

Browser-based Identity Federation

An Introduction

Thomas Groß
thomasgross@acm.org

July 29, 2010

1 The Web Browser as Platform

Many people use web browsers each and every day. Whether for gathering information, reading news, consulting a map, or socializing with friends, the web browser is their trusted companion. The idea of such web-based services is simple: users request services through a web browser with basic capabilities, and then receive the resulting content transparently. In the verge of Web 2.0, the browser has become an application platform in its own right, a versatile operating system for manifold applications. The browser gains a part of its charm as application platform from its *pervasiveness*. Virtually any modern PC and many cell phones are Web-enabled out-of-the-box. With two billion PCs expected till 2014 according to Gartner [4] and nearly five billion cell phones in 2010 according to ITU [7], browser-based protocols may reach their share of the world.

By foregoing installation of special client software, web-based services can benefit from the pervasiveness of web-enabled devices and anticipate rapid adoption. In fact, their ascetic self-constraint to a standard web browser as a client, redounds to their great advantage: they are *zero-footprint*. As these services can reach out to virtually any web-enabled device, organizations can count on cost-efficient deployment and maintenance as well as a low entry barrier for large user populations on the Internet.

2 Identity Federation

Many web-based services require user authentication, be it for personalizing user experience or authorizing access. As Web 2.0 promotes the seamless amalgamation of content from multiple sources, web-based services benefit from authentication by trusted third parties or across trust domains. This very authentication across trust domains is the guiding principle of *identity federation*. This new field aims at establishing mutually authenticated secure channels between users and services providers. Identity federation bootstraps the initial authentication between users and identity suppliers which are essentially trusted third parties. Clearly benefiting from the reach of browser-based protocols, identity federation standards readily embraced the zero-footprint paradigm. The resulting special protocol class spawned the key question: How secure is *browser-based identity federation*?

Definition 1 (Problem Statement).

1. *Analyse the security of browser-based identity federation.*
2. *Create the formal foundations for rigorous security proofs.*
3. *Prove the security of a major browser-based identity federation standard.*

3 Three-Party Authentication

In three-party authentication, two protocol agents mutually authenticate each other with help of a trusted third party. Even though three-party authentication and channel establishment are mature research fields, browser-based identity federation was an uncharted area at the beginning of this work. Earlier results from cryptography and formal methods do not directly apply to identity federation for two reasons: (i) Instead of a protocol agent, identity federation employs a standard browser as a client, limited to few cryptographic capabilities. (ii) Instead of concise protocol definitions, identity federation consists of highly extensible mark-up languages depending on their metadata. Let us expand these thoughts.

Web browser as client. As many identity federation standards indeed arrange for a zero-footprint client, we first explore the protocols' most important distinctive feature: the use of a standard web browser. Browser-based identity federation does not establish secure channels in the prevalent approach, *i.e.*, to exchange a session key and then secure the communication through it. Standard browsers encapsulate the key establishment of their secure channel primitives, notably Transport Layer Security (TLS), and prohibit the use of derived keys in high-level security protocols. This strict modularization is an important robustness principle of web browsers. Also, zero-footprint protocols assume that the browser does not own a client certificate and are therefore restricted to unilaterally authenticated secure channels.

Identity federation overcomes this limitation by establishing a server-authenticated secure channel first, and secondly sending additional information through this channel to vouch for mutual authentication. We depict a simplified example of such a protocol run in Figure 1. Indeed, establishing mutual authentication with a standard browser devoid of client certificates involves further subtleties. Let us branch to the browser's properties as a principal

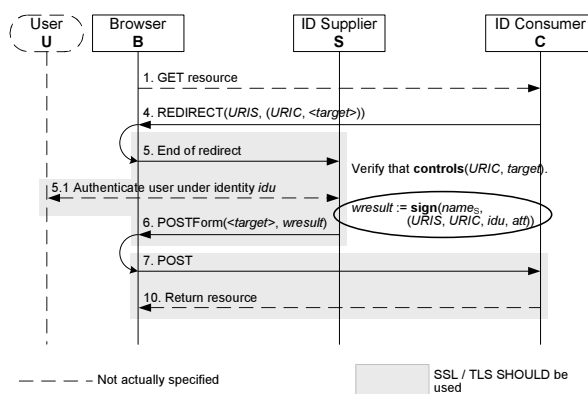


Figure 1: Simplified overview of a federation protocol. It uses a credential $wresult$ signed by ID Supplier S and server-authenticated secure channels to establish a secure channel between the identified user's Browser B and ID Consumer C.

of such protocols: (i) general behavior, (ii) security mechanisms, and (iii) hazards.

First, in contrast to the cryptographic protocol agents in three-party authentication, a browser is passive, as it is not aware of security protocols executed through it. Browsers clearly have no initiative as it would be expected from protocol agents, *i.e.*, they only execute tasks triggered by other principals, be it users or servers.

Second, standard web browsers only command few security capabilities: they understand the Hypertext Transfer Protocol, the Hypertext Markup Language, and the JavaScript language. They establish secure channels with Secure Socket Layer and Transport Layer Security. But, browsers cannot compute cryptographic functions on their own. In conclusion, browsers only contribute a limited security portfolio to identity federation.

Third, browsers can jeopardize a protocol's security. Let us highlight three conceptual hazards: First, browsers can produce involuntary information flow to disk or other servers, which may undermine a security protocol executed through it. Second, browsers cannot distinguish between trusted protocol principals and adversarial servers,

and can therefore fall prey to executing adversarial tasks. Whereas a protocol-specific agent can hold a trust root tailored to the task at hand, a standard browser trusts many root certification authorities with a variety of quality standards. Third, as browsers are restricted to a few rudimentary security policies, such as the same origin of code or the validation of certificates, adversaries can escalate privileges by exploiting policy loopholes. In addition, these hazards are more often than not beyond the user's control.

Let us conclude this branch. As even a correct browser can potentially leak information and jeopardize the protocol's security, the browser behavior tremendously impacts the security analysis of browser-based identity federation. Thus, we handle the browser as a semi-honest principal with two explicit imperfections: (i) The browser behaves according to its own rules, independently of the protocol specification; it is not privy to security protocols executed through it. It is a *protocol-unaware agent*. (ii) The browser leaks information that is confidential to the protocol. With these browser imperfections in mind, we mention a second challenge for the analysis of browser-based identity federation: the actual standards.

Nature of identity federation standards. Identity federation standards are an El Dorado of markup language design: highly extensible and multi-purpose. In particular, they follow a multi-part design principle, *i.e.*, they consist of multiple abstract interfaces with exchangeable realizations. Most standards build upon a core specification, which introduces basic language primitives, and then distinguish between *bindings*, which link to transport protocols, and *profiles*, which specify message exchange patterns. Contrary to cryptographic protocols, identity federation standards do not define precise message flows mathematically. Instead, they determine a set of constraints that limit the protocol space and allow multiple protocol realizations. In particular, these constraints often de-

clare relations and restrictions regarding a protocol's meta-data¹, and to an extent that the protocol's security heavily relies on the meta-data's consistency. These factors increase the complexity of the security protocols, and render earlier analysis and proof techniques inapplicable.

Consequences. We have seen that browser-based identity federation combines a standard web browser with constraint-based multi-part standards. We face a browser which is not aware of the security protocol. We face sets of fine-grained meta-data with subtle relationships, which can break a protocol's security in their own right. Instead of precise protocol definitions, we obtain entire protocol classes. As the subtle interplay of these factors potentially introduces vulnerabilities, browser-based identity federation is a challenging area that requires a new approach for security proofs. For this purpose, we aim at a formal model for rigorous browser-based security analysis that captures the subtleties of browser behavior and identity federation as well as supports constraints and meta-data.

4 Analysis of Identity Federation

As a first step, we have analyzed the Security Assertion Markup Language (SAML) [9] and WS-Federation [8] for security vulnerabilities and found many on different levels. Parallel to this work, multiple research efforts, *e.g.* [2, 3, 5], have analyzed related protocols for vulnerabilities. Most publications, apart from [3], abstract away the actual properties of web browsers and therefore treat the browser similar to a normal protocol agent. Undoubtedly, those threads of work contributed to identity federation analysis. However, we want to analyze the browser-based case from

¹The protocol meta-data include, for instance, identifier strings, Uniform Resource Identifiers (URIs), audience of transactions, and relationships to other principals.

a different perspective and to advance the field in two respects.

First, we improve the analysis precision by employing a detailed formal browser model. Our security analysis of SAML motivated this necessity, because it revealed subtle but severe information flow attacks introduced by a correct browser.

Second, most security research on identity federation pits the protocol’s strength against the analytic capabilities of the researchers or their tools. Thus, those approaches mostly derive vulnerabilities and offer improvements—as we did in the case of SAML. They advance the field through a protocol evolution, with their analysis capabilities as selection pressure. We complement this evolution with security proofs, *i.e.*, with positive statements that vouch for combinations of protocol constraints and fundamental assumptions. Our approach not only proves a safe area, but also provides guidance against new vulnerability vectors: those need to either overcome the fundamental assumptions or change the protocol constraints.

5 Goal: Channel Authenticity

We introduce *channel authenticity* as primary security goal:

Definition 2 (Channel Authenticity).

If a service provider accepts an identity federation protocol run at a secure channel with some entity, then this entity indeed holds the identity certified by the identity supplier.

We complement authenticity with the secondary goal of *freshness*:

Definition 3 (Freshness).

If a service provider accepts an identity federation protocol run, then the identification of the user has taken place in this very protocol run.

Both goals provide stronger security guarantees than the *entity authentication* prevalent in identity federation standards, as they (i) bind the user’s

identity directly to the secure channel and (ii) enforce a overall binding of the identification to the very identity federation run. We therefore establish a *mutually authenticated channel* between the user’s browser and the service provider.

6 Rigorous Security Proof

We aimed at obtaining a positive security statement for browser-based identity federation, and focus on formally capturing the behavior of a real web browser. We can achieve this in three steps:

First, we build on a strong formal machine model with cryptographic soundness. For that, we enhance the Reactive Simulatability (RSIM) framework [1], contributing a UML-flavored graphical specification language. We were the first to lift the state transition specification of its I/O automata to a faithful graph representation, and thus obtain an effective tool for information flow analysis.

Second, we were the first to model a real web browser in such a formal machine model. To do so, we specify generic machines for web browser, user behavior, and server-authenticated secure channels. We derive their behavior from the HTTP specification and explicit assumptions about user and secure channel properties. These machines constitute reusable building blocks for rigorous security proofs of browser-based protocols.

Third, we prove a browser-based identity federation protocol secure in this model. For this purpose, we co-designed a secure instantiation of the WS-Federation Passive Requestor (WSFPI) [6], which became the basis of major interoperability tests between identity federation vendors. We depicted this protocol in Figure 1. By modelling this protocol with protocol-specific machines and complementing them with our generic browser model machines, we obtain a formal proof system for the WSPFI protocol. We depict a simplified overview over such a proof system in Figure 2. We derive rigorous security proofs for WSPFI by static infor-

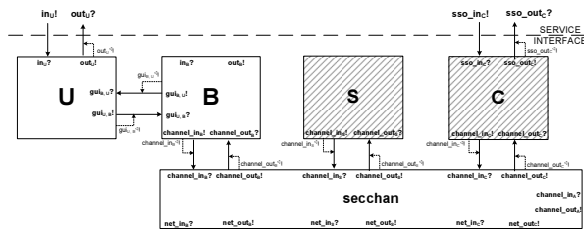


Figure 2: Simplified overview of a federation proof system. We complement the protocol-specific machines for ID Supplier S and ID Consumer C with generic machines for Browser B, User U and Channel Abstraction secchan from our framework.

mation flow analysis and back-tracking.

Theorem 1 (Security of WSFPI). *Assuming a secure signature scheme, a proper certificate setup for the identity supplier and cross-protocol key separation², a WSFPI protocol run interacting with the formal browser model establishes a secure channel between the identified user’s browser and the service provider which fulfills channel authenticity.*

Our security proof for WSFPI was the first security proof for a browser-based identity federation protocol with a formal browser model.

7 Impact on Practice

It was the declared goal of our work to impact the real world, that is, to make actual identity federation standards secure.

To that end, we collaborated with multiple standard bodies, namely OASIS for SAML and WS-Federation as well as the Web Services Interoperability (WS-I) consortium for WSFPI. In the case of SAML, our security analysis was officially acknowledged and lead to significant security improvements in SAML 2.0 [9]. In the case of WS-Federation, we had the opportunity to co-design

²Assumption that signing and long-term channel establishment keys are not reused in other contexts.

the interoperability profile, which we have subsequently proven secure. By that, two of the most important standards employed in industry benefited from this work.

We were also interested in the secure use of identity federation in industry systems, because a secure federation protocol is not sufficient to obtain a secure federation system. To that end, we collaborated with IBM Software Group to develop a product for secure identity federation. Whereas our contributions to the architecture and research prototype of IBM’s *Tivoli Federated Identity Manager (TFIM)* served as reality check and validation of our protocol work, we also contributed a robustness test suite for TFIM derived from the actual provably secure WS-Federation constraint set. By that, we promoted that the production protocol module fulfills the constraints sufficient for channel authenticity.

References

- [1] M. Backes, B. Pfitzmann, and M. Waidner. The reactive simulatability (RSIM) framework for asynchronous systems. *Cryptology ePrint Archive Report 2004/082*, IACR, 2004. <http://eprint.iacr.org/>.
- [2] K. Bhargavan, R. Corin, C. Fournet, and A. Gordon. Secure sessions for web services. In *ACM Workshop on Secure Web Services (SWS)*. ACM Press, 2004.
- [3] S. Gajek, J. Schwenk, M. Steiner, and C. Xuan. Risks of the cardspace protocol. In *Proc. 12th International Conference on Information Security (ISC)*, volume 5735 of *LNCS*, pages 278–293. Springer, 2009.
- [4] Gartner. Gartner says more than 1 billion PCs in use worldwide and headed to 2 billion units by 2014, 2008. <http://tinyurl.com/Gartner2008>.
- [5] S. M. Hansen, J. Skriver, and H. R. Nielson. Using static analysis to validate the SAML single sign-on protocol. In *Proc. 2005 Workshop on Issues in the Theory of Security (WITS '05)*, pages 27–40, New York, NY, USA, 2005. ACM Press.
- [6] M. Hur, R. D. Johnson, A. Medvinsky, Y. Rouskov, J. Spellman, S. Weeden, and A. Nadalin. *Passive Requestor Federation Interop Scenario*, Version 0.4, Feb. 2004.
- [7] ITU. Global mobile cellular subscriptions, total, 2010. <http://tinyurl.com/ITU2010-cell>.
- [8] C. Kaler and A. N. (ed.). *WS-Federation: Passive Requestor Profile*, Version 1.0, July 2003. BEA and IBM and Microsoft and RSA Security and VeriSign, <http://www-106.ibm.com/developerworks/library/ws-fedpass/>.
- [9] OASIS Standard. *Security Assertion Markup Language (SAML) V2.0*, Mar. 2005.