Introducción (I)

- Lightweight Directory Access Protocol (LDAP):
  - Esta basado en X.500.
  - Permite acceder a información guardada de forma centralizada en una red.

Introducción (II)

- La estructura de almacenamiento de LDAP es el servicio de directorio:
  - El termino directorio no se refiere a un directorio como la estructura de un sistema de ficheros.
  - Servicio de directorio es una base de datos especial, desarrollada para:
    - Frecuentes consultas.
    - Muy pocas actualizaciones.
  - No posee control de transacciones ni posibilidad de vuelta atrás (rollback).

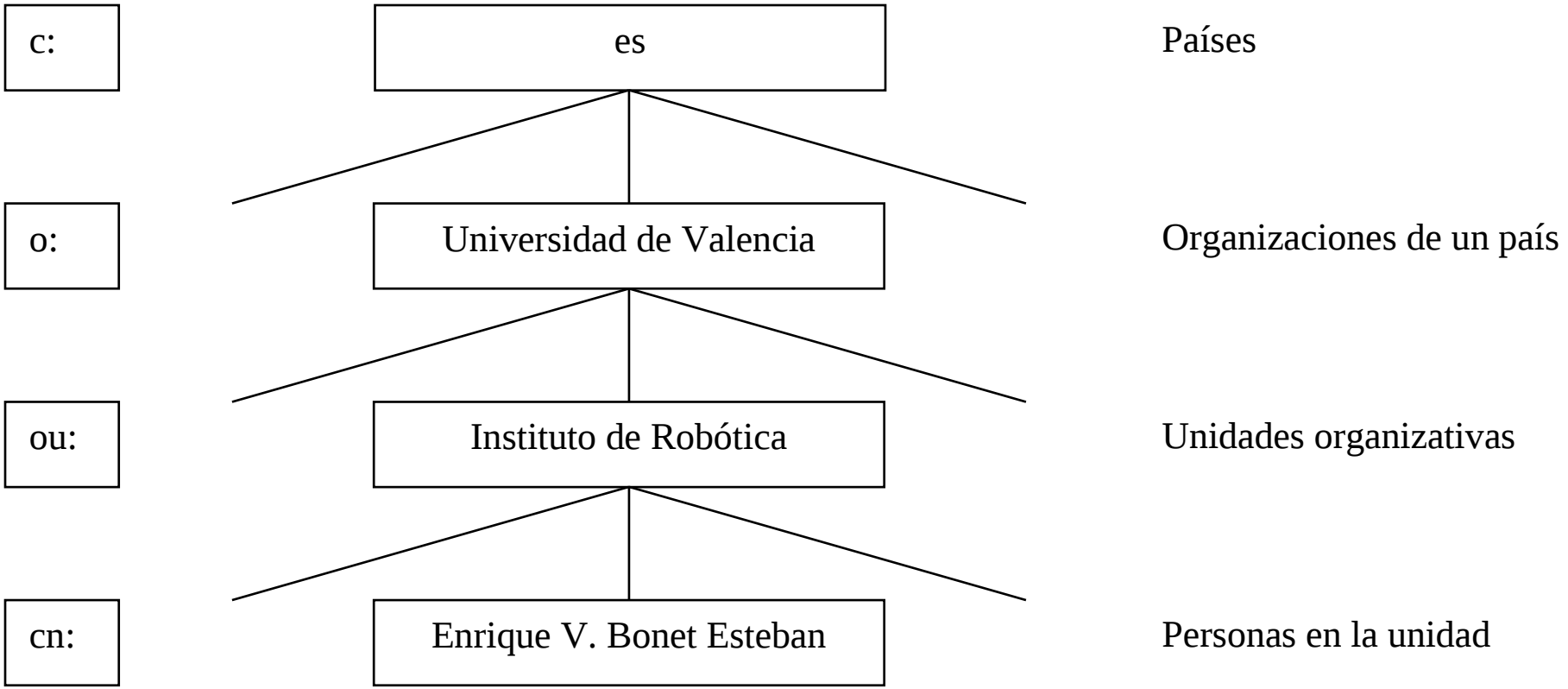
Introducción (III)

- Los servicios de directorios:
  - Pueden ser replicados fácilmente entre distintos ordenadores, incrementado:
    - Disponibilidad.
    - Fiabilidad.
  - Se permiten inconsistencias temporales entre las réplicas.
- Las ventajas de LDAP son:
  - Centralización de la información.
  - Sencillez de administración.
  - Posibilidad de utilizar protocolos seguros.

Estructura de la información (I)

- La información se representa:
  - Mediante un modelo abstracto.
  - Cada elemento debe tener un Distinguished Name (dn).
  - Cada dn debe ser único dentro de la estructura.
- Toda la información de cada elemento se asocia al elemento.

Estructura de la información (II)



Estructura de la información (III)

- La información suele estructurarse mediante Domain Component.
- Esta basado en la estructura de los datos de los DNS:
  - Para la Universidad de Valencia, dc=uv, dc=es.
- Pueden añadirse niveles intermedios si la estructura de DC es muy grande:
  - ou=Instituto de Robótica, dc=uv, dc=es.
  - dc=irobot, dc=uv, dc=es.

Ficheros de extensión LDIF (I)

- Son archivos de texto UTF-8 que permiten insertar y extraer información de LDAP.
- Su estructura es:

```
dn: <distinguished name>
{<attrdesc>: <attrvalue> | <attrdesc>:: <base64-
  encode-value> |<attrdesc>:< <URL>}
...
```

- El final de un elemento y el comienzo de otro se indica con una línea en blanco.
- Las líneas que comienzan por # son comentarios.

Ficheros de extensión LDIF (II)

```
# Comienzo registro 1
dn: <nombre distintivo del registro 1>
<tipo de atributo>: <valor del atributo>
...
# Fin registro 1 (le sigue línea en blanco que indica su fin)

# Comienzo registro 2
dn: <nombre distintivo del registro 2>
<tipo de atributo>: <valor del atributo>
...
# Fin registro 2 (le sigue línea en blanco que indica su fin)

# Registro 3
dn: <nombre distintivo del registro 2>
<tipo de atributo>: <valor del atributo>
...
# Fin registro 3 (le sigue línea en blanco o bien, por ser el
# último registro, el final de fichero).
```

Ficheros de extensión LDIF (III)

- Los archivos LDIF permiten además especificar modificaciones en las entradas de LDAP.
- Su sintaxis es:

```
dn: <nombre distintivo>
changetype: <{add|modify|delete|modrdn}>
...
```

- Donde según el valor de changetype la sintaxis es distinta.
- Más información en los apuntes.

Ficheros de extensión SCHEMA (I)

- La información de los elementos (registros) que se desee introducir en LDAP debe estar definida en unos archivos llamados esquemas (extensión SCHEMA).
- Los esquemas contienen información relativa a:
  - Atributos.
  - Clases de objetos.
- Los esquemas por defecto se encuentran en */etc/openldap/schemas* o sus subdirectorios.

Ficheros de extensión SCHEMA (II)

- Los atributos son los elementos que forman parte de un registro (nombre, apellidos, correo, etc.).
- Las clases de objeto definen el tipo de registro que se desea construir, indicando los atributos existentes y su tipo:
  - MUST: Atributo obligatorio del registro.
  - MAY: Atributo opcional del registro.
- Un registro puede tener atributos de una o varias clases de objeto.

Ficheros de extensión SCHEMA (III)

- Ejemplo de usuarios de ordenador.
- Esquema: nis.schema.
- Clase de objeto: posixAccount.

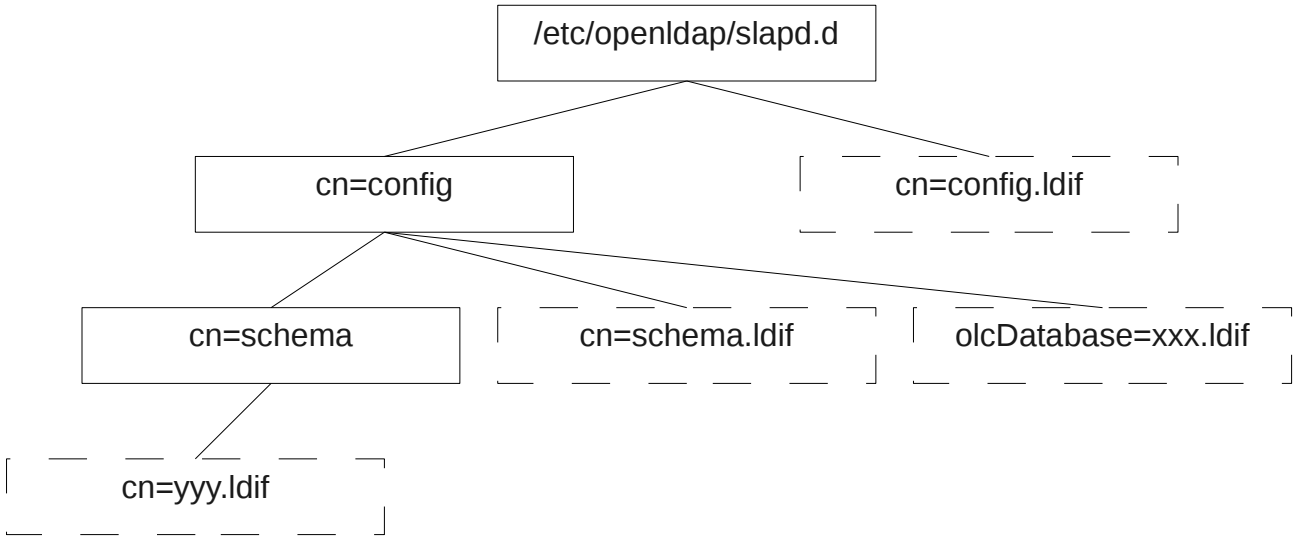
```
object class ( 1.3.6.1.1.1.2.0 NAME 'posixAccount'
DESC 'Abstraction of an account with POSIX attributes'
SUP top AUXILIARY
MUST ( cn $ uid $ uidNumber $ gidNumber $ homeDirectory )
MAY ( userPassword $ loginShell $ gecos $ description ) )
```

- Para estos atributos en un fichero LDIF:

```
objectclass: account
objectclass: posixAccount
```

Configuración del servidor (I)

- El servidor de LDAP es el programa /usr/sbin/slapd.
- La configuración antigua se realizaba en el fichero /etc/openldap/slapd.conf.
- En la actualidad se realiza utilizando una estructura de backend similar a cualquier otro registro de LDAP.



Configuración del servidor (II)

- La configuración general del servidor se realiza en el fichero *cn=config.ldif*.
- La configuración de los backends existentes en el servidor se realiza en los ficheros existentes dentro del directorio *cn=config*.
- Los nombres de los backend existentes se forman como:

  `olcDatabase={número}<tipo de backend>.ldif`
- Los tipos de backend son:
  - *frontend*.
  - Uno de los de la tabla siguiente.

Configuración del servidor (III)

Tipo	Descripción
bdb	Backend transaccional de tipo de base de datos Berkeley.
config	Backend de configuración de slapd.
dnssrv	Backend DNS SRV (especificación de datos a los DNS de los ordenadores y número de puertos de servicios).
hdb	Variante jerárquica de un backend BDB.
ldap	Backend de proxy LDAP.
ldif	Backend LDIF (formato de intercambio de datos de LDAP).
meta	Backend de meta directorio.
monitor	Backend de monitorización.
passwd	Proporciona acceso en modo de solo lectura al fichero de password.
perl	Backend de perl programable.
shell	Backend de shell (programa externo).
sql	Backend programable de SQL.

Configuración del servidor (IV)

- Los backend existentes por defecto son:

```
olcDatabase={0}config.ldif
olcDatabase={-1}frontend.ldif
olcDatabase={1}monitor.ldif
olcDatabase={2}hdb.ldif
```

- Que son creados en la instalación desde el fichero:

```
/usr/share/openldap-servers/slapd.ldif
```

- Con el comando:

```
/usr/libexec/openldap/convert-config.sh -f
  /usr/share/openldap-servers/slapd.ldif
```

Configuración del servidor (V)

- Debemos modificar la configuración de slapd.ldif:

```
...
#
# TLS settings
#
olcTLSCipherSuite: HIGH:MEDIUM:+SSLv2
olcTLSCACertificateFile: /etc/openldap/certs/cacert.pem
olcTLSCertificateFile: /etc/openldap/certs/slapdcert.pem
olcTLSCertificateKeyFile: /etc/openldap/certs/slapdkey.pem
...
```

Configuración del servidor (VI)

```
#
# Schema settings
#
dn: cn=schema,cn=config
objectClass: olcSchemaConfig
cn: schema
include file:///etc/openldap/schema/core.ldif
include file:///etc/openldap/schema/cosine.ldif
include file:///etc/openldap/schema/nis.ldif
...
```

Configuración del servidor (VII)

```
#
# Backend database definitions
#
...
olcSuffix: dc=irobot,dc=uv,dc=es
olcRootDN: cn=administrador,dc=irobot,dc=uv,dc=es
olcRootPW: {SSHA}k97qF7R6BGpHzDgZHt6PX3XoeIAjV5Un
...
```

- La contraseña se crea con:

```
slappasswd -h {modo}
```

Configuración del servidor (VIII)

- El atributo:

`olcDbDirectory`

- Indica el directorio donde se almacena el backend. El directorio debe existir, ser del usuario ldap y tener permisos 0700.
- LDAP sobre Linux funciona por defecto mediante SASL, para no usarlo todo comando debe tener la opción -x.

Inserción de datos (I)

- El primer elemento que se inserta es el que define el elemento raíz:

```
dn: dc=irobot,dc=uv,dc=es
objectclass: dcObject
objectclass: organization
dc: irobot
o: IRTIC
```

- Que indica la organización de la que almacena datos el backend.
- Los datos se insertan con el comando:

```
ldapadd -x –H ldap://localhost.localdomain -D
  "cn=administrador,dc=irobot,dc=uv,dc=es" -W –f
  <fichero>
```

Inserción de datos (II)

- La inserción de otros elementos se realiza de forma similar:

```
dn: uid=ebonet,dc=irobot,dc=uv,dc=es
objectclass: account
objectclass: posixAccount
uid: ebonet
cn: Enrique V. Bonet Esteban
uidNumber: 1000
gidNumber: 100
homeDirectory: /home/ebonet
loginShell: /bin/bash

dn: uid=mamloba,dc=irobot,dc=uv,dc=es
objectclass: account
objectclass: posixAccount
uid: mamloba
cn: M. Amparo Lopez Ballesteros
uidNumber: 1001
gidNumber: 100
homeDirectory: /home/mamloba
loginShell: /bin/bash
```

Inserción de datos (III)

- La inserción de contraseñas se realiza como:

```
ldappasswd -S -x –H ldap://localhost.localdomain -D
  "cn=administrador,dc=irobot,dc=uv,dc=es" -W
  "uid=ebonet,dc=irobot,dc=uv,dc=es"
```

- Introduciendo la contraseña por teclado.

Configuración del cliente (I)

- Se realiza en el fichero */etc/openldap/ldap.conf*.
- Existen tres líneas básicas en la configuración:
  - Definición del servidor/es.

```
URI ldap://<nombre|dirección IP>[:puerto]
```

  - Definición del sufijo por defecto en las consultas.

```
BASE <base>
```

  - Definición de la acción a realizar al comprobar un certificado del servidor.

```
TLS_REQCERT <valor>
```

Configuración del cliente (II)

- Los posibles valores del campo <valor> son:
  - *never*: No se solicita un certificado digital.
  - *allow*: Se solicita un certificado digital:
    - Se continúa la ejecución si no se proporciona
    - Se continúa la ejecución si no puede verificarse.
  - *try*: Se solicita un certificado digital:
    - Se continúa la ejecución si no se proporciona.
    - Se termina la ejecución si no puede verificarse.
  - *demand/hard*: Se solicita un certificado digital:
    - Se termina la ejecución si no se proporciona.
    - Se termina la ejecución si no puede verificarse.

Configuración del cliente (III)

- Si se utilizan certificados, la línea:
  - *TLS_CACERT*: Indica la clave pública de la autoridad de certificación reconocida por el cliente de LDAP.
- Si la configuración es correcta, la consulta:

```
ldapsearch -x -Z
```

Debe funcionar y devolver todo el contenido del backend.

Control de acceso a la información

- Debemos poder controlar el acceso a la información cuando esta es sensible (usuario, contraseña, etc.).
- Para ello se utilizan Access Control List:
  - Se especifican en el fichero:

```
/etc/openldap/slapd.d/cn=config/olcDatabase={1}hdb.ldif
```

- Su sintaxis es:

```
olcAccess: to <a que> [by <por quién> <permisos de acceso> [
   <control>] ]+
```

- Condiciones de <*permisos de acceso*> a entradas y/ó atributos <*a que*> <*por quién*>.

El campo <a que> (I)

- Indica a que entradas se aplican las condiciones.
- Sus valores posibles son:
  - `*`
  - `dn[.<alcance>]=<DN>`
  - `filter=<filtro_ldap>`
  - `attrs=<lista_atributos>[val[.<estilo>]=<valor>]`

El campo <a que> (II)

- *: Todas las entradas.
- *dn[.<alcance>]=<DN>*, según:
  - Nombre distintivo.
  - Valor de <alcance>.

Valor	**Descripción**
base	Es el valor por defecto, e indica los elementos cuyo dn coinciden con el especificado.
one	Indica elementos cuyo padre es el dn especificado.
subtree	Indica todos los elementos que se encuentran en el subárbol que empieza en el dn especificado, incluyendo el propio dn.
children	Igual que *subtree* pero sin incluir el propio dn.

El campo <a que> (III)

- *filter=<filtro_ldap>*: Según un filtro LDAP del formato RFC 2254.

```
filter=(objectClass=account)
```

- *attrs=<lista_atributos>[val.<estilo>]=<valor>]*:
  - Según la lista de atributos especificada.
  - Según el valor particular de un atributo si se utiliza completo.
    - Solo un valor de atributo puede indicarse.
    - Si *<estilo>=exact* el atributo debe ser igual al <valor>.
    - Si *<estilo>=regex*, <valor> es una expresión regular.

El campo <por quién> (I)

- Especifica a que usuarios se aplica el control de acceso.
- Las principales formas de especificación son:
  - `*`
  - `anonymous`
  - `users`
  - `self`
  - `dn[.<estilo>]=<valor>`
  - `dn.<alcance>=<DN>`
- Pueden existir varios campos <quién> dentro de una misma ACL.

El campo <por quién> (II)

- *: Todos los usuarios.
- *anonymous*: Usuarios no autenticados.
- *users*: Usuarios autenticados.
- *self*: La propia entrada (el propio usuario).
- *dn[.<estilo>]=<valor>*: El nombre distintivo
  - Si *<estilo>=exact* debe ser igual al <valor>.
  - Si *<estilo>=regex*, <valor> es una expresión regular.
- *dn.<alcance>=<DN>*: Idéntico a la opción similar del campo <que>.

El campo <permisos de acceso> (I)

- Puede tomar valores:
  - `self`
  - `<nivel>`
  - `<privilegios>`
- Donde self indica la propia entrada.
- <nivel> toma uno de los valores:
  - none
  - auth
  - compare
  - search
  - read
  - write

El campo <permisos de acceso> (II)

- <*privilegios*> tiene la sintaxis:

```
<privilegios> = {=|+|-}{w|r|s|c|x|0}+
```

  - = Comienza la asignación de privilegios.
  - + Añade privilegios.
  - - Elimina privilegios.
  - w: Escritura.
  - r: Lectura.
  - s: Búsqueda.
  - c: Comparación.
  - x: Autenticación.
  - 0: Ningún privilegio.

El campo <control> (I)

- Puede tomar los valores:
  - *stop*: Valor por defecto. Si un campo coincide se dejan de procesar los siguientes <quién> existentes.
  - *continue*: Permite seguir procesando campos <quién> aunque este sea una coincidencia.
  - *break*: Se siguen procesando campos <acceso> aunque este sea una coincidencia.

El campo <control> (II)

- Permitir búsqueda y comparación en el atributo cn a todo el árbol “dc=uv, dc=es” y añadir lectura a los usuarios autenticados:

```
access to dn.subtree="dc=uv,dc=es" attrs=cn by * =cs
  continue by users +r
```

- Permitir búsqueda y comparación en el atributo cn a todo el árbol “dc=uv, dc=es” y añadir lectura para el usuario de id=ebonet:

```
access to dn.subtree="dc=uv,dc=es" attrs=cn by * =cs
  break
access to dn.subtree="id=ebonet,dc=uv,dc=es" by * +r
```

Ejemplo

- Un ejemplo que permite controlar el acceso a un backend que autoriza/deniega el acceso a los sistemas es el siguiente:

```
olcAccess: to attrs=userPassword by
  dn="cn=administrador,dc=irobot,dc=uv,dc=es" write by
  self write by * auth
olcAccess: to * by
  dn="cn=administrador,dc=irobot,dc=uv,dc=es" write by
  * read
```