General Notes on Assignments

Deadlines for Returning

- No fixed deadlines given for assignments 1 and 2
- Use the two week intervals as indicators
  - Just for you to have meaningful milestones and to start early
  - Doing everything in the last week will not work (in most cases)
- Final (real) deadline for Assignment 3: 05.01.2005
  - Short reviews (in groups) probably 2nd week of January
  - For those leaving Finland earlier special arrangements will be made

Notes on Assignment 1

- Test senders
  - **`pc27`**: **`226.226.226.226/62226`**
  - **`solomon`**: **`226.226.226.226/62226`**
  - one packet per second containing some 120 bytes of text and binary data, and a sequence number (both in text and binary)
- If no IP address is given, you may assume IPv4
- Unless you are super-user, you cannot bind to ports < 1024

Questions? Issues?

Assignment 2: udp2rtsp

udp2rtsp

- Receives UDP packets from a specified transport address (command line)
- Works for unicast and multicast addresses (IPv4, optionally IPv6)
- Virtually “any number” of addresses (typically 1 or 2)
- Option: Short and long form for dumping data packets to stdout or file
  - Short output: include SSRC, RTP payload type, timestamp, seq-no, and M bit
- Terminating the program with Ctrl-C (SIGINT) will cause it to dump a summary of the packets received so far.
  - Count missing packets from gaps in RTP sequence number space

- Accept RTSP-based control connection and forward the data packets to a media client
  - Choose one media player, e.g., RealPlayer, QuickTime, mplayer
  - Attention: this will require some trial-and-error testing
  - Will provide additional hints as we go
- Support simple thread of RTSP methods for single run
  - OPTIONS, DESCRIBE, SETUP, PLAY
  - PAUSE, TEARDOWN
- Terminate when the client disconnects

`udp2rtsp -a <addr-spec-audio> -v <addr-spec-video> -i <if-addr> -s -l <dumplen> -f <output-file> -r <addr-spec>`

-a, -v: transport address to receive data on; uses the following format

<IPv4 address>/rtp-port[-rtcp-port][/pt]
/port[-rtcp-port][/pt]
<IPv6 address>/port[-rtcp-port][/pt]
<hostname>/port[rtcp-port][/pt]

May each be specified only once.

-i: address of the local interface to use for listening to multicast packets

-s: packet reports in short form: one line per packet:

reception timestamp (µs!), sender, receiver address, packet size

If “-s” is not specified, the long form is implied. In this case, the above line is followed by a hexdump of (parts of) each packet received:

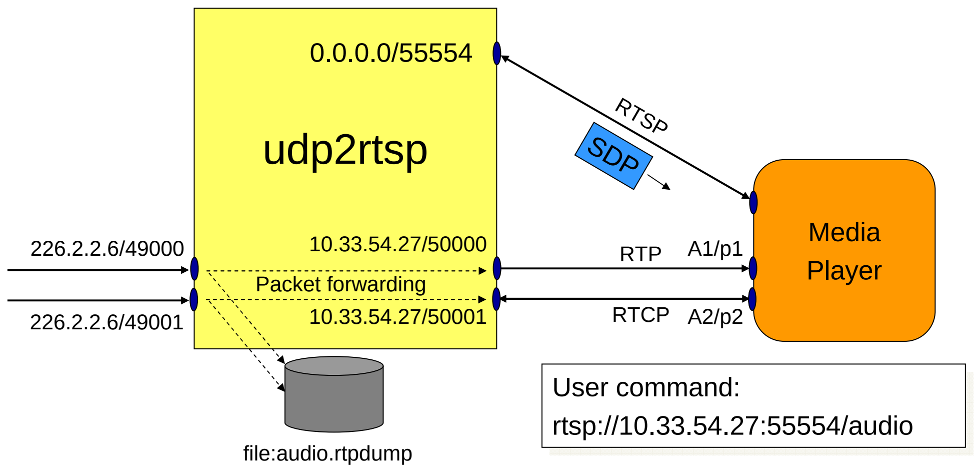
```
000000  xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx    ................
```

-l: Number of bytes to include in the hexdump

-r: transport address to accept RTSP connections

-f: name of the output file to dump to (“-” -> stdout); if not given, be silent

udp2rtsp -a 226.2.2.6/49000-49001 -i 10.33.54.27 -s -f audio.rtpdump -r /55554



Media Stream Control

- RTSP is used to
  - Obtain transport addresses of the client (Transport: header)
  - Obtain information about the media stream (codecs, etc.)
  - Start and stop playing
- SDP is used to describe the media stream
  - Carried in RTSP 200 OK message in response to DESCRIBE

Response to DESCRIBE

```
v=0\r\n
s=This is audio\r\n
o=jo 1234.. 1234.. IN IP4 10.33.54.27\r\n
i=My example audio stream\r\n
t=0 0\r\n
a=recvonly\r\n
c=IN IP4 10.33.54.27\r\n
m=audio 50000 RTP/AVP 0\r\n
```

Response to SETUP

```
RTSP/1.0 200 OK\r\n
CSeq: %d\r\n
Session: %d\r\n
Transport: RTP/AVP;unicast; \
    client_port=p1-p2; \
    server_port=50000-50001\r\n
\r\n
```

Important Hint: Trial and Error!

- Different (versions of) media players behave differently
  - Sometimes not really standards-compliant
- Examples:
  - RealPlayer always sends an OPTION request first
  - RealPlayer sends many SET_PARAMETER requests
  - MS Media Player does not send OPTION
  - Quicktime Player behaves again differently
  - Haven’t checked mplayer yet
- All require quite some experimentation to get it right
  - All suffice to work out the initial RTSP stuff
  - Getting an actual visual or audible result takes longer
  - Further hints to get there to come

Attention: read() from TCP sockets

- Some players send multiple requests and do not wait for the response (e.g., RealPlayer)
  - Need to parse what you read from TCP
    - Also: you may not get all of a (large) request in a single read() system call
  - There MAY be multiple requests received in a single read() system call
  - Example:

```
SET_PARAMETER rtsp://127.0.0.1:554/trailer RTSP/1.0
CSeq: 4
Subscribe: stream=0;rule=0,stream=0;rule=1
Session: 3361875

PLAY rtsp://127.0.0.1:554/trailer RTSP/1.0
CSeq: 5
User-Agent: RealMedia Player (HelixDNAClient)/10.0.0.0 (win32)
Session: 3361875
Range: npt=0-
```

Approach

- Augment your udpspy to understand RTP headers
- Enable packet forwarding
  - Need to turn it on and off even while media is received
  - Note in short output whether packet was forwarded (e.g., “f”)
- Provide an empty container function for RTSP handling

- Write RTSP code separately
  - Test this stand-alone
  - To get the RTSP interaction right you don’t need media streams

- Finally integrate and test
- Test media streams to come…