Design and Verification of Pervasive System Security

1S. Geetha, 2Dr. J. Madhusudanan, 3Dr. V. Prasanna Venkatesan, 4Dr. A. Martin

1Department of Banking Technology, Pondicherry University, Puducherry, INDIA.
2Department of Computer Science and Engineering, Sri Manakula Vinayagar Engineering College, Puducherry, INDIA.
3Department of Banking Technology, Pondicherry University, Puducherry, INDIA.
4Department of Master of Computer Applications, Sri Manakula Vinayagar Engineering College, Puducherry, INDIA.

ABSTRACT

The vision of Pervasive Computing is that every device we communicate today has become a part of everyday objects, augmented with information services. Though there are a number of frameworks is been developed for security of smart systems, there is still space for research in the field of security of these systems. User privacy and Trust needs more concentration in a pervasive environment. Context-awareness, Autonomy and Intelligence are the important features of pervasive computing environment. These systems face more issues with respect to aspects like security and mobility of devices. The context based security policies are more important for the pervasive environment. To make the pervasive systems more secure the security policies provided should be adaptive to the various context available in a single entity/context. This paper is one approach to develop security policies according to the changing context for the pervasive systems.

INTRODUCTION

The Pervasive system need more focus on security concerns and requirements. Addressing the security issues in the smart systems is a major challenge in today’s research field. The pervasive systems that is said to be smart is not been provided with smart security policies. They smartly identify the context of the environment and act accordingly but they cannot locate the threats near to it smartly. Due to the lack of smartness in the security policies the pervasive systems are critical to various threats in the computing environment. Trust is yet another aspect of security in pervasive systems. The communication between various devices in the pervasive environment and their context needs trust to share the information among them. How to trust the devices in the pervasive environment? It is a big question for the pervasive system users. To trust the nearby devices for communication these systems should smartly identify their security levels thereby avoiding the risk of threats. Intelligent behavior of the pervasive systems helps to identify and differentiate the various devices available in the pervasive environment. But the intelligence provided is not up to the level of the needed features.

Based on Eugenia I. Papagiannakopoulou et al (2012), Authentication and authorization of the devices in the environment is not with a fixed infrastructure. As a result it is been prone to malicious snoopers. While discovering the nearby devices in the environment, if any malicious devices is been added it leads to a dangerous situation. Thus intelligence is more important for a pervasive system. Thus a pervasive system with context-awareness, autonomy for making decisions, intelligence and trust to provide security for these systems is needed. This paper is an approach for including security policies for the various context of the pervasive system along with security policy for context as a context for the various available contexts in the pervasive environment.

2. Literature Survey:

According to Keara Barrett (2004), it is said that the traditional security policies of pervasive systems does not address the features like invisibility, heterogeneity, accessibility, etc., Manually adjusting the security policies according the context is not possible. The introduction of context based security policies for pervasive systems will provide more security to these systems. Ruan He, et al (2010), have stated that autonomic
management strategies are to be specified using domain-specific languages but using an objective or function oriented policies will be more desirable for higher-level strategies.

In the journal of Context-aware Security, Konrad Wrona and Laurent Gomez (2005), it is been discussed that context-aware systems face more security challenges in respect to trusting the context. The authors have highlighted that if context information are used in mobile applications applying security policies is more challenging in a pervasive environment. According to Jason Cornwell, et al (2006), the demand for increasing flexibility grows in a pervasive environment it essential to develop a security policy that is understandable for the user’s to be used at various contexts accordingly.

More research is been done to get the overall security of the pervasive systems. Although security is been implemented in the pervasive environment they are implemented into the already existing pervasive computing architectures, so there is a lack in the security measures and they are specific for each applications. According to the analysis done on the existing smart system it is clear that the security policies provided is not to the extent that it can act smartly under critical situations. Due to this lack in security features they become vulnerable to various threats that arise in the pervasive environment. Thus a security model that addresses the needs of security policies in the various context of the pervasive system required for its success in the current computing world.

3. Proposed System:

The proposed system as shown in the figure 1, provides the security policies for different context which includes context itself as one of the context in the pervasive environment that provide better security for the pervasive system. The objective of the proposed work is to develop a new model for security policies that addresses the various issues in the pervasive environment.

The proposed architecture consists of Application layer, Context Layer, Security Layer and Physical Layer.

![Fig. 1: Proposed Architecture.](image)

**Context Manager:**

The context manager gets the various contexts from the application layer and sends it to the context handler to provide the needed context services.

**Context Handler:**

The context handler classifies the context received from the context manager and checks with the analyzer to provide which type of security policy for the existing scenario. If there is any conflict in the context and security policies then the handler will handle it with necessary steps or policy changes to be carried out.

**Context Analyzer:**

The Context Analyzer will analyse the security policies to provide a better policies for the context and will identify which security policy to be given for the existing scenario.
Context Policy Resolver:
The Context policy resolver will identify the situation of the context and matches the correct security policy needed for the context from the security policy builder.

Security Policy Builder:
The Security policy builder will build the various security policies for the context. The policies will be generated according to the context of the environment at different scenarios.

Security Policy Validator:
The Policy validator will validate the security policies build by the policy builder by identifying the proper policies for the context available. The validator will check for the policies correctness that is whether the policy build for the context is an appropriate one or not.

4. Architecture Verification Using CPN Model:
CPN model for proposed architecture is developed and shown in Figure 2. In the proposed Architecture different contexts are congregated and formed as scenarios. According to the scenario, new security policy is derived and it is applied in the system.

4.1 Verification Properties:
To verify the CPN model three key properties are defined to ensure the contexts are combined to build scenarios, to complete the execution process and to avoid deadlock state among the contexts. Three key properties of proposed Security CPN model are

- **P0 (symmetry): different scenario and its I/O are unordered:**
  This property can be verified by different possible context in random order and check whether new scenario is created correctly.

- **P1 (no deadlock): the model may always process different incoming context requests:**
  This property can be verified by observing the number of new scenario generated depending on the rule applied on them and as well as by observing the liveness property which give the dead marking as result of the model.

- **P2 (fairness): every place and transit of CPN model are detected and processed:**
  This property can be verified by observing the fairness property of state space analysis report. It should contain no infinity sequence which mean that the model work completely as well as correctly.

  P0, P1, P2 are difficult to verify, despite it can be achieved through the execution of number of test cases and also to examine all possible execution orders.

Verification is carried out to check the performance, liveness, reliability, and fairness of an application.

Verification of context-aware middleware consists of the following steps.

- **Step 1: Start**
- **Step 2: Get the initial detail E.g. Number of context**
- **Step 3: Add the new context to the existing context until all the context condition is added.**
  For example, contexts are

<table>
<thead>
<tr>
<th>Location</th>
<th>User</th>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: ATM Branch</td>
<td>1: Geetha</td>
<td>1: ATM</td>
</tr>
<tr>
<td>2: Online Bank</td>
<td>2: Madhu</td>
<td>2: Phone</td>
</tr>
<tr>
<td>3: Mobile Bank</td>
<td>3: Deepa</td>
<td>0: Default</td>
</tr>
<tr>
<td>0: Branch</td>
<td>0: Default</td>
<td>0: Default</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operation</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: Cash Withdrawn</td>
<td>1: Morning</td>
</tr>
<tr>
<td>2: Cash Deposit</td>
<td>2: Evening</td>
</tr>
<tr>
<td>3: Enquire</td>
<td>0: Default</td>
</tr>
<tr>
<td>0: Default</td>
<td></td>
</tr>
</tbody>
</table>

- **Step 4: Build the scenario using new context**
- **Step 5: Scenario are stored in the temporary holder**
- **Step 6: Get Security Policy Validator file detail**
Step 7: Get the scenario from step 5 and get the Security Policy Validator file from step 6 match it, if true change the Security policy to the new security policy or ignore it.
Step 8: Now new security policy rule is applied to the System.

![CPN Model for Proposed Architecture](image)

**Fig. 2:** CPN Model for Proposed Architecture.

\[
\begin{align*}
\{ & L1 \quad U1 \quad D1 \quad O1 \quad T1, \ldots \\
& L1 \quad U1 \quad D1 \quad O1 \quad T2, \ldots \\
& L1 \quad U1 \quad D1 \quad O1 \quad T3, \ldots \\
& L1 \quad U1 \quad D1 \quad O1 \quad T4, \ldots \\
& L3 \quad U3 \quad D2 \quad O3 \quad T1, \ldots \\
& L3 \quad U3 \quad D2 \quad O3 \quad T2, \ldots \\
& L3 \quad U3 \quad D2 \quad O3 \quad T3, \ldots \\
& L3 \quad U3 \quad D2 \quad O3 \quad T4, \ldots 
\end{align*}
\]

Three functions are used to verify the new scenario,
- Security Policy Validator function: It gets the security policy rule from the rule file.
- Security Policy Builder: It gets the new scenario from the scenario function and security policy from the Security Policy Validator function. Then compares the new scenario with the security policy, if it matches then the new security policy is sent to the result function.
- New Policy function: The security policies are received from Security Policy Builder function and it is stored in the corresponding security file.

- **Performance:** The state space analysis report from CPN model, will report the time taken to analysis the process and its feasibility.
- **Liveness:** Dead marking function will report the number of solution states for the given application. If dead transition and live transition function will reports none as its result, it indicates that there will be more than one solution for the given application.
- **Reliability:** Complete execution of the process and always produce result to the application.
• **Fairness:** the process should terminate properly and they should not go into infinity sequences.

4.2 Case Study: Verification of CPN model in Bank Environment:

Bank environment is consider with 4 user contexts, 2 operation contexts, 1 location contexts, 2 timing contexts and 1 device contexts. Various contexts are aggregated to form different scenario. These scenarios in the given environment can be verified using the Coloured Petri Net tool.

CPN Tools state space report for:
/cygdrive/C/cpn/gum3.cpn

Statistics
--------------------------
State Space
Nodes: 20868
Arcs: 81703
Secs: 300
Status: Partial

Scc Graph
Nodes: 20868
Arcs: 81703
Secs: 2

Boundedness Properties
---------------------------
Best Integer Bounds

<table>
<thead>
<tr>
<th></th>
<th>Upper</th>
<th>Lower</th>
</tr>
</thead>
<tbody>
<tr>
<td>New_Page_Device 1</td>
<td>11</td>
<td>0</td>
</tr>
<tr>
<td>New_Page_Location 1</td>
<td>11</td>
<td>0</td>
</tr>
<tr>
<td>New_Page_Policy 1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>New_Page_Scenario 1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>New_Page_Security_Policies_Validator 1</td>
<td>66</td>
<td>0</td>
</tr>
<tr>
<td>New_Page_Time 1</td>
<td>33</td>
<td>0</td>
</tr>
<tr>
<td>New_Page_User 1</td>
<td>12</td>
<td>0</td>
</tr>
<tr>
<td>New_Page_operation 1</td>
<td>33</td>
<td>0</td>
</tr>
</tbody>
</table>

Home Properties
--------------------------
Home Markings
None

Liveness Properties
--------------------------
Dead Markings
8015 [20868, 20867, 20866, 20865, ...]

Dead Transition Instances
None

Live Transition Instances
None

Fairness Properties
--------------------------
No infinite occurrence sequences.

4.2.1 Verification of three key properties of CPN Model applied on bank environment:

From the below specification of the input parameter, that are used as input to the CPN model to verify the possible combination of scenarios and its execution process to generate different scenarios are summarized in the Table 1.
Table 1: CPN Model Input Parameters for Smart Bank.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Smart Bank</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of User</td>
<td>4</td>
</tr>
<tr>
<td>Number of device</td>
<td>1</td>
</tr>
<tr>
<td>Number of operation</td>
<td>2</td>
</tr>
<tr>
<td>Number of location</td>
<td>1</td>
</tr>
<tr>
<td>Number of Time</td>
<td>2</td>
</tr>
<tr>
<td>State Space Node</td>
<td>20868</td>
</tr>
<tr>
<td>Strongly Connected Component Node</td>
<td>20868</td>
</tr>
<tr>
<td>Fairness Properties</td>
<td>No Infinity sequence</td>
</tr>
</tbody>
</table>

P0: Bank environment consist of 4 user contexts, 2 operation contexts, 1 location contexts, 2 timing contexts and 1 device contexts. These contexts are combined in different possible combination to generate 16 possible scenarios. From this it is evident that it is the maximum possible scenarios for the given application.

P1: According to the liveness property Kurt Jensen and Lars M. Kristensen (2009), Dirk Fahland and Christian Gierds (2013), G. Clark, T. Courtney et al (2001), Vinai George Bijuand Santanu Kumar Rath (2010), if the result of dead transition and live transition function is none then it indicates that there will be two or more result states. In our experiment, it is observed that the states of liveness properties of dead marking function are 8015 which prove liveness property.

P2: By observing the fairness properties it is verified that it does not have any infinity sequences which mean there is no deadlock situation in the applications.

5. Discussion:

Critical parameters like liveness, fairness and scenario handling of the case study are verified with help of CPN. From the verification carried out, it is evident that the proposed system will work for different domain applications like Smart Home, Smart Hospital, etc... Our verification is limited to functional aspects of the model. Our future work is to study the non functional aspects.

6. Conclusion:

Thus the security model is been verified and the security policies are provided to the context according to the scenario of the application. Since the security policies is been built based on various context the proposed model will help us to develop a security model for all type of pervasive applications to improve its security smartly.

REFERENCES


