Towards Self-Adaptive Security for Pervasive Computing Architectures
Methodology and Algorithms
For Securing Networked Self-Adaptive Embedded Systems

Antonio Vincenzo Taddeo

Advanced Learning and Research Institute (ALaRI)
Faculty of Informatics, University of Lugano
Lugano, 6900, Switzerland
antonio.taddeo@usi.ch

PhD-Proposal Talk, Lugano, September 14, 2010
About My Thesis

Collaborators

- **Research Advisor:** Prof. Mariagiovanna Sami
- **Academic Advisor:** Prof. Laura Pozzi
- **Co-Researcher:** Dr. Alberto Ferrante

*European project AETHER (FP6 EU: IST-2006-27611)*

Self-adaptive Embedded Technology for Pervasive Computing Architecture

*Hasler Foundation project aetherplus (GRANT-09083-2043-021)*

Enhancing Self-adaptive Approaches to Security
Dissertation Goal

Enabling self-adaptive security in Pervasive Systems by means of new design principles and algorithms

My Challenges

- selecting the most suitable security solutions for a given operating environment
- considering constraints in available resources and application requirements
- runtime trade-off between security and performances
Why self-adaptive security?
Context and Motivation

Networked and Distributed Self-Adaptive Embedded Systems (NSAS)

- resource-constrained and mobile nodes
- power consumption is the most important constraint
- communication security is required
- nodes may support HW/SW self-adaptation
- nodes can be heterogeneous both in HW and services provided
Security design is a challenge

- influences both device design and performances
- requires relevant computational capabilities and energy consumption
Traditional Security Solutions

**Typical security solutions**

- ad-hoc approaches for specific protections
- not specifically designed for pervasive systems
- assume that the environment is well-known and static
- worst-case scenario considered
  - enable the strongest security solution
  - power-hungry solution
State of the Art in Adaptive Security

Related Work

- Salehie and Tahvildari [2009]
  - Self-adaptive software solutions analysis
- Dandalis and Prasanna [2004]
  - Adapt security “on the fly” based on IPSec parameters
- Chandramouli et al. [2006]
  - Knapsack Problem for optimizing power consumption
- El-Hennawy et al. [2004]
  - Change the key-length according to a confidentiality level
What is adapted?
Self-adaptive Security: System Model Overview

Derin et al. [2009]; Ferrante et al. [2007]
Security Self-Adaptation Overview

Levels of adaptations

- **Node level**
  - Gradual adaptation of security (*GAS*)
  - Coordinated management of QoS and security (*CMQS*)

- **Network level**
  - Negotiation of security services protocol (*NSSP*)
  - Trusting protocol (*TP*)

My solution provides an integrated approach to self-adaptive security
How is adaptation performed?
Node-level adaptation
**GAS: Gradual adaptation of security**

**Principle**
- Maximization of security and system workload by means of **gradual adaptation**
- “Best Effort” approach: strongest security first
- Security is self-adapted to a given **context** by means of adaptation policies

**Innovation**
- Gradual degradation/upgrade of security
- High flexible security/workload adaptation
- Energy consumption monitoring

Taddeo et al. [2010a]
GAS Adaptation Process

Energy Context

Trigger Conditions

Upper Threshold

Lower Threshold

Battery Capacity

P1

P2

P3

ΔE

Task Degradation Policy

Optimization Policy

Workload Policy

Policies

Controller

Monitor

Adapter

Adaptation Trigger

Running Tasks

Sec. Req.

Monitoring Period t
**CMQS : Coordinated management of QoS and security**

**Principle**
- Dynamic adaptation of security for each packets
- Maximization of security and # of packets sent
- QoS: increase the delivery of high priority packets

**Innovation (similar as before, plus . . . )**
- Energy-aware and QoS-aware security
- Reduced energy consumption per byte

Taddeo and Ferrante [2010]; Taddeo et al. [2010b]
CMQS Adaptation Process

Energy Context

Trigger Conditions

Upper Threshold

Lower Threshold

P1

P2

P3

ΔE

Priority-based packets queue

Sec. Req.

Controller

Monitor

Adapter

Packets Policies

Optimization Policies

Policies
Network-level adaptation
NSSP: Negotiation of Security Services Protocol

Principle

- Run-time negotiation of security solutions between heterogeneous nodes
- Optimization of security utility and energy consumption for both nodes

Innovation

- Use of multiple security suites
- New security utility computation by AHP
- LPP solution based on nodes operating conditions

Taddeo and Ferrante [2009a,b]; Taddeo et al. [2009]
NSSP Adaptation Process

Linear Programming Problem (LPP)

Objective Function:

\[ U = \text{Max} \sum_{i=1}^{n}(U^A_i + U^B_i)x_i \]

subject to constraints:

- \[ \sum_{i=1}^{n} x_i \leq T \]
- \[ \sum_{i=1}^{n} e^A_i x_i \leq E^A_t \]
- \[ \sum_{i=1}^{n} e^B_i x_i \leq E^B_t \]
TP: Trusting protocol

Principle

- Dynamic verification of nodes trustworthiness
- Reputation-based protocol
- Protection mechanism against security attacks

Innovation

- Lightweight solution, high protection
- Decentralized approach
- High customizable trust value function

Ferrante et al. [2008]
**TP Process**

**Trust Evaluation**

- by using direct and indirect experience

\[ T^d_j = w_p \times V^d_j + w_i \times \sum_i V^i_j \]

---

**A. V. Taddeo, ALaRI, University of Lugano**

Proposal Talk—Towards Self-Adaptive Security, 17/19
Where are the advantages?
Evaluating My Solution

Evaluation Methodology

- By mean of simulations
  \((\text{SystemC and } SC^2)\)
- Experiments on real hardware
  \((\text{Sun SPOT})\)
Evaluating My Solution

Evaluation Methodology

- By mean of simulations (*SystemC* and *SC²*)
- Experiments on real hardware (*Sun SPOT*)

Results

- Optimized security for a given operating environment
- Balanced trade-off between performances and security
- Energy consumption saving
- Enhanced QoS management for high priority tasks
Conclusion & Future Work

Conclusion

- Self-adaptive security is feasible
- A new methodology, protocols and algorithms for enabling it have been proposed
- Different adaptation perspectives have been analyzed:
  - Node and Network level
- Solution effectiveness has been tested by simulations and real prototype implementation
Conclusion & Future Work

Conclusion

- Self-adaptive security is feasible
- A new methodology, protocols and algorithms for enabling it have been proposed
- Different adaptation prospectives have been analyzed:
  - Node and Network level
- Solution effectiveness has been tested by simulations and real prototype implementation

Future Work

- Analysis of system security vulnerability
- Enhancing the model of adaptation policies
Appendix
## Index

1. Introduction
2. Related Work
   - 2.1 Self-adaptivity
   - 2.2 Security adaptation
   - 2.3 Approaches comparison
3. Self-adaptive Security Model
4. Node-Level Adaptation
   - 4.1 GAS
   - 4.2 CMQS
5. Network-Level Adaptation
   - 5.1 NSSP
   - 5.2 TP
6. Case Study
   - 6.1 GAS
   - 6.2 CMQS
   - 6.3 NSSP
   - 6.4 TP
7. Conclusion
GAS Results

![Graph showing GAS Results]

- Running Applications [N]
- Charge [%]
- Iteration

- Policy 1
- Policy 2
- Policy 3

- Energy consumption
NSSP Results

Energy vs Byte

- $b_1$
- $b_2$
- $a_1$
- $a_2$
- $\chi_{\text{tot}}$

A. V. Taddeo, ALaRI, University of Lugano

Proposal Talk—Towards Self-Adaptive Security,
TP Results

![Graph showing the relationship between faultiness of SANE and failure rate of task delegation. The graph includes three lines representing different levels of trust: direct + indirect trust, direct trust, and no trust.](image)

- **x-axis**: Faultiness of the SANE (%)
- **y-axis**: Failure rate (%) of task delegation

- **Legend**:
  - Direct + indirect trust
  - Direct trust
  - No trust
Analytic Hierarchy Process (AHP)

\[ O^1 = \begin{bmatrix} 1 & 3 \\ 3^{-1} & 1 \end{bmatrix} \rightarrow o^1 = \begin{bmatrix} 0.75 \\ 0.25 \end{bmatrix} \rightarrow \bar{o}^1 = \begin{bmatrix} 1 \\ 0.33 \end{bmatrix} \]

\[ O^3_{sec} = \begin{bmatrix} 1 & 7 & 3^{-1} \\ 7^{-1} & 1 & 9^{-1} \\ 3 & 9 & 1 \end{bmatrix} \rightarrow o^3_{sec} = \begin{bmatrix} 0.2897 \\ 0.0549 \\ 0.6553 \end{bmatrix} \]

\[ o_{global} = [o^3_{sec} ; o^3_{syscost}] \cdot o^1 \]

AHP How-to

- build the hierarchy;
- construct the comparison matrix \( O \) for each level;
- compute its priority vector \( o \);
- synthesis of the global ranking of different alternatives;

A. V. Taddeo, ALaRI, University of Lugano
Proposal Talk—Towards Self-Adaptive Security,


References II


