CS 564, Assignment 5

This assignment can be solved in teams of no more than *three students*,
is worth *three tokens*, and
is due on *18 April at 11:59 pm*

This assignment consists of two parts. For the first part you need to complete a survey of existing solutions for the remote service discovery problem. The second part asks for an implementation of a particular kind of remote service discovery (namely, peer discovery on a local network). This second part will consist of an enhancement to your solution for Assignment 4 and is to be developed concurrently with your solution for that assignment. As a consequence, the teams being formed for Assignment 4 must be maintained for this assignment. Note that mixed (graduate and undergraduate) teams are allowed, but in such teams the undergraduate members will not receive credit for this assignment.

1 Remote Service Discovery

Remote service discovery is an important problem for many distributed applications. As the name implies, the problem requires that an application identifies a remote application that offers a desired service. Real-world examples abound and include the Dynamic Host Configuration Protocol (DHCP) which allows a machine to join a network automatically, peer-to-peer network clients (such as BitTorrent) that need to connect to their peers, and remote desktop applications (such as TeamViewer) which need to identify the machine to connect to and control. Note that the problem can be stated in terms of a local network or an internet.

There are several solutions for the remote service discovery problem. They can be roughly classified in centralized services (which use well-known servers to act as service directories) and ad-hoc solutions (which accomplish service discovery in a decentralized manner).

The first part of this assignment asks you to perform a critical survey of the existing solutions for the remote service discovery problem. You need to perform a literature search to discover solutions, describe briefly each solution thus found, state its advantages and disadvantages compared to other solutions, and also mention the applicability of the solution (local networks or internets or both). While comparing the solutions do not forget to mention the suitability of each to various real-world environments and applications.

2 Peer Discovery in a Local Network

Peer discovery is a particular instance of remote service discovery. In this variant distributed processing is accomplished by several participant applications that run on separate machines. The IP addresses of these machines are however not known beforehand, so before the distributed processing can begin a supplementary algorithm needs to be run on each participant to discover the other participants. This assignment further assumes that all the participants reside on the same local network.

The second part of the assignment requires that you build an additional functionality into your solution for Assignment 4 as follows:

The new functionality is introduced by the `-x` command line switch. When this switch is not present the file servers behave as described in the handout for Assignment 4. On the other hand, when `-x` is provided on the command line, the server runs an algorithm for discovering the rest of the participants

(which are assumed to all reside on the same local network) and thus populate its list of peers. The server then continues as described in the handout for Assignment 4.

Note that the switch `-x` and the list of peers are mutually incompatible. When both are provided the server should produce an appropriate error message and refuse to launch. Pay particular attention to the fact that at the launch time of one server it is not necessarily the case that the other peers are up. Therefore you should not launch the discovery mechanism right at startup. The most logical time to initiate peer discovery would be either at the time of arrival of the first client, or at the time when a synchronization operation becomes necessary. For the purpose of this assignment the synchronization mechanism must be run at the time the first client connects. You should make sure however that the peer discovery is only run once (it does not make sense to discover more peers in the middle of the run since that would guarantee file inconsistencies) and that all the peers discover each other. In other words, once a server initiate the discovery phase all the peers that are currently up should participate and populate their own list of peers.

While there are multiple ways of accomplishing this, I recommend that you consider a UDP-based implementation.

3 What to submit

You must submit your solution for both Assignment 4 and Assignment 5 together.

For the requirements stated in Section 1 submit a report typeset to PDF. Name this file `GradPaper.pdf` and place it in the root directory of your submission. Your report must include proper citations of references (primary references being strongly recommended). It is not enough to provide a list of references, you must also cite them in the text; in fact I will discard any reference that is not cited.

Your report will be marked for completeness, the appropriateness of references, the organization of your summary, and the clarity of presentation. The length of the report is not part of the marking scheme. I expect that the length of most reports will hover around 5 pages (with 1-inch margins all around and typeset in 10- or 11-point font). For LaTeX users I recommend (though I do not require) using the standard `article` document class with the `fullpage` package.

For the requirements stated in Section 2 provide the switch and functionality described above in your solution for Assignment 4. The messages exchanged during peer discovery as well as the final list of peers *must* be both written in the log file (when the server is detached) or printed to the terminal (when the server is attached).