Maîtrise Informatique
M3: Administration Système et Systèmes Distribués

DS – Session de Février 99

Denis Caromel

Professeur
Université de Nice - Sophia Antipolis, Département Informatique

Les documents et calculatrices sont autorisés.
Les réponses ne sont pas longues, soyez concis!
Notez sur votre feuille les numéros des questions (exemple: 3. b.).
Durée: 2H00

1 (4 points)

D'une façon **synthétique**, expliquer le principe de l'initialisation du système Unix Solaris à partir du lancement du processus **init**.
Expliquer comment configurer cette partie du système, ainsi que la gestion et le principe de fonctionnement des scripts de démarrage.

3/4 de page maxi.

2 (2 points)

a. Dans quels cas doit-on faire un serveur d'installation?
b. Quels sont les moyens disponibles pour que le serveur trouve le numéro IP d'un client à partir de son nom?

3 (2 points)

a. Comment, et où sont attribuées et stockées les adresses Ethernet?
b. Comment une machine standard, qui n'est pas un serveur, accède-t-elle à une machine quelconque, par exemple lors d'un commande du type: ftp enee.usc.edu ?
c. Quel(s) fichier(s) de configuration allez-vous regarder en cas de problème?

4 (2 points)

a. Quel(s) principe(s) permet(tent) au DNS de faire la correspondance entre une adresse IP et un nom de domaine ?
b. Expliquer brièvement son fonctionnement, donner un exemple.

5 (2 points)

Donner, d'une façon structurée en au moins trois groupes (Physique, Réseau/Transport, Application), les noms des fichiers de configuration réseau d'une machine Unix (Solaris).

6 (2 points)

a. Quel est le principal avantage de Corba sur RMI ?
b. Quel est son principal inconvénient ?

7 (2 points)

Voici deux fichiers de configuration. Que se passe-t-il?

```
Sur Nil: /etc/vfstab
--------------------
#device         device          mount           FS      fsck    mount   mount
#to mount       to fsck         point           type    pass    at boot options
#
...
/dev/dsk/c0t0d0s0       /dev/rdsk/c0t0d0s0      /       ufs     1       no      -
/dev/dsk/c0t0d0s6       /dev/rdsk/c0t0d0s6      /usr    ufs     1       no      -
/dev/dsk/c0t0d0s4       /dev/rdsk/c0t0d0s4      /var    ufs     1       no      -
...
thalie:/net1            -                       /net1   nfs     -       yes     bg,suid
eea:/net3               -                       /net3   nfs     -       yes     bg,suid
erato:/net4             -                       /net4   nfs     -       yes     bg,suid
...

Sur thalie: /etc/dfs/dfstab
----------------------------
share -F nfs -o ro=SERVEUR:JAZZ:LANGAGE:MONTAGNE:BD:SONDE:FLEUVE:POETES:MACHPROF:LABOINFO /usr
share -F nfs -o ro=SERVEUR:JAZZ:LANGAGE:MONTAGNE:BD:SONDE:FLEUVE:POETES:MACHPROF:LABOINFO /net
share -F nfs -o ro=SERVEUR:JAZZ:LANGAGE:MONTAGNE:BD:SONDE:FLEUVE:POETES:MACHPROF:LABOINFO /net2
share -F nfs -o rw=SERVEUR:JAZZ:LANGAGE:MONTAGNE:BD:SONDE:FLEUVE:POETES:MACHPROF:LABOINFO /home1
share -F nfs -o ro=SERVEUR:JAZZ:LANGAGE:MONTAGNE:BD:SONDE:FLEUVE:POETES:MACHPROF:LABOINFO /home
share -F nfs -o ro=SERVEUR:JAZZ:LANGAGE:MONTAGNE:BD:SONDE:FLEUVE:POETES:MACHPROF:LABOINFO /opt
share -F nfs -o rw=SERVEUR:JAZZ:LANGAGE:MONTAGNE:BD:SONDE:FLEUVE,anon=0 /export
```

8 (5 points)

En vous appuyant sur cet exemple:

```java
public interface I extends Remote {
    Object foo (Object o) throws RemoteException;
    Object bar (Object o1, Object o2) throws RemoteException;
}
public class II extends UnicastRemoteObject implements I
{
    public I () throws RemoteException {
        super();
    }
    public Object foo (Object o) {
        ...;
    }
    public Object bar (Object o1, Object o2) {
        ...;
    }
    public static void main(String[] args) {
        String name = "//localhost/Exam";
        try {
            I i = new II();
            Naming.rebind(name, i); // (*** 1 ***)
            System.out.println("II bound");
        } catch (Exception e) {
            System.err.println("II exception: " +  e.getMessage());
            e.printStackTrace();
        }
    }
}
public class IC {
    public static void main(String args[]) {
        if (System.getSecurityManager() == null) {
            System.setSecurityManager(new RMISecurityManager());
        }
        try {
            String name = "//" + args[0] + "/Exam";
            I i = (I) Naming.lookup(name); // (*** 2 ***)
            Object a1 = i.foo (...); // (*** 3 ***)
            Object a2 = i.bar (...);
        } catch (Exception e) {
            System.err.println("I exception: " + e.getMessage());
            e.printStackTrace();
        }
    }
}
```

a. Expliquez le fonctionnement de cet exemple, et les opérations réalisées, en particulier aux lignes numérotées.

b. Quel est le rôle de la commande **rmic**? Expliquez le principe de fonctionnement du programme ci-dessus à l'exécution en vous aidant de dessins.

c. Peut-on se passer du **rmiregistry** ? Si oui, comment ?

d. Sur quelle(s) classe(s) va-t-on générer le stub et le skeleton ?

e. Proposer d'une façon synthétique et abstraite, le code des classes X_Stub.java et X_Skel.java de cet exemple.