CS419: Computer Networks

Lecture 11: April 27, 2005

Media over Internet

Media over the Internet

CS419

- Media = Voice and Video
- Key characteristic of media:
  - Realtime
  - Which we've chosen to define in terms of *playback*, not "latency over the network"
- A digitized sample of media must be "played back" at a precise time (relative to the previous sample)

Media samples

CS419

- Media is sampled (and played out) at uniform time period
- CD quality audio: 44100 samples per seconds with 16 bits per sample, stereo sound
  - 44100*16*2 = 1.411 Mbps
- Telephone quality voice: 8K samples per second, 8 bits per sample
  - 8000*8 = 64 Kbps

Media samples

- Video
  - For 320*240 images with 24-bit colors
  - 320*240*24 = 230KB/image
  - 15 frames/sec: 15*230KB = 3.456MBps = 27.6 Mbps

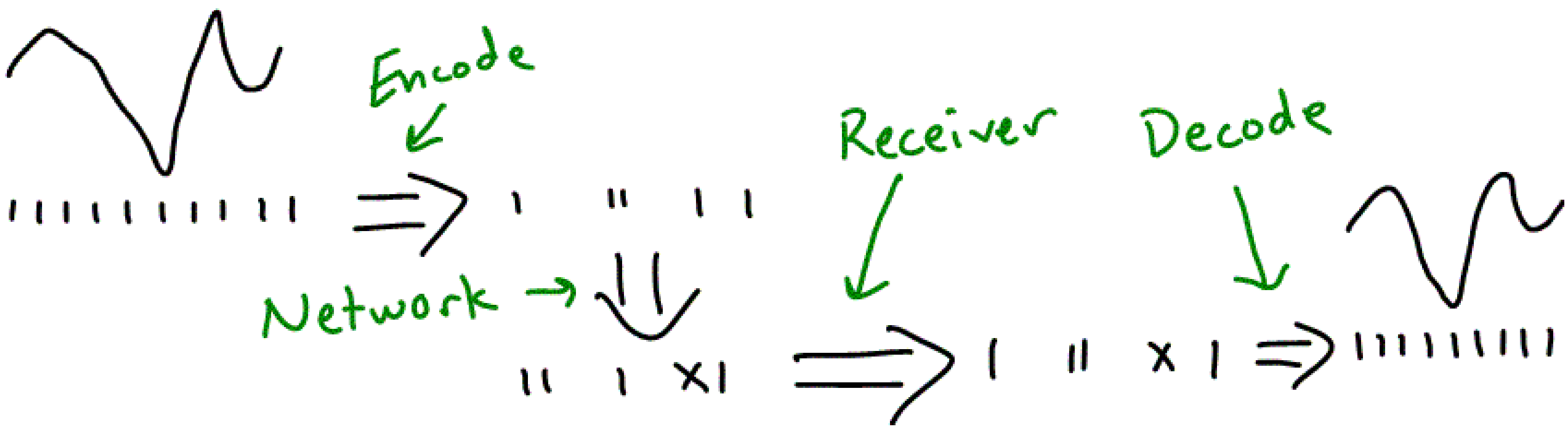
MPEG “compression”

- MP3 audio compression
  - Typical rates are 96kbps, 128kbps, 160kbps
  - From 1.4Mbps: 14.6x, 10.9x, and 8.75x reduction respectively
  - With very little perceived degradation!
- MPEG1 and MPEG2 video compression
  - 1.5Mbps – 6Mbps
  - From 27.6Mbps: 18.4x – 4.6x reduction

What does this compression mean to us?

CS419

- Compressing periodic, fixed-size samples produces:
  - non-periodic, variable-size “units”



It's all about receive buffer…

CS419

- Receiver must reproduce timing of original compressed packets
  - Timing was screwed up by the network (jitter and delay)
- The more we buffer at the receiver, the more jitter we can tolerate
  - Best case: download entire file before playing any of it
  - Worst case: conversational voice
- We mentioned this in QoS lecture . . .

Receive buffer considerations

CS419

- Conversational voice: we can tolerate maybe 250ms latency
  - 150ms or less is better
  - After network delay, 150ms – 200ms buffering
- “Live” media: a few seconds latency ok
- Non-live streaming media: don’t want to wait too long for start of playback

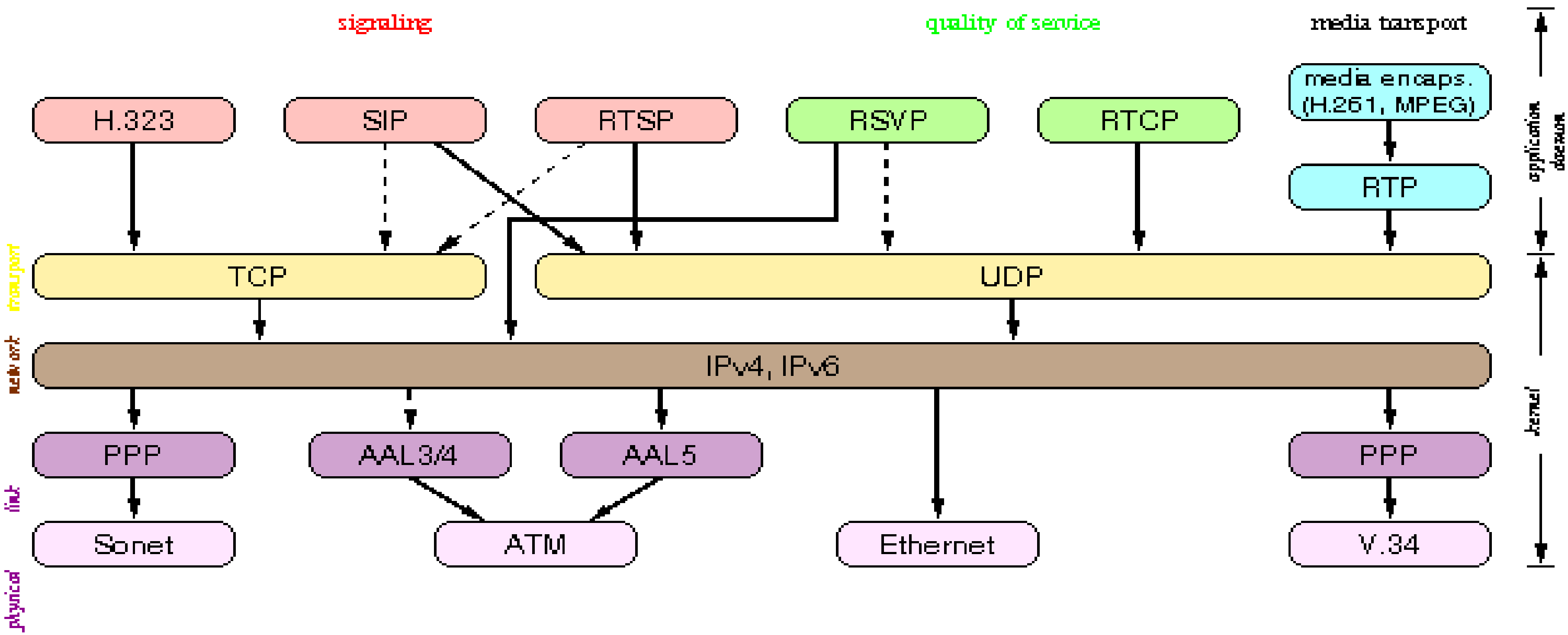
Other realtime considerations

CS419

- In addition to timing and variable size of compression units
- Encoding schemes have different loss tolerance
  - Can use FEC (Forward Error Correction) to an extent
- Some packets better to lose than others
- Encoding schemes may be able to slow down
  - At the expense of quality

Media-related protocols

CS419



Real Time Protocol (RTP) RFC 3550

CS419

- Attempt to provide common transport for many types of media
- In addition to already-stated realtime requirements:
  - Must run over multicast
  - Must allow for “mixing” of streams (i.e. for conferencing)
  - Must be able to combine multiple streams
    - Multi-media, or layered encoding over multiple multicast groups

RTP design approach

- Provide general header with broad capabilities
- Provide separate control protocol for managing RTP stream
  - RTCP: Real Time Control Protocol
- Each encoding type individually specifies how to use RTP

Some RTP usage profiles

CS419

2029 RTP Payload Format of Sun's CellB Video Encoding.
2032 RTP Payload Format for H.261 Video Streams.
2035 RTP Payload Format for JPEG-compressed Video.
2038 RTP Payload Format for MPEG1/MPEG2 Video.
2190 RTP Payload Format for H.263 Video Streams.
2198 RTP Payload for Redundant Audio Data.
2250 RTP Payload Format for MPEG1/MPEG2 Video.
2343 RTP Payload Format for Bundled MPEG.
2429 RTP Payload Format for the 1998 Version of ITU-T Rec. H.263 ...
2431 RTP Payload Format for BT.656 Video Encoding.

More RTP usage profiles

CS419

2435 RTP Payload Format for JPEG-compressed Video.
2658 RTP Payload Format for PureVoice(tm) Audio.
2733 An RTP Payload Format for Generic Forward Error Correction.
2793 RTP Payload for Text Conversation.
2833 RTP Payload for DTMF Digits, Telephony Tones and Telephony...
3016 RTP Payload Format for MPEG-4 Audio/Visual Streams.
3047 RTP Payload Format for ITU-T Recommendation G.722.1.
3119 A More Loss-Tolerant RTP Payload Format for MP3 Audio.
3189 RTP Payload Format for DV (IEC 61834) Video.
3190 RTP Payload Format for 12-bit DAT Audio and 20- and 24-bit...
3389 Real-time Transport Protocol (RTP) Payload for Comfort Noise

RTP header

CS419

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|X|  CC   |M|     PT      |       sequence number         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           timestamp                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           synchronization source (SSRC) identifier            |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|            contributing source (CSRC) identifiers             |
|                             ....                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- version (V)
- padding (P)
- extension (X)

CSRC count (CC)
marker (M)
payload type (PT)

RTP Header

- SSRC identifier
  - Random 32-bit value assigned by the sender
  - Per media stream
  - In multicast, used to distinguish multiple senders
    - RTCP can be used to detect colliding SSRCs
  - Also used to synchronize multi-media streams (image and sound)
    - RTCP announces when SSRCs can be combined

RTP Header

- CSRC: Contributing Source
  - Identifies which sources were combined by a mixer
- Marker: Defined by profile.
  - For example, can indicate frame boundary
- Payload type: some well-known, some defined by profile
  - Indicates type of encoding (MPEG2, MPEG3, etc.)
- Extension: profiles can define their own extension headers

RTP Header: Sequence Number and Timestamp

CS419

- Timestamp indicates when the media should be played back
  - Expressed in units of time defined by the profile
    - e.g., 20 ms block size of 8,000 Hz audio → 160 timestamp units per packet
  - Not absolute time, not “synchronized”
  - Rather, time since initial timestamp
  - Initial timestamp set randomly

RTP Header: Sequence Number and Timestamp

CS419

- Sequence number used to indicate loss and ordering
- Why not use timestamp for this???

Timestamp and talk spurts

CS419

- Receiver does not have to play out packet at exact timestamp time
- In the case of voice (with gaps in between talk spurts)
- Start of talk spurt may vary a little
  - But within a talk spurt, timing must be right
  - Think of a constant C added or subtracted from timestamp during talk spurt
- Why would we do this???

Receive buffer and jitter

- Because of jitter, receive buffer must delay playback of voice a little
  - 10’s of ms
  - More-or-less depending on RTT
  - and on amount of jitter measure over time
- Allows proper playback time even when some packets delayed

Receive buffer and jitter

- Receiver tries to keep a certain amount of voice buffered
  - Enough to recover from jitter
  - But not so much as to introduce too much delay
- If the sender is delayed, the buffer empties a bit
- If the sender is speeded up, the buffer fills a bit
- Either way, the buffer must be brought back to the appropriate size

Receive buffer and jitter

- The receiver can manipulate the buffer by shortening or lengthening the silences between talk spurts
  - As I said, by adding or subtracting a small constant to the timestamp
- If voice and video, must chop out some video to keep lip synch

RTCP: Real Time Control Protocol

CS419

- Runs alongside RTP to control it in various ways
- RTP and RTCP (used to) always run on consecutive port numbers
    - But this was often screwed up by NAT, so SIP allows these numbers to be negotiated individually

RTCP packet types

- SR: Sender report, for transmission and reception statistics from participants that are active senders.
- RR: Receiver report, for reception statistics from participants that are not active senders.
- SDES: Source description items, including CNAME.
- BYE: Indicates end of participation.
- APP: Application specific functions.

Sender Report RTCP Packet (first part)

CS419

```
 0                   1                   2                   3
        0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
header |V=2|P|    RC   |   PT=SR=200   |             length            |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                         SSRC of sender                        |
       +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
sender |              NTP timestamp, most significant word             |
info   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |             NTP timestamp, least significant word             |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                         RTP timestamp                         |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                     sender's packet count                     |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                      sender's octet count                     |
       +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Sender Report RTCP Packet (second part, also RR packet)

CS419

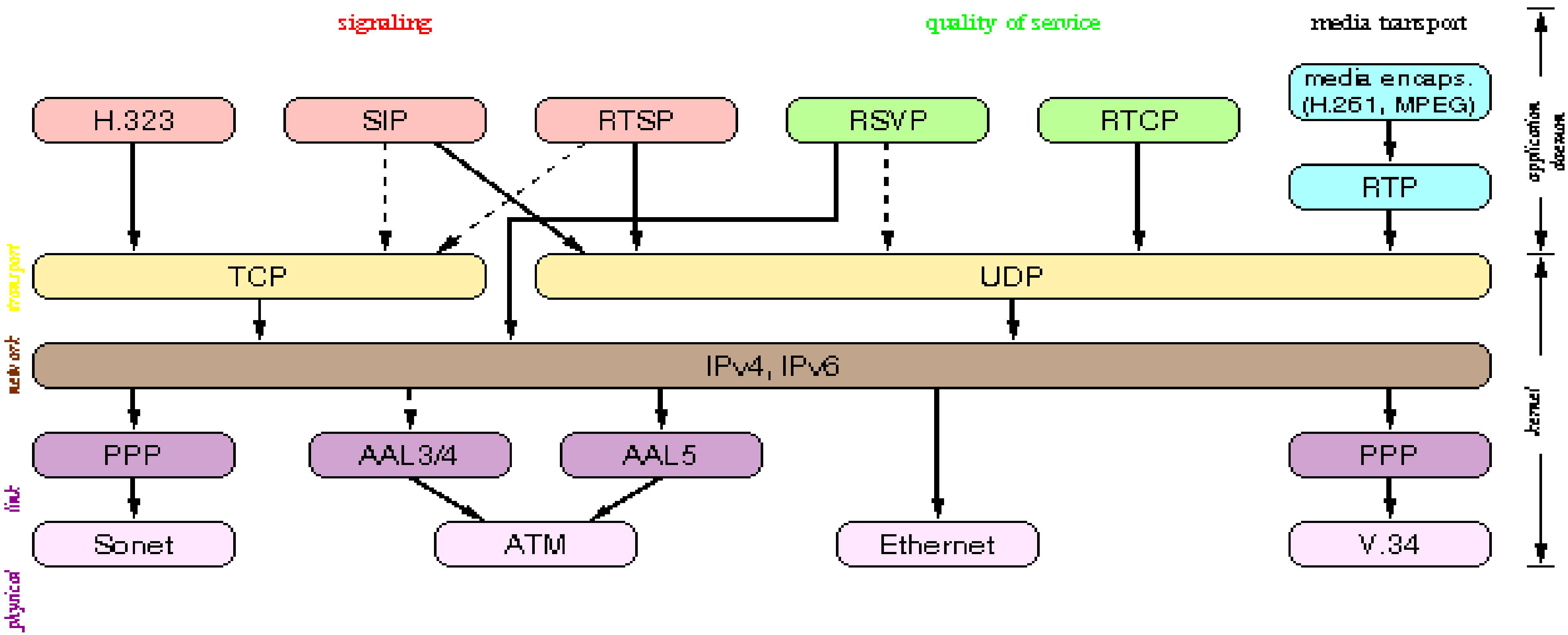
```
        0                   1                   2                   3
        0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
report |                 SSRC_1 (SSRC of first source)                 |
block  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  1    | fraction lost |       cumulative number of packets lost       |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |           extended highest sequence number received           |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                      interarrival jitter                      |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                         last SR (LSR)                         |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                   delay since last SR (DLSR)                  |
       +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
report |                 SSRC_2 (SSRC of second source)                |
block  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  2    :                               ...                             :
       +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Session Initiation Protocol (SIP)

- We've seen how RTP supports a media stream between two or more hosts
- But how did those hosts know to talk in the first place?
  - What ports to use
  - What media stream to use
  - What IP addresses to use
- SIP is one answer

Media-related protocols

CS419



What is SIP?

CS419

- A (formerly) lightweight signaling protocol for IP networks
  - Allows two or more hosts to tell each other what they want to do
  - Way more powerful than simple “ports”, which require a pre-established understanding
- Required for audio/video over IP
  - Because there are many types of audio/video
  - Originally a simple, multicast-aware alternative to H.323
- But has broad applicability
  - Messaging, presence, TCP, etc.

Capabilities of SIP

- Addressing
  - Addresses users or machines
  - user@domain, or +1-234-567-8901
- User location discovery
  - Through registration
- Routing
  - SIP server discovery, redirection
- Signaling
  - Negotiate services, media type, IP type (unicast or multicast), etc.
- Presence and (instant) messaging
  - As SIP “event package” (I.e. application)

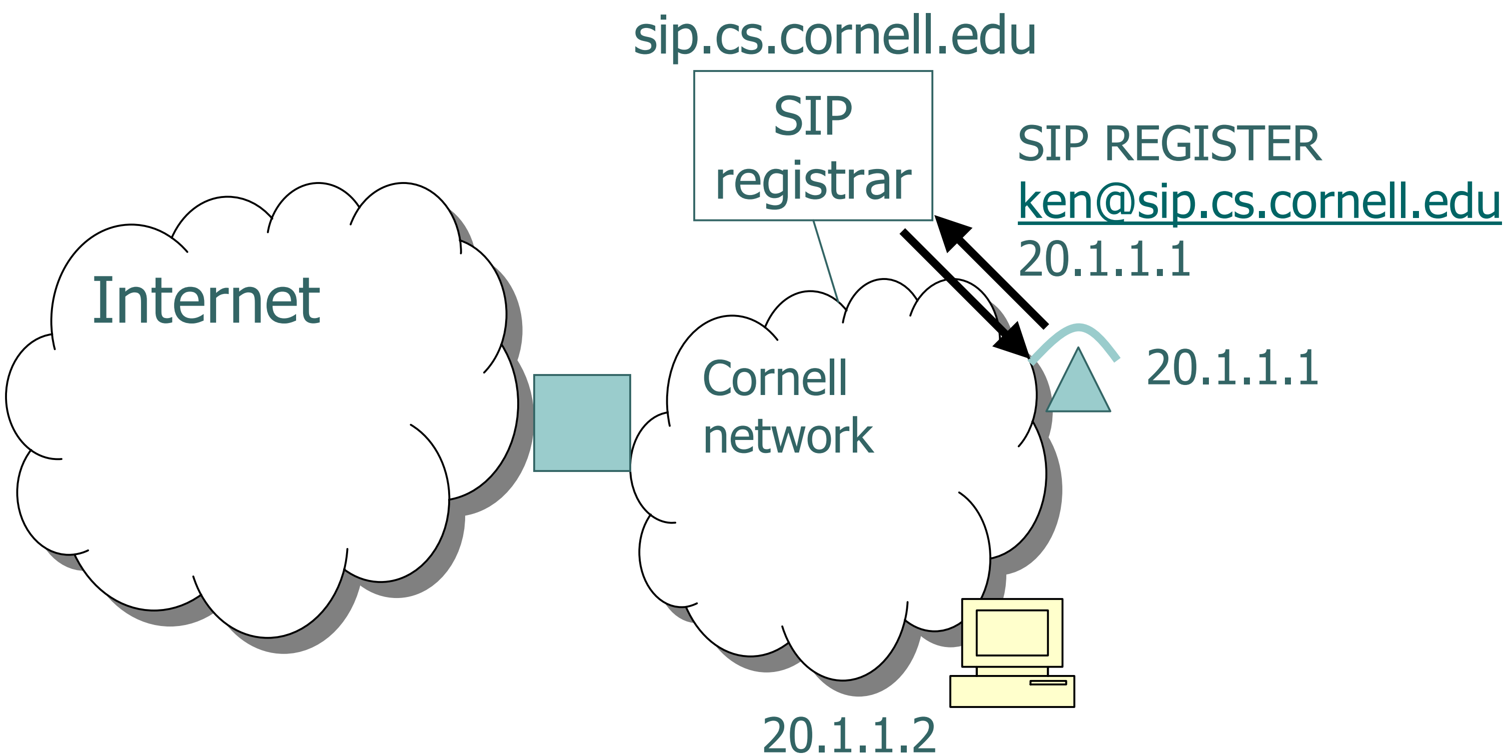
Capabilities of SIP

CS419

- Secure signaling
  - Over TLS
  - Of course, can signal a secure media session, i.e. Secure RTP
- Mobility
  - Of machines across IP (re-INVITE)
  - Of users across machines (REGISTER)
- Service selection
  - Voice, email, fax, messaging, etc.
- “Call” (session) handling
  - Call forward, call transfer, 3rd party conferencing
- Interface with phone network
- NAT traversal (using STUN)

Basic SIP operation

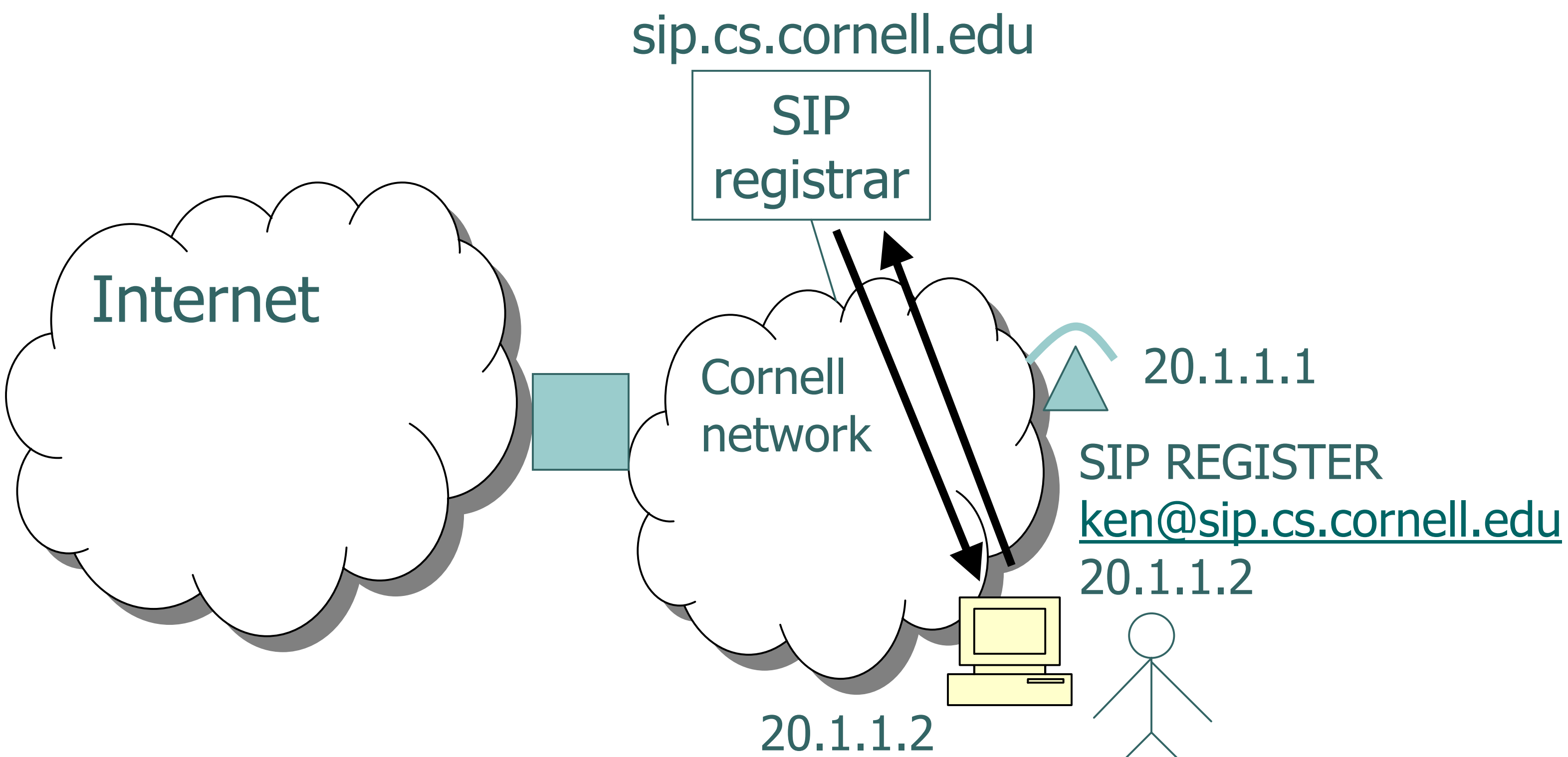
CS419



Ken's VoIP desk phone periodically registers ken@sip.cs.cornell.edu

Basic SIP operation

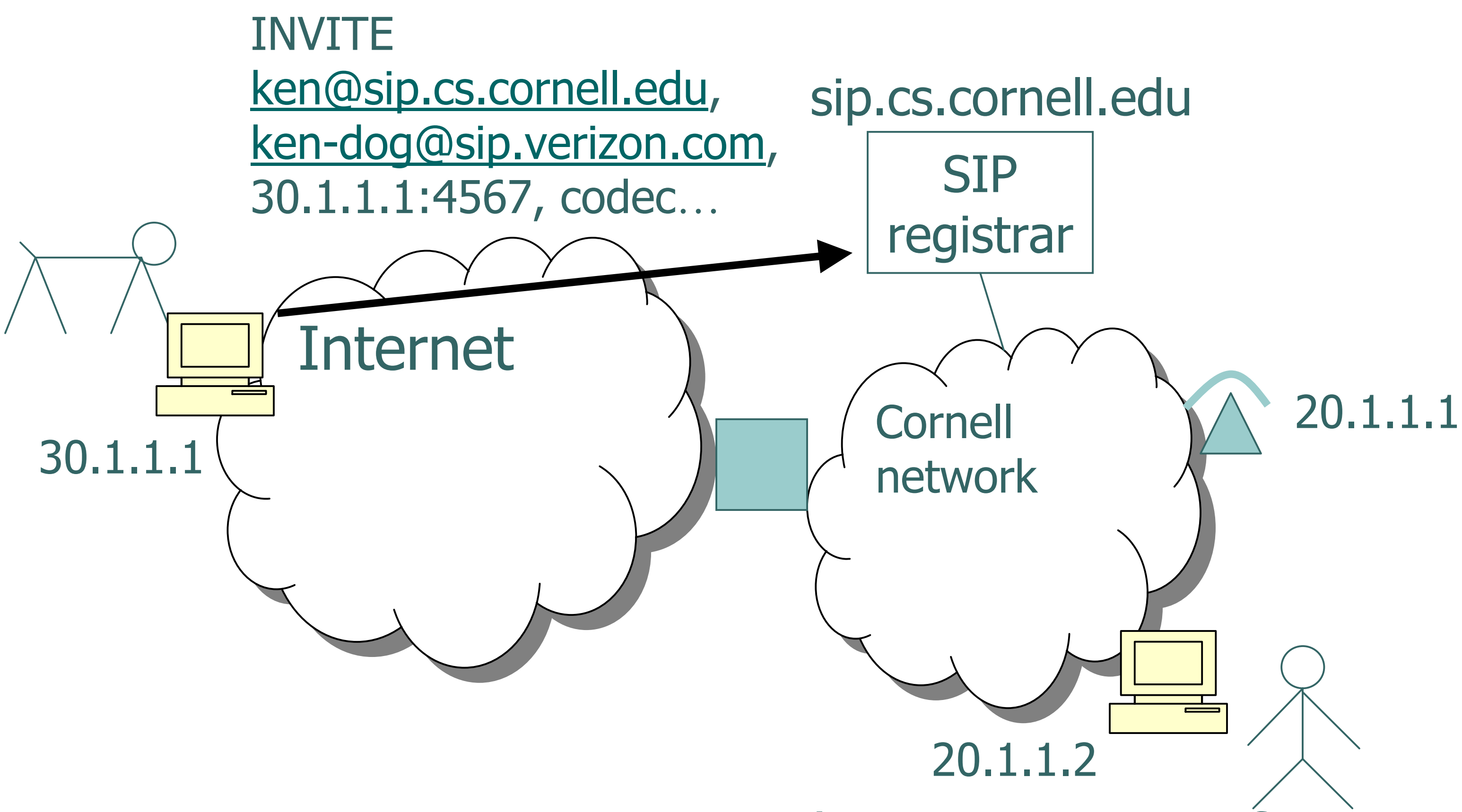
CS419



Ken moves to a computer down the hall, start VoIP app

Basic SIP operation

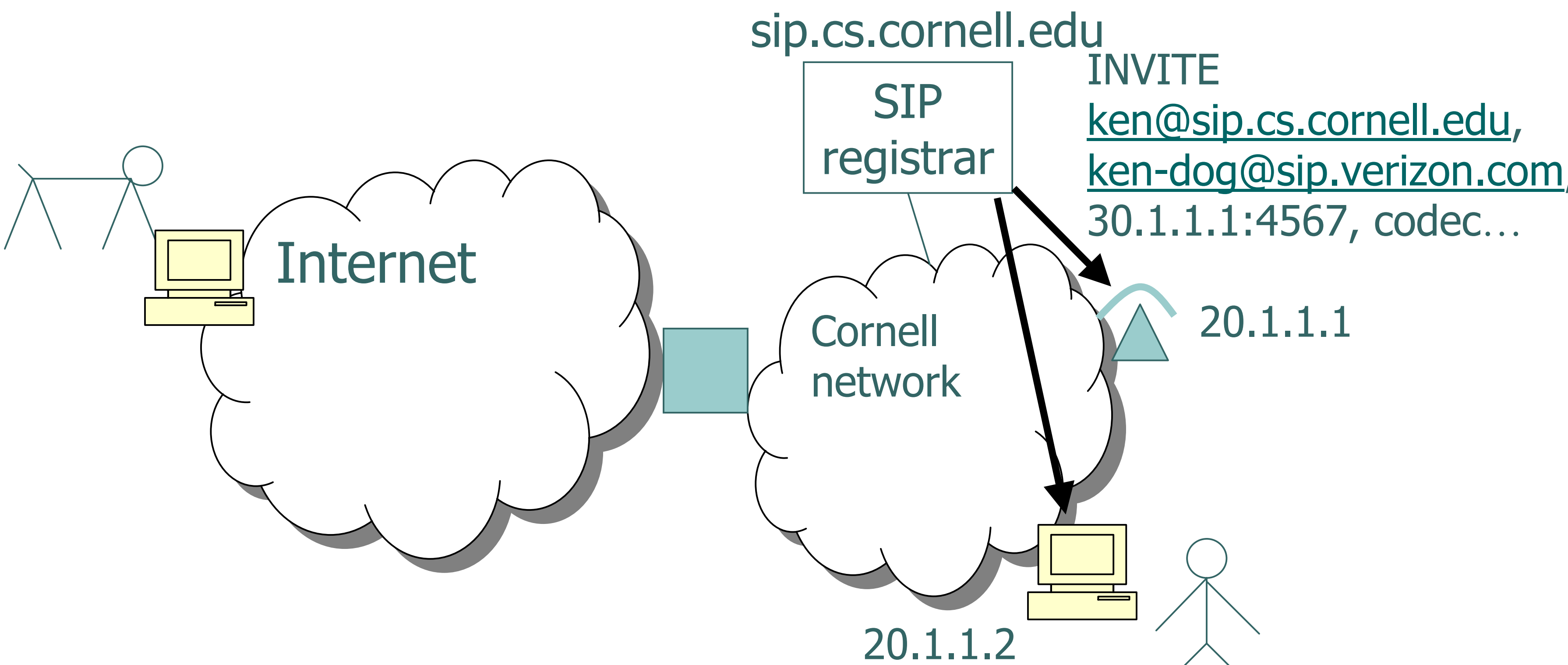
CS419



Ken's dog wants to go for a walk, activates its BoIP phone

Basic SIP operation

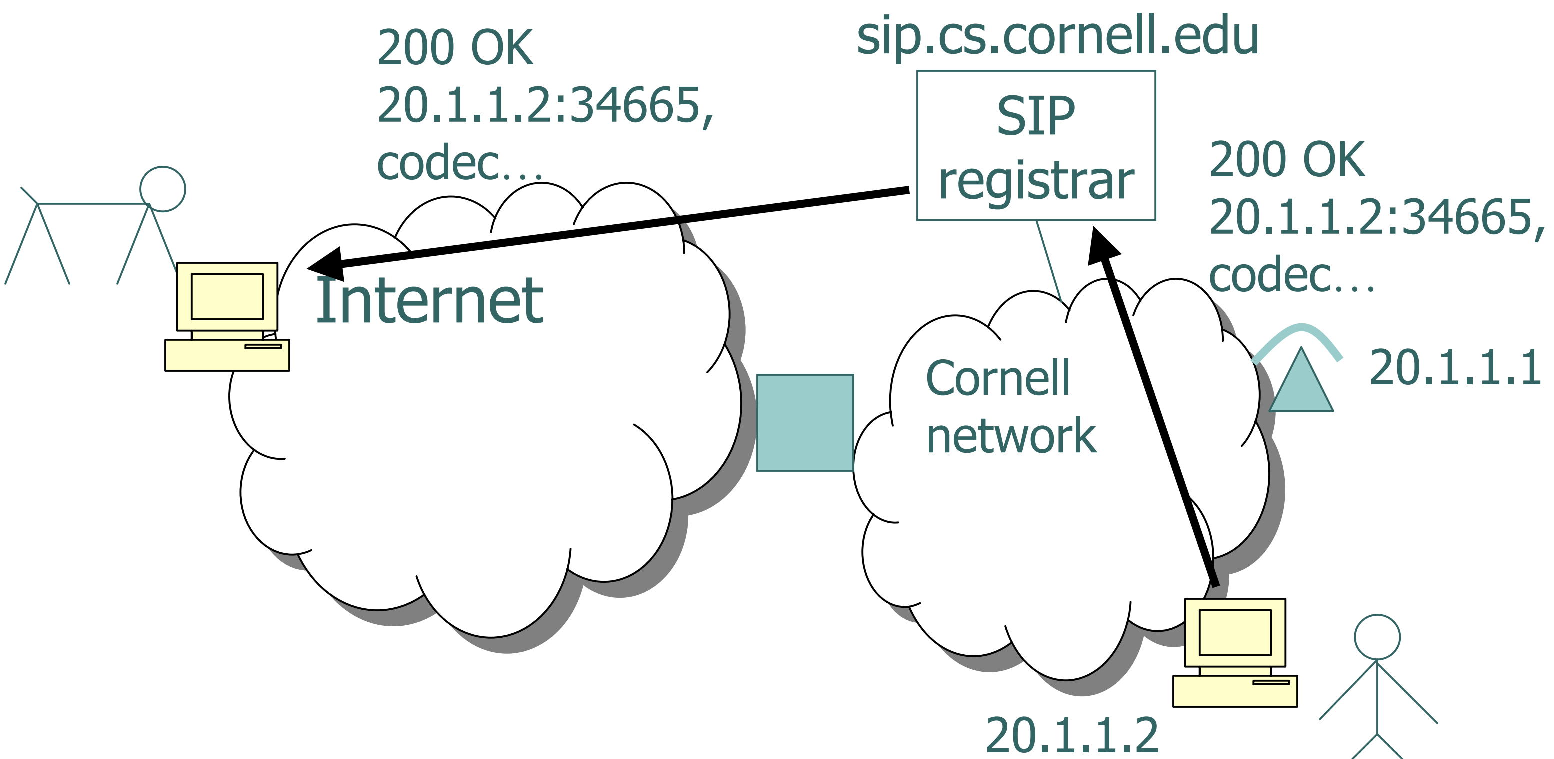
CS419



The SIP registrar forks the INVITE, sends it to both devices

Basic SIP operation

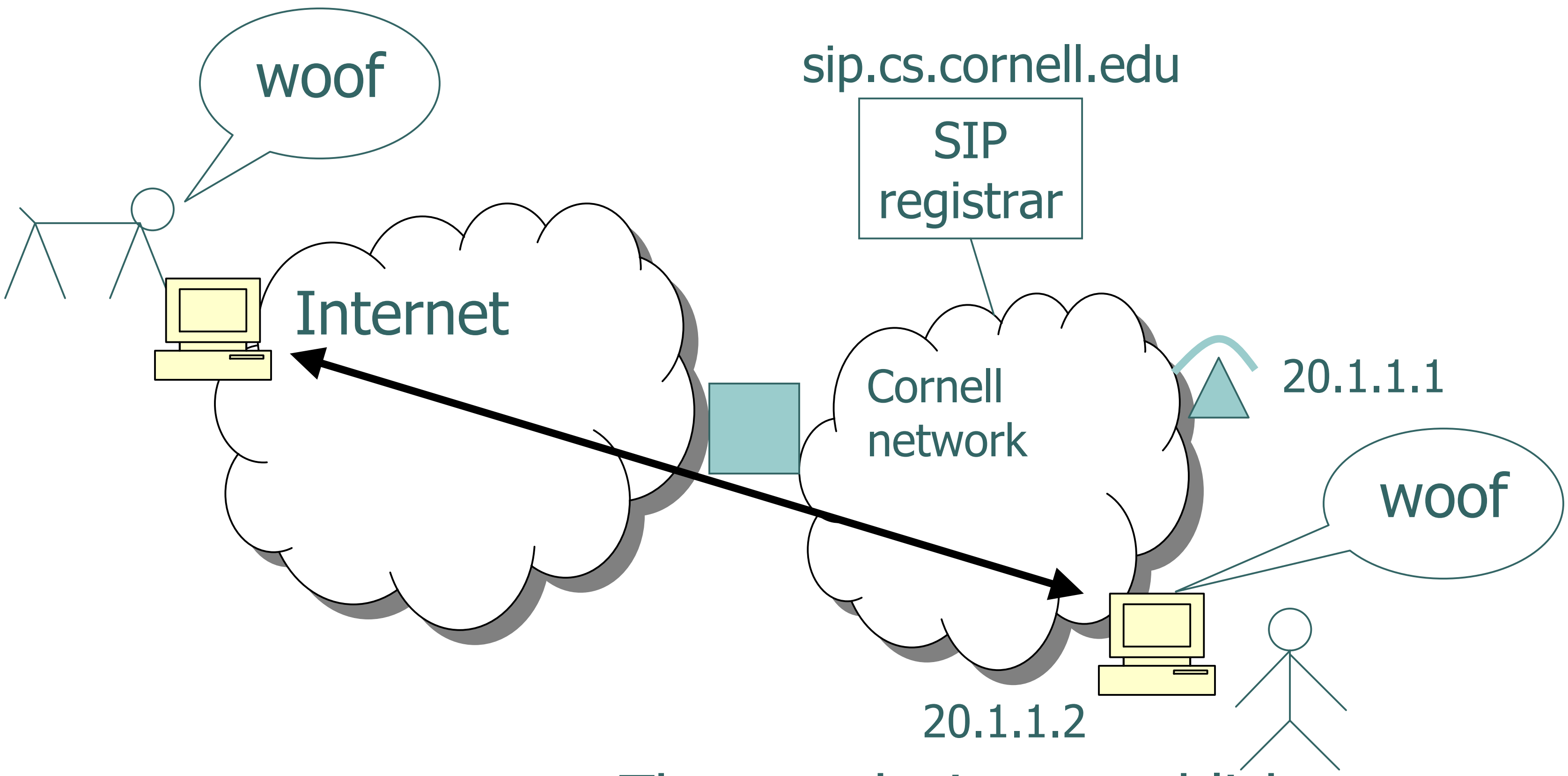
CS419



Ken answers at the computer

Basic SIP operation

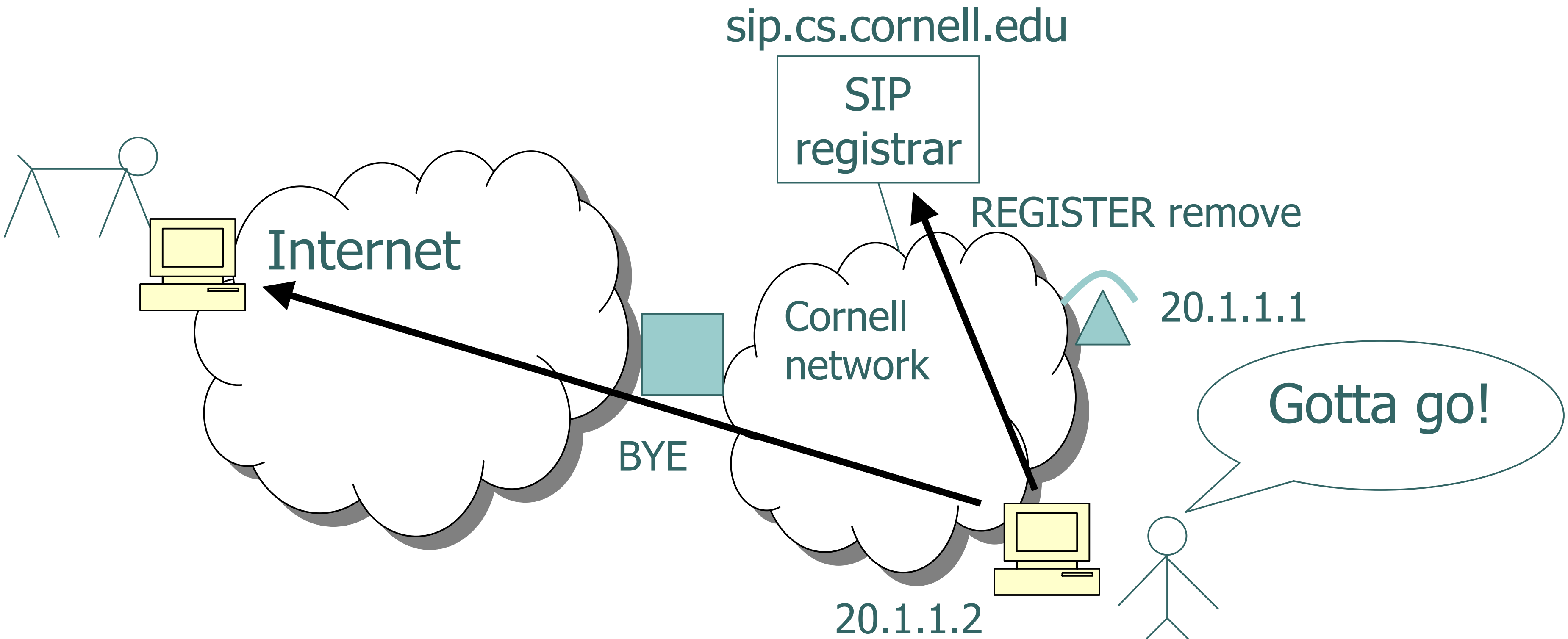
CS419



The two devices establish a media stream over RTP

Basic SIP operation

CS419



Ken hangs up, logs off the computer

SIP methods

- SIP base methods
  - REGISTER, INVITE, ACK, CANCEL, BYE, OPTIONS
- SIMPLE presence methods
  - SUBSCRIBE, NOTIFY
- SIMPLE message method
  - MESSAGE

SIP status

- SIP is taking off
  - In large part because VoIP is taking off
- Microsoft moving to SIP
  - Messenger based on SIMPLE (among other things)
  - VoIP based on SIP
- Unlike IPv6, SIP doesn't have the vicious circle
  - No ISP involvement needed
  - Microsoft can bootstrap SIP all by itself

SIP future (my opinion)

CS419

- Once SIP takes off, every P2P application will be built over it
  - Games, voice, video, chat, voice chat, presence, messaging, file sharing, etc.
  - Because it scales, has security, and allows easier integration of multiple communications channels
  - Example: A web-based help desk will be able to determine what applications you have (through presence, once you approve), and send you web pages, videos, etc., as part of the help service