საქართველოს ტექნიკური უნივერსიტეტი

ირაკლი ჯორჯაძე

მონაცემთა გადაცემის ქსელებით ხმის სიგნალის გატარებასთან დაკავშირებული პრობლემების აღმოფხვრა და შესაბამისი დაცვის მექანიზმების შემუშავება

წარდგენილია დოქტორის აკადემიური ხარისხის მოსაპოვებლად

სადოქტორო პროგრამა: ”ტელეკომუნიკაცია“

შიფრი: 0402

საქართველოს ტექნიკური უნივერსიტეტი

თბილისი, 0175, საქართველო

ივლისი, 2019 წ.

თბილისი

2019 წელი

საქართველოს ტექნიკური უნივერსიტეტი

ენერგეტიკისა და ტელეკომუნიკაციის ფაკულტეტი

ჩვენ, ქვემორე ხელისმომწერნი ვადასტურებთ, რომ გავეცანით ირაკლი ჯორჯაძის მიერ შესრულებულ სადისერტაციო ნაშრომს დასახელებით: „მონაცემთა გადაცემის ქსელებით ხმის სიგნალის გატარებასთან დაკავშირებული პრობლემების აღმოფხვრა და შესაბამისი დაცვის მექანიზმების შემუშავება“ და ვაძლევთ რეკომენდაციას საქართველოს ტექნიკური უნივერსიტეტის ენერგეტიკისა და ტელეკომუნიკაციის ფაკულტეტის სადისერტაციო საბჭოში მის განხილვას დოქტორის აკადემიური ხარისხის მოსაპოვებლად.

_____, 2019 წელი

ხელმძღვანელი:_____ასოც. პროფ. ვ. აბულაძე

რეცენზენტი: _____

რეცენზენტი: _____

საქართველოს ტექნიკური უნივერსიტეტი

2019

ავტორი: ირაკლი ჯორჯაძე

თემის დასახელება: „მონაცემთა გადაცემის ქსელებით ხმის სიგნალის გატარებასთან დაკავშირებული პრობლემების აღმოფხვრა და შესაბამისი დაცვის მექანიზმების შემუშავება.“

ფაკულტეტი: ენერგეტიკისა და ტელეკომუნიკაციის

ხარისხი: აკადემიური დოქტორი

სხდომა ჩატარდა:_____, 2019 წ.

რეზიუმე

სამუშაოში წარმოდგენილია სადისერტაციო თემასთან “მონაცემთა გადაცემის ქსელებით ხმის სიგნალის გატარებასთან დაკავშირებული პრობლემების აღმოფხვრა და შესაბამისი დაცვის მექანიზმების შემუშავება“ დაკავშირებით ჩატარებული თეორიული და ექსპერიმენტული კვლევის შედეგები.

თანამედროვე სატელეკომუნიკაციო პროვაიდერების ქსელები კომპლექსურია და ისინი მომხმარებლებს მრავალფეროვან სერვისებს სთავაზობენ, რომელთა უზრუნველსაყოფად პროვაიდერის ქსელში გამოყენებულია სხვადასხვა ტექნოლოგიები და პროტოკოლები.

ინტერნეტის ქსელში მულტიმედიური აპლიკაციების მოთხოვნის შედეგად უკანასკნელ პერიოდში გაიზარდა მონაცემთა გადაცემის სიხშირე და მოთხოვნები სიჩქარისადმი. საუკეთესო ხარისხის სერვისებით მომხმარებლების უზრუნველყოფა დაკავშირებულია მრავალ პრობლემასთან, როგორიცაა, მაგალითად, პაკეტების დაყოვნება, მათი დაკარგვა, პაკეტებს შორის არსებული დროითი ინტერვალების ცვალებადობა და სხვ.

აღნიშნული პრობლემების გადაჭრისთვის შეიძლება გამოყენებული იყოს ტექნოლოგია, რომელსაც უწოდებენ მრავალპროტოკოლიან კომუტაციას ნიშნულების მიხედვით (MPLS) და რომელიც უზრუნველყოფს გადაცემული პაკეტების უსაფრთხოებას, მათი დაყოვნების დროის მინიმიზაციასა და გადაცემას მაღალი სიჩქარით.

ამრიგად, MPLS-ქსელები სტანდარტული მარშრუტიზაციის ნაცვლად იყენებენ კომუტაციის მეთოდს და ისინი კი არ ენაცვლებიან არსებულ IP-ქსელებს, არამედ მათთან კომბინაციაში უზრუნველყოფენ მომსახურების გაუმჯობესებულ ხარისხსა და შესაფერის გარემოს ნაკადის მართვისთვის. MPLS-ის მნიშვნელოვანი ფუნქციაა ტრაფიკის ინჟინერინგი (TE), რომელიც მნიშვნელოვან როლს ასრულებს ქსელური დატვირთვის მინიმიზაციისთვის, რესურსების რეზერვაციისთვის, ტრაფიკის ბალანსისა და მისი მართვისთვის. ქსელის მუშაობის ეფექტურობის შეფასება ხდება ქსელის გარკვეული პარამეტრების გათვალისწინებით, რომელთაგან ძირითადია დაყოვნება, პაკეტებს შორის დაყოვნების ცვალებადობა (Jitter) და პაკეტების კარგვის სიხშირე (Packet Loss). MPLS-TE-ქსელთან მომსახურების ხარისხის შესაბამისი ტექნოლოგიის (QoS) შერწყმით შესაძლებელია ქსელის ეფექტურობის გაუმჯობესება. ამიტომ MPLS-TE-ს საშუალებით მიიღწევა მონაცემების უსაფრთხო და ხარისხიანი მარშრუტიზაცია. ქსელში ტრაფიკის დიდი მოცულობის არსებობისას და ისეთი მოულოდნელი პრობლემების წარმოშობისას, როგორებიცაა ფიზიკური შეერთების გათიშვა, ქსელზე განხორციელებული ხაკერული თავდასხმები და მათი ისეთი ნაირსახეობა, როგორიცაა ეგრეთ წოდებული განაწილებული შეტევა მომსახურების დაბლოკვის მიზნით (DdoS), ამ დროს ხდება მონაცემთა გადაცემის არხებში ტრაფიკის კრიტიკულ დონემდე მატება და ხშირად არხის

გამტარუნარიანობის ლიმიტის შევსება. ამისთვის საჭიროა ქსელის მონაცემთა ნაკადის მართვა, რომლის განხორციელება მიზანშეწონილია, აგრეთვე, მარშრუტიზატორების, კომუტატორებისა და ქსელის სხვა აპარატურული პრობლემების დროს. მსგავს შემთხვევებში უნდა მოხდეს მარშრუტიზაციის კონფიგურაციის მყისიერი ცვლილება. ქსელის სპეციალისტის ჩართულობის შემთხვევაში ეს პროცესი მოითხოვს დიდ დროს გარდა ამისა, იგი იწვევს მექანიკური შეცდომების ალბათობის გაზრდას.

ამრიგად, მრავალპროტოკოლიანი კომუტაცია ნიშნულების მიხედვით (MPLS) მოქნილი ტექნოლოგიაა, რომელიც უზრუნველყოფს უსაფრთხოებას, ქსელში პაკეტების დაგვიანების მინიმიზაციასა და მონაცემთა გადაცემას მაღალი სიჩქარით. იგი, ძირითადად, გამოიყენება სერვის-პროვაიდერების მიერ და რეალურ დროში გადაცემული ხმოვანი და ვიდეო-სერვისების გადაცემის იდეალურ საშუალებას წარმოადგენს. ხმის გადაცემა ინტერნეტ-პროტოკოლის საშუალებით (VoIP) მრავალი პროტოკოლით უზრუნველყოფილი საბოლოო პროდუქტია და მის განვითარებასთან ერთად საჭიროა მისი კომპონენტების ხელმისაწვდომობის მთლიანობისა და კონფიდენციალობის საკითხების დამუშავება.

გამოცდილება გვიჩვენებს, რომ სერვის-პროვაიდერის ქსელში ტრაფიკის ოპტიმალური მარშრუტიზაცია მრავალ პრობლემასთანაა დაკავშირებული, რადგან პროვაიდერებს მაღალი გამტარუნარიანობის გადაცემის არხები აქვთ და ამ არხების ტევადობის გაზრდა ან მათი რაოდენობის დამატება დიდ ფინანსებთანაა დაკავშირებული. ამიტომ პროვაიდერები იძულებულნი არიან მაქსიმალურად გამოიყენონ თავიანთ ქსელში არსებული რესურსები, რაც ვერ ხერხდება სტანდარტული შიდა მარშრუტიზაციის პროტოკოლების გამოყენებით. სწორედ ამ დროს სპეციალისტები მიმართავენ ტრაფიკის ინჟინერიას (MPLS-TE), რის გარეშეც თითქმის გამორიცხულია სერვის-პროვაიდერის ინტერნეტ-ქსელის ფუნქციონირება.

თუმცა, როგორც აღნიშნული იყო, MPLS-TE-ს გამოყენების შემთხვევაში ქსელი არ არის დაზღვეული გარკვეული სახის პრობლემებისგან, რომელთა დასაძლევად დამუშავდა სისტემა NACS შესაბამისი ალგორითმებით, რომელიც ქსელში დაფიქსირებული პრობლემების დროს გადაწყვეტს ქსელური მოწყობილობების კონფიგურაციის ოპტიმიზაციის ამოცანას და დაახარისხებს შესრულებულ ქმედებებს. ამასთანავე, შესაძლებელი იქნება აღნიშნული პროცესების ვიზუალიზაცია და რეალურ დროში მათი დაკვირვება

ნაშრომში დამუშავებულია მონაცემთა ნაკადის გვირაბების (Tunnels), ვირტუალური მარშრუტიზაცია-გადამისამართების (VRF) პროტოკოლების ავტომატიზაციის პროგრამულ უზრუნველყოფასთან ინტეგრაციის მეთოდი, რამაც უზრუნველყო ხმისა და ინტერნეტ-ტრაფიკის ერთმანეთისა და გლობალური მარშრუტიზაციის ცხრილიდან იზოლაცია.

ნაშრომში დამუშავებულია, აგრეთვე, ეგრეთ წოდებული ქსელის ავტომატური კონფიგურაციის NACS-სისტემის ლოგიკური სქემა,

შემუშავებულია შესაბამისი პროგრამული უზრუნველყოფის კოდი, წარმოდგენილია მისი განხორციელების ტექნიკა და ნაჩვენებია მონიტორინგის სისტემასთან მისი ინტეგრაციის შესაძლებლობა.

ვირტუალურ პროგრამულ EVE-NG-გარემოში რეალური IP-MPLS-ის ანალოგიური ქსელის შემუშავების შედეგად შესაძლებელი გახდა შესაბამისი ალგორითმის მოდელირება, IP-MPLS-ქსელის სიმულაცია და წარმოქმნილი ქსელური პრობლემების მოგვარება.

Resume

The work includes the results of the theoretical and experimental research on the Dissertation topic "Elimination of problems related to the transmission of voice signals with data transmission networks and the elaboration of appropriate protection mechanisms".

The modern telecommunication providers networks are complex and offer users a variety of services that provide various technologies and protocols in the provider's network

The demand for multimedia applications in the internet network has increased the frequency and demands of data transfer in recent years. Providing users with the best quality services is associated with many problems such as delay in packets, loss of them, change of time intervals between the packets, etc. For solving these problems, technology can be used as a Multi-protocol Label Switching protocol (MPLS), which ensures safety of transmitted packets, minimizing their time of delay and transmission at high speed.

Therefore, MPLS-networks use the switching method instead of standard routing, and they do not replace the existing IP-networks, but in combination with them to provide improved quality of services and to manage the suitable environment. MPLS's important function is Traffic Engineering (TE), which plays an important role in minimizing network loads, resource surveillance, traffic balance and management. The network performance efficiency is evaluated by considering certain network parameters, most of which are the delay, the packet delay variation (jitter) and the packet loss (packet loss). With the MPLS-TE quality of service (QoS), it is possible to improve network efficiency. That is why MPLS-TE can provide data safe and quality routing. network traffic, a large volume of existence and the unexpected problems arising, such as the physical link damage , the network of hacker attacks, and in such variety, such as the distributed denial-of-service (DdoS), At the same time, data transfer channels increase the traffic to critical levels and often fill the bandwidth limit. For this it is necessary to manage the network data stream, which is also appropriate for routers, switches, and other hardware problems. In such cases, instant change of routing configuration should be performed. In case of network engagement, this process requires a lot of time and it leads to increasing the likelihood of mechanical errors.

Therefore, Multi-protocol Label Switching (MPLS) is a flexible technology that provides security, minimizing delay of network data packets and high data transmission. It is mainly used by service providers and is the ideal way to deliver real-time voice and video services. VoIP over Internet Protocol (VoIP) is the ultimate product of many protocols, and with its development it is necessary to develop the integrity and confidentiality issues of its components.

Experience shows that optimal routing of traffic in the service provider's network is associated with many problems, as the providers have channels with high

bandwidth transfers, increasing the capacity of these channels or adding them to big finances. Therefore providers are forced to use the resources available on their network, which can not be done using standard internal routing protocols. At this time specialists refer to traffic engineering (MPLS-TE), which is almost excluded from the service provider's Internet-network.

However, as indicated in the use of MPLS-TE, the network is not insured from some kind of problem that has been developed the NACS system with adaptive algorithms to resolve the problem of networking devices and optimize the configuration of network devices. In addition, it will be possible to visualize these processes and monitor them in real time.

In this work the process of integration of data flow tunnels (Tunnels), Virtual Routing and Forwarding (VRF) protocol automation software is developed, which ensured isolation from the Internet and Internet traffic between each other and from the Global Routing Table.

The work has also developed, so called NACS-system logic scheme of the so-called network configuration, developed by the relevant software code, its implementation technique and the ability to integrate with the monitoring system. With the development of a similar network of real IP-MPLS in virtual software EVE-NG environments, modeling of algorithm, IP-MPLS-network simulation and solving network problems.

შინაარსი

ნახაზების ნუსხა

გამოყენებული აბრევიატურების ნუსხა

აბრევიატურა	განმარტება	
1	2	3
IP	Internet Protocol	ინტერნეტ პროტოკოლი
VOIP	Voice Over Internet Protocol	ხმის გადაცემა ინტერნეტ პროტოკოლის საშუალებით
QoS	Quality Of Service	მომსახურების ხარისხი
SIP	Session Initiation Protocol	სესიის ინიცირების პროტოკოლი
TCP/IP	Transmission Control Protocol/Internet Protocol	გადაცემის კონტროლის პროტოკოლი/ინტერნეტ პროტოკოლი
TE	Traffic Engineering	ტრაფიკის ინჟინერია
DDoS	Distributed denial-of-service attack	განაწილებული შეტევა მომსახურებაზე
VRF	Virtual-Routing/Forwarding	ვირტუალური მარშრუტიზაცია/გადამისამართება
RTP	Real Time Transport Protocol	რეალურ დროში გადაცემის პროტოკოლი
MPLS	Multi Protocol Label Switching	მრავალპროტოკოლიანი კომუტაცია ნიშნულების მიხედვით
PSTN	Public Switched Telephone Network	საერთო სარგებლობის სატელეფონო ქსელი
MGCP	Media Gateway Control Protocol	მედია კარიბჭის კონტროლის პროტოკოლი
BGP	Border gateway protocol	სასაზღვრო ქსელთაშორისი პროტოკოლი
TLS	Transport layer security	სატრანსპორტო დონის უსაფრთხოება

1	2	3
S/MIME	Secure multipurpose Internet mail extensions	ელექტრონული ფოსტის გადაცემის უსაფრთხო პროტოკოლი
SDN	Software-defined networking	პროგრამული უზრუნველყოფით განსაზღვრული ქსელი
IPsec	Internet Protocol Security	ინტერნეტ პროტოკოლის უსაფრთხოება
LSP	Label-switched paths	ნიშნულებიანი კომუტირებადი გზა
iBGP	Internal Border Gateway Protocol	შიდა სასაზღვრო ქსელთაშორის პროტოკოლი
RSVP	Resource reservation protocol	რესურსების რეზერვირების პროტოკოლი
ATM	Asynchronous transfer mode	ასინქრონული გადაცემის რეჟიმი
HTTP	Hypertext transfer protocol	ჰიპერტექსტის გადაცემის პროტოკოლი
(IXP)	Internet exchange point	ინტერნეტის მიმოცვლის წერტილი
TDP	Tag Distribution Protocol	ტეგების განაწილების პროტოკოლი
RSVP	Resource Reservation Protocol	რესურსების რეზერვირების პროტოკოლი
LDP	Label Distribution Protocol	ნიშნულების განაწილების პროტოკოლი
LIB	Label information base	ნიშნულების საინფორმაციო ბაზა
TOS	Type of service	სერვისის ტიპი
TTL	Time-to-live	სიცოცხლის ხანგრძლივობა
API	Application programming interface	გამოყენებითი პროგრამების ინტერფეისი
OID	Object identifier	ობიექტის იდენტიფიკატორი

1	2	3
ISO	International Organization for Standardization	სტანდარტიზაციის საერთაშორისო ორგანიზაცია
CPU	Central processing unit	ცენტრალური პროცესორი
ADSL	Asymmetric digital subscriber line	ასიმეტრიული ციფრული სააბონენტო ხაზი
OSI	Open systems interconnection	ღია სისტემების ურთიერთქმედება
IETF	Internet engineering task force	ინტერნეტ-ინჟინერიის სამუშაო ჯგუფი
SDP	Session Description Protocol	სესიის აღწერის პროტოკოლი
UAC	User Agent Client	მომხმარებელი აგენტი კლიენტი
UAS	User Agent Server	მომხმარებელი აგენტი სერვერი
IVR	Interactive voice response	ინტერაქციული ავტომოპასუხე
CRM	Customer-relationship management	მომხმარებელთა ურთიერთობის მართვა
ZRTP	Zimmermann Real-time Transport Protocol	ზიმერმანის რეალურ დროში მონაცემთა გადაცემის ტრანსპორტის პროტოკოლი
SRTP	Secure Real-time Transport Protocol	უსაფრთხო რეალურ დროში მონაცემთა გადაცემის ტრანსპორტის პროტოკოლი
ASIC	Application-specific integrated circuits	პროგრამისათვის სპეციფიკური ინტეგრალური მიკროსქემა
ERO	Explicit route object	ზუსტი მარშრუტის ობიექტი
HDLC	High-level data-link control	მონაცემთა გადაცემის არხის მაღალი დონის მართვა
PPP	Point-to-point protocol	წერტილიდან წერტილამდე გადაცემის პროტოკოლი

1	2	3
PE	Provider Edge	პროვაიდერის მიჯნა
CE	Customer Edge	მომხმარებლის მიჯნა
BES	Back End Service	უკანა დაბოლოვების სერვისი
AS	Autonomous systems	ავტონომიური სისტემა
NPM	network performance monitoring	ქსელის მუშაობის მონიტორინგი
eBGP	External BGP	გარე სასაზღვრო კარიბჭის პროტოკოლი
IGP	Interior Gateway Protocols	შიდა კარიბჭის პროტოკოლი
EGP	Exterior Gateway Protocol	გარე კარიბჭის პროტოკოლი
iBGP	Internal BGP	შიდა სასაზღვრო კარიბჭის პროტოკოლი
HTTP	Hypertext transfer protocol	ჰიპერტექსტის გადაცემის პროტოკოლი
DHCP	Dynamic host configuration protocol	ჰოსტის დინამიური კონფიგურაციის პროტოკოლი
DNS	Domain name system	დომენების სახელების სისტემა
SCTP	Stream Control Transmission Protocol	ნაკადის გადაცემის კონტროლის პროტოკოლი
SDES	Simplified Data Encryption Standard	გამარტივებული მონაცემთა დაშიფვრის სტანდარტი
IS-IS	Intermediate System to Intermediate System	შუალედური სისტემა შუალედური სისტემისკენ
EIGRP	Enhanced Interior Gateway Routing Protocol	გაძლიერებული შიდა კარიბჭის მარშრუტიზაციის პროტოკოლი
NMS	Network management system	ქსელის მართვის სისტემა
SAL-	Service Abstraction Layer	სერვისის აბსტრაქციის ფენა

1	2	3
SNMP	Simple Network Management Protocol	ქსელის მარტივი მართვის ოქმი
IPSLA	Internet protocol service level agreement	ინტერნეტ პროტოკოლის მომსახურების დონის შეთანხმება
WMI	Windows Management Instrumentation	ვინდოუსის მართვის ინსტრუმენტაცია
OVA	Open Virtualization Alliance	ღია ვირტუალიზაციის ალიანსი
ECMP	Equal-cost multi-path routing	თანაბარი ღირებულების მრავალმხრივი მარშრუტიზაცია
CBR	Constraint-Based Routing	შეზღუდვაზე დაფუძნებული მარშრუტიზაცია
CSPF	Constrained Shortest Path First	პირველადი უმოკლესი შეზღუდული მარშრუტი
OSPF	Open Shortest Path First	პირველადი უმოკლესი მარშრუტი
OLT	Optical Line Terminal	ოპტიკური ხაზის ტერმინალი
DSLAM	Digital subscriber line	ციფრული სააბონენტო ხაზის წვდომის მულტიპლექსორი

შესავალი

ინტერნეტმა თანამედროვე კომუნიკაციების ეპოქაში საოცარი ცვლილებები მოიტანა. მულტიმედიური აპლიკაციების მოთხოვნამ გაზარდა მონაცემთა გადაცემის სიხშირე და მოთხოვნები სიჩქარისადმი. აპლიკაციებისთვის საუკეთესო ხარისხის სერვისის მიწოდება მრავალ პრობლემასთან აღმოჩნდა დაკავშირებული. მრავალპროტოკოლიანი კომუტაცია ნიშნულების მიხედვით (MPLS) მოქნილი ტექნოლოგიაა, რომელიც უზრუნველყოფს უსაფრთხოებას, პაკეტების დაგვიანების მინიმიზაციასა და მონაცემთა გადაცემას მაღალი სიჩქარით.

MPLS-ქსელები სტანდარტული მარშრუტიზაციის ნაცვლად იყენებენ კომუტაციის მეთოდს. MPLS არ ცვლის არსებულ IP-ქსელებს, მაგრამ გვთავაზობს მომსახურების უკეთეს ხარისხს და შესაფერის გარემოს ნაკადის მართვისთვის. MPLS-ის ერთ-ერთი მნიშვნელოვანი ფუნქცია ტრაფიკის ინჟინერინგია (TE). ეს ფუნქცია ქსელური დატვირთვის მინიმიზაციისთვის, რესურსების რეზერვაციისათვის, ბალანსისა და მართვისთვის მნიშვნელოვან როლს ასრულებს. ქსელის მუშაობის ეფექტურობის შეფასება ხდება ქსელის გარკვეული პარამეტრების გათვალისწინებით, რომელთაგან ძირითადია დაყოვნება, პაკეტებს შორის დაყოვნების ცვალებადობა (Jitter) და პაკეტების კარგვის სიხშირე (Packet Loss). MPLS-TE-ქსელთან მომსახურების ხარისხის (QoS) ინტეგრაციით შესაძლებელია ქსელის ეფექტურობის გაუმჯობესება.

MPLS-TE-ს საშუალებით განისაზღვრება მონაცემების უსაფრთხო და ხარისხიანი მარშრუტიზაცია. ქსელში ტრაფიკის დიდი მოცულობა და ისეთი მოულოდნელი ინციდენტები, როგორებიცაა ფიზიკური შეერთების გათიშვა, ქსელური შეტევები და განაწილებული შეტევა მომსახურების დაბლოკვის მიზნით (DdoS), იწვევს მონაცემთა გადაცემის არხებში ტრაფიკის კრიტიკულ დონემდე მატებას და ხშირად არხის გამტარუნარიანობის ლიმიტის შევსებას. გარდა ამისა, მარშრუტიზატორის, კომუტატორის ან

ცალკეული აპარატურული პრობლემის დროს ქსელში საჭიროა მონაცემთა ნაკადის მართვა.

ასეთ დროს უნდა მოხდეს მარშრუტიზაციის კონფიგურაციის მყისიერი ცვლილება. ქსელის სპეციალისტის ჩართულობის შემთხვევაში ეს პროცესი მოითხოვს დიდ დროს და, გარდა ამისა, იზრდება მექანიკური შეცდომების ალბათობა. MPLS-TE-ს არ აქვს საშუალება დინამიურად აირჩიოს სასურველი პრიორიტეტული მონაცემებისთვის უსაფრთხო (დაუტვირთავი) მარშრუტები.

თემის აქტუალურობა. დღესდღეობით ინტერნეტ სერვისებზე დამოკიდებული ადამიანების რიცხვი იზრდება. საბანკო, საგანმანათლებლო და სამეცნიერო-კვლევით საქმიანობასა და სოციალურ ქსელებში ხმოვანი და ვიდეო ინფორმაციის ინტერნეტ პროტოკოლის საშუალებით (VoIP) გადაცემა ერთ-ერთი ყველაზე იაფია. VoIP-სისტემებმა პოპულარობა მოიპოვა თავისი ეფექტურობის, ცოცხალი (Live) ვიდეო ზარების, ზარის ხარისხის, მოქნილობისა და იაფი სერვისის გამო. თუმცა ინტერნეტითა და ტრადიციული IP-ქსელის გამოყენებით რეალურ დროში ისეთი ინფორმაციის გადაცემა, როგორიცაა VoIP, რთული ამოცანაა, რადგან IP-ქსელს არ შეუძლია უზრუნველყოს ხმის და ვიდეო სიგნალების საიმედო მიწოდება. ტრადიციული IP-ქსელი იყენებს საუკეთესო მეთოდს, რომელსაც მხოლოდ შეცდომის კონტროლისა და გარკვეული შეზღუდული გადანაწილების სტრატეგიის განხორციელება შეუძლია. ამ და სხვა პრობლემების დაძლევის მიზნით ინტერნეტ ინჟინერიის სამუშაო ჯგუფის მიერ შემუშავდა ტექნოლოგია სახელწოდებით MPLS.

MPLS-ტექნოლოგიით შესაძლებელია ისეთი პრობლემების გადალახვა, როგორიცაა დიდი დაყოვნებები და პაკეტების კარგვა, რაც ხშირია ტრადიციულ IP-ქსელებში. MPLS-ტექნოლოგიაა, რომელიც სერვის-პროვაიდერების მიერ ძირითადად გამოიყენება რეალურ დროში გადაცემული ხმოვანი და ვიდეოსერვისების გაუმჯობესებული ხარისხით

მისაწოდებლად და იგი შეიძლება ჩაითვალოს როგორც ხმის და ვიდეოკომუნიკაციის იდეალური საშუალება.

კომუნიკაციასთან დაკავშირებული პრობლემური საკითხების გადაწყვეტა შესაძლებელია მონაცემთა მართვის ინჟინერინგის (TE) ტექნოლოგიით, რომელიც წარმოადგენს ინსტრუმენტების კომპლექტსა და ტექნიკას ქსელური მონაცემების სრულყოფილად სამართავად.

სადისერტაციო ნაშრომის მიზანი. ზემოთ თქმულიდან გამომდინარე, ნაშრომის ძირითადი მიზანია ისეთი ამოცანის გადაწყვეტის გზების ძიება, როგორიცაა მონაცემთა გადაცემის პაკეტური საკომუტაციო ქსელის მონიტორინგის სისტემის შემუშავება, რომელიც ქსელში დაფიქსირებული პრობლემების დროს შესაბამისი ალგორითმების დახმარებით გადაწყვეტს ქსელური მოწყობილობების კონფიგურაციის ოპტიმიზაციის ამოცანას, დაახარისხებს შესრულებულ ქმედებებს და, ამასთანავე, შესაძლებელს გახდის პროცესებისადმი დაკვირვებას რეალურ დროში.

კვლევის ამოცანები. დასახული მიზნის მიღწევისათვის ნაშრომში გადაწყვეტილი იქნა შემდეგი ამოცანები:

1. VoIP-ქსელების უსაფრთხოების რისკების შეფასება, მათი ანალიზი და უსაფრთხოების შესაბამისი მოდელის შემუშავება;
2. MPLS-ქსელის არქიტექტურის ანალიზი: MPLS-TE, მონაცემთა ნაკადის გვირაბების (Tunnels) ვირტუალური მარშრუტიზაცია-გადამისამართების (VRF) პროტოკოლების ავტომატიზაციის პროგრამული უზრუნველყოფის მოდულებთან მორგება;
3. ქსელის მონიტორინგის პროგრამული უზრუნველყოფის დანერგვა და სატესტო გარემოში სიმულაცია;
4. ინტერნეტის და ხმოვანი სიგნალების მონაცემების ერთმანეთისგან იზოლირება. ხმოვანი სიგნალის ტრაფიკის წყაროზე დაფუძნებული მარშრუტიზაციის (SR) განხორციელება.
5. მონაცემთა გადაცემის პაკეტური საკომუტაციო ქსელის მონიტორინგის სისტემის აგება;

6. ვირტუალურ გარემოში MPLS-ქსელის აგება და მისი სიმულაცია, რომელიც SNMP-პროტოკოლის საშუალებით ინტეგრირდება მონიტორინგის სისტემასთან.

კვლევის მეთოდოლოგია. დისერტაციაში დასმული ამოცანების გადაჭრისათვის თეორიული და პრაქტიკული საკითხების დამუშავებისას გამოყენებულ იქნა ქსელის გრაფიკული ემულატორი EVE-NG, რომელიც მიესადაგება მარშრუტიზატორის, კომუტატორისა და სასერვერო ოპერაციულ სისტემებს. სიმულაციური ქსელი აიგო შემდეგი ქსელის პროტოკოლებისა და ტექნოლოგიების გამოყენებით: IP, MPLS, LDP, RSVP, SNMP, SSH, OSPF, BGP, Te, VRF, QoSIP-MPLS. ქსელის პროგრამული უზრუნველყოფის დასამუშავებლად გამოყენებული იქნა Node.JS-პლატფორმა და პროგრამირების ენა javascript.

მეცნიერული სიახლე. ნაშრომის თემასთან დაკავშირებული კვლევის შედეგად მიღწეული სამეცნიერო სიახლეა შემუშავებული ალგორითმი, რომელიც საჭირო დროს ახდენს ქსელური მოწყობილობების კონფიგურაციის ოპტიმიზაციას, რაც უზრუნველყოფს მონაცემთა პაკეტების უსაფრთხო მარშრუტიზაციას.

პრაქტიკული ღირებულება და სამუშაოს შედეგების გამოყენების სფერო. ნაშრომის პრაქტიკულ ღირებულებას განაპირობებს დამუშავებული პროგრამა, რომელიც უზრუნველყოფს MPLS-ქსელში გამოყენებულ მონიტორინგის სისტემის ავტომატიზაციას და რომლის დანიშნულებაა ქსელში დაფიქსირებული სხვადასხვა ტიპის ავარიების დროს ქსელური მოწყობილობების კონფიგურაციის ავტომატური ცვლილება.

აპრობაცია. სადისერტაციო ნაშრომის ძირითადი შედეგები მოხსენებული და განხილული იქნა: სტუს-ტელეკომუნიკაციის დეპარტამენტში პირველ მეორე და მესამე კოლოქვიუმებზე 2017-2019წწ. სტუ-სა და ფოჯას უნივერსიტეტის პირველ ერთობლივ R&D საერთაშორისო კონფერენციაზე „მრეწველობის დარგების დინამიკა და თანამედროვე ტენდენციები საქართველოსა და ევროკავშირში: საინფორმაციო-

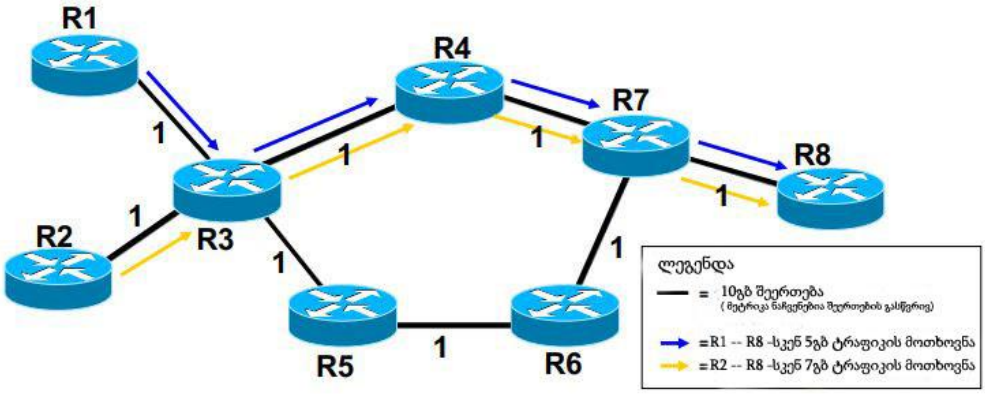
საკომუნიკაციო ტექნოლოგიები მიწოდების ჯაჭვის მენეჯმენტში“ (17-19ოქტომბერი, 2018 წელი, თბილისი, საქართველო); მიღებული შედეგები აისახა სამეცნიერო-ტექნიკურ რეფერირებად ჟურნალში „ენერგია“ (No1 (89), თბილისი 2019 წ.) გამოქვეყნებულ სტატიასა და ჟურნალში "მართვის ავტომატიზებული სისტემები" (სტუ-ს თემატური სამეცნიერო შრომების კრებული No 1 (28), 2019) გამოქვეყნებულ 2 სტატიაში. სტუ-ს ტელეკომუნიკაციების დეპარტამენტში გაფართოებულ სხდომაზე, წინასწარ დაცვაზე 2019წ.

ცნობები დისერტაციის მოცულობისა და სტრუქტურის შესახებ. ნაშრომი შედგება შესავლისაგან, ლიტერატურის მიმოხილვისაგან, ხუთი თავისაგან და დასკვნისაგან. სამუშაოს მოცულობა შეადგენს 153 გვერდს 76 ნახაზით. გამოყენებული ლიტერატურა შეიცავს 44 დასახელებას.

ლიტერატურის მიმოხილვა

სატელეკომუნიკაციო სერვის პროვაიდერები მუდმივად ისწარფვიან თავიანთი IP/MPLS-ქსელის ეფექტურობის გაუმჯობესებისთვის, რის გამოც იზრდება ტრაფიკის ინჟინერიის მნიშვნელობა. ტრადიციული IP-ქსელების შიდა კარიბჭის მარშრუტიზაციის პროტოკოლები (IGP), როგორიცა OSPF და ISIS, პაკეტებს ამისამართებენ IP-ქვექსელის დანიშნულების მიხედვით უმცირესი ღირებულების უმოკლეს გზაზე. უმცირესი ღირებულების გამოთვლა ეფუძნება მარტივი დანამატების მეტრიკას (იგი ასევე ცნობილია როგორც წონა ან ღირებულება), რომელშიც მონაცემთა გადაცემის ყოველ არხს მინიჭებული აქვს საკუთარი მეტრიკა. მაშინ მთელი გზის ღირებულება განისაზღვრება როგორც ამ მეტრიკების ჯამი. უმეტეს შემთხვევაში ქსელში მეტრიკები იქმნება მონაცემთა გადაცემის არხის გამტარუნარიანობის ან ამ არხში მათი გადაცემის შეყოვნების მიხედვით. ამ შემთხვევაში გათვალისწინებული არაა ფაქტობრივი ხელმისაწვდომი (არაუტილიზებული) გამტარუნარიანობა და, შესაბამისად, ამ დროს ტრაფიკი გაიგზავნება კავშირის უმოკლესი გზის გავლით, რაც პოტენციურად იწვევს შემაერთებელი არხის გადავსებას მაშინ, როდესაც ალტერნატიულ მარშრუტებზე მონაცემთა გადაცემის არხები შეიძლება იყოს სრულიად დაუტვირთავი. მაგალითად, განვიხილოთ ნახ. 1-ზე წარმოდგენილი ქსელი, სადაც მონაცემთა გადაცემის თითოეული არხი 10 გბ-იანია და ყოველ აქვს ერთი და იგივე მეტრიკა 1 [1, 2].

R1-დან R8-საკენ და R2-დან R8-საკენ რომ ყოფილიყო შესაბამისად 5 გბ-იანი და 7 გბ-იანი ტრაფიკის მოთხოვნა, მაშინ IGP-პროტოკოლი ორივე მოთხოვნისთვის აირჩევდა ერთი და იმავე მარშრუტს (R1/R2->R3->R4->R7->R8), ვინაიდან გასავლელი არხების მეტრიკათა ჯამია 4 და, აქედან გამომდინარე, ეს მარშრუტი წარმოადგენს უმოკლეს გზას [3].



ნახ. 1. ტრაფიკის ინჟინერია: პრობლემა

ხშირად ამბობენ, რომ კომპიუტერული ქსელი, ძირითად შემთხვევაში, იგეგმება მასში გასატარებელი ტრაფიკის მიხედვით. ტრაფიკის ინჟინერია (TE) არის პროცესი, რომლის დროსაც ტრაფიკი უნდა მოერგოს არსებულ ქსელს ფიქსირებული ქსელის ტოპოლოგიის გათვალისწინებით, რაც ინჟინერს აძლევს ტრაფიკისთვის საჭირო მარშრუტების განაწილებისა და ქსელში არსებული გამტარუნარიანობის რესურსების მაქსიმალურად ეფექტურად გამოყენების საშუალებას. პრაქტიკაში ეს გულისხმობს მთელი რიგი სპეციფიკური ხარისხის მომსახურებების კრიტერიუმების დაბალანსებას, ტრაფიკის ნიმუშების შეცვლას და ზრდას, მონაცემთა პაკეტების დაყოვნებას, მათს სტაბილურობასა და აღდგენის სიჩქარეს გაუმართაობის შემთხვევაში და არხის დატვირთულობის დონეს. დღესდღეობით IP-ქსელებში ტრაფიკის ინჟინერია ხშირად ასოცირდება ნიშნულების მიხედვით მრავალპროტოკოლიან კომუტაციასთან (MPLS) ან, უფრო კონკრეტულად, MPLS-ტრაფიკის ინჟინერიასთან (MPLS-TE). IP-ის მარშრუტიზაცია შესაძლებელია დაიგეგმოს მეტრიკების მანიპულაციითაც MPLS-ის გარეშე. ტრაფიკის ინჟინერიის მთავარი სარგებელი მისი ეკონომიური ღირებულებაა. მაგალითად, ტრაფიკის ინჟინერიის გამოყენებით ნაკლები რესურსებით შესაძლებელია იგივე სერვისის დონის მიღწევა, რაც შეიძლება ყოფილიყო დაგეგმილ ქსელში მისი გამოყენების გარეშეც. მისი გამოყენების შემთხვევაში ასევე შესაძლებელია უფრო მეტი

საერთო ტრაფიკის უზრუნველყოფა ქსელის გამტარუნარიანობის უცვლელობის პირობებში ვიდრე მისი გამოყენების გარეშე. ქსელის ღირებულება ასევე მოიცავს საოპერაციო და მართვის ხარჯებს. ამიტომ ნებისმიერი პოტენციური გამტარუნარიანობის დაზოგვა უნდა იყოს დაბალანსებული ქსელის დამატებით ხარჯებთან და კონკრეტული ტრაფიკის ინჟინერიის გადაწყვეტის სირთულესთან. IP ქსელების უმეტესობა მუშაობს IGP-ის მეტრიკაზე დაფუძნებულ ტრაფიკის ინჟინერიის ხელით მართვის მეთოდით, რომელიც დაკავშირებულია მონაცემთა გადაცემის არხის მეტრიკის მანიპულაციასთან [4, 5].

ერთ-ერთი მცდარი მიდგომაა ტრაფიკით გადავსებული არხის მეტრიკის მცირე ზომებით გაზრდა იქამდე, სანამ ტრაფიკი არ მიაღწევს მისაღებ დონეს. კიდევ ერთი მცდარი მიდგომაა ტრაფიკის ბალანსისთვის მარტივი თანაბარი ღირებულების მქონე მრავალგზიანი (ECMP) iGP-მარშრუტების მოწყობა ქსელის სხვადასხვა კვანძებს შორის ანუ მეტრიკის დაყენება იმდაგვარად, რომ წყვილი კვანძების ორი ან მეტი მარშრუტი გამოვიდეს თანაბარი სიგრძის. ამ შემთხვევაში iGP-მარშრუტიზაცია გაყოფს ტრაფიკს და თანაბრად გადაანაწილებს მონაცემთა გადაცემის არხებში. ეს მარტივი პროცედურაა, თუ ერთ ან ორ კვანძზე ვრცელდება, მაგრამ მრავალი არხის ან კვანძის შემთხვევაში იგი სწრაფად ხდება უმართავი იმდენად, რამდენადაც ყოველ ასეთ არხზე უნდა შეფასდეს მეტრიკის ცვლილების გავლენა. აქედან გამომდინარე, ცხადია, რომ ამგვარი მეტრიკის ცვლილებაზე დაფუძნებული iGP-ის მეთოდებით გამოყენებული ტრაფიკის ინჟინერიის მიდგომები შეზღუდულია. გარდა ამისა, ქსელური ტრაფიკისთვის საჭირო მატრიცის (ე. წ. ტრაფიკის მოთხოვნის მატრიცის) ცოდნის გარეშე მთელ ქსელში შემომავალ და გამომავალ გაერთიანებულ მონაცემთა ნაკადებზე წინასწარ ვერ განისაზღვრება ის, თუ როგორ აისახება არხების დატვირთულობაზე მეტრიკის ცვლილებით გამოწვეული ეფექტი. ამ და სხვა ფაქტორების გამო, დიდი ხნის განმავლობაში, iGP-მეტრიკის მანიპულირებაზე დაფუძნებული ტრაფიკის ინჟინერიის მეთოდი, როგორც

სისტემური, არ გამოიყენებოდა ქსელური ტრაფიკის სამართავად და იგი მონაცემთა გადაცემის არხზე მეტრიკის ცვლილებისას იწვევდა ტრაფიკის გადავსებას სხვა არხში, რის გამოც პრობლემა გადაინაცვლებდა ქსელის სხვა ნაწილში. ამით აიხსნება ის გარემოება, რომ მეტრიკის ცვლილებები უნდა გაკეთდეს ე. წ. ტრაფიკის მოთხოვნის მატრიცის მიხედვით [6].

ტრაფიკის მოთხოვნის დონის გაზომვები ტრადიციულად ხელმისაწვდომი არ იყო IP-ქსელებში. თუმცა, დღესდღეობით არსებობს გარკვეული მეთოდები ასეთი სახის ინფორმაციის მისაღებად [7].

iGP-მეტრიკის ცვლილების მეთოდზე დამყარებული ტრაფიკის ინჟინერიის მეთოდის გარშემო არსებობს რიგი კვლევებისა, სადაც პრობლემების დაძლევის მიზნით განხილულია მისი ოპტიმიზაციის სხვადასხვა პროცედურები. iGP-მეტრიკაზე დაფუძნებული ტრაფიკის ინჟინერია განხორციელდა ავტომატური დაგეგმარების ინსტრუმენტის საშუალებით, სადაც ქსელის ლოგიკური და ფიზიკური ტოპოლოგიისა და ტრაფიკის მოთხოვნის მატრიცის მონაცემების შეყვანის შემდგომ სისტემა გადაცემის არხებისთვის შეიმუშავებს მეტრიკების ოპტიმალურ კომპლექტს. მეთოდის ოპტიმიზაციის მიზნები ძირითადად შეიძლება იყოს გაერთიანებული ტრაფიკის დატვირთვის მინიმუმამდე შემცირება და კლასების მიხედვით ტრაფიკის ოპტიმიზაცია იმ შემთხვევაში, როდესაც გამოყენებულია დიფერენცირებული სერვისების არქიტექტურა. iGP-მეტრიკაზე დაფუძნებულ TE-სა და MPLS-TE-ს მეთოდების ეფექტურობასთან დაკავშირებით აზრთა სხვადასხვაობაა სპეციალისტებს შორის [8].

მარშრუტიზაციის შესახებ გადაწყვეტილების მიღების დროს MPLS-TE, iGP-TE-სგან განსხვავებით, იყენებს MPLS-ის მახასიათებელს. ამ შემთხვევაში მონაცემთა გადაცემისა და მისი კონტროლის ფენები ერთანეთისგან გამოყოფილია და მარშრუტიზაცია აღარ ხდება პაკეტის დანიშნულების მისამართის მიხედვით. MPLS-TE უზრუნველყოფს შეზღუდვებზე დაფუძნებული სამარშრუტო გზის განსაზღვრასა და

გამოკვეთილ მარშრუტიზაციას, რაც შეიძლება გამოყენებულ იქნას მონაცემთა გადაცემის გადავსებული არხებიდან ტრაფიკის მოსახსნელად და სხვა დაუტვირთავ არხში მისი გადამისამართებისთვის ანუ ხელმისაწვდომი რესურსების ოპტიმალურად ათვისებისთვის. ნიშნულებით კომუტირებული გზები MPLS-TE-კონტექსტში ანუ ე. წ. "ტრაფიკის ინჟინერიის გვირაბები" გამოიყენება ქსელში ტრაფიკის ტრანსპორტირებისთვის გადაცემის ისეთი არხებით, რომლებიც iGP-სთვის არ არის უმოკლესი გზა დანიშნულების მისამართისაკენ [9, 10, 11].

MPLS-TE-ს მიერ გამოყენებულ ერთ-ერთ მეთოდს წარმოადგენს შეზღუდვაზე დაფუძნებული მარშრუტიზაციის გზის განსაზღვრა, რომელიც შეიძლება განხორციელდეს ონლაინ რეჟიმში დინამიურად მარშრუტიზაციის გვირაბის წყაროების ("ე.წ. გვირაბის ხელმძღვანელი კვანძების") მიერ, რომლებიც განიხილება როგორც დინამიური გზის პარამეტრი და როგორც ცენტრალიზებული ავტონომიური ფუნქცია. მარშრუტიზაციის გზის განმსაზღვრელი ცნობილია როგორც გვირაბის სერვერი ან მარშრუტიზაციის გზის განმსაზღვრელი ელემენტი, რომელიც განსაზღვრავს გვირაბის ზუსტ გზას და რომელსაც გვირაბის ხელმძღვანელი კვანძი გამოიყენებს კონკრეტული გვირაბისთვის და იგი განიხილება როგორც ზუსტი გზის ვარიანტი. შეზღუდვაზე დაფუძნებული მარშრუტიზაციის (CBR) ალგორითმი იყენებს შეზღუდვების უმოკლესი გზის (CSPF) განსაზღვრის მეთოდს, რომელსაც კონკრეტული გვირაბი გამოიყენებს მოქმედი ქსელის გამტარუნარიანობის რესურსების (საჭიროებისამებრ შეზღუდვებით) და ამ გვირაბისთვის აუცილებელი გამტარუნარიანობის შესაბამისად. არასაკმარისი სიჩქარის ან გვირაბის გამტარუნარიანობის შეზღუდვის პოლიტიკის დარღვევის გამო არსებული ტოპოლოგიიდან გვირაბისთვის შერჩეული იქნება უმოკლესი (მცირე ღირებულების) გზა. ონლაინ ან ავტონომიური გაანგარიშების მეთოდის შემთხვევაში RSVP-გვირაბის სასიგნალო კავშირისთვის კვანძიდან კვანძამდე გვირაბის გზას განსაზღვრავს ე. წ. ზუსტი მარშრუტის ობიექტი

(ERO). ERO-ს შესახებ ინფორმაცია გადაეცემა რესურსების აღდგენის პროტოკოლის საშუალებით (RSVP) და მისი დანიშნულებაა გვირაბის LSP-ს სიგნალიზაცია. უფრო მეტი საჭიროების შემთხვევაში MPLS-TE-ს საშუალებით შესაძლებელია ერთმანეთისგან დამოუკიდებელი მარშრუტიზაციის კონფიგურაციების შედგენა ვიდრე iGP-მეტრიკაზე დაფუძნებულ TE-ში [11].

იმ შემთხვევაში, როდესაც განიხილება ქსელში ავარიის არსებობის სცენარი, MPLS-TE-ს ზუსტი გზის განსაზღვრის პარამეტრის გამოყენებისას მოსალოდნელია უკეთესი შედეგის მიღწევა ვიდრე iGP-მეტრიკაზე დაფუძნებული მარშრუტიზაციის დროს ქსელის უავარიო სცენარში. ნებისმიერი სახის ავარიის შემთხვევაში iGP-მარშრუტიზაცია დამოკიდებულია ერთი და იგივე მეტრიკის პარამეტრებზე. TE-გვირაბების ზუსტი გზის პარამეტრით შესაძლებელია დამოუკიდებელი მარშრუტების განხორციელება. მეორეს მხრივ, MPLS-TE-ს ზუსტი გზების მართვა IP-ქსელში უფრო რთულია, ვიდრე მხოლოდ IP-მარშრუტიზაციისა. ამასთან დაკავშირებით საინტერესოა იმის გარკვევა, თუ რამდენად ამართლებს MPLS-TE-ს გამოყენებით მიღწეული ეფექტურობა მისი მართვით გაჩენილ დამატებით ადმინისტრაციულ სირთულეებს [12].

თანამედროვე IP-ქსელებში ხშირია დაზიანებები, მათ შორის ცალკეული ქსელური ელემენტის, მისი მაკავშირებელი ნაწილების ან ოპტიკური კაბელის დაზიანებები. ამიტომ უზრუნველყოფილი უნდა იყოს ქსელის საკმარისი გამტარუნარიანობა და ტრაფიკის მარშრუტიზაციის შეცვლის შესაძლებლობა. დაზიანების სხვადასხვა სცენარისთვის ქსელის მდგრადობის აღწერის ერთ-ერთი სასარგებლო მეთოდია ყველაზე მაღალი დატვირთვის დონის გამოთვლის მეთოდი.

ტრაფიკის მარშრუტიზაციის გზის განსაზღვრის პროცესში გასათვალისწინებელია დაზიანებების ძირითადი შემთხვევების სამი ჯგუფი [13]:

1. სატრანსპორტო ქსელის დაზიანებით გამოწვეული ერთი არხის დაზიანება;

2. მარშრუტიზატორის ან კომუტატორის დაზიანებით გამოწვეული ქსელური კვანძის დაზიანება;

3. ოპტიკური კაბელის დაზიანებით გამოწვეული საერთო რისკის არხების ჯგუფის (SRLG) დაზიანება.

ქსელური ტრაფიკის მოთხოვნის მატრიცა შეიცავს ინფორმაციას გაერთიანებული ტრაფიკის (მაგალითად, ორ მარშრუტიზატორს შორის გამავალი ტრაფიკის) პარამეტრების გაზომვის შედეგების, მათი პროგნოზირებისა და შეფასების შესახებ. ქსელის ტრაფიკის მატრიცისა და ქსელის მარშრუტიზაციის მოდელისცოდნის საფუძველზე შესაძლებელია იმის დადგენა, თუ რა შედეგები შეიძლება მოყვეს მარშრუტიზაციის სქემის ცვლილებას [14].

თავი 1. მრავალპროტოკოლიანი კომუტაცია ნიშნულების მიხედვით (MPLS)

1.1.1. ნიშნულებიანი მარშრუტიზატორი (LSR)

ნიშნულებიანი მარშრუტიზატორი (Label Switch Router) არის მარშრუტიზატორი, რომელიც გამოიყენება MPLS-ის ფუნქციონირებისთვის და რომელსაც შეუძლია MPLS-ნიშნულების გარჩევა, მიღება და გადაცემა.

MPLS-ქსელში არსებობს სამი სახეობის LSR:

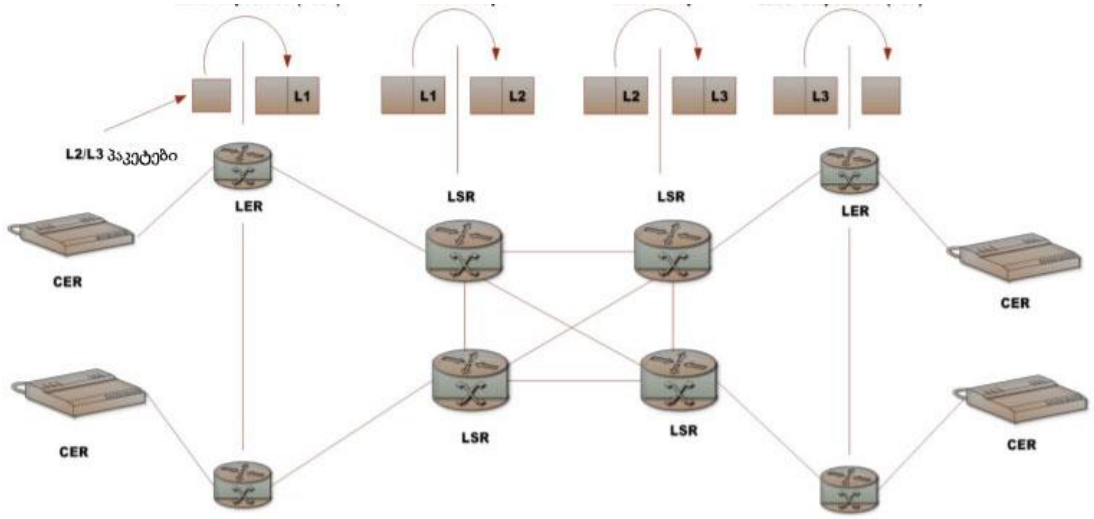
1. შემსვლელი LSR. იგი ღებულობს დაუნიშნავ პაკეტს, უკეთებს ნიშნულს(ებს) პაკეტის წინსართად და გზავნის მონაცემთა გადაცემის შრეზე;
2. გამსვლელი LSR. იგი ღებულობს დანიშნულ პაკეტს, აშორებს ნიშნულს(ებს) და გზავნის მონაცემთა გადაცემის შრეზე;
3. შუამავალი LSR. იგი ღებულობს შემომავალ დანიშნულ პაკეტს, ატარებს მასზე ოპერაციას, ცვლის ნიშნულებს პაკეტზე და გზავნის მონაცემთა გადაცემის შრეზე.

შევნიშნოთ, რომ შემსვლელი და გამსვლელი LSR-ები არის პროვაიდერის მოსაზღვრე LSR-ები.

LSR-ს შეუძლია განახორციელოს სამი ოპერაციიდან ერთ-ერთი: მოშორება (pop); დამატება (push) ან შეცვლა (swap).

LSR-ს პაკეტის გაგზავნამდე უნდა შეეძლოს ერთი ან რამდენიმე ზედა ნიშნულის მოშორება და, აგრეთვე, ერთი ან რამდენიმე ნიშნულის დამატება მიღებულ პაკეტზე. თუ მიღებული პაკეტი უკვე დანიშნულია LSR-დასტაში, მაშინ ის ამატებს ერთ ან რამდენიმე ნიშნულს და შემდგომ გზავნის პაკეტს. თუ პაკეტი ჯერ არაა დანიშნული, მაშინ LSR ქმნის ნიშნულების დასტას და შემდეგ ამატებს ნიშნულს პაკეტზე. LSR-ს ასევე უნდა ქონდეს ნიშნულების ცვლილების შესაძლებლობა, რაც მარტივად ნიშნავს იმას, რომ დანიშნული პაკეტის მიღებისას ნიშნულების დასტაში არსებული ზედა ნიშნული იცვლება ახლით და პაკეტი გადაეცემა გამავალ ინტერფეისს [15, 16].

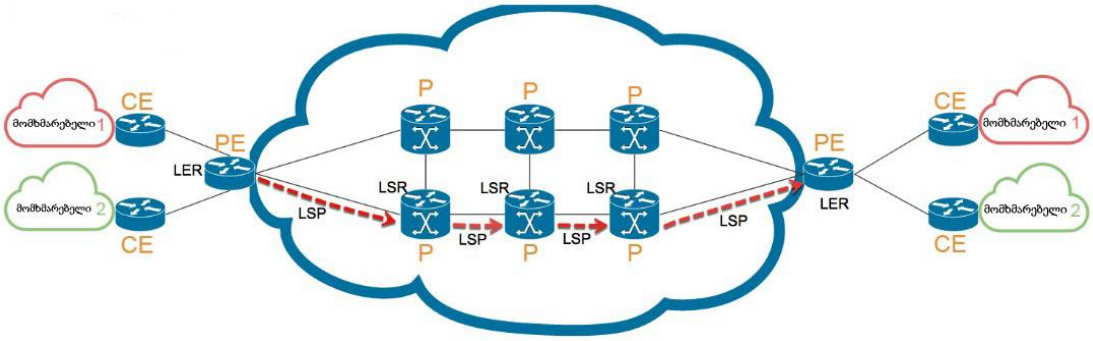
MPLS-VPN-ის შემთხვევაში შემსვლელ და გამსვლელ LSR-ებს ეძახიან პროვაიდერის სასაზღვრო მარშრუტიზატორებს (PE), ხოლო შუამავალ LSR-ს - პროვაიდერის მარშრუტიზატორს (P). ტერმინები PE და P მარშრუტიზატორი იმდენად პოპულარული გახდა, რომ ზოგიერთი სპეციალისტი აღნიშნულ ტერმინებს მაშინაც კი იყენებს, როდესაც MPLS-ქსელში MPLS-VPN არაა გააქტიურებული (ნახ. 2).



ნახ. 2. MPLS-ის ნიშნულების ოპერაცია

1.1.2. ნიშნულებიანი გზა (LSP)

ნიშნულებიანი გზა LSP წარმოადგენს LSR-ების მიმდევრობას, რომლებიც დანიშნულ პაკეტებს გადასცემენ MPLS-ქსელში. LSP არის გზა, რომელსაც პაკეტი გადის MPLS-ქსელში. LSP-ში პირველი LSR არის შემსვლელი LSR, ხოლო ბოლო LSR - გამსვლელი LSR. დანარჩენი ყველა LSR, რომელიც მოთავსებულია მიმღებ და გამსვლელ LSR-ებს შორის, შუამავალი LSR-ია (ნახ. 3).



ნახ. 3. ნიშნულებიანი გზა (LSP)

1.1.3. მარკერების განაწილების პროტოკოლი (LDP)

LDP წარმოადგენს ნიშნულების განაწილების პროტოკოლს და მისი დანიშნულებაა შიდა მარშრუტიზაციის პროტოკოლის პრეფიქსების ფორმირება. LDP არ არის ერთადერთი პროტოკოლი, რომელსაც შეუძლია MPLS-ნიშნულების განაწილება.

ნიშნულების განაწილების პროტოკოლებია:

1. ტეგების განაწილების პროტოკოლი (TDP);

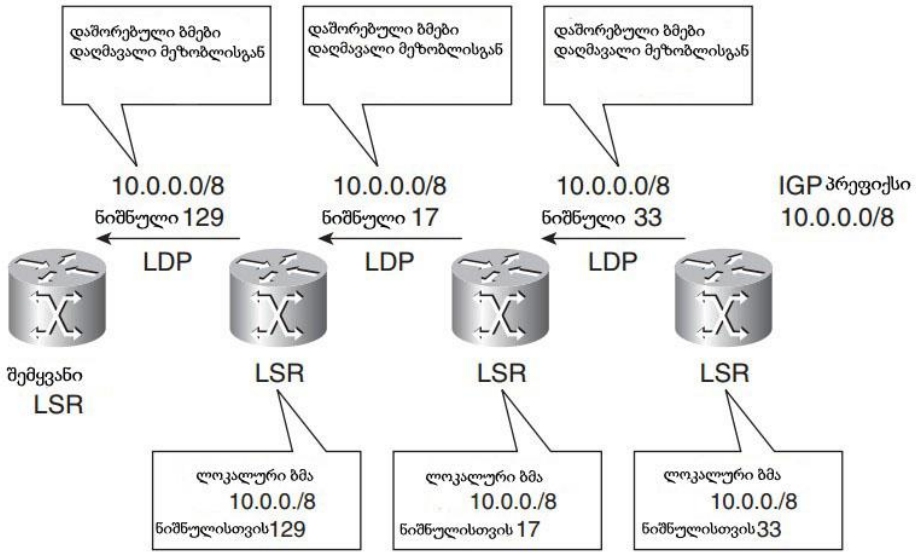
2. ნიშნულების განაწილების პროტოკოლი (LDP);

3. რესურსების რეზერვაციის პროტკოლი (RSVP).

TDP განიხილება როგორც ნიშნულების განაწილებისთვის შექმნილი პირველი პროტოკოლი, რომელიც შექმნა კომპანია Cisco-მ და ის წარმოადგენს მის საკუთრებას. მოგვიანებით შეიქმნა LDP-პროტოკოლი, რომელიც, მოქმედების თვალსაზრისით, იგივეა, რაც TDP-პროტოკოლი, მაგრამ LDP-ს გააჩნია მეტი ფუნქციონალური შესაძლებლობები, რის გამოც მან მალევე ჩაანაცვლა TDP. მარშრუტიზაციის IP-პროტოკოლის მარშრუტიზაციის ცხრილში არსებული თითოეული IP-ის პრეფიქსისთვის ყველა LSR ქმნის ლოკალურ ბმას, ანუ ის ნიშნულს აბამს IPv4-პრეფიქსს. ამავე დროს LSR ყველა LDP-მეზობელთან ანაწილებს თავისი ბმების ცხრილს.

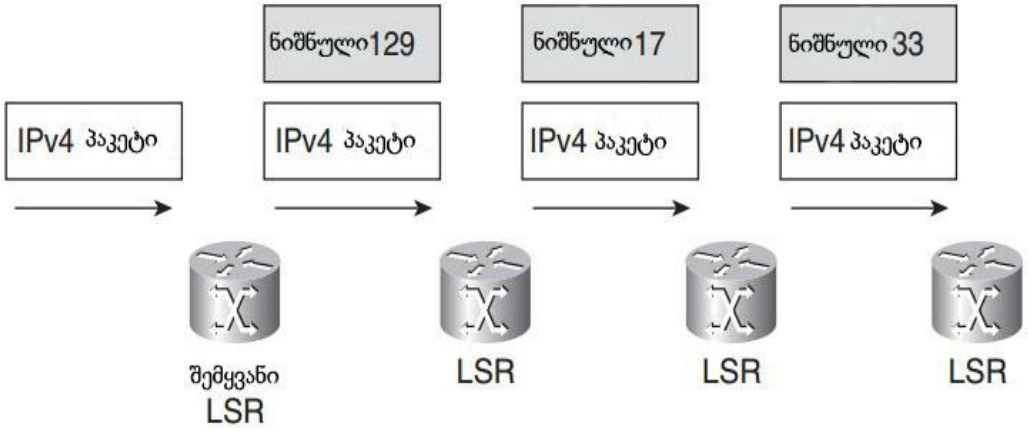
მეზობელი მარშრუტიზატორებისგან მიღებული ბმები დაშორებული ბმებია. მარშრუტიზატორი დაშორებულ და ლოკალურ ბმებს ინახავს სპეციალურ ცხრილში, რომელსაც ნიშნულების ინფორმაციულ ბაზას (LIB) უწოდებენ. თითოეულ LSR-ს თითოეული IP-პრეფიქსისთვის აქვს მხოლოდ ერთი ლოკალური ბმა [16].

ნახ. 4-ზე წარმოდგენილ სურათზე ნაჩვენებია თითოეული LSR-ის მიერ მინიჭებული მარკირება კონკრეტული პრეფიქსისთვის და ამ ინფორმაციის LDP-მეზობლებთან გაცვლა. შემდეგ, როგორც ავღნიშნეთ, ეს ინფორმაცია ინახება სპეციალურ ბაზაში (LIB-ში).



ნახ. 4. ნიშნულების განაწილების პროტოკოლი (LDP)

ნახ. 5-ზე კონკრეტულად ჩანს IPv4-პაკეტი, რომლის დანიშნულების მისამართია 10.0.0.0/8 და რომელიც შედის MPLS-ქსელის მიმღებ LSR-ში, სადაც პაკეტს ემატება ნიშნული ნომრით 129 და გადაეცემა მეორე LSR-ს. მეორე LSR-პაკეტში ცვლის შემომავალ ნიშნულს 129-ს გამავალი ნიშნულით 17-ით და მას უგზავნის მესამე LSR-ს. მესამე LSR შესაბამისად შემომავალ ნიშნულს 17-ს ცვლის გამავალი ნიშნულით 33-ით და ამ უკანასკნელს უგზავნის შემდეგ LSR-ს და ა.შ.



ნახ. 5. პაკეტების გადაცემა MPLS ქსელში

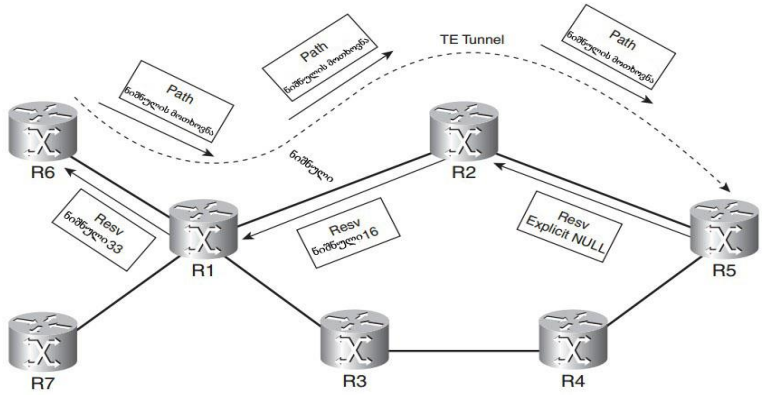
შუამავალ LSR-ებს არ ესაჭიროებათ MPLS-ის შიგთავსის ცოდნა იმიტომ, რომ ერთადერთი ინფორმაცია, რაც მათ პაკეტის გასაგზავნად ჭირდებათ, მხოლოდ ზედა ნიშნულია. თუ მარკირების დასტა შეიცავს ერთზე მეტ ნიშნულს, მაშინ შუამავალ LSR-ს აღარ აინტერესებს დანარჩენი ნუშნულები გარდა ზედა პირველი ნიშნულისა. გამსვლელი LSR-პაკეტს აშორებს ყველა ნიშნულს. მას სწორი გადამისამართებისთვის ჭირდება MPLS-ის შიგთავსის ცოდნა.

1.1.4. რესურსების რეზერვირების პროტოკოლი (RSVP)

ნიშნულების განაწილება RSVP-პროტოკოლით გამოიყენება მხოლოდ MPLS-TE-ს შემთხვევაში. გზის მონიშვნისთვის RSVP იყენებს PATH და RESV-შეტყობინებებს. TE-ს საწყისი მარშრუტიზატორი გზავნის PATH-შეტყობინებას ბოლო მარშრუტიზატორთან, საიდანაც RESV-შეტყობინება საწყის მარშრუტიზატორს უბრუნდება ზუსტად იგივე გზით. თითოეული LSR, რომელიც TE-ს გვირაბის ტრაფიკმა უნდა გაიაროს, იწერება განსაზღვრულ მარშრუტების სიაში, ანუ ERO-ში, რომელიც ინტერფეისების IP-მისამართების მიმდევრობების სიას წარმოადგენს. ყოველი IP-მისამართი

აღნიშნავს გზაში გასავლელ თითო LSR-ს. PATH-შეტყობინება იგზავნება საწყისი მარშრუტიზატორიდან მომდევნო მარშრუტიზატორთან, რომელიც თავის IP-მისამართს ამოშლის ERO-დან, პოულობს მომდევნო IP-მისამართს და აღნიშნულ შეტყობინებას უგზავნის შესაბამის მომდევნო მარშრუტიზატორს. ასე გრძელდება მანამ, სანამ ბოლო მარშრუტიზატორი არ მიიღებს PATH-შეტყობინებას. ბოლო მარშრუტიზატორი RESV-შეტყობინებას უკან აბრუნებს ზუსტად იგივე გზით, რომელიც განვლო PATH- შეტყობინებამ. იმ შემთხვევაში, როდესაც საწყისი მარშრუტიზატორი ყოველგვარი RSVP-შეცდომების გარეშე მიიღებს RESV-შეტყობინებას, მოინიშნება გზა და TE-გვირაბიც ჩაირთვება. TE-გვირაბის საწყისი მარშრუტიზატორი განსაზღვრავს საუკეთესო გზას, რომელიც გვირაბში გამავალმა პაკეტებმა უნდა გაიარონ. გზის დინამიურად განსაზღვრის გარდა ქსელის ადმინისტრატორმა წინასწარ შეიძლება დააკონფიგუროს TE-გვირაბის გზა. ნებისმიერ შემთხვევაში საწყისმა მარშრუტიზატორმა ზუსტად იცის, თუ რა გზით უნდა გაიაროს ტრაფიკმა დანიშნულების ადგილამდე [16, 17, 18].

RSVP-პროტოკოლის დანიშნულებას, TE-გვირაბის გზის მონიშვნის გარდა, წარმოადგენს MPLS-ის ნიშნულის გადატანა იმ მიზნით, რომ გვირაბის გამოყენებულ გზაზე უზრუნველყოფილი იყოს პაკეტების გადაცემა ნიშნულების მიხედვით (ნახ. 6).



ნახ. 6. რესურსების რეზერვაციის პროტოკოლი (RSVP) ტრაფიკის ინჟინერიისთვის

1.1.5. MPLS-ის უპირატესობები

IP-ქსელში MPLS-ის დანერგვის უპირატესობებია:

1. ერთიანი ქსელის ინფრასტრუქტურის გამოყენება;

2. BGP-ისგან თავისუფალი ქსელის ბირთვი;

3. MPLS-VPN-ის ერთრანგიანი მოდელი;

4. ტრაფიკის ოპტიმალური გადაადგილება;

5. ტრაფიკის ინჟინერია.

პირველ რიგში განვიხილოთ მცდარი შეხედულება ქსელში MPLS-ის დანერგვის მიზეზის შესახებ. იგი წინათ შეიძლება ყოფილიყო გონივრული, თუმცა დღესდღეობით ამ მიზეზით MPLS-ის განხორციელება არ არის გამართლებული. IP-პაკეტის კომუტაცია ცენტრალური პროცესორის გამოყენებით მიიჩნეოდა უფრო ნელა განსახორციელებლად, ვიდრე პაკეტების კომუტაცია ნიშნულების მიხედვით. IP-პაკეტების მარშრუტიზაცია ხდება IP-თავსართში მიმღების (დანიშნულების) IP-მისამართის ველის გადამოწმებით და მარშრუტიზაციის ცხრილში მარშრუტიზაციისთვის საუკეთესო ვარიანტის მოძიებით. შესაბამისად, ვინაიდან IP-მისამართი შედგება 4 ოქტეტისგან, შესაძლებელია მარშრუტიზაციის ცხრილში მისამართის ძიება კომპლექსური ყოფილიყო. კომპლექსური ძიება კი ნიშნავს იმას, რომ IP-პაკეტის გაგზავნისთვის გადაწყვეტილების მიღებას შეიძლებოდა დაჭირვებოდა გარკვეული დრო. ამიტომ ზოგი თვლიდა, რომ პაკეტების კომუტაცია ნიშნულების მიხედვით უფრო სწრაფი იყო IP-მისამართის მიხედვით კომუტაციის გამოყენებასთან შედარებით. მაგრამ ვინაიდან დღეისათვის მარშრუტიზატორებს აქვთ მონაცემთა გადაცემის 100 გიგაბიტიანი გამტარუნარიანობის არხები, შესაბამისად შეუძლებელი იქნებოდა ასეთი მოცულობის IP-ტრაფიკის გატარება ცენტრალური პროცესორის გამოყენებით. ცენტრალური პროცესორი არსებობს უმთავრესად იმისთვის, რომ გაუმკლავდეს ე. წ. კონტროლის სიბრტყის (Control plane) ტრაფიკს. Control plane არის

პროტოკოლების ნაკრები, რომელიც გამოიყენება ე. წ. მონაცემთა სიბრტყის “Data plane”-ის შექმნისთვის. Control plane-ის მთავარი კომპონენტებია მარშრუტიზაციის პროტოკოლები, მარშრუტიზაციის ცხრილები და სხვა საკონტროლო თუ სასიგნალო პროტოკოლები, რომლებიც გამოიყენებიან Data plane-ის უზრუნველყოფისთვის. Data plane არის პაკეტების გაგზავნის გზა მარშრუტიზატორის ან კომუტატორის გავლით. დღეისათვის პაკეტების კომუტაცია ხორციელდება სპეციალურად ჩაშენებული ჩიპით (ASIC-ით). მარშრუტიზაციისთვის ASIC-ების გამოყენებამ გაათანაბრა IP-პაკეტებისა და მარკირებული პაკეტების გადაცემის სისწრაფე. ასე რომ, თუ ქსელში MPLS-ის გამართვის ერთადერთი მიზეზია ქსელის სიჩქარის გაზრდა, მაშინ ის არ არის გამართლებული.

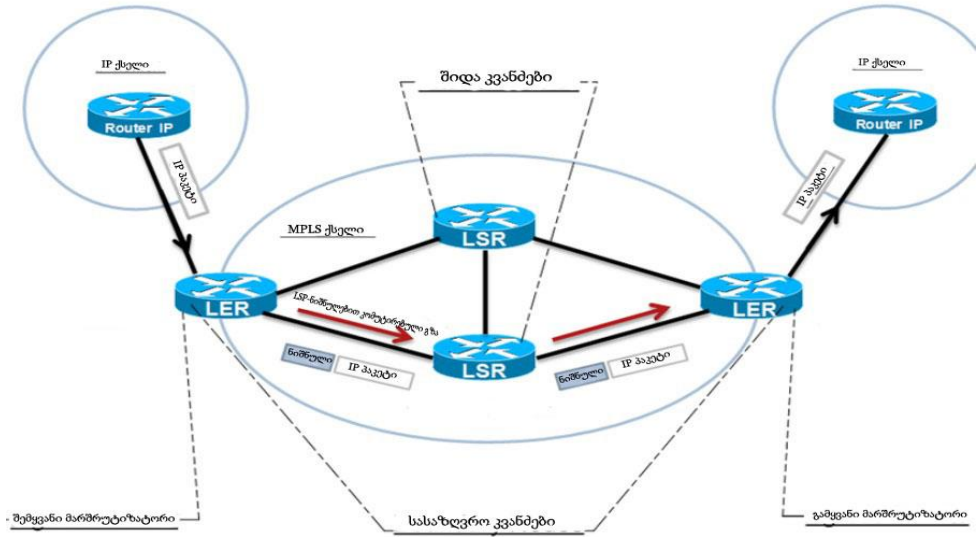
1.2. ერთიანი ქსელის ინფრასტრუქტურის გამოყენება

MPLS-ის დანიშნულებაა შემოსული პაკეტის მონიშვნა ამ პაკეტის მიმღების მისამართის ან სხვა წინასწარ დაკონფიგურირებული კრიტერიუმის მიხედვით და ტრაფიკის კომუტაციის განხორციელება საერთო ინფრასტრუქტურის გავლით, რაც მის უდიდეს უპირატესობას წარმოადგენს მხოლოდ IP-მარშრუტიზაციის გამოყენებასთან შედარებით.

MPLS-ის IP-ისთან ერთად გამოყენებისას შესაძლებელია ქსელური ტრანსპორტირების შესაძლებლობების გაზრდა. პაკეტებზე ნიშნულების დამატებით შესაძლებელია სხვადასხვა (მაგალითად, IPv4, IPv6, Ethernet, HDLC, PPP და სხვა) პროტოკოლების გადატანა IP/MPLS-ქსელში. ფუნქციას, რომლითაც შესაძლებელია ნებისმიერი მეორე შრის კადრების გადატანა MPLS-ქსელში ეწოდება AToM. მარშრუტიზატორებს, რომლებიც ატარებენ AToM-ტრაფიკს, არ ჭირდებათ MPLS-ის შიგთავსის ცოდნა. მათთვის

მთავარია, რომ გაატარონ მონიშნული ტრაფიკი და ამისთვის მათ მხოლოდ ჭირდებათ ნიშნულების ნახვა. მოკლედ, კომუტაცია MPLS-ნიშნულების მიხედვით სხვადასხვა პროტოკოლების შესაბამის საერთო ქსელში გადაცემისთვის მარტივი მეთოდია. სატელეკომუნიკაციო სერვის პროვაიდერებს AToM აძლევს მომხმარებლებისთვის ერთი და იგივე მეორე შრის სერვისის მიწოდების საშუალებას [19].

უკანასკნელ შემთხვევაში სერვის პროვაიდერს მხოლოდ სჭირდება ერთიანი ქსელის ინფრასტრუქტურა მომხმარებლისთვის ნებისმიერი სახის ტრაფიკის მიწოდებისთვის, რაც ნაჩვენებია ნახ. 7-ზე მოყვანილი სქემის სახით.



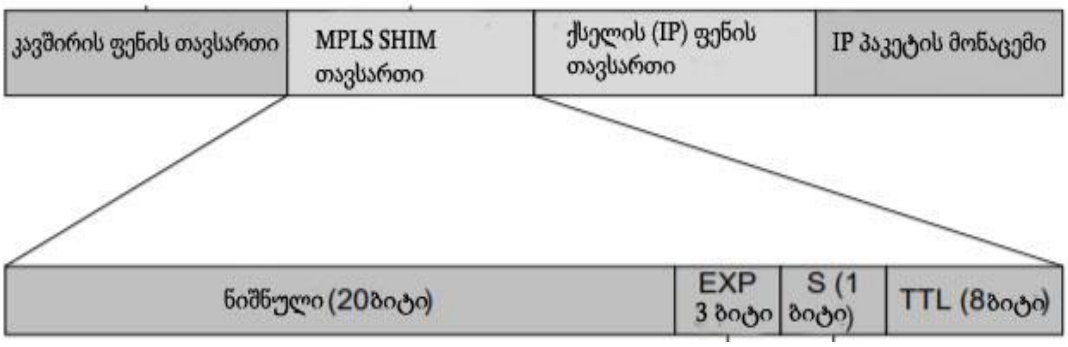
ნახ. 7. IP-პაკეტის ტრანსპორტი MPLS ქსელის გავლით

1.1.MPLS-ის არქიტექტურა

ტრადიციულ IP-ტექნოლოგიაში მარშრუტიზატორის გადამისამართების გადაწყვეტილება ეფუძნება პაკეტის სათაურის შესწავლას და მარშრუტიზაციის ალგორითმის მუშაობის პრინციპს OSI მოდელის ქსელის შრეზე. ჩვეულებრივ შემთხვევაში, როდესაც

მარშრუტიზატორში შედის პაკეტი, მარშრუტიზატორმა უნდა განსაზღვროს პაკეტის მიმართულება. MPLS-ქსელში პაკეტებს ნიშნულის სახით ენიჭება გადამისამართების ექვივალენტობის კლასი (FEC), რომელიც გამოიყენება კვანძებზე გადამისამართებისთვის IP-თავსართის გადამოწმების გარეშე. ყველა მარშრუტიზატორს აქვს FEC-ცხრილი. MPLS მოქმედებს OSI-მოდელის მონაცემთა და ქსელის დონეებს შორის. ამიტომ მას უწოდებენ 2.5 დონის ოქმს [20].

MPLS მრავალპროტოკოლიანი მაღალი წარმადობის WAN-ტექნოლოგიაა, რომელიც აგზავნის მონაცემებს ერთი მარშრუტიზატორიდან მეორესაკენ მოკლე გზის ნიშნულების საფუძველზე და არა IP-ქსელის მისამართებით. MPLS-ს აქვს რამდენიმე განმსაზღვრელი მახასიათებელი. ის მრავალპროტოკოლიანია, რაც იმას ნიშნავს, რომ მას აქვს უნარი გადაიტანოს ნებისმიერი „ტვირთი“ IPv4, IPv6, Ethernet, ATM, DSL და Frame Relay ტრაფიკების ჩათვლით. მონაცემთა გადაცემის დროს გამოიტანება ნიშნულები, რომელთა საშუალებითაც მარშრუტიზატორი იღებს გადამისამართების გადაწყვეტილებას. კომუტაცია ნიშნულების გამოყენებით ნიშნავს იმას, რომ ქსელში გადაცემული პაკეტები აღარაა არც IPv4-პაკეტი, არც IPv6-პაკეტი და არც მეორე შრის კადრი, არამედ იგი წარმოადგენს დანიშნულ პაკეტს. ერთი MPLS-ნიშნული არის 32-ბიტიანი ველი, რომელსაც გააჩნია კონკრეტული სტრუქტურა (ნახ. 8).



ნახ. 8. MPLS-ველი

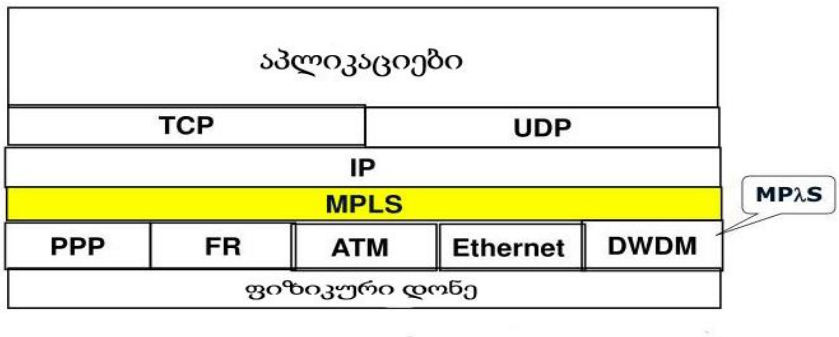
MPLS-ველის პირველი 20 ბიტი (მათი ნომრებია 0-დან 19-ის ჩათვლით) ნიშნულის მნიშვნელობის განმსაზღვრელია და ეს მნიშვნელობა შეიძლება მოთავსებული იყოს 0-დან 2^{20}-1-მდე დიაპაზონში. ამ დიაპაზონის პირველი 16 მნიშვნელობა ამოღებულია ნორმალური ხმარებიდან, ვინაიდან მათ აქვთ სპეციფიკური დანიშნულება. MPLS-ველის მომდევნო 3 ბიტი (20-დან 22-მდე) ექსპერიმენტალური ბიტებია (EXP). ისინი გამოიყენება მხოლოდ სერვისის ხარისხისთვის (QoS). მომდევნო 1 ბიტი (23) ბიტი BoS ბიტია და მისი მნიშვნელობაა 0, როცა ის არ არის დასტის ქვედა ნიშნული, ხოლო 1-ია, თუ ის წარმოადგენს ქვედა ნიშნულს. დასტაში იგულისხმება ნიშნულები, რომლებიც პაკეტზეა მიბმული და დასტა შეიძლება შედგებოდეს ერთი ან რამოდენიმე ნიშნულისგან. დასტაში ნიშნულების რაოდენობა შეუზღუდავია, თუმცა იშვიათად გვხვდება 4 ან მეტი ნიშნული. MPLS-ველის შემდეგი 8 ბიტი (24-დან 31-მდე) გამოიყენება TTL-ისთვის (Time to live), რომელსაც ზუსტად იგივე ფუნქცია აქვს, რაც IP-თავსართში არსებულ TTL-ს. მის მნიშვნელობას მარტივად აკლდება 1 თითოეული მარშრუტიზატორის გავლის შემდეგ და მისი მთავარი ფუნქციაა თავიდან აიცილოს პაკეტის უწყვეტი ტრიალი მარშრუტიზაციის წრეში. TTL-ის გარეშე პაკეტი ყოველთვის იტრიალებს ციკლურად წრეზე. თუ TTL-ის მნიშვნელობა 0-ს მიაღწევს, პაკეტი გაუქმდება.

MPLS-მარშრუტიზატორებს მარშრუტიზაციისთვის შესაძლებელია დასჭირდეთ ერთზე მეტი ნიშნული, რაც მიიღწევა დასტაში ნიშნულების შეფუთვით. დასტაში პირველ ნიშნულს ეწოდება ზედა ნიშნული (top label), ხოლო უკანასკნელს - ქვედა ნიშნული (bottom label). მათ შორის შესაძლებელია ნებისმიერი რაოდენობის ნიშნულების არსებობა.

ნიშნული	EXP	0	TTL
ნიშნული	EXP	0	TTL
...			
ნიშნული	EXP	1	TTL

ნახ. 9. MPLS-ის ნიშნულების დასტა

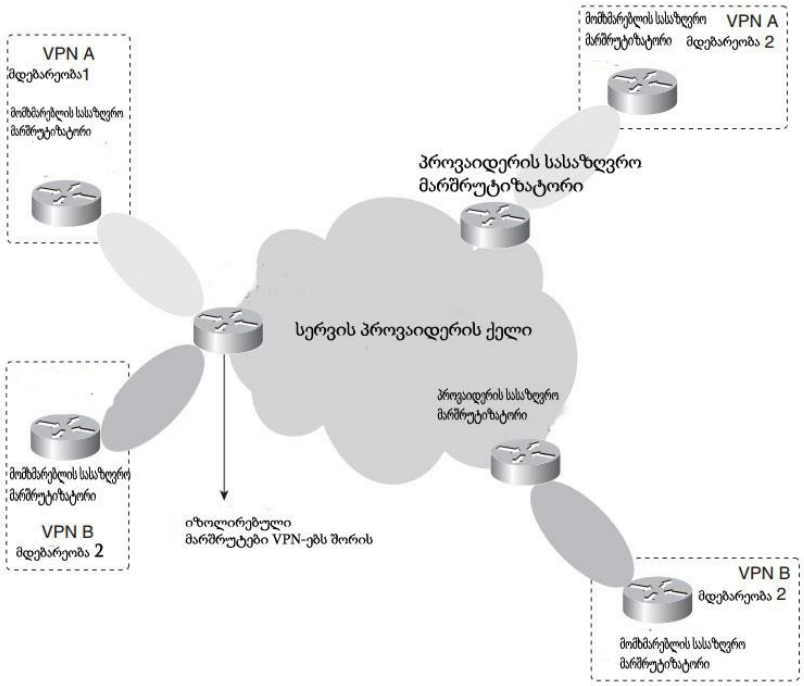
როგორც ნახ. 9-დან ჩანს, BoS ბიტი ყველა ნიშნულისათვის არის 0, გარდა ბოლო ნიშნულისა. ბოლო ნიშნულის BoS ბიტი არის 1. ზოგიერთ MPLS-აპლიკაციას გადამისამართებისთვის დასტაში ესაჭიროება ერთზე მეტი ნიშნული. MPLS-VPN და AtoMMPLS-აპლიკაციებია. ორივე აპლიკაცია დასტაში ორ-ორ ნიშნულს აკრავს. MPLS არაა OSI-მოდელის მეორე შრის პროტოკოლი (Layer 2), რადგან მეორე შრის ენკაპსულაცია მარკირებულ პაკეტთან ერთადაა წარმოდგენილი. MPLS ასევე არ არის მესამე შრის (Layer 3) პროტოკოლი, ვინაიდან მესამე შრის პროტოკოლიც წარმოდგენილია პაკეტში მარკერებთან ერთად. ამიტომ MPLS არც ისე კარგად ერგება OSI-მოდელს. ყველაზე მარტივ გზას წარმოადგენს MPLS-ის განხილვა 2.5 შრის პროტოკოლის სახით (ნახ. 10).



ნახ. 10. MPLS-ის მდებარეობა OSI მოდელში

1.2.2. MPLS-VPN-ის ერთრანგიანი მოდელი

ერთრანგიან VPN-მოდელში სერვის პროვაიდერის მარშრუტიზატორები მომხმარებლის ინფორმაციას ატარებენ თავიანთ ქსელში, თუმცა ისინი ასევე მონაწილეობენ მომხმარებლის მარშრუტიზაციაშიც. სხვა სიტყვებით რომ ვთქვათ, სერვის პროვაიდერის მარშრუტიზატორები ამყარებენ მესამე დონის (Layer 3) პირდაპირ კავშირს მომხმარებლის მარშრუტიზატორებთან. შედეგად სერვის პროვაიდერისა და მომხმარებლის მარშრუტიზატორებს შორის მყარდება ურთიერთკავშირი რომელიმე დინამიური მარშრუტიზაციის პროტოკოლის გამოყენებით (ნახ. 11).



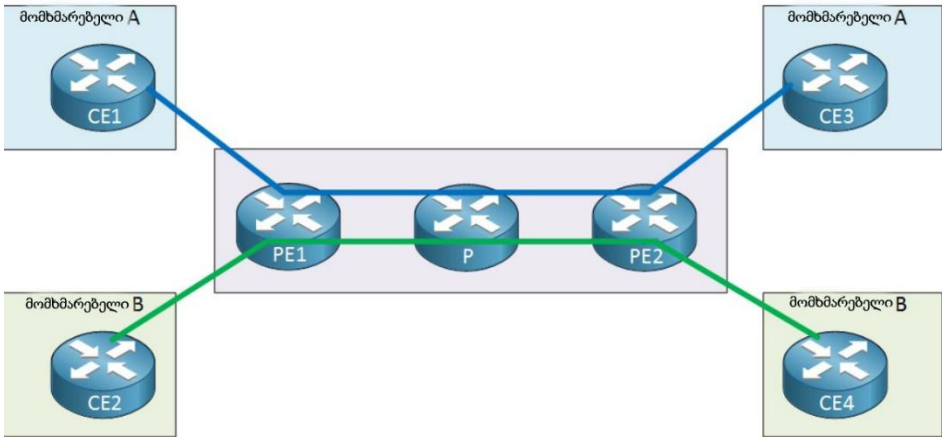
ნახ. 11. MPLS-VPN სერვისი

მომხმარებლის მარშრუტიზატორი MPLS-VPN-სერვისით, რომელსაც მომხმარებლის მოსაზღვრე მარშრუტიზატორსაც (CE) უწოდებენ, IP-შრეზე

ამყარებს კავშირს სერვის პროვაიდერის მინიმუმ ერთ მარშრუტიზატორთან, რომელსაც პროვაიდერის მოსაზღვრე მარშრუტიზატორს (PE) უწოდებენ.

MPLS-VPN-ქსელში მონაცემების კონფიდენციალურობა მიიღწევა ვირტუალური მარშრუტიზაცია-გადამისამართებისა (VRF) და ქსელში მონაცემების გადაცემით მონიშნული პაკეტების სახით.

VRF უზრუნველყოფს სხვადასხვა მომხმარებლების მარშრუტიზაციის შესახებ ინფორმაციის განცალკევებას, ხოლო MPLS-ქსელში - პაკეტების გადაცემას ნიშნულებით და არა IP-თავსართში არსებული ინფორმაციით. მომხმარებლის დამატების შემთხვევაში საკმარისია PE-მარშრუტიზატორზე მხოლოდ CE-მარშრუტიზატორთან კავშირის დამატება, რაც მარტივი და მოსახერხებელია (ნახ. 12) [21].



ნახ. 12. ვირტუალური მარშრუტიზაცია (VRF) MPLS-VPN ქსელში

1.3. ტრაფიკის ოპტიმალური მოძრაობა

იმის გამო, რომ ATM-ისა და Frame Relay-ის კომუტატორები მხოლოდ მეორე შრის (Layer 2) მოწყობილობებია, ამიტომ შესაბამისი პროტოკოლებით ერთმანეთთან დაკავშირებულ მარშრუტიზატორებთან კავშირისთვის საჭიროა ვირტუალური წრედები (Virtual Circuits).

მარშრუტიზატორებისთვის ერთმანეთთან პირდაპირი ტრაფიკის გაგზავნის დროს საჭიროა მათთვის გამოყოფილი ვირტუალური წრის შექმნა. ყველა სამომხმარებლო მოწყობილობის ერთმანეთთან პირდაპირი კავშირის საჭიროების შემთხვევაში აუცილებელი ხდება ე. წ. full mesh ტოპოლოგიის შექმნა ვირტუალური წრეებით, რაც მოუხერხებელი და ძვირია. MPLS-VPN-ის შემთხვევაში კი ტრაფიკი შესაბამის CE-მარშრუტიზატორებს შორის იმოძრავებს ოპტიმალურად.

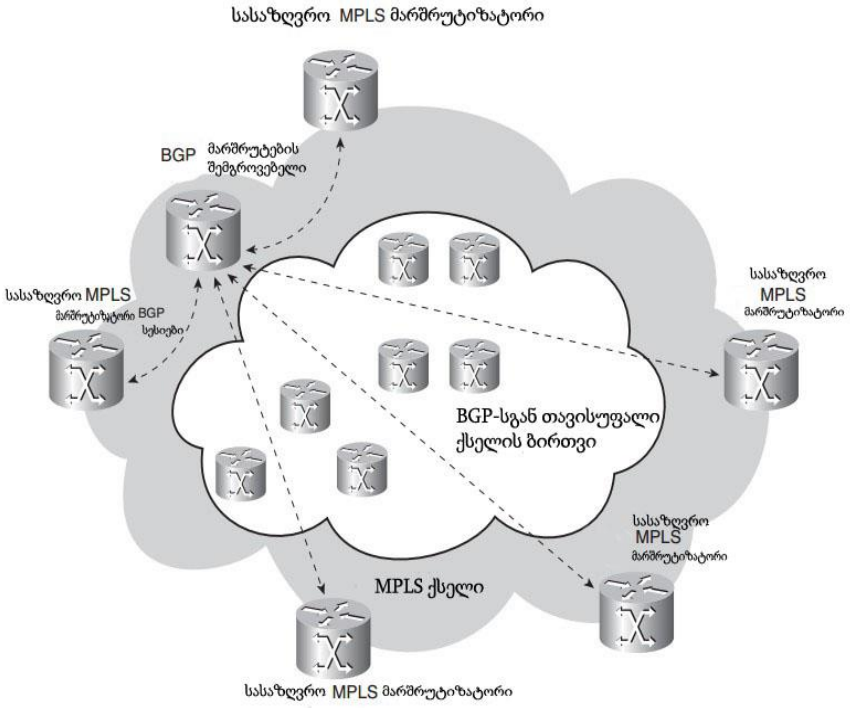
1.2.1. BGP-ისგან თავისუფალი ქსელის ბირთვი

როდესაც სერვის პროვაიდერის IP-ქსელში საჭიროა ტრაფიკის გადამისამართება, მაშინ ყველა მარშრუტიზატორმა უნდა გადაამოწმოს პაკეტში მიმღების IP-მისამართი და იგი შეიტანოს მარშრუტიზაციის თითოეულ ცხრილში. სასაზღვრო ქსელთაშორისი პროტოკოლის (BGP) საშუალებით გადაიცემა მომხმარებლების IP და ინტერნეტის პრეფიქსები, რისთვისაც სერვის პროვაიდერის ყველა მარშრუტიზატორზე გააქტიურებული უნდა იყოს BGP-პროცესი.

როგორც აღნიშნული იყო, MPLS იძლევა პაკეტების გადამისამართების საშუალებას არა IP-მისამართების, არამედ ნიშნულების მიხედვით. MPLS-ქსელში პაკეტის მიმღების IP-მისამართის ცოდნა არაა საჭირო. საჭიროა მხოლოდ გამავალ მარშრუტიზატორთან ასოცირებული შესაბამისი ნიშნულის ცოდნა. ნიშნული არის პაკეტზე მიბმული ინფორმაცია, რომლითაც შუამავალი მარშრუტიზატორები განსაზღვრავენ იმას, თუ რომელ გამავალ მოსაზღვრე მარშრუტიზატორთან უნდა გადამისამართდეს პაკეტი. ქსელის ბირთვის მარშრუტიზატორებს პაკეტის გადამისამართებისთვის აღარ ესაჭიროებათ ინფორმაცია მიმღების IP

მისამართის შესახებ და, შესაბამისად, სერვის პროვაიდერის ქსელის ბირთვის მარშრუტიზატორებს - BGP-ს გააქტიურება.

MPLS-ქსელში ჩართულ მოსაზღვრე მარშრუტიზატორებს ჭირდებათ პაკეტში დანიშნულების IP-მისამართის შემოწმება და, შესაბამისად, BGP-სერვისის გააქტიურება. თითოეულ BGP-პრეფიქსს შემავალ MPLS-მარშრუტიზატორზე აქვს მასთან ასოცირებული ე. წ. BGP next-hop IP-მისამართი და იგი წარმოადგენს გამავალი MPLS-მარშრუტიზატორის მისამართს. რადგანაც ქსელის ბირთვის ყველა მარშრუტიზატორი პაკეტებს გადასცემს MPLS-ნიშნულების მიხედვით, ამიტომ ყველა მოსაზღვრე მარშრუტიზატორის BGP next-hop IP-მისამართი უნდა ქონდეს ქსელის ბირთვის ყველა მარშრუტიზატორს. ამის უზრუნველყოფად კი შეუძლია შიდა კარიბჭის მარშრუტიზაციის ნებისმიერ პროტოკოლს, კერძოდ OSPF-ს ან ISIS-ს (ნახ. 13) [22].



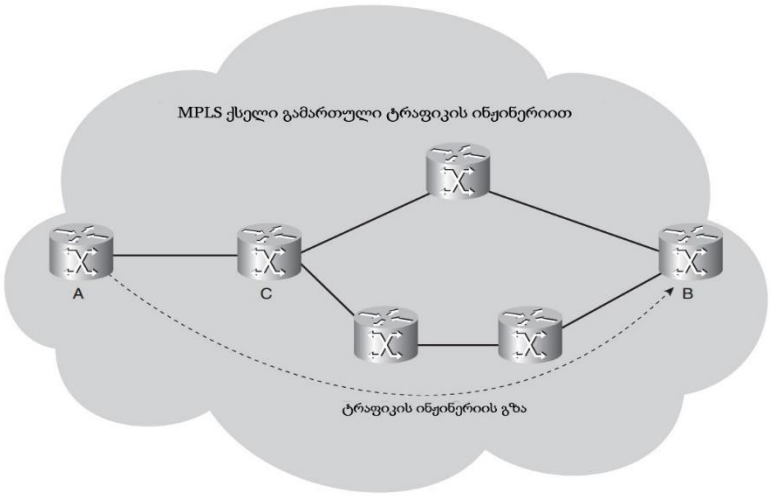
ნახ. 13. BGP-ისგან თავისუფალი ქსელის ბირთვი

1.3.1. ტრაფიკის ინჟინერია (TE)

როგორც ცნობილია, ტრაფიკის ინჟინერიის ძირითადი არსია ქსელის ინფრასტრუქტურის ოპტიმალურად გამოყენების უზრუნველყოფა მონაცემთა გადაცემის შედარებით დაუტვირთავი არხების ჩათვლით. ეს იმას ნიშნავს, რომ ტრაფიკის ინჟინერია ქსელში ტრაფიკის ისეთი მარშრუტით მოძრაობის შესაძლებლობას იძლევა, რომელიც IP-მარშრუტიზაციის მიხედვით ყველაზე ნაკლებად ღირებულია, ანუ ისეთი მარშრუტით, რომელიც დინამიური შიდა მარშრუტიზაციის პროტოკოლით განსაზღვრული მონაცემების მიხედვით ყველაზე ნაკლებადაა პრიორიტეტული.

MPLS-ქსელში ტრაფიკის ინჟინერია იძლევა A წერტილიდან B წერტილამდე ტრაფიკის სასურველი გზით მოძრაობის საშუალებას, ანუ გარკვეულმა ტრაფიკმა შეიძლება იმოძრაოს ნაკლებად დატვირთული ან საერთოდ დაუტვირთავი მაგისტრალით, რაც ქსელის ადმინისტრატორს აძლევს ქსელის არსებული რესურსების ოპტიმალურად გამოყენების შესაძლებლობას. ნახ. 14-ზე მოცემულია მარშრუტის სქემა A-დან B მარშრუტიზატორამდე. აღნიშნული ქსელი მხოლოდ IP-ქსელი რომ იყოს, მაშინ A მარშრუტიზატორზე განხორციელებული ვერანაირი ცვლილებით ვერ ვაიძულებთ C მარშრუტიზატორს განახორციელოს ნახ. 14-ზე მოცემული მარშრუტით ტრაფიკის გაგზავნა, რადგან C მარშრუტიზატორი მხოლოდ თავად წყვეტს, თუ რომელი გზა აირჩიოს. MPLS-TE-ს შემთხვევაში კი იქმნება იმის შესაძლებლობა, რომ A მარშრუტიზატორმა B მარშრუტიზატორის მიმართულებით აირჩიოს ნახაზზე ნაჩვენები გზა. MPLS-TE აიძულებს C მარშრუტიზატორს, რომ A და B მარშრუტიზატორებს შორის ტრაფიკი ატაროს ქვედა, ნაკლებად პრიორიტეტული გზით. ეს სრულდება MPLS-ნიშნულებით გადამისამართების მექანიზმის მეშვეობით. ტრაფიკის ინჟინერიის გზის საწყისი მარშრუტიზატორი (მოცემულ შემთხვევაში A) არის მარშრუტიზატორი, რომელიც უთითებს იმ სრულ გზას, რომლითაც

ტრაფიკმა უნდა იმოძრაოს MPLS-ქსელში. ხშირად ტრაფიკის ინჟინერიას ასევე მოიხსენიებენ როგორც წყაროზე დაფუძნებულ მარშრუტიზაციას (Source-based Routing), რადგანაც, როგორც ცნობილია, საწყისი მარშრუტიზატორი სრული გზის განმსაზღვრელია. საწყისი მარშრუტიზატორის მიერ პაკეტისადმი წინ დართული მონიშვნა განსაზღვრავს სრულ გზას დანიშნულების მისამართამდე და ამიტომ ვერცერთი შუამავალი მარშრუტიზატორი ვერ შეძლებს რომელიმე სხვა გზით ტრაფიკის გაგზავნას [23, 24].



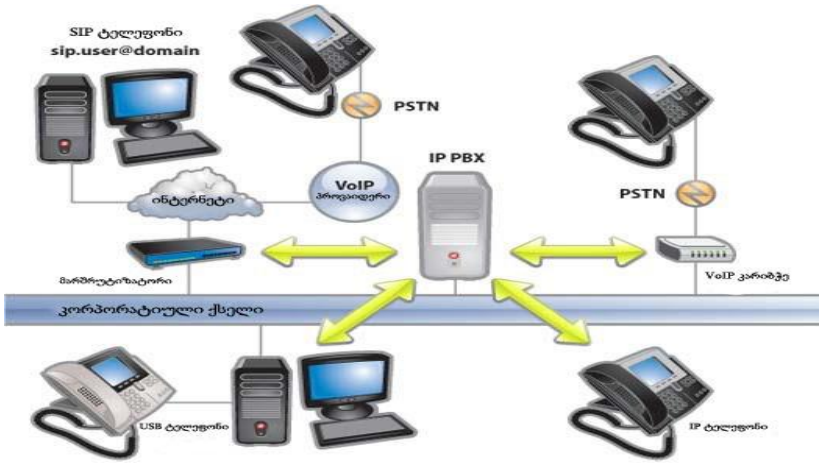
ნახ. 14. MPLS ქსელი გამართული ტრაფიკის ინჟინერიით

თავი 2. VoIP-ქსელები და MPLS-ის უსაფრთხოების კომპონენტები

MPLS-ში უსაფრთხოების უზრუნველყოფა ხდება "ბუნდოვანებით", ანუ MPLS-ის გარემოში მომხმარებლის ტრაფიკი მიუწვდომელია ინტერნეტის სხვა მომხმარებლებისთვის. ეს იმას ნიშნავს, რომ მომხმარებელს ეძლევა კერძო LAN-ქსელის შექმნის შესაძლებლობა, რაც უზრუნველყოფს მისი ტრაფიკის უსაფრთხო გავრცელებას ინტერნეტ სერვის-პროვაიდერის (ISP-ის) საჯარო ქსელში. ამის გამო პოტენციური თავდამსხმელები ვერ მიიღებენ ინფორმაციას მომხმარებლის შესახებ (ე. წ. კრიტიკულ ინფორმაციას), რათა განახორციელონ ისეთი თავდასხმები, როგორიცაა განაწილებული შეტევა მომსახურების დაბლოკვის მიზნით (DdoS). ამასთანავე, MPLS-ის უსაფრთხოება დამოკიდებულია პროვაიდერის კიდის (EDGE) და ძირითადი (CORE) ქსელების უსაფრთხოებაზე. სერვის-პროვაიდერები ინტერნეტიდან წვდომების შეზღუდვისა და გაფილტვრის მიზნით თავიანთ მარშრუტიზატორებზე იყენებენ ისეთ მეთოდებს, როგორიცაა პაკეტის ფილტრაცია და „დაშვების კონტროლის სიები“ (ACL).

VoIP არის ხმისა და მასთან დაკავშირებული სამოსამსახურო ინფორმაციის მარშრუტიზაცია ციფრულ ფორმატში ინტერნეტ პროტოკოლის (IP) გამოყენებით. VoIP-ქსელი არ არის შეზღუდული მხოლოდ ხმოვანი კომუნიკაციით, არამედ მისი საშუალებით შესაძლებელია, აგრეთვე, როგორც ვიდეო ზარების განხორციელება (ვიდეოკონფერენცია), ასევე სხვა ტიპის მონაცემთა გადაცემა საკონფერენციო კავშირისთვის. ის მნიშვნელოვანი ტექნოლოგიაა, რადგან უზრუნველყოფს ადამიანთა ერთმანეთთან კომუნიკაციის საშუალებების ნაირფეროვნებას. გარდა ამისა, რეალურ დროში კომუნიკაციისთვის ერთდროულად შეიძლება იქნეს გამოყენებული როგორც IP, ასევე კომპიუტერზე ჩაწერილი პროგრამული, ვიდეო, მობილური და უკაბელო ტელეფონები. VoIP-ს შეუძლია მონაცემთა გადაცემისთვის სხვადასხვა გადამცემი ტექნოლოგიების გამოყენება და VoIP- ზარების ინიცირება და მიღება როგორც უკაბელო, ასევე მობილური,

საკაბელო და PSTN-ქსელების საშუალებით მათი შესაბამისი არქიტექტურისა და სისტემების გამოყენებით. ეს სისტემები შეიძლება შეიცავდეს ისეთ მოწყობილობებს, როგორიცაა მოდემები, VoIP-მომხმარებლის აპლიკაციები მობილური ტელეფონებზე, VoIP-კარიბჭეები, IP-ტელეფონები, IP-PBX და კომპიუტერები. აქედან გამომდინარე, შეიძლება ითქვას, რომ თავისი გამოყენებადობისა და მოქნილობის წყალობით VoIP-ს შეუძლია სხვადასხვა ქსელების დაახლოება და კონვერგენცია, ანუ ჰიბრიდული ქსელის შექმნა (ნახ. 15) [25, 26].

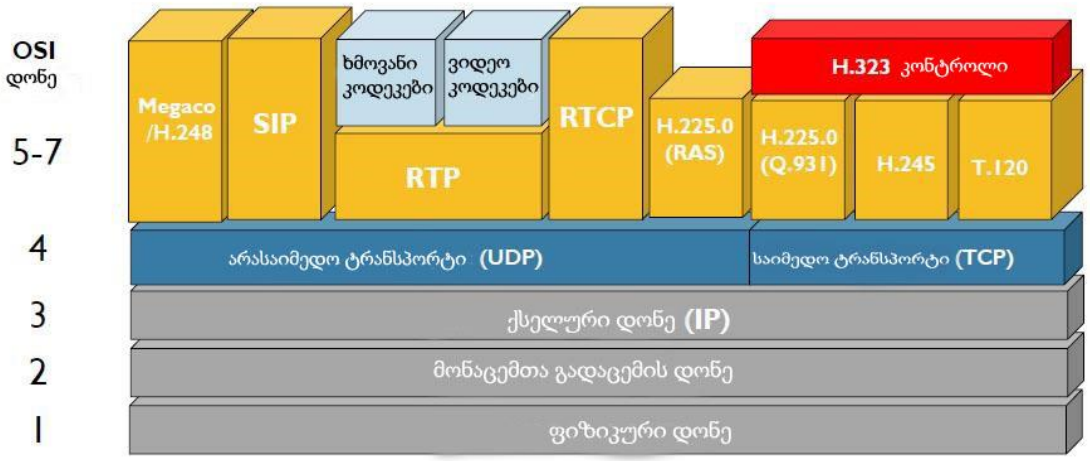


ნახ. 15. ჰიბრიდული VoIP ქსელი

2.1. VoIP-ის არქიტექტურა

VoIP-ის მომხმარებლებისთვის VoIP-ტექნოლოგია შეიძლება განვიხილოთ ინტერნეტ-კომუნიკაციის მეგზურის როლში, ვინაიდან ის მომხმარებლებს აძლევს მომსახურების სხვადასხვა სახეობებისადმი წვდომის საშუალებას. VoIP-ქსელის დონეზე იყენებს ინტერნეტ პროტოკოლს (IP) ტრანსპორტის დონეზე და ისეთ პროტოკოლებს, როგორიცაა გადამცემი საკონტროლო (TCP), სამომხმარებლო დატაგრამებისა (UDP) და ნაკადის კონტროლის გადაცემის პროტოკოლები (SCTP).

აპლიკაციის დონეზე იგი იყენებს სპეციალისტებისთვის კარგად ნაცნობ პროტოკოლებს - DNS-ს, DHCP-ს და ორ ან ორზე მეტ მომხმარებელს შორის ხმისა და/ან მულტიმედიური ტრაფიკის გაცვლისთვის გამოყენებად სასიგნალო პროტოკოლებს (H.323, SIP, MGCP, RTP, SRTP და ZRTP). ნახ. 16-ზე წარმოდგენილია OSI-ის დონეებისა და VoIP-პროტოკოლების შესაბამისობის სქემა.

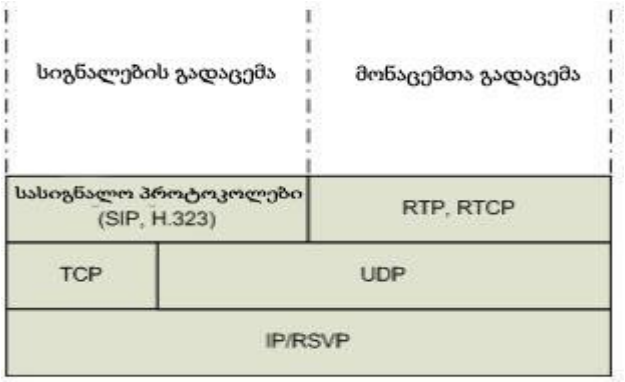


ნახ. 16. VoIP -არქიტექტურა

2.1.1. VoIP-პროტოკოლების აღწერა

როგორც მონაცემთა გადაცემის ქსელების, ასევე ინტერნეტის მომხმარებელთა შორის კომუნიკაციის უზრუნველყოფა ხდება მრავალი პროტოკოლისა და წინასწარ განსაზღვრულ წესების საფუძველზე. ნახ. 17-ის საშუალებით შეიძლება საინფორმაციო და მონაცემთა გადაცემის სიგნალებისა და შესაბამის ძირითად VoIP-პროტოკოლებს შორის ურთიერთკავშირის გარკვევა. პროტოკოლები TCP და HTTP შეიქმნა IP-პროტოკოლისათვის დამატებითი ფუნქციების მისანიჭებლად, რაც ბუნერივად საკმარისი აღმოჩნდა IP-თვის. მაშასადამე, ხმის გადაცემა ინტერნეტ პროტოკოლის საშუალებით (VoIP) მრავალი პროტოკოლით უზრუნველყოფილი საბოლოო პროდუქტია. ამრიგად, VoIP თავის თავში არ

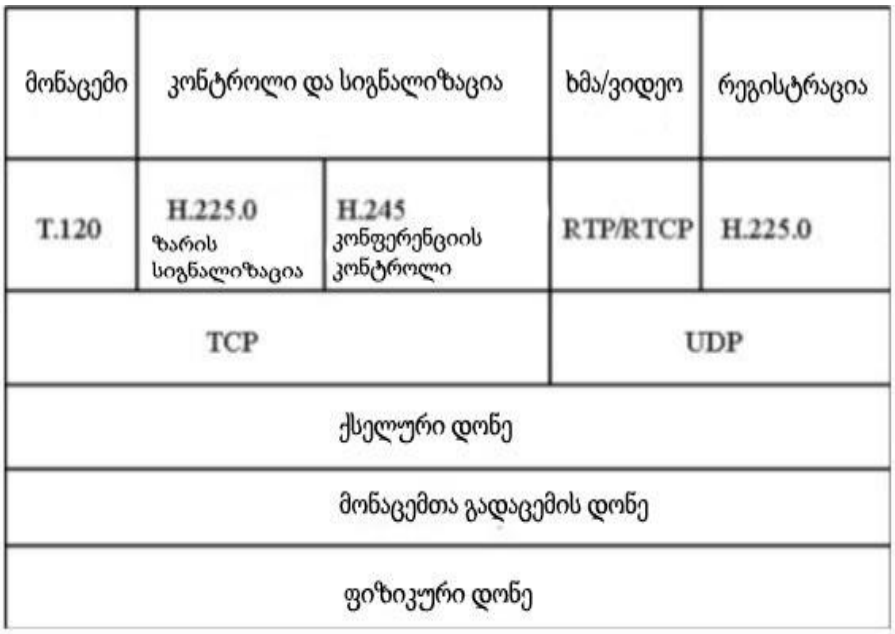
გულისხმობს ოქმს, სერვისს ან მოწყობილობას, არამედ ის პროტოკოლების კომპლექტია, რომელიც უზრუნველყოფს და შესაძლებელს ხდის ტელეფონიის სერვისის ჩართვას მონაცემთა გადაცემის ქსელების ინფრასტრუქტურაში.



ნახ. 17. VoIP- ის ძირითადი პროტოკოლები

ზოგადად, როგორც VoIP, ასევე IP-ტელეფონია შედგება ორი - სასიგნალო და მედია სასაუბრო ფაზისგან. სასიგნალო ფაზის პროტოკოლად დღეისთვის არსებული VoIP-სისტემები იყენებენ ორი სტანდარტიდან ერთერთს - H.323 ან SIP. სასაუბრო მედია ფაზაში ციფრული ხმოვანი პაკეტების ტრანსპორტირებისთვის ძირითადად გამოიყენება RTP პროტოკოლი.

H.323 წარმოადგენს საერთაშორისო სატელეკომუნიკაციო კავშირის მიერ რეკომენდირებული პროტოკოლების ნაკრებს, რომელიც შეიცავს ზარის კონფიგურაციის, მისი შეწყვეტის, აუთენთიფიკაციისა და რეგისტრაციის პროტოკოლებს. H.323 ფართოდ გამოიყენება კორპორატიულ საწარმოებში, რადგან ის ორობითი კოდზე დაფუძნებული პროტოკოლია, რომელიც ადვილად შეიძლება ინტეგრირდეს PSTN-ქსელთან. ნახ. 18-ზე ნაჩვენებია სიგნალის სახეობებსა და შესაბამის პროტოკოლებს (T.120, H.225.0, H.245, RTP/RTCP, H225.0) შორის ურთიერთკავშირი.



ნახ. 18. H.323 ძირითადი პროტოკოლები

სესიის ინიცირების პროტოკოლი (SIP) ინტერნეტის საინჟინრო სამუშაო ჯგუფის (IETF) მიერ და ტექსტზე დაფუძნებული OSI მოდელის მეშვიდე დონის გამოყენებით (Application layer) შექმნილი პროტოკოლია. SIP-პროტოკოლი თავსებადია ტრანსპორტის დონის UDP და TCP პროტოკოლებთან. SIP-ქსელის არქიტექტურა განსხვავდება H.323 ქსელის არქიტექტურისგან. SIP კლიენტ-სერვერული ტიპის პროტოკოლია, სადაც კავშირის დამყარების მოთხოვნის გაცემა ხდება მომხმარებლის მიერ, რომელსაც პასუხი მიეწოდება სერვერიდან. SIP-კლიენტები, როგორც წესი, იყენებენ TCP-სა და UDP-პორტებს, რომელთა ნომრებია 5060 და 5061. SIP- ის პაკეტების ტრანსპორტირებისას დაუშიფრავი სასიგნალო ტრაფიკის გადასაცემად გამოიყენება პორტი 5060, ხოლო დაშიფრული ტრაფიკისთვის - პორტი 5061. RTP-პროტოკოლი გამოიყენება ხმის მონაცემების რეალურ დროში გადასაცემად, ხოლო სესიის აღწერის პროტოკოლი (SDP) - მონაწილე მხარეთა მედიაკომუნიკაციის პარამეტრების (კოდირება, პორტები და ა.შ.) აღსაწერად [27, 28].

2.1.2. SIP-ის კომპონენტები

SIP-ის ინფრასტრუქტურის სერვერული კომპონენტები შეიძლება დაიყოს ოთხ ჯგუფად:

1. **Proxy-სერვერი (Proxy Server)**: პროქსი სერვერი არის შუალედური მოწყობილობა, რომელიც იღებს SIP მოთხოვნებს და ამისამართებს მათ. იგი მუშაობს როგორც კლიენტ-სერვერი და უზრუნველყოფს მომხმარებლებს შორის ზარის განხორციელების შესაძლებლობას. არსებობს ორი სახის Proxy სერვერი:

1.1. ე. წ. **Statefull Proxy**, რომელიც მოთხოვნების დამუშავების პროცესში ინახავს ტრანზაქციების მდგომარეობას და რომელსაც შეუძლია ხელახალი მოთხოვნის გაშვება იმ შემთხვევაში, როდესაც მეორე მხარისგან არ მოდის პასუხი. მის მიერ უზრუნველყოფილი სერვისებია: ზარის გადამისამართება, დაკავება და ა.შ.;

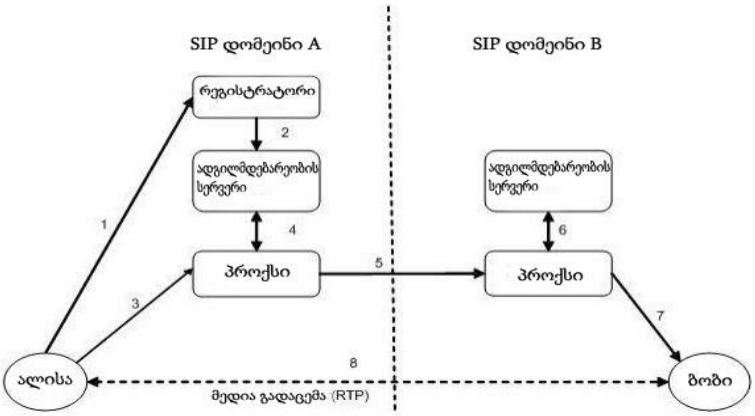
1.2. **Stateless Proxy**, რომელიც მოთხოვნების შესრულების პროცესში კი არ ინახავს ტრანზაქციის მდგომარეობას, არამედ იგი მხოლოდ აგზავნის შეტყობინებებს.

2. **მარეგისტრირებელი სერვერი (Registrar Server):** ეს სერვერი იღებს მოთხოვნებს მომხმარებელთა რეგისტრაციის შესახებ და ინახავს შესაბამის ინფორმაციას იმ მიზნით, რომ მომხმარებელს შესთავაზოს ლოკალიზაციის და მისამართის სერვისი მის მიერ კონტროლირებად არეალში. იგი ასევე უზრუნველყოფს აუთენთიფიკაციას. SIP-მომხმარებელი უნდა დარეგისტრირდეს მარეგისტრირებელ სერვერზე ყოველი ხელახალი ჩართვისას.

3. **გადამამისამართებელი სერვერი (Redirect Server):** ეს არის სერვერის ტიპი, რომელიც შემდეგი სერვერებისაკენ ამისამართებს შემოსულ მოთხოვნებს და მისი ფუნქციაა, აგრეთვე, მომთხოვნს დაუბრუნოს შემდეგი სერვერის მისამართი.

4. **ადგილმდებარეობის სერვერები (Location Server)**: უზრუნველყოფენ ინფორმაციის მიწოდებას ქსელში (ინტერნეტში ან SIP სისტემაში) არსებული

რესურსების ადგილმდებარეობის შესახებ. ადგილმდებარეობის სერვერები, მონაცემთა ბაზების მსგავსად, საჭირო მისამართებით უზრუნველყოფენ თითოეულ დარეგისტრირებულ მომხმარებელს და მომხმარებელთა მობილობას საკომუნიკაციო სისტემის არეალში. SIP-სისტემაში ადგილმდებარეობის სერვერის მონაცემთა ბაზის განახლება ხდება SIP-მომხმარებლის აგენტების რეგისტრაციების შედეგად, რომელიც აიხსნება ნახ. 19-ზე წარმოდგენილი სქემის საშუალებით [29, 30, 31].



ნახ. 19. SIP-ის არქიტექტურა (SIP- ის ურთიერთქმედება)

2.1.3. SIP-ის პროტოკოლის ლოგიკური კომპონენტები და მისი უპირატესობები

SIP-ის პროტოკოლის კომპონენტებია SIP-სერვერების მომხმარებელი აგენტი (UAS), მომხმარებელი აგენტი კლიენტი (UAC) და მომხმარებელი აგენტი (UA). მომხმარებელი აგენტი არის აპლიკაცია, რომელსაც შეუძლია SIP-სესიის დაწყება, მიღება და შეწყვეტა. იგი ასევე შეიძლება განვიხილოთ როგორც პროგრამული ტელეფონი, რომელიც დაინსტალირებულია მომხმარებლის კომპიუტერში ან როგორც ფიზიკური ტელეფონი, რომელსაც შეუძლია VoIP-სერვისის მომსახურება. UAC გამოიყენება SIP-მოთხოვნის

ინიცირებისთვის, ხოლო UAS - მოთხოვნის მიღებისა და მომხმარებლის სახელით პასუხების დაბრუნებისთვის.

სატელეფონო საკომუნიკაციო სერვის პროვაიდერებს შეუძლიათ SIP-ის პროტოკოლის გამოყენებით ერთმანეთთან დააკავშირონ საკაბელო სატელეფონო და IP-ტელეფონიის ქსელები. HTTP-ის პროტოკოლში, მასში SIP-ის პროტოკოლის საფუძვლების არსებობის გამო, გამარტივდა ხმაზე დაფუძნებული ვებ-აპლიკაციების შექმნა და ინტეგრირება, რის გამოც ეს პროტოკოლი გამოიყენება ქსელებში Google, facebook, WebRTC და სხვ.

SIP ტექსტური პროტოკოლია და იგი ადვილად გაფართოვებადია, ახასიათებს მობილურობა, ადვილია მისი დანერგვა და დაპროგრამება, მანამდე დამუშავებულ პროტოკოლებთან შედარებით მოითხოვს ექსპლუატაციაში გაშვების მცირე დროს. SIP იყენებს ერთრანგიან დეცენტრალიზირებულ P2P-პროტოკოლს და არ მოითხოვს ქსელის დონეზე რაიმეს იმპლემენტაციას. SIP შეიძლება გამოყენებულ იქნას მიმდინარე სესიის მოდიფიცირებისთვის მისი საშუალებით ჩვეულებრივი სატელეფონო კავშირის პროცესი შესაძლებელია გარდაიქმნას მრავალმონაწილიან ვიდეოკონფერენციად. ამ შემთხვევაში გარეშე მომხმარებლებს შეუძლიათ მდებარეობის გაუთვალისწინებლად შეუერთდნენ სესიას.

SIP-ის მობილურობა აიხსნება კომპანიის თანამშრომელთა მობილურ ჩართულობას კომპანიაში არსებულ საკომუნიკაციო სისტემებთან ამ თანამშრომლების კომპანიისაგან მოშორებით ყოფნისას, რომელთა მოწყობილობების დაკავშირება მოქნილი და მოსახერხებელი VoIP აპლიკაციების მეშვეობით,

SIP-პროტოკოლის გამოყენებით შესაძლებელია კომპანიის შიდა IP-PBX ან პროვაიდერის მიერ მიწოდებული სერვისის საშუალებით მძლავრი კომპლექსური სატელეფონო სისტემის დანერგვა, რომელიც მორგებული იქნება კომპანიის მოთხოვნებზე. ასევე ამავე სისტემაში შესაძლებელია IVR-ფუნქციის გამოყენება, რომელიც წარმოადგენს სატელეფონო ზარების

ავტომატურ რეჟიმში დამუშავებისა და მომხმარებლებთან ინტერაქციის ტექნოლოგიას. ინტერაქტიული სატელეფონო სისტემები დღეს ხშირად გამოიყენება კომპანიების მიერ მომსახურების დონის ამაღლებისა და ამავდროულად მომსახურების ღირებულების შემცირების მიზნით. აუდიო ჩანაწერებისა და სცენარების საშუალებით, რომლებიც ინტერაქტიული სატელეფონო სისტემების მთავარი შემადგენელი ნაწილია, მომხმარებლებს შეუძლიათ მიიღონ სასურველი ინფორმაცია ან ჩაატარონ რაიმე ოპერაცია ტელეფონის ღილაკების მეშვეობით.

SIP-ის პლატფორმების ინტეგრირება შეიძლება ავტომატიზაციის, CRM, მომხმარებლის მონაცემთა ბაზების მონიტორინგისა და ანალიზის და გაყიდვების განყოფილებებში ზარებისა და შეკვეთების დაფიქსირების სისტემებთან. SIP-პროტოკოლის ერთ-ერთი მთავარი უპირატესობაა კორპორატიულ პროგრამულ უზრუნველყოფაში სატელეფონი სისტემის ინტეგრაციის შესაძლებლობა [32].

2.2. SIP-ზე დაფუძნებული VoIP-ქსელების უსაფრთხოების ხარვეზები

ყველა საკომუნიკაციო ქსელში უსაფრთხოება ქსელის მნიშვნელოვან ნაწილად ითვლება, რადგან მომხმარებლებს ან თანამშრომლებს შორის საუბარი მოითხოვს კერძოობას, მთლიანობას, უსაფრთხოებასა და კონფიდენციალობას. უსაფრთხოების რისკები SIP-ზე დაფუძნებულ VoIP- ში მოიცავს არა მარტო SIP-პროტოკოლში არსებულ ხარვეზებს, არამედ ოპერაციული სისტემებს, აპლიკაციებს და სხვა პროტოკოლებს. უსაფრთხოების რისკები შეიძლება განისაზღვროს როგორც "რაღაც ან ვინმე, რომელმაც სავარაუდოდ შეიძლება გამოიწვიოს საფრთხე ან სირთულე".

SIP-ზე დაფუძნებული VoIP-ის კომპლექსურობა ქმნის უსაფრთხოების დიდი რაოდენობის რისკებს, რომლებიც გავლენას ახდენენ საინფორმაციო უსაფრთხოების შემდეგი სამი ძირითადი მიმართულებით: კონფიდენციალობა, ხელმისაწვდომობა და მთლიანობა. ინტერნეტ-

უსაფრთხოებისა და საფრთხეების ანგარიშები აჩვენებს, რომ SIP-ზე დაფუძნებულ VoIP-ზე შეტევები იზრდება. ძირითადად ამის მიზეზია ის, რომ სულ უფრო მეტი ადამიანი ან ორგანიზაცია ერთვება SIP-ზე დაფუძნებულ VoIP-მომსახურებაში და, აგრეთვე, ისიც, რომ მავნე ჰაკერები და ე. წ. ინტერნეტ-კრიმინალები გააქტიურდნენ VoIP ქსელების თავდასხმის მიზნით. ძირითადად თავდამსხმელები სამიზნედ ირჩევენ მცირე ზომის კომპანიებს მათი უსაფრთხოების ალგორითმების სისუსტის გამო, რადგან SIP-პროტოკოლი თავისთავად არ ითვალისწინებს შიფრაციას. ამიტომ ამ დროს მესამე მხარეს შეუძლია შეაგროვოს მომხმარებლის მონაცემები მათი რეგისტრატორ-სერვერისთვის მიწოდებისას. შედეგად, მესამე მხარეს შეუძლია განახორციელოს ძვირადღირებული საქალაქთაშორისო ზარები ან თუნდაც მომხმარებლის მოპარული მონაცემებით შეიქმნას კარიბჭე და წუთობრივად გაყიდოს ძვირადღირებული მარშრუტების სატელეფონო მომსახურება. SIP-პროტოკოლის უსაფრთხოების რისკები მოიცავს შემდეგ ქმედებებს: მიყურადება, არასანქცირებული წვდომა, შეტყობინებების გაყალბება, გამეორება, 5060 პორტზე თავდასხმა, ე. წ. „შუა ადამიანი“ (Man-in-the-middle), რეგისტრაციის გაუქმება (გაყალბება), DDoS და თავდასხმა ბილინგის სისტემაზე. თუ თავდამსხმელი ქსელის რომელიმე კვანძთან მოახერხებს წვდომას, მას ასევე ექნება წვდომა IP-პაკეტებზე, რომლებიც ამ კვანძში მოძრაობენ. არსებობს უამრავი ქსელის უფასო ანალიზატორი და პაკეტების ჩამწერი ინსტრუმენტები, რომლებსაც შეუძლიათ VoIP-ტრაფიკის Wave (ხმოვანი ჩანაწერის ფაილის გაფართოება) ფაილად დაკონვერტირება. SIP-ზე დაფუძნებული ქსელები დაუცველია მიყურადებისგან და ისინი მარტივად არიან ხელმისაწვდომი IP-ქსელების გამოყენების გამო.

არასანქცირებული წვდომა ნიშნავს იმას, რომ თავდამსხმელს აქვს წვდომა ქსელში არსებულ რესურსებზე მაშინ, როდესაც მათ არ გააჩნიათ ამისი უფლებამოსილება. ორგანიზაციებში გამოიყენება სატელეფონო ზარების კონტროლის, ადმინისტრირების, ბილინგისა და ხმის სატელეფონო ფუნქციების სხვა სისტემები, რომლებიც შეიძლება შეიცავდეს პაროლებს,

მომხმარებლის ვინაობას, მისი ტელეფონის ნომრებსა და პირად ინფორმაციებს. ქსელური მარშრუტიზატორები და კონცენტრატორები ხშირად სტანდარტული მწარმოებლის პაროლებით მუშაობენ, რომელთა მუდმივობის (უცვლელობის) შემთხვევაში თავდამსხმელები ადვილად ახერხებენ წვდომას.

შეტყობინებების გაყალბება ხდება მაშინ, როდესაც თავდამსხმელები SIP-შეტყობინებებს ცვლიან SIP-მოწყობილობებს შორის, რაც შეიძლება მოხდეს რეგისტრაციის მიღების ან პროქსის გაყალბების გამო.

შეტყობინებების გამეორების შემთხვევაში მისი სამიზნეა SIP-ტრაფიკი. ამ დროს თავდამსხმელი აკავებს SIP-ტრაფიკს და გარკვეული პერიოდის შემდეგ ხელახლა აგზავნის მას. თავდამსხმელის მიერ განხორციელებულმა ასეთმა ქმედებამ შეიძლება გამოიწვიოს პრობლემები მონაცემთა მთლიანობის თვალსაზრისით. კერძოდ, იმ დროს, როდესაც დაკავებული და გამეორებული ტრაფიკი შეიცავს პირად მონაცემებს, მაშინ არაავტორიზირებულმა მომხმარებლებმა შეიძლება მიიღონ წვდომა VoIP-სერვისებზე.

5060 პორტზე თავდასხმისას თავდამსხმელებმა შეიძლება მარტივად შეუშალონ ხელი SIP-პორტზე გამავალი ტრაფიკის გადაცემას და მოახდინონ მისი მანიპულირება, ვინაიდან იგი გადაიცემა დაუშიფრავად.

თავდასხმის ქმედებისას სახელწოდებით „შუა ადამიანი“ (Man-in-the-middle) თავდამსხმელი მალულად იღებს და ცვლის მოსაუბრე მხარეებს შორის შეტყობინებებს. ამიტომ თავდასხმელს შეუძლია SIP-სიგნალების, შეტყობინებების მანიპულირება და გადამამისამართებელი სერვერის საშუალებით სატელეფონო ზარებით სარგებლობა.

რეგისტრაციის გაუქმება (გაყალბება) ხდება იმ დროს, როდესაც თავდამსხმელი ცვლის მსხვერპლის UA-ს ლეგიტიმურ IP-მისამართს თავისი ყალბი მისამართით, რის შედეგადაც დაზარალებულის ყველა შემომავალი ზარი გადამისამართდება თავდამსხმელისკენ. რეგისტრაცია ძირითადად მოითხოვს მომხმარებლის სახელისა და პაროლის აუთენტიფიკაციას. თუ

ორგანიზაციის შიგნით დაყენებულია სუსტი პაროლები, თავდამსხმელს ადვილად შეუძლია ისარგებლოს მომხმარებლების რეგისტრაციის პაროლებით და მოიპაროს SIP-ზარები. რეგისტრაციის სესია დაუცველია, რადგან ის იყენებს UDP-პროტოკოლს, რომელსაც აქვს უსაფრთხოების სუსტი მექანიზმი. SIP-ის სარეგისტრაციო სერვერს არ შეუძლია პროქსის სერვერის ან UA-ის ლეგიტიმურობის დადგენა.

თავდასხმის ქმედებისას სახელწოდებით „DDoS“ ადგილი აქვს ქსელის ე. წ. "დატბორვას", რაც იმას ნიშნავს, რომ ამ დროს ლეგიტიმური მომხმარებელი ვერ ღებულობს მომსახურებას SIP-ზე დაფუძნებულ VoIP-ქსელში. თავდასხმის ეს ვარიანტი შეიძლება განხორციელდეს სესიის შეტევით SIP-BYE ან CANCEL შეტყობინებების გამოყენებით, რაც იწვევს SIP-სესიების ან მოთხოვნების შეწყვეტას. გამავალი ზარის დროს თავდამსხმელი აგზავნის გაყალბებულ SIP-BYE ან CANCEL შეტყობინებას, რაც, თავის მხრივ, იწვევს ზარის ან მოთხოვნის შეწყვეტას. სესიების შეწყვეტის გამო შეიძლება წარმოშვას DdoS-თავდასხმის შესაძლებლობა, რომლის დროსაც თავდამსხმელი SIP-ქსელს SIP-BYE-ს ან CANCEL-ის შეტყობინებების დიდ რაოდენობას აწვდის. ამის შედეგად SIP-სესიები წყდება ან VoIP-სერვისი საერთოდ ხდება მიუწვდომელი.

ბილინგის სისტემაზე თავდასხმისას სერვისების პროვაიდერებს ეცვლებათ VoIP-ის ქსელის გარეთ გამავალი ზარების ტარიფები, რომლებიც იანგარიშება კავშირის მანძილისა და ხანგრძლივობის მიხედვით. ამ შემთხვევაში თავდამსხმელს შეუძლია განახორციელოს არასანქცირებული ზარები ლეგალური მომხმარებლის ანგარიშის გამოყენებით, რაც გამოიწვევს არასანქცირებულად გაწეული სატელეკომუნიკაციო მომსახურების მის ანგარიშზე დაფიქსირებას და, ამის შედეგად, ლეგალური მომხმარებლისთვის ფინანსური ზარალის მიყენებას. ბილინგის სისტემაზე თავდასხმა შეიძლება განხორციელდეს SIP-შეტყობინებების მანიპულირებით SIP-პროქსის ჩანაწერებზე თავდამსხმელის წვდომისას. გარდა ამისა, თავდამსხმელმა შეიძლება ხელი შეუშალოს ბილინგის პროცესს

SIP-სესიის დაწყების მიზნით გაგზავნილი ACK-შეტყობინების დაბლოკვით, რის შედეგადაც სერვერი SIP-პროქსი ჩათვლის, რომ ზარი არ შედგა და ის არ დაფიქსირდება [33].

2.3. სასიგნალო უსაფრთხოების მიმოხილვა

ღია ტექსტის შესახებ მონაცემის დაშიფვრის მეთოდი და შესაბამისი ტექნიკა გამოიყენება შეტყობინების კონფიდენციალურობისა და კერძოობისთვის. ზოგ შემთხვევაში ის ასევე გვაძლევს ინფორმაციას იმის შესახებ, დაირღვა (ან შეიცვალა) თუ არა მონაცემების მთლიანობა. კონფიდენციალურობის დაცვისთვის HTTP-ისა და SMTP-ს უსაფრთხოების საუკეთესო მეთოდები შეიძლება გამოყენებულ იქნას SIP-ის უსაფრთხოებისთვის. უსაფრთხოების მექანიზმები უზრუნველყოფილია Ipsec, S/MIME და TLS-პროტოკოლების გამოყენებით. სამივე მათგანს აქვს საკუთარი უპირატესობები და ნაკლოვანებები.

პროტოკოლი TLS შეიძლება გამოყენებულ იქნას უსაფრთხოების არქიტექტურაში, როდესაც კავშირის მონაწილეები თავდაპირველად ერთმანეთს არ ენდობიან. იგი გამოიყენება SIP-შეტყობინებების სატრანსპორტო დონეზე და მოითხოვს TCP-კავშირს (TCP-პროტოკოლის გამოყენებას). ამ პროტოკოლის მიხედვით გამოყენებული უნდა იყოს სიმეტრიული კოდირების გასაღები. TLS-ის გამოყენებას SIP-სიგნალის უსაფრთხოების უზრუნველსაყოფად აქვს თავისი შეზღუდვები, კერძოდ, თითოეული პროქს-სერვერი შეტყობინებინების მარშრუტიზაციის სწორი მიმართულებით განხორციელებისთვის საჭიროებს SIP-ის სათაურის ღია ფორმატის ტექსტით წარმოდგენას იმ დროს, როდესაც დაცულია სეგმენტი UAC-დან UAC-მდე. TLS-მექანიზმით კავშირის განხორციელებისას არ არსებობს კომუნიკაციის საბოლოო წერტილამდე მისი გამოყენების რეალური გარანტია. TLS ასევე შეიძლება გამოყენებულ იქნას მომხმარებლების მოლაპარაკების დაშიფვრის გასაღების დასაცავად. თუმცა,

კავშირის ბოლო წერტილამდე დაცვის არარსებობამ შეიძლება გამოიწვიოს გასაღების გამჟღავნება, რადგან ამ დროს SIP-შეტყობინება ქსელის ნებისმიერ სხვა ნაწილში შეიძლება გადაიცეს ღიად (დაუშიფრავად).

მრავალპროფილიანი ინტერნეტის ელექტრონული ფოსტის გაფართოებები (MIME) განსაზღვრავს ელექტრონული ფოსტის სხვადასხვა სახის მონაცემების გაგზავნის მექანიზმს. MIME ასევე შეიძლება გამოყენებულ იქნას სხვა პროგრამების დაცვის უზრუნველსაყოფად. უსაფრთხოების მექანიზმები, რომლებიც გამოიყენება MIME body-ის ელ-ფოსტაში, შეიძლება, აგრეთვე, გამოყენებულ იქნას SIP-ის სხეულის (Body-ის) დაცვის უზრუნველსაყოფად, რადგან კოდირება ხდება MIME-ით. MIME უსაფრთხოების მექანიზმს ეწოდება S/MIME. შესაბამისი პროტოკოლი აუმჯობესებს შეტყობინების უსაფრთხოებას და იგი შეიძლება გამოყენებულ იქნას როგორც უსაფრთხოების მექანიზმი საბოლოო მომხმარებლამდე. ამ შემთხვევაში შეტყობინებების დაშიფვრა ცოტა უფრო სახიფათოა, ვინაიდან იგი მოითხოვს იმას, რომ მიმღების საჯარო გასაღები წინასწარ უნდა იყოს ცნობილი გამგზავნისთვის ან იგი მიღებული უნდა იყოს ცენტრალური საცავიდან. შემდგომ შეტყობინების დაშიფვრა მოხდება მიმღების საჯარო გასაღებით, ხოლო შიფრის მოხსნა - მიმღების კერძო გასაღებით. S/MIME მექანიზმის ყველაზე დიდი ნაკლოვანება საჯარო გასაღებების შესაბამისი ინფრასტრუქტურის სიმცირეა. გადაწყვეტილება საჯარო გასაღებების გაცვლის შესახებ შეიძლება მიღებული იყოს ასევე დაუცველი "Man in middle" თავდასხმების შემთხვევაში, თუ მომხმარებელი თვითნებურად დამოწმებულ სერტიფიკატებს გამოიყენებს.

უსაფრთხოების ინტერნეტ-პროტოკოლი (IPsec) არის IPv4-ქსელის უსაფრთხოების პროტოკოლი, რომელიც ამოწმებს და შიფრავს მონაცემების პაკეტებს. Ipsec-ს შეუძლია დაიცვას მონაცემთა ნაკადები მომხმარებელთა წყვილების ჰოსტებს შორის, ქსელი კარიბჭეებს შორის და ქსელი კარიბჭესა და ჰოსტს შორის. უსაფრთხოების ინტერნეტ-პროტოკოლი კომუნიკაციის დასაცავად იყენებს უსაფრთხოების კრიპტოგრაფიულ სერვისებს. ამ

პროტოკოლს გააჩნია ქსელის დონის ერთრანგიან ქსელთან ავტორიზაციის, მონაცემთა წარმომშობის ავტორიზაციის, მონაცემთა მთლიანობის, მონაცემთა კონფიდენციალურობისა და გამეორებისგან დაცვის შეაძლებლობის მხარდაჭერა. მისი უპირატესობაა ის, რომ იგი UA-აგენტების გარეშე დამყარებულ კავშირებს ახორციელებს ოპერაციული სისტემის დონეზე. Ipsec-პროტოკოლი შეიძლება, აგრეთვე, გამოყენებულ იქნას UDP-პროტოკოლის საფუძველზე დამყარებული კავშირების დასაცავად. VPN-ქსელი ყველაზე მეტადაა შესაფერისი მომხმარებლებს (საკომუნიკაციო მხარეებს) შორის წინასწარ ჩამოყალიბებული სანდო კავშირისთვის Ipsec-პროტოკოლის გამოყენებისას .

მედიისა და აუდიო მონაცემების შეხაბებ ინფორმაციის ტრანსპორტირება რეალურ დროში ხორციელდება მონაცემთა გადაცემის ტრანსპორტის პროტოკოლით (RTP). მისი საშუალებით პაკეტების მიმოცვლა ხდება ღია ტექსტით. ამიტომ უსაფრთხოების, მონაცემების კონფიდენციალობის, შეტყობინებათა მთლიანობის, RPP-სა და RTCP-ზე თავდასხმებისგან დაცვის საკითხების გადასაჭრელად შემუშავდა უსაფრთხო RTP (SRTP)-პროტოკოლი. SRTP-ს მიერ გამოიყენება ორი ტიპის გასაღებები - სესიისა და მასტერ-გასაღებები. კრიპტოგრაფიული კონტექსტი განსაზღვრულია სხვადასხვა გასაღებების მართვის ისეთი მექანიზმებით, როგორიცაა SDES, ZRTP და MIKEY [33, 34, 35, 36, 37].

2.4. VoIP-ის უსაფრთხოების პრობლემების გამომწვევი მიზეზები

იმ დროს, როდესაც ორგანიზაციები განსაკუთრებულ ყურადღებას უთმობენ თავიანთი ინფორმაციის, შესაბამისი ქსელებისა და ინფრასტრუქტურის დაცვისათვის საჭირო ტექნოლოგიის გამოყენებას, ავიწყდებათ, რომ პროცესში ადამიანებიც მონაწილეობენ. კერძოდ, უნდა აღინიშნოს, რომ სატელეკომუნიკაციო უსაფრთხოებაზე შესაძლოა გავლენა

იქონიოს პირის ინფორმაციული უსაფრთხოების ცოდნის ნაკლებობამ, რაც გამოიწვევს ორგანიზაციის ფინანსურ და, აგრეთვე, კონფიდენციალურობისა და ინფორმაციის მთლიანობის დაცვის პრობლემებს. ამიტომ ორგანიზაციის თანამშრომლებს უნდა ჰქონდეთ ცოდნა ინფორმაციული უსაფრთხოებისა და რისკების შესახებ, რათა გაითვალისწინონ უსაფრთხოების ნიუანსები, სიმძიმე და მათი მასშტაბები. ორგანიზაციამ ასევე უნდა უზრუნველყოს უსაფრთხოების პოლიტიკასთან დაკავშირებული მასალების მომზადება და მათი საქმიანობის განმსაზღვრელი შესაბამისი დოკუმენტის გაცნობა თანამშრომლებისთვის. აღნიშნულ დოკუმენტში დაფიქსირებული უნდა იყოს სათანადოდ დადგენილი საჯარიმო სანქციები გადაცდომების შემთხვევაში. ორგანიზაციამ უნდა უზრუნველყოს პროცესებისა და პროცედურების სწორად დოკუმენტირება, მოწყობილობებისა და სისტემების განახლებისა და მათი კონფიგურაციის შეცვლის უსაფრთხოების პოლიტიკა და სხვადასხვა დონის სატელეკომუნიკაციო ინფრასტრუქტურაზე დაშვების პირობები.

კარგად სტრუქტურირებული ქსელის უსაფრთხოების მოდელის შექმნა ძალზე მნიშვნელოვანია უსაფრთხოების განმტკიცებისა და შენარჩუნებისთვის. იგი უზრუნველყოფს ქსელის ინჟინრებისთვის ქსელის შექმნისა და ექსპლუატაციისას უსაფრთხოების მნიშვნელოვანი დეტალების გაცნობას. უსაფრთხოების მოდელი ასევე სასარგებლოა უკვე არსებული ქსელის უსაფრთხოების უზრუნველსაყოფად, პროფილაქტიკური სამუშაოების გრაფიკების შემუშავებისა და ქსელის განვითარების (სრულყოფის) სიცოცხლისუნარიანი მოდელისთვის.

2.5. MPLS-ზე დაფუძნებული VoIP-ქსელის უსაფრთხოების მოდელი

პრობლემის გადაჭრის მიზანია SIP-ზე დაფუძნებული VoIP-ქსელის უსაფრთხოების დაცვა მისი ინტერნეტისგან იზოლაციის საშუალებით. ხმოვანი მომსახურების მიმწოდებლის ქსელის დასასრულს არსებულ SIP-

სერვერებთან მომხმარებლის კავშირი შეიძლება მიღწეულ იქნას VoIP-ის ჩართვით MPLS-ქსელის ინფრასტრუქტურაში. ამ დროს ორგანიზაციაში განთავსებული მომხმარებლის მოწყობილობები ინტერნეტის ქსელიდან იზოლირებულნი არიან, რითაც გრანატირებული ხდება მათი უსაფრთხოება. ეს კი უზრუნველყოფს მომხმარელთა ხმოვანი ტრაფიკის ხელმისაწვდომობას, მთლიანობასა და კონფიდენციალობას.

SIP-ზე დაფუძნებული VoIP წარმოიქმნა როგორც ხმოვანი კომუნიკაციის დე-ფაქტო სტანდარტი. ამის გამო ღია SIP-ზე დაფუძნებული ინტერფეისების მხარდაჭერა სულ უფრო მნიშვნელოვანი ხდება კერძო სატელეფონო საკომუტაციო სისტემით (IP-PBX-ით) აღჭურვილ ორგანიზაციებში. იგი გამოიყენება მომხმარებლებს შორის ინტერაქტიული კომუნიკაციის დაწყებისთვის. მიუხედავად იმისა, რომ იგი პროგრესულად განვითარებადი ტექნოლოგიაა, მაინც უნდა აღინიშნოს მისი ნაკლოვანება, რაც გამოიხატება უკანონო ან არაავტორიზებული გამოყენების გზით ფინანსების მნიშვნელოვანი დანაკარგების საფრთხის არსებობაში, რაც ყველაზე დიდი პრობლემაა ორგანიზაციებისთვის. VoIP-სისტემების უფრო მეტად გავრცელებასთან ერთად უსაფრთხოების გამოწვევები და რისკები იზრდება და ამიტომ უნდა იყოს გატარებული უსაფრთხოების შესაბამისი ზომები, რადგან, როგორც ცნობილია, ნებისმიერ საკომუნიკაციო ქსელში მომხმარებლებსა ან თანამშრომლებს შორის საუბარი მოითხოვს კერძოობას, მთლიანობას, უსაფრთხოებას და კონფიდენციალობას. უსაფრთხოების რისკები SIP-ზე დაფუძნებულ VoIP-ში მოიცავს არა მარტო SIP-პროტოკოლისთვის დამახასიათებელ ხარვეზებს, არამედ ოპერაციულ სისტემებს, აპლიკაციებსა და სხვა პროტოკოლებს. უსაფრთხოების რისკები შეიძლება განისაზღვროს როგორც "რაღაც ან ვინმე, რომელმაც სავარაუდოდ შეიძლება გამოიწვიოს საფრთხე ან სირთულე".

ინტერნეტ-უსაფრთხოებისა და საფრთხეების ანგარიშები აჩვენებს, რომ SIP-ზე დაფუძნებული VoIP-შეტევები იზრდება. ძირითადად ამის მიზეზია ის, რომ, ჯერ ერთი, სულ უფრო მეტი ადამიანი ან ორგანიზაცია

ერთვება SIP-ზე დაფუძნებულ VoIP-მომსახურებაში და, მეორე, ის, რომ მავნე ჰაკერები და ინტერნეტ-კრიმინალები სულ უფრო მეტად ერთვებიან VoIP-ქსელებზე თავდასხმის პროცესში. თავდამსხმელები, ძირითადად, სამიზნედ ირჩევენ მცირე ზომის კომპანიებს მათი უსაფრთხოების ზომების სისუსტის გამო. ვინაიდან, როგორც ცნობილია, შიფრაცია იშვიათად გამოიყენება SIP-ში და რადგან, ჩვეულებრივ, ის არ არის კონფიგურირებული, ამიტომ მესამე მხარეს შეუძლია შეაგროვოს მომხმარებლის მონაცემები მაშინ, როდესაც მომხმარებელი მათ მიაწვდის მის რეგისტრატორ-სერვერს. ამ დროს მესამე მხარეს შეუძლია განახორციელოს ძვირადღირებული საქალაქთაშორისო ზარები ან თუნდაც მომხმარებლის მოპარული მონაცემებით შეიქმნას კარიბჭე და წუთობრივად გაყიდოს ძვირადღირებული მარშრუტების სატელეფონო მომსახურება. SIP-ის უსაფრთხოების მეთოდებია: დაშიფვრა, Ipsec, VPN, MPLS-გვირაბების გამოყენება, Firewalls, 5060 პორტის შეცვლა და სხვა.

შემოთავაზებული მოდელი მიზნად ისახავს SIP-ზე დაფუძნებული VoIP-ქსელის უსაფრთხოების დაცვას ინტერნეტისგან იზოლაციით.

ნახ. 20-ზე წარმოდგენილია SIP-ზე დაფუძნებული VOIP-ის უსაფრთხოების მოდელის არქიტექტურა (მისი კონცეპტუალური სტრუქტურა), რაც გულისხმობს ქსელურ ინფრასტრუქტურაში არსებული უსაფრთხოების გამოწვევებისა და ორგანიზაციული უსაფრთხოების ინფორმირებულობის ერთმანეთთან დამოკიდებულებას.

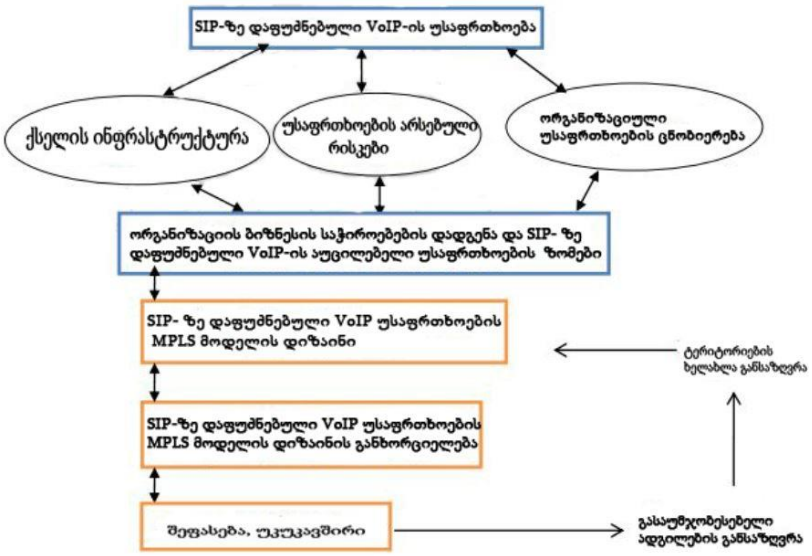
წარმოდგენილი მოდელი შემუშავდა უსაფრთხოების ზომებისა და არქიტექტურის სიღრმისეული შესწავლის საფუძველზე. აღნიშნული მოდელით შესაძლებელია ორგანიზაციის SIP-ზე დაფუძნებული VoIP-ქსელის უსაფრთხოების დაგეგმვა.

SIP-ზე დაფუძნებული VoIP-ის უსაფრთხოების მოდელი შეიქმნა შემდეგი ძირითადი მიმართულებების შესწავლის საფუძველზე:

1. VoIP-ის უსაფრთხოების ანალიზი;
2. SIP-ზე დაფუძნებული VoIP-ქსელის ინფრასტრუქტურა;

3. SIP-პროტოკოლის უსაფრთხოება;

4. ორგანიზაციის უსაფრთხოების ცნობიერების დონე.



ნახ. 20. SIP-ზე დაფუძნებული VOIP-ის უსაფრთხოების მოდელის არქიტექტურა

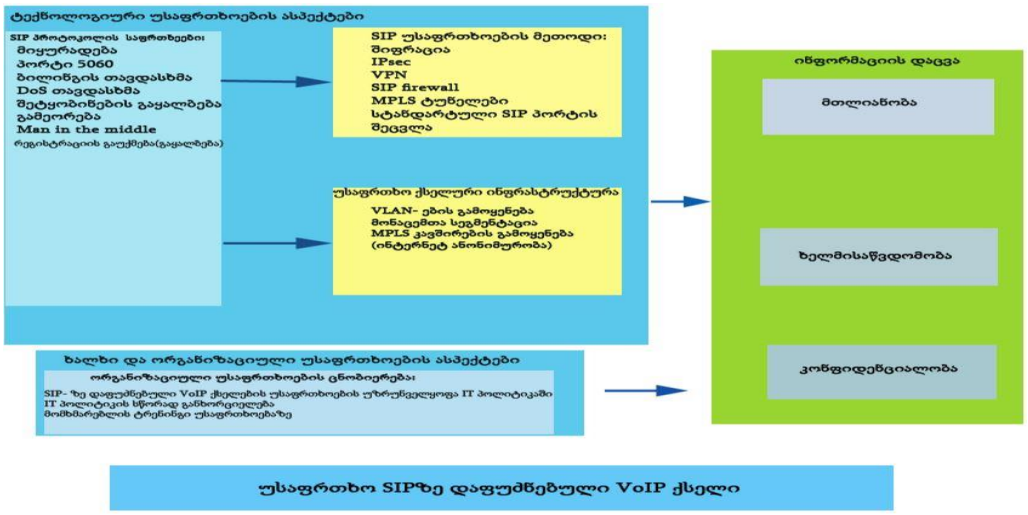
სხვადასხვა კომპანიის მიერ გამოიყენება უსაფრთხოების სხვადასხვა ზომები. თუმცა, უნდა აღინიშნოს, რომ მათ მიერ ხშირად არ ხდება SIP-პროტოკოლის დეტალების გათვალისწინება, რამაც შეიძლება გამოიწვიოს ხმოვანი ზარებისა (QoS-ის) და ინფორმაციის უსაფრთხოების პრობლემები. უსაფრთხოების კომპლექსურმა მეთოდებმა უნდა უზრუნველყოს მომსახურების უსაფრთხოება მისი ხარისხის გაუარესების გარეშე.

ხმოვანი კომუნიკაციის უსაფრთხოებისა და რისკებისაგან თავის არიდების მიზნით ორგანიზაციებმა უნდა განახორციელონ ქსელური ტრაფიკის სეგმენტაცია, რაც, ძირითადად, ხორციელდება სხვადასხვა ვირტუალური ლოკალური ქსელების (VLAN-ების) გამოყენებით.

SIP-ზე დაფუძნებული VoIP-ის ინფრასტრუქტურა SIP-ტრაფიკის მარშრუტიზაციის, მისი შექმნისა და შეწყვეტის მიზნით მოიცავს სხვადასხვა მოწყობილობებსა და შეერთებებს. გამოყენებული შესაბამისი აღჭურვილობა შედგება SIP-სერვერებისაგან, VoIP-კარიბჭეებისაგან,

მარშრუტიზატორებისაგან, ეკრანებისაგან (Firewalls). UA-ის SIP-ტელეფონებისაგან და SIP-ის პროგრამული ტელეფონებისაგან. SIP-ზე დაფუძნებული მეთოდები მგრძნობიარეა გარკვეული ტიპის შეტევების მიმართ, რაც იწვევს უსაფრთხოების ისეთ პრობლემებს, როგორიცაა შეტევები ორგანიზაციის ბილინგის სისტემაზე. 5060 სტანდარტული პორტია, რომელიც გამოიყენება ყველა SIP-ტრაფიკისთვის. თავდამსხმელებმა შეიძლება ადვილად განახორციელონ DDos-შეტევა, თუ პორტი 5060 ღიაა. SIP-მეთოდებისა და ტრაფიკის უსაფრთხოების უზრუნველყოფისთვის საჭიროა, რომ გამოყენებული იყოს 5060-საგან განსხვავებული პორტი და ტრაფიკის დაშიფვრის მეთოდი. უსაფრთხოების მოდელის შემუშავებისას საჭიროა არა მარტო ტექნოლოგიებზე და ინფრასტრუქტურულ ასპექტებზე კონცენტრირება, არამედ ორგანიზაციის უსაფრთხოების მიზნებისა და მომხმარებლის მოთხოვნებთან თანმხვედრი მოდელის ჩამოყალიბება იმისათვის, რომ უზრუნველყოფილი იყოს მაქსიმალური უსაფრთხოება.

მოდელი წარმოადგენს სახელმძღვანელოს და იგი წარმოაჩენს საჭირო პროცედურებს, რომელთა მეშვეობითაც ორგანიზაციებს შეუძლიათ SIP-ზე დაფუძნებული VoIP-ქსელების განხორციელება (ნახ. 21).



ნახ. 21. VoIP-ის უსაფრთხოების მოდელი

მოდელი შედგება სამი ძირითადი კომპონენტისგან: ტექნოლოგია, საინფორმაციო უსაფრთხოება და ორგანიზაცია-ხალხი. ქსელის ინფრასტრუქტურა, არსებული უსაფრთხოების რისკები და ორგანიზაციული უსაფრთხოების ინფორმირების დონე არის მნიშვნელოვანი ფაქტორები, რომლებიც განიხილება შემოთავაზებული მოდელის განვითარების დროს. მოდელი საშუალებას აძლევს ორგანიზაციას უკეთ დაიცვას უსაფრთხოების ძირითადი საინფორმაციო კომპონენტები, ინფორმაციის კონფიდენციალობა, მთლიანობა და ხელმისაწვდომობა. ორგანიზაციის გარემოში მოდელი რომ ეფექტურად ჩამოყალიბდეს, მნიშვნელოვანია, რომ ყურადღება მიექცეს არა მხოლოდ ტექნოლოგიურ ასპექტებს, არამედ ორგანიზაციისა და თანამშრომლების მოლოდინებს. ხალხსა და ტექნოლოგიას შორის მჭიდრო კავშირი ძალზედ მნიშვნელოვანი ფაქტორია. უსაფრთხოება, თავის მხრივ, სხვა ფაქტორებზეცაა დამოკიდებული, კერძოდ VoIP-ქსელის კომპონენტების შესაბამისი აპარატურის მწარმოებლისა და მისი კომპანიაში შემომტანი შუამავლის (ინტეგრატორის) კრიტიკულად შერჩევაზე, რაც მოდელის წარმატებით განხორციელების წინაპირობაა.

განვიხილოთ უსაფრთხოების მოდელის ტექნოლოგიური და ადამიანური ასპექტები. როგორც უკვე აღინიშნა, უსაფრთხოების მოდელი საჭიროა არა მხოლოდ ტექნოლოგიური ასპექტებისთვის, არამედ ორგანიზაციული საჭიროებებისთვისაც. უსაფრთხოება არის კავშირში ადამიანებთან და მათ ურთიერთქმედებასთან ტექნოლოგიებთან. ადამიანები, რომლებიც ურთიერთქმედებენ ამ ტექნოლოგიებთან, არ არიან სათანადოდ მომზადებული მისი გამოყენებისა და უსაფრთხოების რისკებზე, რაც გავლენას ახდენს სისტემის უსაფრთხოებაზე. ამიტომ ორგანიზაციებმა უნდა აწარმოონ სათანადო პოლიტიკა საინფორმაციო ტექნოლოგიებში (IT) გასათვიცნობიერებლად, რომელიც უზრუნველყოფს არა მარტო უსაფრთხოების ტექნოლოგიურ ნაწილს, არამედ იმასაც, რაც დაკავშირებულია პერსონალის გაწვრთნასთან. იმის გამო, რომ ყველა

თანამშრომელი, როგორც წესი, არ არის ტექნიკური პერსონალი, ორგანიზაციებმა უნდა უზრუნველყონ ქსელების შიდა უსაფრთხოება და თანამშრომლების სწორად ინფორმირება ტექნოლოგიის გამოყენების შესახებ. ინფორმაციული ტექნოლოგიების პოლიტიკა არა მარტო უნდა შეიქმნას, არამედ ის ეფექტურად უნდა განხორციელდეს ორგანიზაციაში უსაფრთხოების მაღალი დონის უზრუნველსაყოფად. ინფორმაციული უსაფრთხოების უზრუნველყოფის ძირითადი კომპონენტებია: ინფორმაციის კონფიდენციალურობა, ხელმისაწვდომობა და მთლიანობა [38, 39].

თავი 3. ინტერნეტის სერვის-პროვაიდერის ქსელი, მარშრუტიზაციის პრინციპები, პროტოკოლები და მათი მომავალი

3.1. მარშრუტიზაციის პრინციპები

ქსელური კომუნიკაციისთვის ერთ-ერთ მნიშვნელოვან მოწყობილობას წარმოადგენს მარშრუტიზატორი (Router). მარშრუტიზატორი ერთმანეთთან აკავშირებს სხვადასხვა ქსელებს და მართავს დანიშნულების ქსელის მიმართულებით მონაცემების გადაცემის პროცესს. მონაცემების დანიშნულების მიმართულებით გადასაცემად მარშრუტიზატორები იყენებენ ე.წ. მარშრუტიზაციის ცხრილს (Routing Table), რომელშიც ინახება ინფორმაცია მარშრუტიზატორისთვის ნაცნობი გზების შესახებ. მარშრუტიზაციის განხორციელებისთვის მარშრუტიზატორს უხდება შემდეგი ოპერაციების ჩატარება:

1. სატელეკომუნიკაციო ქსელში ფიზიკურად ან/და ლოგიკურად დაკავშირებულ მარშრუტიზატორებთან მარშრუტიზაციის მონაცემების გაცვლა. მისი განხორციელება ხდება სპეციალური პროგრამის გამოყენებით, რომელიც გათვალისწინებულია მარშრუტიზაციის პროტოკოლებით (routing protocols). მარშრუტიზაციის მონაცემების გასაცვლელად მარშრუტიზატორებზე უნდა დაკონფიგურირდეს მარშრუტიზაციის ერთი და იგივე პროტოკოლი;

2. მარშრუტიზაციის საკუთარი ცხრილის შევსება მარშრუტიზაციაზე მიღებული მონაცემებით;

3. მარშრუტიზაციის საკუთარი ცხრილის სკანირება და იქ არსებული ჩანაწერების მიხედვით დანიშნულების ქსელისკენ მიმავალი გზებიდან ერთი ან რამდენიმე საუკეთესოდ შეფასებული გზის არჩევა;

4. მონაცემების გადაცემისას მისი ურთიერთქმედება იმ მეზობელ მარშრუტიზატორთან, რომელიც განთავსებულია დანიშნულების ქსელის მიმართულების გზაზე;

5. მარშრუტიზაციის საკუთარი ცხრილიდან იმ ადაპტერის იდენტიფიცირება, რომლისკენაც უნდა მოხდეს მონაცემების გადაცემა;

6. მონაცემების მიღების შემთხვევაში შესაბამისი პაკეტის შემოწმება და დანიშნულების ქსელის მისამართის მიღება, მიღებული მისამართის შედარება მარშრუტიზაციის საკუთარ ცხრილში არსებულ ჩანაწერებთან და შემდეგი შესაბამისი მოქმედებების განსაზღვრა. ჩანაწერის პოვნის შემთხვევაში პაკეტი აგრძელებს გზას დანიშნულების ქსელისკენ, წინააღმდეგ შემთხვევაში კი როუტერი არ გააგზავნის მას.

7. ზემოთ ჩამოთვლილი ოპერაციების გარდა ისეთი დამატებითი ფუნქციების შემოწმება, როგორებიცაა პაკეტის სიცოცხლის ხანგრძლივობა (TTL) და მომსახურების ტიპი (TOS). შემოწმების გავლის შემდეგ პაკეტი აგრძელებს გზას დანიშნულების ქსელის მიმართულებით.

ზემოთ აღწერილი ყოველი მოქმედება მეორდება დანიშნულების ქსელისკენ მიმავალ გზაზე განთავსებულ ყველა მარშრუტიზატორზე. მარშრუტიზატორები მონაცემების დანიშნულების მიმართულებით გადასაცემად მიმართავენ მარშრუტიზაციის საკუთარ ცხრილს, რომელიც შეიცავს ინფორმაციას დაშორებული ქსელების შესახებ. მარშრუტიზაციის ცხრილში ინფორმაცია ამა თუ იმ ქსელის შესახებ შეიძლება გამოჩნდეს ორი (სტატიკური და დინამიური) სახით:

1. მარშრუტიზაციის ცხრილში ინფორმაციის სტატიკური სახით გამოჩენის შემთხვევაში ქსელის ადმინისტრატორი მარშრუტიზატორში ბრძანებათა გარკვეული მიმდევრობით შეიტანს ინფორმაციას ამა თუ იმ ქსელის შესახებ;

2. მარშრუტიზაციის ცხრილში ინფორმაციის დინამიური სახით გამოჩენის შემთხვევაში შესაძლებელია მასში ამუშავებულ იქნას სპეციფიკური პროგრამა, რომელიც ერთი და იმავე პროგრამის ფარგლებში

ფიზიკურად ან ლოგიკურად დაკავშირებულ მარშრუტიზატორებს აძლევს მათთვის ნაცნობი ქსელების შესახებ ინფორმაციის ურთიერთგაცვლის საშუალებას [40].

3.2. დინამიური მარშრუტიზაციის პროტოკოლები

დინამიური მარშრუტიზაციის პროტოკოლები იყოფიან ორ - IGP და EGP-პროტოკოლებად. IGP-პროტოკოლები გამოიყენებიან ორგანიზაციის ქსელის შიგნით საქმიანობისთვის. მარშრუტიზაციის ასეთი პროტოკოლებია RIP, EIGRP, IS-IS და OSPF.

EGP-პროტოკოლები შემუშავებულ იქნა მარშრუტიზაციის გაფართოებული ცხრილების სამართავად და, აგრეთვე, ინტერნეტის ქსელის სტრუქტურირების ასამაღლებლად მარშრუტიზაციის დომენების სხვადასხვა ადმინისტრაციულ ერთეულებად დაყოფის საშუალებით. ასეთ ადმინისტრაციულ ერთეულებს ავტონომიური სისტემა (AS) ეწოდებათ. დღეისთვის ინტერნეტის ქსელში საერთაშორისო მარშრუტიზაციისთვის გამოყენებულ ერთადერთ პროტოკოლს წარმოადგენს BGP-4.

მარშრუტიზატორი მის ცხრილში არსებული ინფორმაციის მიხედვით იღებს გადაწყვეტილებას მარშრუტიზაციის შესახებ. მარშრუტები შესაძლებელია დაინიშნოს ადმინისტრატორის მიერ სტატიკურად ან გამოეყოს მას დინამიურად სხვა მარშრუტიზატორის მეშვეობით ან მარშრუტიზაციის პროგრამული პროტოკოლით. მარშრუტის დადგენა ხდება შემდეგი 4 მონაცემის საფუძველზე:

1. მიმღების მისამართი;
2. ქვექსელის ნიღაბი;
3. კარიბჭის (Gateway) მისამართი ან ინტერფეისის სახელი;
4. მარშრუტის ”ღირებულება” ან მეტრიკა.

საკუთარი ინტერფეისისა და სხვა მარშრუტიზატორებისაგან მომავალი ინფორმაციის დინამიური მართვისთვის გამოიყენება მარშრუტიზაციის პროტოკოლები. დინამიური მარშრუტიზაცია ცვლის მარშრუტების სტატიკურად შერჩევის შრომატევად პროცედურას ქსელური ადმინისტრატორის ჩარევის გარეშე. ამ დროს მარშრუტიზაციის ალგორითმისა და ინფორმაციის ანალიზი ხდება ორი ძირითადი კრიტერიუმის (მანძილი და ვექტორი) საფუძველზე:

1. მანძილის კრიტერიუმის შემთხვევაში ხდება იმის გარკვევა, თუ რამდენად დაშორებულია დანიშნულების წერტილი მოცემული მარშრუტიზატორიდან;

2. ვექტორის კრიტერიუმის შემთხვევაში ხდება იმის გარკვევა, თუ რომელი მიმართულებითაა მიზანშეწონილი მოცემულ ქსელში პაკეტების გადამისამართება.

მარშრუტის მანძილი ფასდება ღირებულებით ან მეტრიკით, რომელიც შეიძლება ახასიათებდეს შემდეგი პარამეტრებიდან ერთ-ერთს:

1. გადასვლების რიცხვი;

2. ადმინისტრაციული ირიბი ხარჯები;

3. გამტარუნარიანობა;

4. გადაცემის სიჩქარე;

5. შეფერხების ალბათობა;

6. საიმედოობა;

7. პირდაპირი მარშრუტი.

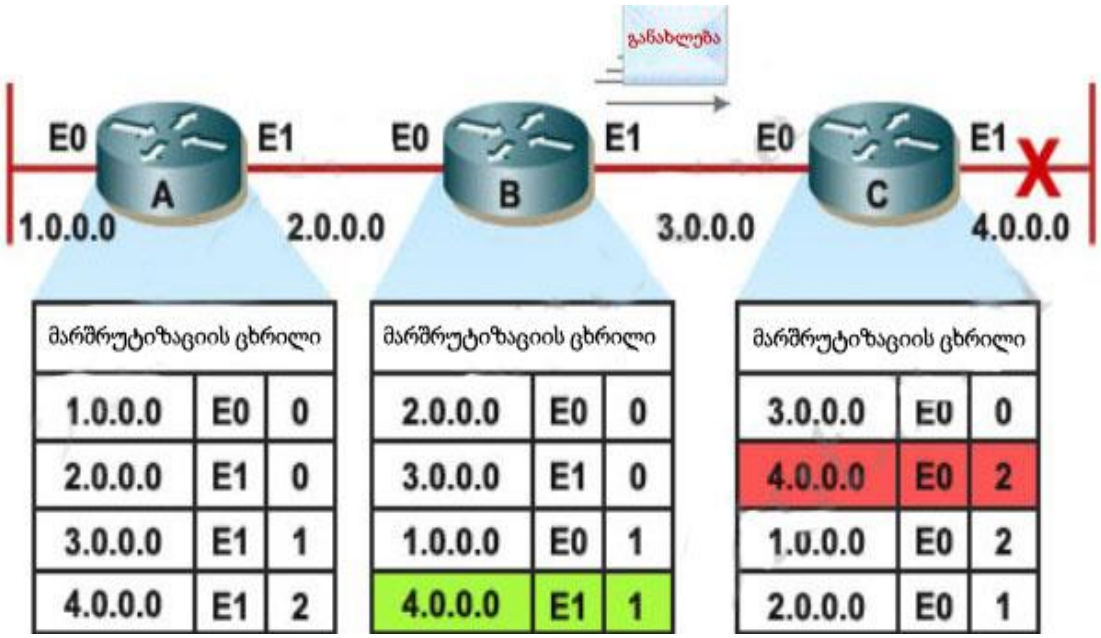
მარშრუტიზატორის ჩართვისთანავე სამუშაო რეჟიმში ერთვებიან მისი გამართული ინტერფეისები და ამ დროს მარშრუტიზაციის ცხრილში ჩაიწერება ინფორმაცია მასთან უშუალოდ მიერთებული ლოკალური ქსელის მისამართების შესახებ. Cisco-ს მარშრუტიზატორებში ამგვარი მარშრუტები აღინიშნება პრეფიქსით C. მათი განახლება ხდება ავტომატურად პირდაპირი მარშრუტის ინტერფეისის ხელახალი გამართვისას ან მარშრუტის გამორთვისას.

ქსელის ადმინისტრატორს შეუძლია ხელით გამართოს კონკრეტული ქსელის სტატიკური მარშრუტი, რომელიც მარშრუტიზაციის ცხრილში აღინიშნება სიმბოლოთი S და რომელიც არ შეიცვლება მანამ, სანამ ადმინისტრატორი არ შეცვლის მას.

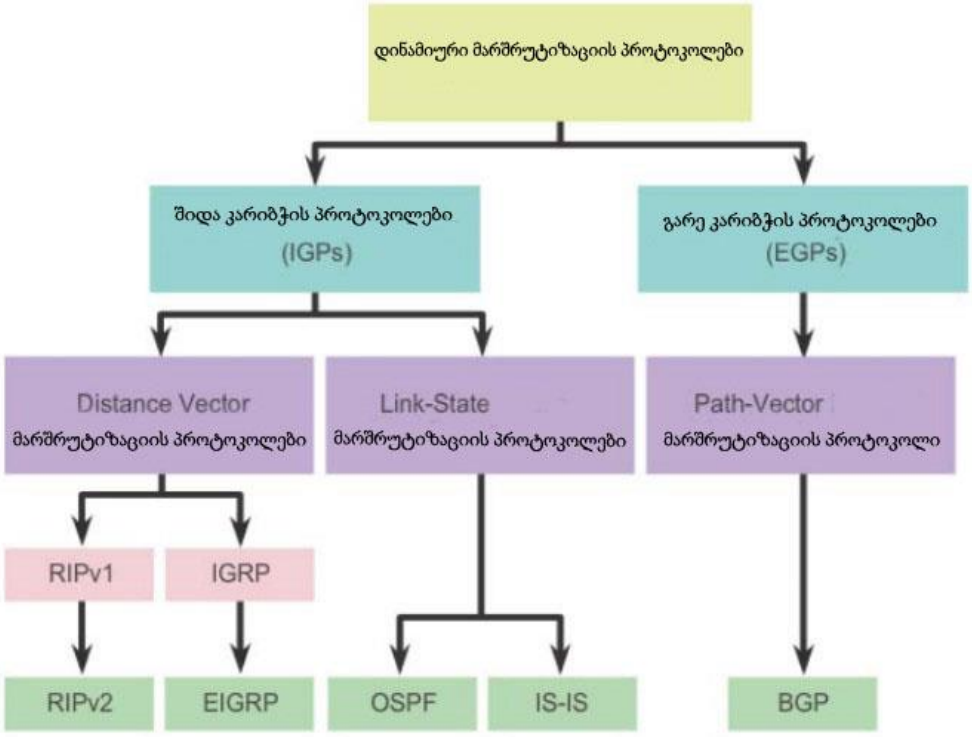
იმ ქსელების დამისამართებისთვის, რომელთა გზაც არ არის ასახული მარშრუტიზაციის ცხრილში, გამოიყენება კარიბჭის მისამართი Default. ჩვეულებრივ, ეს არის შემდეგი მარშრუტიზატორის მისამართი ინტერნეტის სერვის-პროვაიდერისკენ მიმავალ გზაზე ქსელში მხოლოდ ერთი მარშრუტიზატორის არსებობის შემთხვევაში, რომლის ამორჩევა ხდება ავტომატურად. მარშრუტიზაციის ცხრილში შესაბამისი მარშრუტი აღინიშნება პრეფიქსით S*. დინამიური მარშრუტები ავტომატურად იქმნება და ახლდება მარშრუტიზაციის პროტოკოლების მიერ. ეს მარშრუტები მარშრუტიზაციის ცხრილში გამოისახება წინსართით, რომელიც ასახავს მარშრუტის შემქმნელი პროტოკოლის ტიპს. მაგალითად, წინსართი R შეესაბამება მარშრუტის შესახებ ინფორმაციის RIP-პროტოკოლს. მარშრუტიზაციის პროტოკოლების მეშვეობით მარშრუტიზატორები ერთმანეთს უცვლიან განახლებულ ინფორმაციებს და დინამიურად ქმნიან და ანახლებენ მარშრუტიზაციის ცხრილებს. მათი დინამიური განახლების პროცესი წარმოდგენილია ნახ. 22-ზე.

უნდა აღინიშნოს, რომ დინამიური მარშრუტიზაცია საუკეთესოა დიდი მასშტაბის ქსელების შემთხვევაში.

ინტერნეტი ემყარება ავტონომიური სისტემების AS-კონცეფციას, რის გამოც რეალიზებულია პროტოკოლების 2 – IGP და EGP ჯგუფი. პირველი მათგანი გამოიყენება ერთი ავტონომიური სისტემის შიგნით მარშრუტიზაციისთვის, ხოლო მეორე - ავტონომიურ სისტემებს შორის მარშრუტიზაციისთვის. ნახ. 23-ზე ნაჩვენებია დინამიური მარშრუტიზაციის პროტოკოლები და მათი ურთიერთკავშირი [41].



ნახ. 22. მარშრუტიზაციის ცხრილების დინამიური განახლება



ნახ. 23. მარშრუტიზაციის პროტოკოლები და მათი ურთიერთკავშირი

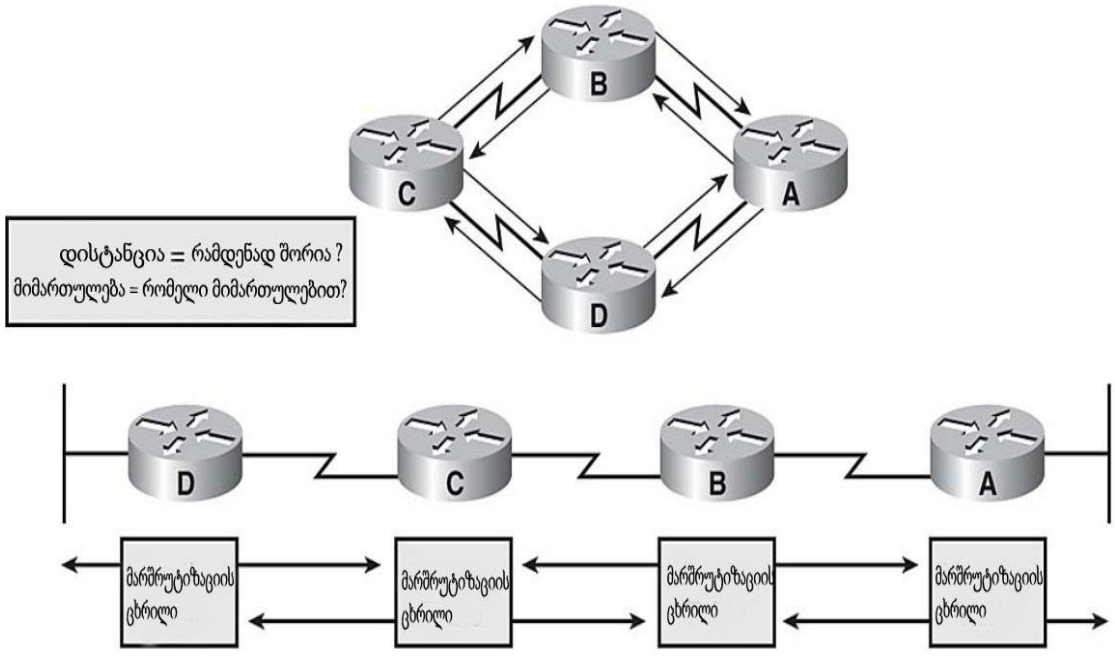
3.3. მარშრუტიზაციის Distance Vector-პროტოკოლები

მარშრუტიზაციის IGPs-პროტოკოლები და მათი განვითარების შედეგად შექმნილი RIPv1, RIPv2, IGRP, EIGRP-პროტოკოლები ეფუძნებიან შემდეგ 2 მახასიათებელს:

1. მახასიათებელი Distance განსაზღვრავს დანიშნულების ქსელამდე სიშორეს და იგი ემყარება ინფორმაციებს გადასასვლელების (Hop) რაოდენობისა, მარშრუტის ღირებულებისა, გამტარუნარიანობისა, მონაცემთა პაკეტების დაყოვნებისა და სხვათა შესახებ;

2. მახასიათებელი Vector განსაზღვრავს მომდევნო გადასასვლელი მარშრუტიზატორის ან გადასასვლელის ინტერფეისს.

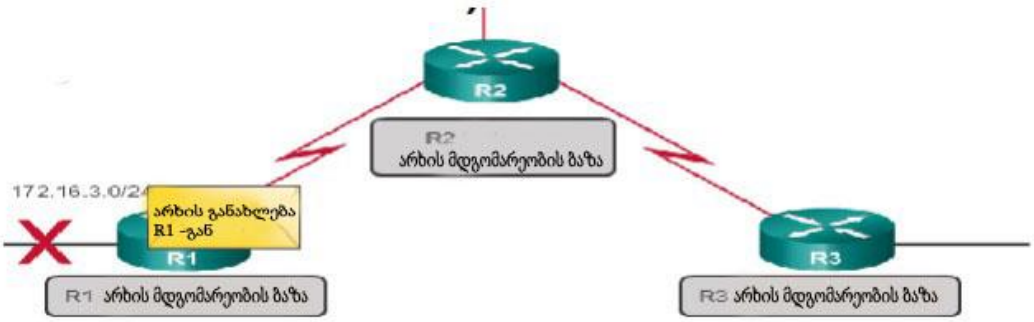
ნახ. 24-ზე წარმოდგენილია მარშრუტიზაციის Distance Vector-პროტოკოლების მოქმედების ვიზუალური წარმოდგენა.



ნახ. 24. მარშრუტიზაციის Distance Vector-პროტოკოლების მოქმედების ვიზუალური წარმოდგენა

3.4. მარშრუტიზაციის Link-State-პროტოკოლები

IPv4, IGPs, OSPF, IS-IS-პროტოკოლებით კონფიგურირებულ მარშრუტიზატორებს ყველა დანარჩენი მარშრუტიზატორიდან ინფორმაციის მიღების შედეგად შეუძლიათ სრული წარმოდგენის შექმნა ქსელის ტოპოლოგიის შესახებ. მოცემული პროტოკოლები განსაკუთრებით ეფექტურია დიდი ზომის იერარქიულ ქსელებში, ანუ მაშინ, როდესაც ქსელების სწრაფ კონვერგენციას აქვს გადამწყვეტი მნიშვნელობა. RIP-პროტოკოლებით კონფიგურირებული მარშრუტიზატორები პერიოდულად აგზავნიან განახლებებს მეზობელ როუტერებზე მაშინ, როდესაც Link-State-განახლებები იგზავნება ქსელის ტოპოლოგიაში ცვლილების განხორციელების შემდეგ, რაც ჩანს ნახ. 25-ზე წარმოდგენილი სქემიდან.



ნახ. 25. მარშრუტიზაციის Link-State-პროტოკოლების მოქმედების ვიზუალური წარმოდგენა

მარშრუტიზაციის პროტოკოლები შეიძლება ერთმანეთს შევადაროთ შემდეგი მახასიათებლების მიხედვით:

1. მახასიათებელი „კონვერგენციის სიჩქარე“ განსაზღვრავს იმას, თუ მარშრუტიზატორები რამდენად სწრაფად უცვლიან ერთმანეთს ინფორმაციას ქსელში მომხდარი ცვლილებების შესახებ;

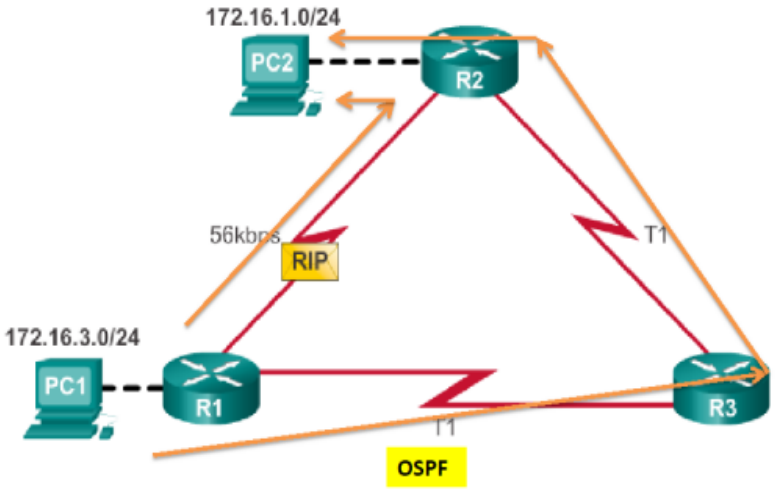
2. მახასიათებელი „მასშტაბურობა“ აღნიშნავს იმას, თუ რამდენად მასშტაბური შეიძლება იყოს ქსელი;

3. მარშრუტიზაციის მახასიათებელი ქვექსელის ნიღაბის შემოწმების მხარდაჭერით (Classful) და მის გარეშე (Classless);

4. მახასიათებელი „რესურსების გამოყენება“ ასახავს მარშრუტიზაციის პროტოკოლების მოთხოვნებს, კერძოდ კი მეხსიერების (RAM-ის) ზომას, პროცესორის სისწრაფესა და გამტარუნარიანობას;

5. მახასიათებელი „რეალიზაცია და მომსახურება“ აღწერს ქსელის ადმინისტრატორის ცოდნის დონეს მარშრუტიზაციის მოცემული პროტოკოლების საფუძველზე ქსელის სათანადო მართვის თვალსაზრისით.

სხვადასხვა პროტოკოლები იყენებენ განსხვავებულ მეტრიკას დანიშნულების მისამართამდე მარშრუტის განსაზღვრისას. მაგალითად, RIP-პროტოკოლი განსაზღვრავს მარშრუტს გადასასვლელების (Hop) რაოდენობის მიხედვით მაშინ, როდესაც OSPF-პროტოკოლი ირჩევს მარშრუტს მონაცემთა გადაცემის არხის გამტარუნარიანობაზე (bandwidth) დაყრდნობით. ნახ. 26-ზე ნაჩვენებია RIP და OSPF-პროტოკოლების მეტრიკებით განსაზღვრული მარშრუტები [41].



ნახ. 26. RIP და OSPF-პროტოკოლების მეტრიკებით განსაზღვრული მარშრუტები

3.5. ინტერნეტ-პროვაიდერების ქსელები

ხშირ შემთხვევაში ინტერნეტ-პროვაიდერების ქსელები, მათი მასშტაბებიდან გამომდინარე, კომპლექსურია, რაც იმას ნიშნავს, რომ მათი ფიზიკური და ლოგიკური ტოპოლოგია იერარქიულადაა განაწილებული. ქსელი იწყება მომიჯნავე მარშრუტიზატორებით, რომლებიც უზრუნველყოფენ კავშირს ე. წ. აღმავალ პროვაიდერებთან, ინტერნეტის მიმოცვლის წერტილებთან (IXP) და, სურვილისამებრ, სხვადასხვა მსხვილ ორგანიზაციებთან. მომიჯნავე როუტერები, როგორც წესი, განთავსებული არიან ევროპის სხვადასხვა ქალაქებში. მათ ფიზიკური და ლოგიკური კავშირი აქვთ როგორც ერთმანეთთან, ასევე ქსელის ბირთვთან (CORE-თან). აქედან გამომდინარე, შემდეგი შრეა ქსელის ბირთვი, რომელიც, ძირითადად, უზრუნველყოფს ქსელის სხვადასხვა ნაწილში ტრაფიკის გატარებას მაქსიმალურად სწრაფად. ქსელის ბირთვის როუტერები, უმეტეს წილად, არ არიან დატვირთულნი კონფიგურაციებითა და ფუნქციებით, რადგან უზრუნველყოფილი უნდა იყოს მათი წარმადობის მაქსიმალურად სრულად გამოყენება. ისინი უზრუნველყოფენ მაღალსიჩქარიან კავშირს განაწილების შრისა და მომიჯნავე როუტერებს შორის. განაწილების შრეზე (Distribution layer) ხორციელდება მომხმარებლებისთვის საჭირო თითქმის ყველანაირი კონფიგურაცია, წესები და დგება დაშვების სიები (Access Lists). ქსელის ბოლო შრეა დაშვების შრე (Acess layer), რომელიც უზრუნველყოფს უშუალოდ მომხმარებლების ჩართვას. დაშვების შრის ქსელური მოწყობილობები სხვადასხვა ტიპისაა, რაც იმაზეა დამოკიდებული, თუ რომელი ტექნოლოგიით (P2P, DSL, GPON, GE-LINK, WIFI და ა.შ.) მიეწოდება მომხმარებელს სერვისი.

ინტერნეტ-პროვაიდერებს საკუთარი ქსელური აპარატურა საქართველოს გარდა გააჩნიათ ევროპის სხვადასხვა ქალაქებშიც, მაგალითად, სოფიასა და ფრანკფურტში. აღნიშნულ ქალაქებში ინტერნეტ-ტრაფიკის გაცვლის რამდენიმე წერტილია (IXP), რომლებიც იძლევიან

სხვადასხვა ავტონომიური სისტემების ერთმანეთთან დაკავშირების საშუალებას საერთო წერტილიდან. ამის გამო სათითაოდ ყველა სისტემასთან ფიზიკურად მიერთება აღარ ხდება საჭირო. მათი ტრაფიკი იცვლება არა ინტერნეტ-პროვაიდერის ქსელის გამოყენებით, არამედ ურთიერთჩართვის წერტილში. ინტერნეტ-ტრაფიკის გაცვლის წერტილებია NETIX, DE-CIX, AMSIX და ა.შ. არსებობენ, აგრეთვე, უმსხვილესი კონტენტ-პროვაიდერები (მაგალითად, Google და Facebook), რომელთანაც აუცილებელია პირდაპირი ფიზიკური მიერთება, რადგან გლობალური ტრაფიკის 70%-მდე მოდის ამ პროვაიდერების სისტემებზე. შესაბამისად, ორივე მხარის - ინტერნეტ-პროვაიდერისა და კონტენტ-პროვაიდერის ინტერესშია ტრაფიკის პირდაპირი გაცვლა. გარდა ამისა, აღნიშნული კონტენტ-პროვაიდერები დროებითი მეხსიერების (Cache) სერვერებს სთავაზობენ დიდი ტრაფიკის მქონე პროვაიდერებს, რაც საგრძნობლად ამცირებს მონაცემთა გადაცემის გლობალური არხების გადატვირთულობის საშიშროებას. გარკვეულ კონტენტ-პროვაიდერებთან პირდაპირი მიერთებების გარდა საჭიროა სხვა ინტერნეტ-კონტენტთან კავშირების არსებობაც, რომელსაც უზრუნველყოფს ე. წ. აღმავალი პროვაიდერი, მაგალითად, Level 3; Telia Sonera; Sofia Connect; NTT და ა. შ. ურთიერთჩართვებისას პროვაიდერი აანონსებს საკუთარი და მასთან ჩართული კლიენტების კუთვნილ IPv4 და IPv6-პრეფიქსებს BGP-პროტოკოლის საშუალებით, ხოლო აღმავალი პროვაიდერებისგან ღებულობს სრულ ინფორმაციას (Full View) ინტერნეტ-პრეფიქსების შესახებ, რათა ზუსტად განსაზღვროს ის, თუ რომელი პრეფიქსი რომელი პროვაიდერისგანაა უფრო „ადვილად“ მიღებადი. ინტერნეტ-პრეფიქსების შესახებ სრული ინფორმაციის გარდა ინტერნეტ-პროვაიდერები აღმავალი პროვაიდერებისგან ღებულობენ ინფორმაციას სტანდარტული მარშრუტის შესახებაც (Default Route), რომელიც იმ შემთხვევაში გამოიყენება, როდესაც ინტერნეტ-პრეფიქსების შესახებ მიღებულ სრულ ინფორმაციაში არ აღმოჩნდება ინფორმაცია იმ ქსელის შესახებ, რომლისკენაც უნდა

გაიგზავნოს პაკეტი. ინტერნეტ-პრეფიქსების შესახებ სრული ინფორმაცია, როგორც წესი, ჭირდება როგორც მომიჯნავე, ასევე ქსელის ბირთვის მარშრუტიზატორებს, ქსელის დანარჩენ ნაწილში კი ვრცელდება ინფორმაცია სტანდარტული მარშრუტის შესახებ. სხვადასხვა ავტონომიური სისტემების (AS) დასაკავშირებლად გამოიყენება გარე სასაზღვრო კარიბჭის პროტოკოლი eBGP, ხოლო იგივე ავტონომიურ სისტემაში მყოფ სისტემებს შორის - ქსელთაშორის პროტოკოლი iBGP. ინტერნეტ-პრეფიქსების შესახებ სრული ინფორმაციის არსებობისას მარშრუტიზატორს უწევს 600 000-ზე მეტი IPv4 და 45 000-მდე IPv6 პრეფიქსის გაანალიზება, რაც დიდ აპარატურულ რესურსს მოითხოვს. დიდი აპარატურული მოთხოვნებიდან გამომდინარე მწარმოებლები პროვაიდერებს სთავაზობენ შესაბამის მოდელებს. ძირითადად პროვაიდერები Cisco-ს წარმოების პროდუქტით აწყობენ ქსელს, თუმცა ასევე პოპულარულია Juniper-ის წარმოების აპარატურაც. პროვაიდერებისთვის წარმოებული მოდელები Cisco, ASR-9000 და სხვადასხვა სერიის Juniper MX-ის მოდელები ერთმანეთისგან განსხვავდება ზომითა და წარმადობით. ერთი და იგივე სერიის მარშრუტიზატორები შესაძლებელია გამოყენებულ იქნეს განაწილების შრეზეც, თუმცა აღნიშნულ შრეზე ასევე გამოიყენება 7600 და 6800 სერიის Cisco-ს წარმოების როუტერებიც. არის შემთხვევები, როდესაც დაშვების შრეზეც გამოიყენება მარშრუტიზაციის ფუნქციების მქონე კომუტატორები. ისინი ამცირებენ L2 დომეინს და ამის გამო მარშრუტიზაციაც მათზეა მორგებული. მსგავსი შემთხვევებისთვის შეიძლება გამოვყოთ Cisco-ს წარმოების 4500, 3600 და ASR920 სერიის კომუტატორები. როგორც უკვე იყო აღნიშნული, დაშვების შრეზე აპარატურის სახეობის შერჩევა დამოკიდებულია გამოყენებული ტექნოლოგიის სახეობაზე, კერძოდ, GPON-ტექნოლოგიის შემთხვევაში გამოიყენება სხვადასხვა მწარმოებლის OLT-ები (Huawei, ZTE, CALIX, ERICSSON და სხვ.) ADSL-ტექნოლოგიის შემთხვევაშიც DSLAM-ების მწარმოებელი კომპანიების ფართო არჩევანია (კომპანიები Huawei, ZTE, Zyxel, Alcatel და სხვ.).

როგორც მომიჯნავე როუტერებს შორის, ასევე მომიჯნავე და ქსელის ბირთვის როუტერებს შორის საჭიროა ფიზიკური კავშირები, რომელიც ხორციელდება ოპტიკური კაბელის გამოყენებით. როგორც წესი, მსგავსი მიერთებები ხორციელდება WDM-ტექნოლოგიის გამოყენებით, რომელიც უზრუნველყოფს 1 ოპტიკურ ბოჭკოში სხვადასხვა სიხშირის გამოყენებით გარკვეული რაოდენობის სხივის გატარებას. DWDM-აპარატურაზეა დამოკიდებული ის, თუ როგორი გამტარობის არხები გვექნება. საქართველოში პროვაიდერები ძირითადად იყენებენ 10 და 100 გიგაბიტიან შეერთებებს.

განაწილების შრის (Distribution) მარშრუტიზატორები განთავსებულია ქალაქების, რეგიონებისა და სეგმენტების მიხედვით და ყველა მათგანი ლოგიკურად დაკავშირებულია ქსელის ბირთვთან. ქსელის შიდა დინამიური მარშრუტიზაციის პროტოკოლის (IGP) მუშაობის უზრუნველსაყოფად მათ შორის ფიზიკური კავშირებისთვის გამოიყენება ლოგიკური დამისამართების პრინციპი. როგორც წესი, პროვაიდერები, სურვილისამებრ, გამოიყენებენ ISIS ან OSPF-პროტოკოლებს. აღნიშნული პროტოკოლები იძლევიან ქსელის ტოპოლოგიის გაანალიზების საშუალებას, რის შემდგომაც ამ პროტოკოლის გამოყენებით მყარდება iBGP-სესიები როგორც ერთმანეთთან პირდაპირ მიერთებულ, ასევე ერთმანეთისგან დაშორებულ მარშრუტიზატორებს შორის. ამავე შრეზე გამოიყენება MPLS-პროტოკოლი, რომელიც მონაცემებს ერთი ქსელური კვანძიდან გადასცემს შემდეგ კვანძს ნიშნულების და არა ე. წ. გრძელი ქსელის მისამართების გამოყენებით, რაც მოწყობილობას თავიდან აცილებს მარშრუტიზაციის ცხრილში კომპლექსური ძიების პროცესს.

3.6. პროგრამული უზრუნველყოფით კონტროლირებადი SDN-ქსელი

პროგრამით კონტროლირებადი ქსელის ძირითადი მიმართულება ქსელური მოწყობილობებიდან ინფორმაციის ამოღებაა. ამ მიზნით

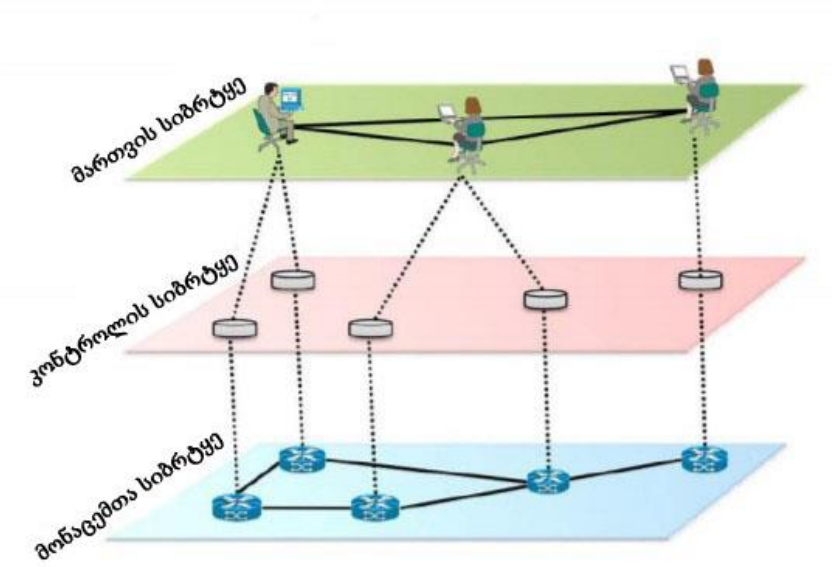
დღეისთვის გამოიყენება თითქმის იგივე არქიტექტურის მარშრუტიზატორები და კომუტატორები, რომლებიც გამოიყენებოდა ათეული წლების წინ. განსხვავება მხოლოდ გაზრდილ სისწრაფესა და მეტ გამტარუნარიანობაშია, ხოლო მოქმედების პრინციპი და ინტელექტუალური შესაძლებლობები უმთავრესად უცვლელია. ქსელური მოწყობილობები კი წლების განმავლობაში დარჩა უცვლელი. SDN-ქსელში ქსელური მოწყობილობები ნაკლებად ინტელექტუალურია. ამ ქსელში მთელი სისტემა აწყობილია ქსელის კონტროლის მექანიზმზე. ისეთი აპლიკაციებისთვის როგორიცაა (youtube, Skype,VoIP) მნიშვნელოვანია ის, რომ დაყოვნება და ჯიტერის პარამეტრები ყოველთვის იყოს სტაბილურად მისაღებ დონეებზე. დღევანდელ ქსელებში, რეალურ დროში კომუნიკაციის პრობლემის დროს, გამოყენებულია QoS-ტექნოლოგია იმ მიზნით, რომ შეიქმნას გარკვეული სახის პრიორიტეტული პაკეტები.

ტრადიციული IP-ქსელები კომპლექსურია და ისინი რთულია სამართავად. იმავდროულად რთულდება ქსელის კონფიგურაცია წინასწარ განსაზღვრული პოლიტიკის მიხედვით მისი რეკონსტრუქციისას ხარვეზების დაფიქსირებისა და დატვირთვისა და განსახორციელებელი ცვლილებების დროს. თანამედროვე ქსელები ვერტიკალურადაა ინტეგრირებული და მათში კონტროლისა და მონაცემთა გადაცემის მოდულები ერთმანეთთანაა შერწყმული. SDN პარადიგმაა, რომელიც ცვლის ვერტიკალურ ინტეგრაციას ქსელის კონტროლის ლოგიკის გამოყოფით მარშრუტიზატორებისა და კომუტატორებისგან. SDN-ით ხდება ქსელის მართვის ლოგიკური ცენტრალიზაციის ხელშეწყობა და ქსელური პროგრამირების დანერგვა. ნახ. 27-ზე წარმოდგენილია SDN-ქსელის არქიტექტურა.

SDN-ქსელი შედგება მონაცემთა გადამტანი მოწყობილობებისაგან, რომლებშიც თავმოყრილია კომუტატორები და მარშრუტიზატორები. მაკონტროლებელი მოწყობილობები შეიცავენ მართვის სერვერებს, რომლებიც ამყარებენ კომუნიკაციას მონაცემთა გადამტან ყველა

მოწყობილობასთან და განსაზღვრავენ მონაცემთა ტიპებს. მონაცემთა ტიპების შეცვლა შეიძლება დროში დინამიურად, რაც იძლევა ერთი წერტილიდან მთელი ქსელის კონტროლის საშუალებას. კონტროლისა და

მონაცემთა მართვის ნაწილების გაყოფის დროს მათთვის საჭიროა კომუნიკაციის ახალი ფორმის შექმნა. ამ მიზნით SDN-ში არსებობს ქსელის მოწყობილობების კონტროლის ოქმი OpenFlow. SDN-ქსელები შესაძლებელია იმართოს აპლიკაციების მეშვეობით, რომლებიც განთავსებულია მართვის მოდულში, რის გამოც SDN-ის არქიტექტურა ხდება მოქნილი, ცენტრალურად მართვადი და პროგრამირებადი [42].



ნახ. 27. SDN არქიტექტურა

3.6.1. SDN-ქსელის კომპონენტები

SDN-ქსელის გადამამისამართებელი მოწყობილობებია (FD) კომუტატორები და მარშრუტიზატორები, რომლებიც პასუხისმგებელნი არიან ნაკადის გატარებაზე ცენტრალური კონტროლერიდან ნაკარნახევი წესების მიხედვით. FD-მოწყობილობები კონტროლერთან დაკავშირებულია Southbound-ინტერფეისით, რაც გათვალისწინებულია OpenFlow-პროტოკოლით.

ქსელის მონაცემთა გადამტან მოწყობილობებში (DP) ერთიანდება ყველა გადამამისამართებელი მოწყობილობა.

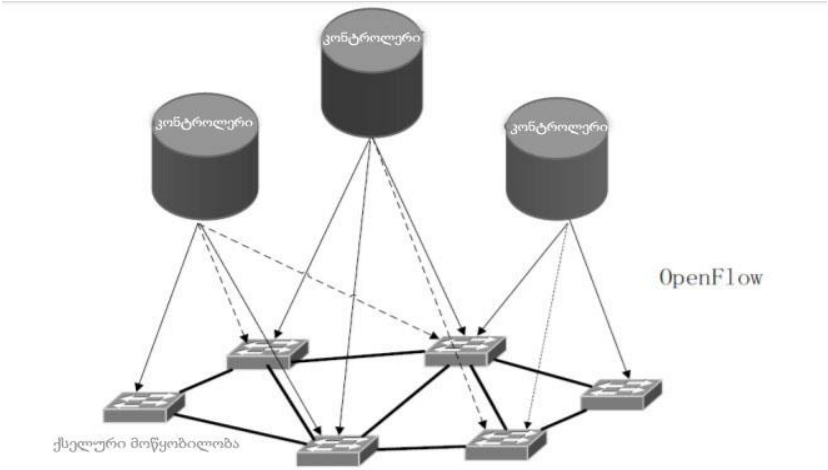
ქსელის საკონტროლო მოდული (CP) ქსელის „ტვინია“, რომელიც მონაცემთა გადამტან მოდულს უკავშირდება Southbound-ინტერფეისით, ხოლო აპლიკაციებს - Northbound ინტერფეისით. Southbound ინტერფეისი (SI) იმ გზის მაჩვენებელია, რომლითაც კონტროლერი უკავშირდება გადამამისამართებელ მოწყობილობებს OpenFlow-პროტოკოლის საშუალებით. Northbound-ინტერფეისის (NI) წყალობით SDN-ის არქიტექტურაში შესაძლებლია კონტროლერის დაპროგრამება და მისი პარამეტრების შეცვლა.

OpenFlow ქსელური ღია კომუნიკაციის სტანდარტული ინტერფეისია, რომელიც გამოყენებულია SDN-ქსელში Control Plane-სა და Data Plane-ს შორის კავშირისთვის. პროტოკოლი ძირითადად იძლევა გადამამისამართებელი მოწყობილობების მარშრუტიზატორების და კომუტატორების კონფიგურების საშუალებას. OpenFlow-პროტოკოლით შესაძლებელია შეიქმნას ქსელის მართვის ერთიანი პოლიტიკა, რომელიც შეიძლება გავრცელდეს მთლიან ქსელზე. პროტოკოლი საშუალებას აძლევს ცენტრალურ კონტროლერს დისტანციურად მართოს მოწყობილობები. ეს გარემოება ქსელს ხდის ავტომატურს, რაც აღმოფხვრის კონფიგურაციის გაკეთებას თითოეულ მოწყობილობაზე. აღსანიშნავია ამ პროტოკოლოს კიდევ ერთი უპირატესობა, რაც გამოიხატება იმაში, რომ არაა კონკრეტულ მწარმოებელზე დამოკიდებული და, შესაბამისად, ეს ფაქტორი ქსელის მშენებლობის პროცესს კიდევ უფრო აადვილებს.

3.6.2. განაწილებული კონტროლერები

SDN-ქსელებში ხელმისაწვდომია განაწილებული კონტროლერების არქიტექტურა. ეს გარემოება სისტემას აძლევს იმის ფუფუნებას, რომ ჰქონდეს ცდომილების ბევრი წერტილი SPOF.

ერთობლივად მომუშავე მრავალრიცხოვანი კონტროლერები ქსელს აძლევენ მასშტაბირების, დატვირთვის დაბალანსებისა და მაღალი ხელმისაწვდომობის შესაძლებლობას. იმისათვის, რომ კონტროლერებმა იმუშაონ განაწილებულ რეჟიმში, საჭიროა მათ შორის გარკვეული ინფორმაცის გაცვლა, რაც მოიცავს ქსელის ტოპოლოგიისა და მართვის დონის პარამეტრების შესახებ ინფორმაციას (ნახ. 28).



ნახ. 28. განაწილებული კონტროლერების არქიტექტურა

3.6.3. ODL-პროტოკოლი

Open Day Light (ODL) არის ღია პროექტი, რომელიც დაარსდა 2013 წლის აპრილში. მისი პირველი გამოცემა მოხდა 2014 წლის თებერვალში. შესაბამისი პროტოკოლის მიზანია ცენტრალიზებული მართვის სისტემის უზრუნველყოფა, რომელიც API-სტრუქტურის საშუალებით იძლევა ქსელის პროგრამირების საშუალებას. მიკრომომსახურების არქიტექტურა მომხმარებლებს აძლევს პროგრამების, პროტოკოლებისა და მოდულების გაკონტროლების საშუალებას. იგი ასევე უზრუნველყოფს კავშირებს გარე

მომხმარებლებსა და პროვაიდერებს შორის. სხვადასხვა მოთხოვნების მიმართ კონტროლერი ძალზე ადაპტირებადია და იგი სხვადასხვა პრობლემის დროს სერვისებისა და პროტოკოლების შერწყმას ხდის შესაძლებელს. ODL ღია კოდის სტრუქტურაა, რაც იძლევა მსოფლიოს მასშტაბით მრავალი პროგრამისტის ჩართულობის შესაძლებლობას. ეს ფაქტი კი ხელს უწყობს მის სწრაფ განვითარებას.

სოციალური მედია, მობილური მოწყობილობები და სერვისების ერთობლიობა „ღრუბლოვანი გამოთვლები“ ცდება ტრადიციული ქსელების შესაძლებლობების ფარგლებს. გამოთვლით და არქივირების (დამახსოვრების, შენახვის) ტექნიკაში უკვე გამოიყენება წარმოუდგენელი ინოვაციები ვირტუალიზაციითა და ავტომატიზაციით, მაგრამ ეს სარგებელი შეზღუდულია ქსელის გამტარუნარიანობის შესაძლებლობების თვალსაზრისით. პროგრამული უზრუნველყოფით განსაზღვრულ ქსელს აქვს პოტენციალი მოახდიონს რევოლუცია მოძველებულ მონაცემთა ცენტრებში ქსელის მოქნილი კონტროლის ისეთი მექანიზმით, როგორიცაა ვირტუალიზაცია. როგორც ზემოთ აღინიშნა, SDN-ით შესაძლებელია სხვადასხვა სარგებელის მიღება სხვადასხვა ტიპის ორგანიზაციებისთვის, მათ შორის სერვის-პროვაიდერებისთვის ღრუბლოვანი გამოთვლების ჩატარების შემთხვევაში, მონაცემთა ცენტრებისა და საწარმოთა კამპუსებისთვის ქსელური ინფორმაციის გადაცემისას [43].

თავი 4. ქსელის მონიტორინგის ალგორითმი, მისი მოდელირება და სიმულაცია

4.1. ქსელის მონიტორინგის საფუძვლები

კომპიუტერული ქსელები წარმოიშვა ერთრანგიანი მარტივი მცირე რაოდენობის მოწყობილობებისგან შემდგარი სისტემებიდან. დღეისთვის უკვე არსებობს კომპლექსური ქსელები, რომლებშიც ისეთი ტექნოლოგიებია გაერთიანებული, როგორებიცაა "ღრუბლოვანი" სისტემები, უსადენო კავშირები, დისტანციური მომხმარებლები, ვირტუალური კერძო ქსელები (VPN), საგნების ინტერნეტი (IoT), მობილური მოწყობილობები და სხვა. მიუხედავად კომპიუტერული ქსელების ასეთი ევოლუციისა, ქსელური მონიტორინგის სისტემის გამოყენების აუცილებლობა მაინც მუდმივ ფაქტორად რჩება. იგი ქსელის სპეციალისტებს აძლევს იმის საშუალებას, რომ მათ ჰქონდეთ ინფორმაცია ნებისმიერი სახეობის თავიანთ ქსელში (WAN, LAN, VoIP, MPLS და სხვ.) არსებული მდგომარეობისა და სხვადასხვა ქსელის ისეთი ელემენტების ან/და კვანძების შესახებ, როგორებიცა კონცენტრატორები, მარშრუტიზატორები, ეკრანები (firewalls), სერვერები და მომხმარებლის სისტემები.

კომპიუტერული ქსელი მოწყობილობათა კომპლექტია, რომლებიც ერთმანეთთან დაკავშირებულია საერთო სატრანსპორტო ან საკომუნიკაციო პროტოკოლის მეშვეობით. კომუნიკაცია შეიძლება განისაზღვროს როგორც მომხმარებლებს ან/და ქსელის კვანძების ისეთ მოწყობილობებს შორის მონაცემთა გადაცემა, როგორებიცაა კომპიუტერები, მობილური მოწყობილობები, გამომავალი მოწყობილობები, მართვის ელემენტები, სერვერები, მარშრუტიზაციისა და გადართვის მოწყობილობები და ა.შ. გეოგრაფიული განაწილების მიხედვით არსებობს ლოკალური (LAN), რეგიონული (WAN) და გლობალური ინტერნეტ-ქსელი.

ქსელის დიზაინის ან ტოპოლოგიის მიუხედავად, ნებისმიერი სახეობის ქსელი ემორჩილება მინიშნებებს და წესებს, რომლებიც აღწერილია მონაცემთა გადაცემისა და კომუნიკაციის OSI-მოდელში.

ქსელის სერვერებისა და სისტემების მონიტორინგისთვის აუცილებელია ქვემოთ ჩამოთვლილი პირობების დაკმაყოფილება:

1. ქსელში არსებული სხვადასხვა ელემენტების იმ მონაცემებისადმი წვდომის შესაძლებლობა, რომლებიც შეიცავენ ინფორმაციას საკონტროლებელი ელემენტის სამუშაო, მიმდინარე და ფუნქციონალური მდგომარეობების შესახებ;

2. მონიტორინგის პროგრამამ უნდა განახორციელოს მონაცემთა შეგროვება, დამუშავება და მომხმარებლისთვის იოლად აღქმად (ე. წ. „მეგობრულ“) ფორმატში მათი მიწოდება და წინასწარ განსაზღვრული ნორმების (პროგრამული ზღვრების) დარღვევის შემთხვევაში უზრუნველყოს მომხმარებლების ინფორმირება მოსალოდნელი პრობლემების შესახებ;

3. მონიტორინგის პროგრამასა და მონიტორინგზე აყვანილ ელემენტს შორის ინფორმაციის გადაცემის პროტოკოლის ან მეთოდის არსებობა.

ქსელში არსებული ინფორმაციის მოპოვება ხელს უწყობს ქსელის უკეთეს მართვასა და კონტროლის განხორციელებას და ქსელის პრობლემების იდენტიფიცირებას მანამ, სანამ პრობლემა გამოიწვევს შეფერხებებს და რომელიმე კვანძის გათიშვას. მუდმივი მონიტორინგი ხელს უწყობს ისეთი ქსელის შექმნას, რომელიც გამოირჩევა ე. წ. მაღალი საშემსრულებლო დისციპლინით.

ქსელის მონიტორინგის მონაცემების შეგროვებისთვის გამოყენებული ზოგადი მეთოდები, პროტოკოლები მოწყობილობები, სისტემები და პროგრამული უზრუნველმყოფებია:

1. Ping-მეთოდი არის IP-ქსელში ჰოსტის ხელმისაწვდომობისა და არსებობის შესამოწმებლად გამოსაყენებული ინსტრუმენტი. Ping-ის მონაცემებზე დაყრდნობით შეიძლება დადგინდეს ინფორმაცია ქსელის

მასპინძელი ჰოსტის აქტიურობის შესახებ. გარდა ამისა, ჰოსტთან კომუნიკაციის დროს შესაძლებელია გაიზომოს პაკეტის გადაცემის დრო და მისი დაკარგვის სიხშირე;

2. ქსელის მართვის მარტივი პროტოკოლი (SNMP) ქსელური მართვის პროტოკოლია, რომელიც გამოიყენება ქსელის მომხმარებელთა შორის იმ ინფორმაციის გასაცვლელად, რომელსაც მოიცავს ქსელური მონიტორინგის პროგრამა. იგი ქსელის მართვისა და მონიტორინგის ყველაზე ფართოდ გამოყენებადი პროტოკოლია.

3. მართვადი მოწყობილობა მხარს უჭერს SNMP-პროტოკოლს და მისი მეშვეობით ხელმისაწვდომი ხდება სპეციფიკური ინფორმაცია;

4. აგენტი პროგრამული უზრუნველყოფაა, რომელიც მართვადი მოწყობილობების ნაწილია. მისთვის ხელმისაწვდომია მოწყობილობის მართვის საინფორმაციო ბაზა (MIB), რითაც NMS-სისტემებს აძლევს ინფორმაციის წაკითხვისა და ჩაწერის საშუალებას;

5. ქსელის მართვის NMS-სისტემა მონიტორინგს უწევს და აკონტროლებს მართვად მოწყობილობებს აგენტის მეშვეობითა და SNMP-ბრძანებების გამოყენებით.

SNMP-მონაცემები გროვდება პერიოდულად გამოკითხვა-მოთხოვნის ან ე. წ. ტრაპ-შეტყობინებების მიღებით. MIB არის ბაზა, რომელსაც გააჩნია ინფორმაცია მართვადი მოწყობილობების სტრუქტურისა და მართვის სისტემის შესახებ. იგი შეიცავს ობიექტის იდენტიფიკატორებს (OID), რომლებიც წარმოადგენენ მართვადი მოწყობილობების მუდმივ ან ცვლად მნიშვნელობებს.

შეტყობინებების ლოგირების სისტემა Syslog მართვად მოწყობილობებს აძლევს ქსელში შეტყობინებების გაგზავნის საშუალებას. შეტყობინებების შესახებ ინფორმაცია შეიძლება გამოყენებულ იქნას როგორც სისტემური მენეჯმენტისთვის, ასევე უსაფრთხოების აუდიტისთვის. ეს სისტემა მხარდაჭერილია უამრავი მოწყობილობის, მარშრუტიზატორების, კომუტატორების, ქსელური ეკრანებისა და სხვათა

მიერ. ქსელებში, სადაც მონიტორინგისთვის NMS არ არის ხელმისაწვდომი ან არსებულ NMS-ს არ გააჩნია კონკრეტული ფუნქციების მხარდაჭერა და ინსტრუმენტების ფუნქციონირების გაფართოება, ქსელის სპეციალისტებს ეძლევათ პროგრამული სკრიპტების გამოყენების შესაძლებლობა. სკრიპტები იყენებენ ისეთ საერთო ბრძანებებს, როგორიცაა ping, netstat, lynx, snmpwalk და ა. შ. სკრიპტები შეიძლება შეიქმნას პროგრამირების სხვადასხვა ენის (Perl, python და სხვ.) გამოყენებით.

მონიტორინგი ქსელისა და სისტემების ადმინისტრატორებს ეხმარება ქსელში რაიმე შეცდომის არსებობისას შექმნილი პრობლემების ძირეული მიზეზების გამოვლენისა და მათი იდენტიფიცირების საქმეში მანამ, სანამ ისინი გავლენას მოახდენენ კავშირის უწყვეტობაზე. ამ შემთხვევაში მნიშვნელობა არ აქვს იმას, რომ საწარმოს საკომუნიკაციო ქსელი მცირე (50 კვანძზე ნაკლები) თუ დიდი (1000 კვანძზე მეტი) მოცულობისაა. უწყვეტი მონიტორინგი ეხმარება ე. წ. მაღალი საშემსრულებლო დისციპლინის მქონე ქსელის განვითარებასა და მისი მუშაობის შენარჩუნებას მინიმალური დროით გათიშვის რეჟიმში.

ქსელის მონიტორინგის განმახორციელებელი სისტემა, მისი ღირებულების გარდა, უნდა იყოს ყოვლის მომცველი, ანუ უნდა მოიცავდეს საწარმოს ყველა ასპექტს, კერძოდ კი ქსელსა და მასთან ასოცირებულ კავშირს, სისტემებსა და მათთან ასოცირებულ უსაფრთხოებას. სისტემა უნდა უზრუნველყოფდეს ქსელის შესახებ ყველა სახის ინფორმაციის (ანგარიშგება, პრობლემის გამოვლენა და მოგვარება და ქსელის მუშა მდგომარეობაში შენარჩუნება) მონიტორინგს. ქსელის მართვა ფართო სფეროა, რომელიც მოიცავს სხვადასხვა ფუნქციებს. ქსელის მენეჯმენტის სხვადასხვა მიზნები კლასიფიცირებულია და დაჯგუფებულია ხუთ სხვადასხვა კატეგორიად, რომელთა შორისაა ბრალეულობის მენეჯმენტი (F), კონფიგურაციის მენეჯმენტი (C), საბუღალტრო მენეჯმენტი (A), შესრულების მენეჯმენტი (P) და უსაფრთხოების მენეჯმენტი (S). მათი საერთო აღნიშვნაა FCAPS.

ყველა ქსელს, მათი მუშაობის სპეციფიკის გათვალისწინებით, გააჩნია თავისი განსხვავებული საბაზისო დონე, რომელიც აღწერს იმას, თუ ქსელის რომელი პარამეტრია ნორმის ფარგლებში. იმ შემთხვევაში, როდესაც ქსელის პარამეტრები განსხვავდებიან მათი საბაზისო დონისაგან, მაშინ წარმოიქმნება უარყოფითი შედეგების წარმოშობის პოტენციური საფრთხე. ასეთ შემთხვევებში განგაშის შესახებ გაფრთხილება ქსელის ამა თუ იმ პარამეტრის საშუალო მნიშვნელობიდან გადახრის შემთხვევაშიც კი ხელს შეუწყობს პრობლემების ადრეულ გამოვლენასა და მის მოგვარებას, რაც, თავის მხრივ, უზრუნველყოფს ქსელის უწყვეტ ფუნქციონირებას. არსებობს გაფრთხილება-განგაშის სიგნალის ფორმირების სხვადასხვა ვარიანტი. გაფრთხილება-განგაშის სიგნალებთანაა დაკავშირებული შემდეგი პროცესები:

1. **ტრიგერი:** ტრიგერი აღნიშნავს მოვლენას, რომელიც იწვევს განგაშის სიგნალს. მოვლენა შეიძლება აღინიშნოს როგორც კვანძის მდგომარეობის ან მისი პარამეტრის წინასწარ განსაზღვრული მნიშვნელობის ცვლილება.

2. **ზღვრები, გამეორების რაოდენობა და დროითი შეფერხებები:** განგაშის შესახებ შეტყობინებათა უმეტესობა დაფუძნებულია ზღვრულ მაჩვენებლებზე. კერძოდ, განგაშის შესახებ სიგნალი ფორმირდება იმ შემთხვევაში, როდესაც ქსელის ამა თუ იმ პარამეტრის საბაზისო მნიშვნელობა გადაკვეთს ზღვრულ მაჩვენებელს. განგაშის შესახებ სიგნალი შეიძლება იყოს მარტივი ან კომპლექსური, რომელიც შეიძლება გაიგზავნოს, მაგალითად, 2-ჯერ 10 წუთის განმავლობაში. დასაშვებია, აგრეთვე, სხვა ვარიანტის არსებობის შესაძლებლობა.

3. **გადატვირთვა, საწყის პარამეტრებზე დაყენება:** იმ დროს, როდესაც ქსელის ესა თუ ის პარამეტრი დაუბრუნდება საბაზისო ზღვრულ მაჩვენებელს, შესაძლებელია განგაშის სიგნალის შეწყვეტა.

4. **ქსელის მართვა და მონიტორინგი:** ქსელის მართვა და მონიტორინგის სტრატეგია ისეთივე მნიშვნელოვანია, როგორც ქსელის დიზაინი და მისი განხორციელება. მონიტორინგის გარეშე კარგად

დაგეგმილი და მოწყობილი ქსელი მცირე პრობლემის შემთხვევაშიც კი შეიძლება გამოვიდეს მწყობრიდან. ორგანიზაციებისა და ქსელის ადმინისტრატორებმა ქსელის მონიტორინგის განხორციელებისას უნდა გაითვალისწინონ, აგრეთვე, საკუთარი და სხვათა მიერ დაგროვილი საერთო გამოცდილებაც, რაც გამოყენებული უნდა იყოს სამომავლო მოქმედებების ძირითადი სტრატეგიის განსასაზღვრავად. თუმცა, ეს სულაც არ ნიშნავს იმას, რომ ქსელის მონიტორინგი უნდა შემოიფარგლოს მხოლოდ საერთო გამოცდილებით.

5. **ხელმისაწვდომობის მონიტორინგი:** ხელმისაწვდომობის მონიტორინგი განსაზღვრავს საინფორმაციო ტექნოლოგიების ინფრასტრუქტურის ყველა რესურსის მონიტორინგს, რათა უზრუნველყოფილი იყოს ორგანიზაციისა და მისი მომხმარებლების მოთხოვნების დაკმაყოფილება. დღევანდელი IT-ინფრასტრუქტურა ბიზნესის მოთხოვნების დასაკმაყოფილებლად საჭიროებს მონიტორინგის დროის უწყვეტობას 100%-ით. ქსელში ჩართული მოწყობილობები და სერვისები ყოველთვის უნდა იყოს ხელმისაწვდომი ბიზნესის უწყვეტობის უზრუნველსაყოფად. რესურსებისა და მომსახურების უწყვეტი მონიტორინგი უზრუნველყოფს ქსელური კვანძის ან/და მომსახურების ხელმისაწვდომობას. ხელმისაწვდომობის მონიტორინგის ზოგიერთი საყოველთაოდ გამოყენებული ტექნოლოგიებია:

1. Ping-ტექნოლოგია ყველაზე ფართოდ გამოყენებადი მეთოდია. მისი გამოყენებისას სიგნალი ICMPpings იგზავნება საკონტროლებელ მოწყობილობაზე და მიღებული პასუხების საფუძველზე ირკვევა საკითხი მოწყობილობის ან სერვისის ხელმისაწვდომობის შესახებ;

2. ტექნოლოგია „ტელნეტი“ გამოიყენება ქსელის მოწყობილობის ხელმისაწვდომობის შესამოწმებლად და მართვისთვის;

3. SNMP-ტექნოლოგია გამოიყენება მოწყობილობის ხელმისაწვდომობის ან მომსახურების არსებული მდგომარეობის შესაფასებლად;

4. WMI-ტექნოლოგია გამოიყენება Windows-სისტემებში მომსახურების ხელმისაწვდომობის შემოწმებისთვის;

5. IPSLA-ტექნოლოგია გამოიყენება Ciscos-ის აპარატურაში და მას შეუძლია გაზომოს WAN-კავშირების ხელმისაწვდომობა და მათი გამტარუნარიანობა კონკრეტული სერვისების მიხედვით;

6. ტექნოლოგია „ინტერფეისის მონიტორინგი“ გამოიყენება ქსელში არსებული მრავალი ტიპის ინტერფეისისთვის, მაგალითად, სწრაფი Ethernet და მაღალსიჩქარიანი Gigabit Ethernet-ნტერფეისებისთვის. ქსელურ მოწყობილობებში ინტერფეისს წარმოადგენს მონაცემთა პაკეტების შესვლისა და გასვლის წერტილი. თუ ინტერფეისზე დაფიქსირდა შეცდომა, პაკეტების დაკარგვა ან თუნდაც მოხდა ინტერფეისის გათიშვა, მაშინ გაუარესდება სერვისის ხარისხი ან სულაც გაითიშება სერვისი. ქსელის მონიტორინგის სისტემები ქსელის მოწყობილობიდან ინფორმაციის შესაგროვებლად იყენებენ პინგს ან SNMP-პროტოკოლს. SNMP-ის გამოყენებით ხდება უფრო მეტი რაოდენობის დეტალური მონაცემის შეგროვება ინტერფეისის გამტარუნარიანობის, სატრანსპორტო სიჩქარის, შეცდომების, უარყოფებისა და სხვათა შესახებ;

7. ტექნოლოგია „დისკის მონიტორინგი“ ელექტრონული მონაცემების საცავი ორგანიზაციისთვის ერთ-ერთი ყველაზე მნიშვნელოვანი რესურსია. საწარმოებში მონაცემები გროვდება მრავალი დისკისგან შემდგარ მასივებზე. ეს ტექნოლოგია მოიცავს დისკზე სათანადო მართვის ეფექტური სივრცის გამოყენებას, დისკზე შესრულების შეცდომებს, დიდი ფაილების სტატისტიკას, თავისუფალ სივრცეს, დისკის მოცულობის შესახებ ცვლილებებს და ა. შ.

8. ტექნოლოგია „აპარატურა“ გამოიყენება მარშრუტიზაციისა და კომუტაციისთვის, მონაცემთა შენახვისთვის, აპლიკაციის სერვერებისთვისა და სხვა მიზნებისთვის, რაც ხორციელდება მრავალი მოწყობილობის გამოყენებით. ამ ტექნოლოგიის საფუძველზე იქმნება მთელი IT-ინფრასტრუქტურის ხერხემალი. ქსელური მოწყობილობის მნიშვნელოვანი

საკონტროლებელი კომპონენტებია: ცენტრალური პროცესორი CPU. თუ იგი გამოყენებულია მაქსიმალურად, მაშინ მოწყობილობა გამოვა მწყობრიდან; ტემპერატურა; გაგრილების სისტემა და ვენტილატორის სიჩქარე; ელექტროენერგიის მიწოდებისა კვების ბლოკის მდგომარეობა. ისინი კრიტიკულად მნიშვნელოვანი კომპონენტებია, რის გამოც უნდა ხორციელდებოდეს მათი მონიტორინგი.

4.2. EVE-NG-პლათფორმა

შემდეგი თაობის ემულირებული ვირტუალური გარემო (EVE-NG) ქსელის გრაფიკული ემულატორია, რომელიც მხარს უჭერს მარშრუტიზატორის ოპერაციულ სისტემებს. ქსელის ემულატორს შეუძლია მუშაობა ე. წ. ვირტუალურ მანქანაზე. ემულატორის მართვის გრაფიკული ინტერფეისი იხსნება ვებ-ბრაუზერში. ემულატორი მომხმარებლებს აძლევს კომპიუტერული ქსელის კვანძების შექმნის, მათი ერთმანეთთან დაკავშირებისა და კონფიგურაციის განხორციელების საშუალებას. შესაძლებელია, აგრეთვე, ახალი ოპერაციული სისტემების დამატება თითქმის ნებისმიერი ტიპის ქსელის მხარდასაჭერად.

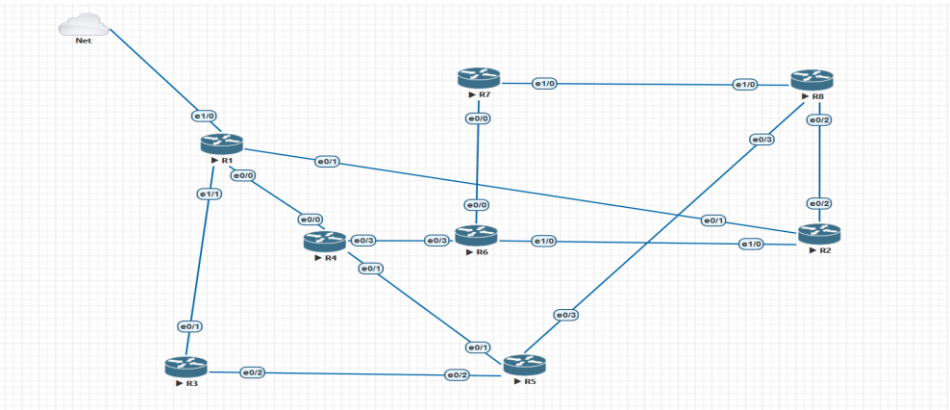
EVE-NG-პლათფორმა შეიძლება გამოყენებულ იქნას სხვადასხვა სახის ტექნოლოგიების შესწავლის მიზნით. ემულატორით შესაძლებელია ზოგადი ტექნოლოგიების ან მწარმოებლის მიერ დამუშავებული სპეციფიკური თემების შესწავლა და დიაგნოსტიკა. EVE-NG-გარემოში შესაძლებელია დაინერგოს სხვადასხვა ტიპის ქსელური ტოპოლოგიები, მოხდეს ქსელური პრობლემის სიმულაცია და შემდეგ ამ პრობლემის მოგვარება (თრაბელშუთინგი), კერძოდ კი პაკეტების ინსპექტირება wireshark-ით. იგი შეიძლება გამოყენებულ იქნას კორპორატიული პროვაიდერის ქსელის სიმულაციისთვის ქსელური ცვლილებების წინასწარი შემოწმებისა და წარმოებაში შემდგომი ჩაშვების მიზნით. EVE-NG-პლათფორმა ხელმისაწვდომია OVA ან ISO-ფაილის ფორმატებში. ღია

ვირტუალური მოწყობილობის (OVA) ფორმატი არის არქივი (TAR), რომელიც შეიცავს პაკეტების დისკებისა და კონფიგურაციის ფაილებს, რომლებიც გამოიყენება ვირტუალური მანქანის აღწერისთვის. EVE-NG-ს ინსტალაციისთვის შეიძლება გამოყენებულ იქნას VM-ის VMware Workstation ან ESXi გარემო.

4.3. IP-MPLS-ისა და TE-ს მოდელირება

წინამდებარე ნაშრომის მიზნების განხორციელების მიზნით MPLS-ქსელის სიმულაციისა და ექსპერიმენტებისთვის გამოყენებულ იქნა მარშრუტიზატორები ოპერაციული სისტემით IOS Version 15.4(2)T. ნაშრომში განხილულ სიმულაციურ ქსელში გამოყენებული ტექნოლოგიები და პროტოკოლები მოიცავენ IP-, MPLS-, LDP-, RSVP-, SNMP-, SSH-, OSPF-, BGP; Te-, VRF- და QoS-პროტოკოლებსა და მეთოდებს.

ნახ. 29-ზე ნაჩვენებ სურათზე მოცემულია საქალაქთაშორისო IP/MPLS-ქსელი. მასში მარშრუტიზატორები განთავსებულია საკვანძო ქალაქებში და ფიზიკურ დონეზე კავშირები დაზღვეულია იმ მიზნით, რომ კაბელის დაზიანებამ არ გამოიწვიოს ამა თუ იმ ქალაქის იზოლირება. ქსელის სიმულაციისისას გამოყენებულია 8 მარშრუტიზატორი პირობითი დასახელებებით R1, R2, ..., R8.



ნახ. 29. MPLS-ქსელის ტოპოლოგია

ერთმანეთთან ლოგიკური კავშირისთვის მარშრუტიზატორებს შორის დამაკავშირებელ ინტერფეისებზე გაწერილია Ipv4-მისამართები. IP-კავშირის დამყარების შემდეგ მარშრუტების დინამიურად საფორმირებლად და გასავრცელებლად ინტერფეისებზე ირთვება დინამიური მარშრუტიზაციის პროტოკოლი, განხილულ შემთხვევაში - OSPF-პროტოკოლი, რომელიც აღნიშნულ ქსელში გამოიყენება მხოლოდ როუტერების Loopback-მისამართების დინამიურად გასაცვლელად. თითოეულ მარშრუტიზატორზე შექმნილია Loopback0-ინტერფეისი და გაწერილია შესაბამისი Ipv4-მისამართი, რომელსაც სხვა მარშრუტიზატორები ეცნობიან OSPF-პროტოკოლის საშუალებით. იმიტაციის პირობებში, სიმარტივისთვის, Loopback-მისამართები ემთხვევა მარშრუტიზატორების ნომრებს. მაგალითად, R1-მარშრუტიზატორს შეესაბამება მისამართი 1.1.1.1, R2-მარშრუტიზატორს - მისამართი 2.2.2.2 და ა.შ. აღნიშნული IP-მისამართების მიხედვით მონიტორინგის სისტემას შემდგომში დაემატება ახალი მარშრუტიზატორები. დინამიური მარშრუტიზაციის გამართვა დადასტურდება რომელიმე მარშრუტიზატორზე გლობალური როუტინგ-ცხრილის ნახვის შედეგად, სადაც აუცილებლად უნდა იყოს დანარჩენი მარშრუტიზატორების შესაბამისი Loopback-მისამართები, რაც წარმოდგენილია დინამიური მარშრუტიზაციის ცხრილის სახით (ნახ. 30).

```
R1_LAB#show ip route ospf
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override

Gateway of last resort is not set

      2.0.0.0/32 is subnetted, 1 subnets
O        2.2.2.2 [110/11] via 10.40.50.6, 1w3d, Ethernet0/1
      3.0.0.0/32 is subnetted, 1 subnets
O        3.3.3.3 [110/11] via 10.40.50.30, 1w3d, Ethernet1/1
      4.0.0.0/32 is subnetted, 1 subnets
O        4.4.4.4 [110/11] via 10.40.50.14, 1w3d, Ethernet0/0
      5.0.0.0/32 is subnetted, 1 subnets
O        5.5.5.5 [110/21] via 10.40.50.30, 1w3d, Ethernet1/1
                 [110/21] via 10.40.50.14, 1w3d, Ethernet0/0
      6.0.0.0/32 is subnetted, 1 subnets
O        6.6.6.6 [110/21] via 10.40.50.14, 1w3d, Ethernet0/0
                 [110/21] via 10.40.50.6, 1w3d, Ethernet0/1
      7.0.0.0/32 is subnetted, 1 subnets
O        7.7.7.7 [110/31] via 10.40.50.14, 1w3d, Ethernet0/0
                 [110/31] via 10.40.50.6, 1w3d, Ethernet0/1
      8.0.0.0/32 is subnetted, 1 subnets
O        8.8.8.8 [110/21] via 10.40.50.6, 1w3d, Ethernet0/1
```

ნახ. 30. დინამიური მარშრუტიზაციის ცხრილი

ქსელში MPLS-პაკეტების ნიშნულების მიხედვით კომუტაციისთვის საჭიროა ინტერფეისებზე ან დინამიური მარშრუტიზაციის პროტოკოლში ჩაირთოს LDP-პროტოკოლი. მისი მთელ ქსელში ჩართვის შემდეგ მარშრუტიზატორები გაცვლიან ინფორმაციებს ნიშნულების შესახებ შესაბამისი ქსელებისთვის. ნახ 31-ზე ნაჩვენებ სურათზე ჩანს ინტერფეისზე გააქტიურებული LDP-პროტოკოლი.

```
interface Ethernet0/1
 ip address 10.40.50.5 255.255.255.252
 ip ospf network point-to-point
 ip ospf 1 area 0
 mpls ip
```

ნახ. 31. LDP-პროტოკოლის გააქტიურება მარშრუტიზატორში.

როგორც ნახ. 32-ზე ჩანს, თითოეულ IP-მისამართს აქვს შესაბამისი ნიშნული. თითოეულ მარშრუტიზატორზე ხდება ამ ნიშნულების ჩანაცვლება, რადგან ნიშნულები ლოკალურადაა დაგენერირებული და ისინი გაიცვლებიან LDP-პროტოკოლის მეშვეობით. შესაბამისად, თითოეულ მარშრუტიზატორს აქვს მომდევნო მარშრუტიზატორის მიერ კონკრეტული ქსელისთვის დაგენერირებული ნიშნული და იგი ჩანაცვლებული ნიშნულით გადასცემს პაკეტებს აღნიშნული მარშრუტიზატორისკენ. თუ საბოლოო IP-მისამართი შემდეგ მარშრუტიზატორზეა გაწერილი, მაშინ მისი წინა მარშრუტიზატორი პაკეტის ნიშნულს კი არ ცვლის, არამედ მთლიანად ხსნის ამ ნიშნულს და მომდევნო მარშრუტიზატორს უგზავნის მხოლოდ IP-პაკეტს ნიშნულის გარეშე. ნახ. 32-ზე კონკრეტული მისამართების გასწვრივ ამიტომაც ჩანს ფრაზა "pop label". აღნიშნულ ქსელში საჭიროა MPLS-TE, რათა წინასწარ განისაზღვროს საწყისი წერტილიდან დანიშნულების წერტილამდე სასურველი ან არასასურველი მარშრუტი, რაც იძლევა ტრაფიკების ოპტიმალურად განაწილების საშუალებას. ამიტომ გლობალური კონფიგურაცია უნდა გამზადდეს წინასწარ. უპირველეს

ყოვლისა, MPLS-ქსელში მყოფ ყველა მარშრუტიზატორზე გლობალური კონფიგურაციის რეჟიმიდან უნდა გააქტიურდეს MPLS-TE-ფუნქცია.

Local Label	Outgoing Label	Prefix or Tunnel Id	Bytes Label Switched	Outgoing interface	Next Hop
18	19	8.8.8.8/32	0	Et0/1	10.40.50.6
19	23	7.7.7.7/32	0	Et0/1	10.40.50.6
	24	7.7.7.7/32	0	Et0/0	10.40.50.14
20	24	6.6.6.6/32	0	Et0/1	10.40.50.6
	25	6.6.6.6/32	0	Et0/0	10.40.50.14
21	26	5.5.5.5/32	0	Et0/0	10.40.50.14
	22	5.5.5.5/32	0	Et1/1	10.40.50.30
22	Pop Label	4.4.4.4/32	0	Et0/0	10.40.50.14
23	Pop Label	3.3.3.3/32	0	Et1/1	10.40.50.30
24	Pop Label	2.2.2.2/32	0	Et0/1	10.40.50.6

ნახ. 32. MPLS-ნიშნულების ცხრილი.

შემდეგ თითოეულ ინტერფეისზე უნდა ჩაერთოს MPLS-TE და RSVP, რომელსაც უნდა მიეთითოს გამტარუნარიანობა კილობიტებში, რომელიც, როგორც ნახ. 33-დან ჩანს, არის 10 გბ=10000000 კბ.

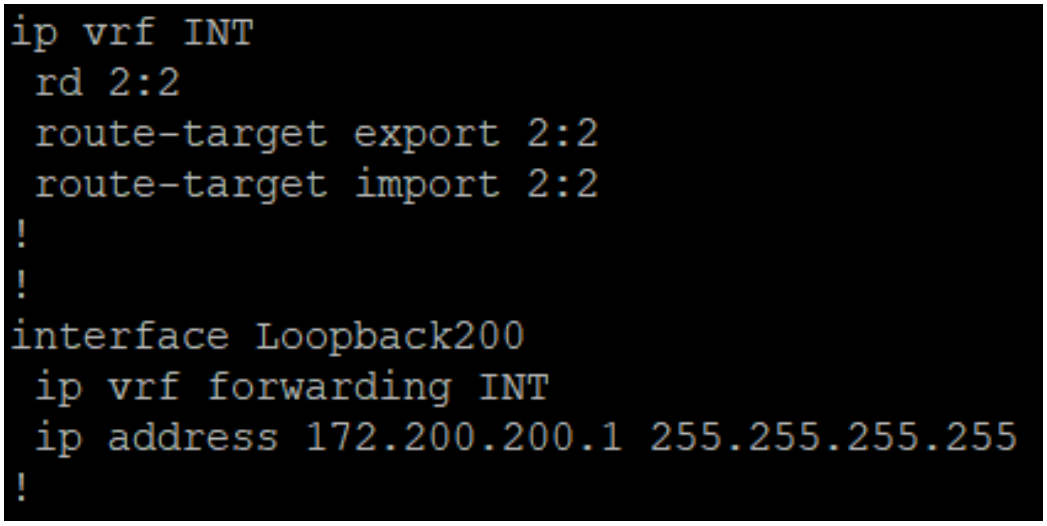
```
(config)#mpls traffic-eng tunnels
(config)#int e0/1
(config-if)#mpls traffic-eng tunnels
(config-if)#ip rsvp bandwidth 10000000
```

ნახ. 33. MPLS-TE-ს გააქტიურება

ამის შემდეგ შიდა მარშრუტიზაციის პროტოკოლში უნდა მოხდეს MPLS-TE-ს გააქტიურება, კერძოდ, მოცემულ შემთხვევაში OSPF-ში და ამავე დროს უნდა მიეთითოს მარშრუტიზატორის იდენთიფიკატორი (Router-id) (ნახ. 34).

ნახ. 34. MPLS-TE-ს გააქტიურება OSPF-ში.

თითოეულ იმ მარშრუტიზატორზე, სადაც ტერმინირდებიან ინტერნეტის მომხმარებლები, საჭიროა შესაბამისი vrf-ის შექმნა, რათა ინტერნეტის მომხმარებლები იზოლირებულნი იყვნენ მარშრუტიზაციის ცხრილში და არ ჰქონდეთ წვდომა გლობალური მარშრუტიზაციის ცხრილთან ან სხვა vrf-ებთან. კონკრეტულ vrf-ში ტრაფიკის სიმულაციისთვის წინამდებარე ნაშრომის მიზნების შესაბამისად შეიქმნა ასევე Loopback 200-ინტერფეისი და იგი გაიწერა შესაბამის vrf-ში (ნახ. 35).



ნახ. 35. VRF INT-ის კონფიგურაცია Loopback 200 ინტერფეისზე

მარშრუტიზაციის vrf INT-ცხრილში გამოჩნდება მხოლოდ შესაბამისი მარშრუტი(ები). ნახ. 36-ზე მხოლოდ ის მარშრუტი ჩანს, რომელიც განხილულ შემთხვევაშია მარშრუტიზატორზე გაწერილი.

```
Routing Table: INT
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override

Gateway of last resort is not set

      172.200.0.0/32 is subnetted, 3 subnets
C        172.200.200.1 is directly connected, Loopback200
```

ნახ. 36. VRF INT-მარშრუტები

ამის შემდგომ უკვე საჭიროა პროცესში BGP-პროტოკოლის ჩართვა, რათა ვირტუალური VPN-მარშრუტები გაიცვალოს IP/MPLS-სატრანსპორტო ქსელის მეშვეობით. შესაბამისი მარშრუტიზატორები ტრანსპორტის TCP-პროტოკოლის საშუალებით დაამყარებენ BGP-სესიებს. შესაბამისად, BGP-სესია შესაძლებელია დამყარდეს არა მხოლოდ ფიზიკურად ერთმანეთთან მიერთებულ მოწყობილობებს შორის, არამედ მათ შორის ლოგიკური კავშირის შემთხვევაშიც. ამისთვის მთავარი პირობაა ის, რომ მარშრუტიზატორებს ერთმანეთთან უნდა ჰქონდეთ IP-კავშირი, რაც წინამდებარე ნაშრომის ფარგლებში აწყობილ ქსელში განხორციელდა OSPF-პროტოკოლის საშუალებით.

განხილული სანიმუშო ქსელის შემთხვევაში უნდა შეიქმნას 2 iBGP VPNV4-სესია R1-სა და R8-ს და R1-სა და R6-ს შორის. ვინაიდან BGP-სესიას აქვს ლუპის საწინააღმდეგო მექანიზმი, რომლის მიხედვითაც ერთი AS-დან მიღებულ მარშრუტებს იმავე AS-ის სხვა მარშრუტიზატორს არ გაუგზავნის, ამიტომ საჭიროა ან ე. წ. Full Mesh-სესიების დაკონფიგურირება, რაც თითქმის წარმოუდგენელია, განსაკუთრებით, დიდ ქსელში, ან რომელიმე მარშრუტიზატორი უნდა წარმოადგენდეს რეფლექტორს. მოცემულ შემთხვევაში ყველა დანარჩენი მარშრუტიზატორი იქნება რეფლექტორ-კლიენტები და ისინი სესიას დაამყარებენ ამ კონკრეტულ

მარშრუტიზატორთან(ებთან). ეს მარშრუტიზატორები რომელიმე მათგანისგან მიღებულ მარშრუტიზაციის ცხრილებს გაუგზავნის ყველა დანარჩენ მარშრუტიზატორს. ნახ. 37-ზე ნაჩვენებ სურათზე მოყვანილია R1-როუტერზე iBGP-კონფიგურაციის შემთხვევა.

```
router bgp 1
 bgp router-id 1.1.1.1
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 neighbor 6.6.6.6 remote-as 1
 neighbor 6.6.6.6 update-source Loopback0
 neighbor 8.8.8.8 remote-as 1
 neighbor 8.8.8.8 update-source Loopback0
 !
 address-family ipv4
 exit-address-family
 !
 address-family vpnv4
  neighbor 6.6.6.6 activate
  neighbor 6.6.6.6 send-community extended
  neighbor 6.6.6.6 route-reflector-client
  neighbor 8.8.8.8 activate
  neighbor 8.8.8.8 send-community extended
  neighbor 8.8.8.8 route-reflector-client
 exit-address-family
 !
 address-family ipv4 vrf INT
  redistribute connected
 exit-address-family
```

ნახ. 37. iBGP-კონფიგურაცია ორგანიზაციის შიგნით VRF-ის გამოყენების შემთხვევაში

სამივე R1, R6 და R8 მარშრუტიზატორზე შესაბამისი კონფიგურაციის გაწერის შემდეგ მარშრუტიზაციის vrf INT-ცხრილი შეიცვლება, ვინაიდან მას დაემატება დინამიური ჩანაწერები (ნახ. 38).

```
Routing Table: INT
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override

Gateway of last resort is not set

      172.200.0.0/32 is subnetted, 3 subnets
C        172.200.200.1 is directly connected, Loopback200
B        172.200.200.6 [200/0] via 6.6.6.6, 1w3d
B        172.200.200.8 [200/0] via 8.8.8.8, 1w3d
```

ნახ. 38. VRF INT-ცხრილის დინამური ჩანაწერები

ერთ-ერთი მიმართულებით ტრეისის საშუალებით განხორციელებული გადამოწმებით შეიძლება იმაში დარწმუნება, რომ ტრაფიკი მოძრაობს vrf INT-ში და, შესაბამისად, ქსელში მოხდა ინტერნეტ-მომხმარებლების გამოყოფა, რაც ჩანს ნახ. 39-დან.

```
R1_LAB#traceroute vrf INT 172.200.200.8 source lo200 numeric
Type escape sequence to abort.
Tracing the route to 172.200.200.8
VRF info: (vrf in name/id, vrf out name/id)
  1 10.40.50.6 [MPLS: Labels 19/33 Exp 0] 2 msec 1 msec 2 msec
  2 172.200.200.8 2 msec *  2 msec
```

ნახ. 39. VRF INT–traceroute-ის შემოწმება

გარდა ინტერნეტ-სერვისისა, პროვაიდერები მომხმარებლებს სთავაზობენ სხვა მრავალფეროვან სერვისებს. ერთ-ერთი მათგანია ხმის გადაცემა ინტერნეტ-პროტოკოლის საფუძველზე (VoIP). VoIP-სერვისის უსაფრთხოებისთვის მისი იზოლირება აუცილებელია მომხმარებლების ქსელისგან. ამისთვის წინამდებარე ნაშრომში განხილულ სანიმუშო ქსელში აღნიშნულ მარშრუტიზატორებზე, ინტერნეტის vrf-ის გარდა, უნდა შეიქმნას vrf VOIP. მისი კონფიგურაცია vrf INT-ს კონფიგურაციის იდენტურია. ექსპერიმენტის ჩატარებისას R1, R6 და R8-მარშრუტიზატორებზე შეიქმნა vrf VOIP-პროვაიდერის რეგიონალური სერვერები მათი ერთმანეთთან კომუნიკაციისთვის. ტრაფიკის კონკრეტულ vrf-ში სიმულაციისთვის შეიქმნა, აგრეთვე, Loopback100-ინტერფეისი, რომელიც გაიწერა შესაბამის vrf VOIP-ში. გარდა ამისა, მარშრუტების შესახებ ინფორმაციის დინამიურად მისაღებად BGP-ში ექსპერიმენტისას vrf VOIP-ს გაუკეთდა რედისტრიბუცია ისევე, როგორც ეს იყო vrf INT შემთხვევაში (ნახ.40).

შედეგად, მარშრუტიზატორი უკვე დინამიურად მიიღებს ინფორმაციას vrf VOIP-ში გაწერილ ქსელების შესახებ (ნახ. 41).

```
ip vrf VOIP
 rd 1:1
 route-target export 1:1
 route-target import 1:1
interface Loopback100
 ip vrf forwarding VOIP
 ip address 172.17.17.1 255.255.255.255
router bgp 1
 !
 address-family ipv4 vrf VOIP
  redistribute connected
 exit-address-family
```

ნახ. 40.VRF VOIP -კონფიგურაცია

```
BGP table version is 18, local router ID is 1.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 1:1 (default for vrf VOIP)
 *>  172.17.17.1/32   0.0.0.0                  0         32768 ?
 *>i 172.17.17.6/32   6.6.6.6                  0    100      0 ?
 *>i 172.17.17.8/32   8.8.8.8                  0    100      0 ?
```

ნახ. 41. vrf VOIP-დინამიური ჩანაწერები

next-hop მისამართები vrf VOIP-შიც იგივეა, რაც იყო vrf INT-ს შემთხვევაში, რის გამოც R1-დან R8-ს მიმართულებით მარშრუტიზაციაც განხორციელდება ანალოგიურად (ნახ. 42).

```
R1_LAB#traceroute vrf VOIP 172.17.17.8 source lo100 numeric
Type escape sequence to abort.
Tracing the route to 172.17.17.8
VRF info: (vrf in name/id, vrf out name/id)
  1 10.40.50.6 [MPLS: Labels 19/34 Exp 0] 2 msec 1 msec 2 msec
  2 172.17.17.8 2 msec *  2 msec
```

ნახ. 22. VRF VOIP–traceroute-ის შემოწმება

მოცემულ ქსელში მარშრუტიზატორებზე გამოყენებულია ურთიერთდამოუკიდებელი მარშრუტიზაციის სამი (გლობალური, vrf INT

და vrf VOIP) ცხრილი. როგორც უკვე იყო აღნიშნული, ქსელები ერთმანეთისგან იზოლირებულნი არიან, რის გამოც ისინი ვერ შეძლებენ ერთმანეთთან წვდომას. ამ ეტაპზე ქსელებში გამართულია სტანდარტული მარშრუტიზაცია, თუმცა, უნდა აღინიშნოს, რომ ხმის ტრაფიკის გამოყოფისთვის საჭიროა წყაროზე დაფუძნებული მარშრუტიზაციის გამოყენებაც, რაც მიიღწევა MPLS-TE-ს საშუალებით. დასახული მიზნის მისაღწევად ხმის ტრაფიკზე ზემოქმედებისთვის ტრაფიკის ინჟინერიის გამოყენება უნდა მოხდეს მხოლოდ vrf VOIP-ისთვის. წინამდებარე ნაშრომში დასმული ერთ-ერთი ამოცანის განხორციელება, რომელიც მდგომარეობს ერთი vrf-ის ტრაფიკზე ზემოქმედებაში, სტანდარტული კონფიგურაციით ვერ მოხერხდება, ვინაიდან, როგორც უკვე იყო აღნიშნული, ორივე vrf-ის next-hop-მისამართი ერთი და იგივეა. ამიტომ ქსელს უნდა დაემატოს ახალი Loopback-ინტერფეისი, მაგალითად Loopback11, რომელიც მარშრუტიზაციის გლობალურ ცხრილში ისე იქნება ჩართული როგორც Loopback0 (ნახ. 43).

```
interface Loopback11
 ip address 11.1.1.1 255.255.255.255
 ip ospf 1 area 0
end
```

```
C        11.1.1.1 is directly connected, Loopback11
O        11.6.6.6 [110/21] via 10.40.50.14, 1w6d, Ethernet0/0
                  [110/21] via 10.40.50.6, 2d20h, Ethernet0/1
O        11.8.8.8 [110/21] via 10.40.50.6, 2d20h, Ethernet0/1
```

ნახ. 43. Loopback11-ის შექმნა დინამიური ჩანაწერებით

აღნიშნული ინტერფეისი უნდა იქნას გამოყენებული vrf VOIP-ის next-hop-სთვის, ვინაიდან vrf INT-სა და vrf VOIP-ს უნდა ქონდეთ სხვადასხვა next-hop-მისამართები, რაც ექსპერიმენტის ჩატარებისას იძლევა კონკრეტული vrf-სთვის წყაროზე დაფუძნებული მარშრუტიზაციის გამართვის საშუალებას. ამისთვის თითოეულ მარშრუტიზატორზე vrf VOIP-

ის კონფიგურაციაში უნდა გაიწეროს ბრძანება „bgp next-hop Loopback11“ (ნახ. 44).

```
ip vrf VOIP
 rd 1:1
 route-target export 1:1
 route-target import 1:1
 bgp next-hop Loopback11
```

ნახ. 44. Next-hop კონფიგურაცია vrf VOIP-სთვის.

აღნიშნული კონფიგურაციის ფორმირების შემდეგ აღმოჩნდება, რომ vrf VOIP-ში შემავალ ქსელებს შეეცვლებათ next-hop ისე, როგორც ეს იყო დაგეგმილი (ნახ. 45).

```
BGP table version is 24, local router ID is 1.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 1:1 (default for vrf VOIP)
 *>  172.17.17.1/32   0.0.0.0                  0         32768 ?
 *>i 172.17.17.6/32   11.6.6.6                 0    100      0 ?
 *>i 172.17.17.8/32   11.8.8.8                 0    100      0 ?
```

ნახ. 45. vrf-დინამიური ჩანაწერები შეცვლილი next-hop-ებით

შემდეგი განსხვავებული next-hop იძლევა უკვე ფორმირებული კონკრეტული vrf-ისთვის წყაროზე დაფუძნებული მარშრუტიზაციის MPLS-TE-ს დაკონფიგურების საშუალებას. ვინაიდან გლობალური კონფიგურაცია უკვე გაკეთებულია, ამიტომ დარჩენილია მხოლოდ კონკრეტული გვირაბის გამართვის პროცესი. იგი განხორციელდა R1-დან R8-მდე (Tunnel 1) და R1-დან R6-მდე (Tunnel 2) გვირაბებისთვის. გამართვის პირველ ეტაპზე გვირაბები დინამიურად აირჩევენ მარშრუტს და ჩაირთვებიან.

ნახ. 46-ზე ნაჩვენებ სურათზე ჩანს რომ გვირაბი არის მზად და მისი შესაბამისი მარშრუტი კანონიერია. ამ კონკრეტულ შემთხვევაში გამოიყენება დინამიურად არჩეული მარშრუტი path option 100, რომელიც ჯერჯერობით ერთადერთი არაა. ნახაზზე ნაჩვენებია, აგრეთვე, არჩეული მარშრუტის თითოეული hop, რაც ნიშნავს იმას, რომ გვირაბი მზად არის გამოსაყენებლად.

```
Name: R1_LAB_t1                          (Tunnel1) Destination: 8.8.8.8
  Status:
    Admin: up         Oper: up     Path: valid       Signalling: connected
    path option 100, type dynamic (Basis for Setup, path weight 20)

  Config Parameters:
    Bandwidth: 0        kbps (Global)  Priority: 7  7   Affinity: 0x0/0xFFFF
    Metric Type: TE (default)
    AutoRoute:  disabled  LockDown: disabled  Loadshare: 0        bw-based
    auto-bw: disabled
  Active Path Option Parameters:
    State: dynamic path option 100 is active
    BandwidthOverride: disabled  LockDown: disabled  Verbatim: disabled


  InLabel  :  -
  OutLabel : Ethernet0/1, 38
  RSVP Signalling Info:
       Src 1.1.1.1, Dst 8.8.8.8, Tun_Id 1, Tun_Instance 539
    RSVP Path Info:
      My Address: 10.40.50.5
      Explicit Route: 10.40.50.6 10.40.50.10 8.8.8.8
      Record   Route:   NONE
      Tspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
    RSVP Resv Info:
      Record   Route:   NONE
      Fspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
  Shortest Unconstrained Path Info:
    Path Weight: 20 (TE)
    Explicit Route: 10.40.50.6 10.40.50.10 8.8.8.8
```

ნახ. 46. დეტალური ინფორმაცია გვირაბის შესახებ

გვირაბში ტრაფიკი არ გაივლის მანამდე, სანამ მარშრუტიზაციის საშუალებით არ მოხდება შესაბამისი მითითების ფორმირება. იმის გამო, რომ გვირაბი სტანდარტულად დინამიურ მარშრუტიზაციას იყენებს, ამ ეტაპზე გვირაბში ტრაფიკის გაშვებას აზრი არ აქვს, რის გამოც “ip explicit-path name EXCLUDED-PATH” ბრძანების საშუალებით, უპირველეს ყოვლისა, იქმნება ალტერნატიული მარშრუტი, სადაც EXCLUDED-PATH მარშრუტის სახელია. აღნიშნული მარშრუტი, მისი პირველობისა და პრიორიტეტულობის გამო, უნდა მიუერთდეს გვირაბებს option 100-ზე დაბალი ნომრით. ამ შემთხვევაში გვირაბს მიუერთდება თავისუფალი მარშრუტი, რომელსაც ჯერჯერობით

არანაირი სტატიკური ინფორმაცია არ მიეწოდება. შესაბამისად თავისუფალი მარშრუტი არ გამოიყენება, რის გამოც მაინც დინამიური მარშრუტი იქნება გამოყენებული (ნახ. 47).

```
R1_LAB#show run int tu1
Building configuration...

Current configuration : 245 bytes
!
interface Tunnel1
 ip unnumbered Loopback0
 tunnel mode mpls traffic-eng
 tunnel destination 8.8.8.8
 tunnel mpls traffic-eng path-option 10 explicit name EXCLUDED-PATH
 tunnel mpls traffic-eng path-option 100 dynamic
 no routing dynamic
end

R1_LAB#show run int tu2
Building configuration...

Current configuration : 245 bytes
!
interface Tunnel2
 ip unnumbered Loopback0
 tunnel mode mpls traffic-eng
 tunnel destination 6.6.6.6
 tunnel mpls traffic-eng path-option 10 explicit name EXCLUDED-PATH
 tunnel mpls traffic-eng path-option 100 dynamic
 no routing dynamic
end
```

ნახ. 47. გვირაბში ახალი path-option-ის დამატება

ვინაიდან ჯერ გვირაბში არ მომხდარა ტრაფიკის გატარება, ამიტომ vrf VOIP-ტრაფიკი გაივლის ისევ სტანდარტული OSPF-მარშრუტით (ნახ. 48).

```
R1_LAB#traceroute vrf VOIP 172.17.17.8 sour lo100 numeric
Type escape sequence to abort.
Tracing the route to 172.17.17.8
VRF info: (vrf in name/id, vrf out name/id)
  1 10.40.50.6 [MPLS: Labels 22/34 Exp 0] 2 msec 2 msec 2 msec
  2 172.17.17.8 2 msec *  3 msec
```

ნახ. 48. vrf VOIP traceroute-ტრაფიკი გვირაბში ჩაშვებამდე

შემდგომ განხორციელდა ტრაფიკის ჩაშვება გვირაბში, რისთვისაც vrf VOIP-ის next-hop-მისამართი სტატიკურად გაიწერა შესაბამისი გვირაბისკენ (ნახ. 49).

```
ip route 11.6.6.6 255.255.255.255 Tunnel2 name To-R6
ip route 11.8.8.8 255.255.255.255 Tunnel1 name To-R8
```

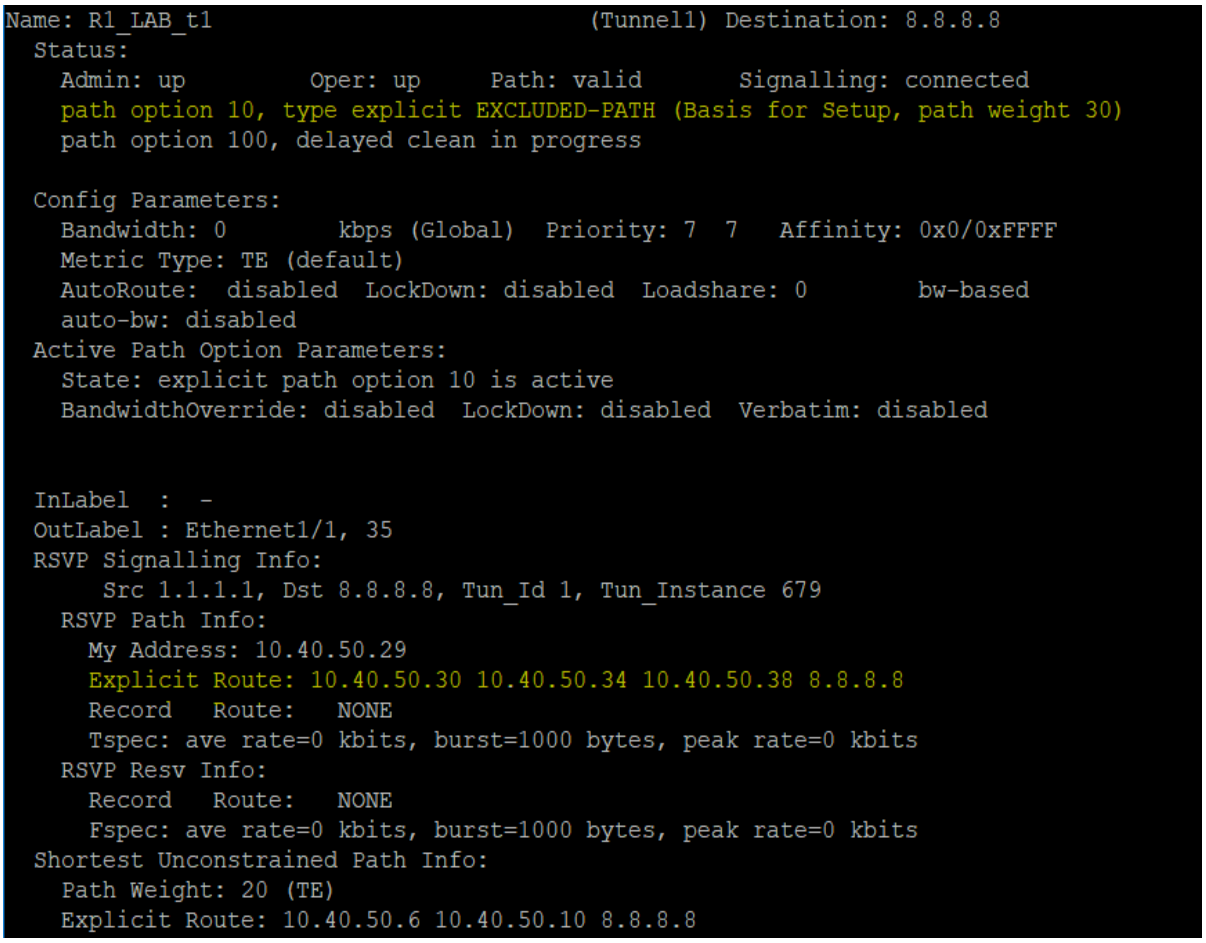
ნახ. 49. სტატიკური მარშრუტიზაცია და vrf VOIP-ტრაფიკის გვირაბში ჩაშვება

ექსპერიმენტის ჩატარებისას სატესტოდ vrf VOIP-ის ტრაფიკისთვის განხორციელდა მანიპულაცია იმ შემთხვევისთვის, როდესაც კონკრეტულ მარშრუტზე მონაცემთა გადაცემის არხს შეექმნება რაიმე პრობლემა. ამისთვის ჯერ EXCLUDED-PATH-ში უნდა დაემატოს ჩანაწერი „excluded-address 10.40.50.6“, რომელიც აღნიშნავს იმას, რომ მარშრუტიზაციისთვის არ უნდა მოხდეს R1-სა და R2-ს შორის შეერთების გამოყენება. აღნიშნული ჩანაწერის გაკეთების შემდეგ მარშრუტიზატორი გამოითვლის ყველაზე მოკლე მარშრუტს, თუმცა ამისთვის არ გამოიყენებს 10.40.50.6 მისამართს, რაც ჩანს ნახ. 50-ზე მოტანილ სურათზე.

```
Explicit Path name EXCLUDED-PATH:
    1: exclude-address 10.40.50.6
```

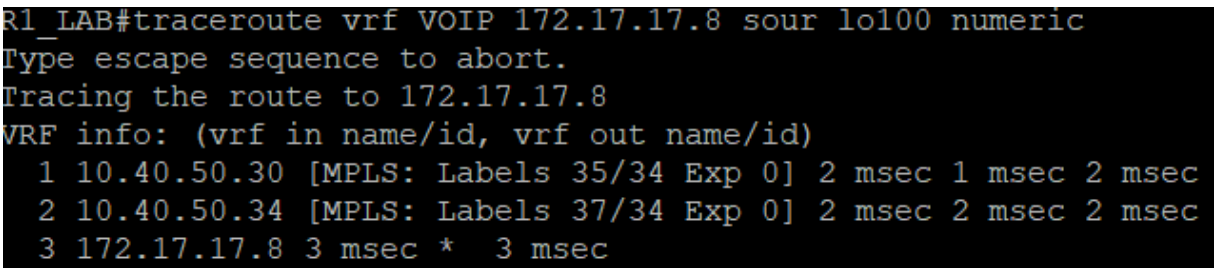
ნახ. 50. ჩანაწერის დამატება Explicit path-ში

აღწერილ პროცესის შემდეგ განხორციელდა „mpls traffic-eng reoptimize“ ბრძანების ფორმირება exec mode-ში იმისთვის, რომ შეტანილი ცვლილება აისახოს დაუყოვნებლივ. ნახ. 51-ზე წარმოდგენილ სურათზე ჩანს გვირაბში პრიორიტეტული გზის შეცვლა.



ნახ. 51. გვირაბის დეტალური ინფორმაცია excluded-address-ის დამატების შემდეგ

ტრეისის ხელახლა შემოწმების შედეგად, როგორც ნახ.52-ზე ნაჩვენებ სურათზე ჩანს, vrf VOIP-ტრაფიკი გვირაბში ჩაშვების შემდეგ მოძრაობს R1-R2 შეერთების გამოყენების გარეშე.



ნახ. 52. Vrf VOIP-ტრაფიკის შემოწმება მისი გვირაბში ჩაშვების შემდეგ

ჩატარებული ოპერაციების შედეგად ტრეისი შეიცვალა წინასწარ გაწერილი წესის შესაბამისად, რომლის მიხედვითაც R1-მა არ უნდა გამოიყენოს 10.40.50.6 IP-მისამართი.

ექსპერიმენტის ჩატარებისას vrf VOIP-ტრაფიკის მიმართულების ცვლილების შემოწმების მიზნით მოხდა მეორე IP 10.40.50.30-ის დამატება. აღნიშნული ქმედების შესაბამისი სურათი წარმოდგენილია ნახ. 53-ზე, საიდანაც ჩანს ის, რომ ამ შემთხვევაში მარშრუტიკვლავ იცვლება, კერძოდ კი არჩეული აღმოჩნდა ის მარშრუტი, რომელიც არაა გათვალისწინებული 10.40.50.6 და 10.40.50.30 მისამართებით.

```
Explicit Path name EXCLUDED-PATH:
    1: exclude-address 10.40.50.6
    2: exclude-address 10.40.50.30
R1_LAB#traceroute vrf VOIP 172.17.17.8 source lo100 numeric
Type escape sequence to abort.
Tracing the route to 172.17.17.8
VRF info: (vrf in name/id, vrf out name/id)
  1 10.40.50.14 [MPLS: Labels 16/34 Exp 0] 3 msec 1 msec 1 msec
  2 10.40.50.46 [MPLS: Labels 27/34 Exp 0] 1 msec 1 msec 2 msec
  3 172.17.17.8 1 msec *  2 msec
```

ნახ. 53. ახალი exclude-address-ის დამატება და traceroute

ექსპერიმენტისას მოხდა, აგრეთვე, პირველი exclude-address-ის ამოშლა და ტრაფიკის დაბრუნება შესაბამის მარშრუტზე. ამოშლის შესაბამისი ბრძანება ასახულია ნახ. 54-ზე.

```
R1_LAB(config)#ip explicit-path name EXCLUDED-PATH
R1_LAB(cfg-ip-expl-path)#no index 1
```

ნახ. 54. Excluded address-ის ამოშლის ბრძანება

ნახ. 55-ზე ნაჩვენებ სურათზე ჩანს, რომ ტრაფიკი კვლავ დაბრუნდა უმოკლეს მარშრუტზე, რომელიც აღარ იკრძალება პროტოკოლის მიერ.

```
R1_LAB#traceroute vrf VOIP 172.17.17.8 source lo100 numeric
Type escape sequence to abort.
Tracing the route to 172.17.17.8
VRF info: (vrf in name/id, vrf out name/id)
  1 10.40.50.6 [MPLS: Labels 35/34 Exp 0] 2 msec 1 msec 1 msec
  2 172.17.17.8 1 msec *  3 msec
```

ნახ. 55. vrf VOIP-ტრაფიკის მარშრუტის შემოწმება

4.4. ქსელის მონიტორგის დროს დაფიქსირებული ძირითადი პრობლემები

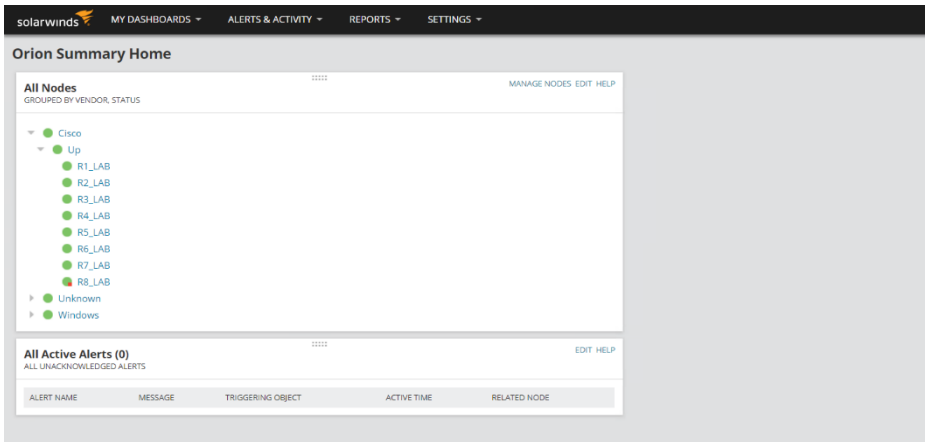
ქსელის ექსპლუატაციისას ერთ-ერთი ყველაზე გავრცელებული შემთხვევაა მონაცემთა გადაცემის არხის მაღალი დატვირთულობა, რომლის დროსაც გამტარში მონაცემების გადაცემის მოცულობა უახლოვდება გამტარის ფიზიკური შესაძლებლობის ზღვარს და, შესაბამისად, ამ დროს მონაცემთა პაკეტები ვეღარ გადაიცემა. კრიტიკულად შეიძლება ჩაითვალოს ზღვარი, რომლის დროსაც გამტარის დატვირთულობა აღწევს 90%-სა და მეტს. ფიქსირდება ისეთი შემთხვევები, როდესაც ფიზიკური დაზიანების გამო კონკრეტული ერთი ან მრავალი მიმართულებით ზიანდებიან მონაცემთა გადაცემის არხები, რაც შეიძლება იყოს სხვა არხების ტრაფიკით გადავსების მიზეზი. მონაცემთა გადაცემის არხის ტრაფიკით გადავსების გამომწვევი მიზეზი შეიძლება იყოს ტრაფიკის უეცარი მატება, სხვა ფიზიკური გამტარების დაზიანება, Ddos-შეტევა და სხვა. მსგავს შემთხვევებში MPLS–TE-ს საშუალებით შესაძლებელია მონაცემთა ტრაფიკის არიდება კონკრეტულ მარშრუტიზატორებს შორის კრიტიკულად დატვირთული გადაცემის არხებისთვის. ამ დროს ლოგიკურ ჯგუფად შეიძლება გამოიყოს ისეთი შემთხვევები, როგორებიცაა ქსელური მოწყობილობის ცენტრალური პროცესორის მაღალი დატვირთულობა, მაღალი ტემპერატურა ან მეხსიერების გადავსება. შესაბამის შემთხვევებში საჭიროა ქსელში სათანადო ცვლილებების გატარება, კერძოდ კი MPLS–TE-ს საშუალებით ტრაფიკის მარშრუტიდან კონკრეტული კვანძის გამოთიშვა.

4.5. ექსპერიმენტული ქსელის მონიტორინგი

როგორც აღნიშნული იყო, თანამედროვე კომპიუტერული ქსელების მუშაობა წარმოუდგენელია მონიტორინგის სისტემის გარეშე. სწორედ რომ მათი საშუალებითაა შესაძლებელი ქსელში ინციდენტების დაფიქსირება და მათზე რეაგირება.

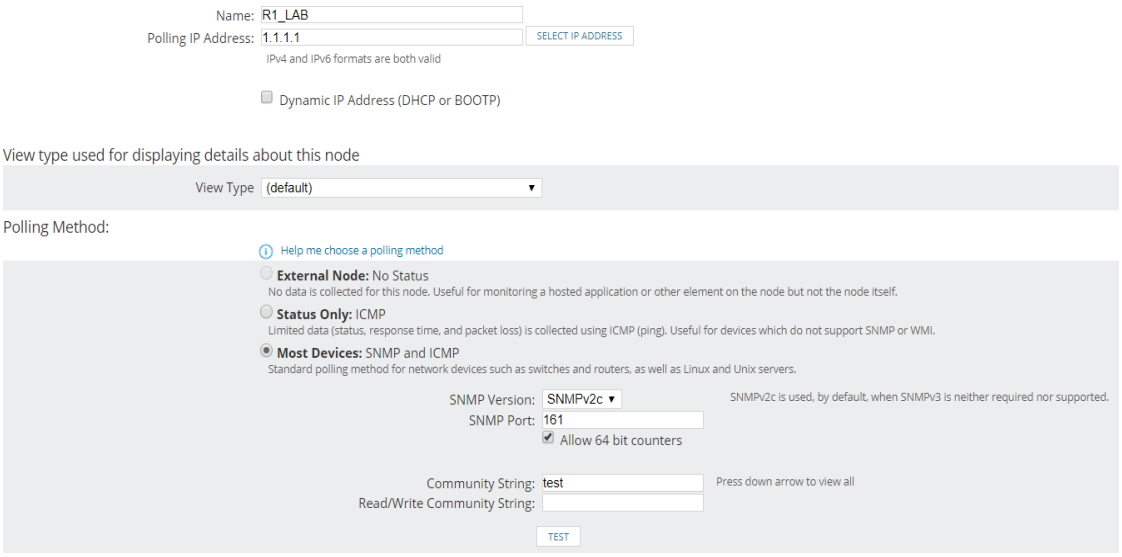
წარმოდგენილ ნაშრომში ექსპერიმენტული მოდელირებისთვის გამოყენებულია მონიტორინგის Solarwinds-NPM-სისტემა, რომელიც დაინსტალირდა ESXI 6.0 გარემოში მომუშავე შემდეგი მონაცემების მქონე ვირტუალურ პლატფორმა Windows Server 2012 R2-ზე: 50GB SSD; 8GB RAM; 6 intel Xeon 2.4 Ghz.

ნახ. 56-ზე ნაჩვენებია მონიტორინგის პროგრამის ვებ-პანელის მთავარი გვერდი. იგი სრულიად კონფიგურირებადია, ვინაიდან შესაძლებელია სხვადასხვა სახის ისეთი ინტერაქტიული ჩანართების გამოტანა, როგორიცაა მაგალითად: ყველა აქტიური ალერტი; ყველა მოწყობილობა სტატუსების მიხედვით (ჩართული, გათიშული); მონაცემთა გადაცემის ყველაზე დატვირთული 25 არხი და სხვა. ნახაზზე ასევე ჩანს ინფორმაცია იმ მარშრუტიზატორების შესახებ, რომლებიც მონიტორინგის სისტემაშია დამატებული. ქვემოთ ნაჩვენები იქნება მოწყობილობის მონიტორინგზე აყვანის პროცესი.



ნახ. 56. მონიტორინგის NPM-სისტემის მთავარი გვერდი

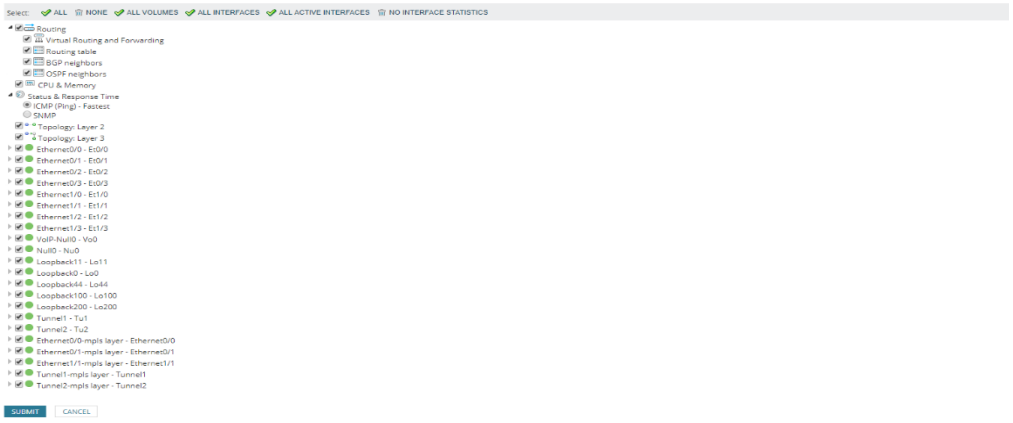
ქსელური მოწყობილობის მონიტორინგის პროგრამაში დასამატებლად სისტემის ადმინისტრატორი პროგრამაში შესაბამის ადგილზე გადასვლით ერთმანეთის მიმდევრობით ახორციელებს ქმედებებს „კვანძების მართვა“ და „კვანძის დამატება“ (Manage Nodes და Add a Node). ამის შემდეგ საჭიროა მოწყობილობის IP-მისამართის შეყვანა, რომელიც წვდომადია მონიტორინგის სერვერიდან. მონიტორინგის პროგრამაში შესაძლებელია მართვადი მოწყობილობა დაემატოს ICMP-პროტოკოლის მეშვეობით, რომლის დროსაც Ping-ის საშუალებით მოხდება მოწყობილობის მარტივი მონიტორინგი და სისტემაში იქნება ინფორმაცია მხოლოდ მოწყობილობის სტატუსის (წვდომადია თუ არაწვდომადი) შესახებ. მოწყობილობის სრულყოფილად მონიტორინგისთვის საჭიროა მასში იყოს დაკონფიგურირებული პროგრამის საჭირო SNMP client-და communit String-ატრიბუტები. ამის შემდეგ SNMP-პროტოკოლის საშუალებით ხდება პროგრამაში მართვადი მოწყობილობის დამატება (ნახ. 57).



ნახ. 57. მონიტორინგის პროგრამაში ქსელური მოწყობილობის დამატება

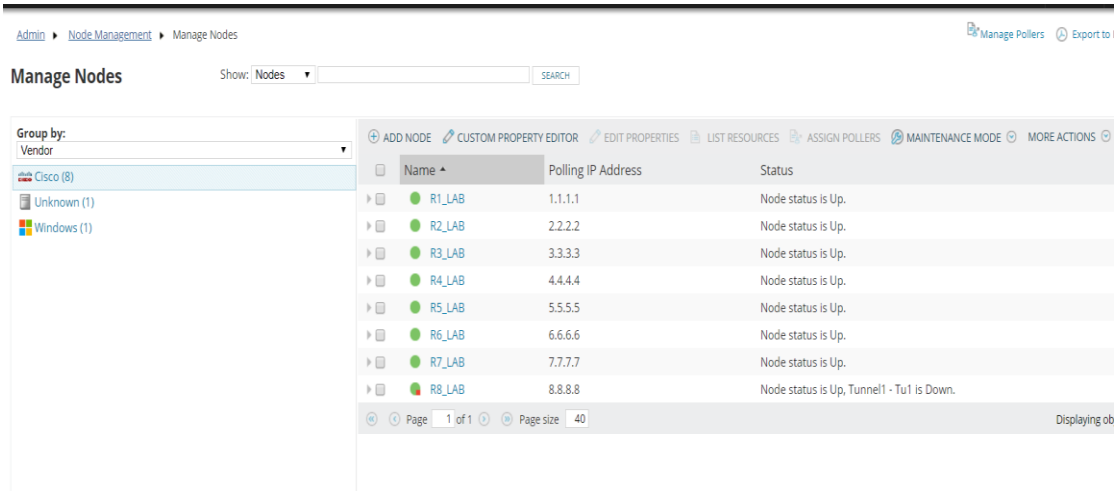
SNMP-ტესტის გავლის შემდეგ მონიტორინგის სისტემა SNMP-პროტოკოლის საშუალებით მართვადი მოწყობილობიდან ამოიღებს

ინფორმაციას პროცესორის, მარშრუტიზაცის პროტოკოლების, ინტერფეისებისა და სხვა რესურსების შესახებ (ნახ. 58).

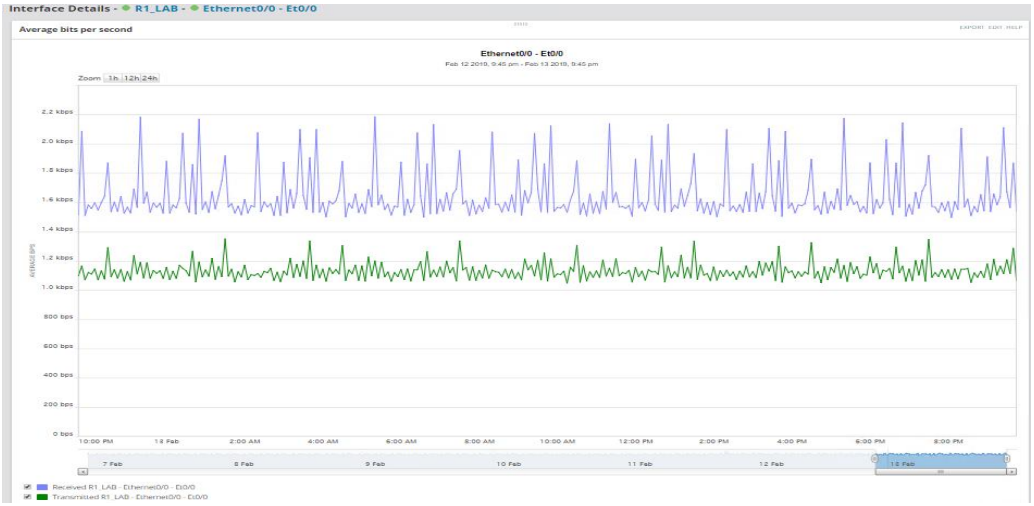


ნახ. 58. R1 მარშრუტიზატორის ფიზიკური და ვირტუალური ინტერფეისები, CPU, MEMORY

ნახ. 59-ზე ნაჩვენებია მოწყობილობების მართვის გვერდის სურათი მონიტორინგზე აყვანილი 8 მარშრუტიზატორის შემთხვევაში. ინტერფეისის დატვირთვის გრაფიკული გამოსახულების ვიზუალიზაციის მიზნით ჯერ შეირჩა კონკრეტული მარშრუტიზატორი, შემდეგ კი სასურველი ინტერფეისი (ნახ. 60).



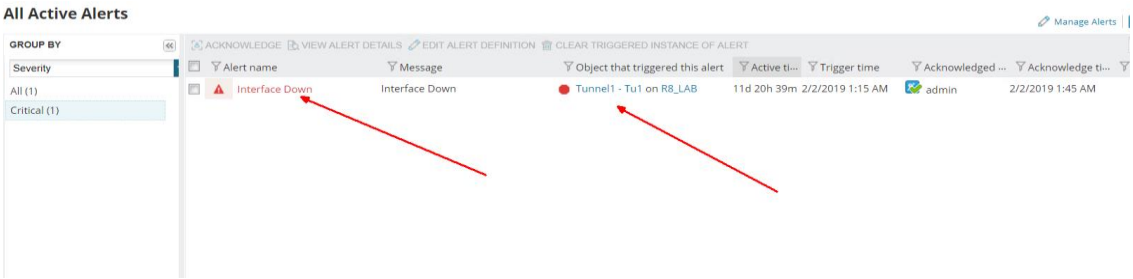
ნახ. 59. მოწყობილობების მართვის გვერდი



ნახ. 60. მარშრუტიზატორის ფიზიკური ინტერფეისის დატვირთვის გრაფიკი

4.6. გაფრთხილების კონფიგურაცია

მონიტორინგის სისტემაში შესაძლებელია განგაშის შეტყობინებების სხვადასხვა ტიპის კონფიგურაციის შექმნა, რაც უზრუნველყოფს იმას, რომ სისტემა დააფიქსირებს მითითებულ ზღვრულ მნიშვნელობაზე გადაცდენას, მოიმოქმედებს წინასწარ გაწერილ გარკვეულ (ელ-ფოსტაზე შეტყობინების გაგზავნა, სხვადასხვა ბრძანების შესრულება, მოწყობილობის გადატვირთვა და სხვა) პროცედურას. სანიმუშოდ ნახ. 61-ზე ნაჩვენებია ისეთი შემთხვევის სურათი, როდესაც ექსპერიმენტის ჩატარების მომენტისთვის გათიშული იყო R8-ინტერფეისის გვირაბი Tunnel 1.



ნახ. 61. განგაშის სიგნალების მართვის გვერდი

ჩატარებული ექსპერმინეტის ფარგლებში ქსელის მონიტორინგის სიმულაციისთვის შეიქმნა ის ძირითადი განგაშის მექანიზმები, რომლებიც ყველაზე ხშირად ფიქსირდება ქსელში. მათ შორისაა მონაცემთა გადაცემის არხის გადავსება, პროცესორის მაღალი დატვირთვა, ინტერფეისის გათიშვა და ქსელური მოწყობილობის გათიშვა (ნახ. 62).

UTILIZATION	ON	UTILIZATION	Interface	LOG
High CPU	ON	High CPU	Node	2 actions
Node	ON	Node ${NodeName} is down	Node	2 actions
Interface Down	ON	Interface Down	Interface	2 actions

ნახ. 62. მონიტორინგის სისტემაში დაკონფიგურირებული განგაშის ფორმები

საბოლოოდ შეიქმნა მონიტორინგის სისტემა, რომელშიც ექსპერიმენტის ჩატარებისას დამატებული იქნა სიმულირებული ქსელის მარშრუტიზატორები. გარდა ამისა, შეიქმნა განგაშის მექანიზმები პრობლემის დაფიქსირების შესახებ, რომლებიც იძლევიან ქსელში დაფიქსირებული ხარვეზების დროული აღმოჩენის შესაძლებლობას.

თავი 5. ქსელის მონიტორინგისა და ავტომატიზაციის ალგორითმი

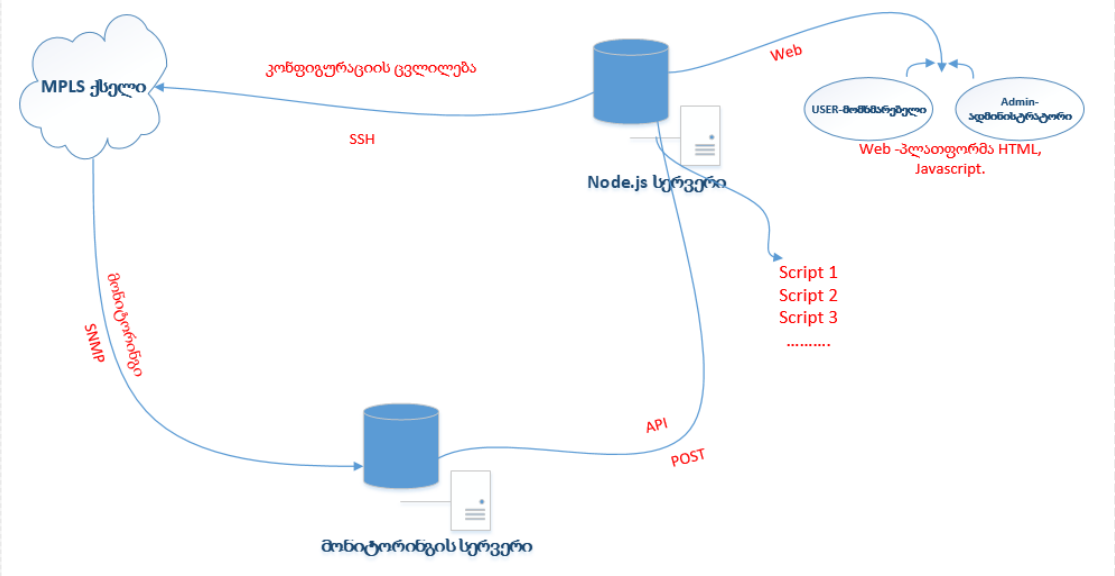
5.1. ავტომატიზაციის ალგორითმი

როგორც ცნობილია, ქსელის უწყვეტი და ხარისხიანი მუშაობისთვის საჭიროა მისი მუდმივი მონიტორინგი გარკვეული ინციდენტების დროს კონფიგურაციის მყისიერი ცვლილების მიზნით, რათა თავიდან იქნეს აცილებული ის შედეგები, რაც შეიძლება მოყვეს ქსელში დაფიქსირებულ ხარვეზებს.

წინამდებარე ნაშრომში დასმული ამოცანების განხორციელების მიზნით შედგენილია პაკეტური საკომუტაციო ქსელის ალგორითმის შესაბამისი პროგრამა და მისი რეალიზაციის ლოგიკური მოქმედებების სქემა, რომელიც ურთიერთქმედებაშია ქსელთან და მონიტორინგის სისტემასთან. დამუშავებული ალგორითმის პროგრამულ ნაწილს შეიძლება ეწოდოს ქსელის ავტომატური კონფიგურაციის სისტემა (Network Auto Configuration System - NACS).

თავდაპირველად მონიტორინგის სისტემა ქსელში მომხდარი ნებისმიერი სახის (უტილიზაცია, პროცესორის მაღალი დატვირთვა, ქსელური მოწყობილობის გათიშვა და სხვა) ინციდენტის დაფიქსირების შემდგომ node.js-პლატფორმაზე გამართულ NACS-ს უგზავნის HTTP-POST/GET-შეტყობინებას. NACS-პროგრამა, შეტყობინების შინაარსიდან გამომდინარე, SSH-პროტოკოლის საშუალებით მოახდენს მარშრუტიზატორის კონფიგურაციის გადამოწმებას და, საჭიროების შემთხვევაში, მის ცვლილებას. საბოლოოდ ცვლილებების შესახებ მონაცემები შეინახება Mysql-ბაზაში მათი შემდგომი ანალიზისთვის. NACS - სერვერს ექნება ვებ-ბრაუზერზე დაფუძნებული მომხმარებლისა და ადმინისტრატორის გარემო. მომხმარებლის User-გარემოში შესაძლებელი იქნება მონაცემთა ბაზაში დაფიქსირებული კონფიგურაციის ცვლილებებისა

და შესრულებული ქმედებების შემოწმება და სტატისტიკების მოძიება. ადმინისტრატორის გარემოდან შესაძლებელი უნდა იყოს ქსელის ელემენტების, განსახორციელებელი მოქმედებების სცენარებისა და პარამეტრების ისეთი ცვლილებები, როგორიცაა, მაგალითად, მარშრუტიზატორის დამატება, მარშრუტიზატორის IP-მისამართის ცვლილება ან მარშრუტიზატორის ექსპლუატაციიდან ამოღება და მისი გარკვეული კონფიგურაციის ცვლილება. გარდა ამისა, შესაძლებელი უნდა იყოს გვირაბის შექმნისა და წაშლის ბრძანებების ფორმირება. ნახ. 63-ზე წარმოდგენილია მონიტორინგის დამუშავებული ალგორითმის რეალიზაციის სქემა.



ნახ. 63. მონიტორინგის ალგორითმის სქემა

შემუშავებული NACS-სისტემის ლოგიკური კავშირები შესაძლოა დაიყოს შემდეგნაირად:

1. მონიტორინგის სერვერიდან POST-მეთოდით შეტყობინებების მიღება node.js-ზე. ამ შემთხვევაში Web-სერვერი იღებს შეტყობინებას ქსელური ინციდენტების შესახებ;

2. node.js-დან SSH-პროტოკოლის საშუალებით მარშრუტიზატორების მართვა და ცვლილებების ინფორმაციის ბაზაში დაფიქსირება;
3. node.js-ზე web-სერვერის გამართვა და მომხმარებლისა და ადმინისტარტორის პანელების შექმნა. იგი, თავის მხრივ, შეიძლება დაიყოს შემდეგნაირად:

3.1. პროგრამირების javascript და HTML ენაზე შესრულებულ მომხმარებლის WEB-გარემო იძლევა შესრულებული ქმედებებისა და სტატისტიკის შემოწმებისა და მონიტორინგის საშუალებას.

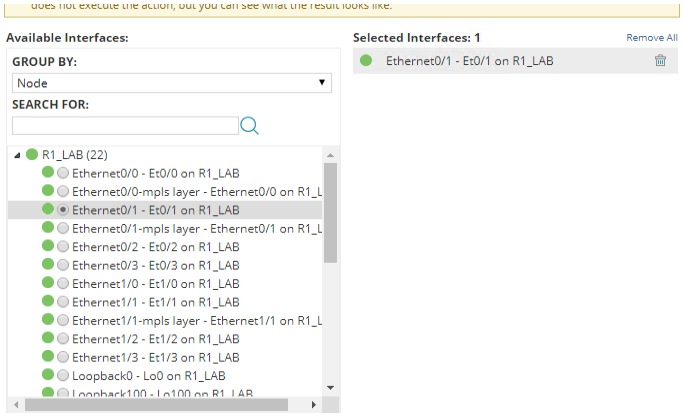
3.2. პროგრამირების javascript და HTML-ენაზე შესრულებული ადმინისტრატორის WEB-გარემო იძლევა კონფიგურაციის სცენარებისა და ქსელის კომპონენტების პარამეტრების ცვლილების საშუალებას [44].

5.2. შემუშავებული ალგორითმის მოდელირება და მისი რეალიზაციის შედეგები

წინამდებარე ნაშრომში შემუშავებული და აღწერილი ალგორითმის მოდელირება განხორციელდა ქსელის ემულატორის EVE-NG-ის გამოყენებით. ექსპერიმენტი ჩატარდა IP-MPLS-ქსელის 8 კვანძზე მათ მარშრუტიზატორებზე შექმნილი OSPF, MPLS-TE, LDP და RSVP-პროტოკოლების შესაბამისი კონფიგურაციით. VoIP-ტრაფიკის Internet-ტრაფიკისგან იზოლაციის მიზნით გამოყენებულ იქნა vrf-ტექნოლოგია. მოდელირებულ IP-MPLS-ქსელში პრობლემების დაფიქსირებისა და იმიტაციისთვის გამოყენებულ იქნა მონიტორინგის NPM-სისტემა. დამუშავებული ალგორითმის ეფექტურობის დასადგენად დაიგეგმა მონიტორინგის სისტემიდან შემდეგი ოთხი ტიპის განგაშის შესახებ შეტყობინების გაგზავნის სცენარი:

1. შეტყობინება მონაცემთა გადაცემის არხის გადავსების შესახებ. ნახ. 64-ზე წარმოდგენილ სურათზე ნაჩვენებია მონიტორინგის გარემო, საიდანაც

მოხდა R1-ის მონაცემთა გადაცემის 0/1 ნომრის მქონე არხის ინტერფეისის გადავსების სიმულაცია.



ნახ. 64. არხის გადავსების პროცესის სიმულაცია

აღნიშნული სურათიდან ჩანს, რომ Node.Js-პლატფორმაზე გამართული და ნაშრომის მიზნებისთვის დამუშავებული პროგრამული ალგორითმი (იხ. დანართი) მონიტორინგის სერვერიდან R1-ის 0/1 ინტერფეისზე იღებს შეტყობინებას ტრაფიკის გადატვირთვის შესახებ. ნახ. 65-ზე მოყვანილია აღნიშნული შეტრობინების მიღებისა და დამუშავების პროცესის ამსახველი სურათი.

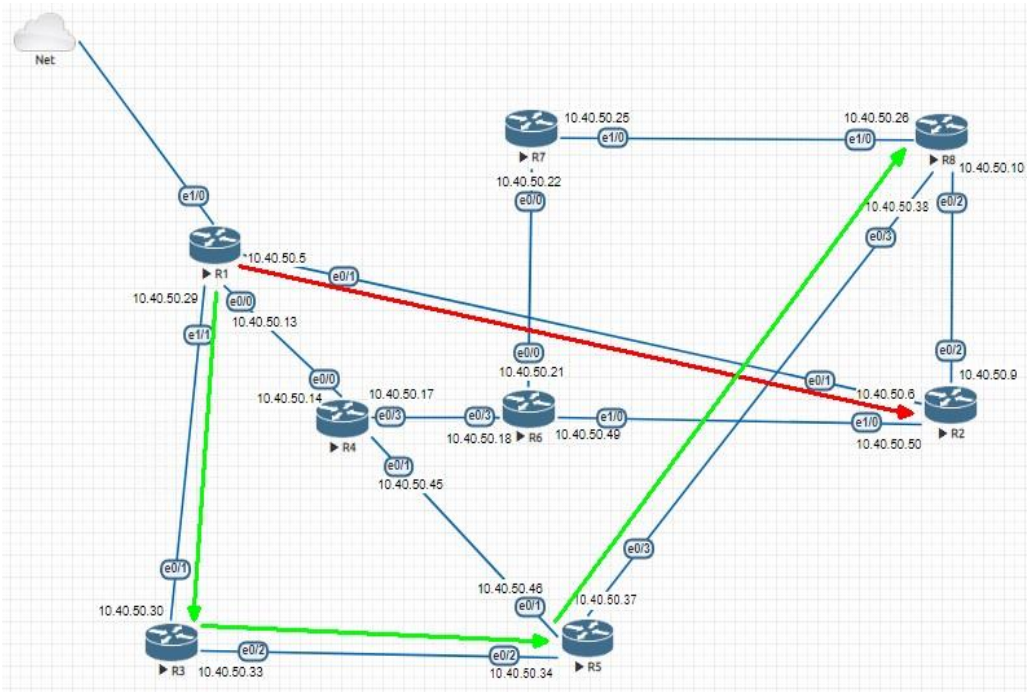
ნახ. 65. სერვერის მიერ შეტყობინების მიღება და დამუშავება

ნახ. 66-ზე ნაჩვენებია R1-დან R8-მარშრუტიზატორისკენ მიმართული გვირაბის კონფიგურაცია, სადაც ჩანს რომ R1-0/1 ინტერფეისი შეიზღუდა, რის გამოც იგი აღარ ფიგურირებს გვირაბში გადაცემული ტრაფიკის მარშრუტში. შესაბამისად, ხმოვანი მონაცემების გადაცემის vrf VOIP-ტრაფიკი უკვე მოძრაობს უსაფრთხო მარშრუტით.

```
R1_LAB#show ip explicit-paths name EXCLUDED-PATH
PATH EXCLUDED-PATH (strict source route, path complete, generation 47)
    1: exclude-address 10.40.50.14
R1_LAB#
R1_LAB#
  InLabel  :  -
  OutLabel : Ethernet1/1, 16
  RSVP Signalling Info:
       Src 1.1.1.1, Dst 8.8.8.8, Tun_Id 1, Tun_Instance 38
    RSVP Path Info:
      My Address: 10.40.50.29
      Explicit Route: 10.40.50.30 10.40.50.34 10.40.50.38 8.8.8.8
      Record   Route:   NONE
      Tspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
    RSVP Resv Info:
      Record   Route:   NONE
```

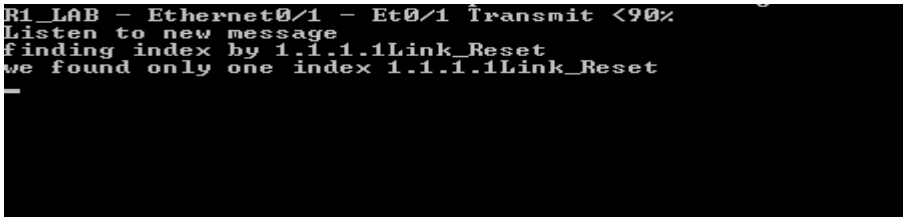
ნახ. 66. დეტალური ინფორმაცია გვირაბში ტრაფიკის გადაცემის შესახებ

ნახ. 67-ზე წარმოდგენილ სქემაზე მწვანე ისრით ნაჩვენებია R1-დან R8-სკენ მოძრავი vrf VOIP-ტრაფიკის მარშრუტი. გვირაბის ახალი მარშრუტი შედგა OSPF-პროტოკოლის უმოკლესი გზის მექანიზმის გამოყენებით. შესაბამისად R1-დან R8-სკენ, აღნიშნული შეზღუდვიდან გამომდინარე, ოპტიმალური აღმოჩნდა მარშრუტი R1->R3->R5->R8. ნახაზზე წითელი ისრით კი ნაჩვენებია პრობლემური (არაოპტიმალური) მარშრუტი.



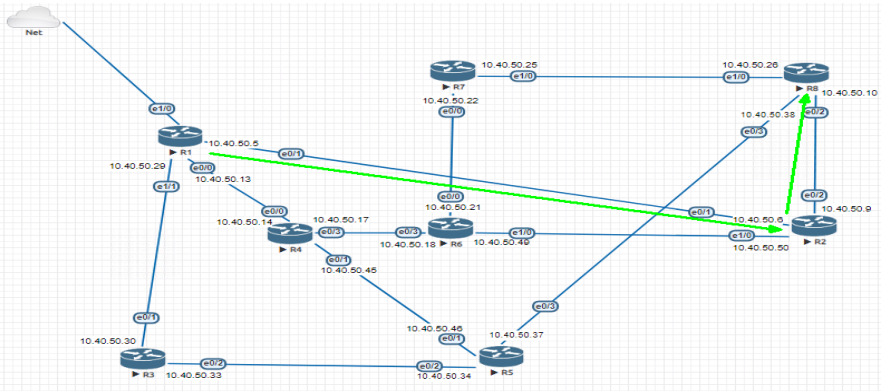
ნახ. 67 vrf VOIP-ტრაფიკის მოძრაობის სქემა

2. მონაცემთა გადაცემის არხში ტრაფიკის მისაღებ დონეზე დაბრუნების შესახებ შეტყობინება და შემუშავებული ალგორითმით გათვალისწინებული შემდგომი ქმედება. ამ შემთხვევაში მონიტორინგის სერვერზე გაკეთდა R1-ის მონაცემთა გადაცემის 0/1 ნომრის მქონე არხის გადავსების პრობლემის მოხსნის იმიტაცია და გაიგზავნა შესაბამისი შეტყობინება პროგრამულ ალგორითმის სერვერზე. ნახ. 68-ზე ნაჩვენებია სერვერის მიერ შეტყობინების მიღებისა და დამუშავების დაწყების ამსახველი სურათი.



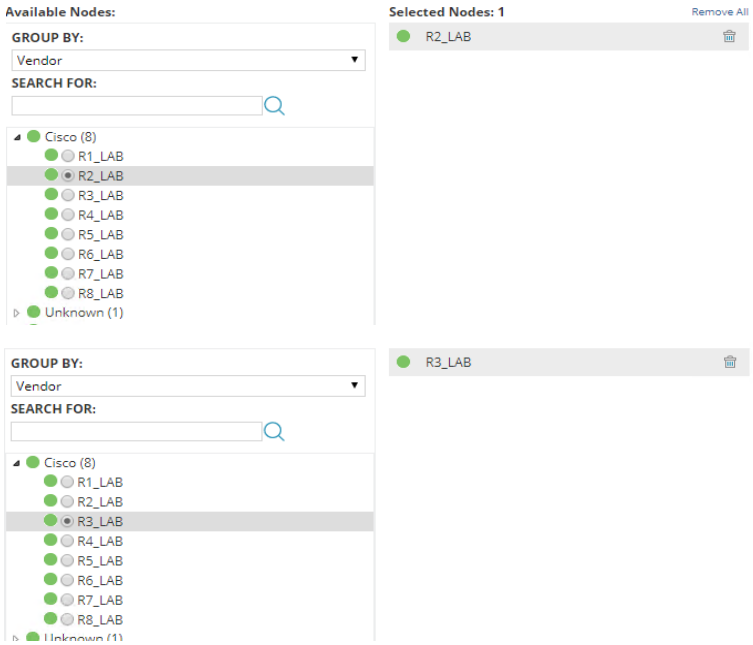
ნახ. 68. NACS-ის მიერ შეტყობინების მიღება და დამუშავება

R1-მარშუტიზატორზე 0/1 ინტერფეისის გადავსების პრობლემის მოგვარების შემთხვევაში აღნიშნული ინტერფეისის შეზღუდვა მოიხსნება და MPLS-TE ხელახლა გამოთვლის მარშრუტს წამოდგენილი სქემის მიხედვით. ამ შემთხვევაში R1-დან R8-მდე უმოკლესი გზაა R1-R2-R8 (ნახ. 69).



ნახ. 69. vrf VoIP-ტრაფიკის მოძრაობის მიმართულების სურათი პრობლემის მოგვარების შემდგომ

3. შეტყობინება ზემოთ წარმოდგენილ ქსელის R2 და R3 მარშრუტიზატორებზე არასტაბილური კავშირის შემთხვევასთან ან/და მათი აპარატურული ხარვეზების წარმოქმნასთან ასოცირებული ისეთი პრობლემების შესახებ, როგორებიცაა: ცენტრალური პროცესორის მაღალი დატვირთვა, აპარატურული მეხსიერების გადავსება, ტემპერატურის მომატება კრიტიკულ ზღვარს ზემოთ და ქსელურ კვანძთან არასტაბილური კავშირი. მოდელირების შესაბამისი სურათი პროცესორის გადატვირთვის შემთხვევაში წარმოდგენილია ნახ. 70-ზე.



ნახ. 70. ქსელური R2 და R3 კვანძების პროცესორის გადატვირთვის პრობლემის სიმულაცია

ნახ. 71-ზე ნაჩვენებია შემთხვევა, როდესაც პროგრამული ალგორითმის სერვერი ცენტრალური პროცესორის გადატვირთვის შესახებ შეტყობინებას იღებს R2 და R3-კვანძებზე, რის შემდეგაც ალგორითმი იწყებს მოქმედებებს.

```
R2_LAB 2.2.2.2 High CPU >90%
[ '1.1.1.1', '6.6.6.6', '8.8.8.8' ]
Listen to new message
finding exclude-address in 1.1.1.1CPU_Trigger
finding exclude-address in 8.8.8.8CPU_Trigger
finding exclude-address in 6.6.6.6CPU_Trigger
find result is true in 1.1.1.1 and execute command CPU_Trigger
find result is true in 8.8.8.8 and execute command CPU_Trigger
find result is true in 6.6.6.6 and execute command CPU_Trigger

R3_LAB 3.3.3.3 High CPU >90%
[ '1.1.1.1', '6.6.6.6', '8.8.8.8' ]
Listen to new message
finding exclude-address in 1.1.1.1CPU_Trigger
finding exclude-address in 8.8.8.8CPU_Trigger
finding exclude-address in 6.6.6.6CPU_Trigger
find result is false in 1.1.1.1 and execute command CPU_Trigger
find result is false in 8.8.8.8 and execute command CPU_Trigger
find result is false in 6.6.6.6 and execute command CPU_Trigger
```

ნახ. 71. NACS-ის მიერ შეტყობინების მიღება და დამუშავება

ალგორითმის ამუშავების შედეგად განხილულ შემთხვევაში ექსპერიმენტულ ქსელში vrf VOIP-გვირაბის მარშრუტის გამოთვლისას მონაწილეობენ პრობლემური R2 და R3 კვანძები, რაც ასახულია ნახ. 72-სა და ნახ. 73-ზე.

```
R8_LAB#show ip explicit-paths name EXCLUDED-PATH
PATH EXCLUDED-PATH (strict source route, path complete, generation 4)
    1: exclude-address 3.3.3.3
    2: exclude-address 2.2.2.2
R8_LAB#
R6_LAB#show ip explicit-paths name EXCLUDED-PATH
PATH EXCLUDED-PATH (strict source route, path complete, generation 6)
    1: exclude-address 3.3.3.3
    2: exclude-address 2.2.2.2
R6_LAB#
R1_LAB#show ip explicit-paths name EXCLUDED-PATH
PATH EXCLUDED-PATH (strict source route, path complete, generation 11)
    1: exclude-address 3.3.3.3
R1_LAB#
R1_LAB#show ip explicit-paths name EXCLUDED-PATH
PATH EXCLUDED-PATH (strict source route, path complete, generation 12)
    1: exclude-address 3.3.3.3
    2: exclude-address 2.2.2.2
R1_LAB#
```

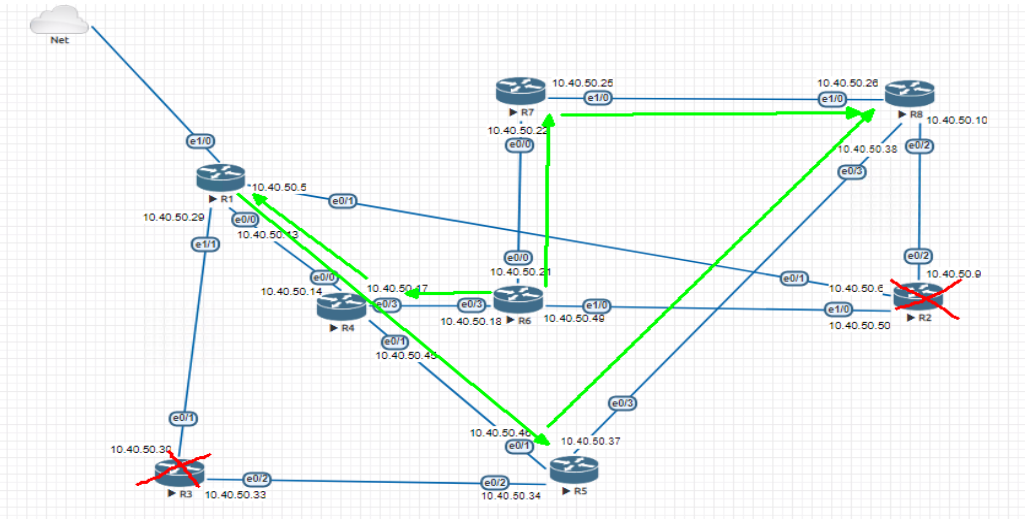
ნახ. 72. vrf VOIP-გვირაბისთვის R2 და R3-მარშრუტიზატორების შეზღუდვა

R2 და R3 მარშრუტიზატორების ცენტრალური პროცესორის გადატვირთვის შესახებ შეტყობინების მიღების შემდეგ vrf VOIP-ის მოქმედებაში არსებულ მარშრუტიზატორებზე ჩართული MPLS-TE გამოთვლის ახალ უსაფრთხო მარშრუტებს (ნახ. 73).

```
 OutLabel : Ethernet1/0, 17
 RSVP Signalling Info:
      Src 8.8.8.8, Dst 6.6.6.6, Tun_Id 1, Tun_Instance 35
   RSVP Path Info:
     My Address: 10.40.50.26
     Explicit Route: 10.40.50.25 10.40.50.21 6.6.6.6
OutLabel : Ethernet0/0, 18
RSVP Signalling Info:
     Src 1.1.1.1, Dst 8.8.8.8, Tun_Id 1, Tun_Instance 44
  RSVP Path Info:
    My Address: 10.40.50.13
    Explicit Route: 10.40.50.14 10.40.50.46 10.40.50.38 8.8.8.8
    Record   Route:   NONE
RSVP Signalling Info:
     Src 6.6.6.6, Dst 8.8.8.8, Tun_Id 1, Tun_Instance 16
  RSVP Path Info:
    My Address: 10.40.50.21
    Explicit Route: 10.40.50.22 10.40.50.26 8.8.8.8
    Record   Route:   NONE
```

ნახ. 73. გვირაბის კონფიგურაცია

ნახ. 74-ზე წარმოდგენილია ისეთი შემთხვევა, როდესაც R2 და R3 პროცესორები გადატვირთულია და ისინი აღარ გამოიყენებიან vrf VOIP-ტრაფიკის გადაცემისათვის.



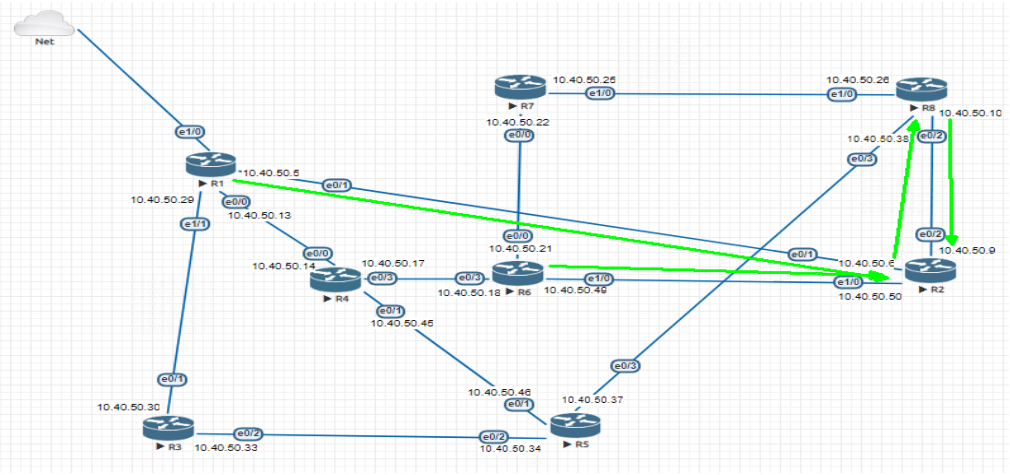
ნახ. 74. R2 და R3 კვანძების პრობლემის დროს vrf VOIP ტრაფიკის გადაადგილება

4. შეტყობინება R2 და R3 მარშრუტიზატორების ცენტრალური პროცესორის ნორმალური დატვირთულობის შესახებ, რომლის მიხედვითაც მე-3 პუნქტში წარმოდგენილი პრობლემის მოგვარების სიმულაციისას NACS-სერვერი იღებს აღნიშნულ შეტყობინებას და ამუშავებს მას (ნახ. 75).

```
R2_LAB 2.2.2.2 High CPU <90%
Listen to new message
finding index by 1.1.1.1CPU_Reset
    2
finding index by 8.8.8.8CPU_Reset
    2
finding index by 6.6.6.6CPU_Reset
    2
we found more then one index and execute config 1.1.1.1CPU_Reset
we found more then one index and execute config 8.8.8.8CPU_Reset
we found more then one index and execute config 6.6.6.6CPU_Reset
R3_LAB 3.3.3.3 High CPU <90%
Listen to new message
finding index by 1.1.1.1CPU_Reset
    1
finding index by 8.8.8.8CPU_Reset
    1
finding index by 6.6.6.6CPU_Reset
    1
we found only one index and execute config 1.1.1.1CPU_Reset
we found only one index and execute config 8.8.8.8CPU_Reset
we found only one index and execute config 6.6.6.6CPU_Reset
```

ნახ. 75. NACS-ის მიერ შეტყობინების მიღება და დამუშავება

CPU-ს გადატვირთულობის პრობლემის მოგვარების შემდგომ R1, R6 და R8-ზე მოიხსნება R2-და R3-ის მარშრუტის შეზღუდვა და vrf VOIP-ტრაფიკი იმოძრავებს OSPF-ით მიღებული უმოკლესი მარშრუტით (ნახ.76).



ნახ. 76. vrf VoIP-ტრაფიკის მოძრაობის სურათი პრობლემის მოხსნის შემდეგ

დასკვნები

ძირითადი შედეგები, რომლებიც მიღებულია წინამდებარე ნაშრომში, შემდეგია:

1. შეფასებულია VoIP-ქსელების კომპონენტები და უსაფრთხოების რისკები და შემუშავებულია VoIP-უსაფრთხოების მოდელი, რომლის განხორციელება უზრუნველყოფს VoIP-ქსელის მომხმარებელთა ხმოვანი ტრაფიკის ხელმისაწვდომობას, მთლიანობასა და კონფიდენციალობას;

2. გაანალიზებულია MPLS-ქსელის არქიტექტურა და MPLS-TE-ტექნოლოგია, რომლის საფუძველზეც დადგინდა, რომ MPLS მოქნილი ტექნოლოგიაა, რომელიც უზრუნველყოფს უსაფრთხოებას, პაკეტების დაგვიანების მინიმიზაციასა და მონაცემთა გადაცემას მაღალი სიჩქარით;

3. ნაჩვენებია, რომ MPLS-ქსელის არქიტექტურისა და MPLS-TE-ტექნოლოგიის ერთობლიობა ძირითადად გამოიყენება სერვის-პროვაიდერების მიერ რეალურ დროში გადაცემული ხმოვანი და ვიდეო-სერვისების გაუმჯობესებული ხარისხით მისაწოდებლად;

4. დადგენილია, რომ MPLS-ის ერთ-ერთი მნიშვნელოვანი ფუნქცია ტრაფიკის ინჟინერინგია (TE), რომელიც მნიშვნელოვან როლს ასრულებს ქსელური დატვირთვის მინიმიზაციისთვის, რესურსების რეზერვაციისთვის, ბალანსირებისა და მართვისთვის და რომლის საშუალებითაც განისაზღვრება მონაცემების უსაფრთხო და ხარისხიანი მარშრუტიზაცია;

5. დამუშავებულია მონაცემთა ნაკადის გვირაბების (Tunnels), ვირტუალური მარშრუტიზაცია-გადამისამართების (VRF) პროტოკოლების ავტომატიზაციის პროგრამულ უზრუნველყოფასთან ინტეგრაციის მეთოდი, რამაც უზრუნველყო ხმისა და ინტერნეტ-ტრაფიკის ერთმანეთისა და გლობალური მარშრუტიზაციის ცხრილიდან იზოლაცია;

6. დამუშავებულია რეალური IP-MPLS-ის ანალოგიური ქსელი ვირტუალურ პროგრამულ EVE-NG-გარემოში, რამაც უზრუნველყო

შესაბამისი ალგორითმის მოდელირების, IP-MPLS ქსელის სიმულაციის, ქსელური პრობლემისა და მისი მოგვარების განხორციელების შესაძლებლობა.

7. დასაბუთებულია ქსელის მონიტორინგის პროგრამული უზრუნველყოფის გამოყენებისა და სატესტო გარემოში სიმულაციის შესაძლებლობა;

8. განხორციელებულია ინტერნეტისა და ხმოვანი სიგნალების მონაცემების ერთმანეთისგან იზოლირებისა და ხმოვანი სიგნალის ტრაფიკის წყაროზე დაფუძნებული მარშრუტიზაციის (SR) ოპერაციები;

9. რეალიზებულია ქსელის მონიტორინგის შემუშავებულ სისტემაში მიმდინარე პროცესების ვიზუალიზაციისა და რეალურ დროში დაკვირვების შესაძლებლობა, რამაც უზრუნველყო მონიტორინგის სისტემის მოქნილობა და მისი ეფექტურობის გაუმჯობესება.

10. შემუშავებლია გადაცემის პაკეტური საკომუტაციო ქსელის მართვის, მონიტორინგისა და ინტელექტუალური გადაწყვეტილებების მიმღები სისტემა, რომელიც ქსელში დაფიქსირებული პრობლემების დროს პროგრამული ალგორითმების დახმარებით გადაწყვეტს ქსელური მოწყობილობების კონფიგურაციის ოპტიმიზაციის ამოცანას და დაახარისხებს შესრულებულ ქმედებებს.

გამოყენებული ლიტერატურა

1. Sendra.S, et al. (2010). “Study and Performance of Interior Gateway IP Routing Protocols.” Network Protocols and Algorithms 2(4).

2. Farhangi S., et al. (2012). “Performance Comparison of Mixed Protocols Based on EIGRP, IS-IS and OSPF for Real-time Applications.” Middle Eastern Journal of Scientific Research 12(11).

3. https://www.ccexpert.us/network-design/metrics-used-by-routing-protocols.html.
   ბოლო გადამოწმება 2.12.2019.

4. Aslam M.N., Aziz Y. Traffic Engineering with Multi-Protocol Label Switching, Blekinge Institute of Technology, August 2008.

5. Multiprotocol Label Switching (MPLS).
   https://searchnetworking.techtarget.com/definition/Multiprotocol-Label-Switching-MPLS
   ბოლო გადამოწმება 4.10.2019.

6. Goldschmidt O. ISP Backbone Traffic Inference Methods to Support Traffic Engineering. In Internet Statistics and Metrics Analysis (ISMA) Workshop, San Diego, CA, December 2000

7. https://team.inria.fr/rap/files/2013/12/BR04.pdf
   ბოლო გადამოწმება 9.10.2018

8. https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/multiprotocol-label-switching-traffic-engineering/whitepaper_c11-551235.html.
   ბოლო გადამოწმება 1.14.2019

9. Varadhan K., Govindan R., Estrin D. Persistent Route Oscillations in Inter-Domain Routing, Computer Networks, 1999.

10. Rosen E., Viswanathan A., Callon R. Multiprotocol Label Switching Architecture, RFC 3031, January 2001.

11. Resource ReSerVation Protocol https://tools.ietf.org/html/rfc2205.
    ბოლო გადამოწმება 2.10.2019

12. http://www.knom.or.kr/tutorial/tut-01/MPLS.pdf.
    ბოლო გადამოწმება 3.10.2019

13. https://en.wikipedia.org/wiki/Shared_Risk_Resource_Group.
    ბოლო გადამოწმება 5.10.2019

14. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.650.9995&rep=rep1&type=pdf.
    ბოლო გადამოწმება 5.1.2019

15. Ghein, Luc De. (2006). MPLS Fundamentals. Cisco Press.

16. https://www.cisco.com/c/en/us/support/docs/multiprotocol-label-switching-mpls/mpls/118846-config-mpls-00.html
    ბოლო გადამოწმება 3.11.2019

17. https://www.cisco.com/c/en/us/products/ios-nx-os-software/resource-reservation-protocol-rsvp/index.html. ბოლო გადამოწმება 2.11.2019

18. https://www4.cs.fau.de/Projects/JRTP/pmt/node25.html.
    ბოლო გადამოწმება 2.11.2019

19. http://icact.org/upload/2010/0264/20100264_finalpaper.pdf.
    ბოლო გადამოწმება 4.18.2019

20. https://www.mplsinfo.org/architecture.html.
    ბოლო გადამოწმება 9.11.2018

21. https://www.cisco.com/c/en/us/td/docs/net_mgmt/vpn_solutions_center/2-0/mpls/provisioning/guide/PGmpls1.pdf.
    ბოლო გადამოწმება 1.30.2019

22. https://searchnetworking.techtarget.com/answer/What-are-the-benefits-of-a-Multiprotocol-Label-Switching-based-Border-Gateway-Protocol-free-core.
    ბოლო გადამოწმება 5.11.2019

23. Fortz B., Rexford J and Thorup M. “Traffic Engineering With Traditional IP Routing Protocols” in IEEE Communications Magazine, October 2002.
24. Awduche D.O., “MPLS and traffic engineering in IP networks.” Communications Magazine, IEEE, vol. 37, no. 12, pp. 42-47, 1999.

25. http://home.zcu.cz/~petrovic/TZ/Security_Considerations_in_IP_Telephony_Network_Configuration.pdf. ბოლო გადამოწმება 11.30.2018

26. https://www.sans.org/reading-room/whitepapers/voip/security-issues-countermeasure-voip-1701. ბოლო გადამოწმება 11.30.2018

27. https://en.wikipedia.org/wiki/Session_Initiation_Protocol. ბოლო გადამოწმება 5.07.2018

28. https://en.wikipedia.org/wiki/Voice_over_IP. ბოლო გადამოწმება 5.07.2018

29. https://www.dialogic.com/webhelp/csp1010/8.4.1_ipn3/sip_software_chap_-_sip_protocol_overview.htm. ბოლო გადამოწმება 5.07.2018

30. Rosenberg J., Schulzrinne H. Sip: locating sip servers, RFC 3263, june 2002.

31. Rosenberg J., Schulzrinne H. June 2002.RFC 3621 Session Initiation Protocol (SIP).

32. https://en.wikipedia.org/wiki/Session_Initiation_Protocol. ბოლო გადამოწმება 5.13.2018

33. Schulzrinne S. H. & Cangussu J. (2009, May 3). Issues and challenges in securing VoIP. Computer & Security

34. Chen & S. E. Voip security threats relevant to speermint draft-niccolinispeermint-voipthreats-02. Technical report, The IETF Trust, SPEERMINT Working Group, August 2007.

35. Borenstein N. November 1996. Multipurpose internet mail extensions (mime) part one: Format of internet message bodies. Internet

36. Juniper Networks, I. Enterprise voip security, best practices. Technical report, Juniper Networks, Inc., 2006.

37. Ramsdell, B. June 1999. S/mime version 3 message specification, request for comments: 2633. Network Working Group.

38. აბულაძე ვ., ხუნწარია ჯ., ჯორჯაძე ი., გიორგაძე გ. ხმოვანი სიგანლის გადაცემის ქსელის უსაფრთხოების მოდელი. "ენერგია", 2019, №1(89), გვ. 90-95.

39. აბულაძე ვ., ხუნწარია ჯ., ჯორჯაძე ი., გიორგაძე გ. MPLS ქსელში მონაცემთა ნაკადის ავტომატური მართვის მოდელი. “მართვის ავტომატიზებული სისტემები”, 2019, №1(28), გვ. 106-111.

40. Balchunas A. (2014), Cisco CCNP Routing Study Guide, www.routeralley.com.
ბოლო გადამოწმება 8.30.2018

41. http://vet.ge/wp-content/uploads/2015/08/studentis-saxelmzgvanelo-qseluri-kavshirebi-da-WAN-teqnologiebi-qselis-admin-2.pdf.
ბოლო გადამოწმება 8.30.2018

42. https://www.slideshare.net/lz1dsb/why-sdn,2015
ბოლო გადამოწმება 12.10.2018.

43. OpenDaylightProject, https://en.wikipedia.org/wiki/OpenDaylight_Project
ბოლო გადამოწმება 1.15.2019

44. გიორგაძე გ., ჯორჯაძე ი., აბულაძე ვ., ხუნწარია ჯ. მეოთხე თაობის რადიო ქსელის დატვირთვისა და გამტარუნარიანობის ანალიზი. “მართვის ავტომატიზებული სისტემები”, 2019, №1(28), გვ. 112-117.

დანართი 1. მოდელირებისთვის გამოყენებული პროგრამული კოდი

```
var http = require('http');
var url = require('url');

const methodClass = require('./myClass.js');
const libTriggerLink = require('./TriggerLink.js');
const libTriggerCPU = require('./TriggerCPU.js');
const libResetLink = require('./ResetLink.js');
const libResetCPU = require('./ResetCPU.js');

http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  /*Use the url module to turn the querystring into an object:*/
  var q = url.parse(req.url, true).query;
  /*Return the year and month from the query object:*/
  var urlPostInfo = q.script; // + " " + q.ip + " " + q.interface + " " + q.attribute;
  //R1_LAB - Ethernet0/0 - Et0/0 Transmit <90%

console.log(urlPostInfo);

var Recognaizer = methodClass.smsRecognaizer(urlPostInfo);
var scriptIndex = Recognaizer[0];
var script_Trigger_Reset = Recognaizer[1];
var script_Cpu_Link = Recognaizer[2];
var scriptRouterIP = Recognaizer[3];

if(script_Trigger_Reset == "TRIGGER" && script_Cpu_Link == "Link")
{
```

```
  //execute code on the Routers Orion Debug Triggerd_Link Message
  for (x in scriptRouterIP) {
    var resultTriggerLink =
libTriggerLink.funcTriggerLink(scriptRouterIP[x],scriptIndex);
  }
  console.log("Listen to new message");
}
else
if(script_Trigger_Reset == "TRIGGER" && script_Cpu_Link == "CPU")
{
  //execute code on the Routers Orion Debug Triggerd_CPU Message
  var router = [];
  switch (scriptRouterIP[0]) {
    case "R1_LAB": router = ["6.6.6.6","8.8.8.8"];
            break;
    case "R6_LAB": router = ["1.1.1.1","8.8.8.8"];
            break;
    case "R8_LAB": router = ["1.1.1.1","6.6.6.6"];
      break;
    default:
      router = ["1.1.1.1","6.6.6.6","8.8.8.8"];
  }
  console.log(router);
  for (x in router) {
      var resultTriggerCPU =
libTriggerCPU.funcTriggerCPU(router[x],"R1_R6_R8_CPU_TRIGGER",scriptRouterI
P[1]);
  }
  console.log("Listen to new message");
}
```

```
else
if(script_Trigger_Reset == "RESET" && script_Cpu_Link == "Link")
{
  //execute code on the Routers Orion Debug Reset_Link Message
  for (x in scriptRouterIP) {
    var resultResetLink = libResetLink.funcResetLink(scriptRouterIP[x],scriptIndex);
  }
  console.log("Listen to new message");
}
else
if(script_Trigger_Reset == "RESET" && script_Cpu_Link == "CPU")
{
  //execute code on the Routers Orion Debug Reset_CPU Message
  var router = [];
  switch (scriptRouterIP[0]) {
    case "R1_LAB": router = ["6.6.6.6","8.8.8.8"];
            break;
    case "R6_LAB": router = ["1.1.1.1","8.8.8.8"];
            break;
    case "R8_LAB": router = ["1.1.1.1","6.6.6.6"];
      break;
    default:
      router = ["1.1.1.1","6.6.6.6","8.8.8.8"];
  }
  for (x in router) {
    var resultResetCPU =
libResetCPU.funcResetCPU(router[x],"R1_R6_R8_CPU_RESET",scriptRouterIP[1]);
  }
  console.log("Listen to new message");
```

```
}

}).listen(8080);

exports.smsRecognaizer = function (script){
	if (script == "R1_LAB - Ethernet0/0 - Et0/0 Transmit >90%")
	{
		return ["R1_LAB_TRIGGER-E0/0","TRIGGER","Link",["1.1.1.1"]];
		console.log("R1_LAB_TRIGGER-E0/0")
	}else
	if (script == "R1_LAB - Ethernet0/0 - Et0/0 Transmit <90%")
	{
		return ["R1_LAB_RESET-E0/0","RESET","Link",["1.1.1.1"]];
		console.log("R1_LAB_RESET-E0/0");
	}else
	if (script == "R1_LAB - Ethernet0/1 - Et0/1 Transmit >90%")
	{
		return ["R1_LAB_TRIGGER-E0/1","TRIGGER","Link",["1.1.1.1"]];
		console.log("R1_LAB_TRIGGER-E0/1");
	}else
	if (script == "R1_LAB - Ethernet0/1 - Et0/1 Transmit <90%")
	{
		return ["R1_LAB_RESET-E0/1","RESET","Link",["1.1.1.1"]];
		console.log("R1_LAB_RESET-E0/1");
	}else
	if (script == "R1_LAB - Ethernet1/1 - Et1/1 Transmit >90%")
	{
		return ["R1_LAB_TRIGGER-E1/1","TRIGGER","Link",["1.1.1.1"]];
		console.log("R1_LAB_TRIGGER-E1/1");
```

```
}else
if (script == "R1_LAB - Ethernet1/1 - Et1/1 Transmit <90%")
{
        return ["R1_LAB_RESET-E1/1","RESET","Link",["1.1.1.1"]];
        console.log(R1_LAB_RESET-E1/1);
}else
if (script == "R4_LAB - Ethernet0/3 - Et0/3 Transmit >90%")
{
        return ["R4_LAB_TRIGGER-E0/3","TRIGGER","Link",["1.1.1.1"]];
        console.log("R4_LAB_TRIGGER-E0/3");
}else
if (script == "R4_LAB - Ethernet0/3 - Et0/3 Transmit <90%")
{
        return ["R4_LAB_RESET-E0/3","RESET","Link",["1.1.1.1"]];
        console.log("R4_LAB_RESET-E0/3");
}else
if (script == "R3_LAB - Ethernet0/2 - Et0/2 Transmit >90%")
{
        return ["R3_LAB_TRIGGER-E0/2","TRIGGER","Link",["1.1.1.1"]];
        console.log("R3_LAB_TRIGGER-E0/2");
}else
if (script == "R3_LAB - Ethernet0/2 - Et0/2 Transmit <90%")
{
        return ["R3_LAB_RESET-E0/2","RESET","Link",["1.1.1.1"]];
        console.log("R3_LAB_RESET-E0/2");
}else
if (script == "R2_LAB - Ethernet0/2 - Et0/2 Transmit >90%")
{
```

```
                return ["R2_LAB_TRIGGER-
E0/2","TRIGGER","Link",['1.1.1.1','6.6.6.6']];
                console.log("R2_LAB_TRIGGER-E0/2");
        }else
        if (script == "R2_LAB - Ethernet0/2 - Et0/2 Transmit <90%")
        {
                return ["R2_LAB_RESET-E0/2","RESET","Link",['1.1.1.1','6.6.6.6']];
                console.log("R2_LAB_RESET-E0/2");
        }else
        if (script == "R5_LAB - Ethernet0/3 - Et0/3 Transmit >90%")
        {
                return ["R5_LAB_TRIGGER-
E0/3","TRIGGER","Link",['1.1.1.1','6.6.6.6']];
                console.log("R5_LAB_TRIGGER-E0/3");
        }else
        if (script == "R5_LAB - Ethernet0/3 - Et0/3 Transmit <90%")
        {
                return ["R5_LAB_RESET-E0/3","RESET","Link",['1.1.1.1','6.6.6.6']];
                console.log("R5_LAB_RESET-E0/3");
        }else
        if (script == "R7_LAB - Ethernet1/0 - Et1/0 Transmit >90%")
        {
                return ["R7_LAB_TRIGGER-
E1/0","TRIGGER","Link",['1.1.1.1','6.6.6.6']];
                console.log("R7_LAB_TRIGGER-E1/0");
        }else
        if (script == "R7_LAB - Ethernet1/0 - Et1/0 Transmit <90%")
        {
                return ["R7_LAB_RESET-E1/0","RESET","Link",['1.1.1.1','6.6.6.6']];
                console.log("R7_LAB_RESET-E1/0");
```

```
	}else
	if (script == "R6_LAB - Ethernet0/0 - Et0/0 Transmit >90%")
	{
		return ["R6_LAB_TRIGGER-
E0/0","TRIGGER","Link",['1.1.1.1','6.6.6.6']];
		console.log("R6_LAB_TRIGGER-E0/0");
	}else
	if (script == "R6_LAB - Ethernet0/0 - Et0/0 Transmit <90%")
	{
		return ["R6_LAB_RESET-E0/0","RESET","Link",['1.1.1.1','6.6.6.6']];
		console.log("R6_LAB_RESET-E0/0");
	}else
	if (script == "R6_LAB - Ethernet1/0 - Et1/0 Transmit >90%")
	{
		return ["R6_LAB_TRIGGER-
E1/0","TRIGGER","Link",['1.1.1.1','6.6.6.6']];
		console.log("R6_LAB_TRIGGER-E1/0");
	}else
	if (script == "R6_LAB - Ethernet1/0 - Et1/0 Transmit <90%")
	{
		return ["R6_LAB_RESET-E1/0","RESET","Link",['1.1.1.1','6.6.6.6']];
		console.log("R6_LAB_RESET-E1/0");
	}else
	if (script == "R4_LAB - Ethernet0/1 - Et0/1 Transmit >90%")
	{
		return ["R4_LAB_TRIGGER-
E0/1","TRIGGER","Link",['1.1.1.1','6.6.6.6']];
		console.log("R4_LAB_TRIGGER-E0/1");
	}else
	if (script == "R4_LAB - Ethernet0/1 - Et0/1 Transmit <90%")
```

```
        {
                return ["R4_LAB_RESET-E0/1","RESET","Link",['1.1.1.1','6.6.6.6']];
                console.log("R4_LAB_RESET-E0/1");
        }else
        if (script.indexOf("CPU") != -1 && script.indexOf(">90%") != -1)
        {
                var myArray = script.split(" ");
                return
["R1_R6_R8_CPU_TRIGGER","TRIGGER","CPU",[myArray[0],myArray[1]]];
                console.log("R1_R6_R8_CPU_TRIGGER");
        }else
        if (script.indexOf("CPU") != -1 && script.indexOf("<90%") != -1)
        {
                var myArray = script.split(" ");
                return
["R1_R6_R8_CPU_RESET","RESET","CPU",[myArray[0],myArray[1]]];
                console.log("R1_R6_R8_CPU_RESET");
        }
}

const runSSH = require('./runCommand.js');
const check = require('./showCommandCheck.js');
const loadJsonFile = require('load-json-file');

exports.funcResetCPU = function (RouterIP,scriptIndex,excludeAddress)
{
   var myResult = "";
   loadJsonFile('scripts.json').then(json => {
      // `json` contains the parsed object
```

```
        var obj = json;

        var showCommand =
obj.scripts[scriptIndex].script_1_ShowCommand.command;

        var attribute = obj.scripts[scriptIndex].script_1_ShowCommand.attribute;

        runSSH.runCommand(RouterIP,showCommand,showCommandResult => {

            console.log("finding index by " + RouterIP + "CPU_Reset");

            var checkResult =
check.resetLinkCheck(showCommandResult,attribute[0],excludeAddress);

                                console.log(checkResult[1]);

                if(checkResult[0] < 2)

                {

                    var runCommand =
obj.scripts[scriptIndex].script_1_RunCommand.only_one_index[RouterIP];

                    runCommand.push("no index " + checkResult[1],"end","wr","exit");

                    runSSH.runCommand(RouterIP,runCommand,runCommandResult => {

                        console.log("we found only one index and execute config " + RouterIP
+ "CPU_Reset");

                        //run command when logic is true

                        myResult = runCommandResult + "|" + RouterIP + "Execute
Command - Reset->CPU->only_one_index -> R1\n";

                    });

                }else

                {

                    var runCommand =
obj.scripts[scriptIndex].script_1_RunCommand.more_index;

                    runCommand.push("no index " + checkResult[1],"end","mpls traffic-eng
reoptimize tunnel 1","mpls traffic-eng reoptimize tunnel 2","wr","exit");

                    runSSH.runCommand(RouterIP,runCommand,runCommandResult => {

                        console.log("we found more then one index and execute config " +
RouterIP + "CPU_Reset");

                        //run command when logic is false
```

```
                    myResult = runCommandResult + "|" + RouterIP + "Execute
Command - Reset->CPU->more_index\n";
                });
            }
        });

    });
    return myResult;
    //console.log(myResult);
}
const runSSH = require('./runCommand.js');
const check = require('./showCommandCheck.js');
const loadJsonFile = require('load-json-file');

exports.funcResetLink = function (RouterIP,scriptIndex)
{
  var result = " ";
  loadJsonFile('scripts.json').then(json => {
    // `json` contains the parsed object
    var obj = json;
    var showCommand = obj.scripts[scriptIndex].script_1_ShowCommand.command;
    var attribute = obj.scripts[scriptIndex].script_1_ShowCommand.attribute;
    runSSH.runCommand(RouterIP,showCommand,showCommandResult => {
      console.log("finding index by " + RouterIP + "Link_Reset");
      var checkResult =
check.resetLinkCheck(showCommandResult,attribute[0],attribute[1]);
      if(checkResult[0] < 2)
      {
        var runCommand =
obj.scripts[scriptIndex].script_1_RunCommand.only_one_index;
```

```
        runCommand.push("no index " + checkResult[1],"end","wr","exit");
        runSSH.runCommand(RouterIP,runCommand,runCommandResult => {
          console.log("we found only one index " + RouterIP + "Link_Reset");
          //when we find only only_one_index
          result = runCommandResult + "|" + RouterIP+"Execute Command - Reset->Link->only_one_index\n";
        });
      }else
      {
        switch (attribute[1]) {
            case "10.40.50.6":
            case "10.40.50.14":
            case "10.40.50.30":
            case "10.40.50.18":
            case "10.40.50.34":
              var runCommand = obj.scripts[scriptIndex].script_1_RunCommand.more_index;
              runCommand.push("no index " + checkResult[1],"end","mpls traffic-eng reoptimize tunnel 1","mpls traffic-eng reoptimize tunnel 2","wr","exit");
              runSSH.runCommand(RouterIP,runCommand,runCommandResult => {
                console.log("we found more then one index and execute config " + RouterIP + "Link_Reset");
                //when we finde more_index
                result = runCommandResult + "|" + RouterIP+"Execute Command - Reset->Link->more_index\n";
              });
              break;
            default:
              var runCommand = obj.scripts[scriptIndex].script_1_RunCommand.more_index;
```

```
            runCommand.push("no index " + checkResult[1],"end","mpls traffic-eng
reoptimize tunnel 1","wr","exit");

            runSSH.runCommand(RouterIP,runCommand,runCommandResult => {

              console.log("we found more then one index and execute config " +
RouterIP + "CPU_Reset");

              //run command when logic is false

              result = runCommandResult + "|" + RouterIP+"Execute Command - Reset-
>Link->more_index\n";

            });

         }

       }

     });

   });

  return result;

  //console.log(result);

}



exports.runCommand = function (LocalIP, commandssh, result) {

   var host = {

     server: {

       host: LocalIP,

       port: "22",

       userName: "cisco",

       password: "cisco",

       algorithms: {

         kex: [

           'diffie-hellman-group1-sha1',

           'ecdh-sha2-nistp256',

           'ecdh-sha2-nistp384',
```

```
            'ecdh-sha2-nistp521',
            'diffie-hellman-group-exchange-sha256',
            'diffie-hellman-group14-sha1'],
          cipher: [
            'aes128-ctr',
            'aes192-ctr',
            'aes256-ctr',
            'aes128-gcm',
            'aes128-gcm@openssh.com',
            'aes256-gcm',
            'aes256-gcm@openssh.com',
            'aes256-cbc' ]
          }
      },
    commands: commandssh,
    msg: {
      send: function( message ) {
        //console.log("message: " + message);
      }
    },
    verbose: false,
    debug: false,
    idleTimeOut: 5000,
    onEnd: function( sessionText, sshObj ) {
      sshObj.msg.send("--------- onEnd has ----------");
      sshObj.msg.send(sessionText);
    }
};
```

```
    var SSH2Shell = require ('ssh2shell'),
    //Create a new instance passing in the host object
    SSH = new SSH2Shell(host);

    //Start the process
    SSH.connect(result);
}

exports.triggeredCheck = function (text,attribute)
{
        if (text.indexOf(attribute) != -1)
        {
                return true;
        }
        else
        {
                return false;
        }
}

exports.resetLinkCheck = function(text,attribute1,attribute2)
{
        var myArray = text.split("\n");
        var ind = 0;
        var indexNumber = 0;
        for(x in myArray)
        {
                if (myArray[x].indexOf(attribute1) != -1)
                {
```

```
                ind++;
            }
            if (myArray[x].indexOf(attribute2) != -1)
            {
                indexNumber = myArray[x].slice(0, myArray[x].indexOf(":"));

            }
        }
        return [ind,indexNumber];
console.log("index number " + indexNumber);
}


const runSSH = require('./runCommand.js');
const check = require('./showCommandCheck.js');
const loadJsonFile = require('load-json-file');

exports.funcTriggerCPU = function (RouterIP,scriptIndex,excludeAddress)
{
   var result = "";
   loadJsonFile('scripts.json').then(json => {
      // `json` contains the parsed object
      var obj = json;
      var showCommand =
obj.scripts[scriptIndex].script_1_ShowCommand.command;
      var attribute = obj.scripts[scriptIndex].script_1_ShowCommand.attribute;
      runSSH.runCommand(RouterIP,showCommand,showCommandResult => {
         console.log("finding exclude-address in " + RouterIP + "CPU_Trigger");
         var checkResult = check.triggeredCheck(showCommandResult,attribute);
```

```
        if(checkResult)
        {
            //run command when logic is true
            var runCommand = obj.scripts[scriptIndex].script_1_RunCommand.true;
            runCommand.splice(2, 0, "exclude-address " + excludeAddress);
            runSSH.runCommand(RouterIP,runCommand,runCommandResult => {
                console.log("find result is true in " + RouterIP + " and execute command
CPU_Trigger");
                result = runCommandResult + "|" + RouterIP+"Execute Command -
Trigger->CPU->TRUE\n";
            });
        }else
        {
            //run command when logic is false
                                    var runCommand1 =
obj.scripts[scriptIndex].script_1_RunCommand.false[RouterIP];
                                    //console.log(runCommand1);
            runCommand1.splice(2, 0, "exclude-address " + excludeAddress);
            runSSH.runCommand(RouterIP,runCommand1,runCommandResult => {
                console.log("find result is false in " + RouterIP + " and execute command
CPU_Trigger");
                //run command with routers
                result = runCommandResult + "|" + RouterIP + " - Execute Command -
Trigger->CPU->FALSE \n";
            });
        }
      });
   });
   return result;
   //console.log(result);
}
```

```
const runSSH = require('./runCommand.js');
const check = require('./showCommandCheck.js');
const loadJsonFile = require('load-json-file');

exports.funcTriggerLink = function (RouterIP,scriptIndex)
{
    var result = "";
    loadJsonFile('scripts.json').then(json => {
        // `json` contains the parsed object
        var obj = json;
        var showCommand =
obj.scripts[scriptIndex].script_1_ShowCommand.command;
        var attribute = obj.scripts[scriptIndex].script_1_ShowCommand.attribute;
        runSSH.runCommand(RouterIP,showCommand,showCommandResult => {
            console.log("finding exclude-address in " + RouterIP + "Link_Trigger");
            var checkResult = check.triggeredCheck(showCommandResult,attribute);
            if(checkResult)
            {
                var runCommand = obj.scripts[scriptIndex].script_1_RunCommand.true;
                runSSH.runCommand(RouterIP,runCommand,runCommandResult => {
                    console.log("find result is true in " + RouterIP + " and execute command
Link_Trigger");
                    //run command when logic is true
                    result = runCommandResult + "|" + RouterIP+"Execute Command -
Trigger->Link->TRUE\n";
                });
            }else
            {
                var runCommand = obj.scripts[scriptIndex].script_1_RunCommand.false;
                runSSH.runCommand(RouterIP,runCommand,runCommandResult => {
```

```
                console.log("find result is false in " + RouterIP + " and execute command
Link_Trigger");

                //run command when logic is false

                result = runCommandResult + "|" + RouterIP+"Execute Command -
Trigger->Link->FALSE\n";

            });

        }

    });

    return result;

    //console.log(result);

});

}
```