Overlay Networks
Jennifer Rexford
COS 461: Computer Networks
Lectures: MW 10-10:50am in Architecture N101
http://www.cs.princeton.edu/courses/archive/spr12/cos461/

Skype
• Niklas Zennström and Janus Friis in 2003
• Developed by KaZaA
• Instant Messenger (IM) with voice support
• Based on peer-to-peer (P2P) networking technology

Skype Network Architecture
• Login server is the only central server
• Both ordinary host and super nodes are clients
• Any node with public IP address and resources can become a super node

Challenges of Firewalls and NATs
• Firewalls
  – Often block UDP traffic
  – Usually allow hosts to initiate connections on port 80 (HTTP) and 443 (HTTPS)
• Network Address Translation (NAT)
  – Cannot easily initiate traffic to a host behind a NAT
• Skype must deal with these problems
  – Discovery: client exchanges messages with super node
  – Traversal: sending data through an intermediate peer

Data Transfer
• UDP directly between the two hosts
  – Both hosts have public IP addresses, no UDP blocks
  – Easy: the hosts can exchange UDP packets directly
• UDP between an intermediate peer
  – One or both hosts with a NAT
  – Neither host’s network blocks UDP traffic
  – Solution: direct UDP packets through another node
• TCP between an intermediate peer
  – Hosts behind NAT and UDP-restricted firewall
  – Solution: TCP connections through another node
Silence Suppression

• What to transfer during quiet periods?
  – Could save bandwidth by reducing transmissions
• Skype does not appear to do silence suppression
  – Maintain the UDP bindings in the NAT boxes
  – Provide background noise to play at the receiver
  – Avoid drop in the TCP window size
• Skype sends data when call is “on hold”
  – Send periodic messages as a sort of heartbeat
  – Maintain the UDP bindings in the NAT boxes

Skype Data Transfer

• Audio compression
  – Voice packets around 67 bytes
  – Up to 140 packets per second
  – Around 5 KB/sec (40 kbps) in each direction
• Encryption
  – Data packets are encrypted in both directions
  – To prevent snooping on the phone call
  – ... by someone snooping on the network
  – ... or by the intermediate peers forwarding data

Overlay Networks

IP Tunneling to Build Overlay Links

• IP tunnel is a virtual point-to-point link
  – Illusion of direct link between two separated nodes
    Logical view: A  B  tunnel  E  F
    Physical view: A  B  tunnel  E  F
• Encapsulation of packet inside an IP datagram
  – Node B sends a packet to node E
  – ... containing another packet as the payload
Tunnels Between End Hosts

Overlay Networks
- Logical network built on top of physical network
  - Overlay link is tunnel through underlying network
- Many logical networks may coexist at once
  - Over the same underlying network
- Nodes are often end hosts
  - Acting as intermediate nodes that forward traffic
- Who controls the nodes providing service?
  - The party providing the service
  - Distributed collection of end users

Case Study: Resilient Overlay Networks

RON: Resilient Overlay Networks
Premise: by building application overlay network, can increase performance and reliability of routing

RON Circumvents Policy Restrictions
- IP routing depends on AS routing policies
  - But hosts may pick paths that circumvent policies

RON Adapts to Network Conditions
- Start experiencing bad performance
  - Then, start forwarding through intermediate host
RON Customizes to Applications

- VoIP traffic: low-latency path
- Bulk transfer: high-bandwidth path

How Does RON Work?

- Keeping it small to avoid scaling problems
  - A few friends who want better service
  - Just for their communication with each other
  - E.g., VoIP, gaming, collaborative work, etc.
- Send probes between each pair of hosts

How Does RON Work?

- Exchange the results of the probes
  - Each host shares results with every other host
  - Essentially running a link-state protocol!
  - So, every host knows the performance properties
- Forward via intermediate host when needed

RON Works in Practice

- Faster reaction to failure
  - RON reacts in a few seconds
  - BGP sometimes takes a few minutes
- Single-hop indirect routing
  - No need to go through many intermediate hosts
  - One extra hop circumvents the problems
- Better end-to-end paths
  - Circumventing routing policy restrictions
  - Sometimes the RON paths are actually shorter

RON Limited to Small Deployments

- Extra latency through intermediate hops
  - Software and propagation delays for forwarding
- Overhead on the intermediate node
  - Imposing CPU and I/O load on the host
- Overhead for probing the virtual links
  - Bandwidth consumed by frequent probes
  - Trade-off between probe overhead & detection speed
- Possibility of causing instability
  - Moving traffic in response to poor performance
  - May lead to congestion on the new paths

Electronic Mail
Mail Servers and User Agents

- **Mail servers**
  - Always on and always accessible
  - Transferring e-mail to and from other servers
- **User agents**
  - Sometimes on and sometimes accessible
  - Intuitive interface for the user

Store-and-Forward Model

- Messages sent through a series of servers
  - A server stores incoming messages in a queue
  - ... to await attempts to transmit them to next hop
- If the next hop is not reachable
  - The server stores the message and tries again later
  - Each server adds a Received header
    - To aid in diagnosis of problems

Scenario: Alice Sends Message to Bob

1) Alice uses UA to compose message "to" bob@someschool.edu
2) Alice's UA sends message to her mail server; message placed in message queue
3) Alice's mail server opens TCP connection with Bob's mail server
4) Alice's mail server sends Alice's message over the TCP connection
5) Bob's mail server places the message in Bob's mailbox
6) Bob invokes his user agent to read message

Identifying the Mail Server

- Alice identifying her mail server
  - User-agent configuration (e.g., smtp.cs.princeton.edu)
- Alice's mail server identifying Bob's mail server
  - From name in Bob's e-mail address (e.g., yale.edu)
  - Domain name is not necessarily the mail server
    - Mail server may have longer/cryptic name
    - Multiple servers may exist to tolerate failures
  - Identifying the mail server for a domain
    - DNS query asking for MX records (Mail eXchange)
    - Then, a regular DNS query to learn the IP address

Simple Mail Transfer Protocol

- Client-server protocol
  - Client is sender, server is receiver
- Reliable data transfer
  - Built on top of TCP (on port 25)
- Push protocol
  - Sending server pushes file to the receiving server
  - ... rather than waiting for the receiver to request it

Sample SMTP interaction

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250 Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
C: How about pickles?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```
**Try SMTP For Yourself**

- **Running SMTP**
  - Run “telnet servername 25” at UNIX prompt
  - See 220 reply from server
  - Enter HELO, MAIL FROM, RCPT TO, DATA commands
- **Spoofing is easy!**
  - Just forge the argument of the “FROM” command
  - ‘…leading to all sorts of problems with spam’
- **Spammers can be even more clever**
  - E.g., using open SMTP servers to send e-mail
  - E.g., forging the “Received” header

---

**Example With Received Header**

```
Return-Path: casado@cs.stanford.edu
Date: Wed, 4 Jan 2006 14:05:35 -0800
Message-ID: <43BC46AF.3030306@cs.stanford.edu>
Content-Transfer-Encoding: 7bit
Content-Type: text/plain; charset=ISO-8859-1; format=flowed
Subject: Using VNS in Class
CC: Martin Casado <casado@cs.stanford.edu>
To: jrex@CS.Princeton.EDU
MIME-Version: 1.0
User-Agent: Mozilla Thunderbird 1.0 (Windows/20041206)
From: Martin Casado <casado@cs.stanford.edu>
```

---

**Multiple Server Hops**

- **Typically at least two mail servers**
  - Sending and receiving email servers
- **May be more**
  - Separate servers for key functions
  - Spam filtering, virus scanning
  - Servers that redirect the message
    - From jrex@princeton.edu to jrex@cs.princeton.edu
    - Messages to princeton.edu go through extra hops
  - Electronic mailing lists
    - Mail delivered to the mailing list’s server
    - ‘… and then the list is expanded to each recipient’

---

**Influence of PCs on E-Mail Retrieval**

- **Separate machine for personal use**
  - Users did not want to log in to remote machines
- **Resource limitations**
  - Most PCs did not have enough resources to act as a full-fledged e-mail server
- **Intermittent connectivity**
  - PCs only sporadically connected to the network
  - Too unwieldy to have sending server keep trying
- **Led to the creation of new e-mail agents**
  - POP, IMAP, and Web-based e-mail

---

**Retrieving E-Mail From the Server**

- **Server stores incoming e-mail by mailbox**
  - Based on the “From” field in the message
- **Users need to retrieve e-mail**
  - Asynchronous from when the message was sent
  - With a way to view and organize messages
  - **In the olden days…**
    - User logged on to machine where mail was delivered
    - Users received e-mail on their main work machine
    - **Now, user agent typically on a separate machine**
    - ‘… and sometimes on more than one such machine’

---

**Post Office Protocol (POP)**

- **POP goals**
  - Support users with intermittent connectivity
  - Retrieve e-mail messages when connected
- **Typical user-agent interaction with a POP server**
  - Connect to the server
  - Retrieve all e-mail messages
  - Store messages on the user’s PCs as new messages
  - Delete the messages from the server
  - Disconnect from the server
### Limitations of POP

- **Does not handle multiple mailboxes easily**
  - Designed to put user’s incoming e-mail in one folder
- **Not designed to keep messages on the server**
  - Instead, designed to download messages to client
- **Poor handling of multi-client access to mailbox**
  - Increasingly important as users have home PC, work PC, laptop, cyber café computer, PDA, etc.
- **High network bandwidth overhead**
  - Transfers all of e-mail messages, often well before they are read (and they might not be read at all!)

### Interactive Mail Access Protocol (IMAP)

- **Supports connected and disconnected operation**
  - Users can download message contents on demand
- **Multiple clients can connect to mailbox at once**
  - Detects changes made to mailbox by other clients
  - Server keeps message state (e.g., read, replied to)
- **Access to parts of messages and partial fetch**
  - Clients can retrieve individual parts separately
  - E.g., message text without attachments
- **Multiple mailboxes on the server**
- **Server-side searches**

### Web-Based E-Mail

- **User agent is an ordinary Web browser**
  - User communicates with server via HTTP
  - E.g., Gmail, Yahoo mail, and Hotmail
- **Reading e-mail**
  - Web pages display the contents of folders
  - “GET” request to retrieve the various Web pages
- **Sending e-mail**
  - User types text into a form and submits to server
  - “POST” request to upload data to the server
  - Server uses SMTP to deliver message to other servers

### Conclusions

- **Overlay networks**
  - Tunnels between host computers
  - Build networks “on top” of the Internet
  - Deploy new protocols and services
- **Benefits of overlay networks**
  - Customization to the applications and users
  - Incremental deployment of new technologies
  - May perform better than the underlying network
- **Precept: Distributed Hash Tables**