Quest for Web Single Sign-on at the University of Michigan

slides from a poster presentation @ Educause 2003

Abstract: Cosign is a Web single-sign-on system recently deployed at the University of Michigan. We will describe the requirements of such a system, the evaluation of other Web single-sign-on systems, and the collaborative development process used to design and implement Cosign.

The University of Michigan
Quest for Web Single Sign-on
Other Authentication Filters
Application Support
Collaborative Portals

First Generation

Since late 1995, the University of Michigan has used a cookie-based web authentication system. The requirements (as presented by Winkel/Kamm at WebDevShare '98) were:

Password Security

Passwords should never go over a wire unencrypted.

Content Security

Content such as grades, patient records, and HR information should not be distributed in clear text.

Simplicity

The user interface must be very simple, because of the wide variety of expertise among University of Michigan web users.

Entirely Web-based

To facilitate as little installation on the client machines as possible.

Cross-Platform

Must support Internet Explorer, Netscape Navigator, and preferably Lynx as a text interface.

Speed

Strengths

- Ran on Solaris, AIX, Linux, Windows NT, and Windows 2000.
- Supported Apache API, NSAPI (iPlanet), and ISAPI (IIS 4.0)
- Many campus web applications already written to work with it.
- Enforced variable length hard session time-outs (8 or 4 hours depending on the service).
- Cookie was sent only to the server that set it: cookie contained only a random key that, even if compromised, revealed no sensitive information about the user or service.
- Worked with virtually any web browser with support for cookies enabled (including text-only browsers and users with javascript disabled).

Weaknesses

- Each web server required a separate login for each web server.
- Each web server that supported X-509 client authentication had to be configured individually to do so.
- There were no idle-timeouts. A user was logged-in until he or she chose to logout or the hard session limit was reached.
- The code required on a departmental web server was too complex. This makes porting more time consuming and maintenance very difficult.

Central Requirements

Central Web Login Server

Passwords, if used, are sent only to the central weblogin service over SSL.

Login Once Per Session

Users need only authenticate once per session to access any number of cosign-protected campus sites.

Global Logout

Users can logout of all cosign-protected services by visiting a single URL.

Idle Timeout

Sessions have both idle and hard timeouts.

No Domain Cookies

There are no domain cookies used in this system.

Untrusted Environment

A compromised service host does not represent a compromise of the cosign system as a whole.

Requirements Evaluation

	A-Select	CAS	Cosign	PubCookie
Filters				
Apache 1.3.x	X	X	X	X
Apache 2.x		X	X	
ISAPI	X		X	X
Java Servlets	X		X	
Authentication Types				
guest accounts			X	
web form	X	X	X	X
Basic Auth			X	
x.509			X	
IP Address	X			
multifactor			X	X
logout/login				
reauthentication				
hard timeout	X	X	X	X
user quits browser	X	X	X	X
idle timeout			X	
global logout			X	
Security				
security compartmentalization		X	X	
no domain cookies	X	X	X	
intercepted cookies valueless			X	
n-tier authentication (proxy)		X	X	X

Sources:

http://middleware.internet2.edu/webiso/docs/webiso-questionnaire.txt
http://www.pubcookie.org/
http://a-select.surfnet.nl/
http://www.yale.edu/tp/auth/
http://et.aset.psu.edu/initiatives/credential/publications/index.shtml

IT Commons

A campus-wide effort to improve collaborative planning
and decision-making for information technology at U-M

What it is

The IT environment at the University of Michigan is a highly distributed one, with many schools, colleges and other units providing substantial services, in addition to those provided by central units. Through the Roadmap Initiative, we are defining a new collaborative method for decision-making about information technology that fits with the highly distributed nature of our institution, but reflects our desire and need for coordination.

How it works

"It's rooted in the core values and mission of the University, it reflects the diverse priorities of our many units and programs, and it emphasizes the advantages of creative collaboration. Moving in this direction will allow students, faculty and staff to use our IT assets more effectively in research, teaching and learning." -J. Hilton

"A critical mass of schools can develop a project and plan for its production, keeping the rest of the campus informed and perhaps attracting more schools to participate. It's a very different approach from having somebody build something off in a corner and either keep it to themselves or force other people to use it." -J. Williams

"There's nothing in this whole model that forces centralization or forces decentralization or forces anything. Units can opt in or opt out. What this really is about is doing IT the way we do the other things that have made this University so successful on the academic side: getting processes to work across the board and collaborating across units." - K. Bridges

Key Players in Cosign

Kevin McGowan, Central IT
Johanna Craig, Central IT
Wesley Craig, Central IT
Jarod Malestein, Central IT
Bill Doster, Central IT
Gavin Eadie, Central IT
Dan Hyde, Central IT

Tony Chan, Administrative Computing
Fritz Freiheit, Administrative Computing
Mark Montague, L, S, & A
Paul Saxman, CAEN
Cory Snavely, University Libraries
Wayne Wilson, Medical School

Community Requirements

Java Authentication Filter

J2EE stand-alone application servers turned out to be more popular than Java applications served via traditional web servers.

Centralized Guest Accounts

The number of locally created 'guest' accounts in use on campus systems was surprising, even among departments using the existing central authentication system.

External Web Hosting

Many departments either already had or were exploring the possibility of hosting their applications with a outside vendors.

SSL Load Balancing

Large critical applications required support for hardware SSL load balancers.

HIPAA Considerations

The University of Michigan Hospital required shorter idle timeout windows in order to comply with HIPAA.

Multi-factor Authentication

Administrative Information Systems required two tiers of authentication in order to access critical financial and personnel data.

Central Components

cgi

The central cgi is responsible for logging users into and out of the central cosign server. It is also responsible for registering each service a user logs into - this action ties the user's central login cookie to their session on individual application servers such as our web mail client, web directory client, or CourseTools environment. The prototype CGI was built to use Kerberos V/GSSAPI to authenticate the user.

daemon

The central daemon is responsible for maintaining the state of all cosign sessions. This includes keeping track of which users have logged in, logged out, and idle timed out. This also means the daemon keeps track of all of the service cookies that represent the authenticated web applications a user has accessed. The daemon has the ability to replicate its cookie database to multiply hosts, so a failure of one server does not constitute a failure of the system. The daemon answers queries of user identity from both the cgi and the filter, and talks to other daemons through a replication protocol. The daemon was written in C and has knowledge of Kerberos V tickets.

filter

The filter resides on an application server, and is not part of the centralized cosign infrastructure. The filter is responsible for determining which areas of a web site are protected by cosign and which are not. If a user attempts to access a protected area, the filter assures the user is authenticated, and obtains their username, authentication realm, IP address, and optionally a Kerberos ticket. This information can then be used by other authorization mechanisms to make further access decisions. The prototype filter was written in C for Apache 1.3.x.

Other Authentication Filters

ISAPI

The ISAPI filter runs on Internet Information Services (IIS) versions 5 and 6. It has the same core functionality as the prototype Apache filter. This filter was written to support the Business School's intranet and alumni web services. The ISAPI filter is also in use at the School of Information as part of their Cold Fusion service.

Java

The Java filter is compatible with J2EE 1.3 and 1.4. It was developed by staff at the College of Engineering (CAEN) and the Administrative Information Systems group (MAIS). The java filter was written for our campus PeopleSoft implementation as well as our CHEF/OKI installation.

Apache 2

The Apache 2.x filter is in alpha. It will be used by the School of Information, the School of Social Work, and the Housing Information Technology Office.

Additional Authentication Methods

Basic/Digest Authentication

The Cosign distribution includes a CGI which allows the deployer to use basic or digest authentication on the central weblogin server. The user authenticates to this one server via the web server's built-in authentication mechanism and Cosign leverages this authentication to the entire SSO environment -- the basic auth HTTP headers are sent only to the central server.

x.509

Similar to its support for basic auth, Cosign supports the use of x.509 certificates for initial sign-on. The user presents a personal certificate to the central weblogin server and Cosign provides access to the rest of the single sign-on environment. Only the central server need be configured to perform SSL mutual authentication directly with the user.

Form & Cookie

The default authentication method used by Cosign is a web form using Kerberos 5. A user submits her username & password and the form posts to a CGI which uses the Kerberos libraries to obtain and verify a Krb5 ticket. It would be trivial, however, to verify this form input with Kerberos IV, a local password file, a MySQL database, PAM, multi-factor, an LDAP directory, etc.

Application Support

Container Model

Cosign, like many other WebISO solutions, places the authenticated user's username in the REMOTE_USER environment variable. This means that any application written to work with standard web server authentication will work without modification on a Cosign-protected server.

PeopleSoft 8

The University of Michigan uses PeopleSoft software to track all human resources, student financial, and academic data. The campus is upgrading to PeopleSoft 8 in February. Load-testing of the Cosign-protected web interfaces to these new modules, using Cosign's J2EE servlet authentication filter, has been completed with excellent results.

FootPrints

FootPrints, from UniPress software, is commercial, web-based, helpdesk software. Like IMP and many commercial web applications, expects to be deployed in an infrastructure-poor environment and, thus, has its own integrated authentication. The department using this software explained the "container model" approach described above to this vendor. UniPrint enthusiastically embraced this model and will include optional reliance on REMOTE_USER in subsequent releases.

IMP & Horde

Mail.umich.edu uses IMP to provide web mail access to more than 70,000 unique users per month with daily peaks near 10,000 simultaneous sessions. Minor modifications were required to remove IMP's built-in login screen, but no changes were required to support Kerberos authentication between IMP and the IMAP servers.

Collaborative Portals

Traditional Portals

Many large organizations desire an integrated set of web applications with a unified interface called a 'portal.' Users enjoy the navigability of features afforded by such an arrangement as well as the visibility of options.

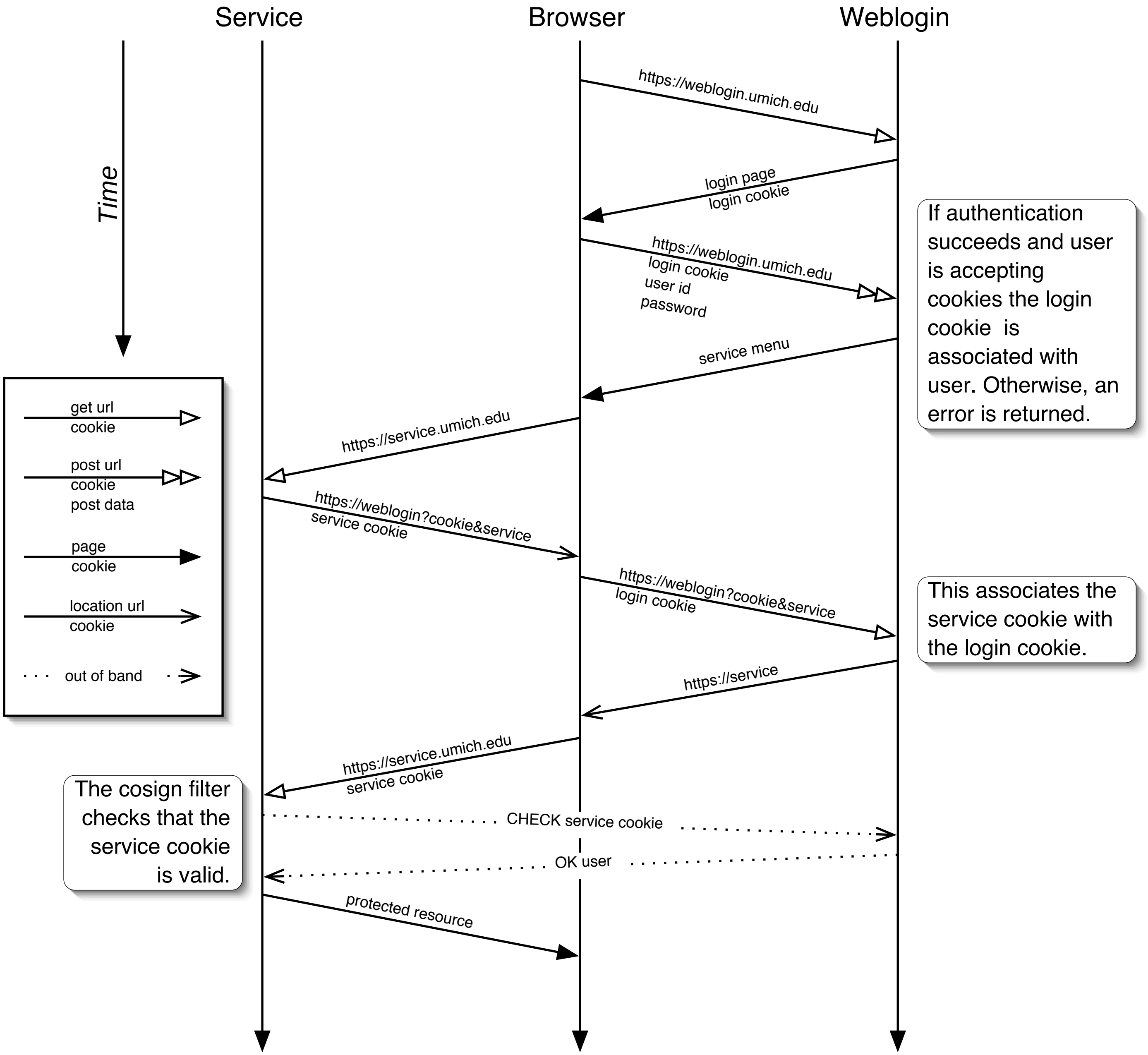
The Portal Problem

One problem with traditional portal application development is that it implicitly makes one of the oldest software design errors -- attempting to solve complex problems across multiple problem domains with a single piece of software. Portal projects become incredibly expensive and difficult to deploy (and, presumably, to maintain) when they attempt, in a single application or application framework, to deliver solutions ranging from web e-mail access to course collaboration. If these systems do manage to deliver all of their intended features, they are likely to do so by compromising performance. Additionally, these systems necessarily create an all-or-nothing environment, e.g., "all of our web applications must be written in language X" or "everything must come from vendor Y."

Collaborative Portals

Cosign, and similarly robust authentication frameworks, can create a "cooperative," lightweight portal. A simple service menu becomes the point of access for all services on campus. Each service provider has the freedom to make implementation decisions while still providing a portal's ease of use. Moreover, users access these cooperating services directly, with no intermediate -- allowing optimum efficiency.

Case 1: User Visits Weblogin First

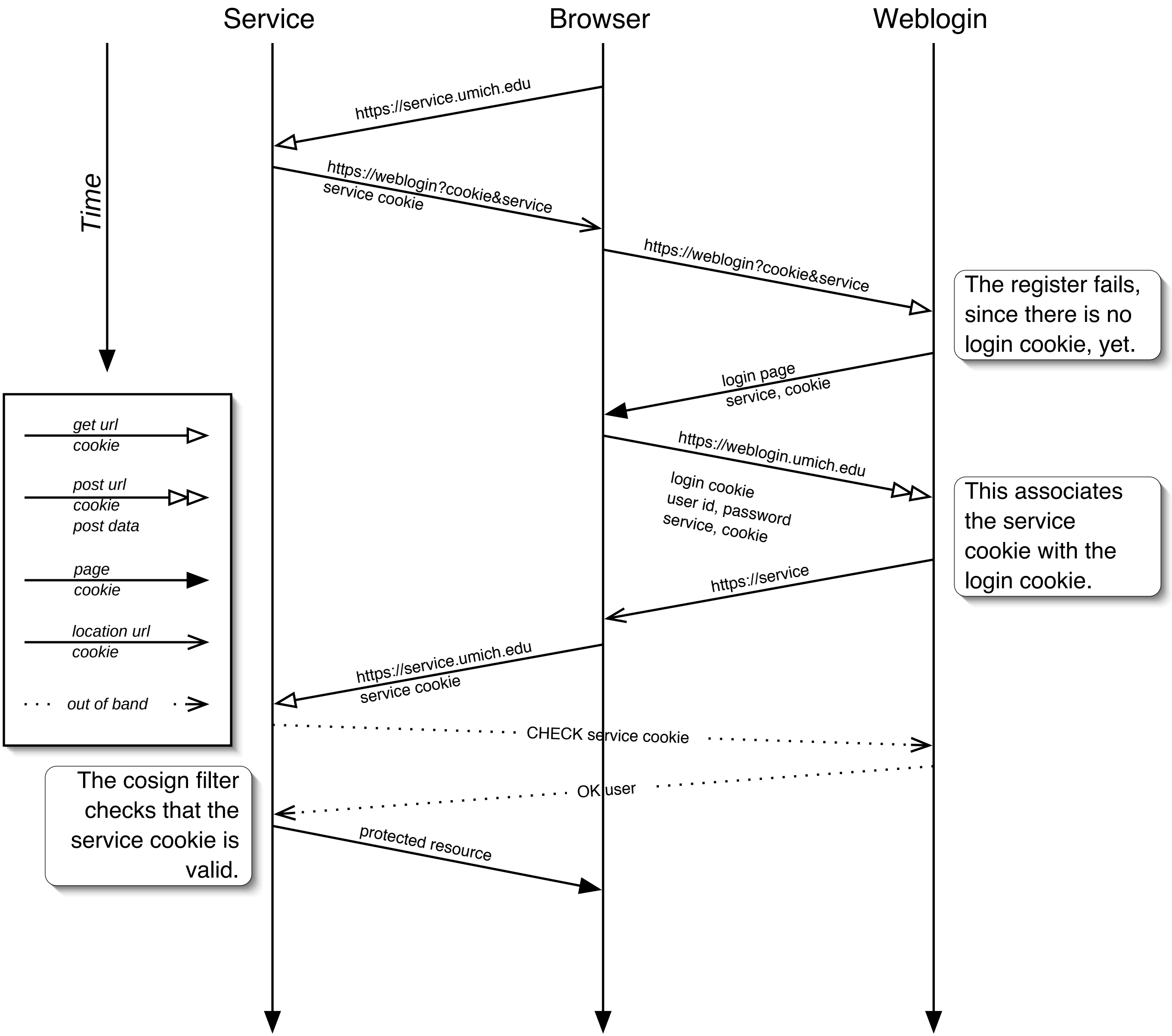


If authentication succeeds and user is accepting cookies the login cookie is associated with user. Otherwise, an error is returned.

This associates the service cookie with the login cookie.

The cosign filter checks that the service cookie is valid.

Case 2: User Visits Service First


Service
Browser
Weblogin
Time
https://service.umich.edu
https://weblogin?cookie&service
service cookie
https://weblogin?cookie&service
The register fails, since there is no login cookie, yet.
login page
service, cookie
https://weblogin.umich.edu
login cookie
user id, password
service, cookie
This associates the service cookie with the login cookie.
https://service
https://service.umich.edu
service cookie
CHECK service cookie
OK user
protected resource
The cosign filter checks that the service cookie is valid.
get url
cookie
post url
cookie
post data
page
cookie
location url
cookie
out of band