GUIDING THE SELECTION OF SECURITY PATTERNS FOR REAL-TIME SYSTEMS

Anas Motii & Brahim HAMID
Institut de Recherche en Informatique de Toulouse (IRIT)
Université de Toulouse Jean Jaurès, France
Motii,hamid@irit.fr
Context of our work

- **SEMCO**: we promote a framework to support PBSE in the domain of assistance to the secure and dependable resource constrained system engineering.

  - **TERESA (FP7-EU)**: To demonstrate and validate an engineering discipline for trust (S&D) that is adapted to Resource Constrained Embedded Systems (RCES)

- **SIRSEC (FUI7 – FR)**: To develop process, technologies and tools for the safety of information exchange

- **This work was also performed within the framework of the Doctoral thesis of Anas MOTII (defense in June 2017), which is co-directed with Pr. Jean-Michel Bruel (IRIT, University of Toulouse Jean-Jaurès), Dr. Agnes Lanusse (CEA LIST) and Brahim HAMID (IRIT, University of Toulouse Jean-Jaurès)**
Agenda

- Motivations and background
- Modeling of secure applications for real-time systems
- Model-driven development
- Tool support
- Case study
- Conclusion and perspectives
Agenda

- Motivations and background
- Modeling of secure applications for real-time systems
- Model-driven development
- Tool support
- Case study
- Conclusion and perspectives
SCADA (Supervisory Control and Data Acquisitions systems) systems are meant to **control processes** and **acquire data**.

- The integrity and confidentiality of these data are essential
  - Critical data and infrastructures: chemical industry
  - The corruption of data can lead to catastrophic damages

**Example:** water treatment alteration in March 2016

- Real-time SCADA systems must meet communication network deadlines (IEEE 1646 and IEC 61850 standards)
  - Evaluating security solution alternatives is needed
Our approach through an example: The Stakeholders

Patterns are effective tools for the reuse of specific knowledge

Patterns realize security policy by capturing best practices

Patterns are a systematic approach to describe best practices

Patterns can describe standards and regulations in a precise way and make them more understandable and usable
Examples of security patterns

Context:
The system delivers functionalities and information to others. Application must exchange data with others. At least a percentage of this data will be sensitive in nature.

Problem:
How do we ensure the security of data in transit?

Solution:
Create secure channels for sensitive data. Exchange information to allow to set up encrypted communication. Reduce associated overhead by using ordinary communication channels for non-sensitive data.

Design pattern notion was introduced by Christopher Alexander – an architect (1977) for the domain of urban architecture.
Background (Pattern-based approaches)
Contributions

- In this work, we go one step further, we study the impact of implementation alternatives of these security solutions onto the system architecture.
  - A special emphasis is paid to timing performance concerns using model-based real-time evaluations
- The approach is defined in the form of a set of DSMLs and model transformation rules
- MDE Tool-Chain
  - Support for representing, storing and retrieving expertise for reuse, analysis and generation
Agenda

- Motivations and background
- Modeling of secure applications for real-time systems
- Model-driven development
- Tool support
- Case study
- Conclusion and perspectives
Proposed approach

**Step 1**: The architect provides the functional architecture specification model of the real-time system together with a security pattern configuration model, including the bindings,

**Step 2**: The architect executes the pattern configuration integration,

**Step 3**: Create a task model based on the functional architecture, the platform specification and the mappings between them,

**Step 4**: Analyze the schedulability of the resulted model,

**Step 5**: The result of the analysis is then delivered for evaluation,
Architecture of the complete system of patterns

Architecture of the initial system of patterns
Pattern configuration integration

Algorithm: Integrate Pattern Configuration

Input: $Ma$, $C$, $B$

Output: $Mac$

for each Pattern in the $C$ do
  for each $bi$ in $B$ do
    if $bi$.patternFunction.pattern = Pattern
      substitute ($Mac$, $bi$.applicationFunction, $bi$.patternFunction)
      save()
    endif
  endfor
endfor

Diagram:
- Alternative 2
  - P3
  - P4
- Alternative 3
  - P5
  - P6
- Alternative 4
  - F1
  - F2
  - F3
  - F4

Security solution alternative 1

Functional architecture
Real-time evaluation of pattern configurations

Algorithm EvaluateArchitectureConfiguration
    Input : T
    Output : M
    for each TMac in T do
        schedulabilityAnalysis(T)
        if T.isSchedulable
            M = M U TMac.Mac
            save()
        endif
    endfor

- Preliminary evaluation
  - Rate Monotonic scheduling (RMS) to check if a node is overloaded
- Response time analysis
  - offset-based scheduling based on the worst case response time to check that all response times must verify $R_{Task} \leq D_{Task}$
Agenda

- Motivations and background
- Modeling of secure applications for real-time systems
- Model-driven development
- Tool support
- Case study
- Conclusion and perspectives
Model-Driven Engineering

- Model-Driven Development
  - Raise the abstraction level (metamodeling)
  - Perform analysis at earlier phases of development (model transformations)
  - System and software artifacts generation (model transformations)

SEPM (System and software Engineering Pattern Metamodel)

M2M transformations:
- Pattern instantiation
- Pattern integration

Modeling system architecture

Modeling security pattern alternatives
Example of SSL

The SSL pattern has the following functions:

- **Fp1**: Verify server certificate
- **Fp2**: Key exchange
- **Fp3**: Encryption
- **Fp4**: Hashing
- **Fp5**: Forwarding data
- **Fp6**: Integrity checking
- **Fp7**: Decryption
## UML + MARTE + SEPM

<table>
<thead>
<tr>
<th>MARTE Stereotype</th>
<th>UML extension</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Functional architecture stereotypes</strong></td>
<td></td>
</tr>
<tr>
<td>GQAM::GaWorkloadBehavior</td>
<td>Activity</td>
</tr>
<tr>
<td>GQAM::GaWorkloadEvent</td>
<td>AcceptEventAction</td>
</tr>
<tr>
<td>SAM::SaEndToEndFlow</td>
<td>ActivityPartition</td>
</tr>
<tr>
<td>SAM::SaStep</td>
<td>CallActionBehavior</td>
</tr>
<tr>
<td>Alloc::Allocate</td>
<td>Abstraction</td>
</tr>
<tr>
<td>Alloc::Allocated</td>
<td>CallAction, Property</td>
</tr>
<tr>
<td><strong>Platform Stereotypes</strong></td>
<td></td>
</tr>
<tr>
<td>GQAM::GaPlatformResources</td>
<td>Class</td>
</tr>
<tr>
<td>SAM::SaExecHost</td>
<td>Property</td>
</tr>
<tr>
<td>SAM::SaCommHost</td>
<td>Connector</td>
</tr>
<tr>
<td>SAM::SaSharedResource</td>
<td>Property</td>
</tr>
<tr>
<td>GRM::SchedulableResource</td>
<td>Property</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source</th>
<th>UML</th>
<th>MARTE annotations and types</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEPM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SepmPattern</td>
<td>Package</td>
<td>N/A</td>
<td>This package is the main container</td>
</tr>
<tr>
<td>Context</td>
<td>Comment</td>
<td>N/A</td>
<td>The textual description is put in this comment inside the UML package</td>
</tr>
<tr>
<td>Problem</td>
<td>Comment</td>
<td>N/A</td>
<td>—</td>
</tr>
<tr>
<td>SepmInternalStructure</td>
<td>Class</td>
<td>N/A</td>
<td>The class describes the functional architecture of the pattern and contains functions</td>
</tr>
<tr>
<td>Function</td>
<td>Property</td>
<td>SAM::SaStep</td>
<td>The functions are put inside the container class. Each SEPM::Function has a S&amp;D property and/or a resource property</td>
</tr>
<tr>
<td>ResourceProperty</td>
<td>Class</td>
<td>NFP_Real</td>
<td>This mapping depends on the resource property category and thus the target is one of the sub types of NFP_Real. E.g., “CPUTime” property is mapped to NFP_Duration</td>
</tr>
</tbody>
</table>
Agenda

- Motivations and background
- Modeling of secure applications for real-time systems
- Model-driven development
- Tool support
- Case study
- Conclusion and perspectives
Tool features

- Repository life cycle
  - Set-up
  - Organisation
- Security patterns life cycle
  - Definition
  - Deposit artefact in repository
- System life cycle
  - System modeling
  - Security requirement definition
  - Retrieve pattern from repository
  - Build configurations
  - Establish bindings
  - Integrate artefacts
  - Analysis (real-time, risk, …)
Tool implementation

Modeling system architecture

M2M transformations (QVT)

Real-time analysis (QOMPASS)

Modeling security pattern alternatives

Papyrus System Architect

SEMCOMDT

Papyrus UML
Agenda

- Motivations and background
- Modeling of secure applications for real-time systems
- Model-driven development
- Tool support
- Case study
- Conclusion and perspectives
Security requirements

In our case we focus of the following requirements:

- **Req1.** There should be mechanism for secure communication that guarantees data integrity, confidentiality and authenticity.
- **Req2.** There should be a mechanism that protects against denial of service attacks at the level of the SCADA master.
Identification of security solution alternatives

- The search in the semco model repository leads to two abstract patterns refined by several concrete patterns:
  - Secure communication pattern
  - Firewall pattern
Identification of security solution alternatives

- The pattern system leads to the generation of four security solution alternatives
SCADA system functional architecture
Secure communication pattern – functional architecture
Secure communication pattern – functional architecture

- SSL pattern is applied on the SCADA architecture to secure the messages transferred between “setPoint_update” and “actuator_cmd_computation” functions. These functions play respectively the roles “send_data” and “receive_data” in the SSL pattern.
security patterns timing parameters and deployment*

<table>
<thead>
<tr>
<th>Functions</th>
<th>Execution time</th>
<th>Task</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setpoint Processing</td>
<td>8.7</td>
<td>$\tau_1$</td>
<td>SCADA master</td>
</tr>
<tr>
<td>Poll Data</td>
<td>9.6</td>
<td>$\tau_1, \tau_6, \tau_7$</td>
<td>SCADA master</td>
</tr>
<tr>
<td>Log Data</td>
<td>8.5</td>
<td>$\tau_5$</td>
<td>SCADA master</td>
</tr>
<tr>
<td>Check Status</td>
<td>9.6</td>
<td>$\tau_6$</td>
<td>SCADA master</td>
</tr>
<tr>
<td>Visualize Data</td>
<td>10.5</td>
<td>$\tau_5$</td>
<td>SCADA master</td>
</tr>
<tr>
<td>Alarm Handler</td>
<td>10.3</td>
<td>$\tau_6$</td>
<td>SCADA master</td>
</tr>
<tr>
<td>Archive Data</td>
<td>9.5</td>
<td>$\tau_7$</td>
<td>SCADA master</td>
</tr>
<tr>
<td>Command Computation</td>
<td>10</td>
<td>$\tau_2, \tau_3, \tau_4$</td>
<td>PLC</td>
</tr>
<tr>
<td>Data Preprocessing</td>
<td>9.5</td>
<td>$\tau_3$</td>
<td>PLC</td>
</tr>
<tr>
<td>Diagnosis</td>
<td>8.9</td>
<td>$\tau_4$</td>
<td>PLC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Patterns</th>
<th>Functions</th>
<th>Execution time</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>SecureCommSSL</td>
<td>Authentication</td>
<td>9.7</td>
<td>$\tau_1$</td>
</tr>
<tr>
<td>(1) SecureCommSSL</td>
<td>Key exchange and encryption</td>
<td>10.1</td>
<td>$\tau_1$</td>
</tr>
<tr>
<td>(2) SecureCommSSL</td>
<td>HMAC</td>
<td>9.9</td>
<td>$\tau_1$</td>
</tr>
<tr>
<td>(2) SecureCommSSL</td>
<td>Decryption</td>
<td>9.2</td>
<td>$\tau_1$</td>
</tr>
<tr>
<td>(2) SecureCommSSL</td>
<td>Integrity checking</td>
<td>10.3</td>
<td>$\tau_2$</td>
</tr>
<tr>
<td>(2) SecureCommSSL</td>
<td>Filtering</td>
<td>10.2</td>
<td>$\tau_2$</td>
</tr>
<tr>
<td>(2) SecureCommSSL</td>
<td>Packet Filtering</td>
<td>10</td>
<td>$\tau_7$</td>
</tr>
</tbody>
</table>

* IEEE 1646 standard and IEC 61850
Schedulability analysis for the alternatives

Preliminary evaluation

Utilization bounds

- SCADA master utilization exceeds the threshold (75.68%)
  - Alternative 2: 83.33%
  - Alternative 4: 103.33%
- Alternatives 2 and 4 are rejected!

Response time analysis

- Alternative 3:
  - Task 2 response time: 280ms
  - Task 2 deadline: 248ms
- Alternative 3 is rejected!
- Alternative 1 is valid
Agenda

- Motivations and background
- Modeling of secure applications for real-time systems
- Model-driven development
- Tool support
- Case study
- Conclusion and perspectives
Conclusion

- The pattern-based security methodology seems to be a very interesting combination of two flavors of security and performance design which has not been addressed sufficiently in literature.

Contribution

- The evaluation of possible security solution alternatives fulfilling real-time requirements
- Model-based approach
- Tool support

Next

- The architecture description needs to be extended with other type of elements, e.g., software components, connectors and deployments units
- This work can make use of optimization techniques during task generation [*]
- We are studying more sophisticated techniques for system of patterns building, including more sophisticated techniques to derive artifacts relationships

Thank you for your attention

Questions?