

Ben-Gurion University of the Negev

Faculty of Engineering Science

School of Electrical and Computer Engineering

Dept. of Communication Systems Engineering

Fourth Year Engineering Project

Preliminary report

Distributed Caching-based Acceleration Mechanisms in Datacenter Networks

**Project Number:** p-2022-093

**Students:** Shir Granit 205531445, Anna Axalrod 324682475

**Supervisors:** Prof. Chen Avin, Dr. Gabriel Scalosub

**Submitting Date:** 21/11/2021

**Table of Contents:**

[Abstract 3](#_Toc54193468)

[Abstract (English) 3](#_Toc54193469)

[Abstract (Hebrew) 4](#_Toc54193469)

[Project’s Goal 5](#_Toc54193471)

[Research Proposal: Spec Sheet 5](#_Toc54193472)

[Product Essence 5](#_Toc54193472)

[Technologies and Methodologies that Will be Used in the Project 5](#_Toc54193473)

[Work Plan 6](#_Toc54193475)

[Application for our Project 6](#_Toc54193476)

[System Performance Spec 6](#_Toc54193477)

General Scheme [7](#_Toc54193477)

[Literature review 8](#_Toc54193479)

[General background 8](#_Toc54193480)

[Motivation 9](#_Toc54193480)

[Different Approaches for the Problem 9](#_Toc54193480)

[Design Approach 10](#_Toc54193479)

[Project Constraints 10](#_Toc54193480)

[Project Assumptions 10](#_Toc54193480)

[Initial Risks 10](#_Toc54193480)

Defining the Content of the Project  [10](#_Toc54193480)

[Final Testing Proposal 11](#_Toc54193480)

[Budget Estimation, Schedule and Work Division](#_Toc54193479) 12

[Budget Estimation 1](#_Toc54193480)2

[Schedule 1](#_Toc54193480)3

[Bibliography 15](#_Toc54193482)

[Appendix 16](#_Toc54193482)

PRE Evaluation [16](#_Toc54193480)

[Copyright and Confidentiality Form 17](#_Toc54193480)

# Abstract

## Abstract (English):

**Distributed Caching-based Acceleration Mechanisms in Datacenter Networks**

Students Names: Axalrod Anna, Granit Shir

*Our Email address:* [*axalrod@post.bgu.ac.il*](mailto:axalrod@post.bgu.ac.il)

Advisers’ Names: Prof. Avin Chen, Dr. Scalosub Gabriel

Due to high data flow through a data center network, switches in the network face a problem of storing an enormous amount of traffic rules which are necessary for correctly transferring packets in the network. Those rules are usually stored in an external device to which the access slows the network’s performance. The purpose of the project is to find a solution that prevents multiple access to the external device, leading to a significant speedup of the routing process in the network.

In our project, we will develop a distributed solution for the problem presented which will be based on a previous solution that uses caches.

Our proposed method includes creating a network with several switches which send queries to each other to get forwarding rules. In case of missing information, we will request the forwarding rule from the external device (the controller), which keeps all the forwarding rules and holds the switch's cache memory.

We expect a decrease in the number of requests to the external device for rules when sharing the cache between the network's switches, as well as faster network performance.

 This is a funded continuation project with NVIDIA.

Keywords – Network Topologies, Distributed Algorithms, P4, Cache, Software-Defined Networking, LPM, Rule, Mininet, Pipeline, OpenFlow.

## Abstract (Hebrew):

**מנגנוני האצה מבוזרים מבוססי מטמון ברשתות מרכזי נתונים**

שמות הסטודנטים: אקסלרוד אנה, גרניט שיר

*כתובת האי-מייל שלנו:* [*axalrod@post.bgu.ac.il*](mailto:axalrod@post.bgu.ac.il)

שמות המנחים: פרופ' אבין חן, ד"ר סקלוסוב גבריאל

בעקבות זרימת נתונים גבוהה למרכז נתונים, בפני המתגים ברשת, ניצבת בעיה לאחסן כמות עצומה של חוקי תעבורה אשר נדרשים לביצוע העברה תקינה של החבילות ברשת. בדרך כלל אחסון החוקים קורה במכשיר חיצוני שהגישה אליו מאטה משמעותית את ביצועי הרשת. מטרת הפרויקט היא למצוא פתרון אשר ימנע את הגישה המרובה למכשיר החיצוני ובכך יאיץ משמעותית את תהליך הניתוב ברשת.

בפרויקט הגמר שלנו, נפתח פתרון מבוזר לבעיה זו המתבסס על פתרון קודם שעשה שימוש במטמון. בעזרת מתגים ברי תכנות אנו נדמה רשת בה לכל מתג יהיה זיכרון מטמון משלו אשר ינוהל ע"י פרוטוקול רשת מבוזר.

שיטת העבודה שלנו תכלול הקמת רשת עם מס' מתגים אשר ישלחו שאילתות אחד לשני כדי לקבל חוקי ניתוב. במידה שהחוק לא נמצא, נפנה לבקר מרכזי ששומר את כל חוקי הניתוב לקבלת חוק הניתוב המבוקש ושמירתו במטמון המתג.

בתוצאות הפרויקט, אנו מצפות כי כמות הפניות לבקר המרכזי, כאשר יהיה שיתוף מטמון בין מתגי הרשת, תקטן באופן משמעותי וביצועי הרשת יואצו.

זהו פרויקט המשך ממומן בשיתוף אנבידיה.

מילות מפתח: טופולוגיות רשת, אלגוריתמים מבוזרים, P4 (שפת תכנות לרשתות תקשורת), זיכרון מטמון, SDN (רשת מוגדרת תוכנה), LPM (התאמת הקידומת הארוכה ביותר(, חוקים, Mininet (תוכנת אמולציה לרשתות), צינור תעבורה, OpenFlow (פרוטוקול תקשורת).

# Project’s Goal

The goal of the project is to implement and evaluate a P4 switch-based distributed caching mechanism for action rules designed to accelerate datacenter networks. Through this, we can implicate and understand what better solutions can be suggested for switches in datacenters. Implementing different distributed topologies contributes a comparison between the different use of cache in each switch and the communication between the switches and the controller (the rule-making mechanism).

# Research Proposal: Spec Sheet

Name of the project:

Distributed Caching-based Acceleration Mechanisms in Datacenter Networks

## Product Essence:

We plan to create and emulate a distributed datacenter environment, which will consist of switches, hosts, and an SDN network controller with in-network cache management capabilities, in several topologies. Then, we will use this environment to evaluate the performance of a given caching algorithm and measure its efficiency and accuracy.

## Technologies and Methodologies that Will be used in the Project:

* + - OpenFlow protocol – A protocol used in SDNs allowing remote administration of packet forwarding rules in the network's switches [1].
    - Programmable switches (P4) – P4 is a programming language used to control the packet forwarding plane in programmable network devices, like routers and switches [2].
    - Emulation software (Mininet) – Mininet is a network emulator widely used to emulate virtual hosts, routers, network controllers, and switches supporting the OpenFlow protocol.
    - SDN controller (Written in Python) – The controller is the decision-making mechanism that supports the cache-based switches.
    - Traffic generators (Written in C, in a Linux OS script format) – Traffic will be generated for tests and evaluations of the simulation we will create [3].
    - Longest Prefix Matching (LPM) – A method to perform packet forwarding by matching the packet destination subnet to the closest matched network in the rules table.
    - Caching algorithms – To maximize the number of Cache-hits, we will implement different existing Cache Replacement Policies (FIFO, LRU, MRU, etc.) and select the best suited for this project [4].
    - Communication protocol - To be able to transfer packets between the switches, we need a communication protocol for the fast and efficient transfer of information.

## Work Plan:

* + - Use a Mininet environment created last year and adjust it for our purposes. It will include a small datacenter network with switches, hosts, an SDN controller, and a traffic generator, in several different topologies.
    - Design the packet pipeline inside the switches using the P4 programming language and implement a cache inside the switches to store the forwarding rules.
    - Create (or use a given) caching algorithm and deploy it into the switches.
    - Configure a network controller using Python and OpenFlow to manage the datacenter network and handle the Cache-misses.
    - Design a traffic generator, made from different distributions [5].
    - Evaluate the performance of different topologies.

## Applications for our Project:

Our purpose is to accelerate traffic forwarding in large networks, allowing networks in datacenters to increase their performance, reduce their costs on expensive hardware and facilitate their support for larger infrastructures with relatively few hardware changes.

## System Performance Spec:

We expect our project implementations to accelerate current datacenter networks performance. The caching mechanism should be easily deployable in all current programmable switches running in an SDN network. Our input is packets corresponding to several flows, and we expect to see a reduction in the network's packet processing time.

The equipment needed for this research is a PC equipped with emulation software, and enough memory to emulate a significant number of switches and hosts.

## General Scheme:

* + - The current state of the system built last year:

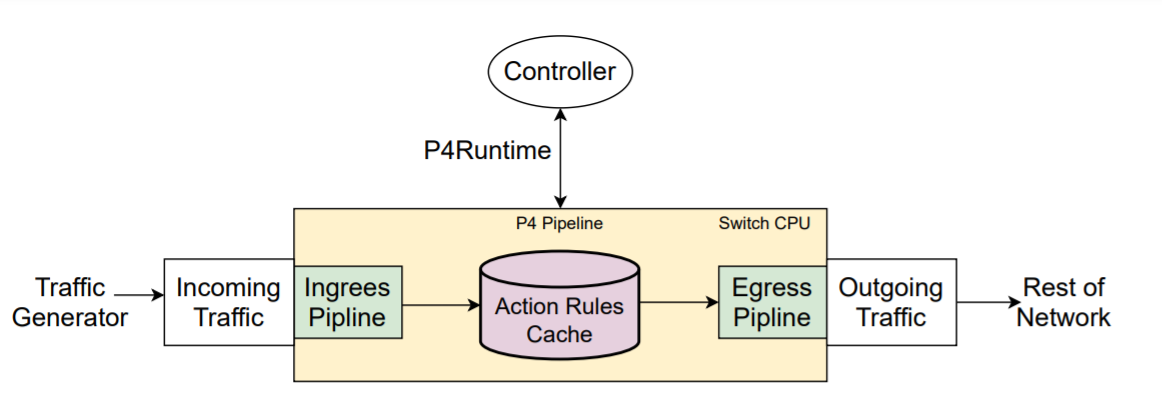
****

Figure 1: General flow of the system that was built last year.

* + - The first desired topology to actualize:

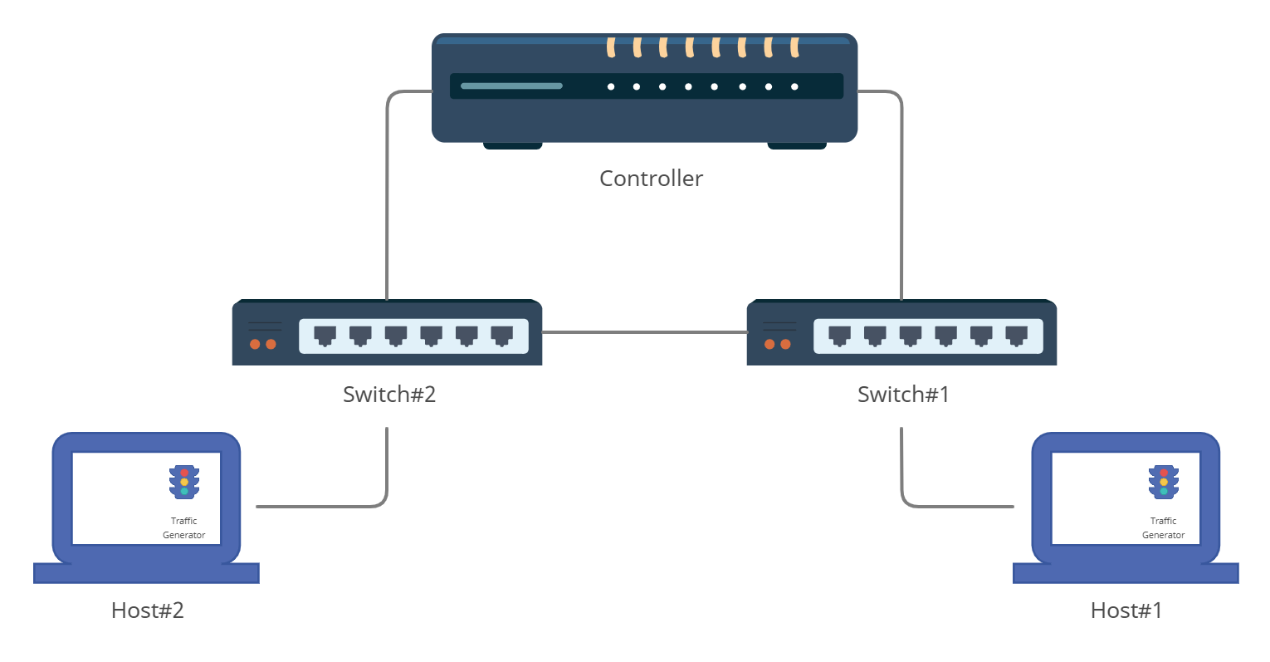


Figure 2: The desired topology of the network.

As described in Fig. 1, the existing model consists only of a single switch with a single controller. In our project, we plan to create the topology described in Fig. 2. This topology will be composed in a distributed way and includes three switches and a single controller. Furthermore, all components will relate to at least one port between one another. In a further stage, we plan to study more complex topologies, in case the topology described in Fig. 2 will yield the expected results. The purpose of this change is to measure how much influence can be driven from connecting and communicating between the two switches.

# Literature review

## General background:

Nowadays modern datacenters that rely on high connectivity between servers and numerous virtual machines, and especially cloud computing solutions, require the use of an approach called Software-defined Network (SDN) [1]. In this approach, the control and management of the network are performed by software in a centralized approach. A network controller software, generally deployed on a server, oversees network management. This controller usually has high computing capabilities and is the “brain” of the network. Fig. 3 attached below is an example of such a network, consisting of seven switches, eight servers (in our model we will treat them as ‘hosts’), and an SDN controller.

Moreover, SDN creates an abstract separation in the way packets are handled in the network. The first abstract layer is the Control Plane, and all the devices that belong to this layer are responsible for defining how packets will be processed in the network. The second layer is the Data Plane, and the job of the devices in this layer is to forward the packets from their source to their destination. In SDN-based networks, the control plane is considered smart and slow, while the data plane is considered fast and simple.

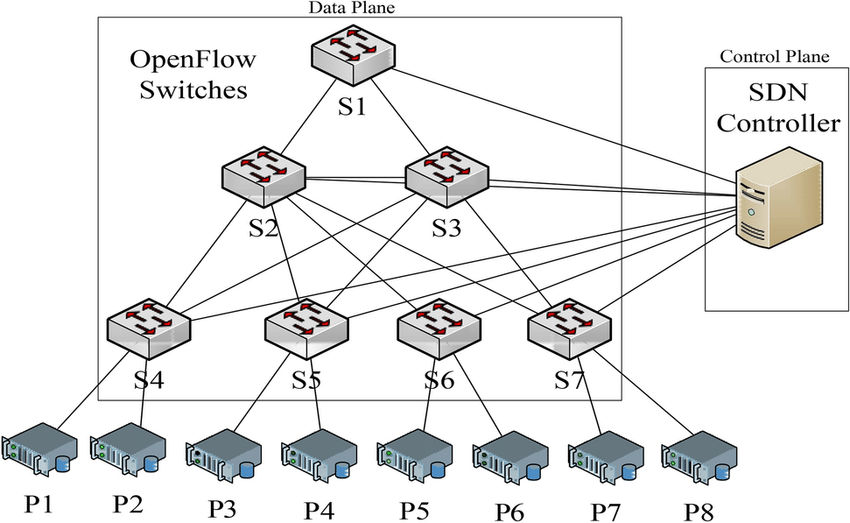


Figure 3: SDN-based three-layer data center architecture [6].

Nowadays, datacenter networks use programmable switches to provide more flexible in-network packet manipulation capabilities. The functionality of these switches can be manipulated by code and introduce new features to the networks that were previously unavailable. The packet processing pipeline of these switches is written in a programming language called P4 [2]. Management protocols like OpenFlow [8] and P4Runtime [9] allow the controller to control and manage these switches.

The forwarding policy of a datacenter is complex and is constantly growing. A modern datacenter might need to handle tens and hundreds of millions of different forwarding rules. Therefore, the controller holds the datacenter forwarding policy, and it is his job to insert and delete the relevant forwarding rules to the switches. These rules tell the switch how to handle a packet that matches a certain rule.

## Motivation:

The use of cloud computing became very popular in the past few years. As a result, datacenter networks are required to handle a great amount of traffic, and the switches in the network are having trouble storing the entire forwarding policy of the network, needed for the traffic to be forwarded correctly. Two main approaches were taken to solve this issue.

## Different Approaches for the Problem:

### Use switches that can store the entire policy that is needed for the traffic:

Usually being used in cloud computing, enabling them to provide line-rate responses to table lookups, which means that all switches are required to hold the forwarding policies of the entire datacenter [10]. The memory used in today's switches to store the forwarding policy is a Ternary Content Addressable Memory (TCAM) [7] which can perform fast parallel table lookup for stored entries. This memory is very expensive and consumes a great amount of energy. Therefore, this solution cannot be scaled to the number of switches needed to operate nowadays datacenters.

### Using cheaper switches and an external device:

A different approach will be to use cheaper switches with relatively small TCAM and offload the entire cloud-scale policy to some external device, with the sole purpose of storing these forwarding rules. Each switch will be connected to such a device, and upon packet arrival, it will ask the device for the correct forwarding rule for that certain packet. This solution degrades the switches' response time to forwarding decisions. Datacenter switches are required to perform at line-rate when deciding where to forward each packet, and the channel between the switch and the external device, as described, creates a bottleneck in the network [7].

# Design Approach

## Project Constraints:

### Mininet-based networks environment cannot (currently) exceed the CPU or bandwidth available on a single server [7].

### Mininet environment cannot (currently) run non-Linux-compatible OpenFlow switches or applications [7].

### There is not currently a way to examine our system in a real-life datacenter network since we do not have access to one.

## Project Assumptions:

### Our main assumption is that the system as described in Fig. 1 is working correctly, and the simulation environment is working with a single switch connected to a single controller, given traffic in the network.

### We assume the availability of an SDN controller.

### We assume that the BMv2 P4 implementation in Mininet is stable.

## Initial Risks:

As explained in ‘Project Constraints’ above, Mininet has limitations, planning our future emulations, we are taking a risk of the emulation system not performing close enough to the real topology of a datacenter.

## Defining the Content of the Project:

* The main content of the project is to create an emulated distributed datacenter environment, which will consist of switches, hosts, and an SDN network controller with in-network cache management capabilities, in several topologies. We will adjust the given emulation environment in Mininet Creating different topologies and adjusting the current algorithms. After implementation, we will generate differently distributed traffic, and evaluate the performance of our implementation under different workloads.
* The main goal of the first part of the project is to check the performance of the distributed system with different cache on every switch. after we finish, we will also have enough data that hopefully can help big data centers to work more effectively in the network. In this project, we will provide an emulation environment with two switches with a cache algorithm, two hosts with a traffic generator, and a controller which will be the "brain" in the emulation. The controller will manage the cache in the switches and have access to all rules.
* In the second part of the project, we will perform the experiments and present the results. In addition, we will be able to measure times and know what is better, i.e., in how many switches we need to check the cache before it is better to send the package to the controller.

## Final Testing Proposal:

* After we will implement our emulation environment and connect the right components, we would like to start to perform the checking of our system. The tests will check what is better, to use distributed cache mechanism as we have shown before, or to turn directly to the controller. Notice that the switches have a limited memory cache so the order of actions is below, every switch will have a different traffic generator so the cache in the switches will be different. The purpose of this action is to use the cache in the switches to the fullest. For every package that the switch will get, it will check if the transfer rules the package is in the cache and if so, we will transfer the package to the destination, in our case we will dump it. On the one side, if the rule is not in the cache, we transfer it to the second switch to check the cache there. If the rule in the other switch does not exist, we will have to transfer the package to the controller, and it will take care of it. In some cases, the controller can decide to change the cache in the switches, so it will send the right message.
* In our project, we will check different types of topologies to know what is the best one for large datacenters. In each test, we will look at the performance of the network after the new topology, compared to the performance of one switch that has its cache and is connected only to the controller. We will start with one neighbor connected to our switch and continue exponentially. For each test, we will measure the time it takes us to finish the trace of data and compare it to one switch that has only his cache. In that way, we can see and examine what is the best number of hopes.

# Budget Estimation, Schedule, and Work Division

## Budget Estimation:

### Working Hours

|  |  |  |  |
| --- | --- | --- | --- |
| Total Cost | Cost per Hour | Amount | Parameter |
|  |  | 12 months | Project Duration |
|  |  | 2 | Number of Students |
| 61,440 ₪ | 80 ₪ | 8 | Weekly Hours per Student |
| 9,600 ₪ | 200 ₪ | 1 | Weekly Consultant Hours with Prof. Avin Chen and Dr. Scalosub Gabriel |
| 14,400 ₪ | 300 ₪ | 1 | Weekly Guidance by Matty Kadosh at Nvidia |
| 85,440 ₪ | **Total Salary** | | |

### Hardware and Software

|  |  |  |
| --- | --- | --- |
| Total Cost | Amount | Device/Software |
| 3,400 ₪ | 1 | Personal Computers |
| 200 ₪ | 1 | Microsoft Office |
| 0 ₪ | 1 | Mininet |
| 3,600 ₪ | **Total Cost** | |

### Total Budget Estimation

|  |  |
| --- | --- |
| Total Cost | Parameter |
| 85,440 ₪ | Salary |
| 21,360 ₪ | Overheads 25% |
| 106,800 ₪ | Salary + Overheads |
| 3,600 ₪ | Hardware and Software |
| 110,400 ₪ | **Summary** |

## Schedule and Work Division:

### Gantt:

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Duration | Estimated Start Date | Estimated Finish Date |
| Learning the systems | 30 days | 01/08/2021 | 31/08/2021 |
| Setting up emulation starting point | 30 days | 01/08/2021 | 31/08/2021 |
| Studying theoretical material | 30 days | 01/09/2021 | 01/10/2021 |
| Writing PDR report | 20 days | 1/10/2021 | 21/10/2021 |
| PDR Submission | 15 days | 09/10/2021 | 24/10/2021 |
| Adjusting the simulation in the network - 2 switches and 1 controller | 63 days | 1/09/2021 | 03/11/2021 |
| PDR-Students Correction | 15 days | 21/10/2021 | 05/11/2021 |
| Open GitHub | 7 days | 4/11/2021 | 11/11/2021 |
| Preliminary Submission | 15 days | 06/11/2021 | 21/11/2021 |
| Read Articles | 41 days | 20/10/2021 | 30/11/2021 |
| PRE report writing | 19 days | 11/11/2021 | 30/11/2021 |
| document last year's code | 48 days | 28/10/2021 | 15/12/2021 |
| Preliminary-Students Correction | 15 days | 17/12/2021 | 01/01/2022 |
| Create video for Technology Ventures | 15 days | 23/12/2021 | 07/01/2022 |
| Technology Ventures Submission | 15 days | 25/12/2021 | 09/01/2022 |
| Exams period | 15 days | 26/01/2022 | 10/02/2022 |
| Technology Ventures-Students Correction | 45 days | 15/01/2022 | 01/03/2022 |
| Progress Submission | 15 days | 26/02/2022 | 13/03/2022 |
| Change topology to 3-switch and 1 controller topology | 10 days | 26/03/2022 | 05/04/2022 |
| Change topology to 3-switch and 2 controllers topology | 10 days | 10/04/2022 | 20/04/2022 |
| Testing our system with overloads of traffic | 15 days | 10/05/2022 | 25/05/2022 |
| Poster Submission | 15 days | 22/05/2022 | 06/06/2022 |
| Presentation Submission | 15 days | 08/06/2022 | 23/06/2022 |
| Results evaluation and conclusions | 30 days | 09/06/2022 | 09/07/2022 |
| Final Submission | 15 days | 09/07/2022 | 24/07/2022 |

### Gantt chart:



# Bibliography

|  |  |
| --- | --- |
| [1] | B. Salisbury, "TCAMs and OpenFlow: What Every SDN Practitioner Must Know" 2 July 2012. |
| [2] | P4 Language Consortium. P4 switch. https://github.com/p4lang. |
| [3] | S. U. A. F. R. S. a. X. H. N. Sarrar, “Leveraging Zipf’s law for traffic offloading”, SIGCOMM Comput. Commun. Rev., 2012. |
| [4] | S. Jiang, "Efficient Caching Algorithms for Memory Management in Computer Systems", College of William & Mary - Arts & Sciences, 2004. |
| [5] | A. D. A. P. Alessio Botta, "A tool for the generation of realistic network workload for emerging networking scenarios", 19 February 2019. |
| [6] | Montazerolghaem, Ahmadreza. "Software-defined load-balanced data center: design, implementation and performance analysis." Cluster Computing 24.2 (2021): 591-610. |
| [7] | V. Ravikumar and R. Mahapatra, "TCAM architecture for IP lookup using prefix properties,", EEE Micro, vol. 24, no. 2, pp. 60-69, March-April 2004. |
| [8] | D. Medhi and K. Ramasamy, "IP Address Lookup Algorithms", Network Routing (Second Edition), 2018. |
| [9] | P4Runtime Protocol. https://github.com/p4lang/p4runtime. |
| [10] | Mininet Overview. http://mininet.org/overview. |

# Appendix:

## PRE Evaluation

**המלצת ציון (ע"י מנחה אקדמי) לדו"ח מכין**

מספר הפרויקט: p-2022-093

שם הפרויקט: מנגנוני האצה מבוזרים מבוססי מטמון ברשתות מרכזי נתונים.

שם המנחים מהמחלקה: פרופסור חן אבין, ד"ר גבריאל סקלוסוב.

שם הסטודנטית: אנה אקסלרוד ת.ז.: 324682475

שם הסטודנטית: שיר גרניט ת.ז.: 205531445

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| % |  | חלש  55-64 | בינוני  65-74 | טוב  75-84 | ט"מ  85-94 | מצוין  95-100 |
| 15 | הבנת הנושא הצורך וסביבת היישום |  |  |  |  |  |
| 15 | חיפוש מקורות והבנת עבודות דומות |  |  |  |  |  |
| 15 | שלמות דף מפרט (הצעת מחקר) |  |  |  |  |  |
| 15 | הצעת תכנון ותכנון הבדיקות הסופיות |  |  |  |  |  |
| 10 | גילוי יוזמה וחריצות |  |  |  |  |  |
| 20 | פתרון בעיות, מקוריות ותרומה אישית  (מעבר למילוי ההנחיות) |  |  |  |  |  |
| 10 | הערכת תקציב, לו"ז וחלוקת עבודה,  ציון מקורות ושלמות כללית |  |  |  |  |  |

הערכת רמת הקושי של הפרויקט: קל מאוד / קל / בינוני / קשה / קשה מאוד.

הערות:

## Copyright and Confidentiality Form:

**טופס כיבוד זכויות יוצרים וסודיות:**

אני מצהיר שלא אעשה שימוש בפרויקט ההנדסי שלי בכל חומר בעל זכויות יוצרים כגון:טקסט, תמונה, אודיו, וידיאו, מוזיקה, סרט, אנימציה, תוכנה, חומרה, תיכנון מעגל, ללא קבלת אישור מראש מבעל הזכויות.

אני מצהיר שאשלב הפרויקט ההנדסי שלי בדוחות סרטונים והרצאות אינפורמציה שאינה נחלת הכלל רק בתנאי שאושרה מראש ע"י בעל הזכויות.

הרישום לפרויקט ההנדסי משמש ההתחייבות שלי לקיים ולכבוד זכויות יוצרים וסודיות.

חתימה: אנה אקסלרוד        תאריך: 10.11.2021

חתימה: שיר גרניט         תאריך: 10.11.2021