Sirannon: Demonstration Guide

Alexis Rombaut Nick Vercammen Nicolas Staelens

Brecht Vermeulen Piet Demeester

Ghent University - IBBT, Department of Information Technology (INTEC)
G. Crommenlaan 8, Bus 201
B-9050 Ghent, Belgium
+32 9 331 49 79
{alexis.rombaut, nicolas.staelens, nick.vercammen, brecht.vermeulen, piet.demeester}@intec.ugent.be

1. INTRODUCTION

Sirannon is an open source streamer which aims at being flexible and modular for research and development. For a description about Sirannon itself, refer to the summary paper included in the submission for the Open Source Software Competition [2], the manual included in the distribution or the full paper submitted to ACM Multimedia 2009 [1].

The five following demonstrations illustrate the different capabilities of Sirannon: streaming, video tool or simulation. These demos work with any of the supported containers and codecs listed in the paper describing the program. However, when using user selected content, unforeseen problems could occur due to uncommon extensions of codecs or not fully compliant bit streams. Therefore, three sample containers, *demo.mov*, *demo.avi* and *demo.mpeg*, are available at the website [**?**] and each demo was tested using these three containers. Figures 6, 7 and 8 present the bit rate evolution of these samples. Copy the different "demo*n*.xml" files located under "dat/xml" in the distribution to the directory where the sample containers are located and invoke the program from here.

In addition, as media player the use of the VLC Media Player [3] for network playback is highly recommended. In order to open a network stream in VLC, go to *File* > *Open Network Stream* in older versions of VLC or *Media* > *Open Network* in the latest version and select the proper protocol, typically RTP (Real-time Transport Protocol [6]) or RTSP (Real Time Streaming Protocol [7]) in the following demonstrations. Sometimes artifacts occur due to starting the playback in the middle of the stream or overflow of the receive buffers which then VLC dynamically increases.

Finally, to view each demo configuration in the user interface, run: `$ python interface/sirannon.py`

ACM Multimedia 2009 '09 Beijing, China

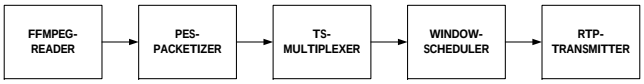


Figure 1: Basic streamer using MPEG Transport Streams sent over RTP.

2. DEMONSTRATIONS

2.1 Basic streaming

The first demo illustrates a simple streaming solution using RTP (see figure 1). The use of MPEG Transport Streams (MPEG-TS [4]) provides a codec independent packetization, making the configuration suitable for different containers and codecs. VLC assumes by default that content received over RTP is multiplexed using transport streams and will play it without additional measures. The demo configuration takes two additional arguments: the path of the container you wish to stream and the destination address.

- `$ sirannon demo1.xml demo.mpeg 127.0.0.1`
- Launch VLC, in the network tab select the RTP protocol using port 1234 (default).
- The content should start playing instantaneously.
- The configuration sends the sample in a continuous loop, therefore, kill the program in the shell to end the demo.

2.2 RTSP

In addition, Sirannon supports RTSP which allows managing a collection of requested streams sent over RTP (see figure 2). The RTSP server can offer multiple content and handle as many concurrent streams as allowed by the processing power of the server machine. The demo configuration takes two additional arguments, firstly the path to the folder which contains the streams to offer and secondly the RTSP server port (should the default port 554 be unavailable).

- `$ sirannon demo2.xml .  5554`
- Launch VLC, in the network tab select the RTSP protocol and use for example rtsp://localhost:5554/FILE/demo.mpeg
- The content should start playing instantaneously.
- Test the concurrent streaming by requesting additional streams using new instances of VLC.

Figure 2: Trivial graph containing just the component *RTSP-server* which dynamically creates a series of nested components for each requested stream.

- End each stream by closing VLC or pressing stop.
- Kill the program in the shell to end the demo.

2.3 Impairment

This demo shows Sirannon can function as a tool for example to analyse and impair a stream (see figure 3). The configuration generates an elementary video stream. To ensure VLC recognizes this raw format, use the extensions ".m1v", ".m2v", ".m4v" and ".264" in the file name, respectively for the codecs MPEG-1, MPEG-2, MPEG-4 and H.264/AVC. The demo configuration takes four additional arguments: the path of the source container, the path of the generated elementary stream, the packet loss for I-frames and the packet loss for P/B-frames. In addition, random packet loss, especially for increasingly higher probabilities, can corrupt a stream in such ways the player crashes during playback.

- `$ sirannon demo3.xml demo.mpeg demo.m1v 0.001 0.005`
- The program impairs the stream as fast as possible and terminates when the container is fully processed.
- Test the results of the impairment by playing the generated elementary stream in a player such as VLC.
- Experiment by changing the values of the packet loss and watching the newly generated stream.

2.4 Differentiated streaming

Sirannon allows sending a stream using multiple transport mechanism (see figure 4a), which distinguishes itself from other streamers. These multiple links allow for novel and experimental transport mechanisms. As demonstration, the configuration uses three types of mechanisms: RTP, UDP and TCP. However, standard players cannot handle these multiple links, requiring a second Sirannon configuration, acting as proxy to rejoin the multiple links into a single link which a standard player supports (see figure 4b). The first configuration takes as extra argument the path of the container you wish to stream and the destination address.

- proxy: `$ sirannon demo4B.xml 127.0.0.1`
- Open an additional shell and execute the following line in the shell:
- server: `$ sirannon demo4A.xml demo.mpeg 127.0.0.1`
- Launch VLC, in the network tab select the UDP protocol using port 1234 (default).
- The content should start playing instantaneously.
- Kill the program in both shells to end the demo. If you kill the streamer first, the proxy will produce buffer underflow warnings since it receives no more data.

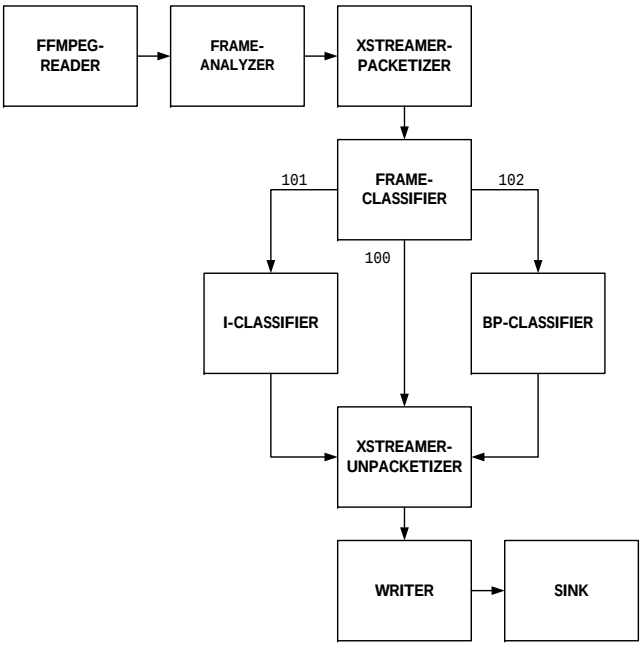


Figure 3: Impairment configuration, *frame-analyzer* parses the frame type from each video frame while *frame-classifier* splits the stream in a stream of I-frames and a stream of P/B-frames and *Sirannon-packetizer* provides a codec independent packetization.

2.5 Simulation

Finally, Sirannon can function as a simulator, for example simulating hours of streaming in mere minutes. This demo simulates the streaming of three streams concurrently through a basic buffer and evaluates the packet loss due to buffer overflows and the average delay of the buffer. By writing your own custom buffer, novel or experimental buffers can be simulated. The configuration takes 7 additional arguments: path of the nested configuration, path of the first, second and third container, number of loops, buffer rate (in kbps) and buffer size (in packets).

- `$ sirannon demo5A.xml demo5B.xml demo5B.xml demo.mov demo.mpeg demo.avi 1 8000 80`
- Play with the values of the buffer size and buffer rate to evaluate the effects on the average delay and packet loss.

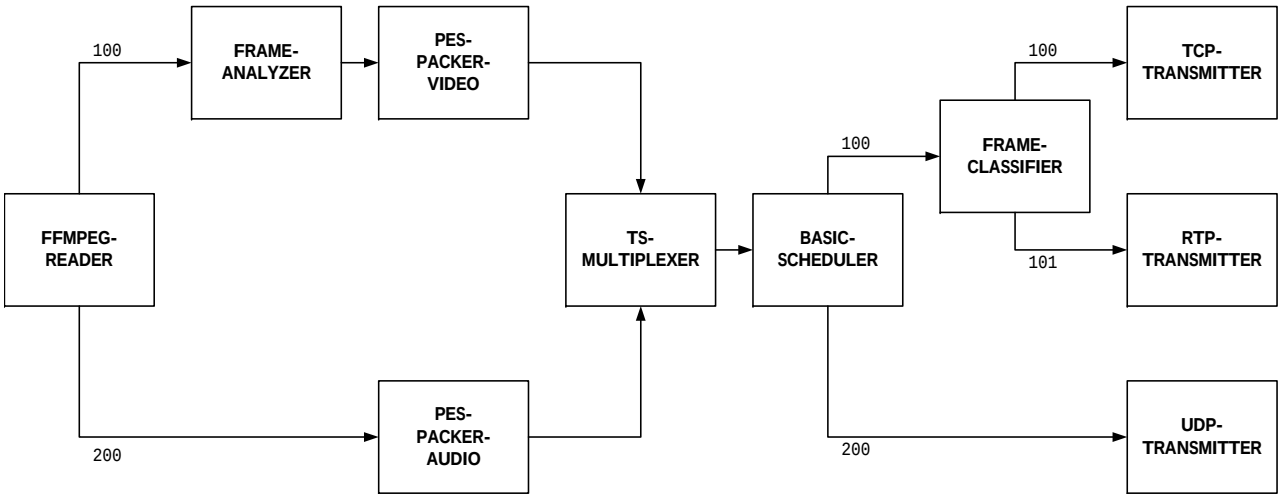
3. REFERENCES

[1] ACM Multimedia 2009. `http://www.acmmm09.org/default.aspx`, October 2009.

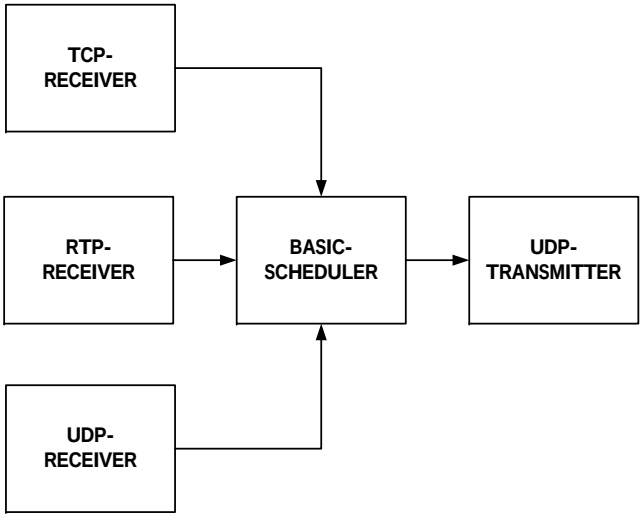
[2] ACM Multimedia 2009, Open Source Software Competition. `http://www.acmmm09.org/OSSC.aspx`, 2009.

[3] L. Aimar, G. Bazin, et al. VLC media player. `http://www.videolan.org/vlc/`, 2009.

[4] ITU-T and ISO/IEC JTC 1. Generic coding of moving pictures and associated audio information: systems, October 2007. ITU-T Rec. H.222 and ISO/IEC 11172-1 (MPEG TS).



(a) Differentiated streamer



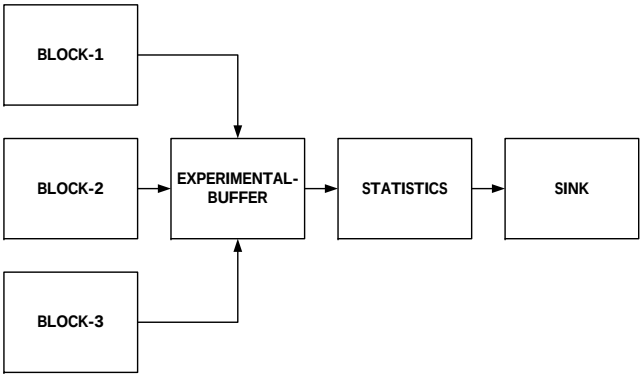
(b) Proxy

Figure 4: Differentiated streaming, in (a) using three different types of transport mechanisms, RTP, UDP and TCP, with the TCP connection handling the I-frames, the RTP link the P/B-frames and the UDP link the audio. In (b) the proxy rejoins the differentiated streams into a single UDP link, receivable and playable by a standard media player.

[5] A. Rombaut. xStreamer: Modular Multimedia Streaming. `http://xstreamer.atlantis.ugent.be/demo.tar.bz2`, 2009.

[6] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. `http://www.ietf.org/rfc/rfc1889.txt`, January 1996. RFC 1889.

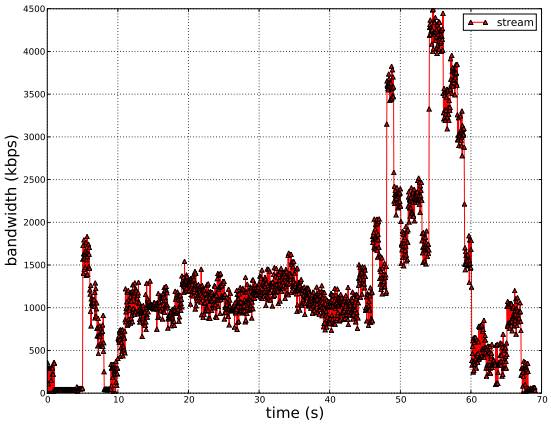
[7] H. Schulzrinne, U. Columbia, A. Rao, and R. Lanphier. Real Time Streaming Protocol (RTSP). `http://www.ietf.org/rfc/rfc2326.txt`, April 1998. RFC 2326.



(a) Simulator

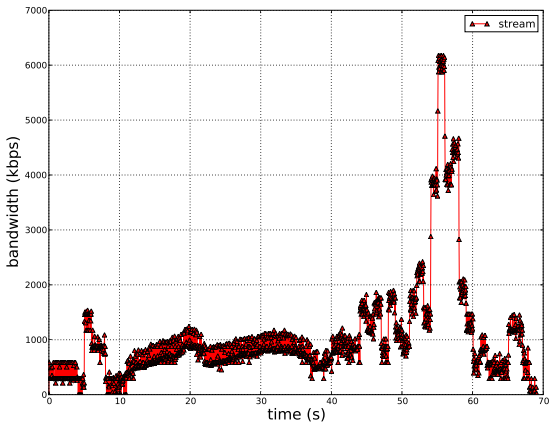
(b) Nested configuration

Figure 5: Simulating network buffers, in (a) three streams are sent through one experimental buffer with the component *statistics* measuring the packet loss and average delay of the buffer over a long period. (b) shows the nested streamer configuration used in each component *block* from figure 5a.



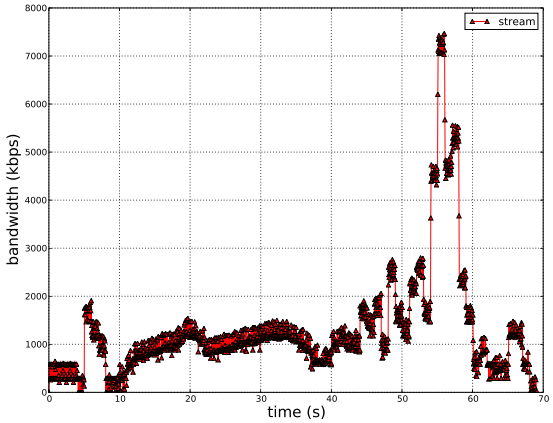
	mean	min	max	sigma	total
size (B)	754	21	1470	581	10367k
bit rate (kbps)	1248	3	4378	923	

Figure 6: Bit rate evolution of *demo.mov* (using a window of 1000 ms)



	mean	min	max	sigma	total
size (B)	1290	21	1470	388	8863k
bit rate (kbps)	1086	17	6084	1007	

Figure 7: Bit rate evolution of *demo.mov* (using a window of 1000 ms)



	mean	min	max	sigma	total
size (B)	1347	22	1470	367	10954k
bit rate (kbps)	1331	8	7307	1190	

Figure 8: Bit rate evolution of *demo.mpeg* (using a window of 1000 ms)