FACULTEIT INGENIEURSWETENSCHAPPEN
EN ARCHITECTUUR

Besturingssystemen III: Examen

Bert De Saffel

Master in de Industriële Wetenschappen: Informatica Academiejaar 2018–2019

Gecompileerd op 22 januari 2019

UNIVERSITEIT
GENT

Inhoudsopgave

1 Modelvragen theorie: reeks A **2**

1.1 Structuur van Active Directory gegevens 2

1.2 attributeSchema objecten (§2.2.4 en §2.2.5) 4

1.3 classSchema objecten (§2.2.4 en §2.2.6) 6

1.4 Active Directory domeinstructuren (§2.4.4, laatste paragraaf §2.4.5 en §2.4.6) 8

1.5 Active Directory server rollen (§2.4.7, §2.3 en fractie §2.4.2) 9

2 Modelvragen theorie: reeks B **11**

2.1 Active Directory functionele niveaus (§2.4.3) 11

2.2 Active Directory replicatie (§2.5) 13

2.3 Gedeelde mappen en NTFS 15

2.4 Machtigingen op bestandstoegang (§3.3) 18

2.5 Gebruikersgroepen (§4.2.2 en §4.2.3) 20

Hoofdstuk 1

Modelvragen theorie: reeks A

Het examen wordt volledig schriftelijk beantwoord. Indien de student dit wenst, wordt het antwoord onmiddellijk na indienen geëvalueerd, en eventueel gevolgd door enkele vragen ter verduidelijking of aanvulling.

1.1 Structuur van Active Directory gegevens

1. Bespreek de *diverse namen* die alle Active Directory objecten *identificeren*. §2.2.1

   - **Relative distinguished name (RDN).** Dit is een identificatie van een object **binnen** een **containerobject**. Een RDN moet dus **niet uniek** zijn op Active Directory niveau. Dit wordt opgeslagen in het attribuut **cn**.
   - **Distinguished name.** Deze **unieke** identificatie wordt opgebouwd uit de RDN van het object zelf en van alle RDNs waarvan het object hiërarchisch deel uitmaakt. Dit wordt opgeslagen in het attribuut **distinguishedName**
   - **Canonieke naam.** Dit heeft dezelfde functie als een distinguished name, maar heeft een eenvoudigere representatie. Dit wordt opgeslagen in het attribuut **canonicalName**
   - **GUID.** Elk object heeft een **unieke** GUID. Dit is een 128-bit getal dat niet kan gewijzigd worden. Een GUID wordt aangemaakt bij de **creatie** van een object en kan dan **niet meer aangepast** worden. Dit wordt opgeslagen in het attribuut **objectGUID**.

2. Wat zijn *SPN objecten*? Bespreek de *aanvullende naamgeving* voor deze objecten. (§2.2.2)

   Security Principal Objects (SPN) zijn objecten die **security IDs (SIDs)** bevatten. Dergelijke objecten worden gebruikt voor het verlenen van toegang tot domeinbronnen en zijn daarom van toepassing op gebruikersaccounts, computeraccounts, groepen en domeinen. Een SID is net zoals een GUID uniek binnen een forest voor elk nieuw aangemaakt object. Bij het verplaatsen en hernoemen binnen hetzelfde domein blijven de GUID en SID behouden van een SPN object. Indien het object naar een andere domein verplaatst wordt, zal enkel GUID hetzelfde blijven. Hierbij wordt het **sIDHistory** attribuut aangevuld met het vorige SID. De aanvullende naamgeving voor deze objecten zijn:

   - Een **gebruikersaccount** heeft nood aan een extra identificatieattribuut om inlogprocedures door een gebruiker te vereenvoudigen. Een **User Principal Name (UPN)** is een vereenvoudigde waarde (loginnaam) dat uitsluitend gebruikt wordt voor aanmelding. Verder moet een UPN uniek zijn binnen het hele forest en heeft standaard de volgende vorm:

     `RDN@UPNsuffix`

Hierbij is RDN de RDN van de gebruiker en UPNsuffix één van de volgende alternatieven:

- De DNS domeinnaam waarin het gebruikersaccount zich bevindt.
- De DNS domeinnaam van het root domein.
- Een willekeurige maar een op voorhand gedefinieerde naam.

Er bestaat ook een alternatieve vorm, om compatibiliteit met vroegere NT versies te behouden. Men kan de NetBIOS naam van het domein en de SAM accountnaam van de gebruiker, aan elkaar gekoppeld door het \-teken te gebruiken.

- Een **computeraccount** bevat drie extra attributen:
  - **SAM accountnaam.** Dit is gelijkaardig aan het UPN attribuut, maar dient voor compatibiliteit met oudere windows systemen (pre 2000). Standaard bestaat deze naam uit de eerste 15 bytes van de RDN, gevolgd door een $ teken. Deze naam kan op elk moment gewijzigd worden. Deze waarde wordt opgeslagen in het attribuut **samAccountName**.
  - **DNS hostnaam.** Deze waarde bevat standaard de eerste 15 bytes van de RDN en de suffix voor primaire DNS.
  - **Service Principal Name.** Dit attribuut wordt gebruikt tijdens de wederzijdse verificatie van client software en de server die een bepaalde service aanbiedt. Het attribuut wordt bepaald door het multi-valued `servicePrincipalName` kenmerk.

3. Enkele veel gebruikte klassen (hiermee worden *attributeschema* en *classschema* objecten niet bedoeld) vertonen nog meer identificerende attributen voor hun instanties. Bespreek deze klassen en attributen.

4. In welke *partities* is de Active Directory informatie verdeeld? Geef de betekenis van elke partitie, hun onderlinge relatie (zowel fysiek als met betrekking tot hun naamgeving), en de replicatiekarakteristieken ervan. (laatste helft §2.2.3)

   - **Domeingegevens.** Deze partitie bevat informatie over alle objecten in het domein (servers, bestanden, printers, accounts, ...). Objecten die aangemaakt of gewijzigd zijn, worden steeds opgeslagen in de domeingegevens. In Active Directory kunnen er meerdere domeinen bestaan in een forest. Deze domeinen vormen een boomstructuur met als wortel de domeingegevens van het **root** domein.

     Domeingegevens van een bepaald domein worden gerepliceerd tussen alle domeincontrollers van enkel dat domein.

   - **Configuratiegegevens.** De configuratiegegevens beschrijven de fysieke topologie van de directory. Het bevat ondermeer een lijst van alle domeinstructuren, de locaties van de domeincontrollers en de global catalog controllers, de sites en de replicatietopologie. Er is slechts één configuratiepartitie per forest en geldt voor alle domeincontrollers in dit forest.

     De configuratiegegevens worden gerepliceerd naar alle domeincontrollers in het forest.

   - **Het schema.** Deze partitie is de formele definitie van alle objecten en kenmerkgegevens die kunnen opgeslagen worden in de directory. Er is slechts één schemapartitie per forest.

     Het schema wordt gerepliceerd naar alle domeincontrollers in het forest.

   - **Applicatiepartitie.** De applicatiepartitie laat toe om een deel van de Active Directory gegevens gescheiden onder te brengen. Indien de DNS gegevens van de domeinen en van het forest geïntegreerd worden in Active Directory, dan worden er respectievelijk `DomainDNSZones` en `ForestDNSZones` applicatiepartities aangemaakt.

     De replicatiestrategie van een applicatiepartie is de enige die dynamisch kan gewijzigd worden, aangezien het ook de enige partitie is dat dynamisch kan aangemaakt worden. De specifieke domeincontrollers instellen die repliceren kan in het `msDS-NC-Replica-Locations` kenmerk van het overeenkomstige crossref object in de configuratiegegevens.

1.2 attributeSchema objecten (§2.2.4 en §2.2.5)

1. Bespreek het *doel* en de *werking* van attributeSchema objecten. Hoe kunnen deze objecten het best *geraadpleegd* en *gewijzigd* worden?

   - Het attributeSchema bevat alle kenmerken die in het schema voorkomen. Een kenmerk is zelf een object. Zo een kenmerk wordt éénmaal gedefinieerd en kan meerdere malen gebruikt worden bij verschillende klassen, wat voor consistentie zorgt. Een goed voorbeeld hiervan is het description kenmerk. Om op een eenvoudige manier objecten te raadplegen of wijzigen, kan men met het schema snap-in mechanisme werken. **Active Directory Schema** is zo een snap-in, dat eenvoudig in twee vensters zowel de kenmerken als de klassen weergeeft. Dubbelklikken op een item geeft de meest relevante eigenschappen van het object, en de mogelijkheid om deze in te stellen. Er kan ook gebruikt gemaakt worden van de opdracht **dsquery ***. Deze opdracht kan om het even welke kenmerken tonen van één enkel object, of van alle objecten in een container, zowel niet-recursief als recursief.

2. Bespreek de *diverse naamgevingen*, specifiek voor attributeSchema objecten.

   - **cn**: De RDN van het attributeSchema object in de Schema container.
   - **schemaIDGUID**: Dit kan automatisch gegenereerd worden bij de creatie van een nieuw kenmerk. Een attribuut krijgt dan wel een verschillende GUID in verschillende forests. Manueel een GUID instellen kan ook, met bv de **guidgen** of **uuidgen** opdrachten.
   - **lDAPDisplayName**: De naam die gebruikt wordt voor LDAP. Deze naam is belangrijk voor programmatische toegang.
   - **attributeID**: De interne representatie van een object. Deze identifiers worden verleend door specialie autoriteiten, en zijn gegarandeerd uniek in alle netwerken over de hele wereld. Een Object Identifier bestaat uit een decimale reeks met punten, waarbij de toekenning op een hiërarchische manier gebeurd. Een Object Identifier kan aangevraagd worden bij een regionale ISO vertegenwoordiger. Indien dit niet gewenst is, kan er ook een Object Identifier gegenereerd worden in een Microsoft subtak, met behulp van de opdracht **oidgen**.

3. Bespreek de belangrijkste *kenmerken* van attributeSchema objecten, en op welke waarden die ingesteld kunnen worden.

   - **attributeSyntax en oMSyntax**: De syntax bepaalt het data type zoals: Object ID, Boolean, Integer, DirectoryString zijn enkele van de 26 mogelijkheden. Slechts 18 van deze 26 worden momenteel gebruikt in Active Directory. Het is **onmogelijk** om een nieuwe syntax te definieëren. Het object ID van een syntax wordt geïdentificeerd in de vorm van 2.5.5.x . Sommige Object Identifiers zijn blijkbaar niet te onderscheiden, waarop beroep moet gedaan worden op een bijkomende Integer waarde: mOSyntax.
   - **rangeLower en rangeUpper**: Bepalen de lengte- of bereikbeperkingen van kenmerken.
   - **isSingleValued**: Geeft aan of een attribuut meerdere waarden kan bevatten (een lijst).
   - **searchFlags**: Dit veld bevat binaire informatie, waarbij elke afzonderlijke bit kan ingesteld worden. Veronderstel de vorm $b_{10}b_9b_8b_7b_6b_5b_4b_3b_2b_1$, dan betekent elke bit het volgende:
     - b_1: Deze wordt meestal gezet, zodat eenvoudige indexering van de waarde van het kenmerk geactiveerd wordt, ongeacht van waar het object zich in Active Directory bevindt.
     - b_2: Indien deze gezet wordt, wordt de waarde van het kenmerk gecombineerd met de identificatie van de container waarin het object zich bevindt. Dit heet een containerized index, en zijn in staat om snel objecten op te sporen in een specifieke container.

- b_3: Dit laat Ambiguous Name Resolution toe. Bij opzoekingen van kenmerken waaraan een bepaalde waarde voldaan moet worden, kan i.p.v.

  $$(|\ (kenmerk1 = waarde)(kenmerk2 = waarde)(kenmerk3 = waarde)...)$$

  het eenvoudigere

  $$(anr = waarde)$$

  gebruikt worden.
- b_5: Deze bit heeft niets met indexering te maken, maar geeft aan of de waarde van een attribuut behouden blijft bij het maken van een kopie van dit object.
- b_6: Deze bit instellen versnelt opzoekingen waarin kenmerken met wildcards vermeld worden. Deze tuple indexen worden best zeldzaam gebruikt, aangezien ze veel resources in beslag nemen.
- b_8: (staat niets over in de cursus)
- b_{10}: (staat niets over in de cursus)

- **systemFlags**: Dit heeft dezelfde vorm als searchFlags, een binair formaat $b_{28}...b_5b_4b_3b_2b_1$ waarvan de bits het volgende betekenen:
  - b_1: Deze bit geeft aan of dat het kenmerk gerepliceerd mag worden naar andere domeincontrollers of niet. Attributen die vaak wijzigen zoals lastLogOn en lastLogOff worden niet gerepliceerd.
  - b_3: Dit geeft aan of een attribuut geconstrueerd is of niet. Een dergerlijk attribuut wordt niet opgeslagen in Active Directory, maar wordt telkens opnieuw berekend op basis van andere kenmerken.
  - b_5: Geeft aan of dat het attribuut een systeemobject is.
  - b_{28}:Geeft aan dat de naam van het object al dan niet gewijzigd kan worden.
- **isMemberOfPartialAttributeSet**: Bepaalt of een attribuut in de global catalog wordt opgenomen of niet.
- **linkID**: Een getal dat aangeeft dat het attribuut een gelinkt attribuut is. Een even getal duidt op een forward-link en het sequentieel daaropvolgend oneven getal duidt op de corresponderende back-link. Dergerlijke linkIDs laten toe om n-tot-n en n-tot-1 relaties te definiëren zoals de member en memberof, alsook de managedBy en managedObjects attributen.

4. Welke andere types objecten bevat het *Active Directory schema*, en wat is hun bedoeling? (o.a. §2.2.7)

   - **subSchema:** Er is slechts één object die behoort tot de klasse subSchema, Aggregate. Dit object bevat een alternatieve, compacte representatie van het gehele schema en wordt het abstracte schema genoemd. De bedoeling is om vereenvoudigde schema gegevens ter beschikking te stellen aan LDAP clients, zonder dat die zich om veel implementatie details hoeven te bekommeren. Voor elk classSchema en attribuutSchema object biedt het abstracte schema slechts enkele kenmerken aan.
   - **classSchema:** Voor elke klasse is er een classSchema object waarmee de klasse kan ingesteld worden. De kenmerken van classSchema objecten definiëren de klasse, en bevatten twee soorten regels: met structuurregels worden mogelijke hiërarchische relaties tussen klassen of objecten gedefinieerd, terwijl met inhoudsregels kenmerken definiëren die beschikbaar zijn voor een exemplaar van die klasse.

5. Via welke attributen kun je de *klasse* van een willekeurig Active Directory object achterhalen? Hoe moet je op zoek gaan naar alle objecten van een bepaalde klasse? Illustreer aan de hand van relevante voorbeelden. (laatste paragraaf §2.2.6)

- **objectClass:** Het attribuut objectClass is multi-valued en niet geïndexeerd en bevat niet enkel de klasse zelf, maar ook alle hiërarchische superklassen. Voor printQueue, user en computer is objectClass ingevuld met respectievelijk {printQueue, connectionPoint, leaf, top}, {user, organizationalPerson, person, top} en {computer, user, organizationalPerson, person, top}.
- **objectCategory:** Het kenmerk objectCategory is single-valued en wordt geïndexeerd. Het bevat echter niet noodzakelijk de klasse van het object, maar bevat de meest typische vertegenwoordiger voor die klasse. Voor printQueue, user en computer is objectCategory respectievelijk printQueue, person en computer.

Het opzoeken van alle objecten kan zowel via *objectClass* als *objectCategory*. Het is soms niet evident wanneer wat moet gebruikt worden. Voor het opzoeken van printers is de selecte van objecten waarvoor de *objectCategory* ingesteld is op *printQueue*, duidelijk de beste keuze, aangezien dit de opzoeking toelaat om op indexering een beroep te doen. Om gebruikers op te zoeken zou de objecten waarvan *objectCategory* gelijkgesteld is aan *person* kunnen gefiltered worden. Dit heeft echter als nadeel dat ook objecten met klasse *contact* teruggegeven worden. Er kan dan besloten worden om niet *objectCategory* te gebruiken, maar objecten te selecteren waarvan *user* tot de *objectClass* behoort. Dit heeft weeral het vervelende effect dat objecten met klasse *computer* ook geselecteerd worden. De juiste objecten kunnen gezocht worden door de combinatie waarvoor zowel de *objectCategory* ingesteld is op *person*, als *user* tot de *objectClass* behoort.

1.3 classSchema objecten (§2.2.4 en §2.2.6)

1. Bespreek het *doel* en de *werking* van classSchema objecten.

   - Voor elke klasse is er een classSchema object waarmee de klasse kan ingesteld worden. De kenmerken van classSchema objecten definiëren de klasse, en bevatten twee soorten regels: met structuurregels worden mogelijke hiërarchische relaties tussen klassen of objecten gedefinieerd, terwijl met inhoudsregels kenmerken definiëren die beschikbaar zijn voor een exemplaar van die klasse.

2. Hoe benadert Active Directory het mechanisme van *overerving*?

   - Een klasse die een andere klasse (de superklasse) overeft, neemt de kenmerken van deze superklasse over, inclusief de structuurregels en de inhoudsregels. Deze overerving werkt recursief: de subklasse erft alle gegevens van opeenvolgende superklassen. Een klasse kan echter maar van één superklasse overerven, en eventueel van speciaal hiervoor bestemde hulpklassen, die zelf geen instanties kunnen bevatten. Vanaf Windows Server 2003 is het mogelijk om dynamische objecten te definiëren. Hierdoor is het mogelijk om hulpklassen dynamisch te gebruiken: tijdens de create van een object kunnen extra dynamische hulpklassen gegenereerd worden, die enkel voor die instantie geldig zijn, door aanvullingen van het objectClass attribuut. Na de creatie kan dit attribuut niet meer gewijzigd worden.

     Het kenmerk `objectClassCategory` geeft de categorie van de klasse aan: structurele klasse, abstracte klasse en hulpklasse. Er kunnen geen instanties aangemaakt worden van een abstracte klasse, maar ze bevatten wel een verzameling kenmerken die door subklassen die van de klasse afgeleid zijn, overgeërfd worden.

     De hiërarchische relaties tussen objecten van klassen wordt gedefinieerd in de kenmerken `possSuperiors` en `systemPossSuperiors`.

3. Bespreek de diverse *naamgevingen*, specifiek voor classSchema objecten.

   komt overeen met vraag 1.2.2, maar enkel de naamgeving is anders

- **cn**: De RDN van het classSchema object in de Schema container.
- **schemaIDGUID**: Dit kan automatisch gegenereerd worden bij de creatie van een nieuw kenmerk. Een klasse krijgt dan wel een verschillende GUID in verschillende forests. Manueel een GUID instellen kan ook, met bv de **guidgen** of **uuidgen** opdrachten.
- **lDAPDisplayName**: De naam die gebruikt wordt voor LDAP. Deze naam is belangrijk voor programmatische toegang.
- **governsID**: De interne representatie van een object. Deze identifiers worden verleend door specialie autoriteiten, en zijn gegarandeerd uniek in alle netwerken over de hele wereld. Een Object Identifier bestaat uit een decimale reeks met punten, waarbij de toekenning op een hiërarchische manier gebeurd. Een Object Identifier kan aangevraagd worden bij een regionale ISO vertegenwoordiger. Indien dit niet gewenst is, kan er ook een Object Identifier gegenereerd worden in een Microsoft subtak, met behulp van de opdracht **oidgen**.

4. Bespreek de belangrijkste *kenmerken* van classSchema objecten, en op welke waarden die ingesteld kunnen worden.

   - **rDNAttID:** bepaalt welk kenmerk van een klasse gebruikt wordt om de RDN van objecten te bepalen. Vaak gebruikte waarden zijn: cn (Common-Name), ou (Organizational-Unit-Name), dc (Domain-Component), o (Organization-Name), c (Country-Name) of l (Locality-Name).
   - **(system)mustContain, (system)mayContain:** Deze vier attributen specificeren welke kenmerken optioneel zijn en welke verplicht ingevuld moeten worden. Voor een kenmerk dat overgeërfd is, en verplicht is, kan het kenmerk niet optioneel gemaakt worden, ook al staat in de subklasse dit kenmerk bij `(system)mayContain`. De prefix `system` duidt op de lijst van attributen die enkel door Active Directory zelf kunnen gewijzigd kunnen worden.
   - **defaultSecurityDescriptor:** bepaalt de explicitiete machtigingen die gelden voor objecten van deze klassen.
   - **systemOnly:** Indien deze bit aanstaat, kunnen de structuurregels en inhoudsregels van de klasse niet gewijzigd worden.
   - **systemFlags:** Kan twee belangrijke bits bevatten:
     - bit 5: geeft aan of dat de klasse een systeemklasse is of niet.
     - bit 28: geeft aan of dat de naam van de klasse mag gewijzigd worden.
   - **objectClass:** een multi-valued en niet geïndexeerd kenmerk. Het bevat niet alleen de klasse van het object zelf, maar ook alle hiërarchische superklassen.
   - **objectCategory:** is single-valued en wordt geïndexeerd. De waarde van dit kenmerk bevat de meest typische vertegenwoordiger uit de verzameling van het objectClass kenmerk.
   - **isDefunct:** duidt aan of al dan niet instanties van de klasse aangemaakt mogen worden.

5. Welke andere types objecten bevat het *Active Directory schema*, en wat is hun bedoeling? (o.a. §2.2.7)

   - **subSchema:** er is slechts één object die behoort tot de klasse subSchema, Aggregate. Dit object bevat een alternatieve, compactie representatie van het gehele schema en wordt het abstracte schema genoemd. De bedoeling is om vereenvoudigde schema gegevens ter beschikking te stellen aan LDAP cliënts, zonder dat die zich om veel implementatie details hoeven te bekommeren. Voor elk classSchema en attribuutSchema object biedt het abstracte schema slechts enkele kenmerken aan.

- **attributeSchema:** Het attributeSchema bevat alle kenmerken die in het schema voorkomen. Een kenmerk is zelf een object. Zo een kenmerk wordt éénmaal gedefinieerd en kan meerdere malen gebruikt worden bij verschillende klassen, wat voor consistentie zorgt.

6. Hoe en met welke middelen kan het Active Directory schema uitgebreid worden? Waarom moet je en hoe kan je hierbij *voorzichtig* te werk gaan? (o.a. §2.2.8, ldifde fractie §2.2.3)

   - Wanneer onvoorzichtig uitbreidingen aan het schema worden toegevoegd, kan dit nefaste gevolgen hebben op domein en forestniveau. Een domein kan beschadigd geraken of zelfs uitschakelen. Bovendien gelden de wijzigingen voor het gehele forest.

     De veiligste manier om uitbreidingen op het schema te ontwikkelen en uit te testen is een geïsoleerd netwerk van een testomgeving. Bovendien worden schemaobjecten beveiligd door Access Control Lists, zodat enkel gemachtigde gebruikers aanpassingen kunnen doen. Verder kunnen ook nog volgende richtlijnen gehanteerd worden:

     - Vermijdt het wijzigen van de attributen van een bestaande klasse.
     - Maak enkel een nieuwe structurele klasse (een subklasse van de superklasse Top) aan als er geen enkel ander object enigszins aan de behoeften voldoet.

     Een schema uitbreiden heeft echter veel potentieel. De aanbevolen manier om het Active Directory op grote schaal uit te breiden is via de **ldifde** opdracht, aangezien één ldifde inputbestand meerdere aanpassingen ineens kan bevatten. Zo een inputbestand heeft als formaat het LDAP Data Interchange Format. Aanpassingen op kleine schaal kunnen eerder gebeuren met de Active Directory Schema snap-in. Om op een veilige manier kenmerken toe te voegen aan objecten, maakt men eerst de kenmerken aan in het attributeSchema, vervolgt door het aanmaken van een hulpklasse die deze kenmerken bevat. De nieuwe hulpklasse kan dan geassocieerd worden met de klasse waaraan de kenmerken moeten toegevoegd worden.

1.4 Active Directory domeinstructuren (§2.4.4, laatste paragraaf §2.4.5 en §2.4.6)

1. Wat is de bedoeling van *vertrouwensrelaties*?

   - Tussen twee domeinen kan er een vertrouwensrelatie tot stand gebracht worden, zodat gebruikers in het vertrouwd domein kunnen geverifieerd worden door de domeincontroller in het vertrouwend domein. Een gebruiker kan slechts domeinbronnen van een ander domein raadplegen indien er een vertrouwenspad bestaat tussen de twee domeinen. Dit wil niet zeggen dat de gebruiker automatisch toegang heeft tot de bronnen in dat domein.

2. Bespreek de verschillende *soorten* vertrouwensrelaties.

   - **Automatische vertrouwensrelaties.**
     - Windows Server maakt automatisch vertrouwensrelaties aan tussen domeinen en hun kinddomeinen bij creatie. Deze automatische vertrouwensrelaties kunnen niet verbroken worden en zijn bovendien bi-directioneel en transitief. Windows Server maakt ook automatisch vertrouwensrelaties aan tussen de trees van eenzelfde forest: de root domeinen van alle trees in het forest vormen transitieve vertrouwensrelaties met het forest root domein van het forest. Een domein dat nieuw aangemaakt wordt in een tree of een forest heeft, door de transitieve eigenschap, automatisch vertrouwensrelaties met alle andere Windows Server domeinen in de tree of het forest.
   - **Expliciete vertrouwensrelaties.**

- **Forest vertrouwensrelatie.** Indien de diverse forests minimaal Windows Server 2003 functioneel niveau hebben, dan kunnen er een bi-directionele en transitieve vertrouwensrelaties tussen de root-domeinen van deze forests gelegd worden.
- **Realm vertrouwensrelatie.** Dit is een veralgemening van een forest vertrouwensrelatie. Deze relatie kan gelegd worden tussen een Windows Server 2008 domein en een willekeurig kerberos v5 realm, onafhankelijk van het besturingssysteem waarop die geïmplementeerd zijn.
- **Verkorte vertrouwensrelatie.** Deze enkelvoudige of bi-directionele relatie kan gelegd worden tussen Windows Server domeinen binnen hetzelfde forest. Deze verkorte vertrouwensrelaties kunnen gebruikt worden om het vertrouwenspad in grote en complexe trees korter te maken. Praktisch is dit enkel nuttig indien het vertrouwenspad minstens een vijftal domeinen overspant, en dan uiteraard nog indien er frequent gebruik van gemaakt wordt.
- **Externe vertrouwensrelatie.** Dit is een enkelvoudige relatie waarbij één domein een ander vertrouwt. Verificatieaanvragen kunnen enkel van het trusting domein naar het trusted domein doorgegeven worden. Om een externe vertrouwensrelatie in twee richtingen te leggen, moeten er twee enkelvoudige relaties gelegd worden. Deze relatie is ook niet transitief.

3. Op welke diverse manieren kunnen vertrouwensrelaties *gecreëerd* en *gecontroleerd* worden? Bespreek ook de *optionele configuratiemogelijkheden*.

   - Om een vertrouwensrelatie aan te maken, moet de domeinnamen en gebruikersaccount met machtigingen om vertrouwensrelaties in beide domeinen te maken, beschikbaar zijn. Een vertrouwensrelatie krijgt een wachtwoord toegewezen, dat bekend moet zijn bij de beheerders van beide domeinen van de vertrouwensrelatie. Dit wachtwoord wordt na het opzetten van de vertrouwensrelatie nooit meer gebruikt. Volgende optionele configuratiemogelijkheden bestaan ook:
     - **Selective Authentication.** Standard worden alle gebruikers van het trusted domein opgenomen in de Authenticated Users groep van het trusting domein. Via selective Authentication moet dit per individuele gebruiker of gebruikersgroep ingesteld worden.
     - **SID Filtering.** Indien SID Filtering aanstaat, wordt enkel rekening gehouden met de SID opgeslagen in het *objectSid* attribuut van de objecten in het trusted domein. Staat dit niet aan, dan verwerkt het trusting domein ook de SIDs opgeslagen in het sIDHistory attribuut.
   - Vertrouwensrelaties kunnen ook aangemaat worden in een *Command Prompt*, met behulp van de **netdom trust** opdracht. Om een overzicht te krijgen van alle vertrouwensrelatie en hun toestand, kan gebruik gemaakt worden van de **netdom query trust** opdracht.

4. Welke verschillen zijn er in praktijk tussen *NT 4.0* en *Windows Server* domeinstructuren? Bespreek onder andere telkens de noodzaak om meerdere domeinen in te voeren. Bespreek de alternatieve mogelijkheden bij de *conversie van een NT 4.0 domeinstructuur* naar een Windows Server omgeving.

   ToDo: Oplossen

1.5 Active Directory server rollen (§2.4.7, §2.3 en fractie §2.4.2)

Welke vragen moet men zich stellen na de initiële installatie van een Windows Server toestel, in verband met *bijzondere functies* die de server kan vervullen met betrekking tot Active Directory? Formuleer bij het beantwoorden van deze vragen telkens (voor zover relevant):

- Hoe bepaald wordt *welke servers* een dergelijke specifieke functie vervullen? *Hoeveel* zijn er nodig (in termen van: *minimaal/exact/maximaal #*, *in functie van* ...), en waarom?
- *Eigenschappen* zoals bedoeling, noodzaak, kriticiteit, inhoud, synchronisatie, voor welke Windows versie(s) van toepassing, ...?
- De *eventuele relatie* tussen de diverse functies. Vermeld bijvoorbeeld welke functies al dan niet door dezelfde server *kunnen* vervuld worden, of misschien wel juist wel door dezelfde server *moeten* vervuld worden.
- Hoe kan achterhaald worden welk(e) toestel(len) de bijzondere functie vervult, en op welke diverse manieren men de *toewijzing* ervan kan instellen, wijzigen en/of ongedaan maken?

  ToDo: oplossen

Hoofdstuk 2

Modelvragen theorie: reeks B

2.1 Active Directory functionele niveaus (§2.4.3)

1. Geef de diverse *functionele niveaus* waarop Active Directory kan ingesteld worden, en welke beperkingen er het gevolg van zijn.

   - **Windows 2000 mixed.** Een forest met dit functioneel niveau stelt geen enkele eis aan het functioneel niveau van de liddomeinen. Een domein met dit functioneel niveau biedt echter de laagste functionaliteit.
   - **Windows 2000 native.** Dit heeft geen impact op een forest. Op domeinniveau legt dit echter de beperking op dat een domeincontroller NT 5+ draait. Lidservers en werkposten hebben deze restrictie niet.
   - **Windows Server 2003.** Een forest met dit functioneel niveau kan enkel domeinen bevatten waarbij hun domein functioneel niveau minstens Windows Server 2003 is. Op domein functioneel niveau laat dit enkel nog Windows Server 2003+ domeincontrollers toe, zonder deze restrictie ook op te leggen aan lidservers en werkposten.
   - **Windows Server 2008.** Een forest met dit functioneel niveau kan enkel domeinen bevatten waarbij hun domein functioneel niveau minstens Windows Server 2008 is. Het biedt echter geen aanvullende functionaliteit. Analoog met Windows Server 2003, zal een domein met functioneel niveau Windows Server 2008 enkel Windows Server 2008+ domeincontrollers toelaten, zonder deze restrictie ook op te leggen aan lidservers en werkposten.

2. Bespreek van elk niveau alle eraan gekoppelde voordelen. Geef hierbij telkens een korte bespreking (verspreid over de cursus !) van ingevoerde begrippen.

   Elk niveau kan ondergebracht worden in het domein functioneel niveau en forest functioneel niveau.

   - **Domein functioneel niveau.**
     - **Windows 2000 mixed.** Geen voordelen, standaardfunctionaliteit.
     - **Windows 2000 native.**
       - Er is de keuze om slechts één global catalog te hebben voor het hele forest, wat voor minder replicatie zorgt. De global catalog bevat een read-only en verkorte inhoudsopgave van elk domein in een groep domeinen. Hierdoor kunnen objecten opgezocht worden, zonder te weten in welk specifiek domein van de directory deze gegevens feitelijk zijn opgeslagen.

        - Transitieve vertrouwensrelaties tussen verschillende domeinen van eenzelfde forest zijn mogelijk. Twee domeinen kunnen een vertrouwensrelatie opstellen, zodat gebruikers in het vertrouwd domein kunnen geverifieerd worden door de domeincontroller in het vertrouwend domein.
        - Domeincontrollers zijn zelf in staat om SPN objecten aan te maken, hiervoor gedelegeerd door de RID master. De RID master is een serverrol dat slechts door één domeincontroller kan vervuld worden. Deze RID master geeft reeksen relatieve SIDs wanneer een domeincontroller zijn RID pool voor 80% heeft opgebruikt.
        - Gebruikers en/of computers kunnen verzameld worden in groepen. Een groep wordt gebruikt om een verzameling van gebruikersobjecten die dezelfde toegangsmachtigingen hebben te groeperen, zodat restricties enkel op deze groep moeten gedefinieerd worden en niet op de individuele gebruikersobjecten.
        - Alle SIDs die in een SPN object in het verleden gehad heeft, worden bijgehouden in het sIDHistory kenmerk.
    - **Windows Server 2003.**
        - Gebruik van aanvullende schema klassen en attributen.
        - Het veranderen van de naam van een domeincontroller, zonder degradatie en promotie.
        - Gebruik van aanvullende opdrachten zoals *redirusr* en *redircmp* om de default Active Directory containers te wijzigen waarin respectievelijk nieuwe gebruikers en nieuwe computers terechtkomen.
        - Caching op domeincontroller niveau van UPN suffixen en het lidmaatschap van universele groepen, zodat het niet meer strikt noodzakelijk is dat tijdens het inlogproces een global catalog bereikbaar is. Een universele groep is een groep die leden kan bevatten uit elk domein van het forest. Restricties op zo een groep is dan ook geldig op elk domein van het forest.
        - Filteren van group policies, nu niet alleen op basis van beveiligingsgroepen, maar ook met behulp van WMI scripts.
    - **Windows Server 2008.**
        - Opnieuw aanvullende schema klasse en attributen.
        - Encryptie van het Kerberos protocol met langere sleutels.
        - Fijnkorrelig wachtwoordbeleid, zodat wachtwoordrestricties niet langer globaal zijn voor het gehele domein, maar specifiek ingesteld kunnen worden voor individuele gebruikers of groepen.
        - Replicatie van DFS namespaces en van de SYSVOL share met behulp van DFS Replication.
- **Forest functioneel niveau.**
    - **Windows 2000 mixed.** Geen voordelen, standaardfunctionaliteit.
    - **Windows 2000 native.** Geen voordelen, standaardfunctionaliteit.
    - **Windows Server 2003.**
        - Het hergebruiken van gedeactiveerde attributen en klassen.
        - Dynamische hulpklassen.
        - Dynamische objecten, met een beperkte levensduur.
        - Efficiënte replicatie van de global catalog gegevens.
        - Het veranderen van de naamgeving en de hiërarchische structuur van domeinen in een forest.
        - Transitieve vertrouwensrelaties tussen verschillende forests.
        - Read-only Windows Server 2008+ domeincontrollers.
        - Efficiëntere KCC algoritmen om de replicatietechnologie te herconstrueren.

        - ◇ Replicatie van de individuele waarden van multi-valued attributen.
    - ○ **Windows Server 2008.** Geen extra functionaliteit ten opzichte van Windows Server 2003.

3. Hoe kan men detecteren op welk niveau een Active Directory omgeving zicht bevindt?

    - Het attribuut **msDS-Behaviour-Version** geeft voor zowel op domeinniveau als forestniveau aan welk functioneel niveau beschikbaar is.

4. Op welke diverse manieren kan men het functionele niveau verhogen of verlagen?

    - Omschakelen naar een bepaald functioneel niveau gebeurd steeds manueel en kan enkel opwaarts: het is niet mogelijk om een hoger niveau om te vormen naar een lager niveau. Er moet steeds rekening gehouden worden met de restricties die elk niveau oplegt, indien deze niet voldaan zijn heeft het geen zin om te verhogen, en zal dit ook met een gepaste foutmelding getoond worden. Een niveau aanpassen kan enerzijds rechtstreeks, door de attributen van het domeinobject te manipuleren, ofwel via een GUI met behulp van de Active Directory Domains and Trust snap-in, beschikbaar in **domain.msc**.

2.2 Active Directory replicatie (§2.5)

1. Wat is de bedoeling van *replicatie*?

    - Gebruikers en services moeten op elk gewenst moment vanaf elke computer in het forest toegang kunnen krijgen tot de directory gegevens. Een domein zonder actieve domeincontrollers functioneert niet langer naar gebruikers toe. Doordat in één domein met meerdere domeincontrollers kan gewerkt worden, worden de fouttolerantie en de belastingsverdelingen verbetert.

2. Hoe wordt dit in Windows Server (ondermeer ten opzichte van NT 4.0) gerealiseerd: bespreek de verschillende *technische kenmerken* en *concepten* van Windows Server replicatie, en hoe men specifieke problemen vermijdt of oplost.

    - Active Directory maakt gebruik van multi-master replicatie, zodat de directory kan bijgewerkt worden vanaf elke domeincontroller, behalve read-only domeincontrollers. NT 4.0 daarentegen maakt gebruik van een master-slave model met primaire domeincontrollers en back-up domeincontrollers die gebruikt worden om de SAM gegevens, de policies, de gebruikersprofielen en de logon scripts te distribueren. In het master-slave model had slechts één enkele server, de primaire domeincontroller, een wijzigbare kopie van de directory. De andere domeincontrollers werden ingeschakelt voor het afhandelen van aanvragen, inclusief aanvragen van gebruikers voor wijzigingen. In huidige Windows Server systemen zijn alle Windows Server domeincontrollers equivalent. Dit biedt meer fouttolerantie omdat met meerdere domeincontrollers de replicatie kan voortgezet worden als één of meerdere andere domeincontrollers uitvallen.
    - Een ander verschil met de replicatietechniek van NT 4 wordt store-and-forward replicatie genoemd. Elke verandering op een domeincontroller wordt slechts uitgewisseld met enkele ander domeincontrollers, die op hun beurt de wijzigingen communiceren met nog enkele andere domeincontrollers. Hoe de domeincontrollers weten naar welke andere domeincontrollers ze hun wijzigingen moeten doorsturen, gebeurd via de **KCC (Knowledge Concistency Checker)** software die op elke Active Directory domeincontroller beschikbaar is. De KCC's proberen steeds een topologie te vormen die ten minste twee paden met elke domeincontroller mogelijk maakt, waarbij elke controller maximaal met drie andere controllers rechtstreeks is verbonden, en waarbij het aantal hops tussen twee willekeurige

domeincontrollers hoogstens drie is. Deze replicatietopologie wordt periodiek of via een trigger (bv een domeincontroller wordt toegevoegd) vernieuwd.

- Het laatste verschil tussen NT 4 en Windows Server is de kleinste replicatie eenheid. In NT 4 is dat altijd het volledig object. In forests met Windows 2000 functioneel niveau de volledige waarde van een individueel attribuut, en in forests met minimaal Windows Server 2008 functioneel niveau zelfs de atomaire waarde van het attribuut.
- Windows Server replicatie gebruikt een pull mechanisme. Elke domeincontroller brengt wel zijn intra-site replicatiepartners op de hoogte indien er wijzigingen in de eigen directory aangebracht zijn, maar hij stuurt deze wijzigingen niet op eigen initiatief door. Het opvragen van de gegevens wordt geïnitieerd door de replicatiepartners. Deze methode is zeer foutbestendig. Indien een update mislukt, vraagt de domeincontroller gewoon opnieuw om de gegevens. Verder wordt er niet bij elke wijziging de replicatiepartners verwittigd. De wijzigingen worden gegroepeerd in een bepaald tijdsinterval. Dit heeft als voordeel dat het netwerk minder belast zal worden. Dit wordt propagation damping of replication latency genoemd. Active Directory garandeerd daarom niet dat elke domeincontroller over de meest actuele directory beschikt, maar dat ze uiteindelijk wel zal convergeren, daarom noemt men dit een loose consistency model. Propagation damping is uitgeschakeld voor sommige attributen die te maken hebben met beveiliging. Gewijzigde lockout kenmerken en wachtwoorden worden bijvoorbeeld binnen de 15 seconden gerepliceerd. Dit noemt men urgent replication en wordt in eerste instantie met de PDC emulator uitgevoerd.
- Windows Server Replicatie is multi-threaded: een domeincontroller kan simultaan repliceren met diverse partners. Bij replicatie tussen controllers in dezelfde site worden gegevens niet gecomprimeerd om de verwerkingskracht van de domeincontrollers minder te belasten.
- Een obstakel is het replicatieverkeer, dat moet relatief beperkt blijven. Daarom wordt in Active Directory enkel gewijzigde directory gegevens gerepliceerd. Om te voorkomen dat een domeincontroller meerdere malen dezelfde wijziging zal doorvoeren, wordt zijn **USN (Update Sequence Number)** verhoogd. De combinatie van USN en GUID van een domeincontroller wordt zijn Up-To-Dateness Vector (UTD vector) genoemd. Elke domeincontroller meldt bij elke wijziging aan een object zijn huidige UTD vector aan de andere domeincontrollers, en houdt in de eigen UTD vectortabel de meest recente UTD vector bij die hij van elke andere controller heeft ontvangen, en waarvan hij de wijzigingen heeft verwerkt.
- Een ander probleem doet zich voor wanneer dat hetzelfde kenmerk van een object op meer dan één domeincontroller quasi tegelijkertijd gewijzigd wordt. De domeincontroller die conflicterende wijzigingen detecteert, lost dit op door naar het tijdstip van wijziging te kijken, en de wijziging met het oudere tijdstip te negeren. Zijn de tijdstippen gelijk, wordt enkel de wijziging van de domeincontroller met het hoogste GUID geaccepteerd. Voor sommige objecten, waarvoor de kans op conflicten klein is door bovenstaande methode, is dit nog altijd niet toelaatbaar, en wordt er beroep gedaan op slechts één domeincontroller, de operations master om de wijzigingen door te voeren.
- Men moet vermijden dat een object opnieuw gecreëerd wordt door replicatie wanneer deze verwijdert wordt. Daarom wordt een object niet onmiddellijk uit Active Directory verwijderd, maar eerst als tombstone gemarkeerd en in een hidden container, *Deleted Items*, geplaatst. Na 60 dagen (180 dagen in Windows Server 2008+ domeinen) wordt een object definitief verwijderd.

3. Welke toestellen repliceren onderling in een *forest*? Welke specifieke gegevens worden hierbij uitgewisseld?

   - Enkel de domeincontrollers repliceren onderling waarbij zowel de directory objecten en corresponderende kenmerken voor het domein, als het schema en de configuratiegegevens van het forest worden uitgewisseld.

4. Welke impact hebben *sites* met betrekking tot de replicatie van Active Directory gegevens? Welke andere Active Directory aspecten worden door sites beïnvloed? (§2.6.1)

   - Replicatiepartners van verschillende sites laten standaard het meldingsmechanisme van gewijzigde UTD vectortabellen achterwege met als gevolg dat er enkel polling kan toegepast worden. Het pollingsinterval staat standaard op 3 uur, maar kan ingekort worden tot 15 minuten.
   - Gegevens worden standaard gecomprimeerd, alhoewel men dit voor elk verbindingsobject kan uitschakelen.
   - Met behulp van sitekoppelingen kan aangegeven worden hoe de verschillende sites onderling, met rechtstreekse fysieke netwerkverbindingen, verbonden zijn. De KCC's genereren automatisch enkel verbindingsobjecten tussen sites als er tussen beide sites een sitekoppeling bestaat. De verbindingsobjecten kunnen dan de feitelijke netwerkverbindingen gebruiken om directory gegevens uit te wisselen.
   - Een aspect dat gewijzigd wordt is de KCC. Om te vermijden dat er meerdere verbindingsobjecten van dezelfde sitekoppeling gebruik maken, waarbij a priori willekeurige domeincontrollers gebruikt worden om informatie tussen de sites uit te wisselen, introduceert men de **ISTG (Inter-site Topology Generator)**. Enkele domeincontrollers krijgen de rol van ISTG, waardoor de functionaliteit van hun KCC wijzigt. De ISTG zorgt ervoor om voor elke partitie hoogstens één verbindingsobject per sitekoppeling aan te maken. De domeincontrollers die van dit uniek verbindingsobject gebruik maken worden bruggenhoofd servers genoemd. Dit zijn de domeincontrollers waar gegevens tussen sites uitgewisseld worden. Elk paar site heeft dus zijn eigen paar domeincontrollers. Door deze invoering zorgt de ISTG ervoor dat de KCC niet de standaard intra-site richtlijnen volgt van de replicatietechnologie, maar zich optimaal zal aanpassen aan de siteconfiguratie.

5. Hoe wordt bepaald *tot welke site* computers, servers in het bijzonder, behoren? (laatste paragraaf §2.6.2 en fractie §2.6.3)

   - Computers worden aan sites toegewezen op basis van de locatie in een IP subnet. Een site kan dus beschouwd worden als een verzameling computers in een of meer IP subnetten. Alle computers die geadresseerd worden door hetzelfde IP subnet maken automatisch deel uit van dezelfde site.
   - Voor een server wordt de locatie bepaald door de vraag tot welke site in de directory het server object van de domeincontroller behoort. Elke site heeft een container met de naam Servers, die alle domeincontrollerobjecten bevatten die in deze site zijn geplaatst. Tijdens de promotie van een server tot domeincontroller wordt de server automatisch toegevoegd aan de site waaraan het subnet, waartoe de server behoort, is gekoppeld. De site van een server ligt hierdoor vast, en kan enkel manueel veranderd worden met **dssite.msc**.

2.3 Gedeelde mappen en NTFS

1. Welke *configuratieinstellingen* kun je maken tijdens of onmiddellijk na het creëren van gedeelde mappen? Bespreek het *doel* van elk van deze diverse instellingen en de belangrijkste *eigenschappen* en *mogelijkheden* ervan. (§3.2.1, §3.2.2, fracties §3.3.1, §3.4.2, §3.4.3, §3.5 en §3.6)

   - **Limit the number of simultaneous users:** Aanduiden hoeveel gebruikers er op hetzelfde moment toegang mogen hebben tot de share. Dit kan nuttig zijn om software, waarvoor een licentie aangekocht is, tot slechts een beperkt aantal gelijktijdige gebruikers, te limiteren.

- **Beheer van shares:**
- **Share machtigingen:**
- **Diskquota's:**
- **File screens:**
- **Distributed File System:** DFS lost twee problemen op: het kan op verschillende servers shares bevatten met exact dezelfde inhoud en laat ook toe om het hele netwerk te bundelen in één enkele naamruimte.
- **Client-side caching:** Client-side caching cachet automatisch vaak gebruikte netwerkbestanden. De cachelocatie is een map op de lokale harde schijf van de gebruiker met als naam *Offline Files*. Als een bestand aangepast wordt, wordt het zowel op het netwerk als in de Offline Files map geplaatst . Bij het openen van dit bestand worden de timestamps van beide versies vergeleken, en wordt enkel de gecachte versie genomen indien beide timestamps gelijk zijn. Caching is een handige manier om het openen van bestanden te versnellen. Ook bij het uitvallen van een netwerk is caching zeer handig. De gebruiker kan doorwerken, en als het netwerk terug bereikbaar is wordt het bestand automatisch naar het netwerk verstuurd. Als het blijkt dan een lokaal bestand en een bestand op het netwerk beiden gewijzigd zijn, moet de gebruiker zelf de juiste versie aanduiden.

  Op een werkpost kan gekozen worden tussen twee methoden:

  - Passieve caching: Netwerkbestanden worden automatisch, maar tijdelijk, gecacht van zodra ze worden gebruiket. Bij het overschijven van de toegestane limiet, zal client-side-caching automatisch de oudste gecachte bestanden verwijderen.
  - Actieve caching: Deze methode laat de gebruiker toe om individuele bestanden te cachen, door dit expliciet aan te geven op het bestand zelf.

  Op een server kan client-side-caching op elk individueel share niveau geconfigureerd worden. Belangrijke opties zijn:

  - ... will not be available offline.
  - Only the files and programs ...
  - All files and programs ...

2. Waar wordt de definitie en (partiële) configuratie van gedeelde mappen *opgeslagen*? Hoe kan men deze wijzigen vanuit een *Command Prompt*?

   ToDo: Oplossen

3. Geef een overzicht van de belangrijkste voordelen van de opeenvolgende versies van het *NTFS bestandssysteem*. Bespreek elk van deze aspecten (ondermeer het doel, de voordelen en de beperkingen ervan), en geef aan hoe je er gebruik kan van maken, bij voorkeur vanuit een *Command Prompt*. (NTFS fractie §1.6, fracties §3.4.1, §3.4.2 en §3.4.4)

   - **NTFS 1.2**
     - **Beveiliging tot op bestandsniveau**: NTFS is het enige Windows bestandssysteem dat voldoende beveiliging biedt voor gebruikers die zich lokaal aanmelden.
     - **Logging van schijfactiviteiten**: Schijfactiviteiten worden vastgelegd in logboekbestanden, kan de oorspronkelijke situatie na een stroomonderbreking herstelt worden.
     - **Grotere volumes, zonder performantiedegradatie**: Naarmate de stationsgrootte toeneemt, nemen de prestaties met NTFS niet af zoals met FAT.
     - **Compressie**: Alleen NTFS volumes ondersteunen compressie. Via het commando `compact` kan je bestanden of folders compresseren. Voer zelf het commando 'compact /?' uit en studeer de opties /C, /F, /I en /S.
     - **Hardlinks**: Je kan diverse mappen in een volume hetzelfde bestand laten bevatten, zonder dit bestand fysiek te dupliceren. Om een hard link naar een bestand te creëren gebruik je ofwel `fsutil hardlink create linkbestand bronbestand` of anders `mklink /H linkbestand bronbestand`

- **Dynamisch uitbreiden van partities/volumes**:_ToDo: vinden (`diskpart extend`)
- **Spanned volumes**: _ToDo: vinden

- **NTFS 3.0**
  - **Mounten van volumes in NTFS mappen** (`mountvol` of `diskpart assign`)
  - **Symbolische links**: Een symbolische link naar een directory, eventueel in een ander volume, kan aangemaakt worden met het commando `linkd linkmap bronmap`. Om ook een symbolische link naar een bestand te maken gebruik je `mklink linkbestand bronbestand`. Dit commando werkt ook met directories met de /D optie: `mklink /D linkmap bronmap`.
  - **Sparse bestanden**: Indien een bestand sparse is, kan NTFS enkel schijfruimte toewijzen aan de delen van grote bestanden waarnaar effectief wordt geschreven. Om een bestand als sparse te markeren, moet je het commando `fsutil sparse setflag bestandsnaam` gebruiken.
  - **File markers voor Remote Storage Service**: _ToDo: vinden
  - **Transparante encryptie en decodering**: _ToDo: §3.4.1 (`cipher /e /a /s:...`)
  - **Individuele diskquota op volumeniveau**: Voor elke gebruiker kan er een diskquota op volumeniveau ingesteld worden, zodat er een maximale beschikbare opslagcapaciteit voor die gebruiker in een specifiek volume is. Om quotabeheer in te schakelen op een specifiek volume gebruik je het commando `fsutil quota enforce volumenaam`. Om na te gaan of er gebruikers over de limiet zitten, gebruik je het commando `fsutil quota violations`. Informatie over het gebruik van volumes wordt geregistreerd per security ID en niet per accountnaam.

- **NTFS 3.1**
  - **Historische versies van bestanden**: De Volume Shadow Copy service (VSS), geïntroduceerd door Windows Server 2003+, gecombineerd met de reparsepunt technologie van NTFS 3+ biedt de Shadow Copy Restore functionaliteit aan. VSS neemt periodieke snapshots van alle bestanden in een volume, en houdt een databank bij van incrementele wijzigingen aan deze bestanden. Deze databank wordt bijgehouden in een verborgen en geëncrypteerde map System Volume Information.Om VSS in te stellen gebruik je het `vsadmin` commando.
  - **Self-healing correctie van corrupties**: Eventuele corrupties worden online opgespoord en gecorrigeerd, waardoor de continue beschikbaarheid van het systeem verhoog, en gegevensverlies geminimaliseerd wordt.
  - **Ondersteuning van transacties**: Een transactie is een reeks opeenvolgende bewerkingen op hetzelfde volume. Een transactie is atomair, zodat op het einde van een transactie ofwel alle wijzigingen doorgevoerd worden, of dat alle wijzigingen ongedaan gemaakt worden. Dit systeem vereenvoudigt de implementaties van transacties door RDBMS'en zoals SQL Server.
  - **Globale diskquota op mapniveau**: Deze diskqouta's kunnen gelden op individuele mappen, maar zijn onafhankelijk van de eigenaar. Elke individuele bijdrage van een gebruiker tot de map zal een impact hebben op het al dan niet overschrijden van de limiet. Een lijst van mapquota's kan opgevgraagd worden met `dirquota template list`. Om een mapquota aan te maken gebruik je dan `dirquota quota add /path:map`
  - **File Screens**: Een file screen verhindert dat in een specifieke maphiërarchie bestanden met bepaalde extensies kunnen opgeslagen worden, en kunnen reacties triggeren als een gebruiker poogt dat toch te doen. OM file screens te beheren kan gebruikt gemaakt worden van het commando `filescrn`.

2.4 Machtigingen op bestandstoegang (§3.3)

1. Welke rol spelen machtigingen bij de beveiliging van bronnen? Geef een gedetailleerd overzicht van het *algemeen* (op alle windows objecten toegepast) mechanisme van *machtigingen*.

   - Bij elke bron moet er bepaald worden wie er toegang heeft tot welke gegevens, en wat die er mee kan doen. Elk object in Active Directory, elk object van een NTFS volume, elke registersleutel, elk proces en ook elke service heeft een security descriptor. Deze bevat:
     - een machtigingsset = Access Control List,
     - de System Access Control List definieert welke acties van welke gebruikers door het auditing systeem gelogd worden,
     - een identificatie (de SID) van de eigenaar van het object. De eigenaar of een beheerder beheert het installen van de ACL op het object.
     - de primaire groep van de maker van het object. Deze identificatie is enkel van belang voor compatibiliteit met POSIX toepassingen.

     De ACL is een verzameling machtigingen die gelden voor bepaalde gebruikers of groepen. Elke machtiging in de ACL wordt een Access Control Entry genoemd. Voor elke gebruiker of groep kan er een ACE zijn die een machtiging ontzegt of een machtiging toekent. Alle ACEs vormen samen de ACL. Eerst worden alle ACEs die machtigingen ontzeggen behandeld, vervolgens pas de ACEs die machtigingen toekennen. Van zodra een specifieke machtiging in een ACE ontzegt is, wordt een eventuele toekenning van de machtiging via een andere ACE genegeerd. Indien een gebruiker tot meerdere groepen behoort, worden alle machtigingen van die groep toegepast op de gebruiker.

     Er bestaan twee soorten machtigingen:

     - Expliciete machtigingen worden rechtstreeks aan een object gekoppeld.
     - Overgenomen machtigingen worden doorgegeven door een bovenliggend object in de hiërarchie.

     De expliciete machtigingen hebben steeds voorrang op overgenomen machtigingen, zelfs op machtigingen die ontkent worden.

2. Bespreek hoe het mechanisme van machtigingen *specifiek* (en op diverse niveaus) *toegepast* wordt op *bestandstoegang*. Geef de verschillende soorten machtigingen, hun onderlinge relaties, en hoe deze kunnen *geanalyseerd* en *ingesteld* worden. Toon hierbij aan dat je zelf met deze configuratietools geëxperimenteerd hebt.

   - Voor bestandstoegang kunnen zowel machtigingen ingesteld worden op share niveau als op NTFS niveau. Verzoeken van gebruikers kunnen onafhankelijk van de NTFS machtigingen door de share machtiging geweigerd worden.
   - **Share niveau**

     Bestandstoegang op share niveau vormt de eerste beveiligingslaag. Elke share heeft toegewezen machtigingen. Om die te bekijken of wijzigen moet de Permissions Editor opgestart worden, bv door `compmgmt.msc` op te starten, de properties opvragen van een share, de tabpagina Sharing selecteren en de Advanced Sharing en Permissions knoppen te selecteren.

     Op share niveau kunnen er drie machtigingen gedefinieerd worden:

     - **Full Control**: Een gebruiker of groep kan alle bewerkingen uitvoeren op alle bestanden en mappen, en op de share zelf.
     - **Read**: Dit laat om om de volledige structuur van de share te zien, om mappen en bestanden te lezen, en om toepassingen in die maphiërarchie uit te voeren.
     - **Change**: Dit laat ook toe om bestanden te wijzigen en te verwijderen, en er de bestandskenmerken van te wijzigen. De share zelf en NTFS machtigingen kunnen niet gewijzigd worden.

- **NTFS niveau**:
  NTFS machtigingen bieden een tweede (beveiliging op mappen) en derde (beveiliging op bestanden) beveiligingslaag aan. Op NTFS niveau bestaan er twee subniveaus:
  - **De atomaire machtigingen**: Deze 13 machtigingen vormen de bouwstenen voor het hogere niveau. De atomaire machtigingen zijn de kleinst mogelijke machtigingen die je kan instellen, en bieden de mogelijkheid om zeer nauwkeurig het toegangsniveau te bepalen. de 13 opties op p85 moet je niet kennen, enkel mechanisme uitleggen.
  - **De moleculaire machtigingen**: Dit niveau omvat zes veel gebruikte combinaties van atomaire machtigingen. ook hier moeten de opties niet gekend zijn.

  NTFS machtigingen kunnen ingesteld worden met de ACL editor, die kan opgeroepen worden via de Security tabpagina van een wilekeurige map of bestand. De security tabpagina bevat een lijst van gebruikers of groepen waarop een ACL gedefinieerd is voor het geselecteerde object. Op een gebruiker of groep klikken geeft de overeenkomstige moleculaire machtigingen in een andere lijst. Grijze selectievakjes duiden overgeërfde machtigingen aan. De knop advanced geeft een nieuw venster met 3 tabbladen:
  - **Permissions**: Dit tabblad geeft opnieuw de lijst van gebruikers of groepen. Er kunnen gebruikers of groepen toegevoegd, verwijderd, of geinspecteerd worden. Door een gebruiker of groep te selecteren en vervolgens op View te klikken, komen terug de 6 moleculaire machtigingen tevoorschijn. Om de atomaire machtigingen te zien klik je op Show Advanced Permissions.
  - **Auditing**: Dit laat toe om de SACL in te stellen voor het specifieke object. Elke atomaire machtiging kan ingesteld worden, en ook of dat de machtiging gelukt of gefaald is.
  - **Effective Access**: Dit tabblad laat toe om de machtigingen van een bepaalde gebruiker of groep te bekijken via hetzelfde algoritme dat de Security Reference Monitor hanteert om de uiteindelijke machtigingen op een object na te gaan.

3. Wat gebeurt er met de machtigingen bij het *verplaatsen* van een bestand? Wat gebeurt er met de machtigingen bij het *kopiëren* van een bestand?

   - In beide gevallen wordt de gebruiker die de verplaats- of kopieeropdracht uitvoert, automatisch de eigenaar van de objecten.
   - **Verplaatsen**: Een bestand of map verplaatsen naar een container binnen hetzelfde volume heeft geen impact op de explicitie machtigingen die zijn toegewezen aan het object, en blijfen dus behouden. Indien deze verplaatst worden naar een andere container binnen een ander volume, dan krijgen de objecten dezelfde machtigingen als hun doelcontainer.
   - **Kopiëren**: Een bestand of map kopiëren naar een container op hetzelfde of een ander volume, vervallen de expliciete machtigingen, en worden de machtigingen van de doelcontainer overgenomen.

4. Op welke *andere objecten* zijn machtigingen van toepassing?

   - Op elk object in Active Directory, elke registersleutel, elk proces en elke service.

5. Wie is *in principe* verantwoordelijk voor het configureren van machtigingen? Door welke instelling is dit zo vastgelegd? Hoe kan ervoor gezorgd worden dat enkel *administrators* verantwoordelijk gesteld worden voor het configureren van machtigingen?

   ToDo: Oplossen

2.5 Gebruikersgroepen (§4.2.2 en §4.2.3)

1. Bespreek in detail het onderscheid tussen de diverse soorten *veiligheidsgroepen*, ondermeer afhankelijk of het toestel al dan niet in een domein is opgenomen. Behandel hierbij vooral de mogelijkheden en beperkingen. Bespreek ondermeer:

   - de *zichtbaarheid* van de diverse soorten groepen,
   - welke objecten er *lid* van kunnen zijn,
   - de onderlinge relaties en de regels voor het *nesten* van de diverse soorten groepen? Stel deze relaties eveneens schematisch voor.

   - **Lokale veiligheidsgroep:** Lokale groepen kunnen leden uit elk domein van het forest of een ander trusted domein bevatten. Lokale groepen zijn enkel zichtbaar en geldig in het eigen domein. Lokale groepen worden dan ook niet gekopieerd naar de global catelog. Lokale groepen kunnen behalve gebruikers, ook globale en universele groepen uit om het even welk domein van het forest omvatten, maar enkel lokale groepen uit hetzelfde domein.
   - **Globale veiligheidsgroep:** Globale groepen kunnen alleen gebruikers en andere globale groepen uit hetzelfde domein omvatten. Ze zijn zichtbaar in elk domein van het forest of een ander trusting domein. Lokale en universele groepen kunnen geen deel uitmaken van een globale groep. Deze groep wordt vaak gebruikt als container voor gebruikers die dezelfde machtigingen of rechten nodig zullen hebben.
   - **Universele veiligheidsgroep:** Universele groepen kunnen leden hebben uit elk domein van het forest, maar niet uit een ander trusted domein, en zijn zichtbaar in elk domein van het forest, maar niet in een ander trusting domein. Een universele groep word, zoals een lokale groep, typisch gebruikt om rechten en machtigingen toe te kennen, maar bieden er tegenover het voordeel dat ze in alle domeinen tegelijkertijd geldig zijn, en bijgevolg slechts éénmaal moeten gedefinieerd worden. Een universele groep kan andere universele groepen, en gebruikers en globale groepen uit alle domeinen van het forest bevatten. Lokale groepen mogen niet opgenomen worden.

2. Hoe en waarom worden deze soorten groepen *in de praktijk* best gebruikt, al dan niet gecombineerd? Van welke omstandigheden is dit afhankelijk? Illustreer aan de hand van concrete voorbeelden.

   - **Lokale groepen** worden typisch gebruikt om rechten en machtigingen toe te kennen, door voor elke bron of verzameling bronnen één of meerdere lokale groepen te creëren en de toegang tot die bron in te stellen door één keer toegang te verlenen aan de lokale groep. De toegang tot de bron wordt nadien enkel gewijzigd door het lidmaatschap te manipuleren. Lokale groepen zijn niet alleen zichtbaar op domeincontrollers, maar ook op alle werkposten en lidservers. Dit neemt de noodzaak weg om uit veiligheidsoverwegingen alle lidservers te configureren als domeincontrollers.
   - **Globale groepen** worden eerder gebruikt ales container voor gebruikers die dezelfde machtigingen of rechten nodig zullen hebben. Als een verzameling gebruikers, afkomstig uit een ander domein toegang wil krijgen tot een bepaalde bron, dan wordt deze verzameling gebruikers best gegroepeerd in een globale groep van het andere domein. Deze groep moet dan lid gemaakt worden van een lokale groep die gekoppeld is aan de bron.
   - **Universele groepen** worden net als lokale groepen typisch gebruikt om rechten en machtigingen toe te kennen, maar bieden het voordeel dat ze in alle domeinen tegelijkertijd geldig zijn en dus slechts eenmaal moeten gedefinieerd worden. Zowel de namen als de leden van universele groepen worden in de global catalog opgenomen en kan vrij groot worden. Bij voorkeur bevatten universele groepen enkel globale groepen.

3. Waar en hoe wordt het (volledige) lidmaatschap van een *user* object tot een groep bijgehouden? Op welke diverse manieren kan men dit lidmaatschap *configureren*? Op welke diverse manieren kan men de volledige verzameling van objecten, die deel uitmaken van een specifieke groep, of de volledige verzameling van groepen, waar een specifiek object deel van uitmaakt, achterhalen? (partim §4.1.2 en §4.2.3)

   - In het properties venster van een gebruikersaccount kan in het tabblad Member Of de lidmaatschap tot groepen geconfigureerd worden. Het memberOf attribuut dat hiervoor gebruikt wordt, is gelinkt aan het member attribuut van de groep en is hier de back-link. Dit kan niet rechtstreeks gewijzigd worden, maar enkel via het member attribuut van de groep. In het properties venster van een groep kan in het tabblad Members de groep bevolkt worden. De groep kan zelf deel uitmaken van een andere groep door dit aan te passen in het tabblad Member Of van de groep. De lijst van members wordt steeds opgeslagen in een link-kenmerk met in de groep de forward-link members, en in het object dat deel uitmaakt van de groep de back-link memberOf.

     Groepsleden kunnen ook uit een trusted domein geselecteerd worden. Active Directory maakt hiervoor een phantomobject aan, dat het object uit het trusted domein representeert. Deze phantomobjecten komen terecht in de container ForeignSecurityPrincipals en kunnen lid worden van lokale groepen in het domein.

4. Door *wie* wordt het lidmaatschap van de diverse groeptypes bij voorkeur ingesteld?

   _ToDo: Oplossen

5. Op welke diverse manieren kan men het beheer van Active Directory objecten, specifieke attributen van groepsobjecten in het bijzonder, *delegeren aan niet-Administrators*? Bespreek een aantal technieken om dit delegeren *zo eenvoudig mogelijk* uit te voeren. (partim §4.1.2, en §4.4.2)

   _ToDo: Oplossen