January 2010


Available at: http://works.bepress.com/mutaz_al-debei/12

Bachar Alrouh  
Brunel University – West London  
Uxbridge, Middlesex, UK  
UB8 3PH  
Bachar.Alrouh@brunel.ac.uk

Mutaz M. Al-Debei  
Brunel University – West London  
Uxbridge, Middlesex, UK  
UB8 3PH  
Mutaz.Al-Debei@brunel.ac.uk

Gheorgita Ghinea  
Brunel University – West London  
Uxbridge, Middlesex, UK  
UB8 3PH  
George.Ghinea@brunel.ac.uk

ABSTRACT
In this paper, we provide a novel analytical hierarchy process decision-making framework for Web service security profiles. This framework aids in solving the dilemma of which Web service security profile is most fitting in a particular situation. This is because the developed framework allows architects and developers to take informed decisions following a systematic and manageable approach. In developing the framework, we followed the design-science research paradigm within which we incorporated a number of laboratory experiments.

Categories and Subject Descriptors
H.3.5 [Online Information Services]: Web-based services.

General Terms

Keywords

1. INTRODUCTION
Web services are software systems supporting interoperable machine-to-machine interaction over networks. This is because Web services are platform neutral and programming language independent technology. Despite its interoperability advantages, Web services raise security threats as they make it possible for applications from different parties to communicate with each other. Hence, message exchange security is an important issue in this context as messages need to be securely communicated and delivered intact.

The main challenge of Web services security is to understand and consider the risks of securing a Web-based service depending on the existing security profiles and simultaneously follow evolving standards in order to fill the gap in Web services security [7].

However, a considerable number of XML-based security profiles (e.g. User name with digest password; SAML authorization over SLL; etc.) are available today and choosing the most appropriate one in a given situation represents a complex dilemma for Web services architects and developers. This is due to the following three reasons: (1) There is no one supreme Web service security profile for all cases; i.e. different Web services security profiles achieving different requirements; (2) Applying different security profiles normally result in different quality measurements of a certain Web service [10, 16, 22]; and (3) Different organizations usually have different requirements and even the same organization may establish different requirements across different software applications.

Therefore, choosing the “best-suited” Web service profile to be deployed for a certain application in a particular organization is a complex undertaking decision-making task. This decision is usually uninformed as in many cases the decision is based on the sole experience of the developer. In some other cases, developers tend to use a profile that is (a) configurable based on the organization’s infrastructure; (b) developers are familiar with; or (c) supported by existing providers, without taking into consideration other security profiles. This is in fact represents the problem space.

Aiming to change this situation into a more favorable one, we in this paper develop a novel Analytical Hierarchy Process (AHP) decision-making framework for Web service security profiles following the design science research paradigm within which we incorporate a number of laboratory experiments. This framework represents the solution space which aims to make the decision in this context more informed. This is because such a framework makes the decision-making process concerning Web service security profiles more systematic, manageable, and effective by providing developers with an approach through which they can prioritize the security requirements, rank the available alternatives accordingly, and then select the profile that best fulfils the defined requirements. The use of AHP -amongst other decision-making methods- in this research is particularly useful. This is because this research partly but importantly requires measurements of intangibles in the form of expert judgments; and AHP is capable of doing that by converting the judgments into appropriate priority scales [20].

The remainder of this paper is structured as follows. In the next section, we provide a concise theoretical background regarding Web services along with their security profiles. We also discuss the AHP decision making approach. Thereafter, we present the research approach showing how we utilized the design science
paradigm for the development of the framework. In this section, we place a particular emphasis on describing the iterations through which the final artifact (i.e., AHP Framework) has been evolved. Furthermore, the final version of the framework is introduced showing and discussing its dimensions and elements. Before presenting the paper’s conclusions, we empirically validate and evaluate the developed framework through the use of a real life scenario so as to indicate the framework’s utility and value.

2. WEB SERVICES’ SECURITY PROFILES
Web services and applications open an organization’s business processes and data to distributed clients, thus it is highly important that messages communicated and received securely [7]. Indeed, the recipient of the message should be able to confirm its integrity and assure that it has not been modified while transferred. The message should also be delivered to the recipient confidentially where only authorized users can read it, know the identity of the sender, and determine the operation requested in the message [14].

For the purpose of securing the message exchanging process of an application’s Web services, developers can deploy one of the existing security profiles or mechanisms. In this paper, we use Metro Web service stack [23] to demonstrate the most common security profiles for Web services, as explained below.

- **Username Authentication with Symmetric Key (UA):** It is a symmetric key cryptography that is used for integrity and confidentiality. It relies on a single, shared secret key, generated at runtime and encrypted using the service’s certificate. The client does not possess any certificates on its own, but instead sends its username/password for authentication.
- **Username with digest passwords (UDP):** It is similar to UA, except that digest passwords are used in the username token and therefore is not required to be encrypted.
- **Mutual Certificates Security (MCS):** It is an asymmetric cryptography profile that adds security via authentication and message protection that ensures integrity and confidentiality.
- **Transport Security using SSL:** It protects the application during transport using SSL. Transport-layer security is provided by transport mechanisms used to transmit information over the wire between clients and providers, thus transport-layer security relies on secure HTTP transport (HTTPS) using Secure Sockets Layer (SSL). Transport security is a point-to-point security mechanism that can be used for authentication, message integrity, and confidentiality.
- **SAML Authorization over SSL (SA):** This profile attaches an authorization token with the message and uses SSL for confidentiality protection. In this profile, the SAML token is expected to carry some authorization information about end users. The sender of the token is actually vouching for the credentials in the SAML token.
- **SAML Sender Vouches with Certificates (SV):** This profile protects messages with mutual certificates for integrity and confidentiality. The sender vouches a SAML token for authorization.
- **STS Issued Token:** This profile enables the use of a single Security Token Service (STS) to establish a chain of trust between servers and clients; especially where service providers and requesters are in different managed environments and confidentiality is a major issue. Instead of services trusting clients directly, services trust tokens issued by a trusted STS. The client then has to securely authenticate to this STS.

Although many Web service security profiles are available, the decision of which one is best to be deployed for a particular application and settings is complex as different Web service security profiles facilitate the achievement of different requirements. Therefore, we postulate that following a structured decision-making process would aid developers in this regards and make their decisions more informed. The decision-making approach used in this paper is Analytical Hierarchy process (AHP) and we discuss it in the next section.

3. MULTI-CRITERIA DECISION MAKING METHODS: AHP
Analytical Hierarchy Processing (AHP) is one of the most considered methods for multiple criteria decision making problems. The wide spread acceptance of AHP is due to its simplicity and ease of use, in addition to its highly recognized utility. Indeed, AHP helps in structuring decision makers’ thoughts and in organizing the problem dimensions. AHP can be defined through the following three characteristics: (1) **AHP is analytic.** It assists in analysing the decision problem logically and in establishing judgments based on decision makers’ intuition and feelings which can be validated, questioned and reviewed by others; (2) **AHP utilizes a hierarchy structure.** This property comes naturally with human tendency to decompose and reduce the complex problems into sub-problems to be tackled one by one; and (3) **AHP is methodical.** It defines a step-by-step process for decision making tasks.

Saaty and Vargas [21] describe the steps to implement the AHP method as follows: (a) **Hierarchy;** that is decomposing the problem into a hierarchy of goal, criteria, sub-criteria and alternatives; (b) **Pair-Wise Comparisons** where data can be collected from experts/decision-makers or through empirical experiments corresponding to the hierarchic structure. The comparisons are made for each criterion and then converted into quantitative measures; (c) **Comparison Matrix** where pair-wise comparisons of various criteria generated at step b are organized into a square matrix; (d) **Eigen Vector** where the principal Eigen value and the corresponding normalized right Eigen vector of the comparison matrix is calculated to give a relative importance for the various criteria being compared; (e) **Consistency Ratio** of the matrix of order n is evaluated; (f) **Ratings** where the rating of each alternative is multiplied by the weights of the sub-criteria and aggregated to get local ratings with respect to each criterion. The local ratings are then multiplied by the weights of the criteria and aggregated to get global ratings; and (g) **Integrating Group Judgments** is used if the judging process involves multiple experts or data are collected from multiple experiments, then a single consolidated judgment is calculated using a geometric mean to integrate the multiple measures.

4. THE DESIGN SCIENCE APPROACH FOR DEVELOPING THE FRAMEWORK

4.1 Design-Science Research (DSR)
The paradigm followed in this research, concerned with analytically developing an AHP framework for Web service
security profiles, is that of Design-Science Research (DSR). This research aims, by utilizing design-science research, at producing an artifact in the form of a framework aiming to help architects and developers in organizations choosing best-suited security profiles for software applications along with their Web services. The aim of this research is highly consistent with the general aim of DSR. This is because the research in information systems and computing is considered a DSR research, if the main aim is to change a current situation related to organizational or social systems into a more desirable one through the development of novel artifacts [6]. Hence, we argue that DSR is highly fitting in the context of this research.

Indeed, design-science research is primarily a problem solving paradigm [6] that seeks to create artifacts addressing the so-called wicked problems [11, 18]. In principle, design-science research attempts to successfully design, develop, and evaluate technology-oriented design artifacts characterized as novel, innovative, and purposeful. Portrayed as purposeful implies that these artifacts would potentially provide organizations and humans with recognizable utility since they should address unsolved problems [6], or provide better solutions and thus enhance existing practices [9]. Hence, these artifacts aim to provide additional improvements to real-world phenomena [8, 12, 19]. Therefore, while humans could change their life styles through the introduction of these novel artifacts, organizations might change the ways in which they do business so as to exploit the opportunities that emerged due to these artifacts.

In the context of this research, whilst an AHP framework for Web service security profiles is the main artifact, selecting the most fitting security profile for a certain software application and its Web services is the tackled wicked problem. In the context of design-science research, the term “wicked problems” can be described as unstructured decision-making activities and settings. This is because these types of decisions are normally “poorly formulated, confusing, and permeated with conflicting values of many decision makers or other stakeholders” [18].

Design artifacts are classified by March and Smith [11], and anchored by Hevner et al. [6], into constructs of vocabulary and symbols, models representing reality with appropriate levels of abstraction, methods in the form of algorithms and practices, and instantiations which are implemented systems and/or their prototypes developed as proof-of-concepts [1]. The developed framework in this paper represents a model artifact which includes constructs. While its application and use in the course of AHP represents a method artifact.

The scheme to construct design artifacts is still very broad. Two main and general processes are identified by [11] as build and evaluate. Whilst building design artifacts demonstrate feasibility, they are evaluated against criteria of value to a community of intended users to ensure utility, quality, and efficacy [6]. Importantly, design science research stresses the importance of iterations in producing the design artifacts and assumes that reality and knowledge emerge throughout the iterations effort [13, 24].

### 4.2 Developing an AHP Framework

Following the DSR paradigm, we aim to develop an AHP hierarchy or Framework for selecting Web service security profiles that are most appropriate for certain applications. To this end and by referring to [21], we recognized that four facets need to be defined: (a) goal; (b) alternatives; (c) criteria; and (d) sub criteria (if there is any). An AHP hierarchy indicates a relationship between elements of one level with those of the level immediately below. This relationship percolates down to the lowest levels of the hierarchy. Accordingly, every element is connected to every other one, at least in an indirect manner. In the hierarchic structure, at the root of the hierarchy is the goal or objective of the problem being studied and analyzed. The leaf nodes are the alternatives to be compared. In between these two levels are various criteria and sub-criteria. It is important to note that when comparing elements at each level, a decision-maker needs just to compare with respect to the contribution of the lower-level elements to the upper-level ones.

In the context of this research, defining the goal of the AHP framework was quite straightforward as it is directly mapped to the current research problem; i.e., choosing the best-suited security profile to be deployed for a particular application including its Web services. Having the AHP goal established, we moved a step further to define the decision alternatives. For this purpose, we deliberately selected seven security profiles that are deemed representatives. These security profiles are: UA, UDP, MCS, TSSL, SA, and SV which are discussed in section 2 of this paper.

However, the issue of defining the decision criteria and sub-criteria was more complex. Three iterations incorporating design, deployment, and evaluation courses of action were needed before the final artifact (i.e., AHP Framework) has been emerged which includes a comprehensive criteria and sub-criteria in this context (see Figure 1). In the following sub-section, we discuss these three iterations in detail.

![Figure 1. DSR Activities](image)

#### 4.2.1 DSR Iteration One: Library Research

In this iteration, we followed a library research in which a comprehensive literature review was undertaken. The main purpose of this library research was to understand the security elements or criteria that are deemed relevant in the domain of Web services. In fact, this iteration was not very challenging given the fact that there is almost a consensus in the relevant literature regarding Web service security measures. The identified security criteria based on which Web service security profiles need to be compared are as follows [5, 17]:

- **Authentication**: It refers to establishing identity which assures that access to data and applications is limited to those who have appropriate proof of identity. Authentication mechanisms are based on the idea of having a token in the possession of the entity that is authenticated and this token is either software or hardware based. When a Web Service starts, information about the authentication status must be carries in the Web Service communication. As in standard Web traffic, service provider
should authenticate service requesters before sending Web Services information. This mechanism is known as peer-to-peer authentication. Another important mechanism is known as message origin authentication. The idea here is that received messages are authenticated based on their origin. This is useful if messages are communicated through a chain of Web services.

- **Integrity**: It refers to the requirement of ensuring that transmitted information has not been changed or modified during transmission. Knowledge about tampering occurrence fulfills integrity requirements. Integrity can be achieved using digital signature, for example.

- **Confidentiality**: It refers to the requirement for exchanged data between two communicating parties not to be available to a third party that may try to pry into the communication. In order to achieve confidentiality, one approach is to use a private connection between the communicating parties, such as a dedicated line or a Virtual Private Network (VPN). However, the critical information of Web services are usually exchanged through entrusted networks, the Internet most likely, where the private connection is not achievable and another approach is used to meet the requirement of confidentiality, which is encryption.

- **Non-repudiation**: It means that the message originator cannot deny sending this message. Any doubt about the message sender throws confidentiality and integrity into question and results could be disastrous. Digital signature and Public Key Infrastructure (PKI) technologies are used to deliver non-repudiation.

- **Authorization**: It refers to granting privileges for users and deciding whether an entity is allowed to access particular resources and services or not. Just because a user is authenticated does not mean that they are always authorized. Authorization software allows administrators to manage a policy for access control to services by giving different privileges to different users and groups. Single sign-on technologies, such as SAML, are used in Web services for both authentication and authorization.

- **End-to-End Security**: It means that communicated data is signed and encrypted between partners communicating the data throughout the chain.

After establishing these criteria, we deploy and utilize them to compare different Web service security profiles. Retrospectively, we found out that although these criteria facilitate reducing the number of favored security profiles for a certain application, yet there is a number of security profiles that can be employed. Consider this example, a certain Web application is highly critical and requires fulfillment of all of the previously discussed security criteria. To choose the best-suited security profile, the developer need to compare the security profiles based on the defined security criteria. These criteria alone will lead the developer to find out that more than one security profile can be selected (e.g. UA, MCS, and STS, see [3]), although there are slight differences in the level of security they can achieve. Moreover, despite the fact that these security profiles can all achieve the defined security requirements, they may have substantial differences in terms of performance. Hence, we recognized that these criteria alone are not sufficient for developers so as to take informed decisions and that performance requirements need to be taken into consideration. This evaluation led us to start the iteration 2 of our DSR research.

### 4.2.2 DSR Iteration Two: Laboratory Experiments

To test the effect of individual security profiles on Web services performance, we have designed and implemented an echo scenario. We actually use a simple JAX-WS echo application, which consists of a Web service and a client. This scenario represents the peer-to-peer mode test; the client sends different size messages (from 1 Byte to 1MByte) and the Web service echoes (send back) the same message received.

NetBeans IDE 6.5 is used to develop the Web service and the client. The Web service is developed as a Web application and deployed on a Glass Fish 2.2 application server. We used a Java SE application to represent the client. The initial data sent from the client to the service are randomly generated before sending the message to avoid any caching. Our data are collected from a local machine; the Web service and the client are deployed on a Dell machine (Pentium D CPU 2.80 GHz / 3GB of RAM) Running Microsoft XP. We measure, using Java’s System.nanoTime(), the time spent in requesting and responding on the client side as round trip time (RTT). We run every test 1000 times and calculate the average, maximum, and standard deviation RTT for each case. The test is then repeated on 10 different occasions. We compare the results using the Round Trip Time Increment Percentage [4] (RTTIP) in order to evaluate the performance overhead for a specific security mechanism deployment:

\[
RTTIP = \frac{RTT_i - RTT_0}{RTT_0} \times 100\%,
\]

- **RTT_0** is the round trip time without applying any security mechanism deployment.
- **RTT_i** is the round trip time of the Web service with a specific security profile i deployment.

Our performance evaluation [2] has shown that security profiles that use transport level security are always faster than message level security profiles. In addition, message level security protocols have a scalability problem if large messages are exchanged, unlike SSL-based profiles. Within message level security profiles, username authentication based profiles perform better than mutual certificates security. However, the difference is insignificant when using very large size messages. Using digest passwords instead of encrypting the whole username token can slightly improve the performance. The performance penalty of using SAML is very small and depends primarily on the underlying security profile. Finally, the performance of STS security profiles is massively less than Non-STS profiles and should only be used when service and client are on different domains.

Having reached this point of the research, we recognized that security profiles need to be selected not only based on their fulfillment of security requirements, but also based on their performance measures. In other words, the AHP hierarchy at this point included two major criteria (security and performance) along with their sub-criteria. By utilizing the current AHP hierarchy we start analyzing different scenarios and cases related to Web applications and services in their natural settings. This step has highlighted an important criterion which has been overlooked, namely configuration requirements. In fact, while conducting these cases, it came into view that different security profiles normally call for different infrastructure to be in place.
Examples of these infrastructure and configuration requirements include certificates and security service tokens.

Having recognized the importance of configuration requirements to the decision of which security profile is most appropriate to be deployed in a certain situation; we start DSR iteration 3 to further enhance the AHP framework.

4.2.3 DSR Iteration Three: Infrastructure and Configuration Requirements

As highlighted in the previous subsection, each security profile requires configuring some options on the Web service host. These configuration requirements reflect the technological constrains and system preference of the Web service provider. The Web service client may need to be configured depending upon the security profile selected by the server side. For Web service security profiles, the regular service configuration requirements [23] are:

- **Certificates stores**: There are two types of certificates stores: Keystore and Truststore. Keystore is used in service and client sides to specify the certificates and private keys for both. Truststore is used in the server side to specify aliases that contain the certificates and trust root of clients and vice versa.
- **Security Token Service**: They are used by Web services to reference a Security Token Service. The security configuration for the client-side of this application is dependent upon the security mechanism selected for the Security Token Service, and not on the security mechanism selected for the application.
- **Users’ database**: They are used by security profiles that require username, and preferably passwords, for authentications.
- **Flexibility**: The configuration requirements vary in terms of flexibility between the different security profiles. Transport level security profiles are usually easier to configure than the message level ones. Furthermore, some profiles require configuring additional options, such as SAML callback handler on the client side. We use the term “configuration flexibility” in this paper to refer to how flexible it is to implement a certain security profile.

After these three iterations of the conducted design-science research, we have reached a point where, from our perspective, the developed framework, in its final version (see Figure 2), is comprehensive and useful. We, in the next sub-section, provide an empirical validation and evaluation of the framework through practical settings. We aim here to show how the framework can be applied by Web service developers to define which profile is best to be implemented. To this end, we have adopted an illustrative scenario from [15] concerning public Web service. The description of this scenario and the application of the developed AHP framework are discussed in the following subsection.

4.3 Empirical Application and Evaluation

The evaluation in this section is devoted to test the performance and the applicability of the proposed framework by testing it in practical settings. We aim here to show how the framework can be applied by Web service developers to define which profile is best to be implemented. To this end, we have adopted an illustrative scenario from [15] concerning public Web service. The description of this scenario and the application of the developed AHP framework are discussed in the following subsection.

4.3.1 Description

A distributor uses Web services to provide catalogue information to online sellers that provide online shopping services. Sellers access the Web service from their Web applications to display current items available from the distributor. The Web service provider has the following requirements: (a) Web services’ clients require direct access to the Web service; (b) sellers accessing the Web service must be authenticated; (c) data passed between the service and clients contains sensitive information, such as account information, that must be protected while in transit; (d) the Web service provider has a database for the sellers that are allowed to use this service; and (e) high performance is expected by the clients as they should get instant response when using the service.

4.3.2 Application of the developed AHP Framework

According to the previous description, the three defined criteria are compared against each other. Within each criterion, the sub-criteria are also pair-compared in terms of their importance to compose the pair-wise comparison matrix. To convert the comparisons into quantitative values, the fundamental scale of absolute numbers [20] has been employed, as shown in table 1:

<table>
<thead>
<tr>
<th>Option</th>
<th>Numerical value(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equal</td>
<td>1</td>
</tr>
<tr>
<td>Marginally strong</td>
<td>3</td>
</tr>
<tr>
<td>Strong</td>
<td>5</td>
</tr>
<tr>
<td>Very strong</td>
<td>7</td>
</tr>
<tr>
<td>Extremely strong</td>
<td>9</td>
</tr>
</tbody>
</table>
| Intermediate values to reflect fuzzy inputs| 2, 4, 6, 8         | Reciprocals

Table 1: A Nine-Point Scale for the pair-wise comparisons

<table>
<thead>
<tr>
<th>Option</th>
<th>Numerical value(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security</td>
<td>1</td>
</tr>
<tr>
<td>Configuration</td>
<td>3</td>
</tr>
<tr>
<td>Performance</td>
<td>1</td>
</tr>
<tr>
<td>Priorities</td>
<td>0.2583</td>
</tr>
</tbody>
</table>

Based on the developed framework, 18 comparison matrices are generated. One of these matrices is for the criteria with respect to the goal, as shown in table 2. The consistency ratio (CI/RI) for each matrix should be less than 10% to accept the judgment. Otherwise, the pair-wise comparisons are reconsidered.

<table>
<thead>
<tr>
<th>Option</th>
<th>Numerical value(s)</th>
<th>Priorities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security</td>
<td>1/3</td>
<td>1/5</td>
</tr>
<tr>
<td>Configuration</td>
<td>1/3</td>
<td>1</td>
</tr>
<tr>
<td>Performance</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

Table 2. Main Criteria with Respect to the Goal

Figure 2. AHP Framework for Web Service Security Profiles
The results are synthesized by multiplying each alternative local weight of each sub-criterion by the priority of its criterion. The next step is to calculate the global weights vector by multiplying the normalized matrix by the number of its elements. The resulting weights are added for each alternative to calculate its final priority as shown in Table 6.

Table 3. Sub-criteria with Respect to Performance

<table>
<thead>
<tr>
<th></th>
<th>AVR RRT</th>
<th>STDEV RRT</th>
<th>MAX RRT</th>
<th>Priorities</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVR RRT</td>
<td>1</td>
<td>5</td>
<td>7</td>
<td>0.7306</td>
</tr>
<tr>
<td>STDEV RRT</td>
<td>1/5</td>
<td>1</td>
<td>3</td>
<td>0.1884</td>
</tr>
<tr>
<td>MAX RRT</td>
<td>1/7</td>
<td>1/3</td>
<td>1</td>
<td>0.0810</td>
</tr>
<tr>
<td>Consistency Ratio = 5.59 %</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The remaining 14 pair-wise comparison matrices are for the seven security profiles and easier to configure. Microsoft Corporation [15] suggests using Username Token and Https in this scenario, which portrays the values of the pair-wise judgments with respect to flexibility.

Table 4. Alternatives with respect to Flexibility

<table>
<thead>
<tr>
<th></th>
<th>UA</th>
<th>UDP</th>
<th>MCS</th>
<th>SSL</th>
<th>SA</th>
<th>SV</th>
<th>STS</th>
<th>Priorities</th>
</tr>
</thead>
<tbody>
<tr>
<td>UA</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1/2</td>
<td>2</td>
<td>2</td>
<td>6</td>
<td>0.1821</td>
</tr>
<tr>
<td>UDP</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1/2</td>
<td>2</td>
<td>2</td>
<td>6</td>
<td>0.1821</td>
</tr>
<tr>
<td>MCS</td>
<td>1/2</td>
<td>1/2</td>
<td>1</td>
<td>1/3</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>0.1023</td>
</tr>
<tr>
<td>SSL</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>7</td>
<td>0.3019</td>
</tr>
<tr>
<td>SA</td>
<td>1/2</td>
<td>1/2</td>
<td>1</td>
<td>1/3</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>0.1023</td>
</tr>
<tr>
<td>SV</td>
<td>1/2</td>
<td>1/2</td>
<td>1</td>
<td>1/3</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>0.1023</td>
</tr>
<tr>
<td>STS</td>
<td>1/6</td>
<td>1/6</td>
<td>1/5</td>
<td>1/7</td>
<td>1/5</td>
<td>1/5</td>
<td>1</td>
<td>0.0271</td>
</tr>
<tr>
<td>Consistency Ratio = 1.12 %</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

On the other hand, the performance of different security profiles can be measured based on the criteria discussed in DSR iteration 2. Thus, the pair-wise comparisons here are done based on actual data rather than judgments as illustrated in Table 5.

Table 5. Alternatives with respect to Average RTT

<table>
<thead>
<tr>
<th></th>
<th>UA</th>
<th>UDP</th>
<th>MCS</th>
<th>SSL</th>
<th>SA</th>
<th>SV</th>
<th>STS</th>
<th>Priorities</th>
</tr>
</thead>
<tbody>
<tr>
<td>UA</td>
<td>1</td>
<td>0.99</td>
<td>1.18</td>
<td>0.76</td>
<td>0.78</td>
<td>1.21</td>
<td>6.8</td>
<td>0.1554</td>
</tr>
<tr>
<td>UDP</td>
<td>1.01</td>
<td>1</td>
<td>1.19</td>
<td>0.76</td>
<td>0.79</td>
<td>1.22</td>
<td>6.85</td>
<td>0.1567</td>
</tr>
<tr>
<td>MCS</td>
<td>0.85</td>
<td>0.84</td>
<td>1</td>
<td>0.64</td>
<td>0.66</td>
<td>1.02</td>
<td>5.75</td>
<td>0.1313</td>
</tr>
<tr>
<td>SSL</td>
<td>1.32</td>
<td>1.31</td>
<td>1.57</td>
<td>1</td>
<td>1.03</td>
<td>1.6</td>
<td>9</td>
<td>0.2057</td>
</tr>
<tr>
<td>SA</td>
<td>1.28</td>
<td>1.27</td>
<td>1.52</td>
<td>0.97</td>
<td>1</td>
<td>1.55</td>
<td>8.72</td>
<td>0.1993</td>
</tr>
<tr>
<td>SV</td>
<td>0.83</td>
<td>0.82</td>
<td>0.98</td>
<td>0.63</td>
<td>0.65</td>
<td>1</td>
<td>5.64</td>
<td>0.1289</td>
</tr>
<tr>
<td>STS</td>
<td>0.15</td>
<td>0.15</td>
<td>0.17</td>
<td>0.11</td>
<td>0.11</td>
<td>0.18</td>
<td>1</td>
<td>0.0229</td>
</tr>
<tr>
<td>Consistency Ratio = 0.00 %</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Finally, the results can also be presented in the ideal form by dividing each priority by the maximum priority to make this first ranked alternative an ideal one with others getting their proportionate value [20] as demonstrated in table 7.

Table 7. Final Results as Normalized and Idealized Priorities

<table>
<thead>
<tr>
<th>Security Profile</th>
<th>Normalized Priorities</th>
<th>Idealized Priorities</th>
<th>Ranking</th>
</tr>
</thead>
<tbody>
<tr>
<td>UA</td>
<td>0.1517</td>
<td>0.7959</td>
<td>4</td>
</tr>
<tr>
<td>UDP</td>
<td>0.1523</td>
<td>0.7991</td>
<td>3</td>
</tr>
<tr>
<td>MCS</td>
<td>0.1378</td>
<td>0.7230</td>
<td>5</td>
</tr>
<tr>
<td>SSL</td>
<td>0.1906</td>
<td>1.0000</td>
<td>1</td>
</tr>
<tr>
<td>SA</td>
<td>0.1669</td>
<td>0.8757</td>
<td>2</td>
</tr>
<tr>
<td>SV</td>
<td>0.1372</td>
<td>0.7198</td>
<td>6</td>
</tr>
<tr>
<td>STS</td>
<td>0.0628</td>
<td>0.3295</td>
<td>7</td>
</tr>
</tbody>
</table>

The AHP framework suggests that transport security using SSL is the best-suited alternative to be deployed in order to secure the service in the given scenario. Although this profile is a point-to-point security mechanism, it is enough to satisfy the security requirements of the proposed application. Transport layer security mechanisms are normally faster than message layer security profiles and easier to configure. Microsoft Corporation [15] suggests using Username Token and Https in this scenario, which is an equivalent to the Transport Security using SSL provided by the Metro Web service Stack [23]. This also confirms the efficacy of the developed framework. In addition, our AHP framework provides an order for other possible alternatives. In this scenario, Because of the tendency towards a high performance, Transport layer based security profiles and peer-to-peer authentication and configuration (SSL and SA) are on the top of the ranking.

5. CONCLUSIONS

There is a considerable number of different security profiles that can be employed for Web applications and services. Selecting a profile that best matches the requirements of a certain application
in particular settings is very challenging. Aiming at helping security developers and architects in this context, we in this paper developed a novel and useful decision-making AHP framework and demonstrated how it can be applied empirically. The framework is developed following design science research which included three main iterations. Each iteration formed a loop which contains design, deploy, and evaluate as three main activities. The main contribution of this framework relies in its utility in guiding developers taking informed decisions in a systematic and manageable manner.

We plan to extend this work by including more security profiles into the framework and exploring other criteria and sub-criteria that can enhance the existing framework. The results can also be improved by involving more experts in the pair-wise judgment process.

6. REFERENCES


