Computer Systems - Notes Week 1

Ruben Schenk, ruben.schenk@inf.ethz.ch

September 20, 2021

Chapter 1 : Introduction

Computer Systems is the field of computer science that studies the design, implementation, and behavior of real, complete systems of hardware and software.

Chapter 2 : Naming

2.1 Basic definitions

A **name** is an identifier used to refer to an object in a system. It might be a character string, or a string of bits - in principle any concrete value. A **binding** is an association of a name to an object and a **context** is a particular set of bindings. We might also refer to a context as a *namespace* or *name scope*.

Resolution describes the process of, given a name and a context, finding the object to which the name is bound in that context.

> Remark: For a name to designate an object, it must be bound to the object in some context. In other words, whenever names are being used, there is always a context, even if it's implicit.

2.2 Naming networks

A **naming network** is a directed graph whose nodes are either naming contexts or objects, and whose arcs are bindings of names in one context to another context. A **pathname** is an ordered list of names, also referred to as *path components*, which therefore specify a path in the network.

A naming network which is a tree is called a **Naming hierarchy**. Pathnames in a naming hierarchy are sometimes called *tree names*. The unique starting point in a naming hierarchy is called the root.

Example (UNIX name resolution): A UNIX filename is a pathname. The UNIX file system is, for the most part, a naming hierarchy of directories which contain references to other directories of files:

- A UNIX filename which stats with `/`, such as `/usr/bin/dc`, is resolved using the root of the file system, i.e. all paths are bound to the current directory but `/` has an implicit binding to the root.
- A filename which doesn't start with a `/` is resolved using the current working directory as the first context.
- Every context (i.e. directory) in the UNIX file system has a binding of the name `.` to the context itself.
- Each context also has a binding of the name `..`. This is always bound to the parent directory of the context.
- A **file** object can have *more than one name bound to it*, but a **directory** *cannot*.

2.3 Indirect entries and symbolic links

An **indirect entry** is a name which is bound not to an object per se, but instead to another path name.

> Remark: Indirect entries can complicate name resolution since there is no guarantee that the binding will end up at an object at all, or if the name to which the indirect entry is bound is even syntactically valid in the context in which it is to be resolved.

2.4 Pure names and addresses

A **pure name** encodes no useful information about whatever object it refers to.

> Remark: The names we have considered so far are arguably pure names - in particular, the only thing we have done with them is to bind them to an object in a context, and look them up in a context to get the object again. Some people use the word *identifier* to mean a pure name.

An **address** is a name which encodes some information about the location of the object it refers to.

> Remark: An IP address is not a pure name, since it can be used to route a packet to the corresponding network interface without you needing to know anything more about the interface.

2.5 Search paths

A **search path** is an ordered list of contexts which are treated as a single context. To look up a name in such a context, each constituent context is tried in turn until a binding for the name is found.

> Remark: The UNIX shell `PATH` variable, for example: `/home/ruben/bin:/usr/bin:/bin:/sbin:/usr/sbin:/etc` - is a search path context for resolving command names: each directory in the search path is tried in turn to look for a command.

2.6 Synonyms and Homonyms

Synonyms are different names that ultimately resolve to the same object. **Homonyms** are bindings of the same name to different objects.

> Remark: The existence of synonyms turns a naming hierarchy into a directed, possibly cyclic, graph. For this reason, the UNIX file system disallows most cycles by preventing directories from having synonyms except for `.` and `..` - *only files can have multiple names.*