

On Similarities between SOA-Based Web Service and Smart Card Application for Ease of Understanding and Securing the Former

Zheng Jianwu Liu Mingsheng Liu Hui

Department of Information Engineering,
Shijiazhuang Railway Institute, Hebei 050043, China
{zhengjw, liums, liuhui}@sjzri.edu.cn

Abstract

This paper is to leverage familiarity with smart card application, i.e. understanding and practical experiences of implementing trustworthy smart card application, to uncover secret veil surrounding the SOA-Based Web Services, and further develop and implement effective strategies for achieving trustworthy Web Services. Aspects being compared, of smart card application and SOA-Based Web Services, include architectures, transmission models, needed security services, and essentials of achieving trustworthy applications. At last, the contemporary strategy, i.e. WS-Security, are introduced and explained for achieving trustworthy SOA-Based Web Services. Especially, its token services for attestation & authorization, and its cryptographic services for data confidentiality & integrity are detailed.

Keywords: Smart Card, SOA, Web Services, APDU, SOAP, Cryptographic Services

1 Introduction

Web Services are modular software applications built to run over the Internet, basing on open standards for definition, discovery, and interoperability, by enterprises for rapidly responding to changing market, enhancing competitive power, and pursuing business profit. Because Service-Oriented Architecture (SOA) is capable of simplifying the application development and integration by providing an architectural framework for building, integrating and deploying, using the same set of standards that enable basic Web Services, SOA is suggested and recommended as underlying architecture for establishing self descriptive, reusable, and loosely coupled Web Services.

However, the exchange of sensitive business information, the exposure of critical resources and services over

the Internet - an untrusted public network - gives rise to considerable security concerns, which in turn inhibit the implementation of SOA and deployment of Web Services.

In this paper, the smart card application is introduced for ease of understanding the SOA-Based Web Services, especially for borrowing systematic knowledge of the card application system and practical experiences of implementing trustworthy smart card application to help the practioners develop and implement effective security measures for securing Web Services.

2 Similarities between SOA-Based Web Service and Smart Card Application

2.1 Architectures

2.1.1 Architecture of Smart Card Application

Fig. 1 illustrates a smart card application system, which is made up of card application, interface device, smart card, and links connecting them. We can view the interface device as a part of the link connecting the card application and the smart card, i.e., two ultimate endpoints.

For intuitively understanding the card application system, as shown Fig. 1, we abstract the card application system and get a four-layer framework, depicted in Fig. 2, over which smart card transaction is executed.

2.1.2 Architecture of SOA-Based Web Service

Fig. 3 illustrates the architecture of SOA-Based Web Service, consisting of three different entities, i.e. Service repository, service requestor, and service provider, and three distinct operations, i.e. creating, provisioning, and using Web Services respectively. Three main

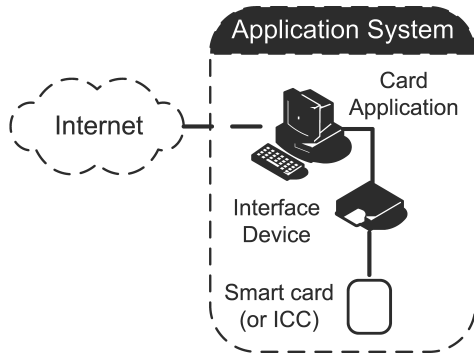


Figure 1. Smart Card Application System

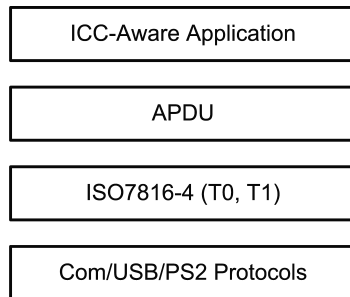


Figure 2. Layered Framework of Smart Card Application

technologies are developed specifically for implementing SOA-Based Web Services, i.e. WSDL, UDDI and SOAP.

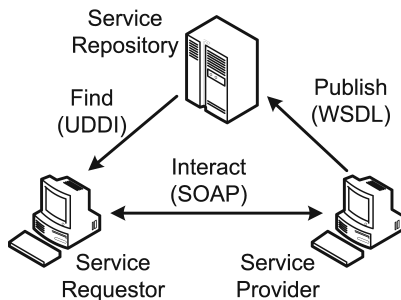


Figure 3. Architecture of SOA-Based Web Service

SOAP: Simple Object Access Protocol [1], which builds on XML [2] and supports the exchange of information in a decentralized and distributed environment. SOAP enables platform-neutral and language-independent interfaces to applications. SOAP message will be detailed Section 2.4.2.

WSDL: Web Services Description Language [3], which defines methods for creating detailed descriptions of Web services. Specifically, it is utilized to encapsulate functionalities implemented with not necessarily standard-compatible or proprietary techniques, for providing platform-neutral and language-independent services across the Internet.

UDDI: Universal Description Discovery and Integration [4], which provides a method for publishing service descriptions so Web services can be located and accessed by other Web services. Specifically, it is a mechanism for discovering where required Web Services are provided and who provides the related services.

The architecture of SOA-Based Web Services can also be abstracted as depicted in Fig. 4.

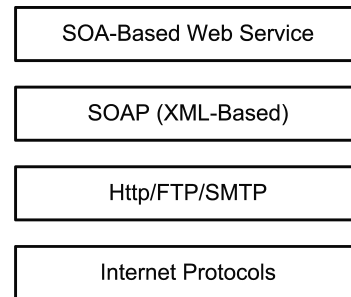


Figure 4. Layered Framework of SOA-Based Web Service

Comparing layered frameworks depicted in Fig. 2 and Fig. 4, it is surprised to discover that they are essentially alike to each other. Namely SOA-Based Web Services and smart card applications are built on frameworks nearly identical. This is the underlying reason that we introduce the smart card application as reference.

2.2 Transmission Models

As the architectures of smart card application and SOA-Based Web Service are nearly identical, it is natural to anticipate that transmission models of them are similar to each other.

2.2.1 Transmission Model of Smart Card Application

According to Fig. 1 and Fig. 2, we can conclude the transmission model of smart card application, as depicted in Fig. 5.

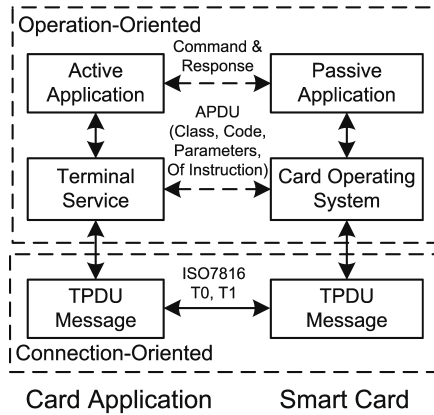


Figure 5. Transmission Model of Smart Card Application

It is instructive to view the smart card as three separate components depicted at the right side of the Fig. 5, especially for intuitively understanding processes for transaction exchange.

2.3 Transmission Model of SOAP-Based Web Services

According to Fig. 3 and Fig. 4, We can conclude the transmission model of SOA-Based Web Services as depicted in Fig. 6.

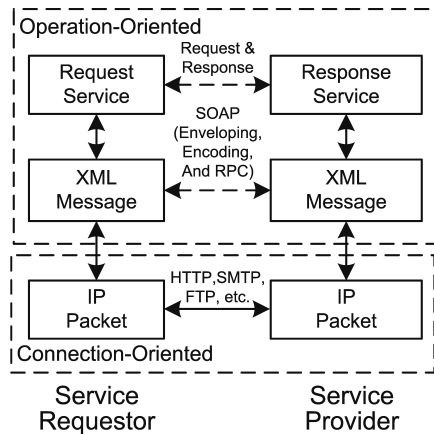


Figure 6. Transmission Model of SOA-Based Web Services

By Fig. 5 and Fig. 6, it is evident that two transmission models perfectly resemble each other.

Service interaction in SOA-Based Web Services incorporates two types of implementing blocks, i.e. application-oriented block and connection-oriented

block. The former block guarantees the flexibility and freedom for the enterprise to be responsive to changing market and business, and the later block ensure the enterprise to leverage IT infrastructures already created and implemented, namely Web Services can be delivered with existing infrastructures. Obviously, the former is needed for achieving operation-oriented security, as far as the security is concerned.

2.4 Application-Oriented Message Structures: SOAP vs. APDU

Similarities between transmission models of SOA-Based Web Services and smart card application have been illustrated, as depicted in Fig. 5 and Fig. 6, we would like to further exhibit the similarities of structured messages (application-oriented) exchanged in SOA-Based Web Services and smart card application, respectively.

2.4.1 Anatomy of APDU Message

As shown in Fig. 5, the card application (actively) initiates the transaction and sends a command to smart card, and the smart card (passively) returns a response after processing the received command to the card application. A specific response corresponds to a specific command, referred to as a command-response pair. An application protocol data unit (APDU) contains either a command data unit (C-APDU) or a response data unit (R-APDU).

Fig. 7 depicts the C-APDU message, consisting of a mandatory header of four bytes, *CLA*, *INS*, *P1* and *P2*, and a conditional body ($[\dots]$) of variable length, i.e. *Lc*, *IDATA* and *Le*.

$$[CLA][INS][P1][P2] [Lc] [IDATA] [Le]$$

Figure 7. Command APDU Message

Fig. 8 depicts the R-APDU message, consisting of a conditional body *ODATA* of variable length and a mandatory trailer of two bytes. Two mandatory bytes are status bytes *SW1* and *SW2*, which code the status of the receiving entity after processing the received command APDU.

$$[ODATA] [SW1][SW2]$$

Figure 8. Response APDU Message

Please refer to reference [5] for detail descriptions about C-APDU and R-APDU.

2.4.2 Anatomy of SOAP Message

In a SOA-Based Web Service, as shown in Fig. 3 and Fig. 6, SOAP messages are exchanged through multi hops (ultimate service endpoints and intermediaries). A SOAP message is an envelope that has child elements of an optional header and a mandatory body containing core service information. The basic form of a SOAP message is identified as an XML 1.0 document, as illustrated with the XML document below, with `Envelope` element qualified with the namespace `http://www.w3.org/2003/05/soap-envelope`.

```
<?xml version='1.0' ?>
<env:Envelope xmlns:env=
  'http://www.w3.org/2003/05/soap-envelope' >
<env:Header>
...
</env:Header>
<env:Body>
...
</env:Body>
</env:Envelope>
```

A SOAP message consists of two key parts, i.e. SOAP message header and SOAP message body.

1. **SOAP Message Header:** which is optional. It contains zero or more namespace-qualified elements. Unlike the SOAP message body, which may not be modified, the message header gives all intermediaries the right of accessing (read and write) to its child elements.

Each header element will be processed by at most one service node, however, others may examine headers not targeted at them. A service node can add header elements for subsequent service nodes before forwarding the SOAP message, however, it should delete any header elements targeted at it.

2. **SOAP Message Body:** which is classified as SOAP request and SOAP response according to the functional characteristics of the message.

SOAP Request: which may contain zero or more child elements. If multiple child elements are present, they represent a single unit of work, multiple units of work, or some combination of work and data. Message elements are analogous to paper documents that have an understood XML-Based structure.

SOAP Response: which can contain a document, an RPC response, or a SOAP fault. SOAP fault is a special form of SOAP response, and is the only body child elements that are defined by SOAP. SOAP faults are generated in response to errors or to carry other status information.

3 Securing IT Infrastructures

When considering to secure SOA-Based Web Services, practical experiences in securing smart card application are valuable, especially for making clear that what lower-level security targets should be achieved, what contemporary technologies should be implemented, and what contemporary strategies is proper to be selected for effectively securing SOA-Based Web Services.

3.1 Related Security Concerns

In smart card application, it is required that both the card application and the smart card are legitimate communication parties, privacy & integrity of transaction information exchanged between them should be guaranteed, and the two parties should implement effective measures for frustrating potential attacks originated by the adversaries, such as replay attack.

In the similar way, it is required that both the service requestor and the service provider are legitimate communication parties, privacy & integrity of service information exchanged between them (through multi-hops and across the Internet) should be guaranteed, and effective measures should be taken to frustrate possible attacks originated by the adversaries. Needed security services are as follows.

- Identity legitimacy of two ultimate service endpoints should be assured.
- Confidentiality & integrity of service messages exchanged should be guaranteed.
- Policy for managing authorization and access control should be implemented.
- The power to fighting attacks, for example replay attack, denial-of-service, phishing attack and so on, is also required in SOA-Based Web Services.

3.2 Essentials of Securing IT Infrastructures

According to the “Trustworthy” notion presented in references [6][7], we can conclude that following two things should be accomplished for successfully securing IT infrastructures, namely to implement effective measures for providing the needed security services mentioned above, specifically for solving security problems facing nowadays networked information systems (e.g. SOA-Based Web Services, smart card application systems).

1. Constructing trustworthy networked endpoints via proper mechanisms for identity authentication and authorization, which assures identity legitimacy of intending communication parties and proper right allocation.
2. Establishing trustworthy communication channel via effective security strategies (protocols), which guarantees the trustworthy transmission, i.e., data privacy & integrity, and power to fighting attacks.

As an instance, the essential tasks of securing SOA-Based Web Services are to complete source authentication and/or peer identity authentication, and to establish trustworthy service channel between service request application and service response application as shown in the transmission model depicted in Fig. 6.

Fig. 9 depicted the scenario of anticipated trustworthy service interaction in SOA-Based Web Services.

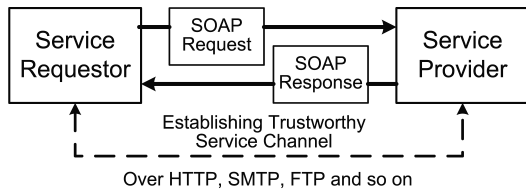


Figure 9. Anticipated Trustworthy Service Interaction

4 Contemporary Technologies and Strategies for Securing IT Infrastructures

In order to open the window of designing an effective strategy for achieving trustworthy SOA-Based Web Services, we first exemplify a strategy for achieving trustworthy smart card transaction (for more detail, please refer to [8]),

4.1 Achieving Trustworthy Smart Card Application

4.1.1 Protocol for Trust Negotiation: STS

Authenticating trustworthiness of the card application and the smart card is to convince them that the card application (resp. smart card) is really the party it claims to be by evidencing notarized document, such as public certificate signed by a proper (publicly recognized) certificate authority.

W. Diffie's STS (Station to Station) protocol [9] supplies a complete and detail solution to tasks of constructing trustworthy endpoints in securing smart card application. Please refer to [9] for further detail.

4.1.2 Strategy for Trustworthy Smart Card Transaction

Because the strategies for securing command APDU and response APDU can be built on the same idea, only strategy, specifically algorithmic steps, for securing command APDU transmission is detailed.

Assuming that the original C-APDU P is expressed as

$$[CLA][INS][P1][P2] [Lc] [IDATA] [Le] .$$

1. If Lc field is absent in P , insert a zero byte to P at Lc position (directly after $P2$). Original message P is updated as

$$[CLA][INS][P1][P2][Lc] [IDATA] [Le] .$$

2. Call MD5 to hash the message P (MD5 is selected just for exemplification, other proper hash algorithms are also Ok.),

$$H = MD5(P) .$$

H is the output message digest, whose length is exactly 16 bytes.

3. Concatenate output message digest H with $IDATA$ in original message P . Concatenating operation is illustrated as

$$IDATA' = H || IDATA .$$

4. Concatenate a transaction sequence number byte SN , and $IDATA'$ is modified as

$$IDATA' = SN || IDATA' .$$

Sequence number for transaction, SN , is introduced for providing anti-counterfeiting service. Sequence number can be negotiated with the use of authentication.

5. Call message encryption function $GE()$ to encrypt $IDATA'$ above with secret session key K . Encryption is expressed as

$$C = GE(IDATA', K) .$$

$GE()$ should be defined according to symmetric cryptography, such as DES and AES. Where does the session key K come from is discussed in Section 4.1.1.

6. Construct a new command APDU P' . P' is constructed as

$$P' = [CLA][INS][P1][P2][Lc'][C] [Le] .$$

Four mandatory bytes are taken directly from original message P . Value of Lc' in P' is different from that of Lc in original message P , Le field in P' is the same as Le field in original P ; if Le is absent in P , Le will not be present in P' , otherwise, Le will be present in P' .

7. Send P' to smart card instead of original command P .

With the use of symmetric cryptography, message digest and sequence number, implementing the strategy above can therefore guarantee data privacy & integrity, anti-counterfeiting, and structure soundness of APDU message as well, for smart card transaction.

4.2 Achieving Trustworthy SOA-Based Web Services

The WS-Security specification [10] is one of standards defined by OASIS (2004, more precisely, it is outlined, developed, and driven by a collaboration between IBM and Microsoft, and now being developed by OASIS consortium.) and provides a mechanism for addressing end-to-end security, specifically for data integrity, confidentiality, and data origin authentication features within a SOAP message. As well, WS-Security defines how to associate security tokens with SOAP messages.

As a result, WS-Security in essence is a systematic strategy for achieving trustworthy SOA-Based Web Services, which can be implemented for dealing with security problems facing Web Services.

For the compactness of this paper, WS-Security-Based SOAP Message will not be illustrated. However, it is recommended that you can refer to Section 11 (Extended Example) of WS-Security Specification [10] for thoroughly understanding the element-based structure, how security tokens are attached to the message, how signature and encrypted data are embedded in the SOAP message (how signature and encrypted data are created and represented is specified in XML Encryption and XML Signature specifications), and also the reference models (to security tokens, encrypted data, etc.) , and so many other things.

4.2.1 Token Services for Proof of Identity and beyond

As with the STS implemented for attestation in smart card application, WS-Security defines a generic mechanism for associating security tokens with the message. Specifically, it defines the $\langle wsse:Security \rangle$ header as the mechanism for conveying security-related information with and about a SOAP message. "Associating a security token" means that one or more tokens are included in $\langle wsse:Security \rangle$ headers in the message and that a referencing mechanism is introduced to refer to these tokens. Tokens generally are either identification or cryptographic material, i.e. security token is a representation of security-related information, such as X.509 certificate, Kerberos tickets and so on. Therefore,

security tokens can be utilized to identify service endpoint and assert its right.

4.2.2 Cryptographic Services for Data Privacy & Integrity and so on

As with the strategy mentioned in Section 4.1.2, which is implemented in smart card application for transaction privacy & integrity, and anti-counterfeiting, WS-Security makes use of the XML Signature [12] and XML Encryption [11] and defines how to include digital signatures, message digests, and encrypted data in a SOAP message. Moreover, the specification goes beyond these two underlying specifications by tailoring them for manipulating SOAP messages, as with the algorithmic steps demonstrated in Section 4.1.2.

Message confidentiality & integrity are provided by leveraging XML Encryption and XML Signature in conjunction with security tokens (which may contain or imply key information) to ensure that messages are transmitted without modification, and to keep messages confidential.

For instance, adversaries cannot read or understand the kernel service information, which is scrambled and encapsulated in `EncryptedData` element, by intercepting or tampering messages exchanged without authority of accessing the proper key. Legitimate communication parties can recover `EncryptedData` to its intelligible representation.

5 Conclusion

It is meaningful to compare SOA-Based Web Services and smart card applications, especially for comprehensively understanding the SOA-Based Web Services, erroneously viewed as peculiar. We find that the SOA-Based Web Service and smart card application system are similar to each other in great extent, or one can be viewed as the variant of the other.

By implementing WS-Security (and its friends, i.e. specifications of WS-* type), it is possible to achieve trustworthy SOA-Based Web Services, however, further work is needed to realize the transfer from the vision to the reality.

Acknowledgments

This work was supported by Natural Science Foundation of Hebei province, China, under Grant No.F2004000427 and Grant No.F2005000515.

References

- [1] SOAP Version 1.2 Part 1: Messaging framework, June 2003. <http://www.w3.org/TR/2003/REC-soap12-part1-20030624/>
- [2] Extensible Markup Language (XML) 1.0, 3rd ed. February 2004. <http://www.w3.org/TR/2004/REC-xml-20040204/>
- [3] W3C Web Services Description Language (WSDL) 1.1, March 2001. <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>
- [4] OASIS universal description, discovery and integration (UDDI), October 2003. http://uddi.org/pubs/uddi_v3.htm
- [5] ISO/IEC 7816, Identification Cards-Integrated circuit(s) cards with contacts, International Organization for Standardization.
- [6] Trusted Computing Group. <https://www.trustedcomputinggroup.org/>
- [7] Microsoft Corporation, NGSCB. <http://www.microsoft.com/resources/ngscb/default.aspx>
- [8] Zheng Jianwu and Liu Mingsheng. Establishing Trustworthy Terminal with Smart Card. Proceedings of the 2005 International Conference on Security and Management. Las Vegas, June 2005, pages 277-282.
- [9] W. Diffie, P. C. van Oorschot, and M. J. Wiener, Authentication and authenticated key exchanges, Designs, Codes and Cryptography, 1992, (2): 107-125.
- [10] WS-Security: SOAP Message Security 1.0 (WS-Security 1.0), March 2004. Available at: <http://docs.oasisopen.org/wss/2004/01/>
- [11] XML Encryption Syntax and Processing, December 2002. <http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/>
- [12] XML-Signature Syntax and Processing, February 2002. <http://www.w3.org/TR/2002/REC-xmlsig-core-20020212/>