I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY

Gavin Jampani
Eish Kapoor
Vivek Ponnala

ENTITLED

SmartNIC Compatible Blockchain

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING

Thesis Advisor

Thesis Advisor

Department Chair

Department Chair
SmartNIC Compatible Blockchain

by

Gavin Jampani
Eish Kapoor
Vivek Ponnala

Submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering
School of Engineering
Santa Clara University

Santa Clara, California
June 9, 2022
SmartNIC Compatible Blockchain

Gavin Jampani
Eish Kapoor
Vivek Ponnala

Computer Science and Engineering
Santa Clara University
June 9, 2022

ABSTRACT

The intent of our Senior Design Thesis was to develop a software infrastructure for Blockchain applications that is compatible with SmartNIC (smart network interface card) hardware to contribute to the future of edge computing and Blockchain. The demand on network infrastructure is exploding, with there estimated to be 41.2 billion active network connections in 2025, 30.8 of those being IOT [13]. Of these active network IOT connections, a growing plurality will contain SmartNICs, which add valuable compute into the crucial intersection between the network and individual server. With it forecasted that three-fourths of all enterprise data will be processed at the edge, SmartNICs will be a giant player in that space [9]. On the other hand, the Blockchain industry is growing at an unprecedented rate, but is running to large problems in its application, specifically its reliance on GPUs and high compute cost- Bitcoin for example consumed 110 terawatt hours in 2021, .55% of the global energy production and roughly equivalent to the size of Malaysia or Sweden [2]. We saw overlap in edge processing requiring security, the SmartNICs offering compute power on the edge, and Blockchain’s need for an alternative compatible hardware that catered to its decentralized qualities.

In our solution, we contributed to both areas, as we made a software infrastructure that is fully run on a fourth generation SmartNIC. This infrastructure allows clients to maintain nodes, mine for currency, and send transactions to other nodes, and more- essentially the key tenets of the blockchain that are the underpinning of Blockchain applications such as smart contracts and cryptocurrency. Using C standard library, openssl, and nanomsg, we created an object oriented Blockchain infrastructure nicknamed “Lampcoin,” which we found to be stable on the SmartNIC as well as operable. The results are extremely encouraging because they point to an unique opportunity to offload a certain part of the larger blockchain work in the future to the SmartNIC, such as Bitcoin hashing work for mining or smart contract verification for Ethereum.
# Table of Contents

1 Introduction  
1.1 Brief Overview of Final Product Vision ......................................................... 1  
1.2 Motivation ............................................................................................................ 1  
1.3 Background and Current Innovations ................................................................. 2  
  1.3.1 SmartNICs ..................................................................................................... 2  
  1.3.2 Blockchain Technologies ............................................................................... 3  
1.4 Proposed Solution and Project Description ......................................................... 4  

2 Use Cases  
2.1 Stakeholders ........................................................................................................ 6  
2.2 Use Cases ............................................................................................................ 6  

3 Proposed Requirements  
3.1 Proposed System Designs .................................................................................... 8  
3.2 Software ................................................................................................................ 10  
3.3 Design Rationale ................................................................................................... 10  
3.4 Technologies Required ....................................................................................... 11  
3.5 Functional Requirements ................................................................................... 11  
3.6 Non-Functional Requirements .......................................................................... 11  
3.7 User and Technical Requirements .................................................................... 11  

4 Development Timeline and Risks  
4.1 Phases of Development ..................................................................................... 13  
4.2 Expected Timeline .............................................................................................. 14  
4.3 Possible Risks Analysis ...................................................................................... 14  
4.4 Problems and Obstacles ..................................................................................... 14  

5 Implementation and Results  
5.1 Implementation .................................................................................................. 16  
5.2 Evaluation and Performance Metrics Used ....................................................... 17  
5.3 Results .................................................................................................................. 17  
  5.3.1 Experiment 1 - Running Solely On SmartNIC Results ................................ 17  
  5.3.2 Experiment 2 - Offload Hashing a Block Results ........................................ 19  
5.4 Analysis ............................................................................................................... 20  
  5.4.1 Experiment 1 - Running Solely On SmartNIC Analysis ................................. 20  
  5.4.2 Experiment 2 - Offload Hashing a Block Analysis ......................................... 20  

6 Social Ethics  
6.1 Accessibility .......................................................................................................... 21  
6.2 Security ................................................................................................................ 21  
6.3 Manufacturability ................................................................................................ 21  
6.4 Environment ........................................................................................................ 21
# List of Figures

2.1 Use Case Diagram ................................................................. 7
3.1 Single Node Abstraction ....................................................... 8
3.2 Multi Node Abstraction ....................................................... 9
3.3 Blockchain Software Architecture ....................................... 10
4.1 Gantt Chart Expected Timeline ........................................... 14
5.1 Server vs. SmartNIC Performance Table 1 .............................. 18
5.2 Server vs. SmartNIC Performance Graph 1 ............................. 18
5.3 Server vs. SmartNIC Performance Table 2 .............................. 19
5.4 Server vs. SmartNIC Performance Graph 2 ............................. 19
8.1 The 3 Server Boxes that We Used Throughout Our Project- the 2 SmartNICs were hosted in them. 25
8.2 Server Box Used for Testing and SmartNIC Hosting .................. 26
8.3 The Nvidia Bluefield 2 DPU, the SmartNIC We Ran Our Experiments On 26
8.4 Basic Netronome SmartNIC .................................................. 27
Chapter 1

Introduction

1.1 Brief Overview of Final Product Vision

Goal is to implement this infrastructure as software on SmartNIC(Smart Network Interface Card), so our final product vision for this senior design project of how our software will run is broken into the following steps:

• Step 1: User logs into Linux server or laptop that includes SmartNIC

• Step 2: User runs a Python or Bash script that sets up node automatically with IP address and unique keys, potentially creating Blockchain if genesis node.

• Step 3: Script will setup API endpoints or user can use command line to interact with deployed Blockchain infrastructure (Methods like make_transaction, all_transactions, performance, etc.)

1.2 Motivation

Ever since the creation of Bitcoin in 2009 by the infamous Satoshi Nakomoto, the revolutionary concept of blockchain has taken the technology world by storm. Blockchain, a system of recording information in a way that makes it difficult or impossible to change, is the security concept behind Bitcoin and the now over 6,000 cryptocurrencies that exist today on the market. Cryptocurrency is only one application. Due to its extensive security applications, the Blockchain model applies across multiple markets, including the food industry, data governance, NFTs, digital voting, healthcare, and more.

As with all innovations, the widespread adoption of Blockchain, especially in cryptocurrency, has shown that it has severe negative impacts. The most problematic impact is the extreme computational demand. When new coins get mined in most cryptocurrencies, the miner has to show proof-of-work, which is extremely computationally expensive and thus expends a great deal of energy- even proof of stake systems and other proof systems create some computational impact. These consequences have created a race to accumulate the most computational power possible, leading to worldwide shortages of graphics cards and other valuable resources. The problem current Blockchain
technology poses is three-pronged: how do we balance security and fallibility with the rational usage of compute and prevent an arms race for computational power while keeping the Blockchain process de-centralized as well-intended?

With the current push for edge computing, SmartNICs are going to be key contributors to this area shifts to more emphasis on network infrastructure. There was not any academic literature on Blockchain and edge computing, so we were motivated to take up the challenge of being the first to enter the intersectional space.

1.3 Background and Current Innovations

1.3.1 SmartNICs

Network Interface Cards (NIC) are Layer-2 (data-link layer) hardware components that can help physical devices connect to wired and wireless networks. NICs can be easily found in most devices in today’s world like laptops, TVs, phones, etc. They are very fast at transmitting and receiving data frames as it is purely implemented in hardware rather than software going up the IP protocol stack. Because of the speed that NICs deliver, there have been recent developments in adding small CPUs and processing abilities to the NICs to offload some intensive processing done by servers in data centers. These advanced NICs with CPU processing power are called SmartNICs.

SmartNICs are starting to become popular requirements for data centers, cloud providers, software-defined networks (SDN), etc [8]. SmartNICs have the same basic capabilities as a regular NIC, except they also integrate other processing abilities. Certain categories of SmartNICs have different CPUs that have specific characteristics for specific types of applications. Here are the 3 main categories [10]:

- **ASIC (Application – Specific Integrated Circuit):** good price/performance, has some programmability and extensibility, and not great flexibility as it is specific for pre-defined tasks.

- **FPGA (Field Programmable Gate Array):** good performance, expensive, difficult to program, and specific for workload optimization.

- **SoC (System – on – Chip):** It is somewhere in between the previous 2. It is a combination of a NIC and CPU. It is good price/performance, very flexible, easiest to program with C/C++.

SmartNICs currently help offload servers’ computation and processing. One paper [7] created a framework called iPipe that offloads distributed applications onto SoC-based SmartNICs. The paper proves that it achieves host CPU and latency savings. One paper [14] aims to offload some of the CPU power needed to support Network Function Virtualization. Another paper [11] aims to offload some of the Functions that Network Function Virtualization contains. In industry, there is starting to be usage for it in offloading ML inference engines and software defined networking, with companies like Nvidia, AMD, and Marvell leading the way.
1.3.2 Blockchain Technologies

Blockchain has taken the world by storm recently in the last decade, with the proliferation of blockchain-based cryptocurrencies such as Bitcoin and Ethereum bringing light to the concept as well as its possible application throughout the technological world. At its core, most blockchains are public distributed ledgers in which transactions are recorded chronologically. These transactions, when recorded to the Blockchain, cannot be removed or altered, preventing hacking concerns. Each transaction has its own “block” in the Blockchain, which can contain various types of data (depending on the Blockchain application), but all contains four primary fields [1]:

- Hash of previous block: How chronology is stored by the blockchain, keeps blocks linked
- Data: Aggregate set of transactions included in block
- Nonce: The random value used to vary the output of the hashing value of the block
- Hash: Traditionally done through passing the previous hash, data, and nonce thru the SHA-256 algorithm, it is what distinguishes the block: the digital signature

These blocks cannot be added by anyone. Each Blockchain implementation requires some proof, or some reason to trust a network node when the node attempts to add a transaction to the ledger. The traditional way of doing so is proof of work. In proof of work, the network node attempting to add a transaction must “mine” for a nonce value that produces a hash value lower than the blockchain’s pre-defined condition. Whoever solves that extremely complex mathematical puzzle is rewarded with some reward, which in traditional cases is some type of coin. The “miner” is rewarded for their validation work, and the transaction is added to the decentralized blockchain ledger.

The design of Blockchain has some huge advantages, which in part is fueling its adoption across the world [3]:

- Security: Since transactions cannot be deleted or altered, this makes hacking or messing with the chain next to impossible. Coupled with the decentralization of the ledger in that every node has its own version of the ledger, it makes attempting to fake a transaction extremely hard as more than ½ of all nodes would need to corroborate the fake transaction for it to be legitimized. This is perhaps its strongest advantage.
- Regulation: Most systems based on trust require a central governing authority, such as the government or a corporation, to mediate and ensure that all parties continue to trust the system. The way Blockchain is designed, trust is governed by the consensus mechanism- the majority Blockchain ledger is the one that rules. This approach not only removes the middle man, but also scales with more users, increasing security with adoption.
- Transaction fees: No intermediaries, so no transaction fees. Rather than paying a bank or intermediary, these costs are now pocketed by the user.
• Transaction Time Speed: Unlike other forms of transactions which can take time to verify or to have the transaction show up, the transaction in Blockchain’s case is almost instantaneous or dependent on network speed, which is usually not a large impediment.

Mentioned earlier is the cryptocurrency application of Blockchain: that application is the one that brought the technology to the forefront, with Satoshi Nakamoto creating Bitcoin in 2009. Fast forward to now, multiple new applications for Blockchain have been created: smart contracts, personal identification, healthcare, and IOT. Each application is just starting to be explored, and with this exploration has come some areas of improvement for Blockchain, areas which it will need to improve on to be the algorithmic backbone of our technological future [6]:

• Environmental cost: The proof of work is very computationally expensive, which means it has a giant financial and economic impact. It’s claimed that the energy required to keep the blockchain Bitcoin running was as much as 159 world nations. It’s a staggering statistic that needs to be addressed moving forward and is being addressed currently with different validation mechanisms such as proof of identity or proof of stake, as well as constant modifications to the algorithmic underpinning of mainstream blockchains.

• Lack of regulation: Because there is no governing authority, there will always be a fairly risky environment for Blockchain, especially for new currencies. Scams and market manipulation are regular in the development of “alt-coins”, and there is always the risk of crypto-wallets being hacked or lost or certain applications being outlawed by governments.

• Awareness: The basic tenets of Blockchain rely on adoption, and although adoption has considerably ramped in the decade plus it has been out since 2009, it is still not enough for the technology to become mainstream. A great deal of the world’s population is not privy to Blockchain, and will be hesitant in any transition to such technologies because of that.

1.4 Proposed Solution and Project Description

Our project aims to solve the issue of by approaching Blockchain from an entirely different perspective. As mentioned earlier in Section 1.2 on Page 1, in most blockchain implementations, a miner needs a plethora of Graphical Processing Units (GPUs). GPUs are preferred in blockchain implementations compared to CPUs since they have more processing power. Our project will experiment with SmartNICs to offload GPU processing on a distributed network of SmartNICs. A regular (network interface controller) NIC allows devices to connect over a computer network such as the Internet. SmartNICs, on the other hand, have a mini CPU that can process data compared to a regular NIC. SmartNICs are usually integrated at the network edge and help allow servers, data centers, GPUs, etc., to offload some of that data processing. For SmartNICs, some data processing gets offloaded, permitting components of the blockchain network
like GPUs to use less energy. Additionally, an NIC is already integrated into the machines, or edge nodes, that make up the Blockchain network. A GPU has to be physically added to these edge nodes which adds more hardware complexity and unnecessary costs.

Our project is research-heavy and experimental, as there are many unknowns to be discovered. Our objective is to create a robust and fallible software infrastructure for implementing the new Blockchain network on the SmartNICs. Using SmartNICs from Nvidia, we aim to experiment with setting up multiple servers that use SmartNICs, instead of using GPUs, to compute the hashing algorithm and compete with SmartNICs in the other servers. We will experiment with offloading hashing and encryption/decryption algorithms to design the ideal infrastructure for future blockchain applications on the SmartNIC device.

The materials needed for current Blockchain mining as well as the energy are massive, and in this undertaking, those same materials are prevented from being utilized by underserved populations. Additionally, the barriers to entry in terms of hardware necessity to participate in the blockchain movement are skyrocketing in terms of cost and complexity. By producing a Blockchain infrastructure on SmartNICs that balances security and fallibility with the rational usage of energy resources, we aim to create a systematic and sustainable software architecture capable of conserving energy resources. For this reason, we strongly believe our project can directly affect the current Blockchain landscape and positively impact several communities.
Chapter 2

Use Cases

2.1 Stakeholders

- Developers:
  - Our Team

- Customer
  - Entities aiming to make Blockchain applications:
    - Corporations
    - Independent developers
    - Government

- Users
  - Application Developers:
    - Frontend
    - Backend
    - etc.
  - Cloud Providers (eg. Azure, AWS, GCP)

2.2 Use Cases

The first use case describes an Individual Miner who wants to install our project on their laptop that has a SmartNIC. The second use case is a Cloud Developer who wants to install our project in a native cloud environment (Refer to 2.1).
Figure 2.1: Use Case Diagram
Chapter 3

Proposed Requirements

3.1 Proposed System Designs

Figure 3.1 displays our single node abstraction. We run the full blockchain technology within the SmartNIC that is stored within the server, and it is given commands by the user through terminal.

Figure 3.2 displays our multi node abstraction. Our intent is to be able to setup a distributed network of servers and SmartNICs to run our Blockchain C program. First, we will try to make the Blockchain run as a stable build by itself, communicating with other SmartNICs in an effort to maintain and mine a SmartNIC-only coin. If enough time, stretch
Figure 3.2: Multi Node Abstraction
goal is to leave some Blockchain tasks on CPU and offload certain others to SmartNIC, to use best of both worlds.

3.2 Software

Figure 3.3: Blockchain Software Architecture

Figure 3.3 displays our high-level software architecture for our Blockchain C program. In total, we have more than 4755 lines of code. Blockchain implementation is not usually done in C because of C’s lack of OOP architecture. For the purposes of our project with SmartNICs, however, we needed to implement it in C programming in order to be more universally runnable on a variety of SmartNICs with some being only C programmable, and this being a recent breakthrough at that with SmartNICs mostly being flexibly programmed in Verilog, VHDL, and P4.

3.3 Design Rationale

The original designs for Blockchain include energy and cost expensive hardware such as GPUs. Our design uses existing cloud architectures and implements Blockchain on them. We will be using servers that have SmartNICs
already integrated in them instead of buying additional GPUs which complicates the existing hardware requirements. (Note: For the purposes of this project, we are going to setup our own test environment purchasing servers and SmartNICs). Cloud platforms like Microsoft Azure already make use of SmartNICs in their servers and the papers currently being published (for example, this paper [4] discusses a new Azure Accelerated Network on top of the Microsoft Azure architecture) are about new implementations on top of this existing network. Our design is less energy and cost expensive and does not introduce unnecessary hardware complexity.

3.4 Technologies Required

The technologies that we need are:

- 3 SmartNICs
- 3 Servers
- 1 Layer-3 switch (eg. router)

3.5 Functional Requirements

- SmartNICs will be able to compute the hashing algorithms
- SmartNICs will be able to manage the ledgers
- SmartNICs will be able to compete with other algorithms
- Each server will have 1 ledger and 1 SmartNIC

3.6 Non-Functional Requirements

- Performance should not be effected too much
- Energy cost and Hardware Complexity should be as minimal as possible
- Should be easily deployable for an Infrastructure-as-Code service

3.7 User and Technical Requirements

- User:
  - SmartNICs that support our type of software we run on them
  - Odd Number of Nodes
- Basic Understanding of how Blockchain works (Ideally a miner)

**Technical**

- Hoping for Only C Programmable SmartNICs
- Odd number of SmartNICs requirement for Blockchain (Industry Standard is minimum 3)
- Linux based Servers
Chapter 4

Development Timeline and Risks

4.1 Phases of Development

1. Phase 1:
   - **Time**: 6 - 8 weeks
   - **Goal**: Understand Blockchain mathematical concepts, what is a SmartNIC.

2. Phase 2:
   - **Time**: 1 - 2 weeks
   - **Goal**: Run Basic Blockchain Algorithm on all our Laptops and Different Machines, Understand How to Write A Blockchain Program

3. Phase 3:
   - **Time**: 5 - 7 weeks
   - **Goal**: Agree on project design constraints, dependencies, object oriented design, and intended implementation.

4. Phase 4:
   - **Time**: 10 - 12 weeks
   - **Goal**: Writing and implementing Blockchain infrastructure code in C on CPU compute.

5. Phase 5:
   - **Time**: 6-8 weeks
   - **Goal**: Write script, test system, reiterate, and ensure program stability on SmartNIC.
4.2 Expected Timeline

Refer to 4.1

4.3 Possible Risks Analysis

1. Our hardware might not come till later in the quarter and our implementation might be delayed

2. Performance Issues: SmartNIC might not be powerful enough as we initially thought and we would have to purchase additional hardware for it to work

3. We might need to start over to debug where our issues might occur

4. Worst Case: the project might be too difficult and we would need to change it

5. Project Scope: Basic goal is to have two SmartNICs “compete” for coin, expansion on scope will happen after used case is done

6. Roadblocks in Software Development: Current plan offers time to deal w/ setbacks

7. Time to Address UX UI: Lowest priority - will reduce emphasis on UX UI if other parts of project require more time

4.4 Problems and Obstacles

A main problem we had was getting the necessary hardware. Because of the chip shortage, we got SmartNICs 1 week before the Senior Design Presentation which was not enough time to generate meaningful results to present.

We also faced problems in learning how to develop software on SmartNICs as the software stack of SmartNICs are still being developed. The hardware exists, but the software is still being invented. We are in an area of research that
really does not have proper documentation as well since everything is still being figured out. It was a very experimental project.

Our Blockchain implementation in C also faced problems. Most Blockchain implementations are done in higher level languages to make use of OOP and other helpful libraries. We ran into issues like figuring out in C programming how to serialize a large structure to input into our hashing functions. We had to design our own data structures and deal with difficult issues that a higher language could solve compared to C programming. But given we are using SmartNICs, we had to code in a lower level language like C as some SmartNICs are only C programmable.
Chapter 5

Implementation and Results

5.1 Implementation

For the implementation of this project, we developed it in C. The reason we did so is because SmartNICs cannot be coded at any higher level of abstraction that would be sufficient for an infrastructure. Since SmartNICs are a fairly new hardware, they have had traditionally any IP coded in them on VHDL or Verilog, with P4 becoming a new entrant. In the fourth generation of SmartNICs, which is the most recent wave, C programmability has been added, albeit with little flexibility as of now. Thus, we only used C standard library functions, the BlueField 2 optimized OpenSSL library for secure network communications, and the nanomsg library for secure and scalable socket communication patterns that underlie our Blockchain design. Since we are in C and using only C-stdlib, this means that we had to build our own data structures, in this case the dictionary and linked list data structures. These data structures would be used as the backbone of the structures that formulate our Blockchain design. For the Blockchain, we have four main structures at a high level, all that are a part of the other:

- **Transaction**: Contains the sender, recipient, time of transaction, signature. Smallest object in the OOP design of the Blockchain.

- **Block**: Contains the previous hash, time of block creation, proof, and the transaction list of the transactions stored in the block. The Blockchain securitizes the transactions within it, so the block of the Blockchain is containing the valuable transaction history, in our case 100 of them.

- **ChainConnect**: The connector of the Blockchain; contains a block and then the reference to the next chain connect.

- **Blockchain**: The structure that is the metaphorical “chain.” Contains length of the Blockchain, number of nodes, number of total transactions stored, the last block, the last hash, the last transaction hash, the last proof of work, first connect of the chain, total amount, and a total transaction list. Has a ledger and verified node dictionary as well.
For the Blockchain to be operable, there have to be nodes and clients, so next we created the design of those.

- **Node**: Actual mining node that participates in the Blockchain, has to be running the whole time. Has IP address as a node relates to one running computer node in the network. Other than mining- its chief Blockchain function, it communicates with other nodes for consensus protocol and registers new nodes.

- **Client**: User of the Blockchain that can get information about the Blockchain contents and send transactions to other users with the chain currency.

So to use the Blockchain, one has to open the command line and compile the node using a script. After the node is running, the user has to open a separate terminal in order to add to the Blockchain, as the client. If the node and client were the same program, then the Blockchain would not work, as the node always has to be running for the Blockchain to work- the Blockchain is non-existent if there are no nodes. Clients can log in any time to post transactions, but the mining and maintenance of Lampcoin is required 24/7, thus the design choice. Our intention was to make the implementation as straightforward as possible.

### 5.2 Evaluation and Performance Metrics Used

We first created various test files and eventually an automated python test script to verify that our Blockchain implementation in C is robust. We tried different inputs and verified our code gave the appropriate response. This is for testing our implementation only for correctness.

We also evaluated the performance of our project by running our implementation on SmartNICs and Servers that were running Linux Ubuntu. We have 3 PowerEdge Servers from Dell Technologies and we have 2 Nvidia DPU BlueField-2 SmartNICs. We then installed Linux Ubuntu on them and necessary packages to get our code running. With these hardware pieces setup we were able to generate results.

The performance metrics we chose to focus on were CPU usage in terms of user % and time in seconds to do a particular part of the Blockchain. We used the sar tool to gather the CPU usage metrics and we used the Linux time command to get the time of running a part of our code.

### 5.3 Results

#### 5.3.1 Experiment 1 - Running Solely On SmartNIC Results

5.1 and 5.2 compares the CPU performance and time it takes to run the mining part of Blockchain separately on the server and SmartNIC. We are taking a closer look into the average amount of CPU taken by our user program (the mining part part of Blockchain) and found that, predictably, the mining used the CPU more on the SmartNIC than the Server. Partly because the CPU inside the SmartNIC is much smaller compared to the server and at the same time, the
Figure 5.1: Server vs. SmartNIC Performance Table 1

<table>
<thead>
<tr>
<th></th>
<th>Time (s)</th>
<th>Average CPU user %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server</td>
<td>20.541</td>
<td>6.3</td>
</tr>
<tr>
<td>SmartNIC</td>
<td>68.697</td>
<td>12.42</td>
</tr>
</tbody>
</table>

Figure 5.2: Server vs. SmartNIC Performance Graph 1
server has more CPU cores. We also found out the time it takes to mine just 3 blocks on the SmartNIC took way more time than the server (keeping in mind that in the real world, the mining is done on GPUs).

### 5.3.2 Experiment 2 - Offload Hashing a Block Results

<table>
<thead>
<tr>
<th></th>
<th>Time (s)</th>
<th>Average CPU user %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server</td>
<td>10.053</td>
<td>0.15</td>
</tr>
<tr>
<td>SmartNIC</td>
<td>10.437</td>
<td>0.29</td>
</tr>
</tbody>
</table>

Figure 5.3: Server vs. SmartNIC Performance Table 2

In our Experiment 2, we tried to just focusing on offloading the hashing part onto the SmartNIC. 5.3 and 5.4 compares the performance of running the hashing part of Blockchain between the server and SmartNIC using a similar approach to gather these metrics as Experiment 1. We can see that the hashing part of Blockchain has very simi-
lar performance between the server and the SmartNIC, which is a good indicator. It means that there is not much performance we are losing when we move the hashing part to the SmartNIC.

5.4 Analysis

5.4.1 Experiment 1 - Running Solely On SmartNIC Analysis

We found out it is possible to run the mining part solely on a SmartNIC, but it is not ideal as 5.2 shows. Our Blockchain implementation is a very lightweight version of Proof-of-Work and we used a server as the benchmark for running it. In the real world, Blockchain is ran on GPUs which has more performance than servers and definitely more than SmartNICs. So, if we were to run the mining part of the Blockchain solely on the SmartNIC, that is not going to work in the real world. The other thing to point out is we are using high performance SmartNICs from Nvidia that are technically DPUs (Data Processing Units), see section 5.2. We would expect if we were running our "Lampcoin" on regular SmartNICs, that probably will take even longer to run. The main component of mining that consumes the most energy and performance is the Proof-of-Work. The last thing to point out is that the SmartNIC does not have enough memory to store and maintain a huge chain of blocks that keeps growing. So from this first test, we know that offloading the entire mining part of Blockchain on SmartNICs is a bad choice.

5.4.2 Experiment 2 - Offload Hashing a Block Analysis

As we see from this experiment, the hashing part did not incur too much of a performance cost when moving it to the SmartNIC. These gathered metrics probably have a little bit of noise as well, but they are very similar between the server and SmartNIC. If we kept this implementation and system design, we are saving about 10 seconds from the server. The CPU usage did not change much probably because the Proof-of-Work and managing the ledger is more CPU usage heavy. But, as we see from Experiment 1, we cannot just offload the Proof-of-Work or ledger management since the average SmartNIC might not have the capacity to handle that much pressure. But, we did help the performance of the overall server (in the real world case, the GPU) offload some parts of the Blockchain in order to perform better and faster. It can also lead to leveraging parallelism to help future the performance of Blockchain. (7.2).
Chapter 6

Social Ethics

6.1 Accessibility

Blockchain currently requires a great deal of money to participate in. Most implementations of the technology require graphics cards, which are in massive shortage and cost thousands of dollars. We want the future of Blockchain to be on hardware that is cheaper and more accessible. With SmartNICs, we can make Blockchain technology more accessible to lower income population as the overall cost of the technology will decrease.

6.2 Security

By 2025, there will be an estimated 30.9 billion IOT connections [5]. That’s alot of security vulnerabilities. Future security attacks will be harder to trace and will need to be detected and eliminated faster than ever, so we will need security architecture on the edge. With the benefits of Blockchain, we believe these security vulnerabilities will be largely mitigated, and the SmartNIC’s proximity to the edge make it an ideal solution. SmartNICs also preserve the idea of distributed computing since they were made specifically to work on the edge in a distributed network which is also crucial to maintaining the security aspect of Blockchain technology.

6.3 Manufacturability

This plays a little in line into the accessibility issue, but SmartNICs cost cheaper to manufacture than GPUs, and are already placed in most hyperscalers’ and cloud providers’ data center architecture. Especially in the current chip shortage, creating hardware that requires less chips can be produced faster and in greater amounts.

6.4 Environment

Blockchain, in its current design, is a very energy consuming process. Currently Bitcoin specifically consumes 122.87 terawatt hours - more than what Netherlands and Argentina consume [12]. With our implementation, we hope to
reduce the actual computational strain as well. We hope to use the networking aspects of the SmartNIC to change the proof of work concept behind Blockchain applications, and thus energy demand.
Chapter 7

Conclusion

7.1 Summary

The potential for creativity and architectural innovation in the edge processing sphere is massive, and in our senior design, we explored a major player in the space: the smartNIC, or the smart network interface card. Specifically, we apply the SmartNIC’s computational capabilities and traits to the growing area of Blockchain. Blockchain is the underlying backbone of numerous applications such as Bitcoin, Ethereum, medical records, and more—its zero trust security approach and consensus algorithm concept prevent malicious attacks on the network stored on its foundation extremely effectively. We saw an overlap in the demand created by network users for security privacy and also the growth of IOT devices as an ideal intersection with Blockchain’s current main issues: high energy expenditure and reliance on graphic cards. Thus, we created the first software implementation of a SmartNIC-compatible infrastructure, one that can be run entirely on SmartNICs. Our team developed Lampcoin, implemented in the C standard library with the help of the networking nanomsg and cryptographic OpenSSL libraries. After many trials and tribulations in obtaining the SmartNIC hardware, we received the hardware at the tail end of Spring Quarter and began testing our code to develop a stable build. We were not only able to deploy a successful build on the Nvidia Bluefield 2 DPU, but we also were able to test the offloading of just hashing onto the SmartNIC, and the results when running the whole Blockchain infrastructure on the card and only a portion of it were both extremely encouraging. The results showed that SmartNICs were capable of running lightweight Blockchain, which lend to the conclusion that they can act as light nodes in certain Blockchain applications, an interesting way of increasing nodes in a potential secure Blockchain network. Additionally, the results suggested that offloading hashing to SmartNIC is not a large cost, and if anything the SmartNIC is more than able to handle offloading duties in large-scale Blockchain networks in the future. Last, but not least, it opens the door up to more parallelism opportunities, an extremely important part of the networking infrastructure’s approach to bandwidth demand.

7.2 Next Steps + Future

For next steps, we aim to continue exploring the application of Blockchain on SmartNICs, expanding our hardware base from outside Nvidia, and expanding our tests to non Proof of Work implementations such as Proof of Stake. We aim to continue reiterating on our infrastructure, as we hope to make it lightweight and secure, but also hugely applicable in the Blockchain space and edge processing space. Overall, we find the security application of the SmartNIC extremely fascinating, and we are working towards publishing academic work on this topic as well as the endeavours that arise from it.
Chapter 8

Appendix

8.1 Photos
Figure 8.1: The 3 Server Boxes that We Used Throughout Our Project- the 2 SmartNICs were hosted in them.
Figure 8.2: Server Box Used for Testing and SmartNIC Hosting

Figure 8.3: The Nvidia Bluefield 2 DPU, the SmartNIC We Ran Our Experiments On
Figure 8.4: Basic Netronome SmartNIC
Bibliography


