Microsoft Windows 2000 Active Directory Service

Technology Overview

Agenda

- Active Directory Structure
  - Logical
  - Physical
  - Replication Operations
- DNS Integration/Interaction
- Kerberos V5 Functionality

Active Directory Logical Structure

The Active Directory
Goals

- Address customer needs for a Directory Service
  - Hierarchical namespace
  - Partitioning for scalability
  - Multimaster replication
  - Dynamically extensible schema
  - Online backup and restore
  - Open and extensible directory synchronization interfaces
- LDAP as the core protocol for interoperability

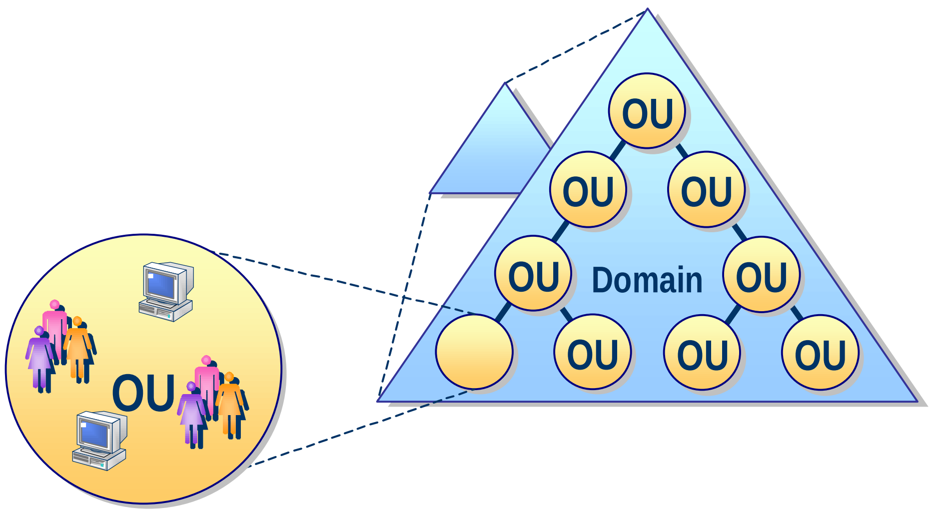
Directory Service

Enabling Technology - Distributed Services

- Replaces registry-based security account manager (SAM)
- 100% backwards compatible
- Adds many new features
  - X.500 and DNS naming
  - LDAP protocol support
  - Domain hierarchy
  - Extensible schema
  - Multimaster replication

DS Structure

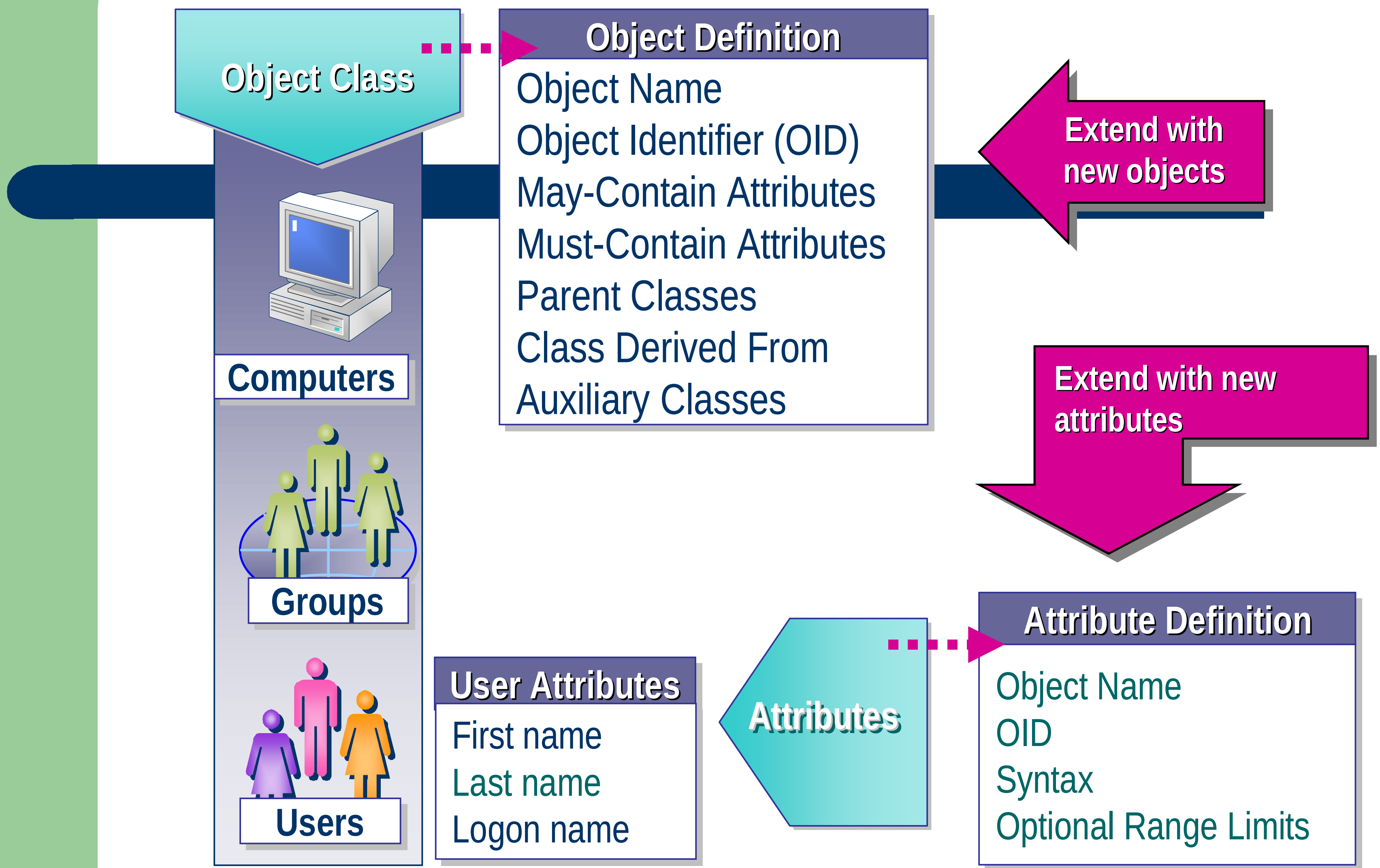
- X500-Like Tree per Domain
- Allows easy segmentation along organizational or geographic lines
- Represents the unit of Replication (Naming Context)
- Enables Granular Administration



DS Schema

- Contains a formal definition of the contents and structure of Active Directory
  - Attributes
  - Classes
- Defines what attributes an instance of a class <u>must</u> have, what additional attributes it <u>may</u> have, and what object class can be a <u>parent</u> of the current object class.

Extensible Schema



Naming Contexts

- Portion of the LDAP namespace
- Specific region inside a DCs database
- Boundary for replication
- Existing NCs:
  - Configuration (Enterprise wide context)
  - Schema (Enterprise wide context)
  - Domains in Enterprise (Domain wide context for full replication, Enterprise wide context for partial replication to Global Catalogs)

Location of Naming Context in the Domain Namespace

- Root of a Domain Namespace is the DNS name of the Domain
- Objects and containers are children of the root
- Configuration is child of the first domain in the enterprise (root domain), Schema is child of Configuration container

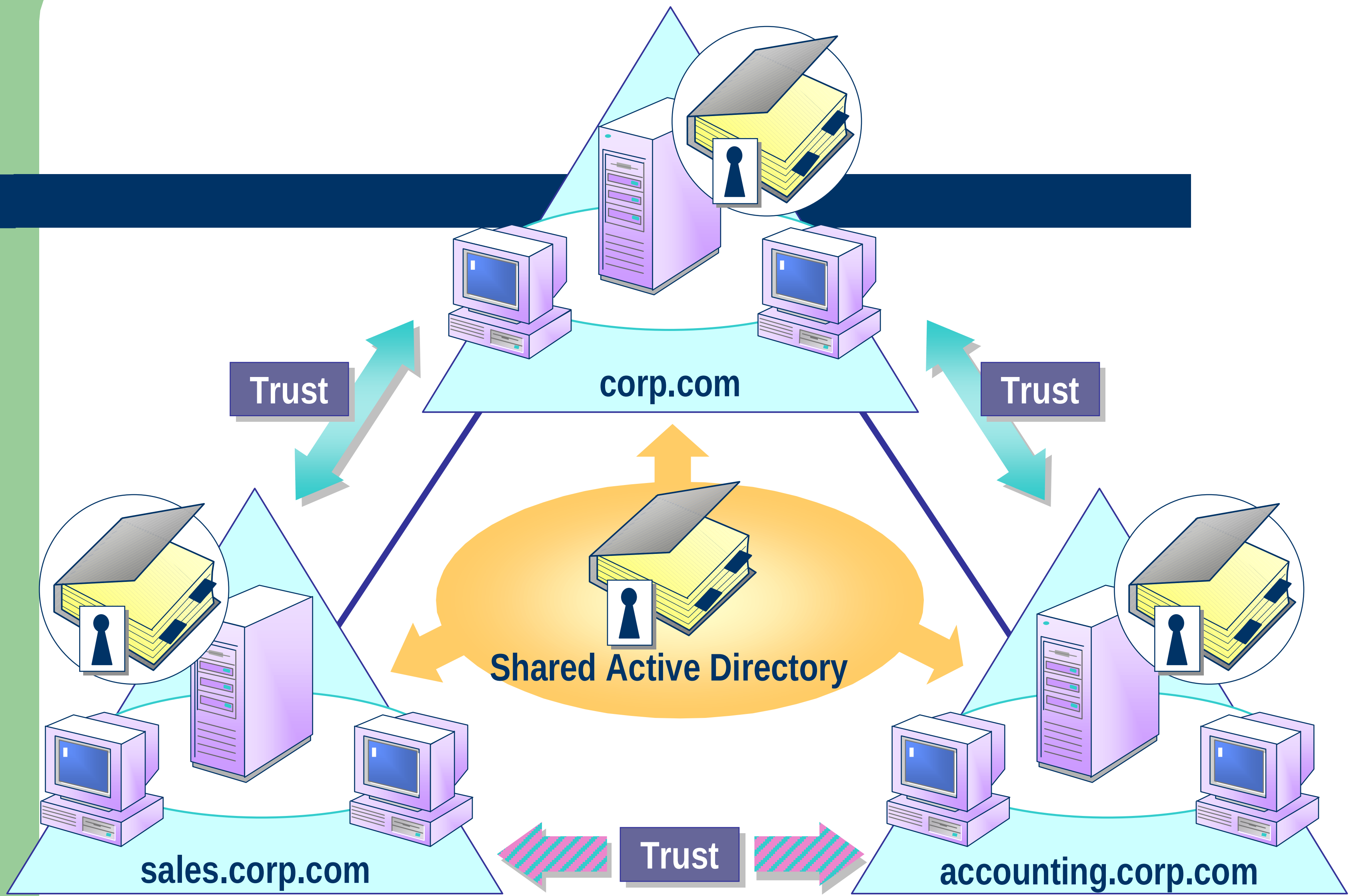
Knowledge References in AD

- All knowledge about the namespace is held by Cross-Ref objects in the Partitions Container
- “Default Referral” supported by:
  - First determining if a value exists for the superiorDnsRoot attribute of the Forest Root, if so referral is generated using this value
  - If not, then best effort to construct a referral using “DC=“ components of DN presented

Domain Tree - Definition

A collection of NT5 domains representative of a contiguous namespace and sharing a common schema and configuration container.

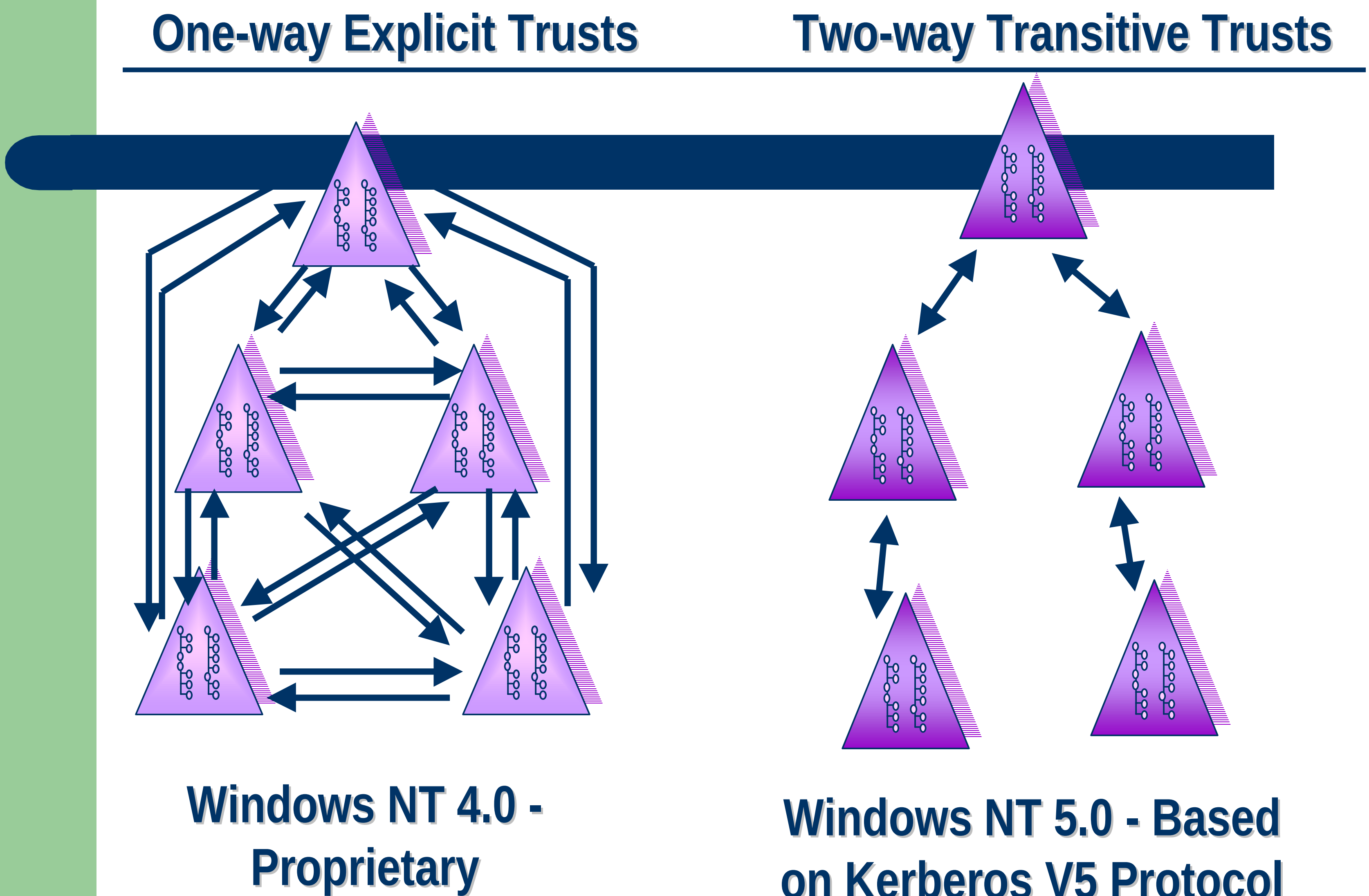
A Domain Tree



Domain Tree

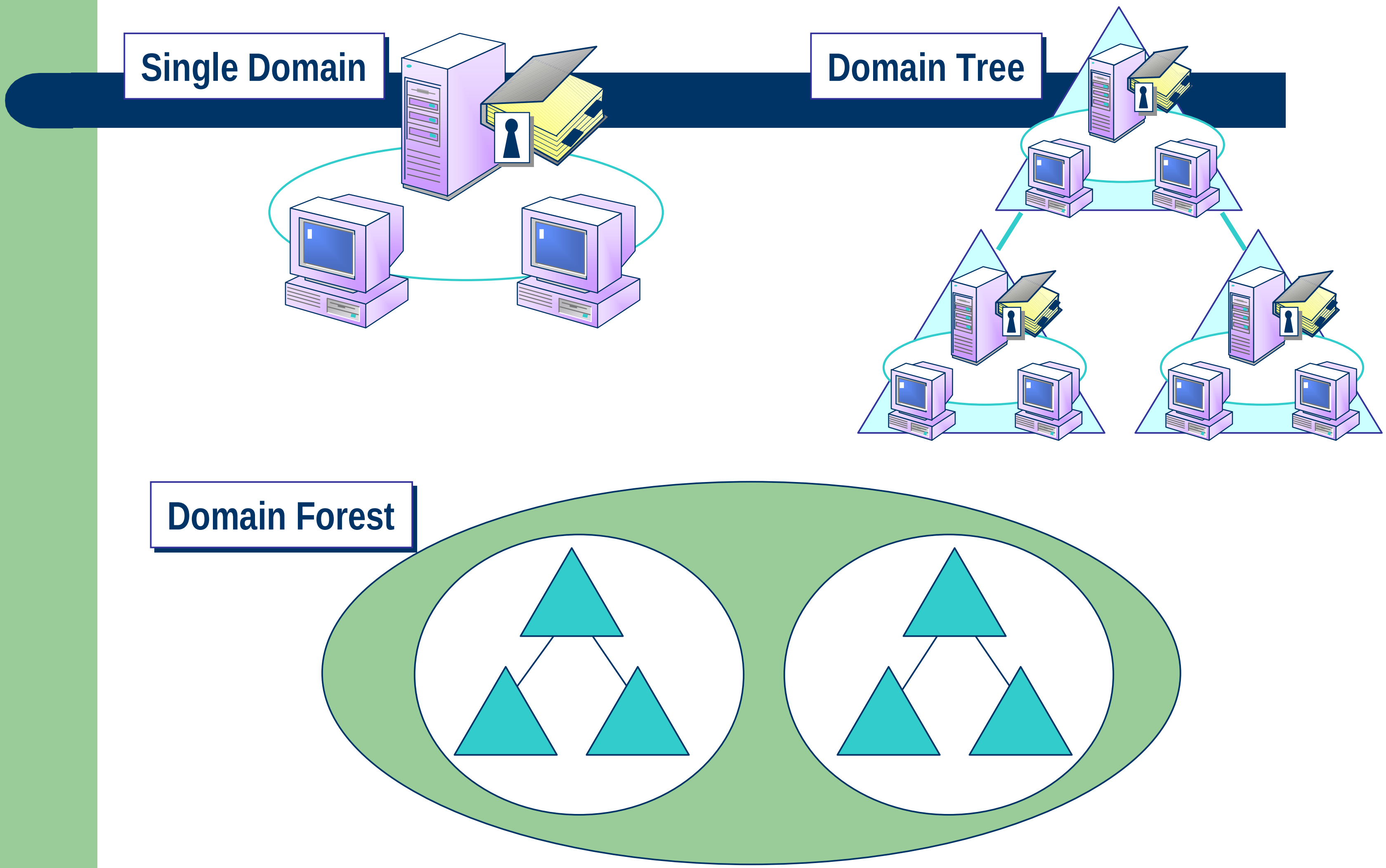
- Tree
  - Renaming parent’s domain will affect all its children’s names.
  - Moving domain in a tree, it will rename the “from” domain and its children.
  - Joining two trees in a forest is possible
  - Deep search on parent’s domain will get referral on its children.

Trust Relationships



Forest

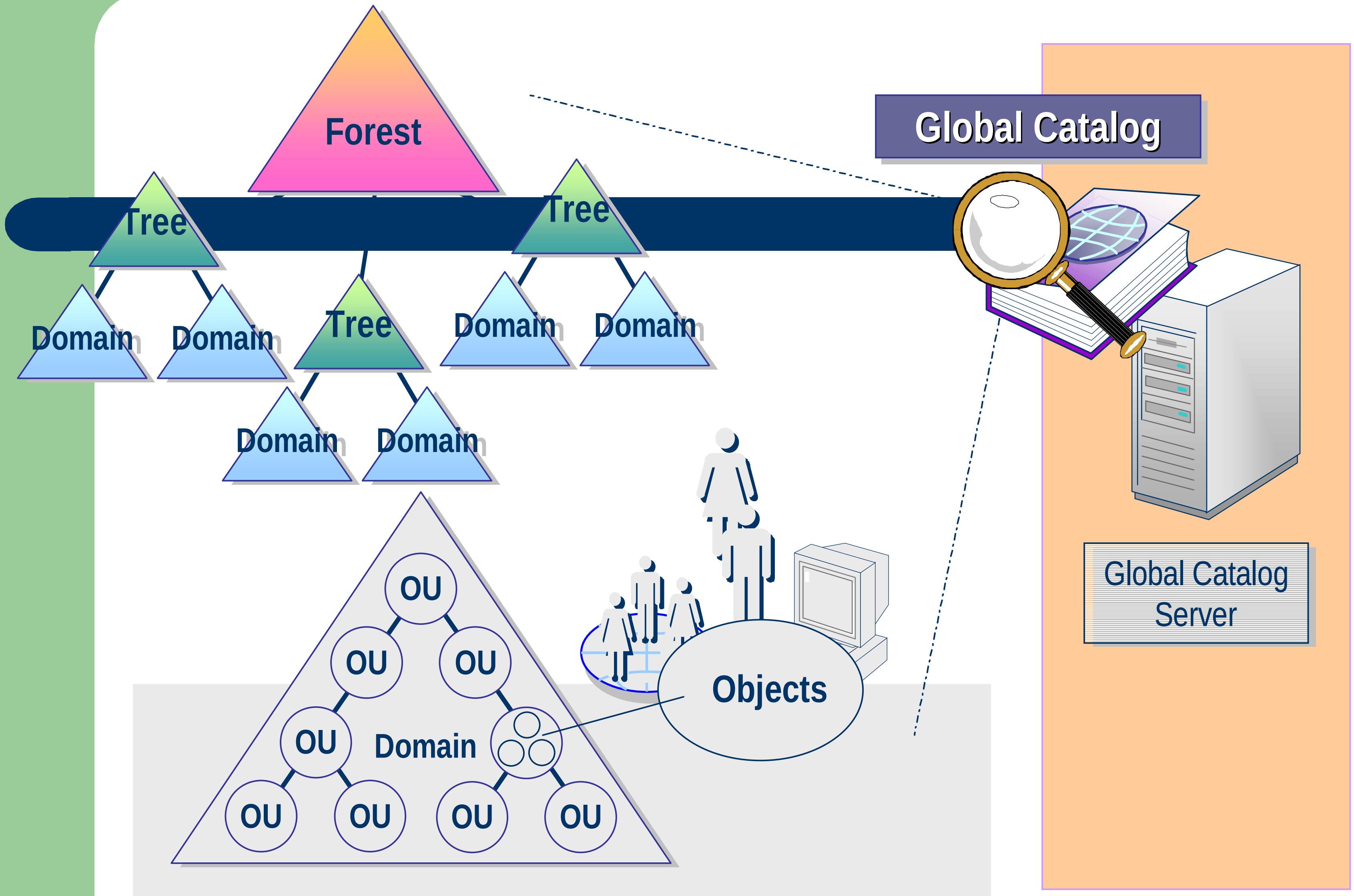
One or More Discontiguous Trees
Intra Tree Relationship: Trust
Share Common Schema & Global Catalog



Global Catalog

- Hosts all objects in the Enterprise
  - It is hierarchical, not flat
- Read-only Mode
- Frequently queried properties enterprise wide are prime candidates for GC
- Port Number is 3268
- Deep search at the root of name space will generate referral to GC

Global Catalog



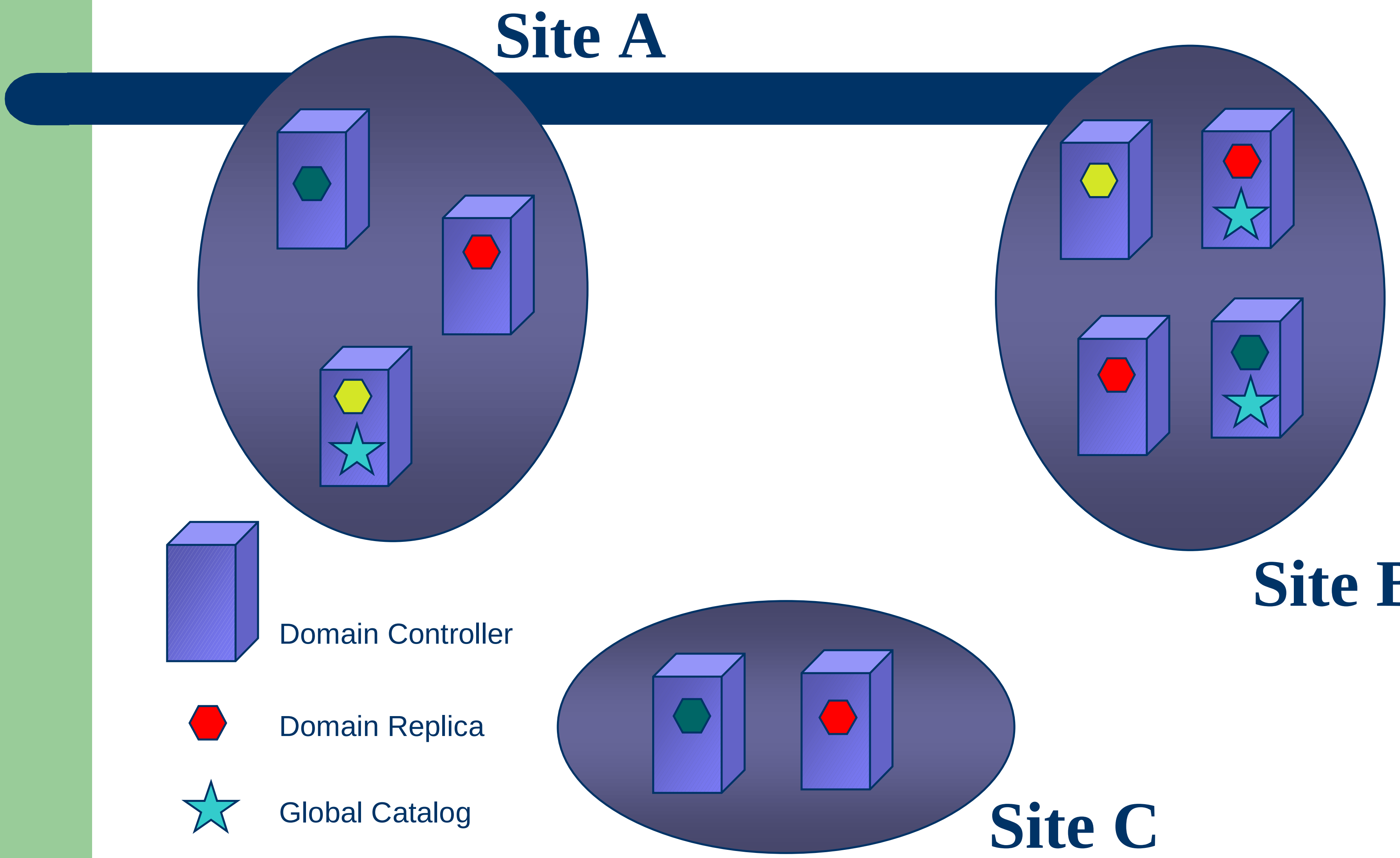
Active Directory Physical Structure And Replication

Active Directory Implementation
Physical Organization: Sites

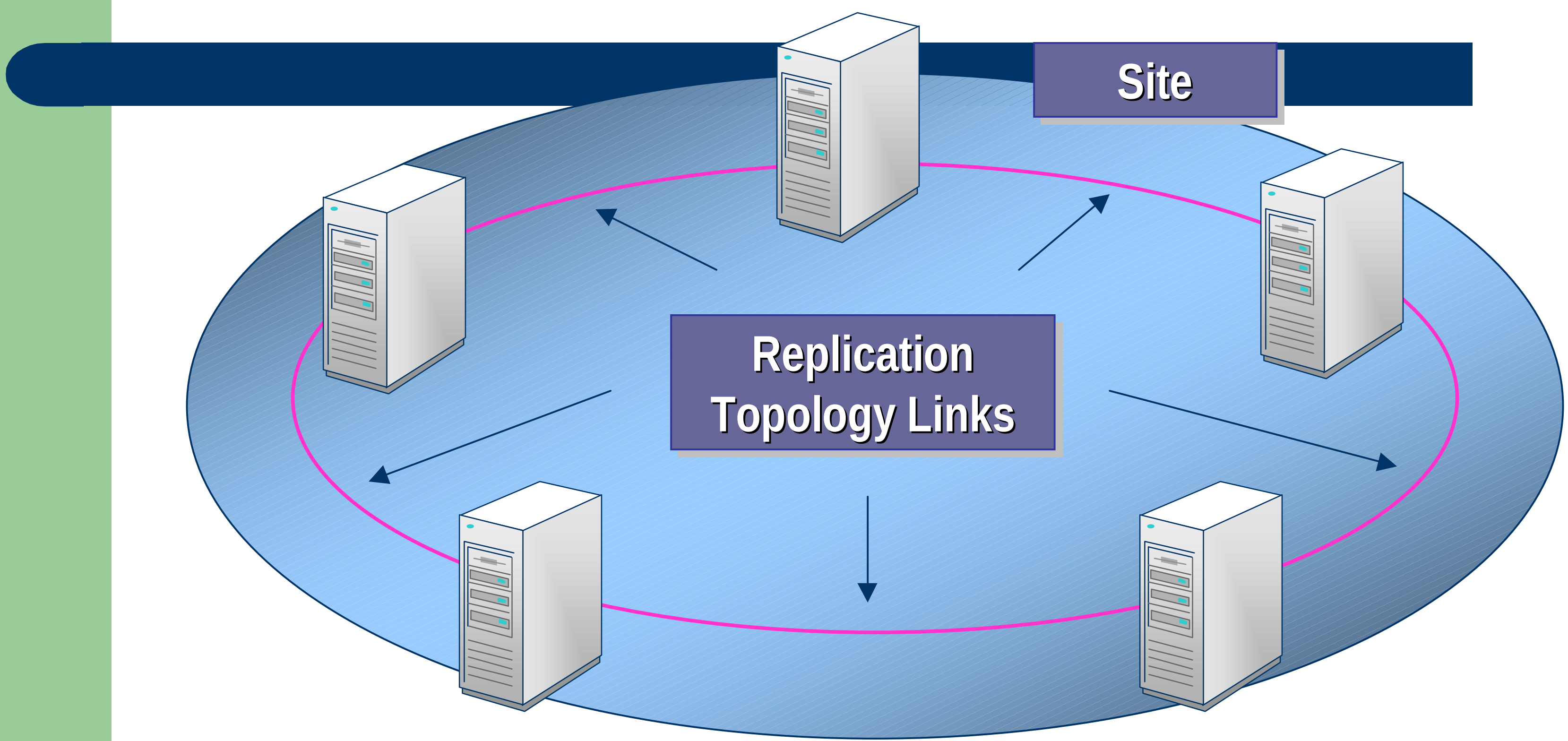
- Sites are areas of good connectivity, e.g. LANs, ATM nets, etc.
- *Not* part of the logical namespace structure
- DCs for a given domain can be distributed across many sites
- A single Site can hold many different DCs
- The *physical* organization provides fault-tolerance and performance for the *logical* organization

Active Directory Implementation

Physical Organization: Sites



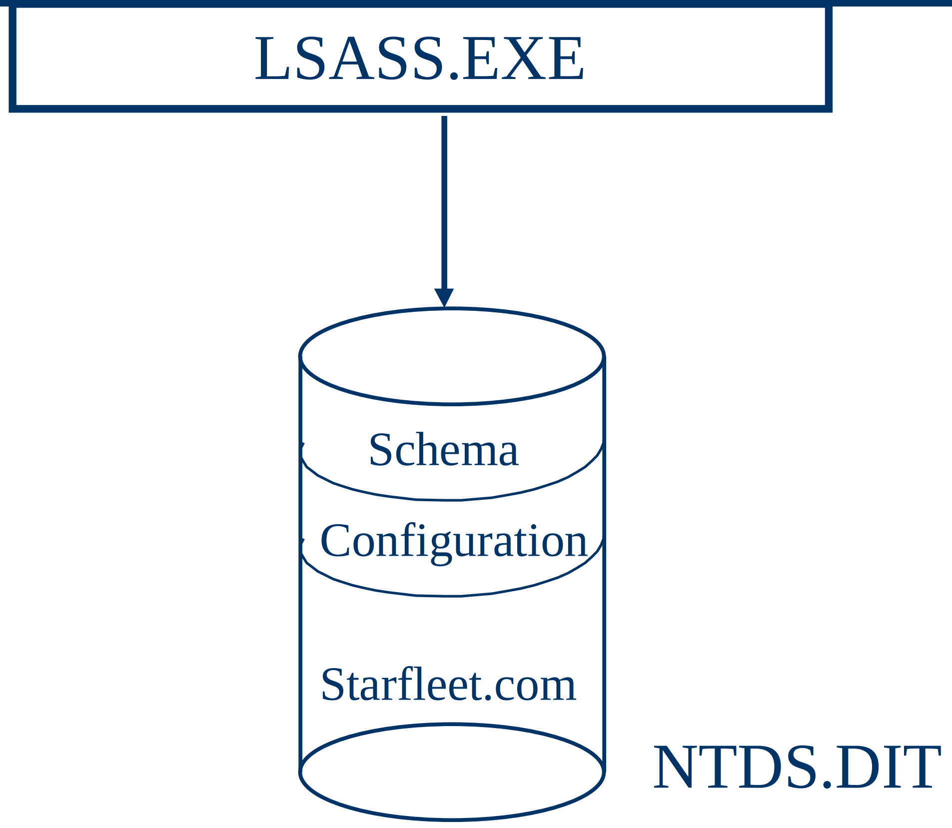
Intra-site Replication



Replication Terminology

- Naming Context
- Update sequence numbers
- Watermark vector
- State vector
- Sites and Domains
- Site links
- Site link bridges

Database content



- Domain Controller in Starfleet.com

Replication Fundamentals

- Multi-master Replication
- Replicated Operations
  - Object Creation
  - Object Manipulation
  - Object Move
  - Object Deletion
- Originating Update
  - Update was initiated by the DC or an application
- Replicated Update
  - Update was replicated from a replication partner
- Object deletions create tombstones

Replication Fundamentals

- Transitiveness of Replication
  - Store/Forward mechanism
  - Propagation dampening based on state vector
- Domain Controller
  - Server object (not machine account)
  - Server GUID: Used to find DC using DNS
  - Database GUID: Used to identify the DC's database in replication calls
    - Initially: Same as Server GUID
    - Changes if database is restored from backup

Replication Architecture

- USN based
- High-watermark vector
  - used to detect updates on replication partner
- Up-to-date Vector
  - used to filter updates that have not yet reached the domain controller
- Conflict reconciliation

Replication Architecture - USNs

- 64 Bit DWORD
- DC local meaning
- Assigned to new object update transaction
  - If transaction is aborted, then the USN is not assigned to any object
- Each object carries two USNs
  - usnCreated, usnChanged
- Each property carries two USNs
- Indexed property in the database

Object Creation

USN: 4710 → **USN: 4711**

Object: usnCreated **: 4711**				**Object:** usnChanged **: 4711**		
Property	**Value**	**USN**	**Version#**	**Timest.**	**Org. DB GUID**	**Org USN**
P1:	***Value***	**4711**	**1**	**TS**	***DS1 DB GUID***	**4711**
P2:	***Value***	**4711**	**1**	**TS**	***DS1 DB GUID***	**4711**
P3:	***Value***	**4711**	**1**	**TS**	***DS1 DB GUID***	**4711**
P4:	***Value***	**4711**	**1**	**TS**	***DS1 DB GUID***	**4711**

Object Replicated

USN: 4711

USN: 1745 → **USN: 1746**

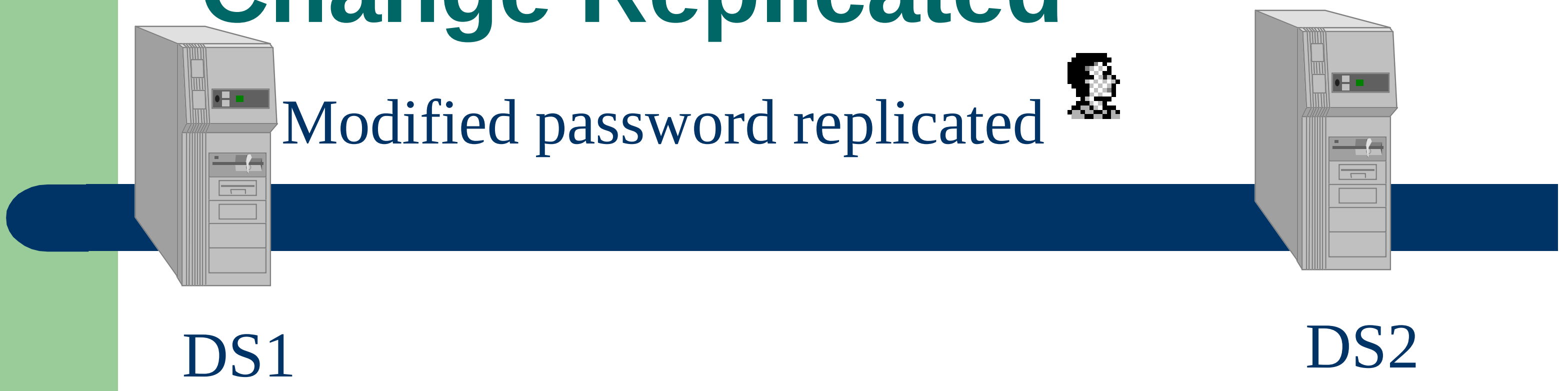
Object: usnCreated **: 1746**				**Object:** usnChanged **: 1746**		
Property	**Value**	**USN**	**Version#**	**Timest.**	***Org. DB GUID***	**Org USN**
P1:	***Value***	**1746**	**1**	**TS**	***DS1 DB GUID***	**4711**
P2:	***Value***	**1746**	**1**	**TS**	***DS1 DB GUID***	**4711**
P3:	***Value***	**1746**	**1**	**TS**	***DS1 DB GUID***	**4711**
P4:	***Value***	**1746**	**1**	**TS**	***DS1 DB GUID***	**4711**

Object Modification

USN: 2001 → **USN: 2002**

Object: usnCreated **: 1746**				**Object:** usnChanged **: 2002**		
Property	**Value**	**USN**	**Version#**	**Timest..**	***Org. DB GUID***	**Org USN**
P1:	***Value***	**1746**	**1**	**TS**	***DS1 DB GUID***	**4711**
P2:	***Value***	**2002**	**2**	**TS**	***DS2 DB GUID***	**2002**
P3:	***Value***	**1746**	**1**	**TS**	***DS1 DB GUID***	**4711**
P4:	***Value***	**1746**	**1**	**TS**	***DS1 DB GUID***	**4711**

Change Replicated



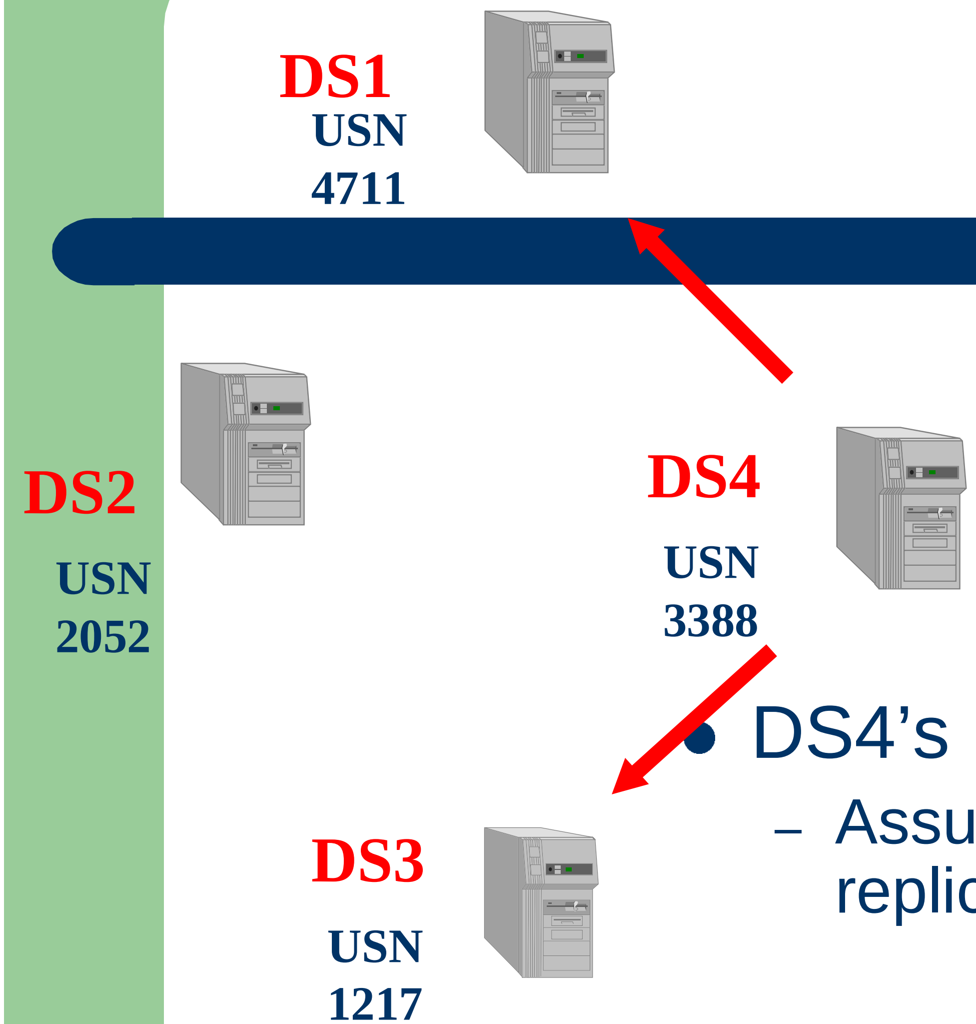
USN: 5039 → **USN: 5040** **USN: 2002**

Object: usnCreated **: 1746**				**Object:** usnChanged **: 5040**		
Property	**Value**	**USN**	**Version#**	**Timest.**	***Org. DB GUID***	**Org USN**
P1:	***Value***	**4711**	**1**	**TS**	***DS1 DB GUID***	**4711**
P2:	***Value***	**5040**	**2**	**TS**	**DS2 DB GUID**	**2002**
P3:	***Value***	**4711**	**1**	**TS**	***DS1 DB GUID***	**4711**
P4:	***Value***	**4711**	**1**	**TS**	***DS1 DB GUID***	**4711**

High-watermark vector

- [redacted]
  - Replication partners
  - Highest known USN
- Used to detect recent changes on replication partners

High-watermark vector DS4



DSA GUID	Highest known USN
DS1 GUID	4711
DS3 GUID	1217

- DS4’s high-watermark vector
  - Assuming, that DS1 and DS3 are it’s replication partners

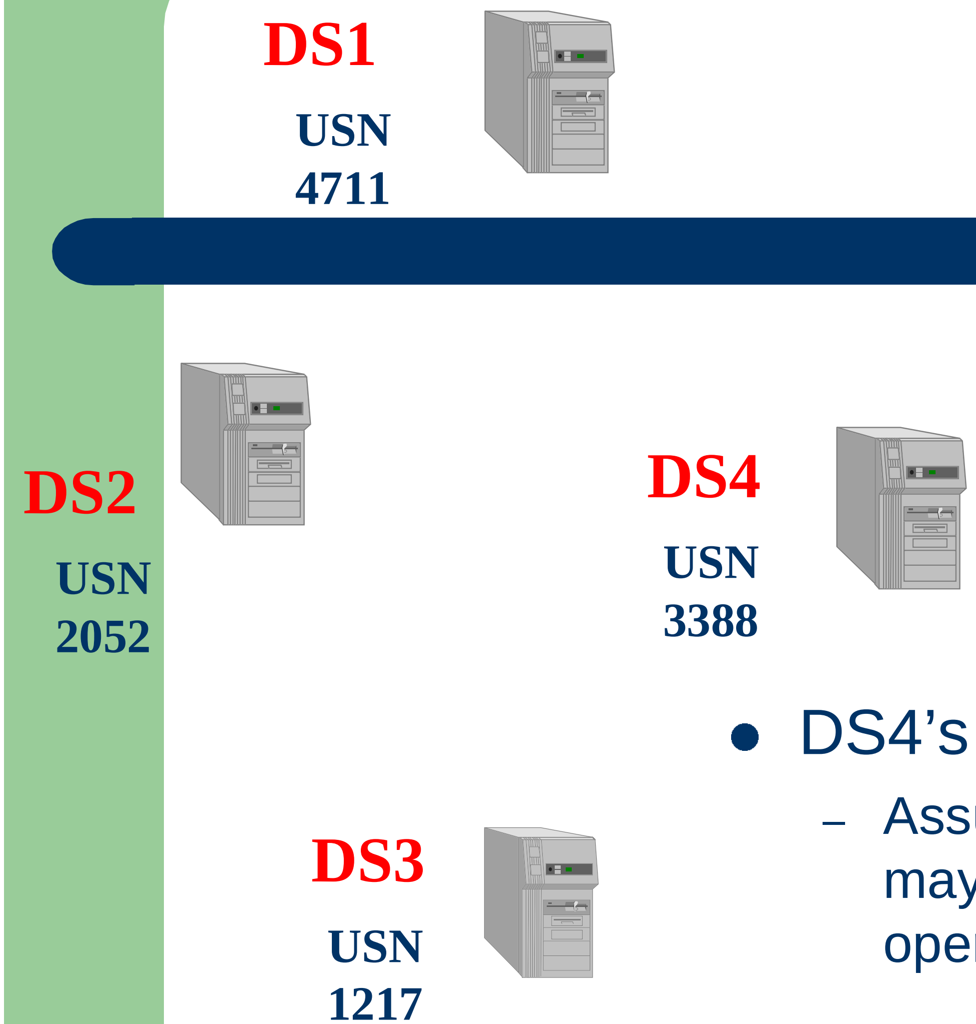
Information sent in preparation of replication

- Naming context for which changes are requested
- Max. # of object update entries requested
- Max. # of values requested
- High-USN-changed value of naming context of replication partner
- Complete up-to-dateness vector
  - Used for propagation dampening

Up-to-dateness vector

- Up-to-dateness related to a specific Naming Context
- List of pairs:
  - Originating-DC-GUID (Database GUID)
  - Highest-Originating-USN
- Only those DCs are added from which originating updates have been received (even through replication)
- Stored as replUpToDateVector, which is a property on the naming context object

Up-to-dateness vector

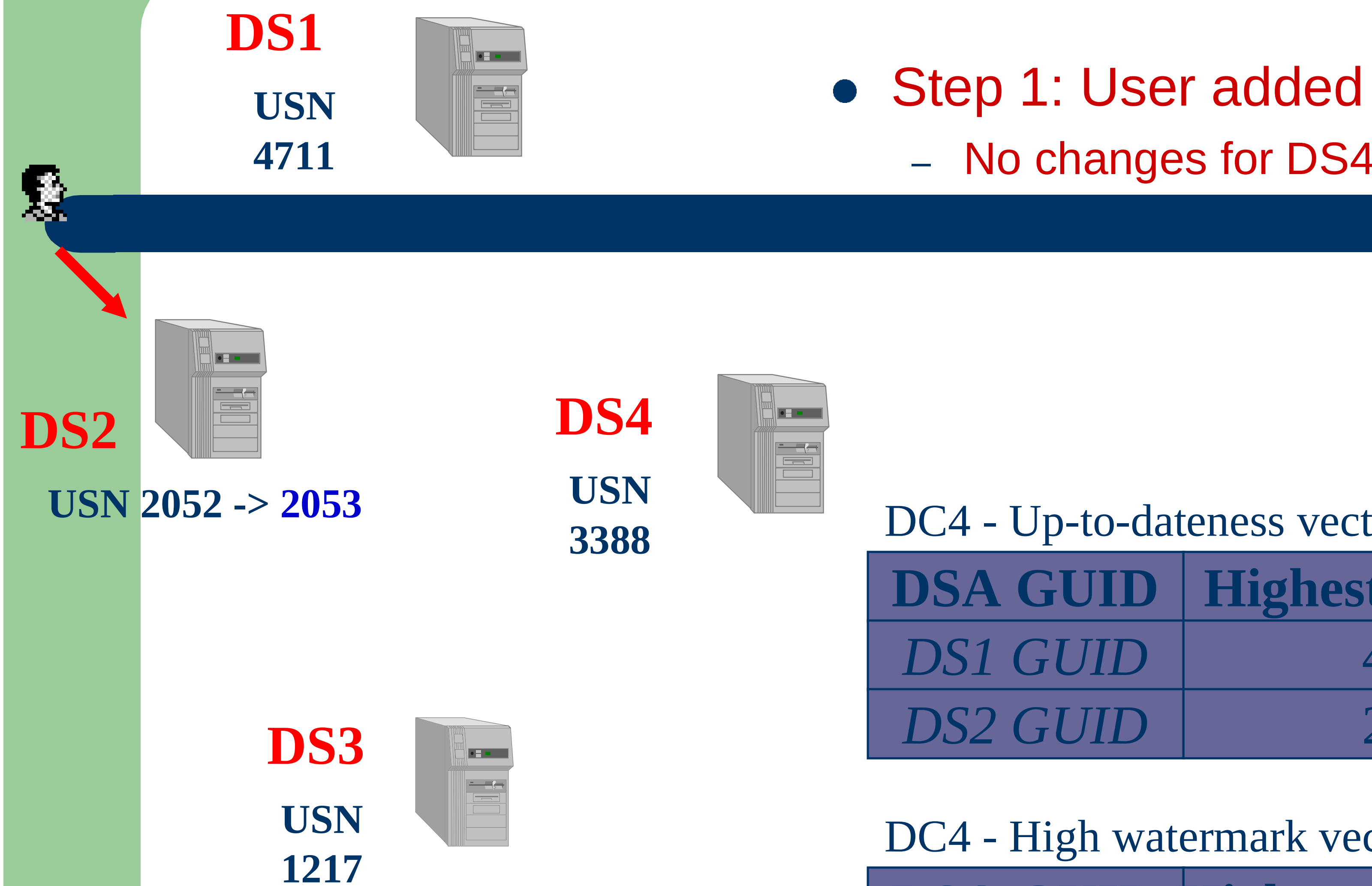


DSA GUID	Highest Org. USN
DS1 GUID	4711
DS2 GUID	2050

- DS4's up-to-dateness vector
  - Assuming, that only DS1 and DS2 (and maybe DS4) performed originating write operations

Replication - DC4

- Step 1: User added to DS2
  - No changes for DS4



DC4 - Up-to-dateness vector

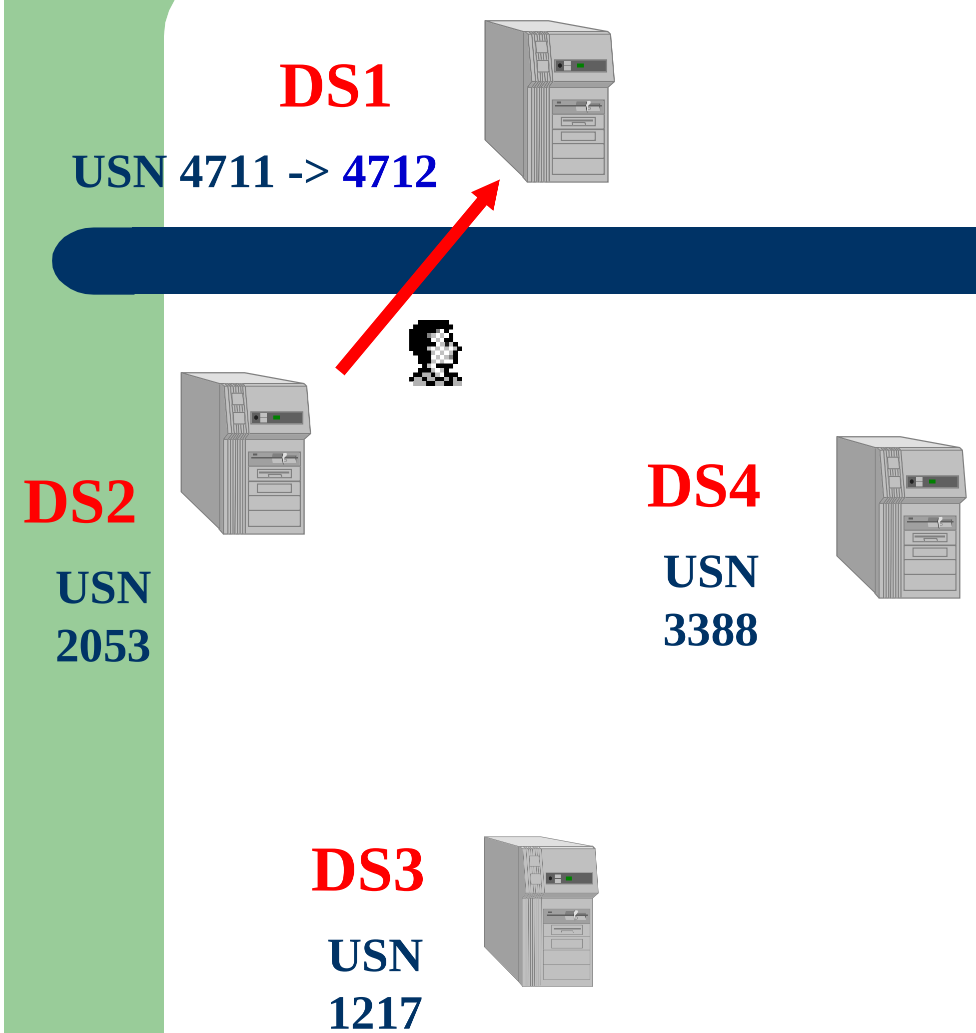
DSA GUID	Highest Org. USN
DS1 GUID	4711
DS2 GUID	2050

DC4 - High watermark vector

DSA GUID	Highest known USN
DS1 GUID	4711
DS3 GUID	1217

Replication - DC4

- Step 2: User replicated to DS1
  - No changes for DS4
  - Note: Write was originated on DS2!



DC4 - Up-to-dateness vector

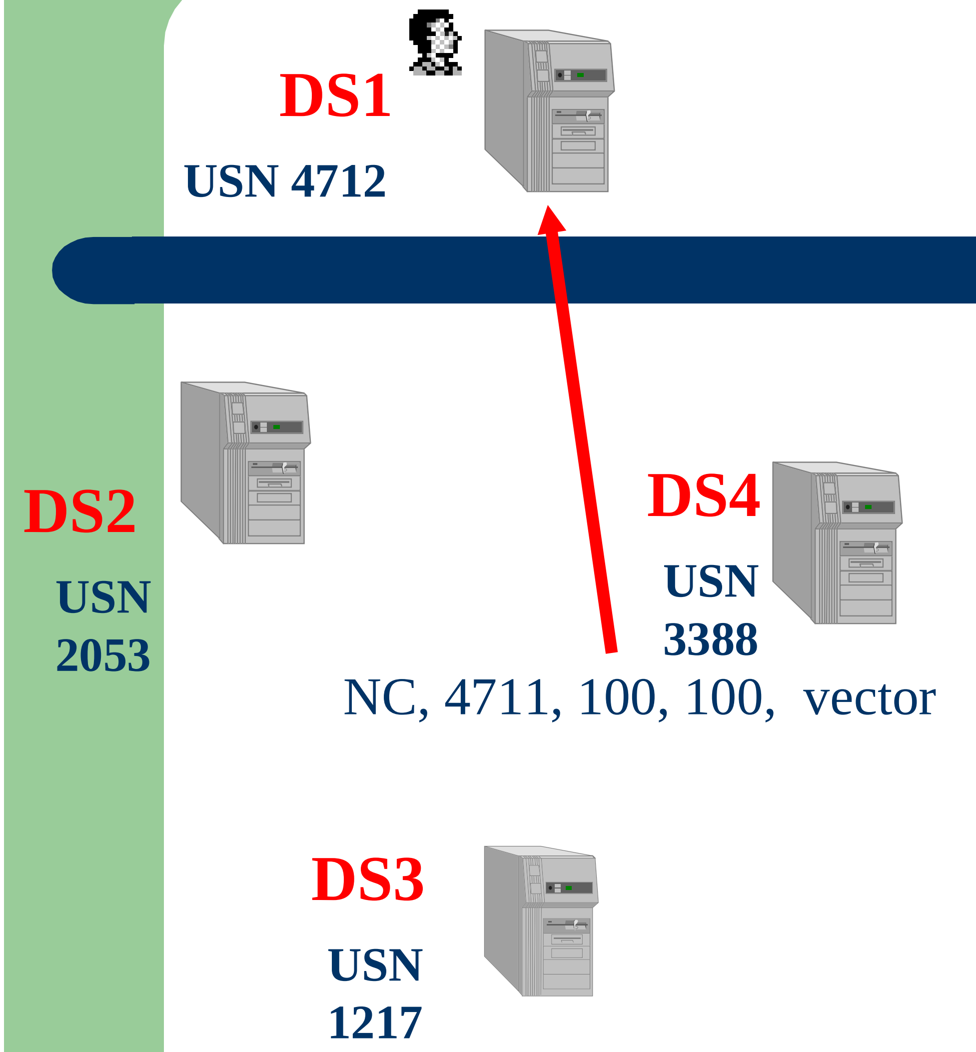
DSA GUID	Highest Org. USN
DS1 GUID	4711
DS2 GUID	2050

DC4 - High watermark vector

DSA GUID	Highest known USN
DS1 GUID	4711
DS3 GUID	1217

Replication - DC4

- Step 3: DS4 initiates replication with DS1
  - Sends NC, highest known USN DS1 for this NC, # objects, # valu up-to-dateness vector



DC4 - Up-to-dateness vector

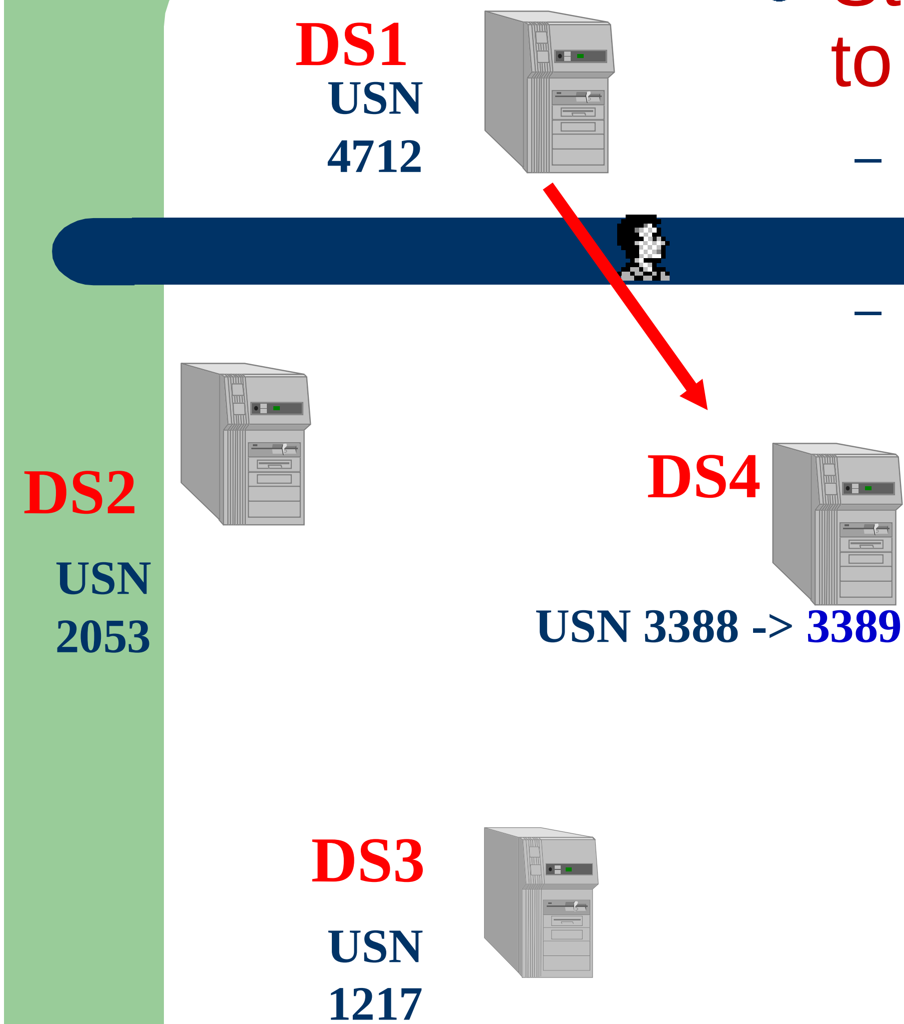
DSA GUID	Highest Org. USN
DS1 GUID	4711
DS2 GUID	2050

DC4 - High watermark vector

DSA GUID	Highest known USN
DS1 GUID	4711
DS3 GUID	1217

Replication - DC4

- Step 4: DS1 replicates new user to DS4
  - Sends data, last-object-changed USN, it's up-to-dateness vector
  - DS4 uses DS1's up-to-dateness vector to determine it's up-to-dateness



DC4 - Up-to-dateness vector

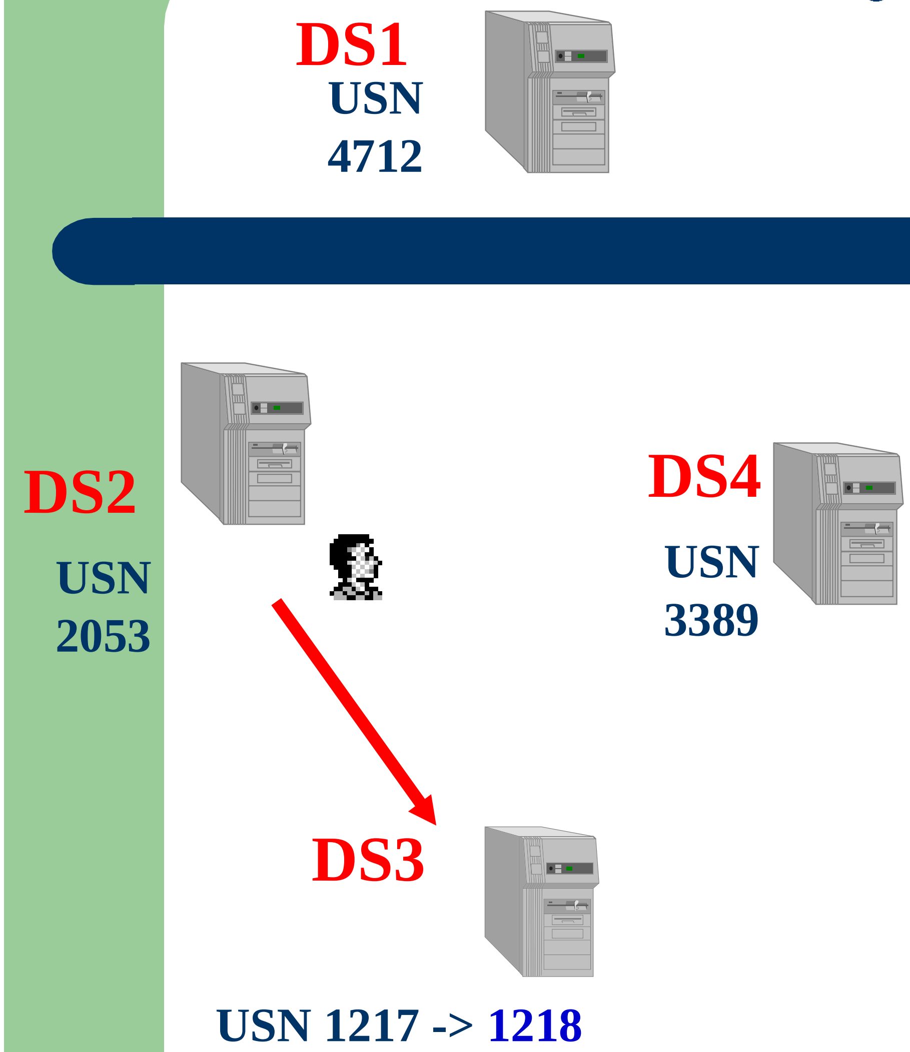
DSA GUID	Highest Org. USN
DS1 GUID	4711
DS2 GUID	**2053**

DC4 - High watermark vector

DSA GUID	Highest known USN
DS1 GUID	**4712**
DS3 GUID	1217

Replication - DC4

- Step 5: DS2 replicates new user to DS3
  - No changes for DS4



DC4 - Up-to-dateness vector

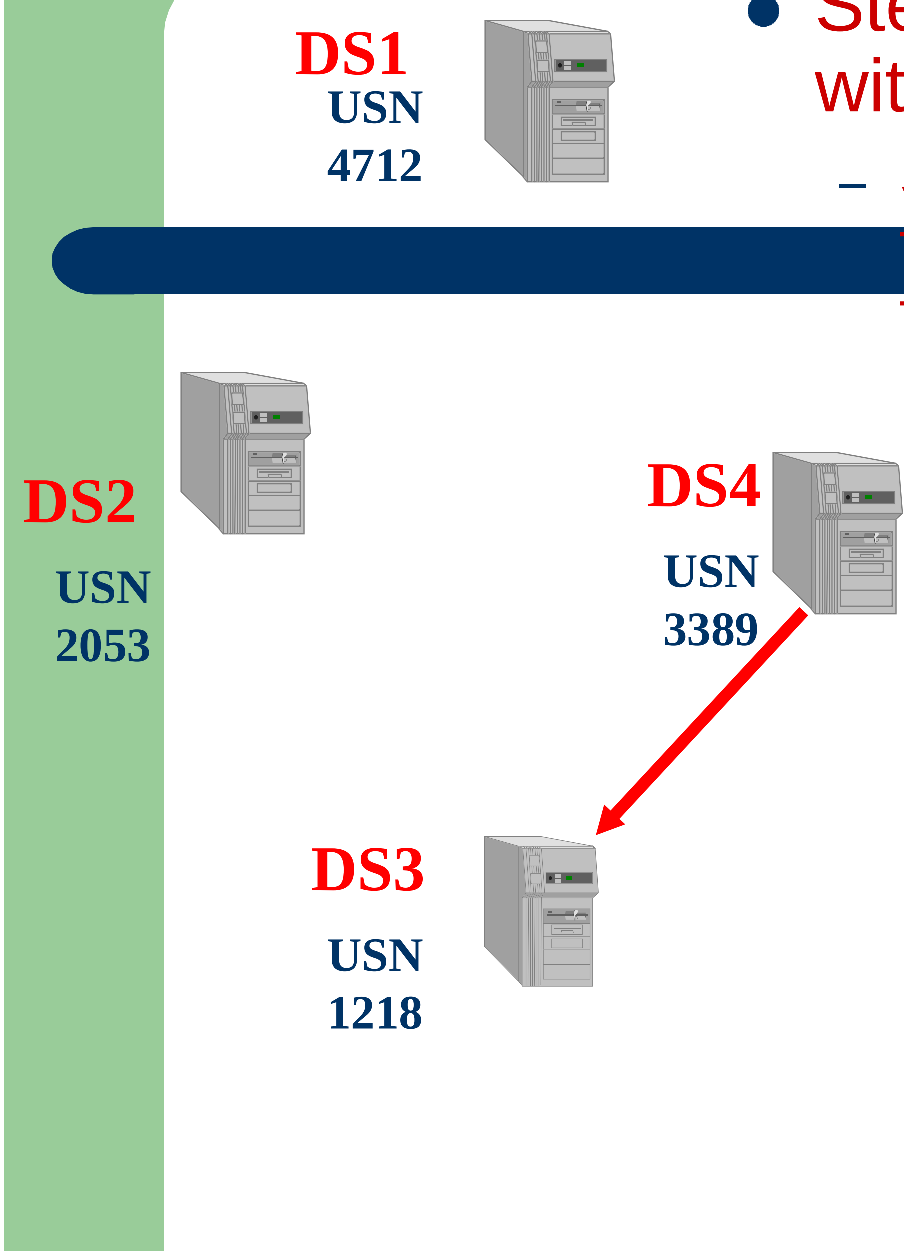
DSA GUID	Highest Org. USN
DS1 GUID	4711
DS2 GUID	**2053**

DC4 - High watermark vector

DSA GUID	Highest known USN
DS1 GUID	**4712**
DS3 GUID	1217

Replication - DC4

- Step 6: DS4 initiates replication with DS1
  - Sends NC, highest known USN DS3 for this NC, # objects, # values, up-to-dateness vector



DC4 - Up-to-dateness vector

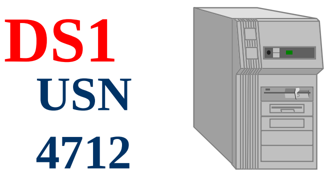
DSA GUID	Highest Org. USN
DS1 GUID	4711
DS2 GUID	2053

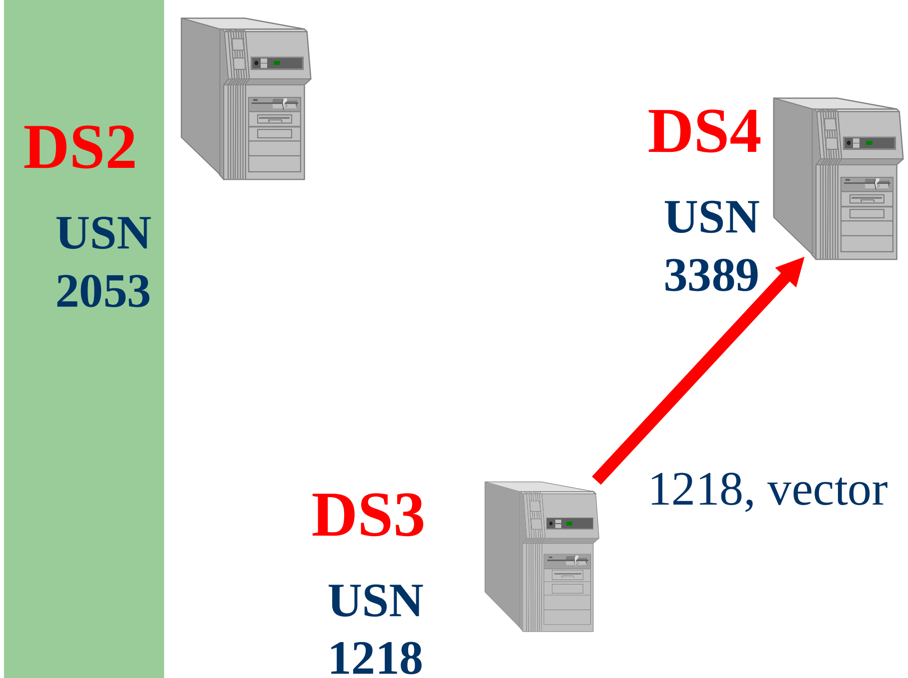
DC4 - High watermark vector

DSA GUID	Highest known USN
DS1 GUID	4712
DS3 GUID	1217

Replication - DC4

- Step 7: DS3 replication reply
  - Determines, that DS4 already is up-to-date
  - Sends last-object-changed USN, up-to-dateness vector, but no data!





DC4 - Up-to-dateness vector

DSA GUID	Highest Org. USN
DS1 GUID	4711
DS2 GUID	2053

DC4 - High watermark vector

DSA GUID	Highest known USN
DS1 GUID	4712
DS3 GUID	**1218**

Conflict Reconciliation - 1

- Attribute Value Conflict
  - I.e., user changes his password on DC1, admin changes user's password on DC2
  - Reconciliation: higher version number -> higher timestamp -> higher GUID of originating write DSA
- Move under deleted parent
  - I.e., Admin creates user in OU1 on DC1, second Admin deletes OU1 on DC2
  - Reconciliation: OU1 is deleted, user moved to "Lost and Found" container

Conflict Reconciliation - 2

- Object creation name conflict
  - I.e., two administrators create two user objects with identical RDNs on two DCs at the same time
  - Reconciliation: One object (identified by it's GUID) receives a system wide unique value on the conflicting attribute (here the RDN)
    - In Beta 2: Both RDNs get new value
  - Reconciliation: higher version number -> higher timestamp -> higher GUID of originating write DSA

Urgent Replication

- Initiated by SAM or LSA (not by LDAP writes) for:
  - Newly locked-out account
  - RID pool changes
  - DC Machine Accounts (Post Beta 2 ?)
- These trigger an immediate replication cycle within the site
- Uses notification

Replication Transports

- Intra-Site
  - DS-RPC
- Inter-Site
  - DS-RPC
  - ISM-SMTP

Replication Topology

- Topology Generator on each DC (KCC)
  - Local Operation
  - Computes topology and creates/deletes connection objects
  - Connection Object defines incoming replication from partner
  - Single connection object per replication partner
- One topology per NC
  - Configuration NC and Schema NC share same topology
  - Each Domain NC has it’s own topology
  - GCs embrace Domain NCs
- Topology is built on top of sites
  - For each NC, a bi-directional ring is automatically built within sites
  - For each NC, a spanning tree topology is automatically built between sites
  - Can be over-written

Replication Model

- Intra Site
  - RPC Replication in a Site - Uncompressed
  - KCC Generates a bi-directional Ring with extra edges
    - Algorithm ensures no more than three hops between any two DCs
- Inter Site
  - Minimum cost Spanning Tree
  - DS-RPC - Can be compressed
  - SMTP

Intra-Site Replication

- DC GUID is used to construct the ring
- New installed DCs add themselves to the ring, and replicate the new configuration information
- Existing DCs add/remove connection objects
- For simplicity:
  - This is a GUID:
    - 1509e139-1dcd-11d2-9e98-98493b0b9910
  - This is what we use: 1

DNS

Role of DNS

- Active Directory Domains named with DNS names
  - europe.microsoft.com
- Machines named with DNS names
  - printserver1.hq.microsoft.com
- REPLACES NETBIOS NAMING! (Negates need for Netbios Dependent Services e.g., WINS, Browser, etc.)

Microsoft DNS

w Features in NT5

- Dynamic DNS
- Active Directory replication integration
- Unicode character support
- Enhanced DNS Manager
- Caching resolver service

SRV RR

ms.com IN SRV 2 10 389 dc01.ms.com
ms.com IN SRV 1 30 389 dc02.ms.com
ms.com IN SRV 1 50 389 dc03.ms.com
ms.com IN SRV 1 20 389 dc04.ms.com

- dc02, dc03, dc04 will be tried first before dc01
- dc02, dc03, dc04 will be picked 30%, 50% and 20% respectively
- If all failed, dc01 will be picked 100%

The Domain Locator

- Components of <domain>
  - Active Directory domain name:

    hq.microsoft.com
  - Optional site identifier:

    redmond.sites.hq.microsoft.com
  - Optional role:

    pdc.ms-dcs, gc.ms-dcs,

    for example: gc.ms-dcs.hq.microsoft.com

Microsoft DNS

Dynamic DNS Protocol

- IETF proposed standard (RFC 2136)
- Capabilities
  - Add and delete records
  - Updates are atomic
- Updates sent to authoritative server
  - Client must locate authority
  - Server with secondary zone forwards to server with primary zone

Microsoft DNS
Dynamic DNS Client

- Hostname

  <Computer Name>.<DNS Domain Name>
- DNS Domain Name is per-adapter
  – Configured via DHCP, or by hand
- Name collision detection:
  – Assert hostname is unique using prerequisites

Microsoft DNS

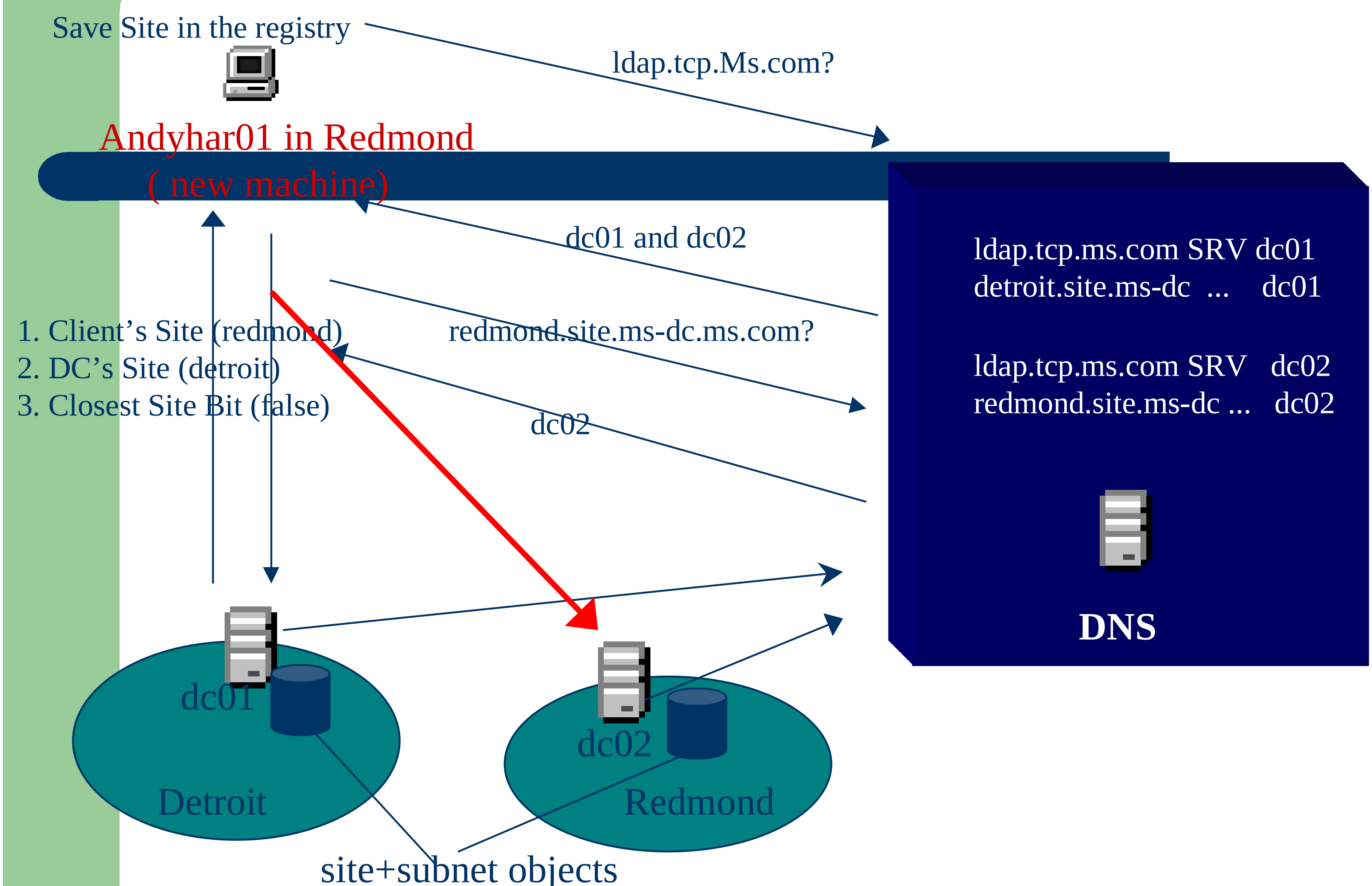
Dynamic DNS Client

- DHCP client
  - Track names and addresses
  - Send A RR updates to DNS
  - Ask DHCP server to update PTR RRs
- DHCP server
  - Register PTR RRs for clients
  - Remove PTR RRs when leases expire
    - Optional - also remove A RR
    - Garbage collection

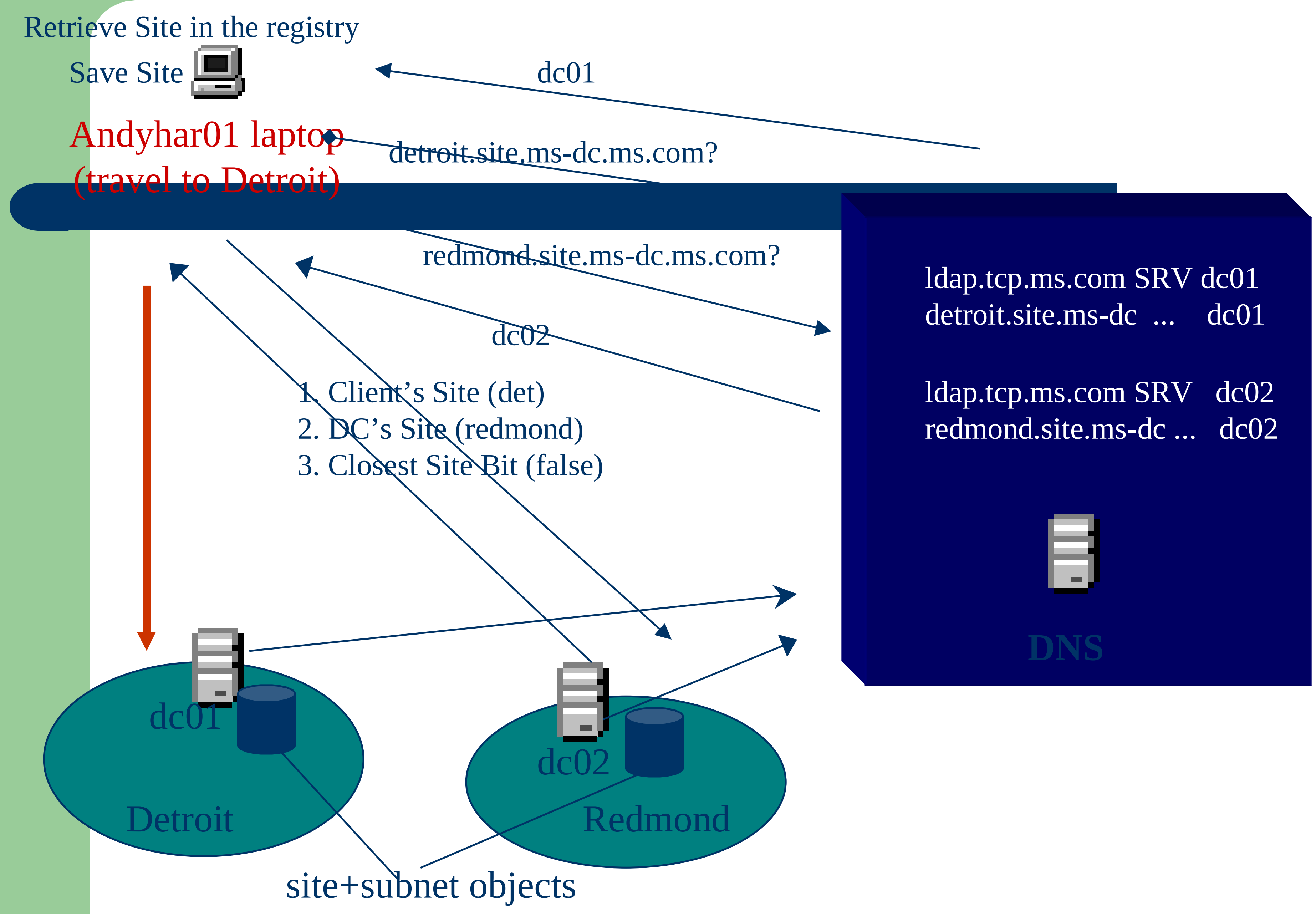
Microsoft DNS
Dynamic DNS Client

- Downlevel hosts
  - DHCP server can be configured to register A, PTR RRs for downlevel clients
- DHCP-DNS interaction is Internet-Draft
  - draft-ietf-dhc-dhcp-dns-04.txt
- Moving toward standards track

DNS and Active Directory



DNS Locator


Retrieve Site in the registry
Save Site
Andyhar01 laptop
(travel to Detroit)
dc01
detroit.site.ms-dc.ms.com?
redmond.site.ms-dc.ms.com?
dc02
1. Client's Site (det)
2. DC's Site (redmond)
3. Closest Site Bit (false)
ldap.tcp.ms.com SRV dc01
detroit.site.ms-dc ... dc01
ldap.tcp.ms.com SRV dc02
redmond.site.ms-dc ... dc02
DNS
dc01
Detroit
dc02
Redmond
site+subnet objects

The Machine Locator

- Machines named with DNS names
- Machines register A RRs in DNS
- Some capability already available in NT 4.0
  - net view \\printserver1.hq.microsoft.com
  - DNS names work in admin tools

Microsoft DNS

ADS Replication Integration

- DNS zone-xfer is single master
- Alternative - store zone in ADS
  - Name is object, RRset is attribute
  - Zone file:

    microsoft.com IN A 207.68.156.54

    microsoft.com IN MX 10 mail1.microsoft.com.
  - ADS object:

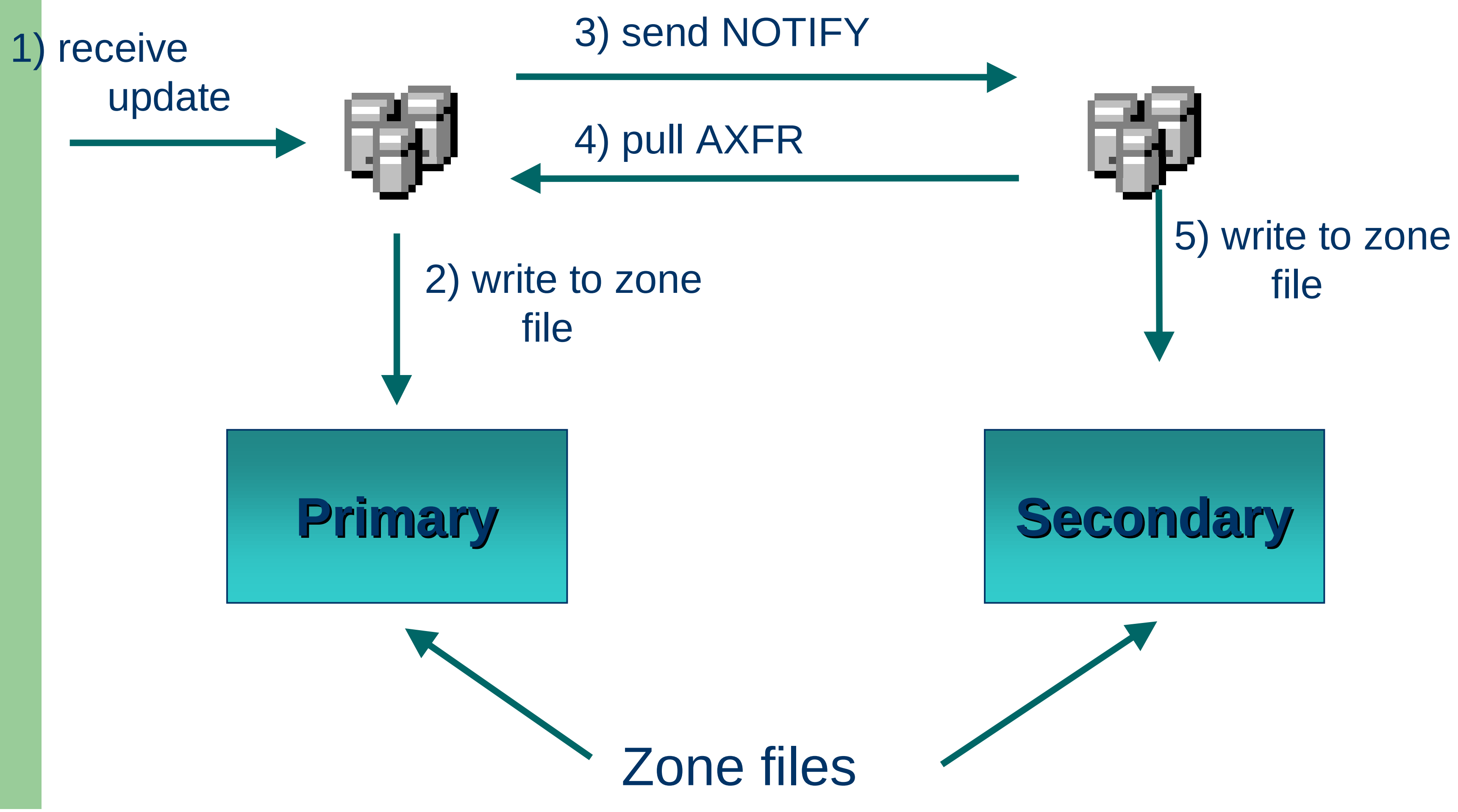
    dnsNode = microsoft.com

        A = 207.68.156.54

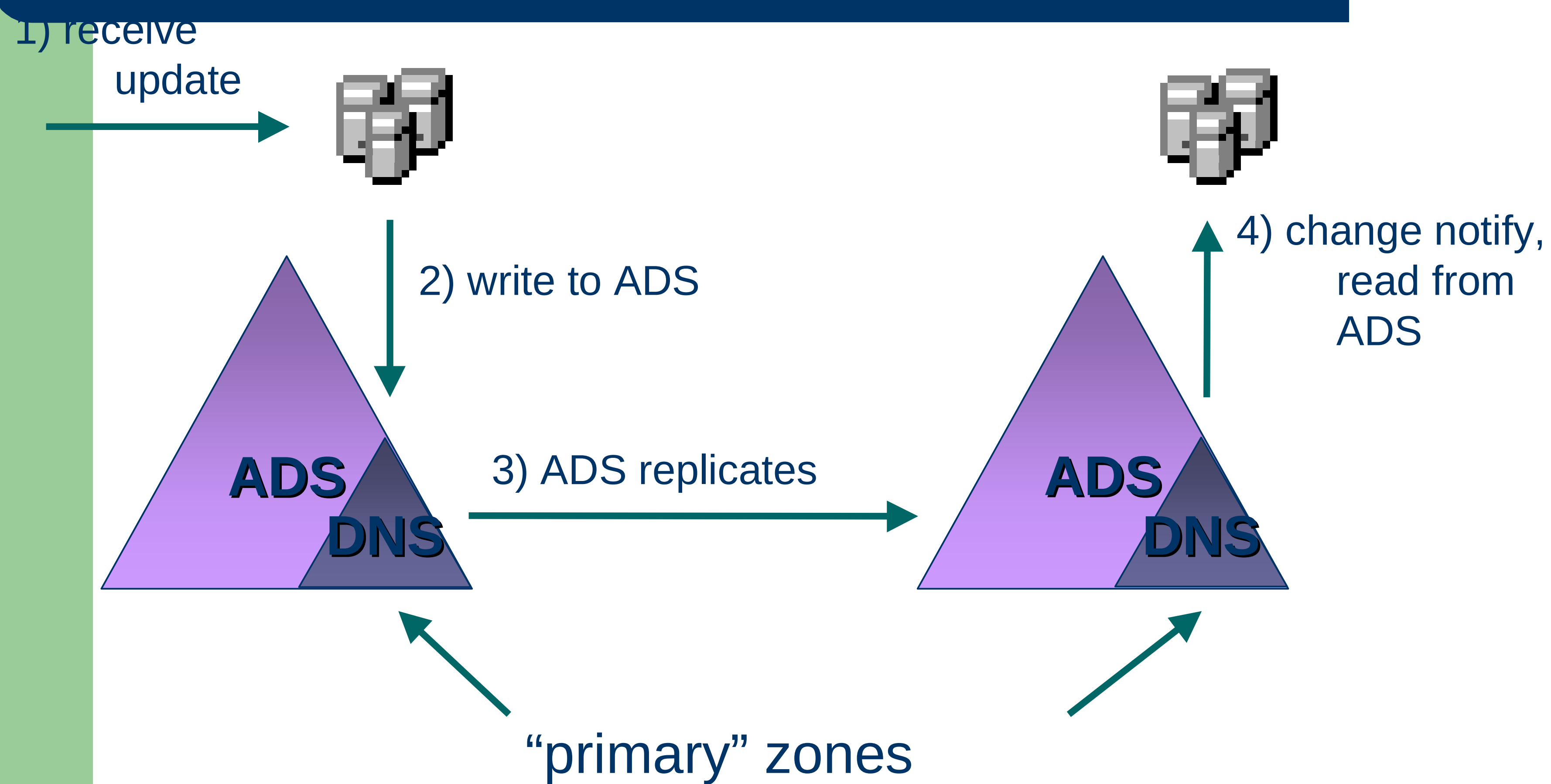
        MX = 10 mail1.microsoft.com.

Microsoft DNS

ADS Replication Integration (cont'd)



Microsoft DNS
ADS Replication Integration (cont'd)



Microsoft DNS

DNS Manager

- Microsoft Management Console Snap-in
- Wizards for
  - Setting up new servers
  - Managing zones, creating common records
- Flexible access control
  - Assign access using ADS groups
  - Per-server and per-zone control
- Monitoring

Windows NT 5.0 Kerberos Implementation

- Single sign on to Windows NT domains and Kerberos-based services
- Integrated Windows NT authorization
- PK extensions for smart card logon
- Active Directory support for account management
- Active Directory trust hierarchy
- Application support through SSPI

Kerberos Protocol Benefits

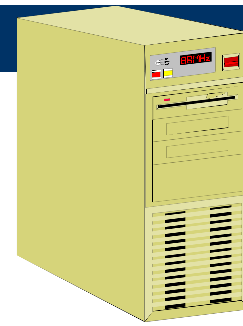
- Faster connection authentication
  - Server scalability for high-volume connections
  - Reuse session tickets from cache
- Mutual authentication of both client, server
- Delegation of authentication
  - Impersonation in three-tier client/server architectures
- Transitive trust between domains
  - Simplify inter-domain trust management
- Mature IETF standard for interoperability
  - Testing with MIT Kerberos V5 Release

Kerberos Integration

Client

Kerberos SSPI provider manages credentials and security context;
LSA manages ticket cache

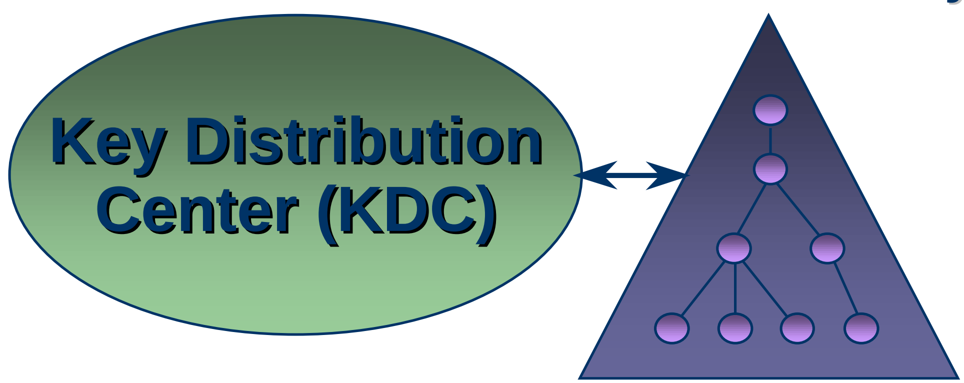
Server



Session ticket authorization data supports NT access control model

KDC relies on the Active Directory as the store for security principals and policy

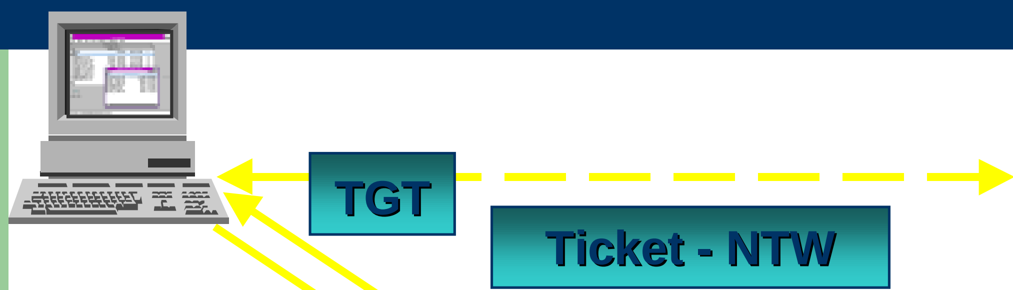
Windows NT Active Directory



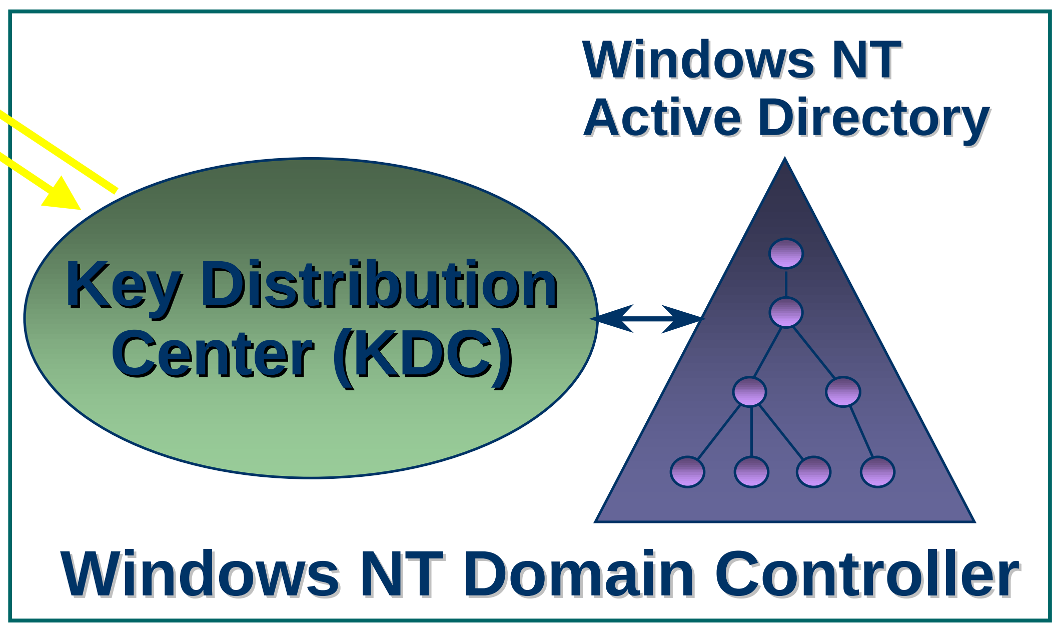
Windows NT Domain Controller

Kerberos Authentication

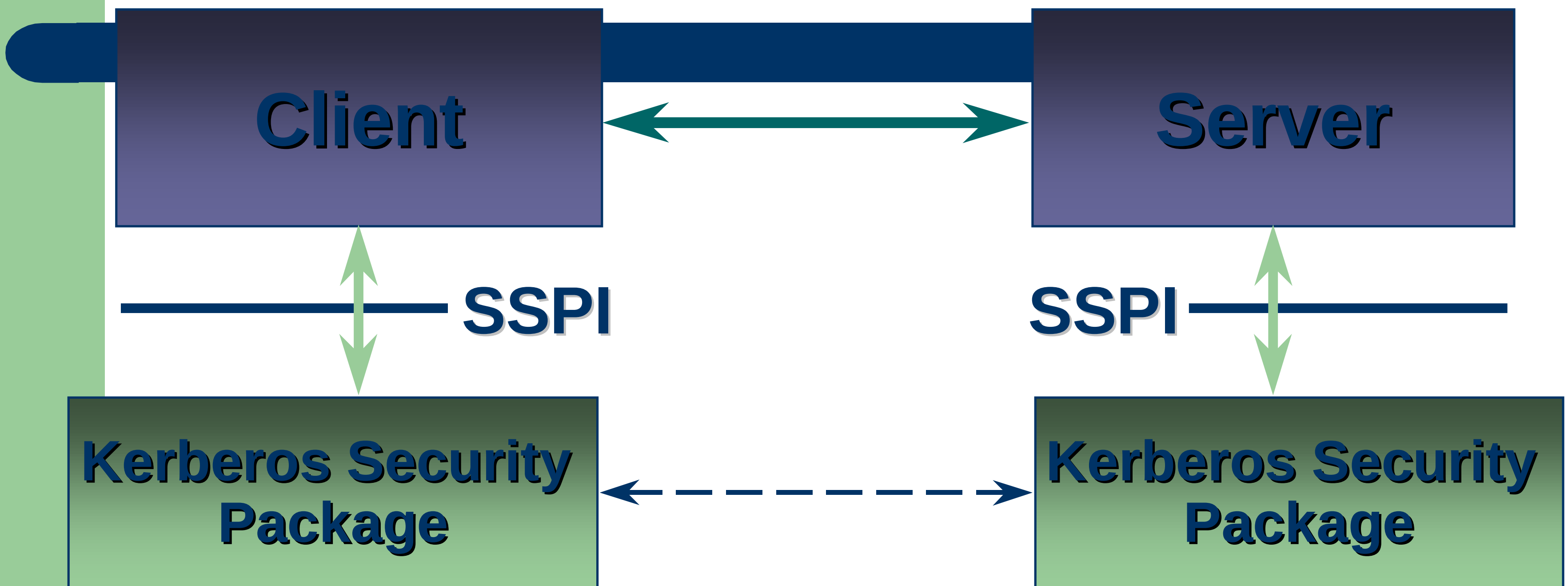
Interactive Domain Logon



1. Locate KDC for domain by DNS lookup for AD service
2. Use hash(pwd) to sign pre-auth data in AS request
3. Group membership expanded by KDC, added to TGT auth data
4. Send TGS request for session ticket to workstation



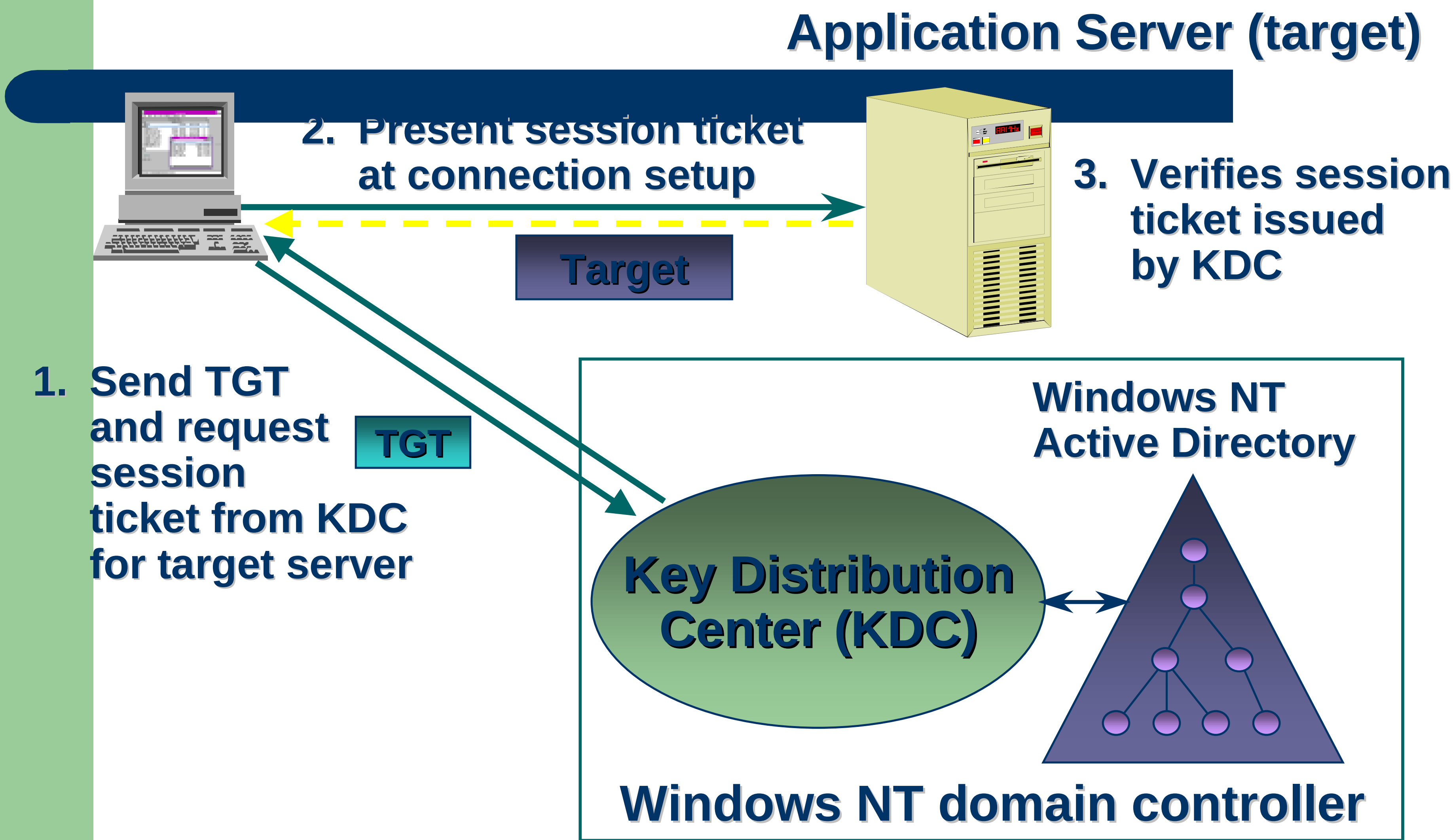
Security Support Provider Interface



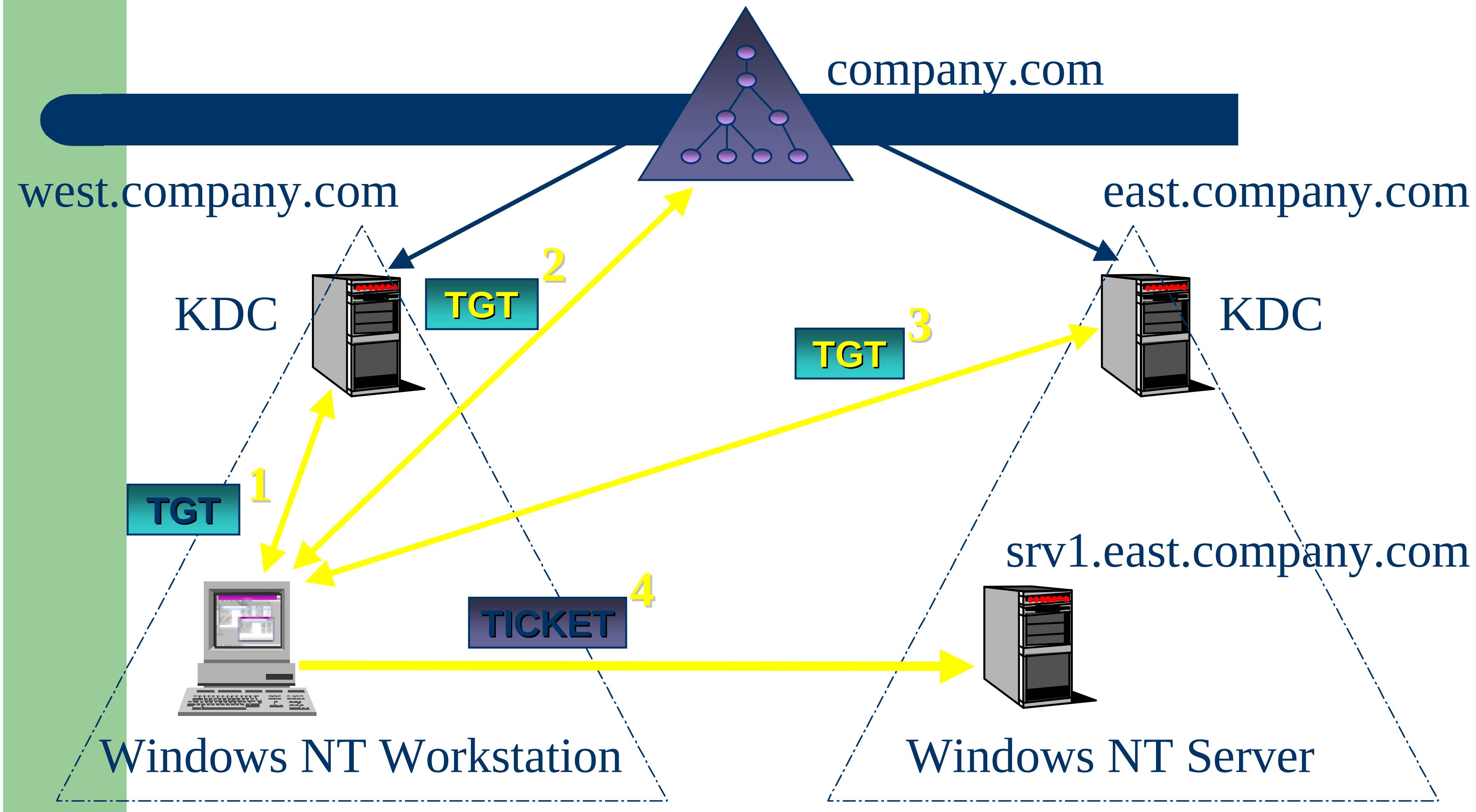
- Application protocol carries all data
- Kerberos SSP manages security context

Kerberos Authentication

Network Server connection



Cross-domain Authentication

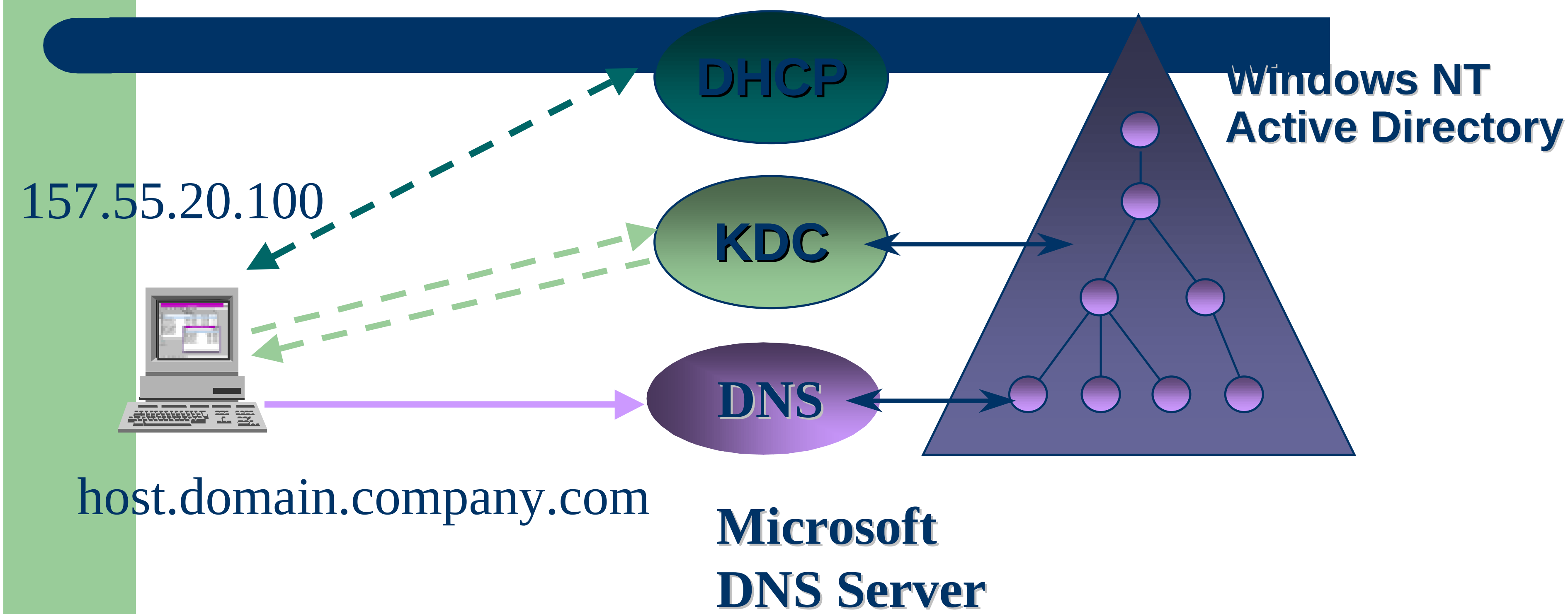


Windows NT 5.0 Integration

Kerberos Authentication Use

- LDAP to Active Directory
- CIFS/SMB remote file access
- Secure dynamic DNS update
- Distributed file system management
- Host-host IP security using ISAKMP
- Secure Intranet web services in IIS
- Authenticate certificate request to Enterprise CA
- DCOM/RPC security provider

Secure Dynamic DNS Update



Authentication and Authorization

- Authenticate using domain credentials
  - User account defined in Active Directory
- Authorization based on group membership
  - Centralize management of access rights
- Distributed security tied to the Windows NT Security Model
  - Network services use impersonation
  - Object-based access control lists

Authorization Data

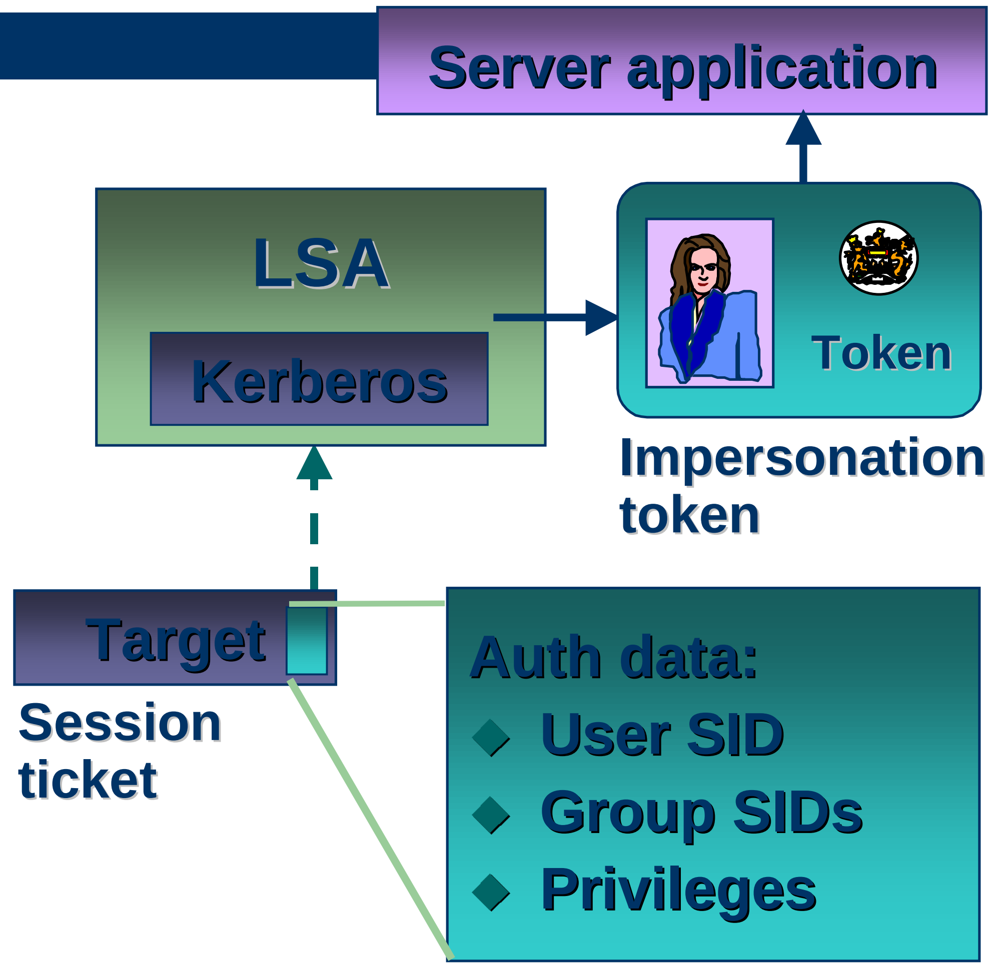
- What is the client allowed to do?
  - Based on Windows NT group membership
  - Identified by Security Ids (SIDs) in NT security architecture
- NT KDC supplies auth data in tickets
  - At interactive logon (AS exchange):
    - User SID, global, universal group SIDs
  - At session ticket request (TGS exchange)
    - Domain local group SIDs

Authorization Data

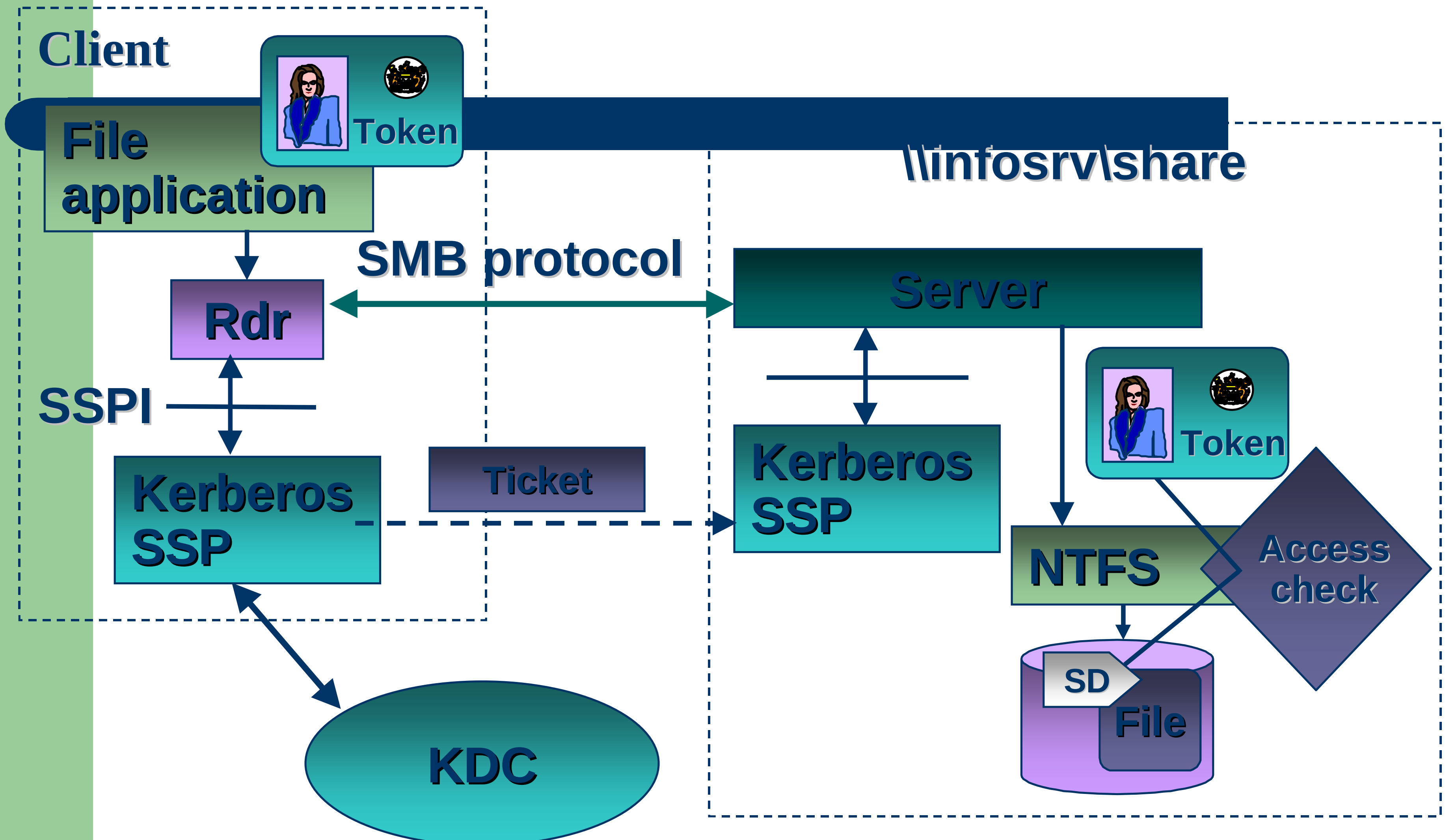
- Kerberos protocol supports auth data in tickets
  - Examples: DCE and Sesame architectures
- Revision to RFC 1510
  - Clarifications on client, KDC supplied data
  - Submitted by Ted Ts'o, Clifford Neuman
- Interoperability issues are minimum
  - NT auth data ignored by UNIX implementations

Building An Access Token From A Kerberos Ticket

- Kerberos package gets auth data from session ticket
- LSA builds access token for security context
- Server thread impersonates client context



Remote File Access Check



Interoperability Goals

- Cross-platform protocol interoperability
  - Authentication
  - Message integrity (sign/verify)
  - Confidentiality (seal/unseal)
- Single user account store
  - Scalability and ease of administration
- Use existing authorization mechanisms
  - Name-based authorization
  - Integrated Windows NT authorization

Summary

- Standards-based secure protocol implementation
- Integration with Active Directory for scalability, ease of management
- Integration with Windows NT distributed system services