- Name no more precise →
  Interesting/advanced Topics in Operating Systems
  - scalability
  - systems security
  - modeling
- Some overlap with „Distributed Systems“ (Prof Schill) and some classes by Prof Fetzer
- In some cases no written material (except slides)
Several lectures presented by research-group members.

Strongly suggested: register for mailing list!!!
- for questions and discussions
- only way to inform about short-term issues
- use: “tu-dresden” mail-addresses
EXAM

- oral
- about 1 exam date per month until September (except August)
- Diplom/Master INF study programmes: can be combined with other classes in complex modules
- for application:
  - email to angela.spehr@tu-dresden.de for exam date/time
  - provide paperwork (forms) at least 2 weeks before exam (after that no more cancellation except for sickness)
SCALABILITY IN COMPUTER SYSTEMS
EXAMPLE: DNS/BIND
GOAL OF LECTURES ON SCALABILITY

Outline:

- scalability: terminology, problems
- basic approaches
- case studies

Goal:

- understand some of the important principles how to build scalable systems
OUTLINE AND GOAL OF TODAY'S LECTURE

Outline:

- scalability ...
- names in Distributed Systems: purposes of naming, terminology
- application of scalability approaches on name resolution

Goal:

- understand some of the important principles how to build scalable systems ... using DNS as example
MORE CASE STUDIES IN THE CLASS

- memory consistency
- locks and advanced synchronization approaches
- file systems
- load balancing (MosiX) and HPC
GENERAL DEFINITION: SCALABILITY

Scalability:

- the ease with which a system or component can be modified to fit the problem area
  http://www.sei.cmu.edu/str/indexes/glossary/

Dimensions of Scalability:

- resources: CPUs, memory
- software (versions, better libs, etc.)
- heterogeneity (different hardware / SW = portability)
A system is described as scalable if it remains effective when there is a significant increase in the number of resources and the number of users. (Coulouris, Dollimore, Kindberg: Distributed Systems)

Scalability [in telecommunication and software engineering] indicates the capability of a system to increase performance under an increased load when resources (typically hardware) are added. (Wikipedia)
A SW ENGINEERING ASPECT OF SCALABILITY

Prepare for change in functionality
- software engineering
- choose sufficiently large logical resources
- provide hooks for extension

Not subject of the course
PROBLEMS

- performance bottlenecks / Amdahl's Law
- failures / abuse
- administration
AMDAHL’S LAW

The speedup of a system can be described by the formula:

\[
\text{Speedup}(P,N) = \frac{1}{1 - \frac{P}{N}}
\]

- **P**: section that can be parallelized
- **1-P**: serial section
- **N**: number of CPUs

If **N** becomes VERY large, the speedup approaches: \(1/(1-P)\).
THE "RPC" PRINCIPLES

- partitioning
  - split systems into parts that can operate independently to a large extent

- replication
  - provide several copies of components
    - that are kept consistent eventually
    - that can be used in case of failure of copies

- locality (caching)
  - maintain a copy of information that is nearer, cheaper/faster to access than the original
MORE PRINCIPLES FOR SCALABILITY

- identify and address bottlenecks (!!!)
- specialize functionality/interfaces
- right level of consistency
  - caches, replicates, ... need not always be fully consistent
- lazy information dissemination
- balance load
MORE ON LOAD BALANCE

- balance load
  - keep load under reasonable threshold
    - at the processing components
    - in the communication subsystems
  - load balancing can be static or dynamic.

- Will study a detailed example for dynamic load balancing later (MosiX).
- minimize the delay induced by “RPC”
- prepare for change
- information dissemination
DOMAIN NAME SYSTEM AS AN EXAMPLE

- names and name resolution etc in general
- a bit of history of internet names
- DNS general properties
- RPC in DNS
NAMES, IDENTIFIERS, ADDRESSES

- names
  - symbolic
  - have a meaning for people

- identifiers
  - identifies a component (uniquely)
  - are used by programs

- addresses
  - locates a component & can change
  - can change
name resolution:

- map symbolic names to objects
- in detail: to a set of attributes such as:
  - identifiers, addresses, other names, security properties

Principle interface:

- Register (Name, attributes, ...)
- Lookup (Name) -> attributes
- compilers
  - statically map names to addresses

- dynamic libraries
  - dynamically remap addresses

- port mapper (SUN RPC)
  - map service to port

- Name resolution is a form of dynamic mapping of pathnames to attributes.
Many services, tools, ... provide their own name resolution

- file systems (UNIX: path names to I-Nodes)
- login
- RPC (remote procedure call) systems (portmapper)
PURPOSE OF DIRECTORY SERVICES

- integration of name services
- generic name service
- world-wide use of names
- pervasively used:
  - email/web
  - computer attributes (IP addresses)
  - people attributes (certificates, ...)
  - ...

A BIT OF HISTORY

- UUCP/MMDF (cum grano salis):
  - `ira!gmdzi!oldenburg!heinrich!user` (path to destination)
  - `user@ira!heinrich%gmdzi` (mixing identifiers and path information)
A BIT OF HISTORY

- ARPA-Net at the beginning:
  - a single file: hosts.txt
  - maintained at Network Information Center of SRI (Stanford)
  - accessed via ftp
  - TCP/IP in BSD Unix => chaos name collisions, consistency, load
- DNS: Paul Mockapetries (84) ...
- **naming domain**
  subtree in the hierarchy of DNS contexts

- **zone**
  (aka Zone of authority) Subset of a domain over which an authority has complete control. Subzones (starting at apices of a zone) can be delegated to other authorities.

- **navigation**
  querying in a set of cooperating name spaces
RECURSIVE ./ ITERATIVE

Scalability in Computer Systems, Example: DNS/BIND
REQUIREMENTS / PROPERTIES

- arbitrarily large numbers
- arbitrary units of administration
- long living names, the higher in the hierarchy the longer
- high robustness
- restructuring of name spaces
- consistency
- efficiency
TODAY: hundreds of “top level domains"
EXAMPLES

- inf.tu-dresden.de: domain
- os.inf.tu-dresden.de: computer
- heidelberg.ibm.com: domain
- ftp ftp.inf.tu-dresden.de
  - DNS: \(\rightarrow\) IP address: 141.76.2.3
  - ftp daemon: IP address, port 21

- properties:
  location independent / not very deeply nested
IMPLEMENTATION STRUCTURE (BIND)

- Resolver (runtime library)
  - name server
    - root name server
    - NS2
    - NS3
RPC: PARTITIONS

- Zones:
  - administrative unit

- Name Server:
  - maps to names and addresses of name servers responsible for sub zones
  - maintains management data
  - process doing the name resolution for one zone
  - key interface: Resource records (RR)
example taken from Coulouris et al, Distributed Systems
2 ways of replication:
- several IPs/names
- "any cast" (send packet to one of many servers with same IP)
- 13 root name server IPs, several hundreds of any cast
- each zone has at least one primary and one secondary IP
- each name server caches resource records
- time to live attribute
- authoritative versus non-authoritative answers
<table>
<thead>
<tr>
<th>Record type</th>
<th>Interpretation</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>address</td>
<td>IPv4 address</td>
</tr>
<tr>
<td>AAAA</td>
<td>address</td>
<td>IPv6 address</td>
</tr>
<tr>
<td>NS</td>
<td>Name server</td>
<td>DNS name</td>
</tr>
<tr>
<td>CNAME</td>
<td>Symbolic link</td>
<td>DNS name of canonical name</td>
</tr>
<tr>
<td>SOA</td>
<td>Start of authority</td>
<td>Zone-specific properties</td>
</tr>
<tr>
<td>PTR</td>
<td>IP reverse pointer</td>
<td>DNS name</td>
</tr>
<tr>
<td>HINFO</td>
<td>Host info</td>
<td>Text description of host OS</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>
- Paul Albitz & Cricket Liu
  DNS and BIND
  O’Reilly & Associates, Inc.

- Mark Hill, Michael Marty
  Amdahl's Law in the Multicore Era IEEE

- Couluris, Tollimore, Kindberg
  Distributed systems