Web Hosting

Michael Tsai
2017/06/03

Web Hosting Basics

- A daemon (server) listens for connection on TCP port 80
- Accept request for documents
- Transmits them to the requesting user's browser

URL

- URI = {URL, URN}
- URI: Unified Resource Identifier
  URL: Unified Resource Locator
  URN: Unified Resource Name
  (e.g., urn:isbn:0-13-020601-6)
- URL: {protocol/app., hostname, [port, directory, filename]}

URL Examples

Proto	What it does	Example
file	accesses a local file	file:///etc/syslog.conf
ftp	accesses a remote file via FTP	ftp://ftp.admin.com/adduser.tar.gz
http	accesses a remote file via HTTP	http://admin.com/index.html
https	accesses a remote file via HTTP/SSL	https://admin.com/order.shtml
ldap	accesses LDAP directory services	ldap://ldap.bigfoot.com:389/cn=Herb
mailto	sends email to a designated address	mailto:linux@book.admin.com

How HTTP works

- Stateless client/server protocol
- A client
  asks the server for the **contents** of
  a specific **URL**
- The server responds with the data (or err. msg.)
- Try it: telnet to port 80

人肉Browser

- telnet to www.csie.ntu.edu.tw port 80 (http default TCP port)
- Type the following:
  GET / HTTP/1.1
  Host: www.csie.ntu.edu.tw
  (hit <enter> twice)
- What do you get?
- Try a nonexistent URL. What do you get?

Content Generation

- Dynamic content is better

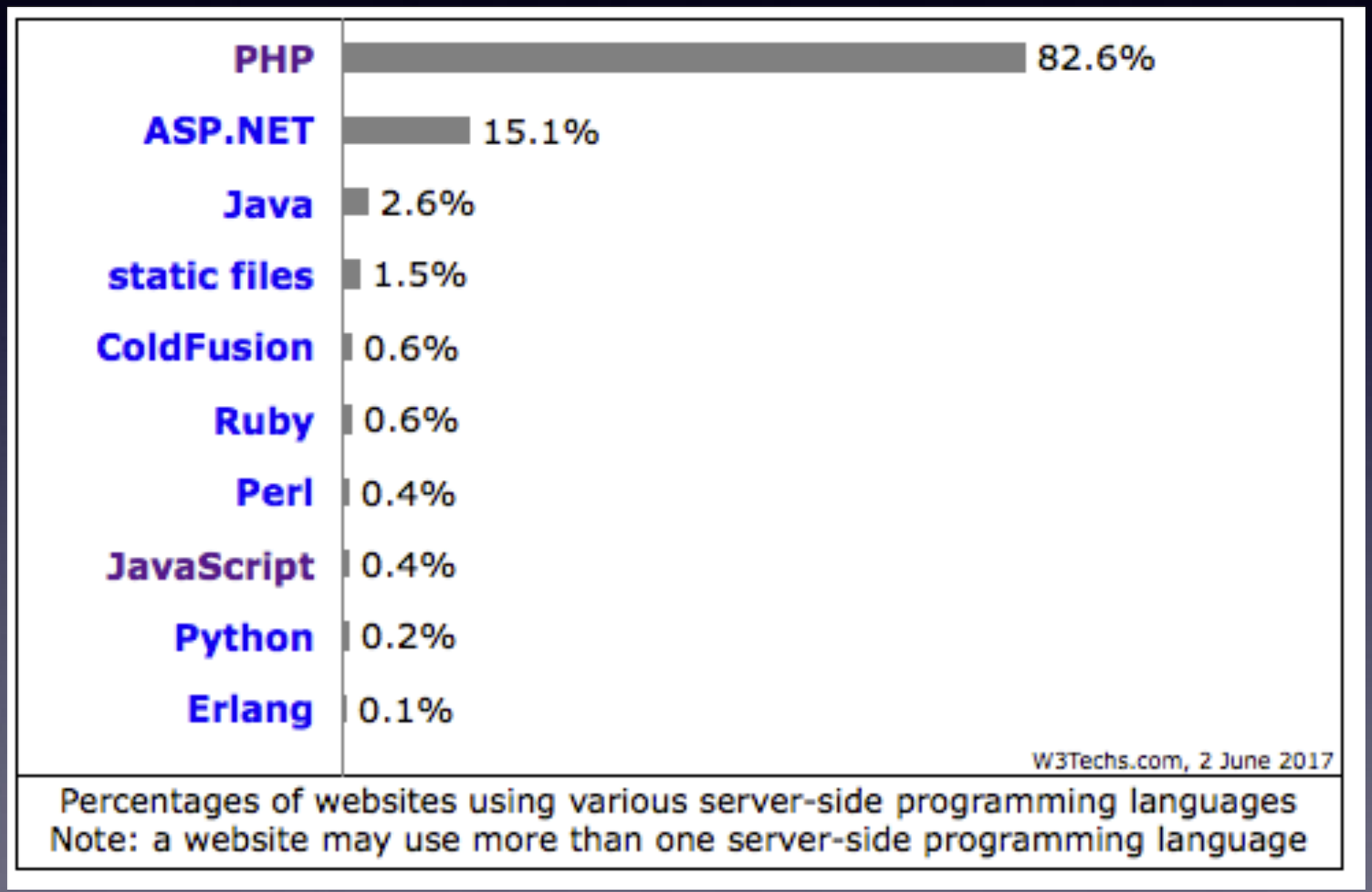
1. CGI (Common Gateway Interface):
   Allow external program to interact with the web server

2. FastCGI: Allow external program to continue running to server multiple requests

3. Embedded interpreters:
   (e.g., Perl, PHP, Python, Ruby on Rails)
   Executing external script within the server (.php, .pl)
   e.g., LAMP: linux + apache + mysql + php/perl/python

4. Application servers:
   Entire, full-fledge, platform for web
   (e.g., Tomcat, WebSphere, WebLogic, Jetty)

Market Share:
Server-Side Programming Language



Security!

- Bottom line:
  you allow **the entire world** to execute a script on your server (access to files, networks, and more!)

- Need to make sure that the script is secure
  (as much as other network-accessible program)

- Read:
  OWASP Top 10 Application Security Risks 2017
  https://www.owasp.org/index.php/Top_10_2017-Top_10

Load Balancing

- Many factors affecting the maximum load a server can handle:
  - Hardware architecture
  - Operating system
  - System tuning
  - Sites being served
    (static vs dynamic (database))
  - And, in addition, network bandwidth
- Stress testing - CPU, I/O, or network-bound? (usually not network)

Create Scalability

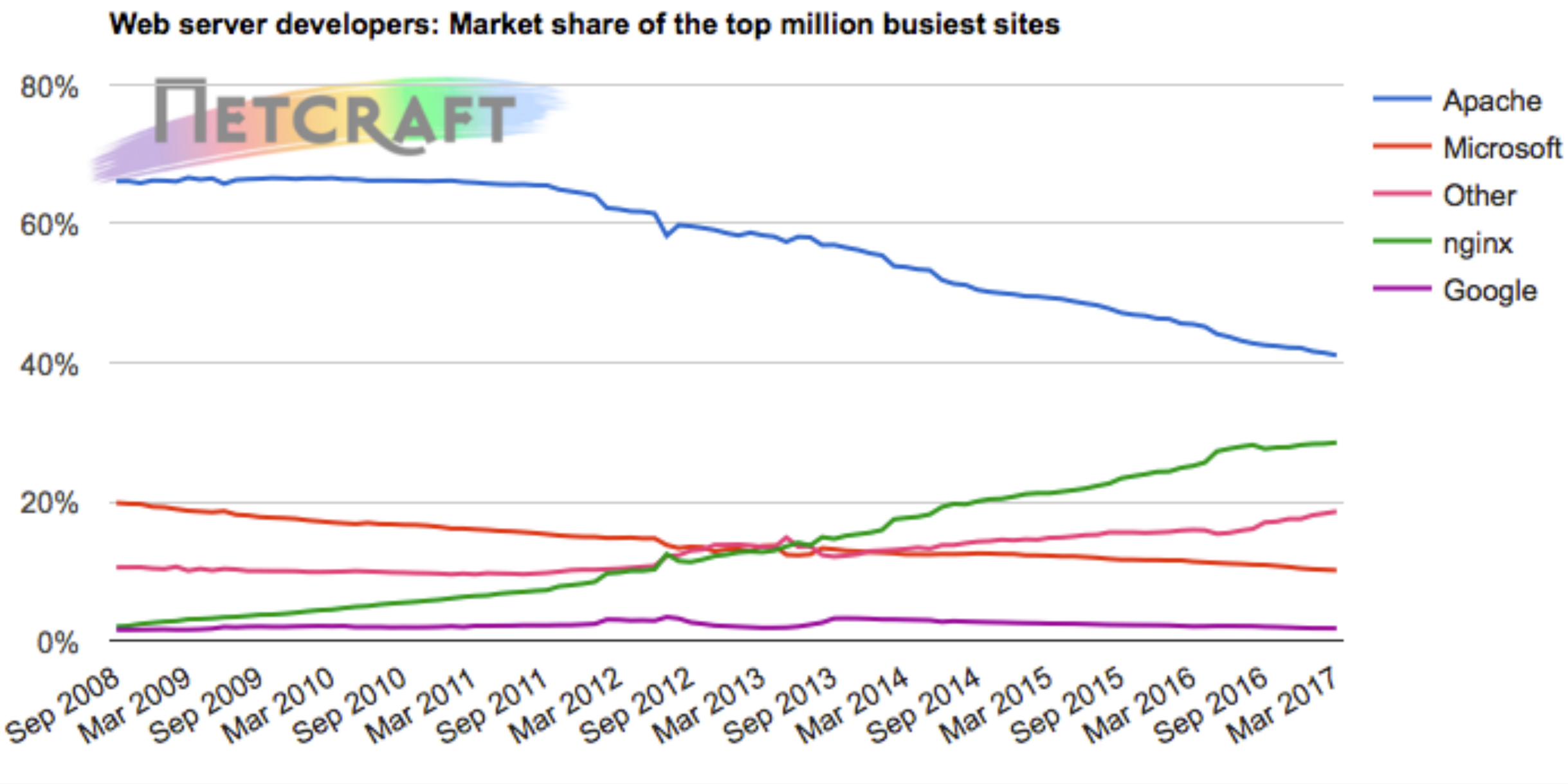
1. Round robin DNS (we've talked about it)
   Note that the order in the DNS record is **irrelevant**.
   (Think about its disadvantage)

2. Hardware solution (e.g., Big-IP from F5)
   Takes response time of individual servers into account

3. Software solution (e.g., Linux Virtual Server, proxy load balancing in Apache)

Scaling Beyond Limits

- Cloud computing (e.g., Amazon Web Services)
- Co-location hosting (like us, or some NTU services)
- Content Distribution Networks (e.g., akamai, limelight, edgecast)
  - Putting static content close to users
  - Try: https://www.cdnplanet.com/tools/cdnfinder/
  - Pick a content-rich website (such as a news website) and see if it uses CDN



source: https://news.netcraft.com/archives/2017/03/24/march-2017-web-server-survey.html

Apache

- Web server with the largest market share (53.8% of top M busiest sites, 2014/04)
- Runner-ups:
  Microsoft & nginx take 17.8% and 12.4%
- Began in 1995
- First web server software to serve more than 100M sites (in 2009)
- Versatile

Nginx

- (pronounced “engine x”)
- Created in 2004
- Used by 57.0% of the top 10,000 websites. (W3Techs)
- Written with an explicit goal to outperform Apache (less memory, 4x more requests per second)
    - Less flexibility