Abstract

It is often useful to be able to use distributed identity—an identity from one organization for accessing resources from another. For example, it would be useful to use an identifier corresponding to your work identity when accessing business partners’ websites. Similarly, collaborative projects can benefit from distributed identity. This memo proposes a scheme for distributed identity that meets requirements to minimize the risk of phishing attacks for HTTP-based applications including websites.
Table of Contents

1. Introduction .................................................. 3
2. Terminology ................................................... 4
3. User Experience ............................................... 5
   3.1. Roaming and Local State ............................... 5
   3.2. Webdav and other HTTP Applications ................. 6
4. Website Author Experience ................................. 7
5. A Proposed Solution ......................................... 8
   5.1. Negotiate Authentication .............................. 8
   5.2. Kerberos ............................................... 9
   5.3. Security Assertion Markup Language ................ 11
   5.4. Local Browser UI .................................... 11
6. Security Considerations ................................. 13
7. References .................................................... 14
   7.1. Normative References ................................ 14
   7.2. Informative References ............................. 14
Author’s Address .............................................. 16
Intellectual Property and Copyright Statements ............. 17
1. Introduction

It is often useful to be able to use distributed identity--an identity from one organization for accessing resources from another. For example it would be useful to use an identifier corresponding to your work identity when accessing business partners' websites. Similarly collaborative projects can benefit from distributed identity. This memo proposes a scheme for distributed identity that meets requirements to minimize the risk of phishing attacks [PHISHING].

Distributed identity solutions will never achieve the goal of having a single identity used for all websites. Some websites will only accept a limited number of identity providers. For example financial sites typically want high degrees of assurance that identity providers will not allow the wrong user to use an identity that has access to a financial account. Similarly users will not always wish to link their identities together. However distributed identity can minimize the number of identities that are in use.

This proposal uses existing technology with incremental improvements to achieve the goal of distributed identity. This document is organized into two parts. The first part (sections 3 and 4) describes the intended user experience for the distributed identity system. The second part describes a specific proposal for realizing distributed identity.
2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

The terms identity, subject, relying party, and claim are used as defined in [LAWS].
3. User Experience

This section describes what Alice, a HTTP user is hoping to get from distributed identity.

Alice wants to go to a website. She’s remembered the name of the website from an ad, received a link in email, or is following a link from another site.

The website wishes Alice to log in. It presents an HTML page which includes a login button. Alice pushes this button and her local identity management UI appears. This UI includes some branding from the website, but it also includes branding Alice chose when she installed her computer.

Alice selects an identity she wants to present to the website. Since she has not recently used this identity either on the web or anywhere else, she must enter the password associated with the identity.

Alice is logged into the website. If the website uses TLS, she knows that the page she sees was created by the same party to whom she authenticated. If her identity is designed to be accepted only by a small number of relying parties, then she knows that she is talking to one of these parties even if she was not careful to check to make sure the URL was not spoofed.

If Alice didn’t have an existing identity she wanted to use and if the website she’s going to is not somehow restricted to existing users, then her identity management UI will provide the option to create a new identity.

The primary benefit from distributed identity is that if the website allows, Alice can choose whatever identity she wishes to use. For example if a business partner has granted her access to some collaboration site, she can use her work identity even if the site is in a completely different organization. Alice can choose to use the same personal identity for many of the sites she shops at, or she can choose to use different identities in order to minimize sites’ ability to correlate what she is doing. She can even choose to use multiple identities within a single site if she wants that level of privacy.

3.1. Roaming and Local State

Alice’s computer may store information on which identity is typically used with which website. However it is important that Alice be able to use her identities from any computer she finds herself at, without needing to bring special hardware or to configure the computer with
her identities. Alice should be able to use an identity by knowing the name of the identity, the identity provider and of course the password.

Special consideration needs to be given to how Alice knows that the local identity management UI is displayed on a computer that she is visiting. On her local computer she has branded the identity management UI.

3.2. Webdav and other HTTP Applications

Alice needs to get the same experience when she uses HTTP applications besides websites. Generally the identity management UI will be invoked when the server requires authentication instead of when she selects a login button, but besides that the experience will be the same.
4. Website Author Experience

This section describes how distributed identity works for Bob, an author of a website.

Bob wants to add support for distributed identity to his website. His server software does already support distributed identity.

Bob adds a new form control to his login page. He can choose one of three ways to select which identities users may use:

1. Any identity
2. Any identity from a set of identity providers specified by Bob
3. Any identity from a single identity provider that Bob specifies

Bob can also specify branding for his site to be included in the local identity management UI. In the interests of extensibility Bob can specify which distributed identity management mechanisms he supports.

In the interests of incremental deployment, Bob needs to be able to use Javascript or some other mechanism to provide a page that works both with traditional authentication and with distributed identity.

If Bob runs a webdav server or some other HTTP application, he sends the same information but it is carried in the HTTP response requesting authentication instead of in an HTML page. It is important that future extensions to distributed identity keep the parameters that can be specified in HTTP and HTML in sync.
5. A Proposed Solution

This section proposes a concrete way to achieve distributed identity. The goals of this proposal are as follows:

1. Demonstrate that distributed identity can be achieved with incremental improvements to existing protocols and technology.

2. Establish a minimum level of detail that needs to be specified to propose a solution and show that it is workable.

3. Propose a solution the author believes is the best compromise between deployability and meeting the requirements.

The solution is comprised of four related components. HTTP negotiate authentication is used to carry security data from the browser or client application to the server and to carry security response information back. Kerberos is used as a way to separate the interactions with the identity provider from the interactions with the HTTP server. In effect, Kerberos provides the distributed identity. Security Assertion Markup Language (SAML) is used to carry assertions (claims) about the identity. The browser or client UI is used to let the user know in a trusted manner that their password is being given to an identity manager on their local system rather than sent in the clear to the remote server.

5.1. Negotiate Authentication

HTTP negotiate authentication has been very successful at providing enterprise websites with a limited form of distributed identity. Enterprise users can log into their operating system and then are authenticated to websites with no additional work. While this protocol was originally introduced for Microsoft it has been adopted by several major browsers, Webdav libraries, web servers and operating systems. Negotiate authentication has enjoyed wide deployment in enterprises--possibly greater than digest authentication. The negotiate authentication method typically provides a better user experience than the digest method because except in error cases, no dialogue or user interaction is required. One significant problem with basic and digest HTTP authentication is that at least today, they provide a interface very different from the rest of the website by popping up a local dialogue; options for forgotten passwords, new accounts or help are not typically available.

In the optimal case, the negotiate method will not add any round trips to the HTTP exchange. If the client knows that negotiate
authentication is needed (for example via an HTML extension like the one contemplated in Section 4), then the client can include the first negotiate message in the with its HTTP request. For the case of Kerberos [RFC4121], only one round trip is required so no round trips are added. Additional round trips may be required if other mechanisms are used.

While negotiate exists today, incremental improvements are required to get an ideal user experience. The following improvements would be needed:

1. Create a standards track version of the negotiate protocol. It is likely necessary to change the name used in the HTTP authentication request in order to gain change control and to add extensibility. Work is already under way to do this as an individual submission.

2. Currently negotiate assumes that servers maintain context if more than one round trip is required. This is unacceptable for some HTTP deployments. Add support for the client to return an opaque context to the server for multi-round-trip negotiation.

3. Currently there is not a well defined way in HTTP to authenticate before sending a request. In the case of PUT requests or other requests with confidential data it would be desirable to authenticate first. Add support for this.

4. Provide some mechanism for carrying hints about the identity that is used. As discussed in Section 5.2, servers may have many identities. Some GSS-API implementations will be much easier to use if the server’s identity is made available to the server as part of the HTTP request.

5.2. Kerberos

Kerberos provides separation between the identity provider and the relying party. In effect, Kerberos allows the identity to be distributed. Kerberos deployments have scaled to sizes comparable to those anticipated for Internet identity providers.

Kerberos has two exchanges that are relevant to distributed identity. The first exchange is the authentication server (AS) exchange [RFC4120] which is an exchange between the identity provider and client. This exchange has a mode described in RFC 4120 that is a traditional challenge/response authentication system [IABAUTH]. Authentication based on public key certificates [RFC4556] can also be performed using the AS exchange. Modes of the AS exchange for token cards and for zero-knowledge password protocols have been implemented.
but not standardized.

The second exchange is the ap-req exchange used to authenticate to the relying party. This exchange is carried in HTTP authentication using the negotiate mechanism.

A feature of Kerberos called authorization data can be used to carry Security Assertion Markup Language (SAML) assertions. Since the assertions are carried in the ticket, Kerberos’ existing mechanisms can be used to protect and authenticate the assertions when the Kerberos server (KDC) is the SAML Authority. This will be much easier to deploy than XML signatures. Of course XML signature can still be used for assertions made by parties other than the KDC.

Kerberos authenticates a client to a given server. Kerberos has mechanisms for cross-realm authentication where a client in one realm (Kerberos namespace) can authenticate to a server in another realm. While this mechanism can be used with distributed identity when the necessary cross-realm relationships exist, distributed identity cannot depend on being able to establish cross-realm relationships. A cross-realm relationship implies that the client realm believes that the KDC of the server realm is responsible for authentication to any server in that realm’s namespace. Unfortunately, there is no easy way to know which servers belong to a given realm’s namespace. So, it will generally be easier to deploy systems where servers have multiple Kerberos principals--one for each identity provider they accept. Since servers only accept identity providers when they have a relationship with someone using that identity provider, then this will not typically involve more than constant increase in state servers already maintain for each user. In several important situations including servers that accept a specified set of identity providers, significantly less state is required.

Kerberos is a good match for many of distributed identity’s needs. However incremental improvement is required in the following ways:

1. Provide a standardized mechanism for enrolling an identity in a Kerberos realm. This is needed so Alice’s local identity management UI can create identities.

2. Develop a mechanism for enrolling a Kerberos server into a realm based on a public-key credential or other non-Kerberos credential. This is needed so that servers can start accepting identities from a particular identity provider. In cases where the identity provider is willing to register any server and the server is willing to accept any identity provider, this operation needs to be something Alice’s browser can do automatically.
3. Kerberos in the enterprise is focused on single sign-on. Users should type their password or present a smart card at login and then should have network credentials cached. This is desirable from a usability standpoint. However security policies of some sites--particularly financial sites--require that the user has recently authenticated. Such sites will often time out a session after a few minutes to avoid problems where a user has left a computer unattended. Kerberos has an integrity-protected mechanism to report when the user actually authenticated to the KDC but no standardized mechanism to allow a server to request a new authentication be performed without using cached tickets.

4. An authorization data element needs to be defined to carry SAML assertions.

5. An HTTP transport for Kerberos should be defined. Currently, Kerberos runs over TCP and UDP. If Kerberos is going to be used for web authentication and HTTP transport is required so that Kerberos has the same firewall traversal properties as HTTP.

5.3. Security Assertion Markup Language

People often desire to use distributed identity systems to convey information such as name and age about a subject to the relying party. SAML is proposed as a mechanism to do this. In order to use SAML, a profile of SAML for this application needs to be created. Such a profile would need to specify:

1. What claims need to be supported
2. How third-party and KDC-issued claims are mixed
3. How clients request particular claims

An alternative that has been proposed is a SAML GSS-API mechanism that wraps the Kerberos mechanism. The problem with this approach is that only the KDC and server share the service key ticket. So, unless the SAML is inside the Kerberos ticket, then the client is responsible for binding the SAML assertions to the Kerberos exchange and proving that the assertions apply to the client. This would typically require XML digital signatures be used for all assertions—not just third-party claims. That would increase code complexity and deployment complexity.

5.4. Local Browser UI

The local browser UI is critical to the security of any web authentication system. The primary purpose of browser UI extensions
is to let a user know whether they are typing a password into a local system or sending the password over the network. The phishing requirements document [PHISHING] discusses UI requirements in more detail.

Specification of the UI is outside the scope of the IETF. However the IETF should be involved in discussing security requirements for the UI, such as the need to distinguish between passwords entered locally and passwords sent over the net. The IETF also needs to be involved in discussing abstract requirements for the inputs and output of the UI such as those discussed in Section 4.
6. Security Considerations

All the security considerations discussed in [PHISHING] apply to the specific proposal in this document.

The security of HTML extensions proposed in Section 4 needs to be carefully considered. In particular, any concrete set of extensions needs to be analyzed for possible vulnerabilities when an attacker can modify the HTML elements used to signal distributed identity.

As discussed in Section 5.2, Kerberos needs to be extended to support enrollment of new identities. The security of these extensions will be critical to the security of the entire system. Often, identity providers will allow new identities to be enrolled without authentication. This allows the identity provider to confirm that subsequent uses of the identity correspond to the same subject but not to confirm that the identity corresponds to some particular real-world subject. This is reasonable provided that relying parties do not place inappropriate trust in information claimed about the subject. In particular, relying parties need to make sure that they have sufficient confidence the identity corresponds to the intended subject before granting access to that identity. Similarly the relying party needs to confirm that it has sufficient confidence in policies and procedures used to run an identity provider.

Similar concerns exist for enrollment of servers into a Kerberos realm. A denial of service condition exists if servers are weakly authenticated before being allowed to join a realm and an attacker is able to prevent the real server from joining by joining a spoofed server. In addition, if users connect to a spoofed server expecting to be connecting to the real server, they may disclose confidential information to that server. However the server authentication provided by TLS will tend to mitigate this attack when distributed identity is used in conjunction with HTTP over TLS. Also, identity providers wishing to provide a high degree of trust should not weakly authenticate servers before allowing those servers to accept identities.
7. References

7.1. Normative References


[MSNEGOTIATE]
Jaganathan, K., Zhu, L., and J. Brezak, "Kerberos Based HTTP Authentication in Windows",

[PHISHING]


7.2. Informative References

draft-iab-auth-mech-05.txt (work in progress), February 2006.


[RFC4556] Zhu, L. and B. Tung, "Public Key Cryptography for Initial
Author’s Address

Sam Hartman
Massachusetts Institute of Technology

Email: hartmans-ietf@mit.edu
Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Copyright Statement

Copyright (C) The Internet Society (2006). This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.