$ wants to compose a service $s$ by using a subset Components of atomic components.

- An atomic component represents a basic functionality whose qualities are described by a set Attributes of attributes.

- Both judgment and recommendation are used to evaluate trustworthiness and trust.

- Recommendations comes from a set $\Phi(s)$ of recommenders chosen depending on the service.

\[ \text{Attributes} \rightarrow \text{Components} \]

\[ \text{system } S \rightarrow \text{history} \rightarrow \text{service } s \rightarrow \text{Attributes} \]

\[ TValues \ni \{0,1,\bot\} \]

\[ \text{Recommenders} = \Phi(s) \]
Abstract Trust Function

It contains concepts of measurement, time, reputation, and recommendation:

\[ rht\text{trust}_{S,s}(p) := rht\text{trust}_{S,s}(p,i,rht\text{trust}_{S,s}^{i-1}(p),\text{Recommendations}^{i}_{\phi - \{S\},s}(p)) \]

Given a component \( p \) (i.e., its attributes), function \( rht\text{trust} \) evaluated at time \( i > 0 \) can be calculated in following way:

Here:

\[ \text{Recommendations}^{i}_{S,s}(p) := \{rht\text{trust}^{i-1}_{x}(p), x \in \Phi(s) \} \]
Trust reasoning framework

Single

Composed

Dynamict

Static

Dynamic

recommendations

weighted recommendations

direct trust

Static

Dynamic

Static

Dynamic

history dependence

dynamicity

history dependence

dynamicity

history dependence

dynamicity

history dependence

dynamicity

IPA herfstdagen, Nov 23, 2005
Status & future work

• Trust4All started in summer 2005 (3 months ago) 2 year ITEA project
• Dutch consortium sponsored by SenterNovem
• Current status:
  – Reasoning framework about trust is stabilizing
    • submitted paper for iTrust2006
    • First implementation in CAML
  – Trust management architecture under definition
• Next steps:
  – Evaluation functions (how to compute trustworthiness)
  – Trust language (how can applications express trust reqs)
  – Composition (system vs component level)
  – Integration in existing low-footprint embedded middleware systems
  – Development of demonstrators (industrial partners)
Questions?

Discussion!