2014

Transmission de données multimédia

TP3 FLUX VIDEO ET AUDIO

Guénon Marie / Favreau Jean-Dominique / Tanguy Arnaud

POLYTECH4 NICE SOPHIA ANTIPOLIS |Master SSTIM / VIM

Table des matières

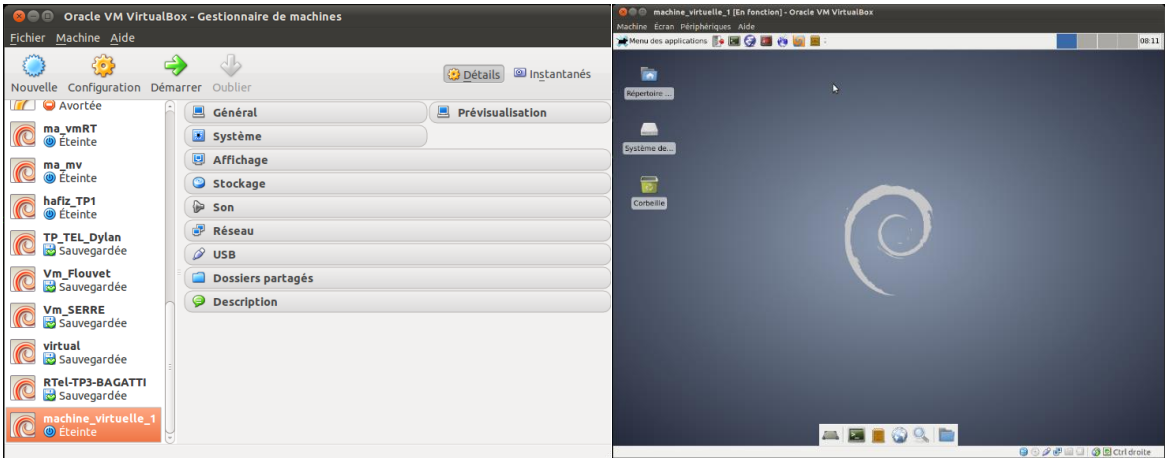
Initialisation 2
1. La salle 310 2
2. Logiciels 2
Flux d'image 3
1. Caméra 3
Avec un navigateur 3
Avec gstreamer 4
Décomposition du flux 4
Ce flux en réseau 5
2. Un autre streamin 6
Avec decodebin 6
On décompose plus finement 6
Sauvegarde directe 6
Sauvegarde et visualisation 7
3. Fichier 8
Sons 9
1. Fichier 9
Lecture 9
Envoi UDP 9
Envoi RTP/UDP 9
Vidéo 10

Initialisation

1. La salle 310

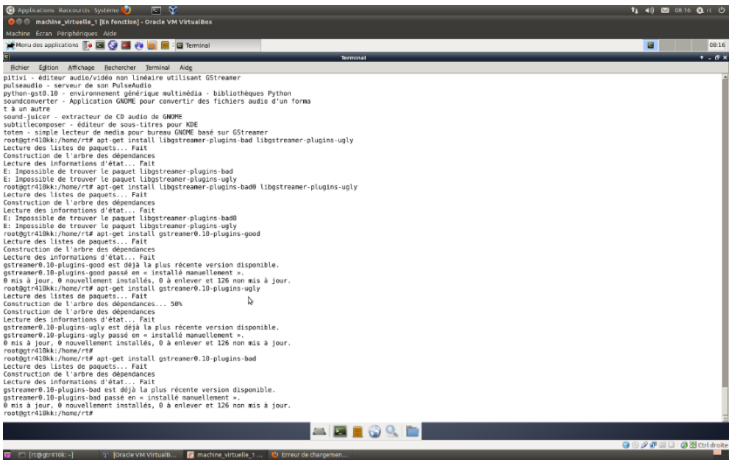
Grâce à la commande *creatvm* nous créons une machine virtuelle :

Puis nous lançons la machine virtuelle crée précédement.



2. Logiciels

Sur la machine virtuelle fraichement crée, les logiciels dont nous allons avoir besoin par la suite ne sont pas à jour. Nous avons donc dû les mettre à jour ainsi qu'installer les logiciels non présents, tels que *vlc, gstreamer*,…



Flux d'image

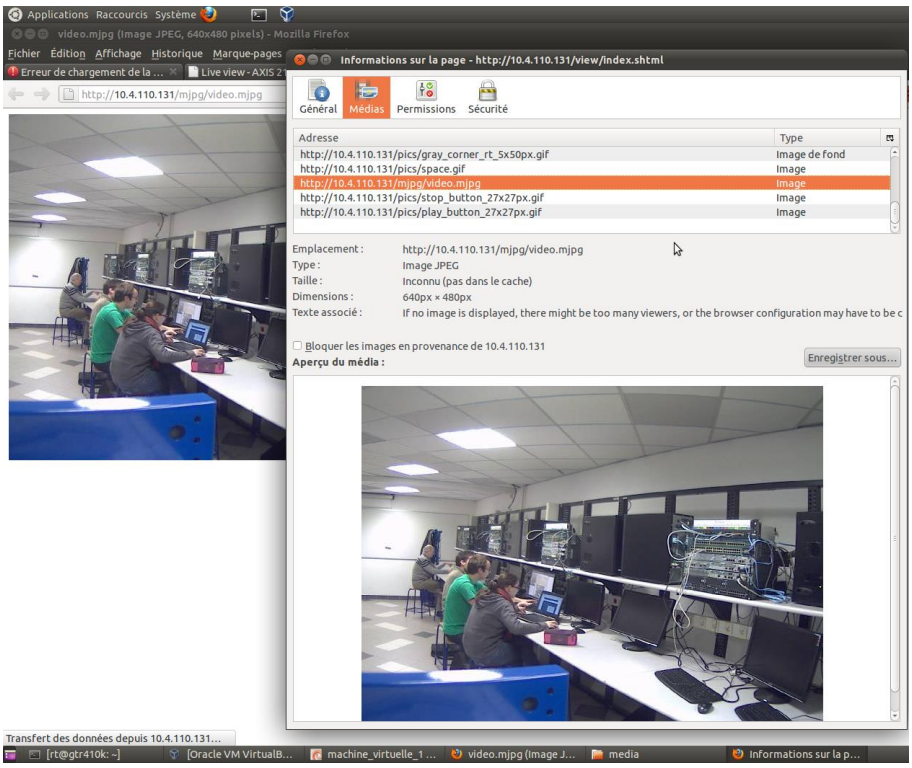
1. Caméra

Une fois la machine virtuelle lancée et à jour, nous avons eu à récupérer et afficher les images fournies en temps réel par la caméra Axis. Pour cela nous avons utilisé deux méthodes :

Avec un navigateur

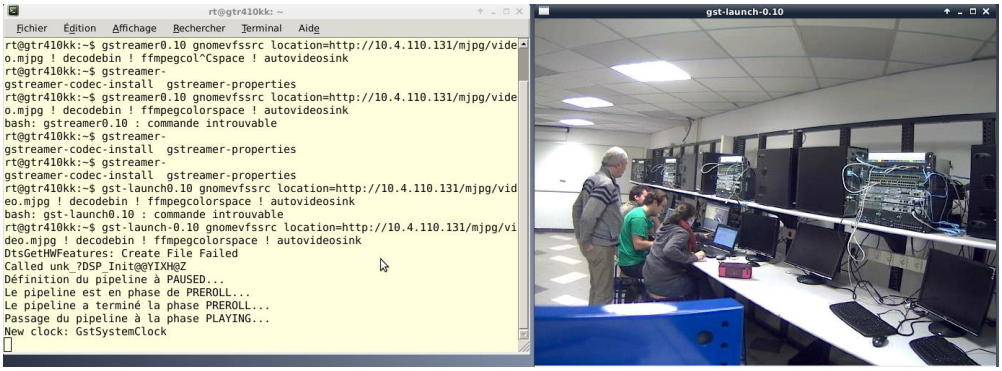
Notre première méthode a été de récupérer les images envoyées par la caméra grâce un navigateur web. Pour cela, nous pouvons accéder à deux adresses différentes et récupérer deux flux de type différents.

- A l'adresse http://10.4.110.131/mjpg/video.mjpg, nous récupérons un flux encapsulé en MJPEG. C'est-à-dire une vidéo.
- A l'adresse rtps://10.4.110.131:554/mpeg4/media.amp, nous récupérons un flux encapsulé en MPEG4. C'est-à-dire une suite d'images statiques affichées et rafraichies suffisamment souvent pour que l'œil humain croie à des mouvements fluides.



Avec gstreamer

La deuxième méthode que nous avons utilisée consistait à utiliser la commande *gstreamer* pour construire une chaîne qui traite notre flux vidéo en agençant les traitements les uns après les autres.



Décomposition du flux

Lorsque l'on décompose la commande *gstreamer*, on se rend compte que le(s) plugin(s) utilisé(s) est (sont) *jpegdec0.*

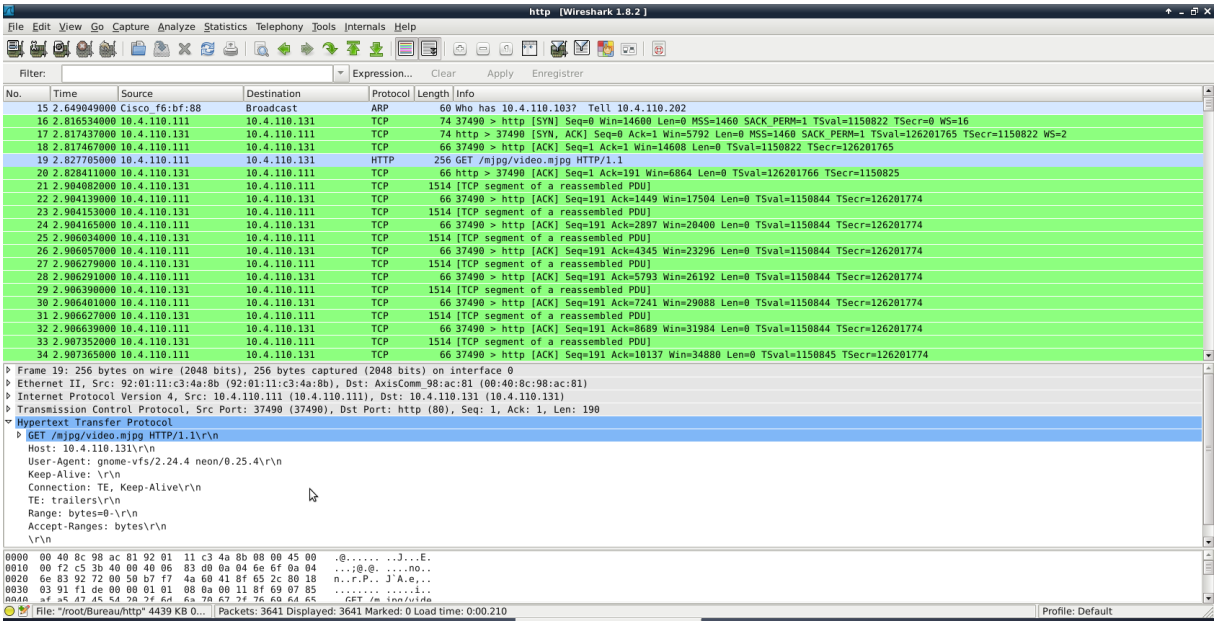
ligne de commande spécifique :

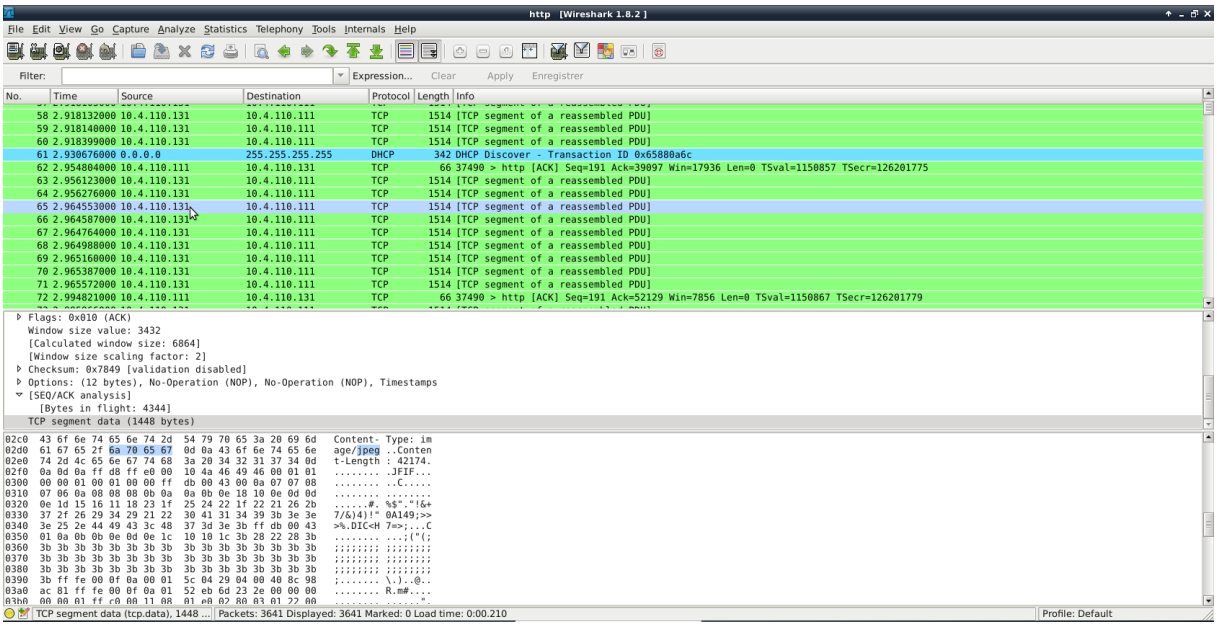
De plus, nous avons pu voir que les images envoyées par la caméra sont des images jpeg codées en YUV (luminance/chrominance). Ceci est lié au fait que ce sont les variations sur ces paramètres qui sont le mieux perçues par l'œil humain. Au contraire, les images sont reconverties pour être affichées en RGB, car c'est ce type d'affichage qui est le mieux géré par l'ordinateur.

```
rt@gtr410kk:~$ gst-launch-0.10 -v gnomevfssrc location=http://10.4.110.131/mjpg/video.mjpg ! decodebin ! ffmpegcolorspace ! autovideosink
Définition du pipeline à PAUSED...
Le pipeline est en phase de PREROLL...
/GstPipeline:pipeline0/GstDecodeBin:decodebin0/GstTypeFindElement:typefind.GstPad:src: caps = multipart/x-mixed-replace
/GstPipeline:pipeline0/GstDecodeBin:decodebin0/GstMultipartDemux:multipartdemux0.GstPad:sink: caps = multipart/x-mixed-replace
/GstPipeline:pipeline0/GstDecodeBin:decodebin0/GstQueue:queue0.GstPad:sink: caps = image/jpeg
/GstPipeline:pipeline0/GstDecodeBin:decodebin0/GstQueue:queue0.GstPad:src: caps = image/jpeg
/GstPipeline:pipeline0/GstDecodeBin:decodebin0/GstJpegDec:jpegdec0.GstPad:sink: caps = image/jpeg
/GstPipeline:pipeline0/GstDecodeBin:decodebin0.GstGhostPad:src0: caps = video/x-raw-yuv, format=(fourcc)I420, width=(int)640, height=(int)480, framerate=(fraction)0/1
/GstPipeline:pipeline0/GstDecodeBin:decodebin0/GstJpegDec:jpegdec0.GstPad:src: caps = video/x-raw-yuv, format=(fourcc)I420, width=(int)640, height=(int)480, framerate=(fraction)0/1
/GstPipeline:pipeline0/GstFFMpegCsp:ffmpegcsp0.GstPad:src: caps = video/x-raw-rgb, bpp=(int)32, depth=(int)24, endianness=(int)4321, red_mask=(int)65280, green_mask=(int)16711680, blue
_mask=(int)-16777216, width=(int)640, height=(int)480, framerate=(fraction)0/1, pixel-aspect-ratio=(fraction)1/1
/GstPipeline:pipeline0/GstFFMpegCsp:ffmpegcsp0.GstPad:sink: caps = video/x-raw-yuv, format=(fourcc)I420, width=(int)640, height=(int)480, framerate=(fraction)0/1
/GstPipeline:pipeline0/GstDecodeBin:decodebin0.GstGhostPad:src0.GstProxyPad:proxypad2: caps = video/x-raw-yuv, format=(fourcc)I420, width=(int)640, height=(int)480, framerate=(fraction
)0/1
/GstPipeline:pipeline0/GstAutoVideoSink:autovideosink0/GstXImageSink:autovideosink0-actual-sink-ximage.GstPad:sink: caps = video/x-raw-rgb, bpp=(int)32, depth=(int)24, endianness=(int)
4321, red_mask=(int)65280, green_mask=(int)16711680, blue_mask=(int)-16777216, width=(int)640, height=(int)480, framerate=(fraction)0/1, pixel-aspect-ratio=(fraction)1/1
/GstPipeline:pipeline0/GstAutoVideoSink:autovideosink0.GstGhostPad:sink: caps = video/x-raw-rgb, bpp=(int)32, depth=(int)24, endianness=(int)4321, red_mask=(int)65280, green_mask=(int)
16711680, blue_mask=(int)-16777216, width=(int)640, height=(int)480, framerate=(fraction)0/1, pixel-aspect-ratio=(fraction)1/1
/GstPipeline:pipeline0/GstAutoVideoSink:autovideosink0.GstGhostPad:sink.GstProxyPad:proxypad1: caps = video/x-raw-rgb, bpp=(int)32, depth=(int)24, endianness=(int)4321, red_mask=(int)6
5280, green_mask=(int)16711680, blue_mask=(int)-16777216, width=(int)640, height=(int)480, framerate=(fraction)0/1, pixel-aspect-ratio=(fraction)1/1
Le pipeline a terminé la phase PREROLL...
Passage du pipeline à la phase PLAYING...
New clock: GstSystemClock
```

Ce flux en réseau

Le but était ici d'analyser les protocoles utilisés pour envoyer la vidéo de la caméra à l'ordinateur.
Nous avons pu constater que la connexion est initialisée par une requête HTTP, après quoi les frames en jpeg sont envoyées par TCP.

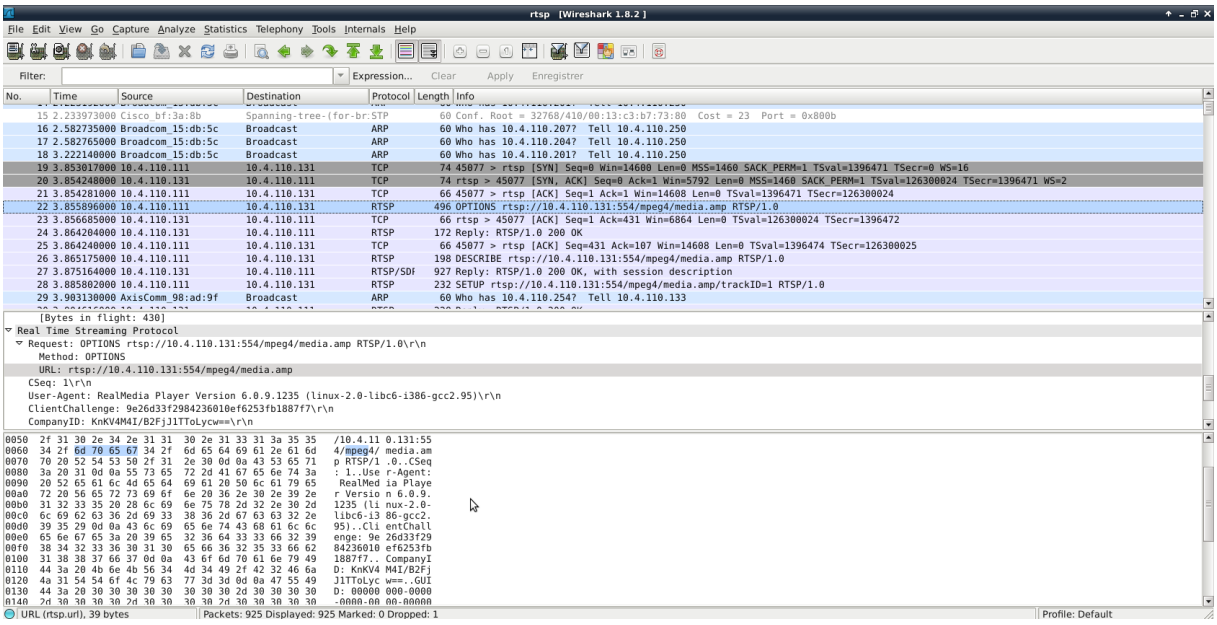




2. Un autre streaming

Avec decodebin

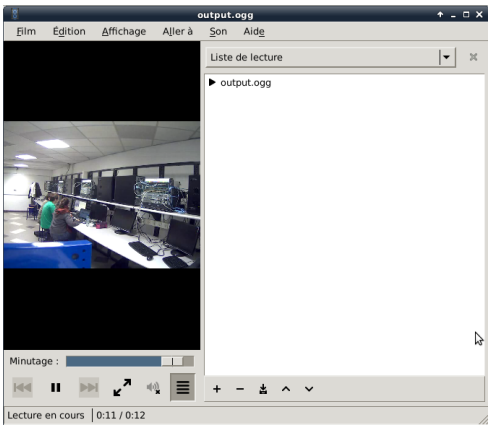
Nous avons lancé la lecture de la vidéo avec *gstreamer* et avec le codec(?) *decodebin*, puis nous avons analysé les flux envoyés grâce à *wireshark*.



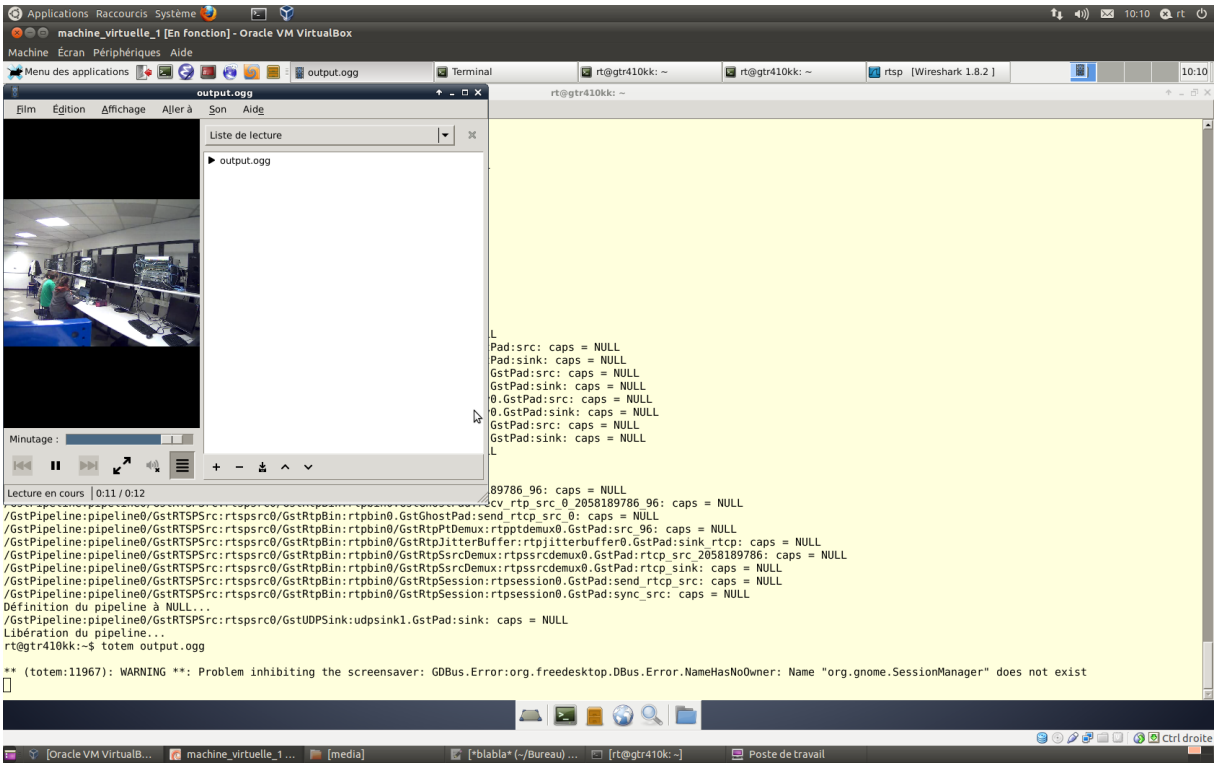
Nous pouvons par ailleurs constater que le transport utilisé ici est le protocole RTP.

On décompose plus finement

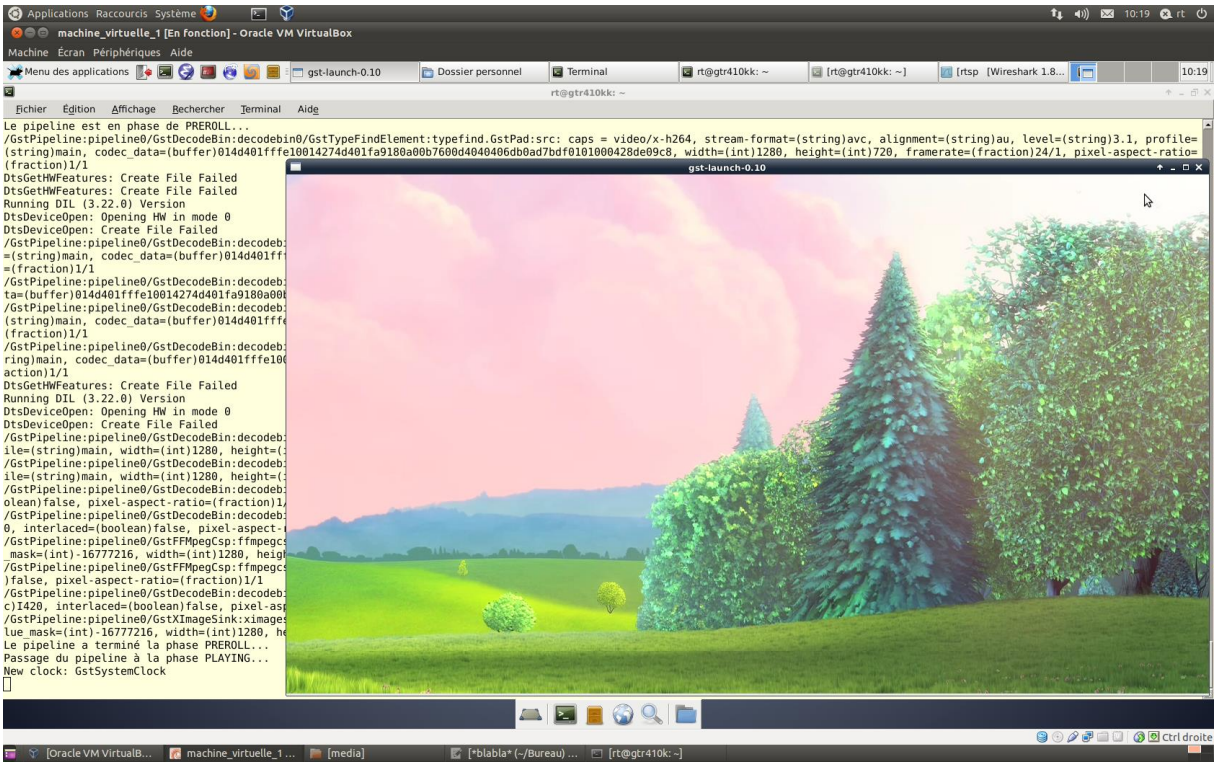
Sauvegarde directe



Sauvegarde et visualisation



3. Fichier


Applications Raccourcis Système
machine_virtuelle_1 [En fonction] - Oracle VM VirtualBox
Machine Écran Périphériques Aide
Menu des applications
gst-launch-0.10
Dossier personnel
Terminal
rt@gtr410kk: ~
[rtsp [Wireshark 1.8...
Fichier Édition Affichage Rechercher Terminal Aide
Le pipeline est en phase de PREROLL...
DtsGetHWFeatures: Create File Failed
DtsGetHWFeatures: Create File Failed
Running DIL (3.22.0) Version
DtsDeviceOpen: Opening HW in mode 0
DtsDeviceOpen: Create File Failed
DtsGetHWFeatures: Create File Failed
Running DIL (3.22.0) Version
DtsDeviceOpen: Opening HW in mode 0
DtsDeviceOpen: Create File Failed
Le pipeline a terminé la phase PREROLL...
Passage du pipeline à la phase PLAYING...
New clock: GstSystemClock
gst-launch-0.10
[Oracle VM VirtualB...
machine_virtuelle_1 ...
[media]
[*blabla* (~/Bureau) ...
[rt@gtr410k: ~]

Sons

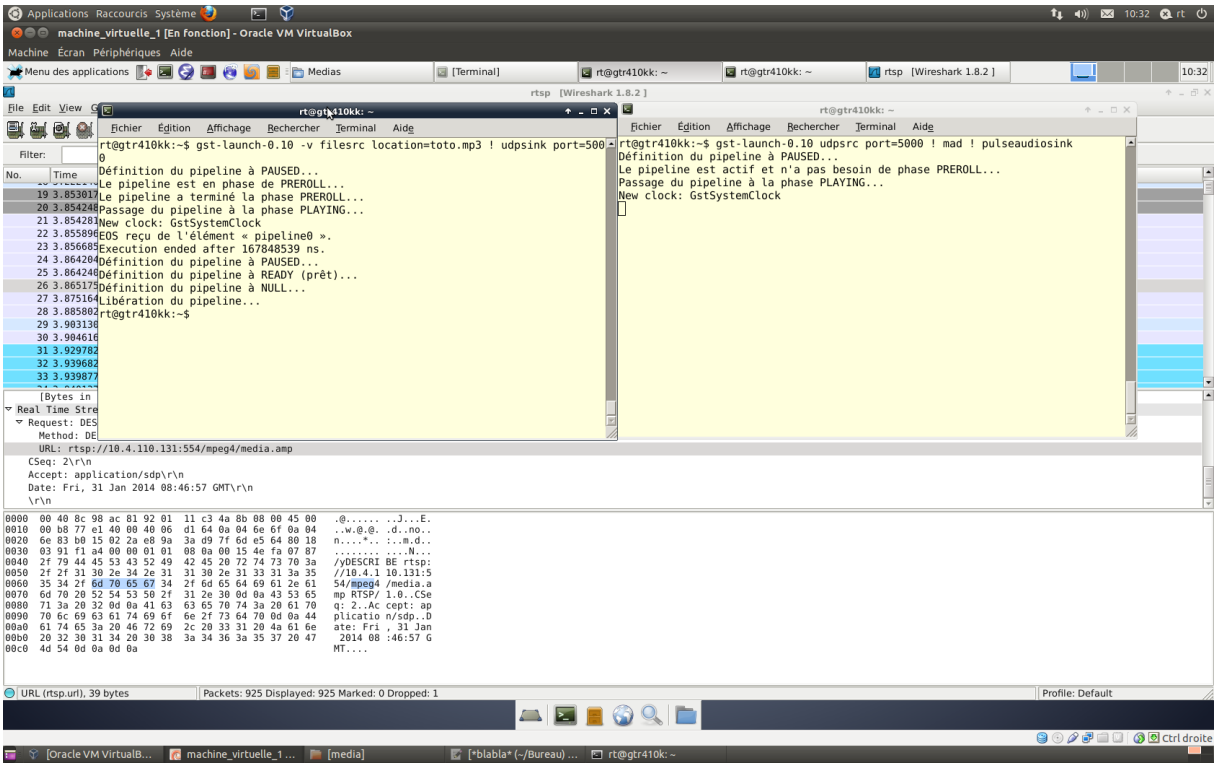
1. Fichier

Lecture

Envoi UDP

Attention, il faut lancer la réception avant l'envoi (à l'inverse, le son n'a rien pour être reçu et n'est donc pas entendu)

Mauvaise qualité audio



Envoi RTP/UDP

Vidéo