



Disruption Tolerant Networks

Acknowledgments: Lecture slides are from Computer networks course thought by Jennifer Rexford at Princeton University. When slides are obtained from other sources, a reference will be noted on the bottom of that slide and full reference details on the last slide.

Goals of Today's Lecture



- Underlying assumptions of the Internet
 - And how they are cooked into the protocols
- Challenging network environments
 - Example networking scenarios
 - Delay-Tolerant Networking architecture
- E-mail as a example of disruption tolerance
 - Mail servers and user agents
 - Simple Mail Transfer Protocol (SMTP)
 - Retrieving e-mail from a mail server
- E-mail message format (if time allows)



Assumptions Underlying the Internet Protocols



Best-Effort Packet Delivery

- Abstract IP datagram
 - Sending a portion of a message in each packet
 - *Assumption: end hosts provide message abstraction*
- No application or transport-level state
 - Routers do not maintain state across a connection
 - *Assumption: communicating hosts can store this state*
- Best-effort delivery
 - Drop packets during times of overload
 - *Assumption: retransmission by end hosts is sufficient*

Stationary Hosts and Stable Topology



- Addressing
 - Hierarchical 32-bit IP addresses
 - *Assumption: end hosts are largely stationary*
- Routing
 - Discover the network topology and compute “best” path
 - *Assumption: topology is relatively stable over time*
- Drop on failure
 - Drop packets when no route currently exists
 - *Assumption: communicating hosts usually connected*



End-to-End Argument

- Link properties
 - Links exist and are generally reliable
 - *Assumption: loss rates typically less than 1%*
- Flow control and congestion control
 - React to flow control on a half round-trip time
 - React to congestion on a full round-trip time
 - *Assumption: end-to-end path has reasonably small RTT*
- Router storage
 - Short-term queuing of a few packets
 - *Assumption: no long-term storage of data in the network*



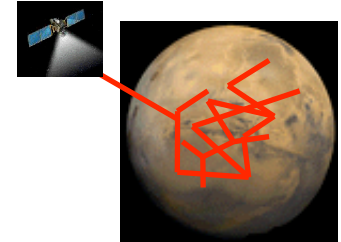
Challenging Network Environments

What are Challenged Networks?



- Unusual

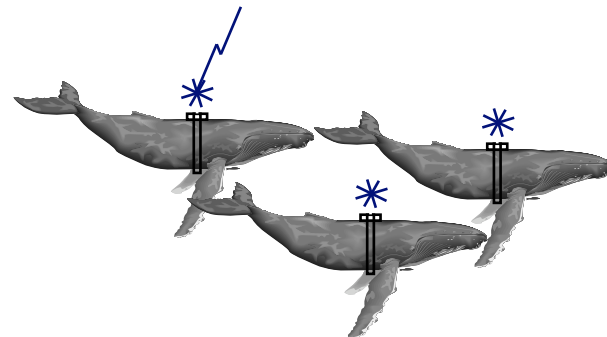
- Containing features or requirements a networking architecture designer would find surprising or difficult to reason about



- Challenged

- Operating environment makes communications difficult

- **Examples:** mobile, power-limited, far-away nodes communicating over poorly or intermittently-available links



Challenging Environments

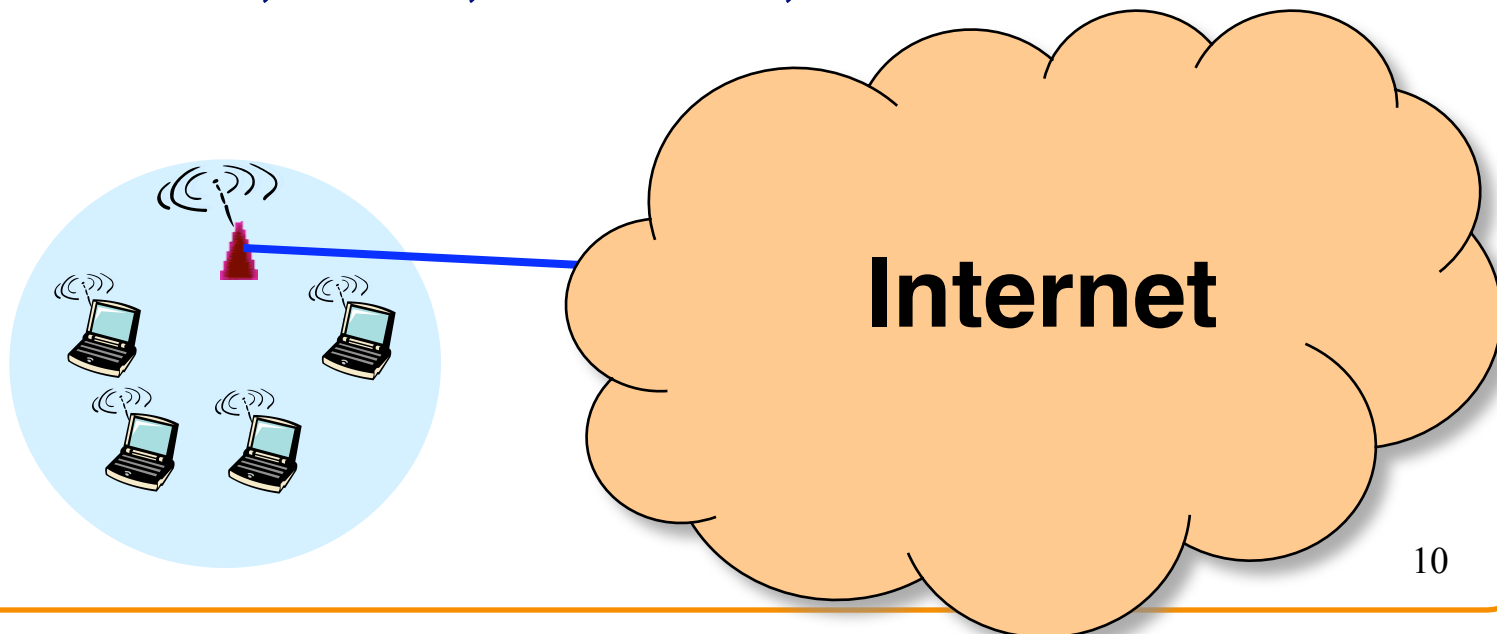


- Random or predictable node mobility
 - Military/tactical networks (clusters meet clusters)
 - Mobile routers with disconnection (e.g., ZebraNet)
 - Daily schedule for a bus passing by a village
- Periods of complete disconnection
 - E.g., the bus is out of range
- Big delays and low bandwidth (high cost)
 - Satellites (GEO, LEO)
 - Exotic links (e.g., deep space or underwater acoustics)
- Big delays and high bandwidth
 - Busses, mail trucks, delivery trucks, etc.

Limp Along With Internet Protocols?



- Run existing Internet protocols
 - And endure the poor performance and poor reliability
 - ... and the risk that communication *never* succeeds
- Deploy proxies at the boundary points
 - E.g., at the wireless/wired boundary
 - To retransmit, cache, transcode, ...





Design New Protocols?

- Revisit the assumptions underlying the Internet
 - Create new assumptions tailored to the environment
 - Design new protocols based on those assumptions
- Advantages
 - More efficient, reliable, and better-performing network
 - Especially for extremely challenging environments
- Disadvantages
 - Additional protocols and complexity, and perhaps cost
 - Significant risk of incompatibility with the Internet

Example Projects



- Digital Gangetic Plains
 - Low-cost networking in rural India
 - Outdoor long-distance directional links using 802.11
 - <http://www.cse.iitk.ac.in/users/braman/dgp.html>
- Sami Network Connectivity Project
 - Internet connectivity for nomadic reindeer herders
 - E-mail, cached Web access, & reindeer herd telemetry
 - Opportunistic relaying of data through gateways
 - <http://www.snc.sapmi.net/>
- ZebraNet
 - Study animal migration and inter-species interaction
 - Tracking collars, P2P communication, and base stations
 - <http://www.princeton.edu/~mrm/zebranet.html>



Disruption Tolerant Networking



DTN Architecture

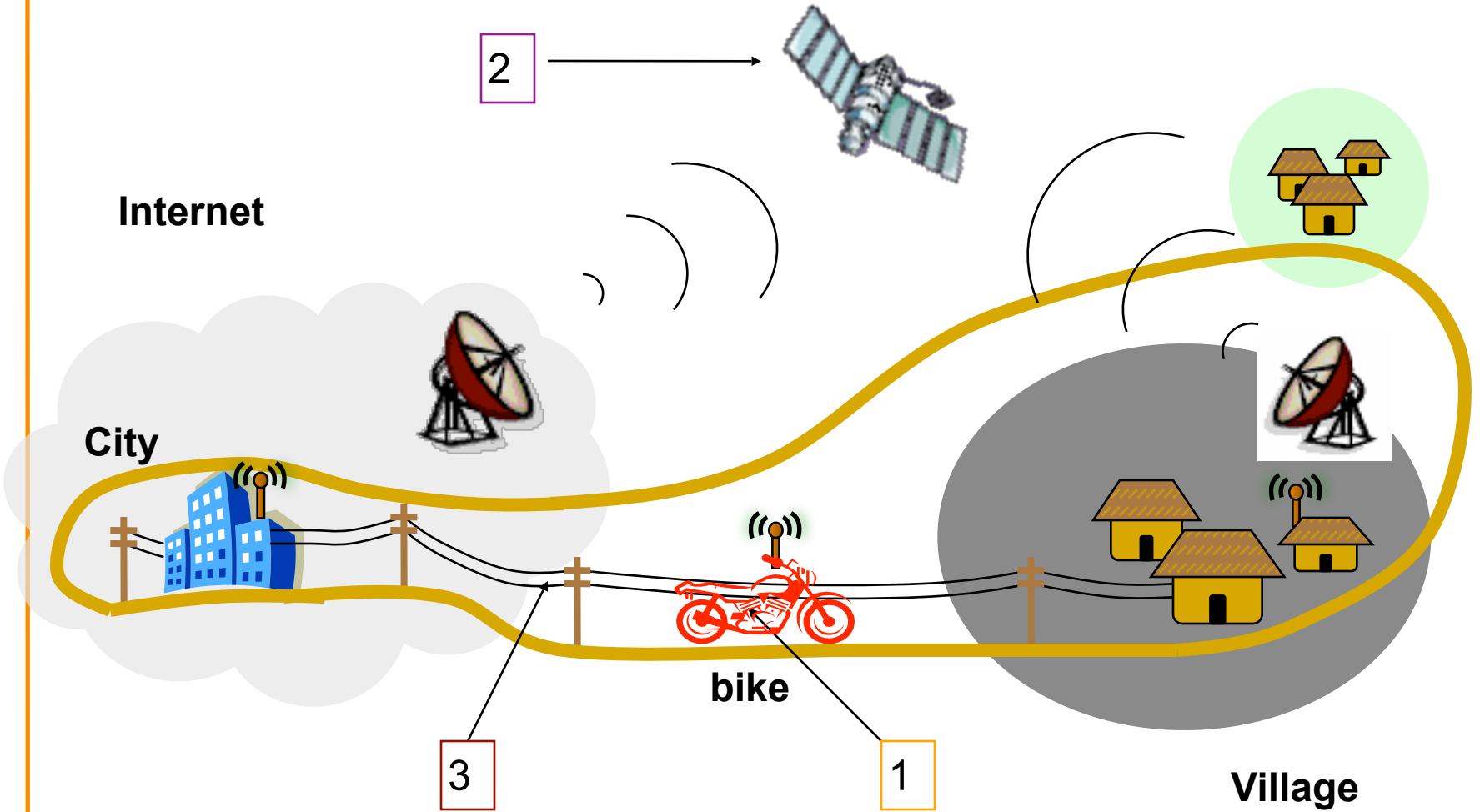
- Goals
 - Interoperability across ‘radically heterogeneous’ networks
 - Tolerate delay and disruption
 - Acceptable performance in high loss/delay/error environments
 - Decent performance for low loss/delay/error environments
- Components
 - Flexible naming scheme
 - Message abstraction and API
 - Extensible Store-and-Forward Overlay Routing
 - Per-(overlay)-hop reliability and authentication



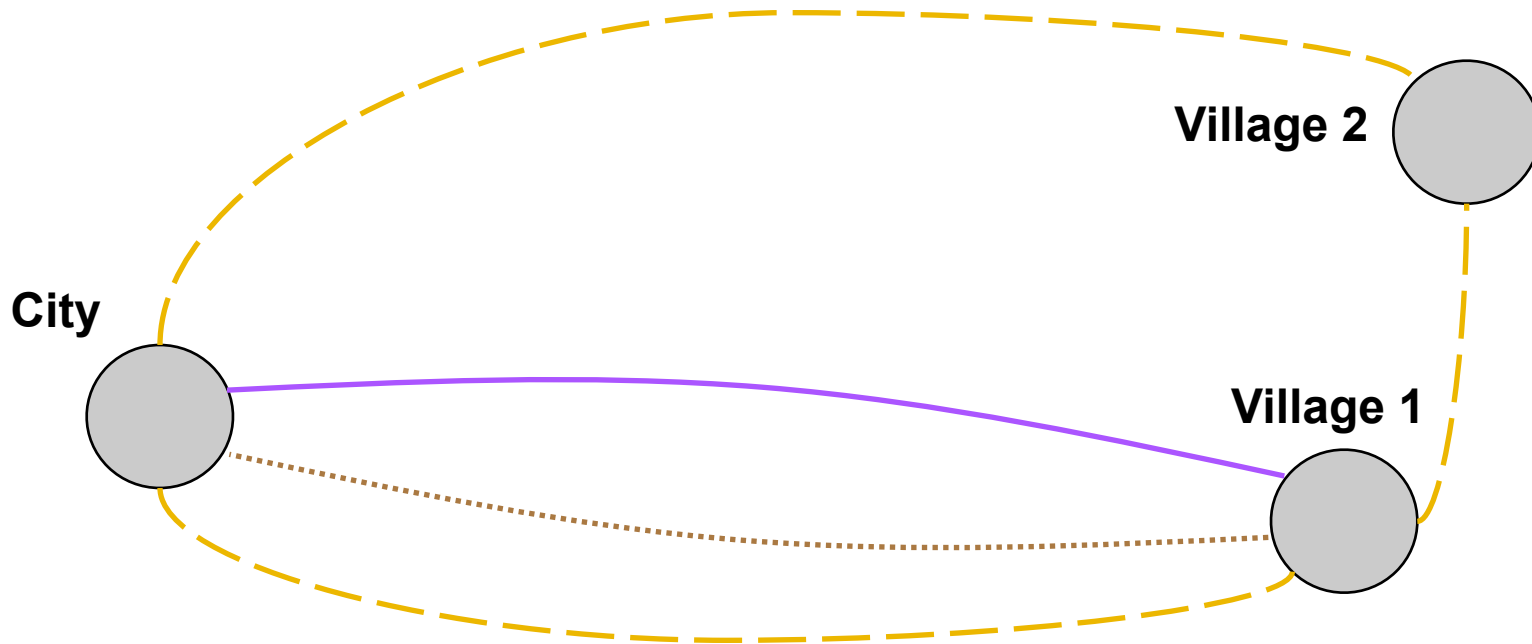
DTN Routing

- DTN Routers form an overlay network
 - Only selected/configured nodes participate
 - Nodes have persistent storage
- DTN routing topology is a time-varying multigraph
 - Links come and go, sometimes predictably
 - Use any/all links that can possibly help
 - Scheduled, Predicted, or Unscheduled Links
 - May be direction specific
 - May learn from history to predict schedule
- Messages fragmented based on dynamics
 - Proactive fragmentation: optimize contact volume
 - Reactive fragmentation: resume where you failed

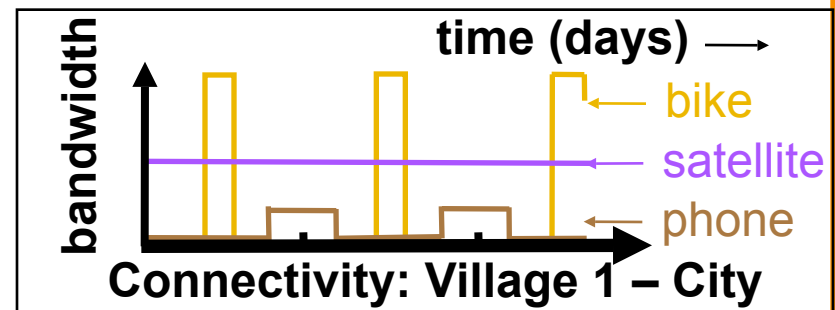
Example Routing Problem



Example Graph Abstraction



- bike (data mule)**
intermittent high capacity
- Geo satellite**
medium/low capacity
- dial-up link**
low capacity



The DTN Routing Problem



- Goal: satisfy message demand matrix
- Vertices have buffer limits
- Edge is possible chance to communicate:
 - One-way: $(S, D, c(t), d(t))$
 - (S, D) : source/destination ordered pair of contact
 - $c(t)$: capacity (rate); $d(t)$: delay
 - A Contact is when $c(t) > 0$ for some period $[i_k, i_{k+1}]$
- Problem: optimize some metric of delivery
 - What metric to optimize? Efficiency? Cost?



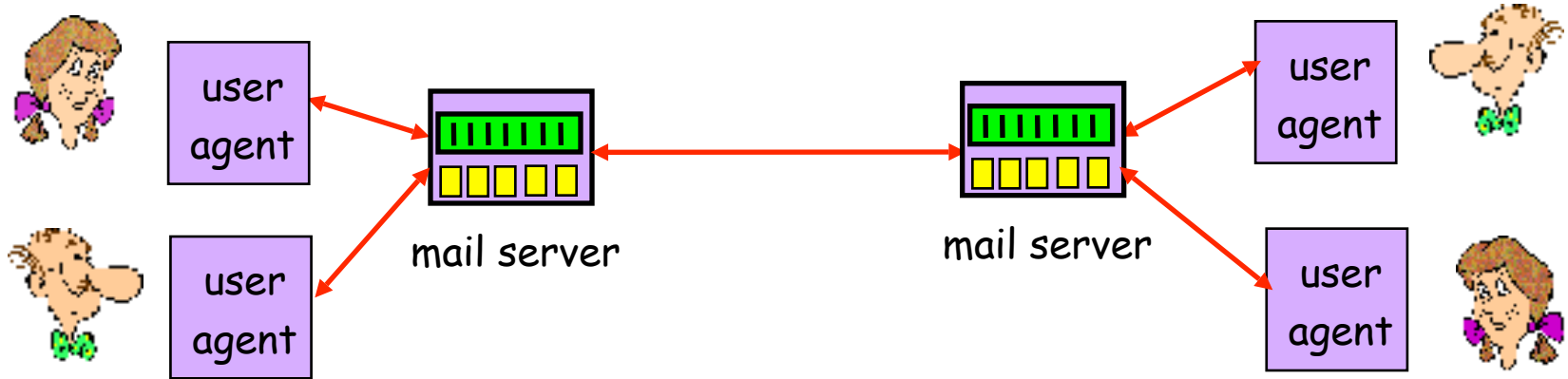
Delivering E-Mail

E-Mail Must Tolerate Disruptions



- Message abstraction
 - Sending a (potentially large) message
 - From one user to another user
 - Okay if there is some delay in delivering the message
- Users may not be online together
 - Receiver may be offline when the sender sends
 - Sender may be offline when the receiver receives
 - Cannot afford to wait until they are both online
- Users may connect from different places
 - Home, work, airport, hotel room, ...
 - Cannot assume a single IP address, or single host

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

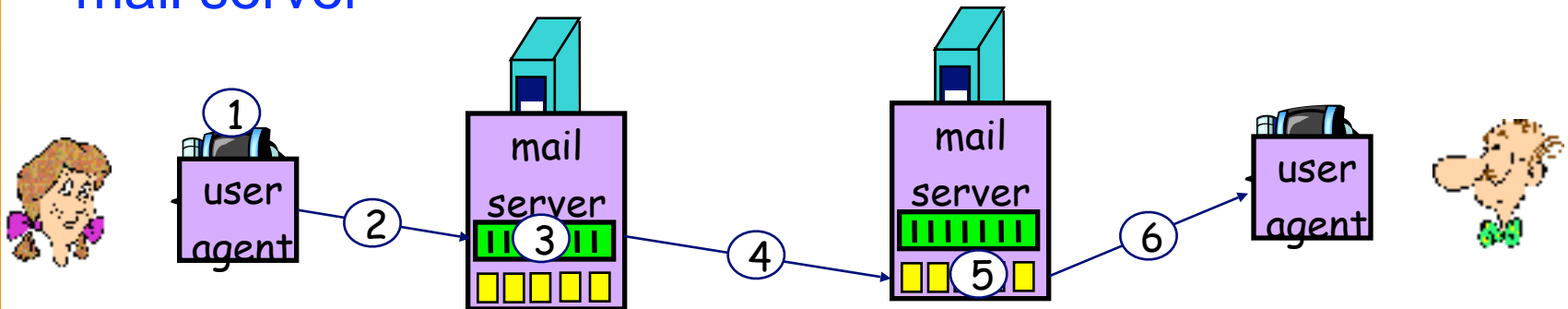


- Messages sent through a series of servers
 - A server stores incoming messages in a queue
 - ... to await attempts to transmit them to the next hop
- If the next hop is not reachable
 - The server stores the message and tries again later
- Each server adds a Received header
 - To aid in diagnosis of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

Scenario: Alice Sends Message to Bob



- 1) Alice uses UA to compose message "to" `bob@some school.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

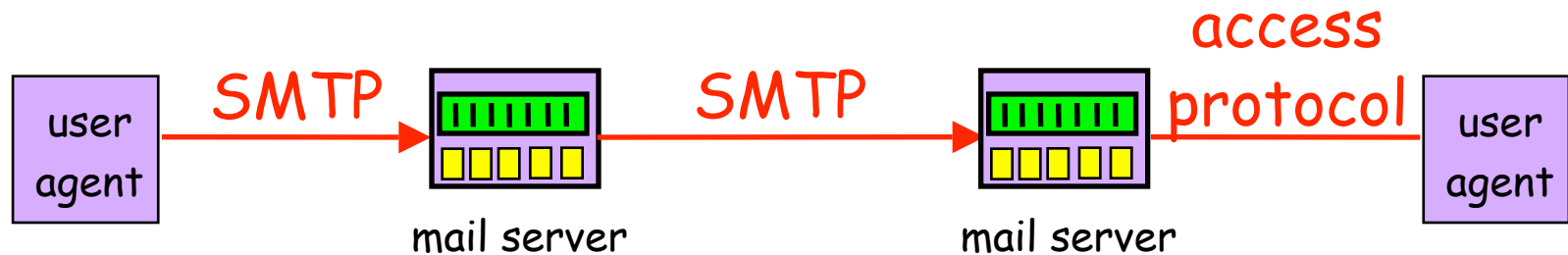




Identifying the Mail Server

- Alice identifying her mail server
 - Explicit configuration of her user agent (e.g., ce.sharif.edu)
- Alice's mail server identifying Bob's mail server
 - From the domain name in Bob's e-mail address (e.g., tehran.ac.ir)
- Domain name is not necessarily the mail server
 - Mail server may have longer/cryptic name
 - E.g., ce.princeton.edu vs. mail.ce.princeton.edu
 - Multiple servers may exist to tolerate failures
 - E.g., cnn.com vs. atlmail3.turner.com and nycmail2.turner.com
- Identifying the mail server for a domain
 - DNS query asking for MX records (Mail eXchange)
 - E.g., nslookup -q=mx tehran.ac.ir
 - Then, a regular DNS query to learn the IP address

Simple Mail Transfer Protocol



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



Multiple Server Hops

- Typically at least two mail servers
 - Sending and receiving sides
- 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

Example With Received Header



Return-Path: <casado@cs.stanford.edu>

Received: from ribavirin.CS.Princeton.EDU (ribavirin.CS.Princeton.EDU [128.112.136.44])
by newark.CS.Princeton.EDU (8.12.11/8.12.11) with SMTP id k04M5R7Y023164
for <jrex@newark.CS.Princeton.EDU>; Wed, 4 Jan 2006 17:05:37 -0500 (EST)

Received: from bluebox.CS.Princeton.EDU ([128.112.136.38])
by ribavirin.CS.Princeton.EDU (SMSSSMTP 4.1.0.19) with SMTP id M2006010417053607946
for <jrex@newark.CS.Princeton.EDU>; Wed, 04 Jan 2006 17:05:36 -0500

Received: from smtp-roam.Stanford.EDU (smtp-roam.Stanford.EDU [171.64.10.152])
by bluebox.CS.Princeton.EDU (8.12.11/8.12.11) with ESMTP id k04M5XNQ005204
for <jrex@cs.princeton.edu>; Wed, 4 Jan 2006 17:05:35 -0500 (EST)

Received: from [192.168.1.101] (adsl-69-107-78-147.dsl.pltn13.pacbell.net [69.107.78.147])
(authenticated bits=0)
by smtp-roam.Stanford.EDU (8.12.11/8.12.11) with ESMTP id k04M5W92018875
(version=TLSv1/SSLv3 cipher=DHE-RSA-AES256-SHA bits=256 verify=NOT);
Wed, 4 Jan 2006 14:05:32 -0800

Message-ID: <43BC46AF.3030306@cs.stanford.edu>

Date: Wed, 04 Jan 2006 14:05:35 -0800

From: Martin Casado <casado@cs.stanford.edu>

User-Agent: Mozilla Thunderbird 1.0 (Windows/20041206)

MIME-Version: 1.0

To: jrex@CS.Princeton.EDU

CC: Martin Casado <casado@cs.stanford.edu>

Subject: Using VNS in Class

Content-Type: text/plain; charset=ISO-8859-1; format=flowed

Content-Transfer-Encoding: 7bit

Retrieving E-Mail From the Server



- Server stores incoming e-mail by mailbox
 - Based on the field in the message
- Users need to retrieve e-mail
 - Asynchronous from when the message was sent
 - With a way to view the message and reply
 - With a way to organize and store the messages
- In the olden days...
 - User logged on to the 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

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
 - ... due to dial-up connections, and shutting down of PC
 - Too unwieldy to have sending server keep trying
- Led to the creation of new e-mail agents
 - POP, IMAP, and Web-based e-mail

Post Office Protocol (POP)



- POP goals
 - Support users with intermittent network connectivity
 - Allow them to retrieve e-mail messages when connected
 - ... and view/manipulate messages when disconnected
- 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 the client
- Poor handling of multiple-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 the 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 the mailbox by other clients
 - Server keeps state about message (e.g., read, replied to)
- Access to parts of messages and partial fetch
 - Clients can retrieve individual parts separately
 - E.g., text of a message without downloading attachments
- Multiple mailboxes on the server
 - Client can create, rename, and delete mailboxes
 - Client can move messages from one folder to another
- Server-side searches
 - Search on server before downloading messages



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
 - ... and allow users to download and view messages
 - “GET” request to retrieve the various Web pages
- Sending e-mail
 - User types the text into a form and submits to the server
 - “POST” request to upload data to the server
 - Server uses SMTP to deliver message to other servers

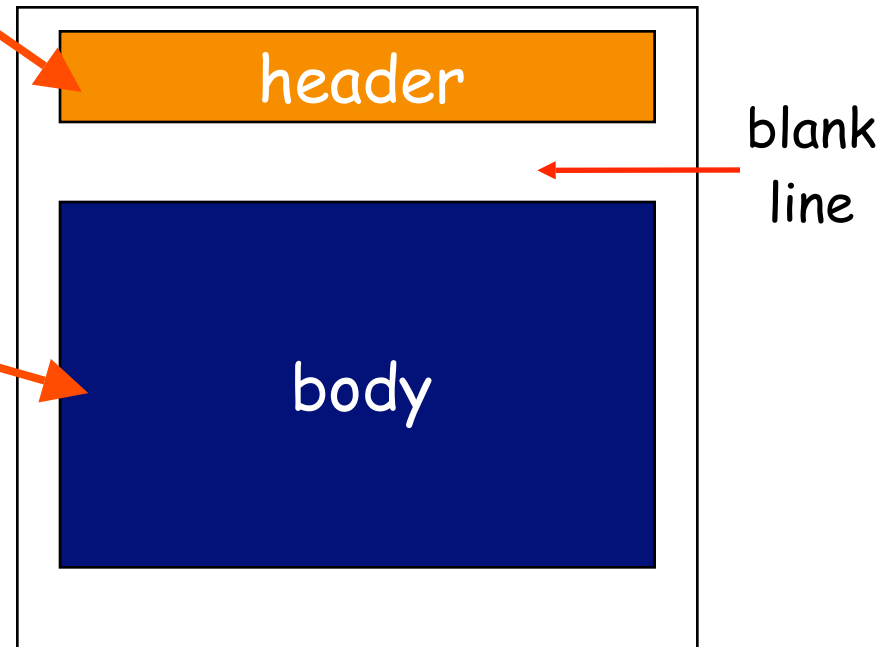


E-Mail Messages

E-Mail Message



- E-mail messages have two parts
 - A header, in 7-bit U.S. ASCII text
 - A body, also represented in 7-bit U.S. ASCII text
- Header
 - Lines with “type: value”
 - “To: jrex@princeton.edu”
 - “Subject: Go Tigers!”
- Body
 - The text message
 - No particular structure or meaning



E-Mail Message Format (RFC 822)



- E-mail messages have two parts
 - A header, in 7-bit U.S. ASCII text
 - A body, also represented in 7-bit U.S. ASCII text
- Header
 - Series of lines ending in carriage return and line feed
 - Each line contains a type and value, separated by “:”
 - E.g., “To: jrex@princeton.edu” and “Subject: Go Tigers”
 - Additional blank line before the body begins
- Body
 - Series of text lines with no additional structure/meaning
 - Conventions arose over time (e.g., e-mail signatures)

Limitation: Sending Non-Text Data



- E-mail body is 7-bit U.S. ASCII
 - What about non-English text?
 - What about binary files (e.g., images and executables)?
- Solution: convert non-ASCII data to ASCII
 - Base64 encoding: map each group of three bytes into four printable U.S.-ASCII characters
 - Uuencode (Unix-to-Unix Encoding) was widely used

```
begin 644 cat.txt
#0V%T
`
end
```

– Limitation: filename is the only clue to the data type

Limitation: Sending Multiple Items



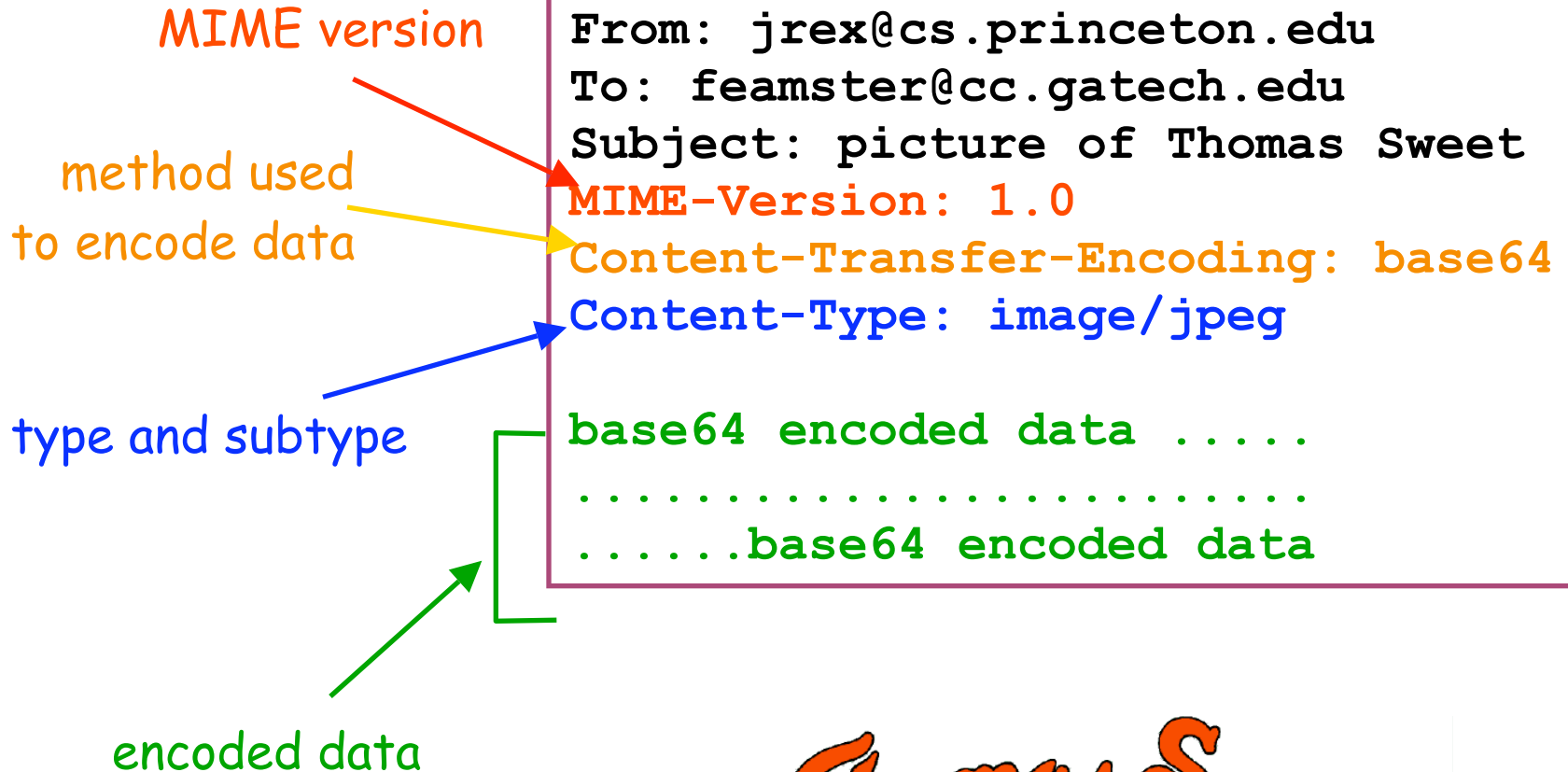
- Users often want to send multiple pieces of data
 - Multiple images, powerpoint files, or e-mail messages
 - Yet, e-mail body is a single, uninterpreted data chunk
- Example: e-mail digests
 - Encapsulating several e-mail messages into one aggregate messages (i.e., a digest)
 - Commonly used on high-volume mailing lists
- Conventions arose for how to delimit the parts
 - E.g., well-known separator strings between the parts
 - Yet, having a standard way to handle this is better

Multipurpose Internet Mail Extensions



- Additional headers to describe the message body
 - MIME-Version: the version of MIME being used
 - Content-Type: the type of data contained in the message
 - Content-Transfer-Encoding: how the data are encoded
- Definitions for a set of content types and subtypes
 - E.g., image with subtypes gif and jpeg
 - E.g., text with subtypes plain, html, and richtext
 - E.g., application with subtypes postscript and msword
 - E.g., multipart for messages with multiple data types
- A way to encode the data in ASCII format
 - Base64 encoding, as in uuencode/uudecode

Example: E-Mail Message Using MIME





Distribution of Content Types

- Content types in my own e-mail archive
 - Searched on “Content-Type”, not case sensitive
 - Extracted the value field, and counted unique types
 - At UNIX command line: `grep -i Content-Type * | cut -d" " -f2 | sort | uniq -c | sort -nr`
- Out of 44343 matches
 - 25531: text/plain
 - 7470: multipart to send attachments
 - 4230: text/html
 - 759: application/pdf
 - 680: application/msword
 - 479: application/octet-stream
 - 292: image (mostly jpeg, and some gif, tiff, and bmp)

Electronic Mailing Lists



- Community of users reachable by one address
 - Allows groups of people to receive the messages
- Exploders
 - Explode a single e-mail message into multiple messages
 - One copy of the message per recipient
- Handling bounced messages
 - Mail may bounce for several reasons
 - E.g., recipient mailbox does not exist; resource limits
- E-mail digests
 - Sending a group of mailing-list messages at once
 - Messages delimited by boundary strings
 - ... or transmitted using multiple/digest format

Conclusions



- **New challenges in data networking**
 - Sensors, intermittent connectivity, long-delay links, ...
 - Require revisiting traditional assumptions
- **Disruption Tolerant Networking (DTN)**
 - Relatively new area of research and standards
 - Many application scenarios with unique properties
- **Electronic mail as an example**
 - Sporadic end-host connectivity
 - Resource constraints on the end host
 - User connecting from different hosts and locations
 - While still relying on the underlying Internet infrastructure