ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/309717226

Using trustworthiness properties to support the procurement of software services

Conference Paper · October 2016

CITATIONS
0

READS
87

3 authors, including:

Stuart Short
SAP Labs France
12 PUBLICATIONS **49** CITATIONS

SEE PROFILE

Sachar Paulus
Hochschule Mannheim
61 PUBLICATIONS **517** CITATIONS

SEE PROFILE

All content following this page was uploaded by Sachar Paulus on 05 November 2016.

The user has requested enhancement of the downloaded file.

Using trustworthiness properties to support the procurement of software services

Antonino Sabetta, Stuart Short
SAP
Sachar Paulus
Mannheim University of Applied Sciences

About the Authors

Stuart Short, has been with SAP since 2006 and is a senior researcher in the SAP Global Security department. He has been the project lead of the European funded project OPTET and has previously been involved in other security and trust European projects such as PrimeLife, TAS3, ASSERT4SOA and Fi-Ware. Prior to working for SAP, he has over ten years experience working in start-ups including his own. He received his research masters from the National University of Ireland, Galway.

Antonino Sabetta is a researcher at the SAP Research Center Sophia-Antipolis in the Security and Trust Research Area. Before joining SAP in 2010, he worked as a researcherat CNR-ISTI, Pisa. He earned his PhD in Computer Science and Automation Engineering from the University of Rome Tor Vergata, Italy in 2007. From the same university he had previously earned his L̈aurea cum Lauded̈egree in Computer Engineering in 2003.

Sachar Paulus is Professor for IT Security at Mannheim University of Applied Sciences. Before this, he was Professor for Security Management at Brandenburg University of Applied Sciences. He was 8 years with SAP in several leading positions related to security, such as SVP Product Security and Chief Security Officer. He was part of RISEPTIS, the Adivdory Board on Ressearch and Innovation in Security, Privacy and Trust in the Information Society of the European Commission and founding member of the ENISA Permanent Stakeholders Group. He has a Ph.D. in number theory.

Keywords

Computer Security, ICT Security, Metrics, Trustworthy Assessment.

Using trustworthiness properties to support the procurement of software services

Abstract

The procurement of software services is a difficult task because its requirements are many and is hard to understand whether a given product satisfies them or not. Furthermore there may be an element of doubt as to the bona fide nature of the description that is supplied by the solution provider. In this paper we discuss how the decision-making process can be facilitated by augmenting the usual description of software services with certificates that represent objective trustworthiness properties of the service at hand. These certified properties need to convey detailed technical information, and at the same time, they must be understandable by the decision maker. We show how to reconcile these conflicting needs using a framework that allows to define and reason about trustworthiness properties in terms of a set of underlying metrics.

1 Introduction

In this section, we will describe the problem that motivated the development of the solution using trustworthiness assertions based on metrics, define what we mean by trustworthiness, present existing approaches to the problem and the drawbacks thereof, describe the requirements for trustworthiness assertions and present application scenarios.

Throughout the introduction, we will come back to three illustrative running examples together with user expectations to demonstrate and explain our rationale. These are:

- A trip expense management application that is run in the Cloud. As a typical software as a service offering, corporate users directly log in to the application using a web browser.
- A runner workout manager app. This is a mobile application that tracks running time, heart beat rate and maybe some more health related information using a device connected to a mobile phone such as a smart watch. Users download and install the app from an app marketplace.
- A cloud-based software development project. Developers as users can choose software components from a variety of potential SOA based

offerings that they can integrate into their own application, such as for example a data analytics package.

1.1 The problem

The examples described above present typical challenges of today's information technology:

- How is the trip expense management cloud application actually selected, given the expectations mentioned? In the best case, the IT department executes some (more or less extensive) testing to verify that confidentiality and availability are met, but in this case, the flexibility and dynamicity of cloud applications is gone; a cloud application selection and testing process could actually take several weeks if not months. In the worst case, testing is not done at all, and there is no guarantee whatsoever that the data is kept confidential and that the application fulfills the availability needs. So either corporations decide to be fast and risk friendly, or need to invest time and effort to identify trustworthiness criteria.
- How do consumers select workout manager apps? They typically don't look at some technical details on the web page of the developer, but trust the information that is presented in the app marketplace. But in today's marketplace, there is only a recommendation from other users, which does not allow to decide on the trustworthiness (in terms of confidentiality and privacy) of the mobile application. Average users are therefore not in a situation to really decide whether the app should be trusted.
- How to developers choose SOA packages? Today, they need to follow a trial and error approach. There is very little information available on criteria that would help the developer in making a trusted decision, so he needs time and effort to see whether his criteria - in this case, expectations on response time and computing accuracy - are met, and even then he might not be able to test the "real" case.

The examples chosen demonstrate that today it is very difficult, if not impossible, as a user of software and or internet-based services to make a trusted decision. This holds both for consumers and corporate users. Corporations may have the resources and the willingness to verify specific requirements to be met, but then one of the big advantages of service-oriented architectures are lost.

But what if we could actually make informed trust decisions? The service application selection process would take a bit more time, but that would minimize risks and or efforts to validate claims of the software manufacturer. Users would be able to trust software and internet-based services after having made a short decision.

What could a software manufacturer do to support this process? Obviously, the manufacturer service provider cannot "inject" trust into the user, since decisions about trust are subjective and make use of conscious and unconscious processes. But: the manufacturer could support this process by providing information about objective, observable properties of the software service or the usage thereof, what we call trustworthiness. This is explained in the next subsection.

Note that throughout this paper, we will not distinguish between software and internet-based services. From a generic trust point of view, there is no substantial difference.

1.2 Trustworthiness

As mentioned earlier, manufacturers cannot directly influence the trust decision making process of a user. But they can provide information about properties that help in that process. To be trusted in general, that information must be verifiable by another party. For the context of this paper, we understand trustworthiness as objective observable properties that contribute to the trust establishment by users.

Obviously, trustworthiness may be different for different use cases, and different user groups. The basic properties may be identical but they differ in their relative importance, or they may be totally different. To demonstrate this, we describe the trustworthiness properties in the three illustrative examples:

– For the trip expense management cloud application, there are three trustworthiness properties: the confidentiality of user information, the confidentiality of company profiles and the availability of the service.

– For the runner workout manager app, the trustworthiness properties are: the confidentiality of the user information, and the privacy respecting usage of the data.

– For the SOA developer scenario, the trustworthiness properties consist of maximum response time and computing accuracy.

1.3 Existing approaches

The currently observable mechanisms to communicate trustworthiness in practice are based on two different approaches:

- *User-centric reputation systems*: Users can rate software / services, typically using a very simple "5 star" approach. This is used to demonstrate how "happy" users are with the software, and to allow new users to potentially rely on this information. This approach has two advantages: it is easy to implement, and it is easy to understand. But it also has a number of disadvantages: it is highly subjective, subject to manipulation and does not take objective information (that is already available) into account. Moreover, it does not differentiate between different use case scenarios or preferences.
- *Technology-centric certification systems*: A completely different approach that can primarily be observed in the professional / business environment is the use of certifications, i.e. assertions from third parties that (seem to) guarantee that specific properties are met (Common Criteria, ISO 27001, TrustE certificates etc.). This approach is often sound and allows an objective comparison of, for example, the security operations employed. But the information cannot be easily understood by users, and often only covers a very small set of the properties that are subject to a trustworthiness evaluation by the user.

From a user's perspective, either the information is easily understandable, but potentially not reliable, or it has a sound technical foundation, but users have a hard time understanding its real meaning and value.

Both approaches, though very different in nature, try to convey evidences which are considered to be of value to the user that tries to make a choice. Many business organizations are investigating valuable evidences that they base their decisions on - from pure qualitative assertions (such as privacy compliance) to quantitative measurements (such as security vulnerabilities and available fixes) when evaluating and selecting apps. Such information is used in professional bidding processes, but to our knowledge there is no standard approach of collecting, describing and verifying such evidences - we call them trustworthiness assertions - upfront by manufacturers or market places and to offer them to the users as decision support.

1.4 Requirements for trustworthiness assertions

Properties of trustworthiness assertions can be generalized from the observations in the last subsection. We propose to develop a mechanism that fulfills the following properties:

- They must have a very good usability. This means that trustworthiness assertions must be easy to unterstand and easy to access. Ideally, to use them for trust decisions, they are as easy as standard reputation systems.
- They must be technically sound. This means that the construction and verification of the trustworthiness assertions follows a technically and ideally scientifically proven approach.
- They should support different user types and usage scenarios, since different user groups may have different expectations in terms of trustworthiness.
- They should support different modes of implementation, such as different certification and validations schemes, and support existing industry norms and standards wherever possible.

Note that the trustworthiness assertions as we describe them here may have a different level of subjectivity - as long as the construction principle is based on objective rules and therefore allows independent verification.

A key observation from our point of view is that existing approaches either try to focus on a good usability, or alternatively try to be technically sound. In this paper, we try to balance both approaches by introducing metrics of observable properties of software / services as a technically sound tool that allows to produce easy-to-grasp trustworthiness assertions.

1.5 Application scenarios

Metrics will be defined in detail in the next section. Using metrics, the selection criteria in our three illustrative examples could look like the following:

- With the trip expense management application users expect the information to be kept confidential, and corporate customers expect the application to be available therefore selection is based on the following two metrics: confidentiality and availability

– For the runner workout manager app users expect the information to be kept confidential, and in addition, that statistics and analytic information is shared in an anonymized manner to preserve the privacy of the user: applicable metrics are confidentiality and privacy

– In a cloud-based software development project developer-type users expect a maximum response time and maybe a certain accuracy of the results, so the metrics concerned with this scenario are response time and accuracy

1.6 Why trustworthiness app metrics?

Increasingly, software and software supported internet services are delivered through so-called app markets or marketplaces. This is primarily a result of the increasing control that operating system manufacturers (for mobile devices, laptop/desktop or server computers) have over the software delivery to their platforms and is motivated by a number of reasons: first, a controlled delivery of software and/or services allows better service (system support, error analysis, etc.), second it allows to monetize the access to the platform (Apple e.g., keeps approx. 60% or the revenue generated by apps) and third it allows to make the platform in principle more trustworthy, since the operating system manufacturer can check the delivered software/service for non-functional properties, such as weaknesses and vulnerabilities or malicious activities prior to the availability in the marketplace.

Whereas some marketplaces are still relatively poor, and users are happy to find any app for their need (e.g., a music streaming app for Firefox OS), the major marketplaces (such as, for example Google Play, Amazon Appstore, and Apple AppStore) offer more than enough apps for the different needs of the users. Users in that case need a simple mechanism to differentiate apps. One of the increasing expectations of users, especially since the "Snowden revelations" took up, is the trustworthiness of the app they are considering to use. The currently observable mechanisms to communicate trustworthiness in practice are based on two different aspects:

– *Reputation*: Users can rate an offering, typically using a very simple "5 star" approach. This is used to demonstrate how happy users are with the software, and to allow new users to potentially rely on this information. This approach has two advantages: it is easy to implement, and it is easy to understand. But it also has a number of disadvantages: it is highly subjective, subject to manipulation and does not take objective information (that is already available) into account. Moreover, it does not differentiate between different use case scenarios or preferences.

– *Certificates*: Another completely different approach that can primarily be observed in the professional / business environment is the use of certificates, i.e. assertions from third parties that (seem to) guarantee that specific properties are met (Common Criteria, ISO 27001, TrustE certificates etc.). This approach is often sound and allows and objective comparison of, for example, the security operations employed. But the information cannot be easily understood by users, and often only covers a very small set of the properties that are subject to a trustworthiness evaluation by the user.

Both approaches, though very different in nature, try to convey evidences that are considered to be of value to the user that tries to make a choice. But there are more evidences that could be used for such a selection process, and indeed, many business organizations are investigating valuable evidences that they base their decisions on - from pure qualitative assertions (such as privacy compliance) to quantitative measurements (such as security vulnerabilities and available fixes) when evaluating and selecting apps. Such information is used in professional bidding processes, but to our knowledge there is no standard approach of collecting, describing and verifying such evidences.

We propose to develop a framework that allows to verifiably express trustworthiness related properties of software and apps. To allow a computationally flexible approach, we are making use of metrics that express the fulfillment of defined trustworthiness properties. Since users may have different preferences, and compliance requirements may be different by location or target group, the metrics must be such that they allow a weighted combination in accordance of the different requirements. For ease of use, they in addition need to be combined to trustworthiness categories for which one computed value is presented as part of the decision support.

Metrics in addition are relatively easily verifiable: if the measurement approach is described, and the metrics definition publicly known, everyone could in principle re-compute the value of the metric (assuming access to the source code, or the conditions for operations, for example). Since measurements and metrics can be expressed in mathematical terms, machine readability can also be realized relatively easily. This would allow marketplaces to automatically make use of the metrics values in their decision support systems. The authenticity of the metrics evidences is assured by using digital certificate technology.

Thus, using our approach based on objective metrics of the software employed, it is possible to adapt the decision making support (such as intelligent search and app propositions) according to the preferences of the user, and in such a way that is easy to communicate.

Note that the principles and ideas presented in this paper are not limited to the trustworthiness aspect of software, it very much could also be used for evaluating other software properties.

1.7 Requirements for the metric approach

To successfully employ a new methodology in practice, it is important to take the requirements for this mechanism from the different stakeholders into account. The following group of relevant stakeholders have been identified in our research work, with the corresponding expectations:

- Users: For users of the app / service, the idea of the trustworthiness metric must be easy to grasp. If there are different properties taken into account, they should be limited in number (say, not more than 5 or 6), and there should be a mechanism to express preferences or priorities for these different properties. Moreover, the user expects a certain validity / objectiveness of these metrics, so they should not (solely) depend on recommendations from other users.
- Software manufacturer / Service operator: For those developing applications/services, the metrics employed should be relatively easy to determine, and there should be a mechanism that allows a "preview" of the metrics at development time so that the software/service can be adjusted to the different needs early in the process.
- Marketplace: For the marketplace owner, it is important that the metrics can easily be compared, so that rankings according to different properties and/or preferences is possible, as well as grouping apps with specific trustworthiness properties or that fulfill sector-specific requirements.

To fulfill these requirements, a set of relevant trustworthiness attributes have been defined(see [10]) and grouped into trustworthiness objectives, and designed a metrics framework(see [11]). The set of relevant trustworthiness attributes counts in total more than 40, and for each of these attributes, one or more metrics have been identified. Attributes are grouped to objectives, and the corresponding metrics can be combined meaningfully (as explained later in this paper) to reflect the trustworthiness objective value. Finally, trustworthiness objective measures can be combined using so-called weights to express the preferences of users, and/or requirements from different sectors, to a one-dimensional score, the trustworthiness metric.

To make this logic work, the metrics must fulfill a number of properties. In contrast to many other applications of metrics, where the nature of each individual metric may be very different, we need a homogeneous set of metrics. The required properties are as follows:

- The metrics employed must be numbers between 0 and 1.
- If the metric equals 1, then the corresponding attribute resp. observable property related to the attribute in question is fully present in the software.
- If the metric equals 0, then it is not present at all.
- Cardinal metrics (returning integer values), ordinal metrics (return boolean values) and qualitative metrice (returning something like "low", "medium" or "high") must be mapped to metrics between 0 and 1, and may require to define target values as part of this normative process.

To identify metrics, we have extensively used the GQM approach (Goal - Question - Metric) [12], and tried as much as possible to integrate existing / known metrics into this framework where appropriate. As explained in the list above, where needed, a normalization of the metrics could have been necessary. For more background on the construction of metrics according to GQM, see [11].

1.8 Potential applications

Using metrics that have capabilities as described in the previous sections would allow a number of user-friendly decision support options. For example, if the preferences of a user for different trustworthiness objectives are stored in a personal profile, the marketplace could offer those apps that have a high probability of being the most trustworthy in view of the expectations of exactly this user. The profile could either be edited manually, or the marketplace analyzes previous decisions and deduces a profile based on the trustworthiness focus elements of the apps the user has chosen before (or even is frequently using).

Another example could be that the profiling is done for specific user groups, such as for example, users belonging to certain professions, or status groups, or having a certain age etc. Correspondingly, the trustworthiness computation is done according to the (implicit) preferences of that user group and the suggestions for apps are displayed according to the weighted trustworthiness metric. There are a multitude of options possible, based on the fact that the computation relies on objectively measurable criteria of the software and an observed (or actively managed) profile of the users. This would highly increase the quality of the suggestions based on the non-functional expectations of the user.

2 Concepts

In this section, we explain how we define metrics, how we obtain values for those metrics starting from measurements, and how we can combine metrics to define trustworthiness attributes.

2.1 Metrics definitions, metrics values, and measurements

Metric definitions (or simply metrics) define how to quantify facts about a process, software artifact, execution, etc In practice, metrics define the objective *measurables.*

In our framework, metrics are characterized by:

1. Unique identifier (e.g., a URI)
2. Optional descriptive name
3. Unit of measurement (e.g., meters, seconds, kilograms, lines of code, number of tests, person-months)
4. The set O of values taken by all measurements that refer to this metric (see below).
5. A mapping function from O to the interval $[0, 1]$

It is important to distinguish between metrics definitions (as defined above) from metric values and measurements.

A key goal of our framework is to make it easy to reason quantitatively about measurable security characteristics; in practical terms, this translates into making the mathematics that underpins such reasoning easy. For this reason, we force metric values to be in the interval $[0, 1]$, so that their interpretation as percentages can be supported in an intuitive manner.

A measurement is the concrete value, expressed in a particular metric, observed on a particular subject. In this document, the term "observation" is used to mean "measurement".

Examples: size in lines of code of a software project; effort in person-month spent for a project; time in seconds spent for a computation, and so on.

Measurements can take values in different domains, depending on the metric. Some are in the interval $[0, 1]$, others in the set [true, false], other over the set of natural or real numbers, and so on. This is fixed as part of the metric definition.

2.2 From metrics to trustworthiness attributes

Trustworthiness attributes (TWA) are generic categories of system qualities. When assessing a system, be it during a certification process or at run-time, it is necessary to identify which metrics can help to understand whether the system has a certain quality or not. For instance, the GQM methodology can be used in order to set up the assessment context (the system under analysis), a set of goals (the system qualities and their points of view) and questions to highlight particular system aspects. At this point, it becomes possible to select a set of metrics whose interpretation gives indications with respect to the presence of TWA in the system being analysed.

At this point, a TWA computation can be defined in terms of metrics in such a way that a numerical value can be derived once a set of suitable measurements is available.

A trustworthiness attribute computation is characterized as follows:

1. A descriptive name (need not be unique, although it'd rather be)
2. A unique identifier (e.g., a URI)
3. A set of metrics, used to define the attribute
4. A formula or algorithm defining how to compute a value for the TWA starting from all the above

For example, the TWA A_k could be computed using metrics $M_A = (mi_1, \ldots, m_n)$ as follows:

Figure 1: Examples of $T_k(o) = 1 - e^{-ko}$ with different values of k

$$A_k = \sum_{i=1}^{n} T_i(o_i) \times w_i \tag{1}$$

where o_i denotes a measurement for metric m_i, T is a function that maps the domain of o_i to the interval $[0, 1]$, and w_i are the weights assigned to metric m_i (both o_i and w_i are in $[0, 1]$, which makes the math easy to handle). Also, the weights are such that $\sum_i w_i = 1$. Because the weights are defined in this way, there is no need to normalize in order to have the attribute value stay in $[0, 1]$

Concerning the function T, it could be defined as in the following *examples*. Note that o is assumed to be non-negative(but smilar definitions could be devised to accomodate any real-valued observation).

$$T(o) = \frac{o}{o_{MAX}} \tag{2}$$

$$T(o) = 1 - e^{-o} \tag{3}$$

Or, introducing a parameter k

$$T_k(o) = 1 - e^{-ko} \tag{4}$$

Other formulas or algorithms are possible, as long as they yield values in $[0, 1]$. A transformation function examples can be as follows:

$$T(m) = e^{-\frac{m}{k}}$$

For small values of m, $T(m) \rightarrow 1$ conversely, for large values of m, $T(m) \rightarrow 0$. The value of k determines *how fast* $T(m)$ goes to 0. Changing the sign of k reverses the plot.

Another useful way of defining T is using the tan^{-1} function.

$$T(m) = \frac{1 - tan^{-1}(kx - s)}{2}$$

As above, for small values of m, $T(m) \to 1$ conversely, for large values of m, $T(m) \to 0$.

Here the $\frac{s}{k}$ controls the inflection point: changing s shifts the inflection point, changing k changes the slope of the curve at the inflection point. Changing the sign before tan^{-1} reverses the plot.

The interpretation in our framework is that larger values of T are desirable. In the examples above, low values of the response time are desirable. In different contexts, large values of time intervals are desirable (e.g., the mean time to failure of a system), in which case a different metric definition is to be used.

3 Practical considerations

3.1 Storage of metrics and attribute definitions

A metrics database is necessary to be able to refer to a shared, rich library of "what can be measured". Reference to a metric (definition) is done via URIs. Human readable names are nice to have, so that users can easily find the right metrics they need to define trustworthiness attributes, but they are not essential.

And the metrics database need not be unique (multiple databases, run by different parties are possible).

Similarly, the definition of trustworthiness attributes need to be accessible and referenceable via URIs. Again, categorization (design-time, run-time, etc) and human readable descriptions are useful, but not necessary. I find that too much emphasis has been put on those categories, whereas what matters is the formal definition (formula + metrics) and the unique ID.

3.2 Relation with certification process

What is certified is that the value of a certain TWA computation (identified with its unique ID) is A_k and that the measurements for the underlying metrics $(m_1, \ldots, m_n)$ are $(o_1, \ldots, o_n)$. This can be described in a digital trustworthiness certificate (see [8]) also permitting to describe why and how a TWA is associated to a system, by means of the Trustworthiness Property Specification concept and its various relations and sub-concepts.

3.3 Catering for user preferences (user profiles)

User preferences can be taken into account, e.g., by altering the formula above including a vector of preferences $U = [u_1, \ldots, u_k]$

$$A_k = \sum_{i=1}^{n} T_i(O(m_i)) \times w_i \times u_i \tag{5}$$

Alternatively, thresholds on specific attributes or metrics can be imposed, that candidates need to satisfy in order for them to match the profile.

3.4 Who specifies the weights w_i and the $T(m)$ function

In principle, anyone could host a database of metrics and attributes for the world to use. As long as the metrics and attributes are specified correctly (and addressable via a unique URI), they can be used in a certificate or in any other piece of metadata. That said, we can decide that a specific trusted actor has to maintain the database of metrics and attributes and its content. Who this actor should be is to be discussed (mostly for the trust implications that the choice may have)

4 Prototype

4.1 Existing Marketplaces

The structure of existing cloud software marketplaces contains many common elements such as the actors involved namely, solution providers, end-users and marketplace operators and also in the context of internal and front-end operations such as the submission and treatment of applications so that they can be discovered and deployed. In order to submit an application the solution provider needs to create an account; the purpose for this is two-fold, firstly it enforces the authentication of the user and secondly it allows for the latter to avail of facilities such as having an overview and management of activities. Once submitted an application is then subjected to an approval process which is governed by the policies stipulated by the marketplace. This could, for instance, contain checks on viruses and security scans and the verification of product details against the marketplaces guidelines. The solution provider can access his account details (this may be in the form of a dashboard) and see the status of his submissions; usually this is communicated as pending approval, approved or rejected. When an app is approved the solution provider can then have it published on the marketplace and may have it also unpublished if necessary. The end-user can browse a marketplace without having an account but would have to attain one for purchases. He is helped in his decision making by a search listing, ratings made by other users and recommendations based on the sites browsing history (if the user has logged in then he may have recommendations referring to previous searches/purchases). The marketplace operator is responsible for regulating the operations and guidelines of the marketplace system and is the intermediary in the software acquisition process between software providers and end-users. He looks after aspects such as performing the vetting process on submitted offerings (and their updates), deployments, revocations, managing users and the billing process.

We propose to enhance the aforementioned marketplace paradigm by, on the one hand, permitting end-users to assign metrics and acceptable levels to their functional and non-functional requirements in a profile that can be used as a basis for their search for applications on the marketplace. On the other hand service providers would have to assign metrics and certified values in a machine readable document. These two factors can be compared in the marketplace and used as a way to support the purchasing decision of an end-user.

- End-User Profile: This can be seen as a baseline that applications have to meet in order to satisfy end-user requirements. We can further extend this concept by considering a subject matter expert creating a profile based on regulations that need to be adhered to according to a legal domain or specific country requirements. This profile could be made available for end-users in the admin section of the marketplace.
- Solution Provider: In order for these requirements to match application descriptions there has to be a formal way to consume this information in a machine readable manner. Consequently the marketplace would have to impose a structure in the submission process in order for information to be easily processed and aligned with the structure of end-user requirements contained within the marketplace. We are assuming that this submitted information would contain claims that have been verified by a trusted third party.
- Decision Support: Applying the metrics to both end-user requirements and software claims would facilitate comparability and enable the use of thresholds for benchmarking expectations. For example, an end-user is obliged to have sensitive customer data protected when stored; we could classify this security objective or attribute as confidentiality. In

order to fulfil this there could be put in place a security mechanism such as encryption. We can then measure this by assessing all the sensitive data that needs to be encrypted and compare it to all data that is actually encrypted.

– Example: Given this example the end-user can add this attribute to his profile and set a threshold. He can then add other attributes such as integrity and availability and likewise establish acceptable levels. When the end-user performs for instance a keyword search for an application based on perhaps functional requirements, the profile can be appended to this and matched against the descriptions and metrics outlined in the service providers application. The search results would show a listing of applications that best fit the requirements with an illustration of the deviations if present.

5 Related work

It is stated in [2] that services that demonstrate security properties that have been verified by a trusted third party or a qualified authority[7] can assist in a better understanding of a marketplace offering and improve the decision making in the selection process.

This can be seen in the case of certified products however the consumption of this information in different contexts can be problematic as has been pointed out by [13] since the certification models are not necessarily conducive to service-based software.

Given the distributed nature of software packages especially in a service-oriented environment, there is a tendency for developers and enterprises in general to have trust concerns about the security of the offerings especially when the nature of the information being dealt with is of a sensitive nature [4, 6].

For applications that are bought through a cloud marketplace it is difficult to ensure that there will not be any potential security problems when it is used/downloaded [3, 9]. There are evidently vetting processes [3] that exist within some marketplace structures even though this is not always the case [1] but the analysis are usually in a generic nature and are insufficient to run an in-depth analysis of a particular submission. In [9] it is suggested that tests in the submission phase should be automated and the results should be made available to the end-users. This would allow for a comparison with their security requirements and to put the onus on the person using the software to decide whether or not to proceed with an installation. Privacy concerns could also be waylaid by a proactive assessment of applications on a regular basis in conjunction with the standard analysis in the vetting process [5]. Disclosure of this information can furthermore improve the choices that end-users make, and also permitting a greater knowledge of the risks that may be involved in using a particular product or service.

6 Conclusion and Future Directions

Purchasing software can be seen as an important and delicate process for organisations and an incorrect choice may have a negative impact from both a business and technical perspective. In order to, on one hand, make the most out of a software procurement model (for instance, mobile and cloud) that renders applications more easily available and, on the other hand, to facilitate end-users and developers in meeting security requirements, a usable and technical solution needs to be proposed. More particularly it is necessary for applications and software to describe their trustworthiness in a machine-understandable way in order for this information to be processed and compared to end-user profiles that contain their trustworthiness preferences for both functional and non-functional aspects. Furthermore it is essential that there is an approach that renders the trustworthiness attributes comprehensible and easy to describe by all of the actors involved. For this we proposed a metrics based solution that enhances the idea of digital trustworthiness certificates and that can be easily used by end-users, developers and marketplace owners. We believe that this method goes in the right direction to solving the confusion that surrounds the complex software procurement decision making process. We plan to further develop this idea with a more in-depth investigation on metrics that are appropriate to specific business contexts.

Acknowledgment

This work was partly supported by the EU-funded project OPTET [grant no. 317631].

References

[1] Towards a trustworthy service marketplace for the future internet. In: lvarez, F., Cleary, F., Daras, P., Domingue, J., Galis, A., Garcia, A., Gavras, A., Karnourskos, S., Krco, S., Li, M.S., Lotz, V., Mller, H., Salvadori, E., Sassen, A.M., Schaffers, H., Stiller, B., Tselentis, G.,

Turkama, P., Zahariadis, T. (eds.) The Future Internet, Lecture Notes in Computer Science, vol. 7281 (2012)

[2] Anisetti, M., Ardagna, C., Damiani, E.: Toward certification of services. In: International Workshop on Business System Management and Engineering (BSME 2010). Malaga, Spain (June 2010)

[3] Barrera, D., van Oorschot, P.: Secure software installation on smartphones. IEEE Security & Privacy 9(3), 42–48 (2011)

[4] Bezzi, M., Kaluvuri, S.P., Sabetta, A.: Ensuring trust in service consumption through security certification. In: Proceedings of the International Workshop on Quality Assurance for Service-Based Applications. pp. 40–43. QASBA '11, ACM, New York, NY, USA (2011), `http://doi.acm.org/10.1145/2031746.2031758`

[5] Gilbert, P., Chun, B.G., Cox, L.P., Jung, J.: Vision: Automated security validation of mobile apps at app markets. In: Proceedings of the Second International Workshop on Mobile Cloud Computing and Services. pp. 21–26. MCS '11, ACM, New York, NY, USA (2011), `http://doi.acm.org/10.1145/1999732.1999740`

[6] IDC: Enterprise it in the cloud computing era. Tech. rep., International Data Corporation IDC (2008)

[7] ITSEC: Common criteria for information technology security evaluation. Tech. rep., Commission of the European Communities (1991)

[8] Lotz, V., Kaluvuri, S.P., Di Cerbo, F., Sabetta, A.: Towards security certification schemas for the internet of services. In: 5th International Conference on New Technologies, Mobility and Security, Istanbul, Turkey, NTMS 2012, May 7-10, 2012. pp. 1–5 (2012), `http://dx.doi.org/10.1109/NTMS.2012.6208771`

[9] McDaniel, P., Enck, W.: Not so great expectations: Why application markets haven't failed security. Security & Privacy, IEEE 8(5), 76–78 (2010)

[10] Mohammadi, N.G., Paulus, S., Bishr, M., Metzger, A., Koennecke, H., Hartenstein, S., Pohl, K.: An analysis of software quality attributes and their contribution to trustworthiness. In: CLOSER 2013 - Proceedings of the 3rd International Conference on Cloud Computing and Services Science, Aachen, Germany, 8-10 May, 2013. pp. 542–552 (2013)

[11] Mohammadi, N.G., Paulus, S., Bishr, M., Metzger, A., Könnecke, H., Hartenstein, S., Weyer, T., Pohl, K.: Trustworthiness attributes and metrics for engineering trusted internet-based software systems. In: Cloud Computing and Services Science - Third International Conference, CLOSER 2013, Aachen, Germany, May 8-10, 2013, Revised Selected Papers. pp. 19–35 (2013)

[12] Paulus, S., Mohammadi, N.G., Weyer, T.: Trustworthy software development. In: Communications and Multimedia Security - 14th IFIP TC 6/TC 11 International Conference, CMS 2013, Magdeburg, Germany, September 25-26, 2013. Proceedings. pp. 233–247 (2013)

[13] Pazzaglia, J., Lotz, V., Cerda, V.C., Damiani, E., Ardagna, C., Gürgens, S., Maña, A., Pandolfo, C., Spanoudakis, G., Guida, F., et al.: Advanced security service certificate for soa: Certified services go digital. In: ISSE 2010 Securing Electronic Business Processes, pp. 151–160. Springer (2011)

View publication stats