Security Considerations of the SchedSP Scheduling Application Service Provider

George Goulas, Vassilios Barkayannis, and Efthymios Housos

Computer Systems Laboratory
Department of Electrical and Computer Engineering
University of Patras, 26500 Patras, Greece

Abstract. In this paper, a framework for the secure provisioning of real world scheduling applications (SchedSP) as services over the Internet is presented. The interaction between SchedSP and GRID resources is also examined. The important security issues of the Application Service Providing (ASP) model are discussed, in the process of analyzing and defining the particular security considerations of the SchedSP framework. A prototype security model for the effective protection of the SchedSP application environment is presented.

1 Introduction

While the computational power of personal computers is fast increasing making them appear as personal workstations, the networking capabilities are also increasing quite fast and this has led to an environment of huge numbers of personal workstations being always on and connected [1]. The problems in maintenance and upgrading of such a large number of installed computers by properly qualified personnel along with the available networking capabilities gave birth to a new computing paradigm. In this paradigm, these networked personal computers would require a minimal software installation and the other software applications needed would reside, run and live on remote servers of the network. The remote server acts as an Application Service Provider (ASP) whose responsibility is to offer software as a service over the Internet [2]. Shortly after the initial appearance of the ASP concept, several new derivative ideas evolved specialized in offering a particular kind of application. Such specialized service providers, collectively called xSP, include the Storage Service Provider, the Database Service Provider, the Management Service Provider and many others [3].

In this paper SchedSP, which is a framework for the creation of a secure service provider specialized in the creation of scheduling solutions is described. Scheduling applications use algorithms that best allocate specific resources to tasks in order to create an acceptable work schedule, subject to several linear and non-linear constraints. A common characteristic of scheduling problems is their NP-hard complexity, and their large computational and memory needs. Since SchedSP tries to offer these algorithms as services, there is a significant problem with respect to the existence and management of all the required computational resources. This issue seems to find a reasonable solution in a distributed computing environment such as the recently defined Internet based Grids [4,5]. There are several proposals for the formation of Grids and one such

system named PLEIADES has been previously developed in the Computer Systems Laboratory of the University of Patras in order to offer distributed computational and application resources as a service over the Internet and a parallel/distributed computational platform for general use. Such a system, in analogy to the xSP terminology, where small x stands for any such provider, has been coined Computational Infrastructure Service Provider (CISP) [6, 7].

SchedSP, as an ASP provider that is available over the Internet is vulnerable to the same security problems and threats that the ASP business model inherits to it. The Internet is still an insecure environment breading many threats to any online enterprise and e-business, this being in many cases a restraining factor to the wider adoption of these business models [8, 9]. Moreover, due to the specific nature of such systems, new security concerns have arisen and have to be confronted and provide users and systems with the necessary protection against these malicious acts. SchedSP has to fulfill both the general network security requirements and the specialized security issues of its particular environment. Issues like confidentiality and integrity of the data, nonrepudiation, user authentication and authorization must be taken into account when the security subsystem is designed. In addition, as XML over HTTP provides the main communication protocol for SchedSP, the new XML security standards [10] are investigated and their recommendations are taken into consideration.

In the rest of this paper, the SchedSP service provider framework for scheduling solutions is presented, after an introduction of the Grid-Enabled Application Service Providing and a quick review of some scheduling applications. Next follows a discussion and analysis of the security considerations of a general ASP, followed by the specific security considerations of the SchedSP framework as a special service provider case, concluding with the definition of the present SchedSP security prototype.

2 Grid-Enabled Application Service Providing and PLEIADES

An Application Service Provider (ASP) is an organization that provides software as a service over the Internet, either based on the pay per use model or the fixed periodic fee approach. In contrary to the traditional software-in-the-box business model, the ASP approach temporarily leases to the customer both the application and the computational resources required. For this special case of software distribution, different types of license agreements between the customers and the ASP are used, which are called Service-Level Agreements (SLA). An SLA describes the Quality of Service offered including application and disk storage availability, response times and computational throughput. Without making any distinctions among the various user groups of special computational requirements it is assumed that the resource needs of an ASP are proportional to the number of simultaneous users served. The overall system design for the provision of ASP services must guarantee that responsibilities and QoS requirements are enforced.

Besides the networking issues that are often highly unpredictable due to the nature of Internet and have not been examined in this research effort, the computational power is the key quality measure that an ASP should focus. The computational needs can

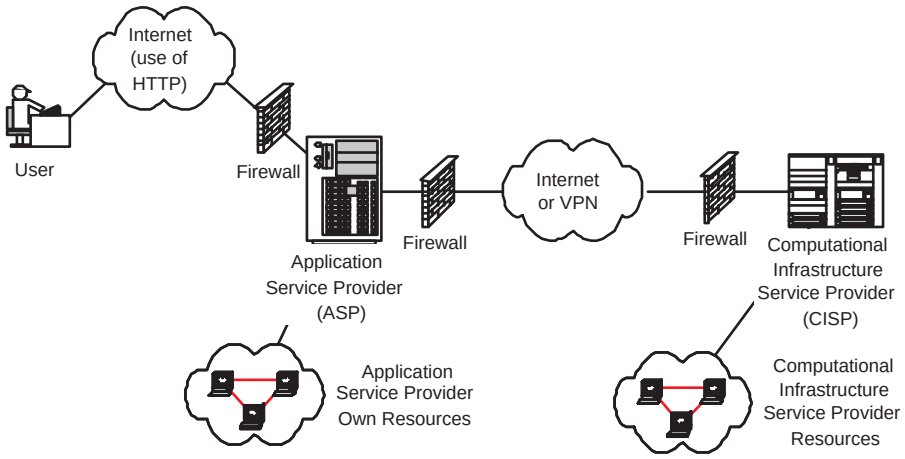

Fig. 1. The CISP concept

be distinguished into two different levels. The first computational need has soft real-time requirements and involves the front-end and some middle-tier components of the ASP environment. These front-end procedures are amenable to parallel computation and this fact could be utilized if difficulties in meeting their performance guarantees did arise. This type of parallelism could be easily explored using an application server software toolkit that many vendors do offer. One such system exists in the Microsoft Application Center included in current server versions of the MS Windows Operating System. The second computational level requirement could be satisfied by an offline computational service offered by a batch-type system, which is suitable for large computations such as simulations or human resource scheduling applications, which often run for large amounts of time. The latter case of computational service is suitable for distributed resources offered by an environment that can provide computational power on demand, like Condor Resource Manager [11–13] or a Grid [4]. When an ASP provides a computational intensive service, it can easily fail to deliver the computational throughput described in the effective SLA contracts when many users appear at once. However, the relaxation of real-time requirements for this task leads to the observation that an ASP could easily and seamlessly utilize such a service from third-party providers. This delivery of computational power in the form of a Grid or Cluster as service over the Internet often called Computational Infrastructure Service Provider (CISP) [6], as illustrated in figure 1. Such a separation of tasks and responsibilities in these two different entities decreases the risks and simplifies the start-up costs for the ASP Service Provider. Furthermore, this model can be expanded to a one-to-many or many-to-many model where various ASP providers would use more than one CISP providers and vice versa. A federated model could finally emerge, where complex ASP - CISP organizations share resources in order to deliver the required quality of service level. In the federated model, each ASP would specialize in developing services that constitute its business competence while incorporating services that excel from

the competition. The result could resemble the present view of the software market with the vendors offering either end-user software or components for use in end-user software environments.

A CISP system under the name of PLEIADES [6, 7] has been developed recently. The PLEIADES system utilizes a pool of computers managed by the Condor Resource Manager for the computational infrastructure. Figure 2 presents the services offered by PLEIADES, which are accessible by a web-based interface and by XML Web Services over HTTP as well. In summary, PLEIADES offers a space for file storage of input and output files of the programs running and job submission and monitoring services for guided submission of programs for execution and monitoring of their progress. An exceptional feature is the cross-compile, which uses the pool of resources to compile ones program in selected platforms offered by PLEIADES, to be able to utilize the maximum of the system. PLEIADES is based in a volunteer model for resource availability, since it was designed a tool for university use, but the resources could very well be owned by the organization running a PLEIADES system for profit.

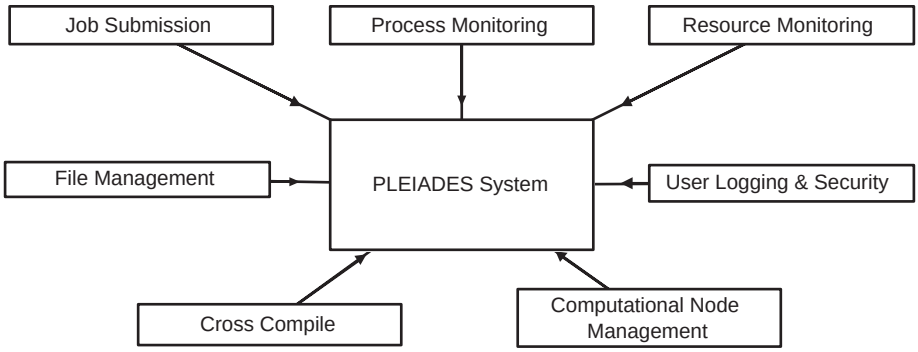

Fig. 2. The PLEIADES Services

3 Scheduling Applications

A scheduling application takes in most cases input files with resources that need to be matched with other resources, which are in most cases humans and machines, in an optimal manner and obeying a set of rules and constraints. The human resources can be for example teachers in the case of school timetabling, or personnel in the case of more general personnel scheduling. The other resources in the above examples would be classes and classrooms or work shifts. The main characteristic of the problems solved by most scheduling applications is the complexity of the solution process which is of exponential order of the problem size. This complexity is referred to as combinatorial explosion and for this reason the results obtained from most real-world scheduling problems are near optimal. A mathematical expression that evaluates the quality of the present allocations of the various resources, often referred to as the objective function,

is repeatedly calculated during the solution process. Many aspects of the overall quality of the result are directly dependent on the expressions that constitute the objective function.

From this discussion, it is obvious that the scheduling problem is transformed to a mathematical model that is solved used various optimization techniques. A common need of these techniques involves the solution of linear or integer optimization problems with constraints. For this problem, several libraries and tools can be utilized. If the scheduling application depends on such tools, they need to be installed on the host computer. In addition, scheduling applications typically require considerable computational and memory resources. These computational necessities along with the fact that such scheduling applications in many instances are not used every day in order to justify the maintenance and support of private solutions qualifies these applications as good candidates for use through the newly developed application service providing deployment scheme.

4 The SchedSP Framework

The SchedSP framework design aims at providing scheduling applications as services over the Internet [14]. The special input/output and modeling nature of scheduling applications together with their demanding computational needs were the basic considerations in the design of the core services of SchedSP. For the user interface, a thin-client approach is able to satisfy the user requirements for the creation of the necessary interface modules and is the selected operation mode. This practically means that there are minimal user equipment expectations meaning that a machine capable to present rich web content containing either Java applets or ActiveX components is the only SchedSP requirement. The thin client interface components communicate with SchedSP using the XML messaging over the HTTP protocol. Figure 3 presents the communication layers, where SchedSP layer is responsible for authentication, authorization as well as other similar issues of managerial nature and the SchedSP Service Provisioning layer is responsible for providing the actual services.

SchedSP provides several services, which fall into two main categories: File storage, execution and monitoring of scheduling applications. The file storage support the need for storing files needed by the scheduling and other applications offered by the SchedSP framework, the execution and monitoring of scheduling applications utilize the PLEIADES system for Grid-based execution of these computationally demanding applications. An architecture overview of the SchedSP framework is presented in figure 4, while the present prototype architecture is presented in. In the following section, an attempt will be made to present the steps taken for a typical use scenario.

A user, using the SchedSP interface, connects to SchedSP using the HTTP protocol over the Internet. The user requests are channeled through a web server to the Gateway component, which is responsible for web server security issues and for forwarding the secure requests to the Executive component. The Executive component implements the SchedSP layer of the communication stack, and is responsible for the user authentication and authorization. The Executive component also distinguishes the various requests in accordance with the service component required for its service and

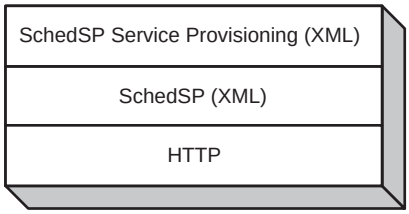

Fig. 3. SchedSP Communication layers

forwards the necessary portion of the request to the appropriate component. The appropriate component also called a Service Provisioning Component (SPC) serves the user request and the corresponding response returns to the user interface following the same path backwards, through the Executive and the Gateway component.

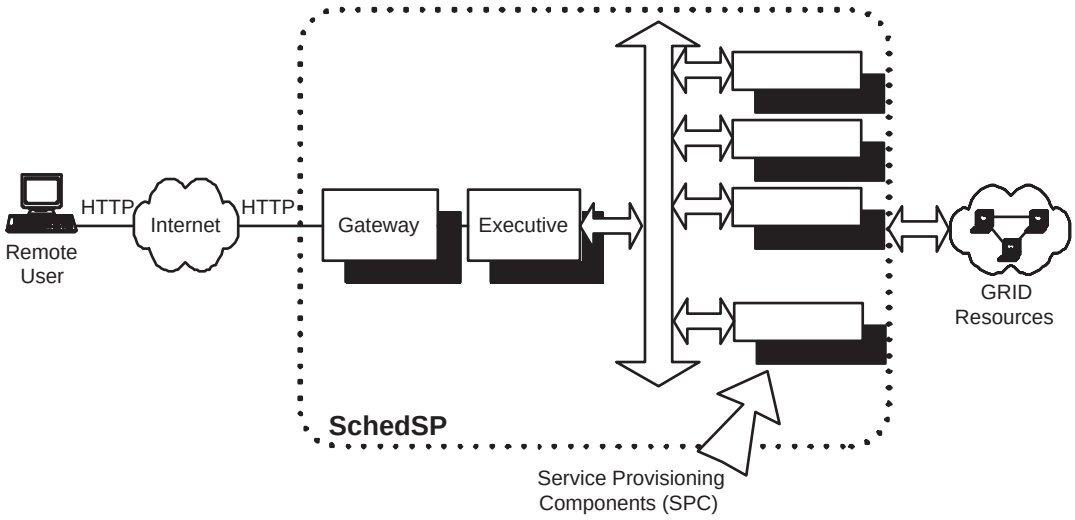

Fig. 4. SchedSP Architecture Overview

Figure 5 presents the present set of SPC components of the SchedSP prototype. The File space SPC serves the file operations. The XML Transformations SPC performs several XML transformations including XSLT and XPath-based modifications to XML files. The Execute and Monitor components support the execution and monitoring of scheduling applications. The ACL (Access Control List) SPC provides uniform role-based access control to resources across the SchedSP components while the Settings component provides other uniform functionality. Finally, the MultiServe SPC provides a useful script for a sequence of service requests in order to save the round-trip delays during complex operations.

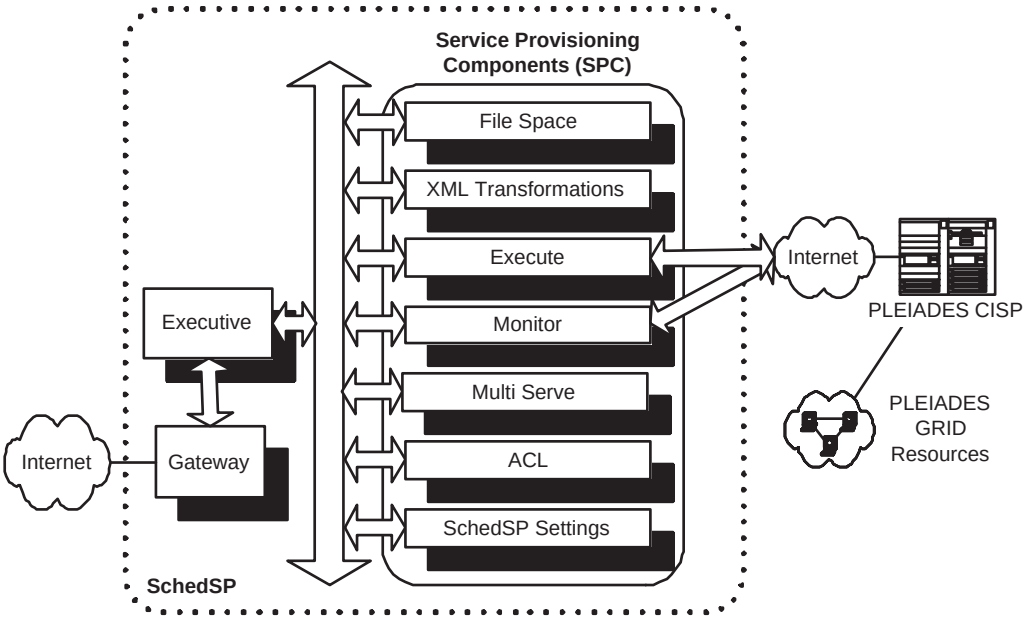

Fig. 5. SchedSP Prototype Architecture

5 Security Issues of the ASP Model

Although the Application Service Providing (ASP) and Web Services models are being fast accepted as a viable alternative to the classical software ownership solution the security risks that these technologies have provide one of the main reasons that such technologies have not yet been widely accepted [8]. Especially in services and applications where the need to secure sensitive users data is necessary, the ASP and Web Services approach lacks a reliable and fully acceptable and efficient security model [9].

Since the ASP model relies on the Internet set of protocols (TCP/IP, HTTP, SMTP, etc.), it is obvious that this approach inherits all of the security weaknesses and threats related to the above protocols. Actually, some of the main security 'headaches' that trouble the ASP and Web Services system developers and administrators relate with the security of the servers hosting their Web applications. These servers are constantly attacked either by simple Internet attacks or by more complex and sophisticated attacks, such as a Distributed Denial of Service (DDoS) attack, which is more difficult to confront and more harmful to the attacked enterprises. The solutions to this kind of threats involve the use of traditional techniques such as firewalls, Virtual Private Networks (VPN) and Intrusion Detection Systems (IDS) [15] that are already in use for the security of web servers. In addition to that, as Boyens and Gunther allege in [16], the basis of trust on which rely many of the existing online services, cannot be considered adequate, especially in distributed systems where multiple parties are involved. Issues like honesty and truthfulness among the involved parties that often govern the human relationships, trustworthiness of the partners that the Service Provider works

with or even the integrity of the staff of the involved Providers and partners have to be taken into account in security system design.

Each ASP and Web Service system has to fulfill both a set of general network security requirements and a set of specialized security issues [17, 18]. Confidentiality refers to the need that the communication between two parties is safe so that the transferred data cannot be available to a third malicious party. It has to be ensured that even if that data is monitored and stolen, it will not be possible for the eavesdropper to understand and use it. This can be partially guaranteed with the use of encryption. Integrity is the requirement of detecting if a third party has tampered the transferred data. This can be ensured with the use of digital signatures. Non-repudiation refers to the need that a communicating party cannot claim not to have transmitted a certain packet of data, which can be ensured by the use of digital certificates. Authentication corresponds to the need that each transacting party can provide enough evidence for its identity, similarly to nonrepudiation, by the use of digital certificates or more simple authentication techniques like passwords. Authorization refers to the need of determining which actions each authenticated user is allowed to perform. This can be ensured with the use of Access Control Lists (ACL) and well definition of each system's set of policies. The more specialized security issues involve the use of particular methodologies to overcome the additional security weaknesses that arise from the use of multiple computer systems [19] and applications requesting services from each other [17].

In order to satisfy the above security needs that the ASP and Web Services model require, new security technologies are being developed. Since the communication is mostly done using the XML protocol, efforts of implementing the present security technologies within the XML standard are currently under way [20]. Organizations like W3C, IETF and OASIS are developing new XML security standards such as XML Encryption [21], XML Signature [22], XKMS [23], XACML [24], etc. These new standards manage to include XML-formatted security data into HTTP and SOAP messages, providing persistent security in the way the data is not only secured during the communication between the parties but remains secure even after the communication has ended or even during new transactions have begun. A consortium of corporations that include IBM, Microsoft and Verisign has already adopted the standards mentioned above, and as a result, the WS-Security [25] standard has emerged.

6 SchedSP Security Requirements

SchedSP, as a service provisioning system, inherits most of the above security risks. Currently, SchedSP uses XML over HTTP for transmitting messages between the user and the Gateway component that acts as the entry point to the SchedSP environment. In addition, SchedSP communicates in the same manner with the PLEIADES GRID system that comprises the CISP partner, in order to get access to a pool of distributed computer resources. Both of these communication channels have to be taken into consideration in the design of the proposed security framework.

As mentioned above, confidentiality is one of the most important security requirements of the ASP and Web Services models. Although in many scheduling applications the data that is required does not include extremely sensitive information about the re-

sources that are planned or the human resources involved, in general, especially where business enterprises are concerned, the need of confidentiality is more intense. So, confidentiality of the data cannot be limited just to the communication part, where some form of message security is always necessary, while on the server the data remain unprotected. Data encryption has to be enforced in order to ensure confidentiality during all the phases of the transactions in order to avoid the need for the users to completely sanitize the data before submitting it to SchedSP.

On the other hand, integrity of the data is quite important in scheduling applications. It has to be ensured that the data is not altered in any manner during the communication or while stored in the server. In addition, nonrepudiation must be ensured. Since users usually are working for a particular organization, this organization needs to know which user did actually create or modify the specific problem solution and the set of input data.

Similarly to nonrepudiation, authentication of users is also required. Since SchedSP is a web-based platform, exposed for potential access to anyone in the Internet, there has to be a scheme for the proper authentication of the system users. Moreover, authentication of users provides the means of keeping track of each action and correctly credits these actions to each user. Another important aspect has to do with authorization. Since filespace is provided for every user on the SchedSP platform, the access rights of each user have to be well defined and enforced. Policies concerning the access rights of the various user types have to be created. This involves not only a single user-level access control but also role based access control.

The security model that we propose for the SchedSP platform is mainly focused on implementing the XML security standards, wherever possible. Since SchedSP does not use the SOAP protocol for data transfer, existing security architectures and products such as WS-Security cannot be used. A new model that utilizes the XML standards and remains close to the ASP and Web Services model is proposed. In the present prototype of SchedSP, SSL is used in order to guarantee confidential communication between the user front-end and the Gateway component. The SSL standard provides safe data transmission by the use of algorithms for encrypting the messages, although it provides a significant overhead for ciphering/deciphering the messages [17]. Since most scheduling applications are not time-critical, this overhead is not significant. The main drawback is the fact that SSL does not provide persistent encryption, meaning that the data remain encrypted only during the communication between the parties and not while stored on the servers [16]. If this is a priority, it is proposed to adopt the use of the XML Encryption standard instead of SSL. The main advantage of XML Encryption is that portions of an XML document can be selectively encrypted. This contributes to reducing significantly overhead because only sensitive portions of the message need to be encrypted and thus avoiding the encryption of the whole set of data [26, 27]. An example of the XML Encryption structure is shown in figure 6, where the structure of the <EncryptedData> element is presented. For the communication between SchedSP and PLEIADES the SSL protocol is used.

The use of SSL also provides integrity of the message during its transfer. That means that the data cannot be tampered during the time of transmission. On the server, SSL does not provide data integrity. Data have to remain digitally signed, so that any

```
<EncryptedData Id? Type?>
        <EncryptionMethod/>?
        <ds:KeyInfo>
                <EncryptedKey>?
                <AgreementMethod>?
                <ds:KeyName>?
                <ds:RetrievalMethod>?
                <ds:*>?
        </ds:KeyInfo>?
        <CipherData>
                <CipherValue>?
                <CipherReference URI?>?
        </CipherData>
        <EncryptionProperties>?
</EncryptedData>
```

Fig. 6. The <EncryptedData> Element Structure

alteration would be detected. In that case it is proposed to use the XML Signature standard, developed by W3C and IETF. XML Signature defines how data, mainly contained in XML formatted messages, can be digitally signed by using the existing algorithms for signatures and message digests and then represent the resulting signature in XML. The XML Signature structure is shown in figure 7. Tampering with the data causes the generation of different message digests that do not verify the original digest, therefore unveiling the alteration. In addition the use of digital signatures can be used for satisfying the nonrepudiation and authentication requirements, since each digitally signed message contains information about its sender that cannot be disputed [26, 27]. For authentication purposes, the SchedSP platform is currently using a password based entry.

```
<Signature>
        <SignedInfo>
        (CanonicalizationMethod)
                (SignatureMethod)
                (<Reference (URI=)? >
                        (Transforms)?
                        (DigestMethod)
                        (DigestValue)
                </Reference>)+
        </SignedInfo>
    (SignatureValue)
   (KeyInfo)?
   (Object)*
</Signature>
```

Fig. 7. XML Signature Structure

For authorization purposes the ACL component is used. Access control is defined in user level and in role level, depending on the type of the applications that each user wants to access. Users can be divided into two main categories: the end user and the developer user. The main difference of the developer role definition is that a developer has the ability to create new roles and enforce specific user rights policies. The above proposed security solution is contained in the Security Component SPC of the SchedSP, shown in figure 8.

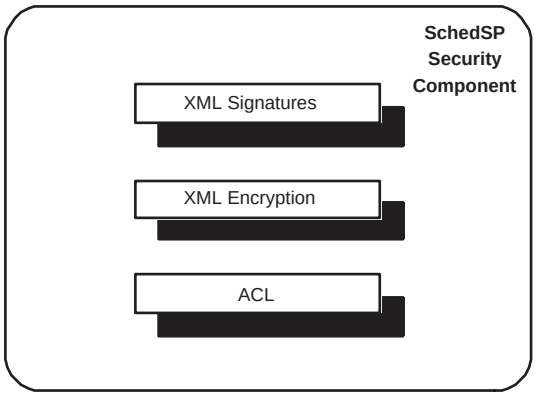

Fig. 8. SchedSP Security Component

7 Conclusions - Future Work

The SchedSP system is presented and analyzed with particular emphasis on the security issues that arise from this computational model. Emphasis was given on the security aspects of the SchedSP environment assuming that network and server security issues have already been addressed. We tried to locate the main security considerations of on-line services and in particular the application level security policies and trust issues are addressed. Then we mapped these security concerns on the present SchedSP prototype platform and various solutions were presented.

In the immediate plans of the team is the further enhancement of the security level of the various SchedSP interactions. Since the SchedSP system might interact with other online services, confidentiality becomes extremely critical and will be further enhanced. In addition the use of the SOAP protocol is within the future considerations, so that a more extended use of the XML security standards can be adopted in the SchedSP system. Finally, following the ASP and Web Services model, many of the security features could be provided by an independent online service that plays the role of the third trusted party between the user and the SchedSP. In a more extreme case, an external security service provider could be defined which would then provide all security related services.

References

1. Always on: Living in a networked world. *IEEE Spectrum*, 38(8), (2001)
2. R. Fourer, J.-P. Goux: Optimization as an Internet Resource. *Interfaces*, 30(2):130-150, (2001)
3. ASP - Application Service Providing, The Ultimate Guide to Hiring than Buying Applications. SCN Education B.V. (ed), Vieweg, (2000)
4. I. Foster, C. Kesselman (eds). *The Grid: Blueprint for a New Computing Infrastructure*. Morgan-Kaufmann, (1998)
5. The Global Grid Forum Website: http://www.ggf.org/
6. Koulopoulos D., Papoutsis K., Goulas G., Housos E.: PLEIADES: an Internet-based Parallel Distributed System. *Software: Practice and Experience*, 32(11):1035-1049 (2002)
7. Koulopoulos D., Goulas G., Papoutsis K., Housos E.. A Parallel/Distributed Platform for University Computational Infrastructure Service Provisioning. *Proceedings CSIT Workshop*. Patras, Greece, (2002)
8. Web services: Still not Ready for Prime Time. CIO Magazine, Sep. 1 (2002)
9. Web Services - not so Fast: a New Era for e-Commerce is Poised on the Fast Track, but Security Raises Caution Flags. Available from: http://www.infosecuritymag.com/2002/oct/webservices.shtml.
10. Standards for XML and Web Services Security. *IEEE Computer*, April (2003), 96-98
11. The Condor Project Homepage: http://www.condorproject.org/
12. J. Basney, M. Livny: Deploying a High Throughput Computing Cluster. In: R. Buyya (ed.), *High Performance Cluster Computing*, Vol 1, Chapter 5, Prentice Hall
13. M. Livny, J. Basney, R. Raman, T. Tannenbaum: Mechanisms for High Throughput Computing. *SPEEDUP Journal*, 11(3), (1997)
14. Goulas G., Housos E.: SchedSP: Providing GRID-enabled Real-world Scheduling Solutions as Application Services. *Proceedings Euroweb Conference*. Oxford, (2002)
15. Anderson R.: *Security Engineering*. Wiley, (2001)
16. Boyens C., Gunther O.: Trust is not Enough: Privacy and Security in asp and web Service Environments, *Proceedings 6th ADBIS Conference*, Springer LNCS, (2002)
17. O'Neill M. et al: *Web Services Security*. McGraw-Hill/Osborne, (2003)
18. Hartman B., Flinn D., Beznosov K., Kawamoto S.: *Mastering Web Services Security*. Wiley, (2003)
19. Moffett J.: Distributed systems security. In: A.I. Kent, J.G. Williams (eds.), *Encyclopaedia of Microcomputers*, Vol.15. Marcel Dekker Inc, New York, (1995)
20. Eugene X. P.: XML Based Security for e-Commerce Applications. *Proceedings 8th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS)*. (2001)
21. XML Encryption Working gGroup, W3C, http://www.w3.org/encryption/2001/
22. IETF/W3C XML Signature Working Group, http://www.w3.org/signature/
23. XML Key Management Working Group, http://www.w3.org/2001/xkms/
24. OASIS XACML Technical Committee, http://www.oasis-open.org/committees/xacml/
25. IBM and Microsoft: Security in a Web Services World: a Proposed Architecture and Roadmap (2002) http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwssecur/html/securitywhitepaper.asp
26. Eastlake III D., Niles K.: *Secure XML: the New Syntax for Signatures and Encryption*. Addison-Wesley, (2003)
27. Dournaee B.: *XML Security*. McGraw-Hill/Osborne, (2002)