OS Mechanisms Posix 1003.1c

Linux Threads

INTERNET APPLICATION & SYSTEM DESIGN

- Dynamic Creation
  - Pthread_create
- Concurrent Execution
  - All threads appear to execute at the same time.
- Preemption
  - Sched_yield to voluntary release allocated time.
- Private Local Variables
  - Each thread has private stack.
- Shared Globals
  - All threads within the same process share global variables.
- Shared File Descriptions
  - All threads within the same process shares fds.

Synchronization

INTERNET APPLICATION & SYSTEM DESIGN

- Mutex
  - Generally a separate mutex variable is used for each variables to be protected.
  - pthread_mutext_init
  - pthread_mutext_lock
  - pthread_mutext_unlock.
- Semaphores
  - Generally one semaphore for each resource with count N.
  - Sem_init
  - sem_wait
  - sem_post.
- Condition Variables
  - A tool for applications in need of busy wait

Concurrency and Advanced Network-based Applications

Issues in Client Design-1

INTERNET APPLICATION & SYSTEM DESIGN

- Choosing A Local Port Number
  - A client can choose any port number.
  - No conflict with other port numbers in use.
  - Not a well known server port number.
  - If a process does not explicitly call bind, Connect or listen, which ever is called kernel automatically picks a valid ephemeral port and source IP.
- Choosing A Local IP address
  - In a multi-homed computer applications may not know which IP adapter is in use.
  - While binding, setting IP address =0 (INADDR_ANY) will allow the kernel to pick the correct IP address.

Once a value is assigned by the KERNEL how can application find out the assigned address?

getsockname(socket, &localaddr,&addlen)

Issues in Client Design-2

INTERNET APPLICATION & SYSTEM DESIGN

- Reading from a Stream
  - Only one write operation may require multiple read at the other end.
  - The flag can be 0 or AND of few values such as:
  - MSG_PEEK: look at the data that is available to read, but witout having system discard it from system buffer.
  - MSG_WAITALL: do not return until the specified amount of data is received.
- /* Repeatedly read data from socket and write to user's screen. */

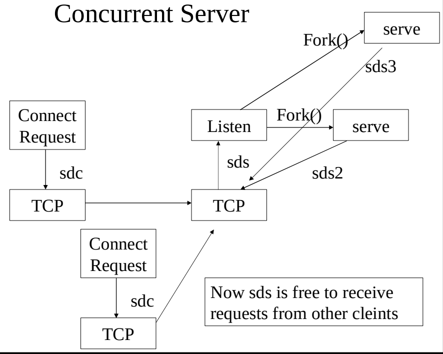
```
n = recv(sd, buf, sizeof(buf), 0);
while (n > 0) {
    write(1,buf,n);
    n = recv(sd, buf, sizeof(buf), 0);
}
```

Issues in Server Design-1

- Binding to a Well Known Port
  - If the service have to be available to a wide audience the server must bind to a well known port.
- Setting up the IP address
  - like clients a server may leave it upto the kernel to set the IP address to avoid confusion in multi-homed hosts.

INTERNET APPLICATION & SYSTEM DESIGN

Concurrent Server



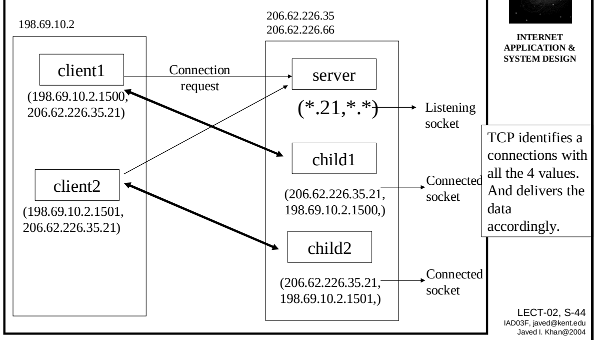
INTERNET APPLICATION & SYSTEM DESIGN

Concurrent server.c

Click Here

INTERNET APPLICATION & SYSTEM DESIGN

TCP Port Numbers and Concurrent Servers



INTERNET APPLICATION & SYSTEM DESIGN

Connection Oriented & Connectionless Communication

- UDP Client
- UDP Server

INTERNET APPLICATION & SYSTEM DESIGN

Issues in Server Design-2

- Concurrent vs. Iterative server
  - simplicity, response time
  - real vs. apparent concurrency
- Connection-Oriented vs. Connectionless Access
  - reliability, persistence, data volume, flow-control

Iterative connectionless	*Iterative connectionoriented*
Concurrent connectionless	*Concurrent connectionoriented*

INTERNET APPLICATION & SYSTEM DESIGN

Concurrency Management

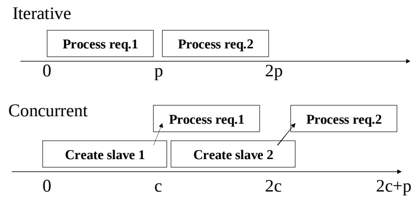
Perspectives:
- User perspective: response time
- System perspective: impact on resources.

Issues:
- How can programmer know whether concurrency is warranted?
- How to determine which design is optimal?
- How can a programmer estimate demand or service time?

Concepts:
- Level of Concurrency
- Demand-Driven Concurrency

Cost of Concurrency

- Overhead and Delay



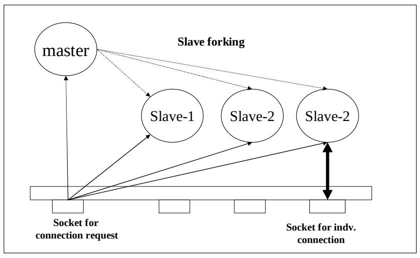
Quiz: if the rate at which requests arrives exceeds 1/c but is less than 1/p which implementation can handle the load?

Small additional delay can be significant for continuous operation of a server under heavy load.

Process Preallocation

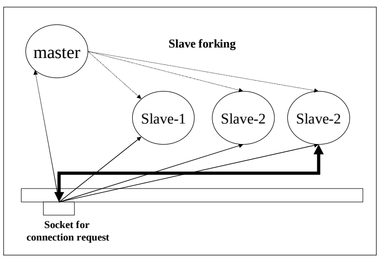
- Demand-driven concurrency can be avoided by limiting the maximum level of concurrency.
- To avoid the run time process creation delay, preallocate service processes.
- Design:
  - Master server creates N slaves at the beginning.
  - Each slave waits/sleeps using OS support.
  - When service request arrives each slave by turn picks it.
  - When done slaves do not exit.
  - Preallocation allows the server process to switch and move to next process faster.

Preallocation in a Connection Oriented Server



QUIZ: How do the slaves decide that two of them do not jump for serving one client?

Preallocation in a Connectionless Server



QUIZ: Just like before mutual exclusion requires OS support.

Slaves use the same descriptor to listen for request and send data. Requests arrive in UDP. Descriptors are automatically freed after each communication.

Delayed Process Allocation

- Standby slave approach solve runtime process creation
- but they too costs in terms of OS resource management.
- Iterative server can yield faster service and higher throughput if the service time is small.
- But, how can programmer know the service time?
- Design:
  - the server estimates the processing time dynamically by looking into service parameters (I/O size/bound).
  - server starts processing a new request iteratively.
  - and starts a timer.
  - On time out, it invokes a concurrent slave.

QUIZ: Delayed process creation and preallocation are they are just opposite principles?

Combined Technique

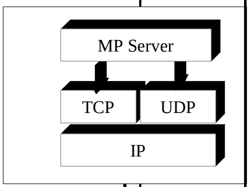
- Design:
  - server starts without any slave.
  - creates a slave only when timer expires.
  - but, once a slave has been created, it does not exit immediately.
  - Slaves can also spawn slaves, if needed.
- How to control concurrency?
  - master specifies limit of concurrency MAX to slaves.
  - Slave exits after a specified period of inactivity.

INTERNET APPLICATION & SYSTEM DESIGN

LECT-02, S-53
IAD03F, javed@kent.edu
Javed I. Khan@2004

Multi Protocol Servers

- One server can talk via several underlying transport layer network protocols.
- Example: DAYTIME server maintained in UNIX systems can talk via TCP and UDP both.
- Advantages:
  - easy software maintenance
  - easy debugging
  - service code can be reused..
  - less impact on system resources.
- Disadvantage:
  - less flexibility for network administrator.
- Design:
  - multiple protocol specific listening ports
  - iterative polling.

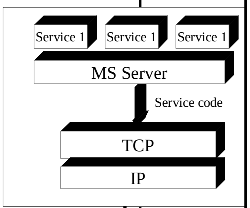


INTERNET APPLICATION & SYSTEM DESIGN

LECT-02, S-54
IAD03F, javed@kent.edu
Javed I. Khan@2004

Multi Service Servers

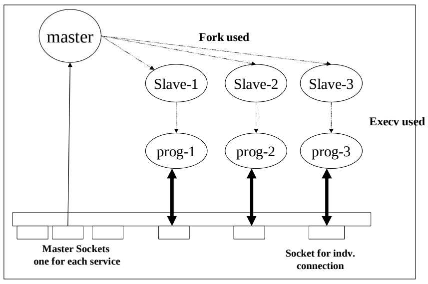
- A typical host may run multiple services. One server can be designed to provide a set of services instead of one.
- Advantages:
  - easy maintenance.
  - communication code can be reused.
  - less impact on system resources.
- Design:
  - requires service descriptor code.
  - server use descriptor code based switch.
  - single process approach
  - concurrent approach
  - super server (UNIX INETD)



INTERNET APPLICATION & SYSTEM DESIGN

LECT-02, S-55
IAD03F, javed@kent.edu
Javed I. Khan@2004

Multiservice Server



INTERNET APPLICATION & SYSTEM DESIGN

LECT-02, S-56
IAD03F, javed@kent.edu
Javed I. Khan@2004

Advantages of Concurrency to Servers

- Improves observed response time.
- Thus it can increase client throughput.
- It can eliminate potential deadlocks and some denial of service attacks.
- It makes server design modular.
- Concurrent execution can easily be ported on multiprocessor hardware.

QUIZ: Can concurrency be of any use to Clients? Clients do not halt others. Possibility of deadlock is less..

INTERNET APPLICATION & SYSTEM DESIGN

LECT-02, S-57
IAD03F, javed@kent.edu
Javed I. Khan@2004

Concurrency in Client Design

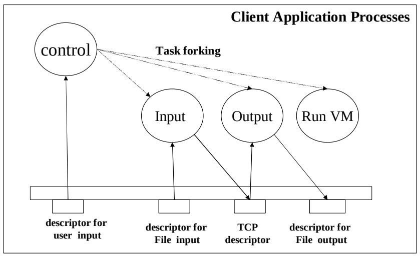
- Concurrency makes client design simpler and modular too.
- Concurrent client can contact several servers without being held by one.
- Concurrency can allow user to change parameters, inquire about client parameters, or control client processing dynamically.

QUIZ: What if when you are reading a web page and it is taking too long to download, and you want to move on to next hyperlink?

INTERNET APPLICATION & SYSTEM DESIGN

LECT-02, S-58
IAD03F, javed@kent.edu
Javed I. Khan@2004

Design of Concurrent Client



- If the server sending data blocks, it can still keep on sending data to server in other direction.

Case Study: Concurrency in Apache

Directive	Definition (default)
StartServers	Initial number of child processes (5)
MaxClients	Maximum number of child processes (256)
MinSpareServers	Target Minimum number of idle children (5)
MaxSpareServers	Target maximum number of idle children (10)
MaxRequestsPerChild	Maximum number of requests per child (30)
ListenBacklog	Maximum number of pending connections (511)
SendBufferSize	Size of the TCP sendbuffer (OS default)
maxKeepAliverequests	Maximum number of requests per connection (10)
KeepAliveTimeout	Maximum idle time for connection (15 sec)

Web Servers

Web Server

- Client Request Handling
- Resource & Access Control
- Dynamical Response Handling
- Performance Enhancement

Client Request Handling

- Read and parse the HTTP request message
- Translate the URL to the file name.
- Determine if the request is authorized.
- Generate and transmit the response.

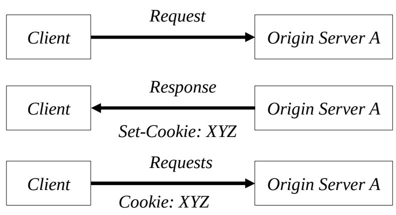
Access Control

- Authentication
  - Identify the user.
  - Must be performed for each requests.
  - Browser typically helps user.
  - Users can be lumped in user groups.
- Authorization
  - Which user has access to which resource.
  - Policy is written in a “Access Control List”.
  - Can create severe server load, as “subdirectories” exists.

Dynamic Response Handling

- Server Side Include
  - <!-#echo var =“LAST_MODIFIED”-> etc.
  - Requires every HTML docs to be parsed and scanned.
  - Use of filename extension *.shtml.
  - *.php: PHP (Personal Home Page) starts a PHP processor.
  - *.asp: Microsoft Active Server pages.
- Server Scripts
  - Not the HTML but the URL points to a script, CGI etc.
  - Separate process can be invoked.
  - Software module in same process.
  - Persistant process contacted by server.

Server Performance: Cookies & State Management



Server A sends Set-Cookie response in response headers to client C. Then all subsequent requests from client C to URL A contains the cookie.

Cookies

- In index to a particular user which can be used by server to identify an user across connection, or even user session.
- Reduces data volume as all user data can be stored in a server side database and cookies can be used as index.
- Also a threat to user privacy.
- First introduced in 1994 in Netscape 4.0. standardized in October 2000 in RFC 2965.
- Typical defaults: each one upto 4KB, maximum of 20 Cookies per server, and total of 300 per browser.

Privacy Concern from Cookies

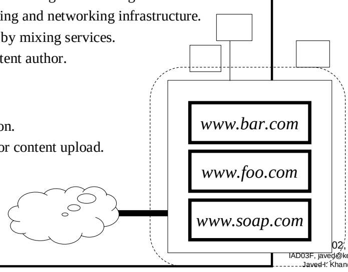
- Web-sites can identify users and start accumulating session history.
- 3-rd party sites can also collect information about used by embedded URLs (Web Bug).
- User Controls:
  - Decide if one will accept it or not.
  - Set limit on size and number.
  - Set acceptance based on site site/domain.
  - Set limit on the duration.
  - Set limit on current page sites.

Performance Enhancement

- Store in memory the popular files and save disk reading time.
  - How to maintain consistency?
    - If file changes the memory must be invalidated.
    - Check last-update time before serving.
- Store dynamically generated responses for popular queries.
  - How to maintain consistency?
    - QUIZ
- Fast Request & Processing
  - Store URL to file name mapping.
  - Catalogue access and basic file attributes (size, last update time) which are to used in response.
  - Response Header information (such as type) stays unchanged and can be pre-catalogued.

Server Farm

- Multiple Web site on a Single Machine
  - Separates content authoring from hosting.
  - Advanced computing and networking infrastructure.
  - Cost amortization by mixing services.
  - Privacy of the content author.
- Concerns:
  - IP address depletion.
  - Synchronization for content upload.
  - Virtual hosting.



Virtual Hosting

- HTTP/1.0 browsers. To get bar.html from www.foo.com may send a request "GET /bar.html HTTP/1.0" to www.foo.com.
- However, if www.virtualhost.com does not see the vanity URL, then it cannot tell which one is the intended server. Thus HTTP/1.0 requires distinct IP for each Web server.
- Solution: HTTP/1.1 introduced the "Host" header, which can specify the vanity URL.
  - GET /bar.html HTTP/1.1
  - Host: www.foo.bar
- The vanity URL can be resolved to www.virtualhosts.com's server IP. But the server by looking into vanity URL header be redirect the message to appropriate server.

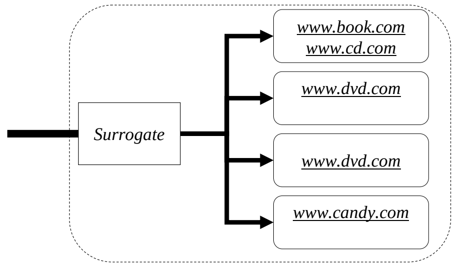
INTERNET APPLICATION & SYSTEM DESIGN

Server Replication

- Mirror Site:
  - Replicate the content in multiple servers to enable load distribution.
  - How to update and keep them synchronized?
  - How to redirect requests?
- Server Bank
  - Keep all in one logical front end server. Locally distribute the load to individual servers working in the backend.
  - How to distribute request sequences evenly?
  - Replica file system updates.
  - Access control/ cookie management.

INTERNET APPLICATION & SYSTEM DESIGN

Server Bank



- Serving high rate of requests requires multiple web-server machines to work concurrently.

INTERNET APPLICATION & SYSTEM DESIGN

Case Study: A Business Site

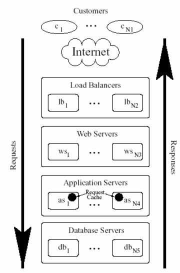


Fig. 1. Logical diagram of the system under study.

Reference: Arlitt, Krishnamurthy and Rolia, Hewlett-Packard Laboratories, Aug 2001]

INTERNET APPLICATION & SYSTEM DESIGN

Workload Characterization
(Resource Types)

Table I. Breakdown of Requests by Resource Type

Resource Type	March 2000 (%)	July 2000 (%)
Dynamic	96.58	95.73
HTML	2.58	3.67
Image	0.37	0.16
Text	0.03	0.01
Other	0.44	0.43
Total	100.00	100.00

In this table, all requests containing a parameter list(e.g., /example.html?parameter_list) are placed in the dynamic category. They can only be generated by application servers, not web servers. And, of course, not all data generated by app servers is dynamic.

INTERNET APPLICATION & SYSTEM DESIGN

Workload Characterization
(Usage Analysis)

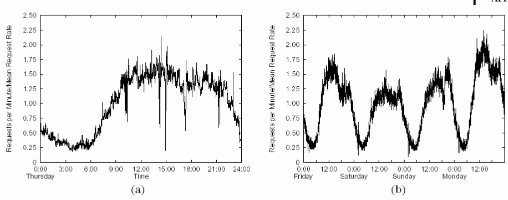


Fig. 2. Traffic volume: (a) March 2000; (b) July 2000.

INTERNET APPLICATION & ·TEM DESIGN

Classes of Requests

Types		Cacheable	Noncacheable	Search
Examples		Product descriptions Photos	Adding items to shopping carts	Search by Keywords
S or D?		Static	Dynamic	Dynamic
Personality		No	Yes	Yes
%	March	73.30	15.89	10.81
	July	70.53	17.94	11.94

Classes of Requests
(continue)

Table II. Relative Mean CPU Demands for Request Classes

Request Class Name	Ratio of CPU Demand
Cacheable (response can be cached)	1 (cache hit)
Cacheable (response can be cached)	500 (cache miss)
Noncacheable (response cannot be cached)	100
Search	40

Based on benchmark and MeasureWare.

INTERNET APPLICATION & SYSTEM DESIGN

Session Analysis

- Every session is assigned a session number.
- A single session times out when over 15 mins
- Robots consume resouces, which we don't want to see, but it's difficult to tell the difference between a human user behavior and a robot behavior.
- In order to identify a robot in our survey, we assume sessions consisting of more than 30 requests are from robots. A detail analysis found that 9% of those sessions characterized as robots performed checkout operations!

INTERNET APPLICATION & SYSTEM DESIGN

Session Analysis
(Classes of Sessions)

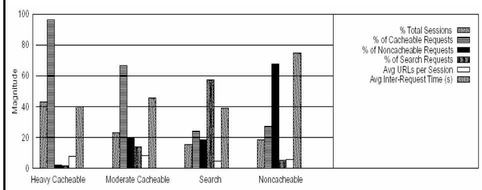


Fig. 7. Clustering based on request cache mix for users, July data ($\beta_{cv}@k = 4$ is 1.67) (Min $\beta_{cv}@k = 7$ is 1.12).

Over 40% of user sessions are characterized as heavily cacheable. 96%(avg) requests by these users are cacheable, which means they seldom do searching or check out. They are "window shoppers".

INTERNET APPLICATION & SYSTEM DESIGN

Next Class..
Cacheing..