Authentication and Authorization in Spatial Data Infrastructures

Bastian Schäffer
52°North GmbH, Münster, Germany

ABSTRACT

Spatial Data Infrastructures (SDI) provide access to different kinds of georesources such as data and models. It is evident that in several SDI use cases some of provided georesources have to be protected for various reasons such as confidentiality, privacy or commercial value. This paper summarizes state of the art concepts and technologies for the two major concepts for protecting a georesources in SDIs: Authentication and Authorization. One well known scheme (HTTP-Basic Authentication) is demonstrated to protect a geospatial Web Service as a reference example.

1 INTRODUCTION

Geospatial Web Services organized in a Spatial Data Infrastructures (SDIs) are designed for the purpose of providing and sharing georesources (data and models) across organizational and technical boundaries. The real potential lies in the agility of Geospatial Web Service via SDIs to access external georesources on-demand and to integrate them into business process on the fly (Groot & McLaughlin, 2000). This goal is mostly reached on a technical level by the provision of data encoding and service interface standards, such as established by the Open Geospatial Consortium (OGC) or INSPIRE.

However, not all provided georesources in an SDI can be offered freely and openly, i.e. anybody can access them. In some cases, the access to these georesources has to be protected and restricted due to e.g. confidentiality or privacy reasons or because the georesource should be exploited commercially.

Since in an SDI these georesources are provided via Geospatial Web Service, the access to these Geospatial Web Services has to be restricted. In this context a thorough overview of general security concepts such as authentication, authorization, cryptography and trust in a Geospatial Web Services setting is provided at first. This is followed by a pragmatical demonstration of a simple authorization and authentication solution.

2 BACKGROUND

This section provides a review of basic concepts and related work in the context of rights management, security and licensing for Geospatial Web Services.

2.1 Web Service Security Definition

The definition of computer security in general has been defined in multiple ways, as, for example, by Gollmann (1999) and Bishop (2005). ISO defines it as:

> “Information held by IT products or systems is a critical resource that enables organizations to succeed in their mission. Additionally, individuals have a reasonable expectation that their personal information contained in IT products or systems remain

private, be available to them as needed, and not be subject to unauthorized modification. IT products or systems should perform their functions while exercising proper control of the information to ensure it is protected against hazards such as unwanted or unwarranted dissemination, alteration, or loss. The term IT security is used to cover prevention and mitigation of these and similar hazards". (ISO 1999, p. 9)

To protect the exchange of information between secured systems and the management of the stored data, ISO also states that "security services may apply to the communicating entities of systems as well as to data exchanged between systems, and to data managed by systems". (ISO 1996a, p. 1)

Going one step further and taking Web Services into account in this context, Hafner & Breu (2009, p. 28) define Web Services security as "the sum of all techniques, methods, procedures and activities employed to maintain an ideal state specified through a set of rules of what is authorized and what is not in a heterogeneous, decentralized and interconnected computing system", which is used as the definition for Web Service security in this paper, because it takes the properties of SDIs, such as Web Services in general, heterogeneity and decentralization into account as well as interconnection, which is relevant for workflows.

2.2 Building Blocks of Web Service Security

This section discusses the most relevant security requirements derived from the literature review. Based on Hafner & Breu (2009), ISO (1996a), ISO (1996b) ISO (1999), Kanneganti & Chodavarapu (2008) and Bertino et al. (2009), authentication, authorization, confidentiality, integrity, non-repudiation, protection and privacy are the most relevant aspects. These requirements only address the message exchange, which is relevant when defining Web Service interfaces and protocols. There may be other requirements addressing physical or organizational protection, but this is out of the scope of this paper.

Authentication

Authentication is defined by ISO (1996b, p. 3) as "the provision of assurance of the claimed identity of an entity". In other words, authentication describes the verification that a communication partner is what it pretends to be. In general, authentication answers the question 'Who is accessing a resource?'. This differs from the concept of identification, which describes the process of providing an identity but not of verifying it. The verification part is then referred to as authentication (Kanneganti & Chodavarapu 2008). In principle, there are three different methods of authentication according to Kanneganti & Chodavarapu (2008):

- Authentication by knowledge
- Authentication by possession
- Authentication by personal attributes

Authentication by knowledge uses secrets such as passwords or personal questions only known to the holder of that information. Many internet protocols such as HTTP, telnet, FTP or SMPT support this type of authentication. It is necessary to distinguish between simple passwords with an unlimited lifespan, one-time use passwords such as TANs, and more complex challenge–response protocols (Schmeh 2009).

Authentication by possession typically uses digital signatures as described in Section 2.3.4. Only the holder of a specific key (a shared key in symmetric systems and a private key in asymmetric systems; see Section 2.3.1 and 2.3.2) is able to create a digital signature which can be verified by the receiver.

Authentication by personal attributes leverages biometric authentication. Fingerprints, iris scans, face, hand, voice or ear detection are common methods, but with limited adoption in Web Service environments.

Authenticity

Authenticity describes the concept that a piece of information, such as a received message, originates from the expected sender (Rosado et al. 2006). While the concept of authentication (see above) is typically active where an entity authenticates itself, authenticity is passive where a receiver validates the undisputed credibility of, for example, a message. Digital Signatures combined with trust (see Section 2.3.4) are a typical means of realizing authenticity.

Authorization

Authorization is not directly related to message exchange, but nevertheless it is one of the key security requirements. Authorization aims to control access to resources (Kanneganti & Chodavarapu 2008). For access-controlled services, incoming requests are matched against policies which define access rights to certain resources for certain subjects (requestors). If these access rights cover the requested action, access is permitted, otherwise access will be denied.

There are multiple access control models which apply the authorization principle (see Section 2.5).

Confidentiality

Providing confidentiality means protecting messages against unauthorized reading (Kanneganti & Chodavarapu 2008). It has to be ensured that only the designated communication partners (typically the sender and the receiver of a message) can access the content of a message and not a middleman observing the communication. Encryption is a standard technique to ensure confidentiality (see Section 2.3.3).

Integrity

Integrity protects messages against unnoticed modifications (Kanneganti & Chodavarapu 2008). On the message level, integrity is typically provided by the use of digital signatures (see Section 2.3.4). These signatures are tightly bound to the message to be protected. Whenever there has been a modification of a message after the signature was applied, a validation of this signature will fail. If security at the transport level is provided, integrity is ensured once the secure communication session is established via for example SSL/TLS.

Non-Repudiation

Non-repudiation provides evidence to the receiver of the existence of a message (Kanneganti & Chodavarapu 2008). This becomes important to prevent fraud claims against the message receiver. Non-repudiation is ensured by storing messages together with a valid signature of the sender. If a sender denies having submitted a certain message afterwards, the receiver can expose the signed message. Since no one else but the sender is able to generate the signature (see Section 2.3.4), the stored message proves that the message was signed by the holder of the public key (see Section 2.3.2) corresponding to his signature.

Protection

Protection against attacks in the network environment is an elementary aspect. There are three main kinds of vulnerabilities which allow attacks, according to Kanneganti & Chodavarapu (2008):

- Vulnerabilities at the application code level, which allow, for example, an SQL injection
- Vulnerabilities at the administration level, such as unchanged default passwords
- Vulnerabilities in the computing and network infrastructure, such as weaknesses in operating systems or underlying protocols, for instance, TCP/IP

Since this paper focuses more on architectural aspects, protection against these low-level vulnerabilities is not further considered but is included here for reasons of completeness.

Privacy

Privacy deals with protecting confidential private or personal data against disclosure to third parties. Legal issues also play a role here as shown, for instance, by Critchell-Ward & Landsborough-McDonald (2007). In Web Service environments, user identities can be, amongst other things, masked by pseudonyms coming from dedicated pseudonym services. A Policy Information Point can be used for this purpose. Since privacy is hard to enforce, this correlates strongly with trust (see Section 2.4).

2.3 Cryptographic Background

Cryptography is a key technology for meeting the security building blocks described in Section 2.2. According to NIST (2010, p. 55), cryptography is “the discipline that embodies the principles, means, and methods for the transformation of data in order to hide their semantic content, prevent their unauthorized use, or prevent their undetected modification”.

This section briefly discusses cryptographic methods used for encrypting and signing messages as a typical means of addressing the Web Service security building blocks explained in Section 2.2.

2.3.1 Symmetric Cryptography

Symmetric cryptography is sometimes called *secret key cryptography* (versus *public key/asymmetric cryptography*, see Section 2.3.2) because the entities that want to exchange an encrypted message share a single common artefact referred to as *key*. Such a key allows the encryption and decryption of a secret message and needs to be kept secret for this reason.

One classical example is the Caesar cipher, which is a substitution cipher. Each letter in the plaintext is replaced by a letter which is shifted a fixed number of positions in the alphabet. The shift number is therefore the shared secret key. For example, with a shift of 5, *A* is replaced by *F*, *B* becomes *G*, etc. The message sender and receiver need to know the same secret key (shift of five positions to the right in this example) in order to encrypt/decrypt the message. An example of modern symmetric cryptography algorithms are the Advanced Encryption Standard (AES) (Daemen & Rijmen 2002) and Twofish (Schneier 1999).

Keeping the shared key secret entails both cost and risk when the key is distributed. In particular, for n partners, n/2*(n-1) keys have to be exchanged (Schmeh 2009). Thus, symmetric cryptography has a disadvantage in terms of key management compared to asymmetric cryptography.

2.3.2 Asymmetric Cryptography

Asymmetric cryptography, also known as *public key cryptography*, uses a distinct pair of keys for any participant of a secure communication. One key has to be kept secret (the private key), while the other key is public (the public key) and has to be shared with the communication partners (Kanneganti & Chodavarapu 2008) (for a review of Public Key Infrastructures see Section 2.4.4).

The basic principle of asymmetric cryptography is that plaintext encrypted with one key of this pair can only be decrypted with the other one. If a message is encrypted with a public key it can only be decrypted with the corresponding private key and vice versa. Mathematically, this concept can be expressed as:

$$c = e(a,m) \text{ and } m = d(x,c) \qquad (1)$$

as well as

$$c = e(x,m) \text{ and } m = d(a,c) \qquad (2)$$

where the sender has the public key a of the receiver and their own private key x, m is the message in plaintext, c the ciphertext, e the encryption function and the d the decryption function. In case (1), the sender only has to know the public key (a) of the receiver and both parties need to use the same cryptographic algorithm (encrypt function (e), decrypt function (d)). In case (2), the sender encrypts the message with their private key (x), which allows the receiver to verify with the sender's public key (a) the authenticity of that message. A typical asymmetric cryptography algorithm is RSA, which is based on prime number multiplication (Adleman et al. 1978).

Since the described mechanism works in both ways it can be used for encrypting and signing messages. Both concepts are fundamental for this paper in order to meet the described Web Service security building blocks (Section 2.2).

2.3.3 Encryption

Encryption is defined by IETF (2000, p. 68) as a "cryptographic transformation of data (called 'plaintext') into a form (called 'ciphertext') that conceals the data's original meaning to prevent it from being known or used. If the transformation is reversible, the corresponding reversal process is called 'decryption', which is a transformation that restores encrypted data to its original state".

For instance, if user *A* wants to send an encrypted message to user *B*, user *A* encrypts the message with *B's* public key using asymmetric cryptography, so only *B* is able to decrypt this message with his own private key.

Since asymmetric cryptography needs far more computation than symmetric encryption (Schmeh 2009), both technologies are typically used in combination. First, the initiator of the communication creates a session key which is asymmetrically encrypted and sent to the communication partner. Now that a session key is securely exchanged, it can be used together with symmetric encryption for further communication between *A* and *B*. This hybrid encryption technique is, for instance, used in common TLS/SSL technologies.

In terms of Web Services, XML Encryption (Imamura et al. 2002) is the most relevant standard for message-based encryption. It defines how to encrypt the contents of an XML document. For transport-level encryption, SSL/TLS is the most widespread technology.

2.3.4 Digital Signatures

IETF (2000, p.58) defines Digital Signatures as "a value computed with a cryptographic algorithm and appended to a data object in such a way that any recipient of the data can use the signature to verify the data's origin and integrity".

In particular, digital signatures rely on asymmetric cryptography, which requires a pair of cryptographic keys for each participant involved in secure communication (Kanneganti & Chodavarapu 2008). Therefore, sending a signed message requires asymmetric encryption of the message with the sender's private key (see equation 2 in Section 2.3.2). This mechanism allows anybody to decrypt the message with the sender's public key and thus prove that nobody else could have encrypted this message but the owner of the corresponding private key.

For performance improvements, instead of encrypting the whole message, typically a hash value is computed out of the message and then this hash value is signed while the message itself is sent in plaintext.

Of course, encryption and signatures can also be used in combination, by first computing a signature with the sender's private key, and then encrypting the signed message with the recipient's public key.

For Web Services, XML Signature (W3C 2002) is the most relevant standard. It specifies how an XML message can be digitally signed.

2.3.5 Security Levels

It is, in principle, possible to differentiate between transport-level security and message-level security (Schmeh 2009).
Transport-level security deals with securing the transport protocol (such as HTTP, FTP, SMTP, etc.) instead of securing the message itself.

The most relevant technology is the Secure Socket Layer (SSL) developed by Netscape, and in a slightly modified version standardized by the IETF, thereafter known as *Transport Layer Security protocol (TLS)* (Schmeh 2009). This paper uses SSL as a synonym for both protocols. SSL makes use of symmetric and asymmetric encryption technologies using X.509 certificates. A functional Public Key Infrastructure (see Section 2.4.4) is therefore the foundation for using SSL.

Message-level security is focused on the means for securing the message instead of the transport protocol. The most relevant standard is WS-Security defined by OASIS (2004). It is focused on the SOAP protocol and introduces a <wsse:Security> element, which is embedded in the SOAP Header element. It serves as a container for other tokens and defines how to embed, for example, encryption with XML-Encryption (Imamura et al. 2002), Digital Signatures with XML-Signatures (W3C 2002), and other elements such as SAML tokens.

2.4 Trust

Trust, and especially trustworthiness, is a fundamental aspect of security in distributed systems. This is especially important in the context of SDIs, where business transactions span across enterprise and security domain borders. Thus, this section will review the concept of *trust* in the context of this paper.

Basically, trust is an intrinsic part of all business transactions; over the internet or at the supermarket around the corner. On the one hand, customers need to rely on the fact that the seller really provides the service or goods they advertise and does not disclose confidential information such as name, credit card number, etc. On the other hand, sellers need to rely on the fact that, for example, a customer is the person stated on their ID card and therefore is old enough to buy a pack of cigarettes. Therefore, trust is an important factor for business decisions.

2.4.1 Definition

In general, it is possible to differentiate between a *trustor*—an entity that trusts a target entity—and a trustee—an entity that is trusted. The concept of trust involves multiple aspects like belief in truthfulness, reliability, reputation, risk etc. (Grandison & Sloman 2000). There is no general accepted definition of trust in the literature; however, its importance has been recognized and multiple approaches can be found.

As one of the first definitions, Kini and Choobineh (1998, p. 9) defined trust from a socio-economical and social psychology perspective: trust is "a belief that is influenced by the individual's opinion about certain critical system features". Their definition covers several aspects of human trust in computers but lacks the definition of trust in e-commerce scenarios.
Grandison and Sloman (2000, p. 4) define trust after a survey of various definitions as "the firm belief in the competence of an entity to act dependably, securely and reliably within a specified context" (by assuming that dependability covers reliability and timeliness).

The Joint Research Centre (Jones & Morris 1999) defined trust from a business perspective in a way that business partners and the whole business transaction can be relied on. The time variant and measurable notions were missing in these definitions and were added by Dimitrakos (2003, p. 56), resulting in the definition that "Trust of a party A in a party B for a service X is the measurable belief of A in B behaving dependably for a specific period within a specified context in relation to X".

This definition also regards trust from a service-oriented point of view. Therefore, it can be used as the definition of trust in this paper.

2.4.2 Trust Relationships

Trust has to be regarded independently for every business transaction. However, there are some common aspects of all trust relationships according to Grandison & Sloman (2000) and Dimitrakos (2003).

In principle, trust is not absolute, nor always bound to a specific action in the context of a business transaction. For instance, A trusts B to be able to handle a car but not a gun. Even though there are ideas for a dual approach where a trustee is trusted or untrusted in general (Yahalom et al. 1993), this may result in an oversimplified view which cannot distinguish between the characteristics of different business tasks. It is interesting to note that not even trust in oneself is absolute: for instance, protection against accidentally overwriting of files.

This leads to the proposition that trust is measurable. For instance, A trusts B but A does not trust C to do the same task. In the literature, no general metric can be found and has been established for specific domains and problems (e.g. eBay user ratings). However, it is possible to distinguish between a relative measurement and an absolute measurement.

Additionally, trust is directed, in other words not symmetric. Consider that A trusts B but B does not necessarily have to trust A. Generalized, this means that trust depends on the role of an entity in a transaction. Especially when it comes to trust between groups, trust is not necessarily distributed to the members of the group. This means that A can trust a collective C :=: $(C_1 \ldots C_n)$ but does not necessarily have to trust each member of C to the same degree. For instance, A trusts that C delivers a project in time, but does not trust C_i, $C_i \in C$, to the same degree to deliver because C_k, $C_k \in C$, may compensate the deficits of C_i.

By taking a closer look at the definition stated in Section 2.4.1, we see that trust may be time-variant. In other words, A trusted B in the past for a specific business transaction but will not necessarily trust B again in the future. For instance, if B has abused the given trust, trust may be revoked for the future.

Trust can be transitive. For example, A trusts B and B trusts C. Therefore, A trusts C. However, the literature recommends avoiding transitivity (see e.g. Romberg (2002) or Christianson & Harbison (1997)), because B can trust D without consensus from A leading to unintentional transitivity. In some cases transitive trust is necessary and therefore has to be carefully assessed for unintended side effects.

2.4.3 Trust Classification

In the literature, there are different classifications of trust, as for instance from Grandison & Sloman (2000) and Dimitrakos (2003), as outlined below.

Resource Access Trust

First, consider a resource (e.g. Web Service) R controlled by A. If B wants to access R, A has to trust B to be sure that the requesting entity claiming to be B really is B. Otherwise, A cannot grant the requesting entity access to R. This type of trust is the basis for access control. On the basis of this fundamental trust relationship, predefined policies (in RBAC) can be evaluated and fine-grained decisions can be taken.

Provision of Service Trust

Provision of Service Trust describes that B trusts A for a Resource R. For example, if A advertises specific quality of service parameters P for R, then B has to trust A that R complies with P. This type of trust becomes extremely important in SOAs with an ad hoc Publish-Find-Bind pattern.

Certification-Based Trust

This type of trust allows A to trust B based on a set of presented certificates issued by an authority C. Therefore, A has to trust C (transitive trust, see Section 2.4.5). This is common practice for authentication in internet applications (Grandison 2003) as well as in the real world: for example, certified medical doctors. Section 2.4.4 will present a specific solution (PKI) for this type of trust.

Reputation-Based Trust

In contrast to certification-based trust, reputation-based trust provides a way of building trust through social control without trusted third parties.

In the real world, this type of trust is common: A trusts B because B´s peers have a high opinion of B. Something similar could be observed with the advent of the Web 2.0. Collaborative actions, as one of the driving factors of this phenomenon, lead to, for example, ratings pages about hotels[1] or auction sellers/buyers.[2]

Trust Delegation

Trust delegation implies that for a resource R controlled by A, A trusts B to make decisions on A's behalf. A typical example is a person delegating all financial decisions to a financial advisor.

Infrastructure Trust

This type of trust refers to basic trust in the underlying infrastructure. For instance, A has to trust the network, his workstation, local servers, etc. To address this problem, a consortium led by Intel and Microsoft and consisting of several leading technology companies and research centers formed the Trusted Computing Platform Alliance.[3]

2.4.4 Trust Management

The term *Trust Management* was first introduced by Blaze et al. (1996). They defined it as "a coherent intellectual framework […] for the study of security policies, security credentials, and trust relationships" (Blaze et al. 1996, p. 164). In other words, trust management is intended to determine the trustworthiness (or, negatively expressed, the risk) of A to interact with B in a specific context. Therefore, Trust Management always has to deal with a balance between increasing trust and thus exposing more vulnerabilities, or decreasing the level of trust in favor of more security mechanisms leading to less efficiency and making transactions between two entities virtually impossible.

Over the decades, several solutions have been proposed, such as PolicyMaker (Blaze et al. 1998; Chu et al. 1997), REFEREE (Chu et al. 1997) or KeyNote (Blaze et al. 1998). The most influence in current internet applications have public key certificates relying on Public Key Infrastructures (PKI). With the advent of asymmetric cryptography (public key cryptography; see Section 2.3.2), arbitrary communication partners can exchange encrypted and/or signed messages, as long as the public keys of the communication partners are known. This implies the need for a mechanism to securely exchange public keys between communication partners and to verify the validity of a received public key. Since the knowledge about these keys is a prerequisite for communicating securely, a secure method for the key exchange is needed. Sending a public key by email or downloading it from a web page via a potentially insecure communication channel such as the Internet does not fulfill these requirements: a potential man-in-the-middle could easily alter the key without leaving a trace.

Public Key Infrastructures (PKIs) solve this problem by introducing Certificate Authorities (CAs). These CAs are assumed to be trustworthy either per definition or within a certain security domain (Kanneganti & Chodavarapu 2008). CAs issue digital certificates for owners of asymmetric key pairs, asserting that the actual public key belongs to a certain owner. Such a certificate is signed by the CA to ensure integrity and authenticity (see Section 2.2).

Once a communication partner trusts a CA, it is assumed that he also trusts all certificates being issued by this CA. So for exchanging public keys, communication partners may then exchange

[1] Tripadvisor website: http://www.tripadvisor.com/.

[2] Ebay website: http://www.ebay.de/.

[3] Trusted Cloud Computing website: http://www.trustedcomputing.org/.

their certificates including those public keys and the related identity information provided by the CA. If there is a valid signature of the CA on this certificate, the communication partners can be sure to have received the right public keys.

Of course, PKIs still need the public keys of the CAs to be known, but this affects only a very limited number of public keys (those of the CAs) instead of all keys of all communication partners. The so-called root certificates, including the identity and the public keys of the most popular CAs, are already included in many software components, such as Browsers or Operating Systems.

Besides the central CAs, a PKI has a Registration Authority (RA) for registering new certificates. A Certificate Revocation List is also necessary to list all invalid certificates. A certificate becomes invalid if it expires or is actively revoked due to abuse. Additionally a PKI has a Directory Service to list and search for certificates (Schmeh 2009).

Typical standards for PKIs are ITU-T X.509 (ITU-T 2005).

It is important to note that certificates can be chained. For instance, a CA named *rootCA* can issue a certificate for entity e_1. If an entity e_2 wants to access a resource *r* that only trusts certificates from rootCA but no local certificates from e_2,, e_2 can get a certificate either directly from *rootCA* or also let its certificate being signed by e_1. Entity *r* then has to go through the chain ($e_2 \rightarrow e_1 \rightarrow rootCA$) until it finds an entity that it trusts.

2.4.5 Trust Models

When it comes to trust between entities, it is possible to distinguish between different kinds of trust models on the basis of PKIs. According to the already existing trust relationships of the entities, OASIS (2005c) describes the following trust models (Figure 1).

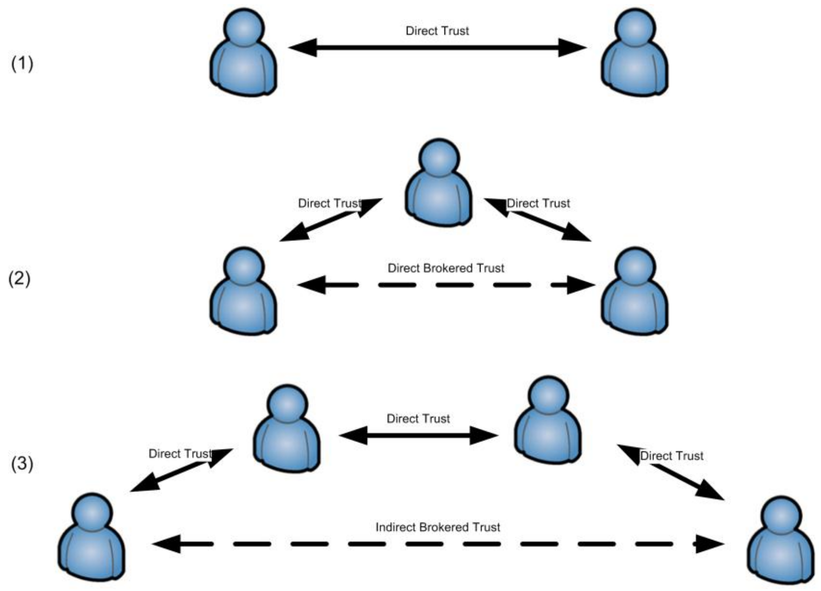

Figure 1. Different trust relationship models: (1) Direct Trust, (2) Direct Brokered Trust, (3) Indirect Brokered Trust.

Direct Trust

Communication partners have a direct trust relationship if there is a pre-established trust relationship between these partners. This already established trust relationship can for example be technically based on exchanged public keys. Because these two partners already trust each other there is no need to exchange any additional information as a prerequisite for secure communication. Thus, no CA is needed for establishing trust (see Figure 1).

Direct Brokered Trust

If two communication partners have no pre-established trust relationship, trust can be established by brokering. Trust brokers in PKIs are typically CAs, where a commonly accepted and trusted CA vouches for the identity of both communication partners (see Figure 1).

Indirect Brokered Trust

Indirect brokered trust is an extension of direct brokered trust, where there is no single CA which has a trust relationship to both communication partners. Thus, trust has to be brokered between several CAs, resulting in a trust chain between the communication partners (see Figure 1).

Privacy has been already introduced in Section 2.2. How privacy is dealt with is usually expressed in terms of use on the basis of certain legal frameworks such as Scheurle et al. 2002. However, a consumer must trust the entity which is expressing the terms of use that it really complies with them. In Web Service environments certificates and privacy seals have been introduced to establish trust (Benassi 1999; Head & Hassanein 2002). Trust is established via certification-based trust as explained in Section 2.43. Examples are on the mass market level TRUTe,[4] TrustedShops[5] or TÜV S@fer-Shopping.[6]

The basic principle behind acquiring those seals and certificates is that a generally trusted authority guarantees a certain compliance level at the certified services side in terms of privacy. The certification process typically involves a self-assessed questionnaire. This questionnaire is reviewed by the trust organization and in terms of compliance to the trust organization’s rules a trust seal is issued. On a technical level, a B2C e-commerce website displays an icon, which identifies the trust seal. Users can click on such a seal and get redirected to the trust organization’s website, where further information can be displayed in proprietary and often not machine readable format. If the user trusts the trust organization, it is assumed that it also trusts the certified services.

The Web 2.0 movement mainly relies on certified but quantifiable user ratings (Tredinnick 2006) as a combination of certification-based trust and reputation-based trust (see Section 2.4.2). Commonly known examples are eBay vendor ratings or hotel ratings on TripAdvisor.

Other stricter privacy compliance standards are for instance ISO 27001/27002, ISO 20000 or TÜV e-commerce seals. The certification process is performed on side by certified security accountants at security audits (Gora 2009). However, issued certificates resulting of that process can only be reviewed offline. These seals and certificates are typically non comparable. However, there are sub standards with stricter rules for certain areas such as ISO 27011 as substandard of ISO 27002.

[4] TRUSTe website: http://www.truste.com

[5] Trusted Shops website: http://www.trustedshops.de/

[6] TÜV Safer-Shopping website: http://www.safer-shopping.de/

Related work has been achieved on combining several privacy compliance aspects. In terms of Web Services, Blake & Cummings (2007) showed how SLA aspects can be combined but did not take privacy aspects into account. The main focus was put classical SLA properties such availability and resiliency. Composition of privacy aspects was addressed by Xu et al. (2006), defining a fine granular model of privacy aspects. Negotiation of privacy compliance between consumer and service was possible with the proposed approach. Hassan & Logrippo (2011) focused especially on legal aspects and the creation of ontologies for representing privacy aspects. Platform for Privacy Preferences (P3P) (Reagle & Cranor 2007) is a protocol for checking consumer privacy requirements against privacy offers. However, trust is assumed between consumer and service for compliance to the stated privacy requirements. Peyton & Nozin (2004) and Peyton et al. (2007) propose to use audit trails for privacy compliance checking with regard to composite Web Services.

2.5 Access Control Models

Authorization as described in Section 2.2 is based on a certain set of rights. The decision is based on the rights which are maintained in specific access control models. The most important classical access control models are Protection Matrices, Discretionary, Mandatory and Role-Based Access Control (Schmeh 2009). These models are described below. Credential-Based Access Control is also described as a newer model for authorization.

Protection Matrices

Protection Matrices, as introduced by Harrison et al. (1976), are abstract representations of access control policies. Such access control policies provide the basis for an access control model by defining the rights and actions a subject is allowed to perform on a resource. Protection Matrices are realized as a two-dimensional array which holds subjects as rows and protected resources as columns. Therefore, each entry (s_i,r_j) defines what subject s_i is allowed to do with resource r_j.

Typically many entries are empty for a large set of subjects and resources, due to the fact that not every subject is usually allowed to perform an action on each resource. Different structures have been developed as an optimization of Protection Matrices. The most important one is the concept of Access Control Lists (ACL). ACLs store only the relevant entries and ignore empty entries. Therefore, an ACL is associated with a resource and defines for each subject which action the subject is allowed to perform on a given resource.

Discretionary Access Control

The Discretionary Access Control (DAC) model is based on the identity of subjects and ownership of objects (Sandhu & Samarati 1994). Subjects are defined as owners of objects (resources) and have discretionary authority over who has access to those objects. This means that DAC is based on the administration of owner-based access rights. Therefore, DAC allows the delegation of rights over an object to other subjects.

Mandatory Access Control

In the Mandatory Access Control (MAC) system, the authorization decision is taken on the basis of a certain set of rules and polices enforced by a central authority. The most important implementation is the Bell–LaPadula model (Bell & LaPadula 1973). This model defines a set of access control rules which define security labels on resources and clearances for subjects on resources. In detail, it defines a hierarchical order of security labels such as *Top Secret* at the top, down to the least sensitive such as *Unclassified*. Subjects are allowed to see all resources for their clearance level and below.

Role-Based Access Control

Role-Based Access Control centers on the idea of enforcing access control according to specific pre-defined policies, which assign certain rights to specific roles (Sandhu & Samarati 1994).

Multiple subjects can be aggregated into a certain role and, vice versa, a subject can have multiple roles. Role hierarchies with inheritance are also possible, which ease the administration overheads. Specific roles (e.g. administrator, firefighter, etc.) are granted all rights associated with the role, such as access to specific resources.

Credential-Based Access Control

To reflect the developments in current IT Infrastructures towards distributed, decentralized, dynamic and interoperable environments, Credential-Based Access Control was introduced (Ardagna et al. 2010; Agarwal et al. 2004) on the basis of work from Bonatti & Samarati (2000) and Seamons et al. (1997).

Credentials are understood in this context as digitally signed documents which can be transmitted via untrusted channels like the Web; see for example Biskup & Karabulut (2003).

In Credential-Based Access Control Models, a credential is sent from a client to a server and the server takes its authorization decision on the basis of the presented credential. Depending on the request and the presented and trusted credential, access is either granted or not.

2.6 Digital Identities

The term *digital identity* is defined as “the digital representation of the information known about a specific individual or organization” (Bertino et al. 2009, p. 80). In addition, an entity may have several (digital) identities, depending on the context and role which it represents (Pfitzmann & Hansen 2005). Various forms are known to represent digital identities. In SOA environments, profiles of attributes (known as *identifiers*) are typically used. These profiles are stored and maintained by Identity Providers (IdPs). The literature distinguishes between strong and weak identifiers. Strong identifiers uniquely identify an entity, such as a MAC or IP address. Weak identifiers are not necessarily unique, such as last names (Schmeh 2009).

2.6.1 Digital Identity Management Models

Bertino et al. (2009, p. 80) define Digital Identity Management as “the set of processes, tools, social contracts and supporting infrastructure for creating, maintaining, utilizing and terminating a digital identity”. It is possible to distinguish between the roles of a *Client,* who needs to present its Digital Identity, an *Identity Provider* (IdP), storing and maintaining the Client’s Identity, and a *Service Provider* (SP), who provides services and needs to verify the Client’s identity.

There are three different Digital Identity Management models that make use of these roles: Isolated, Centralized and Distributed Digital Identity Management (Ahn & Lam 2005). In the isolated Digital Identity Management model, users have different identities at different SPs. This may result in inconsistency, replication and management overheads, since all of the credentials, such as usernames and passwords, have to be remembered and maintained.

The centralized Identity Management model utilizes only a single (central) IdP and a set of SPs, all trusting the IdP. Even though seamless work is guaranteed, the central IdP represents a single point of failure. Besides, all clients and SPs need to agree to use the same IdP, which results in a closed environment. The Liberty Alliance, as briefly introduced later in this section, represents an implementation of this model which also takes the federation of IdPs into account.

The distributed Identity Management model deals with the idea of distributed but federated IdPs. The IdPs are federated, which means that they trust each other and, especially, identities vouched for by other IdPs. Thereby, no centralized IdP for managing identities is necessary, because the various components are distributed following the SOA approach. Typical aspects of this model are that clients have their identity certified by at least one IdP, which is typically not the IdP of the SP. Another benefit of this model is Single Sign-On (SSO) (De Clercq 2002). SSO allows the client to sign-on once with its IdP and use the provided identity with all SPs (if they trust the IdP). WS-Federation presents a specification of distributed and federated Identity Management and is explained in the following. Shibboleth as an emerging trend for Identity Management is also explained in in the following.

Liberty Alliance

The Liberty Alliance[7] is a standardization organization founded in 2001. It provides specifications for Digital Identity Management based on the idea of a *circle of trust*. A circle of trust is the concept of mutually trusting IdP and SP. The building blocks are the Liberty Identity Web Service Framework (ID-WSF) and the Liberty Identity Services Interface Specifications (ID-SIS). The ID-WSF defines a SOAP-based model for discovery and registration of IdPs. In addition, different interfaces are specified, as for instance for administration of user profiles.

WS-Federation

WS-Federation (Bajaj et al. 2003) is a specification created by EA Systems, BMC Software, CA Inc., IBM, Layer 7 Technologies, Microsoft, Novell and VeriSign. It build on the WS-* stack and especially WS-Trust (OASIS 2005c). In particular, the WS-Trust specification is extended to accept and translate digital identities from other domains into identities trusted and understood in their own domain. This aspect allows Single Sign-On in particular, because a client has to obtain an Identity only once and the WS-Federation services try to translate the identity if possible. The brokered trust principle described in Section 2.4.5 is fundamental for this approach. An own metadata model is defined to support this behavior.

In addition, single log-out and pseudonyms are also supported.

Shibboleth

Shibboleth[8] is an open middle-ware architecture introduced by the Internet2 consortium. It provides means for Identity Management but also for authorization purposes. SSO is a central aspect of Shibboleth with SAML 2.0 as its technical backbone.

The typical workflow follows the principles of:

[7] Liberty Alliance website: http://www.projectliberty.org/.

[8] Shibboleth website: http://shibboleth.internet2.edu/.

1. A client, when accessing a Shibboleth-protected SP, is redirected to a page where the client has to select its IdP.
2. The client then redirected to its IdP to authenticate with its (if on the list).
3. It is then redirected back to the SP.

Research in broadening the scope has been undertaken by Barton et al. (2006) and Spence et al. (2006) but seems to be limited to specific application domains. Shibboleth is therefore mainly used in higher education user-centric scenarios (Kanneganti & Chodavarapu 2008).

3 SECURITY ARCHITECTURE

A security architecture is never self-contained, rather to being an overlay to a domain-specific architecture. OGC defines such a service architecture, which is based on the OWS Common standard (OGC, 2006) and reference model (OGC, 2003) and extended by other implementation standards, such as WMS (OGC, 2002). Based on the security concepts introduced in Section 2, this chapter demonstrates a commonly used architecture for securing georesources provided via OGC Web Services. In particular a static model is presented describing the overall architecture and a dynamic model which describes the behavior.

3.1 Static Model

The static model as the general architecture is depicted in Figure 2. It follows the commonly used XACML pattern (OASIS 2005a).

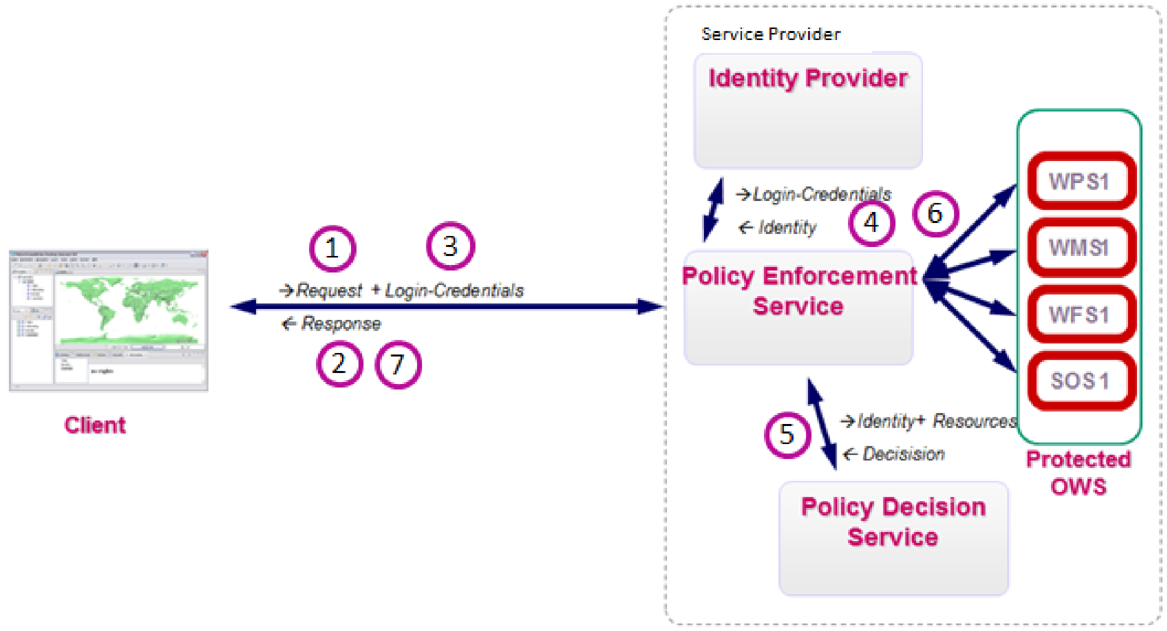

Figure 2. Static and Dynamic Model.

From the static model, it can be distinguished between five components.

Client
The Client Component is an ordinary client capeable of requesting a non-secured OWS providing a georesources such as a WMS provinding a layer.

In order to access a protected OWS it must be able to understand and fullfill certain security preconditions. Such preconditions, depending on the authentication and authoriation method used, describe what a client has to do in order to access the protected service such as the login credential format, encryption mechanism etc.
One typical entry point for such preconditions are the <*AccessConstraints*> element in the GetCapabilities Response of a OWS as mandated by OGC Common (OGC 2006). However, the <*AccessConstraints*> element allows only plain text, which is not sufficient for machine-readable use due to the fact that the element is explicitly not supported to hold other XML elements (see element definition in OGC 2006) necessary for machine readability.
Another alternative is to use simple HTTP status codes as used by HTTP-Basic Authentication. In this case, a HTTP status code 403 is returned if a client tries to request a protected resource without authentication. In this case, the client needs to understand HTTP status codes and apply the HTTP-Basic Authentication scheme. For HTTP-Basic Authentication, typically username and password have to be provided.

The client also has to put the actual login credentials in the request. In case of HTTP-Basic Authentication, username and password are put into the HTTP header acquired typically from a pop-up window directed at the end user.

Policy Enforcement Service
A Policy Enforcement Service manages the access to and from a secured OWS as a separate infrastructure service and thereby applies the security as a service principle (Hinton et al. 2005).

The protected OWS as the back-end for the Policy Enforcement Service, can easily be configured in a way (e.g. by means of a firewall) that it only allows access from its shielding Policy Enforcement Service’s IP address. Thereby, any communication directed at the protected OWS has to go through the Policy Enforcement Service security layer. Thus, the Policy Enforcement Point serves as a transparent Proxy component securing the back-end OWS without touching its implementation. To be transparent in regard to the Client component, the Policy Enforcement Service has the same interface as the secured OWS, but adds additional security functionality, such as issuing preconditions by means of amending the metadata and enforcing usage rights by means of the Policy Enforcement Service.

To manage access to and from a back-end service, a Policy Enforcement Service analyzes incoming requests for completeness in terms of fulfilling the requirements stated in the preconditions. It also extracts the login credentials and forwards them to the Identity Provider as well as the results from the Identity Provider and the requested georesources extracted from the request to the Policy Decision Service.
On the basis of the decision from the Policy Decision Service, the Policy Enforcement Service grants access to or denies access to a protected service based on the given request. A third option is to allow a request but filter the response as shown in the following example:

WMS A offers georesources X,Y,Z
User U is allowed to see/request only georesources X and Y but not Z.

In this case, the Policy Enforcement Service would filter a GetCapabilities response from A to show only X,Y in case user U is requesting. For a GetMap requested it would only allow user U to request georesources X and Y but deny access to resource Z.

Identity Provider

An Identity Provider allows clients to retrieve identity credentials which are trusted in the server domain as well as the transformation of incoming identities into an internal role identity.
In case of HTTP-Basic Authentication, no identity credentials have to be requested from an Identity Provider by the client, because the username and password are assumed to be known in advance.
In this case the Identity Provider authenticates the requesting client with the given username and password and issues an internal identity/role description.

Policy Decision Service
A Policy Descision Service is responsible for authorizing incoming requests. For this reason it receives the requested georesource and the internal id/role description. Via an underlying authoriation model (see Section 2.5) it determines if the requesting user/role is allowed to fetch the requested georesources. On this basis, a decision is issued.

Protected OWS
The protected OWS is a plain OGC Web Service such as a WMS or WFS without any security capabilities. It is configures by e.g. means of a firewall to be only requestable from the Policy Enforcement Service.
By keeping the OWS independent of security logic, COTS service can be simply reused.

3.2 Dynamic Model

Based on the static model described in the previous section, this section describes the behaviour and interactions of the different components for the HTTP-Basic authentication based SDI.

As for every OGC Web Service, a georesources has to be requested via an operation exposed by the service. For this reason, the client sends a plain request without knowing that it requests a protected resource (Figure 2, (1)). The Policy Enforcement Point component retrieves the request and denies access to the protected resource because no credentials for authentication and authorization are provided. Thus, the Policy Enforcement Point returns a message asking for username and password according to the HTTP-Basic Authentication scheme (2). The client has to understand the response and provide username and password in the HTTP-Basic Authentication scheme (3). A HTTP connection secured with SSL/TLS is preferably used to prevent others to read the transferred user name and password as clear text.
On the server side, the Policy Enforcement Point retrieves the message and after validating the integrity, it extracts the credentials. These credentials (username and password) are sent to the Identity Provider which validates them in a way that requesting username is a) known and b) the correct password is provided. On this basis, an internal ID is returned to the Policy Enforcement Point (4). It forwards this ID and the requested georesources to the Policy Decision Point (5), which validates that the requesting user is allowed to see the requested georesources. Based on the returned decision, the Policy Enforcement Point forwards the request to the protected georesources or denies access (6). If access is granted, the protected service returns the requested georesources to the Policy Enforcement Point, which forwards it to the requesting client (7). In a more fine grained authorization scheme, one option for the Policy Decision Point is to return a constrained access permission resulting in an obligation to the Policy Enforcement Point to filter the response, such as layer names from a GetCapabilities response or feature attributes from a GML response (see conceptual example in Section 3.1)

5 SUMMARY

This paper summarizes the basic concepts for authentication and authorization in spatial data infrastructures. A sample architectural approach is presented based on the commonly used HTTP-Basic Authentication scheme and Role-based Access Control.
It became clear that several SDI use cases require such an approach. However, it has to be taken into account that by introducing authentication and authorization in SDIs a significant overhead on several levels is created:

- On an **architectural level**, new services have to be introduced and maintained such as Policy Enforcement Service, Policy Decision Service and Identity Provider.
- On a **governance level**, roles, rights and credentials have initially be assigned to each user and maintained over time. A special focus has to be put on managing new users and expired users such as employees leaving a company and loss of credentials. A process has to be set up for noticing these events.
- On a **security level**, confidentiality of granted credentials on the client side has to be maintained and on the server side protection against different threat models (such as the Internet Threat Model (Rescorla & Korver 2003) or the STRIDE model (Swiderski & Snyder 2004)) has to be exercised.

These aspects have to be taken into account and thoroughly evaluated, if authentication and authorization should be introduced in SDIs.

From a practical point of view, a couple of frameworks for authentication and authorization in SDIs exist such as (non-exhaustive list):

- 52North Security Suite
- Conterra Security Manager
- Intevation Inteproxy
- Camp2Camp SecureOWS

Also some WMS/WFS vendors support natively authentication and basic authorization in their products (non-exhaustive list):

- Geoserver
- ESRI
- Intergraph

REFERENCES

Adleman, L., Rivest, R. L., & Shamir, A. (1978). A method for obtaining digital signatureand public-key cryptosystems. *Communication of the ACM*, *21*(2), 120-126.

Agarwal, S., Sprick, B., & Wortmann, S. (2004). Credential based access control for semantic web services. *Proceedings of the AAAI Spring Symposium-Semantic Web Services,* San Diego, CA. 44-52.

Ahn, G. J., & Lam, J. (2005). Managing privacy preferences for federated identity management. *Proceedings of the 2005 Workshop on Digital Identity Management*, Fairfax, VA. 28-36.

Ardagna, C. A., di Vimercati, S. D. C., Paraboschi, S., Pedrini, E., Samarati, P., & Verdicchio, M. (2010). Expressive and deployable access control in open web service applications. *IEEE Transactions on Services Computing, 99*(PrePrints)

Bajaj, S., Della-Libera, G., Dixon, B., Dusche, M., Hondo, M., & Hur, M. (2003). *Web services federation language (WS-federation) V1.1.* Technical Rreport, IBM Corporation, Microsoft Corporation, BEA Systems, Inc., RSA Security, Inc., Verisign, Inc.,.

Barton, T., Basney, J., Freeman, T., Scavo, T., Siebenlist, F., Welch, V., . . . Keahey, K. (2006). Identity federation and attribute-based authorization through the globus toolkit, shibboleth, gridshib and myproxy. *Proceedings of the 5th Annual PKI R&D Workshop.* NIST, Gaithersburg, MD. Retrieved 12/31, 2010, from http://grid.ncsa.uiuc.edu/papers/gridshib-pki06-final.pdf

Bell, D., & LaPadula, J. (1973). Secure computer systems : A mathematical model. *Journal of Computer Security, 4*(2), 229-263.

Benassi, P. (1999). TRUSTe: An online privacy seal program. *Communications of the ACM, 42*(2), 56-59.

Bertino, E., Martino, L., Paci, F., & Squicciarini, A. (2009). *Security for web services and service-oriented architectures*. New York: Springer.

Bishop, M. (2005). *Introduction to computer security*. Boston: Addison-Wesley.

Biskup, J., & Karabulut, Y. (2003). A hybrid PKI model with an application for secure mediation. *Proceedings of the 16th Annual IFIP WG 11.3 Working Conference on Data and Application Security,* Cambridge, England. 271-282.

Blake, M. B., & Cummings, D. J. (2007). Workflow composition of service level agreements. *Proceedings of the International Conference on Services Computing, 2007. SCC 2007.* 138-145.

Blaze, M., Feigenbaum, J., & Lacy, J. (1996). Decentralized Trust Management. *Proceedings of the Symposium of Security and Privacy , IEEE Computer Society Press, Los Alamitos, 1996.* 164-173.

Blaze, M., Feigenbaum, J., & Keromytis, A. (1998). KeyNote: Trust management for public-key infrastructures. *Proceedings of the 1998 Security Protocols International Workshop,* Camebridge, UK. 59-63.

Bonatti, P., & Samarati, P. (2000). Regulating service access and information release on the web. *Proceedings of the 7th ACM Conference on Computer and Communications Security,* Athens, Greece. 134-143.

Christianson, B., & Harbison, W. S. (1997). Why isn't trust transitive? In M. Lomas (Ed.), *Security protocols* (pp. 171-176). Berlin/Heidelberg: Springer.

Chu, Y. H., Feigenbaum, J., LaMacchia, B., Resnick, P., & Strauss, M. (1997). REFEREE: Trust management for web applications. *Computer Networks and ISDN Systems, 29*(8-13), 953-964.

Critchell-Ward, A., & Landsborough-McDonald, K. (2007). Data protection law in the european union and the united kingdom. In D. Campbell, & A. Alibekova (Eds.), *The comparative law yearbook of international business* (pp. 515-578). Alphen uan den Rhin: Kluwer Law International.

Daemen, J., & Rijmen, V. (2002). *The design of rijndael : AES - the advanced encryption standard*. Berlin u.a.: Springer.

De Clercq, J. (2002). Single sign-on architectures. In D. George, Y. Frankel & O. Rees (Eds.), *Proceedings of the international conference on infrastructure security* (pp. 40-58). London, UK: Springer.

Dimitrakos, T. (2003). A service-oriented trust management framework. *Proceedings of the 2002 International Conference on Trust, Reputation, and Security: Theories and Practice,* Bologna, Italy. 53-72.

Gollmann, D. (1999). *Computer security*. Chichester, N.Y.: Wiley.

Gone, M., & Schade, S. (2007). Towards semantic composition of geospatial web Services–Using WSMO in comparison to BPEL. *International Journal of Spatial Data Infrastructures Research,* (3), 192-214.

Gora, S. (2009). Security audits. *Datenschutz Und Datensicherheit-DuD, 33*(4), 238-246.

Grandison, T., & Sloman, M. (2000). A survey of trust in internet applications. *IEEE Communications Surveys and Tutorials, 3*(4), 2-16.

Grandison, T. W. A. (2003). *Trust management for internet applications.* (Dissertation, Imperial College).

Groot, R., & McLaughlin, J. (2000). *Geospatial data infrastructure: Concepts, cases, and good practice*. Oxford: Oxford University Press.

Hafner, M., & Breu, R. (2009). *Security engineering for service-oriented architectures*. Berlin u.a.: Springer.

Harrison, M. A., Ruzzo, W. L., & Ullman, J. D. (1976). Protection in operating systems. *Communications of the ACM, 19*(8), 461-471.

Hassan, W., & Logrippo, L. (2011). Validating compliance with privacy legislation. *In Press,*

Head, M. M., & Hassanein, K. (2002). Trust in e-commerce: Evaluating the impact of third-party seals. *Quarterly Journal of Electronic Commerce, 3,* 307-326.

Hinton, H., Hondo, M. & Hutchison, D. B. (2005). Security patterns within a service-oriented architectur. *IBM whitepaper.* Retrieved 12/31, 2010, from http://www.ebizq.net/topics/woa/features/6535.html

IETF (2000). *RFC 2828 - internet security glossary.* Retrieved 12/31, 2010, from http://tools.ietf.org/pdf/rfc2828

Imamura, T., Dillaway, B., & Simon, E. (2002). *XML Encryption syntax and processin.* W3C Recommendation.

ISO (1996a). *Information technology-open systems interconnection-security frameworks in open systems: Overview*. ISO/IEC 10181-1.

ISO (1996b). *Information technology-open systems interconnection-security frameworks in open systems: Authentication Framework.* ISO/IEC 10181-2.

ISO (1999). *Information technology-Securitytechniques-Evaluation Criteria for IT Security. Part 1: Introduction and general model.* ISO/IEC 15408-1.

ITU-T (2005). *Recommendation X. 509*. ITU-T.

Kanneganti, R., & Chodavarapu, P. (2008). *SOA security*. Greenwich, Conn.: Manning.

National Institute of Standards and Technology (NIST) (2010). *Glossary of Key Information Security Terms.* Retrieved 12/31, 2010, from http://csrc.nist.gov/publications/nistir/ir7298-rev1/nistir-7298-revision1.pdf

OASIS (2004). *Web services security: SOAP message security 1.0 (WS-security).* OASIS Standard.

OASIS (2005). *Web services trust standard 1.3 (WS-trust).* OASIS Standard.

OGC (2002). *OpenGIS web map service (WMS).* OGC Implementation Specification.

OGC (2003). *OpenGIS reference model.* OGC Abstract Specification.

OGC (2006). *OpenGIS web service common implementation specification (OWS-common).* OGC Implementation Specification.

Peyton, L., Doshi, C., & Seguin, P. (2007). An audit trail service to enhance privacy compliance in federated identity management. *Proceedings of the 2007 Conference of the Center for Advanced Studies on Collaborative Research*, 175-187.

Peyton, L., & Nozin, M. (2004). Tracking privacy compliance in B2B networks. *Proceedings of the 6th International Conference on Electronic Commerce*, 376-381.

Pfitzmann, A., & Hansen, M. (2005). *Anonymity, unlinkability, unobservability, pseudonymity, and identity management-a consolidated proposal for terminology.* Retrieved 12/31, 2010, from http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.28.pdf

Reagle, J., & Cranor, L. F. (2007). The platform for privacy preferences. *Communication of the ACM, 42*(2), 44.

Rescorla, E., & Korver, B. (2003). *Guidelines for writing RFC text on security considerations.* IETF RFC 3552.

Romberg, M. (2002). The UNICORE grid infrastructure. *Scientific Programming*, *10*(2), 149-157.

Rosado, D. G., Gutiérrez, C., Fernández-Medina, E., & Piattini, M. (2006). Security patterns and requirements for internet-based applications. *Internet Research*, *16*(5), 519-536.

Sandhu, R. S., & Samarati, P. (1994). Access control: Principle and practice. *IEEE Communications Magazine*, *32*(9), 40-48.

Scheurle, K. D., Mayen, T., & Bergmann, B. (2002). *Telekommunikationsgesetz (TKG).* Beck.

Schmeh, K. (2009). *Kryptografie verfahren, protokolle, infrastrukturen* (4., aktual. u. erw. Aufl. ed.). Heidelberg: dpunkt-Verl.

Schneier, B. (1999). *The twofish encryption algorithm : A 128-bit block cipher*. New York u.a.: Wiley.

Seamons, K. E., Winsborough, W., & Winslett, M. (1997). Internet credential acceptance policies. *Proceedings of the Joint Conf. on Declarative Programming, APPIA-GULP-PRODE'97,* Grado, Italy. 415-432.

Spence, D., Geddes, N., Jensen, J., Richards, A., Viljoen, M., Martin, A., . . . Trefethen, A. (2006). Shibgrid: Shibboleth access for the uk national grid service. *Proceedings of the Second IEEE International Conference on e-Science and Grid Computing,* Amsterdam, The Netherlands. 71-75.

Swiderski, F., & Snyder, W. (2004). *Threat modeling*. Redmond, WA: Microsoft Press.

Tredinnick, L. (2006). Web 2.0 and business. *Business Information Review, 23*(4), 228-232.

W3C (2002). *XML-signature syntax and processing.* W3C Recommendation.

Xu, W., Venkatakrishnan, V., Sekar, R., & Ramakrishnan, I. (2006). A framework for building privacy-conscious composite web services. *International Conference on Web Services, 2006. ICWS'06.* 655-662.

Yahalom, R., Klein, B., & Beth, T. (1993). Trust relationships in secure systems-a distributed authenticationperspective. *Proceedings of IEEE Computer Society Symposium on Research in Security and Privacy, 1993,* 150-164.

Yu, G., & Di, L. (2010). Secure service composition in sensor web. *Geoscience and Remote Sensing Symposium, 2009,* 457-460.