Hermann Härtig
Senior Professor
Head of OS group 1994 - 2019
Scalability in Computer Systems, Example: DNS/BIND
- Lecturer in charge of DOS:
  Dr. Carsten Weinhold, Barkhausen Institute TUD
- Several lectures presented by research-group members.
- register for mailing list !!! see Website
  - only way to inform about short-term issues
  - for questions and discussions and lecturing methods
  - must use: “tu-dresden.de” mail-addresses
oral,

details to follow after COVID-19

may be similar as last years’ (see next slide)
LAST YEARS’ EXAMS

- about 1 exam date per month
- for exam appointments:
  - email to angela.spehr@tu-dresden.de for exam date/time
  - provide paperwork (forms) at least 2 weeks before exam:
    if you don’t: appointment is automatically canceled
    you can cancel until 2 weeks before date
    after that, no more cancellation except for sickness
- Diplom/Master INF study programmes:
  can be combined with other classes in complex modules
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)
1.0) DOS ORGANISATION

1.1) SCALABILITY IN COMPUTER SYSTEMS

1.2) EXAMPLE: DNS/BIND
GOAL OF ALL LECTURES ON SCALABILITY

Topics:
- scalability: terminology, problems, principle approaches
- case studies, all layers of compute systems

Goal:
- understand (some of the) important principles how to build scalable systems
Outline:

- scalability ... and simple model to reason about 1 aspect
- names in Distributed Systems: purposes of naming, terminology. (DNS)
- application of scalability approaches on name resolution

Goal:

- understand some of the important principles how to build scalable systems ... using DNS as example
- memory consistency
- locks and advanced synchronization approaches
- file systems
- load balancing (MosiX) and HPC(MPI)
Scalability:

Scalability is the property of a system to handle a growing amount of work by adding resources to the system.

Wikipedia (2019) and many other sources
ability of a system to use growing resources ...

- weak:
  to handle growing load, larger problem, ...

- strong:
  accelerate existing work load, same problem
problems

THINK and PAUSE!
PAUSE THE VIDEO HERE
AND THINK before continuing
PROBLEMS

- performance bottlenecks / Amdahl's Law
- failures / abuse
- administration
RESOURCES AND PERFORMANCE

- processors
- communication
- memory (remember basic OS course: “thrashing”)

\[
\text{Speedup} = \frac{\text{original execution time}}{\text{enhanced execution time}}
\]
SIMPLE MODEL: AMDAHL’S LAW

Speedup: \[ \frac{\text{original execution time}}{\text{enhanced execution time}} \]

Parallel Execution

red: cannot run in Parallel

green: runs \textit{perfectly} parallel

unlimited processors maximum speedup: blue/red
AMDAHL’S LAW

Parallel Execution, N processors

red: cannot run in Parallel

green: runs perfectly parallel

N processors maximum speedup: \( \text{blue}/(\text{red}+\text{green}/N) \)
Parallel Execution, N processors

- red: cannot run in Parallel
- green: runs perfectly parallel

maximum speedup: blue/(red+green/N)
AMDAHL’S LAW

Speedup: $\frac{\text{original execution time}}{\text{enhanced execution time}}$

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

$$\text{Speedup}(P,N) = \frac{1}{1 - P + \frac{P}{N}}$$

- if N becomes VERY large, speedup approaches: $1/(1-P)$
principle approaches

THINK and PAUSE!

PAUSE THE VIDEO HERE
AND THINK before continuing
THE “RPC” PRINCIPLES

- partitioning
  split systems into parts that can operate independently/parallel 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
- identify and address bottlenecks
- specialize functionality/interfaces
- right level of consistency
  caches, replicates, ... need not always be fully consistent
- lazy information dissemination
- balance load (make partitioning dynamic)
1.0) DOS ORGANISATION
1.1) SCALABILITY IN COMPUTER SYSTEMS
1.2) EXAMPLE: DNS/BIND
- 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) ...
Scalability in Computer Systems, Example: DNS/BIND

The Domain Name System (DNS) is a service that translates domain names into IP addresses. The diagram above illustrates the structure of the DNS. Each level represents a domain, and the child nodes represent subdomains. The top level includes generic domains like .com, .org, .edu, etc., and country code domains like .us, .au, .de, .at, .ch, etc. At the bottom level, we have specific host names like .tu-dresden.de, .amazon.com, .bahn.com, etc.

For example, the domain name `tu-dresden.de` can be decomposed as follows:

1. `.de` (country code domain for Germany)
2. `.tu-dresden` (top-level domain for TU Dresden)
3. `.edu` (educational domain)
4. `.org` (generic domain for organizations)
5. `.wikiped` (subdomain)
6. `.tudos` (subdomain)
7. `studium` (subdomain)
8. `inf` (subdomain)
9. `os` (subdomain)
10. `forschung` (subdomain)
11. `xy..` (subdomain)
12. `can..` (subdomain)
13. `erwin` (subdomain)

The root node represents the top level of the DNS hierarchy, which contains a pointer to the root server. The root server does not have any hosts directly associated with it and is responsible for routing requests to the appropriate top-level domain servers.

The email address `haertig@os.inf.tu-dresden.de` corresponds to the domain name system structure shown in the diagram.
NAMES, IDENTIFIERS, ADDRESSES

- names
  - symbolic, many names possible for 1 entity
  - have a meaning for people

- identifiers
  - identifies an entity uniquely
  - are used by programs

- addresses
  - locates an entity
  - changes occasionally (or frequently)
name resolution:
map symbolic names to a set of attributes such as: identifiers, addresses, alias names, security properties encryption keys

Principle interface:
- Register (Context, Name, attributes, ...)
- Lookup (Context, Name) -> attributes
domain subtree in DNS hierarchy:

- de
- tu-dresden.de
- os.inf.tu-dresden.de
- todos.org and os.inf.tu-dresden.de are aliases
- **zone**: Subset of a domain over which an authority has complete control. Controlled by a name server. Subzones can be delegated to other authorities.

- **navigation**: Querying in a set of cooperating name servers.
POTENTIAL ZONES

- complete tu-dresden domain
- complete tu-dresden domain
- with sub zone os (possible but unliked by ZIH)
CACHING

- remember intermediate results
- @ root NS makes no sense! (overload)
- @ NS i !!!
CACHING

......

NS i

NS2

NS3

root NS
RECURSIVE ./ ITERATIVE

Diagram showing the interaction between root NS, NS2, NS3, and NSi in a DNS/BIND scalability example.

 TU Dresden, Hermann Härtig, Distributed Operating Systems, SS2020
 Scalability in Computer Systems, Example: DNS/BIND 36
2 techniques for 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
<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>
main problems for scalability ....

simple model: Amdahl’ law)

few principle approaches ...

DNS as fine example ... more examples to come study DNS it in your first exercise

register in mailing list
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