
# INNOV 2021

The Tenth International Conference on Communications, Computation, Networks and Technologies

ISBN: 978-1-61208-900-3

October 3 -7, 2021

Barcelona, Spain

**INNOV 2021 Editors**

Pascal Lorenz, University of Haute Alsace, France

# INNOV 2021

# Forward

The Tenth International Conference on Communications, Computation, Networks and Technologies (INNOV 2021), held on October 3 - 7, 2021 in Barcelona, Spain, aimed at addressing recent research results and forecasting challenges on selected topics related to communications, computation, networks and technologies.

Considering the importance of innovative topics in today's technology-driven society, there is a paradigm shift in classical-by-now approaches, such as networking, communications, resource sharing, collaboration and telecommunications. Recent achievements demand rethinking available technologies and considering the emerging ones.

The conference had the following tracks:

☐ Communications
☐ Networking
☐Computing
☐Web Semantic and Data Processing
☐Security, Trust, and Privacy

We take here the opportunity to warmly thank all the members of the INNOV 2021 technical program committee, as well as the numerous reviewers. The creation of such a high quality conference program would not have been possible without their involvement. We also kindly thank all the authors that dedicated much of their time and effort to contribute to INNOV 2021. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions.

Also, this event could not have been a reality without the support of many individuals, organizations and sponsors. We also gratefully thank the members of the INNOV 2021 organizing committee for their help in handling the logistics and for their work that made this professional meeting a success.

We hope that INNOV 2021 was a successful international forum for the exchange of ideas and results between academia and industry and to promote further progress in the areas of communication, computation, networks and technologies..

**INNOV 2021 Steering Committee**

Sean Sturley, University of the West of Scotland, UK
Yeim-Kuan Chang, National Cheng Kung University, Taiwan

**INNOV 2021 Publicity Chair**

José Miguel Jiménez, Universitat Politecnica de Valencia, Spain
Lorena Parra, Universitat Politecnica de Valencia, Spain

# INNOV 2021

## Committee

**INNOV 2021 Steering Committee**

Sean Sturley, University of the West of Scotland, UK
Yeim-Kuan Chang, National Cheng Kung University, Taiwan

**INNOV 2021 Publicity Chair**

José Miguel Jiménez, Universitat Politecnica de Valencia, Spain
Lorena Parra, Universitat Politecnica de Valencia, Spain

**INNOV 2021 Technical Program Committee**

Kishwar Ahmed, University of South Carolina Beaufort, USA
Amjad Ali,University of Swat, Pakistan
Zahra Ebadi Ansaroudi, University of Salerno, Italy
Eugen Borcoci, University "Politehnica"of Bucharest (UPB), Romania
YK Chang, National Cheng Kung University, Taiwan
DeJiu Chen, KTH Royal Institute of Technology, Sweden
Yung-Yao Chen, National Taiwan University of Science and Technology (NTUST), Taiwan
Albert M. K. Cheng, University of Houston, USA
Enrique Chirivella-Perez, University of the West of Scotland, UK
Karl Cox, University of Brighton, UK
Daniela D'Auria, Free University of Bozen-Bolzano, Italy
Panagiotis Fouliras, University of Macedonia, Thessaloniki, Greece
Marco Furini, University of Modena and Reggio Emilia, Italy
Nikolaos Gorgolis, University of Patras, Greece
Rafael Götzen, Institute for Industrial Management (FIR) at RWTH Aachen University, Germany
Victor Govindaswamy, Concordia University Chicago, USA
Qiang He, Swinburne University of Technology, Australia
Shih-Chang Huang, National Formosa University, Taiwan
Wen-Jyi Hwang, National Taiwan Normal University, Taipei, Taiwan
Sergio Ilarri, University of Zaragoza, Spain
Brigitte Jaumard, Concordia University, Canada
Thomas Jell, Siemens Mobility GmbH, Germany
Alexey Kashevnik, SPIIRAS, Russia
Khaled Khankan, Taibah University, Saudi Arabia
Vasileios Komianos, Ionian University, Greece
Igor Kotenko, SPIIRAS, Russia
Boris Kovalerchuk, Central Washington University, USA
Maurizio Leotta, University of Genova, Italy
Yiu-Wing Leung, Hong Kong Baptist University, Hong Kong

**Copyright Information**

For your reference, this is the text governing the copyright release for material published by IARIA.

The copyright release is a transfer of publication rights, which allows IARIA and its partners to drive the dissemination of the published material. This allows IARIA to give articles increased visibility via distribution, inclusion in libraries, and arrangements for submission to indexes.

I, the undersigned, declare that the article is original, and that I represent the authors of this article in the copyright release matters. If this work has been done as work-for-hire, I have obtained all necessary clearances to execute a copyright release. I hereby irrevocably transfer exclusive copyright for this material to IARIA. I give IARIA permission or reproduce the work in any media format such as, but not limited to, print, digital, or electronic. I give IARIA permission to distribute the materials without restriction to any institutions or individuals. I give IARIA permission to submit the work for inclusion in article repositories as IARIA sees fit.

I, the undersigned, declare that to the best of my knowledge, the article is does not contain libelous or otherwise unlawful contents or invading the right of privacy or infringing on a proprietary right.

Following the copyright release, any circulated version of the article must bear the copyright notice and any header and footer information that IARIA applies to the published article.

IARIA grants royalty-free permission to the authors to disseminate the work, under the above provisions, for any academic, commercial, or industrial use. IARIA grants royalty-free permission to any individuals or institutions to make the article available electronically, online, or in print.

IARIA acknowledges that rights to any algorithm, process, procedure, apparatus, or articles of manufacture remain with the authors and their employers.

I, the undersigned, understand that IARIA will not be liable, in contract, tort (including, without limitation, negligence), pre-contract or other representations (other than fraudulent misrepresentations) or otherwise in connection with the publication of my work.

Exception to the above is made for work-for-hire performed while employed by the government. In that case, copyright to the material remains with the said government. The rightful owners (authors and government entity) grant unlimited and unrestricted permission to IARIA, IARIA's contractors, and IARIA's partners to further distribute the work.

**Table of Contents**

# Range Encoding and Hash table based Packet Classification for Global View Networking

Yeim-Kuan Chang, Yi-Hao Lai, and Che-Wei Li
Department of Computer Science and Information Engineering
National Cheng Kung University, Taiwan

***Abstract*—Packet classification is an important functionality of the Internet router for many network applications. With the emergence of software-defined networking (SDN), packet classification for global view networking is used to search the actions taken at multiple routers, not only at a single router. The control plane provides a global view of the network, which allows applications to identify the network-wide behavior of a packet, defined as the combination of actions taken at all routers. In this paper, we propose a two-layer scheme named range encoding hash table (REHT) that can search the network-wide behaviors of packets efficiently. In layer one, the header field values of all fields are encoded separately. In layer two, hash tables are used for the encoded values to achieve high classification speed. Based on our experiments using real network configurations, REHT performs much faster than BDDs and MDD schemes.**

***Keywords- Packet classification; IP lookup; Encoding; Hash table***

## I. INTRODUCTION

A router forwards packets between networks. When a packet comes in, the router uses packet headers to determine the next hop obtained from routing table. Also, routers need packet classification [2] to support many network applications by classifying packets into *flows*. Flows are specified by *classifier*. The packets classified as a flow are processed in the same manner as defined in the action associated with the flow. Each rule specifies a flow based on five header fields, source and destination address IP fields, source and destination port fields, and protocol field. Each field value may be formatted as a prefix, a range, or a singleton value. Each rule also has a priority. When a packet matches multiple rules, this packet is classified as the flow with the highest priority.

With the development of Internet and emergence of SDN, packet classification is no longer just to classify the packets at a single router. The network behaviors for multiple routers need to be considered at the same time. SDN architecture decouples network control and forwarding function. OpenFlow is the de facto standard for SDN where the control plane operated as a centralized controller provides a global view of the network to allow applications to identify network-wide behaviors of packets. Network-wide behaviors are defined by the routing tables and rulesets in all routers of the network. Network-wide behavior can display how a packet traverses in the network and whether a packet will be discarded. Figure 1 shows the global view of a network with 3 routers. Each router maintains a routing table and one rule table. Figure 1(h) shows the routing behavior table computed from three routing tables. Figure 1(g) shows the network topology. The 2-D header space of these rule tables consists of six disjoint blocks for four distinct rule behaviors in Figure 1(j). Each rule behavior is represented by a three-action tuple

| | Dst IP | next port |
|---|---|---|
| R1 | 2:3 | A1 |
| R2 | 4:5 | A2 |
| R3 | 0:3 | A4 |
| R4 | 6:6 | A3 |
| R5 | * | drop |

(a) Routing table A

| | Dst IP | next port |
|---|---|---|
| R1 | 2:3 | B1 |
| R2 | 6:7 | B2 |
| R3 | 5:5 | B3 |
| R4 | 4:7 | B4 |
| R5 | * | drop |

(b) Routing table B

| | Dst IP | next port |
|---|---|---|
| R1 | 0:1 | C3 |
| R2 | 6:7 | C2 |
| R3 | 2:3 | C1 |
| R4 | * | drop |

(c) Routing table C

| Field1 | Field 2 | Action |
|---|---|---|
| * | 0:3 | permit |
| * | 6:7 | permit |
| 2:3 | * | permit |
| * | * | deny |

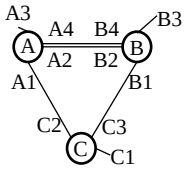
(d) Rule table A

| Field1 | Field2 | Action |
|---|---|---|
| * | 0:3 | permit |
| * | 4:5 | permit |
| 0:3 | * | permit |
| * | * | deny |

(e) Rule table B

| Field1 | Field2 | Action |
|---|---|---|
| * | 4:7 | permit |
| 1:1 | 2:3 | permit |
| * | * | deny |

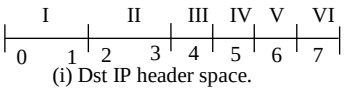
(f) Rule table C



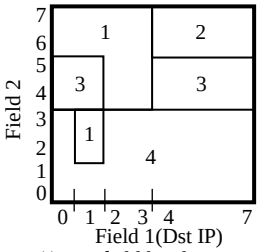

(g) Network topology

| | A | B | C |
|---|---|---|---|
| I | A4 | drop | C3 |
| II | A1 | B1 | C1 |
| III | A2 | B4 | drop |
| IV | A2 | B3 | drop |
| V | A3 | B2 | C2 |
| VI | drop | B2 | C2 |

(h) Routing behaviors




(i) Dst IP header space.

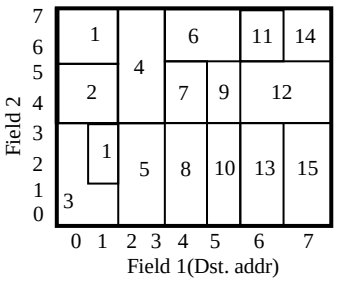



(j) two-field header space

| | A | B | C |
|---|---|---|---|
| 1 | P | P | P |
| 2 | P | D | P |
| 3 | D | P | P |
| 4 | P | P | D |

(k) Rule behaviors

Figure 1. Example of a global view network.




| | |
|---|---|
| 1 | (A4, Drop,C3) |
| 2 | (Drop, Drop,C3) |
| 3 | (A4, Drop, Drop) |
| 4 | (A1,B1,C1) |
| 5 | (A1,B1, Drop) |
| 6 | (A2, Drop, Drop) |
| 7 | (Drop,B4, Drop) |
| 8 | (A2,B4 Drop) |
| 9 | (Drop,B3, Drop) |
| 10 | (A2,B3, Drop) |
| 11 | (A3, Drop,C2) |
| 12 | (Drop,B2,C2) |
| 13 | (A3,B2, Drop) |
| 14 | (Drop, Drop,C2) |
| 15 | (Drop,B2, Drop) |

Figure 2. Example of network-wide behaviors.

 ISBN: 978-1-61208-900-3

($action_A$, $action_B$, $action_C$) in Figure 1(k). After obtaining the routing behavior and rule behavior, we obtain the network-wide behavior of the packets as shown in Figure 2.

Our scheme allows SDN applications running on the control plane to identify network-wide behaviors. If a packet cannot reach its destination, then the border router can directly drop it. The main task is to classify the incoming packets with the classifiers of all routers. The methods designed for a single router is not efficient for global networks since we have to repeat the process until we get all flows. To classify a network-wide behavior efficiently, we need to combine all classifiers into one. However, conventional methods like HiCuts [4] and HyperCuts [7], do not support network-wide behaviors since the search space becomes too complicated.

In this paper, we propose a novel scheme named *Range Encoding Hash Table (REHT)*. REHT is a two-layer hash table-based scheme for solving packet classification problem for global view networking. In layer one, we construct five range encoders for five fields by classifiers. In layer two, we use hash tables to record rules. For an incoming packet, we encode its five field values and access hash tables to obtain its network-wide behavior.

The rest of this paper is organized as follows. Section II briefly reviews related work. Section III shows the overview and details of the proposed REHT scheme. Also, optimization is proposed for ACL rule table. Grouping optimizations which can be adopted by REHT to improve memory usage are proposed in Section IV. The performance evaluation is shown in Section V and conclusions are shown in the last Section.

## II. Related Work

In past years, numerous packet classification schemes have been proposed in the literature. These schemes can be categorized as software solutions and hardware solutions. Software algorithms can also be divided into two categories, decision-tree and field decomposition schemes. HiCuts [4], HyperCuts [7], and EffiCuts [1] are well-known decision trees. BDDs [10] and MDD [11], which can solve network-wide behavior problem are also decision trees. Decision trees see the packet classification problem in the geometric view to cut the search space into smaller subspaces. Field decomposition schemes include BV [5] and RFC [6]. They perform independent searches on each field and combine the intermediate results of all fields to obtain the final results. Hardware schemes usually use parallel search engine, such as Ternary content addressable memory (TCAM) and pipelined design using FPGA. Other encoding schemes can be found in [2][12][13].

Binary decision diagram [8] (BDD) is a decision tree that is used to represent a Boolean function. BDDs can be considered as a compressed representation of sets or relations. In [10], they proposed a control plane tool for packet behavior identification, as known as network-wide behaviors. They first convert forwarding table and rule table to a list of *predicates*. A predicate represents a discontinuous space for an output port or action. Then, they compute separate sets of *atomic predicates* for rule and forwarding predicates. An atomic predicate can be represented by a BDD. For *n* atomic predicates, it can be represented by *n* BDDs. Then, they build the AP tree based on atomic predicates to reduce the number of times to search BDDs.

Boolean functions can be merged into a single multi-valued function to represent a whole classifier by itself. This multi-valued function is represented by a data structure named multi-value decision diagram [9] (MDD), a variant of BDD. In [11], they proposed some optimized method for MDD to update a new behavior. Also, they proposed an algorithm to accelerate classification process by regarding a bunch of header bits as a single variable to analyze time-space tradeoff.

## III. Proposed Scheme

The packet classification (PC) for global view networking depends on multiple routing and rule tables. The address space cutting procedure to get an action, or a network-wide behavior is complicated. We can divide the PC problem of global view network into problems of identifying the routing behavior and rule behavior for the incoming packets. Then we can get the final network-wide behavior by combining the identified routing and rule behaviors. A routing behavior refers to the set of next ports defined by all routing tables of the routers. Similarly, a rule behavior refers to the set of actions defined by all rule tables of the routers.

After obtaining routing and rule behaviors of a packet, we have to combine the next ports and actions. If the rule action of a router is permit, its next port of network-wide behavior remains the same. Otherwise, it changes with rule action. Assume the routing and rule behaviors of a packet with header values (6, 2) are (A3, B2, C2) and (permit, permit, deny), respectively, as shown in Figure 1. After combining operations, the output port of router C becomes “drop”.

The proposed Range Encoding Hash Table (REHT) is a two-layer field-independent based scheme that uses hash tables to store the rule tables. For an incoming packet, REHT encodes the five field values of the packet separately and computes the routing behavior based on the destination IP field encoded value in layer one. In layer two, REHT uses the encoded values of layer one to access the hash tables and compute rule behaviors. Finally, we can obtain the network-wide behavior by combining the routing and rule behaviors.

We first combine all routing tables into a big integrated routing table and all rule tables into a big integrated rule table. Then, we divide the address space of each field into several intervals based on the concept of elementary interval. For the destination address field, we construct the destination address elementary intervals by the destination IP field values of both routing table and rule table. Then, we encode field values into interval numbers call *interval ID*. Layer one contains five range encoders where each range encoder inputs the corresponding field values and outputs the interval IDs. We call the encoded value of the source address field as *source address interval ID*, the encoded value of the destination address field as *destination address interval ID*, and so on. For routing behaviors, we use the integrated routing table to precompute the routing behavior where each destination address interval ID corresponding to a routing behavior.

 ISBN: 978-1-61208-900-3 

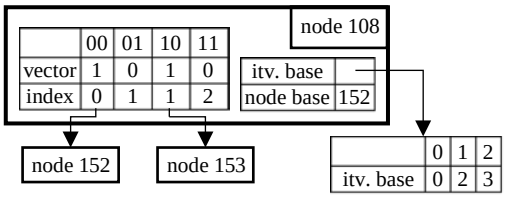



Figure 3. Example node of 2-bit multiway range tree.

Based on encoded values of five dimensions, the rule behavior can be easily stored in the hash tables and the queries of rule behaviors from incoming packet headers can also be computed efficiently.

We know that each field value in five-dimension rule set is either exact value or wildcard (i.e., don't care). Since we cannot hash a rule containing a wildcard without serious duplications, we simple divide the rules into groups based on whether the value of a field is exact value or wildcard, and each group is given a hash table. For each group, we can only consider the field values that are exact values. After accessing all the hash tables from incoming packet headers, we can obtain rules behaviors by the priority encoder. Since we need to access all hash tables to get the rule behavior and it takes a lot of memory accesses, we use possibility bitmap to reduce unneeded memory accesses, described in detail later.

### A. Range Encoder

We first build the respective elementary intervals [3] for each field by both integrated routing and rule tables. The purpose of the encoder is to get the interval ID to which the field value belongs. As the size of each field is different, we use three different methods for the encoder to balance number of memory accesses and memory usage as follows.

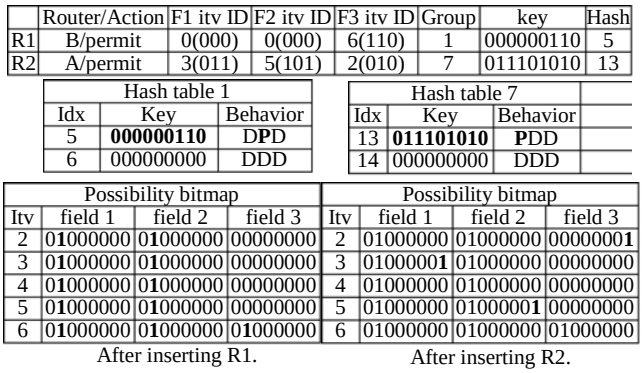
1) *Direct entry mapping for port field (small size)*

For the field of length *m*, we use an array of $2^m$ entries to directly map the address to the corresponding interval ID with only one memory access.

2) *Multiway range tree for IP address (large size)*

Multiway range tree is similar to multiway tries, except for building by endpoint not prefix. Multiway range tree consists of internal nodes and leaf nodes. For the *k*-bit multiway range tree, each internal node holds an array with $2^k$ element corresponding to the value of the *k*-bit chunk in the input key and each element contains a vector and an index. Besides, it holds a node base and an array pointer which points to an array with *n* interval base where *n* depends on the max value of index. The vector is configured so that bit-1 indicates there is a descendant node where the node ID is the sum of base and corresponding index, and the bit-0 indicates there is no descendant node. Also, each leaf node holds an interval base array with $2^k$ elements. Figure 3 shows an example of the data structure of 2-bit multiway range tree.

The searching process starts at the root node and the interval ID is set to 0. Each traverse to a node, we use the most-significant *k*-bit of the input address as the searching key, and then we add the $i^{th}$ item of the interval base array to interval ID which *i* is the index corresponding to the searching key. If the corresponding vector is 1, we shift the input address *k* bits to the left and go to the descendant node. Otherwise, we output the interval ID and terminate the process.



| | Router/Action | F1 itv ID | F2 itv ID | F3 itv ID | Group | key | Hash |
|---|---|---|---|---|---|---|---|
| R1 | B/permit | 0(000) | 0(000) | 6(110) | 1 | 000000110 | 5 |
| R2 | A/permit | 3(011) | 5(101) | 2(010) | 7 | 011101010 | 13 |

Hash table 1

| Idx | Key | Behavior |
|---|---|---|
| 5 | **000000110** | D**P**D |
| 6 | 000000000 | DDD |

Hash table 7

| Idx | Key | Behavior |
|---|---|---|
| 13 | **011101010** | **P**DD |
| 14 | 000000000 | DDD |

Possibility bitmap

| Itv | field 1 | field 2 | field 3 |
|---|---|---|---|
| 2 | 0**1**000000 | 0**1**000000 | 00000000 |
| 3 | 0**1**000000 | 0**1**000000 | 00000000 |
| 4 | 0**1**000000 | 0**1**000000 | 00000000 |
| 5 | 0**1**000000 | 0**1**000000 | 00000000 |
| 6 | 0**1**000000 | 0**1**000000 | 0**1**000000 |

After inserting R1.

Possibility bitmap

| Itv | field 1 | field 2 | field 3 |
|---|---|---|---|
| 2 | 01000000 | 01000000 | 0000000**1** |
| 3 | 0100000**1** | 01000000 | 00000000 |
| 4 | 01000000 | 01000000 | 00000000 |
| 5 | 01000000 | 0100000**1** | 00000000 |
| 6 | 01000000 | 01000000 | 01000000 |

After inserting R2.

Figure 4. Example of inserting a rule.

### B. Hash Table

Given *n*-field rules, we divide them into $2^n$ groups with *n*-bit group IDs. The rule whose field-*i* value is wildcard must belong to the group whose bit-*i* is 0. Each group is given a hash table. Each entry of hash tables holds a key initialized to 0 and a rule behavior initialized to default. To distinguish the rule behaviors of incoming packets and the rule behaviors recorded in hash tables, we call the rule behaviors recorded in hash tables *group behaviors*. The packets' rule behaviors are determined by the query results of group behaviors from incoming packet headers in each hash table.

1) *Insert Rules*

To insert a rule into hash table, we first convert every field value of the rule into interval IDs and combine them as the inserting key. Given hash function *h*, the corresponding index of this rule is *h*(inserting key). If the key of this entry is 0 or equal to the inserting key, we set the value of key to inserting key and update the group behavior as shown in Figure 4. If the key of this entry is neither 0 nor inserting key, which means a collision, we increase the number of entries for hash table until we can put all the rules into hash table without collision.

If any field value of a rule covers multiple intervals, we need to duplicate this rule *m* times where *m* is the product of the number of intervals that each field covers. It is an important issue that a rule may be duplicated hundreds of times in the worst case. To solve this problem, we use two optimized grouping schemes which will be introduced later. In our experiment, we also use cuckoo hashing for layer two. It has better memory usage, but it needs more memory accesses since it has multiple hash functions.

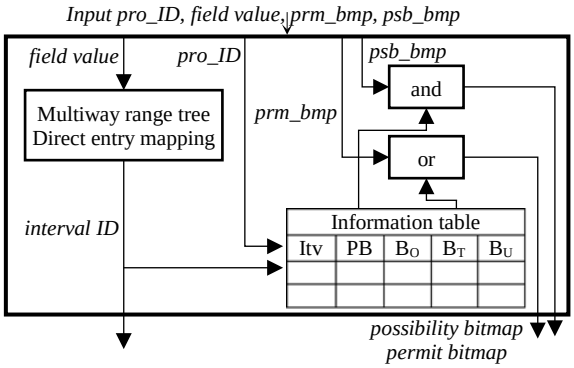



Figure 5. Data structure of the field encoder in layer one.

*2) Possibility Bitmap*

The possibility bitmap is used to confirm which hash tables an interval ID may hit. Given *n* hash tables, we configure a set of *n*-bit possibility bitmap for each interval ID, and each bit corresponds to a hash table. The most significant bit of the bitmap corresponds to hash table 0. If bit-*i* of possibility bitmap for an interval ID is 1, the hash key with this interval ID may hit hash table *i*. Otherwise, the hash key with this interval ID will never hit hash table *i*. Figure 4 shows how to set possibility bitmaps when inserting two rules R1 and R2, where non-relevant entries are not shown.

*3) Searching Process*

To find out the rule behavior of an incoming packet, we use the interval IDs converted from the encoders of layer one as the searching key, and we use and operation on the possibility bitmaps of these intervals. Then, we use the searching key to access the hash tables which the corresponding bit of possibility bitmap is 1. To access a hash table, we identify the key in the corresponding index with searching key. If these two are identical, this group behavior is matched. Otherwise, this hash table is missed. Finally, we can obtain the rule behavior by combining all of matched group behavior from priority encoder.

*C. Optimized Scheme for ACL Rule*

We propose an optimized scheme for the packet classification with ACL rule, such as Stanford backbone network. Based on the property of ACL rule and network-wide behavior, we can improve the performance.

*1) ACL Rule*

As we known, ACL rule contains five fields and only two kinds of actions, permit and deny. In the optimized scheme, we replace the rules of deny action with the rules of permit action in the case of maintaining the original property. Since the actions of rules after replacement are all permit, for group behaviors, we can use *n* bits to represent *n* actions of *n* routers where bit 1 represents permit and bit 0 represent deny. Then, we can use or operation to obtain the rule behaviors instead of priority encoder.

Furthermore, the source port field of ACL rule is always wildcard, so we can use $2^4$ hash tables to record the group behaviors. Also, based on the analysis of ACL rule, some groups may be empty. For ACL rule in Stanford backbone network, the number of hash tables is ten.

```
Input Header H, the input header;
Initial pmb = (all false), permit bitmap;
Initial pbb = (all true), possibility bitmap;
pro_ID = pro_enc(H.protocol);
dstIP_ID = dstIP_enc(H.dstIP, pro_ID, &pmb, &pbb);
srcIP_ID = srcIP_enc(H.srcIP, pro_ID, &pmb, &pbb);
dstPort_ID = dstPort_enc(H.dstPort, pro_ID, &pmb, &pbb);
rout_b = routing _behavior_table[dstIP_ID];
key = key_combiner(srcIP_ID, dstIP_ID, dstPort_ID, pro_ID);
HT = 1;
For i = 0 to 4
START
    If(pmb == all true) Return rout_b;
    If(pbb & HT) hash_table(i, key, &pmb);
    HT = HT << 1;
END
```

Figure 6. Searching process in optimized scheme.

We know that some hash tables only contain one exact field like source or destination address, which means we can obtain group behavior from this field interval ID instead of hash procedure. So, we can remove these two hash tables and put the rules of these tables into the source and destination address *information table* which use the same interval index as possibility bitmap. In other words, each interval $i = 1$ to S is associated with a possibility bitmap and a group behavior. Moreover, since the protocol field only contains three values, TCP, UDP, and don't care, we can merge the two hash tables of which exact fields and wildcard fields are the same except for the protocol field by using three group behaviors to record the case of TCP, UDP and don't care, respectively, denoted by ($B_T$, $B_U$, $B_O$). We then can reduce the number of hash tables to five.

*2) Network-wide Behavior*

We know that we can get the network-wide behavior by combining the routing behavior and rule behavior, but there are two cases that we can get the network-wide behavior without complete procedure. In the first case, if the routing behavior of an incoming packet will eventually drop the packet, then we do not have to get its exact rule behavior. Since no matter what its rule behavior is, the packet will drop. In the second case, since all the actions of rules are permit, for an incoming packet, if one of matched group behaviors for switch *k* is permit, the action of rule behavior for switch *k* is permit. If the rule behavior of an incoming packet is all permit, its network-wide behavior is its routing behavior. So, in the optimized scheme, we can access to the hash table one by one and use or operation to record the matched group behavior. Once all the actions of the rule behavior are permit, the query process can be terminated.

*3) Permit Bitmap*

In the searching process, we use a set of permit bitmap to record the current rule behavior of the packet. The length of permit bitmap is same as the rule behavior, and it is initialized to all false (not permit). Every time accessing to a

TABLE I. STATISTICS OF NETWORK CONFIGURATION.

| | Internet2 | Stanford |
|---|---|---|
| # of routers | 9 | 16 |
| # of prefixes (FIB) | 126,017 | 757,170 |
| # of rules (ACL) | 0 | 1,584 |
| # of header bits of interest | 32 | 88 |

TABLE II. STATISTICS OF STANFORD ACL TABLE.

| | Src. addr | Dst. addr | Dst. port | Protocol |
|---|---|---|---|---|
| # of intervals | 418 | 226 | 55 | 3 |

| Group | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| # of rules | 0 | 0 | 0 | 8 | 16 | 35 | 0 | 57 |
| Group | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| # of rules | 77 | 31 | 0 | 23 | 161 | 11 | 0 | 29 |

| | Information table | Hash table |
|---|---|---|
| # of rules | 167 | 281 |

field encoder or a hash table, the permit bitmap is updated if a group behavior is matched. Once the permit bitmap is all true (permit), the query process can be terminated. If the permit bitmap is not all true at the end of query process, the rule behavior is represented by permit bitmap. Figure 5 shows the complete field encoder in layer one. Figure 6 shows the pseudo code of searching process in optimized scheme.

## IV. GROUPING OPTIMIZATIONS

As described earlier, duplication is an important issue of packet classification that a rule may be duplicated hundreds of times in the worst case. To solve this problem, we propose another two grouping methods for hash tables. In our experiment, we can improve the performance by more than ten times. In other words, we can reduce the total number of duplications to less than one tenth.

The normal grouping is based on whether each field is exact field or wildcard field. However, in some rules, the length of the source and destination address field may be short, or the field value may cover many intervals. We call these heavy rules that duplicate many times. Our goal is to reduce this kind of rules. In the first grouping method, grouping by prefix length, we define the field with length less than or equal to $k$ as wildcard. For example, given $k = 2$, the field value 128.0.0.0/1 is a wildcard field. In our experiment, this method can probably reduce the total number of duplications to less than half. In the second grouping method, grouping by the number of duplications, we decide whether a field value is wildcard or exact field according to the number of intervals it covers. Then, we construct respective elementary interval for wildcard field and exact field. For incoming packets, each encoder output two interval IDs, one used for wildcard field, and another used for exact field. In our experiment, the second grouping method can reduce the total number of duplications to less than one tenth.

## V. PERFORMANCE EVALUATION

Our scheme is evaluated with two real networks: Internet2 and Stanford backbone networks [11]. The network statistics of Internet2 and Stanford backbone network are shown in Table I.

TABLE III. STATISTICS OF MULTIWAY RANGE TREE.

| Header field | | # of nodes | Memory (KB) |
|---|---|---|---|
| Internet2 Dst. address | 8-8-8-8 | 18,416 | 898.5 |
| | 16-8-8 | 18,241 | 985.1 |
| | 12-10-10 | 11,849 | 1,974.5 |
| Stanford Src. address | 8-8-8-8 | 244 | 11.7 |
| | 16-8-8 | 230 | 66.3 |
| | 12-10-10 | 151 | 27.5 |
| Stanford Dst. address | 8-8-8-8 | 1,052 | 46.5 |
| | 16-8-8 | 1,033 | 108.7 |
| | 12-10-10 | 547 | 87.5 |

TABLE IV. MEMORY USAGE.

| | Traditional hashing | Cuckoo hashing |
|---|---|---|
| Memory(KB) | 767.33 | 255.77 |

| | Routing | Src. addr | Dst. addr | Dst. port |
|---|---|---|---|---|
| Memory(KB) | 5.44 | 2.71 | 1.46 | 0.36 |

TABLE V. RESUTLS OF THREE GROUPING.

| Grouping | # of items in hash tables |
|---|---|
| Original | 17,034 |
| Optimization 1 | 7,307 |
| Optimization 2 | 1,092 |

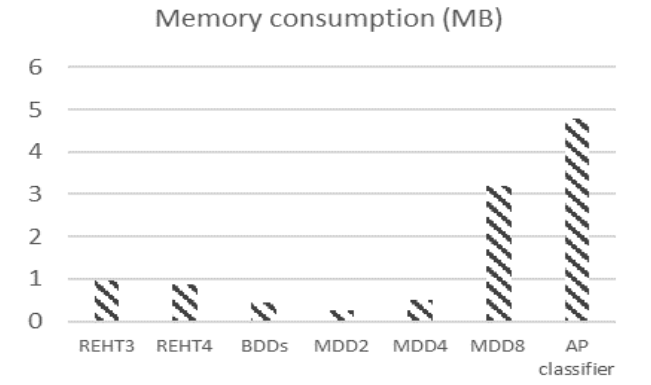
### A. Experimental Analysis

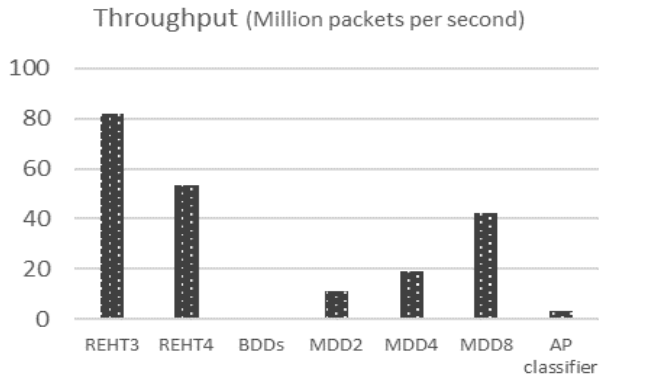
For our proposed scheme, the performance depends on an important factor, the number of intervals in each dimension. If the number of intervals is large, the data structure for encoder is large. Also, the duplications in hash tables may be large. Furthermore, the number of intervals in destination address field is larger than other fields, so the most important factor is the number of intervals in destination address field. The average number of intervals in Internet2 integrated routing table is 14383, and the number of routing behaviors is 457. The number of intervals in Stanford integrate routing table is 2086, and the number of routing behaviors is 507. As a result, the multiway range tree for Stanford is better than Internet2. Table II shows the statistic of Stanford integrate ACL rule table. In optimized scheme, there are probably 37% of the rules that can be recorded in the information tables.

### B. Experimental Results

We show the performance results of the proposed scheme in two parts, range encoding and hash procedure. In range encoding, the multiway range trees for the IP address fields are implemented in three different configurations, denoted by 8-8-8-8, 16-8-8, 12-10-10. Notation 8-8-8-8 means that the multiway range tree is organized as a four-level data structure and each level takes 8 bits of the 32-bit address space. Notations 16-8-8 and 12-10-10 mean that the multiway range tree is organized as a three-level data structure such that the first level takes 16 and 12 bits; the next two levels take 8 and 10 bits each, respectively. Table III shows the statistics of multiway range tree. The number of nodes is associcated with the number of intervals. By comparing three-level and four-level data structure, three-level needs more memory consumption but less memory accesses. By comparing

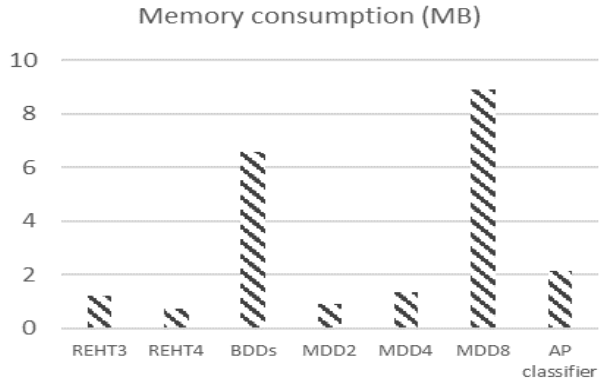
 ISBN: 978-1-61208-900-3 

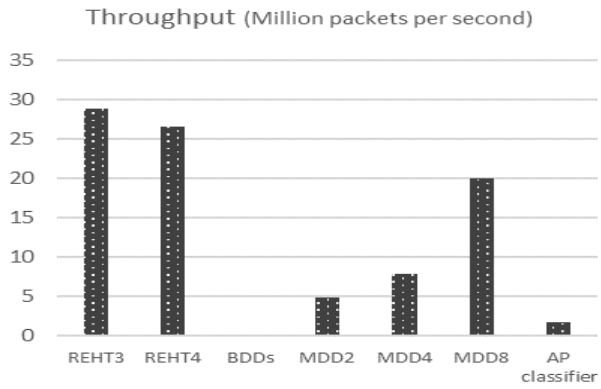



| | REHT3 | REHT4 | BDDs | MDD2 | MDD4 | MDD8 | AP classifier |
|---|---|---|---|---|---|---|---|
| Memory | 0.962 | 0.877 | 0.454 | 0.26 | 0.51 | 3.2 | 4.79 |
| Throughput | 82.05 | 53.33 | 0.085 | 11.3 | 18.95 | 42.6 | 3.4 |

Figure 7. Performance comparison of Internet2.




| | REHT3 | REHT4 | BDDs | MDD2 | MDD4 | MDD8 | AP classifier |
|---|---|---|---|---|---|---|---|
| Memory | 1.23 | 0.753 | 6.568 | 0.9 | 1.35 | 8.89 | 2.15 |
| Throughput | 28.828 | 26.6 | 0.006 | 4.95 | 7.86 | 20.02 | 1.8 |

Figure 8. Performance comparison of Stanford.

configurations 16-8-8 and 12-10-10, configuration 16-8-8 is more suitable for Internet2 and configuration 12-10-10 is more suitable for Stanford backbone network.

Hash procedure is implemented by two hashing methods, traditional hashing and cuckoo hashing. Traditional hashing in REHT is implemented as follows. We set the size of the hash table as (3 * *# of hash items*) and each hash entry can hold three items. According to our experiment, it is the smallest hash table size that can record all rules without collision. Table IV shows the memory consumption of two hashing method for Stanford. The memory consumption of cuckoo hashing is smaller, but traditional hashing needs less memory accesses.

Table IV also shows the memory consumption of information table. Routing information table contains the routing behaviors. Other information tables contain the possibility bitmaps and group behaviors. TABLE V shows the results of three grouping method. Optimization 1 is divide the rules by field length, and optimization 2 is divide the rules by number of intervals that a field value covers. Optimization 2 can reduce the number of items to less than 10%.

We compare our proposed scheme with BDDs and MDD schemes by the same network configurations, Internet2 and Stanford backbone networks. We use instruction 'rdstc' (read time stamp counter) to measure the CPU clock ticks of the searching process and compute the average throughput that is defined as CPU cycles per search. The performance results are shown in Figure 7 and Figure 8. REHT3 is constructed by range encoding configuration 16-8-8 (Internet2) or 12-10-10 (Stanford), and REHT4 is constructed by range encoding configuration 8-8-8-8. Both REHT3 and REHT4 use traditional hashing since it has better classification speed. MDD2/4/8 represents 2/4/8-bit multiway MDD. Since the REHT3 and REHT4 for one dimensional lookup just consist of multiway range tree, the number of memory accesses of REHT3/4 is equal to or less than 3/4. The number of memory accesses of MDD2/4/8 for Internet2 is 16/8/4. So, the throughput of REHT is much better than BDDs and MDD (k = 2/4). For Stanford backbone network, REHT3 has the highest throughput and REHT4 has the smallest memory usage. REHT4 also has second high throughput. The worst case number of memory accesses of REHT for Stanford is sum of the accesses in range encoders and hash tables. The worst case of REHT3 is 12, and the worst case of REHT4 is 14. The number of memory accesses of MDD8 for Stanford is always 11. However, REHT can avoid unnecessary memory accesses, so the throughput of REHT is better than MDD8. For five-dimension header, the memory usage of decision-tree based schemes increase. On the other hand, the memory usage of REHT do not increase too much. Also, REHT can reach

 ISBN: 978-1-61208-900-3 

fast classification speed due to hash procedure. Since AP classifier [10] optimized BDD by reducing the number of searched BDDs, its throughput is only better than original BDD and its memory usage is higher than original BDD.

For throughput, as our single operation is simple enough like BDDs and MDD, the number of memory accesses of REHT is less than or equal to BDDs and MDD. This is why the throughput of the proposed scheme is better than BDDs and MDD. For memory usage, as described in the grouping optimization, duplication is an important issue in our encoding scheme. It may cause serious rule duplication when the IP/port fields have more wildcard (but actually not). For the limitation of REHT, the rule table configuration can't consist of a deny rule that has a higher priority than any permit rule since we only consider the permit action in our method. It can be extended to the configuration with more than one action, which is our future work.

To allow the proposed scheme working with switches, we only need to find another way to efficiently encode the mac address which is a singleton value field. Also, we have add another field VLAN ID in the rules to make sure to which VLAN the classified network behaviors are related. Classbench is a suite of tools for benchmarking packet classification algorithms to produce synthetic filter or rule sets that accurately model the characteristics of various types of networks. We need such tool to support that REHT is suitable for some kinds of network configuration. For the future work, we try to develop a rule generator for different network configurations. The difference from ClassBench is that the rule generator for global view networking has to identify the correctness for packet routing. Also, the rules gererated for different routers should have some common prefixes. Other than generating the tables to model the characteristics of real networks, we have to do research to identify that every route and rule action in the global network is reasonable.

## VI. CONCLUSIONS

In this paper, we proposed the Range Encoding Hash Table (REHT) packet classification scheme for global view networking. For multiple routing tables and rule tables, we first build five encoders for 5 fields and convert the corresponding field values into interval IDs. The destination address interval IDs can correspond to the matched routing behaviors and possibility bitmaps. Other fields interval IDs can correspond to the associated possibility bitmap. By using these interval IDs, we can record and query the rule behaviors efficiently in hash tables. Also, the possibility bitmap can reduce the unneeded hash table accesses. Finally, we can obtain the network-wide behaviors by combining the routing behaviors and rule behaviors.

As we encode the field values of packet headers separately, we can avoid the memory explosion that decision-tree based schemes may happen. Also, we use hashing method to record the rules instead of cross-products so that the memory consumption of REHT can be small while the classification speed can be fast.

## REFERENCES


[1] B. Vamanan, G.Voskuilen, and T. Vijaykumar, “EffiCuts: Optimizing Packet Classification for Memory and Throughput,” in *ACM SIGCOMM*, pp. 207-218, 2010.

[2] D. E. Taylor, “Survey and Taxonomy of Packet Classification Techniques,” in *ACM Computing Surveys*, vol. 37, no. 3, pp. 238-275, Sep. 2005.

[3] Y.-K. Chang and Y.-C. Lin, "Dynamic Segment Trees for Ranges and Prefixes," IEEE Transactions on Computers, VOL. 56, NO. 6, pp. 769-784, June 2007.

[4] P. Gupta and N. McKeown, “Packet Classification Using Hierarchical Intelligent Cuttings,” in *Proceedings of IEEE High-Performance Interconnects*, pp. 34-41, 1999.

[5] T. V. Lakshman and D. Stiliadis, “High-Speed Policy-based Packet Forwarding Using Efficient Multi-dimensional Range Matching,” in *Proceedings of the ACM SIGCOMM '98 conference on Applications, technologies, architectures, and protocols for computer communication*, pp. 203-214, 1998.

[6] P. Gupta, and N. McKeown, “Packet Classification on Multiple Fields,” in *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication*, pp. 147-160, 1999.

[7] S. Singh, F. Baboescu, G. Varghese, and J. Wang, “Packet Classification Using Multidimensional Cutting,” in *Proceedings of ACM Special Interest Group on Data Communication*, pp. 213-224, 2003.

[8] R.E. Bryant, “Graph-based algorithms for boolean function manipulation,” *IEEE Transactions on Computers*, pp.677–691, 1986.

[9] A. Srinivasan, T. Ham, S. Malik, and R.K. Brayton, “Algorithms for discrete function manipulation,” in *IEEE ICCAD*, pages 92–95, 1990.

[10] H. Z. Wang, C. Qian, Ye Yu, H. K. Yang and Simon S. Lam, “Practical network-wide packet behavior identification by AP classifier,” in *IEEE/ACM Transactions on Networking*, vol. 25, pp. 2886-2899, 2017.

[11] T. Inoue, T. Mano, K. Mizutani, S. Minato, and O. Akashi, “Fast packet classification algorithm for network-wide forwarding behaviors,” in *2018 Computer Communications*, vol. 116, pp. 101-117.

[12] Y. K. Chang, C. C. Su, Y. C. Lin, and S. Y. Hsieh, “Efficient Gray Code Based Range Encoding Schemes for Packet Classification in TCAM”, *IEEE/ACM Transactions on Networking*, pp. 1201-1214, 2013.

[13] Y. K. Chang, Y. S. Lin, and C. C. Su, “A High-Speed and Memory Efficient Pipeline Architecture for Packet Classification,” *Proc. the International IEEE Symposium on Field-Programmable Custom Computing Machines* (*FCCM*), pp.215 - 218, 2010.


 ISBN: 978-1-61208-900-3 

# Comparative Study of RIP, OSPF and EIGRP Protocols to Manage WSN-IoT Traffic vs IPTV Traffic Using Cisco Packet Tracer

José Luis García-Navas, Laura Garcia, Oscar Romero, Jaime Lloret
Instituto de Investigación para la Gestión Integrada de Zonas Costeras
Universitat Politècnica de València
Gandía, Valencia (Spain)
email: jogarna3@teleco.upv.es, laugarg2@teleco.upv.es, oromero@dcom.upv.es, jlloret@dcom.upv.es

Pascal Lorenz
Network and Telecommunication Research Group
University of Haute Alsace
Colmar, France
email: lorenz@ieee.org

***Abstract*— Wireless Sensor Networks (WSN) play a highly important role in current life. WSN implementation is constantly growing because of their wide range of applications, such as agriculture, health care, sport, etc. There is no doubt about the advantages of WSN, but there is a problem derived from the fact that the networks are not designed to managed and prioritize Internet of Things (IoT) and WSN traffic. Regarding to this problem, in this paper, a study of IoT data traffic through a traditional network is carried out using Packet Tracer tool. This research compares and analyzes Routing Information Protocol (RIP), Open Shortest Path First (OSPF) and Enhanced Interior Gateway Routing Protocol (EIGRP) routing protocols performance to manage IoT data traffic coming from two different agriculture WSN deployed in different subnetworks and directed to an IoT server. Moreover, IoT data Traffic is going to be analyzed in three different scenarios, transmitted without any other traffic through the network and two different scenarios with Internet Protocol Television (IPTV) traffic sent across the network. IoT update packets will be captured and Round Trip Time (RTT) data will be obtained and analyzed to compare routing protocols performance.**

***Keywords- Wireless Sensor Network; Routing Protocol; RIP; OSPF; EIGRP; IoT traffic; Round Trip Time.***

## I. INTRODUCTION

WSN include several sensor nodes whose applications are widely different, such as agriculture, health care, sport, energy, traffic management, etc. As WSN can be used for monitoring, different applications through the Internet appears the concept Internet of Things (IoT) [1]. Moreover, WSN are playing a key role in the industry and in academic research. They make possible to offer solutions for a wide range of applications. To achieve the necessities and requirements of WSN and the applications used on them, to provide efficient network architectures and suitable telecommunication standard is mandatory [2].

IoT is an emerging field both for researchers and for end-users who wants to benefit of it. IoT creates a world where all electronic devices are connected between them and communicate to monitor human life parameters in different fields. IoT's goal is to create a better world for humans [3]. IoT technologies can be used for multiple reasons, such as Health Care, i.e., monitoring temperature and heartbeat for tracking patients' health and using wireless devices to communicate data [4], or agriculture that is one of the areas on which IoT is widely implemented. A well, IoT can be useful to monitor different aspects, such as controlling production of fruit-bearing trees using image processing techniques [5].

IoT presents a massive ecosystem involving elements, applications, functions, services, and network traffic management. A possible problem derived from this type of ecosystem is that traditional networks are not designed to transmit IoT information, and IoT traffic management can be a challenge [6]. In fact, one of the main lines of investigation related to IoT traffic is focused on characterizing and classifying IoT traffic. The amount and variety of data will be useful to predict and improve traffic management. To obtain this type of information, it is important to collect and synthesize traffic traces in different scenarios with a diversity of IoT devices and in different periods of time. After collecting, traffic traces can be analyzed characterize statistical attributes and develop a classification method [7].

Regarding to data traffic analysis, Packet Tracer [8] is a virtual networking simulation software developed by Cisco that can be used to learn and understand concepts of networks. Cisco Packet Tracer offers a variety of network components that represent real devices as they look in reality and work the same way. Material offered by Packet Tracer is beneficial to simulate real network and gives the opportunity to interconnect and configure devices to create a network. By the same way, Packet Tracer includes IoT devices and implements IoT functionalities, such as smart devices, sensors, actuators, etc. Because of that, it is considered as a useful researching tool in the design and modernization of networks and in the educational process for the study of networks [8].

This paper presents a study of IoT data traffic through a traditional network. Two different WSNs are deployed in different subnets of the network. WSNs are composed of microcontrollers that monitors environmental parameters, such as temperature and humidity and send this information

 ISBN: 978-1-61208-900-3 

to an IoT server located in another subnet. IoT data are analyzed through the network and compared using different routing protocol, such as RIP, OSPF and EIGRP. These protocols have been selected for being the most used in traditional networks, they are easy to implement and to be computed. Moreover, IoT data are analyzed in a clear scenario referring to traffic as well as in a scenario on which IPTV traffic is managed by the network.

The remainder of this paper is organized as follows. Section II presents some related work. The network structure and design carried out in this paper is detailed in Section III. Section IV describes how the simulation has been performed in this research. Simulation and results are summarized in Section V. Finally, Section VI describes conclusions obtained and future work.

## II. RELATED WORK

This section presents different works related with WSN, IoT traffic analysis and the network simulation tool Packet Tracer.

García-Navas et al., [9] presented a practical study that shows an IoT prototype for WSN to measure soil moisture and compare it with a commercial soil moisture sensor. Rocher et al. [10] proposed a WSN solution based on ESP32 board program with Arduino IDE and Wi-Fi technology to control sewerage. The system controls if it is raining or not and controls different scenarios inside the pipelines. Elkin et al. [11] discussed existing methods and algorithms for automated management of traffic flows, with the purpose of applying the Internet of Things technology to the organization of road infrastructure for the dynamic management of traffic flows. Their results showed that IoT technologies significantly reduce the waiting time for cars in the queue at intersections, the total travel time, save fuel, reduce harmful emissions into the atmosphere, reduce the travel time of emergency vehicles to their destination, solve the parking problem and still show many other positive effects.

Some other authors have based their investigations on data traffic analysis, such as Hamid et al. [12] who presented a survey of emerging trends of network traffic classification in IoT and the utilization of traffic classification in its applications. The paper compared the legacy of traffic classification methods and presented an overview of traditional models. Moreover, it included a taxonomy of the current network traffic classification within the IoT context. In their paper, they tried to expose different issues raised in IoT that have been addressed with traffic classification. They concluded that traffic classification in the IoT domain is more challenging in comparison with non-IoT domain, because of the high heterogeneity in the IoT domain. Finally, they highlighted current challenges and possible future direction in IoT traffic network traffic classification. Charyyev et al. [13] demonstrated that an external observer passively sniffing the network traffic can infer IoT device activities, after classifying device events. They evaluated and compared ten machine learning algorithms to classify IoT device events, analyze the impact of different interaction modes with devices, on the performance of classifiers, determine the influence of Local Area Networks (LAN) vs Wide Area Networks (WAN) interaction with the device, and ascertain the effect of region from which the device is connected.

Packet Tracer as a simulation tool can be used to analyze and predict network performance. Teshabaev et al. [14] analyzed, studied, simulated, and modelled a multiservice network on Packet Tracer, to determine the value of delays to increasing value size of Internet Control Message Protocol (ICMP) packet using OSPF protocol. They concluded that the longer the ping length, the more information passes per unit of time. Finally, they considered their research useful in the design and modernization of networks and in the educational process for the development and use of the Packet Tracer program for the study of path of various networks. Dumitrache et al. [15] focused their study on the comparative analysis of the routing protocols RIPv2, OSPF and EIGRP using the soft Cisco Packet Tracer. Their study is an example on how Packet Tracer can be used to analyze network performance and compare routing protocols. Packet Tracer allows designing and simulating virtual networks and strengthen the network security.

As well as Packet Tracer can be used to analyze network traffic performance, thanks to the last Packet Tracer software versions IoT devices and IoT traffic can be analyzed. Ashok et al. in [16] worked with the basic idea that cisco packet tracer can be used to implement smart home. Cisco Packet Tracer offers a variety of network components that represent a real network and gives the opportunity to interconnect and configure devices to create a network. Moreover, it implements IoT functionalities (smart devices, sensors, actuators, etc.). Authors used the latest cisco packet tracer version to introduce smart home, using the home portal for home automation and record smart devices for monitoring them and Microcontroller (MCU-PT) to connect various sensors as well as Internet of Everything (IoE) devices. Gwangwava et al. main objective in [17] was to advance research in the development and implementation of IoT systems. The article bridges the gap on IoT development and deployment. It lays out a quick rollout strategy by using a digital platform that has inbuilt IoT objects and programming capabilities. The article reviewed literature on IoT at different levels and it is a case study of a fertilizer manufacturing plant. Their simulation was only focused on monitoring saturated steam temperature, converter head temperature, and neutralization temperature respectively. Finally, they explain that the model can be extended including key process parameters and adding more levels to the network.

Regarding the related work shown above, it is noted that IoT and WSN are two highly studied topics due to their possible applications and their importance in the modern lifestyle. Furthermore, the analysis of network performance has been studied since the beginning of the use of networks. But it is not so common the study of the network performance managing IoT-WSN traffic. Because of that fact in this paper, we decided to analyze how the most used traditional network's Routing Protocols (RIP, OSPF and EIGRP) manage to exchange IoT packets and how IoT data

 ISBN: 978-1-61208-900-3 

traffic vary depending on the routing protocol. To carry out this study we decided to use Packet Tracer tool to simulate the network performance.

## III. NETWORK DESIGN

The aim of this paper is to analyze how IoT data traffic properties vary depending on the routing protocol used and the type of traffic that the network is managing. The scenario to be analyzed consists of a network divided on 4 subnets, two of them will be WSN and one of them will have an IoT server and an IPTV server. The Core Network will consist of several routers organized to allow multipath between subnets. To create a good scenario to carry out this analysis, some designing criteria have been stablished.

### A. Designing criteria

In this section, the criteria to design the network analyzed in this paper is defined:

- The network will have two different WSN.
- Each WSN will consist of MCU (microcontroller) boards with one humidity sensor and one temperature sensor.
- All MCU boards of each WSN will be connected to a Gateway, through which sensor data will be sent.
- Apart from the two WSN, the network will have one subnetwork containing one IoT server and one IPTV server, and another subnetwork containing some computers.
- The Core Network will consist of various routers interconnected allowing multiple paths between subnetworks.
- Each subnetwork will have a Dynamic Host Configuration Protocol (DHCP) server to automatically assign IP address.

Figure 1 shows the network designed in Packet Tracer. The network consists of 4 subnetworks interconnected by a Core Network that allows multipath between the WSN and servers.

### B. Network Performance

Every microcontroller will be programmed to read temperature and humidity data from sensors and to send it to the IoT Server every second. MCUs will connect to the IoT server using an admin account, which can be used as well to remotely control sensors data.

Once the network is totally operational and MCUs are connected to Gateways, paths between MCUs and IoT Server are created and MCUs stablish connection with the server. MCUs are programmed to connect both to their Gateway and to the IoT server. Gateways are configured to accept connections by password. Finally, the IoT server is configured to accept connections by creating accounts. As soon as the MCU is connected it starts to send information to the server. Information sent to the Server is updated every time a packet is received, so the server shows Real-Time temperature and humidity measured for each MCU.

### C. Routing protocols

The main objective of the paper is to analyze IoT traffic management in different network scenarios. For that reasons, three different scenarios have been defined. Each scenario differs from the rest on the routing protocol used on the routers on the network. RIP, OSPF and EIGRP are the routing protocols selected to analyze how the IoT traffic management can vary.

## IV. SIMULATION DESIGN

The simulation carried out in this paper is focused on obtaining data about IoT traffic on traditional networks. IoT traffic can be considered critical depending on his use and applications, because of that, it is important how this traffic is managed by the network and how different situations can affect to IoT Traffic.

The data that is going to be captured is time since every MCU update sensor data until it receives an ACK message, that means RTT of update packets. This data is going to be captured in three different scenarios, using RIP, OSPF and EIGRP. After analyzing IoT traffic in scenarios on which no more information is exchanged through the network, this data is going to be captured when IPTV traffic is exchanged through the network.

This simulation is carried out to analyze how routing protocols and traffic flows with high quality of service requirements, such as IPTV, can affect IoT traffic management.

Regarding to the type of packets exchanged during the simulation, apart from the ones needed to update RIP, OSPF and EIGRP routing protocol tables, packets exchanged are IoT packets, Transmission Control Protocol (TCP) IoT packets, TCP packets and IPTV packets.

- IoT packets transmit sensor data from temperature and humidity sensor to the MCU microcontrollers.
- TCP IoT packets transmit temperature and humidity sensors data updates from the MCU microcontrollers to the IoT Server.
- TCP packets transmit ACK packets from IoT server to MCU microcontrollers, to inform that the server has received the TCP IoT packet.
- IPTV packets are packets generated by a server using the Traffic Generator tool provided by Packet Tracer. The IPTV service that is going to be used is High-Definition Television (HDTV) with a bit rate of 8 Mb/s.

For each one of the three scenarios proposed, RTT information of IoT traffic is going to be captured in three different data flows situations. First, IoT traffic is captured without any other flows through the network. Secondly, IoT traffic is captured whereas IPTV traffic is sent from the IPTV server to one PC on subnetwork on which no WSN is configured. Finally, IoT traffic is captured whereas three different IPTV flows are sent by the IPTV server, one for each subnetwork.

All data will be captured and analyzed to obtain every significant information that can be used in future work to improve IoT traffic management.

 ISBN: 978-1-61208-900-3 

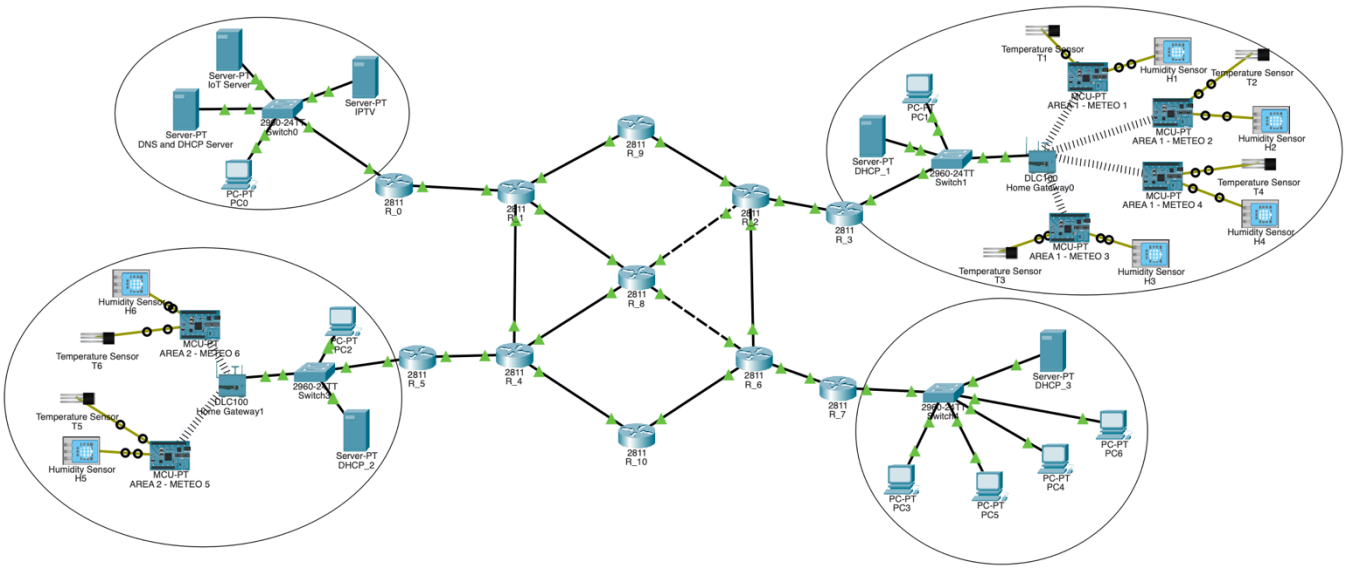



Figure 1. Network designed in Packet Tracer to analyze IoT traffic management

## V. SIMULATION AND RESULTS

In this section, simulation results obtained by the research carried out are shown and analyzed. Routing protocols performance is compared on three different situations which vary on the type and amount of traffic managed by the network.

Two different WSN have been designed and simulated so it is important to bear it in mind whereas analyzing data. The two different WSN are identified by Area 1 and Area 2, and MCU microcontroller are identified by Meteo X. Each data shown and analyzed in this paper is the mean of 5 different data measures. The network designed on packet tracer has been studied using packet tracer simulation mode, and 5 consecutive RTT IoT packets have been captured for each one of the six MCU microcontrollers.

Figure 2 shows RTT data of IoT traffic on A Situation on which no other traffic is managed by the network. As it is shown, the three routing protocols perform in similar ways, but EIGRP seems to be lower than RIP and OSPF. These other two protocols show similar results comparing theirs results between them.

Figure 3 shows RTT data of IoT traffic on B Situation on which IPTV traffic is sent from the IPTV server to one PC on subnetwork on which no WSN is configured. Comparing Figure 2 and Figure 3 results, RIP protocol shows worst results when IPTV traffic is managed by the network, whereas OSPF and EIGRP routing protocols perform in the same way in both situations.

Finally, Figure 4 shows RTT data of IoT traffic on C Situation on which three different IPTV flows are sent by the IPTV server, one for each subnetwork. While the variation on RTT between situation A and B is very soft and, in some cases, RTT decreases when IPTV traffic is managed, on situation C RTT shows a high increase. Moreover, it seems to be that EIGRP is the one that performs better than the others when high amount of IPTV traffic is managed by the network.

Furthermore, protocols can be compared by analyzing them individually studying the three different situations (A, B, C) for each routing protocol. Table 1 shows the RTT data measured on A Situation, considered as a base situation, and the increase or decrease experienced by the RTT on B situation (△1) and C situation (△2). As can be seen, RIP protocol performs better when there is no other traffic, but if there is other type of traffic, IoT traffic suffers higher delays. OSPF and EIGRP routing protocols performs better on situations on which there is more traffic through the network. EIGRP is the best one to transmit IoT data traffic when IPTV traffic is managed by the network.

 ISBN: 978-1-61208-900-3 

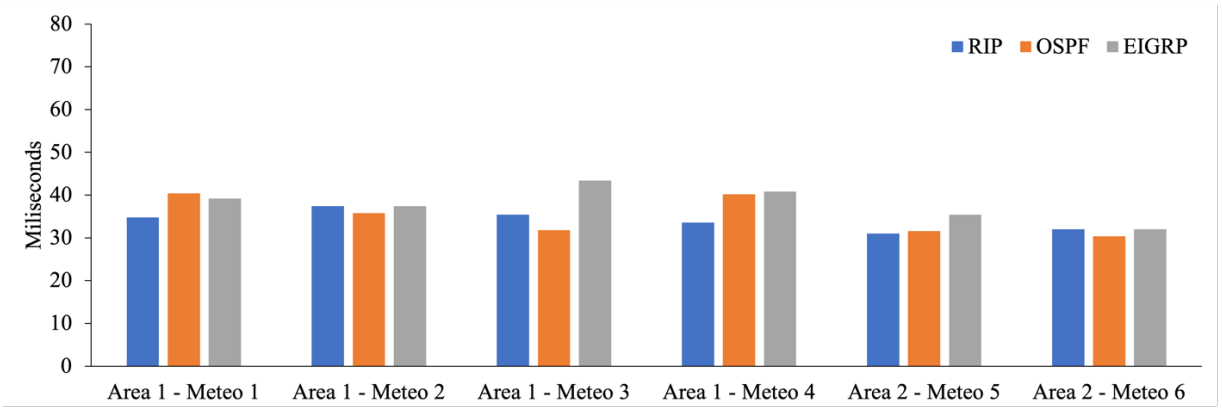



Figure 2. RTT data of only IoT Traffic

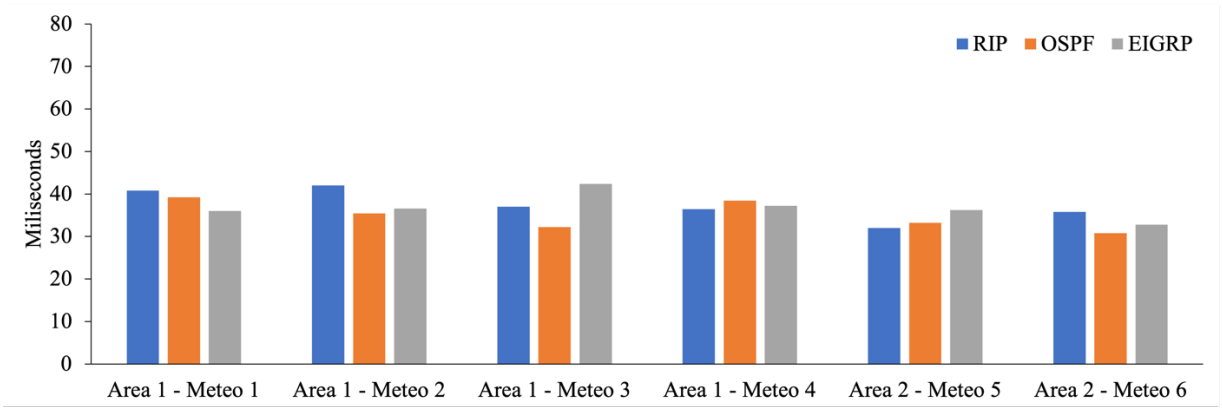



Figure 3. RTT data of IoT Traffic whereas IPTV traffic is sent to network 3 (network without WSN)

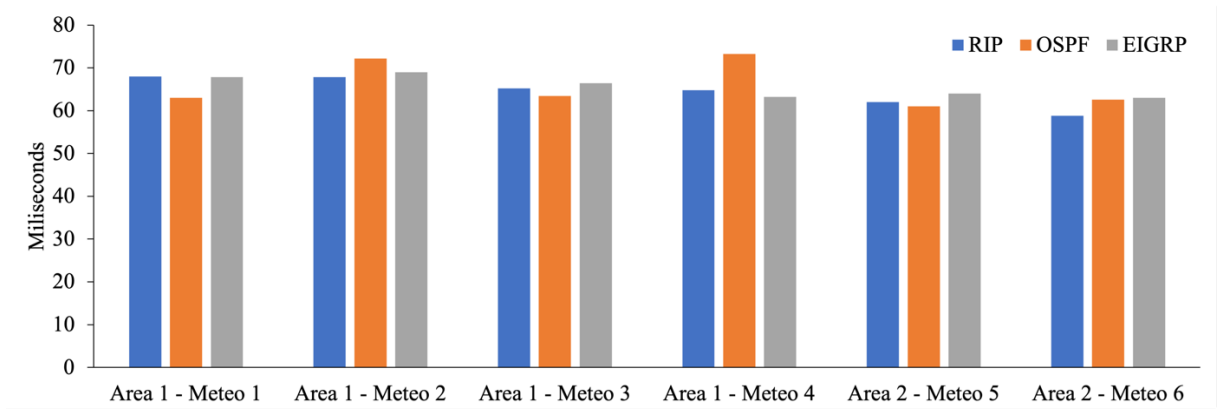



Figure 4. RTT data of IoT Traffic whereas IPTV traffic is sent to the three subnetworks

 ISBN: 978-1-61208-900-3 

TABLE I. COMPARISON BETWEEN RTT IOT TRAFFIC WITHOUT ANY OTHER TRAFFIC AND WITH IPTV TRAFFIC

| | RIP | | | OSPF | | | EIGRP | | |
|---|---|---|---|---|---|---|---|---|---|
| | *RTT IoT Traffic* | *Δ1* | *Δ2* | *RTT IoT Traffic* | *Δ1* | *Δ2* | *RTT IoT Traffic* | *Δ1* | *Δ2* |
| **Area 1 - Meteo 1** | 34.80 | 6.00 | 33.20 | 40.40 | -1.20 | 22.60 | 39.20 | -3.20 | 28.60 |
| **Area 1 - Meteo 2** | 37.40 | 4.60 | 30.40 | 35.80 | -0.40 | 36.40 | 37.40 | -0.80 | 31.60 |
| **Area 1 - Meteo 3** | 35.40 | 1.60 | 29.80 | 31.80 | 0.40 | 31.60 | 43.40 | -1.00 | 23.00 |
| **Area 1 - Meteo 4** | 33.60 | 2.80 | 31.20 | 40.20 | -1.80 | 33.00 | 40.80 | -3.60 | 22.40 |

| | RIP | | | OSPF | | | EIGRP | | |
|---|---|---|---|---|---|---|---|---|---|
| | *RTT IoT Traffic* | *Δ1* | *Δ2* | *RTT IoT Traffic* | *Δ1* | *Δ2* | *RTT IoT Traffic* | *Δ1* | *Δ2* |
| **Area 2 - Meteo 5** | 31.00 | 1.00 | 31.00 | 31.60 | 1.60 | 29.40 | 35.40 | 0.80 | 28.60 |
| **Area 2 - Meteo 6** | 32.00 | 3.80 | 26.80 | 30.40 | 0.40 | 32.30 | 32.00 | 0.80 | 31.00 |

## VI. CONCLUSION

In this paper, a study of how routing protocols (RIP, OSPF and EIGRP) and IPTV traffic can affect to WSN-IoT data traffic have been carried out using Packet Tracer. The routing protocols performance to manage IoT traffic has been studied through a network consisting of 4 subnetworks, on which two of them are WSN. Each one of the routing protocols has been tested on three different situations depending on how much IPTV traffic is managed by the network. In conclusion RIP routing protocol is the one that best manages IoT traffic when no IPTV traffic is needed to be sent. OSPF performs well in similar conditions whereas EIGRP is the worst. On the other hand, when IPVT traffic is managed by the network, EIGRP is the better option while RIP is worst. Finally, OSPF can be considered as a neutral option between RIP and EIGRP on managing WSN-IoT traffic vs IPTV traffic.

In future works, another comparative study can be carried out using different types of traffic instead of IPTV traffic. In addition, this type of study can be implemented in real scenarios so real devices can be tested. Moreover, this type of research can help to create data bases of different types of traffic, to classify and prioritize.

## REFERENCES


[1] S. Bera, S. Misra, S. K. Roy, and M. S. Obaidat, "Soft-WSN: Software-Defined WSN Management System for IoT Applications," IEEE Systems Journal, vol. 12, no. 3, pp. 2074-2081, Sept. 2018.

[2] R. Fantacci, T. Pecorella, R. Viti, and C. Carlini, "A network architecture solution for efficient IOT WSN backhauling: challenges and opportunities," IEEE Wireless Communications, vol. 21, no. 4, pp. 113-119, August. 2014.

[3] S. G. H. Soumyalatha, "Study of IoT: understanding IoT architecture, applications, issues and challenges," In 1st International Conference on Innovations in Computing & Net-working (ICICN 16), CSE, RRCE. International Journal of Advanced Networking & Applications, May. 2016, No. 478, pp. 477-482

[4] D. S. R. Krishnan, S. C. Gupta, and T. Choudhury, "An IoT based Patient Health Monitoring System," 2018 International Conference on Advances in Computing and Communication Engineering (ICACCE), 2018, pp. 01-07.

[5] L. Garcia, et al., "Quantifying the production of fruit-bearing trees using image processing techniques," Proc. The Eighth International Conference on Communications, Computation, Networks and Technologies, IARIA, (INNOV 2019), Nov. 2019, pp. 14-19.

[6] B. K. J. Al-Shammari, N. Al-Aboody, and H. S. Al-Raweshidy, "IoT Traffic Management and Integration in the QoS Supported Network," IEEE Internet of Things Journal, vol. 5, no. 1, pp. 352-370, Feb. 2018.

[7] A. Sivanathan et al., "Characterizing and classifying IoT traffic in smart cities and campuses," 2017 IEEE Conference on Computer Communications Workshops, IEEE (INFOCOM WKSHPS), 2017, pp. 559-564.

[8] S. R. Javid, "Role of packet tracer in learning computer networks," International Journal of Advanced Research in Computer and Communication Engineering, Vol. 3, no. 5, pp. 6508-6511, May. 2014.

[9] J.L. García-Navas, et al., "Practical Study of the Temperature Effect in SoilMoistureMeasurements," Proc. The Eighth International Conference on Communications, Computation, Networks and Technologies, IARIA, (INNOV 2019), Nov. 2019, pp. 7-13.

[10] J. Rocher, J.L. García-Navas, O. Romero, and J. Lloret, "A WSN-based Monitoring System to Control Sewerage," 2019 Sixth International Conference on Internet of Things: Systems, Management and Security, IEEE (IOTSMS 2019), Oct. 2019, pp. 277-282.

[11] D. Elkin, and V. Vyatkin, "IoT in Traffic Management: Review of Existing Methods of Road Traffic Regulation" In Applied Informatics and Cybernetics in Intelligent Systems. Proc. 9th Computer Science On-line Conference. (CSOC 2020) 2020, Vol. 3, pp 536-551.

[12] H. Tahaei, F. Afifi, A. Asemi, F. Zaki, and N. B. Anuar, "The rise of traffic classification in IoT networks: A survey," Journal of Network and Computer Applications, Vol. 154, Article. 102538, pp. 1-20, March. 2020.

[13] B. Charyyev, and M. H. Gunes, "IoT Event Classification Based on Network Traffic," 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2020, pp. 854-859.

[14] T. Z. Teshabaev, M. Z. Yakubova, and O. A. Manankova, "Analysis, research and simulation of a multiservice network based on the Packet Tracer software package to determine the value of delays to increasing value size of ICMP packet," 2020 International Conference on Information Science and Communications Technologies (ICISCT 2020), 2020, pp. 1-4.

[15] C. G. Dumitrache, G. Predusca, L. D. Circiumarescu, N. Angelescu, and D. C. Puchianu, "Comparative study of RIP, OSPF and EIGRP protocols using Cisco Packet Tracer," 2017 5th International Symposium on Electrical and Electronics Engineering (ISEEE), 2017, pp. 1-6.

[16] G. Ashok, P. Akram, M. Neelima, J. Nagasaikumar, and A. Vamshi, "Implementation of smart home by using Packet Tracer," International Journal of Scienfic & Technoloy Research, Vol. 9, no. 2, pp. 678-685, 2020.

[17] N. Gwangwava, and T. Mubvirwi, "Design and Simulation of IoT Systems Using the Cisco Packet Tracer," Advances in Internet of Things, Vol. 11, no. 2, pp. 59-76, 2021.


 ISBN: 978-1-61208-900-3 