Trust4All: a Trustworthy Middleware Platform for Component Software

ZHENG YAN, CHRISTIAN PREHOFER, VALTTERI NIEMI
Nokia Research Center, Itämerenkatu 11-13, 00180 Helsinki
FINLAND

Abstract: - Trust plays an important role in a software system, especially when the system is component based and varies due to component joining and leaving. How to manage trust in such a system is crucial for an embedded device, such as a mobile phone. This article introduces a trustworthy middleware architecture that can manage trust in an autonomic way through adopting a number of algorithms for trust assessment and maintenance with regard to software component download and execution.

Key-Words: - Trust management, Trusted computing, Security, Dependability, Component software

1 Introduction

The growing importance of software introduces special requirements on trust. This normally implies that system software consists of a number of components that are combined to provide user features. Components interact over well defined interfaces; they are exported to applications that can combine and use the components to provide features to consumers. Thus, common components can be effectively shared by applications. A typical feature of devices with component software support is to allow addition of components after deployment, which creates the need for trust management with regard to software component download and execution.

We adopt a holistic notion of trust which includes several properties, such as security, availability and reliability, depending on the requirements of a trustor. Hence trust is defined as the assessment of a trustor on how well the observed behavior (quality attributes) of a trustee meets the trustor's own standards for an intended purpose [1]. From this, the critical characteristics of trust can be summarized. It is both subjective and dynamic. Concretely, trust is different for each individual in a certain situation and, sensitive to change due to the influence of many factors.

EU ITEA Trust4All project aims to build up a trustworthy middleware architecture in order to support easy and late integration of software from multiple suppliers and still have dependable and secure operation of the resulting system. Nokia Research Center participates in this project as a partner.

In this article, we introduce Nokia's work conducted in the Trust4All Project towards autonomic trust management for a component software platform. Obviously, it does not suffice to require the trustor (e.g. most possibly a digital system user) to make a lot of trust related decisions because that would destroy any attempt at user friendliness. For example, the user may not be informed enough to make sound decisions. Thus, establishing trust is quite a complex task with many optional actions to take. Rather trust should be managed automatically following a high level policy established by the trustor, for example a software component or the user of a component software platform. We call such trust management autonomic.

Autonomic trust management concerns trust management in an autonomic processing way with regard to evidence collection, trust evaluation, and trust (re-)establishment and control. We need a proper mechanism to support autonomic trust management not only on trust establishment, but also on trust sustaining. This is important for a component software platform that should support trustworthy downloading and executing of the software components. We develop a trustworthy middleware architecture that can manage trust in an autonomic way through adopting a number of algorithms for trust assessment and maintenance with regard to software component download and execution.

2 Trust Issues of Component Software

In mapping trust to the component software system we can categorize trust into two aspects: trust in the component, and trust in a composition of components. For the component-centered aspect we must consider trust at several decision points: at download time and during execution. At a component download time, we need to consider whether a software provider can be trusted to offer a component. Furthermore, we need to predict whether the component is trustworthy for installation. More necessarily, when the component is executed, we have to ensure it can cooperate well with other components and the system provides expected performance and quality. The trust relationship between system entities changes during the above procedure.

When discussing a component software system, the execution of components in relation to other entities of the system needs to be taken into account. Even though the component is trustworthy in isolation, the new joined component could cause problems because it will share system resources with others. This may impact the trustworthiness of the whole system. Consequently, the system needs mechanisms to control its performance, and to ensure its trustworthiness even if internal and external environment changes. Additionally, some applications (e.g. a health care service) need special support for trust management because they have high priority requirements, whereas other applications (e.g. games), while exhibiting similar functionality (e.g. a network connection) will not have the same priority. Therefore, system-level trustworthiness is dependent on the application domain. So the system needs a trust management framework that supports different trust requirements for the same software components, depending on the context they are used.

3 Trust4All Architecture

The architecture of the component software system consists of layered structure: an application layer that provides features to a user; a component-based middleware layer that provides functionality to applications; and, a platform layer that provides access to lower-level hardware. Using components to construct the middleware layer divides this layer into two sub-layers: a component sub-layer that contains a number of executable components and a runtime environment (RE) sub-layer that supports component development.

The component runtime supporting frameworks also exist at the RE sub-layer. They provide functionalities for supporting component properties and for managing components. These frameworks also impose constraints on the components, with regard to mandatory interfaces, associated metadata etc. The runtime environment consists of a component framework that treats DLL (Dynamic Link Library)-like components. It provides a system-level management of the component configuration inside a device. Each component contains services that are executed and used by applications. The services have interactions with other services; they consume resources; and, they have metadata attached. The trust model of the software component is one kind of the metadata. It indicates required resources for providing specified performance, the trust priority level and composition rules for composing this model with other trust models [1].

Some frameworks in the runtime environment have to be supported with platform functionality. For example, for a resource framework, support for resource usage accounting and enforcement is required from the platform layer. In terms of trust management, the platform needs to provide security mechanisms, such as access control, memory protection and encryption/decryption. In this case the security framework offers functionalities for the use of security mechanisms, provided by the platform, to requests raised by a trust management framework in order to develop and maintain a secure system. The platform layer also provides trusted computing support on the upper layers [2].

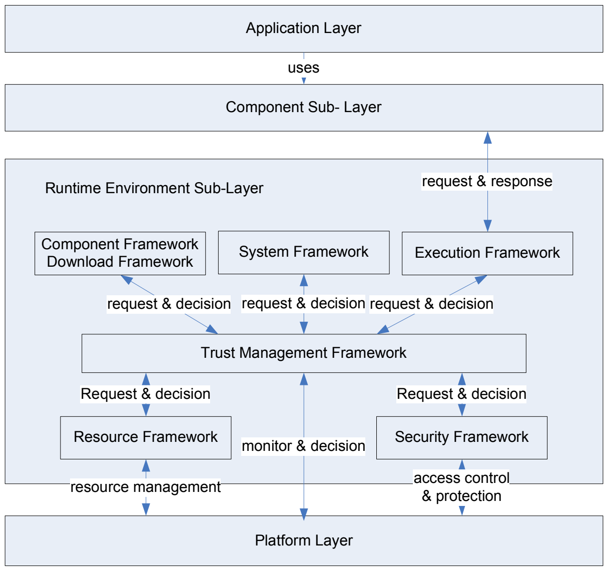

Figure 1: Relationships among trust framework and other frameworks

Figure 1 describes interactions among different functional blocks inside the running environment sub-layer. Placing trust management inside this architecture means linking the trust management framework with other frameworks responsible for the component management (including download), the security management, the system management and the resource management.

The trust management framework is responsible for the assessment on trust relationships and for automatically selecting suitable trust control mechanisms, system performance monitoring and autonomic trust management. The download framework requests the trust management framework for trust assessment about components to decide if to download a component and which kind of mechanisms should be applied to this component. When a component service needs cooperation with other components' services, the execution framework will be involved, but the execution framework will firstly request the trust management framework for decision. The system framework takes care of system configurations related to the components. Similarly, the trust management framework controls the security framework, to ensure that it applies the proper security mechanisms to maintain a trustworthy system. The trust management framework is located at the core of the runtime environment sub-layer. It monitors the system performance and instructs the resource framework to assign suitable resources to different processes. This allows the trust management framework to shut down any misbehaving component, and to gather evidence on the trustworthiness of a system entity. So briefly, the trust management framework acts like a critical system manager, ensuring that the system conforms to its trust policies.

4 Autonomic Trust Management for Component Software Platform

As defined in [3], trust management is concerned with collecting the information required to make a trust relationship decision; evaluating the criteria related to the trust relationship as well as monitoring and re-evaluating existing trust relationships; and automating the process. We think that this concept needs to be extended in order to automatically control and ensure trust in a dynamically changed software platform. We employ autonomic trust management, which includes the following four aspects:

- Trust establishment: the process for establishing a trust relationship between a trustor and a trustee.
- Trust monitoring: the trustor or its delegate monitors the behaviour of the trustee. The monitoring process aims to collect useful evidence for trust assessment.
- Trust assessment: the process for evaluating the trustworthiness of the trustee by the trustor or its delegate with respect to specified criteria or policy. The trustor assesses the current trust relationship and decides if this relationship has changed.
- Trust control and re-establishment: if the trust relationship has changed, the trustor will find reasons and make a decision if and which measures should be taken in order to control or re-establish the trust relationship.

4.1 Factors related to trust

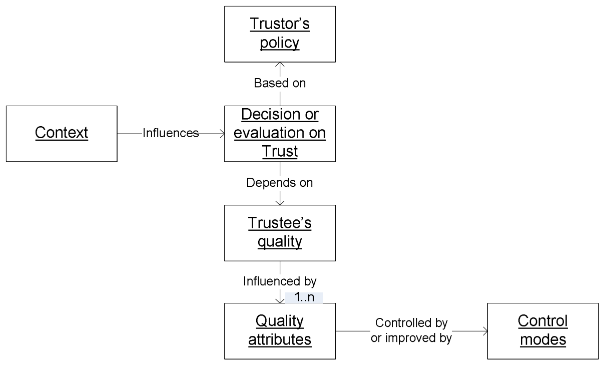

Figure 2: Factors related to trust

We consider a component software platform which is composed of a number of entities, e.g. a component (composition of components), an application, a sub-system and the whole platform system. The trustworthiness of a platform entity depends on a number of quality attributes of this entity. The quality attributes can be the entity's trust properties (e.g. security, availability and reliability) and recommendations or reputations with regard to this entity. The decision or assessment of trust is conducted based on the trustor's (e.g. a platform user or his/her delegate) subjective criteria or policies and the trustee entity's quality attributes, as well as influenced by context information. Context includes any information that can be used to characterize the situation of the involved entities. The quality attributes of the platform entities can be controlled or improved by applying a number of control modes. Particularly, a control mode contains a number of control mechanisms or operations, e.g. encryption, authentication, hash code based integrity check, access control mechanisms, duplication of process, man-in-middle solution for improving availability, etc. It can be treated as a special configuration of trust management that can be provided by the system. The relationships of those factors related to the trustworthiness of a platform entity are illustrated in Figure 2.

4.2 A procedure of autonomic trust management

Based on the above understanding, we propose a procedure to conduct autonomic trust management in the component software platform targeting at a trustee entity specified by a trustor entity, as shown in Figure 3.

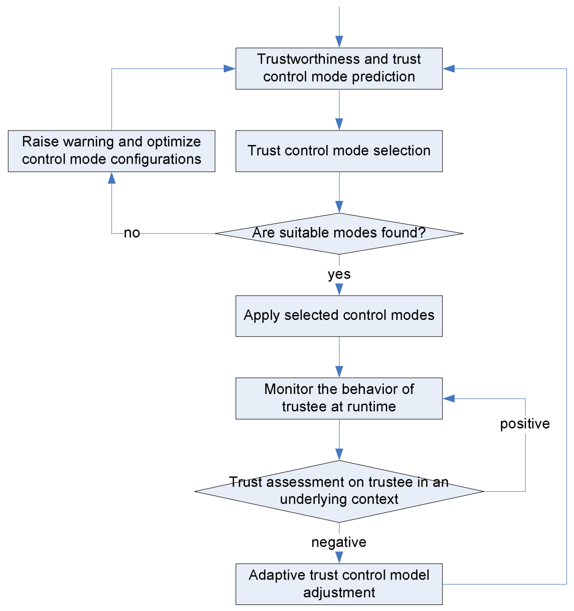
Trustworthiness and trust control mode prediction
Raise warning and optimize control mode configurations
Trust control mode selection
no
Are suitable modes found?
yes
Apply selected control modes
Monitor the behavior of trustee at runtime
positive
Trust assessment on trustee in an underlying context
negative
Adaptive trust control model adjustment

Figure 3: Autonomic trust management procedure at runtime

Trust control mode prediction is a mechanism to anticipate the performance or feasibility of applying some control modes before taking a concrete action. It predicts the trust value supposed that some control modes are applied before the decision to initiate those modes is made. Trust control mode selection is a mechanism to select the most suitable trust control modes based on the prediction results.

For a registered trustor at the trust management framework, the trustworthiness of its specified trustee can be predicted regarding various control modes supported by the system. Based on the prediction results, a suitable set of control modes could be selected to establish the trust relationship between the trustor and the trustee. Further, a runtime trust assessment mechanism is triggered to evaluate the trustworthiness of the trustee through monitoring its behavior based on the instruction of the trustor's policies, as described in [1]. According to the runtime trust assessment results in the underlying context, the system conducts trust control model adjustment in order to reflect the real system situation if the assessed trustworthiness value is below an expected threshold. This threshold is generally set by the trustor to express its real expectation on the assessment. Then, the system repeats the procedure. The context-aware or situation-aware adaptability of the trust control model is crucial to re-select suitable control modes in order to fulfill autonomic trust management.

4.3 A trust control model

We developed a trust control model based on Fuzzy Cognitive Map to support autonomic trust management [4, 5]. It is a signed directed graph with feedback, consisting of nodes and weighted arcs. Nodes of the graph are connected by signed and weighted arcs representing the causal relationships that exist between the nodes. There are three layers of nodes in the graph. The node in the top layer is the trustworthiness of the platform entity. The nodes located in the middle layer are the quality attributes of the entity, which have direct influence on the entity's trustworthiness. The nodes at the bottom layer are control modes that could be supported and applied inside the system. These control modes can control and thus improve the quality attributes. Therefore, they have indirect influence on the trustworthiness of the entity.

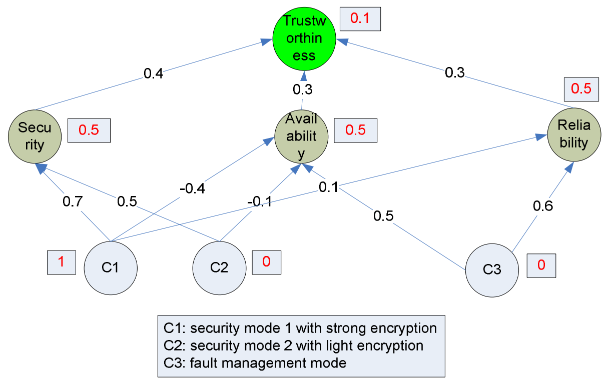
Trustworthiness
0.1
0.4
0.3
0.3
0.5
Security
0.5
Availability
0.5
Reliability
0.7
0.5
-0.4
-0.1
0.1
0.5
0.6
1
C1
C2
0
C3
0
C1: security mode 1 with strong encryption
C2: security mode 2 with light encryption
C3: fault management mode

Figure 4: An example of trust control model

An example of this model is shown in Figure 4. The trustworthiness of the trustee entity is influenced by three quality attributes: QA_1 - Security; QA_2 - Availability; QA_3 - Reliability, with important rates $w_1 = 0.4$, $w_2 = 0.3$, and $w_3 = 0.3$, respectively. There are three control modes that could be provided by the system:

- C_1: security mode 1 with a strong encryption service for encrypting data, but medium negative influence on availability.

- C_2: security mode 2 with a light encryption service for encrypting data and light negative influence on availability.
- C_3: fault management mode with positive improvement on availability and reliability.

The influence factors of each control mode to the quality attributes are specified by the arc weights. The values in the square boxes are initial values of the concept nodes. In practice, the initial value can be set as asserted one or expected one, which can be specified in the trustor's policy profile.

4.4 Algorithms applied for autonomic trust management

There are a number of algorithms adopted by the trust management framework for autonomic trust management. For details, refer to [1, 4, 5]

- **Trust prediction for component software downloading and execution**

Trustworthiness prediction is one of important issues that should be considered with regard to trust management of component software. The trustworthiness of a component should be predicted before initiating a concrete action, and this prediction should be comprehensive regarding multiple factors that could influence trust. We proposed two algorithms to predict trustworthiness for software components downloading and execution. The methodology is based on a trust model, which indicates the component's asserted performance and requirements for achieving the performance. Through evaluating the related trust models, the software component's reputation and the component software platform's competence, the algorithms can predict the trustworthiness of the software components. The prediction result is significant to determine whether to initiate the component downloading or start the execution of the component services. It also helps in locating system resources according to the trust priority level in the case of any conflict [12].

- **Trust assessment at runtime**

We develop a formal trust model to specify, evaluate and set up trust relationships amongst system entities. Based on this model, we apply a simplified scheme of the Subjective Logic to conduct runtime trust assessment based on observation [1].

- **Control mode prediction and selection**

The trust control mode prediction is a mechanism to anticipate the performance or feasibility of some control modes supposed that those modes are applied before the decision to initiate them is made. We developed an algorithm based on the trust control model to conduct the trust control mode prediction as described in [4]. We further developed another algorithm in order to select the most suitable control modes based on the above prediction results. In the component software platform, the control mode prediction and selection are important functionalities with regard to the automatic processing of trust management [5].

- **Adaptive trust control model adjustment**

It is important for the trust control model to reflect the real system situation and context precisely. The influencing factors of each control mode should be context-aware. The trust control model should be dynamically maintained and optimized in order to reflect the real system situation. Thereby, it is sensitive to indicate the influence of each control mode on different quality attributes in a dynamically changed context. For example, when some malicious behaviors or attacks happen, the currently applied control modes can be found not feasible based on trust assessment. In this case, the influencing factors of the applied control modes should be adjusted in order to reflect the real system situation. Then, the system can automatically re-predict and re-select a set of new control modes in order to ensure the trustworthiness. In this way, the system can avoid using the attacked or useless trust control modes in a special context. Therefore, an adaptive trust control model is important for supporting autonomic trust management for the component software platform. We developed a couple of schemes to adaptively adjust the trust control model in order to achieve the above purpose [5].

5 Related Work

A number of trusted computing and management work have been conducted in the literature and industry, which mostly focus on some specific aspects of trust. For example, TCG (Trusted Computing Group) aims to build up a trusted computing device on the basis of a secure hardware chip [2]. Some of trust management systems focus on protocols for establishing trust in a particular context, generally related to security requirements. Others make use of a trust policy language to allow the trustor to specify the criteria for a trustee to be considered trustworthy [3]. However, the focus on the security aspect of trust tends to assume that the other non-functional requirements [6], such as availability and reliability, have already been addressed. In addition, TCG based trusted computing solution can not handle the runtime trust management issues of component software.

Recently, many mechanisms and methodologies are developed for supporting trustworthy communications and collaborations among computing nodes in distributed systems [7-9]. These methodologies are based on digital modeling of trust for trust evaluation and management. However, most of existing solutions focus on the evaluation of trust, whilst they lack a proposal regarding how to manage trust based on the evaluation result. They generally ignore the influence of trust control mechanisms on trustworthiness. We found that these methods are not feasible for supporting the trustworthiness of a device software platform.

Regarding software engineering, trust has been recognized as an important factor for the component software platform. A couple of interesting models have been proposed to ensure the quality of component services at runtime and protect the users [10, 11]. However, we found that the trust model proposed in [10] mainly focuses on the runtime component configuration support, while the model in [11] aims to prevent that a component user sends wrong reports resulting in a bad trust value of the component, especially at component download time. The on-going TrustSoft project aims to study a holistic approach to software trustworthiness through certifying multiple quality attributes of the software [13]. We argue that trust can be controlled according to its prediction or assessment result. Special control modes can be applied into the software platform in order to ensure a trustworthy system in an autonomic approach.

6 Conclusions

In this article, we summarized our results towards autonomic trust management for the component software platform. Our main contributions include that we developed several trust models to specify, predict, assessment, set up and maintain the trust relationships that exist among system entities for the component software platform. We further design an autonomic trust management architecture that adopts a number of algorithms for trust assessment and maintenance during component download and execution. These algorithms make use of recent advances in Subjective Logic and Fuzzy Cognitive Map to ensure the management of trust within the component software platform in an autonomic way.

For future work, we will further study the performance of the algorithms towards practical use of our results.

References:

[1] Z. Yan, R. MacLaverty, Autonomic Trust Management in a Component Based Software System, *ATC'06*, LNCS vol. 4158/2006, pp. 279-292, China, Sept. 2006.

[2] Trusted Computing Group (TCG), TPM Specification, version 1.2, 2003. https://www.trustedcomputinggroup.org/specs/TPM/

[3] T. Grandison, M. Sloman, A Survey of Trust in Internet Applications, *IEEE Communications and Survey*, Forth Quarter, 3(4), pp. 2–16, 2000.

[4] Z. Yan, A Methodology to Predict and Select Control Modes for a Trustworthy Platform, *WSEAS Transactions on Computer*, Issue 3, Vol. 6, pp. 471-477, March 2007.

[5] Z. Yan, C. Prehofer, An Adaptive Trust Control Model for a Trustworthy Component Software Platform, *ATC'07*, LNCS, Springer, July 2007.

[6] S. Banerjee, C. A. Mattmann, N. Medvidovic, L. Golubchik, Leveraging Architectural Models to Inject Trust into Software Systems, ACM SIGSOFT Software Engineering Notes, *Proceedings of the 2005 workshop on software engineering for secure systems—building trustworthy applications*, Volume 30, Issue 4.

[7] Z. Zhang, X. Wang, Y. Wang, A P2P Global Trust Model Based on Recommendation, *Proceedings of 2005 International Conference on Machine Learning and Cybernetics*, Vol. 7, Aug. 2005, pp.3975 – 3980.

[8] C. Lin, V. Varadharajan, Y. Wang, V. Pruthi, Enhancing Grid Security with Trust Management**,** *Proceedings of IEEE International Conference on Services Computing* (SCC 2004), Sept. 2004, pp. 303 – 310.

[9] Y. Sun, W. Yu, Z. Han, K.J.R. Liu, Information Theoretic Framework of Trust Modeling and Evaluation for Ad Hoc Networks, *IEEE Journal on Selected Area in Communications*, Vol. 24, Issue 2, Feb. 2006, pp. 305 – 317.

[10] M. Zhou, H. Mei, L. Zhang, A Multi-Property Trust Model for Reconfiguring Component Software, *QAIC'05*, pp. 142 – 149.

[11] P. Herrmann, Trust-Based Protection of Software Component Users and Designers. *iTrust 2003*, Crete, Greece, May 2003.

[12] Z. Yan, Predicting Trustworthiness for Component Software, *IEEE SecPerU'07*, July, 2007.

[13] W. Hasselbring, R. Reussner**,** Toward Trustworthy Software Systems, *IEEE Computer*, Volume 39, Issue 4, April 2006, pp. 91 – 92.