Network Programming Part 2

MCIT 595

Naming and Addressing

- IP address
  - Identifies a single host
  - 32 bits (IPv4), 128 bits (IPv6)
  - Written as dotted octets (8 bits)
    - e.g., 0x0a000001 is 10.0.0.1
- Host name
  - Identifies a single host
  - Variable length string
  - Maps to one or more IP address
    - e.g., www.upenn.edu
  - gethostbyname() translates name to IP address

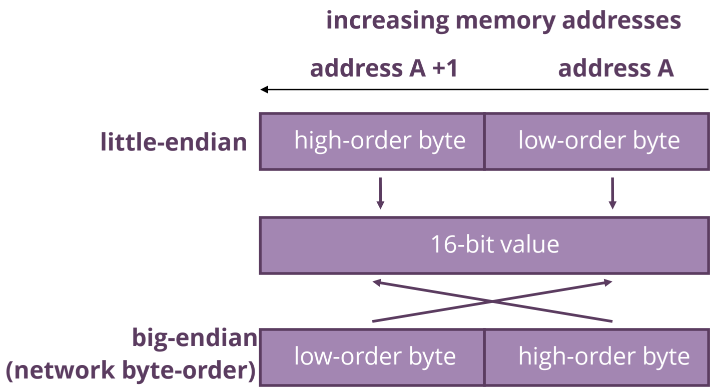
Knowing What Port Number To Use

- Popular applications have well-known ports
  - E.g., port 80 for Web and port 25 for e-mail
  - See http://www.iana.org/assignments/port-numbers
- Well-known vs. ephemeral ports
  - Network service has a well-known port (e.g., port 80)
    - Between 0 and 1023
  - Unused ephemeral (i.e., temporary) port
    - Between 1024 and 65535

Byte Ordering

- Hosts differ in how they store data
  - E.g., four-byte number (byte3, byte2, byte1, byte0)
- Little endian (“little end comes first”) ← Intel PCs!!!
  - Low-order byte stored at the lowest memory location
  - Byte0, byte1, byte2, byte3
- Big endian (“big end comes first”)
  - High-order byte stored at lowest memory location
  - Byte3, byte2, byte1, byte 0

Byte Ordering



increasing memory addresses

address A +1

address A

little-endian

high-order byte

low-order byte

16-bit value

big-endian
(network byte-order)

low-order byte

high-order byte

Byte Ordering Solution

```
uint16_t htons(uint16_t host16bitvalue);
uint32_t htonl(uint32_t host32bitvalue);
uint16_t ntohs(uint16_t net16bitvalue);
uint32_t ntohl(uint32_t net32bitvalue);
```

- Use for all numbers (int, short) to be sent across network
  - Typically used for port numbers