Towards a Security Framework for a WS-HumanTask Processor

Gerhards, M., Skorupa, S., Sander, V., Pfeiffer, P.
FH Aachen University of Applied Sciences
Juelich, Germany
{M.Gerhards|Skorupa|V.Sander}@fh-aachen.de
Pascal.Pfeiffer@dialup.fh-aachen.de

Belloum, A.
University of Amsterdam
Amsterdam, Netherlands
A.S.Z.Belloum@uva.nl

Abstract—**BPEL4People and WS-HumanTask (WS-HT) specify models to integrate human resources into business processes. The emerging use of these standards will result in more challenging authorization decisions in the context of the execution of workflows. Compared to a classical scenario where a workflow management system controls the mapping of tasks to a specific set of resources, human interaction introduces an additional mapping scheme in which humans were separately mapped to tasks. Basically the authorization framework needs to be adapted from a push-based model to a push-/pull-based approach. This short paper introduces the concepts of a security framework for a WS-HT implementation. It presents a generic framework that supports a pull-based work distribution strategy in distributed environments with the help of a task repository that mediates tasks between resources and workflow instances.**

WS-HumanTask; Web Service Security; Secure Token Service; Access Control; Authorization; Authentication

I. INTRODUCTION

Motivated by the demand to integrate human interactions consistently into e-Science, we extended the UNICORE workflow system. By the realization of an actor-driven approach for workflow execution in distributed environments [5], we integrated the missing pull patterns of the well known workflow resource patterns [4] into the UNICORE Workflow System [6]. This was done with the help of a service-enabled task repository deployed to the UNICORE hosting environment. In this extension we followed the path that has been established in 2005, where the white paper "WS-BPEL Extension for People – BPEL4People" was published by IBM and SAP [3]. There, first efforts have been made towards a consistent model to describe the interaction between human beings and business processes. The published proposal strongly relates to the popular industry standard WS-BPEL, which is commonly used to model processes by the orchestration of web services. By now, on the basis of this white paper with BPEL4People and WS-HT two new specifications have been developed, that are expected to be standardized by OASIS. While the BPEL4People specification is closely coupled to the WS-HT proposal, their scopes and intentions are described first. Thus, BPEL4People provides an extended schema to the existing WS-BPEL standard to support a broad range of scenarios that involve people within business processes. However, the new capabilities are limited, because the most essential contribution is the extension of an "Activity" to a "PeopleActivity" enabling the definition of tasks within a BPEL process that have to be executed by human resources. In contrast to that, WS-HT comes up with a large amount of features, which range from the task definition, distribution, and allocation to task lifecycle management and coordination protocols to mediate between different participants. In August 2010, these two specifications were published as committee versions, which mean that the technical work is viewed as completed. The publication as committee version motivated the migration of our task repository, which was implemented and integrated into the UNICORE Grid middleware within the scope of the HiX4AGWS project [5] to WS-HT. As a consequence of this decision, there was a demand for the design of an according authorization framework. The short paper is organized as followed. First of all the WS-HT specification is introduced including their concepts related to security issues. After this, in section III, the resulting requirements for development of a proper security framework are pointed out. In section IV the suitable technologies to come up with the discussed requirement are described, before summarizing the paper.

II. THE WS-HUMANTASK SPECIFICATION

In this section, the scope and the features of the WS-HT specification is described superficially. First of all the core concepts are introduced, before focusing the security concerns. The overall architecture proposed by the WS-HT specification is represented in Figure 1 as a simplified version reduced to the essential elements. The blue framed components belonging to the task processor are introduced in more detail within the specification. WS-HT provides an XML schema to specify available data types and structures to describe task definitions referred to as abstract tasks hereafter. Such abstract tasks contain information about properties that describe particular conditions related to the execution, for example which people should be assigned at what point of time, how a task should be represented to particular clients or what escalation routines should be used if deadlines are missed. An abstract task is a conceptual entity that is modeled explicitly. Particularly, it is possible for a modeler of an abstract task to define a task-specific interface in form of a WSDL port type, which should be used to create task instances. WS-HT does not specify how a task parent interacts with an abstract task.

This domain-specific interface is typically used by a task parent, which might be a workflow or a business process.

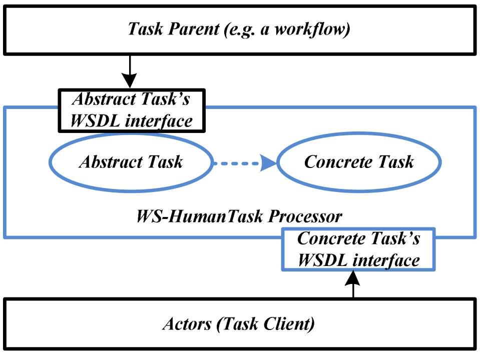
Task Parent (e.g. a workflow)
Abstract Task's WSDL interface
Abstract Task
Concrete Task
WS-HumanTask Processor
Concrete Task's WSDL interface
Actors (Task Client)

Figure 1. WS-HumanTask architecture proposal.

However, the invocation is usually directly concerned with the instantiation of the corresponding abstract task. The behavior of the resulting task instances, which are referred to as concrete tasks, is specified in the WS-HT proposal, i.e. they run through well defined states during their lifecycles (Figure 2). The different states and their transitions can be influenced by actors, who are authorized to perform operations that are well defined by the specification as well. An excerpt of the available task operations is listed in the left column of TABLE 1.

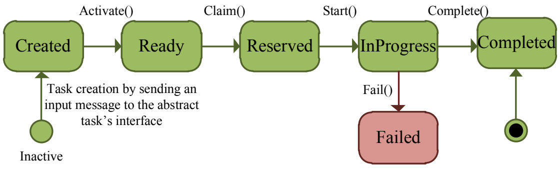

Figure 2. The state diagram shows the lifecycle of a concrete task.

WS-HT provides a WSDL definition that specifies all task operations within a particular port type. The corresponding endpoints have to be provided by a task processor implementing WS-HT. The clients that use these endpoints are referred to as actors in the following. Most specified operations get passed a task identifier as an argument to refer to the corresponding concrete task. In summary, there exist three participants that need to be considered in an authorization framework: Task parent, Task Processor, and Actors. A task parent is interested in the execution of concrete tasks. Therefore, a task parent has to invoke the particular interface of the corresponding abstract task by sending an input message containing e.g. domain-specific context data to the related endpoint. It is assumed that this interface and the description of the abstract task are deployed in advance. The task processor provides the corresponding endpoints and triggers the instantiation of abstract tasks, which always results in the creation of a concrete task. Additionally, the task processor manages the lifecycles of these concrete tasks according to the diagram of Figure 2. The actors are able to perform task operations on concrete tasks by using the specified concrete task's interface provided by the task processor. Actors produce the desired outcome to come up with the requirements of the task parent. The focus of this paper is on introducing concepts for an authorization framework for actors that wants to perform task operations by using the concrete tasks' interface.

So far, we have identified three relevant entities that need to be considered for authorization decisions. We will now discuss the related aspects in more detail. During the lifecycle of a concrete task different actors are allowed to perform certain operations or to view particular task specific data. The modeler of an abstract task as well as the task parent, who triggers the creation of a corresponding concrete task, should be able to influence the set of possible actors. Therefore, WS-HT provides a specific XML container that represents this relation within an abstract task. This container is called people assignments. People assignments are generally used to enable the modeler of an abstract task to define which human resources are assigned to which particular roles. Such roles that are related to different authorization rules are called generic human roles according to the WS-HT proposal. The existing set of generic human roles is represented in Figure 3 as top layer. The related authorizations are additionally illustrated in TABLE 1. The sign "+" means that an actor who is assigned to this generic human role is allowed to perform the operation on the concrete task. The "-" sign indicates that the operation is not available to that particular role and "MAY" means that it is up to the implementation.

TABLE I. TASK OPERATIONS AND AUTHORIZATIONS

Role	**Task Operations that cause state transitions**				
Operation	***Task Initiator***	***Task Stakeholders***	***Potential Owners***	***Actual Owner***	***Business Admin***
activate	+	+	-	-	+
claim	-	MAY	+	-	MAY
complete	-	MAY	-	+	MAY
delegate	MAY	+	MAY	+	+
fail	-	MAY	-	+	MAY
forward	MAY	+	MAY	+	+
nominate	MAY	-	-	-	+
release	-	MAY	-	+	MAY
resume	MAY	+	MAY	MAY	+
skip	+	+	MAY	MAY	+
start	-	MAY	+	+	MAY
stop	-	MAY	-	+	MAY
suspend	MAY	+	MAY	MAY	+

The modeler of an abstract task can assign human resources to these generic human roles by defining people assignments, represented by the second layer. The people assignment evaluation represented by the layer in between can be executed at different points of time. The modeler is able to determine this time, which is called evaluation time. Furthermore, he can decide which data should be used for evaluation. By using literal assignments, the modeler can directly assign users or groups to particular roles at deployment time of the abstract task. The use of expressions enables the modeler to defer the evaluation time until corresponding concrete tasks are instantiated from the abstract description. In the following, we refer to this time as creation time. So, it is for example possible to relate to a certain part of the input message of a task parent by an XPath expression in the abstract task. This would enable a task parent to limit the set of possible actors. So, depending on the definition of the people assignments the evaluation time can be influenced. People assignments can even be changed during runtime of a concrete task by using operations like delegate or forward.

The data container, which can be assigned to a generic human role is also fixed by the specification by an organizational entity. An organizational entity can contain multiple users or groups as well as a mixture of both. So, the evaluation of people assignments should always result in organizational entities assigned to generic human roles independent to the evaluation time. Therefore, at deployment time, each people assignment is bound to a so called people query, which would be evaluated at creation time. The evaluation of such people queries, which is needed to assign real persons to generic human roles, is out of scope of the specification. Therefore, a security framework should provide a people query evaluator that cares for the binding to user directories. Beside the identity management and the binding of the identities to generic human roles by evaluating people queries, the security framework should act as policy decision point (PDP). This means the authorization decision for expected actors should be made by a component of the security framework. It should be explicitly outsourced from the task processor. Further details and requirements for the security framework will be discussed in the next section.

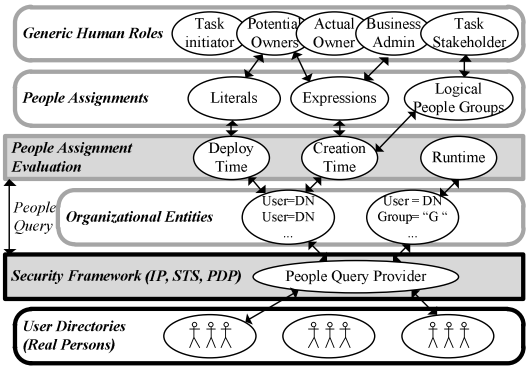

Figure 3. Different layers for assigning actors to concrete tasks according to WS-HumanTask. Whereby, black framed layers are out of scope of the specification and therefore implementation specific, the grey framed layers are concepts that are defined by the WS-HumanTask specification.

III. REQURIMENTS FOR A PROPER SECURITY FRAMEWORK FOR WS-HUMANTASK

In this section the requirements for the design of a security framework for WS-HT should be elaborated. The requirements are related to the security of the concrete task's interface (Figure 1), which is used by actors to perform task operations. There are several requirements that depend on the specification as well as requirements that are specific for our implementation. Some basic and important demands are listed in this section:

(01) **User dictionary and authentication as prerequisite:** Providing and querying a user directory is out of scope of WS-HT, but it is needed to manage available user identities, their corresponding roles and attributes. So, a rather prerequisite requirement is to establish a user directory and, to support single-sign-on scenarios, an identity provider (IP) within the security framework. Users have to authenticate against the security framework for example by using a certificate. In this case we require an according public key infrastructure (PKI).

(02) **Third party support for authorization decisions:** The authorization should be performed by the security framework which means that the task processor is not required to make access decision for concrete task operations by itself. A token-based approach, where the security framework issue security tokens that contain access decision for particular operation(s) on specific concrete tasks appears to be an appropriate solution for this request. To support WS-HT, the management of the people assignments of each concrete task has to be done by the security framework considering the concepts of generic human roles and their fixed authorizations for task operations (TABLE 1).

(03) **Lifecycle support:** An access decision for specific operations on concrete tasks depends on the information specified in the people assignments by the modeler of an abstract task.

(04) Figure 3 shows that the specified people assignments might be evaluated at different points of time. Furthermore, they depend on the lifecycle of a concrete task. Hence, there must be a set of services in the security framework that enable the task processor to send notifications that contain updates for the people assignments of concrete tasks.

(05) **Invalidation of authorization attributes:** According to WS-HT, it should be possible to delegate and forward concrete tasks for specific users. If one of these operations is performed the corresponding behavior of a task processor induces changes in the people assignments. For example, if a potential owner forwards a concrete task to another user, he should be removed from the list of potential owners. The related rights to claim or to start a task have to be removed as well. Because of this, previously issued tokens have to be invalidated due to the updated access policies and people assignments.

IV. A SECURITY FRAMEWORK FOR WS-HUMANTASK

In section III, four major requirements have been identified that have to be considered by a security framework for a WS-HT specification. The fundamental decision we made during the formulation of the requirements is to implement an external security framework separately from the task processor. So, the authentication of actors as well as the authorization and the corresponding access decision should completely be done by the security framework by using a token-based approach. This results in an overall architecture represented in figure 4.

To come up with the prerequisite requirement (1), we establish a user dictionary by using LDAP (Lightweight Directory Access Protocol) and an identity provider by using Shibboleth. In a LDAP dictionary users and their attributes are managed and stored in a tree hierarchy. Because of the widespread use of these technologies, we skip further details on LDAP and Shibboleth at this point.

The second requirement (2) demands for performing the authorization and the authentication within the security framework outside of the task processor. As a consequence, we decide for a collocation of the task processor and the security framework in a single trust domain. This decision allows both, a task model specific authorization scheme and a human task protocol specific handling. Further on, this collocation offers the opportunity to use a token-based approach to handle authorization and access control of actors. This is directly related to the pattern of brokered authentication and authorization [7]. Core components of this pattern were based on the Kerberos authentication scheme. Tokens that are issued and signed by the security framework are the core indicator for any authorization decision of the task processor.

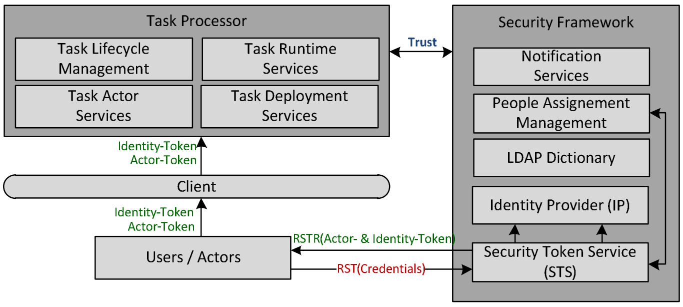

Figure 4. Overall architecture.

Figure 5 illustrates a simplified sequence diagram representing the single steps to get a token and perform a task operation. A user, who wants to become an actor for a particular concrete task is requested to first acquire a specific token for the intended operation. Therefore, he has to send a request for a security token message (RST) according to WS-Trust [8] to a security token service (STS), which is illustrated in figure 5. A STS is part of the security framework (see figure 4) and implements protocols that are specified in the WS-Trust standard. These protocols define formats and exchange patterns for issuing, renewing, and validating security tokens. Within the RST message, the actor has to request for a token that assures his identity (identity token) and a token with a SAML assertion containing a defined set of claims that can be used to authorize the user at the task processor (actor token). Claims are statements about users associating them with particular attributes (e.g. a role).

```
<wst:RequestSecurityToken>
 <wst:TokenType>
   http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.1#SAMLV2.0
 </wst:TokenType>
 <wst:RequestType>
   http://docs.oasis-open.org/ws-sx/ws-trust/200512/Issue
 </wst:RequestType>
 <wsp:AppliesTo xmlns:wsp="..." xmlns:wsa="...">
   <wsa:EndpointReference>
     <wsa:Address>
      http://fh.aachen.de/TaskProcessor/ConcreteTaskInterfaceService
     </wsa:Address>
   </wsa:EndpointReference>
 </wsp:AppliesTo>
 <wsp:PolicyReference xmlns:wsp="..."
   URI="http://fh.aachen.de/TaskProcessor/ConcreteTaskInterfaceService/IdentityPolicy"/>
</wst:RequestSecurityToken>
```

Figure 5. Excerpt for an example request for an identity token.

An example request for an identity token is represented in Figure 5. If such a request is received at the security framework, the STS first delegates the credentials to the identity provider who uses the LDAP dictionary to verify them. If the user identity can be authenticated, a corresponding identity token can be issued and signed, to be used for authentication against the task processor. The corresponding token contains a SAML authentication assertion.

In a second step, the authorization policies that are related to the identity are examined by querying the people assignments of the corresponding concrete task. The people assignments of each concrete task are managed within the security framework. Consequently, the actor has to put the identifier of the concrete task he intends to perform an operation on into the request for an actor token. With the help of this identifier the people assignments of the concrete task can be identified within the security framework and authorization rules can be checked according to TABLE1. A signed actor token can be issued if the user identity is authorized to perform the requested operation, because he is allowed to take over the particular generic human role. An actor token contains SAML assertions about a certain set of claims that the user requires to be authorized at the task processor. To specify this set of claims, we define a certain claims profile which is WS-HT specific. The actor and the identity token are included into the response message (RSTR) from the security framework to the requesting actor as signed SAML assertions. After the actor receives the response, he can send it together with the request message to the task processor. The task processor first checks the issuer of the tokens by using the public key of the security framework. If the issuer is trusted, the content of the tokens can be checked. Therefore, the identity token has to be validated and also the actor token must contain the required claims to perform the requested operation.

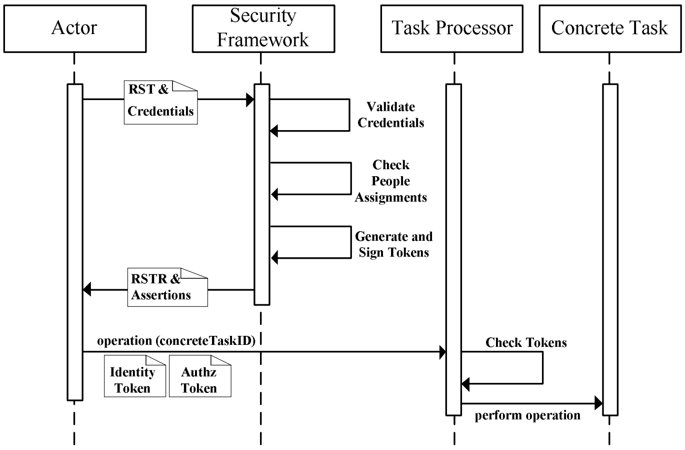

Figure 6. Sequence diagram that shows the process of performing a task operation from an actor's point of view.

To come up with requirement (05), we use an approach, where the assertions concerning the generic human roles are related to particular versions. These versions have to be kept synchronously on the task processor and the security framework for each generic human role of each concrete task. If a task is delegated or forwarded a notification is sent to the security framework, which updates the people assignments, increments the related generic human role version and responds these new version to the task processor. As a result, all token with older versions become invalid.

V. Summary and Outlook

This paper presents fundamental concepts to design a security framework for WS-HumanTask compliant task processor. Therefore, it first introduces the security model of the WS-HumanTask specification, before deriving important requirements concerning the security framework. Finally, an overview is given about the resulting architecture of the security framework realizing a token-based approach for authorizing actors to perform task operations. In the future it is planned to put these concepts into concrete terms by publishing a first version of the security framework.

REFERENCES

[1] Web Service HumanTask V1.1 Committee Specification, August 2010, http://docs.oasis-open.org/bpel4people/ws-humantask-1.1.pdf.

[2] WS-BPEL Extension for People V1.1 Committee Specification, August 2010, http://docs.oasis-open.org/bpel4people/bpel4people-1.1.pdf.

[3] WS-BPEL Extension for People – BPEL4People, A Joint White Paper by IBM and SAP, http://public.dhe.ibm.com/software/dw/specs/ws-bpel4people/BPEL4People_white_paper.pdf, July 2005.

[4] N. Russel and W.M.P. van der Aalst, “Workflow Resource Patterns as a Tool to Support BPEL4People Standardization Efforts”, BPTrends Journal, March 2008.

[5] S. Skorupa, F.Berretz, V. Sander, A. Belloum, M.Bubak: Actor-Driven Workfow Execution in Distributed Environments, Euro-Par 2010, Ischia, Italy, August 31 – September 3, 2010.

[6] Schuller et.al.: Chemomentum - UNICORE 6 based infrastructure for complex applications in science and technology. Euro-Par Workshops: Parallel Processing, Rennes, France, pp. 82-93, 2007.

[7] Brokered Authentication: Security Token Service (STS), http://msdn.microsoft.com/en-us/library/ff650503.aspx, 2005.

[8] Web Service Trust V1.3 Oasis Standard, March 2007, http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-os.pdf

[9] Web Services Security: SAML Token Profile 1.1, OASIS Standard, http://www.oasis-open.org/committees/download.php/16768/wss-v1.1-spec-os-SAMLTokenProfile.pdf, February 2006.

[10] The Apache Dictionary Project, Apache DS, http://directory.apache.org/, 2011.

[11] Shibboleth Identity Provider, version 2.2.1 http://shibboleth.internet2.edu/, 2011.