Understanding Kerberos Constrained Delegation for Azure Active Directory Application Proxy Deployments with Integrated Windows Authentication

September 2015

Copyright

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

The descriptions of other companies’ products in this document, if any, are provided only as a convenience to you. Any such references should not be considered an endorsement or support by Microsoft. Microsoft cannot guarantee their accuracy, and the products may change over time. Also, the descriptions are intended as brief highlights to aid understanding, rather than as thorough coverage. For authoritative descriptions of these products, please consult their respective manufacturers.

© 2015 Microsoft Corporation. All rights reserved. Any use or distribution of these materials without express authorization of Microsoft Corp. is strictly prohibited.

Microsoft, Azure, Active Directory, Office 365, SharePoint, Windows, Windows Intune, Windows PowerShell, Windows Server, and Xbox Live are either registered trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Contents

[1. User scenario 4](#_Toc430512689)

[2. Kerberos Constrained Delegation Overview 4](#_Toc430512690)

[3. Kerberos Constrained Delegation (KCD) Core Concepts 5](#_Toc430512691)

[3.1. What is an Integrated Windows Application (IWA) 6](#_Toc430512692)

[3.2. Kerberos 6](#_Toc430512693)

[3.2.1. Protocol Transitioning 6](#_Toc430512694)

[3.3. Service Principal Names (SPN) 7](#_Toc430512695)

[3.4. Three types of Kerberos Delegation: 7](#_Toc430512696)

[3.4.1. Kerberos Delegation 7](#_Toc430512697)

[3.4.2. Kerberos Constrained Delegation (KCD) 8](#_Toc430512698)

[3.4.3. Resource Based Constrained Delegation 9](#_Toc430512699)

[3.5. Front and Back End Servers 11](#_Toc430512700)

[3.5.1. Azure Active Directory Application Proxy 11](#_Toc430512701)

[3.5.2. Windows Server Web Application Proxy 11](#_Toc430512702)

[4. Supported Topologies 12](#_Toc430512703)

[4.1. Forest and Domain Considerations 12](#_Toc430512704)

[5. Example of Configuring KCD End to End 15](#_Toc430512705)

[5.1. Configure Back End Server Authentication 15](#_Toc430512706)

[5.2. Register Service Principal Name (SPN) 16](#_Toc430512707)

[5.3. Front End Configuration | Trust Computer Account for Delegation 16](#_Toc430512708)

[5.3.1. Azure AD App Proxy Requirements and Setup 16](#_Toc430512709)

[5.3.2. Key things to know about SPNs 17](#_Toc430512710)

[5.3.3. Windows Server Web Application Proxy Requirements and Setup and Configuration 19](#_Toc430512711)

[5.4. KCD Configuration Checklist 22](#_Toc430512712)

[5.4.1. Additional Things to watch out for… 23](#_Toc430512713)

[6. Troubleshooting 24](#_Toc430512714)

[6.1. Troubleshooting Helpdesk Checklist 24](#_Toc430512715)

[6.2. Network Trace 25](#_Toc430512716)

[6.3. Using a Website name different than the Hostname 25](#_Toc430512717)

[6.4. Resource-based constrained delegation failures 26](#_Toc430512718)

[6.5. Event Log Errors 27](#_Toc430512719)

[7. Conclusion 30](#_Toc430512720)

# User scenario

In the past access to applications was simpler, people worked mostly in their organization premises and have access to all their resource with single identity to all corpnet resources. The next step in this evolution more and more organization needed to find a solution for their employees and to allow them access from home. Today, most organization need access to corpnet data and applications from any device, any platform, almost anytime from anywhere with a single identity.

This business needs increases dramatically the challenge of organization IT as they need to maintain remote access infrastructure. Adding to that the security modern challenges which requires the IT to have multi-factor authentication (MFA), conditional pre authentication and reports – the efforts to maintain this process become bigger and bigger.

Azure AD Application Proxy goal is providing easy, secure and reliable service and allowing easy publishing of organization on-prem applications to users outside the corporate network and allowing your employees to have SSO access to applications in the organization corpnet.

This SSO is achieved using Kerberos Constrained Delegation (KCD).

The benefits of using KCD for SSO in these scenario are:

1. Cross platform and cross devices access to corpnet on premise resources.
2. The IT gets the benefits from AD Service such as MFA, Reports, Audit, Security Reports and more
3. Single Sign-on experience from Azure Active Directory to on-Prem applications
4. No need to change the backend applications
5. No need to install agents on backend applications
6. No need to expose on-Prem apps directly to the Internet

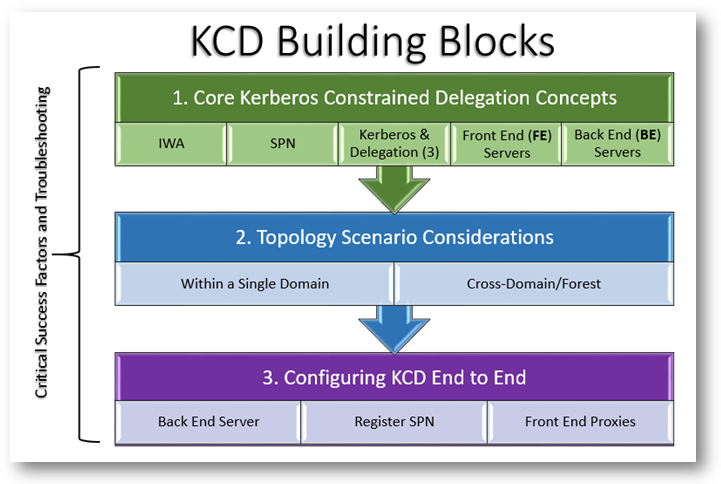
More information on Azure AD Application Proxy can be found in section ‎3.5.1.

While the above scenario is pretty straight forward to configure, KCD can also be complex – especially when your on-Prem infrastructure is complicated. In order clarify the process and help you troubleshoot the complex scenarios

In this document we will try to cover this process end to end –from terminology and implication scenarios as well as common troubleshooting tips

# Kerberos Constrained Delegation Overview

Configuring Kerberos Constrained Delegation (KCD) with Integrated Windows Authentication applications, can be easy to setup and configure. But, it can also be easy to not do it right. The key to successfully deploying IWA based applications is understanding the required building blocks that must be in place. These three core building blocks are included below in the following diagram that will be the basis for how this paper will break it all down to both configure it properly and to also troubleshoot and diagnose issues later down the road if needed.



We will break down these building blocks in each section below. Once you understand the building blocks of Kerberos Constrained Delegation (KCD) and the topology scenario that applies to you, then configuring KCD end-to-end will become easy for you to implement with your IWA applications.

It is important to note that the first two building blocks above are independent of using Azure AD Application Proxy or Windows Server Web Application Proxy. Once the KCD elements are all configured properly for the front end servers, then third building block of “Configuring Back End Application Services” is where you will specifically configure either Azure Active Directory Application Proxy or Windows Server Web Application Proxy. If you do the first two building blocks correctly, then the last part becomes easy. This paper will help to not only help with all three building blocks, but it also will provide some of the exceptions to the rules, challenges and troubleshooting too.

# Kerberos Constrained Delegation (KCD) Core Concepts

When there are issues or problems authenticating with IWA applications through Azure AD Application Proxy or Windows Server Web Application Proxy, it is very likely that there is some element KCD configuration at the root cause. While there is some configuration required for Azure Active Directory Application Proxy (Azure AD App Proxy) and Windows Server Web Application Proxy, the KCD elements should be validated both during the configuration and also later, if needed, for troubleshooting. Understanding the core components of KCD with our IWA scenario will help you get it right the first time. If it is not, we will cover some basic troubleshooting steps below to get you back on track.

Before diving into the nuts and bolts of each of the various components to be discussed, let briefly summarize what must fundamentally happen for KCD to be successful:

1. The client authenticates with their authentication provider e.g. Windows Server Active Directory, AD FS or Azure Active Directory
2. Next, the client request is sent to Azure AD App Proxy or Windows Server Web Application Proxy with an authentication token
3. The token is validated and the UPN is extracted
4. The Service retrieves the ticket for the published web server using the configured SPN using Kerberos Constrained Delegation
5. The request is sent to published web server with the Kerberos ticket added

## What is an Integrated Windows Application (IWA)

Microsoft Integrated Windows Authentication supports multiple negotiated authentication mechanisms. These include: SPNEGO(Simple and Protected GSS-API Negotiation authentication mechanism), Kerberos and NTLM. In the case of KCD, we use Negotiate (Kerberos). If your application is claims-based authentication, then it does not need or use KCD. The focus of this paper is on Integrated Windows applications that use Kerberos for authentication.

Examples of IWA applications would include:

* SharePoint Server
* SQL Server Reporting Services
* Outlook Web Access
* Apache
* SAP
* Microsoft Dynamics CRM
* Multi-tiered Web Applications

A Kerberos discussion is outside the focus of this paper. But you can read more about [Kerberos](https://msdn.microsoft.com/en-us/library/windows/desktop/aa378747(v=vs.85).aspx) on TechNet.

## Kerberos

To understand KCD, you just need to break down the “constrained delegation” part, plus understand another key aspect of it, which is protocol transition. But first, we’ll start with a short overview about Kerberos, which has been the authentication protocol used since Windows Server 2000 Active Directory was introduced. Then, we will discuss Protocol Transition, which was an extension to the Kerberos protocol in Windows Server 2003. Once those foundations are laid, then we can focus on Kerberos Constrained Delegation or KCD. It is the “constrained delegation” part which is important to understanding for either configuration or troubleshooting of IWA applications.

Kerberos is an authentication protocol. While the details of Kerberos for this paper are not necessary to describe in depth here, a basic understanding of it is important since it is Kerberos that provides the final authentication to Integrated Windows Applications (IWA). While Kerberos is not very successful in an internet environment, it is well suited to traverse the internal domains and forests.

When specifically looking at Azure Active Directory Application Proxy or Windows Server Web Application Proxy, they both work using claims and tokens for authentication. This provides a challenge of providing a seamless single sign-on experience as we transition from a claims world in-bound through the Proxy, to one or more web applications within one or more trusted domains or forests in a Windows Server Active Directory environment. This is where the magic happens for the Windows Servers that support integrated windows authentication (IWA). In this case, the user is authenticated once, and then they gain access through the proxy all the way into the internal application. This transition of authentication leads us to what is known as protocol transition as described in the next section below.

Much about Kerberos has been well documented on TechNet. The two links below can expand your knowledge as far as you would like to take it beyond the scope of this paper.

* [How the Kerberos Version 5 Authentication Protocol Works](https://technet.microsoft.com/en-us/library/cc772815)
* [Kerberos Survival Guide](http://social.technet.microsoft.com/wiki/contents/articles/4209.kerberos-survival-guide.aspx)

### Protocol Transitioning

Protocol transition provides application designers with increased flexibility and security by enabling applications to support different authentication mechanisms at the user authentication tier, and by switching to the Kerberos protocol for security features, such as mutual authentication and constrained delegation, in the subsequent application tiers.

Web applications require firewall friendly authentication at the authentication tiers. Since Kerberos is not firewall friendly as described above, protocol transitioning is needed. Protocol transitioning is the ability of a service to get a Kerberos service ticket for delegation in the name of the original client, no matter which authentication method the client originally use for authentication. Therefore, if any authentication protocol other than Kerberos was used, the originating authentication protocol can be transitioned to Kerberos. Non-Kerberos authentication protocol examples could be:

* NTLM
* Basic
* Digest
* SSL Client Certificate
* Forms-based Authentication
* Claims Based Authentication
* Custom Authentication

The protocol transition extension to Kerberos has been done since ever since the Kerberos extensions were introduced in Windows Server 2003. Other than the minimum server version of 2003, the only other requirement in a single domain is that the any participating forests must be at Windows Server 2003 Forest functional level.

##### Additional Resource Links

* [Kerberos Protocol Transition and Constrained Delegation](http://www.microsoft.com/en-us/download/details.aspx?id=20218)

## Service Principal Names (SPN)

Security Principal Names are account attributes that are stored in the Active Directory that register a service name, host and port against a machine or service account. Think of an SPN like a handle that is linked to real entities; such as machines in our examples. This is a very important part of configuring KCD. If the SPN is not configured correctly, Kerberos will fail. Likewise, if Kerberos fails, then KCD will fail. This SPN configuration is one of the first things that should be checked whenever KCD fails. Every service that uses Kerberos authentication needs to have a SPN set for it so that clients can identify the service on the network. If a SPN is not set for a service, then clients will have no way of locating that service. Without properly set SPNs, Kerberos authentication is not possible.

## Three types of Kerberos Delegation:

Delegation is a frequently desired security feature for n-tier Web applications. Without it, the user’s authentication credentials would have to be passed all the way from the client to the server. This could result in a security breach such as identity spoofing. The Kerberos protocol extensions introduced in Windows Server 2003 were adopted to help address delegation issues that Web applications encounter.

Now that the basics of Kerberos have been explained, it is worth discussing the difference between Kerberos Delegation versus Kerberos Constrained Delegation (KCD). If KCD is not configured properly, you just may end up with only Kerberos Delegation. From a security perspective, this is not a preferred option as we should always practice the concept of “least privilege” versus granting unlimited access.

### Kerberos Delegation

Kerberos Delegation is a feature that allows an application to reuse the end-user credentials to access resources hosted on a different server. You should only allow that if you really trust the application server, otherwise the application may use your credentials for purposes that you didn't think of, like sending e-mails on your behalf or changing data in a mission critical application pretending that you made that change.

The first implementation of Kerberos in Windows Server 2000 was an all or nothing implementation. Once that was enabled, it allowed the domain account to use your credentials to connect to any server within the domain, forest or any domain in a forest with a forest trust. This is not constrained at all. You just clicked a checkbox on the computer account for “Trust computer for delegation” or “Account is trusted for delegation” on the user’s object Account tab in Active Directory users and computers.

**Note:** *this only applies where the client to the front-end server authentication uses Kerberos.*

Services that are enabled for Kerberos authentication can delegate identity multiple times. As an identity travels from service to service, the delegation method can change from basic Kerberos to Kerberos constrained. However, the reverse is not possible. The delegation method cannot change from Kerberos constrained to basic Kerberos. Therefore, it is important to anticipate and plan for whether a back-end service will require basic Kerberos delegation. This can affect the planning and design of domain boundaries.

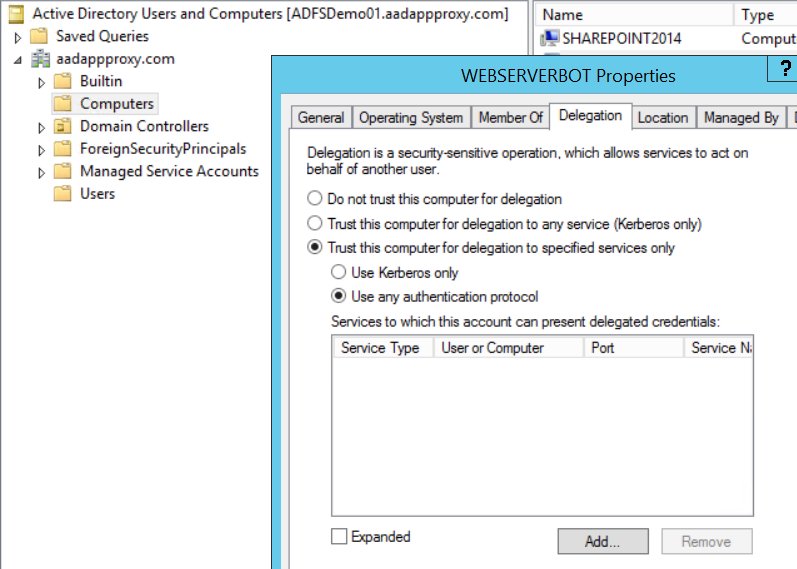
### Kerberos Constrained Delegation (KCD)

Kerberos Constrained Delegation is a Windows extension to the MIT-created authentication protocol. The way the works, is the same as it has been since Windows Server 2003. Kerberos Constrained Delegation allows administrators to restrict which services an account is trusted to delegate to. This is better than normal Kerberos delegation that enables the account to delegate to ANY service.

To achieve this, we need to enable Kerberos delegation. To enable Kerberos delegation, we need to configure the service account of the application that will receive the first authentication from the client so it is able to delegate the client’s credentials to a back end server.

When looking at the delegation tab of a typical web server in Active Directory Users and Computers, many of the scenarios described above can all be illustrated by simply looking at the radio button options.

**Note:** *the scenario below is a case where the web service is running under a machine account context as opposed to a service account.*



On the left side of the table below, if you select that option in the tab shown above, then the right side equals what type of delegation you get.

|  |  |
| --- | --- |
| Do not trust this computer for delegation | *As stated, no delegation here!* |
| Trust this computer for delegation to any service (Kerberos only) | *This is the pre-windows 2003 scenario described for Kerberos Delegation only i.e. it is NOT constrained.* |
| Trust this computer for delegation to specified services only | |
| * Use Kerberos only | *While “Use Kerberos only” is constrained delegation, it does NOT provide protocol transitioning as described above. Therefore, if any other authentication method happens other than Kerberos, this will fail as the protocol will not be transitioned to Kerberos. Selecting this is another common error made.* |
| * Use any authentication protocol | *This is the option to select for KCD that will enable protocol transitioning for other authentication protocols other than Kerberos.* |

**Note:** *When using constrained delegation, confirm that the appropriate SPN is in the Services to which this account can present delegated credentials list box. This should be the SPN for the web application that is registered to the service account it is running under (in the case of the ASP.NET scenario this would be the account under which ASP.NET runs). See* [*SetSPN above*](#SPN) *for syntax examples and resources*

#### KCD Functional Requirements

The rules for KCD must be followed:

1. The front end and backend servers must be in the same domain
   1. This is “traditional” KCD, where the domain administrator defines delegation permissions within their domain.
   2. For cross domain or cross forest scenarios, we will discuss those further below
2. The accounts of users accessing the services do not have to be in the same domain as the services.
   1. Both forests MUST be operating at the Windows Server 2003 forest functional level or higher
   2. A two way transitive trust must be established between the forests
   3. This is relevant for where the user sits in a trusted forest. See [Resource Based Constrained Delegation](#RBCD) for more information below.
3. The front end needs to be configured with the services (by SPN) to which it can delegate authentication. This will require domain credentials to make this happen.

Delegation is a security pattern that occurs frequently in n-tier applications. Constrained delegation gives administrators the ability to specify and enforce application trust boundaries by limiting the scope where application services can act on a user’s behalf. This flexibility to constrain a service’s authorization rights, helps improve application security design by reducing the opportunities for compromise by untrusted services.

### Resource Based Constrained Delegation

Resource Based Constrained Delegation (RBCD) was introduced in Windows Server 2012. In all that has been described thus far, one could think that the evolution to KCD solves the original security issue with the original implementation of the Kerberos protocol. However, there has still existed another limitation that was not previously overcome before. Until Windows Server 2012 added Resource Based Constrained Delegation, KCD only worked across domain boundaries if the front end and back end servers were in the same domain.

Using Windows Server 2012 or greater, you can now use KCD across a domain or forest boundary by using Resource Based Constrained Delegation. There are a number of requirements for Resource Based Constrained Delegation.

As the name suggests, Resource Based Constrained Delegation, must be configured on the resource\backend server service or machine account. This can only be configure through PowerShell. The PowerShell cmdlet will also reference the fully qualified domain name of the Domain Controller hosting the impersonating account. Here is an example of how this would be used with an Azure AD App proxy server called aadappsvr in Contoso.com hosting the *besvcacct* service account for a front end web server: Limitations in this scenario will be described in “Forest and Domain Considerations” below.

$connector= Get-ADComputer -Identity aadappsrv -server aadappsrv.contoso.com

Set-ADComputer -Identity besvcacct -PrincipalsAllowedToDelegateToAccount $connector

Get-ADComputer besvcacct -Properties PrincipalsAllowedToDelegateToAccount

**NOTE:** Using the command above will overwrite any existing settings. To simply update and not overwrite, use the –Instance parameter instead of the –Identity parameter. See more examples at [Set-ADComputer](https://technet.microsoft.com/en-us/library/ee617263.aspx).

**Resource Based Constrained Delegation Advantages**

Resource Based Constrained Delegation in Windows Server 2012 moves the authorization decision to the resource-owners, which provides these advantages:

* Permits back-end to authorize which front-end service-accounts can impersonate users against their resources
* Supports cross-domain, cross-forest scenarios
* No longer requires Domain Admin privileges
  + Requires only administrative permission to the back-end service-account

**Resource Based Constrained Delegation Requirements**

* Clients run Windows XP or later
* Client domain’s domain controllers running Windows Server 2003 or later
* Front-end server running Windows Server 2012
* One or more domain controllers in front-end domain running Windows Server 2012
* One or more domain controllers in back-end domain running Windows Server 2012
* Back-end server account configured with the accounts that are permitted for impersonation
  + Not exposed through Active Directory Administrative Center
  + Configured through Windows PowerShell:
    - New/Set-ADComputer [-name] <string> [-PrincipalsAllowedToDelegateToAccount <ADPrincipal[]>]
    - New/Set-ADServiceAccount [-name] <string> [-PrincipalsAllowedToDelegateToAccount <ADPrincipal[]>]
* Windows Server 2012 schema update in back-end server’s forest
* Back-end application server running Windows Server 2003 or later

#### Additional Resource Links

* [What’s New in Kerberos Authentication](https://technet.microsoft.com/en-us/library/hh831747.aspx)
* [What’s New in Active Directory Domain Service](https://technet.microsoft.com/en-us/library/hh831477.aspx) (AD DS)
* [Kerberos Constrained Delegation Overview](https://technet.microsoft.com/en-us/library/jj553400.aspx)
* [Summary (Kerberos Protocol Transition and Constrained Delegation)](https://technet.microsoft.com/en-us/library/cc772683(v=ws.10).aspx)
* [About Kerberos constrained delegation](https://technet.microsoft.com/en-us/library/cc995228.aspx)
  + *While this is older and for TMG, still another good overview*
* [Understanding Kerberos Double Hop](http://blogs.technet.com/b/askds/archive/2008/06/13/understanding-kerberos-double-hop.aspx)
* [KCD with Cross-Forest Accounts](https://technet.microsoft.com/en-us/library/cc752953.aspx) (2008)
* [Kerberos for the Busy Admin](http://blogs.technet.com/b/askds/archive/2008/03/06/kerberos-for-the-busy-admin.aspx)
* [Windows Server 2003 Kerberos Extensions](https://technet.microsoft.com/en-us/library/cc738207(v=ws.10).aspx)
* [Remote Administration Without Constrained Delegation Using PrincipalsAllowedToDelegateToAccount](http://blogs.msdn.com/b/taylorb/archive/2012/11/06/remote-administration-without-constrained-delegation-using-principalsallowedtodelegatetoaccount.aspx)
* For more information on “Resource-based constrained delegation across domains and forest”, see the same section in [What's New in Kerberos Authentication](https://technet.microsoft.com/en-us/library/hh831747.aspx)

## Front and Back End Servers

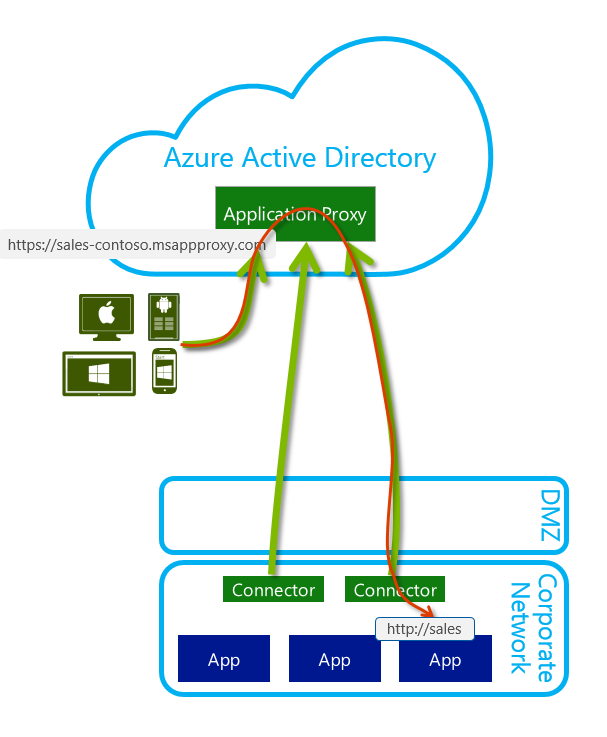
The Front End Servers will either be Azure AD App Proxy or Windows Server Web Application Proxy. In either case, they will proxy the internal IWA application request from a user external to the corporation. Functionally the way these two work will be the same in the sense of acting as a Reverse Proxy.

The Back End Server will be the application server(s) that will be using integrated windows authentication (IWA). Examples of IWA applications were described above, but a couple of common examples could be web servers or a SharePoint farm.

Let’s define Azure AD App Proxy and Windows Server Web Application Proxy in case you are not very familiar with each them already.

### Azure Active Directory Application Proxy

Azure AD Application Proxy (Azure AD App Proxy) lets you publish applications, such as SharePoint sites, Outlook Web Access and IIS-based apps, inside your private network and provides secure access to users outside your network. Employees can log into your apps from home, on their own devices and authenticate through this cloud-based proxy. Application Proxy is a feature that is available only if you upgrade to the Premium or Basic editions of Azure Active Directory. For more information, see [Azure Active Directory Editions](https://msdn.microsoft.com/en-us/library/azure/dn532272).

Azure AD App Proxy now does the job that Windows Server Web Application Proxy did on premises. The main difference is that Azure AD App proxy works from within Azure AD and Windows Server Web Application Proxy works in conjunction with AD FS on-premises. It works like this:

1. You need either Azure Active Directory Basic or an Azure Active Directory Premium license, and then [enable Azure AD App Proxy](https://msdn.microsoft.com/en-us/library/azure/dn768214.aspx) in Azure AD
2. Deploy connectors on-premises on any box running Windows Server 2012 R2, Windows 8.1 or higher
   1. Multiple connects can be deployed for redundancy and scale
3. The traffic goes to the cloud and is routed to the internal server via the on-premises connectors

Azure AD App Proxy thus gains all of the many cloud advantages over Windows Server Web Application Proxy such as:

* Terminating all HTTP/S traffic in the cloud, which blocks most HTTP level attacks on-premises
* Unauthenticated traffic is filtered in the cloud and thus will not arrive on-premises
* There are no incoming connections to your corporate network
* This cloud service is always patched and updated so you don’t have to do or think about this
* Azure AD provides reporting on login abnormalities, and additional reporting and auditing capabilities
* [https://myapps.microsoft.com](https://myapps.microsoft.com/) - Any Azure AD App Proxy applications that you publish, will appear in the MyApps portal.

#### Resource Links

* [Enable Your On-Premises Apps for the Cloud with Microsoft Azure Active Directory Application Proxy](https://channel9.msdn.com/Events/Ignite/2015/BRK3864) – Ignite 2015
* [Using Application Proxy to publish applications for secure remote access](https://msdn.microsoft.com/library/azure/dn768219)
* [Enable Application Proxy services](https://msdn.microsoft.com/en-us/library/azure/dn768214.aspx) – has step-by-step walkthroughs
* [SSO for On Prem IWA Apps Using KCD with Application Proxy](https://msdn.microsoft.com/en-us/library/azure/dn879065.aspx)
* [Application Proxy Blog](http://blogs.technet.com/b/applicationproxyblog/)
  + This blog provides additional information about the Application Proxy service of Windows Server and Azure Active Directory.

### Windows Server Web Application Proxy

Windows Server Web Application Proxy is a new role service that started with Windows Server 2012 R2. Previous reverse proxy solutions of the past were separate add-ons and installations on top of an existing operating system. This added cost and complexity, in addition to the standard server maintenance in place. When Microsoft planned for an integrated reverse proxy and publishing service for Windows Server, they look at the top scenarios that most organizations often use. If features were not used or needed, then they become deprecated. The result is a seamless and fully integrated solution that is easier to install, configure and maintain. Since it is already part of the operating system, its updates are part of the Windows update process.

Another key service that Windows Server Web Application Proxy provides, in addition to publishing internal applications securely to external users, is the Active Directory Federation Services (AD FS) 2012 R2 Proxy role. The AD FS Proxy service was previously a separate role on a separate machine designed to protect the internal AD FS server, which contains digital signing certificates. Likewise, the Web Application Proxy server must also be a separate machine from the AD FS server. The Web Application Proxy server combines the Web Application Proxy and AD FS Proxy services on the same box. The net result is to proxy the AD FS endpoints and also the published applications. Published applications will either have pass-through or pre-authenticated access. In the discussions that follow, it can be understood that any such applications have been published with pre-authentication, which would be required for IWA scenarios.

#### Resource Links

* [Windows Server Web Application Proxy](https://technet.microsoft.com/en-us/library/dn584107.aspx) on TechNet
* [Application Proxy Blog](http://aka.ms/appproxyblog)
* [AD FS and Windows Server Web Application Proxy Links](http://aka.ms/adfslinks)
* [Deploy the Windows Server Web Application Proxy](https://technet.microsoft.com/en-us/library/dn268288.aspx)
* [Windows Server Web Application Proxy PS Cmdlets](https://technet.microsoft.com/library/dn283404%28v=wps.630%29.aspx)

# Supported Topologies

There are a number of topology variations that you may run into while configuring Resource Based Constrained Delegation. The location of the users, front end servers and backend servers are important pieces of the puzzle to understanding if KCD or RBCD is required.

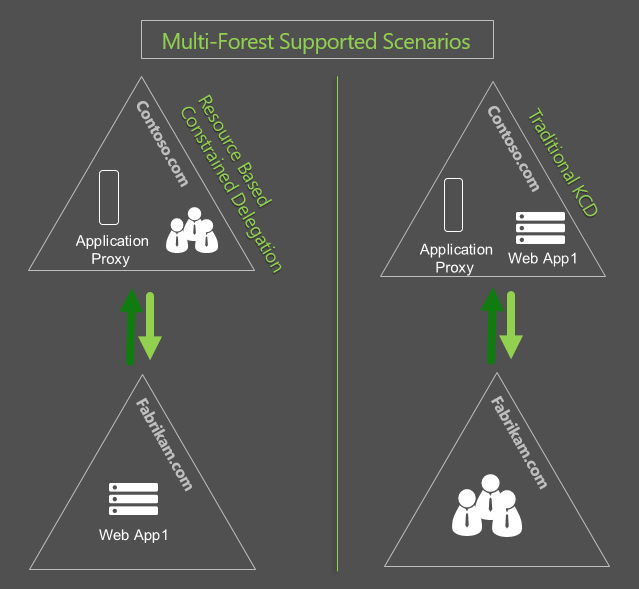
## Forest and Domain Considerations

For simplicity below, “Application Proxy servers” will refer to either:

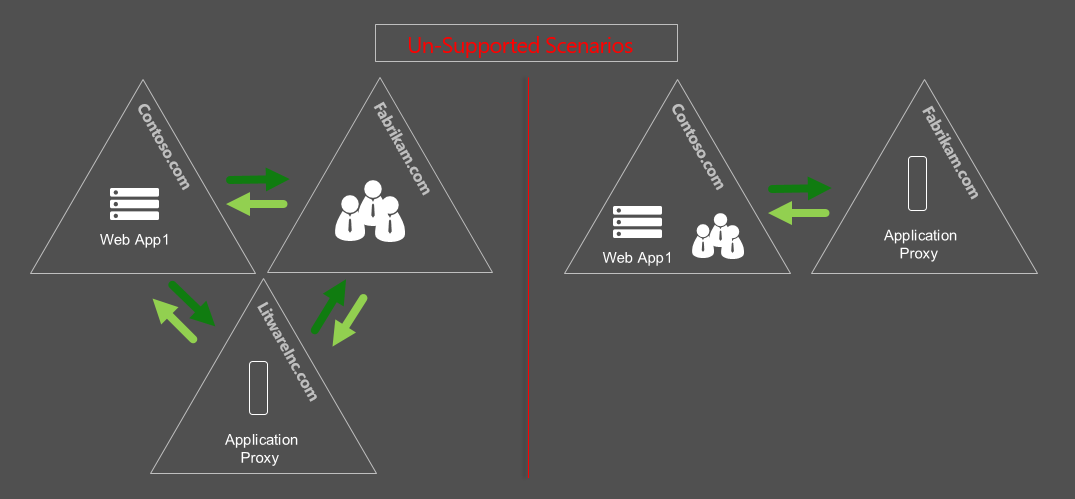
* The server where the Azure Active Directory Application Proxy connector is installed
* The Web Application Proxy Server(s)

The following lists the domain and forest considerations for a deployment using Resource Based Constrained Based Delegation.

1. Deployments where users, resources, and Application Proxy servers are all in the same forest and or domain are functioning as in Traditional KCD scenario.
2. In deployments with *multiple* forests where there is a user forest, a resource forest, and an Windows Server Web Application Proxy forest, the following deployments are supported:
   1. Users and Application Proxy servers are in the same forest, but resources are in a different forest.
   2. Resources and Application Proxy servers are in the same forest, but users are in a different forest. *Traditional KCD*



1. In deployments with multiple forests where there is a user forest, a resource forest, and an Windows Server Web Application Proxy forest, the following deployments will not work:
   1. Users, resources, and Application Proxy servers are all in different forests.
   2. Users and resources are in the same forest and application proxy servers are in a different forest



Where delegation between the front end server and back end server is across a forest boundary, the front end server can only delegate for accounts within its own forest. This is an Active Directory, Directory Services limitation for Kerberos Constrained Delegation and is not Application Proxy specific.

**NOTE:** *Whenever there are cross-forest scenarios our recommendation is to plan for front-end and back-end servers to be in the same forest.*

In multi-forest deployments:

1. The user forest must trust the Application Proxy forest, and the Application Proxy forest must trust the resource forest i.e. a two-way forest trust is required.
2. All of the Active Directory domains in a multi-forest deployment must have at least one Windows Server® 2012 or higher domain controller. For more information, see [Kerberos Constrained Delegation across Domains](http://go.microsoft.com/fwlink/?LinkId=389630)

# Example of Configuring KCD End to End

Now that the KCD concepts and topologies have been covered above in detail, let’s walk through an example of configuring KCD end-to-end using SharePoint Server 2013. There are 3 parts that need to be completed to configure KCD as described in the initial overview.

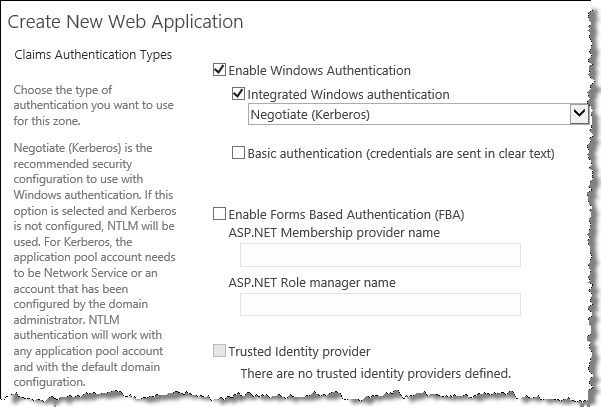


1. Configure Backend Application Server. In this example, it will be SharePoint 2013.
2. Registering the SPN. Now we know the service account we can register the SPN against that account.
3. Configuring Front End Servers’ delegation permissions. References to instructions below will include both Azure AD App Proxy as well as Windows Server Web Application Proxy.

**NOTE:** *A* [*Troubleshooting checklist*](#Checklist) *further down this paper ties these pieces together to give a systematic approach in diagnosing some of the fundamental components that are often overlooked or misconfigured.*

## Configure Back End Server Authentication

Since every back-end application may work differently in the way they are configured, we will not cover specific configuration details, other than highlighting what specific points need to be addressed. The first and foremost point is how the application will authenticate. We discussed earlier that Kerberos authentication is at the heart of KCD. But in the case of IWA applications, they can use either NTLM or Negotiate/Kerberos. KCD required Negotiate. In the example below in creating a new SharePoint application, this is what must be selected as shown below. Therefore, if you have any other application, it likewise must be configured to support Negotiate/Kerberos authentication.



**NOTE:** *in this example of SharePoint, it is not configured for Claims Based Authentication, and hence will use Integrated Windows Authentication and thus require Negotiate/Kerberos to be configured for KCD.*

Here are some other additional scenarios of where Microsoft recommends to set the application to Negotiate: publishing Dynamics and SSRS.

* In the case of Dynamics, some cases have been found where only NTLM was enabled as the authentication provider for Windows Authentication in IIS.
  + The resolution was to change the authentication configuration to add negotiate as an authentication provider
  + This was followed up with an IISReset
* SSRS was essentially the same issue discovered, which returned only NTLM and Kerberos, but not Negotiate
  + The resolution was adding RWWindowsNegotiate to the AuthenticationTypes in the RSReportServer.config
  + Then the report server service was restarted
  + Read more on this at <http://blogs.technet.com/b/rob/archive/2011/11/23/enabling-kerberos-authentication-for-reporting-services.aspx>

## Register Service Principal Name (SPN)

Once the application has been configured to support negotiate/Kerberos authentication, next identity what service account the application is running under. Many applications by default run under local, system or machine account where you cannot register an SPN. Using a domain service account is critical. The only exception to this is with Exchange, which uses an alternate service account (ASA). Read more on the [Exchange Team blog](http://blogs.technet.com/b/exchange/archive/2011/04/15/recommendation-enabling-kerberos-authentication-for-mapi-clients.aspx) on this.

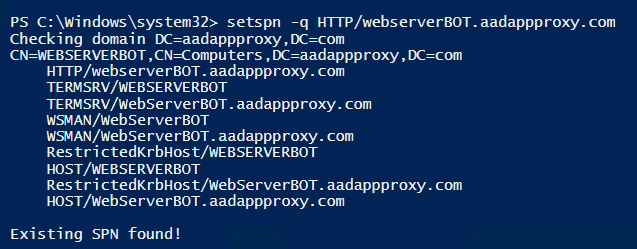
Create the Service Principal Name (SPN) for your application to be published. Some applications may automatically register their SPN, so confirm with **SetSPN –q**.

To set the SPN from an elevated command prompt requires that you use at least Domain Administrator credentials or higher. These rights can also be delegated to the account you use if you are not a domain or enterprise administrator.

setspn -s HTTP/webserverBOT.aadappproxy.com

The advantage of using the “-s” switch above, is that this switch will not add the SPN, if a duplicate one already exists.

To verify it from SetSPN you can run it with the –q switch as shown below. In the picture below, it actually verified that the SPN already existed!



## Front End Configuration | Trust Computer Account for Delegation

Once the SPN exists, then delegation must be configured on the front end server, to impersonate the users to access the published applications on the backend.

* + - * + This will be either on the Azure AD App Proxy connector machine or else on the Web Application Proxy server object.

While we do not want to get into prescriptive guidance for the installation and configuration of Azure AD App Proxy, we’ll summarize the steps from the beginning and provide detailed references. Beyond that, we’ll outline the specifics required for configuring Azure AD App Proxy first, followed by similar references for Windows Server Web Application Proxy.

### Azure AD App Proxy Requirements and Setup

The basic requirements to enable Azure AD App Proxy are well known and documented. Also, the process to publish applications generically is known as well. Here are the two resources to provide 1. Enabling and 2. Publishing with Azure AD App Proxy:

* 1. [Enable Application Proxy services](https://msdn.microsoft.com/library/azure/dn768214.aspx)
  2. [Publish applications with Application Proxy](https://msdn.microsoft.com/library/azure/dn768220.aspx)

The other consideration in an Azure AD environment is the service account properties. The User Principal Name (UPN) that we obtain from Azure AD must be available in the local on-premises domain in order to get a Kerberos ticket. Therefore, for KCD to work properly with Azure AD App Proxy and Azure, there needs to be authentication by either Azure AD password hash synchronization or else federated authentication with AD FS.

Here are some reference articles for more in-depth reading which were used for the summary information above.

* Read the step-by-step procedures and walk through at
  + [Enabling Azure AD Application Proxy](https://azure.microsoft.com/en-us/documentation/articles/active-directory-application-proxy-enable/).
* Further detailed documentation is provided at MSDN with
  + [Using Application Proxy to publish applications for secure remote access](https://msdn.microsoft.com/en-us/library/azure/dn768219.aspx)
* For even more specific information regarding IWA and KCD using Azure AD Application Proxy, see
  + [SSO for On Prem IWA Apps Using KCD with Application Proxy](https://msdn.microsoft.com/en-us/library/azure/dn879065.aspx). This includes step by step procedures for creating the SPN and setting the delegation on the proxy server.

**Note**: *For Azure AD App Proxy, make sure that the server running the Connector and the server running the app you are publishing are domain joined and part of the same domain. If you are using Web Application Proxy for application publishing, then it MUST be domain joined to work.*

### Key things to know about SPNs

1. An SPN must be unique in the forest where it is registered.
   1. Duplicate SPNs are a common point of failure.
   2. If you manually create SPNs, make sure to use **SetSPN –S** and not **SetSPN –A** (or an AD or ADSIEdit graphical editor), because **–S** will check for a duplicate SPN before creating the specified SPN.
   3. A pool of web applications cannot all use a default inbuilt principal, such as network service, the domain controllers could not differentiate the service requests. Therefore, either each application pool needs its own unique domain service account.
      1. Likewise, in a load balanced scenario of multiple application servers, there should be one unique SPN name registered for that farm of servers that are load balanced. An example of this is using one federation service name for a pool of AD FS servers. That one federation service name has just one SPN set for the entire pool of AD FS servers.
      2. An Exchange Client Access server array solves this problem by using an alternate service account (ASA). Read more on the [Exchange Team blog](http://blogs.technet.com/b/exchange/archive/2011/04/15/recommendation-enabling-kerberos-authentication-for-mapi-clients.aspx) on this.
2. Each account can have one or more SPNs associated with it
3. SPN syntax has three parts: <service name> /< host> :< port number>

**Note**: Be aware that the SPN syntax uses a forward slash (/) to separate elements. The < host> in this case refers to the backend server e.g. SharePoint for your applications. In cases where you are using the default ports, for example port 80 for http, then it does not need to be specified.

* 1. <service name> and < host> are required, but port is optional. The colon between host and port is only required when a port is present and not using a default port, such as port 80 for HTTP
  2. For Web applications < service name> will be **HTTP**. Other typical service name examples could be **MSSSQLSvc** and **HOST**.

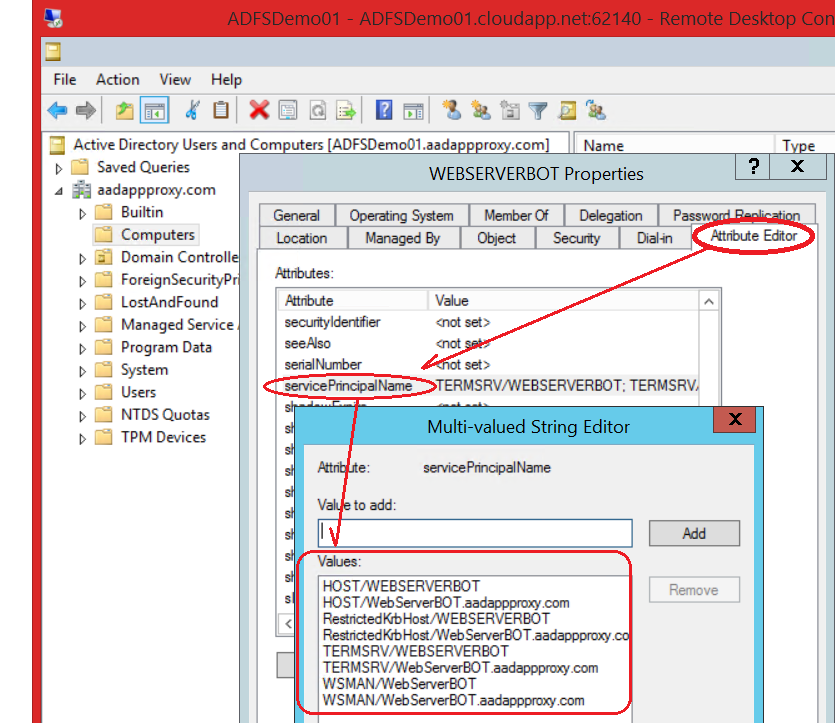
**Note**: Use **HTTP** and not HTTPS

* 1. Examples
     1. **HTTP/www.contoso.com** – Any page on the Web site for http://www.contoso.com.

1. They are stored in Active Directory (AD) in the servicePrincipalName attribute of a service or machine account
   1. There are a few ways to see the servicePrincipalName attributes in Active Directory. Below are two of the more common methods:
      1. One of the easiest ways is to use [**SetSPN**](http://social.technet.microsoft.com/wiki/contents/articles/717.service-principal-names-spns-setspn-syntax-setspn-exe.aspx) **–q**.This switch will check forest-wide.
         1. For example, if you had a SharePoint server sp2013.contoso.com and want to check the SPN registration for the service account used for the SharePoint server, you would type the following…

*Setspn –q http/sp2013.contoso.com*

* + 1. The “Attribute Editor” in AD Users and Computers (**dsa.msc**) in Windows Server 2012 or greater
       1. This only appears on an account object once **Advanced Features** is selected from the **View** menu of **dsa.msc**.
       2. This is a very easy method now to use in Windows Server 2012 and above!
          1. As shown in the picture below with the “Attribute Editor” feature on AD objects. This appears once “Advanced Features” are enabled from the menu of Active Directory Users and Computers …
          2. If you use a graphical editor like this, it is possible to create duplicate SPNs. If this method is used, the follow-up with running **SetSPN –X**, to check for duplicate SPNs.



**Attribute Editor in Windows Server 2012 R2**

If the SPNs are configured properly, this greatly increases your success with KCD as it is one of the core building blocks. If it still is not working, then you would look into configuration of the backend servers (BE) or the front end (FE) application servers.

#### Additional Resource Links

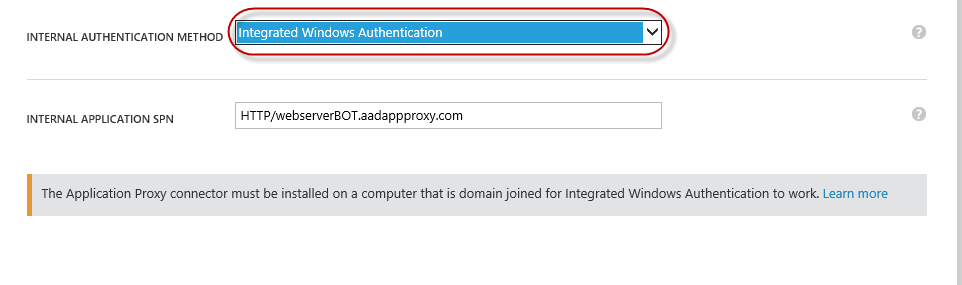
* [Service Publication and Service Principal Names Technical Reference](https://technet.microsoft.com/en-us/library/cc759300(v=ws.10).aspx)
* [Service Principal Names](https://msdn.microsoft.com/en-us/library/ms677949.aspx) overview on MSDN
* [Service Principal Names Syntax](https://technet.microsoft.com/en-us/library/cc961723.aspx)
* [SetSPN Syntax](http://social.technet.microsoft.com/wiki/contents/articles/717.service-principal-names-spns-setspn-syntax-setspn-exe.aspx) Wiki
* [Authentication and Authorization](http://blogs.msdn.com/b/autz_auth_stuff/) Blog | *oldie but goodie!*
* [Interesting Findings on SETSPN](http://blogs.technet.com/b/askds/rss.aspx?Tags=kerberos)
* [Time-out error when you run Setspn.exe in Windows 8 or Windows Server 2012](http://support.microsoft.com/en-gb/kb/2799960/en-us)

#### Setting Integrated Windows Authentication for Azure AD App Proxy

Integrated Windows Authentication is not something that a typical administrator would explicitly be determining for the applications to be published. It may be built into or designed in the application itself. Nonetheless, when configuring Azure AD App Proxy you have to ensure that you not only select the correct configuration when you do have Integrated Windows Applications (IWA), but conversely, do not select those options in the case where the applications were not designed or configured for IWA.

Below is a quick and useful reference to assist with your deployments.

1. The picture below from the Azure Portal shows the selection for the “internal authentication method” for an application configured with Integrated Windows Authentication when configuring Azure AD App Proxy. See the Steps to configure this in [SSO for On Prem IWA Apps Using KCD with Application Proxy](https://msdn.microsoft.com/en-us/library/azure/dn879065.aspx).
2. We also need to set the internal application SPN in accordance with the SPN that we will use as discussed above.



### Windows Server Web Application Proxy Requirements and Setup and Configuration

Here are reference articles to assist in installation and configuration of AD FS and Windows Server Web Application Proxy, which must be in place first, to then complete the remaining part of the end-to-end KCD configuration.

* Windows Server Web Application Proxy will first require a separate instance of Active Directory Federation Services (AD FS).
  + - See [AD FS requirements](https://technet.microsoft.com/en-us/library/dn554247.aspx)
    - See [Windows Server 2012 R2 AD FS Deployment Guide](https://technet.microsoft.com/en-us/library/dn486820.aspx)
* Windows Server Web Application Proxy is available is a remote access server role available only on Windows Server 2012 R2
  + Make sure first that the Web Application Proxy servers are domain joined. This is required for Kerberos Constrained Delegation to work.
  + See [Install and Configure the Windows Server Web Application Proxy Server](https://technet.microsoft.com/en-us/library/dn383662.aspx)
  + Additionally, we have a blog written to [Back-up your Windows Server Web Application Proxy Settings](http://blogs.technet.com/b/applicationproxyblog/archive/2015/03/10/backup-your-web-application-proxy-settings.aspx). This blog also has the installation and configuration of Windows Server Web Application Proxy scripted all together in PowerShell. It also provides PowerShell scripts to capture all the settings into one convenient web page.

Now that the basics above are covered for installing and configuring AD FS and Windows Server Web Application Proxy, you can then walk through the process to complete the end-to-end scenario below.

For a complete sample walkthrough for a proof of concept, see [Configure a site to use Integrated Windows Authentication](https://technet.microsoft.com/en-us/library/dn308246.aspx) which covers the following steps

1. Install a Windows Authentication on a Web Server
2. Create a new website
3. Create a non-claims-aware relying party trust
4. See also [Create a Non-Claims-Aware Relying Party Trust](https://technet.microsoft.com/en-us/library/dn508281.aspx)
5. Configure KCD
6. Test application access
7. Publish the application in Windows Server Web Application Proxy
   1. Screenshots are provided below to further elaborate on publishing with preauthentication options
   2. See also [Publish Applications using AD FS Preauthentication](https://technet.microsoft.com/en-us/library/dn383640.aspx)
8. Test external access to the published application

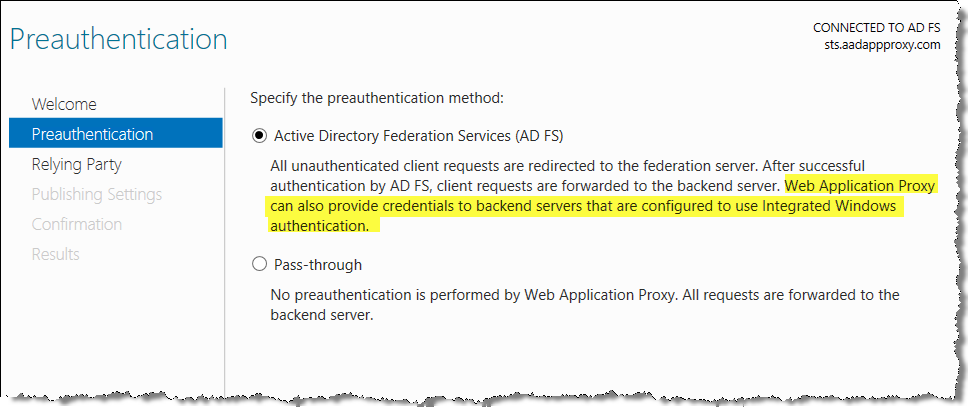
**NOTE:** *Ensure that the* ***Service Principal Name*** *for the non-claims app is mapped to the account that the application pool is running under on the web or application server; which in this case is the web server’s machine account in AD.*

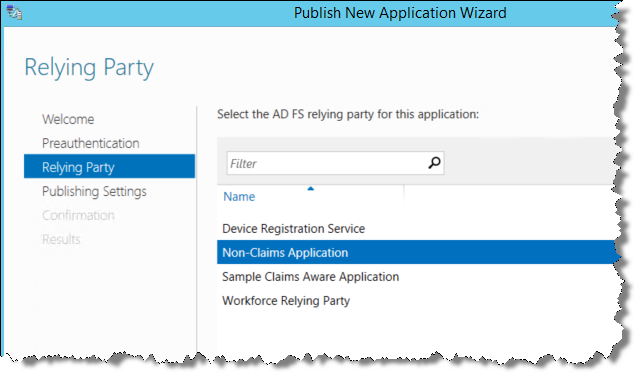
The following sample screenshots below, are based upon steps for publishing in [Publish Applications using AD FS Preauthentication](https://technet.microsoft.com/en-us/library/dn383640.aspx).

When publishing applications with Windows Server Web Application Proxy, the publishing wizard will present two options for Preauthentication:

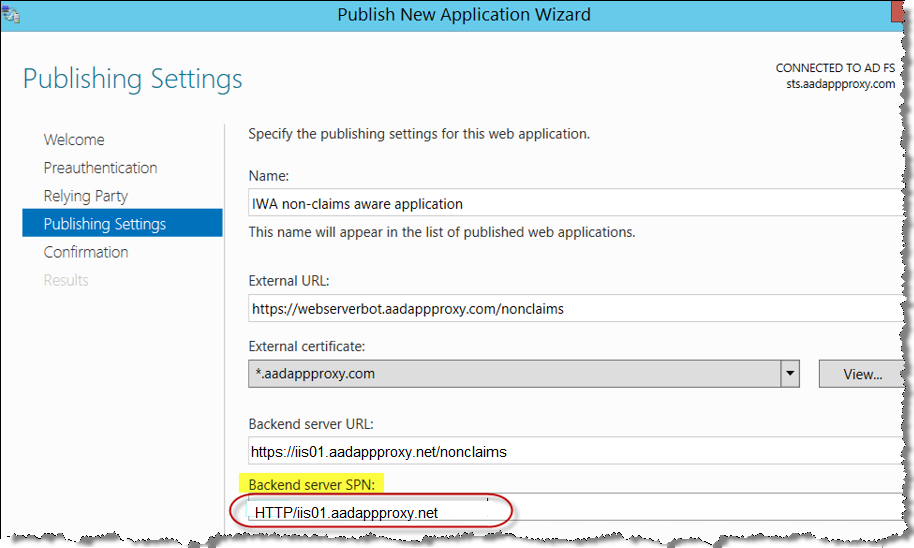
* Active Directory Federation Services (AD FS)
* Pass-through

As shown in the snapshot below, selecting the first option for AD FS, provides the capabilities to IWA applications for performing KCD. This screen is the second window in the interface after the Welcome page where you simply click next to get to this page.



On the **Relying Party** page, you select the non-claims-aware relying party trust that you had created in AD FS.

Next, in the **Publishing Settings** page, you fill in the publishing settings for the web application



Just before you finalize publishing the web application, the **Confirmation** page gives you a summary of all of the PowerShell settings that will be used to configure this web application.

Even though the Windows Server Web Application Proxy publishing wizard is simple and easy to use, if you learn to script these in PowerShell, you can do this all with just one line of PowerShell which will look like this…

Add-WebApplicationProxyApplication -BackendServerUrl https://iis01.aadappproxy.net/nonclaims/ -ExternalCertificateThumbprint 53XXXXXXXXXXXXXXXXXXXXXXXXXEF -ExternalUrl https://webserverbot.aadappproxy.com/nonclaims/ -Name "IWA non-claims aware application" -ADFSRelyingPartyName "Non-Claims Application" -BackendServerAuthenticationSPN HTTP/iis01.aadappproxy.net -ExternalPreauthentication ADFS

**NOTE:** *in the example above, there are two different URLs for the internal and external links. This is possible and works as long as the path trailing the fully qualified domain names matches; in this case /nonclaims/.*

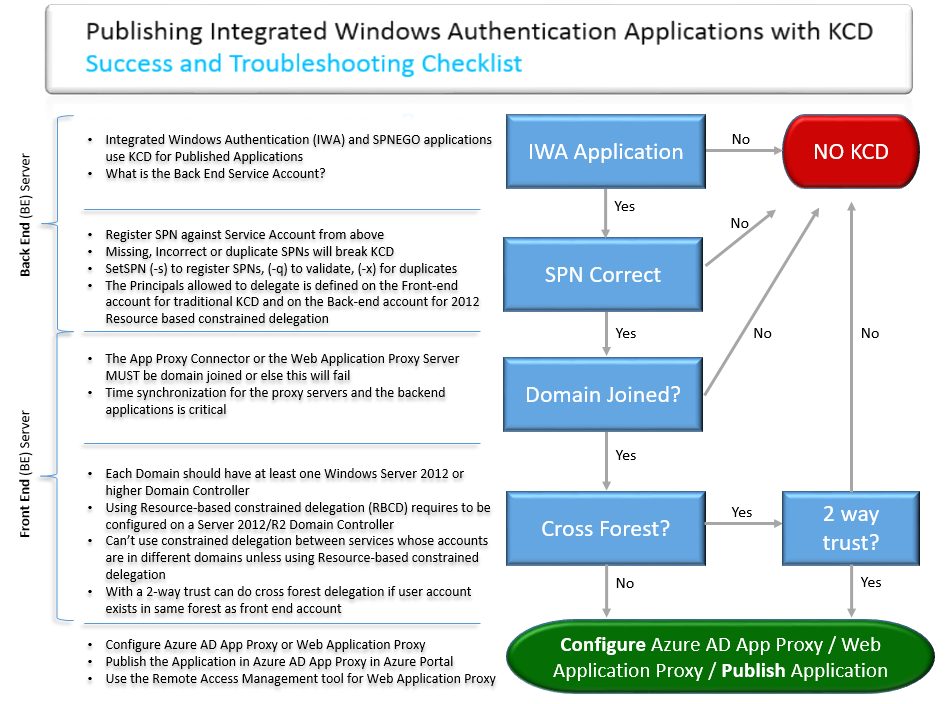
Likewise, once it is published, you can see more of the details from the PowerShell as it was configured, versus the four field view above in the graphical interface. Just run a command like this one below, but specify your published application name:

Get-WebApplicationProxyApplication -Name "IWA non-claims aware application" | fl

Learn more about [Windows Server Web Application Proxy Cmdlets in Windows PowerShell](https://technet.microsoft.com/library/dn283404(v=wps.630).aspx) to script and simplify the configuration and deployment of Windows Server Web Application Proxy.

## KCD Configuration Checklist

Before diving into a few specific issues that have been encountered, let’s summarize the key things to watch out for to be successful in publishing IWA applications with KCD. We have collected many of the configuration points as noted in the sections above into a flow chart to help you. The following diagram shows the multiple checkpoints or gateways that need to be passed to finalize the proper configuration of KCD for IWA applications. If an attempt was already made to publish an IWA application and you are getting KCD failures or related errors, then this chart gives you an overview of the key building blocks of a successful KCD deployment to check and validate. Beyond this checklist, the section following this will get into deeper and more specific issues and their known fixes.



### Additional Things to watch out for…

Apart from the core configuration steps and checklist above, we wanted to highlight a few additional common stumbling blocks customers have encountered.

#### UseAppPoolCredentials

If the IIS app pool is running under a service account, the machine account is still used for ticket decryption when Kernel Mode Auth is enabled. Kernel mode Auth is enabled by default. There is an IIS setting UseAppPoolCredentials which tells the kernel mode layer to decrypt tickets using the App Pool account. You need to change the **UseAppPoolCredentials** from False to True. Read more at [use AppPoolCredentials = True with Kerberos Delegation on 2008](http://blogs.technet.com/b/proclarity/archive/2011/03/08/useapppoolcredentials-true-with-kerberos-delegation-on-2008.aspx).

This would be especially relevant where the backend servers are load balanced as the services will need to run under a common service account.

#### Infrastructure Requirements

Problems with Kerberos authentication can often involve technologies on which the Kerberos protocol depends, or can stem from easy-to-correct oversights in the configuration of Kerberos settings. This section reviews dependencies and summarizes how each dependency relates to troubleshooting Kerberos authentication. While most of these are very fundamental and most likely not the causes in a properly functioning network, any of these fault points will cause KCD with IWA applications to fail.

##### Operating System

Kerberos authentication relies on client functionality that is built in to the Microsoft® Windows Server™. If a client, domain controller, or target server is running an earlier operating system than Windows Server 2003 or Windows XP, it cannot natively use Kerberos authentication and therefore we cannot do Kerberos Constrained Delegation either. This should not be an issue, unless there are very old servers running application services on the network.

##### TCP/IP Network Connectivity

For Kerberos authentication to occur, there must be TCP/IP network connectivity between the Front End Application Proxy Servers, the domain controller, and the target Back End servers.

If you use a firewall, be sure that the Kerberos ports are enabled on the network. For more information about common ports that domain controllers use, see “A List of the Windows Server Domain Controller Default Ports” at <https://technet.microsoft.com/en-us/library/dd772723(v=ws.10).aspx> . Within a single domain or forest this is less likely to be an issue, compared to a scenario where

##### Domain Name System

The client uses the fully qualified domain name (FQDN) to access the domain controller.

* DNS must be functioning for the client to obtain the FQDN.
* For best results, do not use Hosts files with DNS.

##### Time Service

For Kerberos authentication to function properly, it is vital that the time on computers on a network be synchronized. All domains and forests in a network use the same time source. An Active Directory domain controller will act as an authoritative source of time for its domain, which guarantees that an entire domain will have the same time.

Because of these mechanisms, Kerberos authentication relies on the date and time that are set on the KDC and the Front End server; WAP or AADAP. If there is too great a time difference between the KDC and a client requesting tickets, the KDC cannot determine whether the request is legitimate or a replay. Therefore, it is vital that the time on all of the computers on a network be synchronized in order for Kerberos authentication to function properly. Clock skew can be easily diagnosed by reviewing the information in the System log.

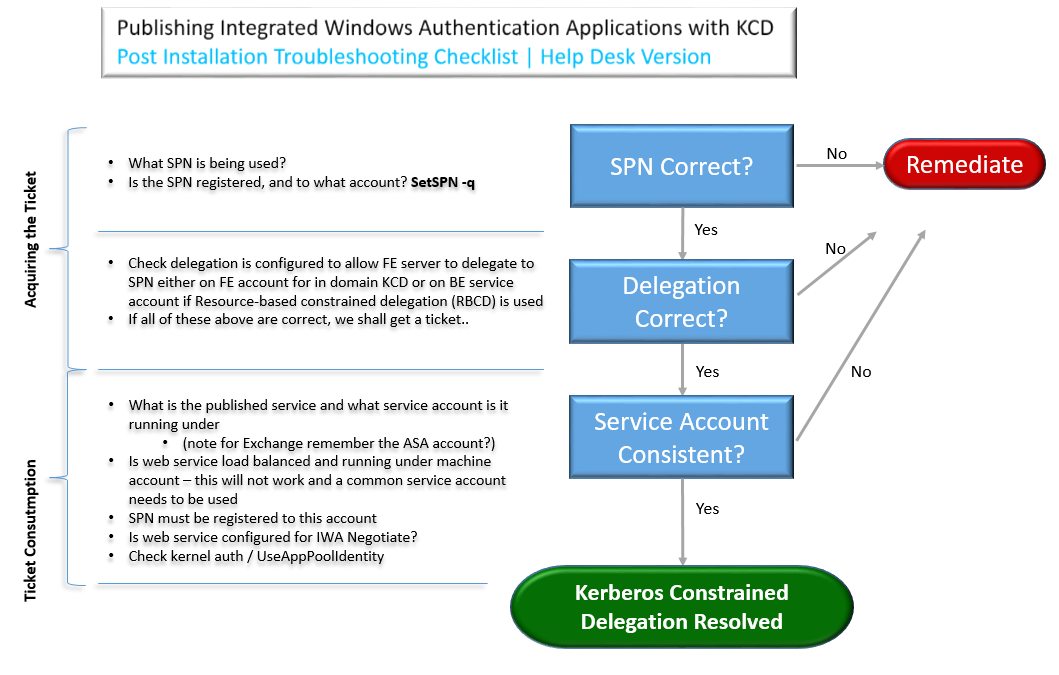
# Troubleshooting

In the previous section, we covered a KCD configuration checklist. That is to be used for checks and balances as you go through the process of configuring IWA applications for KCD. But now let’s consider the post deployment situation. You, or someone else had this all configured, but now it does not work. While all of the bits and pieces for troubleshooting are in the checklist above, we will reorganize the troubleshooting flow as would be done by a support help desk. In fact, this process could be shared with your help desk to help resolve KCD issues with IWA applications. After the Troubleshooting checklist, we have a few more known issues and event log errors to note.

## Troubleshooting Helpdesk Checklist

There are essentially two main things that can go wrong with KCD amongst all of the pieces when have discussed earlier.

1. You must first get a ticket
   1. SPN issues or improper delegation can be a blocker here
   2. You can use [Klist](http://blogs.technet.com/b/tspring/archive/2014/06/23/viewing-and-purging-cached-kerberos-tickets.aspx) to verify the ticket exists
2. Then the ticket is presented to the Back End server e.g. Web Server.
   1. Incorrect configuration on the SPN, incorrect Authentication method set, can be possible causes to prevent the backend server from being able to use the ticket.



## Network Trace

There are many other errors for which the cause might be duplicate SPNs. Kerberos authentication is not possible without properly set SPNs. Source (4me) <https://technet.microsoft.com/en-us/library/cc772897.aspx>

* [Service Principle Name: Registration Duplication](http://blogs.technet.com/b/qzaidi/archive/2010/10/12/quickly-explained-service-principal-name-registration-duplication.aspx)
* [Service Logons Fail Due to Incorrectly Set SPNs](https://technet.microsoft.com/en-us/library/cc772897.aspx#BKMK_Setspn)
  + While the above link is for *Logons* it covers how to find duplicate SPNs which will break KCD as well.

## Using a Website name different than the Hostname

Determine whether you are connecting to the Web site by using the actual NetBIOS name of the server or by using an alias name, such as a DNS name (for example, www.microsoft.com). If you are accessing the Web server by using a name other than the actual name of the server, a new Service Principal Name (SPN) must have been registered by using the SetSPN tool. Because the Active Directory directory service does not know this service name, the ticket-granting service (TGS) does not give you a ticket to authenticate the user. This behavior forces the client to use the next available authentication method, which is NTLM, to renegotiate. If the Web server is responding to a DNS name of www.microsoft.com but the server is named webserver1.development.microsoft.com, you must register www.microsoft.com in Active Directory on the server that is running IIS. To do this, you must run the SetSPN tool on the server that is running IIS.

## Resource-based constrained delegation failures

Consider the following scenario:

* An Active Directory domain (domain A) has two child domains (domain CA and domain CB).
* A client is in domain A, a middle-tier server is in domain CA, and a back-end tier server is in domain CB.
* The middle-tier server uses the resource-based Kerberos Constrained Delegation (KCD) authentication method to obtain a service ticket so that the client can access the back-end tier server. The resource-based constrained delegation uses one of the Windows Server "8" Beta-based domain controllers in domain CA and in domain CB.
* A referral path for Kerberos authentication begins at domain CA, passes through domain A, and ends at domain CB.

In this scenario, the middle-tier server cannot access the back-end tier server.  
  
**Note** *This issue occurs on a domain controller that is running Windows Server 2008 R2 or Windows Server 2008.*

This issue occurs because of the security identifier (SID) filtering feature. When the referral path from the middle-tier to the back-end tier passes through the domain where the client resides, the SIDs of the client in Privilege Attribute Certificate (PAC) are filtered out. Therefore, a valid service ticket for the client cannot be created. This behavior causes a "KDC\_ERR\_POLICY" error to occur.

For hotfix information and more details see:

[Resource-based constrained delegation KDC\_ERR\_POLICY failure in environments that have Windows](http://support.microsoft.com/en-us/kb/2665790)

## Event Log Errors

There are a few places where you can find IWA and KCD related errors. Here are a few to help you out.

|  |  |  |
| --- | --- | --- |
| Event or Symptom | Possible Cause | Resolution |
| AZURE AD APP PROXY ERRORS | also see [Troubleshoot Application Proxy](https://msdn.microsoft.com/en-us/library/azure/dn768218.aspx) | | |
| 12008 - Azure AD exceeded the maximum number of permitted Kerberos authentication attempts to the backend server. | This event may indicate incorrect configuration between Azure AD and the backend application server, or a problem in time and date configuration on both machines. | The backend server declined the Kerberos ticket created by Azure AD. Verify that the configuration of the Azure AD and the backend application server are configured correctly. Make sure that the time and date configuration on the Azure AD and the backend application server are synchronized. |
| 13016 - Azure AD cannot retrieve a Kerberos ticket on behalf of the user because there is no UPN in the edge token or in the access cookie. | There is a problem with the STS configuration. | Fix the UPN claim configuration in the STS. |
| 13019 - Azure AD cannot retrieve a Kerberos ticket on behalf of the user because of the following general API error | This event may indicate incorrect configuration between Azure AD and the domain controller server, or a problem in time and date configuration on both machines. | The domain controller declined the Kerberos ticket created by Azure AD. Verify that the configuration of the Azure AD and the backend application server are configured correctly, especially the SPN configuration. Make sure the Azure AD is domain joined to the same domain as the domain controller to ensure that the domain controller establishes trust with Azure AD. Make sure that the time and date configuration on the Azure AD and the domain controller are synchronized. |
| 13020 - Azure AD cannot retrieve a Kerberos ticket on behalf of the user because the backend server SPN is not defined. | This event may indicate incorrect configuration between Azure AD and the domain controller server, or a problem in time and date configuration on both machines. | The domain controller declined the Kerberos ticket created by Azure AD. Verify that the configuration of the Azure AD and the backend application server are configured correctly, especially the SPN configuration. Make sure the Azure AD is domain joined to the same domain as the domain controller to ensure that the domain controller establishes trust with Azure AD. Make sure that the time and date configuration on the Azure AD and the domain controller are synchronized. |
| 13022 - Azure AD cannot authenticate the user because the backend server responds to Kerberos authentication attempts with an HTTP 401 error. | This event may indicate incorrect configuration between Azure AD and the backend application server, or a problem in time and date configuration on both machines. | The backend server declined the Kerberos ticket created by Azure AD. Verify that the configuration of the Azure AD and the backend application server are configured correctly. Make sure that the time and date configuration on the Azure AD and the backend application server are synchronized. |
| The website cannot display the page. | Your user may get this error when trying to access the app you published if the application is an IWA application, the defined SPN for this application may be incorrect. | For IWA apps: Make sure that the SPN configured for this application is correct. |
| The website cannot display the page | Your user may get this error when trying to access the app you published if the application is an OWA application, this could be caused by one of the following:   * The defined SPN for this application is incorrect * The user who tried to access the application is using a Microsoft account rather than the proper corporate account to sign in, or the user is a guest user.   The user who tried to access the application is not properly defined for this application on the on-premises side. | The steps to mitigate accordingly:   * Make sure that the SPN configured for this application is correct. * Make sure the user signs in using their corporate account that matches the domain of the published application. Microsoft Account users and guest cannot access IWA applications.   Make sure that this user has the proper permissions as defined for this backend application on the on-premises machine. |
| This corporate app can’t be accessed. You are not authorized to access this application. Authorization failed. Make sure to assign the user with access to this application. | Your user may get this error when trying to access the app you published if the user who tried to access the application is using a Microsoft Account rather than the proper corporate account to sign in, or the user is a guest user. | Microsoft Account users and guest cannot access IWA applications. Make sure the user signs in using their corporate account |
| WINDOWS SERVER WEB APPLICATION PROXY RELATED ERRORS | Also see [Web Application Proxy Troubleshooting](https://technet.microsoft.com/en-us/library/dn770156.aspx) | | |
| 12008  Web Application Proxy exceeded the maximum number of permitted Kerberos authentication attempts to the backend server. | This event may indicate incorrect configuration between Web Application Proxy and the backend application server, or a problem in time and date configuration on both machines. | The backend server declined the Kerberos ticket created by Web Application Proxy. Verify that the configuration of the Web Application Proxy and the backend application server are configured correctly.  Make sure that the time and date configuration on the Web Application Proxy and the backend application server are synchronized. |
| 12027  Proxy encountered an unexpected error while processing the request. The name provided is not a properly formed account name. | This event may indicate incorrect configuration between Web Application Proxy and the domain controller server, or a problem in time and date configuration on both machines. | The domain controller declined the Kerberos ticket created by Web Application Proxy. Verify that the configuration of the Windows Server Web Application Proxy and the backend application server are configured correctly, especially the SPN configuration. Make sure the Windows Server Web Application Proxy is domain joined to the same domain as the domain controller to ensure that the domain controller establishes trust with Windows Server Web Application Proxy. Make sure that the time and date configuration on the Web Application Proxy and the domain controller are |
| 13016  Web Application Proxy cannot retrieve a Kerberos ticket on behalf of the user because there is no UPN in the edge token or in the access cookie. | There is a problem with the STS configuration. | Fix the UPN claim configuration in the STS |
| 13019  Web Application Proxy cannot retrieve a Kerberos ticket on behalf of the user because of the following general API error | This event may indicate incorrect configuration between Web Application Proxy and the domain controller server, or a problem in time and date configuration on both machines. | The domain controller declined the Kerberos ticket created by Web Application Proxy. Verify that the configuration of the Web Application Proxy and the backend application server are configured correctly, especially the SPN configuration. Make sure the Web Application Proxy is domain joined to the same domain as the domain controller to ensure that the domain controller establishes trust with Web Application Proxy. Make sure that the time and date configuration on the Web Application Proxy and the domain controller are synchronized. |
| 13020  Web Application Proxy cannot retrieve a Kerberos ticket on behalf of the user because the backend server SPN is not defined. | This event may indicate incorrect configuration between Web Application Proxy and the domain controller server, or a problem in time and date configuration on both machines. | The domain controller declined the Kerberos ticket created by Web Application Proxy. Verify that the configuration of the Web Application Proxy and the backend application server are configured correctly, especially the SPN configuration. Make sure the Web Application Proxy is domain joined to the same domain as the domain controller to ensure that the domain controller establishes trust with Web Application Proxy. Make sure that the time and date configuration on the Web Application Proxy and the domain controller are synchronized. |
| 13022  Web Application Proxy cannot authenticate the user because the backend server responds to Kerberos authentication attempts with an HTTP 401 error. | This event may indicate incorrect configuration between Web Application Proxy and the backend application server, or a problem in time and date configuration on both machines. | The backend server declined the Kerberos ticket created by Web Application Proxy. Verify that the configuration of the Web Application Proxy and the backend application server are configured correctly. Make sure that the time and date configuration on the Web Application Proxy and the backend application server are synchronized. |

# Conclusion

When configuring IWA applications with Azure AD App Proxy or Web Application Proxy, it is essential to understand and properly configure the required building blocks. Get all three right, and then integration is easy. But if you get it wrong, now you also have some troubleshooting steps to help isolate and correct what needs to be done right.

Some scenarios or exceptions may also cause challenges along the way. When initially identifying those situations, you can now be aware of those that are exceptions and thus will not play nicely like the traditional KCD scenarios described.

In addition to the breakdown to understanding and troubleshooting Kerberos Constrained Delegation, you know have a complete end-to-end picture of the building blocks required for these scenarios within a domain and across domains and forests.