

More SharePoint Security. Less Effort.

Claims Based Authorization in SharePoint 2010

Antonio Maio
Senior Product Manager, SharePoint Solutions
antonio.maio@titus.com

SharePoint blog: www.titus.com/blog
Twitter: AntonioMaio2



Agenda

- Introduction
- Claims Based Security in SharePoint 2010
 - What is a Claim?
 - Authentication vs Authorization
 - Information Sharing Considerations - Why this is important
- Architecture and Trusted Identity Providers
 - Active Directory Federation Services 2.0
 - Claims Rules
- Custom Claim Providers (Developer Focused)
- Summary

TITUS Overview



- Data Security & Classification Market Leader
- Over 300 Enterprise Customers
- Over 2 Million Users Deployed

- SharePoint Security
- Email and Document Marking
- Data Loss Prevention

Check out our SharePoint blog:
<http://www.titus.com/blog>

Gartner | 2011
COOL VENDOR



Security & Compliance Solutions

Claims Based Security in SharePoint 2010

- SharePoint 2010 introduced a new method of authenticating user identities - Claims Based Authentication
 - Still supports Classic Windows Authentication
 - Still supports Form Based Authentication, but must go through Claims Based Authentication
- Claims are User Attributes – actually they're more than that...
- Enables Enforcement of Robust Security Policies
- Enables Security Policies to be Dynamic and Very Specific
- Enables Single Sign On Across Domains (through Federation)
- Scalable Alternative to Security Groups

What is a Claim?

- User attributes
- Metadata about a user
- AD attributes/LDAP attributes
- Claims are trusted assertions I make about myself
 - Claims are trusted user attributes
 - Retrieved from a trusted identity provider
 - Packaged and signed in a standards-based format (ex. SAML)
 - Allow me to take my identity across network boundaries in a trusted and secure manner
- Examples
 - name, email, place of birth, security clearance, age, isOver18, rank, etc.

What is Authentication?

- Determining if someone is who they say they are
 - Typically done today through username/password
- How do Claims go beyond this?
 - Verify other information about a user – their Claims
 - Facilitate more complex authentication processes – Ex. 2 factor auth
 - Single Sign-On across systems in different domains

What is Authorization?

- Determining what resources users are permitted to access and what actions they're permitted to perform
 - Typically done through policies using information about the user, information about the resource, etc...
- Using Claims can enhance Authorization to...
 - Be specific to the user
 - Be done without knowing who the user is
 - Be dynamic – ex. changes in a user's security clearance
 - Include environmental attributes (current time, GEO location, connection type, etc.)
 - Be an alternative to security groups – Groups do not scale
 - Policy Example: user must be part of GroupA and GroupB and GroupC to access a resources

Protecting vs Sharing Information

- Dealing with multitude of information
 - From Many Sources – Internal, External, System Gen, Historical, etc...
 - Content Growing at Incredible Rate
- Military/Government – Success requires sharing data
 - Enterprises Sharing Data also critical – departmental, with partners
- Protecting sensitive information also very important
- Ensure right people are accessing the right information
- Sensitive data sitting beside non-sensitive data

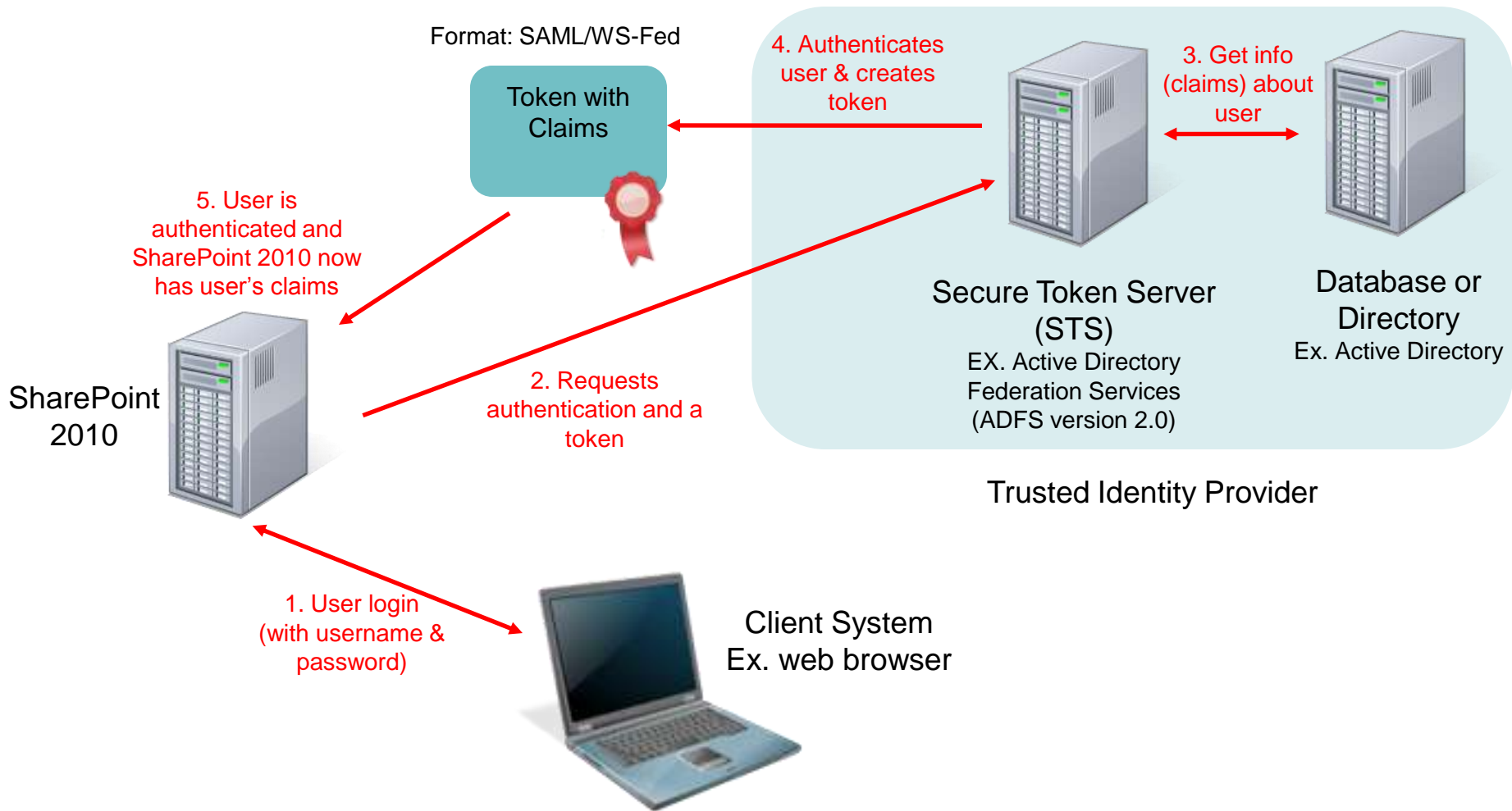
Information Sharing Considerations

- User Identity or Trusted Claims
 - Who am I, What's my clearance level, etc...
- All Methods of Accessing/Viewing Content
 - Web view, Search, Explorer, Roll-ups, custom web parts, Client Obj Model, Server Obj Model
- Leverage Document Metadata
 - What's the classification on this document
- Automation is Critical
 - Ensures access control policies are consistently applied
- Environmental data (if necessary for policies)
 - Time of day, Geo-location, Connection type, Device

Enabling Authorization in SharePoint 2010

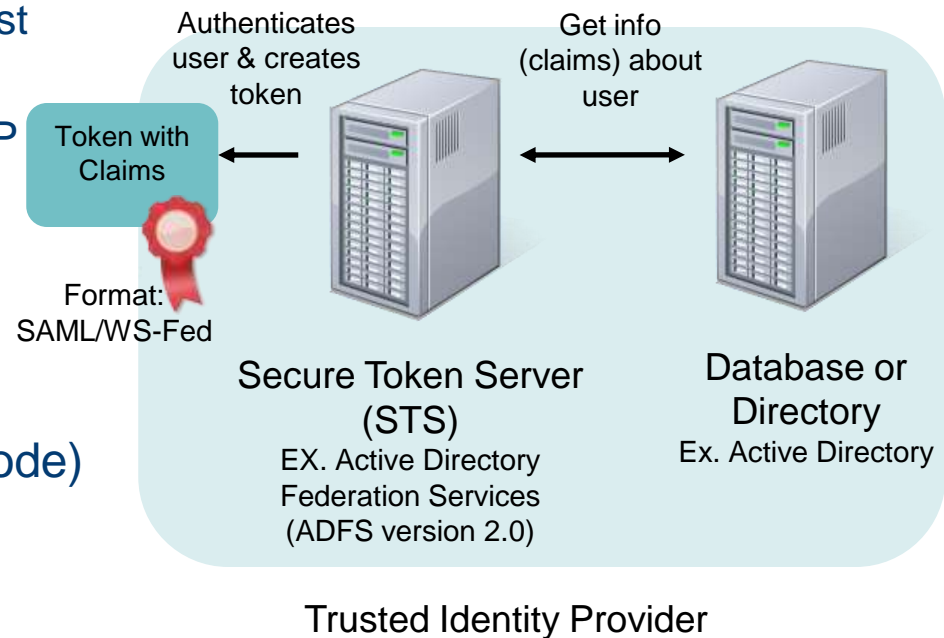
- **Infrastructure and Configuration Required**
 - Storing, managing, retrieving, transforming, trusting claims
 - Configuring SharePoint and Trusted Identity provider
- **Planning Required**
 - Policies to enforce, Claims to use, Metadata to use
 - Getting stakeholders to agree
- **Assign Permissions to Resources based on Claims**
 - Manual Permissions Assignment
 - Third Party Applications to automate permissions and keep up to date

Claims Based Security Architecture



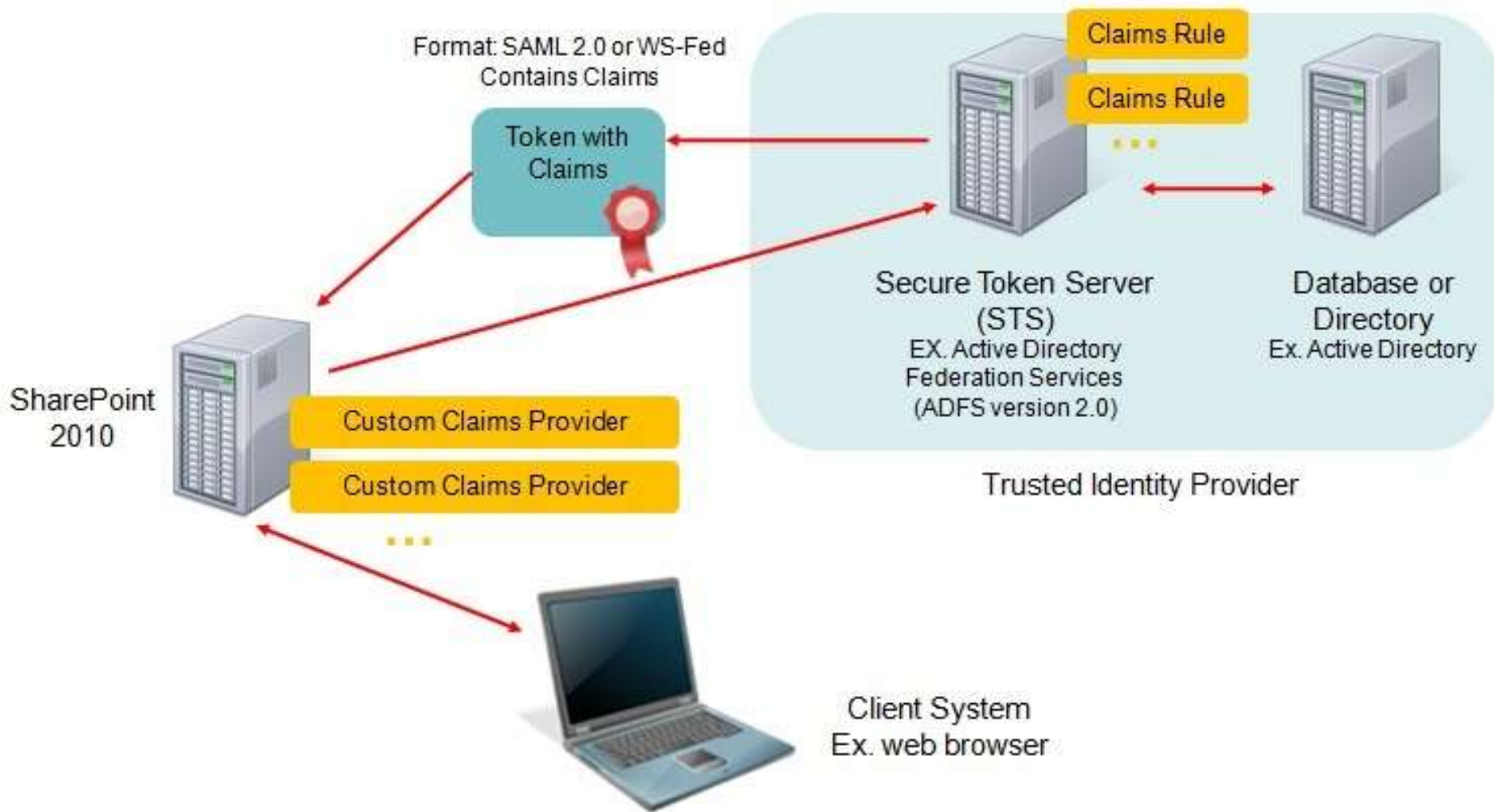
Trusted Identity Providers

- Standards-Based Trusted Identity Provider
 - SAML (SharePoint 2010 supports SAML 1.1 tokens and SAML 2.0 protocol)
 - WS-Federation
- ADFS 2.0 - Consider differences: Identity Store vs Attribute Store
 - Active Directory must be the Identity store – users must be authenticated against Active Directory
 - Active Directory and other SQL/LDAP repositories can be attribute stores
- Consider Claims Augmentation
 - Thru Claims Rules (ADFS2)
 - Thru Custom Claim Providers(code)



Claims Based Security Architecture

Options for Claims Augmentation



Custom Claim Provider

- SharePoint Feature written through code
- Can have multiple deployed
- Custom claim providers called after User Authentication
 - ...and after Trusted Identity Provider may have already returned claims
 - Can manipulate claims returned from a trusted identity provider
 - Can augment claims

Building a Custom Claim Provider

1. Add necessary References

```
Microsoft.SharePoint  
Microsoft.IdentityModel
```

- Browse to find it in \Program Files\Reference Assemblies\Microsoft\Windows Identity Foundation\v3.5\Microsoft.IdentityModel.dll

2. Add necessary Using statements

```
using Microsoft.SharePoint;  
using Microsoft.SharePoint.Administration;  
using Microsoft.SharePoint.Administration.Claims;  
using Microsoft.SharePoint.WebControls;
```

3. Add a Class which inherits from SPClaimProvider

```
namespace SampleClaimProvider  
{  
    public class ClearanceClaimProvider : SPClaimProvider  
    {  
        public ClearanceClaimProvider (string displayName)  
            : base(displayName)  
        {  
        }  
    }  
}
```

Building a Custom Claim Provider

4. Implement the Abstract class

(Right click on base class (SPClaimProvider) and select Implement Abstract Class)

Methods:

FillClaimTypes

FillClaimValueTypes

FillClaimsForEntity

FillEntityType

FillHierarchy

FillResolve(2 overrides)

FillSchema

FillSearch

Properties:

Name

SupportEntityInformation

SupportHierarchy

SupportResolve

SupportSearch

Building a Custom Claim Provider

5. Implement Required Properties

```
public override string Name  
    {get { return ProviderInternalName; }}
```



Returns the Claim Provider unique name

```
public override bool SupportsEntityInformation  
    {get { return true; }}
```



Must return True for Claims Augmentation

```
public override bool SupportsHierarchy  
    {get { return true; }}
```



Supports hierarchy display in people picker

```
public override bool SupportsResolve  
    {get { return true; }}
```



Supports resolving claim values

```
public override bool SupportsSearch  
    {get { return true; }}
```



Supports search operation

Building a Custom Claim Provider

6. Create Static Properties for Name

```
internal static string ProviderDisplayName
{
    get { return "Security Clearance"; }
}
```

```
internal static string ProviderInternalName
{
    get { return "SecurityClearanceProvider"; }
}
```

Building a Custom Claim Provider

7. Create Data Source and Helper Functions

```
private string[] SecurityLevels = new string[]  
    { "None", "Confidential", "Secret", "Top Secret" };
```

Retrieve security clearances from an Array – sample only.
Typically retrieve from database or LDAP.

```
private static string ClearanceClaimType  
{  
    get { return "http://schemas.sample.local/clearance"; }  
}  
  
private static string ClearanceClaimValueType  
{  
    get { return Microsoft.IdentityModel.Claims.ClaimValueTypes.String; }  
}
```

- Adding a claim with type URL `http://schemas.sample.local/clearance` and the claim's value is a string

Building a Custom Claim Provider

7. Implement Methods to Augment Claims

FillClaimTypes

FillClaimValueTypes

FillClaimsForEntity

```
protected override void FillClaimTypes(List<string> claimTypes)
{
    if (claimTypes == null)
        throw new ArgumentNullException("claimTypes");

    claimTypes.Add(ClearanceClaimType);
}
```

```
protected override void FillClaimValueTypes(List<string> claimValueTypes)
{
    if (claimValueTypes == null)
        throw new ArgumentNullException("claimValueTypes");

    claimValueTypes.Add(ClearanceClaimValueType);
}
```

Building a Custom Claim Provider

7. Implement Methods to Augment Claims

```
protected override void FillClaimsForEntity(Uri context, SPClaim entity,
    List<SPClaim> claims)
{
    if (entity == null)
        throw new ArgumentNullException("entity");
    if (claims == null)
        throw new ArgumentNullException("claims");
    if (String.IsNullOrEmpty(entity.Value))
        throw new ArgumentException("Argument null or empty",
            "entity.Value");

    //if existing Clearance claim is 'top secret' then add lower levels clearances
    if (. . .)
    {
        claims.Add(CreateClaim(ClearanceClaimType, SecurityLevels[0],
            ClearanceClaimValueType));

        claims.Add(CreateClaim(ClearanceClaimType, SecurityLevels[1],
            ClearanceClaimValueType));

        claims.Add(CreateClaim(ClearanceClaimType, SecurityLevels[2],
            ClearanceClaimValueType));
    }
    . . .
}
```

Building a Custom Claim Provider

Some Considerations

- Typically - reach out to SQL database, LDAP, Repository for attributes which will get added as claims
- Custom Claim Provider running in the context of the web application, and not the site the user is logging into
 - Logged in as the Central Admin Service Account
 - Do not have context - Most methods have no HTTP Context nor SPContext.Current
 - Cannot directly access data on the Site you authenticated to
 - Retrieve Site list data - must use SharePoint REST interface
- For Debugging use a Claims Testing Web Part in SharePoint:
<http://blogs.technet.com/b/speschka/archive/2010/02/13/figuring-out-what-claims-you-have-in-sharepoint-2010.aspx>

Will display all of the logged in user's claims in a data grid.

Building a Custom Claim Provider

7. Other Important Methods – Replacing the People Picker

`FillEntityType`

Set of possible claims to display in the people picker

`FillHierarchy`

Determine hierarchy for displaying claims in the people picker

`FillResolve` (2 overrides)

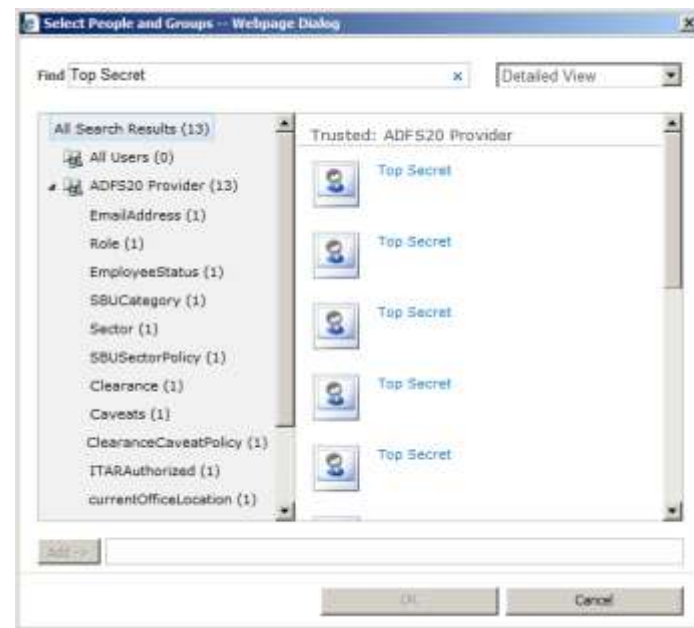
Logic for resolving claims specified in the people picker

`FillSchema`

Specifies the schema that is used by people picker to display claims/entity data

`FillSearch`

Fills in search results in people picker window



Deploying Custom Claim Provider

8. Deployment and Considerations

- Deployed as a Farm Level Feature Receiver – requires more code
 - Must inherit from SPClaimProviderFeatureReceiver (lots of examples)
- Can deploy multiple claim providers – called in order of deployment
- Once deployed - available in every web app, in every zone
 - When user logs in, all Custom Claim Providers deployed to farm get called
 - Set **IsUsedByDefault** property in Feature Receiver Def'n to **False**; then turn it on manually for required web apps
- If using FAST Search Server 2010 for SharePoint
 - Custom claim type mappings must be registered – for security trimming
 - Any custom claims provider used in SharePoint must also be deployed in the FAST Search Server

Summary

- Authentication & Authorization - different but both important
 - Can use Claims today for Authentication in SharePoint 2010
- Claims are great tool for Enterprise-Grade Authorization to
 - Strengthen your Data Governance Strategies
 - Can do manually today in SharePoint 2010
 - Consider Trusted Identity Providers and Custom Claim Providers
- Infrastructure and Planning Required
- Plan policies with business stakeholders – Keep Simple to Start!
- Connect if you have any questions:
 - Antonio Maio – antonio.maio@titus.com
 - SharePoint blog: www.titus.com/blog
 - Twitter: AntonioMaio2