

# Scalable Internet Servers and Load Balancing



---

**Dept. of Computer Science, University of Rochester**



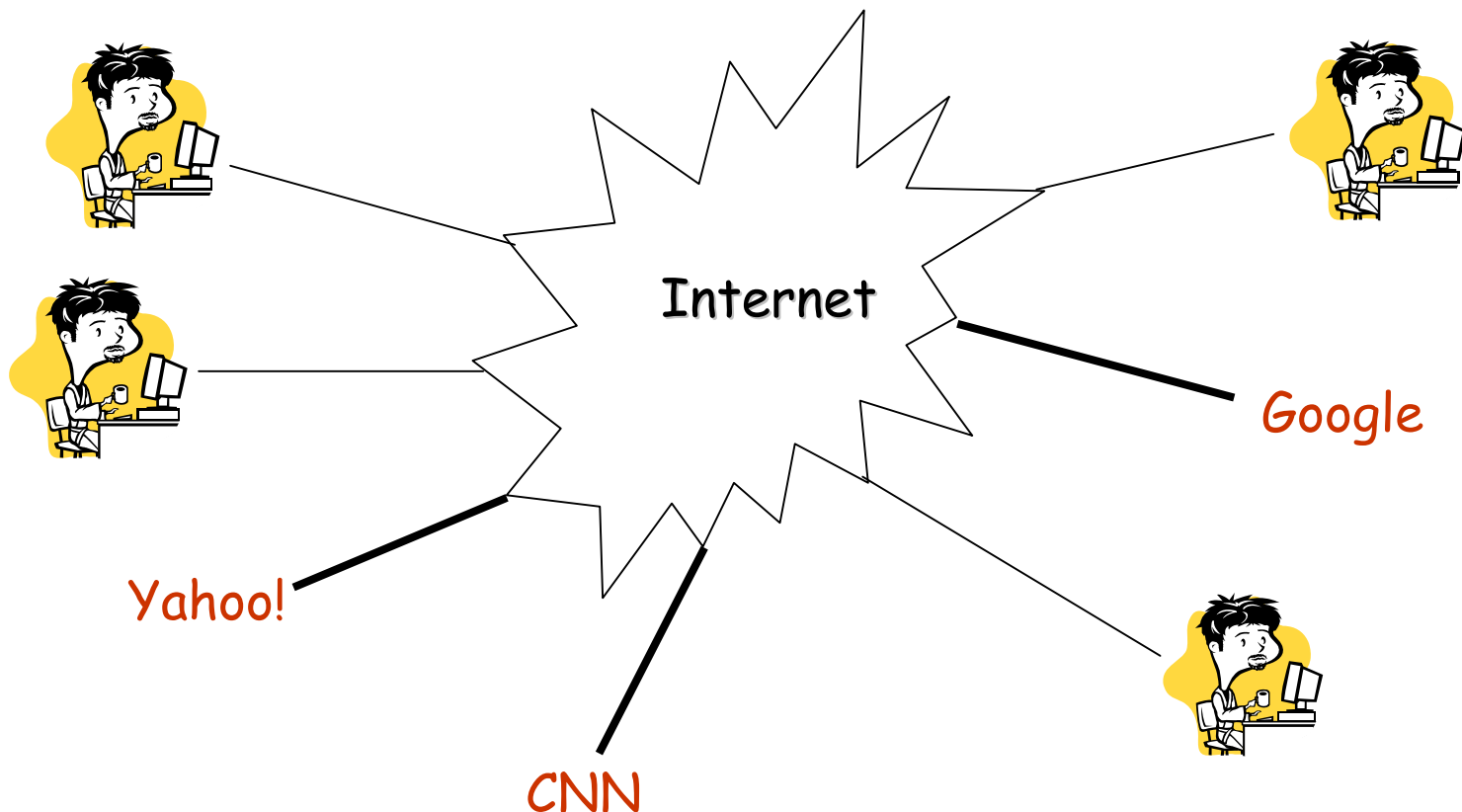
# Internet Services and Servers

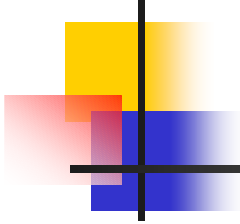
---

- Internet Services
  - Services accessible to online users through Internet.
- Services on the Internet
  - Online keyword search engine: **Google**.
  - Web email service: **Hotmail**.
  - News service: **CNN**.
  - Other portal services: **Yahoo!**, **AOL**, **MSN**.
- Scalability requirements
  - Many simultaneous user accesses; large amount of hosted data, ...
- Internet Servers
  - Computer systems that host Internet services.

# Internet Services are at the Application Layer

- Normally on the end hosts, involving no routers
- Function on transport-layer protocols TCP/UDP





# Search Engine as An Example: Step 1 - Crawling

---

- Crawling - get all these Web pages out there:
  - First retrieve some root pages;
  - Parse their content and follow hyperlinks to retrieve more pages;
  - Depth-first search or breadth-first search?

# Performance Analysis for Crawling

- The key to make it run fast - relieve the performance bottleneck.
- What are the resources involved?
  - CPU processing for TCP/HTTP protocol handling and the parsing of page content
  - local disk bandwidth
  - network bandwidth to remote web sites
- Assume average page size 10KB
  - raw processing power of a single CPU
    - one thousand fetches/second  $\Rightarrow$  around 10MB/s
  - I/O bandwidth of a single disk
    - up to 30MB/s (disk write)
  - network bandwidth from/to the Internet
    - T1 link (1.5Mbit/s); T3 (45Mbit/s)

# Search Engine as An Example: Step 2 - Indexing

- Indexing
  - crawled raw web pages are not easy to search.
  - we index them to formats that are easy to search.
- As part of indexing, we need to give each page an ID
  - using a hash function.

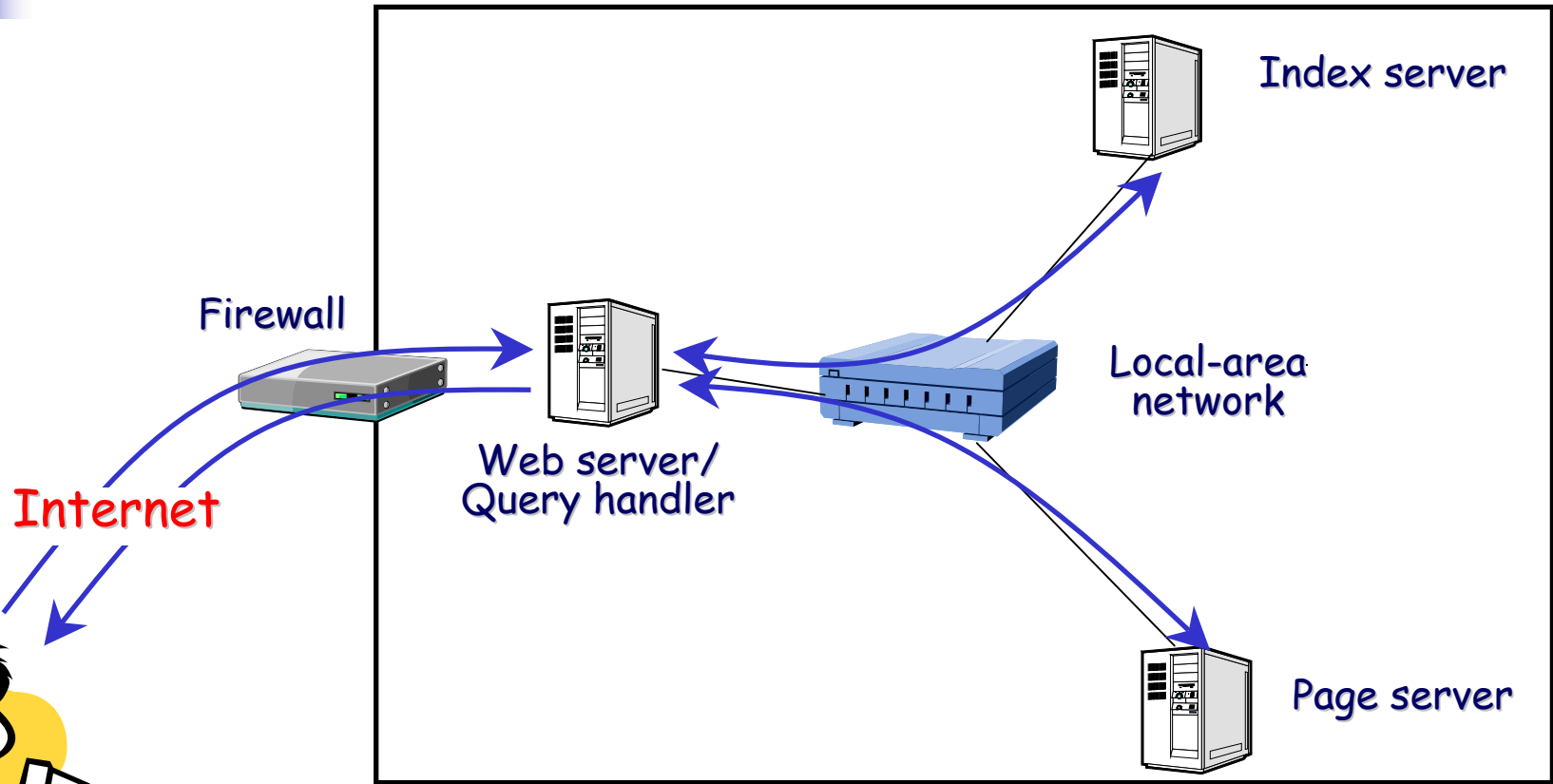
Computer:

Page #123	Page #357	... ..
-----------	-----------	--------

Networks:

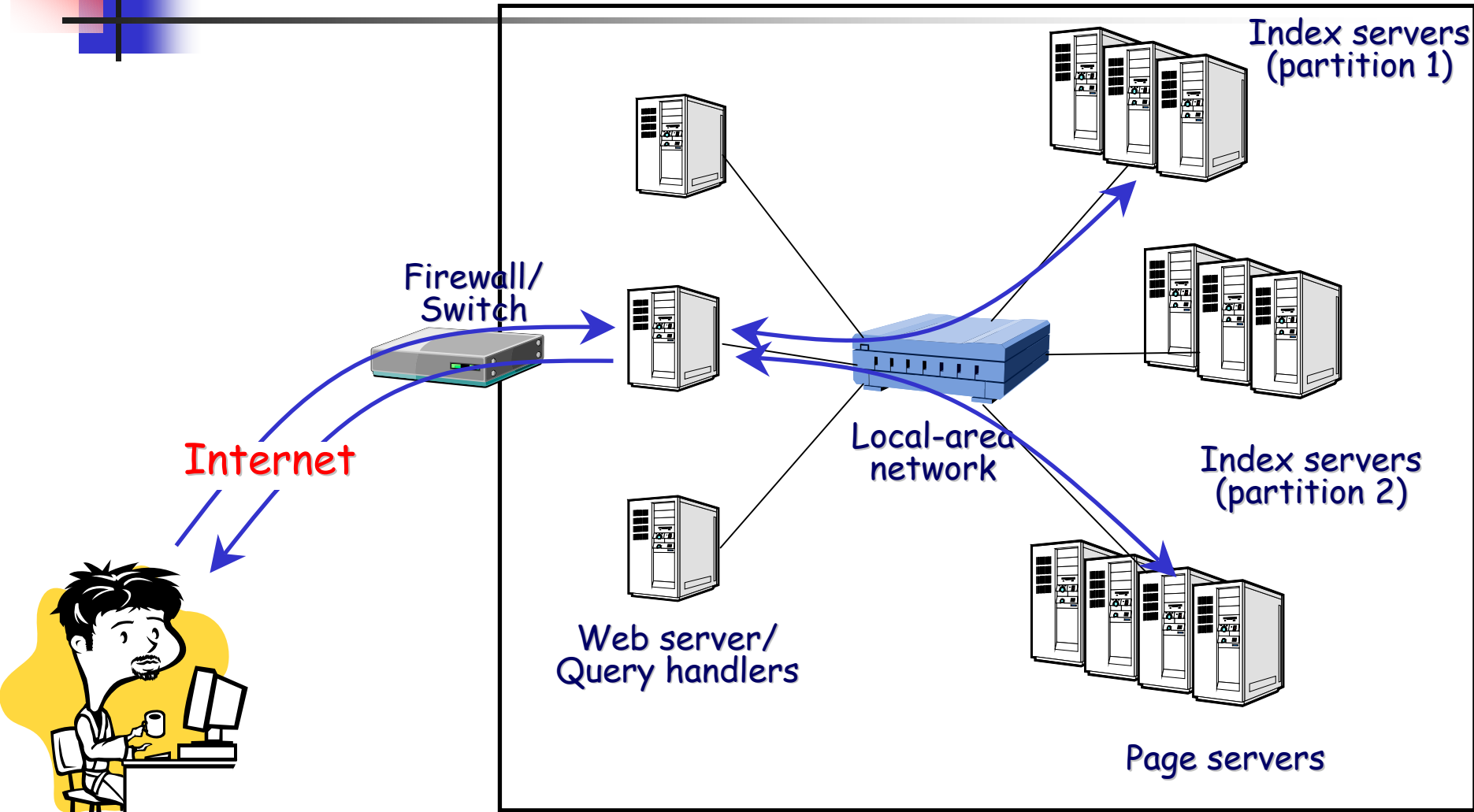
Page #124	Page #468	... ..
-----------	-----------	--------

# Search Engine as An Example: Step 3 - Online Search



**Scalability, reliability**

# Partitioning and Replication



# Load Balancing over Internet Servers

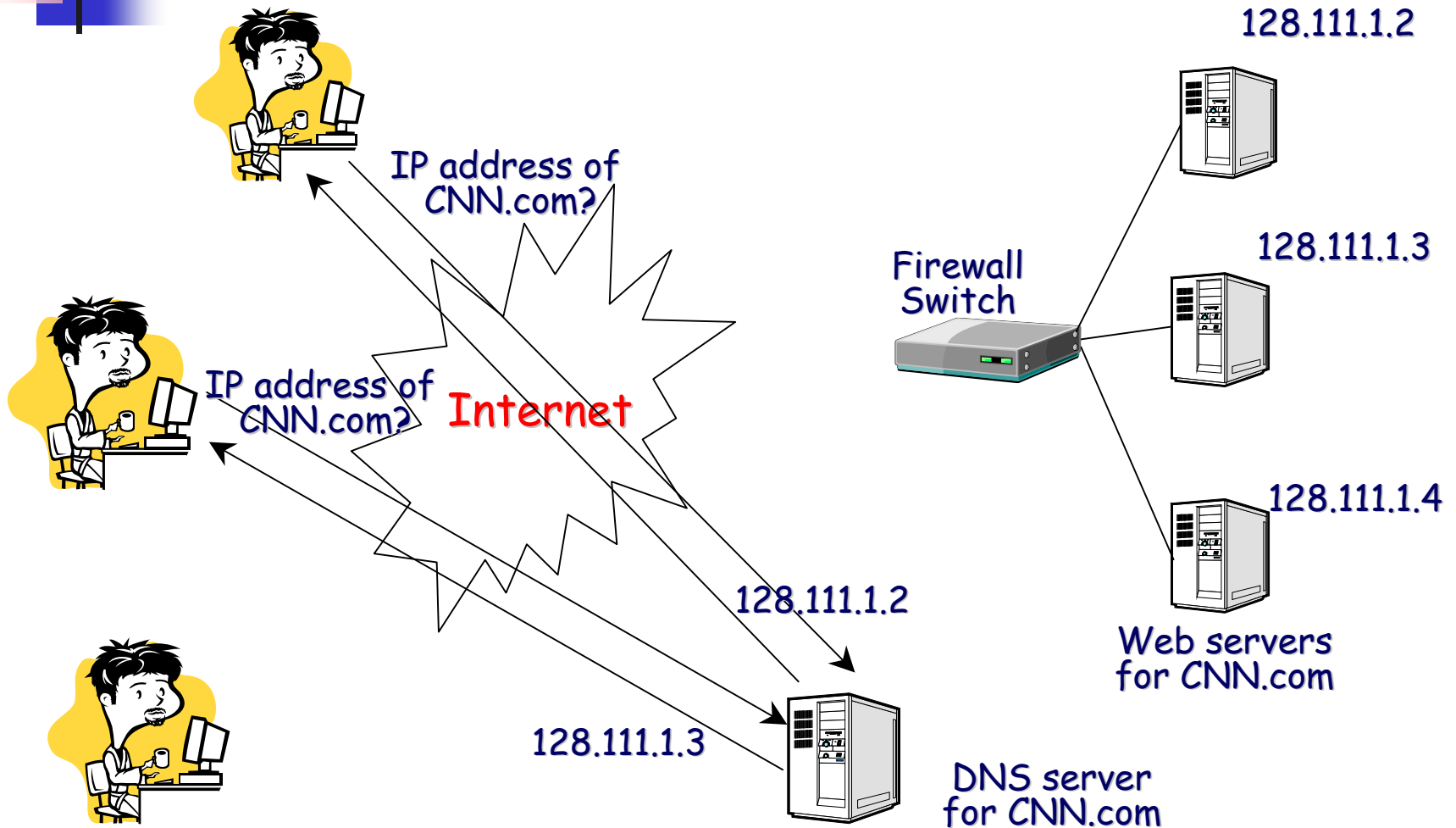


---

- Popular sites like Google or CNN receive tens or hundreds of millions of hits per day.
- A large number of replicated servers are used at these sites.
- Key question: how to balance client requests over these servers?

# Load Balancing on Internet Servers

## Technique 1 - DNS Rotation





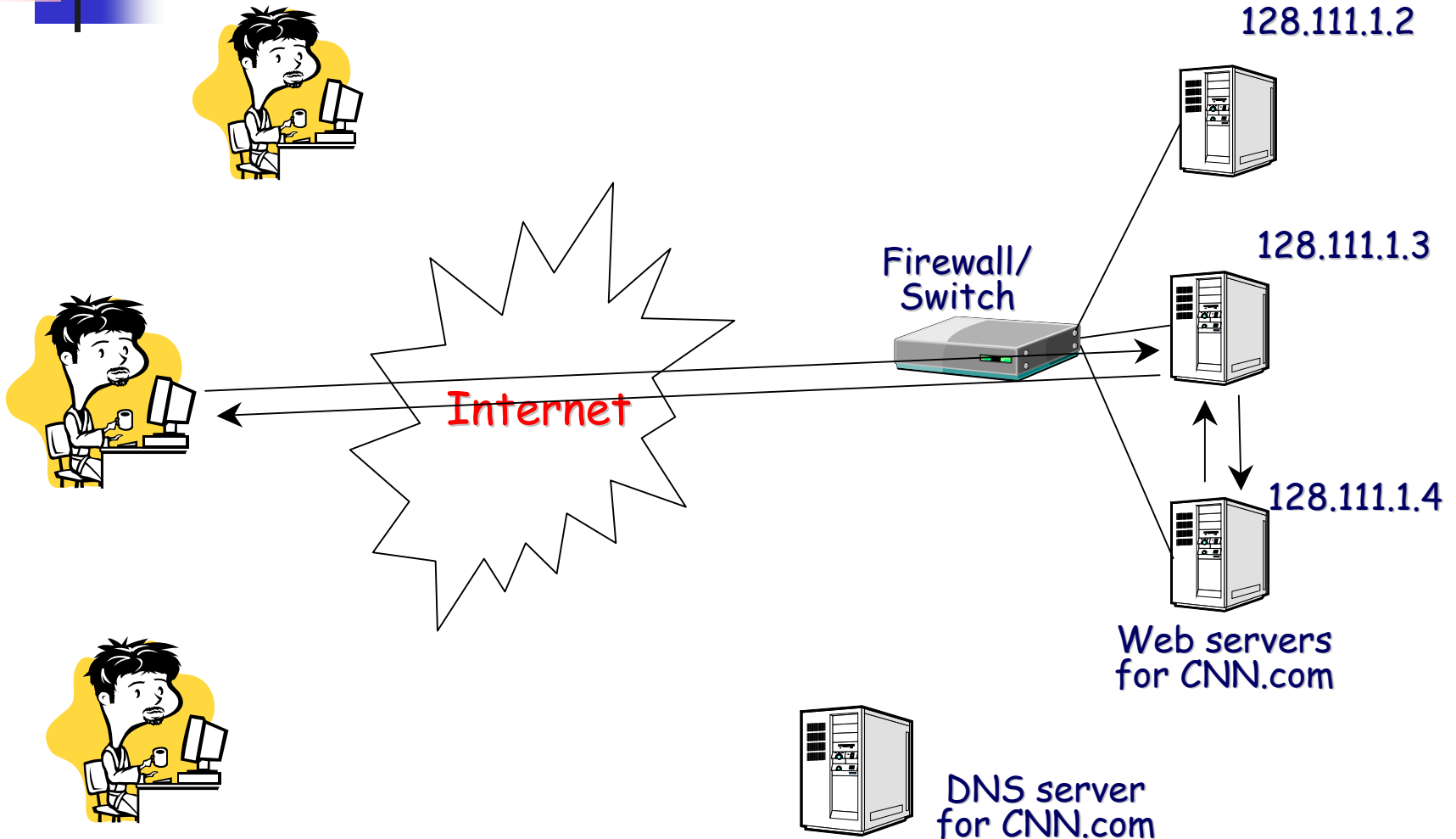
# Discussions on DNS Rotation

---

- Advantages
  - Require almost no change on the existing Internet architecture
- Problems
  - DNS Caching
  - Rigid load balancing policy
    - can't balance based on runtime load changes
    - slow or no adjustment in response to failures

# Load Balancing on Internet Servers

## Technique 2 - Cooperative Offloading



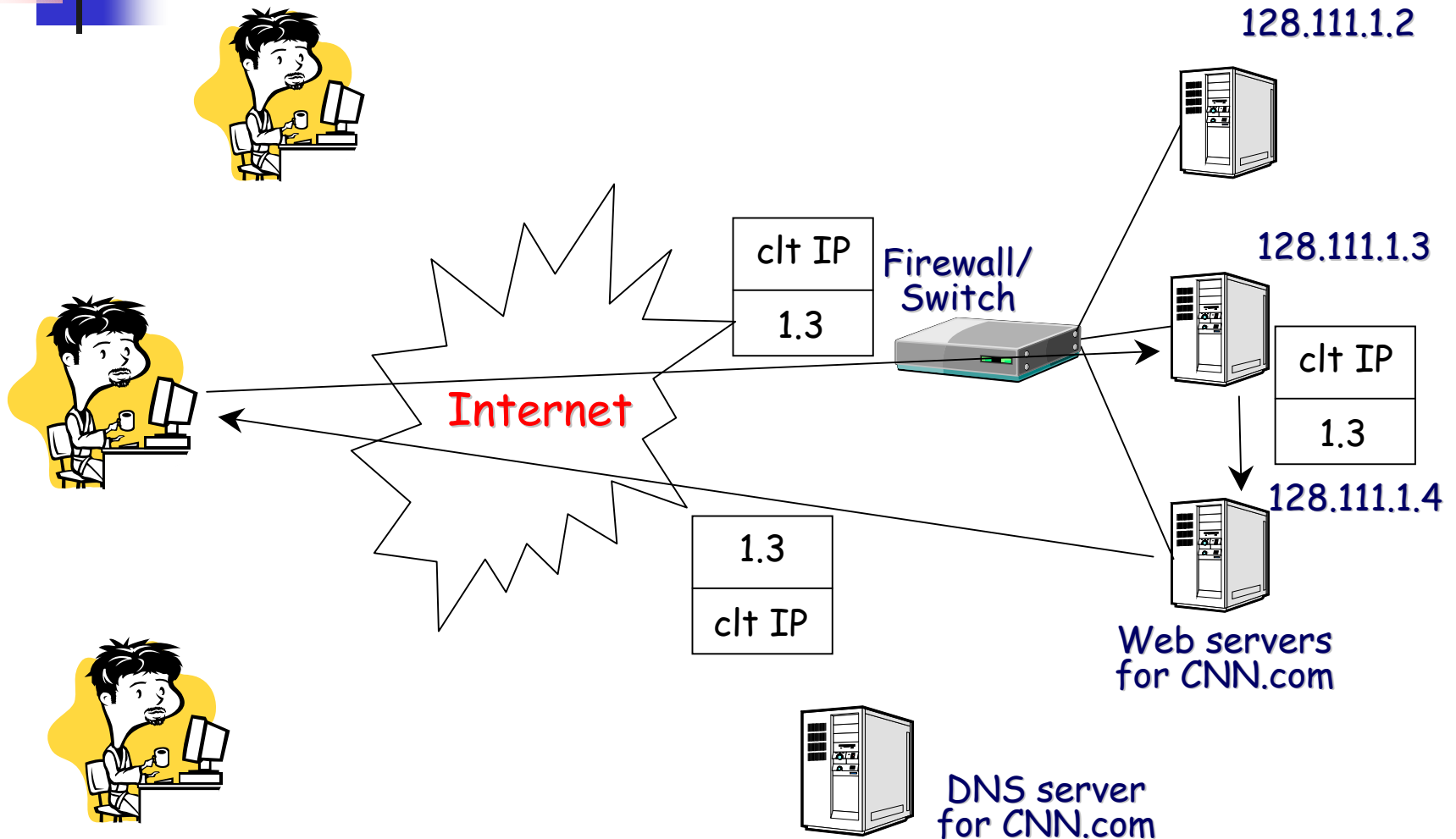
# Discussions on Cooperative Offloading



---

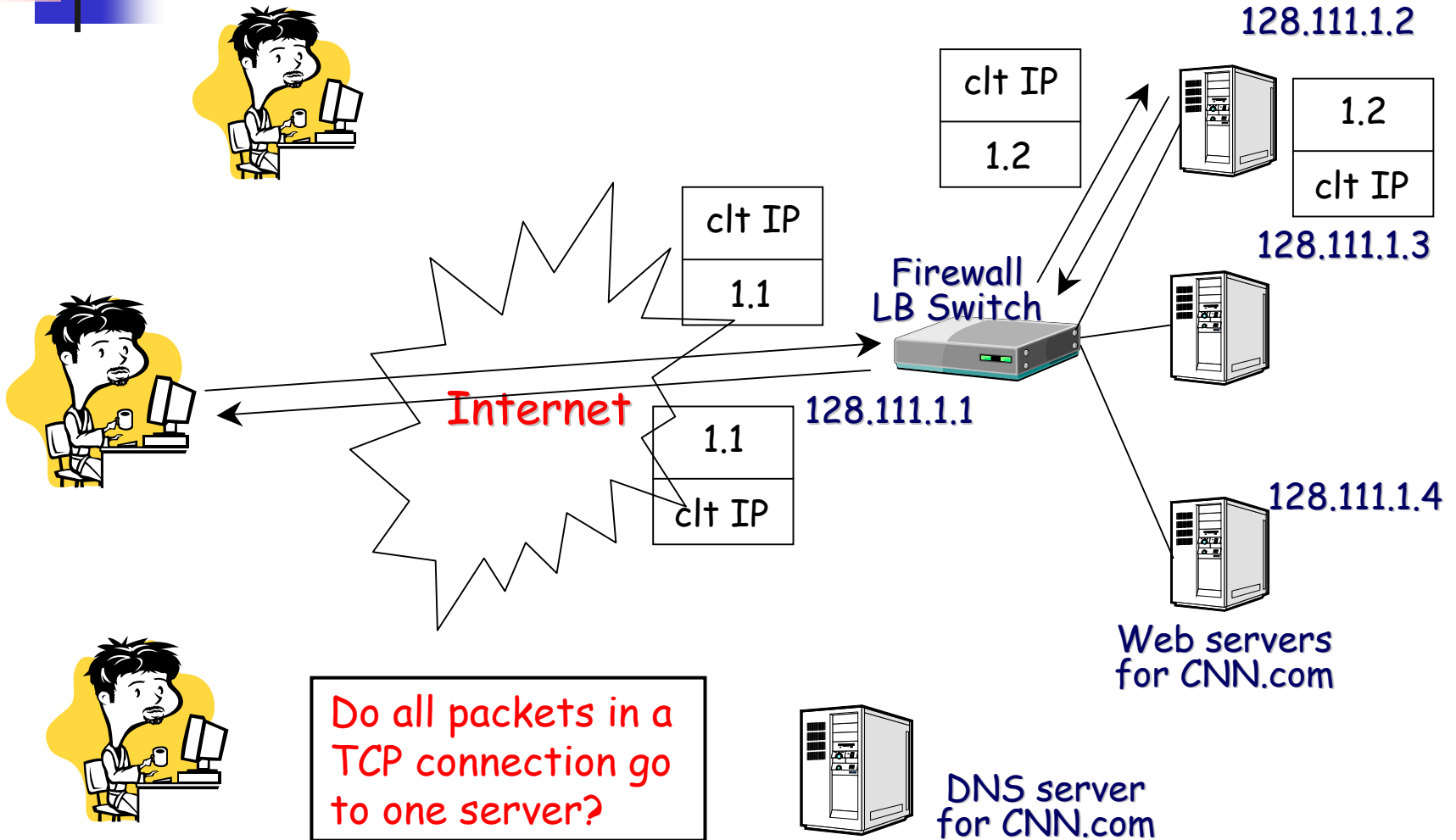
- Can be combined with the DNS rotation.
- Advantages:
  - More flexible policy is possible
  - Be more responsive to runtime workload and server failures (to a certain degree)
- Problems
  - Need a lot more software
  - Longer delay

# Cooperative Offloading with TCP Handoff [Pai et al. ASPLOS1998]



# Load Balancing on Internet Servers

## Technique 3 - Load Balancing Switch





# More About Load Balancing Switch

---

How deep do we need to look into the network protocol stack?

- Network layer (IP)?
- Transport layer (TCP/UDP)?
- Application layer?

Load balancing policies in LB switches (Goal: transparency, plug-and-play)

- Simple rotation
- Least number of active requests
- Shortest response time



# Summary

---

- Scalable Internet servers
  - partitioning
  - replication
- Load balancing on Internet servers
  - DNS rotation
  - cooperative offloading (w. TCP handoff)
  - LB switches
- For each technique, changes required on the components:
  - DNS server??
  - Web server??
  - client??
  - switch??