

Session Authentication Protocol For Web Services

An analysis of Hada and Maruyama's proposed multi-party authentication mechanism for Web Service technologies

Kevin Berkowitz
CS 574 FA 2002
December 4, 2002

The fundamental concepts of data integrity, availability and confidentiality may be viewed as the three golden rules for computer security. Most if not every security policy or protocol strives to ensure that these three golden rules are met to guarantee the overall protection of a computing platform. One such computer security philosophy is the Web-Services Security Specification (WS-Security) proposed in a joint effort by Microsoft, and IBM [1]. The WS-Security white paper highlights potential security threats to the objectives of data integrity, availability, and confidentiality well as proposes means by which a Web Services platform may be secured. It is critical to note that while WS-Security proposes methods for controlling certain threats and protecting data , the WS-Security document does not provide any insight on how to perform session authentication between interacting Web-Services or multiple instances of the same Web-Service. Satoshi Hada and Hiroshi Maruyama (IBM Research, Tokyo Research Labs) note this shortcoming of current Web Service security technology and propose a Session Authentication Protocol specific to Web Services in their joint paper "Session Authentication Protocol for Web Services." [2]

Hada and Maruyama's research makes great strides towards providing a scalable and robust solution to the complex problem of authentication between a multitude of disparate Web-Services. The paper highlights several security threats that may arise without proper session authentication and provides practical and concrete examples on how to mitigate these threats via a session authentication protocol. The act of identifying threats to the existing WS-Security specification coupled with viable controls for these threats makes Hada and Maruyama's work a valuable contribution to the computer

security field and to Web Service security in particular. Many of the concepts provided by Maruyama and Hada are strong and offer an improvement to current Web Service Session Authentication techniques. However, the proposed Session Authentication Protocol does seem to have some weak points that need to be further explored and refined to produce a more robust solution. Indeed one may view the Hada/Maruyama protocol as most nascent research is viewed; an excellent stride in the right direction requiring further refinement and input to produce a consistently viable solution to a problem.

A key strength of the Hada/Maruyama paper is the identification of the need for a "multi-session authentication protocol" based on "existing protocols as the underlying means" of communication security. The IBM researchers argue that while there are several security protocols for securing Internet communications, such as S/MIME, IPsec, SSL/TLS, or Kerberos, none of these existing protocols are sufficient to provide session authentication for a Web Service. A Web Service requires authentication among multiple parties, which potentially may not have prior knowledge of one another. While existing mechanisms for securing Internet communications provide for authentication between a pair of objects (two-party communication model), the existing mechanisms fail to some degree when forced to scale to authentication between multiple parties. Kerberos is the closest to scaling to multiple party authentication via the *forwardable* bit in a Kerberos ticket which allows the ticket to be forwarded to a third party. However, Maruyama and Hada point out that even this flexibility is limited as a ticket and corresponding key must be "generated for each pair of communicating entities." The researchers propose that the existing authentication technologies be used as building blocks for a new technology that will allow for message authentication between an

unlimited number of parties. The IBM researchers continue to describe how such a multi-session authentication protocol can be created by use of their proposed "Session Management" and "Message Authentication" protocols. While the merits of the proposed protocols may need to be investigated further, the identification of a technical need as well as a presentation of a potential solution is a valuable contribution.

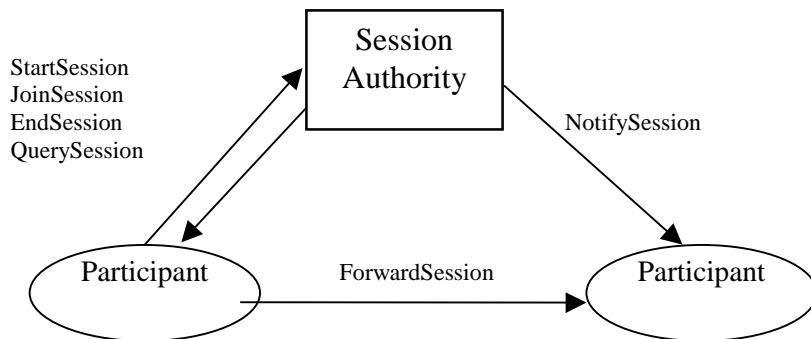
Another positive aspect of the Session Authentication Protocol for Web Services presented by Hada and Maruyama is the open-source like nature of the protocol. The proposed security techniques are not obscure, based on some mysterious black box component, or reliant on fundamentally difficult concepts. Rather, the proposed protocol is cleanly structured and utilizes a message format familiar to those working with Web Services. The proposed protocol relies on the openly adopted building blocks of PKI, SOAP, SOAP Digital Signatures (SOAP-DSIG) and XML-Encryption to create an extension to an existing message format. As such, the protocol can be easily examined and discussed. Knowledgeable peers may review the strengths and weaknesses of the protocol, hopefully yielding constructive criticism that eventually contributes to advances in the technology. The simple format of the proposed protocol will also help to ensure the protocol is correctly implemented. Web Service developers are less likely to make implementation errors with a simple, easy to understand procedure than a complex and intricate algorithm. The open source nature of the protocol will facilitate further analysis by subject matter experts as well as help to ensure correct implementation and should be viewed as a positive attribute of Maruyama and Hada's research.

Hada and Maruyama also prescribe a viable and secure mechanism for an instance of Web Service to send a data packet to a receiving instance of a Web Service. The

mechanism is based on a basic SOAP envelope and relies on two extension mechanisms proposed by the IBM researchers to add authentication features. The first mechanism is the addition of authentication information to the SOAP header and has been termed the "authentication header entry." The authentication header consists of "the session handle, a message identifier, the sending service's certificate and the receiving service's certificate." These authentication header elements are positive enhancements to the basic SOAP envelope. The session handle and message identifier help to avoid against Denial Of Service (DOS) attacks, or what the authors term replay attacks. The message identifier must be unique for the specific message, and hence a given message will be processed by the receiving service only after the uniqueness of the message identifier has been verified. The requirement for both the sending and receiving service entities certificates to be included in the authentication header helps to ensure that messaging is occurring between the expected parties, and that both parties trust one another for communication. The certificate requirement will protect against rogue Web Services that attempt to gather information from other Web Services. As an example, consider a Web Service that pretends to be credit consumer agency. The Web Service could attempt to communicate with a credit bureau (TRW, etc) and query the bureau for a consumer's personal data, such as credit history, contact information or social security number. With the addition of certificate authentication, the credit bureau will be able to verify that the agency requesting data is valid before responding to the request. All communication without expected and verified certificates will be ignored. This addition of an authentication mechanism will prove quite important as Web Service technology becomes more popular and a plethora of Web Services of varying emerge. The

certificate mechanism for authentication is robust, proven and widely used. Inclusion of this technique to aid in authentication is a strong point of Maruyama and Hada's protocol and would be a beneficial addition to the WS-Security mechanism which does not detail any specific authentication mechanism at the SOAP envelope level.

As mentioned in the introduction, the proposed Session Authentication Protocol does have some questionable components. One such gray area is the specification of the Session Authority that is at the heart of the proposed Session Management Protocol. The Session Authority is responsible for allowing participating Web Services to join a session and enforcing some rules for behavior in a session. Specifically, Maruyama and Hada state that the Session Authority is responsible for "assigning session ids, creating session secrets, maintaining the status information for each session" as well as keeping all session participants informed of the status. The figure below depicts Maruyama and Hada's notion of the Session Authority:



All these responsibilities seem like a reasonable task for a trusted piece of software and the concept of a Session Authority appears to be a viable idea. However, the specification for the Session Authority states that the Session Authority can either be a "separate entity or the initiator of the session filling a Session Authority role." While the notion of the Session Authority being a dedicated entity that is universally trusted is

plausible (trust must be placed somewhere), the notion that a Web Service itself could be a Session Authority is troublesome. Consider the case of a 'Trojan Web Service' that implements its own Session Authority. This Trojan service could present itself as an expected service and behave in an expected manner. However, the rogue service could also provide unwanted behavior such as allowing malicious participants to join a session, sharing a session secret among members not in the session or simply passing data learned in the session to a third party outside of the session. The allowance of a same party Session Authority appears to be an opening in the relatively strong protocol presented by Maruyama and Hada. It allows for a multitude of exploits to creep in to a key piece of the authentication protocol. Mandating that the Session Authority be implemented and maintained by some globally trusted entity, much as the Access Mediator is controlled by an Operating System Kernel, will help prevent against malicious attacks and would provide an authentication mechanism less prone to threats and misuse.

Another potential weak point in Hada and Maruyama's proposed Session Authentication protocol may be found in *ForwardSession* capability of the Session Management protocol. Using a *ForwardSession* message, a Web Service may directly invite another party to join the session. This task is accomplished by providing the recipient party the session secret and admission policy for the session. The *ForwardSession* approach will suffer from many of the same issues that plague private key implementations. Namely, it will be hard to keep a session secret truly secret, as any instance of a Web Service must be trusted not to expose the session secret to an unnecessary party. Much as an individual may expose their private key, the ability of session participants to share the secret with external parties can quickly lead to the

session secret being widely known. If the session secret is forwarded, via malicious intent or software bug, to a rogue Web Service, the rogue service will potentially have complete access to private information in the session. Maruyama and Hada address this issue briefly point stating that the participant may share the session secret with another party with the proviso that the forwarder "trusts that the forwarder will conform to the policy." However, this caveat is far from sufficient for creating a truly secure policy. One forwarder may trust the immediate forwarder (recipient), that recipient may in turn chose to trust another party, with the cycle of forwarding continuing in an inductive manner. It is quite possible that each forwarder will have a progressively lower burden for establishing trust and that the session secret for a dedicated session will quickly be shared among unnecessary parties. Eventually, the session secret may fall into the possession of a malicious process and data security could be compromised. It is odd that the Maruyama and Hada go to great lengths to establish a dedicated Session Authority to manage session and session policy and then allow a loophole such as the ForwardSession message. The overall security of the Session Management Protocol could be strengthened by the removal of the ForwardSession capability. Once again it is the case that Hada and Maruyama present an overall strong concept for a security protocol with some weak areas that need further refinement.

The Session Management protocol proposed by Hada and Maruyama also specifies that all session participants will share the same session secret for the duration of the session. The NotifySession message can be used to renew or change a session secret during an active session, but unfortunately Hada and Maruyama provide no schema for when such a change is necessary. Surely, it will be desirable to change the session secret

in a variety of circumstances. For example, consider a session in which only two services are required during initial phases, many services are required for a middle phase and solely two services remain in the last phase. It is unwise not to change the session secret once the majority of participants involved in the session have changed, as many uninvolved parties will have access to a session they should no longer have access to. Also, the longer a session secret is active, the greater the possibility is may fall into the wrong hands (Possibly through the ForwardSession message discussed above). If the session secret persists for the duration of the session and a session is significantly long, a malicious service may come upon the session secret and gain access to session data. Some sort of time based mechanism for creating a new session secret could help to avoid this issue. Admittedly it would be quite difficult for Maruyama and Hada to prescribe a solution for session secret cycling that fits the needs of every potential user group. However, some basic guidelines could provide a good deal more protection. One potential solution would be to institute the following simple policy: change the session secret each time a participant leaves a session or a specified time quanta has elapsed. This provision would improve on Maruyama and Hada's basic notion and hopefully provide more robust session security.

The multi-party session oriented authentication protocol proposed by IBM's Maruyama and Hada makes great strides to improving the authentication methods currently used in Web Service applications. The researchers' description of Session Authentication and Message Authentication protocols accomplish much of their goal of filling a security gap in an existing technology. The protocols have many strong attributes and overall appear to be reliable, scalable and viable solutions to the problem of

authentication in an environment with multiple interacting transient parties. Surely, the open-source like nature of the protocols, the relative simplicity of prescribed mechanisms and the scalability of the approach are key strengths of Maruyama and Hada's work. While the proposed protocols are an overall improvement to existing two-party authentication methods used in Web Services, the prescribed multi-party session authentication method is not without flaw. The research is still young and would likely benefit from further scrutiny and refinement. In particular, the effects and pitfalls of a Session Authority being a part of a Web Service, the *ForwardSession* message in the Session Authentication protocol, and the persistence of the session secret should be investigated further. However, the questionable features of Maruyama and Hada's proposed security mechanism are not sufficient to negate the central notions of the proposed protocol. Rather these potentially flawed design decisions are part of an overall viable security mechanism. Much like any young technology, refinement of initial flaws will lead to a more robust system. In the case of Maruyama and Hada's multi-party session oriented authentication protocol, analysis and refinement of questionable features will hopefully lead to a robust authentication method to be used by future Web Service applications. It is quite conceivable that Maruyama and Hada's work will form the basis for a standardized mechanism for session authentication in Web Services and hence form the cornerstone of a valuable solution to an existing problem in the emerging Web Services field.

References - Primary

The following references are referenced directly by the proceeding paper, and quotes are cited from specific authors. These papers are listed in order of appearance with reference 2 being the central object of discussion for the paper.

- [1] IBM Corporation, Microsoft Corporation, "Security in a Web Services World: A Proposed Architecture and Roadmap", <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwssecur/html/securitywhitepaper.asp>
- [2] Satoshi Hada and Hiroshi Maruyama, IBM Research, Tokyo Research Laboratory, "Session Authentication Protocol for Web Services", IEEE Proceedings of the 2002 Symposium on Applications and the Intranet

References - Supplementary

The following references were used to research items and topics discussed in the primary references. These references are not directly cited in the proceeding paper but concepts from each reference are mentioned and discussed. These references are listed in alphabetic order.

- [1] IBM, "Web Services Flow Language (WSFL 1.0)", <http://www-4.ibm.com/software/solutions/webservices/pdf/WPS.pdf>
- [2] IETF, Kerberos, <http://www.ietf.org/html.charters/krb-wg-charter.html>
- [3] W3C, Simple Object Access Protocol (SOAP), <http://www.w3.org/TR/SOAP>
- [4] W3C Working Group, "XML Encryption Syntax and Processing", <http://www.w3.org/TR/xmlenc-core>
- [5] W3C, "SOAP Security Extensions: Digital Signature", <http://www.w3.org/TR/SOAP-dsig>