Trust in business processes

Nicolai Kuntze, Andreas U.Schmidt, Zaharina Velikova, Carsten Rudolph
Fraunhofer Institute for Secure Information Technology SIT
Rheinstrasse 75, 64295 Darmstadt, Germany
{nicolai.kuntze|andreas.schmidt|zaharina.velikova|carsten.rudolph}@sit.fraunhofer.de

Abstract

The service oriented architectures (SOA) approach designing the interaction between various business entities leads to complex and highly automated relations between formerly separate organisations. In particular the automation of these processes demands a high level of trust of each partner in the underlying IT infrastructure and the results produced. This paper discusses the security and trust implications and shows how to utilize Trusted Computing within the SOA approach introducing means for integrity and authenticity.

1. Introduction

Service Oriented Architecture (SOA) [8] is a paradigm for creating and using business processes using services as the basic functional elements. These are logically linked by workflows defining the interaction and timing between them. In most cases services are servers which are serving requests using a web service interface orchestrated e.g. by a Workflow Management System (WFMS). By serving automatically requests these servers are creating value for their owners as well as for the customers. A standard for specifying such workflow processes is the Web Services Business Process Execution Language (WSBPEL) [2], or BPEL in short.

Documentation of the processes within the value cration chain are common in the paper based world are common and lead to standards like the IT Infrastructure Library (ITIL) [11] or ISO 9000 [18]. As for any other service it is necessary to ensure and validate that assets and artifacts within the architecture are acting as expected and maintaining a certain level of quality. The trust of the customer in the offered service depends on this. Trusted Computing offers means to verify the state of the service before a contract is concluded and can also be applied as a mean for ex post verification of a certain state of the service during process execution.

In section 2 existing work in the field of security for workflow architectures is discussed. Relevant security issues for these architectures are explored in section 3 and the necessary trusted computing basics are introduced in section 4. The core concepts are presented in section 5 and the paper concludes with an outline of their deployment, distinguishing centralized and distributed business processes.

2. Related Work

Web services based on WSDL, SOAP, and UDDI developed to the standard designing the interaction between different services [1] and by this they are providing the basis for service oriented architectures. The WS-* family of protocols cover a variety of middleware related topics like messaging [15], transactions [7, 6], and security [16]. Moreover, best practice approaches exists providing patterns and tools for the application of security and trust mechanisms for these protocols [19].

Short change cycles in the industry require flexible solutions with respect to integration and of new services and processes. Process definition languages like BPEL [2] are used together with additional concepts as defined for example by the Enterprise Service Bus [4]. These workflows defined by BPEL are abstractions of business processes and have inherent security requirements with respect to their order of execution and the data they contain. In [21] a decentralized framework for workflow execution that ensures exception safety and considers security issues is proposed. Secure execution orders are also considered in [3] supporting the case of a decentralised system by proposing a container structure with authentication mechanisms for data access. A similar approach is shown by [5] defining a container and environment to add certain middleware requirements like persistence, security, and reliable messaging dynamically to the BPEL description. This is done by using an aspect oriented approach similar to AspectJ [13]. In the centralised scenario with a dedicated workflow management system as the central authority existing WS-* protocols can offer a certain level of security on the link level for example by using the above mentioned WS-Security. Means of documentation of the involved parties to gain non-repudiation are not within the focus of the development. [14] shows how to describe non-repudiation protocols in BPEL.

On the level of document security the TransiDoc Project [17] proposed a scheme to extend the concept of tranformations of paper documents to their digital counterparts. By this each change to a document like translation can be digitally signed and non-repudiation of these processes is achieved.

3. Security requirements

A wide variety of security requirements can occur for workflows. These requirements are either identified based on static trust relations on the organisational, contractual, or IT infrastructure level, or for single services within the workflow. However, in particular in the case of decentralised workflows it is not trivial to derive the correct combination of security mechanisms that satisfies all requirements of the different partners involved in the workflow. Therefore, it is necessary to precisely define security requirements on the level of the workflow itself. A full formal framework for the specification of security requirements for workflows is out of the scope of this paper, but we can refer to the generic framework for security requirements by Gürgens et al. [10].

Aspects of information governance like authenticity of an entity performing a particular service in a workflow, integrity or confidentiality of data that is transported between entities involved in the workflow, or the enforcement of particular (distributed) sequences of actions (or services) in the workflow define the security requirements for the actual workflows. To fulfill these requirements a combination of security mechanisms for single services as well as for the overall workflow can be required.

We distinguish four classes of properties: authenticity, non-repudiation, confidentiality and enforcement of workflow sequences and introduce them in the following.

Authenticity In general, authenticity of a particular action is satisfied for one partner P in a workflow if P can deduce that this action has occured from its knowledge about the global behaviour of the workflow system (including all partners and also including possible malicious behaviour) and the view of the current beaviour that the action has occured. Stronger authenticity requirements can restrict the occurence of the action to be authentic to the current instance of the workflow or even to a paticular phase of the current instance of the workflow. In a centrally controlled workflow, a workflow engine or in service-oriented architectures an enterprise service bus are in a position to control the workflow and reliably log and report all actions in the workflow. Furthermore, central control also obviously allows a workflow engine to enforce particular sequences or workflow behaviour. Thus, in centralised workflows authenticity can usually be reduced to authenticity of single service executions. In distributed decentralised workflows all players need to contribute to the control and enforcement of authenticity.

Non-repudiation Non-repudiation is strongly related to authenticity but in addition requires that one partner can prove to other partners that a particular action or sequence of actions has occured. Thus, the partner executing those action cannot repudiate this execution. Again, a trusted central entity could simply collect evidence (e.g. digital signatures) and in case of dispute provide this evidence for dispute resolution. In a decentralized workflows partner have to collect evidence and might even have to rely on other partners for the enforcement of non-repudiation requirements. If security is based on such a mutual trust between partners in a distributed workflow, overall trust requirements have to be considered in the design of the workflow and the security policies.

Confidentiality A large variety of information can be required to be confidential in a workflow. This information can include data transferred or computed within the workflow, identity information of the partners involved, order of execution of workflow actions, parts of the workflow specification, security policies, cryptographic keys, and audit trails or evidence collected for non-repudiation. The electronic process slip introduced in Section 5 can provide security mechanisms for a large part of this variety of requirements. It can also be used as a basis to reason about the requirements in order to find conflicting requirements or contradictions. Formalisation of confidentiality is often based on non-interference and information flow properties formalizing the requirement that the occurence of some actions cannot interfere with the behaviour of those partners not allowed to gain knowledge about these actions. Non-interference properties can be used to formalize confidentiality of workflow data towards exernal attackers. However, non-interference properties are not suitable for properties within a workflow where a partner might know about the occurence of some or all workflow actions but might not be allowed to know the values of all parameters in the workflow data. The notion of *parameter confidentiality* [9] is more suitable to formalise confidentiality requirements for workflows.

Enforcement of sequences Security of a workflow very often depends on the order of actions in the workflow. A particular action can depend on a number of other actions occured before or a particular binding phase can only be finished if all goals of the involved partners are satisfied. Again, a trusted central entity can enforce these properties and reduce these rather complex requirements to requirements for single actions or services within the workflow. It should be noted that this reduction is not always as easy as in the case of authenticity of particular actions. It might require a combination of combination of mechanisms for confidentiality, authenticity and non-repudiation.

The lack of central control in distributed workflows increases the complexity of the problem. It might be even impossible to achive the most efficient decentralisation if control has to be given to an untrusted entity and if it is necessary to actually prevent a deviation from the workflow specification. Additional communication between partners might be required to release confidential data that is necessary to continue with the workflow. A realisation is easier if it is satisfactory to detect violations of the requirement after they have occured. Many of these different combinations of security requirements can be realised by the electronic process slip introduced in the following section.

4. Trusted Computing functionalities

The Trusted Computing Group is the main industrial effort to standardize TC technology. Trust as defined by the TCG means that an entity always behaves in the expected manner for the intended purpose. The trust anchor, called Trusted Platform Module (TPM), offers various functions related to security. Each TPM is bound to a certain environment and together they form a trusted platform (TP) from which the TPM cannot be removed.

Through the TPM the Trusted Platform (TP) gains a cryptographic engine and a protected storage. Each physical instantiation of a TPM has a unique identity by an Endorsement Key (EK) which is created at manufacture time. This key is used as a base for secure transactions as the Endorsement Key Credential (EKC) asserts that the holder of the private portion of the EK is a TPM conforming to the TCG specification. The EKC is issued as well at production time and the private part of the key pair does not leave the TPM. There are other credentials specified by the TCG which are stating the conformance of the TPM and the platform for instance the so called platform credential. Before a TPM can be used a take ownership procedure must be performed in which the usage of the TPM is bound to a certain user. The following technical details are taken from [20].

The TPM is equipped with a physical random number generator, and a key generation component which creates RSA key pairs. The key generator is designed as a protected capability, and the created private keys are kept in a shielded capability (a protected storage space inside the TPM). The shielded capabilities protect internal data structures by controlling their use. Three of them are essential for applications. First, key creation and management, second the ability to create a trust measurement which can be used to assert a certain state toward a, remote party, and finally sealing methods to protect arbitrary data by binding it (in TCG nomenclature) to TP states and TPM keys. For the TPM to issue an assertion about the system state, two attestation protocols are available. As the uniqueness of every TPM leads to privacy concerns, they provide pseudonymity, resp., anonymity. Both protocols rest on Attestation Identity Keys (AIKs) which are placeholders for the EK. An AIK is a 1024 bit RSA key whose private portion is sealed inside the TPM. The simpler protocol Remote Attestation (RA) offers pseudonymity employing a trusted third party, the Privacy CA (PCA), which issues a credential stating that the respective AIK is generated by a sound TPM within a valid platform. The system state is measured by a reporting process with the TPM as central reporting authority receiving measurement values and calculating a unique representation of the state using hash values. For this, the TPM has several Platform Configuration Registers (PCR). Beginning with the system boot each component reports a measurement value, e.g., a hash value over the BIOS, to the TPM and stores it in a log file. During RA the communication partner acting as verifier receives this log file and the corresponding PCR value. The verifier can then decide if the device is in a configuration which is trustworthy from his perspective. Apart from RA, the TCG has defined Direct Anonymous Attestation. This involved protocol is based on a zero knowledge proof but due to certain constraints of the hardware it is not implemented in current TPMs.

AIKs are crucial for applications since they can not only be used, according to TCG standards, to attest the origin and authenticity of a trust measurement, but also to authenticate other keys and data generated by the TPM. Before an AIK can testify the authenticity of any data, a PCA has to issue a credential for it. This credential together with the AIK can therefore be used as an identity for this platform. The protocol for issuing this credential consists in three basic steps. First, the TPM generates an RSA key pair by performing the TPM_MakeIdentity command. The resulting public key together with certain credentials identifying the platform is then transferred to the PCA. Second, the PCA verifies the correctness of the produced credentials and the AIK signature. If they are valid the PCA creates the AIK credential which contains an identity label, the AIK public key, and information about the TPM and the platform. A special structure containing the AIK credential is created which is used in step three to activate the AIK by executing the TPM_ActivateIdentity command.

Using the AIK as a signing key for arbitrary data is not possible due to the given constrains of the TPM. To sign data using the AIK it is therefore necessary to employ an indirection using a TPM generated signing key and certify this key by signing it with an AIK. By certifying a specific key the TPM makes the statement that ”this key is held in a TPM-shielded location, and it will never be revealed”. For this statement to have veracity, a challenger or verifier must trust the policies used by the entity that issued the identity and the maintenance policy of the TPM manufacturer. This indirection creates to each AIK a certified key (by the namely AIK) that can be used for signing data. This scheme is shown in detail in [12].

5. Trusted Process Concepts

Trust into a particular service can be established at two points during execution of workflows which can be supported by Trusted Computing. Before a single service is put in charge the calling entity can perform an attestation of the service platform. Alternatively this can be done by the service platform during the execution of the request adding the attestation values to the results.

If the service is invoked this service can perform the attestation and attach the respective data structure to the data produced as a conformity statement.Both approaches result in a conformity statement which vouches for the service platform if it is attached to the data. Based on these statements the requested authenticity and non-repudiation of the results is based on. The first additionally leads to a secured execution flow allowing only for attested services taking part. Additional means are required to bind the result to the attestation values offered before hand to the invoking entity. Otherwise it is not stated that the offered attested state was existing as the result was computed. It is required that the producer of the result takes it into possession and vouches for the integrity.

Using Trusted Computing each service has a unique identity represented by the EK credential bound to the hardware. Building a trusted business process on top on this identity as a prerequisite it is required that the service provider takes possession of this particular platform and declares this possession accordingly. Basically this take ownership process is an organisational process which is to be which is to be implemented and supported by protocols.

Trust in an entity is established in the Trusted Computing Groups reference architecture by using the ability to perform an attestation process. Basically, such an attestation consists out of a log book containing information about all relevant elements the system consists of beginning with the boot process up to the operating system. Whereas the attestation of the boot process is well understood and projects like TrustedGrub are already providing tools to establish a Trusted Boot process attestation of the running operating system is still a ongoing research topic. The challenge is that every change in the actual status of the running processes has to be logged which results in long and complex data structures. As an alternative approach we favor the attestation of a whole image of the operation system including the service before the system is started.

Providing a description on the state of the service requires to attach authenticated attestation data to the result of the service. Each attestation is signed by an AIK as introduced in section 4. Building one coherently signed data package including the result computed by the service as well as the attestation values it also required to sign the whole package. As this is not possible as the AIK can only sign TPM created data it is necessary to apply the indirection introduced also in section 4. Such a data packe states that it is created by a certain service in a testified state and that this service is owned by an organisation operating the PCA that issued the AIK certificate.

Traditional paper based workflows use signed reports carrying the relevant information. These reports are produced wherever it is required to grant the owner the access to certain services. This token based concept can also be applied to digital business processes by establishing a t rustworthy data structure as a technical basis for the workflow. It is a digital form of a report description of the execution of a business process. This token transfers all kind of data between the participants in the process and stores their signatures as proofs for the executed tasks by recording every single step of the whole process. To prevent the invocation of malicious requests or the receiving any kind of data from a malicious service TPMs could be used, which consist of a hardware module that functions as a root of signature verification. We define the following subcontainers of the process slip.

- **Data** - Here is stored the input and output data between the consecutive steps of the process. According to the needed security level some (or all) data can be encrypted for the corresponding receiver to ensure confidentiality and integrity of the data. Instead of using the traditional PKI techniques, where a public/private keypair and a public key certificate is issued for each partner, we can use private and public part of the corresponding TPM's EK. In this case the encryption and the decryption of the data as well as the verification of signatures is carried on by the TPM. Since the private part of the TPM's key pair does not leave the TPM we can be sure that each time these tasks are performed from the same machine.

- **Audit Data** - In the audit trail data subcontainer each involved partner should write according to the workflow or security definition log data for the recently performed tasks. Each service is only allowed to add process information and documentation data relevant for the service's assignments and their output. These entries must be signed by the corresponding system and/or responsible entity in order to ensure the integrity of the inserted information. By adding an authenticated log of each step to the final result it is later on possible to track down the process and assign responsibility. This provides for the base offering non-repudiation as the main advantage.

- **Security Policies** - Security policies are situated in a separate data structure which is logically linked to the workflow definition. They specify security boundaries for each step or sequence of steps in the workflow such as the permitted activities that a given partner can apply on a specified data.
- **Workflow Description** - This static data structure is not changed during the execution of the workflow and can either be a full description of the workflow or be a reference to a location where the definition is kept.

When the business process ends and all sections are filled in correspondingly, our process slip structure can be sent to a central verification unit, which can verify the proper execution of the process. This approach can be applied in a centralised and distributed business processes. However, there exist some main differences between both cases.

5.1 Trusted process slip base data structure

There are three requirements on trusted process slips, i.e., *seals* obtaining secure, auditable information on the system(s) on which the processes are performed:

- **Binding to the technical system.** The system and its current state at the processing time must be uniquely identified.
- **Binding to the single processing steps.** The mentioned information must be bound to the particular processing steps.
- **Binding to the target.** The mentioned information must be bound to the processed data.

This complete set of bindings of the audit data to the processed data yields the desired *attestation* of the process slip which in turn obtains probative force to it. A trusted platform containing a hardware security anchor and capable of performing secure, or authenticated boot processes and to remotely attest its state is a conceptual prerequisite for trusted process slips. It is now straightforward to devise a structure which extends transformation seals as used in [17] to yield the bindings. This is shown in Figure 1. During processing in a single step, the system collects state information, for instance the stored measurement log of the underlying TP and adds them to the pertinent audit data piece — the *processing report*, which in particular contains automatically generated protocols of the process. The TP also adds a security digest, e.g., PCR values to this data for later verification. The single steps as well as the whole transformation report are then secured by machine signatures, e.g., based on AIKs as described above. The outermost of these machine signatures establishes the binding to the converted contents. The seal is completed by a signature of the responsible party.

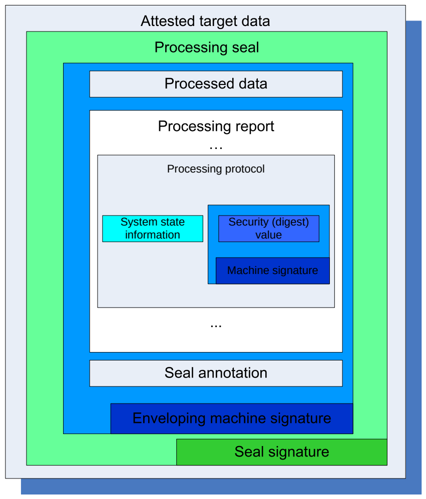

Figure 1. Information structure and signatures of a seal for trusted process slips.

5.2 Trusted process slip in a centralized business process

In orchestration (centralized business process), a central process takes control of the involved services and coordinates the execution of different operations on the services involved in the operation. The involved services do not k̈now(̈and do not need to know) that they are involved in a composition process and that they are taking part in a higher-level business process. Only the central coordinator of the orchestration is aware of this goal, so the orchestration is centralized with explicit definitions of operations and the order of invocation of Web services. Therefore the process slip is structured and used only by the coordinater of the current process.

Depending on the level of security and the security policies the central service sends the input data encrypted for the corresponding service receiver. The central coordinator should be able to interpret and enforce the specified security policies and to provide accordingly the required data access of the services, deployed in the process. After invoking an operation or replying to a service the central manager inserts a task entry in the Audit Data section for the performed action. In order to ensure non-repudiation of the receiver, for each fulfilled task each service must send a c̈onfirmation(̈a digital signature) for the fullfilled task which is inserted by the coordinator in the Audit Data section of the process slip.

5.3 Trusted process slip in a distributed business process

Choreography, in contrast to the orchestration, does not rely on a central coordinator. Rather, each service involved in the choreography knows exactly when to execute its operations and with whom to interact. Choreography is a collaborative effort focusing on the exchange of messages in public business processes. All participants in the choreography need to be aware of the business process, operations to execute, messages to exchange, and the timing of message exchanges. There may exist multiple engines, each executing a composite service specification (a small part of the original process specification but complete in itself) at distributed locations.

In a distributed environment the process slip structure is initialized by the initiator of the whole process and travels from manager to manager[1]We treat the separate services also as managers if they are not entitled in a subprocess.). Each separate manager should be able to interpret the process slip structure as explained in the previous subsection. Additionally the process slip can be initilized in such a way that the separate entries in the security policies and the worflow definition subsection may be encrypted for the corresponding service machines. In this way each service in the workflow is limited in his knowledge to the absolute minimum of knowledge regarding the workflow he is working in.

References

[1] G. Alonso, F. Casati, H. Kuno, and V. Machiraju. *Web Services: Concepts, Architecture and Applications*. Springer Verlag, 2004.

[2] A. Alves, A. Arkin, S. Askary, C. Barreto, B. Bloch, F. Curbera, M. Ford, Y. Goland, N. Kartha, C. K. Liu, S. Thatte, P. Yendluri, A. Yiu, A. Guízar, R. Khalaf, D. König, M. M. V. Mehta, and D. van der Rijn. Web services business process execution language version 2.0, April 2007.

[3] J. Biskup, B. Carminati, E. Ferrari, F. Muller, and S. Wortmann. Towards secure execution orders for compositeweb services. *Web Services, 2007. ICWS 2007. IEEE International Conference on*, pages 489–496, 9-13 July 2007.

[4] D. A. Chappell. *Enterprise service bus*. 2004.

[5] A. Charfi and M. Mezini. An aspect-based process container for bpel. In *AOMD '05: Proceedings of the 1st workshop on Aspect oriented middleware development*, New York, NY, USA, 2005. ACM.

[6] D. L. (Eds.). WS-BusinessActivity, 2004.

[7] D. L. (Eds.). WS-Transaction, 2004.

[8] T. Erl. *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall PTR Upper Saddle River, NJ, USA, 2005.

[9] S. Gürgens, P. Ochsenschläger, and C. Rudolph. Abstractions preserving parameter confidentiality. In *European Symposium On Research in Computer Security (ESORICS 2005)*, pages 418–437, 2005.

[10] S. Gürgens, P. Ochsenschläger, and C. Rudolph. On a formal framework for security properties. *International Computer Standards & Interface Journal (CSI), Special issue on formal methods, techniques and tools for secure and reliable applications*, 2005.

[11] A. Hochstein, R. Zarnekow, and W. Brenner. Itil as common practice reference model for it service management: Formal assessment and implications for practice. *eee*, 00:704–710, 2005.

[12] N. Kuntze, D. Mähler, and A. U. Schmidt. Employing trusted computing for the forward pricing of pseudonyms in reputation systems. In K. Ng, A. Badii, and P. Bellini, editors, *Axmedis 2006 : proceedings of the 2nd international Conference on automated production of cross media content for multi-channel distribution; Workshops Tutorials Applications and Industrial*, Atti del Convegno, Leeds (UK), pages 145–149. Firenze University Press, 2006.

[13] R. Laddad. *AspectJ in Action: Practical Aspect-Oriented Programming*. Manning Publications Co., Greenwich, CT, USA, 2003.

[14] J. P. T. M Bilal, M. Thomas, and S. Abraham. Fair bpel processes transaction using non-repudiation protocols. In *SCC '05: Proceedings of the 2005 IEEE International Conference on Services Computing*, pages 337–342, Washington, DC, USA, 2005. IEEE Computer Society.

[15] OASIS. WS-Reliability, 2004.

[16] OASIS. WS-Security version 1.0, 2004.

[17] J. Piechalski and A. U. Schmidt. Authorised translations of electronic documents. In H. S. Venter, J. H. P. Eloff, L. Labuschagne, and M. M. Eloff, editors, *Peer-reviewed Proceedings of the ISSA 2006 From Insight to Foresight Conference*. Information Security South Africa (ISSA), 2006.

[18] R. Randall. *Randall's practical guide to ISO 9000 : implementation, registration, and beyond*. Addison-Wesley, 1995.

[19] M. Tatsubori, T. Imamura, and Y. Nakamura. Best-practice patterns and tool support for configuring secure web services messaging. In *ICWS '04: Proceedings of the IEEE International Conference on Web Services*, page 244, Washington, DC, USA, 2004. IEEE Computer Society.

[20] TCG. Trusted platform module specification (tpm), version 1.2, revision 94. https://www.trustedcomputinggroup.org/specs/TPM, 2005.

[21] H.-H. Vagts. Control flow enforcement in workflows in the presence of exceptions. Master's thesis, TU Darmstadt, Dec. 2007.

[1](