



Intelligent
Systems
Lab

Administration of Operating Systems

NIS & LDAP

Chapter 21

November 21, 2011

Network Information Service



- Client-server directory service protocol
 - distributes configuration data
 - user and password information
 - ...
- Originally developed by Sun Microsystems
 - for their Solaris operating system
- Simplifies maintenance and administration of multiple computers on a network
 - by keeping information in one place
- Makes it easier for users
 - network environment becomes transparent
- Similar to Active Directory from Windows world

Local Administration



- Local user info stored in `/etc/passwd`
 - and `/etc/shadow`
- Users can be created by adding new lines
 - use `useradd` command
- Users can change their data using `passwd`
 - password, login shell, etc.
 - still just modification of those files
- In a networked environment, this does not work
 - users don't want to remember different password for different computers
 - administrators should not need to update configuration in multiple places

NIS Maps



Intelligent
Systems
Lab

- Useful for roaming profiles and passwords
- Files indexed for efficient access
 - `passwd.byname`
 - `passwd.byuid`
- Client-server model
 - servers can be replicated
 - master/slave
- Originally called *Yellow Pages*
 - a registered trademark of British Telecom
- NIS+
 - more features and more security
 - more administration overhead

NIS Domain



- Each NIS domain has a name
 - exactly one master server
 - can have multiple slave servers
- NIS server can serve multiple domains
- NIS client belongs to one domain at a time
 - but can move between domains easily
 - for example, work/home setup
- NIS traffic is not encrypted
 - this can be a problem
- It only has weak security model
 - mainly based on IP address
- Do not put sensitive information in NIS maps

Name Service Switch



- `/etc/nsswitch.conf`
 - lists sources of information
 - and their lookup order
- Controls how system obtains information
 - `passwd`, `group`, `aliases`, `hosts`, `netmasks`, ...
- Supported in most modern C libraries
 - thus available to user programs
 - not requiring them to implement NIS protocol
- Sample entries
 - `passwd: files ldap`
 - `hosts: files ldap dns`
 - `netmasks: files`

Installation



- NIS client
 - `sudo apt-get install nis`
 - automatically starts NIS client
 - `/etc/defaultdomain`
 - client immediately tries to find and connect to one of NIS servers for this domain
- NIS server
 - `sudo apt-get install nis`
 - `/etc/default/nis`
 - `NISSERVER=false|slave|master`
 - `NISCLIENT=true`
- One machine can be running both

NIS Operation



- `/etc/yp.conf`
 - specify address of NIS server
 - broadcast mode
- `passwd` versus `yppasswd`
 - make sure typing `passwd` does the right thing for your users
- `ypinit`
 - rebuild NIS database
 - for example, after adding new users
- Access rules
 - `host:domain:map:security`
 - `none, port or deny`

LDAP



- Lightweight Directory Access Protocol
 - originally called LDBP, Lightweight Directory Browsing Protocol
- Significantly improved security model
- An alternative to X.500 series of standards
 - directory of arbitrary data
 - organised in a tree hierarchy
- Each directory entry consists of a set of attributes
 - name + value, as defined in a *schema*
- Each entry has a *Distinguished Name*
 - Relative Distinguished Name (RDN)
 - plus parent entry's DN

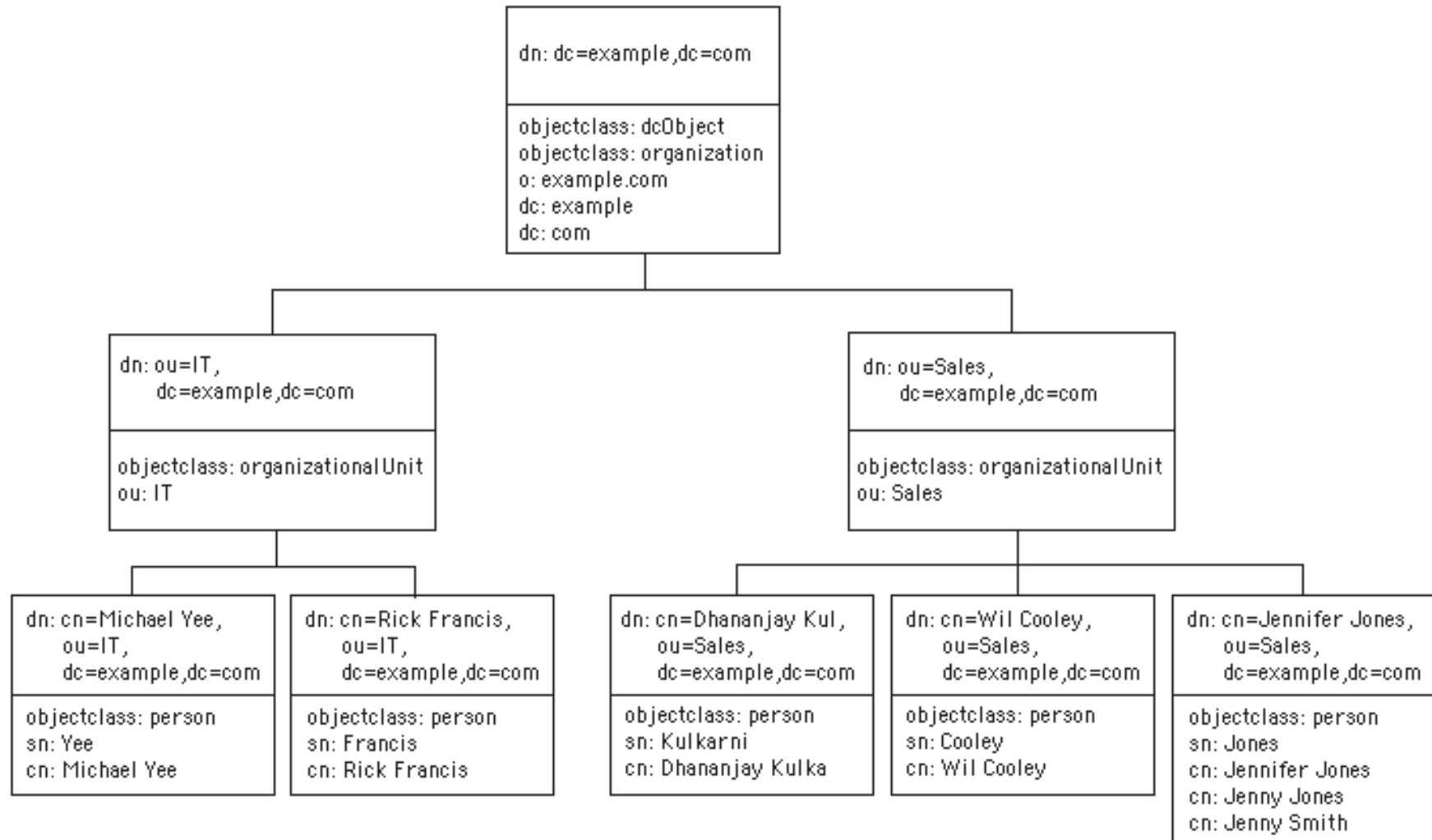
LDAP Data



Intelligent
Systems
Lab

- Specialised database
 - optimised for reading and searching
 - updates are less efficient
- Can store arbitrary data
 - passwords
 - phone numbers
 - date of birth
 - jpeg photos
 - ...
- Client-server model
- OpenLDAP is an open source implementation

LDAP Hierarchy



LDAP Information Model



- LDAP entry is a collection of attributes
 - it has a unique Distinguished Name (DN)
 - uid=jane,ou=People,dc=cims,dc=nyu,dc=edu
- Each attribute has one or more values
 - of an appropriate type
 - telephoneNumber: 212-995-1234
- objectClass attributes decide what attributes are required and allowed for particular entries
 - as defined in the schema
- ldapsearch & ldapadd/ldapmodify/ldapdelete
- *Bind* as some DN or anonymously
 - LDAP uses access control list

LDAP Schema



- Rules that describe what kind of data is stored
- Helps maintain consistency and quality of data
- Reduces duplication of data
- *Object class* rules that the entry must follow
- Schema contains the following:
 - required attributes
 - allowed attributes
 - how to compare attributes
 - attributes types
 - structure of the hierarchy
- Many schemas for various kinds of directories are publicly available



- Pluggable Authentication Module
 - first proposed by Sun Microsystems in 1995
- Centralised authentication mechanism
 - simplifies application design
- “Plug in” different authentication methods
 - different services can have different authentication policies
 - highly secure systems can require multiple different ways to authenticate
 - biometric data in addition to passwords
 - access during work hours can be easier than in the evenings

Kerberos



- Distributed authentication protocol
 - probably most commonly used in practice
- Designed in order to allow workstation users access network resources in a secure way
- Kerberos distinguishes four entities
- Alice, the user, sitting at a client workstation
- Three kinds of servers
 - Authentication server — verifies user login
 - Ticket-granting server — issues *proof of identity* tickets for user-server pairs
 - Bob the server — the server providing applications or services Alice wants to use

Authentication Server

- AS shares a secret password with every user
 - Alice trusts AS with her password
 - but does not want to send it over network
 - Alice does not trust Bob
 - at least not enough to send password to him
- Alice sits down at an arbitrary workstation and types in her name
 - workstation sends it to AS *in plain text*
 - AS sends *session key* and a *ticket* $K_{TGS}(A, K_S)$
 - encrypted using Alice's secret key
 - workstation asks Alice for password and decrypts session key and $K_{TGS}(A, K_S)$
 - and deletes Alice's password from memory



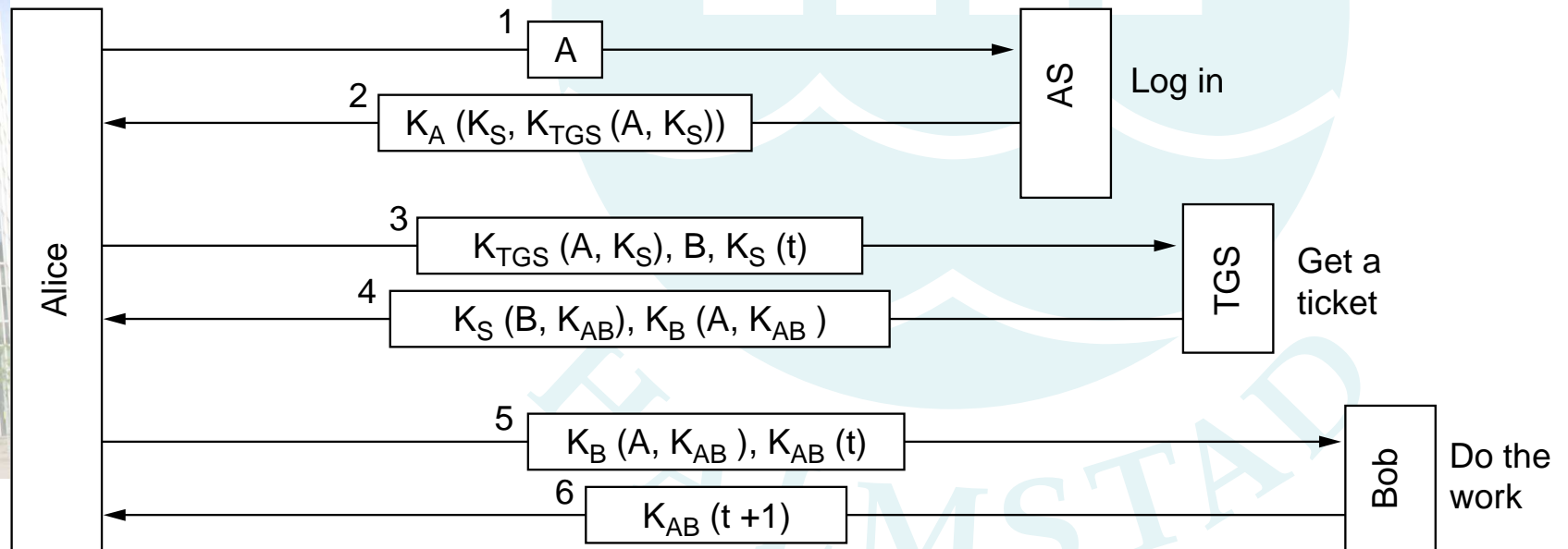
Ticket-Granting Server



- After Alice logs in, she may decide that she wants to contact Bob the server
- Workstation sends message to TGS, asking for ticket to be used for communication with Bob
 - the message contains $K_{TGS}(A, K_S)$
 - i.e. is encrypted with TGS public key
 - so that TGS, and only TGS, can read it
- TGS sends back a session key, K_{AB} , to be used for conversation between Alice and Bob — in two versions
 - one encrypted using K_S , so Alice can read it
 - one encrypted using K_B , Bob's secret key
 - so that Bob can be sure it comes from TGS

Bob the Server

- Finally, Alice sends session key, K_{AB} , to Bob
 - she also sends encrypted timestamp, to guard against replay attacks
- Bob replies with current timestamp, encrypted
 - a proof for Alice that she is indeed talking to Bob and not to Trudy



Kerberos



- Alice can now easily access whichever server she wants on the whole network
 - without ever showing her password to any single one of them
 - actually, without *ever* sending her password through the network to anyone
- The password is only stored in her local workstation memory for a few milliseconds



Intelligent
Systems
Lab

Questions?

