Open Source in
the Corporate World

Open Source Single Sign-On

Erin Mulder

CHARIOT SOLUTIONS

Agenda

- Introduction
- Single Sign-On for Multiple Applications
  - Shared directory (e.g. OpenLDAP)
  - Proxy systems (e.g. Yale CAS)
  - X.509 certificates
  - NTLM
  - Kerberos/SPNEGO
- Integration with Workstation Login
  - Shared directory
  - Kerberos
- Cross-Domain & Federated Single Sign-On

Not on the Agenda

- Authorization
- Auditing
- Identity Management
- Commercial Options
  - RSA ClearTrust
  - Netegrity SiteMinder
  - IBM Tivoli
  - etc.

Introduction

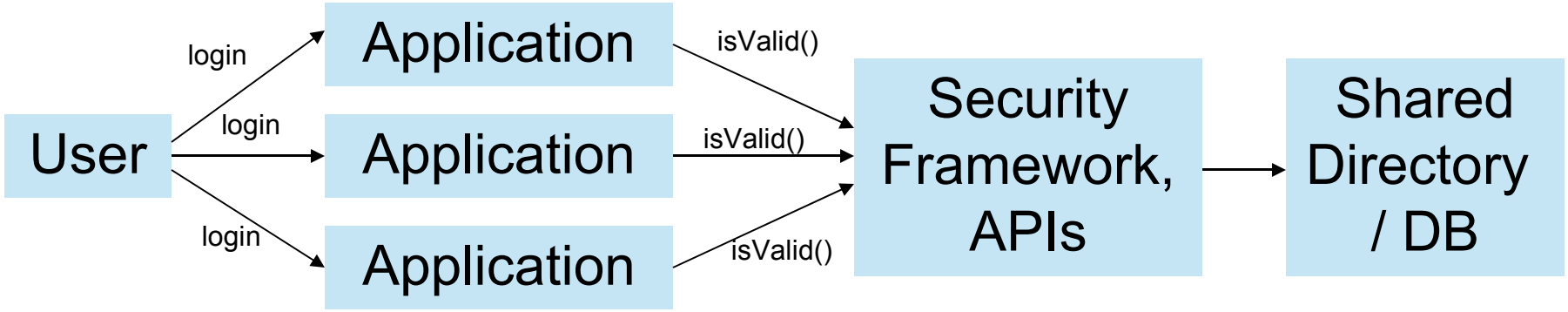
- **Authentication** vs. **Authorization**
  - Authentication verifies who you are
  - Authorization says what you can do
  - Not always the same service providing both
- **Single** password vs. **Seamless** sign-on
  - For today, single means you have one password
  - Seamless means you're not prompted more than once
  - Seamless isn't always single; single isn't always seamless
- **Passwords/keys** vs. **tickets**
  - Passwords allow proxies to authenticate you directly
  - Tickets vouch for your identity after authentication

Application SSO

- A little integration goes a long way
- Easy starting options include:
  - Shared directory (e.g. OpenLDAP)
  - Proxy systems (e.g. YaleCAS)
  - X.509 certificates
- If you're looking for true, seamless SSO:
  - NTLM
  - Kerberos/SPNEGO

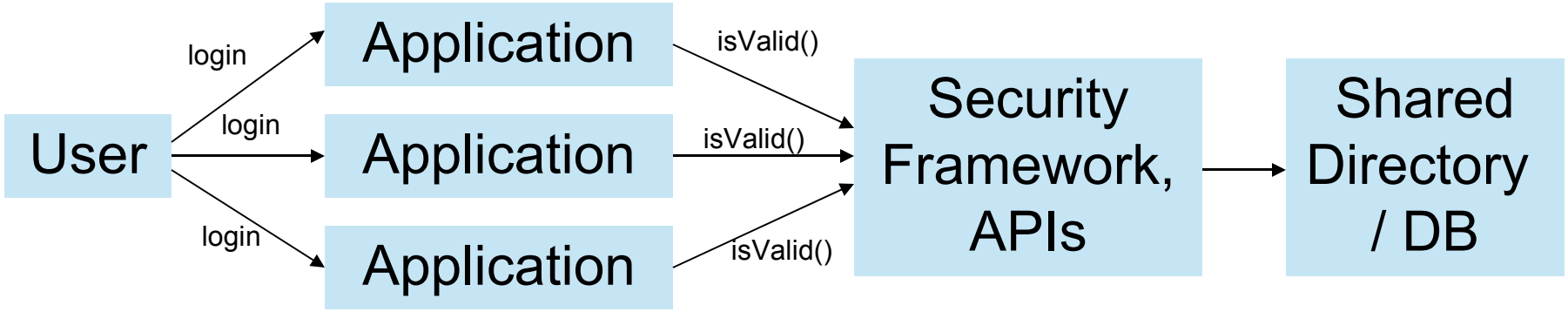
Shared Directory

- User accesses application
- Application collects username/password
- Application validates credentials using common security framework/APIs
- Security framework validates against single LDAP directory or database



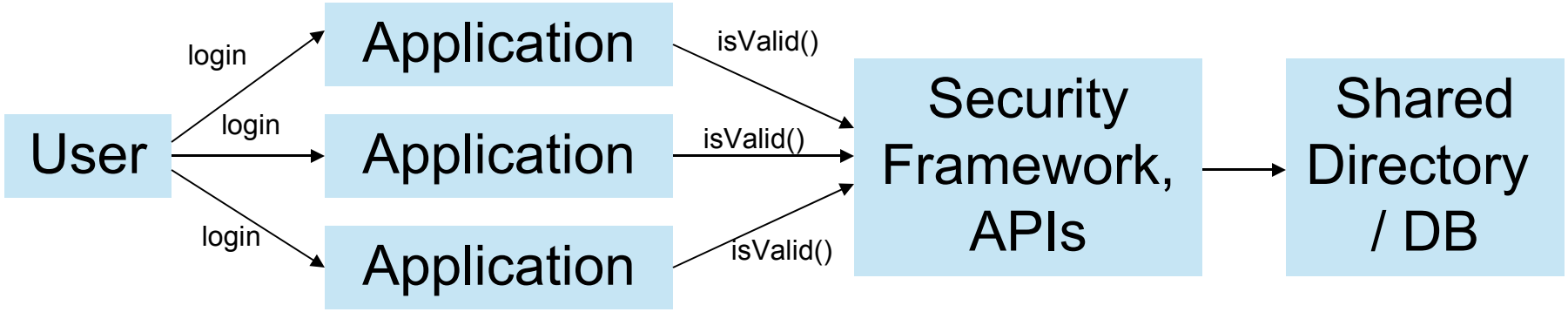
Shared Directory

- Advantages
  - Users only need one password
  - Easy to integrate with application servers
  - Can handle authentication, authorization and identity management
  - Nothing to configure on the client-side
  - Works on all platforms



Shared Directory

- Disadvantages
  - Users have to enter password again for each app
  - Every app needs to worry about authentication
  - Shared credentials are exposed to every application
  - Hassle to maintain/upgrade framework over time

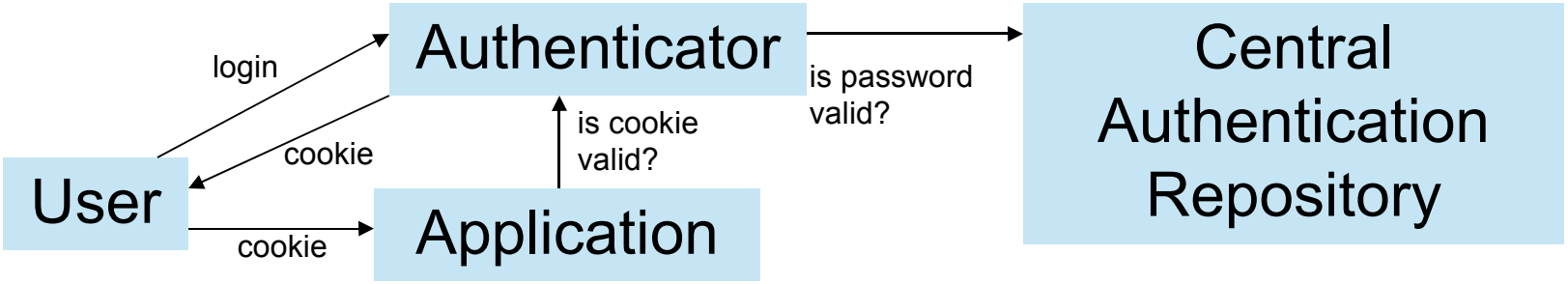


Shared Directory

- Implementing with Open Source
  - Look at OpenLDAP, MySQL, PostgreSQL
  - (In an existing windows domain, integrate with ActiveDirectory to reuse OS passwords)
  - Most application servers include security modules that can talk to an LDAP directory or RDBMS
  - Integrate with standard security framework (J2EE) or open source package (Acegi Security, SecurityFilter…) for login forms and authorization
  - Grab an open source directory/db browser:
    - LDAP browsers: Luma, JXplorer
    - DB browsers: SQuirreL, TORA

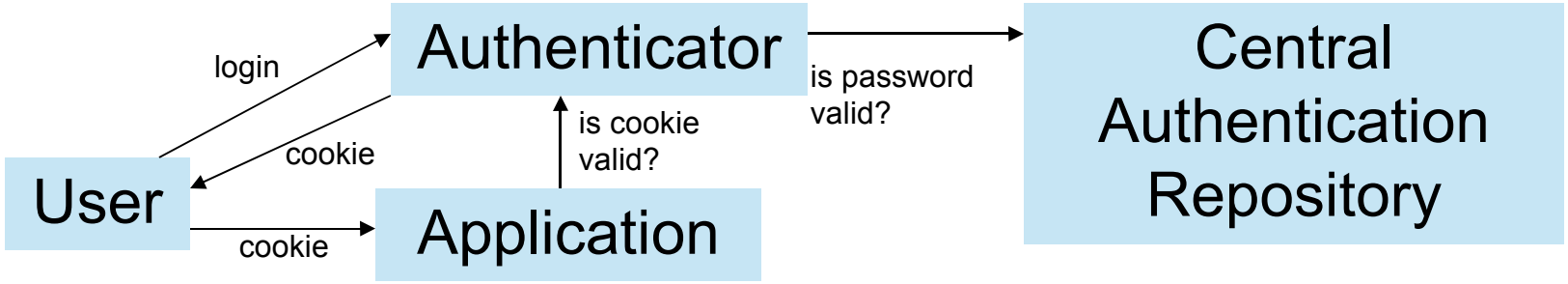
Proxy Systems

- User accesses web application and is redirected to authentication web app
- Authenticator collects username/password and validates against central auth store
- Sets a cookie and redirects user back to original application
- Application communicates with authenticator to ensure cookie is valid



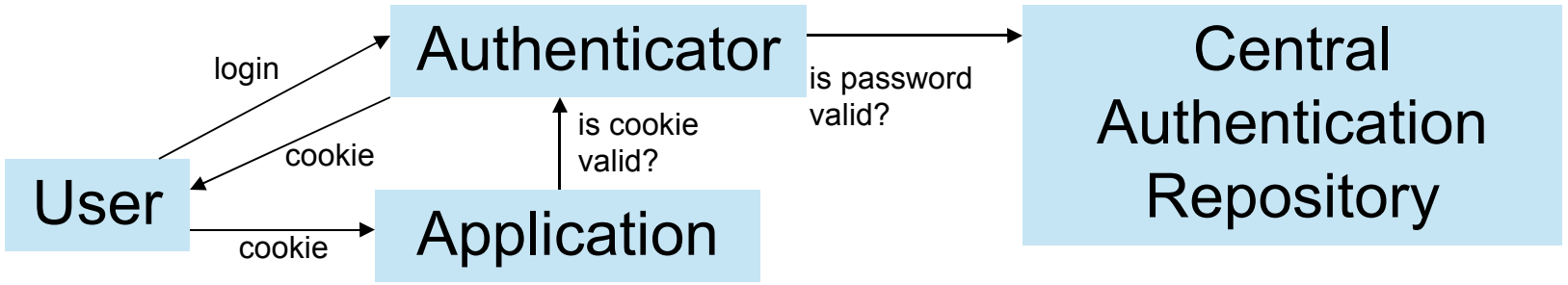
Proxy Systems

- Advantages
  - User only has to log in once per “session” to access multiple web applications
  - Credentials are only passed across wire once
  - Applications don’t need to trust each other



Proxy Systems

- Disadvantages
  - Still requires at least one login per browser “session”
  - Credentials are still passed across network
  - Applications have to communicate with authenticator to validate cookie
  - Doesn’t integrate seamlessly with J2EE declarative security (though you can usually fake this)



Proxy Systems

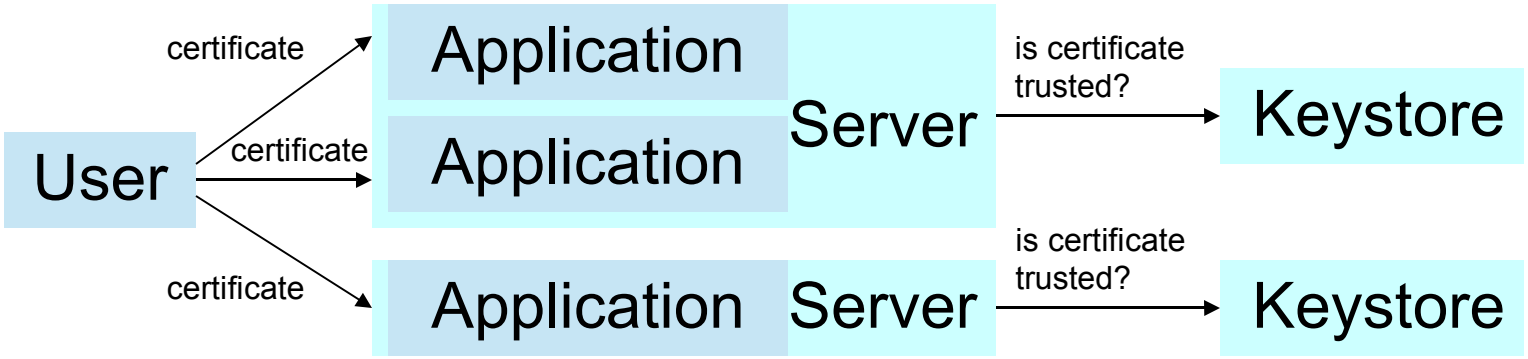
Yale CAS Server

ESUP Portail

- Implementing with Open Source
  - Check out the Central Authentication Service (CAS) project
  - Most widely used implementation is **Yale CAS Server**
  - Hook up to an auth store with **CAS Generic Handler**
  - Get started in minutes with the "Quick Start" packages distributed by **ESUP-Portail**, which include:
    - Yale CAS Server
    - Tomcat
    - CAS Generic Handler (CGH)
    - Ant scripts to install and run it all
  - **CASFilter** supports protecting your application's pages with Servlet filters, JSP custom tags and direct API calls

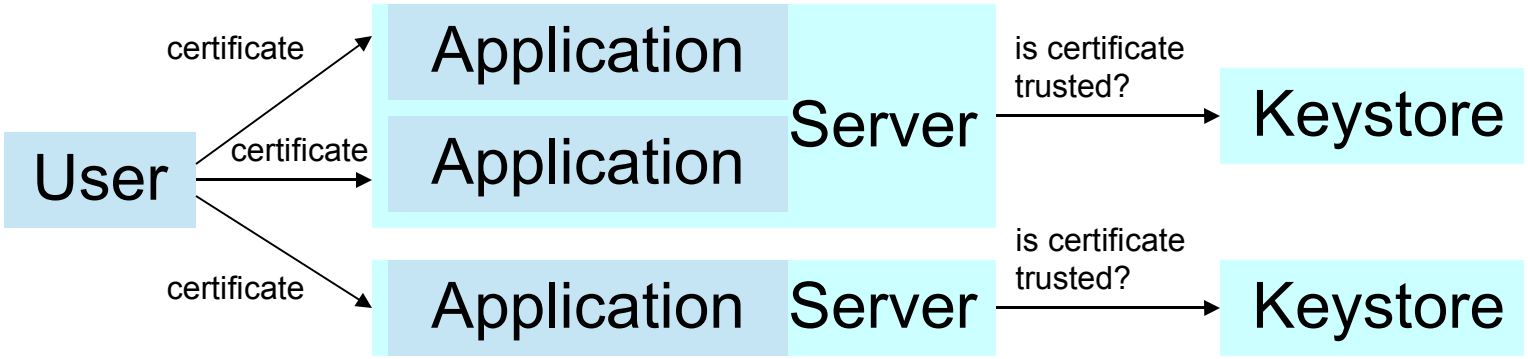
X.509 Certificates

- Certificate is installed in user's browser
- User visits web application and browser authenticates without user interaction
- Server checks certificate against trust store to validate user's identity
- Server asserts user identity to application



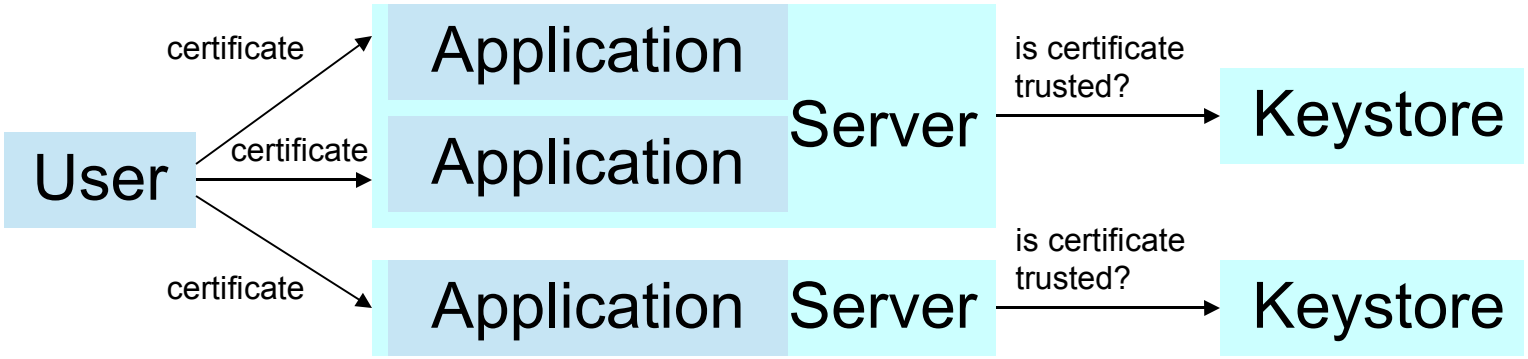
X.509 Certificates

- Advantages
  - Credentials never pass over the wire
  - Seamless sign-on, even across domains
  - Can be configured externally without changing web application
  - Can be configured at either web server or application server level
  - Works with J2EE declarative security



X.509 Certificates

- Disadvantages
  - Must generate and install certificate in every user's browser
  - Must re-install certificates upon expiration
  - Applications on the same server must share trust (only for authentication, not authorization)
  - Need to distribute keys to every server's keystore or implement a sharing mechanism

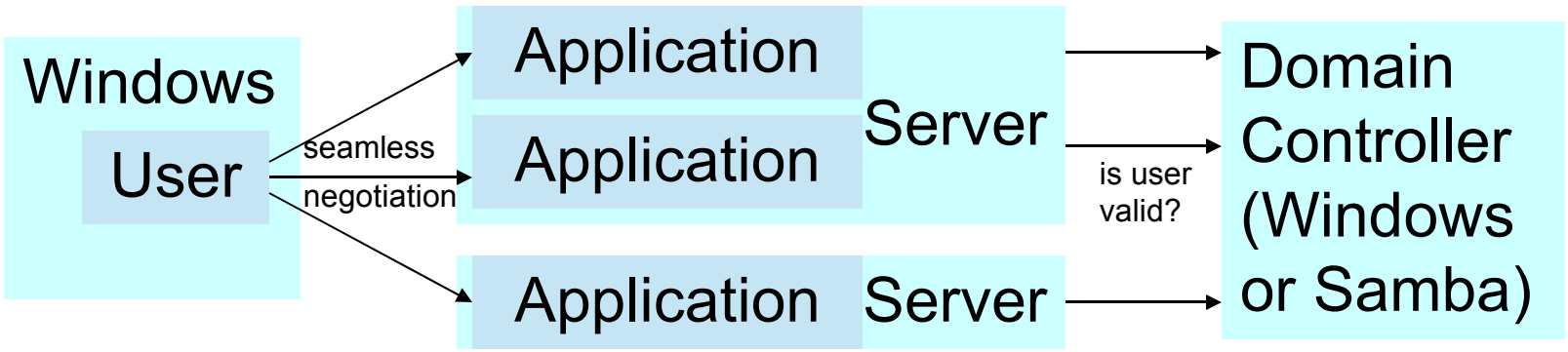


X.509 Certificates

- Implementing with Open Source
  - Use OpenSSL or Java's keytool to generate client and server certificates
  - *Optionally, sign all client certificates with a private certificate authority to simplify trust maintenance*
  - Configure server to trust all client certificates
  - Install certificates in each user's browser
  - *Optionally, configure browsers to trust server certificate (if not signed by trusted CA)*
  - Configure web server, web container or both to require client certificates
  - If using J2EE declarative security, configure web application to use CLIENT-CERT authentication
  - Check out certificate management tools like TinyCA and roCA

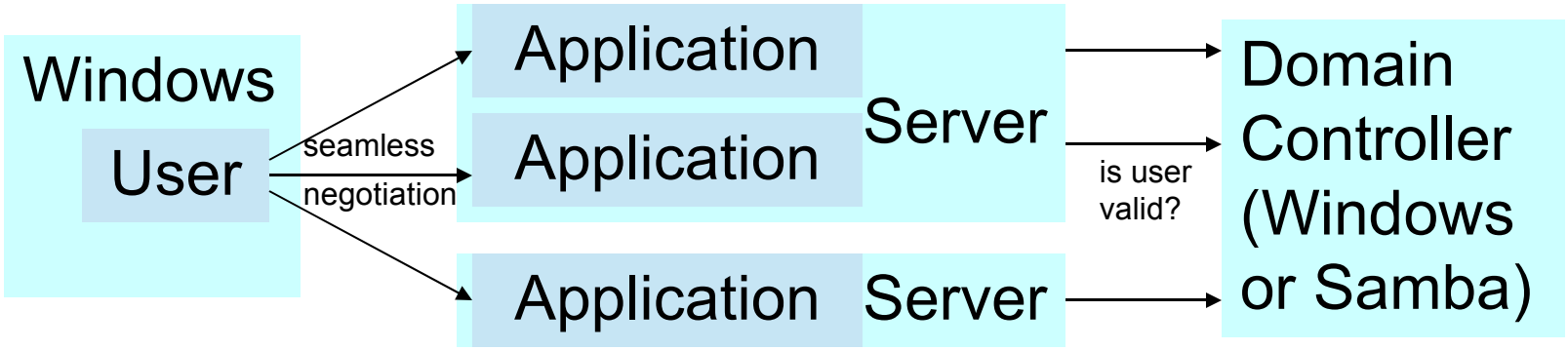
NTLM Authentication

- User logs into Windows workstation and accesses a web application
- Server sends NTLM challenge (from PDC)
- Browser responds without user interaction
- Server validates with domain controller



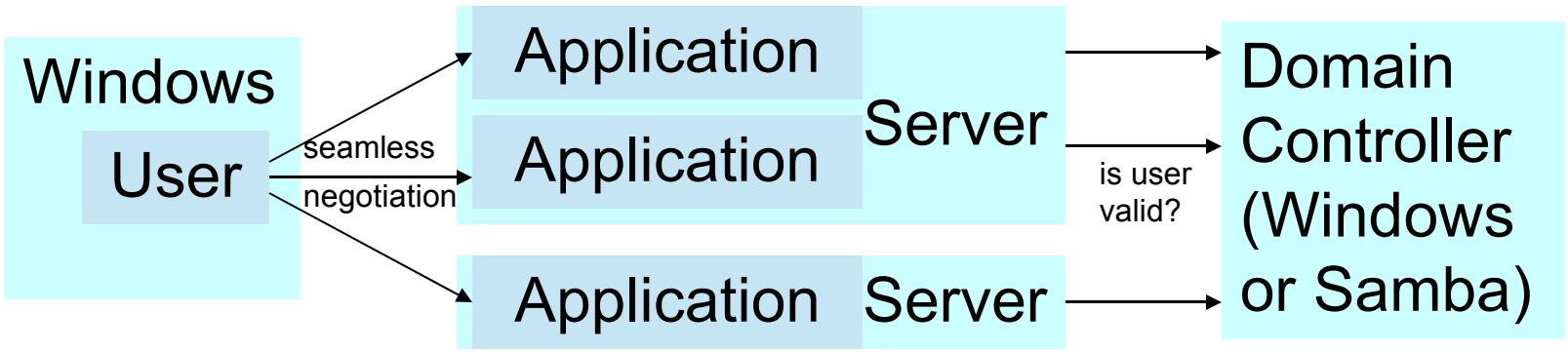
NTLM Authentication

- Advantages
  - True, seamless, single sign-on
  - Works in both IE and Firefox (with configuration)
  - Can be configured to fall back on other types of authentication (e.g. basic)
  - Works against both Windows and Samba
  - Fairly easy to set up



NTLM Authentication

- Disadvantages
  - Somewhat deprecated
  - Web server modules no longer actively supported
  - Known security vulnerabilities
  - Users must be part of the same domain



NTLM Authentication

- Implementing with Open Source
  - Install Apache
  - Install mod_ntlm (no longer well supported)
  - Configure Apache virtual host directory entry to require NTLM authentication
  - Optionally include other authentication types
  - Configure Firefox to allow NTLM authentication
  - Configure web application and/or security filter to get user identity from remote_user header

The Apache Software Foundation
http://www.apache.org/

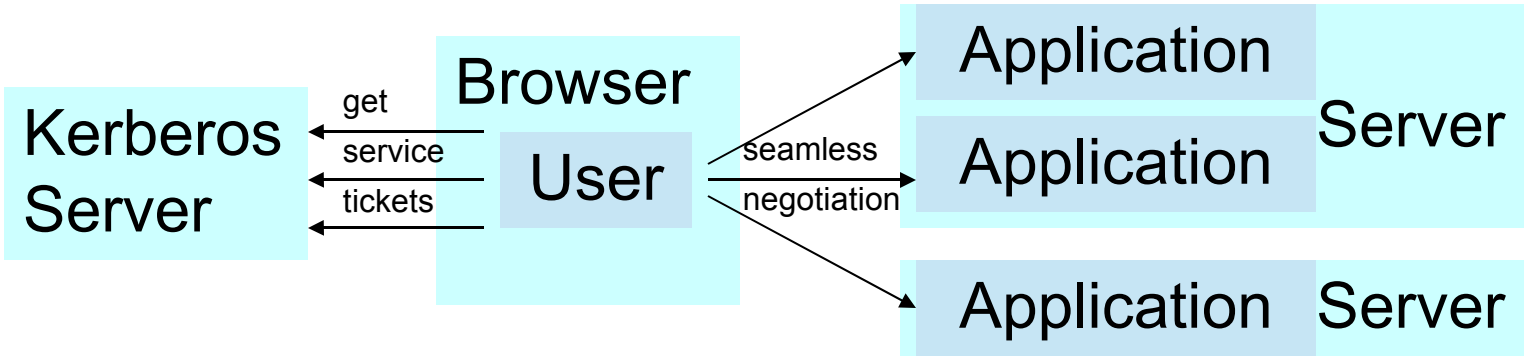
mod_ntlm

samba

The browser you can trust Firefox

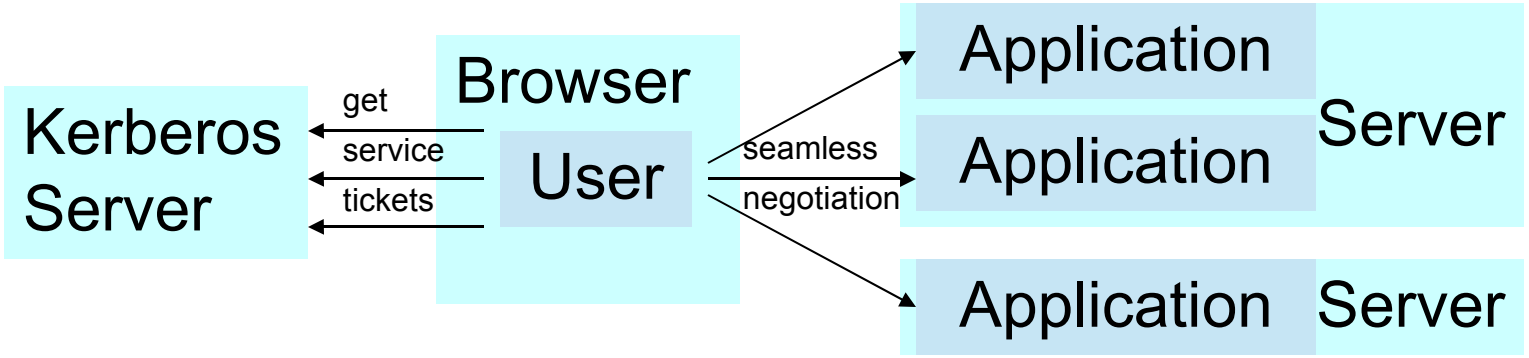
Kerberos/SPNEGO

- User logs into workstation and accesses a web application
- Server sends Kerberos challenge
- Browser gets service ticket from KDC and sends it to server
- Server decrypts ticket and trusts user



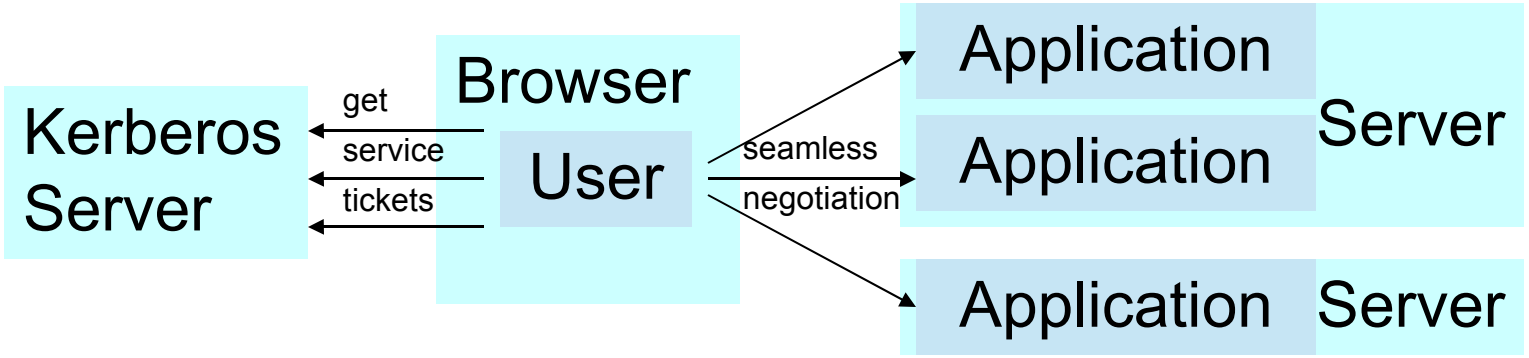
Kerberos/SPNEGO

- Advantages
  - True, seamless, single sign-on
  - Credentials never leave user's computer
  - Applications don't need to trust each other
  - User doesn't need to trust applications
  - Tickets expire periodically
  - Very secure



Kerberos/SPNEGO

- Disadvantages
  - SPNEGO support is still maturing
  - Generally painful to configure and maintain
  - Kerberos servers must be accessible to client machines (more critical network hops to maintain)
  - Requires rough time sync on all machines involved



Kerberos/SPNEGO

- Implementing with Open Source
  - Set up a Kerberos realm
  - Configure service/application in Kerberos
  - Configure users in Kerberos
  - Install Apache 2.0 with mod_spnego
  - Configure Apache to require Kerberos authentication for protected paths
  - Install negotiateauth extension in Mozilla/Firefox
  - Configure browser to enable kerberos authentication

Workstation SSO

- Three basic enabling configurations:
  - Shared LDAP directory (e.g. Active Directory)
  - NTLM/Samba-based domain
  - Kerberos-based domain
- Windows (2K and later) domains are already doing this
- Unix variants can be configured to handle any of them with the right Pluggable Authentication Modules (PAM)
- Mac OS X uses netinfo, which can handle kerberos, and OpenDirectory for LDAP

Cross-Domain SSO

- Cookies don't work
- Client certificates do work
- Kerberos supports domain trust relationships
- Yale CAS 2.0 supports proxy ticketing
- Plenty of standards (Liberty, SAML, etc.) but little stable open source support
- Keep an eye on this area, but look to commercial tools for out-of-the-box features

Open Source in
the Corporate World

Open Source Single Sign-On

Discussion

CHARIOT SOLUTIONS