Name Services

- **names:** used to refer to resources
  - computers
  - services
  - remote objects
  - files
  - users
- processes need to be able to name resources to share them
- sometimes descriptive **attributes** of a resource uniquely identify it
- **address:** attribute of an object. Cannot be used as a name, because the object may change its address
- **human-readable names:** e.g. `/etc/passwd`, `http://www.registermachine.com`
- **identifier:** not usually human-readable, e.g. remote object references, NFS file handles
  - more efficiently stored and processed by software
- **pure name:** contains no information about the object itself. Must be looked up to obtain an address before the named resource can be accessed (e.g. names of people are pure names)
- **non-pure name** contains some information about the object, typically address information
- a name is **resolved** when its translated to data about the named resource
- **binding:** association between a name and an object
  - DNS maps human readable domain names to IP addresses/other attributes
  - X500 directory service: can map a person's name onto attributes e.g. email address, phone number

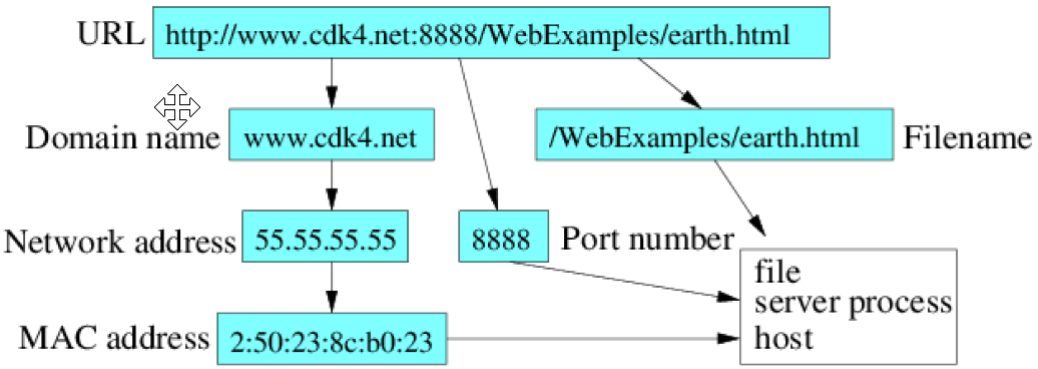

Figure 1: e.g. name

- **local significance:** some names only have meaning to the service that creates it
- services may need to cooperate to have name consistency
  - e.g. NFS users need the same unique ID on both the client and server

Uniform Resource Identifiers

- **URI:** identifies resources on the Web and other Internet resources (email boxes)
  - generic way of specifying identifier to make it easy for common software to process it
  - allows new types of identifiers to be introduced easily and existing identifiers to be used widely
  - a URL (locator) is a URI
  - scheme at start of URI indicates kind of resource being named `scheme:`
- **Uniform Resource Names (URNs):** URIs used as pure resource names rather than locators
  - requires resolution service/name service to translate URN into an actual address
  - `urn:` prefix is allocated for URNs
  - e.g. `urn:ISBN:0-201-62433-8` identifies a resource (book) by ISBN number
- `doi:10.1007/s10707-005-5887-8`
  - lookup service is `http://dx.doi.org/10.1007/s10707-005-5887-8`
  - resolves to `http://springerlink.com/content/c250mnlu2m7n5586/`
  - refers to a document *Building and Querying a P2P Virtual World*

Name Services

- purpose: resolve a name: lookup attributes bound to a name
- separated from other services because:
  - **unification:** resources managed by different services use the same naming scheme, as with URIs
  - **integration:** to share resources in different administrative domains requires naming them. without a common naming service, administrative domains may use different name formats, getting difficult very quickly

Goals of Global Name Service

- handle arbitrary number of names and to serve an arbitrary number of administrative organisations

- **a long lifetime:** over many changes to the names and system
- **high availability:** dependent services stop working if the name server is unavailable
  - e.g. WikiLeaks DNS blocked by US government
- **fault isolation:** local failures do not cause entire service to fail
- **tolerance of mistrust:** large open system cannot have any component that is trusted by all clients
  - false attributes given to names

Name spaces

- **name space:** defines set of names valid for a given service
- structure
  - can be hierarchical, like DNS and UNIX filenames
  - can be flat: e.g. randomly chosen integer ID
- structured names allow
  - efficient lookup
  - name can incorporate semantics about the resource
- length
  - fixed: e.g. 32 bit; easier to store and process
  - unbounded
- **alias:** an alternative name for a resource. Provides transparency.
- **naming domain:** name space for which there exists a single administrative authority for assigning names within it
- administrative authority is usually delegated by division of domain into subdomains, with each sub-domain sharing a common part of the overall name in that name space

Name Resolution

- typically an iterative process: name either resolves to a set of primitive attributes, or it resolves to another name
- aliases mean resolution cycles can occur. Solutions
  - abandon resolution after fixed number of iterations
  - require admins to ensure no cycles occur

Distribution

- large name databases need to be distributed across multiple services
- bottlenecks:
  - network I/O
  - server reliability
- replication can increase availability
- when you delegate name service authority, the service is naturally distributed over delegates: service data is usually distributed with respect to domain ownership

Navigation

- **navigation:** resolve request propagates from one server to another
- **iterative navigation:** client makes request at different servers one at a time, visiting increasingly more specific parts of the domain hierarchy
- **multicast navigation:** multicasts request to group/subset of name servers. Only server with the named request returns a result
- **non-recursive server-controlled navigation:** client sends request to server and the server continues on behalf of the client iteratively
- **recursive server-controlled navigation:** client sends request to a server and server sends request to another server recursively

Caching

- critical to performance of name services
- binding of names to attributes changes infrequently in most circumstances
- results of resolution can be cached by client and server
- eliminates high level name servers from navigation path and allows resolution to proceed despite some server failures

Domain Name System

- name service design whose main naming database is used across the Internet
- prior to DNS, a single central master file was maintained and downloaded to all computers that needed it
  - doesn't scale

  - local organisations cannot administer their own naming systems
  - general name service was needed, not just one for looking up computer addresses

- DNS is designed for use in multiple implementations, each with its own name space

Name space

- name space is partitioned organisationally and geographically
- hierarchical from right to left, delimited by `.`
- each domain authority can specify their own subdomains

Queries

- applications use DNS to resolve host names into IP addresses
- also used to make requests for other services that support a domain, e.g. `MX` for mail server
- **reverse resolution:** allows IP address to be resolved into a domain name
- **host information:** allows information about a host to be obtained. Usually blocked due to security
- **well-known service** allows info about services run by a computer to be returned

Name servers

- database is distributed across a logical network of servers
- DNS naming data are divided into **zones**, containing:
  - attribute data for names in a domain (excluding those contained within a subdomain)
  - name/addresses of at least 2 name servers providing **authoritative** data for the zone
  - names/addresses of name servers holding authoritative data for delegated subdomains
  - zone management parameters: e.g. caching, replication
- 2 name servers need to be specified for each domain to ensure availability in event of a single crash
- **primary/master server** reads zone data from a file
- **secondary server** downloads zone data from primary server
- both primary/secondary servers provide authoritative data for the zone
- any DNS server can cache data from other servers. They need to inform clients that data is not authoritative

Database: Resource Records

- resource records carried by DNS replies are 4-tuples:

```
(Name, Value, Type, TTL)
```

Type	Value
A	IPv4 address for hostname Name
AAAA	IPv6 address for hostname Name
NS	Hostname of authoritative DNS server for domain Name
CNAME	Canonical hostname for alias hostname Name
MX	Mail exchange. Canonical name of a mail server. Allows company to have same aliased name for mail and Web