RED DE WINDOWS: ACTIVE DIRECTORY

David Pérez Martínez

Omar Montenegro Macía

Abel González Vázquez

Índice

CÓMO SE ALMACENAN E IDENTIFICAN LOS OBJETOS:

Active Directory (AD) es un servicio de directorio para su uso en un entorno Windows Server. Se trata de una estructura de base de datos distribuida y jerárquica que comparte información de infraestructura para localizar, proteger, administrar y organizar los recursos del equipo y de la red, como archivos, usuarios, grupos, periféricos y dispositivos de red.

GUID

Para poder identificar de forma única los objetos en Active Directory, Microsoft utiliza un identificador único global o GUID de 128 bits. Debe tener en cuenta que, si mueve un objeto en su árbol AD, o incluso si cambia el nombre del objeto dentro de Active Directory, este GUID permanece sin cambios. Hay una excepción importante a esto, y sería un movimiento a través de un bosque a otro bosque utilizando algo como la Herramienta de migración de Active Directory (ADMT). Esta situación no conserva el GUID.

Nombres distinguidos

Los humanos ciertamente no vamos a trabajar con GUID para identificar objetos en AD. Afortunadamente, existe otro método de identificación llamado nombres distinguidos (DN).

Por ejemplo, podríamos tener un dominio de labs.cbtnuggets.com. El DN sería: dc=laboratorios,dc=cbtnuggets,dc=com

También tenemos nombres distintivos relativos (RDN) para identificar el objeto en un contenedor principal. Por ejemplo, considere este DN: cn=Administrador,cn=Usuarios,dc=cbtnuggets,dc=com

Tipos de atributos

Aquí hay una lista de estos tipos de atributos:

- CN = Nombre común
- L = Nombre de la localidad
- ST = Nombre del estado o provincia
- O = Nombre de la organización
- OU = Nombre de la unidad organizativa
- C = Nombre del país

- CALLE = Dirección de la calle
- DC = componente de dominio
- UID = ID de usuario

DOMINIOS

Tanto el **Active Directory** como **DNS** establecen espacios de nombres. Podemos entender un espacio de nombres como un área delimitada en la cual un nombre puede ser resuelto. La resolución de nombres es el proceso de traducción de un nombre en un objeto o información que lo representa. Por ejemplo, el sistema de ficheros NTFS puede ser considerado un espacio de nombres en cual un nombre de fichero puede ser resuelto en el fichero propiamente dicho.

Por otro lado, cada ordenador basado en Windows Server que forma parte de un dominio tiene un nombre DNS cuyo sufijo es precisamente el nombre DNS de dicho dominio (siguiendo con el ejemplo, un ordenador de dicho dominio podría denominarse pc0100.miempresa.com). De esta forma, los dominios y ordenadores que se representan como objetos en Active Directory, son también nodos en DNS. Por tanto, resulta fácil confundir ambos espacios de nombres, ya que comparten idénticos nombres de dominio. La diferencia es que, aunque comparten la misma estructura, almacenan información diferente: DNS almacena zonas y registros de recursos y el Directorio Activo almacena dominios y objetos de dominio.

Active Directory utiliza DNS para tres funciones principales:

- Resolución de nombres
- Definición del espacio de nombres
- Búsqueda de los componentes físicos de AD

Se hablará con algo más de profundidad sobre el protocolo DNS en puntos posteriores de este pdf.

El **uso de dominios** permite conseguir los siguientes objetivos:

- Delimitar la seguridad
- Replicar información
- Aplicar Políticas (o Directivas) de Grupo
- Delegar permisos administrativos

Múltiples dominios en la misma organización

Existen muchos casos, especialmente en organizaciones grandes, en los que es interesante que una misma organización disponga de varios dominios (por ejemplo, para reflejar una distribución geográfica o departamental, distintas empresas, etc.). El **Active Directory** permite almacenar y organizar la información de directorio de varios dominios de forma que, aunque la administración de cada uno sea independiente, dicha información esté disponible para todos los dominios. Como se explica a continuación, el conjunto de dominios de una organización pertenece a una estructura lógica denominada bosque, que puede estar formado por uno o varios dominios, distribuidos en uno o varios árboles de dominios.

La estructura de dominios de una organización se basa en los nombres de sus dominios. Puesto que, en Windows Server, estos nombres se basan en el estándar DNS, los dominios se crean en una estructura de árbol invertida, con la raíz en la parte superior. Sin embargo, aunque la estructura se basa en los nombres, la vinculación entre dominios se establece explícitamente mediante las denominadas relaciones de confianza, que se describen más adelante.

Cuando se instala el primer controlador de dominio en la organización se crea lo que se denomina el dominio raíz del bosque, el cual contiene la configuración y el esquema del bosque (compartidos por todos los dominios de la organización). Más adelante, podemos agregar dominios como subdominios de dicha raíz (**árbol de dominios**) o bien crear otros dominios "hermanos" del dominio inicial (es decir, ampliando el número de árboles del **bosque de dominios**), debajo del cual podemos crear subdominios, y así sucesivamente.

Árbol de dominios

Un **árbol** es un conjunto de uno o más dominios dentro de un bosque que comparten un espacio de nombres contiguo, es decir, comparten un sufijo de DNS común. Como hemos dicho, si en una organización existe más de un dominio, estos se disponen en una o varias estructuras de árbol jerárquicas.

El primer dominio que se crea en una organización es el **dominio raíz del bosque**, y crea el propio bosque y el primer árbol de este. Cuando se agrega un dominio a un árbol existente, éste pasa a ser un dominio secundario (o hijo) de alguno de los dominios existentes, que pasa a ser su dominio padre. Los dominios secundarios pueden representar entidades geográficas (Valencia, Madrid, Barcelona), entidades administrativas dentro de la organización (departamento de ventas, departamento de desarrollo

...), u otras delimitaciones específicas de una organización, según sus necesidades.

Los dominios que forman un árbol se vinculan mediante relaciones de confianza bidireccionales y transitivas. La relación **padre-hijo** entre dominios en un árbol de dominio es simplemente una relación de confianza. Sin embargo, los dominios siguen siendo independientes entre sí: los administradores de un dominio padre no son automáticamente administradores del dominio hijo y el conjunto de políticas de un dominio padre no se aplican automáticamente a los dominios hijo.

Por ejemplo, en la Universidad Politécnica de Valencia cuyo dominio actual de Active Directory es **upv.es** se crean dos nuevos departamentos: DSIC y DISCA. Con el fin de permitir la administración de los dominios por parte de los técnicos de los respectivos departamentos, se decide agregar dos nuevos dominios a su árbol de dominios existente en lugar de crear dos unidades organizativas en el dominio principal. Los dominios resultantes, **dsic.upv.es** y **disca.upv**.es forman un espacio de nombres contiguo, cuya raíz es upv.es. El administrador del dominio padre (upv.es) puede conceder permisos para recursos a cuentas de cualquiera de los tres dominios del árbol, pero por defecto no los puede administrar.

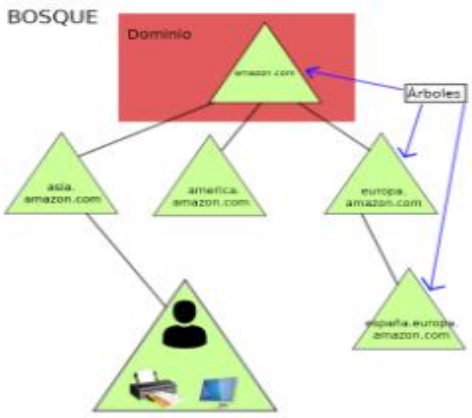
Bosque de dominios

Un **bosque** se define como un grupo de árboles que no comparten un espacio de nombres contiguo, y que se conectan mediante relaciones de confianza bidireccionales y transitivas. A efectos prácticos, se debe recordar que sea cual sea la cantidad y estructuración de dominios de una organización, todos ellos constituyen un único bosque. Por lo tanto, aunque en la organización exista un único dominio, o varios dominios en un único árbol, dicho dominio o árbol constituyen por sí mismos el bosque de la organización. En un bosque, todos los dominios comparten la **misma configuración**, el mismo esquema de directorio, y el mismo catálogo global.

Añadir nuevos dominios a un bosque es fácil. Sin embargo, existen ciertas limitaciones que hemos de tener en cuenta al respecto:

- No se pueden mover dominios de Active Directory entre bosques.
- Sólo se puede eliminar un dominio de un bosque si este no tiene dominios hijo.
- Después de haber creado el dominio raíz de un árbol, no se pueden añadir al bosque dominios con un nombre de dominio de nivel superior.
- No se puede crear un dominio padre de un dominio existente

En general, la estructuración de los dominios de una organización mediante un bosque con uno o varios árboles permite mantener convenciones de nombres de dominio tanto contiguos como discontinuos, lo cual puede ser útil en organizaciones con divisiones independientes que quieren mantener sus propios nombres DNS. Finalmente, debemos relacionar estos conceptos con el procedimiento para crear un dominio. Esto se hace mediante la ejecución de un asistente denominado **dcpromo.exe** en el sistema Windows Server 2008 que queramos promocionar a controlador de dominio.



NIVELES FUNCIONALES

Un **nivel funcional**, que puede estar definido a nivel de dominio o de bosque, establece simultáneamente una serie de características o funcionalidades disponibles en el dominio/bosque y la posibilidad de ser compatible con una versión previa de Windows Server a nivel de servidor (DC). Es decir, cuando situamos el nivel funcional del dominio/bosque en un valor determinado, podemos tener en dicho dominio DCs de cualquier versión de Windows Server que admita dicho nivel simultáneamente.

Si elevamos el nivel funcional, ampliamos las posibilidades del dominio/bosque, pero a costa de no poder tener DCs de versiones previas de Windows que no sean compatibles con dicho nivel funcional. Una vez elevado el nivel funcional de un dominio/bosque, no puede volver a ponerse en el nivel previo. A efectos prácticos, podemos entender los niveles funcionales como una forma razonable de actualizar los servidores (DCs) de los dominios de una organización a una versión superior de Windows Server. Lo habitual es instalar un DC con la versión nueva.

SERVIDOR DE CATÁLOGO GLOBAL

El catálogo global es una partición de sólo lectura que almacena una copia parcial de las particiones de dominio de todos los dominios del bosque. La copia es parcial porque, aunque incorpora todos los objetos de cada dominio, de cada uno sólo almacena un subconjunto reducido de atributos. En particular, se guardan aquellos que se utilizan más frecuentemente para las consultas.

Un servidor de catálogo global es un controlador de dominio que almacena una copia del catálogo y procesa las consultas al mismo. En cada bosque debe existir al menos un DC configurado como servidor de catálogo global, y esta función puede incorporarse a cualquier otro DC del bosque que se desee.

FSMO ROLES

Los Flexible Single Master Operations, conocidos como «FSMO roles» o también «Operations Master Roles», no son roles como los que se instalan a través del Server Manager en Windows Server. Son una serie de funciones de Active Directory para prevenir conflictos de replicación.

Cambios como actualizar las propiedades de un objeto, cambiar la contraseña, etc. Se pueden hacer desde cualquier DC de escritura. En este modelo los conflictos se resuelven dando prioridad al último que haga el cambio. Si, por ejemplo, reseteamos la contraseña de un usuario desde 2 DCs diferentes, el último que lo haga prevalece.

Sin embargo, hay ciertos cambios que son demasiado importantes y críticos en AD para que se puedan hacer desde cualquier DC, y para evitar conflictos se recurre a un sistema Single-Master. Estos cambios tan importantes son llevados a cabo por los FSMO roles, que no tienen por qué estar en un mismo DC, pueden distribuirse entre varios DCs.

Existen 5 FSMO roles, 2 de ellos son a nivel de bosque, y 3 a nivel de dominio. En un bosque sólo puede haber un **Schema Master** y un **Domain Naming Master**. Y por cada dominio del bosque existirá un **PDC Emulator**, un **RID Master** y un **Infrastructure Master**.

Nota: Se utilizará la palabra dominio para referirse a árbol

FSMO roles a nivel de bosque:

- **Schema Master** (Maestro de Esquema): El DC que tenga este rol es el único autorizado para modificar el esquema de AD. Cualquier cambio en el esquema, no importa desde que DC lo hagamos, siempre estará

«conectado» al DC con el FSMO rol de Schema Master. Y éste tiene que estar siempre online cuando actualicemos el esquema.

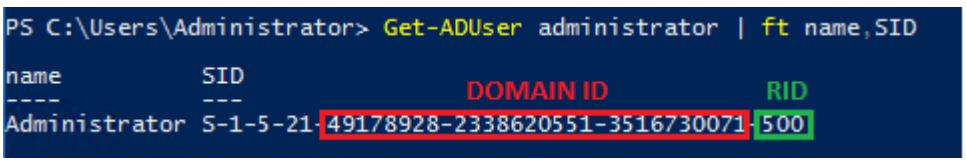
- **Domain Naming Master** (Maestro de Nombres de Dominio): Se encarga de autorizar el alta o baja de dominios en el bosque, así como de mantener una lista actualizada de todos los dominios existentes (para que, por ejemplo, no se dé el caso de intentar crear dos dominios con el mismo nombre al mismo tiempo). También tiene la función de añadir o eliminar particiones de aplicación en AD. Tiene que estar online cuando añadimos o eliminamos dominios o particiones de aplicación.

FSMO roles a nivel de dominio:

- **PDC Emulator** (Emulador de PDC – Primary Domain Controller). Tiene varias funciones:
  - Recibe las actualizaciones de las contraseñas de usuarios y equipos.
  - Crea y actualiza las políticas de grupo (cuando modificamos las políticas del dominio desde la GPMC (Group Policy Management Console), ésta trata de conectarse siempre al DC con el FSMO rol de PDC).
  - Sincroniza tiempos, sólo él puede sincronizar la hora con un servidor externo
  - Debe estar siempre online y accesible.

    Es recomendable que el PDC Emulator y el RID Master residan en el mismo DC.

- **RID Master** (Relative Master ó Maestro de RID): En un dominio, cuando se crea un objeto (sea un usuario, un grupo o un equipo), se le asigna un SID (Security IDentifier) que es único y que nunca es reutilizado incluso aunque el objeto sea eliminado. Haciendo una analogía, es algo así como el documento de identidad personal. Un **SID** consta de tres partes:

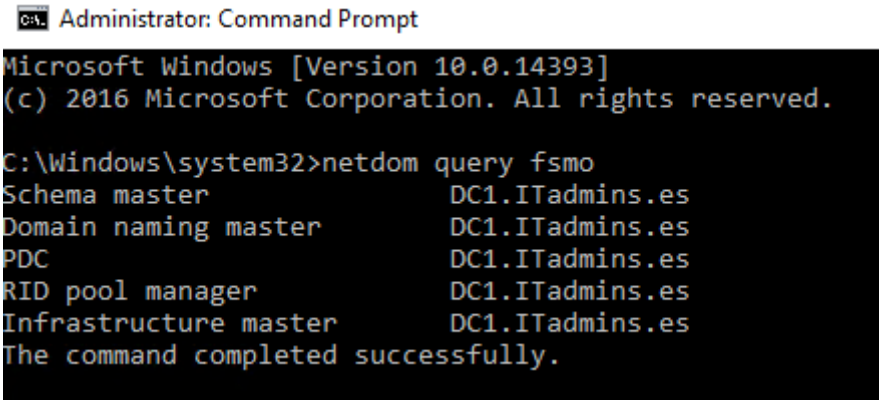


  - **SID:** Es un identificador asignado por ISO que especifica:

        - S-1 : Especifica que la cadena es un SID y la versión de revisión actual es 1.
        - 5: Identifica a la autoridad (5 es para NT Authority, indica que es un SID específico de Windows).
        - 21: Indica que lo que viene a continuación es un Domain ID.

- **DOMAIN ID:** Es un identificador del dominio, todas las cuentas de usuario, grupo y equipo de un mismo dominio comparten estos datos.
- **RID:** Es un número que se asigna de forma secuencial a partir del número 1000 a cada nuevo objeto que se crea en el dominio. Para los objetos creados por defecto con el sistema existen una serie de RIDs estándar: 500 para el administrador, 501 para el usuario invitado, 502 para el Kerberos Key Distribution Center.

- **Infrastructure Máster** (Maestro de Infraestructuras): En un entorno multidominio, es el responsable de actualizar las referencias de los objetos de un dominio en los objetos de otros dominios. En otras palabras, se encarga de traducir los GUIDs (Global Unique IDentifiers), los SIDs y los DNs (Distinguished Names) de los objetos de otros dominios.
    - Este rol sólo tiene «trabajo» cuando estamos en un entorno multidominio, y nunca debería estar ubicado en un DC que sea a su vez Catálogo Global. Cuando un DC es Catálogo Global, éste recibe actualizaciones periódicas de los objetos de todos los dominios mediante la replicación, de forma que la información sobre esos objetos siempre está actualizada.
    - Si el maestro de infraestructuras encuentra objetos sin actualizar, solicita la información actualizada a un GC y después los replica a los otros DCs del dominio.

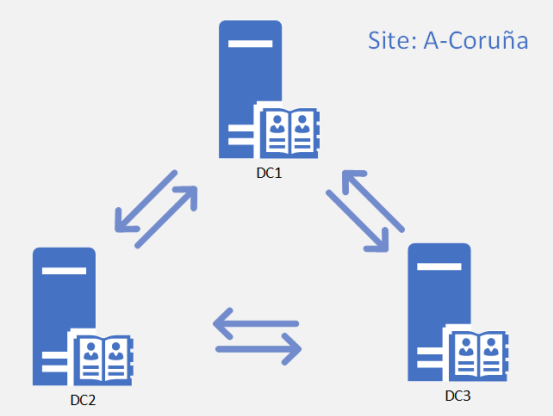


REPLICACIÓN

¿Cómo funciona la replicación?

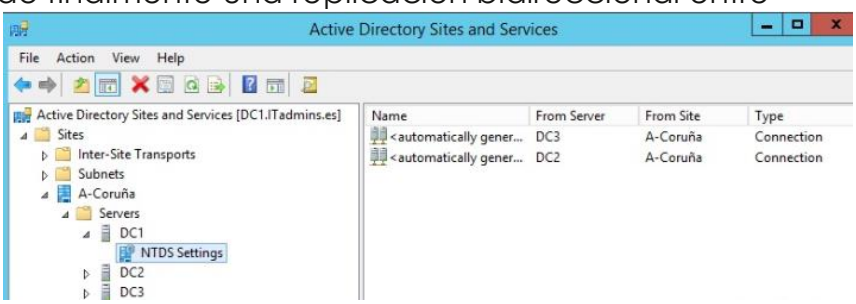
Un Sitio (**Site**) es una o más redes IP, con conectividad permanente, confiable y con suficiente ancho de banda disponible para la replicación. La **replicación** tiene distintos objetivos en cada caso. Si no tenemos problemas de conectividad, el objetivo principal es que un cambio que se produce en un **Controlador de Dominio** se replique en muy poco tiempo en el resto de los Controladores de Dominio. En cambio, si tenemos posibles restricciones de conectividad como puede suceder con enlaces WAN, el objetivo es diferente, como, por ejemplo, controlar y disminuir dicho **tráfico de replicación**. Nos podemos encontrar con dos escenarios diferentes, cuando tenemos a todos los DCs en el mismo Site (Replicación Intrasite) y cuando tenemos los DCs en diferentes Sites (Replicación Intersite).

Replicación Intrasite:



En el anterior ejemplo tenemos un escenario con 3 DCs ubicados en el Site: A Coruña, los cuales replican todos entre sí. El sentido de las flechas indica que hay dos procesos de replicación distintos: en un sentido y en el otro (entrante y saliente), consiguiendo finalmente una replicación bidireccional entre todos los DCs.

Podemos comprobar que automáticamente se crean unos



«conectores de replicación» entre los distintos DCs.

Con lo que vemos que la replicación tiene lugar entre pares de DCs

¿Quién crea estos conectores? Pues hay un proceso en AD llamado Knowledge Consistency Checker (KCC) que se encarga de ello. Tiene como responsabilidad crear y mantener los conectores de replicación ante posibles cambios en la topología de nuestra organización.

Cuando hacemos un cambio en un DC cualquiera, por ejemplo, si creamos un usuario en el DC1, este se guardará en la base de datos del DC1, y le será asignado un USN. Un USN no es más que un valor numérico que se asigna de forma secuencial, incrementándose en 1 con cada cambio. Son únicos para cada DC.

Cada par de DCs mantiene su propia conversación de replicación, la cual extrae información de dos tablas de valores: «High-Watermark Vector» y «Up-To-Dateness Vector».

El **High-Watermark Vector** (HWMV) es el valor USN más alto que un DC ha recibido desde un partner de replicación directo. Cada DC mantiene una tabla con estos valores.

El **Up-To-Dateness Vector** (UTDV) es el valor USN más alto que un DC ha recibido de otro DC cualquiera. Cada DC mantiene una tabla de Up-To-Dateness Vector de cada uno de los DCs que existen o han existido en el bosque. Esta tabla también contiene una marca de tiempo (timestamp) de la última vez que el DC replicó satisfactoriamente con cada uno de esos DCs.

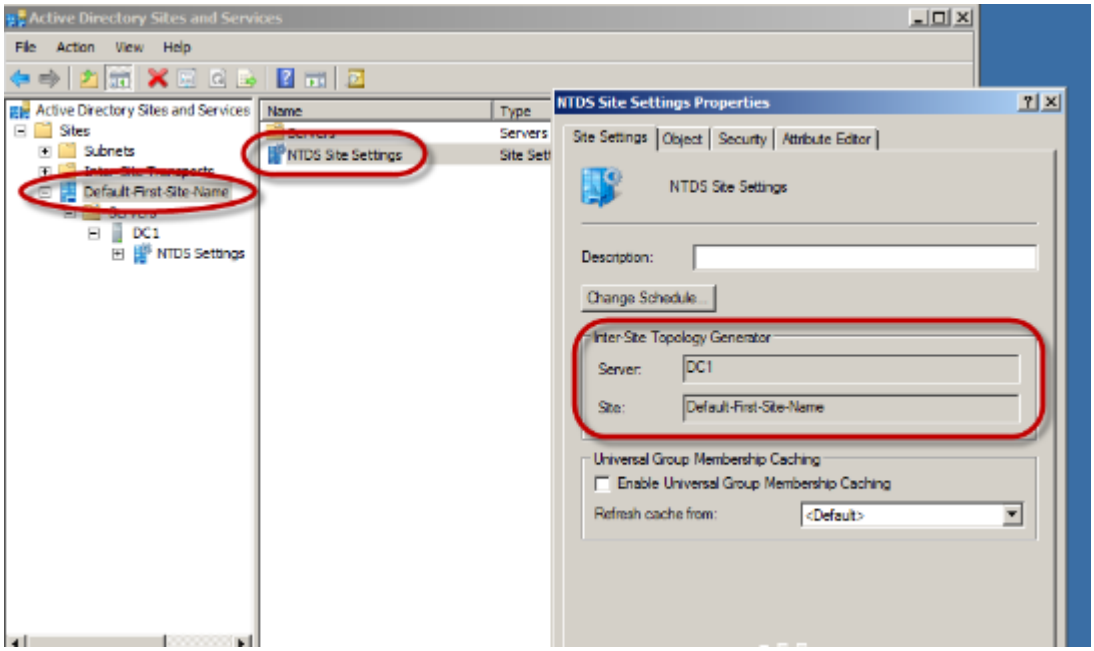
En un momento dado, o en intervalos de una hora si no ha tenido lugar ningún cambio antes, DC2 envía una solicitud a DC1 preguntando si hubo algún cambio (si hubiese algún cambio antes de este tiempo en DC1, éste notificaría a DC2 para que se iniciase la replicación). Como curiosidad, la replicación dentro de un mismo Site tarda 15 segundos en notificar un cambio, esto es así para que puedan producirse más cambios (objetos a replicar) dentro de una misma transacción, y si hubiera más de un partner de replicación los iría notificando con incrementos de 3 segundos a cada uno de esos partners. A efectos prácticos se considera que la **Replicación Intrasite** es instantánea. En nuestro ejemplo sólo hay un partner que sería DC2.

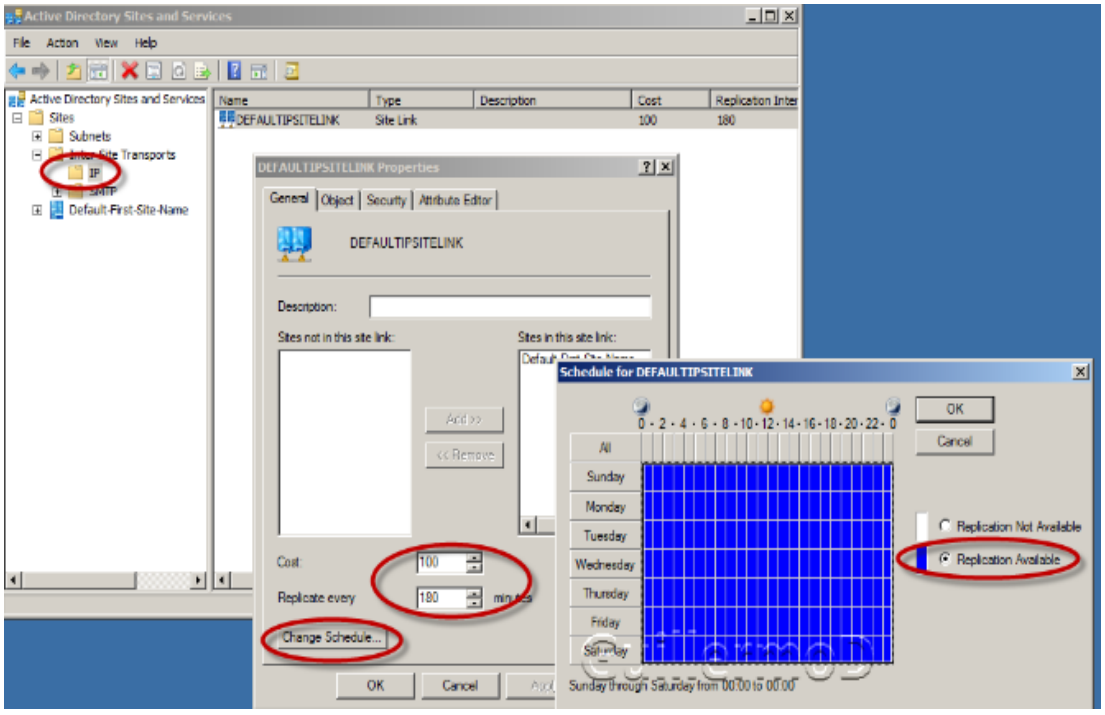
Replicación Intersite:

La replicación se maneja de forma totalmente diferente, ya que el objetivo es controlar y disminuir el tráfico.Dentro de cada Site, hay un Controlador de Dominio que asume la función de ISTG = Inter-site Topology Generator.

Éste es quien nombra al **Bridgehead Server** (Servidor Cabeza de Puente) que será el que replicará con otro Site. Se pueden configurar los **Bridgehead Server** preferidos por cada protocolo (IP o SMTP) pero debemos tener cuidado porque en ese caso el sistema lo elegirá solamente entre los preferidos. Si elegimos uno o varios y ninguno de esos estuviera disponible, se cortaría la replicación entre Sites.

La **frecuencia de replicación** entre Sites por omisión es de 3 horas, configurable entre 15 minutos, y una vez a la semana, y además se pueden configurar: los días y horarios que esté habilitada la replicación, asociarle un **Costo**, que no es función del ancho de banda, sino que es una preferencia administrativa para definir los Objetos Conexión, cuando hay posibilidades redundantes.



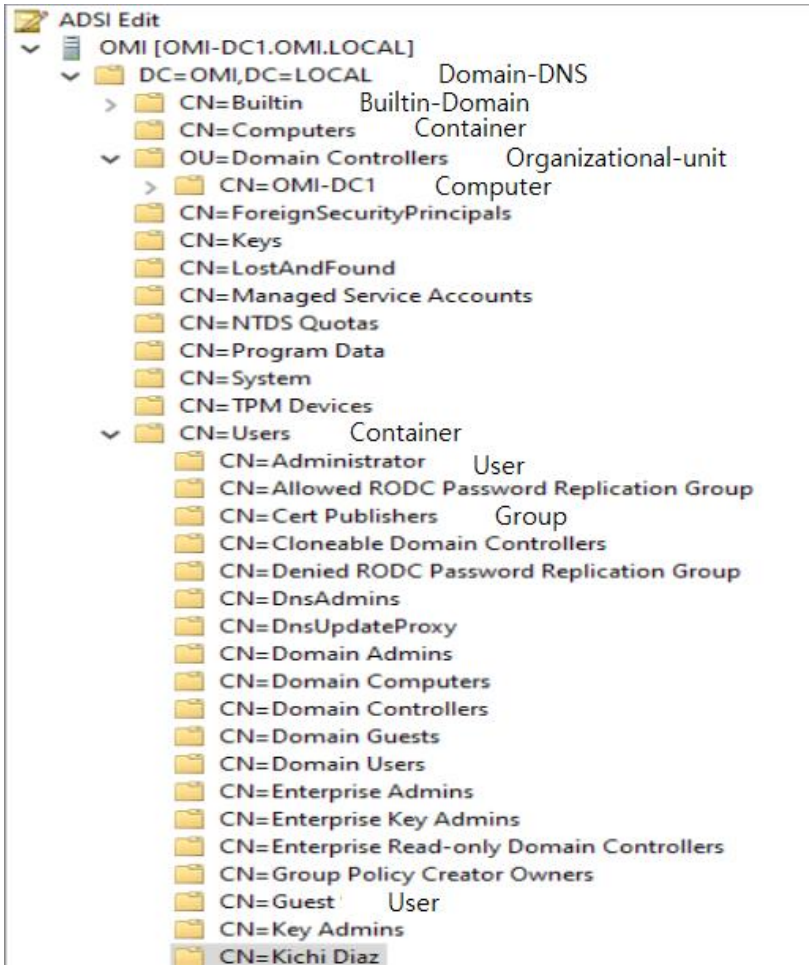


ACTIVE DIRECTORY SCHEMA:

El objetivo de entender los fundamentos del esquema de AD es que el administrador sea capaz de poder **extender** dicho esquema para adaptarse a las necesidades de la organización. Por ejemplo, una universidad necesita guardar los créditos obtenidos de cada alumno, como no existe por defecto nada relacionado con los créditos para asociar a un usuario, es necesario implementar un objeto en el esquema que represente los créditos de un alumno.

El esquema de Windows Active Directory es una especie de plano separado del bosque, que contiene las definiciones de los posibles objetos que se pueden crear dentro del bosque. **Cualquier objeto del bosque en AD es una instancia de una clase en el esquema,** incluso el propio esquema es una instancia de una clase. Una clase es una estructura de datos y una instancia de dicha clase es el almacenamiento de los datos en esa estructura, si yo quiero guardar datos de un empleado como su nombre, nºtlf, email... tendré que crear una instancia de la clase que tenga dichos datos en su estructura.

Para todo un bosque sólo hay un esquema de AD, todos los objetos creados en un bosque son instancias de clases definidas en el esquema asociado a ese bosque:

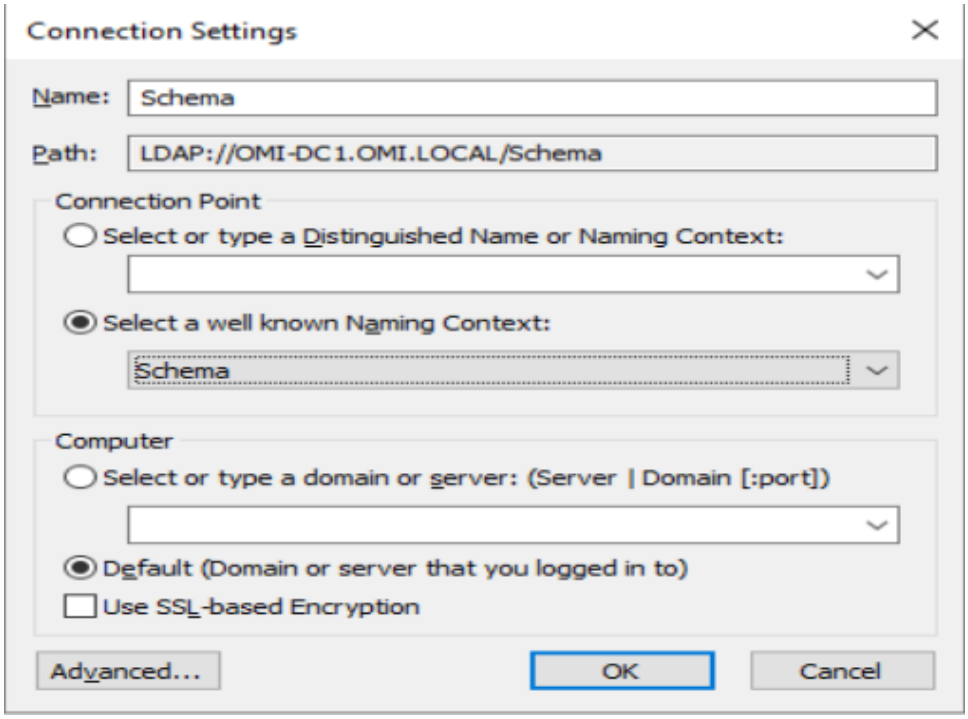


Se especifica a qué clase pertenece algunas de las instancias del bosque

Para que el esquema del bosque mantenga la coherencia al tratar con varios subdominios, se le concede la gestión centralizada a un domain controller, que es el que recibe el rol de **SchemaMaster**.

Para cada bosque el esquema de partida contiene una serie de definiciones por defecto, en Windows Server se puede consultar su contenido mediante consultas de PowerShell o utilizando el snap-in **Active Directory Services Interfaces Editor (ADSI Edit).**

Para poder ver el esquema en ADSI Edit, tendremos que conectarnos al path LDAP correspondiente al esquema:



Al presionar click derecho en ADSI Edit y seleccionar Connect to...

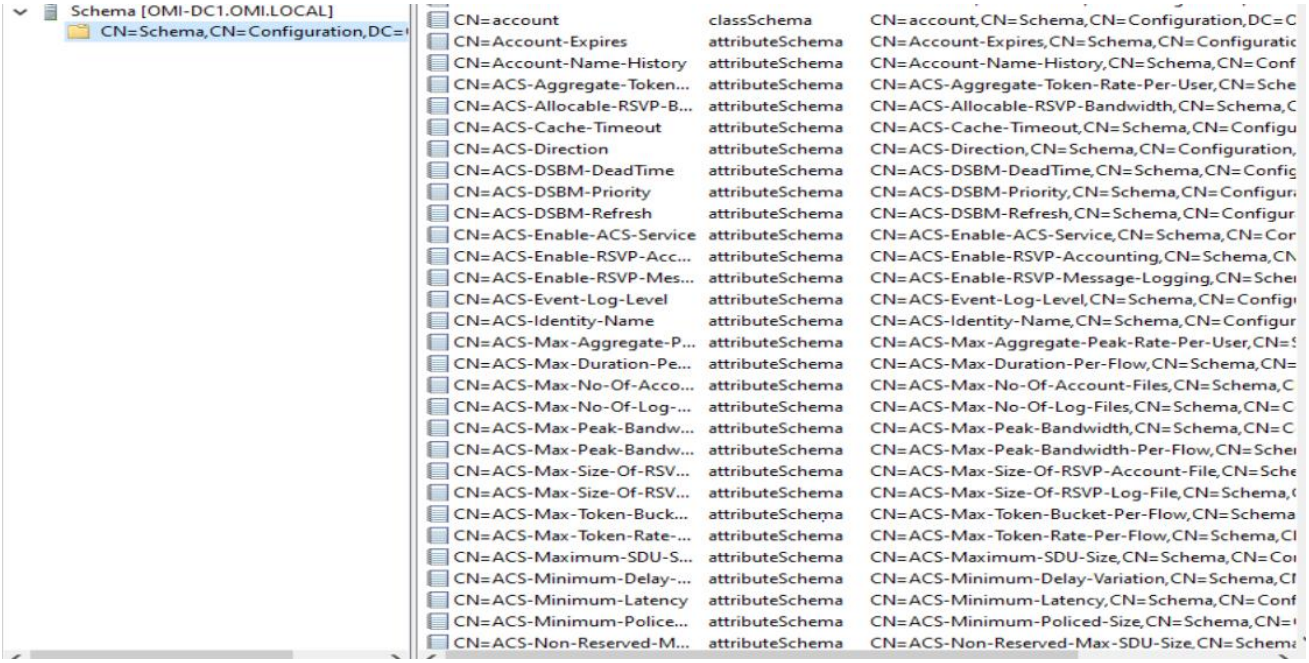
ESTRUCTURA DEL ESQUEMA DE ACTIVE DIRECTORY:

En el esquema se guardan **únicamente** 2 elementos fundamentales:

-Clases: las estructuras de datos

-Atributos: los datos y su tipo dentro de una clase.

Una clase puede tener varios atributos, y un mismo atributo puede ser usado en varias clases, por eso el esquema guarda por separado estos 2 elementos. Todas las clases del esquema son **instancias** de la clase **classSchema** y todos los atributos del esquema son **instancias** de la clase **attributeSchema**.



Esquema por defecto de AD

Al esquema se le identifica por el **nombre distinguido: CN=Schema, CN=Configuration, DC=OMI, DC=LOCAL** lo que quiere decir que el esquema es un contenedor (CN) que se encuentra dentro del contenedor "Configuration" en el **bosque OMI.LOCAL (DC).**

En el plano lógico si es verdad que el esquema reside dentro del contenedor de configuración, pero en el plano físico el esquema se encuentra en una partición independiente.

Todos los objetos que residen dentro del esquema tendrán de nombre distinguido el propio del objeto más el nombre distinguido del esquema del bosque. Por ejemplo, para la clase account, su nombre distinguido es **CN=account, CN=Schema, CN=Configuration, DC=OMI, DC=LOCAL.**

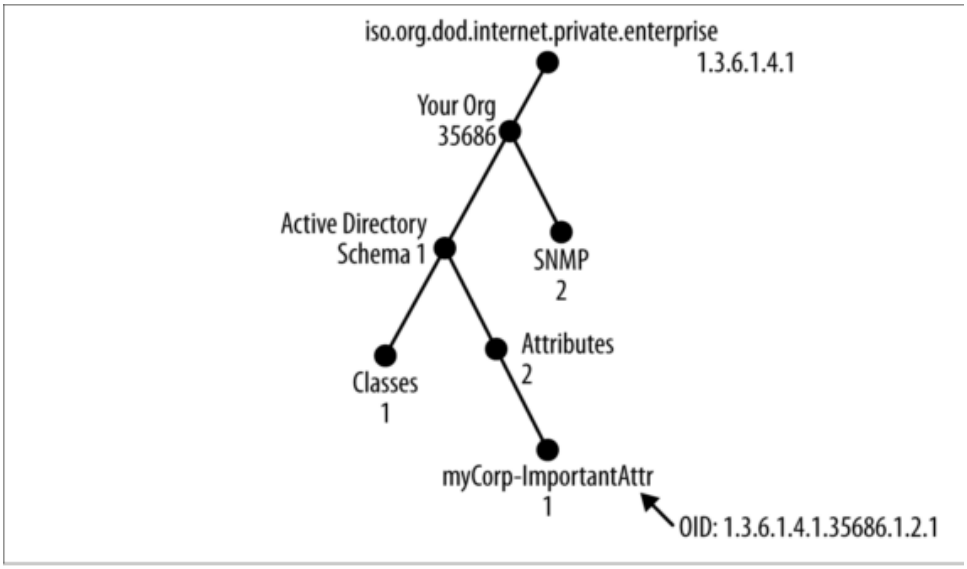
Estos objetos también tienen otro nombre, que puede ser utilizado para realizar consultas al usar LDAP o ADSI guardado en el atributo **lDAPDisplayName**. El nombre LDAP es el nombre del objeto, pero empezando por minúscula y sin guiones. Por ejemplo, para el objeto Domain-Policy-Object, su nombre LDAP es **domainPolicyObject.**

X.500 OID NAMESPACE:

Active Directory está basado en el estándar X.500 desarrollado por ISO e ITU en 1988. Este estándar utiliza un método especial para identificar objetos de forma única a **nivel mundial,** la idea del **Object IDentifier(OID)** es análoga a la de las direcciones IP pero en vez de redes y subredes tenemos OID base y OIDs derivados de ese OID base, los OID **no se agotan.**

Si queremos crear un objeto **en el esquema,** necesitamos solicitar un OID único para nuestra organización a través de la IANA, ya que si utilizamos un OID ya ocupado, puede **generar conflictos** con otros dominios visibles. Alternativamente, Microsoft provee de un script para autogenerar OIDs únicos cada vez que es ejecutado, aunque las buenas prácticas recomiendan adquirir los OIDs a través de la IANA.

Una vez obtenido un OID, por ejemplo el 1.3.6.1.4.1.35686, tenemos total control de ese OID con el que podremos crear ramas y hojas como queramos. El OID de la IANA es el 1.3.6.1.4.1, por lo que la IANA puede otorgar los identificadores que quiera a partir de este OID base.



Árbol OID de ejemplo

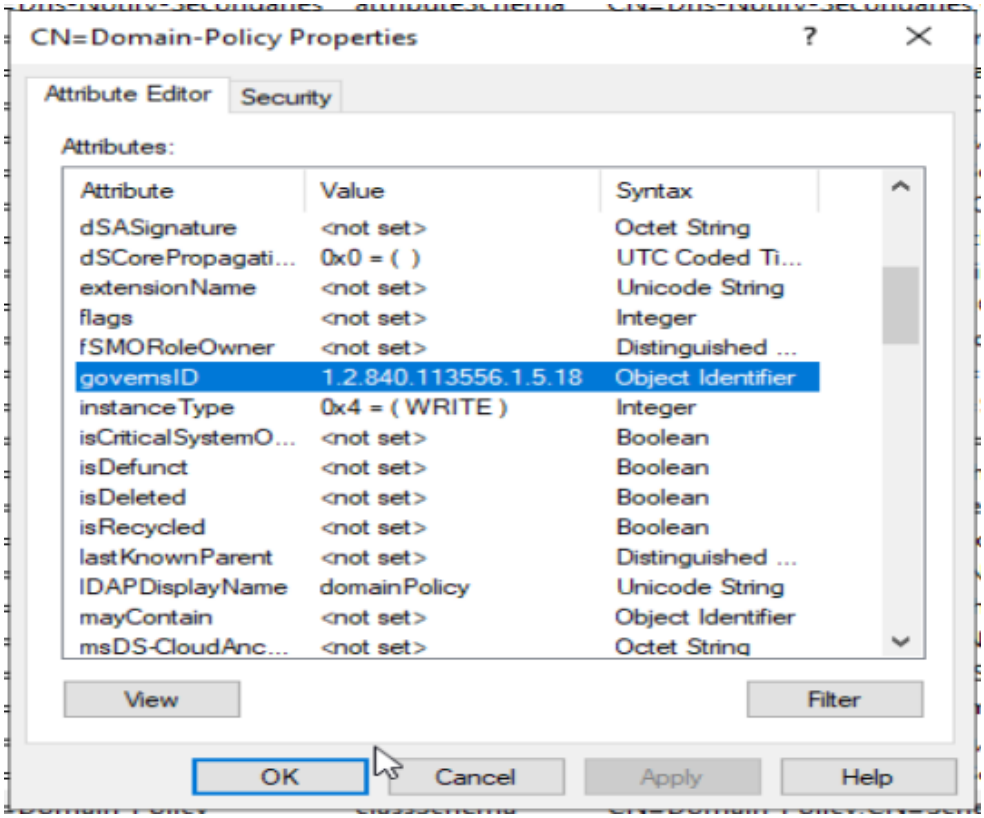
El OID (object identifier) está compuesto de dos partes:

-La primera identifica la rama que contiene el objeto

-La segunda indica el objeto

Por ejemplo, el OID 1.3.6.1.4.1.3385.12.497 identifica al objeto 497 en la rama 1.3.6.1.4.1.3385.12

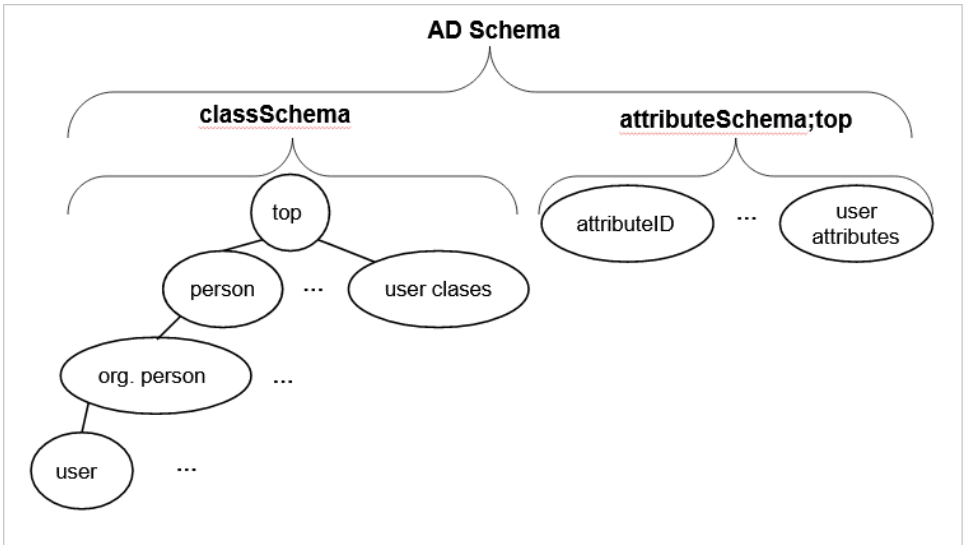
Al consultar la información de una clase en el esquema, podremos consultar su OID a través del atributo **governsID**, de sintaxis **Object Identifier**, la sintaxis es el tipo de dato de un atributo. Para un atributo se llama **attributeID**.



OID de una clase del esquema

CLASES DEL ESQUEMA (classSchema Objects):

Las clases son objetos que definen una estructura de datos y otras propiedades que veremos. Podemos crear instancias de las clases y sobretodo **realizar herencias** de otras clases (crear nuevas clases a partir de otras existentes, el nuevo objeto contiene un atributo "subClassOf" que dice de qué clase es subclase). Al final todas las clases del esquema heredan de la clase **top, que es la clase raíz del esquema.**



Jerarquía de clases dentro de classSchema

En la anterior imagen, se puede ver como la clase "user" es subclase de "organizationalPerson" que es subclase de "person" y que es subclase de "top".

Es importante no confundir el concepto de **crear una nueva clase** y el concepto de **crear una instancia de una clase,** ya que en el primer caso simplemente estamos definiendo las propiedades de la clase, mientras que crear una instancia significa **crear un objeto con todas las propiedades de la clase** para guardar nuestra información (por ejemplo, a la hora de crear un objeto usuario llamado "Pepito" estamos creando una instancia de la clase user).

Las clases además de simplemente guardar datos contienen otras propiedades como **su categoría.** Según la categoría, las reglas de creación de instancias y herencia pueden cambiar para una clase. Existen las siguientes 4 categorías:

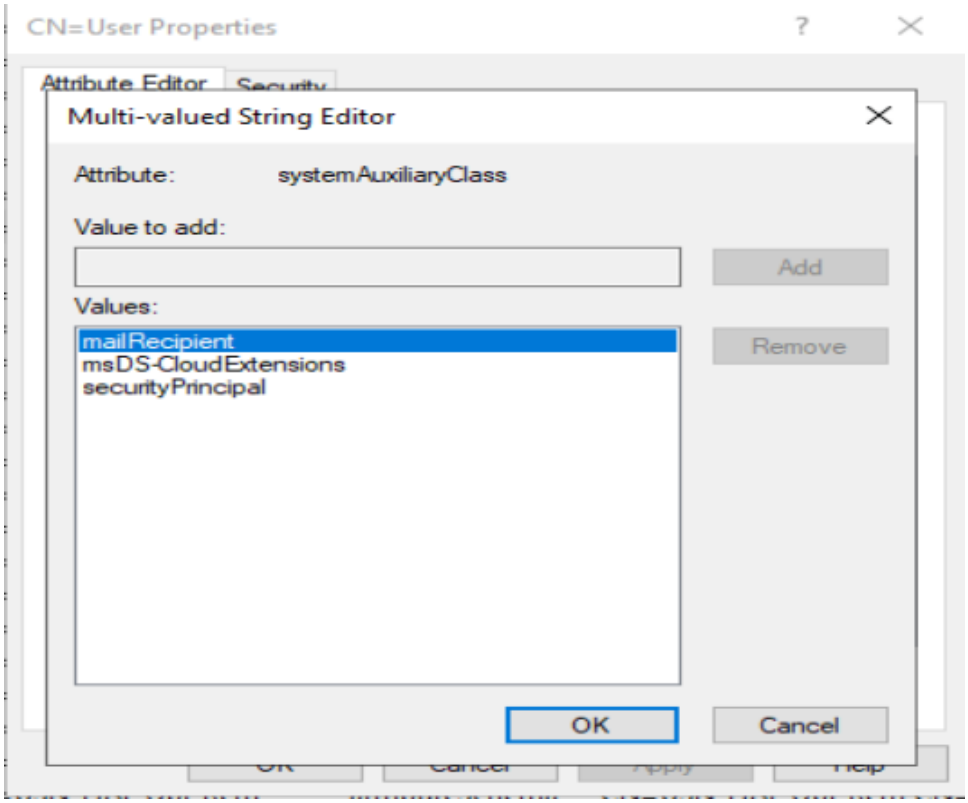
-**Estructurales**: se pueden crear directamente instancias de dicha clase. Un ejemplo es la clase user, en la que una instancia de la clase representa un usuario de la organización.

-**Abstractas**: el concepto de clase abstracta en AD es análogo al de programación orientada a objetos. No se pueden crear instancias directamente de una clase abstracta. Un ejemplo es la clase organizationalPerson, que contiene información básica de un empleado como el nº de empleado, departamento... entonces todas las clases que traten con personas dentro de una organización deberían de heredar de esta clase (como por ejemplo la clase "User"). Una clase abstracta **sólo** puede heredar de otra clase abstracta.

-**Auxiliares**: están hechas con el fin de agrupar atributos, permite que otras clases puedan heredar colecciones de atributos sin que ellas mismas los definan y **tampoco** se pueden crear instancias directamente de una clase auxiliar. Por ejemplo, más de una clase en el esquema necesita guardar configuración del correo, entonces entre varias clases se repite la definición de los atributos relativos al correo, la clase Mail-Recipient almacena todos esos atributos por lo que las clases anteriores pueden utilizar esos atributos con hacer una herencia de Mail-Recipient. Una clase auxiliar **no puede heredar** de una clase estructural.

Para utilizar los atributos contenidos en una clase auxiliar no hace falta heredar directamente de ella, Active Directory provee de un atributo multivaluado que contiene todas las clases auxiliares asociadas llamado

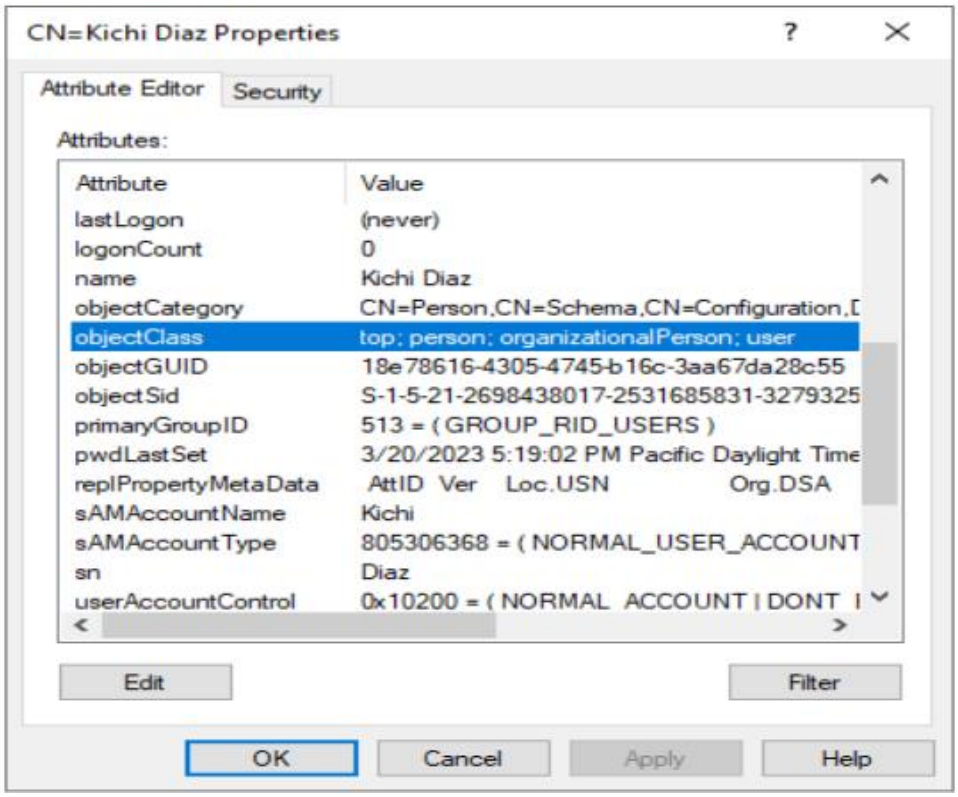
AuxiliaryClass, el atributo llamado **systemAuxiliaryClass** es lo mismo, pero está restringido por el sistema**.**



Clases auxiliares que está usando la clase "User"

-**Clases del 88**: la organización de clases en categorías viene del estándar X.500 de 1993, esta categoría hace referencia a las clases anteriores al estándar, concretamente del año 88.

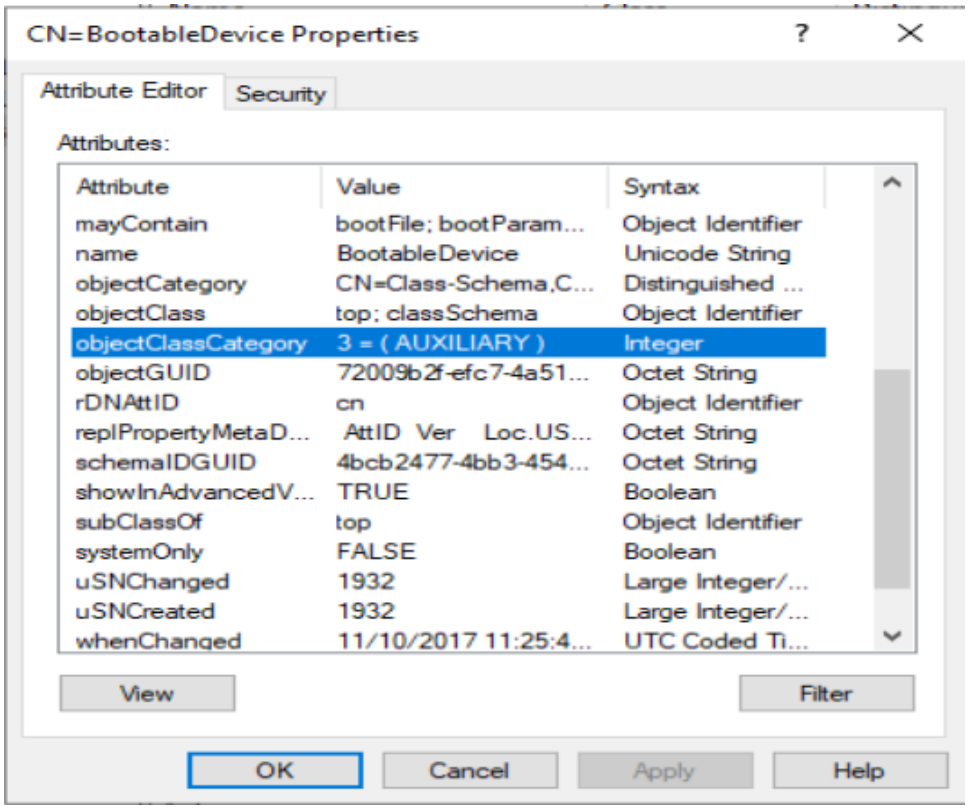
Si queremos saber de qué clase se ha creado una instancia podemos consultar el atributo **ObjectClass,** que nos dice de que clases deriva, la última clase listada sería de la que es instancia. Este atributo coincide muchas veces con el **ObjectCategory,** pero no son lo mismo (la información de ObjectCategory es sacada del atributo defaultObjectCategory de la clase de la que el objeto es instancia, que de valor puede contener la propia clase o una de sus superclases). Por ejemplo, si creamos un nuevo usuario, **objectClass** nos dirá que pertenece a la clase user (de la que realmente es instancia), pero **objectCategory** que pertenece a la clase person.



Kichi Diaz es una instancia de la clase user, que también deriva de las clases que se listan antes que user

Para saber de qué categoría es una clase, basta con consultar el atributo "objectClassCategory" de sintaxis integer, que puede adoptar los valores siguientes:

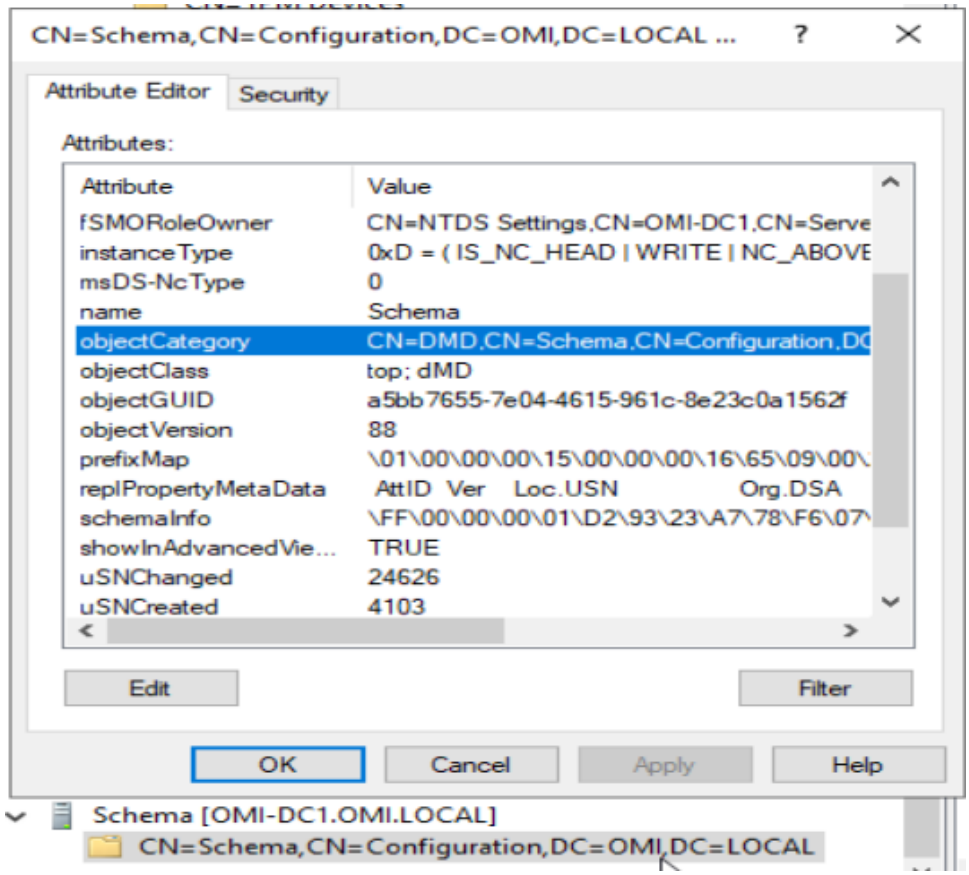
- 0: clase del 88
- 1: estructural
- 2: abstracta
- 3: auxiliar



Clase auxiliar en AD

Si al crear una nueva clase no se le especifica categoría, por defecto será una **clase del 88.**

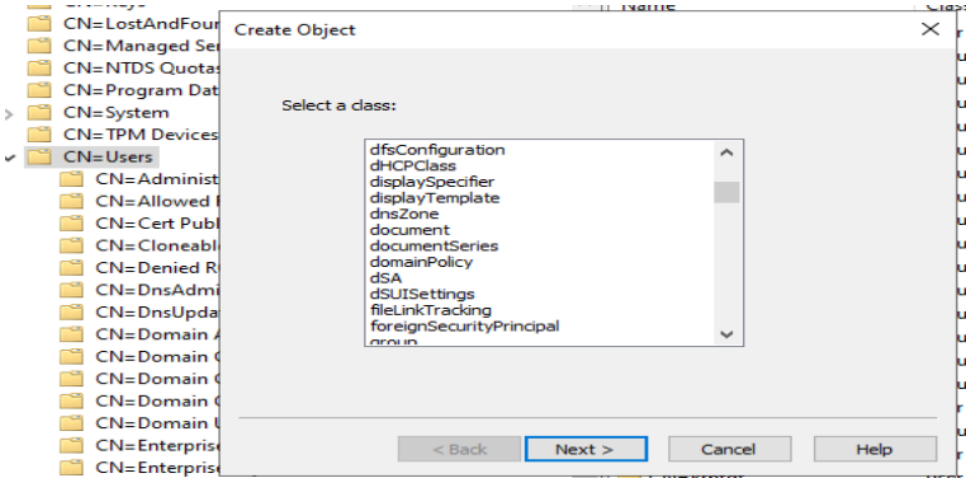
Clase DMD (Directory Management Domain): es la clase que contiene al esquema.



El propio esquema es instancia de la clase DMD

CREACIÓN DE INSTANCIAS EN UN BOSQUE:

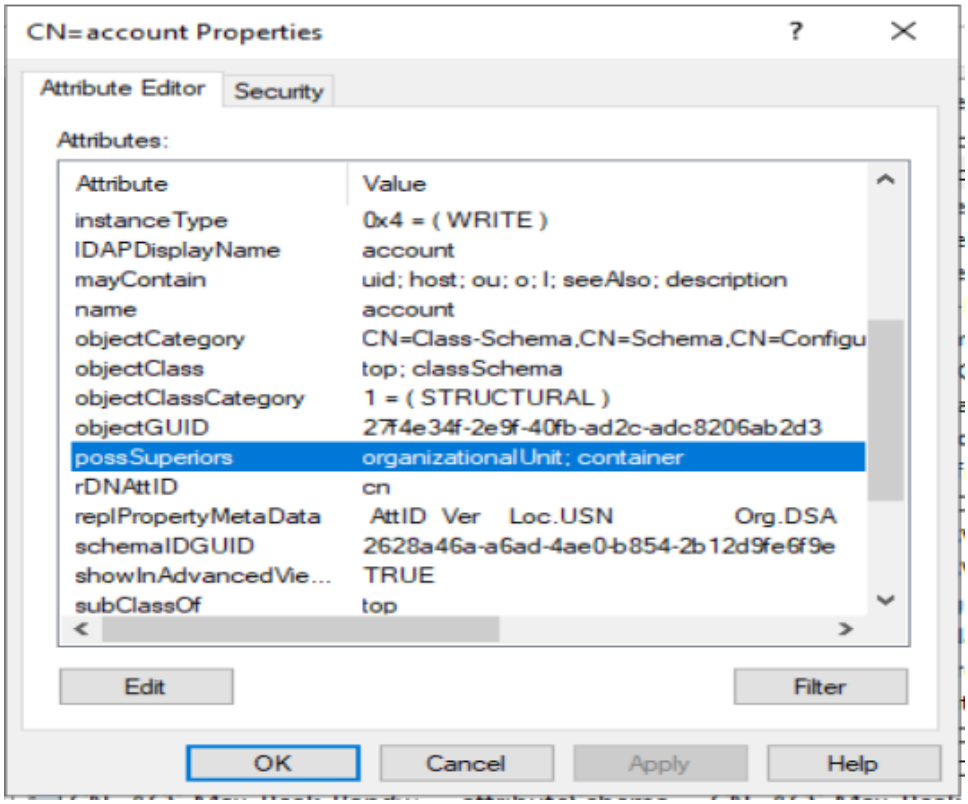
Dentro de un objeto en el bosque se pueden **crear otros objetos,** según el objeto se pueden crear **instancias de ciertas clases**. Por ejemplo, la clase Organizational-Unit te permite crear instancias de la clase container, pero la clase container no te permite crear instancias de la clase Organizational-Unit.



Posibles instancias de clases que se pueden crear dentro del container "Users"

Users es una instancia de la clase container y no de una clase especial para usuarios, por lo que dentro de esta instancia se pueden crear más que sólo usuarios como se ve en la anterior imagen.

También podemos ver de qué padres se puede crear una determinada clase con el atributo **PossSuperiors** (lista de objetos que pueden contener dicha clase) o **systemPossSuperiors** (lista de clases que pueden contener dicha clase, la lista sólo puede ser modificada por el sistema):

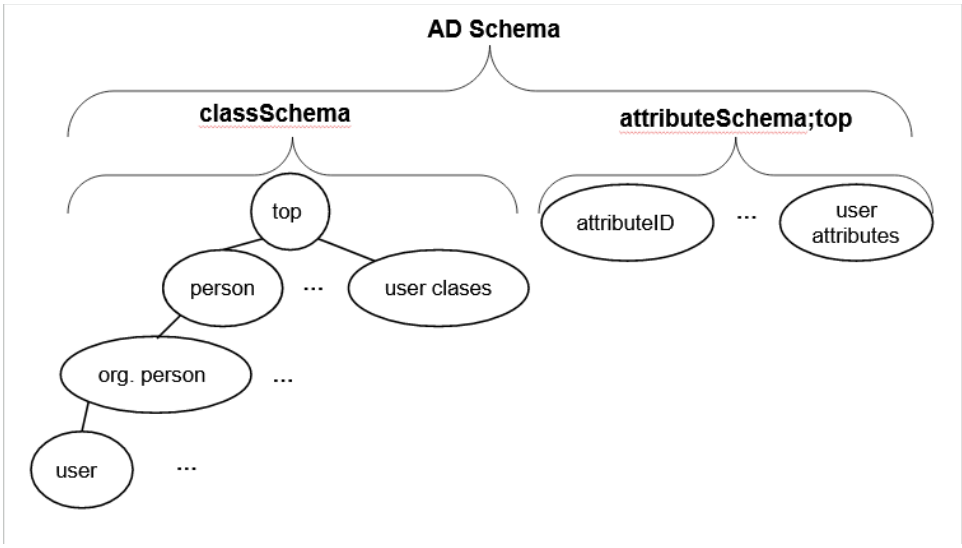


Se pueden crear instancias de la clase account por medio de una OU (las carpetas con prefijo OU) o de un container (las carpetas con prefijo CN).

ATRIBUTOS EN EL ESQUEMA (attributeSchema Objects):

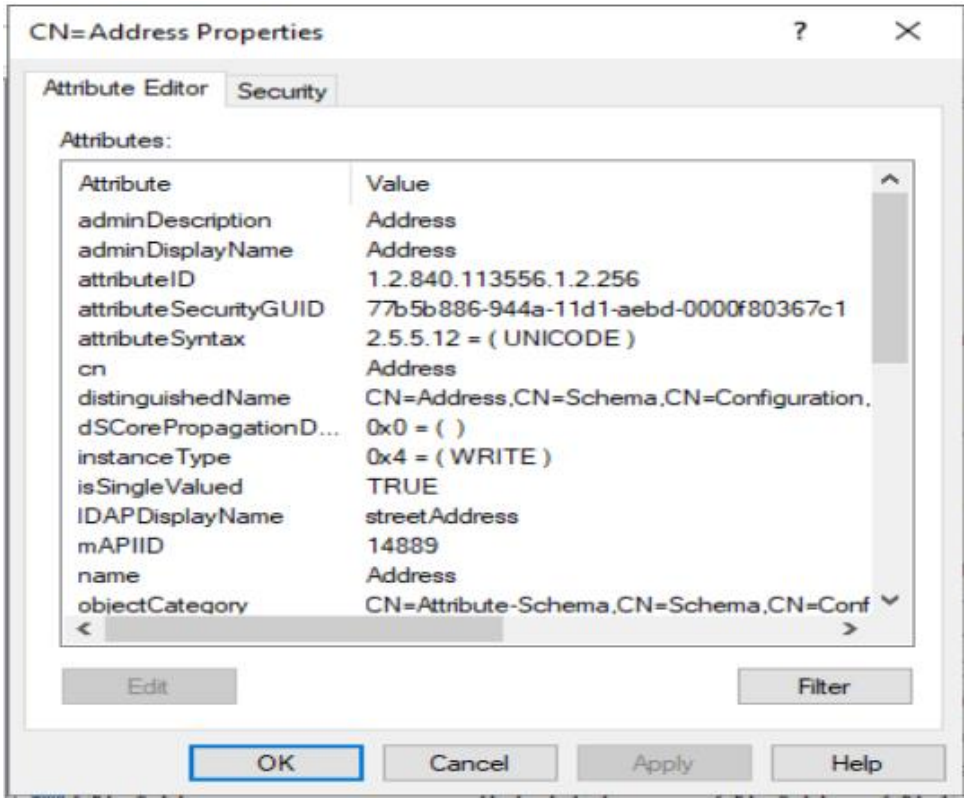
Los atributos son ítems de datos que en conjunto definen un objeto de una clase, residen separados de las clases, ya que un atributo se puede aplicar a varias clases.

Los objetos en el esquema que son atributos son instancias de la clase **attributeSchema.** Aunque todos estos objetos heredan atributos de la clase **top y attributeSchema**, los atributos no permiten **herencias.**



Atributos dentro del attributeSchema

Los atributos están compuestos de otros atributos:



Atributos del atributo "Address"

El attributeID es igual al **governsID**, pero este es propio de los atributos.

Todos estos atributos con un valor asignado son **instancias** de dichos atributos. Por ejemplo, si queremos crear un nuevo atributo en el esquema siempre hay que asociarle un attributeID en el momento de la creación, entonces al momento de crear el atributo **estamos creando una instancia de tipo attributeID** con el valor que le asignamos (aparte de otras instancias de atributos que también se crean: cn, attributeSyntax, systemOnly...).

Los atributos tienen una serie de **propiedades** que es importante conocer, debido a que afectan al **uso y a la funcionalidad** de los mismos. A continuación, se enseñan de forma resumida los más importantes:

SINTAXIS DE UN ATRIBUTO:

La sintaxis es el **tipo de dato** que un atributo puede contener, las sintaxis no están guardadas como objetos en el esquema, sino que están codificadas directamente en Active Directory, por lo que **no se pueden crear nuevas sintaxis**.

Al crear un nuevo atributo debes indicar su sintaxis, para ello debemos introducir el OID de la sintaxis y el **OM syntax**, esto es porque más de una sintaxis tienen el mismo OID.

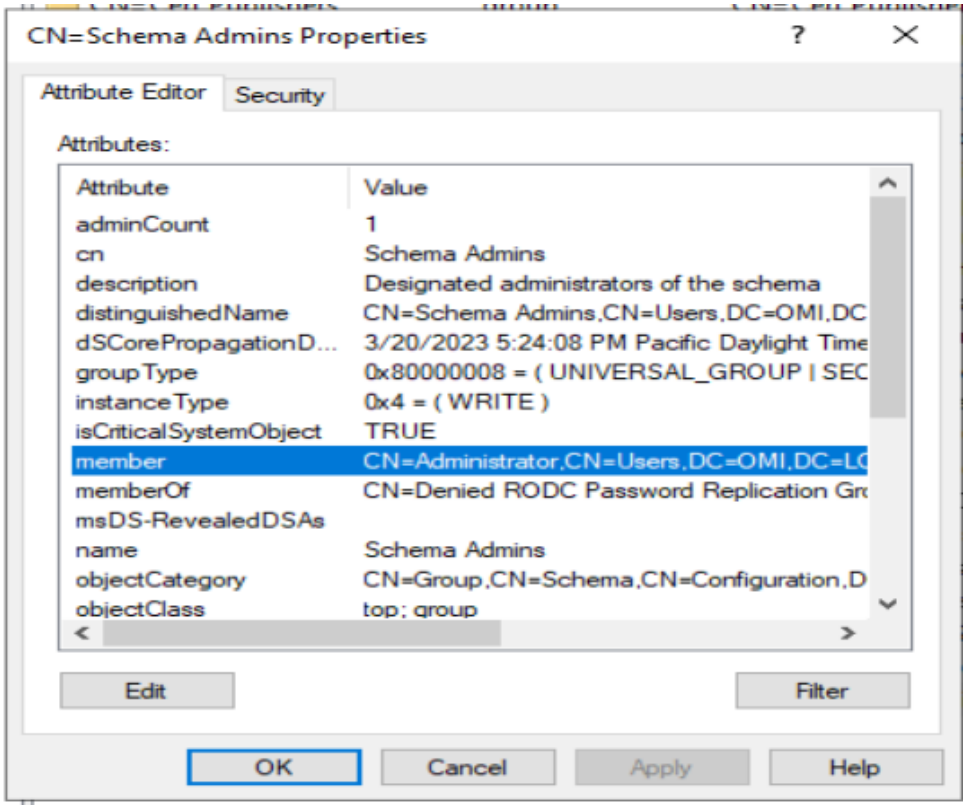
Syntax	OID	OM syntax	Description
Address	2.5.5.13	127	Used internally by the system.
Boolean	2.5.5.8	1	True or false.
Case-insensitive string	2.5.5.4	20	A string that does not differentiate between uppercase and lowercase.
Case-sensitive string	2.5.5.3	27	A string that differentiates between uppercase and lowercase.
Distinguished name	2.5.5.1	127	The fully qualified domain name (FQDN) of an object in Active Directory.
DN-Binary	2.5.5.7	127	Octet string with binary value and DN. Format: *B:<char count>:<binary value>:<object DN>*.
DN-String	2.5.5.14	127	Octet string with string value and DN. Format: *S:<char count>:<string value>:<object DN>*.
Generalized-Time	2.5.5.11	24	ASN1.1 time format, e.g., 20040625234417.0Z.
Integer (enumeration)	2.5.5.9	10	A 32-bit number.
Integer (integer)	2.5.5.9	2	A 32-bit number.
Large integer	2.5.5.16	65	A 64-bit number.
NT security descriptor	2.5.5.15	66	A security descriptor (SD).
Numeric string	2.5.5.6	18	A string of digits.
Object ID	2.5.5.2	6	An OID.
Octet string (Octet-String)	2.5.5.10	4	A byte string.
Print case string (IA5-String)	2.5.5.5	22	A normal printable string.

Tabla con algunas sintaxis

ATRIBUTOS ENLAZADOS (LINKED ATTRIBUTES):

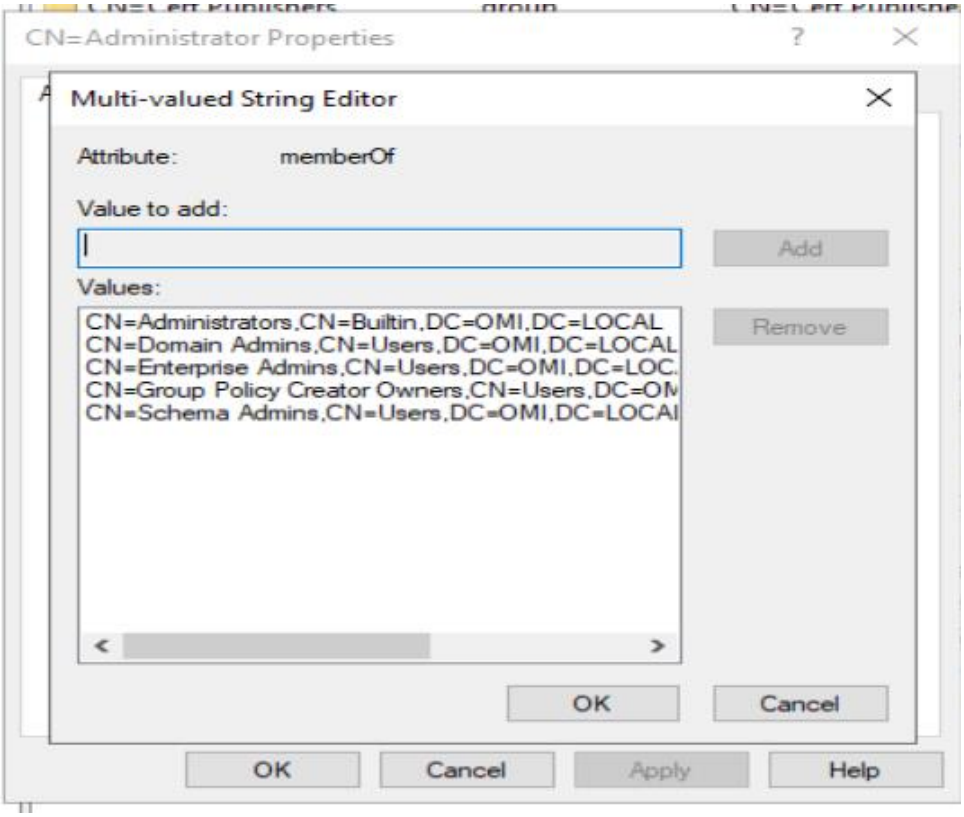
Estos atributos definen la **relación entre objetos** en AD. Un ejemplo sería la relación entre un usuario y el grupo al que pertenece. Existen dos tipos de enlaces:

-**Forward link**: siguiendo el anterior ejemplo, un forward link nos permite saber la lista de usuarios que pertenecen a un grupo.



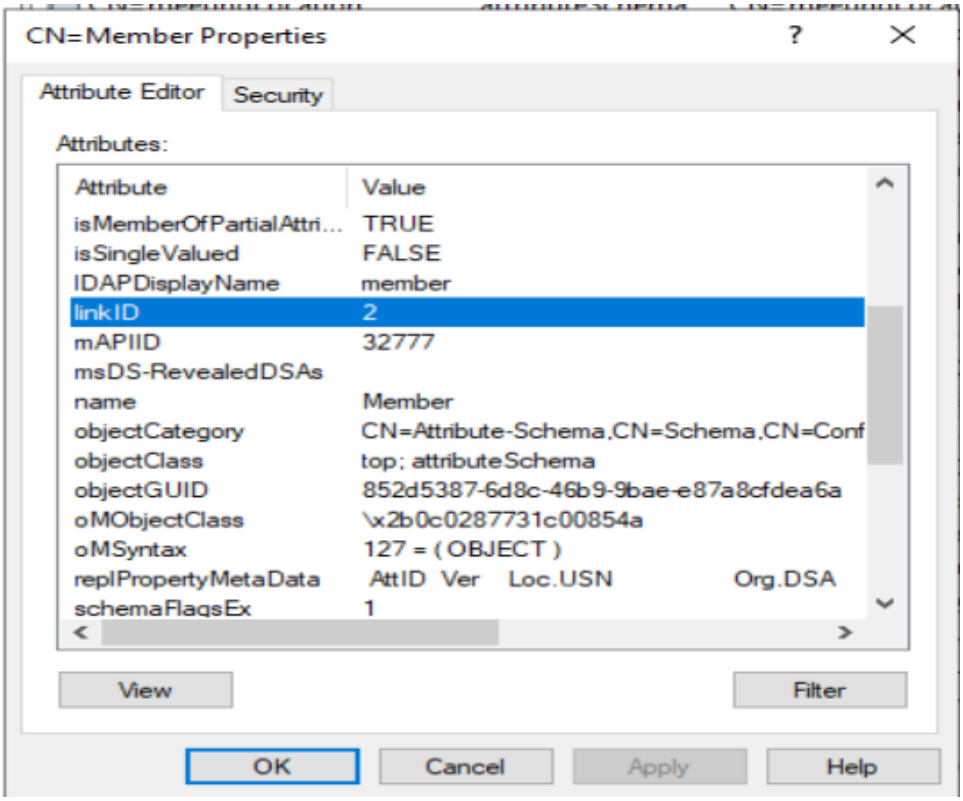
El atributo member es exclusivo de la clase group y guarda la lista de usuarios que pertenecen a un grupo.

-**Back link**: este enlace en el mismo ejemplo nos permite saber a qué grupos pertenece un usuario.



memberOf es exclusivo de la clase User y dice a qué grupos pertenece un usuario.

Para identificar de forma unívoca un atributo enlazado existe el **linkID,** asignado en el momento de la creación:



linkID del atributo Member

-Si el atributo es un forward link, su linkID siempre es **par.**

-Si es un back link, su linkID siempre es **impar.** Para asegurarlo, el nº es igual al del forward linkID relacionado pero incrementado en 1.

Teniendo esto en cuenta, el atributo member es un **forward link** y su back link asociado (memberOf) tiene el linkID **2+1=3**.

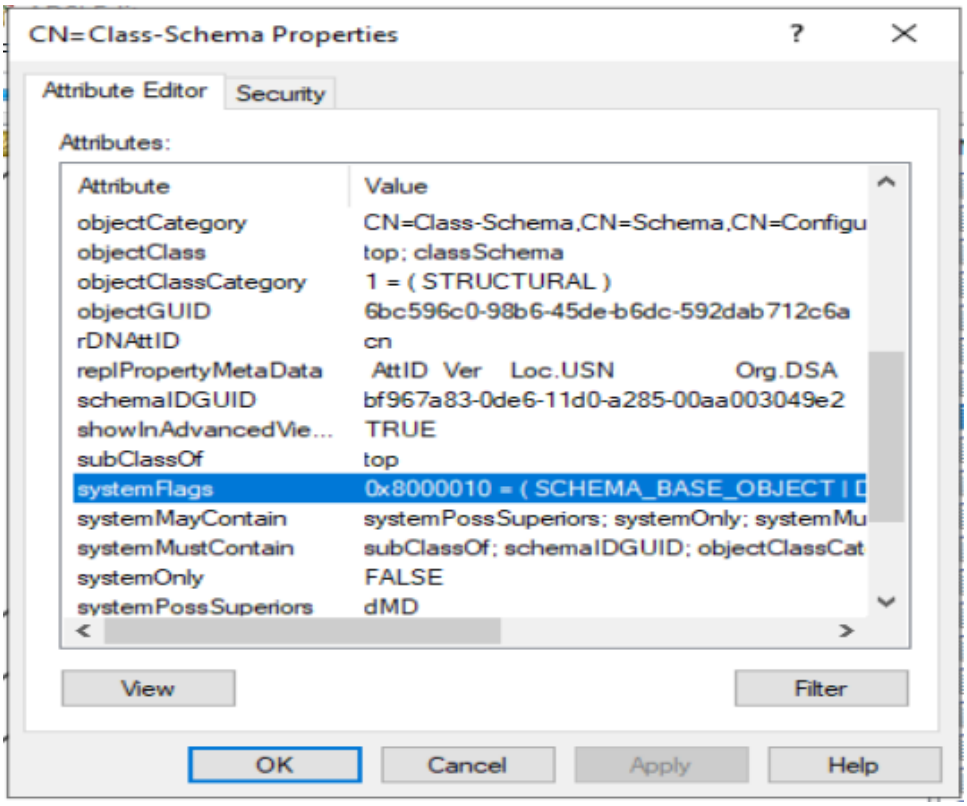
SYSTEMFLAGS:

El systemFlag es un atributo que toma diferentes valores a través de una **máscara de bits,** hay diferentes valores definidos para las systemFlags:

Value	Description
1 (0x0001)	Attribute is not replicated.
2 (0x0002)	Attribute will be replicated to the Global Catalog. This value should only be set by Microsoft; do not use it. Instead, use the `isMemberOfPartialAttributeSet` attribute for adding attributes to the partial attribute set.
4 (0x0004)	Attribute is constructed, not stored in the database. This should only be set by Microsoft; do not use it.
16 (0x0010)	Category 1 attribute or class. Category 1 objects are classes and attributes included in the base schema with the system. Note: not all classes and attributes in the base schema are marked Category 1.
33554432 (0x02000000)	The object is not moved to the Deleted Objects container when it is deleted. Instead, the tombstone remains in the original container.
134217728 (0x08000000)	The schema object cannot be renamed.
268435456 (0x10000000)	For objects in the configuration partition, if this flag is set, the object can be moved with restrictions; otherwise, the object cannot be moved. By default, this flag is not set on new objects created under the configuration partition. This flag can only be set during object creation.
536870912 (0x20000000)	For objects in the configuration partition, if this flag is set, the object can be moved; otherwise, the object cannot be moved. By default, this flag is not set on new objects created under the configuration partition. This flag can only be set during object creation.
1073741824 (0x40000000)	For objects in the configuration partition, if this flag is set, the object can be renamed; otherwise, the object cannot be renamed. By default, this flag is not set on new objects created under the configuration partition. This flag can only be set during object creation.
2147483648 (0x80000000)	The object cannot be deleted.

SystemFlags para clases y atributos

Como se puede ver las systemFlags definen como un atributo o clase puede ser gestionado en Active Directory. El hecho de que sea una máscara de bits permite que un objeto adopte varios valores definidos de las systemFlags.



Un ejemplo es la clase classSchema, que según las systemFlags el objeto no se puede borrar y es de categoría 1 (viene en el esquema por defecto)

SEARCHFLAGS:

También contiene una máscara de bits con los posibles valores, este atributo sólo se aplica en **la definición de los atributos** del esquema.

Bit number	Value	Description
1	1 (0x0001)	Create an index for the attribute. All other index-based flags require this flag to be enabled as well. Marking linked attributes to be indexed has no effect.
2	2 (0x0002)	Create an index for the attribute in each container. This is only useful for one-level LDAP queries.
3	4 (0x0004)	Add an attribute to ambiguous name resolution (ANR) set. ANR queries are primarily used for Exchange and other address book tools. ANR attributes must be indexed and must be in either UNICODE or Teletex string attribute syntax. Adding attributes to this set can have performance implications in Microsoft Exchange.
4	8 (0x0008)	Preserve this attribute in a tombstone object. This flag controls what attributes are kept when an object is deleted.
5	16 (0x0010)	Copy this value when the object is copied. This flag doesn't do anything in Active Directory; tools such as Active Directory Users and Computers that copy objects can look at this flag to determine what attributes should be copied.
6	32 (0x0020)	Create a tuple index. Tuple indexing is useful for medial searches. A medial search has a wildcard at the beginning or in the middle of the search string. For example, the medial search (`drink=*coke`) would match Cherry Coke, Diet Coke, etc.
7	64 (0x0040)	Create a subtree index. This index is designed to increase the performance of virtual list view (VLV) queries.
8	128 (0x0080)	Mark the attribute as confidential. Only users with both Read Property *and* Control Access rights to the attribute so marked can view it when it is so marked.
9	256 (0x0100)	Never audit changes to this attribute. This flag was new in Windows Server 2008. Auditing is covered in Chapter 16.
10	512 (0x0200)	Include this attribute in the RODC filtered attribute set. RODCs and the filtered attribute set are covered in Chapter 9.

searchFlag bits

Las searchFlags se encargan de gestionar **índices**, que permiten hacer consultas más eficientes (no siempre). El atributo abarca más utilidades aparte de los índices:

-ANR (Ambiguous Name Resolution): filtro que permite expandir una simple petición a una petición de múltiples atributos deseados por el administrador y que estén soportados por ANR. Se suele utilizar para encontrar objetos relacionados con usuarios.

```
(
(|
  (displayName=joe richards*)
  (givenName=joe richards*)
  (legacyExchangeDN=joe richards)
  (msDS-AdditionalSamAccountName=joe richards*)
  (msDS-PhoneticCompanyName=joe richards*)
  (msDS-PhoneticDepartment=joe richards*)
  (msDS-PhoneticDisplayName=joe richards*)
  (msDS-PhoneticFirstName=joe richards*)
  (msDS-PhoneticLastName=joe richards*)
  (physicalDeliveryOfficeName=joe richards*)
  (proxyAddresses=joe richards*)
  (name=joe richards*)
  (sAMAccountName=joe richards*)
  (sn=joe richards*)
  (&
    (givenName=joe*)
    (sn=richards*)
  )
  (&
    (givenName=richards*)
    (sn=joe*)
  )
)
)
```

Posible filtro de búsqueda de ANR al poner en la petición anr=Joe Richards

-Conservar los atributos en una tumba: cuando un objeto se elimina, conservar datos de algunos de sus atributos. Útil si no se dispone de la papelera de reciclaje. Los atributos enlazados no soportan esta funcionalidad.

-No replicar ciertos atributos a otros DC, concretamente a los Read-Only DC (son utilizados en ramas que no pueden tener un DC completo). Este conjunto de atributos es conocido como **RODC filtered attribute set**.

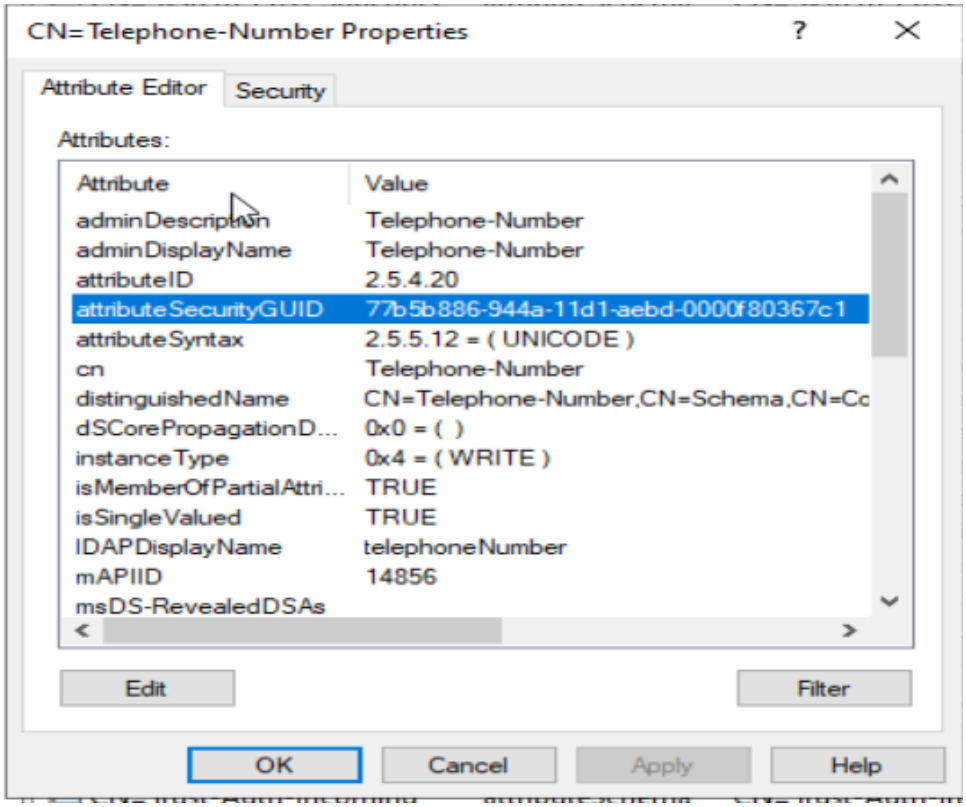
-Marcar un atributo como confidencial: en este caso, para consultar los datos de un atributo necesitas permisos de lectura y de control de acceso.

-Cambio de auditoría: la auditoría de servicios está dividida en 4 subcategorías (acceso, cambios, replicación, replicación detallada), para reducir los problemas de gestión. Puedes elegir entre auditorías o escoger todas o ninguna.

PROPERTY SETS:

Permite asociar un atributo a un conjunto de otros atributos que comparten una propiedad. Un atributo que guarde el nº de teléfono de los empleados pertenecerá al property set de información personal.

El uso de property sets por parte del administrador le permite añadir **solo una regla de control de acceso (ACE)** para un conjunto de atributos, lo que hace más eficiente el otorgar o denegar el acceso a un objeto. El concepto de ACE y ACL se tratará más adelante.



El atributo pertenece a Personal-Information

El identificador de un property set es el **rightsGUID**. A través del atributo attributeSecurityGUID sabemos a qué propertyset pertenece el atributo, porque tiene que coincidir con el rightsGUID.

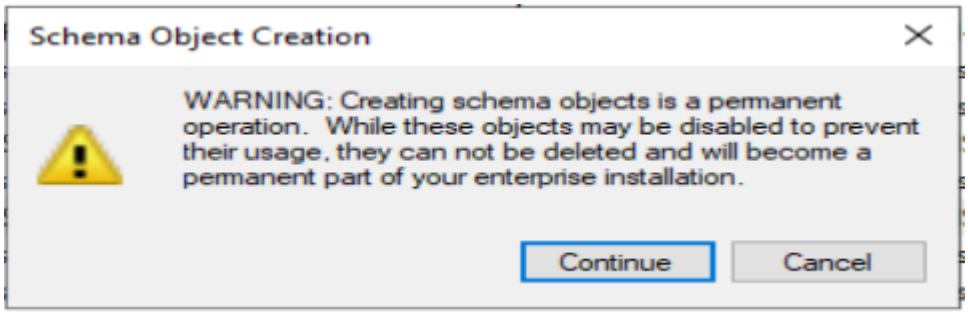
MAPI ID:

Messaging **A**pplication **P**rogramming **I**nterface

es un componente de Windows que permite enviar y recibir diversas peticiones de correo electrónico. Cualquier atributo usado por MAPI tiene un **MAPI ID**.

EXTENDER EL ESQUEMA:

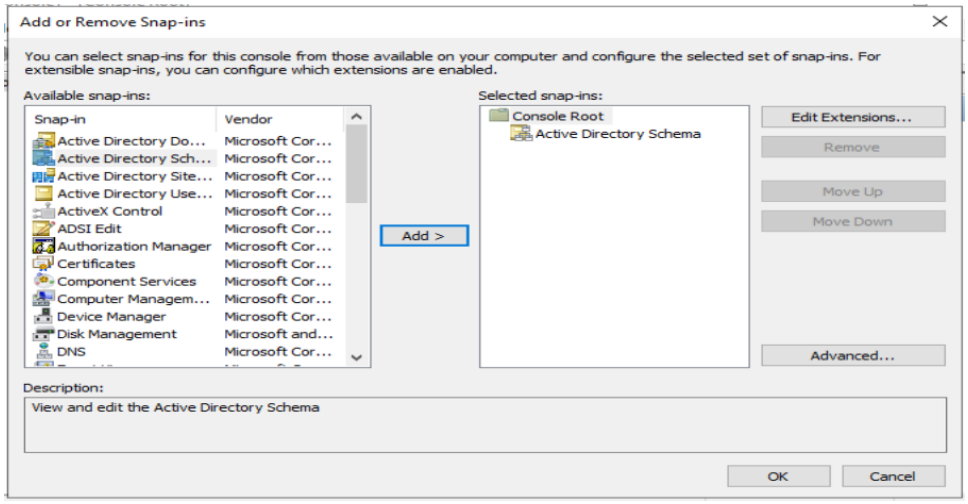
La creación ya sea de un atributo o de una clase es una operación **permanente**, por lo que no se podrá deshacer o destruir del esquema a menos que se restaure una copia anterior del mismo.



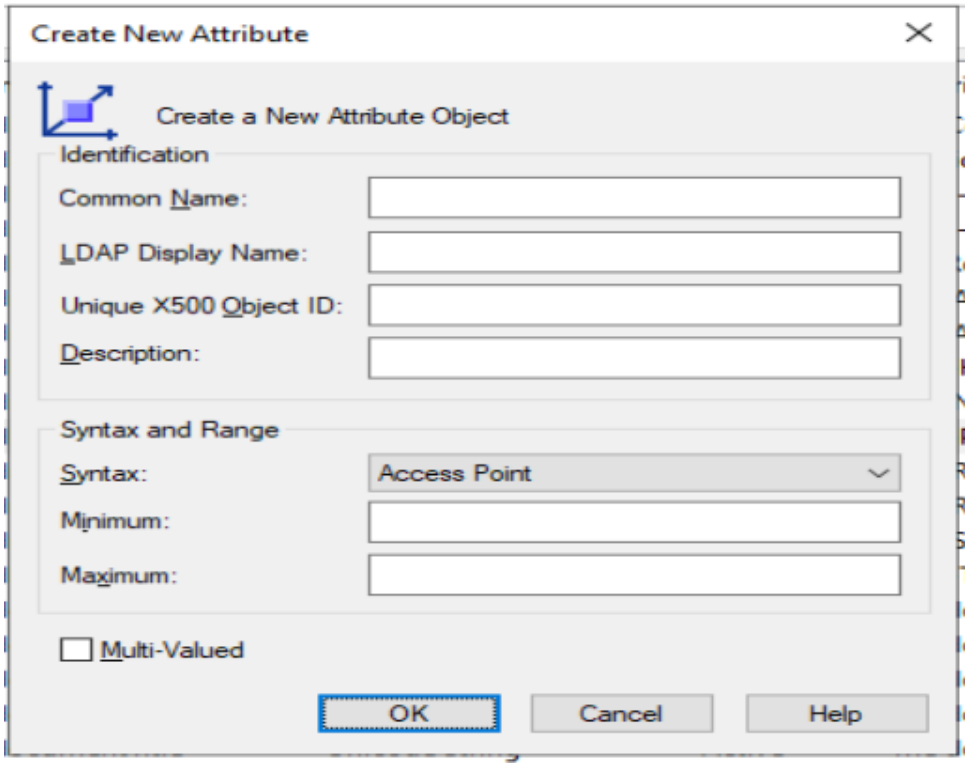
No hay vuelta atrás

Para crear nuevos objetos es recomendable usar el snap-in de AD Schema desde la MMC console. Por defecto no están todas las funcionalidades de la mmc console activadas, para activarlas ejecutar el comando ***RegSvr32 SchmMgmt.dll*** desde una terminal.

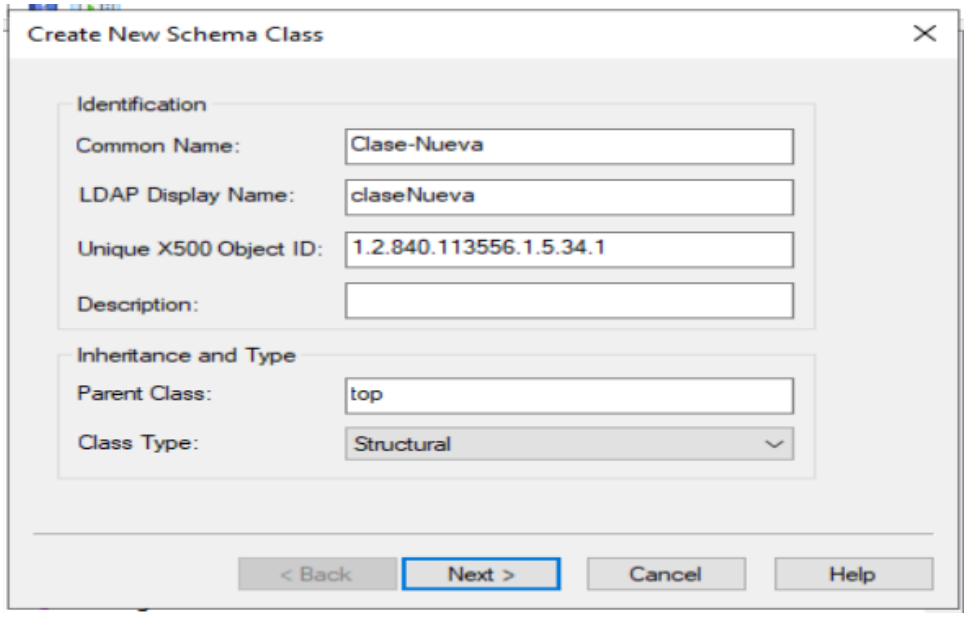
Para entrar a la MMC console, pulsar Windows + R y teclear "mmc". Dentro de la consola, para añadir el snap-in, pulsar Ctrl + M y selecciona el esquema:

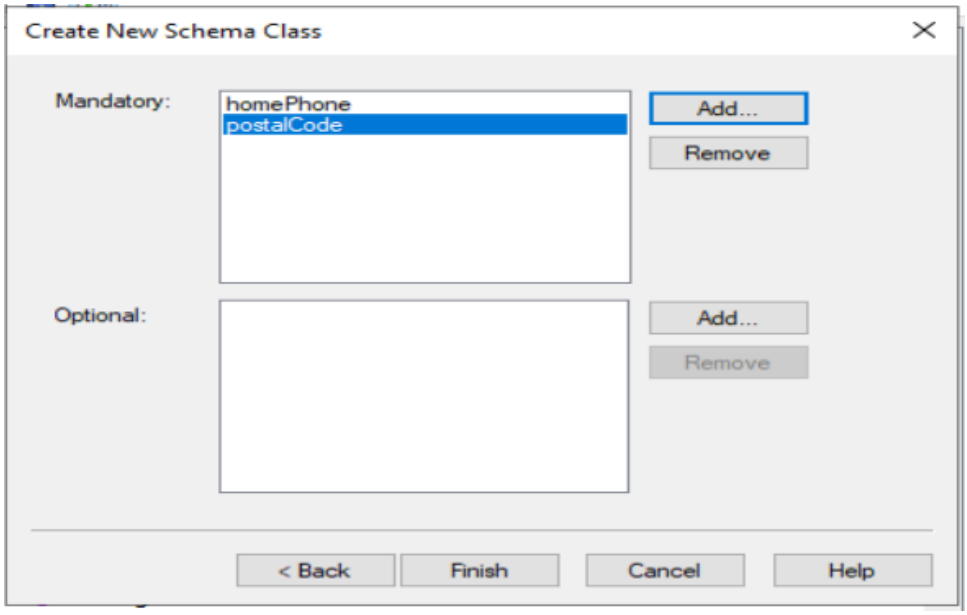


Ya hemos añadido el snap-in a la consola, si lo desplegamos nos saldrán dos containers, las clases y las instancias, desde los que podremos crear nuevos atributos y clases.



Creación de un atributo





Creación de una clase

Infraestructura de red en Active Directory:

En la infraestructura de red de Active Directory tenemos 3 protocolos que destacan sobre los demás: DNS, DHCP e IPAM.

Aparte de estos tres protocolos existe también la virtualización de red Hyper-V, este protocolo proporciona "redes virtuales" (denominadas red de vm) a máquinas virtuales similares a la forma en que la virtualización del servidor (hipervisor) proporciona "máquinas virtuales" al sistema operativo. Lo cual permite la virtualización de las redes de los clientes en una infraestructura de red física compartida. No hablaremos más a profundidad sobre el dado que consideramos necesario para su entendimiento un control y conocimientos sobre otros procesos en una profundidad mayor a la que poseemos actualmente.

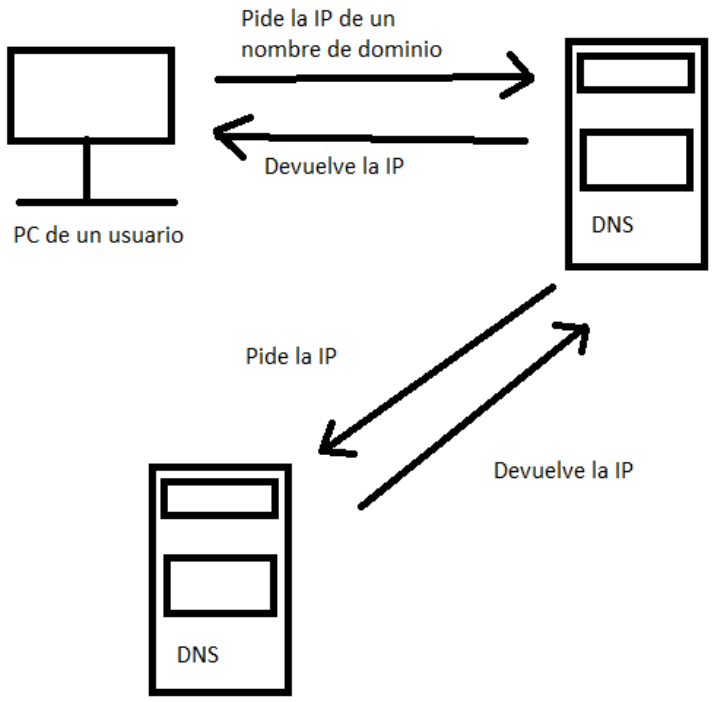
Pasamos entonces a hablar del primero de los tres protocolos, el protocolo DNS.

Domain Name Service o Sistema de Nombres de Dominio (DNS):

DNS es un protocolo de resolución de nombres para redes TCP/IP, como Internet o una red de organización. La gran mayoría de compañías emplean este protocolo para la traducción de sus nombres de dominio a IP e inversa o para alternar su nombre a un alias y que este sea reconocido.

Vamos a dar una breve explicación antes de pasar a sus partes y a como se integra en AD.

Planteando una situación general, donde un usuario hace una búsqueda sobre internet a cierta web, el servidor DNS le dará la IP necesaria para acceder, en caso de que este servidor no tenga la IP, preguntara a otro DNS hasta dar con ella o enviar un error.

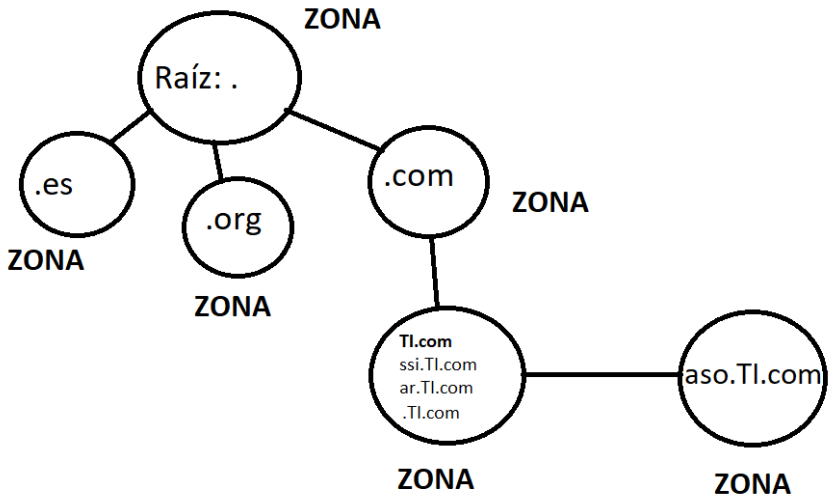


Un DNS tiene una colección de nombres, para tener un orden sobre estos y agilizar los procesos de búsqueda se emplean zonas.

Zonas DNS

Una zona es un conjunto de nombre de dominio, los cuales están diferenciados por áreas que se gestionan en el DNS. El DNS se puede ver como un espacio de nombres jerárquico, el dominio raíz del DNS es la parte superior y cada área compone una rama formada por subdominios.

Vamos a emplear el ejemplo de TI.com. Este dominio tiene 3 subdominios aso.TI.com, ar.TI.com y ssi.TI.com. Vamos a suponer que ar y ssi pueden administrarse de manera conjunta y aso al ser una asignatura especial debe administrarse por separado. Podríamos verlos como en el siguiente esquema:



AD hace uso de los DDNS (Dynamic DNS), un método para que los clientes envíen peticiones a un servidor DNS para añadir o eliminar recursos en una zona. Esto permite agilizar la gestión en el DNS si tenemos un entorno muy grande, ya que anteriormente se debía de modificar un fichero de texto por zona para su control.

Ahora que sabemos que es DNS y para qué sirve en general, pasemos a saber cómo es integrado en AD:

Integración de DNS en AD

AD integra el protocolo DNS para que le sirva como un localizador de Controladores de Dominio para permitir al usuario autenticarse contra este. Veámoslo con un ejemplo:

1. El cliente A estaba situado en un sitio A y fue trasladado a un sitio B.

2. Cuando este cliente inicia sesión este pensará que sigue estando en A, así que trae de su cache la información del DC A y trata de comunicarse con él.

3. El DC A recibe la información, pero se percata que la IP viene de B, por lo que determina que este cliente debería de usar un DC de B.

4. El cliente entonces lanza un DNS lookup para encontrar aquellos DC que trabajan en B

5.El cliente hace contacto con un DC de B y entonces pasará una de estas 3 opciones:

1- El DC de B responde y autentica al cliente de B

2- El DC no responde (está caído), y el cliente entonces intenta contactarse con otro

3- Todos los servicios de B han muerto (asesinados por systemd) fallando en su respuesta y, por lo tanto, el cliente procede a buscar todos los DC que sirven al dominio e intenta autenticarse contra cualquiera de estos.

Pasemos ahora a conocer nuestro segundo proceso, el proceso DHCP.

Protocolo de configuración dinámica de host o Dynamic Host Configuration Protocol (DHCP):

"DHCP es un protocolo es un protocolo cliente-servidor que proporciona automáticamente un host de protocolo de Internet (IP) con su dirección IP y otra información de configuración relacionada."

De una forma simple, DHCP viene a ser un protocolo mediante el cual un servidor DHCP proporciona, de manera directa, una IP a un nuevo cliente cuando este inicia sesión en la red. De esta forma, la implementación de DHCP evita problemas de configuración al hacerlo de manera manual o conflictos con usuarios que empleen un portátil y cambien su localización frecuentemente.

Active Directory no necesita de manera obligatoria un servidor DHCP, a partir de Windows Server 2016 los servidores DHCP son roles de red que pueden seleccionarse

durante la creación del dominio o ser añadidos a posteriori, pero el tenerlo aporta las siguientes ventajas:

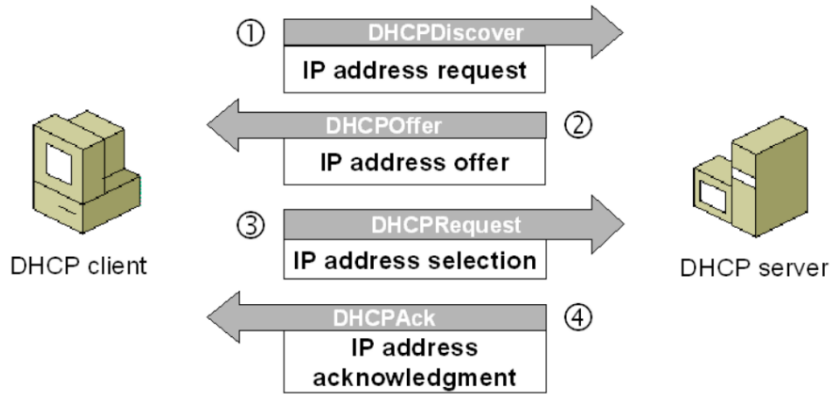
1.Configuración fiable de las IP.

2.Configuración centralizada y automatizada.

3.Control sobre cambios de IP en los clientes.

4.Reenvío de mensajes DHCP, evitando la necesidad de un servidor DHCP por subred.

El protocolo DHCP funciona mediante 4 pasos:



Descubrir: El cliente trasmite un intento de descubrir un servidor DHCP

Ofrecer: El servidor DHCP recibe la petición DHCPDISCOVER del cliente y le ofrece una dirección IP

Solicitar: El cliente solicita al servidor la configuración IP para usarla

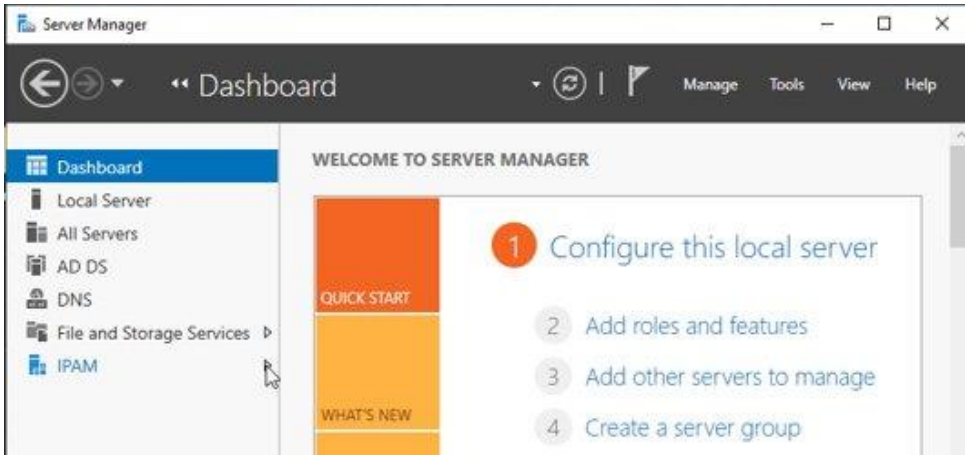
Reconocer: El servidor DHCP otorga la dirección IP y no ofrecerá esa dirección a ningún otro cliente

Pasemos a hablar del último protocolo, IPAM.

El IP Address Management o Administración de direcciones IP (IPAM)

Es simple de entender. Se trata de un proceso mediante el cual podemos administrar y controlar de manera centralizada los protocolos DNS y DHCP en nuestro dominio. En concreto la descripción que da Microsoft sobre el protocolo es:

"La Administración de direcciones IP (IPAM) es un conjunto integrado de herramientas que permite el planeamiento, la implementación, la administración y la supervisión de un extremo a otro de la infraestructura de direcciones IP, con una experiencia de usuario avanzada. IPAM detecta automáticamente los servidores con infraestructura de direcciones IP y los servidores del Sistema de nombres de dominio (DNS) de la red y permite administrarlos desde una interfaz central usando una terminal."



IPAM al igual que DHCP es un rol opcional en la configuración de nuestro AD, que podemos implementar o no.

¿Ahora bien, si solo el protocolo DNS es "obligatorio" entonces que tipo escenarios de redes son admitidos para construir un entorno de AD en Windows Server?

Escenarios de red admitidos en Windows Server:

1.Escenario de redes definidas por software (SDN).

2.Escenario de virtualización de funciones de red (NFV)

3.Escenarios de conmutador virtual de Hyper-V

4.Escenario de servidor DNS

5.Escenario de IPAM con DHCP y DNS

6.Escenarios de formación de equipos NIC

7.Escenarios se Switch Embedded Teaming (SET)

Seguridad en Active Directory:

La seguridad en Active Directory podemos diseccionarla en 3 partes:

1. **Como puede un usuario probar que es quien dice ser (Autenticación)**

2. **Que tiene permitido hacer como usuario o que funciones puede emplear y de qué manera un equipo (Política de grupos)**

3. **A que ficheros puede acceder y que permisos tiene para su uso (ACLs)**

Empezaremos hablando de las Políticas de grupo en AD.

Política o Directiva de grupo en AD:

Las políticas de grupo en AD determinar el comportamiento y privilegios de un usuario o un ordenador dentro del dominio. Se usan principalmente para fortalecer la seguridad en el entorno.

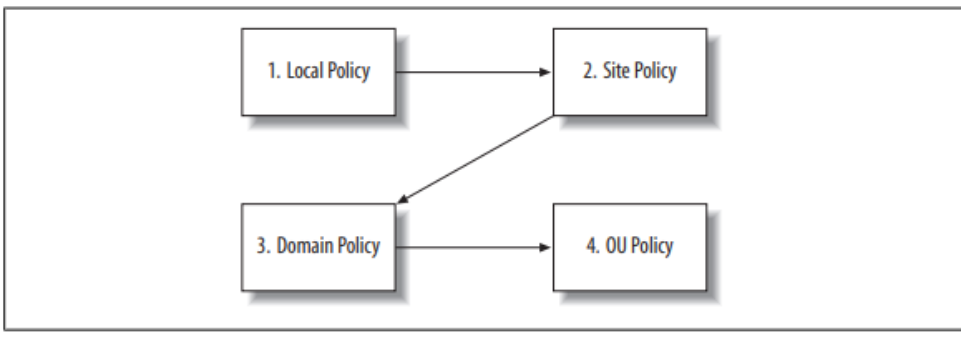
Los administradores configuran y crean objetos de directivas de grupo (GPO) que luego enlazaran con cuatro tipos de contenedores:

-**Local:** Las políticas que implementa Windows al iniciar, sin importar que usuario este iniciando, luego de esto podrán ser sobrescritas por cualquiera de los demás niveles al iniciar la sesión si se establece como tal.

-Sites: redes IP que definen la estructura física de AD.

-Dominios: los dominios dentro de nuestro AD.

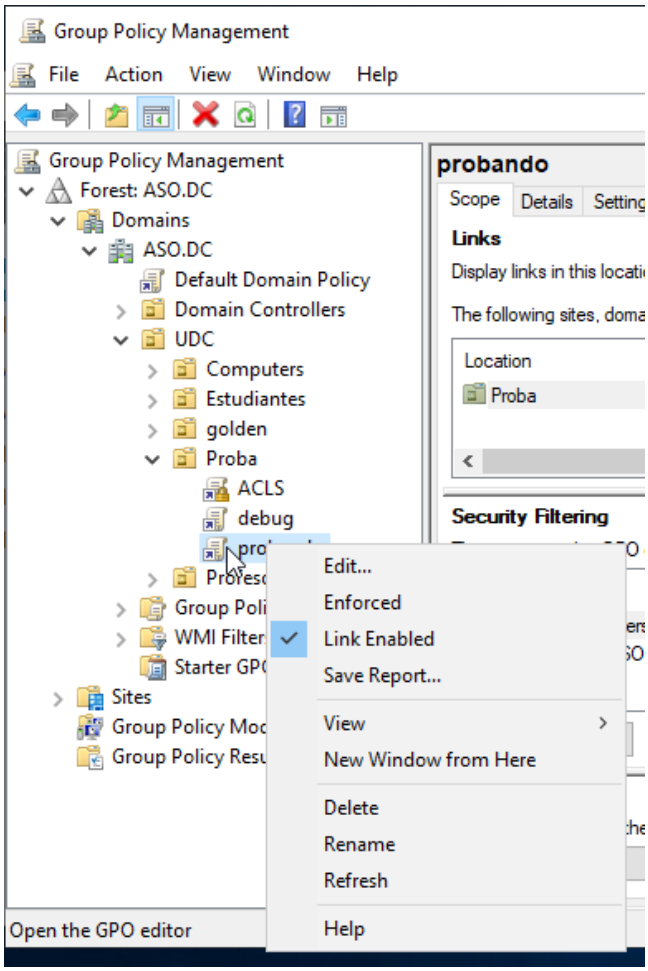
-Unidades Organizativas (OU): "Contenedores" donde se almacenan usuarios u ordenadores para la aplicar de manera controlada las GPO, es el nivel más bajo en la escala de aplicación de políticas de seguridad.



Estos niveles de mando son así por defecto, pero si tenemos una política de grupo dentro de una OU, esta bloqueara la herencia de cualquier política que modifique

características de usuario (EJ: Fondo de pantalla, accesos directos...), y fusionara otras que pueden ser comunes dentro de las referentes a los ordenadores (EJ: Plan de gestión de batería) siempre y cuando la GPO de un nivel superior no sea forzada, en cuyo caso está ultima primara por sobre las demás. Las políticas locales son las únicas que no pueden forzar el no heredar valores, sería algo contraproducente usar un dominio y forzar políticas locales.

Group Policy Objects (GPO): es un conjunto de políticas de seguridad que actúan sobre ordenadores o usuarios para determinar sus privilegios y capacidades. Son creadas a través del Editor de objetos de políticas de grupo en AD.

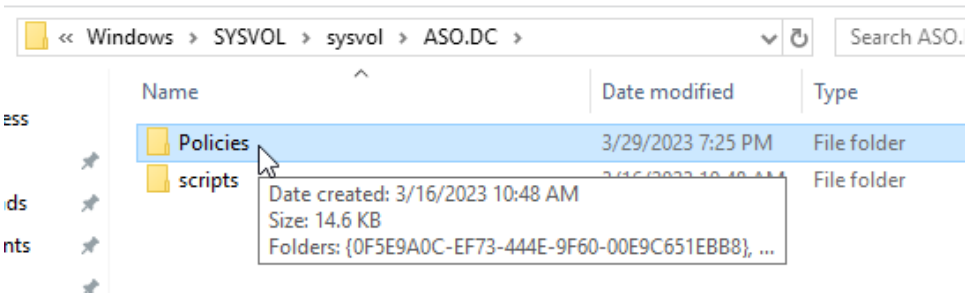


Donde se almacenan las GPO en AD:

Las GPO están almacenadas en dos sitios diferenciados:

-**Contenedor de políticas de grupo (GPC):** un objeto que contiene información como puede ser el nombre de la GPO, sus ACLs, la información de su versión o su estado actual. Se almacenan el el propio AD en el contenedor CN=Policies o CN=System.

-**Plantilla de políticas de grupo (GPT):** es un fichero en el directorio SYSVOL de nuestro sistema, su contenido se replica a todos los demás controladores de dominio que tengamos, aquí se guardan plantillas administrativas, scripts relacionados con las GPO, configuraciones de seguridad, información de software disponible para su instalación...

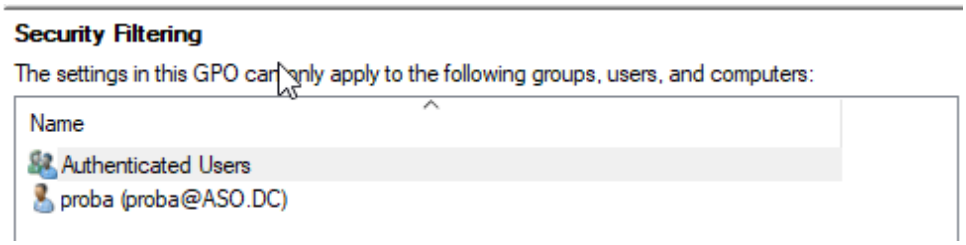


En nuestros ordenadores con Windows podemos ver las políticas de seguridad de la misma manera que en un AD, solo debemos de buscar por el "Editor de Directivas de Grupo Local".

¿Como podemos manejar las políticas de grupo de una manera rápida y sencilla?

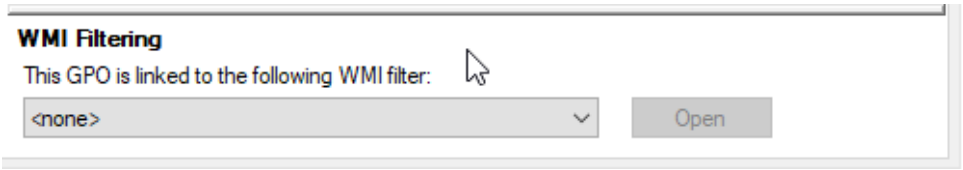
Microsoft proporciona la posibilidad de aplicar filtros en nuestras GPO para hacer su administración más sencilla. Por defecto si creamos una GPO en cualquier nivel, esta será aplicada a todos los usuarios contenidos en este nivel.

Existes 2 filtros principales en las GPO:

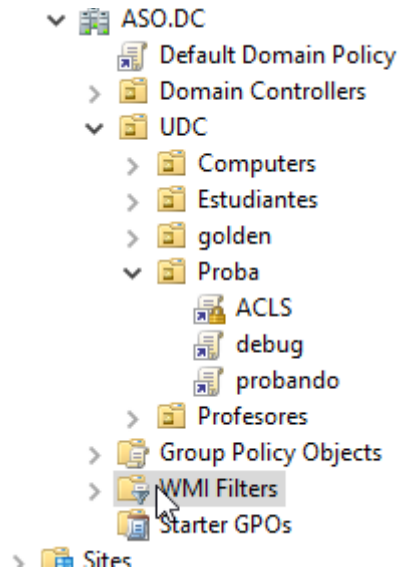


Opción para añadir un filtro Security dentro de la GPO

Security Filter: Antes de Windows Server 2016, incluido este, se podía decidir mediante los filtros de seguridad (Security Filters) eliminar la opción de que todos los usuarios autenticados se viesen afectados por la política de grupo y poder decidir nosotros quien, y como se vería afectado, grupos o usuarios. A partir de Windows Server 2019, está función ya no está, podemos añadir usuarios específicos al filtro, pero no tendrá ningún efecto diferente a restringir los permisos de todos los usuarios para los que se diseñe esta política. Podemos seguir usando el filtro de la misma forma que en versiones anteriores si revocamos el permiso de "Autenticated Users" en el apartado de "Delegations" de la GPO que permite que se aplique la política de grupo sobre él, pero esto no es recomendable.



Opción de selección del filtro WMI dentro de la GPO



Carpeta de configuración de los WMI filters

WMI filter: los filtros WMI permiten al administrador seleccionar ordenadores específicos que cumplan las características del filtro y aplicar la GPO correspondiente, especificando una query como puede ser:

*select * from Win32_OperatingSystem WHERE Version like «6.1%» AND ProductType=»2"*

Opciones disponibles para crear la query:

Tipo de producto:

ProductType 1 = Desktop OS

ProductType 2 = Server OS - Solo Domain Controller

ProductType 3 = Server OS - Excepto Domain Controller

Versiones de sistema operativo:

Windows Server 2019, Windows Server 2016 y Windows 10 = 10.%

Windows Server 2012 R2 y Windows 8.1 = 6.3%

Windows Server 2012 y Windows 8 = 6.2%

Windows Server 2008 R2 y Windows 7 = 6.1%

Windows Server 2008 y Windows Vista = 6.0%

Windows Server 2003 = 5.2%

Windows XP = 5.1%

Windows 2000 = 5.0%

Tipo de arquitectura:

32 bits = Win32_Processor where AddressWidth="32" // OSArchitecture="32-bit"

64 bits = Win32_Processor where AddressWidth="64" // OSArchitecture="64-bit"

Vídeo implementando una política de grupo: politicas.mp4

Pasemos a hablar ahora de las listas de control de acceso y su importancia a nivel de seguridad.

Acces Control List o Listas de control de acceso (ACL):

Son un conjunto de entradas de control de acceso (**ACE**), estas controlan el acceso a nuestros archivos especificando quien es un usuario confiable y quien no lo es para poseer ciertos privilegios, como lectura, escritura, etc.

Todos los ficheros de nuestro sistema hacen uso de las ACLs, para dividir los privilegios de un usuario, un administrador y el SYSTEM (el usuario con mayores privilegios en Windows).

Las ACLs se pueden dividir en 2 partes, DACL y SACL:

Listas de control de acceso discrecional (DACL): En un archivo, identifica a grupos o usuarios sobre los que se aplican las ACEs correspondientes para cada uno. Si el archivo no contiene una DACL se permite el acceso a todo el mundo y si el archivo tiene una DACL, pero no ACEs , se deniega a todo el mundo.

Lista de control de acceso al sistema (SACL): permite a los administradores registrar intentos de acceso a un archivo. Cada ACE puede auditar el tipo de intento de acceso al archivo, si se denegó, si se permitió o ambos. De esta manera un administrador puede llevar cuenta de quien y como accede a que.

Existen 3 tipos de ACEs:

- **ACE de denegación de acceso:** Se usan con las DACL para expresar que un grupo no puede acceder - **ACE de permiso de acceso:** Se usan con las DACL para expresar que un grupo tiene acceso

- **ACE de auditoría del sistema**: Se usan con la SACL para generar logs cuando un usuario o grupo intenta acceder

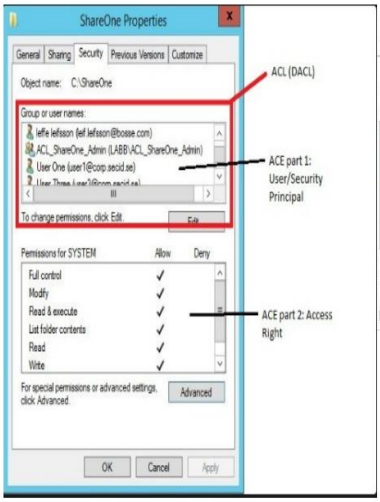
Cada ACE está conformada por 4 componentes:

-**Identificador de seguridad (SID)** que identifica el usuario o grupo de confianza al que se aplica la ACE.

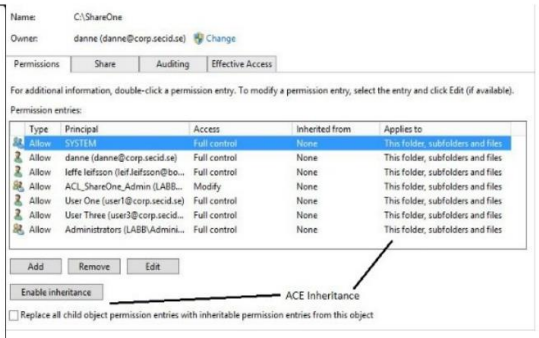
-Una máscara de acceso que especifica los derechos de acceso controlados por la ACE.

-**Un flag** que indica el tipo de ACE.

-**Un conjunto de marcas de bits** que determinan si los contenedores u objetos secundarios pueden heredar la ACE del objeto principal al que está asociada la ACL.



Opciones estándar



Opciones Avanzadas

Sabiendo que es y para qué sirven las ACLs, entendemos que son necesarias a nivel de seguridad para impedir que cualquier usuario haga lo que quiera con un archivo o ejecute cualquier programa desde algún lugar al que no debería de tener acceso. Las ACLs se pueden configurar de manera manual accediendo al apartado de seguridad de un archivo en sus propiedades, o especificándolas en AD haciendo uso de las GPO (aunque esto último da problemas a veces). Lo recomendable en AD es modificar una carpeta de manera manual en la cual se hereden las ACLs a su contenido para poder compartir la carpeta con todos en el dominio.

Pasemos ahora a hablar de los métodos de autenticación en AD.

Autenticación:

Cuando nombramos a la autenticación, nos referimos a un procedimiento mediante el cual un usuario mediante el uso de medios de verificación comprueba que en verdad es quien dice ser y accede a su información.

En AD existes varios métodos de autenticación, nosotros hablaremos de NTLM, Kerberos y LDAP.

En primer lugar, hablaremos del protocolo NTLM.

Net New Technology LAN Manager (net-NTLM):

Primero hemos de aclarar que:

Net-NTLM y NTLM no son lo mismo. NTLM es una función hash que emplea Microsoft para transformar sus contraseñas y almacenarlas, net-NTLM es un protocolo de autenticación en red que funciona mediante desafío-respuesta. Es necesaria esta aclaración debido a que Microsoft nombra a los dos de igual manera, NTLM, sin hacer ningún tipo de distinción. Aquí hablaremos de net-NTLM empleando solamente "NTLM", haremos aclaraciones si es necesario.

Empecemos entonces a entender este protocolo. NTLM fue el primer protocolo de autenticación empleado por Microsoft y actualmente está obsoleto, aunque por motivos de compatibilidad podemos seguir implementándolo.

El protocolo NTLM se basa en un proceso de desafío-respuesta por el cual el cliente se comunica con el servidor y el servidor con el DC para completar el proceso de autenticación. Veamos los pasos:

1. El usuario le comparte a nuestra máquina cliente su usuario, contraseña y nombre de dominio. La máquina cliente crea un hash de la contraseña que el usuario le pasa y borra la contraseña original.
2. El cliente envía una versión en texto plano del nombre de usuario al servidor de aplicaciones
3. El servidor genera un numero aleatorio de 8 bytes, esto es el llamado desafío.
4. El cliente encripta el desafío con el hash creado en el paso 1 y se lo devuelve al servidor, esto es la respuesta
5. El servidor envía 3 objetos al DC:
    a. El nombre de usuario
    b. El desafío
    c. La respuesta
6. El DC usa el nombre de usuario para obtener la contraseña asociada a este de la base de datos. Usa el hash que recibe para encriptar el desafío
7. El DC compara el resultado obtenido del paso 6 y la respuesta del 4, si son idénticos la autenticación es exitosa y si no se le comunicará al usuario por medio de un mensaje de error.

NTLM se considera obsoleto debido a que emplea funciones hash MD4, consideradas poco eficientes en la actualidad y haciéndolo vulnerable a ataques de fuerza bruta para crackear el hash y obtener la contraseña o ataques "Pass The Hash", un ataque en el cual si se conoce el hash NTLM (función hash) de un usuario

se pueden ejecutar comandos o abrir un shell en el equipo de ese usuario sin problema de autenticación.

NTLM tiene dos versiones, NTLM v1 y v2, la v1 funciona exactamente como se describe arriba, la v2 solamente incluye una nueva característica: NTLM v2 emplea marcas de tiempo que no deben de ser superiores a 30 minutos entre cliente, servidor y DC para que la autenticación sea exitosa. Las dos versiones caen en las mismas vulnerabilidades.

Hablemos ahora del sustituto actual de NTLM y proceso de autenticación por defecto en AD, Kerberos.

KERBEROS:

Kerberos es un sistema de autenticación basado en el uso de tickets. En el protocolo de kerberos tenemos 3 actores principales:

1. Un cliente que quiere autentificarse
2. Un servicio al que el cliente quiere acceder
3. Un centro de distribución de claves o tickets (KDC).

Vamos a explicar primero que es el KDC para entender mejor cómo funciona el protocolo.

El KDC es el servicio encargado de distribuir los tickets a los clientes. Este servicio se divide en dos partes:

1. Servicio de autentificación (AS)
2. Servicio de concesión de tickets (TGS)

El AS es el encargado de validar la identidad del usuario y devolverle un ticket TGT (Ticket Granting Ticket) que demuestra la identidad del usuario en este dominio, este ticket permite la obtención de los TGS. El TGS es el encargado de dar al usuario un TGS (Ticket Grating Service) lo cual le brinda acceso al servicio solicitado.

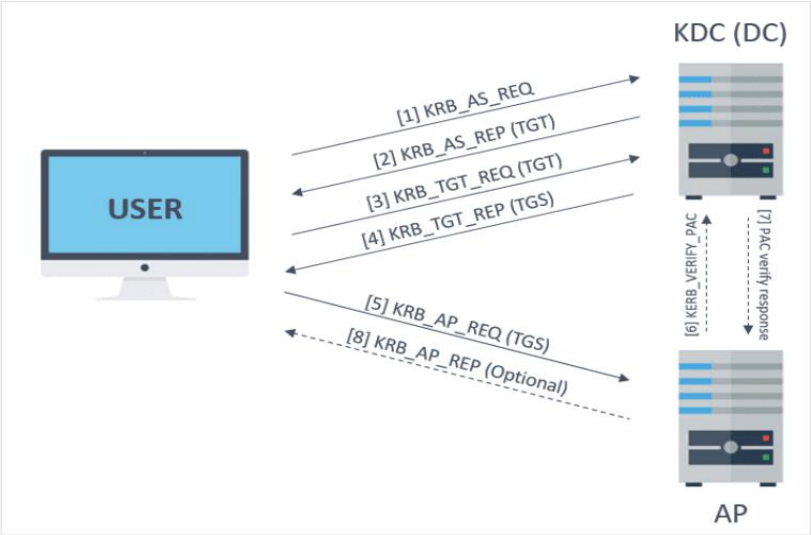
Los tickets empleados por el KDC tienen un tiempo de validez, si este se sobrepasa se deberá de renovar o volver a pasar por el proceso de autenticación, además de esto, los tickets tienen dentro de si toda la información necesaria para identificar al usuario, sus permisos y servicios de los que puede hacer uso.

Que el ticket sea válido no implica que el usuario tenga autorización para acceder a los servicios, todo dependerá de sus privilegios.

Pasemos entonces a ver como es el proceso de autenticación:

En este proceso intervienen el usuario, el KDC y el servidor de aplicaciones (donde están expuestos los servicios a los que un usuario puede acceder). Se enviarán una serie de mensajes:

1. KRB_AS_REQ: Mensaje que emplea el usuario para solicitar el TGT al KDC empleando sus credenciales:
    a. Un timestap cifrado con la contraseña
    b. Nombre de usuario
    c. El identificador del servicio al que quiere acceder (SPN)
    d. Un numero nonce generado por el usuario (evita ataques replay)
2. KRB_AS_REP: Respuesta del KDC para enviar el TGT al usuario:
    a. Ticket TGT:
        i. Nombre de usuario
        ii. Clave de sesión
        iii. Expiración del TGT
        iv. PAC con los privilegios del usuario
    b. Datos cifrados con la clave del usuario
3. KRB_TGS_REQ: El usuario emplea el TGT para solicitar el TGS al KDC, envía:
    a. Datos cifrados con la clave
    b. TGT
    c. SPN del servicio solicitado
    d. Un nonce
4. KRB_TGS_REP: Respuesta del KDC para enviar el TGS solicitado al usuario, envía:
    a. Nombre de usuario
    b. TGS:
        i. Clave de sesión del servicio
        ii. Nombre de usuario
        iii. Expiración del TGS
        iv. PAC con los privilegios del usuario
    c. Datos cifrados con la clave del usuario
5. KRB_AP_REQ: El usuario se identifica contra el servicio usando el TGS:
    a. TGS
    b. Datos cifrados



Resumen de los mensajes de Kerberos en funcionamiento

Kerberos también tiene fallas de seguridad y la mayoría de los ataques se realizan enfocándose en el proceso de obtención de tickets, como pueden ser los ataques "Golden Ticket" y "Silver Ticket".

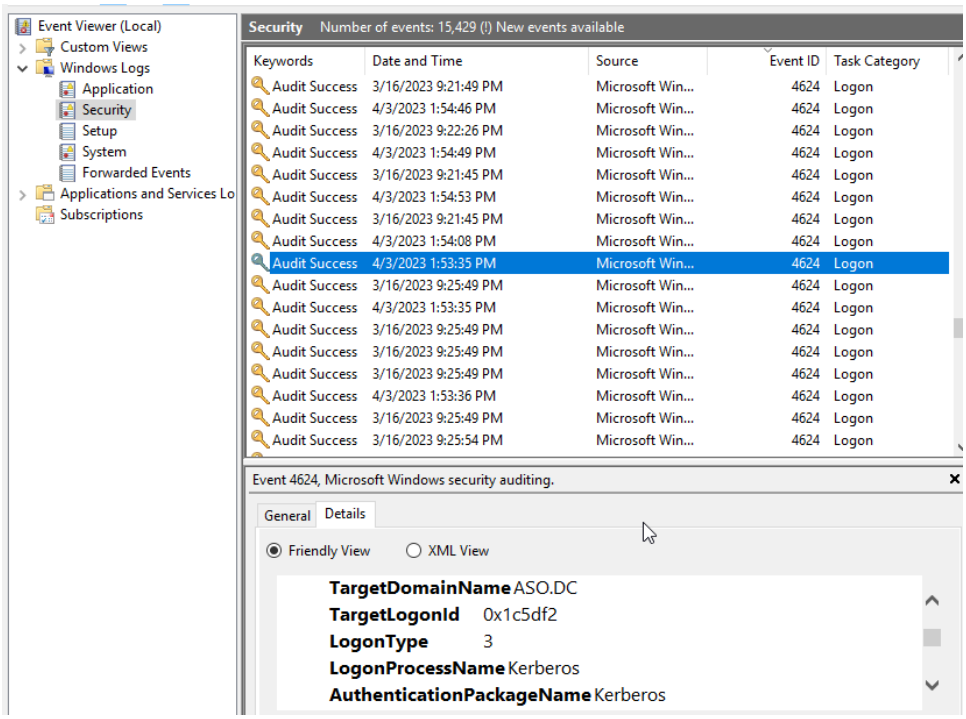
El objetivo del ataque **Golden Ticket** del es construir un TGT, para lo cual se necesita la clave del krbtgt. Por tanto, si se obtiene el hash NTLM de la cuenta krbtgt, es posible construir un TGT. Dicho TGT puede contar con la caducidad y permisos que se desee, consiguiendo incluso privilegios de administrador de dominio.

El ticket continuará siendo válido, aunque el usuario incluido cambie su contraseña. El TGT solo podrá ser invalidado si expira o cambia la contraseña de la cuenta krbtgt.

El **Silver Ticket** es similar, pero esta vez se construye un TGS y lo que se requiere es la clave del servicio al que se quiere acceder. Esta clave se deriva del hash NTLM de la cuenta propietaria del servicio. Esta técnica no funcionará si el servicio verifica el PAC, ya que al no conocer la clave de krbtgt, no es posible firmarlo correctamente.

Video sobre Golden Ticket: kerberos.mp4

AD emplea por defecto Kerberos como sistema de autenticación, podemos verlo si vamos al "Visor de Eventos" y buscamos por el código 4624.



Hablemos ahora entonces de LDAP.

El protocolo ligero de acceso a directorios (LDAP):

Es un protocolo para trabajar con varios servicios de directorio, al igual que AD. LDAP puede trabajar en conjunto con AD a pesar de ser dos cosas diferenciadas completamente.

LDAP es usado por aplicaciones que tengan este protocolo activado.

LDAP puede emplearse para que cuando un usuario intente acceder a un servicio, LDAP busque sus credenciales en la base de datos de AD y permitirle el acceso si está configurado de una de estas dos formas:

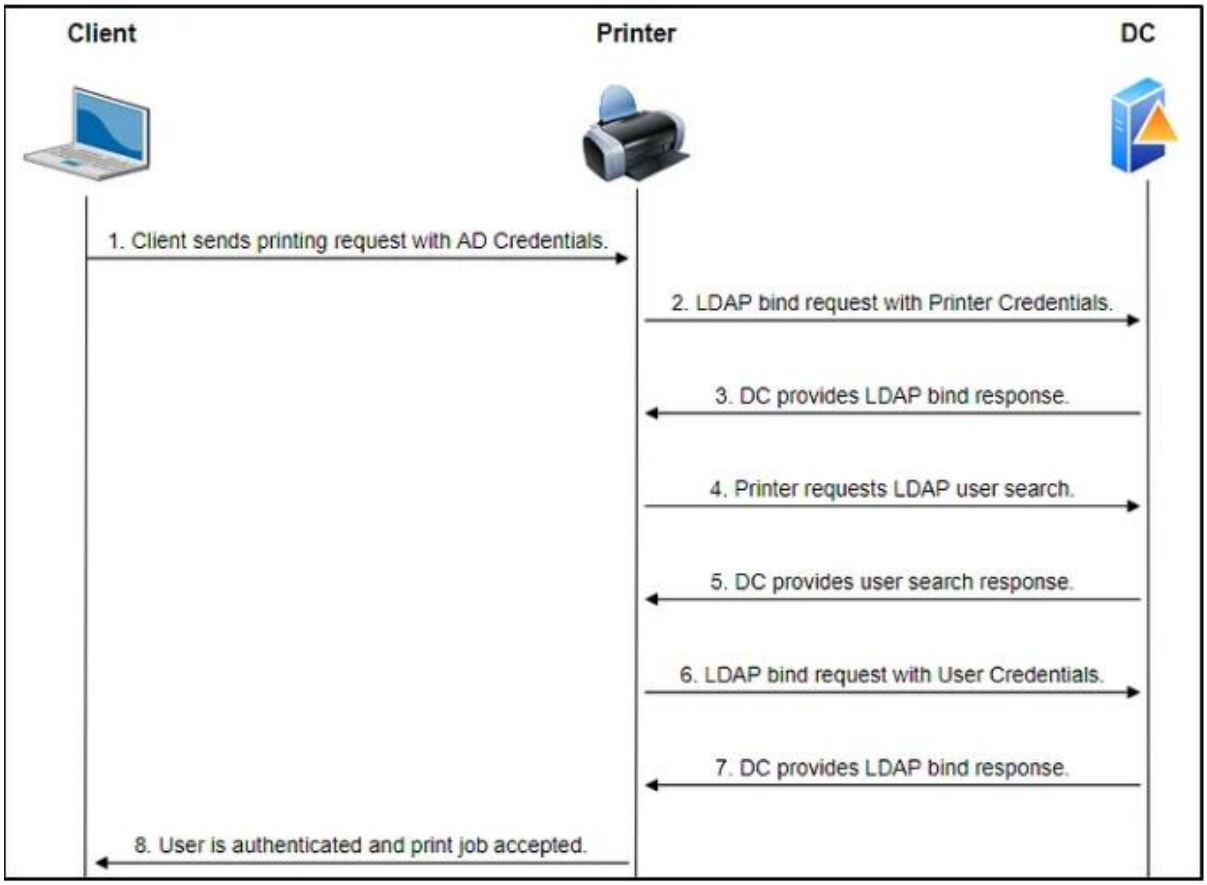
1. Autentificación simple: Las credenciales que es usuario uso para iniciar sesión son correctas y son las necesarias para acceder al servicio indicado además de que posee los privilegios necesarios.
2. Autentificación simple y capa de seguridad: emplea el método de autenticación simple pero además usa un servicio de autentificación secundario como lo puede ser Kerberos para añadir un nivel de seguridad mayor al proceso.

LDAP hace que la aplicación verifique directamente las credenciales del usuario. Las aplicaciones tienen un par de credenciales de AD que pueden usar para llamar a LDAP y que este verifique contra la base de datos de AD que el usuario es quien dice ser.

LDAP es un protocolo de autenticación y/o autorización de recursos IT, mientras que AD es más una base de datos.

Ejemplo: un usuario quiere imprimir usando una impresora con LDAP habilitado:

1. El cliente envía una petición usando sus credenciales de AD
2. La impresora usa sus propias credenciales para crear una petición LDAP, que se usa para autenticar usuarios
3. El DC indica si la autenticación de la impresora en el dominio fue correcta
4. La impresora manda otra petición LDAP para buscar un directorio LDAP, esto se hace para ver si se puede emplear LDAP para autenticación (funciona casi como un ping)
5. El DC indica si fue exitosa la petición
6. La impresora ahora manda la petición LDAP juntos con las credenciales de usuario para verificar si este usuario está autenticado en el dominio
7. El DC lanza su respuesta, indicando si está o no autenticado
8. La impresora dependiendo de la respuesta del DC indicará que el usuario está o no autenticado y si le permite o no ejecutar la acción de imprimir



BIBLIOGRAFÍA:

Active Directory

Active Directory, 5th Edition Designing, Deploying, and Running Active Directory (Brian Desmond, Joe Richards, Robbie Allen etc.). Pág 17-25

FSMO Roles

Ifadmins | conceptos ad fsmo roles

Replicación

Itadmins | conceptos ad replicacion

AD Schema:

ACTIVE DIRECTORY SCHEMA:

Active Directory, 5th Edition Designing, Deploying, and Running Active Directory (Brian Desmond, Joe Richards, Robbie Allen etc.) Pág 73

Theitbros.com | adsi-edit

ESTRUCTURA DEL ESQUEMA DE ACTIVE DIRECTORY:

Active Directory, 5th Edition Designing, Deploying, and Running Active Directory (Brian Desmond, Joe Richards, Robbie Allen etc.) Pág 74 y 75

Schema location

Windowstechno.com | what-is-active-directory-schema

X.500 OID NAMESPACE:

Active Directory, 5th Edition Designing, Deploying, and Running Active Directory (Brian Desmond, Joe Richards, Robbie Allen etc.) Pág 75-79

Impacto de un OID incorrecto

Social.technet.microsoft.com | Windows Server/custom oids in active directory

OIDs asignados

Techgenix.com | schemaobjectsidentifiersoids

What is an OID?

Alverstrand.no | objectid/index.htlm

CLASES DEL ESQUEMA:

Active Directory, 5th Edition Designing, Deploying, and Running Active Directory (Brian Desmond, Joe Richards, Robbie Allen etc.) Pág 95-104

Microsoft.learn | adschema/classes

Object class vs object category

Windows-Active_Directory | active-directory-object-classes-and-attributes

ATRIBUTOS DEL ESQUEMA:

Active Directory, 5th Edition Designing, Deploying, and Running Active Directory (Brian Desmond, Joe Richards, Robbie Allen etc.) Pág 79-81

Microsoft.learn | adschema/attributes

SINTAXIS DE UN ATRIBUTO:

Active Directory, 5th Edition Designing, Deploying, and Running Active Directory (Brian Desmond, Joe Richards, Robbie Allen etc.) Pág 82-83

LINKED ATTRIBUTES:

Active Directory, 5th Edition Designing, Deploying, and Running Active Directory (Brian Desmond, Joe Richards, Robbie Allen etc.) Pág 94

Neroblanco.co.uk | links-and-backlinks-in-active-directory-for-exchange

SYSTEMFLAGS:

Active Directory, 5th Edition Designing, Deploying, and Running Active Directory (Brian Desmond, Joe Richards, Robbie Allen etc.) Pág 84-86

SEARCHFLAGS:

Active Directory, 5th Edition Designing, Deploying, and Running Active Directory (Brian Desmond, Joe Richards, Robbie Allen etc.) Pág 86-93

RODC

Micrososft.learn | ad/rodc-and-active-directory-schema

PROPERTY SETS:

Active Directory, 5th Edition Designing, Deploying, and Running Active Directory (Brian Desmond, Joe Richards, Robbie Allen etc.) Pág 94

Itprotoday | compute-engines/using-ad-property-sets

MAPI ID:

Active Directory, 5th Edition Designing, Deploying, and Running Active Directory (Brian Desmond, Joe Richards, Robbie Allen etc.) Pág 95

Mail2.cni.org | MAPI.html

EXTENDER EL ESQUEMA:

Windowstechno.com | how-to-create-custom-attributes-in-active-directory

DNS

Windows-Active-Directory.com | dns-and-active-directory

Libro: "Active Directory, 5th Edition Designing, Deploying, and Running Active Directory (Brian Desmond, Joe Richards, Robbie Allen etc.)". Capítulo 6: "*Active Directory and DNS*".

Microsoft.learn.com | windows/server/networking/dns/dns-top

Zona DNS:
Cloudflare.com | learning/dns/glossary/dns-zone

Libro: "Active Directory, 5th Edition Designing, Deploying, and Running Active Directory (Brian Desmond, Joe Richards, Robbie Allen etc.)". Capítulo 6: "*Active Directory and DNS*"

Microsoft.com | /azure/dns/dns-zones-records#dns-zones

DHCP

Microsoft.learn.com | windows-server/troubleshoot/dynamic-host.configuration-protocol-basics

Microsoft.learn.com | windows-server/networking/technologies/dhcp/dhcp-top

Computernetworkingnotes.com | how-dhcp-works-explained-with-examples

IPAM:

Microsoft.learn.com | windows-server/networking/technologies/ipam/ipam-top

Redes permitidas en Windows Server

Microsoft.learn.com | windows-server/networking/windows-server-supported-networking-scenarios

Políticas de grupo:

Libro: "Active Directory, 5th Edition Designing, Deploying, and Running Active Directory (Brian Desmond, Joe Richards, Robbie Allen etc.)". Capítulo 8: "*Group Policy Primer*".

Windows-Active-Directory.com | create-enable-disable-gpo

Windows-Active-Directory.com | active-directory-policies

Windows-Active-Directory.com | security-filtering-and-wmi-filtering

Microsoft.learn.com | active-drectory-domain-services/manage-group-policiy

ACLS:

HackTricks | ACLs – DACLs/SACLs/ACEs - HackTricks

Mirosoft.learn.com | Access control list –Win32 apps

NTLM

Microsoft.learn.com | Microsoft-ntlm –Win32 apps

Windows-Active-Drirectory.com | ntlm-and-kerneros-authentication-protocols

Infosecwriteups | ntlm-authentication-in-active-directory

Kerberos

Tarlogic | como-funciona-kerberos

Microsoft.learn | microsoft-kerberos

LDAP

Infosecwriteups | ldap-in-active-directory

Okta.com | What is LDAP and How does it works

Connect2id.com | ldapauth/auth-explained

Jumpcloud.com | what-is-ldap-authentication

Wikipedia.org | Protocolo-ligero-de-acceso-a-directorios