



Dynamic Access Control for Microsoft SharePoint

Leveraging the Safewhere access management suite to securely extend SharePoint to provide rich and flexible access control for inside as well as outside users

June, 2009

Table of Contents

Authentication and authorization challenges with SharePoint	3
User centric authentication and Resource centric authorization	3
Beyond SharePoint Role Based Access Control	4
Dynamic, rich authorizations with Safewhere	6
A note on SharePoint Groups	7
The underlying Safewhere web and SOA Access Management infrastructure.....	8
Centralized management of authorizations and policies	8
Distributed policy enforcement.....	9
Consolidated audit trail	9

Authentication and authorization challenges with SharePoint

With the newest release of Microsoft SharePoint (WSS 3.0 and MOSS 2007), organizations are increasingly looking to extend their SharePoint portals to serve external users.

In particular, many organizations are extending SharePoint in a business to business scenario requiring secure identification and authorization of business partners and their users.

The general response so far has been to create separate user databases for maintaining application specific accounts for outside users. This approach carries not only significant development costs but also an ongoing administrative overhead.

The challenge in this scenario is that you – and your business partners – get yet another user directory to manage, with the risk of getting out of sync with organizational structure and actual users. The most obvious risk is a former partner employee that never gets removed from the extranet user database. The extranet is thus left open to misuse by the proverbial disgruntled former employee.

To eliminate these and other issues, Safewhere offers a complete solution tightly integrated into SharePoint, with broad access control enforcement across not only SharePoint, but the entire web and service oriented architecture.

User centric authentication and Resource centric authorization

To solve this problem, many organizations look to web single sign-on with *federated* authentication, which lets users seamlessly and securely transfer their identity from their own to another organization. This is most often done using either the SAML¹ standard or the Microsoft backed WS-Federation approach².

Federation supports what is sometimes referred to as *user centric identity*; the user establishes her identity by authentication somewhere outside the control of your SharePoint application.

On the other hand you must adopt a *resource centric authorization* approach to protecting your SharePoint application and supporting services. This goal is to manage access independent of how and where the user authenticates, e.g. through federation from business partners, through the internal Active Directory domain, or through dedicated extranet user database.

Combining user centric authentication with resource centric authorization leads to a “hybrid” federated solution as illustrated in figure 1.

Without the burden of authentication, a single SharePoint application may uniformly support any number of authentication mechanisms. This includes the need for varying strengths of authentication, e.g. single factor for some content (username/password) and 2-factor for other content (RSA SecurID, biometric devices, SMS-based one time passwords, OTP, etc.).

¹ SAML: Security Assertion Markup Language. Widely adopted in government and elsewhere. The latest and most likely final version is SAML 2.0.

² As of October 2008, Microsoft has just publicly announced that they will include SAML 2.0 support in the next version of their federation framework and server, codenamed “Geneva”.

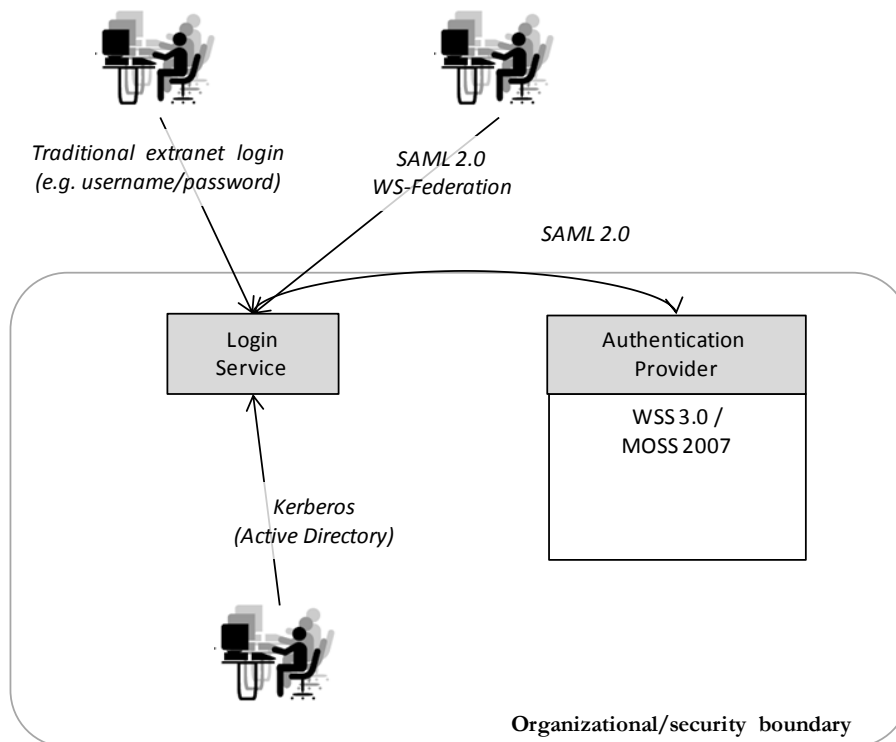


Figure 1. Support for multiple authentication mechanisms

In figure 1 SharePoint accepts only federated login through a central login service. The login service in turn supports both federation and login using any range of authentication mechanisms.

Beyond SharePoint Role Based Access Control

Access control in SharePoint is implemented in a model very similar to that of the file system, leveraging the core RBAC concepts of *Users*, *Roles*, *Objects*, *Operations*, and *Permissions*. Using these concepts you specify *Access Control Lists*, *ACL*, to control access to information in SharePoint (Sites, Lists, Items, etc.).

This approach works well for small groups of users and small SharePoint sites. But as you expand your services beyond your own organization, the administrative burden will grow exponentially as the number of roles explodes. As with any RBAC model, also SharePoint falls short when trying to express things like having to be member of two different roles in order to obtain a certain permission level. As this is simply not possible, you end up having to set up “combination roles” such as *PartnerX-Editor* and *PartnerY-Administrator*.

What is needed is the possibility to combine roles – or *claims*³ as it is in the general case – as illustrated in figure 2.

³ A *claim* is like an attribute and state things like you name, age, organization, favorite color, etc. Relative to attributes, claims employs the notion of *issuer*, as in “the issuer claims something about the subject (e.g. you)”.

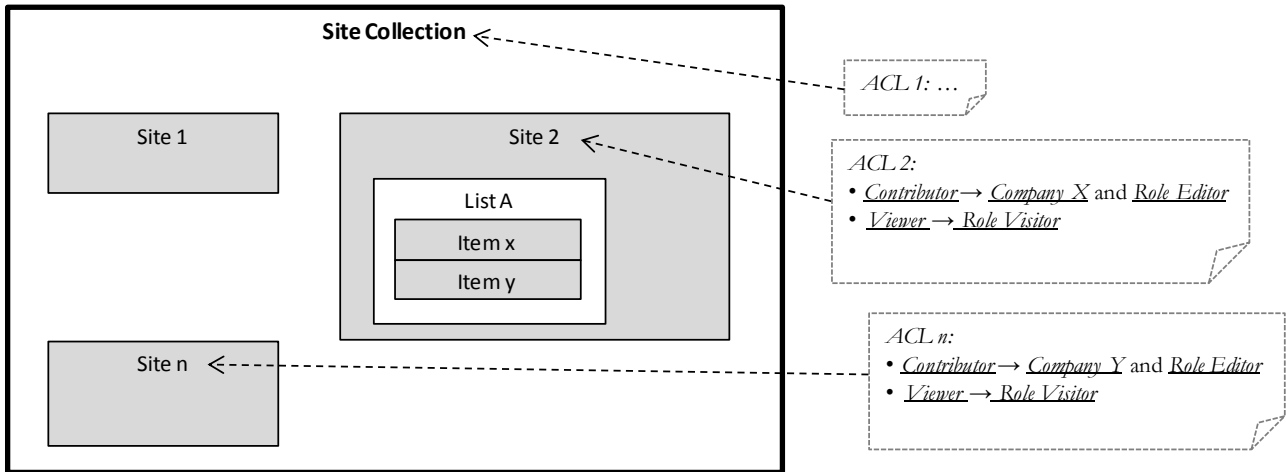


Figure 2. Rich expressions of SharePoint access control using general claims

Using this approach you will work directly of the *claims* made in the token issued by the Login Service shown in figure 1.

The following sections describe how to achieve the benefits outlined so far using the Safewhere authorization and access control infrastructure for web and service oriented architectures.

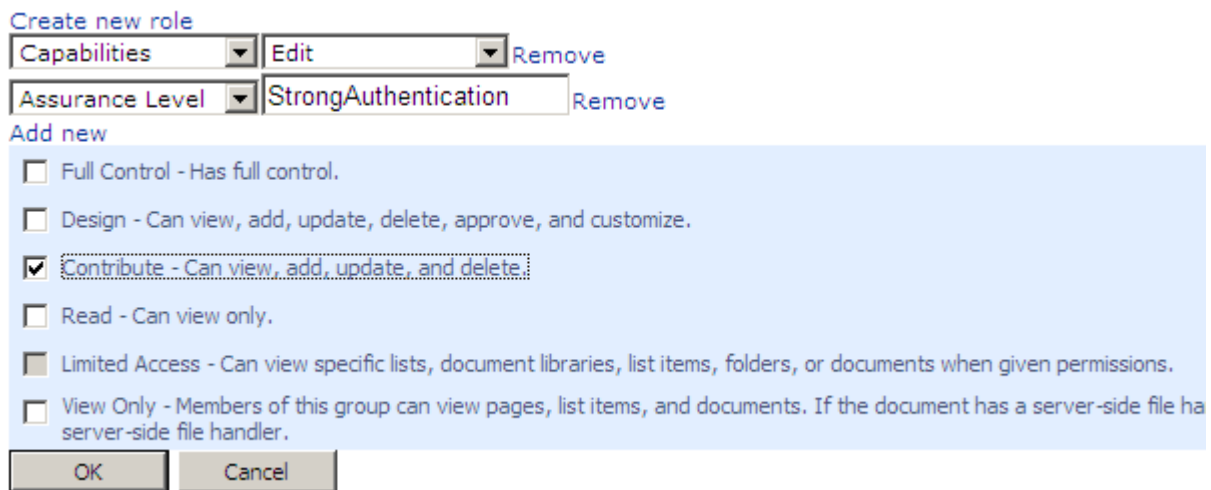
Dynamic, rich authorizations with Safewhere

Enabling external and diverse audiences access to SharePoint applications requires the ability to manage two more aspects:

- Multiple authentication mechanisms – including federation – should be possible in one SharePoint application. *Permissions* should be expressed in *claims* independent of authentication origin and mechanism (figure 1)
- It should be possible to express *permissions* for SharePoint objects in richer terms than is currently possible. We need to express things like “you must be 2-factor authenticated *and* you must be assigned the ‘edit’ right in order to act as a contributor to this SharePoint list” (figure 2).

By leveraging the Safewhere authorization and access control infrastructure for web and service oriented architectures this is possible.

Figure 3 show an example of how this may look in SharePoint: Setting the requirement that the “Contribute” *permission level* requires the user to have two *claims*, namely a “Capability” (C) with the value “Edit” and an “Assurance Level” (AL) with the value “StrongAuthentication”.



The screenshot shows a 'Create new role' dialog box in SharePoint. At the top, there are two dropdown menus: 'Capabilities' with 'Edit' selected and 'Assurance Level' with 'StrongAuthentication' selected. Below these are 'Add new' checkboxes for various permission levels: 'Full Control - Has full control.', 'Design - Can view, add, update, delete, approve, and customize.', 'Contribute - Can view, add, update, and delete.' (which is checked), 'Read - Can view only.', 'Limited Access - Can view specific lists, document libraries, list items, folders, or documents when given permissions.', and 'View Only - Members of this group can view pages, list items, and documents. If the document has a server-side file handler.'. At the bottom are 'OK' and 'Cancel' buttons.

Figure 3. Setting claims-based Permissions for an object in SharePoint

Having defined a permission like that of figure 3, figure 4 shows a typical setting of all permissions for a SharePoint object. Note how one permission level requires both editing rights and strong authentication, where as the view right alone is enough to get read only access to the object.

Use this page to assign users and groups permission to this list item. This list item does not inherit permissions from its parent folder or list.

Inherit permissions.

Required claims or Roles	Sharepoint permissions
Rasmus Helwigh	Full Control
Viewers	View Only
DocCenter Owners	Full Control
DocCenter Visitors	Read
DocCenter Members	Contribute
View	Read
Edit and StrongAuthentication	Contribute
Create new role	

Figure 4. Example of permissions for a SharePoint object

In the above illustrations the notion of a *Capability* claim is from the underlying Safewhere model, which among other claims, introduces modeling of *capabilities*, i.e. things you can do. This is also sometimes referred to as *rights, entitlements, or authorizations*.

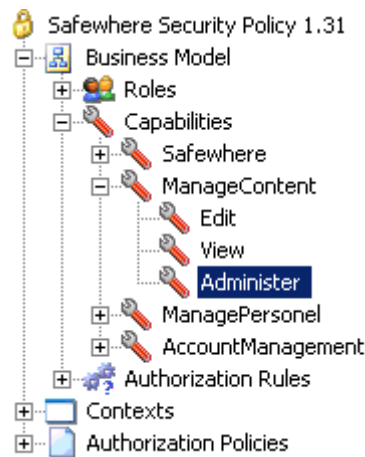


Figure 5. Capability modeling in the Safewhere authorization model

As indicated in figure 5, access to Safewhere itself is governed by its very own models and enforcement policies. This means that Safewhere is one hundred percent bootstrapped with no direct reliance on any specific authentication mechanism and underlying user store.

A note on SharePoint Groups

Traditional practice when working with permissions in SharePoint is to employ the SharePoint group concept as a convenient abstraction and extra degree of freedom.

This is still possible using the Safewhere solution. Still our experience with this new rich type of permissions shows, that it may often be a better practice to work directly with claims as indicated in the screen shot in figure 3 and 4.

The underlying Safewhere web and SOA Access Management infrastructure

*Safewhere*identify* coupled with *Safewhere* authorize* and *Safewhere*protect* each and in combination bring new possibilities and unique features to the web and SOA landscape.

This underlying Safewhere suite offers central administration and overview coupled with decentralized enforcement of access control policies and audit logging. This includes the dedicated tight integration with SharePoint as illustrated in this document. But in a broader context all web resources, that is, web sites and web services – in Microsoft .NET or Java – are protected from unauthorized access.

Centralized management of authorizations and policies

At the core of the offering is the *Safewhere*authorize* which are managed centrally using the user interface illustrated in figure 6. As the figure indicates, three separate concepts are used:

1. *Business Model*. This is where you define what you can do in the organization. This is typically defined along business process at a level of granularity at which you want to distinguish the users of your IT infrastructure. As an example one organization may simply specify a *Capability* to handle personnel, whereas a more complex organization may specify detailed *Capabilities* of hiring, firing, giving leave, etc.

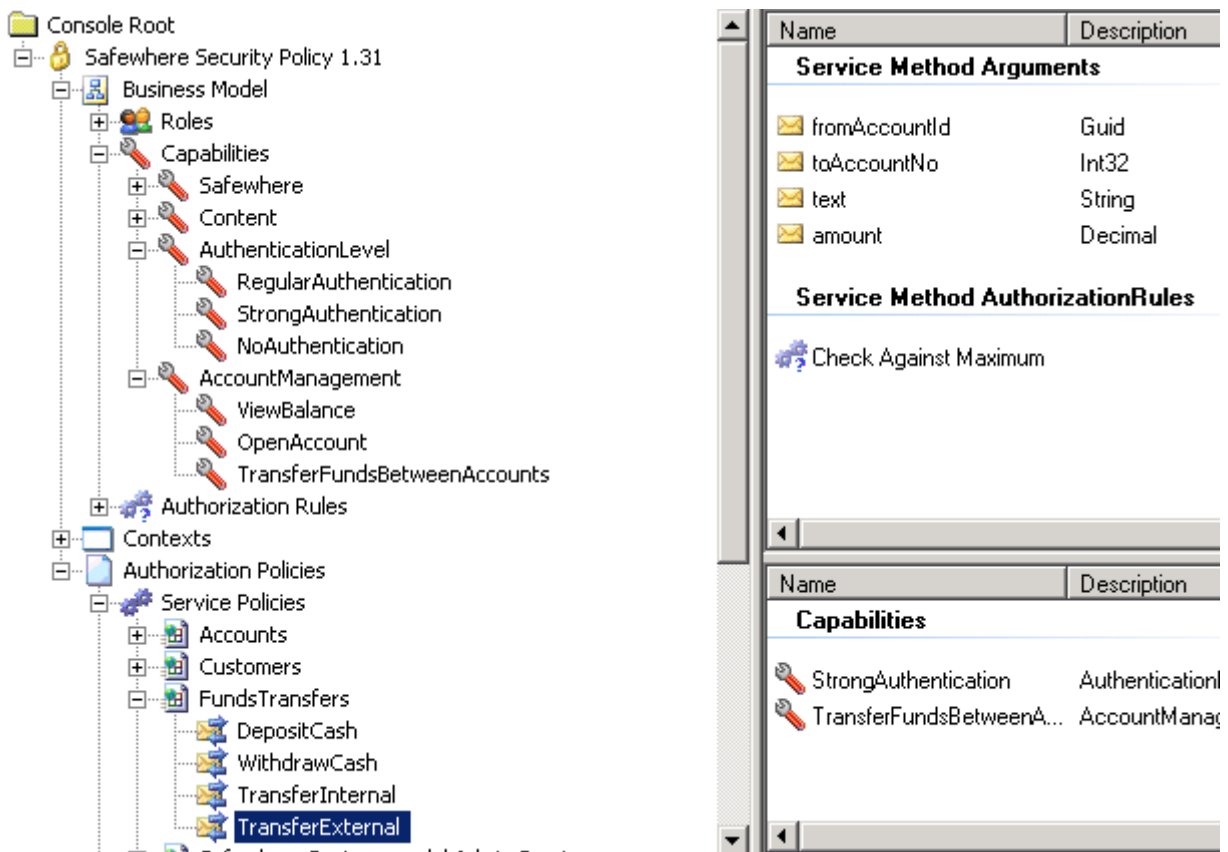


Figure 6. The Safewhere*authorize and Safewhere*protect modeling user interface

2. *Contexts*. You assign specific users and groups to Safewhere *Roles*. This is where the gap is bridged to user stores and authentication mechanisms, including federated authentication.

Once the *business model* and the – contextual – *role assignments* are in place the system is ready to issue *tokens* to users.

Missing are the requirements for granting or denying access to concrete resources.

3. *Access Control Policies*. This is where access control policies are set for concrete systems and applications. Figure 6 shows how a policy is set for a particular method of a web service, the “TransferExternal” method of the “FundsTransfer” service. Note how two *Capabilities* are required in addition to an *Authorization Rule* checking the actual amount being transferred against the allowed amount. Policies are actively pushed to protection points, e.g. web sites or web services.

Note that there is no representation of a SharePoint policy as SharePoint itself includes an access control infrastructure in the form of permissions and access control lists, which may in turn integrate into the above mentioned Safewhere *Business Model*.

Distributed policy enforcement

By application of the issued tickets/tokens the distributed Safewhere protection infrastructure enforces the distributed policies, thus effectively and transparently guarding access to all web sites and web services.

The Safewhere infrastructure will – transparent to applications – ensure that tokens are propagated (sometimes referred to as *identity propagation*) and subsequently checked as part of the access control check at the receiving end of the request.

Consolidated audit trail

Safewhere dynamic access control delivers comprehensive auditing and reporting capabilities, enabling organizations to track potential security problems, help ensure user accountability, and analyze evidence in the event of a security breach.

Safewhere modeling and enforcement create a formal and extensive data foundation on top of which a set of reports are built to document who may do what, and not least, who actually did what.

For more on the Safewhere offering please refer to products sheets on each individual component, *Safewhere*identify*, *Safewhere*authorize*, and *Safewhere*protect*.

About Safewhere

Safewhere represents the next generation in single sign-on and access control for web and service oriented architectures. By innovative application of communication and security standards with current and emerging technology platforms Safewhere has formed partnerships with global and regional system integrators and software vendors of complementary offerings. Read more at www.safewhere.net.