**ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ**

**ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ**
**ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ**

**ΔΙΑΤΜΗΜΑΤΙΚΟ ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ ΣΤΟΝ ΗΛΕΚΤΡΟΝΙΚΟ ΑΥΤΟΜΑΤΙΣΜΟ**

**ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ**

# ΠΟΛΥΜΕΣΙΚΕΣ ΕΦΑΡΜΟΓΕΣ ΣΤΟ ΥΠΟΛΟΓΙΣΤΙΚΟ ΝΕΦΟΣ: ΕΙΚΟΝΙΚΟΠΟΙΗΣΗ ΤΗΣ ΟΙΚΙΑΚΗΣ ΠΥΛΗΣ

**ΣΤΑΥΡΟΣ Μ. ΜΠΑΛΑΣΗΣ**

**Επιβλέπων** **Ευστάθιος Χατζηευθυμάδης,** Αναπληρωτής Καθηγητής

**ΑΘΗΝΑ**

**ΑΠΡΙΛΙΟΣ 2017**

**ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ**

# ΠΟΛΥΜΕΣΙΚΕΣ ΕΦΑΡΜΟΓΕΣ ΣΤΟ ΥΠΟΛΟΓΙΣΤΙΚΟ ΝΕΦΟΣ: ΕΙΚΟΝΙΚΟΠΟΙΗΣΗ ΤΗΣ ΟΙΚΙΑΚΗΣ ΠΥΛΗΣ

**Σταύρος Μ. Μπαλάσης**

**Α.Μ.:** 2012526

**ΕΠΙΒΛΕΠΩΝ** **Ευστάθιος Χατζηευθυμιάδης,** Αναπληρωτής Καθηγητής

**ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ:** **Λάζαρος Μεράκος,** Καθηγητής
**Ευστάθιος Χατζηευθυμιάδης,** Αναπληρωτής Καθηγητής
**Άννα Τσανακάκη,** Επίκουρη Καθηγήτρια

Απρίλιος 2017

# ΠΕΡΙΛΗΨΗ

Σκοπός της παρούσας διπλωματικής είναι η υλοποίηση της εικονικοποίησης του οικιακού εξοπλισμού για τη μεταφορά, τροποποίηση και αποθήκευση πολυμεσικών ροών. Για την κατασκευή της υποδομής χρησιμοποιήθηκε το εργαλείο ανάπτυξης υπολογιστικού νέφους OpenStack ενώ για τη διαχείριση των πολυμεσικών ροών η βιβλιοθήκη gstreamer. Το τελικό αποτέλεσμα εξετάζεται τόσο στην δικτυακή του απόδοση όσο και στον καταμερισμό των υπολογιστικών πόρων μεταξύ των 3 κόμβων.

**ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ**: Υπολογιστικό νέφος, εικονικοποίηση
**ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ**: OpenStack, Network Function Virtualization, πολυμέσα, IaaS, IoT

## ABSTRACT

The main aim of this thesis is the virtualization of the customer premises that manage, process and transfer multimedia streams. The infrastructure design is implemented with OpenStack, a free and open source cloud deploy platform. The streaming media handling is implemented through the gstreamer framework. The final result is examined for its network efficiency but also for its computational distribution between the three nodes.



**SUBJECT AREA**: Cloud, Virtualization
**KEYWORDS**: OpenStack, Network Function Virtualization, multimedia, IaaS, IoT

## ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω θερμά τον αναπληρωτή καθηγητή του τμήματος Πληροφορικής & Τηλεπικοινωνιών κ. Ευστάθιο Χατζηευθυμιάδη για την πολύτιμη καθοδήγηση του και την άριστη συνεργασία που είχαμε, στα πλαίσια της εκπόνησης της διπλωματικής μου εργασίας στο Μεταπτυχιακό Δίπλωμα Εξειδίκευσης στον Ηλεκτρονικό Αυτοματισμό. Επίσης θα ήθελα να ευχαριστήσω τη Μαρία και την οικογένεια μου για τη στήριξη που έδειξαν κατά τη διάρκεια της υλοποίησης και εν γένει των σπουδών μου.

# ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

# Πίνακας Εικόνων

## ΠΡΟΛΟΓΟΣ

Σκοπός αυτής της εργασίας είναι η αποτίμηση των ωφελιμάτων από την εικονικοποίηση του οικιακού περιβάλλοντος στον τομέα των πολυμέσων. Η συγγραφή τόσο του θεωρητικού μέρους όσο και του πηγαίου κώδικα έγιναν στην Αθήνα ενώ η εκπόνηση αυτής είναι στα πλαίσια του Διατμηματικού Προγράμματος Μεταπτυχιακών Σπουδών στον Ηλεκτρονικό Αυτοματισμό με σκοπό την απόκτηση του Μεταπτυχιακού Διπλώματος.

# 1. ΕΙΣΑΓΩΓΗ

Στη σημερινή εποχή, οι ανάγκες της συνδεσιμότητας και φορητότητας αυξάνονται ολοένα και περισσότερο. Σχέδον όλες οι υπηρεσίες προσφέρουν ορισμένο αποθηκευτικό χώρο σε όλους τους χρήστες τους στο υπολογιστικό σύννεφο είτε για αποθήκευση, επεξεργασία ή αποστολή. Τα οφέλη από αυτή την ενέργεια εντοπίζονται τόσο στον προμηθευτή - κατασκευαστή της υπηρεσίας, όσο και στον τελικό χρήστη.Από τη μετάβαση στο υπολογιστικό σύννεφο δεν λείπουν και οι πολυμεσικές δραστηριότητες, δηλαδή εφαρμογές οπτικοακουστικού περιεχομένου.

Σκοπός αυτής της εργασίας είναι να σκιαγραφήσει την υλοποίηση ενός σεναρίου όπως περιγράφεται από το Ευρωπαϊκό Ινστιτούτο Τηλεπικοινωνιακών Προτύπων[1] για την εικονικοποίηση δικτυακών λειτουργιών στα πλαίσια του οικιακού περιβάλλοντος[2]. Τελικό αποτέλεσμα είναι η εικονικοποίηση της οικιακής πύλης αλλά και η κατασκευή της δικτύωσης που θα μπορούσε να την φιλοξενήσει χωρίς επίπτωση στις παρεχόμενες υπηρεσίες του πελάτη και προσφέροντας του την φορητότητα/συνδεσιμότητα ελλατώνοντας τον εξοπλισμό του και αυξάνοντας παράλληλα την πολυπλοκότητα στο υπολογιστικό σύννεφο.

Στο πρώτο τμήμα της παρούσας εργασίας γίνεται παρουσίαση του θεωρητικού μοντέλου πίσω από το οποίο στηρίχθηκε η κατασκευή του εικονικοποιημένου οικιακού περιβάλλοντος,των δικτυακών πρωτοκόλλων που χρησιμοποιούνται για την μεταφορά του πολυμεσικού περιεχομένου, των εργαλείων για την ανάπτυξη του υπολογιστικού νέφους πάνω στο οποίο στήθηκε η εικονικοποιημένη πύλη και η περιγραφή των βιβλιοθηκών για την συγγραφή του προγραμματιστικού κώδικα.

Στο δεύτερο τμήμα αναλύεται η υλοποίηση της δικτύωσης και η κατασκεύη της πλατφόρμας υπολογιστικού νέφους με χρήση του εργαλείου OpenStack καθώς επίσης και τα αρχεία με τους πηγαίους κώδικες για κάθε κόμβο του δικτύου.

# 2. ΕΙΚΟΝΙΚΟΠΟΙΗΣΗ ΤΟΥ ΟΙΚΙΑΚΟΥ ΠΕΡΙΒΑΛΛΟΝΤΟΣ

Εξ'ορισμού, η εικονικοποίηση περιγράφει έναν αφαιρετικό μηχανισμό που αναγκάζει ένα υπολογιστικό, δικτυακό ή αποθηκευτικό πόρο να ενεργεί ως πλειάδα αυτών ή και το αντίθετο αγνοώντας το υποκείμενο υλικό ή/και λογισμικό. Αυτή η μέθοδος μπορεί να αξιοποιεί με τον καλύτερο τρόπο το υλικό ενώ εισάγει μια κεντροποιημένη διαχείριση του συστήματος, βελτιώνοντας με αυτό τον τρόπο τις κεφαλαιακές δαπάνες και το λειτουργικό κόστος αντίστοιχα.

Στο οικιακό περιβάλλον, η εικονικοποίηση της οικιακής πύλης μπορεί να εξασφαλίσει τα παραπάνω, επιτρέποντας επίσης και στον πελάτη να κάνει χρήση της φορητότητας της υπηρεσίας εξασφαλίζοντας του πρόσβαση στην υπηρεσία και στις εγγραφές του ανεξάρτητα του χώρου ή του χρόνου.

Τα παραπάνω συνάδουν με την φιλοσοφία του Internet of Things, της νέας τάσης που αξιοποιεί την διασύνδεση πολλών 'έξυπνων' συσκευών με σκοπό τη συλλογή και την ανταλλαγή δεδομένων.

## 2.1 Internet of Things

Το σενάριο που θα περιγραφεί αποτελεί ένα παράδειγμα του Internet of Things (IoT), της νέας ιδεολογικής δομής που συνδυάζει χαρακτηριστικά από διαφορετικές τεχνολογίες με σκοπό τη σύγκλισή τους. Τα διάφορα διαδικτυακά προτόκολλα μαζί με τις τεχνολογίες τηλεσκόπησης, τις ενσωματωμένες συσκευές και τις μεθόδους διάχυτου υπολογισμού συνεννώνονται σχηματίζοντας ένα σύστημα στο οποίο ο πραγματικός με τον ψηφιακό κόσμο βρίσκονται σε συνεχή διαδραστικότητα.

Δομικό στοιχείο της IoT θεώρησης είναι το 'έξυπνο αντικείμενο', δηλαδή συσκευές που χρησιμοποιούμε καθημερινά στις οποίες έχουμε εμφυσήσει λογική. Με αυτό τον τρόπο αποκτούν την ικανότητα συλλογής πληροφοριών από το περιβάλλον καθώς και τον έλεγχο αυτού (του περιβάλλοντος), όπως επίσης και την ικανότητα σύνδεσης με το διαδίκτυο ανταλλάσοντας έτσι δεδομένα με τις υπόλοιπες έξυπνες συσκευές.

Ο αριθμός των συσκευών που αναμένεται να διασυνδεθούν καθώς και ο όγκος των διαθέσιμων δεδομένων δημιουργεί ευκαιρίες για την σύνθεση υπηρεσίων, των οποίων τα κέρδη μπορεί να καρπωθεί η κοινωνία, το περιβάλλον και ο άνθρωπος ως οντότητα.

## 2.2 Network Function Virtualization (NFV)

Τα τηλεπικοινωνιακά δίκτυα αποτελούνται από έναν αυξανόμενο αριθμό μηχανημάτων που εκτελούν μια συγκεκριμένη εργασία. Η εύρεση διαθέσιμου χώρου και η παροχή τροφοδοσίας για τον νέο εξοπλισμό σε μια νέα δικτυακή υπηρεσία γίνεται ολοένα και πιο δύσκολη όπως και η εφαρμογή του στο δίκτυο ενώ η τεχνολογία από την οποία αποτελείται μπορεί να ξεπεραστεί σε μικρό χρονικό διάστημα, λόγω της επιταχυνόμενης καινοτομίας.

Το NFV αποτελεί μια πρωτοβουλία που έχει σκοπό να εικονικοποιήσει τις δικτυακές λειτουργίες που συσχετίζονται με αυτό τον εξοπλισμό, μειώνοντας έτσι το κόστος κατασκευής και διαχείρισης δικτυακών υπηρεσιών. Οι δικτυακές δραστηριότητες που προσφέρονταν από δρομολογητές, firewalls, και άλλες δεσμευμένες προς τις υπηρεσίες συσκευές, δύναται να εξυπηρετηθούν από εικονικές μηχανές.

### 2.2.1 Πλεονεκτήματα του NFV

- Μείωση των Capex & Opex λόγω του ελαττωμμένου κόστους εξοπλισμού και κατανάλωσης.

- Μειωμένο χρόνο διάθεσης στην αγορά των νέων υπηρεσιών
- Βελτιωμένη απόδοση των επενδύσεων από τις νέες υπηρεσίες
- Μεγαλύτερη ευελιξία ως προς την κλιμάκωση του δικτύου και στην εξέλιξη της υπηρεσίας.
- Άνοιγμα στην αγορά εικονικοποίησης και των προϊόντων καθαρού λογισμικού.
- Η εγκατάσταση και ο έλεγχος νέων υπηρεσιών γίνεται με χαμηλότερο ρίσκο.

Λαμβάνοντας υπ'όψιν τα παραπάνω, η αποφυγή εγκατάστασης νέου υλικού μπορεί να ενισχύσει τα κέρδη που προέρχονται από το οικιακό περιβάλλον εάν αυτό εξεταστεί με την προσέγγιση του NFV.

## 2.3 Το οικιακό περιβάλλον

Η αρχιτεκτονική του δικτύου που προσφέρει μέχρι τώρα τις οικιακές υπηρεσίες περιλαμβάνει συστήματα υποστήριξης και συσκευές στις εγκαταστάσεις του πελάτη (cpe). Τα cpe συνήθως περιλαμβάνουν την οικιακή πύλη (residential gateway / RGW) που προσφέρει τις υπηρεσίες διαδικτύου και voip καθώς και τον αποκωδικοποιητή (Setup Box / STB) για τις πολυμεσικές υπηρεσίες που συνήθως υποστηρίζουν τοπική αποθήκευση(personal video recording/ PVR).

### 2.3.1 Πλεονεκτήματα & μειονεκτήματα της εικονικοποίησης στο οικιακό περιβάλλον

Τα κέρδη της εικονικοποίησης της οικιακής πύλης τόσο στον πάροχο όσο και στον τελικό πελάτη συνοψίζονται παρακάτω:

- Μείωση των κεφαλαιακών δαπανών (CAPEX) λόγω της εξάλειψης του κόστους των STB/RGW (ένα για κάθε τηλεόραση).
- Μείωση του λειτουργικού κόστους (OPEX) της συντήρησης και αναβάθμιση των CPE καθώς και από την χρήση απομακρυσμένων διαγνωστικών για εξεύρεση λύσης των προβλημάτων που προκύπτουν στο οικιακό δίκτυο.
- Βελτιωμένη ποιότητα της εμπειρίας (QoE) λόγω της απομακρυσμένης συνδεσιμότητας με πρόσβαση σε όλα τα περιεχόμενα και τις υπηρεσίες.
- Η εισαγωγή νέων υπηρεσιών γίνεται πιο ομαλά καθώς η εξάρτηση από τα CPE και η διαδικασία εγκατάστασης στον χρήστη ελαχιστοποιούνται.

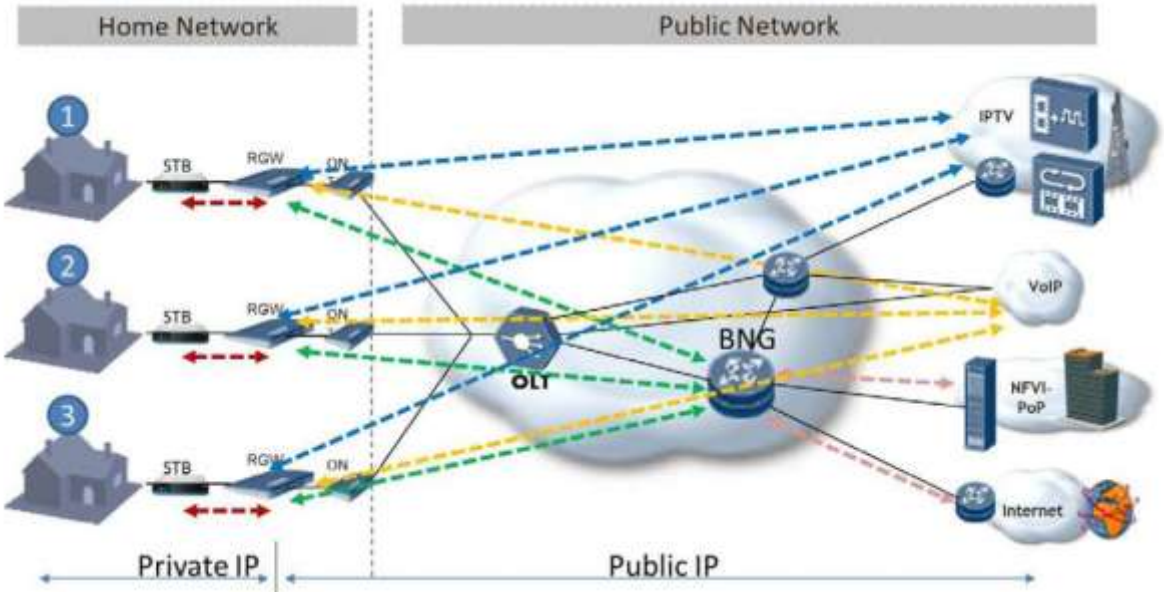
Έτσι λοιπόν ο συνδυασμός των δικτύων μαζί με την τεχνολογία του nfv διευκολύνει την εικονικοποίηση του οικιακού περιβάλλοντος απαιτώντας μόνο φυσική σύνδεση και απλές συσκευές με χαμηλό κόστος συντήρησης στις εγκαταστάσεις του πελάτη.

Όμως, η υλοποίηση της απ'ευθείας ανάθεσης μιας εικονικής μηχανής για κάθε εικονικοποιημένο vSTB/vRGW απαιτεί ένα υπολογιστικό νέφος με αυξημένες απαιτήσεις σε πόρους. Για αυτό το λόγο οι πάροχοι έχουν τη δυνατότητα να πρόσφερουν μικτές υπηρεσίες, τόσο με την εγκατάσταση της φυσικής συσκευής όσο και με την παροχή του νέου εικονικοποιημένου μοντέλου.

## 2.4 Περιγραφή σεναρίου

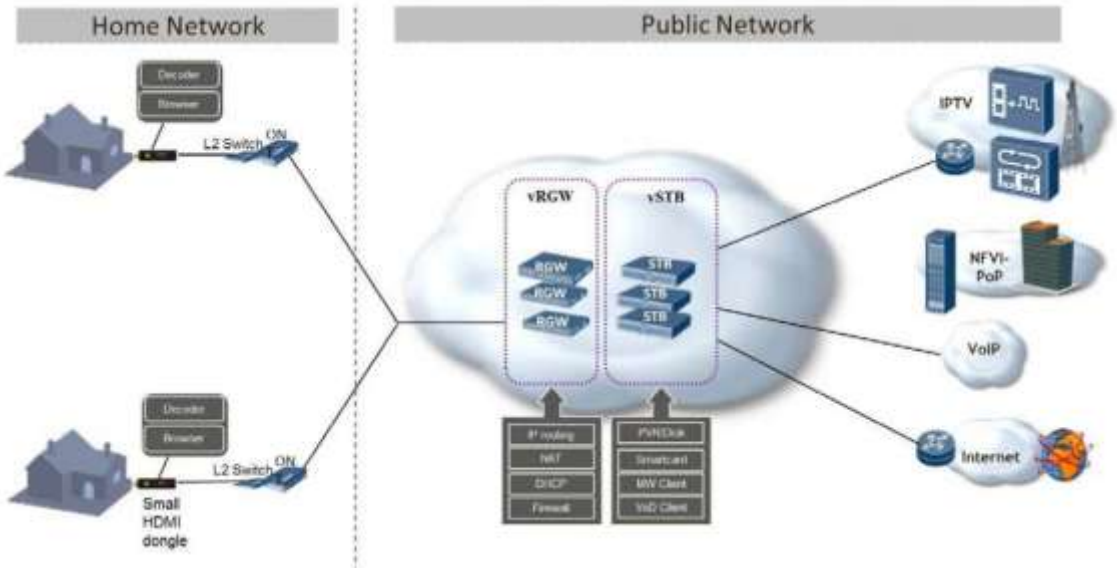
Στην παρακάτω εικόνα, δίνεται ένα παράδειγμα του οικιακού περιβάλλοντος χωρίς εικονικοποίηση. Σε αυτό, κάθε σπίτι είναι εξοπλισμένο με ένα RGW και ένα IP STB.

Η τεχνολογία NFV διευκολύνει την εικονικοποίηση των υπηρεσιών και την μετακίνηση στο νέφος. Στην παρακάτω εικόνα φαίνεται αυτή η μετακίνηση όπως προτείνεται από το NFV, διατηρώντας ένα εικονικοποιημένο αντίγραφο της αρχική συσκευής.




**Εικόνα 1: Το οικιακό περιβάλλον χωρίς εικονικοποποίηση**

Έτσι το RGW μετρατρέπεται σε εικονικό RGW (vRGW) και το STB σε vSTB αντίστοιχα.




**Εικόνα 2: Λειτουργικότητα του εικονικοποιημένου οικιακού περιβάλλοντος**

Στην εικόνα 2 εμφανίζεται το μοντέλο της τοπολογίας δικτύου που θα χρησιμοποιηθεί για αυτή την εργασία. Σε αυτή διακρίνονται οι 3 κόμβοι, δηλαδή αυτοί της προέλευσης του πολυμεσικού περιεχομένου, ο ενδιάμεσος κόμβος που βρίσκεται στο υπολογιστικό νέφος και αποτελεί την εικονικοποιημένη οικιακή πύλη και ο τελικός προορισμός.

Ο κόμβος της εικονικοποιημένης οικιακής πύλης έχει επωμιστεί τις λειτουργίες τόσο της δικτύωσης όσο και της διαμόρφωσης, τροποποίησης και αποθήκευσης του περιεχομένου και γι'αυτό και στο εξής θα καλείται και ως vSTb/vRGW. Αυτό έρχεται σε

αντίθεση με το παραδοσιακό μοντέλο όπου οι πολυμεσικές λειτουργίες εκτελούνταν διαφορετικά από ότι οι δικτυακές.

Για τις ανάγκες της υλοποίησης αυτής της τοπολογίας, χρησιμοποιείται το εργαλεία κατασκευής και διαχείρισης υπολογιστικού νέφους OpenStack, καθώς και η βιβλιοθήκη διαχείρισης πολυμεσικών ροών gstreamer. Για τη μεταφορά των πακέτων στο δίκτυο χρησιμοποιείται το πρωτόκολλο rtp σε συνδυασμό με το πρωτόκολλο rtcp.

# 3. ΠΡΩΤΟΚΟΛΛΑ ΕΠΙΚΟΙΝΩΝΙΑΣ

Ο κλασσικός τρόπος μεταφοράς αρχείων περιλαμβάνει το κατέβασμα ενός αρχείου πρωτού χρησιμοποιηθεί από τον χρήστη. Το κατέβασμα όμως περιέχει ορισμένα μειονεκτήματα όταν αφορά πολυμεσικό περιεχόμενο, όπως ο μεγάλος χρόνος ολοκλήρωσης πρωτού χρησιμοποιηθεί. Ένας εναλλακτικός μηχανισμός είναι η καταμερισμός του περιεχομένου σε αυτοδύναμα πακέτα τα οποία μπορούν να μεταδωθούν ξεχωριστά. Αυτό επιτρέπει στον δέκτη την αποκωδικοποίηση & αναπαραγωγή των πακέτων που έχουν ήδη ληφθεί.

## 3.1 Συνοπτική περιγραφή

Ο παραδοσιακός τρόπος χρήσης ενός αρχείου απαιτούσε το πλήρες κατέβασμά του. Παρά το ότι αυτή η μέθοδος είναι αρκετά αξιοπίστη, κρύβει επίσης και δύο σημαντικά μειονεκτήματα όταν πρόκειται για πολυμεσικές εφαρμογές. Η απαιτούμενη μνήμη που χρειάζεται για την προσωρινή αποθήκευση του αρχείου αλλά και ο χρόνος που απαιτείται πρωτού ο χρήστης μπορεί να αξιοποιήσει την πληροφορία που μπορεί να εκτείνεται από λεπτά μέχρι ώρες μειώνει την ευελιξία τόσο στον χρήστη όσο και κατασκευαστή εφαρμογών.

Μια εναλλακτική μέθοδος αντί του κατεβάσματος είναι το streaming. Με αυτή τη μέθοδο η ροή των bit χωρίζεται σε ξεχωριστά πακέτα τα οποία μπορούν να μεταδωθούν ανεξάρτητα. Αυτό επιτρέπει στον δέκτη να αποκωδικοποιήσει και να αναπαράξει τα κομμάτια τα οποία έχουν ληφθεί ενώ ταυτόχρονα συνεχίζει να λαμβάνει τα επόμενα πολυμεσικά πακέτα από την πηγή. Με αυτόν τον τρόπο μειώνεται κατά πολύ ο χρόνος από την εκπομπή των δεδομένων μέχρι την προβολή στον χρήστη κάτι που είναι ύψιστης πρωτεραιότητας για διαδραστικές εφαρμογές όπως VoD στις οποίες ο χρήστης μπορεί εάν το επιθυμεί να αλλάξει γρήγορα κανάλι ή σε τηλεδιάσκεψη με άγνωστη διάρκεια. Ένα άλλο πλεονέκτημα που έχει το streaming είναι οι μειωμένες απαιτήσεις σε μνήμη. Το συμπέρασμα από τα παραπάνω είναι η αυξημένη ευελιξία στον χρήστη, η οποία έρχεται με χρονικές και προθεσμιακές απαιτήσεις που να εξασφαλίζουν την αναπαραγωγή του μέσου σε πραγματικό χρόνο. Αυτό οδηγεί σε καινούριες προκλήσεις και νέες τεχνικές σχεδιάσης τηλεπικοινωνιακών συστημάτων που να υποστηρίζουν κατάλληλα streaming εφαρμογές.

Το πρωτόκολλο rtp(real time-transport protocol [3]) παρέχει κατάλληλες λειτουργίες μεταφοράς για εφαρμογές που μεταδίδουν δεδομένα σε πραγματικό χρόνο όπως εικόνα και ήχο. Η ποιότητα της υπηρεσίας αλλά και ο έλεγχος των πόρων είναι αντικείμενο του πρωτόκολλου ελέγχου rtcp(real time control protocol [3]), το οποίο πραγματοποιεί τα παραπάνω τηρώντας κλιμακούμενη επίβλεψη σε μεγάλα δίκτυα ευρυεκπομπής. Τα rtp,rtcp είναι σχεδιασμένα έτσι ώστε είναι ανεξάρτητα του υποκείμενου δικτύου μεταφοράς.

Το rtsp(real time streaming protocol [4] εγκαθιδρύει και ελέγχει μία ή περισσότερες συγχρονισμένες ροές συνεχόμενων πολυμέσων όπως εικόνα και ήχος. Οι ροές που ελέγχει το rtsp μπορεί να χρησιμοποιούν το πρωτόκολλο rtp αλλά η ενέργεια του rtsp δεν εξαρτάται από τον μηχανισμό μεταφοράς της ροής πολυμέσων. Το rtsp λειτουργεί σαν ένα δικτυακό τηλεχειριστήριο για εξυπηρετητές πολυμέσων.

## 3.2 rtp

### 3.2.1 Αποστολή πακέτων με χρήση του πρωτοκόλλου rtp.

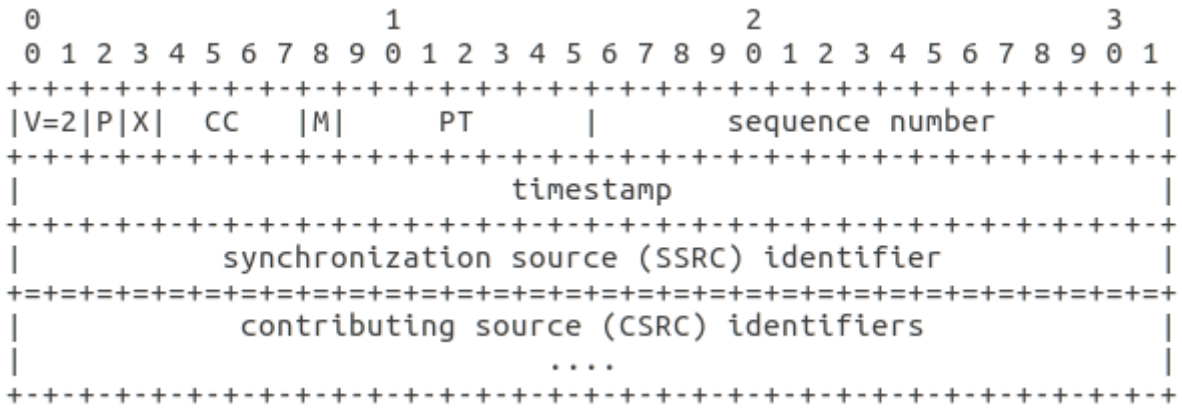
Κάθε ροή εικόνας ή ήχου μεταδίδεται σαν μία διαφορετική συνεδρία στο πρωτόκολλο rtp. Αυτό σημαίνει για παράδειγμα ότι σε μία διάσκεψη που θα χρειαστεί εικόνα και ήχος,

τότε αυτά θα μεταδωθούν σαν 2 ξεχωριστές ροές. Αυτό σημαίνει πως πακέτα rtp και rtcp μεταδίδονται για κάθε ροή χρησιμοποιώντας 2 διαφορετικές UDP πόρτες. Οι δύο ροές, δηλαδή αυτές της εικόνας και ήχου δεν συνδέονται απ'ευθείας. Ένας χρήστης ο οποίος λαμβάνει και της δύο ροές θα πρέπει να χρησιμοποιήσει το ίδιο (διακριτό) όνομα. Αυτό διευκολύνει τους χρήστες οι οποίοι επιθυμούν να λάβουν τη μία εκ των δύο ροών.

Παρά τον διαχωρισμό, ο συγχρονισμός εικόνας και ήχου μπορεί να επιτευχθεί με την αποστολή πληροφορίας χρονισμού στα RTCP πακέτα των δύο συνεδριών.

### 3.2.2 Η κεφαλίδα ενός RTP πακέτου

Οι πρώτες δώδεκα οκτάδες είναι παρών σε κάθε RTP πακέτο. Η λίστα με τους CSRC είναι παρών μόνο όταν εισέρχονται από έναν μίκτη.




**Εικόνα 3: Η κεφαλίδα ενός RTP πακέτου**

- Έκδοση (version): 2 bits
  Το πεδίο που προσδιορίζει την έκδοση του RTP. Η παρούσα τιμή που λαμβάνει είναι (2).
- padding (P): 1 bit
  Εάν το πεδίο padding έχει τιμή, τότε το πακέτο περιέχει μία ή περισσότερες οκτάδες bit στο τέλος του οι οποίες δεν είναι μέρος των δεδομένων εικόνας/ήχου. Η τελευταία οκτάδα προσδιορίζει τον αριθμό των οκτάδων που θα πρέπει να αγνοηθούν, περιλαμβανομένης και της ίδιας. Το γέμισμα αυο μπορεί να χρειάζεται έτσι ώστε το συνολικό πακέτο να αποκτήσει ένα συγκεκριμένο μέγεθος, επειδή έτσι μπορεί να ορίζεται από κάποιον αλγόριθμο κρυπτογράφησης.
- extension (X): 1 bit
  Εάν το πεδίο extension έχει πάρει τιμή, τότε η κεφαλίδα θα πρέπει να ακολουθείται από μία επιπλέον κεφαλίδα, με προσδιορισμένο συντακτικό.
- Synchronization source (SSRC):
  Η πηγή της ροής των RTP πακέτων. Ταυτοποιείται από μία 32-bit ταυτότητα έτσι ώστε να μην βασίζεται στην ταυτότητα δικτύου.
- payload type (PT): 7 bits
  Το πεδίο που προσδιορίζει τη μορφή των δεδομένων του RTP πακέτου και το τρόπο που θα ερμηνευθεί από την εφαρμογή. Το RTP μπορεί να προσδιορίσει από το πεδίο PT τη μορφή του πακέτου[5], ενώ επιπλέον προσδιορισμοί μπορεί να επέλθουν από πρόσθετα πρωτόκολλα και μηχανισμούς.
- sequence number: 16 bits

Ο αριθμός ακολουθίας αυξάνει κατά ένα για κάθε πακέτο RTP που στέλνεται, και μπορεί να χρησιμοποιηθεί από τον δέκτη για την ανίχνευση απώλειας πακέτων και για την επαναφορά της αλληλουχίας. Η αρχική τιμή πρέπει να είναι τυχαία (μη προβλέψιμη) για λόγους ασφαλείας.

- timestamp: 32 bits
  Η χρονοσφραγίδα αντανακλά τη δειγματοληπτική στιγμή της πρώτης οκτάδας στα δεδομένα του πακέτου RTP. Η δειγματοληπτική στιγμή πρέπει να προέρχεται από ρολόι κατάλληλο έτσι ώστε να επιτρέπεται ο συγχρονισμός και η τοποθέτηση σε διάταξη.

Από τα παραπάνω φαίνεται πως η χρονοσφραγίδα τοποθετεί τα πακέτα στη σωστή χρονική στιγμή ενώ ο αριθμός ακολουθίας για τον εντοπισμό απωλειών. Ο αριθμός ακολουθίας αυξάνει κατά ένα για κάθε πακέτο RTP που αποστέλλεται ενώ η χρονοσφραγίδα αυξάνει ανάλογα τον χρόνο που περιέχεται στο ωφέλιμο φορτίο του πακέτου. Η χρονοσφραγίδα μπορεί να μείνει ίδια κυρίως στην περίπτωση αποστολής βίντεο όπου ένα καρέ κατατμίζεται σε περισσότερα από ένα πακέτα. Αποτέλεσμα αυτού είναι πως οι χρονοσφραγίδες από πολλαπλές ροές εξελίσσονται διαφορετικά και ενώ για μια ροή αρκεί η χρονοσφραγίδα για το συγχρονισμό της, για περισσότερες από μία είναι απαραίτητη η ύπαρξη μια πηγής χρονισμού.

Για την παρουσίαση της απόλυτης ώρας και ημερομηνίας χρησιμοποιείται το πρωτόκολλο NTP (Network Time Protocol), που προσδιορίζει τα δευτερόλεπτα στην 00:00 ώρα της 1 Ιανουαρίου του έτους 1900. Η χρήση του πρωτοκόλλου NTP είναι χρήσιμη στον συγχρονισμό ροών από διαφορετικές πηγές, αλλά δεν είναι υποχρεωτική για το πρωτόκολλο rtp. Ο κατασκευαστής μια εφαρμογής μπορεί εάν θέλει να χρησιμοποιήσει κάποια άλλη πηγή χρονισμού ή και καμμία εντελώς.

### 3.3 Πρωτόκολλο ελέγχου RTP (RTCP)

Το πρωτόκολλο ελέγχου RTP (RTCP), βασίζεται στην περιοδική μετάδοση των πακέτων ελέγχου προς όλους του συμμετέχοντες σε μια συνεδρία, χρησιμοποιώντας τον ίδιο μηχανισμό κατανομής όπως στα πακέτα δεδομένων. Το υποκείμενο πρωτόκολλο θα πρέπει να παρέχει την πολυπλεξία μεταξύ των πακέτων δεδομένων και των πακέτων ελέγχου, χρησιμοποιώντας για παράδειγμα διαφορετικές UDP πόρτες.

#### 3.3.1 Λειτουργίες του πρωτοκόλλου rtcp

Το RTCP εκτελεί τις εξής λειτουργίες:

- Παρέχει ανάδραση στην ποιότητα του διαμοιρασμού δεδομένων. Αυτό είναι αναπόσπαστο κομμάτι του RTP ως προτόκολλο μεταφοράς και σχετίζεται με τον έλεγχο ροής και συμφόρησης που ισχύει και στα υπόλοιπα. Αυτή η ανατροφοδότηση έχει αποδειχθεί πως είναι πολύ χρήσιμη όταν παρέχεται από τον δέκτη ή τους δέκτες διότι προσφέρει πολύ χρήσιμα στοιχεία στον διαχειριστή (όπως ο έλεγχος τοπικού ή συνολικού προβλήματος).
- Το rtcp φέρει ένα παραμένων αναγνωριστικό της RTP πηγής προέλευσης, που καλείται CNAME (Canonical Name). Επειδή το SSRC μπορεί να αλλάξει, σε περίπτωση που διαπιστωθεί μια διένεξη, οι δέκτες κάνουν χρήση του CNAME για την παρακολούθηση κάθε συμμετέχοντα.

#### 3.3.2 Το RTCP πακέτο

Ένα πακέτο RTCP φέρει μια ποικιλία πληροφοριών ελέγχου όπως

- SR: αναφορά αποστολέα (Sender Report), που περιέχει στατιστικά στοιχεία από συμμετέχοντες οι οποίοι είναι ενεργοί αποστολείς
- RR: αναφορά παραλήπτη (Receiver Report), που περιέχει στατιστικά στοιχεία από συμμετέχοντες που δεν είναι ενεργοί αποστολείς.

Οι αναφορές των συμμετεχόντων θα πρέπει να αποστέλλονται όσο πιο τακτικά όσο το εύρος ζώνης του δικτύου επιτρέπει με σκοπό την λεπτομερέστερη ανάλυση των στατικών.

- SDES: Περιέχει πηγαία περιγραφικά αντικείμενα, συμπεριλαμβανομένου του CNAME. Οι νέοι παραλήπτες χρειάζεται να λαμβάνουν το CNAME όσο πιο γρήγορα γίνεται έτσι ώστε να προχωρούν σε ενέργειες συσχετισμών των διαφόρων ροών, όπως πχ ο συγχρονισμός εικόνας - ήχου.
- BYE: Υποδεικνύει το τέλος της συμμετοχής σε μια ροή. Όλα τα πακέτα RTCP μπορούν να ληφθούν με οποιαδήποτε σειρά, πλην του BYE το οποίο θα πρέπει να είναι το τελευταίο για ένα δοσμένο SSRC.
- APP: Λειτουργίες σχετικές με την εφαρμογή.

Το κάθε rtcp πακέτο είναι αυτόνομο και μπορεί να δεχτεί επεξεργασία αναξάρτητα της σειράς του. Παρ'όλ'αυτά, τα πακέτα rtpc μπορούν να συσσωρευτούν σε ένα σύνθετο πακέτο.

### 3.3.3 Διαστήματα αποστολής RTCP

Παρ'όλ'αυτά, ο όγκος δεδομένων που προέρχεται από αναφορές ελέγχου δεν είναι αυτοπεριοριστικός αλλά γραμμικός και ανάλογος του πλήθους των συμμετεχόντων. Για αυτό το λόγο, ο ρυθμός μετάδοσης θα πρέπει να μειωθεί υπολογίζοντας δυναμικά τα διαστήματα μεταξύ των αποστολών των RTCP πακέτων.

Για κάθε συνεδρία, ορίζεται ένα όριο ως εύρος ζώνης κίνησης το οποίο διαιρείται μεταξύ των συμμετεχόντων. Αυτό το εύρος μπορεί να είναι λογικά κανονισμένο ή ορισμένο από το δίκτυο. Εάν δεν υπάρχει κάποια δέσμευση, καθορίζεται ένα 'εύλογο' εύρος ζώνης συνεδρίας. Συχνά, αυτό το εύρος ζώνης συνεδρίας είναι το άθροισμα των ονομαστικών ευρών ζώνης των αποστολέων που αναμένεται να είναι ταυτόχρονα ενεργοί. Ο υπολογισμός αυτός μπορεί να προκύπτει από μια εφαρμογή διαχείρησης συνεδριών. Όλοι οι συμμετέχοντες πρέπει να χρησιμοποιούν την ίδια τιμή ώστε να υπολογίζεται κοινό διάστημα μεταξύ των rtcp αναφορών.

Ο υπολογισμός για το εύρος ζώνης των δεδομένων και ελέγχου από το σύστημα δέσμευσης πόρων συμπεριλαμβάνει το υποκείμενο πρωτόκολλο μεταφοράς και τα δικτυακά πρωτόκολλα. Από τον τελικό υπολογισμό, τα δεδομένα ελέγχου θα πρέπει να περιοριστούν σε ένα μικρό μέρος του εύρους ζώνης συνεδρίας, τέτοιο ώστε η κύρια λειτουργία του πρωτοκόλλου μεταφοράς – που είναι η μεταφορά των δεδομένων – να μην επηρεάζεται. Επίσης θα πρέπει να είναι και γνωστό, για να μπορεί ο κάθε συμμετέχων να υπολογίζει ανεξάρτητα το δικό του ποσοστό. Η κίνηση των δεδομένων που αφορούν τον έλεγχο προτείνεται να είναι της τάξης του 5% ενώ επίσης προτείνεται το ¼ αυτής να αφιερώνεται στους συμμετέχοντες που κάνουν αποστολή δεδομένων έτσι ώστε να λαμβάνουν οι υπόλοιπο τα CNAME πιο γρήγορα.

# 4. ΚΑΤΑΣΚΕΥΗ ΤΗΣ ΥΠΟΔΟΜΗΣ

Η εικονικοποίηση έχει ως κύριο χαρακτηριστικό σκοπό τη μετατροπή μιας συσκευής σε ένα αντίγραφό του από λογισμικό, εύκολα προσβάσιμο από παντού και ανεξαρτήτος ώρας. Έτσι λοιπόν για την εικονικοποίηση της οικιακής πύλης κατασκευάζεται ένα αντίγραφο αυτής που τοποθετείται στο ‘νέφος’ με σκοπό τη συνεχής χρήση της υπηρεσίας και την διαθεσιμότητα του χρήστη στις εγγραφές του. Η μεταφορά αυτή πραγματοποιείται με τη χρήση ενός εργαλείου ανάπτυξης υπολογιστικού νέφους, σκοπός του οποίου είναι η ανάπτυξη της κατάλληλης πλατφόρμας από άποψη δικτυακών, υπολογιστικών και αποθηκευτικών πόρων.

Αντικείμενο αυτών των εργαλείων είναι η διατεταγμένη υλοποίηση υπηρεσιών όπως περιγράφονται στο μοντέλο του cloud computing. Με αυτόν τον όρο, εννοείται κυρίως η παροχή υπολογιστικών πόρων σε χρήστες μέσω ενός απομακρυσμένου περιβάλλοντος. Για αυτό το λόγο για αυτές τις υπηρεσίες χρησιμοποιείται ο όρος ‘ως υπηρεσία’ (as a Service)Μερικές από αυτές τις υπηρεσίες είναι η PaaS (Platform as a Service), η IaaS (Infrastructure as a Service), η SaaS (Software as a service) κλπ.

## 4.1 OpenStack

Το OpenStack είναι μια δωρεάν και ανοιχτού λογισμικού πλατφόρμα υπολογιστικού νέφους η οποία αναπτύσσεται ως Infrastructure as a service(IaaS). Η πλατφόρμα αποτελείται από συγγενικά στοιχεία τα οποία ελέγχουν "δεξαμενές" από υπολογιστικούς, αποθηκευτικούς και δικτυακούς πόρους μέσω ενός κέντρου δεδομένων(data center). Ο χρήστης μπορεί να χειριστεί την πλατφόρμα μέσω ενός δικτυακού ταμπλό (dashboard) ή με γραμμή εντολών.

Στις επόμενες παραγράφους αναλύονται τα στοιχεία του Openstack που χρησιμοποιήθηκαν για την ανάπτυξη της υποδομής για την στέγαση της εικονικοποιημένης οικιακής πύλης:

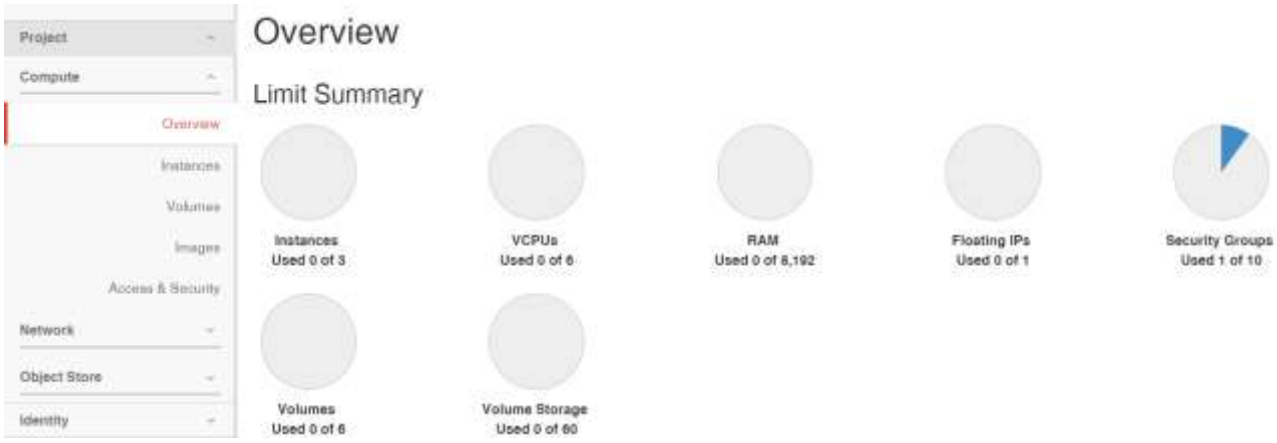
- Horizon: Το Horizon προσφέρει ένα γραφικό περιβάλλον προς τις υπόλοιπες υπηρεσίες του OpenStack (Nova, Neutron κλπ). Όπως και το τερματικό γραμμής εντολών αποτελεί τον σύνδεσμο μεταξύ του Openstack με τον χρήστη. Για να αποκτήσει κάποιος πρόσβαση στο δικτυακό γραφικό περιβάλλον αρκεί να συνδεθεί με έναν φυλλομετρητή στην ip του horizon.

  Στην πρώτη οθόνη του OpenStack Dashboard ο χρήστης έχει μια γενική επισκόπηση των διαθέσιμων/δεσμευμένων πόρων του συστήματος που του αναλογεί. Με αυτόν τον τρόπο μπορεί να πάρει γρήγορες αποφάσεις αναφορικά με τον επιμερισμό των εργασίων δεσμεύοντας περισσότερους επεξεργαστικούς πόρους ή εντατικοποιώντας τις διεργασίες δίνοντας περισσότερη μνήμη στις ήδη υπάρχουσες υπολογιστικές μηχανές.

- Nova: Η διαχείριση μεγάλων δικτύων εικονικών μηχανών με σκοπό την πρόσβαση στους υπολογιστικούς πόρους είναι αντικείμενο της συνιστώσας Nova. Το Nova προσφέρει μαζικά κλιμακούμενη πρόσβαση κατά βούληση σε υπολογιστικούς και αποθηκευτικούς πόρους. Όπως με τις υπόλοιπες περισσότερες κατανεμημένες συνιστώσες στην πλατφόρμα του OpenStack, το Nova επικοινωνεί με το Keystone για την αυθεντικοποίηση, το Horizon μέσω της δικτυακής επαφής του και το Glance για την επιλογή του λειτουργικού συστήματος των εικονικών μηχανών.

- Neutron: Το Neutron προσφέρει τις δικτυακές δυνατότητες του OpenStack, βεβαιώνοντας ότι κάθε στοιχείο του ανεπτυγμένου νέφους επικοινωνεί με τα υπολοίπα.

- Keystone: Προσφέρει τις υπηρεσίες αυθεντικοποίησης του OpenStack. Στην πραγματικότητα είναι μια κεντρική λίστα όλων των χρηστών του OpenStack, χαρτογραφημένοι πάνω στις διαθέσιμες υπηρεσίες του νέφους που έχουν τα δικαιωμάτα να χρησιμοποιούν.




**Εικόνα 4: OpenStack Dashboard**

Στην παραπάνω εικόνα απεικονίζεται η αρχική κατάσταση του συστήματος. Όλοι οι υπολογιστικοί, δικτυακοί & αποθηκευτικοί πόροι είναι διαθέσιμοι.

Η υπηρεσία glance επιτρέπει στους χρήστες την διαχείριση των virtual machine images. Μέσω αυτής της υπηρεσίας, ο χρήστης μπορεί να αποθηκεύσει και να ανακτήσει αυτά τα images από διάφορες τοποθεσίες.

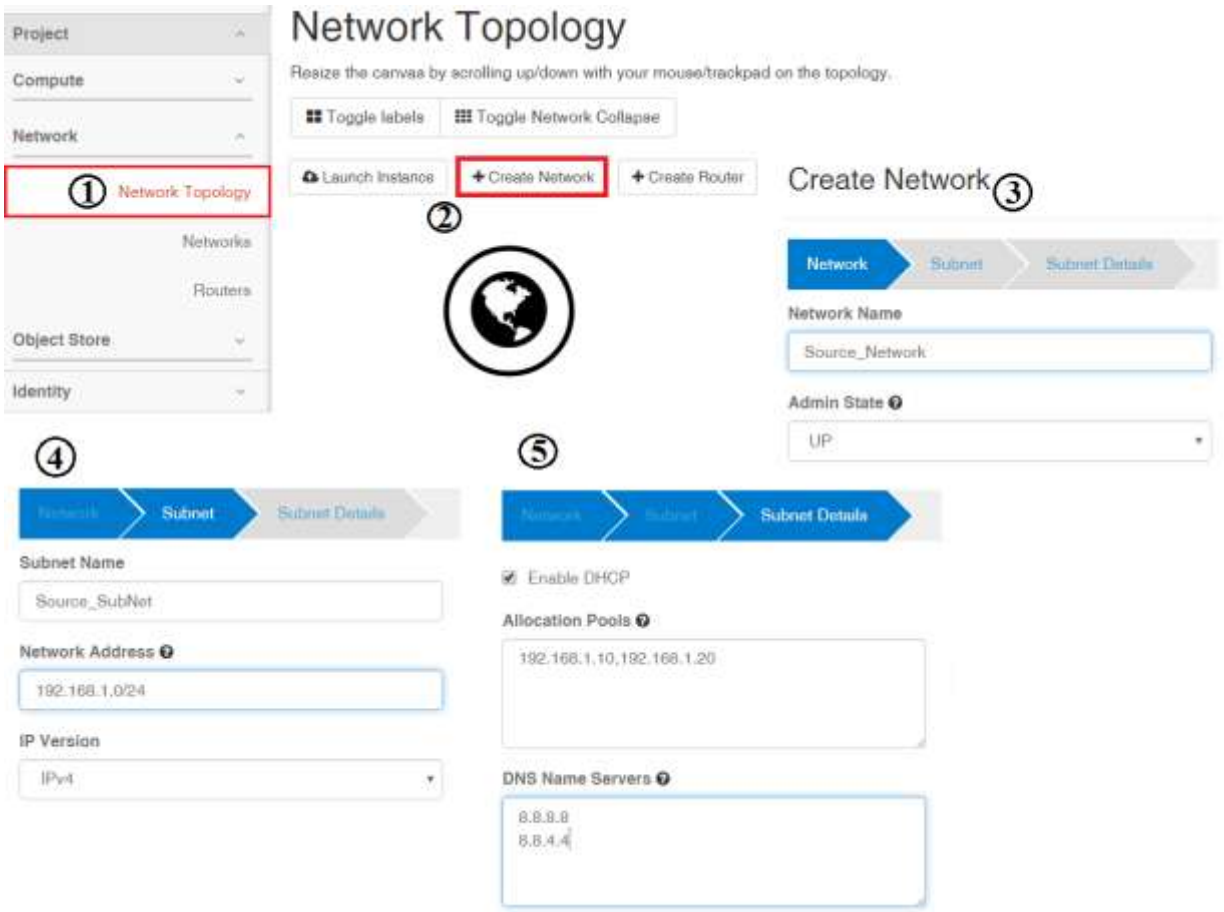
### 4.2 Κατασκευή της δικτύωσης

Η δικτύωση του OpenStack, διαχειρίζεται την υποδομή της εικονικής δικτύωσης και περιλαμβάνει τα δίκτυα, τα switches, τα υποδίκτυα και τους δρομολογητές των μηχανών που χειρίζεται η υπηρεσία του Nova. Οι χρήστες μπορούν να προσδιορίσουν τη συνδεσιμότητα και διευθυνσιοδότηση στο υπολογιστικό νέφος μέσω ενός API (Application Programming Interface).

Οι δικτυακές υπηρεσίες του OpenStack, καλούνται με ένα κοινό όνομα, το Neutron.

#### 4.2.1 Κατασκευή του δικτύου

Σε αυτή την ενότητα περιγράφεται η κατασκευή της απαραίτητης υποδομής για τις ανάγκες της εικονικοποιημένης οικιακής πύλης. Το συνολικό μοντέλο αποτελείται από τρεις κόμβους, αυτούς του αποστολέα, του vSTB/vRGW και αυτόν του πελάτη. Για την δημιουργία του κάθε κόμβου κατασκευάζεται ένα ξεχωριστό δίκτυο με μια εικονική μηχανή ενώ η δρομολόγηση των πακέτων μεταξύ των τριών δικτύων γίνεται μέσω ενός εικονικού δρομολογητή.




**Εικόνα 5: Κατασκευή του δικτύου με χρήση του dashboard**

Εκτός όμως από το γραφικό περιβάλλον, το OpenStack επιτρέπει τη διαχείριση του υπολογιστικού νέφους μέσω της γραμμής εντολών, ενός ενοποιημένου cli (Command Line Interface) για κάθε υπηρεσία. Η σύνταξη των εντολών έχει σχεδιασθεί με γνώμονα την ευκολία ενώ η ύπαρξη της απλοποιεί την αυτοματοποίηση των διαδικασιών με το πέρασμα των εντολών σε κάποιο script.

Η κατασκεύη ενός δικτύου μέσω του OpenStack cli γίνεται με χρήση της εντολής neutron ενώ όλα τα υπόλοιπα χαρακτηριστικά (dhcp, διαθέσιμες διευθύνσεις κλπ.) δίνονται ως ορίσματα ακολουθώντας το συντακτικό της εντολής.

```
stavros@stavros-Lenovo-V110-15ISK:~/Desktop$ neutron net-create Source_Network
Created a new network:
+-----------------+--------------------------------------+
| Field           | Value                                |
+-----------------+--------------------------------------+
| admin_state_up  | True                                 |
| id              | d720e46e-f5ba-4695-af65-caa89a52da4a |
| mtu             | 0                                    |
| name            | Source_Network                       |
| router:external | False                                |
| shared          | False                                |
| status          | ACTIVE                               |
| subnets         |                                      |
| tenant_id       | ff4f063248034b18bd9375bb2b392d00     |
+-----------------+--------------------------------------+
```

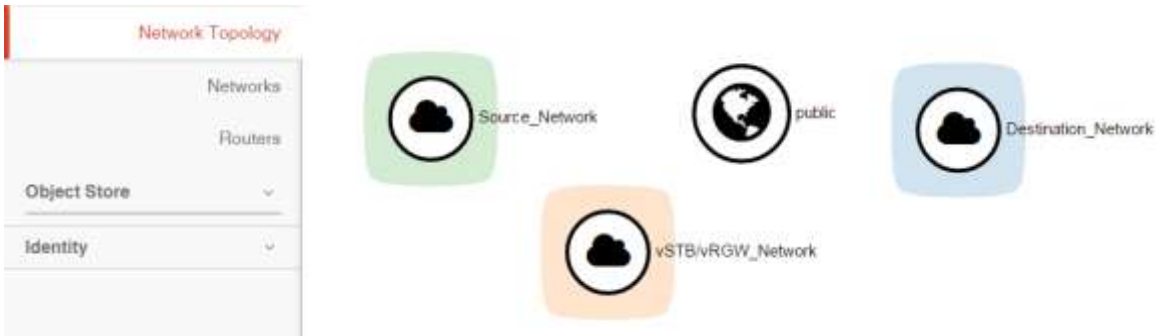
**Εικόνα 6: Κατασκευή του δικτύου με χρήση της γραμμής εντολών**

```
stavros@stavros-Lenovo-V110-15ISK:~/Desktop$ neutron subnet-create Source_Network --name Source_Subnet 192.168.1.0/24 --enable-dhcp --allocatio
n-pool start=192.168.1.10,end=192.168.1.20 --dns-nameserver 8.8.8.8 --dns-nameserver 8.8.4.4
Created a new subnet:
+-------------------+--------------------------------------------------+
| Field             | Value                                            |
+-------------------+--------------------------------------------------+
| allocation_pools  | {"start": "192.168.1.10", "end": "192.168.1.20"} |
| cidr              | 192.168.1.0/24                                   |
| dns_nameservers   | 8.8.8.8                                          |
|                   | 8.8.4.4                                          |
| enable_dhcp       | True                                             |
| gateway_ip        | 192.168.1.1                                      |
| host_routes       |                                                  |
| id                | 5ea9a581-b49d-4510-8be7-1092aa6f8ac4             |
| ip_version        | 4                                                |
| ipv6_address_mode |                                                  |
| ipv6_ra_mode      |                                                  |
| name              | Source_Subnet                                    |
| network_id        | d720e46e-f5ba-4695-af65-caa89a52da4a             |
| subnetpool_id     |                                                  |
| tenant_id         | ff4f063248034b18bd9375bb2b392d00                 |
+-------------------+--------------------------------------------------+
```

**Εικόνα 7: Κατασκευή του υποδικτύου με χρήση της γραμμής εντολών**

Η παραπάνω διαδικασία γίνεται τρεις φορές, μια για κάθε κόμβο του δικτύου.

Μόλις ολοκληρωθούν τα παραπάνω βήματα, ο διαχειριστής ενημερώνεται για το αποτέλεσμα των ενεργειών του ενώ μπορεί να το επαληθεύσει μέσω της γραφικής διεπαφής στην καρτέλα Network Topology.




**Εικόνα 8: Γραφική απεικόνιση του δικτυώματος**

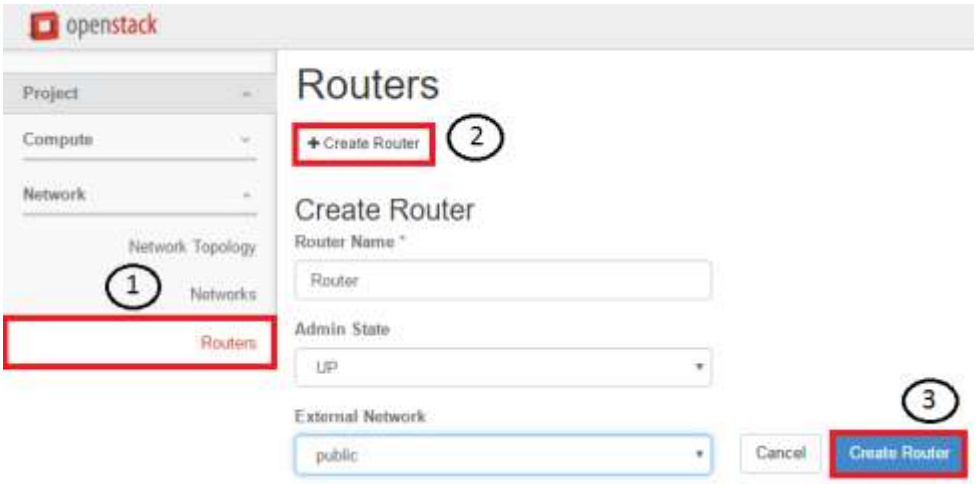
Η επαλήθευση μπορεί επίσης να πραγματοποιηθεί και μέσω της γραμμής εντολών, με χρήση της εντολής neutron net-list. Με αυτήν, ο διαχειριστής ενημερώνεται για το σύνολο των υπαρχόντων δικτύων και για τις παραμετροποιήσεις που αυτά έχουν δεχτεί.

```
stavros@stavros-Lenovo-V110-15ISK:~/Desktop$ neutron net-list
+--------------------------+--------------------+--------------------------+
| id                       | name               | subnets                  |
+--------------------------+--------------------+--------------------------+
| 1fd0a21e-e700-46ae-      | public             | 7913ade1-01e3-44fa-9863-91 |
| 9f05-0b3164daafcc        |                    | 867c0d23ab               |
| 0570b6a5-b49d-48d0-871a- | vSTB/vRGW_Netwrok  | b07b01b5-2055-4ade-      |
| a4e3489a9e74             |                    | 9bc8-3a700d7958e3        |
|                          |                    | 192.168.2.0/24           |
| b5ae2d6d-f8c9-41ac-      | Source_Netwrok     | 86a19641-3fbd-           |
| 8d83-28afedefee1e        |                    | 43e6-b147-87592b4e9770   |
|                          |                    | 192.168.1.0/24           |
| 9552c00b-bee4-4107-892b- | Destination_Netwrok | 596a7ace-7a49-48b2-a1b4-90 |
| 3ebb2daadd9b             |                    | 7254708d6a 192.168.3.0/24  |
+--------------------------+--------------------+--------------------------+
```

**Εικόνα 9: Απεικόνιση του δικτυώματος με χρήση της γραμμής εντολών**

### 4.3 Κατασκευή του δρομολογητή

Μόλις ολοκληρωθεί η κατασκευή των τριών δικτύων ακολουθεί η κατασκευή του εικονικού δρομολογητή που θα μεταφέρει τα πακέτα δεδομένων μεταξύ αυτών.




**Εικόνα 10: Κατασκευή του router με χρήση του dashboard**

Η παραπάνω διαδικασία μπορεί να πραγματοποιηθεί και εκτός της γραφικής διεπαφής, με χρήση της εντολής neutron router-create μέσω του cli.

```
stavros@stavros-Lenovo-V110-15ISK:~/Desktop$ neutron router-create Router
Created a new router:
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| admin_state_up        | True                                 |
| external_gateway_info |                                      |
| id                    | 2c69f882-35cb-4827-8880-dcb0d05d8d87 |
| name                  | Router                               |
| routes                |                                      |
| status                | ACTIVE                               |
| tenant_id             | ff4f063248034b18bd9375bb2b392d00     |
+-----------------------+--------------------------------------+
```

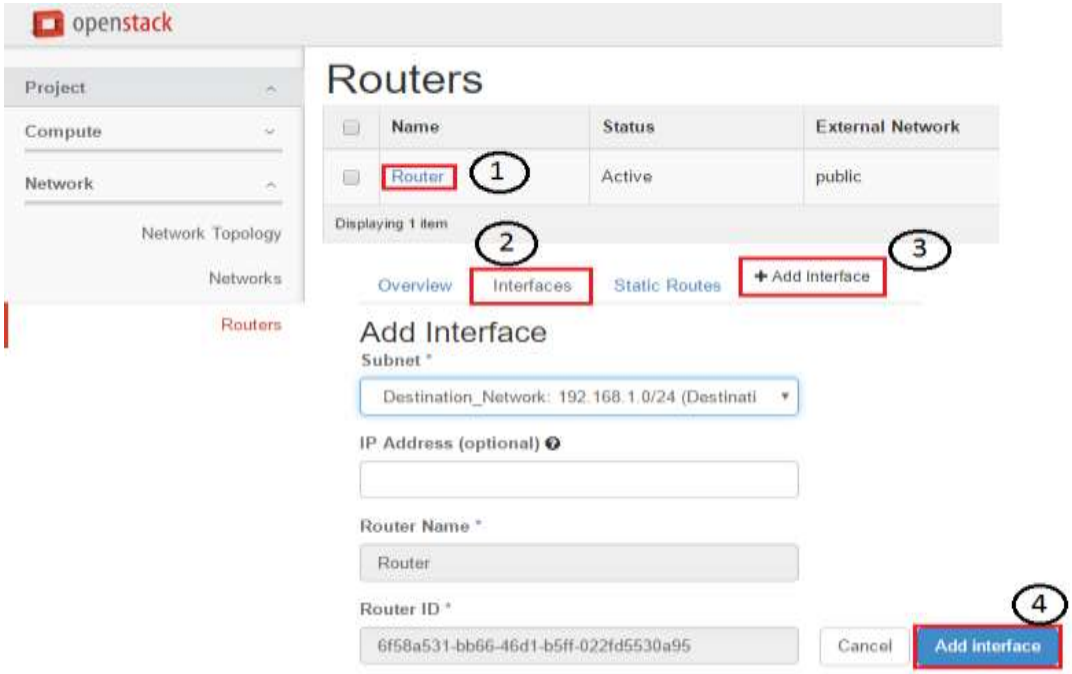
**Εικόνα 11: Κατασκευή του router με χρήση της γραμμής εντολών**

### 4.3.1 Κατασκευή της διεπαφής

Αμέσως μετά την κατασκευή του δρομολογητή ακολουθεί η δημιουργία διεπαφών μεταξύ των 3 δικτύων με τον δρομολογητή. Αυτό είναι πολύ σημαντικό διότι χωρίς αυτό ο δρομολογητής δεν θα γνωρίζει τις διευθύνσεις των κόμβων αλλά ούτε και σε ποιό δίκτυο ανήκουν. Όπως με τα προηγούμενα βήματα, έτσι και σε αυτό η υλοποίηση μπορεί να γίνει τόσο με χρήση της γραφικής διεπαφής όσο και του τερματικού εντολών.

```
neutron router-gateway-set Router public
neutron router-interface-add Router Source_Subnet
Added interface df9a9d89-8f2a-4293-9de3-63b6cd775122 to router Router.
```

Η κατασκευή της διεπαφής δεν παράγει κάποιο γραφικό πίνακα αλλά εκτυπώνει το αντίστοιχο μήνυμα στο τερματικό όπως φαίνεται παραπάνω.
Η ίδια ενέργεια πραγματοποιείται στη γραφική διεπαφή με τον παρακάτω τρόπο:




**Εικόνα 12: Κατασκευή της διεπαφής με χρήση του dashboard**

## 4.4 Διαμόρφωση των εικονικών μηχανών

Με την ολοκλήρωση της δικτύωσης, η διαδικασία συνεχίζει με τον καταμερισμό των υπολογιστικών πόρων. Σε αυτό βοηθούν οι εικονικές μηχανές, δηλαδή οι διαθέσιμοι υπολογιστικοί πόροι οι οποίοι μπορούν να συνθέσουν τα εργαλεία που χρειάζονται για την κατασκευή των τριών κόμβων της δικτύωσης.

### 4.4.1 Security Group

Το OpenStack επιτρέπει την δημιουργία διαφόρων προφίλ ασφάλειας. Αυτά τα προφίλ αποτελούνται από σετ κάνονων που εφαρμόζονται σε μια εικονική μηχανή κατά τη δημιουργία της. Με αυτόν τον τρόπο προσδιορίζεται η δικτυακή πρόσβαση της εικονικής μηχανής. Ένα προφίλ ασφαλείας μπορεί να ελέγξει:

- την -από και προς- κατεύθυνση της ροής δεδομένων
- το πρωτόκολλο επικοινωνίας

- την προέλευση της κίνησης δεδομένων

Εάν δεν έχει επιλεχθεί κάποιο σετ κατά την εκκίνηση της, τότε δίδεται σε αυτή το προκαθορισμένο σετ που επιτρέπει την ελεύθερη κίνηση όλων των δεδομένων από οποιαδήποτε προέλευση. Κατ'όπιν επισκόπησης, διαπιστώθηκε πως για την εικονικοποίηση της οικιακής πύλης είναι απαραίτητη η ύπαρξη τριών μηχανών, με την κάθε 1 να πρεσβεύει και έναν κόμβο όπως περιγράφηκε στο σχήμα προηγούμενης ενότητας.

Κάθε κόμβος, ανάλογα με την λειτουργία του δέχεται και από διαφορετικό προφίλ ασφαλείας, με το κάθε ένα να ελέγχει την προέλευση/προορισμό της κίνησης δεδομένων. Το σπουδαιότερο συμπέρασμα που μπορεί να εξαχθεί είναι πως δεν υπάρχει απ'ευθείας επικοινωνία μεταξύ της πηγής του πολυμεσικού περιεχομένου με τον τελικό προορισμό. Οποιαδήποτε δικτυακή υπηρεσία μεταξύ αυτών των δύο συμβαίνει μόνο μέσα απο τον κόμβο vSTB/vRGW. Περιορισμός στους τύπους δεδομένων μεταξύ των τριών κόμβων δεν υπάρχει.

### 4.4.2 Αυτόματη αρχικοποίηση

Το μοντέλο της εικονικοποιημένης πύλης θα μπορούσε να εξυπηρετήσει έναν τεράστιο αριθμό συνδρομητών. Κρίνοντας από τα μεγέθη (1 για κάθε συνδρομητή) απαιτείται η αυτοματοποίηση των διαδικασιών στο υπολογιστικό νέφος του παρόχου της υπηρεσίας. Αυτό μεταφράζεται ως σωστά παραμετροποιημένες & με προεγκατεστημένες τις απαραίτητες βιβλιοθήκες εικονικές μηχανές οι οποίες θα είναι έτοιμες προς εκμετάλευση.

Το cloud-init είναι το πακέτο που χειρίζεται την αρχικοποίηση μιας εικονικής μηχανής, δηλαδή επεμβαίνει αμέσως μετά την κατασκευή της. Οι οδηγίες δίνονται σε αρχείο κειμένου με μορφή shell script (αρχίζουν με #!) ή cloud config αρχεία (ξεκινούν με \#cloud-config) και περιλαμβάνουν:

- την αναβάθμιση & εγκατάσταση πακέτων λογισμικού
- την διαχείριση του δίσκου
- την εκτέλεση εντολών
- την δημιουργία κλειδιών ασφαλείας

ενώ επιπλέον λειτουργίες μπορούν να γραφούν σε γλώσσα Python.

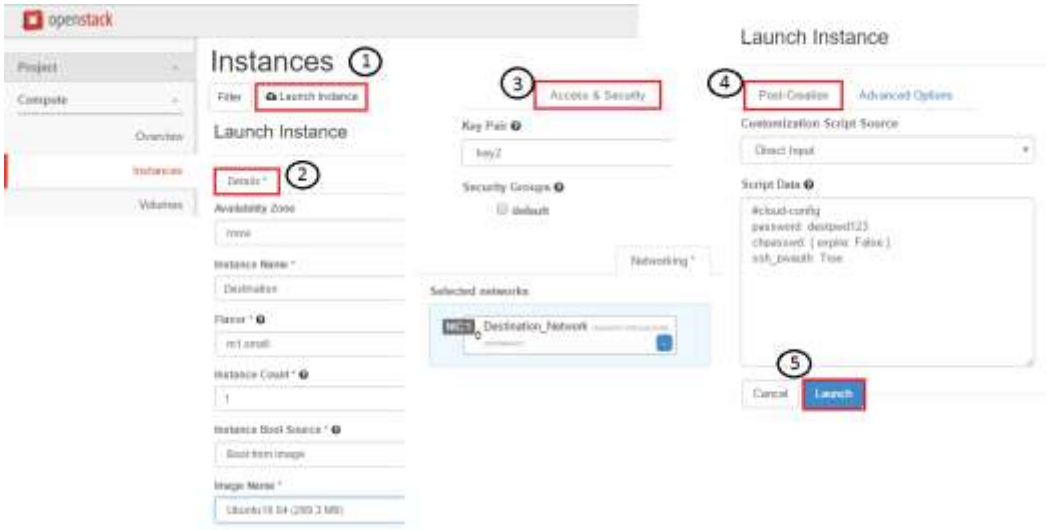
Στο OpenStack υπάρχει ειδικό πεδίο για την τοποθέτηση αυτών των εντολών στο DashBoard ενώ με χρήση της γραμμής εντολών δίνεται ως όρισμα της παραμέτρου user-data στην εντολή nova boot το αρχείο που περιέχει τις οδηγίες:

```
#cloud-config
password: mysecret
chpasswd: { expire: False }
ssh_pwauth: True
apt_update: true
package_upgrade: true
packages:
  - libgtk-3-dev
  - gstreamer1.0-tools
  - gstreamer1.0-plugins-base
  - gstreamer1.0-plugins-good
  - gstreamer1.0-plugins-bad
```

```
  - gstreamer1.0-plugins-ugly
  - gstreamer1.0-libav
  - libgstreamer1.0-dev
  - libgstreamer-plugins-base1.0-dev
  - libgstreamer-plugins-bad1.0-dev
runcmd:
  - mkdir /home/ubuntu/try1
  -       wget       "https://gstreamer.freedesktop.org/src/gst-rtsp-server/gst-rtsp-server-
1.6.0.tar.xz"
    -P /home/ubuntu
  - tar xf /home/ubuntu/gst-rtsp-server-1.6.0.tar.xz  -C /home/ubuntu
```

### 4.4.3 Κατασκευή των εικονικών μηχανών

Επόμενο βήμα προς την ολοκλήρωση της υποδομής είναι η δημιουργία των εικονικών μηχανών, με γνώμονα τις παραγράφους 4.4.1 & 4.4.2 που αφορούν την ασφάλεια και την αρχική τους παραμετροποίηση έτσι ώστε να επιτυγχάνεται στο μέγιστο βαθμό η αυτοματοποίηση της διαδικασίας. Τα βήματα απεικονίζονται στην παρακάτω εικόνα ενώ οι οδηγίες δίνονται και για την γραμμή εντολών.




**Εικόνα 13: Κατασκευή της εικονικής μηχανής με χρήση του dashboard**

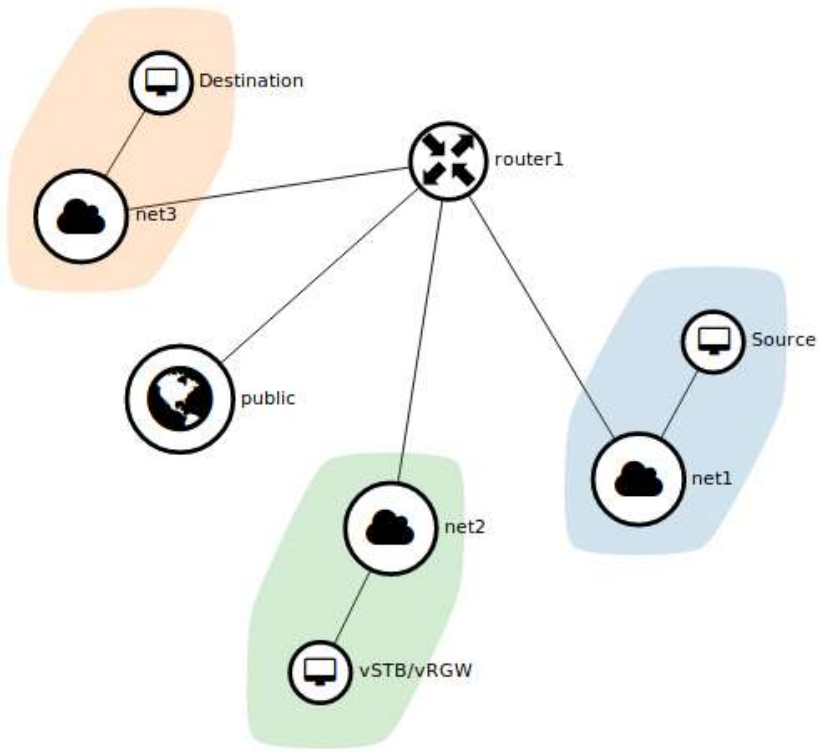
Για την κατασκευή μιας εικονικής μηχανής μέσω του cli χρησιμοποιείται η εντολής nova boot ενώ ως ορίσματα χρησιμοποιούνται οι επιλογές που φαίνονται και στην κατασκευή μέσω του dashboard.

```
stavros@stavros-Lenovo-V110-15ISK:~/Desktop$ nova boot --image Ubuntu16.04 --flavor m1.small --nic net-name=Source_Network --security-groups de
fault  --user-data /home/stavros/Desktop/user_data.file Source
+--------------------------------------+--------------------------------------------------+
| Property                             | Value                                            |
+--------------------------------------+--------------------------------------------------+
| OS-DCF:diskConfig                    | MANUAL                                           |
| OS-EXT-AZ:availability_zone          |                                                  |
| OS-EXT-STS:power_state               | 0                                                |
| OS-EXT-STS:task_state                | -                                                |
| OS-EXT-STS:vm_state                  | building                                         |
| OS-SRV-USG:launched_at               | -                                                |
| OS-SRV-USG:terminated_at             | -                                                |
| accessIPv4                           |                                                  |
| accessIPv6                           |                                                  |
| adminPass                            | EeC8zoiUEkrb                                     |
| config_drive                         |                                                  |
| created                              | 2016-12-12T12:39:17Z                             |
| flavor                               | m1.small (2)                                     |
| hostId                               |                                                  |
| id                                   | 1dd92201-2adb-412d-a33a-678d32b478a4             |
| image                                | Ubuntu16.04 (76f5f4aa-a78f-4703-b738-cab967957431) |
| key_name                             | -                                                |
| metadata                             | {}                                               |
| name                                 | Source                                           |
| os-extended-volumes:volumes_attached | []                                               |
| progress                             | 0                                                |
| security_groups                      | default                                          |
| status                               | BUILD                                            |
| tenant_id                            | ff4f063248034b18bd9375bb2b392d00                 |
| updated                              | 2016-12-12T12:39:18Z                             |
| user_id                              | 0d0ff9ff261e4e7a867c030fc80ee496                 |
+--------------------------------------+--------------------------------------------------+
```

**Εικόνα 14: Κατασκευή της εικονικής μηχανής με χρήση του command line interface**

### 4.4.4 Συνολική απεικόνιση της δικτύωσης

Μόλις ολοκληρώθουν όλα τα βήματα για την κατασκευή της δικτύωσης, ο χρήστης μπορεί να επαληθεύσει το αποτέλεσμα στη γραφική διεπαφή και στο κουμπί επισκόπησης.




**Εικόνα 15: Η απεικόνιση του δικτύου**

# 5. ΒΙΒΛΙΟΘΗΚΕΣ

## 5.1 gstreamer

Το gstreamer παρέχει ένα πλαίσιο κατάλληλο για τη δημιουργία πολυμεσικών εφαρμογών. Τα περισσότερα πλεονεκτήματά του προέρχονται από την αρθρωτή δομή του καθώς για τον σχηματισμό εφαρμογών συνδέονται μεταξύ τους τα κατάλληλα plugin μέσα από μια δεξαμενή στοιχείων. Ο χρήστης που χρησιμοποιεί το gstreamer έχει την δυνατότητα να χτίσει ένα πολυμεσικό σωλήνα, δηλαδή μια σειρά από στοιχεία που χειρίζονται το πολυμεσικό περιεχόμενο. Επίσης του δίνεται η δυνατότητα να δημιουργήσει τα δικά του στοιχεία.

- Στοιχεία (elements): Το στοιχείο είναι η βασικότερη κλάση στοιχείων στο GStreamer. Κάθε στοιχείο επιτελεί μια συγκεκριμένη λειτουργία που μπορεί να εκτείνεται από το διάβασμα ενός αρχείου, την αποκωδικοποίηση δεδομένων, την έγχυση της εξόδου στην κάρτα ήχου κλπ. Μια εγαρμογή συνήθως αποτελείται από μια σειρά από στοιχεία που συνδέονται μεταξύ τους επιτρέποντας με αυτό τον τρόπο να περνάει από μέσα τους η ροή των δεδομένων.
- Pads: Τα pads αποτελούν την είσοδο/έξοδο των στοιχείων που περιγράφηκαν παραπάνω. Σκοπός τους είναι η διαπραγμάτευση για την σύνδεση των στοιχείων αλλά και τη μεταφορά των δεδομένων μεταξύ τους.Τα pads έχουν συγκεκριμένες δυνατότητες χειραγώγησης δεδομένων, επιτρέποντας τη ροή μόνο συγκεκριμένων τύπων δεδομένων . Με ένα αλληγωρικό παράδειγμα, ένα pad μπορεί να θεωρηθεί και ως το βύσμα στα ήχεια του υπολογιστή.
- Bins: Η μείωση της πολυπλοκότητας σε μια εφαρμογή συνεπικουρείται από τα bin, τα οποία αποτελούν 'δοχεία' για συλλογές από στοιχεία. Επειδή το bin αποτελεί υποκλάση των στοιχείων που περιγράφηκαν παραπάνω, ο χρήστης μπορεί να αντιμετωπίσει το ίδιο και ως ένα στοιχείο. Με αυτό τον τρόπο, αλλάζοντας πχ. την κατάσταση στο bin, αλλάζει και η κατάσταση των επιμέρους στοιχείων από τα οποία αποτελείται, ενώ η ροή των ενεργειών υποστηρίζεται και από τις δύο κατευθύνσεις, δηλαδή από το bin προς τα επί μέρους στοιχεία του αλλά και από τα στοιχεία του προς το αυτό.
- Pipelines: Το pipeline είναι το ανώτατο επίπεδο που μπορεί να φτάσει ένα bin. Παρέχει ένα δίαυλο από και προς την εφαρμογή και διαχειρίζεται τον συγχρονισμό των στοιχείων. Η ροή των δεδομένων και η επεξεργασία τους ξεκινά, μόλις το pipeline αλλάξει κατάσταση σε PLAYING.

Το gstreamer διαθέτει και άλλα στοιχεία όπως buffers για τη μεταφορά των δεδομένων μεταξύ των στοιχείων, γεγονώτων μεταξύ των στοιχείων ή μεταξύ της εφαρμογής και των στοιχείων. Επίσης το gstreamer διαθέτει ένα μηχανισμό μηνυμάτων, δηλαδή αντικειμένων που εκδίδονται από τα στοιχεία στο δίαυλο και από τα δίαυλο παρέχονται στην εφαρμογή για αντιμετώπιση.

Στην επόμενη ενότητα παρουσιάζεται ο κώδικας της οικιακής πύλης με συνοδεία αναλυτικής περιγραφής και των στοιχείων που την αποτελούν.

## 5.2 gLib

Η βιβλιοθήκη gLib προσφέρει τα εργαλεία για την κατασκευή εφαρμογών και βιβλιοθηκών στη γλώσσα προγραμματισμού C καθώς ένα μεγάλο εύρος εργαλείων και δομών δεδομένων. Προσφέρει επίσης τον *κυρίως βρόχο* (main loop) δηλαδή έναν βρόχο που ελέγχει συνεχώς τους πόρους και τα γεγονότα που χειρίζονται οι χειριστές γεγονότων (event handlers).

### 5.3 GObject

Οι περισσότερες γλώσσες προγραμματισμού έρχονται με τα δικά τους συστήματα αρχείων και τις δικές του βασικές αλγοριθμικές δομές. Έτσι όπως η βιβλιοθήκη GLib υλοποιεί αυτές τις δομές, έτσι και η βιβλιοθήκη GLib Object (GObject) παρέχει τις απαραίτητες υλοποιήσεις για ένα ευέλικτο αντικειμενοστραφές πλαίσιο για την γλώσσα C.

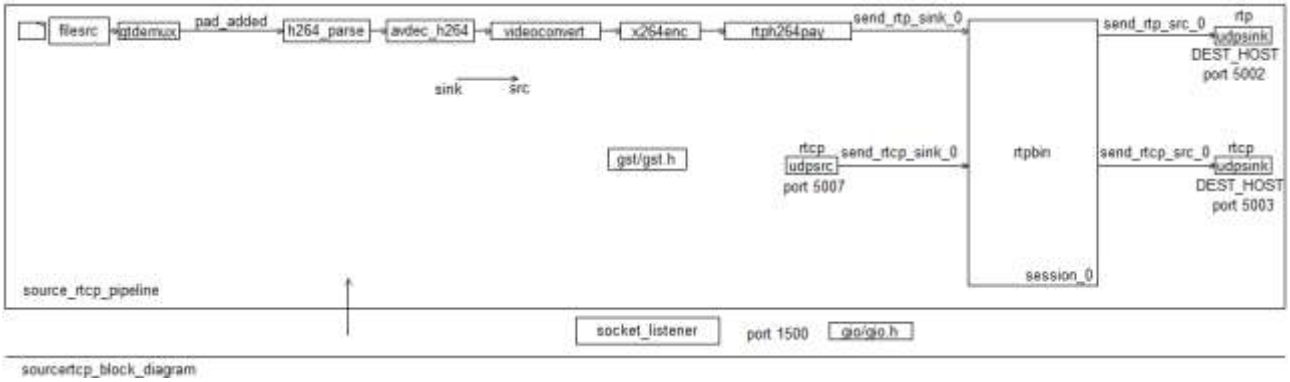
### 5.4 GTK+

Το GTK+ είναι 'εργαλειοθήκη' που χρησιμοποιείται για την κατασκευή γραφικών διεπαφών. Τα εργαλεία της είναι κυρίως γραφικά στοιχεία ελέγχου (widgets) γραμμένα στη γλώσσα προγραμματισμού C. Η βιβλιοθήκη του GTK+ χρησιμοποιεί την βιβλιοθήκη GObject για να αποδώσει αντικειμενοστρεφή χαρακτηριστικά ενώ η ίδια μπορεί να χρησιμοποιηθεί από διάφορα λειτουργικά συστήματα.

# 6. ΧΑΡΑΚΤΗΡΗΣΤΙΚΑ ΤΗΣ ΟΙΚΙΑΚΗΣ ΠΥΛΗΣ

Η σύγκλιση του μοντέλου τοπολογίας, των πρωτοκόλλων δικτύωσης, των εργαλείων διαχείρισης του υπολογιστικού νέφους, της βιβλιοθήκης χειρισμού των πολυμέσων & και των βοηθητικών εργαλείων διαγράφει το πλαίσιο μέσα στο οποίο κατασκευαζέται η οικιακή πύλη. Για τον ολοκληρωμένο έλεγχο της όμως κρίθηκε απαραίτητη η ύπαρξη μιας πηγής που κάνει εκπομπή προς αυτήν αλλά και η ύπαρξη ενός τελικού δέκτη με σκοπό την προβολή του πολυμεσικού περιεχομένου. Έτσι η οικιακή πύλη αποκτάει τον ρόλο του αναμεταδότη από την πηγή προς τον τελικό προορισμό ενώ ταυτόχρονα δέχεται τον έλεγχο του πελάτη. Με αυτόν τον τρόπο ο δέκτης δεν έχει καμμία απευθείας επικοινωνία με τον πομπό και το αντίστροφο ενώ οι όποιες αλλαγές προκύψουν στην εσωτερική δομή του υπολογιστικού νέφους, θα αφήσουν ανεπιρέαστο τον δέκτη. Έτσι κάθε κόμβος του δικτύου, ανάλογα το ρόλο του έχει και ένα εκτελέσιμο αρχείο στο οποίο περιέχονται οι κατάλληλες λειτουργίες. Για την απλοποίηση των διαδικασιών, όλες οι λειτουργίες αφορούν μόνο τη ροή βίντεο από ένα αρχείο mp4.

Στον κόμβο προέλευσης διαβάζεται το αρχείο το οποίο περιέχει τις ροές εικόνας ήχου σε μια πεπλεγμένη μορφή κωδικοποιημένων ροών ενώ έπονται οι ενέργειες της αποπολύπλεξης, αποσυμπίεσης και αποκωδικοποίησης, ανάλυσης βάσει του τύπου της ροής, εκ νέου κωδικοποίηση, κατάτμιση σε πακέτα rtp και δημιουργίας πακέτων rtcp και έξοδος προς το δίκτυο σε ενθυλακωμένα πακέτα udp.

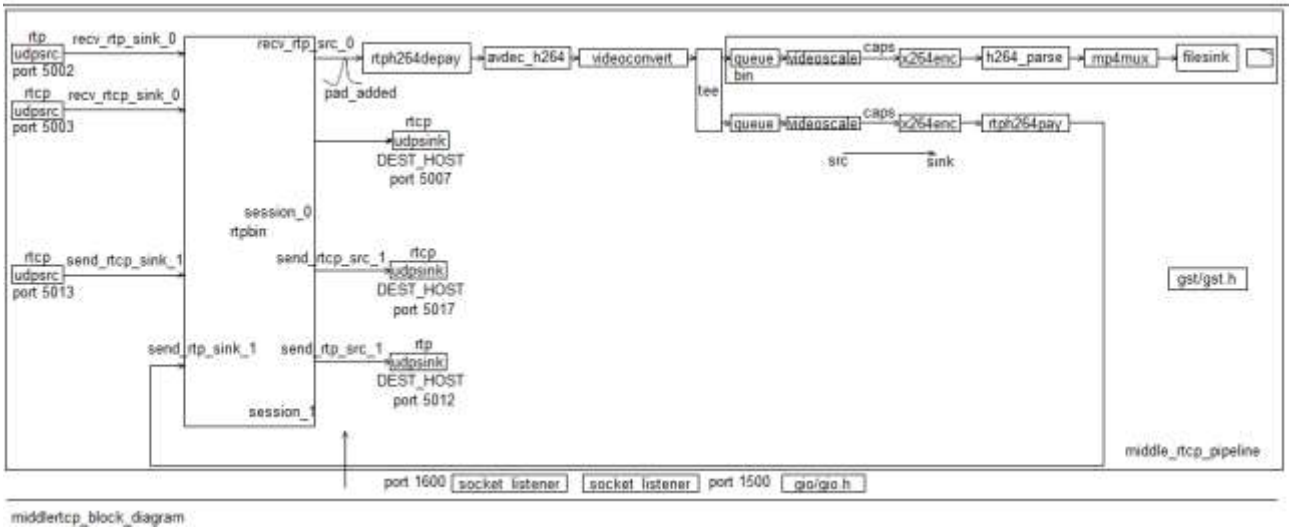



**Εικόνα 16: Το μπλοκ-διάγραμμα του κόμβου προέλευσης.**

Το εννοιολογικό μπλοκ διάγραμμα με τα αντίστοιχα elements του gstreamer δίνεται στην παραπάνω εικόνα.

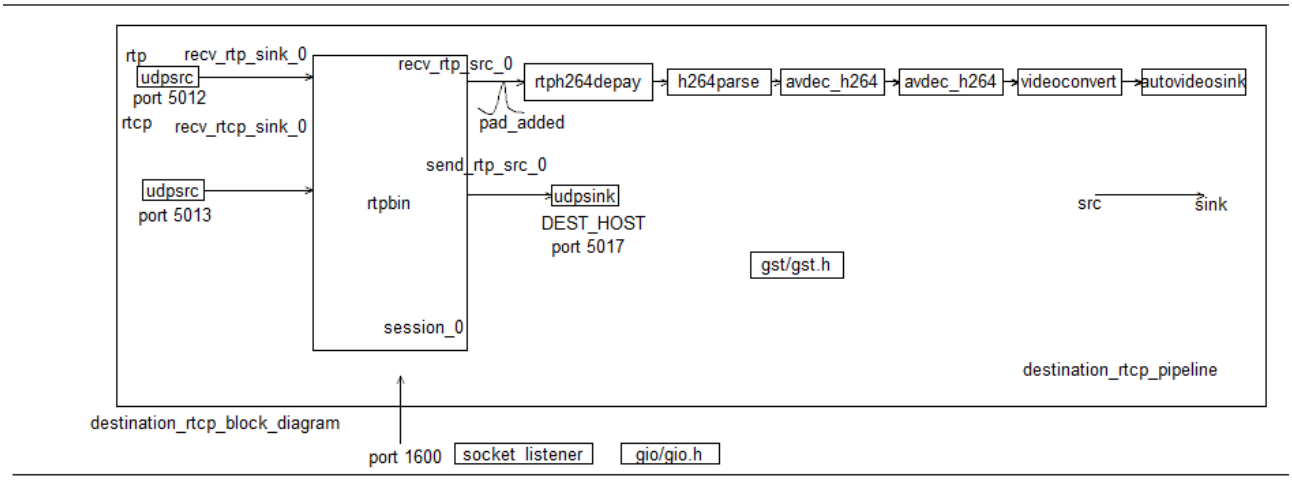
Στον ενδιάμεσο κόμβο λαμβάνονται τα πακέτα udp, αποθηλακώνονται σε πακέτα rtp, τοποθετούνται σε σωστή σειρά και αναλύονται σε ροή βίντεο βάσει του πρωτοκόλλου συμπίεσης. Στη συνέχεια η ροή του βίντεο αντιγράφεται σχηματίζοντας έτσι δύο ίσοδύναμες ροές που επεξεργάζονται διαφορετικά.

Η πρώτη ροή καταλήγει σε αρχείο το οποίο αποθηκεύεται στον κόμβο της οικιακής πύλης για προβολή κάποια άλλη στιγμή ενώ η δεύτερη ροή τροποποιείται δυναμικά στην ανάλυση της για να αποδόσει την ποιότητα της υπηρεσίας (βλ. παράγραφος QoS). Εν συνεχεία ενθυλακώνεται σε πακέτα rtp, ύστερα σε πακέτα udp και αποστέλλεται στο δίκτυο του πελάτη.




**Εικόνα 17: Το μπλοκ-διάγραμμα του κόμβου της οικιακής πύλης.**

Στον κόμβο του δέκτη γίνεται η λήψη των πακέτων udp, αποθηλάκωση σε πακέτα rtp, ευθυγράμμιση και επεξεργασία βάσει του πρωτοκόλλου συμπίεσης και τέλος προβολή στην οθόνη του δέκτη. Ταυτόχρονα στην οθόνη του χρήστη εμφανίζεται και η γραφική διεπαφή του χρήστη με όλες τις παρεχόμενες λειτουργίες.




**Εικόνα 18: Το μπλοκ-διάγραμμα του κόμβου προορισμού.**

Η γραφική διεπαφή αποτελεί τον έλεγχο του χρήστη που εφαρμόζεται στις εικονικές μηχανές της πηγής και της οικιακής πύλης. Μέσω αυτής, αποτέλεσμα της βιβλιοθήκης gtk+3.0, γίνεται η επιλογή της λειτουργίας, που μεταφέρεται σε πακέτα udp στα δίκτυα της πηγής και της οικιακής πύλης, αποτέλεσμα της βιβλιοθήκης GIO, αποκωδικοποιείται και τέλος εκτελείται με χρήση της βιβλιοθήκης του gstreamer.

Σε κάθε περίπτωση τα 3 pipeline μοιράζονται την ίδια κατάσταση: Αναπαραγωγής ή Παύσης.

Αρχικά η κατάσταση των πολυμεσικών pipeline και στους 3 κόμβους είναι σε κατάσταση παύσης, δίνοντας έτσι την δυνατότητα στο χρήστη να εκκινήσει την εφαρμογή όποτε

αυτός επιθυμεί. Μόλις ο χρήστης πατήσει το κουμπί της εκκίνησης, η εντολή μεταφέρεται και εκτελείται σε κάθε ένα από τους κόμβους. Αυτό έχει ως αποτέλεσμα την ροή των πακέτων από τον ένα κόμβο στον άλλο και μόλις τα δεδομένα φτάσουν στον δέκτη, αυτά απεικονίζονται στην οθόνη του. Στη συνέχεια και κατά βούληση του χρήστη μπορεί να γίνει η παύση της ροής. Η ενέργεια αυτή εφαρμόζεται και στους τρεις κόμβους διότι διαφορετικά το αποτέλεσμα θα είναι εκτός προδιαγραφών:

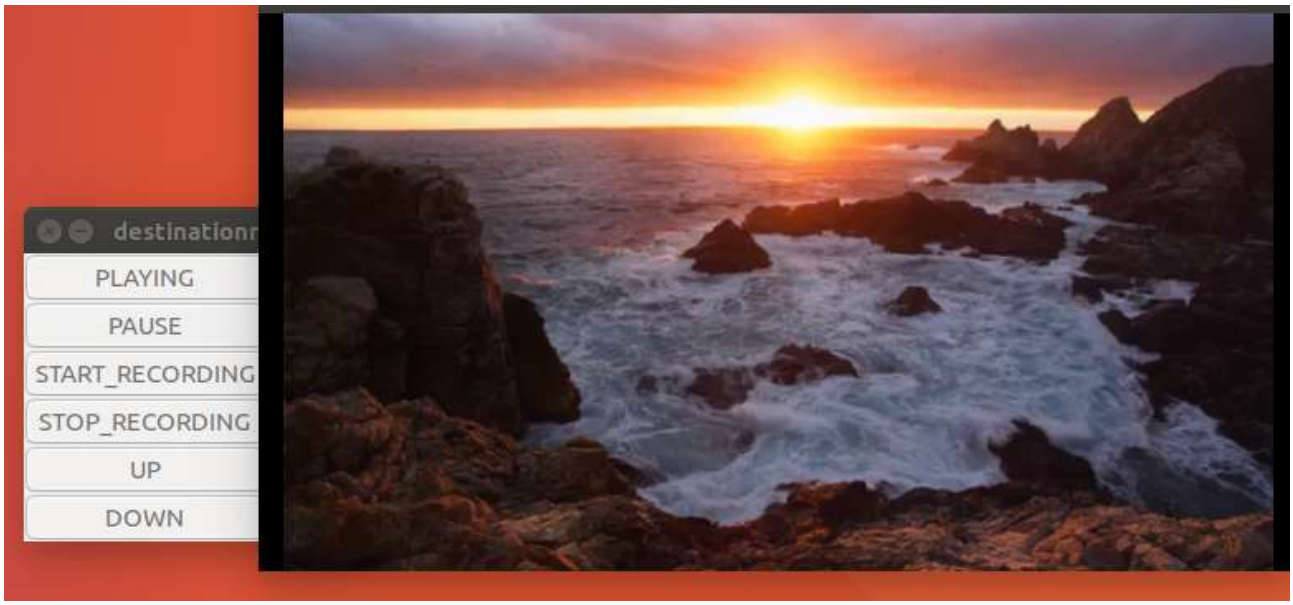
- Εάν εφαρμοστεί μόνο στον κόμβο της πηγής και στον κόμβο της οικιακής πύλης έχει δωθεί η εντολή της εγγραφής, τότε όταν επιλεγεί ο τερματισμός εγγραφής, το αρχείο που θα παραχθεί θα έχει αποθηκευμένο το παγωμένο καρέ για όση διάρκεια κράτησε η παύση. Αυτό συμβαίνει διότι ο ενδιάμεσος κόμβος δεν έχει περάσει στην κατάσταση της παύσης, με αποτέλεσμα να αποθηκεύει ότι έχει ως είσοδο, δηλαδή το τελευταίο καρέ που έλαβε από την πηγή.
- Εάν εφαρμοστεί μόνο στον ενδιάμεσο κόμβο, ο κόμβος της πηγής θα συνέχισει να στέλνει την ροή πακέτων (γιατί δεν πέρασε ποτέ στην παύση), αγνοώντας την κατάσταση του, με αποτέλεσμα να υπάρχει σπατάλη δεδομένων και καταστροφή των καρέ γιατί η πηγή κάνει εκπομπή και ο ενδιάμεσος δεν προχωράει σε λήψη και αναμετάδοση. Όταν ο χρήστης επιλέξει την συνέχεια της αναπαραγωγής, αυτή θα ξεκινήσει από το σημείο που βρίσκεται ο πομπός, σημείο διαφορετικό από αυτό της παύσης.

Η οικιακή πύλη σύμφωνα με το αρχικό πλάνο διεξάγει την εγγραφή του πολυμεσικού περιεχομένου, έτσι ώστε αυτό να είναι διαθέσιμο στον χρήστη όταν αυτός το επιθυμεί. Όταν ο χρήστης επιλέγει την εκκίνηση της εγγραφής, τότε προστίθενται μερικά στοιχεία που αποτελούν έναν κλάδο του πολυμεσικού pipeline της οικιακής πύλης. Όταν επιλεγεί ο τερματισμός της εγγραφής, ο κλάδος αποκόπτεται, του δίνεται EOS για να παραχθεί το αρχείο της εγγραφής και στη συνέχεια αφαιρείται από το pipeline μέχρι να επιλεγεί εκ νέου η εκκίνηση της εγγραφής. Κατά τη διάρκεια της εγγραφής είναι δυνατόν να περάσει το pipeline στην κατάσταση της πάυσης και άρα και ο βρόχος της εγγραφής.

### 6.1 Γραφική διεπαφή χρήστη

Η γραφική διεπαφή του χρήστη με την υπηρεσία έχει σχεδιαστεί όμοια με αυτή του τηλεκοντρόλ για την τηλεόραση, πρόκειται δηλαδή για ένα πλαίσιο με πλήκτρο
όπου το κάθε ένα αντιστοιχεί και σε μία λειτουργία.
Αρχικά κατασκευάζεται το πλαίσιο, μέσα στο οποίο θα τοποθετηθούν τα πλήκτρα.

```
GtkWidget *window_control, *grid;
g_print("CREATING GRID\n");
window_control = gtk_window_new
            (GTK_WINDOW_TOPLEVEL);
gtk_window_set_resizable (GTK_WINDOW
            (window_control), FALSE);
gtk_window_move (GTK_WINDOW (window_control), 10,10);
grid = gtk_grid_new ();
gtk_container_add (GTK_CONTAINER (window_control),
    grid);
```




**Εικόνα 19: Η γραφική διεπαφή του χρήστη**

Μόλις ολοκληρωθεί η κατασκευή του πλαισίου, δημιουργούνται τα εικονικά πλήκτρα που θα τοποθετηθούν μέσα σε αυτό. Όλα τα κουμπιά αναγνωρίζουν το γεγονός της επιλογής τους, δηλαδή πότε γίνεται η επιλογή τους με κλικ. Μόλις συμβεί ένα τέτοιο γεγονός, το πλήκτρο εκπέμπει ένα κατάλληλο σήμα το οποίο αντιμετωπίζεται καλώντας την κατάλληλη callback συνάρτηση μέσω της συνάρτησης g_signal_connect.

- Πλήκτρο PLAYING: Τροποποιεί την κατάσταση των pipeline των τριών κόμβων σε PLAYING. Είναι απαραίτητο ο χρήστης να πιέσει πρώτα αυτό το πλήκτρο για να ξεκινήσει η ροή του βίντεο. Το πλήκτρο αποκτά ξανά ισχύ όταν τα pipeline είναι σε κατάσταση PAUSE, οπότε και πρέπει να ξαναγυρίσουν σε PLAYING, κατ'όπιν επιλογής του χρήστη.

```
   //control source play.
   g_print("CREATING PLAY BUTTON\n");
   button_state_play = gtk_button_new_with_label
                ("PLAYING");

        g_signal_connect (G_OBJECT (button_state_play),   "clicked",G_CALLBACK
(source_state_play),pipeline);

        gtk_grid_attach (GTK_GRID (grid), button_state_play,
        0, 1, 1, 1);
   gtk_widget_show (button_state_play);


static void
source_state_play (GtkWidget * widget, GstElement * pipeline)
{
      g_print ("ATTEMPTING TO SET SOURCE ON PLAY
        STATE\n");
      source_state_change ("1");
```

```
    gst_element_set_state (pipeline,
      GST_STATE_PLAYING);
}
```

Στον παραπάνω κώδικα φαίνεται η κατασκευή του πλήκτρου PLAYING καθώς και η callback συνάρτηση που καλείται από το γεγονός "clicked".

- Πλήκτρο PAUSED: Μεταβάλλει την κατάσταση των pipeline των τριών κόμβων σε PAUSED. Ο χρήστης κατ'επιλογήν του έχει την ικανότητα να παγώσει το πολυμεσικό περιεχόμενο πιέζοντας αυτό το κουμπί ενώ σε περίπτωση που κάνει εγγραφή, παγώνει και η εγγραφή στον ενδιάμεσο κόμβο, αυτόν δηλαδή της εικονικοποιημένης οικιακής πύλης.

```
    //control source pause
    g_print("CREATING PAUSE BUTTON\n");
    button_state_pause = gtk_button_new_with_label
            ("PAUSE");
    g_signal_connect
                    (G_OBJECT (button_state_pause),
      "clicked",G_CALLBACK(source_state_pause),
                     pipeline);

    gtk_grid_attach (GTK_GRID (grid),
      button_state_pause, 0,2, 1, 1);
    gtk_widget_show (button_state_pause);
```

Και η callback συνάρτηση:

```
static void source_state_pause
          (GtkWidget * widget, GstElement * pipeline)
{
    g_print ("ATTEMPTING TO SET SOURCE ON PAUSE
      STATE\n");
    source_state_change ("-1");
    gst_element_set_state (pipeline,
      GST_STATE_PAUSED);
}
```

- Πλήκτρο START_RECORDING: Εκκινεί την εγγραφή του πολυμεσικού περιεχομένου. Κάνοντας click ο χρήστης σε αυτό το πλήκτρο, στέλνει ένα σήμα στην εικονικοποιημένη οικιακή πύλη για να ξεκινήσει την αποθήκευση της ροής του βίντεο.

```
    //control start recording
    g_print("CREATING START RECORDING BUTTON\n");
    button_start_record = gtk_button_new_with_label
            ("START_RECORDING");

    g_signal_connect (G_OBJECT (button_start_record),   "clicked",
      G_CALLBACK(start_recording_cb),pipeline);

    gtk_grid_attach (GTK_GRID (grid),
      button_start_record, 0, 3, 1, 1);
    gtk_widget_show (button_start_record);
```

```
static void
start_recording_cb (GtkWidget * widget, GstElement * pipeline)
{
        g_print ("ATTEMPTING TO START RECORDING\n");
        source_state_change ("2");
}
```

- Πλήκτρο STOP_RECORDING: Τερματίζει την εγγραφή του πολυμεσικού περιεχομένου. Όπως και με το πλήκτρο εγγραφής, έτσι και με το πλήκτρο τερματισμού της, ο χρήστης έχει τον έλεγχο της εικονικοποιημένης οικιακής πύλης.

```
//control stop recording
g_print("CREATING STOP RECORDING BUTTON\n");
button_stop_record = gtk_button_new_with_label
  ("STOP_RECORDING");
g_signal_connect (G_OBJECT (button_stop_record),
  "clicked",G_CALLBACK (stop_recording_cb),  pipeline);

gtk_grid_attach (GTK_GRID (grid),
  button_stop_record, 0, 4, 1, 1);
gtk_widget_show (button_stop_record);
```

```
static void
stop_recording_cb (GtkWidget * widget, GstElement * pipeline)
{
        g_print ("ATTEMPTING TO STOP RECORDING\n");
        source\_state_change ("-2");
}
```

- Πλήκτρο UP: Με αυτό το κουμπί ο χρήστης μπορεί να ανεβάσει την ποιότητα του βίντεο που παρακολουθεί. Η τροποποίηση αυτή συμβαίνει στον κόμβο της εικονικοποιημένης οικιακής πύλης, επάνω στη ροή του βίντεο που επεξεργάζεται.

```
//control up resolution
g_print("CREATING RESOLUTION UP BUTTON\n");
up = gtk_button_new_with_label ("UP");

g_signal_connect (G_OBJECT (up), "clicked",
      G_CALLBACK (up_cb), pipeline);

gtk_grid_attach (GTK_GRID (grid), up, 0, 5, 1, 1);

gtk_widget_show (up);
```

```
static void
up_cb(GtkWidget * widget, GstElement * pipeline)
{
        g_print("Changing resolution higher\n");
        source_state_change ("3");

}
```

- Πλήκτρο DOWN: Όπως και με το κουμπί UP, ο χρήστης επεμβαίνει απομακρυσμένα επάνω στην ποιότητα του βίντεο, αυτή τη φορά όμως μειώνοντας την ποιότητα του. Αποτέλεσμα αυτού είναι η μείωση της

εισερχόμενης κίνησης. Παρακάτω δίνεται ο κώδικας για το κουμπί DOWN αλλά και η συνάρτηση που καλείται όταν αυτό πιεστεί:

```
//control down resolution
g_print("CREATING RESOLUTION DOWN BUTTON\n");
down = gtk_button_new_with_label ("DOWN");
g_signal_connect (G_OBJECT (down), "clicked",
      G_CALLBACK (down_cb), pipeline);
gtk_grid_attach (GTK_GRID (grid), down,
                                          0, 6, 1, 1);
gtk_widget_show (down);
```

```
static void
down_cb(GtkWidget * widget, GstElement * pipeline)
{
        g_print("Changing resolution lower \n");
        source_state_change ("-3");
}
```

### 6.2 vRGW & vSTB

Στο παραδοσιακό μοντέλο της αρχιτεκτονικής, η οικιακή πύλη βρισκόταν εντός των εγκαταστάσεων του χρήστη. Αυτό σημαίνει την υποδοχή της πληροφορίας από το δίκτυο του παρόχου και την προβόλη στην οθόνη της τηλεόρασης. Στο νέο μοντέλο της αρχιτεκτονικής,η εικονικοποιημένη πια οικιακή πύλη βρίσκεται εντός του δικτύου του παρόχου και μακριά από τον χώρο του πελάτη. Αυτό συνεπάγεται πως η οικιακή πύλη θα πρέπει να δέχεται τη ροή των δεδομένων από τον πάροχο αλλά θα πρέπει και να τη διαβιβάζει στο δίκτυο του πελάτη.

#### 6.2.1 Έναρξη και λήξη της αποστολής

Η διανομή του περιεχομένου μπορεί να ξεκινήσει ομαλά όταν και τα τρία πολυμεσικά pipelines έχουν δημιουργηθεί. Κατά την εκκίνηση τους, τοποθετούνται σε κατάσταση PAUSED, έτσι ώστε να δωθεί η ευκαιρία στον χρήστη να εκκινήσει την προβολή μέσω της γραφικής διεπαφής, όποτε θελήσει αυτός πιέζοντας το πλήκτρο PLAYING. Μόλις συμβεί αυτό, η κατάσταση του κόμβου του χρήστη αλλάζει σε PLAYING αλλά επίσης στέλνεται το σήμα για εκκίνηση στον ενδιάμεσο κόμβο.

Μόλις γίνει λήψη από τον ενδιάμεσο κόμβο του σήματος για έναρξη προβολής, η αρχική κατάσταση του pipeline περνάει επίσης από PAUSED σε PLAYING ενώ το μήνυμα μεταβιβάζεται και στον κόμβο προέλευσης. Όλα αυτά φαίνονται στις παρακάτω συναρτήσεις που εκτελούνται στον ενδιάμεσο κόμβο διότι μόνο σε αυτόν γίνεται ταυτόχρονη λήψη και εκπομπή αντίστοιχων μηνυμάτων:

```
//PLAYING STATE
else if (res == 1)
{
  g_print("Attempting to set middle state to
        PLAYING:\n");
  gst_element_set_state (pipeline, GST_STATE_PLAYING);
  transmit (message);
}
```

```
void
transmit (gchar *message)
{
  GError * error = NULL;
  /* create a new connection */
  GSocketConnection * connection = NULL;
  GSocketClient * client = g_socket_client_new();

  /* connect to the host */
  connection = g_socket_client_connect_to_host (client,
                                          (gchar*)"localhost",1500, NULL,
                                         &error);
  /* don't forget to check for errors */
  if (error != NULL)
  {
    g_error (error->message);
  }
  else
  {
    g_print ("Connection successful!\n");
  }
  /* use the connection */
  GInputStream * istream = g_io_stream_get_input_stream
                  (G_IO_STREAM (connection));
  GOutputStream * ostream = g_io_stream_get_output_stream
                  (G_IO_STREAM (connection));
  g_output_stream_write  (ostream, message, 13, NULL,
                           &error);
  /* don't forget to check for errors */
  if (error != NULL)
  {
    g_error (error->message);
  }
}
```

Τέλος, στον κόμβο προέλευσης μόλις ληφθεί το μήνυμα, τροποποείται η κατάστασή του σε PLAYING σηματοδοτόντας την έναρξη της αποστολής του βίντεο.

Η αντίστοιχη διαδικασία στον τερματισμό του βιντέο έχει αντίθετη φορά, καθώς στην παρούσα υλοποίηση τυχόν τερματισμός του περιεχομένου εκλαμβάνεται και ως τερματισμός της λειτουργίας όλων των κόμβων της τοπολογίας. Έτσι μόλις λήξει η ροή, αποστέλεται ένα πακέτο RTCP BYE προς την εικονικοποιημένη οικιακή πύλη, το οποίο μόλις ληφθεί μεταβιβάζεται από αυτήν προς τον χρήστη. Στη συνέχεια το κάθε pipeline τερματίζει με φυσιολογικό τρόπο. Η υλοποίηση των παραπάνω περιγράφεται στο απόσπασμα του κώδικα της εικονικοποιημένης πύλης. Όπως και στην έναρξη της εκπομπής, είναι ο μόνος κόμβος που εκτελεί εκπομπή και λήψη:

```
/*Signals*/
g_signal_connect (rtpbin, "on-bye-ssrc", G_CALLBACK (on_bye_cb), pipeline);
```

Αρχικά δηλώνεται το event της λήψης του bye πακέτου και στη συνέχεια η callback συνάρτηση που θα το διαχειριστεί:

```
static void
on_bye_cb (GstElement * rtpbin, GstPad * new_pad,
        GstElement * pipeline)
{
  g_print ("RECEIVED BYE");
  g_print ("End of stream\n");
  sleep(2);
  g_main_loop_quit (loop);
  g_print ("Returned, stopping playback\n");
}
```

### 6.2.2 Έλεγχος μετάδοσης πολυμεσικής ροής

Η μετάδοση και προβολή του πολυμεσικού περιεχομένου γίνεται κατά βούληση του χρήστη της υπηρεσίας. Αυτό, του δίνει την ευκαιρία είτε να μην χάσει κάποιο κομμάτι του αρχείου αφού μπορεί να συνεχίσει την προβολή του στο σημείο που σταμάτησε σε δεύτερο χρόνο αλλά επίσης και να απελευθερώσει του πόρους του δικτύου για κάποια άλλη χρήση.

Η διαδικασία του ελέγχου μετάδοσης έχει φορά αντίθετη από τη φορά της πολυμεσικής ροής, δηλαδή ξεκινάει από τον οικιακό κόμβο και τελειώνει στον κόμβο του πομπού, ενώ εκτελείται σε τρία βήματα, ένα για κάθε κόμβο της τοπολογίας του δικτύου:

- Στον κόμβο του χρήστη, το πολυμεσικό PIPELINE αλλάζει κατάσταση σε PAUSE (PLAYING → PAUSE) ενώ ειδοποιείται και ο ενδιάμεσος κόμβος μέσω της συνάρτησης source_state_play πως πρέπει να περάσει και αυτός στην ίδια κατάσταση
- Ο ενδιάμεσος κόμβος, μόλις λάβει το μήνυμα αλλαγής κατάστασης περνάει και αυτός σε κατάσταση PAUSE, ενώ με τη σειρά του ειδοποιεί και τον κόμβο προέλευσης πως θα πρέπει να παγώσει την αποστολή του. Αυτά περιγράφονται στα σώματα των συναρτήσεων που ακολουθούν:

```
//Message identification
gboolean incoming_callback  (GSocketService *service,
        GSocketConnection *connection, GObject  *source_object,
                        GstPad * tee_save_pad)
{
        …
  //PAUSED identified
  if (res == -1)
  {
        g_print("Attempting to set middle state to        PAUSED:\n");
    gst_element_set_state (pipeline,       GST_STATE_PAUSED);
    transmit (message);
  }
```

Η παραπάνω συνάρτηση ανιχνεύει κάποιο μήνυμα από τον δέκτη, το αποκωδικοποιεί και αφού αναγνωρίσει πως πρέπει να αλλάξει η κατάσταση σε PAUSED, προχωρεί σε αυτή την κατάσταση ενώ ειδοποιεί και τον κόμβο αφετηρίας με την συνάρτηση transmit:

```
void
transmit (gchar *message)
{
  GError * error = NULL;
  /* create a new connection */
  GSocketConnection * connection = NULL;
  GSocketClient * client = g_socket_client_new();
  /* connect to the host */
  connection = g_socket_client_connect_to_host
        (client, (gchar*)"localhost",1500,                    NULL, &error);
  /* don't forget to check for errors */
  if (error != NULL)
  {
        g_error (error→message);
  }
  else
  {
    g_print ("Connection successful!\n");
  }
  /* use the connection */
  GInputStream * istream =
        g_io_stream_get_input_stream
        (G_IO_STREAM (connection));
  GOutputStream * ostream =
        g_io_stream_get_output_stream
        (G_IO_STREAM (connection));

  g_output_stream_write  (ostream,message, 13, NULL,
                        &error);
  /* check for errors */
  if (error != NULL)
  {
    g_error (error→message);
  }
}
```

- Ο κόμβος προέλευσης μόλις λάβει το μήνυμα παύσης, αλλάζει σε κατάσταση PAUSE και παραμένει εκεί μέχρι να ειδοποιηθεί για αλλαγή:

```
/* connect to the port */
g_socket_listener_add_inet_port((GSocketListener*)          service,  1500,
NULL, &error);
```

Η συνάρτηση που προετοιμάζει την πόρτα 1500 να δεχθεί το μήνυμα από τον κόμβο vSTB/vRGW δίνεται παρακάτω καθώς και το σώμα της callback συνάρτησης:

```
/* listen to the 'incoming' signal */
g_signal_connect (service, "incoming", G_CALLBACK
        (incoming_callback), pipeline);
```

```
gboolean incoming_callback  (GSocketService *service,
        GSocketConnection *connection, GObject *source_object, gpointer
                                    pipeline)
{
  gint res;
  g_print("Received Connection from client!\n");
  GInputStream * istream = g_io_stream_get_input_stream (G_IO_STREAM
                                                (connection));
  gchar message[1024];
  g_input_stream_read  (istream, message,1024, NULL,
                  NULL);
  g_print("Message was: %s\n", message);
  res = atoi(message);
  g_print ("Result:%d\n", res);
  if (res == -1)
  {
    g_print("Attempting to set state to PAUSED:");
    gst_element_set_state (pipeline, GST_STATE_PAUSED);
  }
  return FALSE;
}
```

Η παραπάνω συνάρτηση ορίζει το event incoming και την callback συνάρτηση που θα το διαχειριστεί.

### 6.2.2 Κατασκευή της δικτύωσης

Σύμφωνα με τα παραπάνω ο κόμβος της εικονικοποιημένης οικιακής πύλης δέχεται τα rtp πακέτα δεδομένων σε ένα udp port, ενώ ταυτόχρονα από ένα άλλος ζεύγος udp ports λαμβάνει και στέλνει rtcp πακέτα πίσω στην πηγή. Παράλληλα από ένα άλλο udp port εκπέμπει τα rtp πακέτα προς το οικιακό δίκτυο και χρησιμοποιεί άλλο ένα ζεύγος udp ports για αποστολή και λήψη rtcp πακέτων από τον πελάτη. Τέλος η οικιακή πύλη χρησιμοποιεί άλλο ένα ζεύγος από udp ports για την λήψη κωδικοποιημένων εντολών από τον πελάτη και μεταβίβαση κατά συνθήκη στην πηγή προέλευσης. Όλα τα παραπάνω συνοψίζονται στο παρακάτω σχήμα:

Για κάθε ροή πακέτων (εισερχόμενη ή εξερχόμενη) χρησιμοποιείται και ένα αντίστοιχο στοιχείο που θα τη χειρίζεται. Έτσι από την πηγή προς τον ενδιάμεσο κόμβο κατασκευάζονται 3 στοιχεία (ένα για αποστολή πακέτων rtp και 2 για αποστολή/λήψη πακέτων rtcp). Το είδος των στοιχείων δεν αλλάζει βάσει του τύπου της ροής δεδομένων, αλλά αυτό που τα διαφοροποιεί είναι η κατεύθυνση της ροής. Έτσι κατασκευάζονται δύο στοιχεία udpsink για την αποστολή rtp/rtcp πακέτων και ένα στοιχείο udpsink για την λήψη των πακέτων:

```
rtpsink  = factory_make (rtpsink, "udpsink",
                "rtpsink");
rtcpsink = factory_make (rtcpsink, "udpsink",
                "rtcpsink");
rtcpsrc  = factory_make (rtcpsrc, "udpsrc", "rtcpsrc");
```

Για την αποστολή των rtp πακέτων χρησιμοποιείται ένα udp port με άρτιο αριθμό, ενώ για την αποστολή των rtcp πακέτων χρησιμοποιείται το αμέσως επόμενο περιττό udp

port, όπως αυτό συστήνεται από την IETF. Τα παραπάνω χαρακτηριστικά αποδίδονται στα αντικείμενα μέσω των ορισμάτων της συνάρτησης g_object_set:

```
g_object_set (rtpsink, "port", 5002, "host", DEST_HOST,
        NULL);
g_object_set (rtcpsink, "port", 5003, "host",DEST_HOST,
        NULL);
g_object_set (rtcpsrc, "port", 5007, NULL);
```

Η εξερχόμενη ροή των πακέτων της πηγής, γίνεται εισερχόμενη στον κόμβο της οικιακής πύλης. Για αυτό το λόγο λοιπόν θα χρειαστούν 2 στοιχεία udpsrc για τα πακέτα της εισερχόμενης rtp/rtcp ροής και ένα στοιχείο udpsink για την αποστολή rtcp πακέτων πίσω στην πηγή:

```
rtpsrc = factory_make (rtpsrc, "udpsrc", "rtpsrc");
rtcpsrc = factory_make (rtcpsrc, "udpsrc", "rtcpsrc");
rtcpsink = factory_make (rtcpsink, "udpsink",
                "rtcpsink");

g_object_set (G_OBJECT (rtpsrc), "port", 5002, "caps", caps, NULL);
g_object_set (rtcpsrc, "port", 5003, NULL);
g_object_set (rtcpsink, "port", 5007, "host",
        DEST_HOST, NULL);
```

Στο στοιχείο udpsrc δίνεται επίσης και το πεδίο caps, από το ακρωνύμιο *capabilities*, στο οποίο περιγράφονται στοιχεία της ροής τα οποία είναι πολύ κρίσιμα για την διαχείριση της.

```
/* Set up the caps */
caps = gst_caps_new_simple ("application/x-rtp", "media", G_TYPE_STRING, "video",
                "clock-rate", G_TYPE_INT, 90000,
                "encoding-name", G_TYPE_STRING, "H264", NULL);
```

Οι ροές των πακέτων *rtp, rtcp* λαμβάνονται από τα *rtpsink, rtcpsink* στοιχεία αντίστοιχα τα οποία με τη σειρά τους τροφοδοτούν το στοιχείο *rtpbin*. Το *rtpbin* είναι το στοιχείο που λειτουργεί ως διαχειριστής ροών, επιτρέποντας εκτός των άλλων τον συγχρονισμό πολλαπλών RTP συνεδριών μέσω των RTCP SR πακέτων.

Το στοιχείο rtpbin χρησιμοποιεί έναν αριθμό απο pads, που προσδιορίζουν την λειτουργικότητα που ενεργοποιείται. Έτσι λοιπόν εάν ο προγραμματιστής θελήσει το rtpbin να λειτουργήσει ως δέκτης rtp πακέτων, *αιτείται* ένα *recv_rtp_sink_%u pad,* όπου στο %u δηλώνει τον αριθμό της συνεδρίας. Κατ'όπιν, κάθε πακέτο που λαμβάνεται, προωθείται και υπόκειται σε ένα πλήθος επεξεργασιών που περιλαμβάνουν τον έλεγχο εγκυρότητας, αποπολύπλεξης βάση του SSRC και αποπολύπλεξης βάση του τύπου του περιεχομένου. Αφού ολοκληρωθεί η διαδικασία, το rtpbin θα δημιουργήσει ένα recv_rtp_src_%u_%u_%u pad με τον αριθμό της συνεδρίας, το SSRC και τον τύπο του περιεχομένου αντίστοιχα στο όνομα του.

Για να λειτουργήσει το rtpbin και ως δέκτης rtcp πακέτων, ο προγραμματιστής αιτείται ένα *recv_rtcp_sink_%u* pad όπου στο %u δηλώνεται ο αριθμός της συνεδρίας ενώ για την παραγωγή και αποστολή rtcp πακέτων θα πρέπει να αιτηθεί ένα *send_rtcp_src_%u* με τον αριθμό της συνεδρίας στη θέση του %u.

```
/*Starting with session 0: recv_rtp_sink_0,
recv_rtcp_sink_0, recv_rtcp_sink_0,.*/
/* Start by getting an RTP recv-sink pad for session 0 */
  rtpbin_sink_adding(rtpsrc, "src", rtpbin,
                     "recv_rtp_sink_0");
/* Getting RTCP recv-sink pad for session 0*/
  rtpbin_sink_adding(rtcpsrc, "src", rtpbin,
                     "recv_rtcp_sink_0");
/* Getting an RTCP srcpad for sending RTCP back to the sender.*/
  rtpbin_src_adding(rtpbin, "send_rtcp_src_0",
                rtcpsink, "sink");
```

Η συνάρτηση *rtpbin_sink_adding* δέχεται ως ορίσματα το αντικείμενο που θα συνδεθεί με το rtpbin, τον τύπο του pad (src) που θα ζητηθεί από το rtpbin καθώς και το ίδιο το rtpbin. Τέλος, το τελευταίο όρισμα που δέχεται η συνάρτηση είναι το όνομα του pad μαζί με τον αριθμό της συνεδρίας.

```
static void
rtpbin_sink_adding(GstElement * src, gchar * src_val, GstElement * sink, gchar * sink_val)
{
  GstPad *srcpad, *sinkpad;
  srcpad = gst_element_get_static_pad (src, src_val);
  sinkpad = gst_element_get_request_pad (sink, sink_val);
  lres = gst_pad_link (srcpad, sinkpad);
  g_assert (lres == GST_PAD_LINK_OK);
  gst_object_unref (srcpad);
  gst_object_unref (sinkpad);
}
```

Στο σώμα της συνάρτησης αρχικά γίνεται η αίτηση των 2 pads από τα δύο src & sink elements αντίστοιχα. Στη συνέχεια καλείται η συνάρτηση *gst_pad_link* που επιστρέφει το αποτέλεσμα της ένωσης των δύο στοιχείων σε επίπεδο pad το οποίο και ελέγχεται ως προς την εγκυρότητα του.

Το στοιχείο rtpbin μπορεί να λειτουργήσει και ως αποστολέας πακέτων, αρκεί ο προγραμματιστής να αιτηθεί ένα *send_rtp_sink_%u pad,* το οποίο θα δημιουργήσει αυτόματα ένα *send_rtp_src_%u pad.* Εάν ο αριθμός της συνεδρίας δεν δίνεται στη θέση του %u, τότε θα επιστραφεί ένα pad που να αντιστοιχεί στη συνεδρία με το χαμηλότερο διαθέσιμο αριθμό.

Τα παραπάνω περιγράφονται στο σώμα της συνάρτησης *rtpbin_src_adding:*

```
static void
rtpbin_src_adding(GstElement * src, gchar * src_val, GstElement * sink, gchar * sink_val)
{
  GstPad *srcpad, *sinkpad;
  srcpad = gst_element_get_request_pad (src, src_val);
  sinkpad = gst_element_get_static_pad (sink, sink_val);
  lres = gst_pad_link (srcpad, sinkpad);
  g_assert (lres == GST_PAD_LINK_OK);
  gst_object_unref (sinkpad);
```

```
  gst_object_unref (srcpad);
}
```

Η συνάρτηση rtpbin_src_adding δέχεται ως ορίσματα το element rtpbin καθώς και το όνομα που θα καθορίσει το είδος του pad και τον τύπο των πακέτων που θα χειριστεί μαζί με τον αριθμό της συνεδρίας στη θέση του %u. Τα υπόλοιπα ορίσματα είναι το στοιχείο που θα συνδεθεί με το rtpbin και το όνομα του pad.

Στο σώμα της συνάρτησης θα γίνει η αίτηση των δύο pad, ένα από το μέρος του rtpbin με χρήση της συνάρτησης *gst_element_get_ request_pad* η οποία θα επιστρέψει το pad του στοιχείου και ένα από το στοιχείο που θα συνδεθεί κάθε φορά με χρήση της συνάρτησης gst_element_get_static_pad. Στη συνέχεια ακολουθεί η απόπειρα συννένωσης των 2 elements μέσω των αντίστοιχων pads με τη συνάρτηση gst_pad_link.

Η ροή που δέχεται ο κόμβος της εικονικοποιημένης οικιακής πύλης προωθείται προς τον οικιακό εξοπλισμό του πελάτη σαν μια διαφορετική συνεδρία από αυτή της λήψης. Για να γίνει αυτό καλούνται τα απαραίτητα pads στο rtpbin όπως φαίνεται παρακάτω:

```
  /* Session 1 handling */
  /* Getting RTP send-sink pad for session 1 */
  rtpbin_sink_adding(rtph264pay, "src", rtpbin,
        "send_rtp_sink_1");

   /* Getting RTCP recv-sink pad for session 1*/
   rtpbin_sink_adding(rtcpsrc2, "src", rtpbin,
                "recv_rtcp_sink_1");
   /* Getting RTP send-src pad for session 1*/
   rtpbin_src2_adding(rtpbin, "send_rtp_src_1", rtpsink,
                "sink");

   /* Getting RTCP send-src pad for session 1*/
   rtpbin_src_adding(rtpbin, "send_rtcp_src_1", rtcpsink2,
                "sink");
```

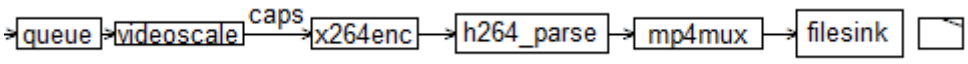
### 6.2.3 Η λειτουργία της εγγραφής (PVR)

Η λειτουργία της εγγραφής έχει κατασκευαστεί με στόχο την ελλάττωση του εξοπλισμού στις εγκαταστάσεις του πελάτη. Ο χρήστης έχει τη δυνατότητα της εκκίνησης της εγγραφής στο σημείο που αυτός επιθυμεί και την τερματίζει επιλέγοντας το αντίστοιχο κουμπί από τη γραφική διεπαφή του χρήστη. Το παραγώμενο αρχείο αποθηκεύεται στον κόμβο που υλοποιείται η εγγραφή, δηλαδή στον ενδιαμέσο. Σε αυτό τον κόμβο η ροή του βίντεο διακλαδώνεται, με έναν από τους δύο κλάδους να καταλήγει σε ένα αρχείο.

Με την ελλάττωση του εξοπλισμού στον πελάτη, οφελείται ο ίδιος ο χρήστης της υπηρεσίας γιατί οι επιλογές του αποθηκεύονται στο σύννεφο, με αποτέλεσμα την διαθεσιμότητα από οποιοδήποτε σημείο. Έτσι μπορεί να μετακινείται από έναν χώρο σε έναν άλλο χωρίς να χρειάζεται να φέρει μαζί του κάποιο αποθηκευτικό μέσο γιατί οι εγγραφές του είναι αποθηκευμένες στον δικό του προσωπικό λογαριασμό και διαθέσιμες όποτε το επιθυμεί.

Εκτός από τον χρήστη όμως, οφελείται και ο πάροχος της υπηρεσίας, καθώς μειώνεται ο εξοπλισμός που παραδίδει στον πελάτη (1 για κάθε STB) αλλά και το λειτουργικό κόστος διότι σε περίπτωση βλάβης δεν θα χρειαστεί επέμβαση στις εγκαταστάσεις του (έξοδα μετακίνησης & επισκευής) αφού όλη η δομή εντοπίζεται στο νέφος.

Τέλος το παραγόμενο από τη λειτουργία της εγγραφής αρχείο έχει ποιότητα ίδια με αυτή της πηγής, επιτρέποντας όμως τη τροποποίηση της κατά τη διάρκεια της προβολής (βλ. κεφ. Ποιότητα της της υπηρεσίας). Η τροποποίηση όμως μπορεί να συμβεί και κατά τη διάρκεια της εγγραφής επάνω στο παραγώμενο αρχείο. Προβλέπεται έτσι και η περίπτωση του μειωμένου αποθηκευτικού χώρου στον λογαριασμό του χρήστη, δίνοντας του έτσι τη δυνατότητα της χρήσης της υπηρεσίας έστω και με μειωμένα χαρακτηρηστικά.




**Εικόνα 20: Τα στοιχεία του σωλήνα για την εγγραφή της ροής.**

Η διαδικασία της έναρξης εγγραφής χωρίζεται σε δύο μέρη. Το πρώτο μέρος ξεκινάει από τον κόμβο του χρήστη όπου και εκεί ουσιαστικά αυτός δηλώνει τη βούληση του για εγγραφή του περιεχομένου τη στιγμή που επιθυμεί με πάτημα του πλήκτρου START_RECORDING.

```
static void
start_recording_cb (GtkWidget * widget, GstElement * pipeline)
{
	g_print ("ATTEMPTING TO START RECORDING\n");
	source_state_change ("2");
}
```

Το πλήκτρο αυτό μεταφέρει την εντολή του χρήστη προς τον κόμβο της οικιακής πύλης για έναρξη της εγγραφής του περιεχομένου:

```
static void
source_state_change (gchar * message)
{
	GError * error = NULL;
	/* create a new connection */
	GSocketConnection * connection = NULL;
	GSocketClient * client = g_socket_client_new();
	/* connect to the host */
	connection = g_socket_client_connect_to_host (client,
	(gchar*)DEST_HOST, 1600, NULL, &error);
	/* error check */
	if (error != NULL)
	{
		g_error (error->message);
	}
	else
	{
		g_print ("Connection successful!\n");
	}
	/* use the connection */
	GInputStream * istream = g_io_stream_get_input_stream
	(G_IO_STREAM (connection));
	GOutputStream * ostream = g_io_stream_get_output_stream
		(G_IO_STREAM (connection));
```

```
        g_output_stream_write  (ostream, message, 13, NULL,
        &error);

        /* don't forget to check for errors */
        if (error != NULL)
        {
                g_error (error->message);
        }
}
```

Η συνάρτηση source_state_change ανοίγει ένα δίαυλο επικοινωνίας για αποστολή σημάτων ελέγχου με διεύθυνση αντίθετη της πολυμεσικής ροής, δηλαδή από τον πελάτη προς τον πάροχο. Με αυτή γίνεται αποστολή όχι μόνο του σήματος για την έναρξη της εγγραφής, αλλά και όλων των λειτουργιών που μπορεί να διαχειριστεί η γραφική διεπαφή.

Το δεύτερο μέρος της εγγραφής περιλαμβάνει την παραλαβή του μηνύματος από τον οικιακό πελάτη, την αποκωδικοποίηση του και έναρξη της εγγραφής.

```
        /* listen to the 'incoming' signal */
        g_signal_connect (service,  "incoming", G_CALLBACK
        (incoming_callback),tee_save_pad);

        /* start the socket service */
        g_socket_service_start (service);
```

Μόλις οριστεί ο listener για τα εισερχόμενα μηνύματα, ορίζεται και η callback συνάρτηση incoming_callback για τον κατάλληλο χειρισμό του γεγονότος.

```
/* This function will get called everytime a client attempts to connect */
gboolean
incoming_callback  (GSocketService *service,
        GSocketConnection *connection, GObject *source_object, GstPad *
                                tee_save_pad)
{
  gint res;
  g_print("Received Connection from client!\n");
  GInputStream * istream = g_io_stream_get_input_stream
                (G_IO_STREAM (connection));
  gchar message[1024];
  g_input_stream_read  (istream,
                   message, 1024, NULL, NULL);

  g_print("Message was: \"%s\"\n", message);
  res = atoi(message);
  … … …

  else if (res == 2)
  {
    g_print("START RECORDING:\n");
    gst_pad_add_probe (tee_save_pad,
                GST_PAD_PROBE_TYPE_BLOCK_DOWNSTREAM,
```

```
                link_recording, tee_save_pad, NULL);

  }
```

Μόλις ληφθεί το πακέτο ελέγχου για την έναρξη εγγραφής της ροής, καλείται η συνάρτηση gst_pad_add_probe η οποία θα επιχειρήσει να συνδέσει στο pipeline και τον κλάδο της εγγραφής. Επίσης δηλώνεται και το αρχείο αποθήκευσης καθώς και η ονομασία αυτού, του οποίου η μορφή ένας αύξοντας αριθμός με κατάληξη ‘.mp4’.

```
static GstPadProbeReturn
link_recording (GstPad * pad, GstPadProbeInfo * info,
        GstPad * tee_save_pad)
{
  GstElement *next;
  gst_pad_remove_probe (pad, GST_PAD_PROBE_INFO_ID
                        (info));
  gchar str;
  g_print ("Attempting to link\n");
  g_object_set (filesink, "location", g_strconcat( g_strdup_printf("%i", i),".mp4",
        NULL), NULL);
  i++;
  gst_bin_add (GST_BIN (pipeline), bin);
  queue_save_pad = gst_element_get_static_pad
                (save_queue, "sink");
  gst_element_add_pad (bin, gst_ghost_pad_new ("sink",
        queue_save_pad));
  bin_pad = gst_element_get_static_pad (bin, "sink");

  if (gst_pad_link (tee_save_pad, bin_pad) !=
        GST_PAD_LINK_OK)
  {
    g_printerr ("Save-Tee could not be linked.\n");
    gst_object_unref (pipeline);
    return -1;
  }
  gst_element_set_state (bin, GST_STATE_PLAYING);
  return GST_PAD_PROBE_DROP;
}
```

Στην παραπάνω διαδικασία, τα στοιχεία του βρόχου εγγραφής ομαδοποιούνται σε ένα κοινό στοιχείο. Αυτό απλοποιεί την διαδικασία διότι όλες οι ενέργειες θα εφαρμοστούν σε αυτό, θα εφαρμοστούν και σε όλα τα υπόλοιπα.

Ο βρόχος της εγγραφής παραμένει για όσο διάστημα επιθυμεί ο χρήστης. Μόλις όμως αποφασίσει ότι θέλει να την τερματίσει επιλέγει το πλήκτρο STOP_RECORDING από τη γραφική διεπαφή του και η μεταφορά της εντολής είναι ίδια με αυτή της έναρξης εγγραφής.

```
static void
stop_recording_cb (GtkWidget * widget, GstElement *
        pipeline)
{
```

```
        g_print ("ATTEMPTING TO STOP RECORDING\n");
        source_state_change ("-2");
}
```

Στον κόμβο της οικιακής γίνεται λήψη του μηνύματος και αναγνωρίζεται η λειτουργία που αιτείται:

```
else if (res == -2)
{
        g_print("STOP RECORDING:\n");
        gst_pad_add_probe (tee_save_pad,
                GST_PAD_PROBE_TYPE_BLOCK_DOWNSTREAM,
                        pad_probe_cb, tee_save_pad, NULL);
}

static GstPadProbeReturn
pad_probe_cb (GstPad * pad, GstPadProbeInfo * info,
        GstPad * tee_save_pada)
{
          GstPad *srcpad, *sinkpad;
          g_print ("pad is blocked now");
          GST_DEBUG_OBJECT (pad, "pad is blocked now");
          /* remove the probe first */
          gst_pad_remove_probe (pad, GST_PAD_PROBE_INFO_ID
                                (info));
          gst_object_ref (bin);
          gst_bin_remove (GST_BIN (pipeline), bin);
          gst_pad_send_event (bin_pad, gst_event_new_eos ());
          usleep(500000);
          gst_element_set_state (bin, GST_STATE_NULL);
          g_print ("Done recording\n");
  return GST_PAD_PROBE_OK;
}
```

### 6.3 Ποιότητα της υπηρεσίας (Quality of Service/ QoS)

Η υπάρχουσα δικτυακή υποδομή μαζί με την ικανότητα προβολής του πολυμεσικού περιεχομένου στην προκαθορισμένη ανάλυση του αλλά και μια σειρά από άλλα θέματα κρίνουν την ποιότητα της παρεχόμενης υπηρεσίας.

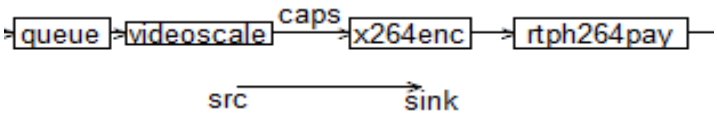
Η κάθε περίπτωση αντιμετωπίζεται με τον ίδιο τρόπο, δηλαδή με την τροποποίηση του αρχικού περιεχομένου σε ποιότητα ανώτερη ή κατώτερη με βάση τις εκάστοτε συνθήκες. Τη λειτουργία της τροποποίησης έχει επωμιστεί η οικιακή πύλη, καθώς είναι ο κόμβος του υπολογιστικού νέφους που συνδέεται με τις εγκαταστάσεις του χρήστη.

#### 6.3.1 Μετατροπή της κωδικοποίησης (transcoding)

Ένας παράγοντας που μπορεί να επιρεάσει την ποιότητα της υπηρεσίας είναι η οθόνη που χρησιμοποιεί ο χρήστης για την προβολή του περιεχομένου. Στο εμπόριο κυκλοφορεί μια τεράστια ποικιλία οθονών, οι οποίες δεν μπορούν όλες να ανταποκριθούν σε περιεχόμενο υψηλής ανάλυσης, με αποτέλεσμα τη σπατάλη δικτυακών πόρων. Για αυτό το λόγο δίνεται στον χρήστη η επιλογή τροποποίησης της ποιότητας κατ'απαίτησής του μέσω του γραφικού περιβάλλοντος του CPE.

**Εικόνα 21: Το ίδιο καρέ με τροποποιημένη την ποιότητα του**

Άλλος παράγοντας που θα μπορούσε να επιρεάσει την ποιότητα της υπηρεσίας είναι η αποθήκευση περιεχομένου στο υπολογιστικό νέφος σε διαφορετική ανάλυση από το αρχικό. Οι λόγοι για αυτή την ενέργεια ποικίλλουν και μπορεί να αφορούν τον περιορισμένο χώρο που προσφέρεται στο υπολογιστικό νέφος ή το είδος του περιεχομένου που πρόκειται να αποθηκευθεί.




**Εικόνα 22: Ο κλάδος που τροποποιεί το περιεχόμενο**

Η διαδικασία ξεκινάει από τη γραφική διεπαφή του χρήστη, πιέζοντας τα πλήκτρα UP ή DOWN ανάλογα με το αν επιθυμεί να βελτιώσει την ποιότητα του βίντεο ή να την ελλαττώσει.

```
/*Callback function for upgrading quality*/
static void
up_cb(GtkWidget * widget, GstElement * pipeline)
{
  g_print("Changing resolution higher\n");
  source_state_change ("3");
}

/*Callback function for downgrading quality*/
static void
down_cb(GtkWidget * widget, GstElement * pipeline)
{
  g_print("Changing resolution lower \n");
  source_state_change ("-3");
}
```

Στη συνέχεια, μόλις το μήνυμα ληφθεί στον ενδιάμεσο κόμβο, μεταφράζεται και χειρίζεται ανάλογα το περιεχόμενο του. Στην περίπτωση που αυτό αφορά τροποποίηση της ποιότητας, γίνεται ένας έλεγχος για το εάν πρόκειται για υποβάθμιση ή αναβάθμιση και στη συνέχεια δίνεται το αντίστοιχο capsfilter (Εικ.21) το οποίο περιέχει και τις απαραίτητες πληροφορίες.

```
 else if(res == 3)
 {
```

```
 if(resol == 7){
  g_object_set (G_OBJECT(capsfilter), NULL);
  g_print("Changing resolution to %d:\n", resol);
 }
 else if(resol == 6)
 {
  g_object_set (G_OBJECT(capsfilter), "caps", caps6,
                                  NULL);
   resol ++;
   g_print("Changing resolution to %d:\n", resol);
  }
……
else if(res == -3)
{
 if(resol == 7)
 {
   g_object_set (G_OBJECT(capsfilter), "caps", caps6,
                                  NULL);
  resol --;
  g_print("Changing resolution to %d:\n", resol);
 }
 else if(resol == 6)
 {
  g_object_set (G_OBJECT(capsfilter), "caps", caps5,
                                      NULL);
  resol --;
  g_print("Changing resolution to %d:\n", resol);
 }
……
```

Για τις ανάγκες της υλοποίησης, έχουν δηλωθεί 6 επιπλέον επιλογές για τις διαστάσεις του μήκους και πλάτους του περιεχομένου. Έτσι ο χρήστης μπορεί να επιλέξει την ποιότητα του βίντεο που παρακολουθεί.

```
/* setting up the caps on videoscale elements, both rtp & save branches*/
caps6 = gst_caps_new_simple ("video/x-raw","width",
                        G_TYPE_INT, 640, "height",
                        G_TYPE_INT, 360, NULL);
caps5 = gst_caps_new_simple ("video/x-raw","width",
                        G_TYPE_INT, 360, "height",
                        G_TYPE_INT, 280, NULL);
caps4 = gst_caps_new_simple ("video/x-raw", "width",
                        G_TYPE_INT, 280, "height",
                        G_TYPE_INT, 240, NULL);
caps3 = gst_caps_new_simple ("video/x-raw","width",
                        G_TYPE_INT, 240, "height",
                        G_TYPE_INT, 180, NULL);
caps2 = gst_caps_new_simple ("video/x-raw","width",
                        G_TYPE_INT, 180, "height",
                        G_TYPE_INT, 120, NULL);
caps1 = gst_caps_new_simple ("video/x-raw","width",
                        G_TYPE_INT, 120, "height",
                        G_TYPE_INT, 80, NULL);
```

### 6.3.2 Στατιστικά & φόρτος δικτύου

Ο έλεγχος που στηρίζεται στη δικτυακή υποδομή μπορεί να πραγματοποιηθεί στα πακέτα RTCP τα οποία φέρουν και στατιστικά στοιχεία. Τα στατιστικά στοιχεία είναι πολύτιμα καθώς επαληθεύουν το θεωρητικό μοντέλο με αξιόπιστα στοιχεία οδηγώντας σε βάσιμα συμπεράσματα.

Στην τοπολογία του δικτύου, ο κόμβος της εικονικοποιημένης πύλης λαμβάνει στατιστικά στοιχεία και από τους 2 κόμβους, δηλαδή τον κόμβο προέλευσης & προορισμού, κάνοντας τον τον πιο ενδιαφέρον ως προς την ανάλυσή του.

```
/* print stats every second */
g_timeout_add_seconds (1, (GSourceFunc) print_stats,
                rtpbin);
```

Αρχικά δηλώνεται η συνάρτηση η οποία θα εκτελείται κάθε 1 sec μαζί με την callback συνάρτηση καθώς και τα απαραίτητα στοιχεία.

```
static gboolean
print_stats (GstElement * rtpbin)
{
  GObject *session;
  GValueArray *arr;
  GValue *val;
  guint i;
  g_print ("***********************************\n");

  /* get session 0 */
  g_signal_emit_by_name (rtpbin, "get-internal-session",
                         0, &session);


  /* print all the sources in the session, this includes the internal source */
  g_object_get (session, "sources", &arr, NULL);
  for (i = 0; i < arr->n_values; i++) {
    GObject *source;
    val = g_value_array_get_nth (arr, i);
    source = g_value_get_object (val);
    print_source_stats (source);
  }
  g_value_array_free (arr);
  g_object_unref (session);

  /* get session 1 */
  g_signal_emit_by_name (rtpbin, "get-internal-session",
                         1, &session);
  /* print all the sources in the session, this includes
                  the internal source */

  g_object_get (session, "sources", &arr, NULL);
  for (i = 0; i < arr->n_values; i++) {
    GObject *source;
    val = g_value_array_get_nth (arr, i);
    source = g_value_get_object (val);
    print_source_stats (source);
```

```
  }
 g_value_array_free (arr);
 g_object_unref (session);
 return TRUE;
}
```

Στη συνέχεια και για κάθε δευτερόλεπτο, καλείται η συνάρτηση print_stats, η οποία λαμβάνει από τον διαχειριστή των συνεδριών (session manager) του rtpbin τα στατιστικά στοιχεία των δύο συνεδριών. Επειδή όμως η μορφή στην οποία βρίσκονται τα στοιχεία δεν είναι αναγνώσιμη από τον άνθρωπο, καλείται η συνάρτηση print_source_stats που έχει σκοπό την μετατρόπη των στατιστικών σε μια μορφή αναγνωρίσιμη από τον άνθρωπο πρωτού τα εκτυπώσει στην οθόνη.

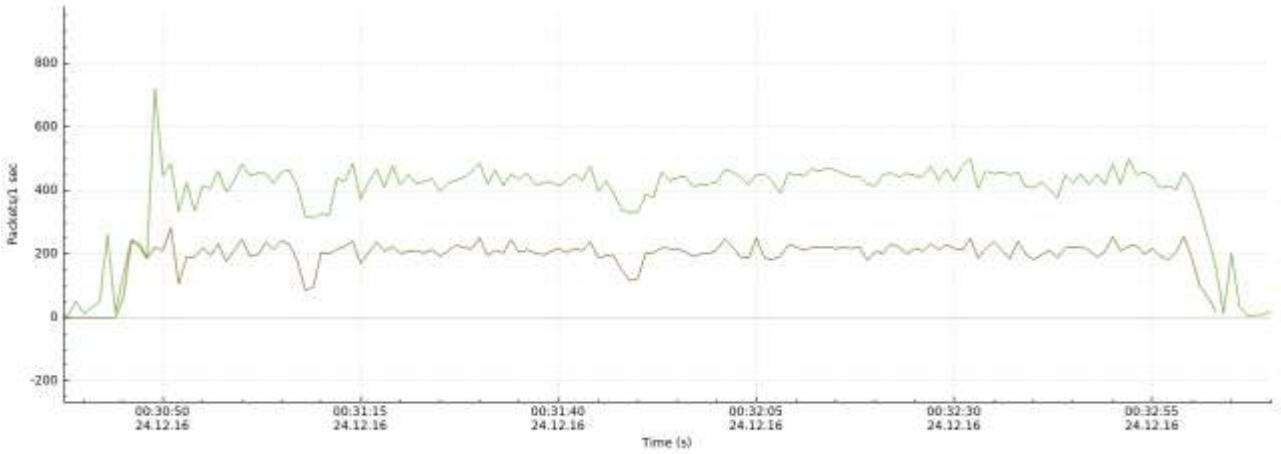
```
/* print the stats of a source */
static void
print_source_stats (GObject * source)
{
 GstStructure *stats;
 gchar *str;


 /* get the source stats */
 g_object_get (source, "stats", &stats, NULL);
 /* simply dump the stats structure */
 str = gst_structure_to_string (stats);
 g_print ("source stats: %s\n", str);
 gst_structure_free (stats);
 g_free (str);
}
```


**Εικόνα 23: Τα στατιστικά στοιχεία που εκδίδει ο ενδιάμεσος κόμβος.**

Η τροποποίηση του περιεχομένου -όπως περιγράφηκε στην παράγραφο 6.3.1- επιφέρει και αλλαγές στον ρυθμό αποστολής των δεδομένων. Ο έλεγχος του ρυθμού αποστολής δεδομένων είναι πολύ σημαντικό χαρακτηριστικό σε δικτυακές εφαρμογές και ειδικότερα σε streaming & live streaming επειδή το περιεχόμενο πρέπει να είναι διαθέσιμο άμεσα και χωρίς καθυστέρηση.

Έτσι στις περιπτώσεις που η αποστολή δεδομένων είναι μειωμένη ή οι συγκρούσεις των πακέτων είναι αρκετά υψηλές, η οικιακή πύλη θα μπορεί να δράσει με ανάλογη τροποποίηση της ποιότητας του περιεχομένου ομαλοποιόντας την εμπειρία θέασης. Στα παρακάτω στιγμιότυπα πραγματοποιούνται οι μετρήσεις αποστολής των δεδομένων και πως μεταβάλλεται ο ρυθμός αυτός με διάφορες αλλαγές στην ποιότητα του περιεχομένου. Οι μετρήσεις λαμβάνονται μεταξύ των δικτύων προέλευσης – vSTB/vRGW (δίκτυο A), vSTB/vRGW – προορισμού (δίκτυο B) και αθροιστικά στα δύο δίκτυα (AB) ενώ χρησιμοποιείται και το ίδιο πολυμεσικό περιεχόμενο.

Στο πρώτο στιγιότυπο φαίνεται η συνολική κίνηση και η κίνηση στο δικτυο A.




**Εικόνα 24: Συνολική κίνηση (AB) & κίνηση βίντεο στο δίκτυο A.**

Στο επόμενο στιγμιότυπο φαίνεται η κίνηση στα δίκτυα A, B & η συνολική (AB).

Η συνολική κίνηση μπορεί να υπολογιστεί από τον τύπο:

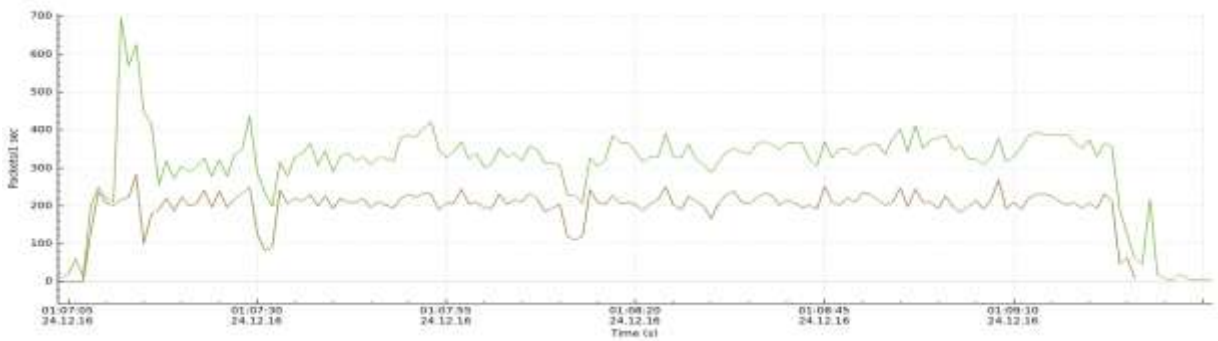
$$sum = udpA + udpB$$

Επειδή όμως οι δύο ροές είναι ίδιες καθώς αφορούν το ίδιο πολυμεσικό περιεχόμενο(δηλ. *udpA = udpB)*, η συνολική κίνηση μετατρέπεται:

$$sum = 2 \cdot udptrafficA$$

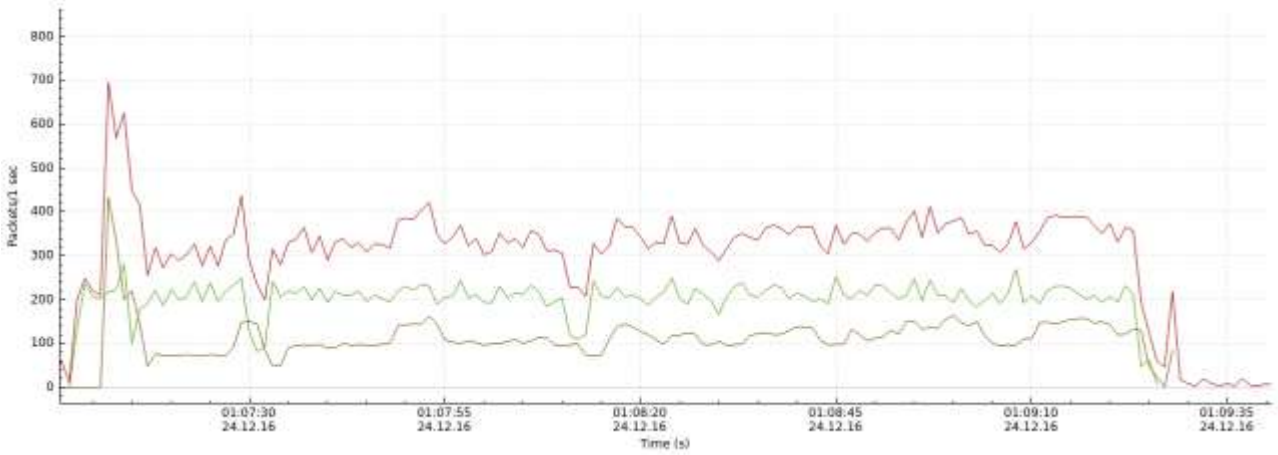
Στο παραπάνω άθροισμα πρέπει να αθροιστούν και οι ροές των rtcp πακέτων, οι οποίες είναι όμως είναι αμελητέες μπροστά στο σύνολο του συνολικού αθροίσματος.

Στη συνέχεια λαμβάνεται ένα στιγμιότυπο από την επανάληψη της αποστολής του περιεχομένου, αυτή τη φορά με μειωμένο το ρυθμό αποστολής λόγω της ελλάτωσης της ποιότητας του περιεχομένου.




**Εικόνα 25: Συνολικές κινήσεις πακέτων βίντεο με μειωμένη ανάλυση στον πελάτη (δίκτυο B)**

Στο παρακάτω στιγμιότυπο μπορεί να διακρίνει κανείς τις κινήσεις στα δίκτυα Α, Β και τη δυνολική (ΑΒ).




**Εικόνα 26: Η κίνηση των πακέτων στα 2 δίκτυα & η συνολική με μειωμένη ανάλυση στο δίκτυο του πελάτη**

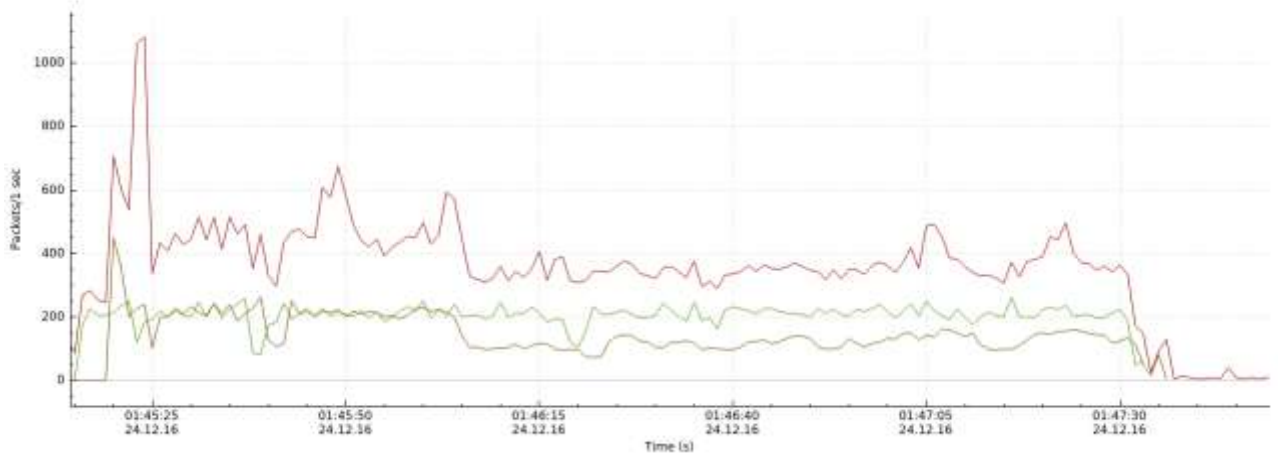
Επειδή οι δύο ροές είναι ίδιες, μπορεί να πεί κανείς ότι η συνολική κίνηση υπολογίζεται από τον τύπο:

$$sum = udptrafficA + udptrafficB$$

αλλά επειδή $udpB = x \cdot udptrafficB$, όπου x είναι ο συντελεστής που εκφράζει την ελάττωση στην κίνηση udp του δικτύου Β εξ'αιτίας της μειωμένης ποιότητας, η συνολική κίνηση και με υπολογισμό των rtcp πακέτων δίνεται από τον τύπο:

$$sum = (1+x) \cdot udptrafficA + rtcpA + rtcpB$$

Στο τελευταίο στιγμιότυπο εμφανίζεται η κίνηση στα 2 δίκτυα αλλά με υποβάθμιση του περιεχομένου στο δίκτυο Β περίπου στο πρώτο λεπτό.




**Εικόνα 27: Η κίνηση των πακέτων στα 2 δίκτυα & η συνολική με μειωμένη ανάλυση στο δίκτυο του πελάτη κατά τη διάρκεια της αναπαραγωγής**

Τέλος, στη συνολική κίνηση θα πρέπει να υπολογιστεί και ο φόρτος από τα μηνύματα ελέγχου του χρήστη μέσω της γραφικής διεπαφής. Έτσι η συνολική κίνηση στα δύο δίκτυα δίνεται από τον τύπο:

$$sum = (1+x) \cdot udptrafficA + rtcpA + rtcpB + udpcontrol$$

## 6.4 Καταμερισμός του φόρτου εργασίας

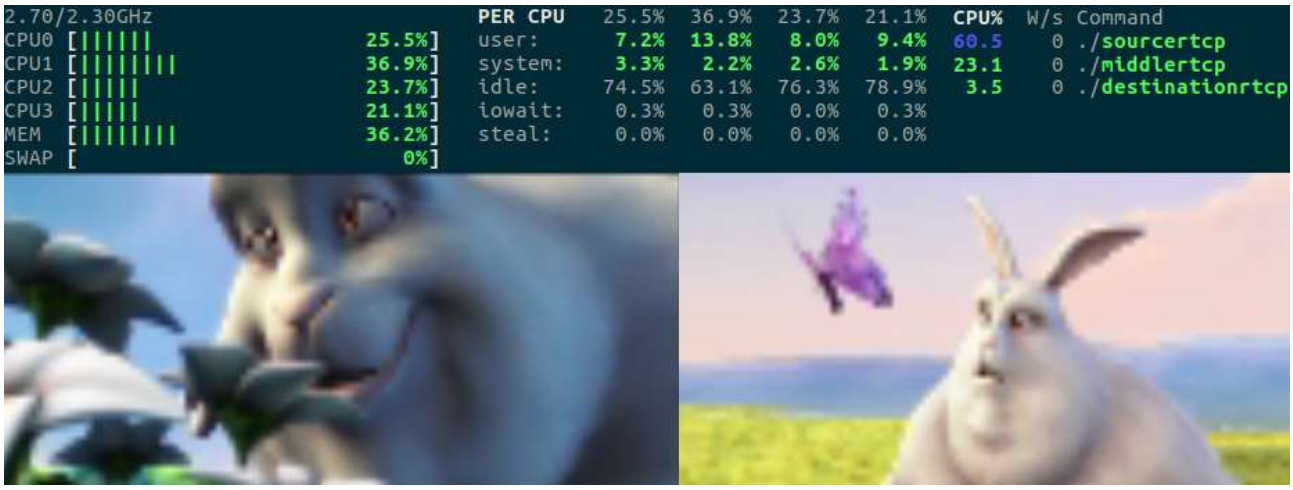
Ο φόρτος εργασίας κάθε κόμβου είναι ανάλογος του πλήθους και του είδους των εργασιών που εκτελεί. Έτσι οδηγούμαστε στο συμπέρασμα, πως ο ενδιαμέσος κόμβος με τις ενέργειες της λήψης, μεταβίβασης, εγγραφής και επεξεργασίας της πολυμεσικής ροής θα έχει και τον περισσότερο φόρτο σε σχέση με τους υπόλοιπους δύο κόμβους. Αυτό απεικονίζεται και στα παρακάτω στιγμιότυπα με χρήση ενός προγράματος επισκόπησης διεργασιών σε ένα φυσικό σύστημα ώστε να μπορούν να συγκριθούν εύκολα οι 3 διεργασίες μεταξύ τους.

Τα στιγμιότυπα που ακολουθούν έγιναν με γνώμονα την μέτρηση του φόρτου σε κάθε στάδιο του κύκλου εργασιών. Έτσι έχουν ληφθεί τέσσερα στιγμιότυπα:

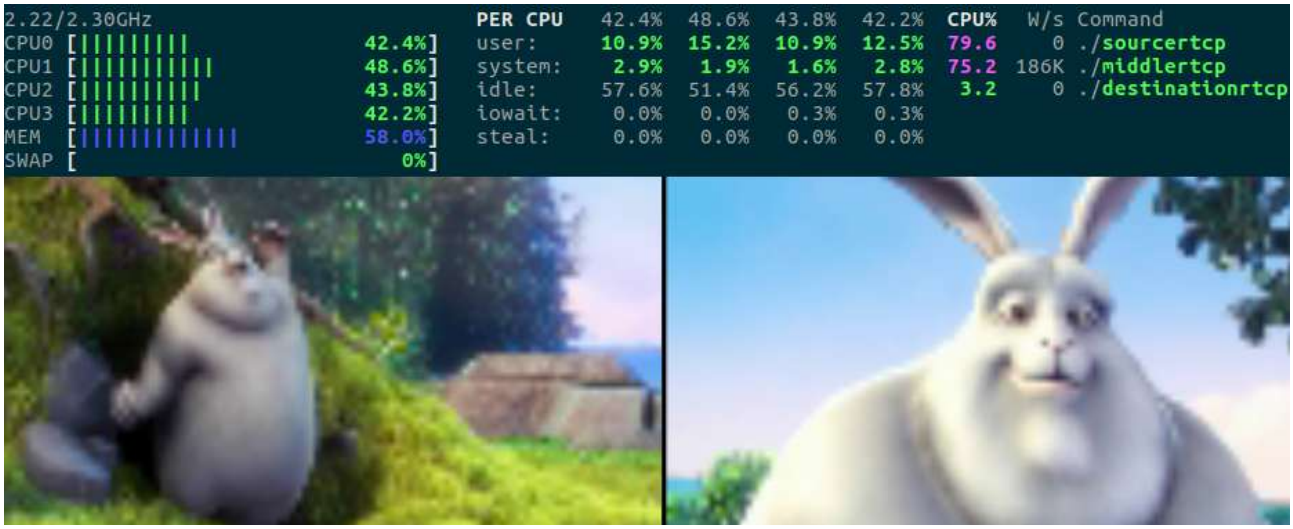
- Χωρίς transcoding και χωρίς εγγραφή του περιεχομένου
- Χωρίς transcoding και με εγγραφή του περιεχομένου
- Με transcoding και χωρίς εγγραφή του περιεχομένου
- Με transcoding και με εγγραφή του περιεχομένου


**Εικόνα 28: Οι τρεις διεργασίες χωρίς τροποποίηση του περιεχομένου και χωρίς εγγραφή**


**Εικόνα 29: Οι τρεις διεργασίες με τροποποίηση του περιεχομένου και με εγγραφή**




**Εικόνα 30: Οι τρεις διεργασίες με τροποποίηση του περιεχομένου και χωρίς εγγραφή**




**Εικόνα 31: Οι τρεις διεργασίες με τροποποίηση του περιεχομένου και χωρίς εγγραφή**

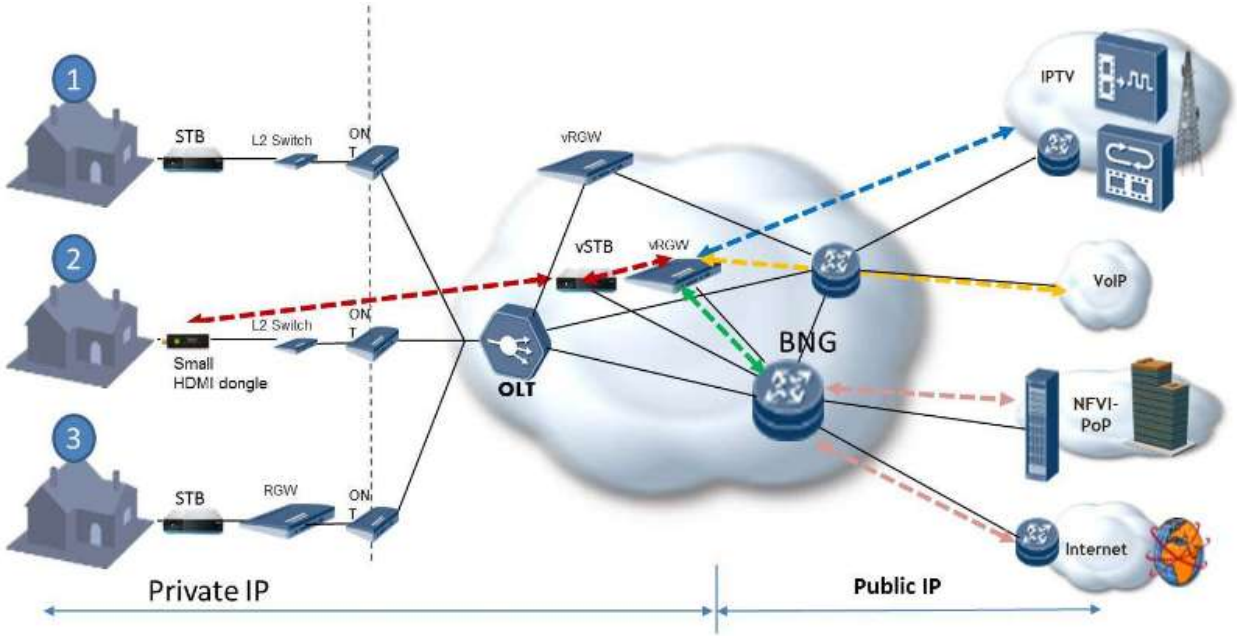
Από τα στιγμιότυπα, εκτός από το προφανές συμπέρασμα πως ο φόρτος του κόμβου προέλευσης παραμένει σταθερός ανεξάρτητα από την ενέργεια που έχει ζητηθεί από την οικιακή πύλη, ο φόρτος του κόμβου της οικιακής πύλης είναι ο μεγαλύτερος από τους τρεις. Τέλος ο φόρτος του κόμβου του προορισμού είναι δραστικά υποπολλαπλάσιος του ενδιάμεσου κόμβου. Αυτό επιτρέπει στον πάροχο της υπηρεσίας την κατασκευή ενός CPE με χαρακτηριστικά συγκριτικά ελαφρύτερα από αυτά του αρχικού. Τα οφέλη από αυτό εντοπίζονται στον πάροχο της υπηρεσίας λόγω της μείωσης των εξόδων CAPEX & OPEX καθώς και της μείωσης κατασκευής του οικιακού εξοπλισμού και συντήρησής του, αλλά επίσης βελτιώνεται η ανταγωνιστικότητα του λόγω της μειωμένης τιμής που θα προκύψει. Επίσης ο χρήστης μπορεί να αγοράσει φθηνότερα την υπηρεσία αλλά και η κοινωνία έχει όφελος από την κατασκευή ενός 'πράσινου' CPE με λιγότερα εξαρτήματα.

# 7. ΣΥΜΠΕΡΑΣΜΑΤΑ

Η εικονικοποίηση της οικιακής πύλης ακολουθεί το μοντέλο της συγκέντρωσης όλων δικτυακών λειτουργιών στο υπολογιστικό νέφος. Η τάση αυτή, όπως περιγράφετα ιαπότοInternet of Things απλοποιεί τις διαδικασίες στο οικιακό περιβάλλον ενώπροσφέρειμειωμένο όγκο συσκευών σε αυτό.

Τα πλεονεκτήματα αυτής της ενέργειας στον πάροχο της υπηρεσίας είναι η κατασκευή ελαφρύτερων συσκευών και άρα με μικρότερη πιθανότητα βλάβης ενώ η μεταφορά των υπηρεσιών στο υπολογιστικό νέφος του δίνει τη δυνατότητα να δοκιμάσει νέες τεχνολογίες με μεγαλύτερη ασφάλεια. Ο πελάτης εκτός από το προφανές πλεονέκτημα του μικρότερου όγκου, επωφελείται της φορητότητας των υπηρεσιών, καθώς μπορεί να έχει πρόσβαση τόσο τρέχων πρόγραμμα όσο και στο αποθηκευμένο μέσω του PVR. Επίσης η λειτουργία του transcoding στο πολυμεσικό περιεχόμενο από τον κόμβο του vSTB/vRGW του δίνει την δυνατότητα να επιλέξει μια ποιότητα ελαττωμένη από την αρχική και να διαθέσει το υπόλοιπο εύρος ζώνης του δικτύου του στις υπόλοιπες δικτυακές του υπηρεσίες.

Στα μειονεκτήματα, ο όγκος των συνδρομητών αναμένεται να είναι πολύ υψηλός με αποτέλεσμα να μην μπορέσει ο πάροχος να εξυπηρετήσει όλοκληρο τον απαιτούμενο όγκο σε δικτύωση και υπολογιστικούς πόρους. Για αυτό το λόγο πολλές εταιρείες καταφεύγουν στην λύση με υβριδική υλοποίηση. Σε αυτή, ένα εκ των STB, RGW εικονικοποιούνται ενώ το άλλο δίδεται στον πελάτη ως φυσική συσκευή που θα τον εξυπηρετεί.




**Εικόνα 32 : Υβριδικές μορφές της υπηρεσίας**

Με όποιον τρόπο επιλεγεί να γίνει η δρομολόγηση -είτε με πλήρης εικονικοποίηση είτε με την υβριδική μέθοδο- το σίγουρο είναι πως η ανάπτυξη των δικτύων υψηλών ταχυτήτων και η προσφορά τους σε χαμηλότερες τιμές θα μπορέσει να βελτιώσει την εικονικοποίηση των υπηρεσιών και να βελτιώσει την εμπειρία των πολυμεσικών εφαρμογών στο οικιακό περιβάλλον.

# ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ

| Ξενόγλωσσος όρος | Ελληνικός όρος |
|---|---|
| European Telecommunications Standards Institute | Ευρωπαϊκός Οργανισμός Τηλεπικοινωνιακών Προτύπων |
| Virtualization | Εικονικοποίηση |
| Network Function Virtualization | Εικονικοποίηση Δικτυακών Λειτουργιών |
| Capital Expenditure | Κεφαλαιακές Δαπάνες |
| Operational Expenditure | Λειτουργικές Δαπάνες |
| Customer Premises Equipment | Εξοπλισμός στο χώρο του πελάτη |
| Residential Gateway | Οικιακή Πύλη |
| Set-top Box | Αποκωδικοποιητής |
| Personal Video Recorder | Προσωπική Εγγραφή Βίντεο |
| Quality of Experience | Ποιότητα της Εμπειριάς |
| Real Time Protocol | Πρωτόκολλο Πραγματικού Χρόνο |
| Real Time Control Protocol | Πρωτόκολλο Ελέγχου RTP |
| Timestamp | Χρονοσφραγίδα |
| Sequence Number | Αριθμός Ακολουθίας |
| Payload Type | Τύπος δεδομένων |
| Network Time Protocol | Πρωτόκολλο Δικτυακού Χρόνου |
| BYE | Πακέτο τερματισμού συνεδρίας |
| Data Center | Κέντρο δεδομένων |
| Dashboard | Διεπαφή χρήστη |
| Virtual Machine | Εικονική μηχανή |
| Multimedia Pipeline | Πολυμεσικός Σωλήνας |
| Transcoding | Διακωδικοποίηση |
| Session Manager | Διαχειριστής συνεδριών |
| Streaming | Ροή |

# ΣΥΝΤΜΗΣΕΙΣ – ΑΡΚΤΙΚΟΛΕΞΑ – ΑΚΡΩΝΥΜΙΑ

| | |
|---|---|
| IoT | Internet of Things |
| NFV | Network Function Virtualization |
| CPE | Customer Premises Equipment |
| VoIP | Voice over IP |
| RGW | Residential Gateway |
| STB | Set-top Box |
| PVR | Personal Video Recorder |
| QoS | Quality of Service |
| QoE | Quality of Experience |
| RTP | Real-time Transport Protocol |
| RTCP | Real-time Transport Control Protocol |
| RTSP | Real Time Streaming Protocol |
| VoD | Video on Demand |
| UDP | User Datagram Protocl |
| SSRC | Synchronization Source |
| PT | Payload Type |
| NTP | Network Time Protocol |
| CNAME | Canonical Name |
| SR | Sender Report |
| RR | Receiver Report |
| PaaS | Platform as a Service |
| IaaS | Infrastructure as a Service |
| SaaS | Software as a Service |
| Cli | Command Line Interface |
| API | Application Programming Interface |
| DHCP | Dynamic Host Configuration Protocol |
| EOS | End of Sequence |

## ΑΝΑΦΟΡΕΣ

**[1]** European Telecommunications Standards Institute
url: http://www.etsi.org/
**[2]** Network Functions Virtualisation (NFV) Use Cases url: http://www.etsi.org/deliver/etsi_gs/nfv/001_099/001/01.01.01_60/gs_nfv001v010101p.pdf
**[3]** RTP: A Transport Protocol for Real-Time Applications
url: https://www.ietf.org/rfc/rfc3550.txt
**[4]** Real Time Streaming Protocol (RTSP)
url: https://www.ietf.org/rfc/rfc2326.txt
**[5]** RTP Profile for Audio and Video Conferences with Minimal Control
url: https://tools.ietf.org/html/rfc3551#section-6