Computer Communication Midterm

By Natthapach Anuwattananon

Chapter 1 : Introduction

1.1 What's the Internet: "nuts and bolts" view

Internet
- เครือข่ายที่เชื่อมต่อเครือข่ายเข้าด้วยกัน (network of networks)
- การเชื่อมต่อ network ทั่วโลกเข้าด้วยกัน

ISP (Internet Service Provider)

mobile network
- จะมีเสาสัญญาณส่งสัญญาณไปที่ smartphone/notebook
- เป็น cellular network

institutional network
- เครือข่ายในองค์กร
- ex. เครือข่ายในมหาวิทยาลัยเกษตร (KU win เป็นแค่ส่วนหนึ่ง)

host
- เครื่องปลายทางที่ run network application

communication link
- ตัวกลางที่ทำหน้าที่ในการส่งข้อมูลใน network
- ex. fiber, copper, radio, satellite
- มีอัตราการส่งข้อมูล เรียกว่า bandwidth

packets
- ชิ้นส่วนของ data

packets switch
- จุดส่ง packet ไปยังยังปลายทาง
- ex. router, swirch

protocol
- กำหนด format, order ขอข้อมูลที่รับ/ส่ง
- action takens : การกระทำเมื่อได้รับข้อมูล

1.2 Network edge

network edge
- host แบ่งเป็น client และ server
- ส่วนมาก sever จะเป็น data center

access network, physical media
- มีทั้งแบบ wire (มีสาย) และแบบ wireless (ไร้สาย)

network core
- จุดที่ router เชื่อต่อกัน

host connect to edge router
- residential access เชื่อมต่อกันภายในบ้าน
- institutional access เชื่อมต่อกันภายในองค์กร
- mobile access เชื่อมต่อกันผ่านมือถือ

bandwidth in link
- shared มีการแบ่ง bandwidth, ยิ่งคนใช้งานใน network เยอะ bandwidth ยิ่งต่ำ
- dedicated ไม่มีการแบ่ง bandwidth

Access net : Digital Subscriber Line (DSL)
- มี spliter แยกสัญญาณ (voice / data) ที่เข้ามาไปยังอุปกรณ์
    - ถ้าเป็น voice เข้าโทรศัพท์, data เข้า dsl modem
- มี DSL access multiplexer (DSLAM) แยกสัญญาณที่ส่งออก
    - ถ้าเป็น voice เข้า telephone network, data เข้า ISP

By Natthapach Anuwattananon

- upstream < 2.5 Mbps (typically < 1 Mbps)
- downstream < 24 Mbps (typically < 10 Mbps)
- สายที่เข้าบ้านเป็นแบบ dedicated

Access net : Cable network
- ส่งข้อมูลมากับสาย cable TV
- สายที่ใช้เรียกว่า Coaxial Cable
- ใช้วิธีการส่งที่เรียกว่า frequency division multiplexing
    - แบ่งช่องสัญญาณตามความถี่ (channel)
    - channel 1-6 : video
    - channel 7-8 : data
    - channel 9 : control
- สายที่เข้าบ้านเป็นแบบ shared
- มี cable modem termination system (CMTS)
    - ทำหน้าที่หยุดการสะท้อนกลับของสัญญาณที่ส่งออกไปยัง ISP
- สาย hybrid fiber coax (HFC)
    - upstream : 2 Mbps
    - downstream : 30 Mbps

Enterprise acess networks (Ethernet)
- มี institutional router คอยรับสัญญาณมาจาก ISP แล้วกระจายต่อให้ Ethernet switch
- end system ต่างๆ จะ connect เข้าหา Ethernet switch
- link จาก ethernet switch ไปยัง end system จะเป็น dedicate

Wireless access networks
- เป็นการส่งสัญญาณไร้สายแบบ shared
- จุดส่งสัญญาณเรียกว่า access point

Wireless LANs
- ระยะประมาณ 10 m.
- desktop ใช้ระบบ 802.11 b/g/n : 11/54/450 Mbps
- mobile ใช้ระบบ 802.11 a/c

Wide-area wireless access
- ใช้ระบบ cellular
- ระยะประมาณ 10 km
- bandwidth ~ 1 - 10 Mbps

Host : sends packets of data
- เวลาในการส่ง packets = ขนาด packet (bits) / อัตราการส่งของ link (bits/sec)

Physical media

physical link
- เส้นทางจาก transmitter (ผู้ส่ง) ไปยัง receiver (ผู้รับ)

guided media
- ส่งข้อมูลผ่านทางของแข็ง
- ex. copper, fiber

unguided media
- ส่งข้อมูลผ่านทาง
- ex. radio

twisted pair (TP)
- เป็นสายที่เกิดจากสายเล็กๆ บิดกันเป็นเกลียว
- ถ้าใช้ในโทรสัพท์จะมี 6 เส้น, ใช้ใน LAN มี 8 เส้น

By Natthapach Anuwattananon

coaxial cable
-
fiber optic cable
- ส่วนมากใช้ใน back bone ของระบบ เพราะแตกหักง่าย
- มีความเร็วสูงในการส่งข้อมูลแบบ point-to-point
- มีอัตรา error ที่ต่ำ
  - ไม่ถูกรบกวนจากสนามแม่เหล็ก
  - ไม่มีการแผ่วเบาลงของสัญญาณ

radio link
- เป็นการส่งข้อมูลผ่านคลื่นแม่เหล็กไฟฟ้า
- มีข้อเสียคือ
  - มีการสะท้อนของสัญญาณ
  - สามารถบังสัญญาณได้
  - อาจเกิดการชนกันของสัญญาณในคลื่นความถี่เดียวกัน

terrestrial microwave
- 45 Mbps

LAN
- 11 Mbps, 54 Mbps
- e.g., WiFi

wide-area
- 3G ~ few Mbps
- e.g., cellular

satelite
- Kbps - 45 Mbps
- ใช้เวลา 270 msec สำหรับการไปกลับ (end-to-end)

1.3 network core

The network core
- เครือข่ายของ router ที่เชื่อมต่อกัน
- packet-switching
  - แตก message ที่จะส่ง ออกเป็น package
  - ส่ง packet จาก router นึงไปอีก router นึง ตาเส้นทาง
  - packet จะถูกส่งเต็มความจุของ link

Packet-switching

store-and-forword
- การส่ง packets จะส่งไปเก็บไว้ใน buffer ของ router ก่อน
- เมื่อ packets มาครบ จึงจะลงไปยัง router ต่อไป
- การส่งแต่ล่ะครั้งจะเรียกว่า 1 hop
- delay ในการส่งแต่ล่ะ hop = L/R

queueing delay, loss
- packet ที่ส่งมาและเก็บไว้ใน buffer จะต้องเข้า queue เพื่อส่งต่อ
- ถ้าหาก packet เข้ามาตอนที่ buffer เต็ม จะถูกโยนทิ้งไป ทำให้ข้อมูลหาย
- ถ้าข้อมูลนั้นจำเป็นต้นทางจะต้องส่งมาใหม่

Two key network-core function

routing
- มี routing algorithm ในการคำนวณเส้นทางที่สั้นที่สุด
- จะได้ local forwarding table

By Natthapach Anuwattananon

forwarding

- ส่ง packet ที่เข้ามา ไปต่อตาม local forwarding table

Circuit switching

- มีการจองเส้นทางตลอดเส้นทาง
- ได้ bandwidth เต็มที่ เพราะไม่มีการ shared
- ใช้ในระบบโทรศัพท์แบบดั้งเดิม

Frequency Division Multiplexng (FDM)

- แบ่งช่องสัญญาณตามความถี่
- มีข้อจำกัดเรื่องการแบ่ง frequency ทำได้จำกัด

Time Division Multiplexing (TDM)

- แบ่งช่องสัญญาณตามเวลา

Packet switching VS Circuit switching

- ถ้ามี 1 Mbps link และให้ user คนล่ะ 100 kbps เมื่ออยู่ในสถานะ active
  - circuit switching : 10 users
  - packets switching : 35 users
- packet switching เหมาะกับ bursty data
  - bursty data คือรูปแบบการส่งข้อมูลที่มีปริมาณการส่งสูงขึ้นและต่ำลงเป็นช่วงๆ
  - e.g., voice, web
  - ไม่ต้องเสียเวลา set up (จองเส้นทางแบบ circuit switching)
- packet switching มีโอกาสเกิดการติดขัดของเส้นทาง และข้อมูลศูนย์หาย
  - มี protocol เพื่อแก้ปัญหานี้

Internet structure : network of networks

- โครงสร้างของ internet เกิดจากมีผู้ลงทุนโยงสายข้ามประเทศต่างๆ เป็น network ของตัวเอง ที่มีขนาดใหญ่ network นี้เรียกว่า Tier 1 ISP (global ISP)
- Tier 1 นั้นมีหลายบริษัท และจะมา peer กันที่ IXP (Internet Exchange Poin) เกิดเป็นแกนกลางของ internet
- ลำดับชั้นต่อมาจะเป็น Tier 2 ISP (regional ISP) ที่มา peer กับ Tier 1
- ลำดับชั้นต่อมาอีกจะเป็น Tier 3 ISP (local ISP) ที่มา peer กับ Tier 3
- end system จะเชื่อมต่อกับ local ISP (access ISP) เพื่อทำการใช้ internet
- นอกจากนี้ยังมี content provider network ที่เก็บข้อมูลต่างๆไว้ และเชื่อมต่อ data center ของพวกเค้าเข้ากับ Tier 1 หรือ Tier 2 เพื่อให้ end user เข้าถึง content ของพวกเค้าเร็วขึ้น
- โครงข่ายของ Tier 1 นั้น แต่ละจุดจะเชื่อมต่อกันแบบ POP (point of presence)

1.4 delay, loss, throughput in networks

Four source of packet delay

$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$

- d_{proc} : nodal process
  - ใช้เวลาในการ check error
  - ใช้เวลาในการหาเส้นทาง
  - น้อยมากๆ < msec
- d_{queue} : queueing delay
  - ใช้เวลาในการรอ queue ออกจาก router
  - ขึ้นอยู่กับความหนาแน่นใน router
- d_{trans} : transmission delay
  - เวลาในการส่งข้อมูล 1 packet
  - $d_{trans} = L_{(bits)} / R_{(bps)}$
  - ใช้เวลามากสุด

By Natthapach Anuwattananon

- d_{prop} : propagation delay
    - เวลาในการส่งข้อมูล 1 บิท จากต้นทางไปปลายทาง
    - $d_{prop} = d / s$
    - d : ความยาวของ link, s : propogation speed (~$2x10^8$ m/sec)

Queueing delay

- R : link bandwidth (bps)
- L : packet lenght (bits) ขนาด
- a : average packet arrival rate (bits/s) อัตราส่งสำเร็จ

- La/R ~ 0 delay น้อย
- La/R → 1 delay มาก
- La/R > 1 delay มากเป็นอนันต์ (คิวยาวมากกกกก)

“Real” Internet delays and routes

- traceroute
    - ใช้ในการวัด delay จาก source ไปยัง router แต่ละตัว
    - ทำงานโดยการส่งข้อมูลไป 3 probe แล้วรอ router ตอบแต่ละ probe กลับมา

Packet loss

- เหตุการณ์ที่ buffer ใน router เต็มแล้ว แต่ยังมีข้อมูลส่งมาอยู่ ข้อมูลนั้นจะถูกโยนทิ้ง
- ปัญหา packet loss จะแก้โดย protocol ชั้นล่างๆ

Throughput

- throughput คือ อัตราในการส่งข้อมูลจาก sender ไปยัง receiver
    - instantaneous : อัตรา ณ ขณะใดขณะหนึ่ง
    - average : อัตราเฉลี่ย
- ปัญหาคอขวด : sender และ receiver มี bandwidth ที่ไม่เท่ากัน
    - throughput จะเท่ากับ bandwidth ที่น้อยที่สุด
- ถ้ามีการส่งผ่าน internet จะต้องนำ bandwidth ของ internet มาคิดหา throughput ด้วย

1.5 Protocol layers, service models

layer

- แต่ละ layer จะทำงานให้กับ layer ข้างบนของตัวเอง
- การเปลี่ยนการทำงานของ layer จะต้องไม่ส่งผลต่อการทำงานของ layer อื่น
- แต่ละ layer ก็จะมีหลาย protocol ต้องเลือกใช้ให้เหมาะกับงาน

Internet protocol stack

- applications layer
    - ใช้ติดต่อระหว่าง user กับ network application
    - ใช้ทำงานร่วมกับ network application
    - FTP (File Transfer Protocol) ใช้ในการส่งไฟล์
    - SMTP (Simple Mail Transfer Protocol) ใช้ในการส่งเมล์
    - HTTP (Hyper Text Transfer Protocol) ใช้ในการส่งหน้าเว็บ
- transport layer
    - เชื่อมต่อกับapplications layer ในการใช้ network sevice (app.) จากต้นทาง-ปลายทาง โดยใช้ Port Number
    - ใช้ส่งข้อมูลระหว่าง process ใน OS
    - TCP
    - UDP
- network layer

- มีหน้าที่ส่งข้อมูลข้ามเครือข่าย โดยผ่าน IP และใช้ switch, router
- ใช้ในการค้นหาเส้นทาง
- IP

- link layer
  - กำหนดรูปแบบการส่งข้อมูล โดยใช้ mac address ระบุต้นทาง-ปลายทาง
  - รวมถึงจัดการกับerrorในการรับส่งข้อมูล
  - ส่งข้อมูลระหว่าง network element ใกล้เคียง
  - Ethernet, WiFi
- physical layer
  - กำหนดคุณสมบัติทางกายภาพของฮาร์ดแวร์ที่ใช้เชื่อมต่อระหว่างต้นทาง-ปลายทาง
  - เช่น สายที่ใช้, ความเร็ว, รูปร่างของสัญญาณ
  - ควบคุมการส่ง data ผ่าน connect link

ISO/OSI reference model

- เพิ่มอีก 2 layer แยกออกมาจาก application layer
- presentation layer
  - ใช้ในการแปลงข้อมูลเพื่อส่งออก
  - ใช้ในการแปลข้อมูลที่ส่งเข้ามา
- session layer
  - ใช้ในการกำหนด checkpoint และ recovery data
  - 1 connection = 1 session
  - ควบคุมการเชื่อมต่อ session เพื่อติดต่อจากต้นทางไปปลายทาง

Encapsulation

- การส่งข้อมูลออกจาก source
  - การส่งข้อมูลจะส่งจาก application layer
  - แต่ละ layer จะนำ header data ของตัวเอง เพิ่มเข้าไปในข้อมูล
- การรับข้อมูลและส่งต่อใน switch
  - physicalจะรับข้อมูลมาทีละbit เมื่อได้ครบทั้งframeแล้วก็จะส่งต่อให้ link layer
  - แต่ที่ switch ไม่มี network layer จึงใช้ mac address ที่อยู่ใน link layer เพื่อดูปลายทางที่จะส่งข้อมูล นั่นคือ router
  - และให้ physical layerส่งไปยัง router
- การรับข้อมูลและส่งต่อใน router
  - physicalจะรับข้อมูลมาทีละbit เมื่อได้ครบทั้งframeแล้วก็จะส่งต่อให้ link layer
  - link layer จะทำการ decapsulate frame ได้เป็น datagram แล้วส่งให้ network layer
  - network layer จะตรวจสอบ IP address ของปลายทาง จากนั้นทำการเลือกเส้นทาง
  - ส่งต่อให้ link layer encapsulation เพื่อระบุ mac address ของ switch ที่ส่งมา
  - และให้ physical layer ส่งไปยัง receiver
- การรับข้อมูลเข้าไปยัง receiver
  - ข้อมูลจะเข้ามาจาก physical layer
  - แต่ละ layer จะทำการถอด header ที่ถูกใส่ด้วย layer เดียวกัน

1.6 network under attack : security

Bad guys : put malware into hosts via Internets

- malware เข้าไปยัง host ได้โดย
  - virus
    - ได้รับโดยการ receive/execute ไฟล์ที่เป็นไวรัส
    - เกิดจากคนที่ไม่ระวัง ไปกดรันไวรัส

By Natthapach Anuwattananon

- worm
    - ได้รับแม้เพียงเชื่อมต่อ internet เฉยๆ
    - สามารถ execute ได้เอง ไม่ต้องมีคนกด
- spyware malware
    - สามารถดักจับการกด key board ของ host เพื่อดัก password
    - สามารถดักจับการเข้า website ของ host
- botnet
    - เครื่องที่ถูก malware ชนิดพิเศษ สามารถควบคุมเพื่อใช้ในการโจมตีแบบ DDoS

Bad guys : attack server, network infrastucture

- Denial of Service (DoS)
    - การโจมตีด้วย traffic ปลอมจำนวนมหาศาล เพื่อขัดขวาง traffic จริง
    - ถ้ามีการใช้ botnet ในการโจมตี จะยิ่งเพิ่มดาเมจจำนวนมาก
        - เรียกว่า Distributed Denial of Service (DDoS)

Bag guys : can sniff packets

- packet sniffing
    - คือการที่เครื่องกลางทาง ดักจับ packet ที่ส่งจาก sender ไปยัง receiver
    - การดักจับนี้สามารถดักจับ password ได้ด้วย
    - ปัจจุบันป้องกันโดย protocal https
- wireshark
    - โปรแกรมฟรีที่เอาไว้ทำ sniffing

Bad guys : can use fake address

- IP spoofing
    - การส่งข้อมูลไปยัง receiver ด้วย address ปลอม

ช่วงปนเล่นๆ

public key and private key

- เป็นการเข้ารหัสแบบที่มีความปลอดภัยสูงมากกกก
- โดยหลักการจะมีคนอยู่ 2 กลุ่มคือ sender และ receiver
- receiver นั้นจะมีกุญแจ 2 ดอก คือ public key และ private key เป็นของตัวเอง
- กุญแจ 2 ดอกนี้มีสกิลพิเศษคือ ของที่ถูกล็อคด้วย public key
  จะสามารถเปิดได้ด้วย private key เท่านั้น
- อีกสกิลนึงคือ จะไม่สามารถสร้าง private key จาก public key ได้ (หมดห่วงเรื่องกุญแจผี)
- receiver จะแหกปากบอกชาวบ้านว่าตัวเองมี public key เป็นอะไร แต่จะไม่บอก private
- sender เมื่อจะส่งข้อมูลหา receiver จะเอา public key มาล็อคข้อมูลที่จะส่ง
- ต่อให้มีโจรมาขโมยของกลางทางก็จะเปิดข้อมูลไม่ได้ เพราะไม่มี private key
- เมื่อถึงปลายทาง receiver ก็จะใช้ private key เปิดข้อมูลมาดู

By Natthapach Anuwattananon

Chapter 2 : Application Layer

2.1 principles of network application

Creating a network app

- write program
  - สามารถรันบน end systems หลายแบบได้
  - ต้องสามารถสื่อสารผ่านทาง network ได้
- ไม่จำเป็นต้องมี application บน network core

Application architectures

- สถาปัตยกรรม network application แบ่งออกเป็น 2 แบบ
- client-server
  - server
    - ทำงานอยู่ตลอดเวลา
    - IP address ไม่เปลี่ยนแปลง
    - data center for scaling
      - ใช้ data center เพื่อขยายขนาดของเครือข่าย
  - client
    - ไม่ต้องทำงานตลอดเวลา
    - IP address เปลี่ยนแปลงได้
    - ไม่สามารถติดต่อ client อื่นได้โดยตรง
- peer-to-peer (p2p)
  - ไม่มี server
  - end systems ใดๆ สามารถติดต่อกันโดยตรงได้เลย
  - peer จะร้องขอข้อมูลจาก peer อื่น และสามารถส่งข้อมูลไปหา peer อื่นได้
  - self scalability
    - สามารถขยายเครือข่ายได้ด้วยตัวเอง
  - peer ไม่จำเป็นต้อง connect กันตลอด

Process communication

- process
  - คือโปรแกรมที่ run อยู่บนเครื่อง host
- communication
  - process ใน host เดียวกัน จะติดต่อกันด้วย inter-process communication
  - process ต่าง host กัน จะติดต่อกันด้วย message
- client-server
  - client process จะเป็นฝ่ายที่เริ่มการสื่อสาร
  - server process จะเป็นฝ่ายนั่งรอคนมาขอคุยด้วย
- peer-2-peer
  - application จะเป็นทั้ง client และ server process

Sockets

- เป็นเหมือนประตูบ้านของ process
- process จะรับ/ส่ง message กันผ่าน socket
- socket จะเชื่อมระหว่าง application layer และ transport laye

Addressing process

- เวลาส่งข้อมูลไปให้ receiver ถ้ามีแค่ ip address แล้วจะรู้มั๊ยว่าส่งไปแล้วมันเข้า process ไหน เพราะใน receiver ก็รันเป็นร้อยๆ process
- เพราะงั้นจะต้องมี port number เพื่อระบุ process ที่จะรับข้อมูลด้วย
- เหมือนกับว่า ip address เป็นที่อยู่ของตึก แล้ว port number จะเป็นเหมือนเลขที่ห้อง

By Natthapach Anuwattananon

- เพราะงั้นเวลาส่งจะต้องระบุทั้ง ip address และ port number

App-layer protocol defines

- application layer protocol จะต้องกำหนดดังนี้
  - type of message exchanged
    - ชนิดของข้อมูลที่จะส่ง
    - e.g., response, request
  - message syntax
    - รูปแบบการวาง field ใน message
  - message semantics
    - ความหมายของแต่ละ field
  - rule
    - กำหนดเวลาและวิธีการ ที่ process จะส่ง/ตอบสนอง message
- open protocols
  - ใช้ฟรี, ใครๆก็ใช้ได้
  - e.g., HTTP, SMTP
- proprietary protocols
  - สร้างขึ้นมาใช้กับ application ของตัวเองโดยเฉพาะ
  - e.g., Skype

What transport service does an app need?

- ความต้องการของ network application มี 4 ด้าน
  - data integrity
    - ความต้องการความถูกต้องของข้อมูล
    - บาง app ต้องการความถูกต้อง 100% (100% reliable data)
    - บาง app ก็ยอมให้ผิดๆได้บ้าง
  - throughput
    - ความต้องการด้านอัตราการส่งข้อมูล
    - บาง app ก็มี throughput ขั้นต่ำ
  - timing
    - ความต้องการด้านความเร็ว
    - บาง app ต้องการ delay ที่น้อยๆ
  - security
    - ความต้องการด้านความปลอดภัย
    - บาง app ต้องการมีการเข้ารหัส

Transport service requirements : common apps

Internet transport protocols services

- TCP service (Transmission Control Protocol)
  - relible transport
    - ข้อมูลที่รับ/ส่ง มีความถูกต้องแน่นอน
  - flow control
    - สามารถควบคุมอัตราการส่งจากผู้ส่งได้
  - congestion control
    - ควบคุมการจราจรหนาแน่นด้วยกรบังคับให้ sender ลดปริมาณการส่ง
  - connection-oriented
    - เป็นการเชื่อต่อแบบต้องมีการสถาปนาการเชื่อต่อ (setup connection)
    - เสียเวลาเพิ่ม 1 round-trip time
  - does not provide

- ไม่เหมาะกับ app ที่ต้องการ timing, minimum throughput, security
- UDP service (User Datagram Protocol)
  - unreliable data transfer
    - ไม่รับรองว่าข้อมูลที่ส่งจะมีความถูกต้อง
  - non connection-oriented
    - ไม่จำเป็นต้องมีการ setup connection
  - does not provide
    - reliability, flow control, congestion control
    - security, timing, minimum throughput
- TCP VS UDP
  - งานที่ต้องขนส่งข้อมูลเยอะๆแล้วต้องการข้อมูลชัวร์ๆ → TCP
  - งานที่ส่งข้อมูลน้อยๆ ต้องการความเร็ว → UDP (ข้อมูลน้อย โอกาสผิดจะน้อย)

Internet app: application, transport protocols

Securing TCP

- transport layer ทั้ง 2 ตัวนั้นไม่รองรับ security
- ถ้าส่ง password ก็จะส่งแบบธรรมดา ถ้าโดนดักข้อมูลก็ได้ password ไปเลย
- SSL (Socket Secure Layer)
  - เป็น library ในชั้นของ application layer
  - สามารถเข้ารหัสข้อมูลก่อนส่งได้
  - end-point authentication
    - เข้ารหัสที่ต้นทาง → ถอดรหัสที่ปลายทาง
- HTTPS
  - เป็น HTTP ที่ผสมเข้ากับ SSL ทำให้มีความปลอดภัยมากขึ้น

2.2 Web and HTTP

First review

- web page นั้นประกอบขึ้นมาด้วย object
- object ก็เช่น HTML file, image, audio …
- web page นั้นจะมี base HTML-file อันนึง ภายในจะเก็บ reference ของ object อื่น
- address ของ object แต่ละตัวนั้นจะเป็น URL

HTTP oerview

- Hyper Text Transfer protocol
- เป็น web's application protocols
- client-server model
  - client
    - browser จะมีการส่ง request เพื่อขอ web page
    - หลังจากได้รับ response จะทำการแสดง web page
    - จำนวนการส่ง request เท่ากับจำนวน object ในหน้าเว็บ
  - server
    - web server จะส่ง object ผ่านไปทาง response
- use TCP
  - client สถาปนาการเชื่อมต่อ TCP กับ server ที่ port 80 (default)
  - server ยอมรับการเชื่อมต่อจาก client
  - แลกเปลี่ยน HTTP message กัน
  - ยุติ TCP connection
- HTTP is "stateless"
  - HTTP ไม่รองรับการเก็บ state ของผู้ใช้

- server จะไม่เก็บ request ครั้งก่อนๆของ client
- ถ้าอยากให้เก็บ state จะต้องทำเองใน application

HTTP connections
- non-persistent HTTP (HTTP 1.0)
  - ส่งข้อมูลแค่ 1 object ก็จะปิด TCP
  - ถ้าอยากได้หลาย object ก็เปิด connection หลายๆรอบเอาสิ
  * เสียเวลากับการ inititial connection มากเกินไป
  - response time
    response time per object = 2RTT + file transmission
    response time per page = sum(2RTT + file transmission)
- persistent HTTP (HTTP 1.1)
  - เปิด connection ครั้งเดียว สามารถส่ง object หลายๆอันได้
  - เปิด connection ทิ้งไว้จนไม่มีข้อมูลส่งมาสักพัก (timeout) ก็จะปิดเอง
  - response time
    response time per object = RTT + file transmission
    response time per page = RTT + sum(RTT + file transmission)

HTTP request message
- request line
  <method> <url> <protocol>/<version>\r\n
- header line
  <header-field-name>: <value>\r\n
  …
  <header-field-name>: <value>\r\n
  \r\n
  - header file name
    - Host: server-name
    - User-Agent: user-application
    - Accept: MIME-type
    - Accept-Language: region
    - Accept-Encoding: encoding-type
    - Accept-Charset: char-set
    - Keep-Alive: connection-timeout
- body
  - entity body

HTTP request : Method type
- GET
  - ส่งข้อมูลผ่านทาง url
  - ไม่เหมาะกับข้อมูลที่ต้องการความปลอดภัย
  - ไม่เหมาะกับข้อมูลที่มีจำนวนมาก
  - ประวัติการส่งข้อมูลจะเก็บไว้ใน history (เพราะติดอยู่กับ url)
- POST
  - ส่งข้อมูลผ่าน entity body
  - เหมาะกับข้อมูลที่ต้องการความปลอดภัย
  - สามารถใช้กับข้อมูลจำนวนมากได้
- HEAD
  - ใช้แค่ตอน test

By Natthapach Anuwattananon

- PUT (HTTP/1.1)
    - ใช้เพื่อ upload file เข้าไปใน url ที่ระบุ
    - ไม่มีใครใช้ เพราะไม่ปลอดภัย
- DELETE
    - delete file ที่ระบุใน url
    - ไม่มีใครใช้ เพราะไม่ปลอดภัย

URL form

<protocol>://<server-name>[:<port-number>][/path]...[?<param>=<value>[&<param>=<value>]...]

HTTP response massage

- status line
    <protocol>/<version> <response-code>\r\n
- header lines
    <header-field-name>: <value>\r\n
    - header-field-name
        Date: วันที่ server ตอบกลับ
        Server: <server-app> <server-OS>
        Lasted-Modified: วันที่แก้ไข data ล่าสุด
        ETAG: etag number
- data

HTTP response : response status code

- 200 OK
    - request สำเร็จ
    - request object อยู่ใน massage นี้
- 301 Moved Permanently
    - object ที่ request อยู่ที่อื่น
    - จะบอก location ที่มี object ใน massage นี้
- 400 Bad Request
    - server อ่าน request ไม่รู้เรื่อง
- 404 Not Found
    - ไม่มีของที่อยากได้
- 505 HTTP Version not Supported

User-server state : Cookies

- การจะทำ cookies นั้น ประกอบด้วย 4 ส่วน
    - cookie header line ใน HTTP response
    - cookie header line ใน HTTP request
    - cookie file เก็บในเครื่อง user (จัดการโดย browser)
    - back-end database ใน web site ต้องมีวิธีจัดการกับ cookie
- การเข้าเว็บครั้งแรก
    - client ส่ง request ไปยัง server
    - server ส่ง response กลับพร้อม header set-cookie:<cookie-id>
    - client จะเก็บ cookie-id ตัวนี้ไว้ในเครื่องพร้อมกับ server-name
- การเข้าเว็บครั้งต่อๆไป
    - client ส่ง request ไปยัง sever พร้อม header cookie: <cookie-id>
    - server จะส่ง response สำหรับ cookie-id นี้กลับไป (แล้วแต่ว่า แต่ละเว็บจะทำอะไร)
- cookies and privacy
    - การใช้ cookie นั้น เป็นเหมือนการอนุญาตให้ web site เรียนรู้สิ่งเกี่ยวกับเรา

By Natthapach Anuwattananon

- ถ้าแคร์เรื่องความส่วนตัว ก็อาจไม่ค่อยโอเค

Web caches : proxy server

- การที่ client ส่ง request ไปยัง server ตลอดนั้นช้า และทำให้การจราจรติดขัด
- ใช้แนวคิดเรื่อง caches มาช่วยในการแก้ปัญหา
- proxy server
    - เป็นเครื่อง server ที่ตั้งอยู่ใน local network เดียวกับ client
    - client จะส่ง request ไปยัง proxy server ก่อน
    - ถ้า requested object มีอยู่ในเครื่อง proxy, proxy ก็จะส่งกลับไปให้ client
        - จะเห็นว่าไม่มี request ส่งออกไปนอก local network เลย
    - ถ้า requested object ไม่มีใน proxy, proxy จึงจะไปขอจาก origin server
    - การเช็คตรงนี้ จะใช้ etag
- ตัว proxy server นั้นทำหน้าที่เป็นทั้ง client และ server

* นอกจาก proxy แล้ว browser ที่ client เองก็มี cache อีกเหมือนกัน เพื่อให้ทำงานเร็วขึ้น

Web caches : conditional GET

- ถ้าใน cache ของ client มี object ที่ต้องการเก็บไว้
    - จะส่ง request ด้วย header If-modified-since: <date>
        - แนบวันที่ cache ไว้ เพื่อให้ server เช็คว่าหลังจากวันนั้น object ยังตัวเดิม
    - ถ้า object ไม่เปลี่ยนแปลง
        - ส่ง response มาด้วย status 304 Not Modified
    - ถ้า object เปลี่ยนแปลง
        - ส่ง response มาด้วย status 200 OK พร้อม object ใหม่

2.3 electronic mail

User Agent

- ก็คือ mail reader นั้นเหละ
- ความสามารถ : edit/read mail

Mail Server

- mailbox
    - เก็บข้อความขาเข้า
- message queue
    - เก็บข้อความขาออก (กำลังจะส่ง)

SMTP

- มี 2 ฝ่ายคือ
    - mail server ฝั่ง client (ผู้ส่ง)
    - mail server ฝั่ง server (ผู้รับ)
- ใช้ TCP, port 25

* message ที่ส่ง อยู่ในรูป 7-bit ASCII

Computer Communication Midterm [Analysis]

By Natthapach Anuwattananon

Protocols

what

- มาตราฐานสำหรับการส่ง message จากต้นทาง ไปปลายทาง

why

- host ใน internet มีจำนวนเยอะมาก ความหลายหลากของ host ก็มีมากตามมาด้วย
- ถ้าไม่มีมาตราฐานกลาง ก็คุยกันไม่ค่อยรู้เรื่อง

how

- protocol จะกำหนด format, order และ action taken ให้
- host จะติดต่อกันด้วยรูปแบบตามที่ protocol กำหนด

FDM

advantage

- ส่งสัญญาณด้วย bandwidth ที่สูงเพราะข้อมูลถูกส่งตลอด

cons

- รองรับผู้ใช้ได้น้อยเพราะคลื่นความถี่แบ่งได้จำกัด

TDM

advantage

- bandwidth ต่ำ เพราะข้อมูลไม่ได้ถูกส่งตลอดเวลา

cons

- รองรับผู้ใช้ได้มาก เพราะแบ่งตามเวลา แต่ยิ่งแบ่ง bandwidth ก็ยิ่งน้อย

packet-switching

circuit-switching

global-ISP

what

- แกนกลางของระบบ internet ที่ทำให้ end user สามารถติดต่อถึงกันได้

why

- ถ้าให้ ISP แต่ละที่ connect กันเอง จะต้องมีสายมากถึง $O(n^2)$ สาย
- แถมแต่ละสายยังต้องโยงข้ามทวีปอีก ใครมันจะไปทำได้

how 1

- สร้าง global ISP ขึ้นมาตรงกลางสิ แล้วก็ให้ isp อื่นมา connect
    - problem
    - งั้นคนที่เป็นเจ้าของ global ISP ก็จะมีอำนาจในการเป็นเจ้าของ internet ด้วยสิ
    - มันจะขัดกับหลักการของ internet ที่ว่า “ไม่มีใครเป็นเจ้าของ internet”

how 2

- งั้นก็มี global ISP หลายๆอัน เพื่อคานอำนาจกันเองล่ะกัน

content provider network

what

- เครือข่ายของผู้ให้บริการ content บน internet

why

- ผู้ให้บริการมี content ที่ต้องการให้ ผู้ใช้ internet เข้าถึง content เหล่านี้ได้

how

- ผู้ให้บริการจะ connect data center ที่เก็บ content ไว้ เข้ากับ ISP
- ส่วนมากจะ connect กับ Tier 1 หรือ regional เพื่อให้ผู้ใช้ เข้าถึงได้เร็วขึ้น

By Natthapach Anuwattananon

traceroute

what

- โปรแกรมที่ไว้ใช้ในการติดตามเส้นทางจาก เครื่องที่กำลังใช้งาน ไปยังเครื่องปลายทาง

how

- จะส่ง packet เล็กๆ ไปยังแต่ละ router ที่ผ่าน router ล่ะ 3 probe
- จากนั้นจะแสดงชื่อของ router นั้น พร้อมกับ delay ในการตอบกลับสำหรับแต่ละ probe

protocol layer

why

- การแยกส่วนการทำงาน ทำให้ง่ายต่อการบำรุงรักษา, ปรับปรุง

how

- แบ่งการทำงานของ layer ออกเป็นชั้นๆ
- การเปลี่ยนรูปแบบการทำงานของแต่ละ layer ไม่ส่งผลแต่การทำงานของ layer อื่น

DDoS Attack

what

- การกระหน่ำโจมตีแบบ DoS ด้วย host จำนวนมากที่กระจายอยู่ตาม network

why

- การโจมตีแบบ DoS นั้นมีประสิทธิภาพ แต่ยังไม่รุนแรงพอเพราะถูกจำกัดด้วยจำนวนเครื่องที่มี

how

- กระจาย spam ไปตาม network เพื่อสร้าง botnet
- โจมตีแบบ DoS ด้วย botnet ที่สร้างไว้ ความรุนแรงจะเพิ่มขึ้นตาม botnet ที่มี

packet sniffing

what

- การดักจับ packet กลางทางเพื่อขโมยข้อมูล

how

- ดักข้อมูลที่จะส่งจาก source ไปยัง destination ที่กลางทาง

IP spoofing

what

- การปลอมแปลง IP เพื่อส่ง source ปลอมไปยัง destination

client-server architecture

advantage

- รับประกันได้ว่า client จะได้รับข้อมูลตลอดเวลา เพราะ server จะทำงานตลอด

cons

- ภาระจะตกอยู่ที่ฝั่ง server เพราะต้องทำงานตลอดเวลา
- การขยายขนาดเครือข่าย (scalability) ทำได้ยาก ถ้าจะทำต้องเพิ่ม bandwidth ของ server ให้เพียงพอกับจำนวน client

peer-to-peer

advantage

- ไม่จำเป็นต้องมี host ไหนที่ต้องทำงานตลอดเวลา ทุกคนจะทำงานหนักเท่าๆกัน
- สามารถขยายขนาดเครือข่ายได้ด้วยตัวเอง (self scalability) เพราะ …

cons

- อาจเกิดเหตุการณ์ที่คนที่มี chunk ที่ต้องการปิดเครื่องหนี ทำให้ไม่สามารถโหลดข้อมูลให้เสร็จได้

By Natthapach Anuwattananon

Socket

what

- เป็น API ใน application layer (ไม่ได้เป็น protocol)
- ใช้สำหรับการสื่อสารระหว่าง process

why

- การส่ง message ด้วยการระบุเพียง IP address นั้นไม่เพียงพอ เพราะ IP address ระบุได้เพียงเครื่อง host แต่ไม่สามารถระบุ process ที่จะรับ message ในเครื่อง host ได้

how

- มี port number สำหรับเป็นหมายเลขของ socket ของแต่ละ process
- 1 socket = 1 port number
- 1 process สามารถมีได้หลาย socket แต่ 1 socket เป็นของ process เพียงแค่ process เดียว
- เวลาส่ง message จะต้องระบุทั้ง IP address และ port number

persistent HTTP

why

- protocol แบบ non-persistent HTTP ที่เกิดขึ้นมาก่อนนั้นเสียเวลาในการ setup TCP มากเกินไป

how

- จากเดิมที่ 1 TCP-connect ต่อ 1 objects เปลี่ยนเป็น 1 TCP-connection โหลด object ทั้งหมด จนกว่าจะ timeout

cookie

what

- เป็น API ที่ช่วยการเก็บ state

why

- HTTP เป็น protocol แบบ stateless เพราะฉะนั้นอยากเก็บ state เลยต้องทำเองที่ application

how

- ที่ฝั่ง client จะมี cookie file สำหรับเก็บ cookie ของแต่ละ web site
- ครั้งแรกที่เข้า web ฝั่ง server จะสร้าง cookie ของ client และส่งกลับด้วย header set-cookie
- หลังจากนั้นทุกครั้งที่ client ส่ง request จะส่ง cookie ผ่านทาง header cookie
- server จะต้องมีวิธีในการจัดเก็บ,เรียนรู้ จาก request ที่ client นี้ส่งมา
- เมื่อ client ส่ง request, server จะเลือกข้อมูลที่เหมาะกับ cookie ที่ส่งเข้ามาให้

proxy

what

- เป็น infrastructure ที่ออกแบบมาเพื่อลด traffic ของ origin server

why

- นับวัน internet ยิ่งขยายขนาด ทำให้มี client จำนวนมาก

By Natthapach Anuwattananon