WebISO
Internet2 Draft
Expires: April 22, 2003

R. Morgan
University of Washington
October 22, 2002

WebISO: Service model and component capabilities

draft-morgan-webiso-model-01

Status of this Memo

Abstract

This document describes a component and service model for systems providing organization-wide Web-based sign-on services, dubbed "WebISO systems", including authentication service, application-server components, and message formats. It defines a set of detailed capabilities for each component and for the overall system. Requirements for deployments at particular sites, and actual WebISO-like packages, can be compared to the model and to these capabilities.

1. Introduction

As computer-based systems and services proliferate, new systems almost universally provide user access via the Web. That is, systems provide information and operations using HTML-structured objects, using the HTTP network protocol; and people use standard web browsers to interact with them. While much of the Web is "open to all", many important academic and business functions provided by these systems require access control, to ensure that information and operations are accessible only to authorized users. While access control can be based on many factors, in many cases the first step is for the identity of the user operating the browser to be securely established by the web-based service (just as with traditional services); this is generally called "user authentication".

It is attractive for large organizations to provide common infrastructure for authentication across the organization. Benefits include lowering process costs, establishing a common reasonable level of security, and providing a common experience for users and system deployers. While a comprehensive authentication infrastructure should ideally support all mainstream applications and services, including the Web, in practice the combination of the Web's pervasiveness and its unique protocol characteristics have led many organizations, even those with existing authentication infrastructures based on standards like Kerberos, to deploy new infrastructure specifically to support authentication for Web applications. This document refers to this kind of infrastructure as "Web initial sign-on" or WebISO.

This document describes a common model for providing WebISO service, based on a number of existing systems. Based on this model, it defines a number of technical capabilities, broken out by component, that are desirable to have in a WebISO implementation. It is intended that implementors and deployers be able to use these capabilities as input to their process of establishing requirements for particular implementations and deployments.

---

2. Basic Model

This section presents a basic model for WebISO services. The model is intended to provide concepts and terms for use in defining system capabilities. The model is not intended to describe precisely any particular WebISO system.

2.1 System Scope and Design Goals

- Authentication: Authentication is the process of a client supplying to a server evidence of the client's security characteristics (in most cases its security identity), and the verification of this evidence by the server. Generally this is used to establish a security context in a server process that the server uses to make security decisions about further communications between the client and server.
- Sign-on: Sign-on is the process of a user establishing a session, to accomplish some task or set of tasks, by knowingly supplying to the system a security identity and some evidence (generally involving secret information such as a password) of that identity.
- User database: This is a database (or repository) containing basic security information about system users, including individuals and others (such as organizational units or processes). Security information includes identifiers, passwords, keys, group memberships, etc.
- Application:
- Web-based: A system is web-based if it uses the HTTP protocol and HTML-format messages for most or all of its communications.
- Organization: From a security point of view an organization is characterized by having many different business processes, hence many distinct applications supporting those processes, but having, at least potentially, a single user database containing security information about all users of organizational applications.

The basic functionality of a WebISO system, then, is to provide the ability for web-based applications across an organization to rely on a shared user database, and to provide user access to applications with minimal number of sign-ons, while meeting various other organizational security goals as described below.

The model assumes these basic design goals and constraints.

1. Work with standard unmodified web browsers.

   A key advantage of web-based applications is that a web browser comes pre-installed on almost every client system. A WebISO system is much more deployable if it can work with standard browsers and operating systems without requiring installation of WebISO-specific software. (This goal doesn't rule out having optional client-side software that provides improved security or ease of use. Consideration of these methods is out of scope of this document, however.)

2. Support organizational username/password authentication.

   Almost all deployed organizational authentication systems, even those based on more sophisticated technology such as Kerberos, support user authentication via username/password. A WebISO system is easy to integrate if it can use the organizational username/password store. Despite the support of X.509 client-certificate authentication, via SSL/TLS, in almost all web browsers and web servers, deployability and user-readiness problems lead most designers away from relying solely on this method for insititution-scale user authentication.

3. Reduce exposure of user passwords to application web servers.

   Web-based applications span a wide range of risk and administration scenarios. It is desirable for the authentication system to be usable by applications even if their system administration practices are not of the highest standard. Thus it is attractive for a WebISO system to work with username/password authentication without exposing user passwords to application servers, and to protect against other possible attacks based on compromised or malicious application servers.

4. Support a single sign-on experience for users.

   As authenticated web-based applications proliferate, it becomes onerous for users to re-enter username/password (or other authentication information) as they begin a session with each new application. A single sign-on scheme lets users create a sign-on session (typically by entering username/password once) such that entry to subsequent applications doesn't require re-entry of username/password.

5. Integrate easily with common web application systems.

   Integrating a web-based application with a WebISO system involves some effort, ranging from simple configuration to substantial programming. Obviously the easier this is, the better. Providing ease of integration may involve integrating the WebISO system with common web server packages (e.g. Apache, Microsoft IIS), or application frameworks such as Java servlets.

WebISO systems also try to meet the general goals of any infrastructure service: robustness, scalability, security, ease of use, ease of administration, integration with existing infrastructure, and reliance on standards.

2.2 Domain Model

This section describes the components of a WebISO system. The description also includes those related components, not part of the WebISO system itself, which form the domain, or environment, in which the system components function.

2.2.1 Weblogin service

The weblogin service is the component of the WebISO system responsible for handling authentication requests, interacting with users to gather authentication information, verifying that information, and issuing authentication responses. Its internal structure is covered in more detail in **a later section**. It is a stand-alone web-based service, independent of any other web-based applications. The weblogin service typically depends on some existing authentication service (or username/password repository) to provide verification of user-entered username/password data. The recipient of its authentication responses is the web application agent on the application web server.

2.2.2 Web application agent

The web application agent is the component of the system that provides integration of the WebISO service into a web-based application. It generates requests to the weblogin service, interprets authentication responses from the weblogin service, and provides user identity information to the web-based application. It may be implemented as a code library to be linked into application code; or as an extension to a web-application environment such as a web server (e.g. Apache); or in various other ways.

2.2.3 Web application

The web application is the "client" of the WebISO infrastructure, in that the WebISO system needs to meet its requirements for user identification, as input to its access control, authorization, auditing and other security functions. There is a huge variety of potential applications ranging from simple to complex.

The simplest "application" is providing access to static web content. Traditional applications in a university environment include university administrative functions such as accounting and HR, and learning-related functions such as course registration and course management, and research functions such as scientific and medical computing. Some web-based applications are "front ends" to other kinds of application services which are accessed by end-users using other protocols (e.g. email protocols such as IMAP). A comprehensive WebISO system should meet the requirements of all of these applications.

2.2.4 Verification service

In most cases the username/password verification in a WebISO system is provided by some existing authentication service, which also provides service to other, non-web-based, applications. Kerberos, LDAP-based directories, and NIS are common examples. In some cases a WebISO deployment may provide integration by being able to access more than one verification service. In some cases a WebISO system may use an authentication method that doesn't require username/password verification, in particular X.509 client certificates.

2.2.5 Web browser

End-users interact with web-based applications (hence with the web application agents) and with the weblogin service via web browsers. A WebISO system generally requires some standard browser features, in particular redirection, SSL/TLS, forms, and cookies. Javascript/ECMAscript is generally encouraged but not required. In some cases support of web (i.e., HTTP/HTML) user agents that are not classic browsers (e.g. limited browsers on PDAs, automated agents such as wget, WebDAV agents, etc, may be useful.

2.2.6 Organizational directory

An organizational directory contains shared information about system objects of interest to the organization, including users, applications, services, network nodes, etc; and it makes this information available to systems and users across the organization. A directory may be relied on by the other WebISO system components for various kinds of information: passwords, user attributes, system configurations, etc.

---

3. System components and interfaces

It is useful to decompose the WebISO system components into subcomponents, since many system features are based on capabilities of these subcomponents and interaction among them. This section describes typical subcomponents of the weblogin service and the web application agent.

3.1 Weblogin service

3.1.1 Browser interface

The weblogin service browser interface interacts with browsers on client systems via HTML/HTTP(s). It accepts authentication requests from web application servers, sent via the browser. It sends pages to the browser to prompt for authentication information, and accepts the responses. It generates redirection information to return the browser to the web application server. It parses input data received from the browser via cookies, URLs, form input, etc, and sends data out in these forms.

3.1.2 Authentication processing

This is the primary "business logic" of the weblogin service. Authentication requests received via the browser interface are handled by this component, including any context information such as modifiers included in the request from the web application agent, application-server-specific policies, single sign-on processing, session lengths and timeouts, etc. The output of authentication processing includes (depending on specific system designs) a response to be sent to the web application agent, login session state saved in the weblogin service, and login session state sent to the browser as a cookie.

3.1.3 Verification adapter

This interface provides access to the verification service or services, generally for verification of a username/password pair. The result of verification is usually simply success or failure. The verification adapter may include additional features such as checking with multiple verification services, return of security tokens (such as a Kerberos TGT), etc.

3.1.4 Session management

A single sign-on service involves some notion of a "sign-on session" that starts at the initial sign-on event, continues for some time during which single sign-on is active, and terminates at some point due to timeout or explicit action of the user or the system. The session management component of the weblogin service may involve state information maintained within the service itself, state stored in the browser via cookies, or both. Session information may, in some designs, be queried by web application agents.

3.1.5 Policy management

A weblogin service will typically have a number of different policies that can be managed by administrators, including authentication methods and interactions, session lengths, per-user or per-system policies, etc. Policies may be managed in many ways, including compile-time settings, configuration files, interactive management UIs, etc.

3.1.6 Administration/logging

logging of authentication events; turning system on/off; etc.

3.2 Web application agent

3.2.1 Web application interface

The web application agent is the means by which the web-based application gets access to the services of the WebISO system. A system that supports different application environments generally will have to provide application agents specific to each environment.

One way of providing agent services is to extend the capabilities of a standard web server such as Apache or Microsoft IIS. Standard web servers provide a number of services to applications, including making available information about requests, providing maangement methods for configuration and policy, process and thread services, etc. The services of the web application agent can be offered to applications in the same way as these existing server features, in some cases overriding existing authentication methods. This is attractive to applications that work with standard web servers.

Another method is to provide agent services as libraries that can be linked with application code. Generally this means writing the libraries in the same language as the application (Java, Perl, Python, C, etc).

In any case the application interface needs to provide access to common funcationality: invoking the authentication interaction including providing possible modifiers to the interaction, and conveying the results of the authentication interaction.

3.2.2 weblogin interface

formatting commands for weblogin, dealing with encryption/verification of messages, encoding optional params, etc.

---

4. System capabilities

This section describes various capabilities, or features, of WebISO systems, in terms of the system components that provide those features.

4.1 General

4.2 Weblogin service

4.2.1 Single sign-on

Single sign-on is a feature of most WebISO systems. With this feature, a user's initial sign-on action requires an explicit user authentication interaction, typically entering a username/password into a weblogin-provided web form; but authenticated access to subsequent webapps does not require such an interaction.

4.2.2 Single sign-on session end/logout

With this feature a user can interact with the weblogin service to end the user's single sign-on session. After this, sign-on using that browser will require another explicit user authentication interaction.

4.2.3 Multi-application logout

4.2.4 Multiple sign-on policies

With this feature a weblogin service can support distinct sign-on policies. This is often to distinguish different authentication levels in terms of strength, e.g. a sensitive webapp might require a special password or one-time token value to be entered.

4.2.5 User identifier mapping

4.2.6 Kiosk mode

This capability permits a weblogin server to identify specific end-user browsing conditions (e.g. User-Agents, IP addresses) and adjust interactions accordingly, e.g. disable, or greatly shorten, single sign-on sessions.

4.2.7 Per-session end-user preferences

The capability permits a weblogin server to customize interactions based on options selected by end-users, e.g. "I'm using a shared computer", enable/disable SSO, enabled/disable reauthentication warnings (aka "pass- or click-through pages").

4.2.8 Skip/Cancel option

This capability allows a user to decline the weblogin service's request to authenticate, with the appropriate return value communicated to the web application agent.

4.2.9 No-prompt option

With this capability an application can request the weblogin service to do authentication on the condition that the interaction requires no prompting of the end-user. That is, the weblogin service does not attempt to prompt the end-user for sign-on information; it only checks for a valid SSO session. This feature allows applications to offer end-users more than one sign-on choice (e.g. weblogin service, local ID, or enter as guest) if the WebISO system hasn't already authenticated them.

4.2.10 Co-branding

With this capability a weblogin service can tailor the login, logout, and click-through pages displayed to end-users with customized text and/or icons for each application.

4.3 Web application agent

References

Author's Address

RL "Bob" Morgan
University of Washington
4545 15th Ave NE
Seattle, WA 98105
US
Phone: +1 206 221 3307
EMail: rlmorgan@washington.edu
URI: http://staff.washington.edu/rlmorgan/

Appendix A. Acknowledgements

The author acknowledges contributions of many members of the Internet2 WebISO working group, in particular Larry Geenfield, Scott Fullerton, and Nathan Dors.