Remote Learning Laboratory using Surveillance cameras and VPN Solutions

Florin Drăgan, Romul Copindean

Faculty of Electric Engineering, Technical University of Cluj-Napoca, Romania

Abstract: *This paper describes a solution for the remote teaching of courses and laboratory work in the COVID-19 pandemic. This project implemented several communication solutions so that students have access, from home, to applications running on computers in the laboratory, to the workstations there and can see what happens during the experiments. For this, communication solutions based on VPN (Virtual Private Network) tunneling, gateway based port forwarding or proxy servers were used. In order for the students to see what was happening in the laboratory, during the experiments, a group of surveillance cameras mounted next to each work stand was used. Communication with them was designed to allow more participants to attend these laboratory experiments, regardless of where they connect to the network. To create the system, a server computer was used that runs the LINUX operating system, the paper presenting the practical solutions needed to configure the applications that provide all these services. This system is used, starting with the autumn of 2020 within the department of which the authors are part, to teach an appreciable number of disciplines that require the realization of practical works on computers and laboratory stands.*

Keywords: *Remote Learning and Remote Laboratory, IP surveillance cameras, video access over RTSP, Linux, TCP or UDP port forwarding and proxy server, VPN connections, Remote Desktop.*

1. INTRODUCTION

Due to pandemic, many schools closed teaching activities that cause less access to practical lessons. In our technical university there are many practical experiments that impose students to work directly on dedicated laboratory platforms. Because of this conditions, access solutions must be provided to them, such as Remote Desktop applications, Virtual Private Network (VPN), Virtual Serial Ports connections. If experimental platforms display some results or movements that must be seen, several other audio and video resources must be provided [1].

Most of the devices are equipped with display elements such as LEDs, or digital displays and others, such as actuators, have moving parts whose positions or movements must also be seen.

Web or surveillance cameras are very useful on presented conditions, but Real Time Streaming Protocol applications allow to students multiple and long distance access to audiate conferences and see direct shown experiments. In this project, each work stand includes a camera with which the students can follow in real time what happens during the work and can track, or record, the evolution of the process or experiment.

To access all cameras with RTSP integrated servers, a port forwarding, or proxy server applications must be also included in main communication gateway [2].

The main system components, providing VPN connection, remote desktop application, WiFi cameras with RTSP server, and many others are shown in figure 1.

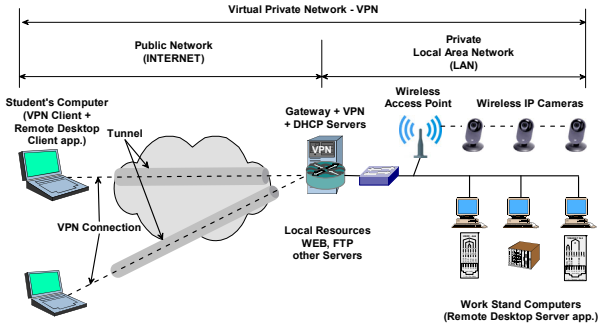

Fig. 1. The Remote Laboratory system's main components

One of the very important central components is the PC based machine that acts as a network gateway, a VPN server for external access, with functions of authentication, data encryption and connections logging. Many other services also run on this computer, such as Web, FTP, Database, DHCP and SSH, last one used mainly for remote system management [3].

Due to the fact that the operating system used is a Linux distribution, installed without support for X windows, this computer does not require demanding resources, the only important requirement being a high reliability and stability in operation.

2. HARDWARE COMPONENTS

2.1. General description

The laboratory room was dedicated for several domains like SCADA systems, microcontroller and PLC learning, data communication protocols and remote energy meter solutions.

All work stands consist in a personal computer, part of a private network connected to Internet through an Ethernet switch and TCP/IP based gateway. If web camera are used, these are each directly connected to the computers, but wireless surveillance cameras must use a WiFi access point. So, another device must be used. In this example, a WiFi extender connected to the same Ethernet infrastructure is used. The gateway runs a DHCP server providing all network addresses, for most of all terminals and TCP/IP devices. Routing, filtering and TCP or UDP port forwarding functions are also provided.

2.2. Central Processing Unit

A dedicated server computer, HP Proliant DL360 G5 Intel(R) Xeon(R) 5160 @ 3.00GHz 4GB RAM with 2x60GB hard drives in RAID 1, was used for this project, but any other desktop computer with similar performance can be used [4].

The gateway runs a DHCP server providing all network addresses, for most of all terminals and TCP/IP devices. Routing, filtering and TCP or UDP port forwarding functions are also provided. The gateway runs a DHCP server providing all network addresses, for most of all terminals and TCP/IP devices. Routing, filtering and TCP or UDP port forwarding functions are also provided. Two Web and FTP servers are used to allow students to download most important documentation materials [3].

The configuration example files, for most important services, are described in the next chapter.

2.3. Video Cameras

Due to mobility reasons, a couple of WiFi IP cameras are used to acquire and transmit, in live video during working, the laboratory lessons. To be more flexible in operation, cameras must be integrated in same private network as the computers used in working stands.

For this purpose, Xiaomi Mijia 1080p smart security cameras were chosen and purchased, that do not require a wall mount and they can be easily placed next to the work stands.

Unfortunately, the acquired models don't run the RTSP server, by default, so the firmware was slightly modified to allow running it [5].

Another nice and functional solution can be based on the use of smartphones, equipped with WiFi and video cameras, which although functional are no longer used, as shown in figure 2. To do this, you need to install applications that can provide the RTSP service, many of which can be found and downloaded for free. A few have been tested, which have worked satisfactorily, developed for Android operating systems.

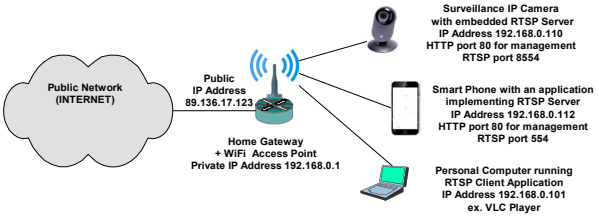

Fig. 2. Using smart phone with RTSP server application as an IP camera

To be used at the desired performance, these applications must be configured in accordance with the rest of the information and communication processing system. Most of these also allow remote configuration using a web browser, but this requires additional configurations of the gateway device, but which are not the subject of this paper. Because most of the time the cameras do not require frequently changed settings, they are made locally directly from the application on the smartphone, no additional functions are required in the gateway for this.

3. SOFTWARE COMPONENTS

3.1. DHCP server configuration file

For better management, computers and other network devices are statically configured, unless when some of them use dynamically configuration procedures. Usually wireless (WiFi) devices search for an access point and ask for various configuration parameters.

In this project, computers, virtual serial port devices, some PLCs and measuring devices are statically configured with IP/gateway/domain-name-server addresses. For surveillance cameras a DHCP with fixed addresses allocation (IP address reservation for each camera, based on MAC addresses), method is used.

The important parts of `dhcpd.conf` file are listed below [6]:

```
# dhcpd.conf
# Sample configuration file for ISC dhcpd
#
# If this DHCP server is the official DHCP server
for the local
# network, the authoritative directive should
be uncommented.
authoritative;
ddns-update-style ad-hoc;

# Configuration for an internal subnet.
# This section is for my local dhcp addresses
# Random addresses allocation
subnet 192.168.120.64 netmask 255.255.255.192 {
#option domain-name "example.com";
option broadcast-address 192.168.120.127;
option    domain-name-servers    193.226.5.151,
193.226.5.33;
option subnet-mask 255.255.255.192;
option routers 192.168.120.65;
range 192.168.120.66 192.168.120.79;
default-lease-time 600;
max-lease-time 7200;
```

```
}

# This section is for fixed defined
# addresses of XIAOMI cameras:
# Camera00 - Camera06...

host Camera00 {
hardware ethernet 64:90:c1:45:e6:5b;
fixed-address 192.168.120.80;
}

host Camera01 {
hardware ethernet 64:90:c1:45:d5:b6;
fixed-address 192.168.120.81;
}
....

host Camera06 {
hardware ethernet 64:90:c1:45:d3:a9;
fixed-address 192.168.120.86;
}
```

[6]

3.2. VPN communication link overview

A virtual private network (VPN) is the extension of a private network that encompasses links across shared or public networks like the Internet. A VPN enables sending data between two computers belonging to different local networks but separated geographically, across a shared or public internetwork in a manner that emulates the properties of a point-to-point private link. The action of configuring, creating and suing a virtual private communication link is known as virtual private networking [7].

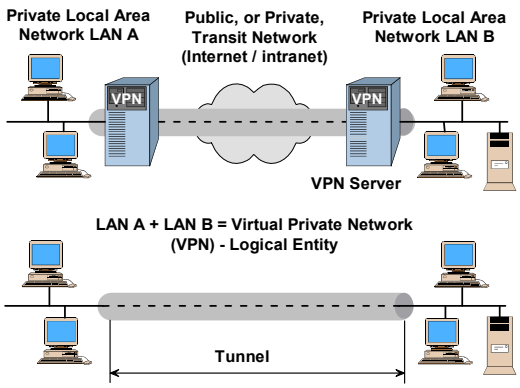

Fig. 3. Concept of Virtual Private Network [8]

To emulate a private communication link, the data being sent is encrypted for confidentiality. Packets that are intercepted on the shared or public network are indecipherable without knowing the encryption keys. The link in which the private data is encapsulated and encrypted is known as a virtual private network (VPN) connection [7].

VPN connections allow users working at home or on the road to obtain a remote access connection to an organization server using the infrastructure provided by a public internetwork such as the Internet [8].

From the user's perspective, the VPN is a "point-to-point" connection between the computer, the VPN client, and an organization server, the VPN server. The exact infrastructure of the shared or public network is irrelevant because it appears logically as if the data is sent over a dedicated private link.

VPN connections also allow organizations to have routed connections with geographically separate offices or with other organizations over a public internetwork such as the Internet while maintaining secure communications. A routed VPN connection across the Internet logically operates as a dedicated WAN link.

With both the remote access connection and with the routed connection, VPN connections allow an organization to trade in long distance dial-up or leased lines for local dial-up or leased lines to an Internet service provider (ISP) [7].

VPN connections use various tunneling protocols, most popular are L2TP/ IPSec (Layer Two Tunneling Protocol)/ (IP Security), PPTP (Point to Point Tunneling Protocol), OpenVPN, SSTP (Secure Socket Tunneling Protocol).

3.3. PPTP-VPN server configuration

Point-to-Point Tunneling Protocol is one of the oldest VPN protocols, developed in the mid-90s by Microsoft. PPTP was integrated first into Windows 95 being specifically designed for dial-up connections. PPTP's basic encryption was quickly cracked, compromising its underlying security. but it lacks many of the security features found in other modern protocols. At this moment it can deliver the best connection speeds for users who may not need heavy encryption. Because PPTP is still used in certain applications, most operating systems continue to use it.

To allow most of computer clients to access VPN connections, the following files must be configured [9]:

A. Primary `pptpd` configuration file `/etc/pptpd.conf`

```
# Sample PoPToP configuration file
#
#speed 115200
#
option /etc/ppp/options.pptpd
#
debug
# bcrelay: Specifies the gateway interface
bcrelay eth0
#
localip 192.168.122.1
remoteip 192.168.122.2-10
#
pidfile /var/run/pptpd.pid
```

B. `pppd` options configuration file [10][9] `/etc/ppp/options.pptpd`

```
# This file should work for Win9x/98/ME and
NT/2000/XP/7/10 clients
#
lock
mtu 1400
mru 1400
login
auth
nodetach
proxyarp
asyncmap 0
```

```
refuse-pap
require-mschap-v2
require-mppe-stateless
lcp-echo-failure 30
lcp-echo-interval 5
ms-dns 193.226.5.35
ms-dns 193.226.5.151
```
[9]

C. Secrets for authentication using CHAP [9]
/etc/ppp/chap-secrets

```
#
# CHAP authentication file:
# This file should have a permissions of 600
to avoid any users to read it. To change, type
# chmod 600 /etc/ppp/chap-secrets [ENTER]
#
# If using login sequence to authenticate
# system registered users, use the following
line:
# Username Server Passw Client_IP_addr
#    *       *       ""      *
# If using additional non registered users,
# create dedicated line for each one:

# Secrets for authentication using CHAP
# User   Server   Secret   Client_IP_addr
Florind     *    paRoLa123        *
Tomi        *     Xcz175aR        *
......
# IP addresses filtering is possible
# "*" sign means "any" value
```
[10][9]

D. Firewall configuration file
/etc/rc.d/rc.firewall

To allow access to the VPN server, from another location, in this firewall configuration file some lines must be added [11][12].

```
#
# For each remote user IP address
# VPN connection from Florin's computer
# (read it as a single line!)
iptables -A INPUT  -p tcp -s 89.137.113.31/32
--dport 1723 -j ACCEPT
#
# For a local subnetwork access:
iptables -A INPUT -p tcp -s 192.168.120.0/26 -
-dport 1723 -j ACCEPT
# For any outside client input connection
iptables -A INPUT  -p tcp --dport 1723 -j
ACCEPT
```

3.4. Starting the VPN server

To run PoPToP simply type: `pptpd`, or `/usr/local/sbin/pptpd` if you don't have `/usr/local/sbin` in your search path.

If starting up the server automatically is needed, a `/etc/rc.d/rc.pptpd` file is required, with `chmod 755` value.

This file is provided within PoPToP package and it don't need to be modified. To start, stop or restart the server, the following commands can be used [9]:

```
root@masserv:~#/etc/rc.d/rc.pptpd start
root@masserv:~#/etc/rc.d/rc.pptpd stop
root@masserv:~#/etc/rc.d/rc.pptpd restart
```

4. VIDEO STREAMING SOLUTIONS

4.1. Video streaming overview

There are few solutions to allow students to view what happened in laboratory room. Web cameras directly connected to each computer on work stand or independent surveillance cameras located nearby them. Instead of surveillance cameras, old smartphones which are no longer used to communicate with them, but due to its inner camera and with WiFi connectivity, may also be used. In this project, the second solution was implemented, by using cameras with RTSP server implementation [2].

The RTSP *(Real Time Streaming Protocol)* is an application-level protocol for control over the delivery of data with real-time properties setting. RTSP provides an extensible framework to enable controlled, on-demand delivery of real-time data, such as audio and video. Sources of data can include both live data feeds and stored clips. This protocol is intended to control multiple data delivery sessions, provide a means for choosing delivery channels such as UDP, multicast UDP and TCP, and provide a means for choosing delivery mechanisms based upon RTP [RFC 7826].

RTP (Real-Time transport Protocol) provides end-to-end network transport functions suitable for applications transmitting real-time data, such as audio, video or simulation data, over multicast or unicast network services [RFC 3550].

In order to increase the efficiency of information transfer through streaming channels, more and more complex and efficient solutions and protocols are continuously developed. The Peer-to-Peer Streaming Peer Protocol (PPSPP) is a protocol for disseminating the same content to a group of interested parts in a streaming fashion. PPSPP supports streaming of both prerecorded (on-demand) and live audio/video content [RFC 7574].

To access surveillance cameras inside private network, simplest solution is to run RTSP client application directly on same computer like the other applications. If the video client must be run on each student's computer, to access cameras beyond gateway device, one of three solutions listed below, is needed to be used:

- using a RTSP communication service running in public domain (cloud);
- gateway configured to allow ports forwarding;
- running a RTSP proxy server application on gateway.

4.2. Peer to peer communication manner

Most cameras use to transfer videos and sounds a communication channel, based on "client/server" streaming pair connection, through various network paths [RFC5694]. This manner require a RTSP streaming server located somewhere in public network, commonly named today "Cloud" as shown in figure 4.

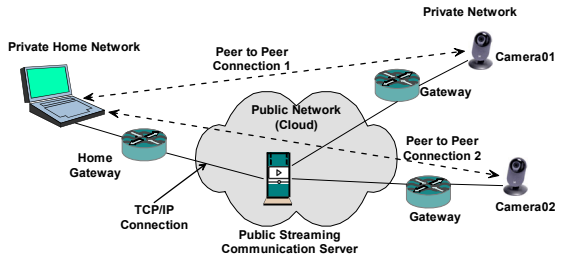

Fig. 4. Concept of Peer to Peer (P2P) connection

Camera (one of the "peer") acts as an image and sound source running a client application, that transmits data to the streaming server. At the other end (the second "peer"), the streaming player is another client application connected to the same server, as the camera and uses the same channel identifier.

By using the application which accompanies each acquired camera and following the instructions, the task must be very easy. Due to security reasons, an important drawback of using this solution in this case, is that some credentials of the camera channel account must be shared with students.

4.3. Gateway providing RTSP ports forwarding

When cameras can run a RTSP server, it is necessary to configure the gateway device to provide a TCP or UDP protocols port forwarding service, depending on each camera working way. In this project, a group of six surveillance IP cameras, in each laboratory, was used.

Cameras was configured to run RTSP based streaming server using TCP 8554 port. Because more than a single camera is used, one socket must be provided for each one. In this case, not only a port forwarding but a port redirection (PAT - *Port Address Translation*), to different values, is also needed [6][12].

The gateway device based on a Linux machine, uses the `iptables` application to define, access control, filtering and packets processing rules. Each rule consist of a set of three lines, starting with `iptables` keyword. First line prepare a "room" for each packet that may be received, for pre-routing process, and the second one instructs where to forward it. It can be seen that for each camera different destination port number `(8551-8556)` is used outside gateway (public network), but the same port number `(8554)` inside private network as shown in figure 5.

The third line complete the access control list to allow packets with appropriate destination port number to be accepted by the system, as part of firewall rules.

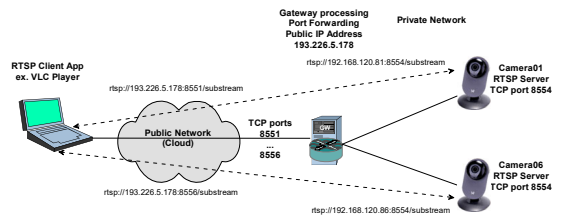

Fig. 5. Accessing multiple RTSP cameras with port forwarding and redirecting (PAT)

```
#
# Camera01

iptables -A PREROUTING -t nat -p tcp -d
193.226.5.178 --dport 8551 -j DNAT --to
192.168.120.81:8554

iptables -A FORWARD -p tcp -d 192.168.120.81 --
dport 8554 -o eth0 -j ACCEPT

iptables -A INPUT -p tcp -d 192.168.120.81/32 -
-dport 8551 -j ACCEPT
#
.........
#
# Camera06

#iptables -A PREROUTING -t nat -p tcp -d
193.226.5.178 --dport 8556 -j DNAT --to
 192.168.120.86:8554

#iptables -A FORWARD -p tcp -d 192.168.120.86 -
-dport 8554 -o eth0 -j ACCEPT

#iptables -A INPUT -p tcp -d 192.168.120.86/32
--dport 8556 -j ACCEPT
# [12]
```

Even if it is not a very important detail, it can be seen that the IP address of each camera has been correlated with the external TCP destination port number. For example, the camera with IP address `192,168,120.81` was assigned port `8551` and so on.

An important drawback of this solution is that it allows most often only one client connection at a time, depending on camera software resources. So, if more simultaneous connections are needed, a RTSP proxy server application may be a better solution to be used.

4.4. Running a RTSP proxy server on gateway

RTSP proxy is a program that allow public RTSP clients (network video player applications) to connect to multiple RTSP servers (IP cameras in this case) on a private network behind a gateway device having one public IP address. It waits for TCP connections, and forward them to the host specified in the RTSP request (figure 6) [2].

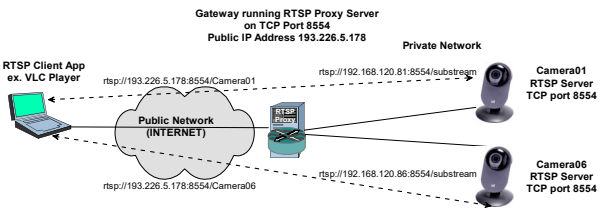

Fig. 6. Accessing multiple RTSP cameras using a proxy server

There are many proxy server application on marketplace, the choice of a variant depending on the complexity of the required tasks and the operating system used. In this project a simple but efficient Linux version, named `rtsp-simple-server`, was used.

The main features of this application are described in reference [13]. It is compatible with Linux, Windows and macOS and does not require any dependency because runs as a single executable file that can be

downloaded already compiled for various operating systems. The application consists in two files, an executable `rtsp-simple-server` and a configuration text file `rtsp-simple-server.yml`.

To configure the proxy server, some lines in rtsp-simple-server.yml need to be edited, like in example below:

```
# supported stream protocols (the handshake is
always performed with TCP).
protocols: [udp, tcp]
# port of the TCP RTSP listener.
rtspPort: 8554
# port of the UDP RTP listener (used only if
udp is in protocols).
#rtpPort: 8000
# port of the UDP RTCP listener (used only if
udp is in protocols).
#rtcpPort: 8001
...
# if the source is an RTSP url, this is the
protocol that will be used to
# pull the stream.
#   sourceProtocol: udp
   sourceProtocol: tcp
# these settings are path-dependent.
#
paths:
#  all:
# proxied:
# url of the source stream, in the format
# source: rtsp://original-url
 camera01:
  source: rtsp://192.168.120.81:8554/substream
...
 camera06:
  source: rtsp://192.168.120.86:8554/substream

# rtsp://user:pass@host:port/path
# To access cameras behind gateway, running
# the proxy server without any authentication
# method, the network resources paths will be:

# rtsp://193.226.5.178:8554/Camera01
...
# rtsp://193.226.5.178:8554/Camera06
# where the IP address above is the public
# address of the gateway.
# For more information please read detailed
# instructions.[13]
#[10]
```

The main advantages of using proxy server are clients simultaneous multiple accesses and less resources demand from cameras.

5. REMOTE LABOTATORY APPLICATIONS

5.1. Running Remote Desktop application through VPN.

Remote Desktop is one of the application allowing to access a computer from other location and run applications in same way as using it's local console, and consist of a client/server pair. Local computer runs client part, and the remote one (Lab computer), the server. The connection uses communication protection methods, by user accounts, passwords and data encryption.

Remote Desktop is one of the application allowing to access a computer from other location and run applications in same way as using it's local console, and consist of a client/server pair. Local computer (student) runs client part, and the remote one (Lab), the server, the two computers forming a single entity, as if they were "soldered". The figure 6 represents the model of how to multiple tunneling processes be used and how the remote desktop tunnel works inside the VPN tunnel [7].

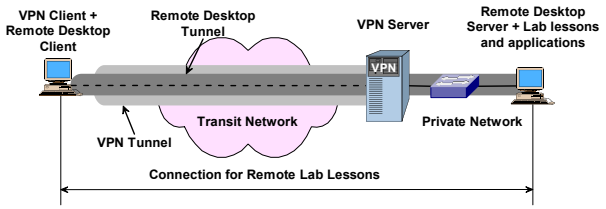

Fig. 7. Multiple tunneling processes, the Remote Desktop tunnel running inside the VPN tunnel

The connection uses communication protection methods, by user accounts, passwords and data encryption. Remote Desktop communication can also be done via a VPN connection, which allows students to use computers in the lab, as if operating directly on them, even if they are elsewhere.

6. CONCLUSIONS

This paper describes solutions developed for remote learning, based on practical lessons, running on computers inside a laboratory placed in other location, if there is a network connection between the student computer and those in laboratory.

Students can run practical applications on workstations and see what happens during the experiments. For this, each stand is equipped with an IP surveillance camera, which the students can access using a common media player application (ex. VLC), in real time.

The solution provides communication virtual connections, based on tunneling techniques as VPN and Remote Desktop, port forwarding and redirection, or proxy servers. One of the easiest implementation consists on a Linux based computer configured to run these tasks, very simple to be configured. The paper describes the principle of this system, main parts, and several very helpful configuration examples.

7. ACKNOWLEDGMENTS

This project was initiated and developed as a result of the pandemic situation COVID-19, when education adopted the way of working online. For its realization, the cheapest, easy and fast solutions to be implemented were analyzed and adopted. The knowledge and experience, gained in this field by all members of the department's staff over time, were used.

REFERENCES

1. Ravindran, R., Huang, C.C., Thulasiraman, K. and Lin, T.C. (2018) "Topology Abstraction Service for IP VPNs: Core Network Partitioning for Resource Sharing". American Journal of Operations Research, 8, 167-202.
2. Shibeshi, Z., S., Terzoli, A., Bradshaw, Ka., "Using an RTSP proxy to implement the IPTV Media Function via a streaming

server", 2010 International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), November 2010
3. https://www.researchgate.net/publication/224208370_Using_an_RTSP_proxy_to_implement_the_IPTV_Media_Function_via_a_streaming_server
4. Drăgan, F., Linux based Remote Access Communication, Data Base and Assessment server, using Apache web server, PHP and MySQL, The 2nd International Conference on Engineering Graphics and Design - ICEGD - 7–10 iunie, 2007, Galaţi, România.
5. https://support.hpe.com/hpesc/public/docDisplay?docId=c00715072&docLocale=en_US
6. https://github.com/cmiguelcabral/mjsxj05cm-hacks
7. Bautts, O., Dawson, T., Purdy, G., N., "Linux Network Administrator's Guide", 3rd Edition, O'Reilly Media, Inc. 2005.
8. * * * Microsoft Windows 2000 Resource Kit, Cap. 9, Virtual Private Networking,
9. Drăgan, F., "Protocoale de comunicaţie", UT Press, ISBN978-973-662-378-3, Cluj-Napoca, 2008.
10. https://wiki.archlinux.org/index.php/PPTP_server
11. Light-Williams, C., Drake, J., „Linux PPP HOWTO", Linux Distribution, Command Prompt Inc. 2000.
12. Ranch, D. A., „Linux IP Masquerade HOWTO", v2.00.110903, Linux Distribution, 2003
13. Russel, R., "Linux 2.4 Packet Filtering HOWTO", Revision 1.21, 2001/08/15,
14. http://www.redes-linux.com/manuales/seguridad/packet-filtering-HOWTO.a4.pdf
15. https://github.com/aler9/rtsp-simple-server
16. https://doi.org/10.4236/ajor.2018.83
17. RFC3550, "RTP: A Transport Protocol for Real-Time Applications", IETF, 2003/07.
18. RFC5694, "Peer-to-Peer (P2P) Architecture: Definition, Taxonomies, Examples, and Applicability", IETF, 2007/11.
19. RFC7574, "Peer-to-Peer Streaming Peer Protocol (PPSPP)", IETF, 2015/07.
20. RFC7826, "Real-Time Streaming Protocol Version 2.0", IETF, 2016/12.

Florin Drăgan
Faculty of Electrical Engineering, Technical University of Cluj-Napoca, 26-28, G. Bariţiu st., Cluj-Napoca, Romania
Florin.Dragan@ethm.utcluj.ro

Romul Copîndean,
Faculty of Electrical Engineering, Technical University of Cluj-Napoca, 26-28, G. Bariţiu st., Cluj-Napoca, Romania
Romul.Copindean@ethm.utcluj.ro