 CMC, vol.57, no.3, pp.353-363, 2018

Secure Video Streaming with Lightweight Cipher PRESENT in an SDN Testbed

Pengcheng Liu[1, †], Xiaojun Wang[1, †], S. R. Chaudhry[1], Khalid Javeed[2], Yue Ma[3, *] and Martin Collier[1]

Abstract: The combination of traditional processors and Field Programmable Gate Arrays (FPGA) is shaping the future networking platform for intensive computation in resource-constrained networks and devices. These networks present two key challenges of security and resource limitations. Lightweight ciphers are suitable to provide data security in such constrained environments. Implementing the lightweight PRESENT encryption algorithm in a reconfigurable platform (FPGAs) can offer secure communication service and flexibility. This paper presents hardware acceleration of security primitives in SDN using NETFPGA-10G. We implement an efficient design of the PRESENT algorithm for faster, smaller and lower power consumption hardware circuit using Verilog. We evaluate the performance of the hardware and software implementations of PRESENT. Experimental results prove that the proposed hardware design is a viable option for use in resource constrained devices in future networks and their applications.

Keywords: Lightweight cipher, netFPGA, openFlow, RESENT encryption.

1 Introduction

Over the past decade, multimedia has been driving an increasing amount of the traffic over networks. By 2021, it is expected that 78 percent mobile traffic will be video data [Cisco (2016)]. As mobile networks are migrating to 5G, the method for video transmission over the networks is evolving, especially in terms of encryption. This evolution includes load balancers, Network Address Translators, QoE and Value-Add Services (VAS) such as content/URL filtering and video compression. Securing all these network devices in the cloud computing environment can be enhanced by incorporating the security functions. Here, the context of security refers to the data encryption at data flow-level in the cloud. In other words, if the security feature is enabled by a software control node for a specific flow, the packets belonging to that flow will be encrypted

[1] School of Electrical and Computer Engineering, Dublin City University, Dublin, Ireland.

[2] Department of Computer Engineering, Bahria University, Islamabad, Pakistan.

[3] School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, 100876, China.

[†] These authors contributed equally to this work.

[*] Corresponding Author: Yue Ma. Email: mayue@bupt.edu.cn.

before departing the network device and will safely pass through the public/private networks to reach end-users. The private encryption key is stored in a flow table created inside the network device. The encryption algorithm uses the associated per-flow key to encrypt packets, the same key is utilized to decrypt the packets and restore the original data.

This paper implements the light-weight PRESENT algorithm on NetFPGA-10G platform in our SDN testbed. The prime motivation to design, implement and evaluate the proposed security enabled network function in the proposed environment using PRESENT are:

- PRESENT is an ultra-light encryption algorithm, designed for efficient hardware implementation. It is highly suitable for power and hardware resource constrained networks.
- PRESENT is an ISO/IEC standard block cipher recommended for lightweight cryptography [Bogdanov, Knudsen, Leander et al. (2007)].
- The 64 bits block size exactly matches the OpenFlow (OF) data-path structural requirements. This feature relieves us from bus width conversion.
- Our target applications will require moderate security levels. Consequently, the 80-bit security key of PRESENT is considered adequate for this study.

We assume that not all the data-flows require encryption service. Therefore, the required per-port throughput of the security engine is less than 10 Gbps. In our work, we design and implement an encryption enabled FPGA accelerator in a 10 G OF-switch [NetFPGA-10G (2009)] using the PRESENT algorithm and investigate its performance in a SDN testbed.

The remainder of this paper is organized as follows. Section II presents the related works and motivations. Then, the detailed implementation is presented in Section III. Section IV describes the experimental setup with detailed analysis of performance outcomes. Finally, Section V states the conclusions of the paper and directions for future work.

2 Related works and motivations

The increase in mobile access to data services places a tremendous load on networks. Traditionally, a majority of mobile data traffic is HTTP over TCP. However, many services are in the process of transitioning to HTTPS [Bruneau, Lacaud, Negru et al. (2018)]. The reasons for moving services to HTTPS are ensuring security (i.e. media request cannot be intercepted), privacy (viewing habits cannot be inferred by inspecting traffic) and popularity (Google page rank favoring sites delivered under HTTPS [Tepsic (2018)]. The growing use of encrypted protocols is causing fundamental shifts, which results in added network burden for security functions in real-time applications.

2.1 Lightweight cryptography

Lightweight cryptography requires cryptographic algorithm that is tailored for low-resource devices and must address three challenges: Minimal overhead (silicon area or memory footprint), low-power consumption, and adequate security level [Biryukov and

Perrin (2017)]. Lightweight cryptographic algorithms are often implemented in Application Specific Integrated Circuits (ASICs). FPGAs are low implementation cost as well as a faster design cycle.

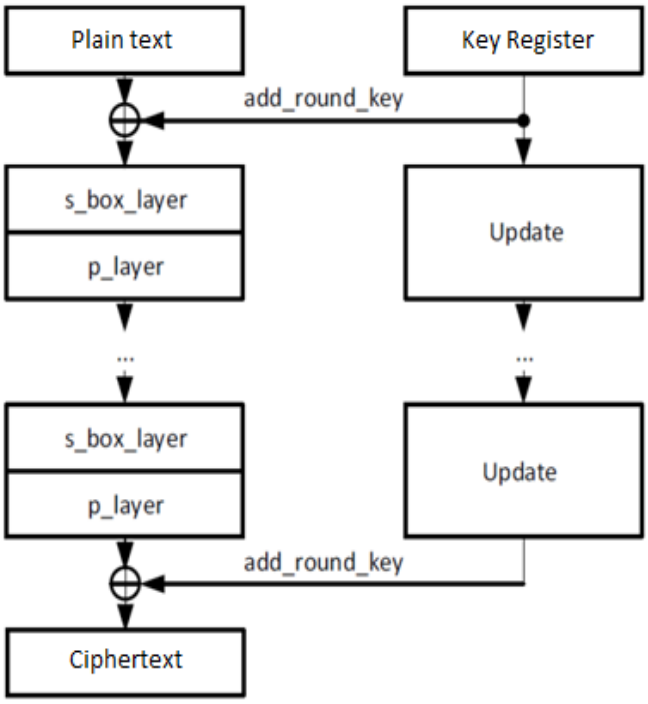


Figure 1: High-level PRESENT encryption [Bogdanov, Knudsen, Leander et al. (2007)]

A plethora of lightweight symmetric cryptography algorithms have been designed [Biryukov and Perrin (2017)]. However, only a few have been used in practice including PRESENT which is standardized by the International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC) [ISOIEC (1987)]. In this paper, we integrate the PRESENT algorithm in a SDN architecture to enhance data security. Block cipher operating in stream mode is stream cipher cores such as eStream [eSTREAM (2016)]. A low-cost implementation of AES [AES (2001)] for RFID application is proposed in [Feldhofer, Dominikus and Wolkerstorfer (2004)]. However, the high-volume resources required for this cipher are around 3600 GE (Gate Equivalence). GE is the well-known metric, according to which the lightweight algorithms are evaluated in terms of chip area usage. It is equal to the chip area consumed by the algorithm normalized to the area of a 2-input NAND gate in a given standard cell library. mCrypton [Lim and Korkishko (2006)], HIGHT [Hong, Sung, Hong et al. (2006)], SEA [Standaert, Piret, Gershenfeld et al. (2006)] and PRESENT [Bogdanov, Knudsen, Leander et al. (2007)] algorithms have been proposed. The resource assessments show that these four algorithms require 2949 GE, 3000 GE, 2280 GE and 1570 GE respectively. The low GE of PRESENT motivates this work to study its

performance in a SDN environment.

2.2 The PRESENT algorithm

In this paper, we implement the PRESENT lightweight block cipher as our encryption core. The algorithm encrypts a 64-bit block with an 80-bit key in 31 rounds. Each of the 31 rounds consists of a linear bitwise permutation and a non-linear substitution layer as well as a XOR operation to introduce a round key K_i for $1 \le i \le 31$. Fig. 1 shows a top level functional flow and layout of PRESENT algorithm.

A substitution box layer (S-Box) is implemented in this algorithm. S-Box takes an input of 4-bits and replaces it with another 4-bit value as the output. Afterwards, a permutation layer (P-Layer) is designed to jumble the bits. The key scheduler is a critical part of the PRESENT algorithm, which is used for updating the key with every iteration during the different states of the algorithm. The PRESENT algorithm is an iterative process, this means that the data will be passed through the S-box, P-layer and add round key process multiple times before the cipher-data is produced at the end.

There are several research works on cryptanalysis of the PRESENT algorithm [Cho (2010); Liu, Jin and Kong (2016); Wang (2008)]. Multiple attack approaches are used to investigate how secure the PRESENT algorithm is. These attacks include, linear cryptanalysis of reduced round PRESENT [Cho (2010)], differential cryptanalysis of reduced round PRESENT and key recovery attack for PRESENT using slender-set linear cryptanalysis [Liu, Jin and Kong (2016)]. Due to the high scale of time taken to crack the PRESENT encryption, the attacks presented in these works would be impractical to apply in reality in an attempt to crack the algorithm and get any useful information secured through the algorithm.

2.2 NetFPGA: A programmable hardware platform

NetFPGA is an open programmable platform with wide recognition by both industries and academic institutions around the world. The NetFPGA-10G board was designed by the NetFPGA team and is available at HTG [NetFPGA (2007)]. The main features of the board are four 10GigE SFP+ interfaces, a 8 channels PCI Express interface, one Xilinx Virtex-5 TX240T FPGA and up to 27 MBytes QDRII SRAM and 288 MBytes RLDRAM-II.

Anwer et al. [Anwer and Feamster (2009)] designed a virtual router based on a NetFPGA-1G board [NetFPGA-1G (2007)] to support up to 8 identical virtual data plane instances. Each data-plane has its exclusive forwarding logic with an independent forwarding table at the hardware level. This design achieves ideal isolation and provides throughput twice as much as a legacy software router. However, it uses only on-chip memories and fixed resources for each virtual data path, constraining the scalability and flexibility of the entire system to a large extent. D. Unnikrishnam et al. [Unnikrishnan, Vadlamani, Liao et al. (2010)] built a scalable virtual data-plane. This design uses both a hardware data-plane on a FPGA and a software data-plane on a commodity server. It resolves the issue of scalability to some extent. Besides, the design uses reconfiguration technology to support the dynamic migration of data-planes between hardware and

software without interfering with regular traffic. Thus, it achieves a limited level of flexibility.

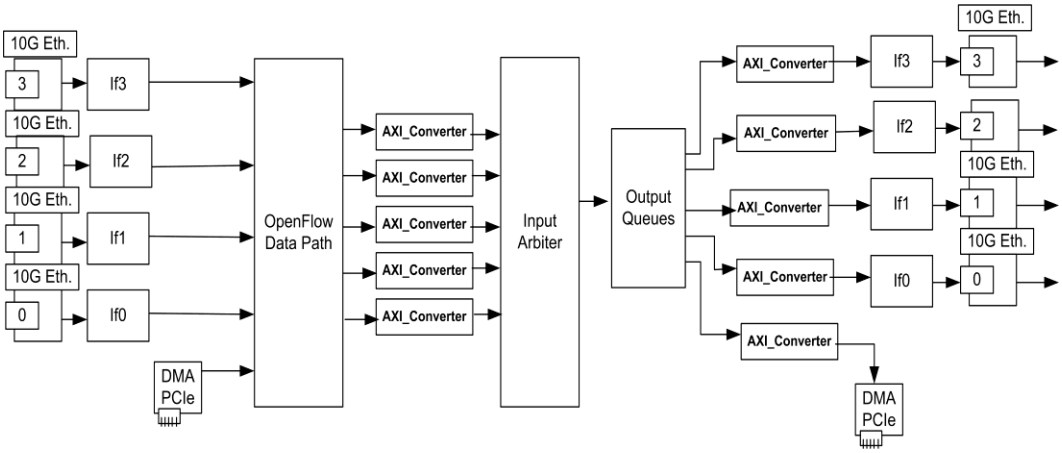


Figure 2: Internal hardware data path using OpenFlow switch

The OpenFlow project of NetFPGA platform [NetFPGA-1G (2007)] is chosen as the footstone of our work, and its top-level architecture is demonstrated in Fig. 2. Packets from each external interface and packets from host goes through dedicated pipelines. Other substantial components of the switch are OF Data Path, Input Arbiter and Output Queues.

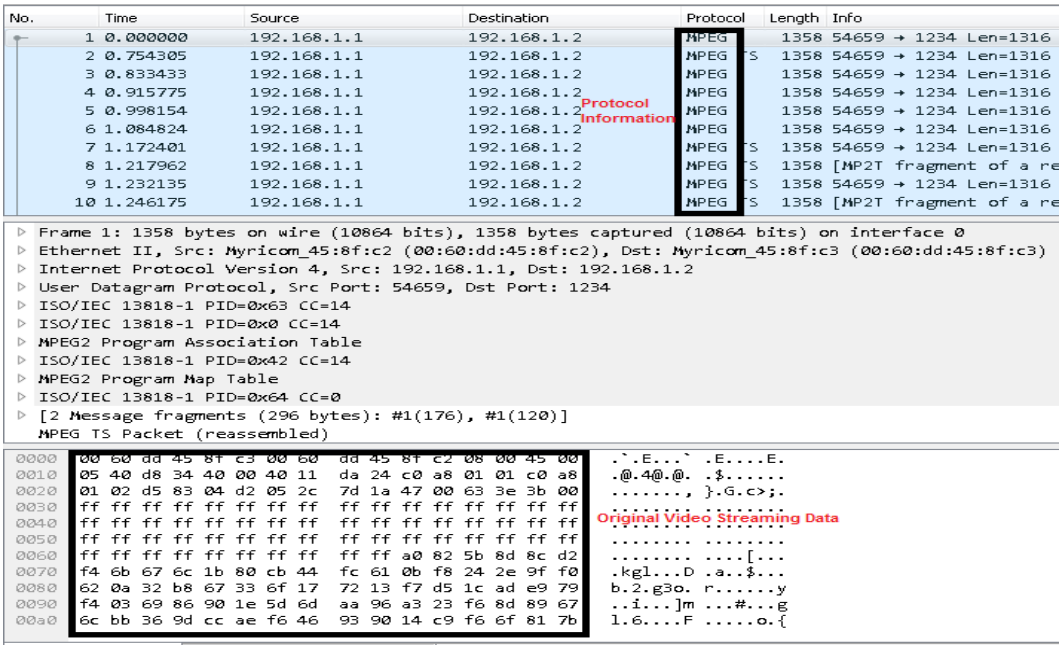
3 System implementation

For the hardware implementation, the PRESENT encryption and decryption algorithms were programmed in Verilog using the Xilinx Vivado software. The programmable logic area in NetFPGA-10G is used for hardware implementation of the PRESENT encryption and decryption. The security function consists of encryption and decryption of packets. The former encrypts packets inside the switch, after all other actions have been performed on the packet header and before sending the packet to the output queues. The latter, decrypts incoming encrypted packet before performing other functionalities. Both encryption and decryption functions are implemented inside OF-data path module.

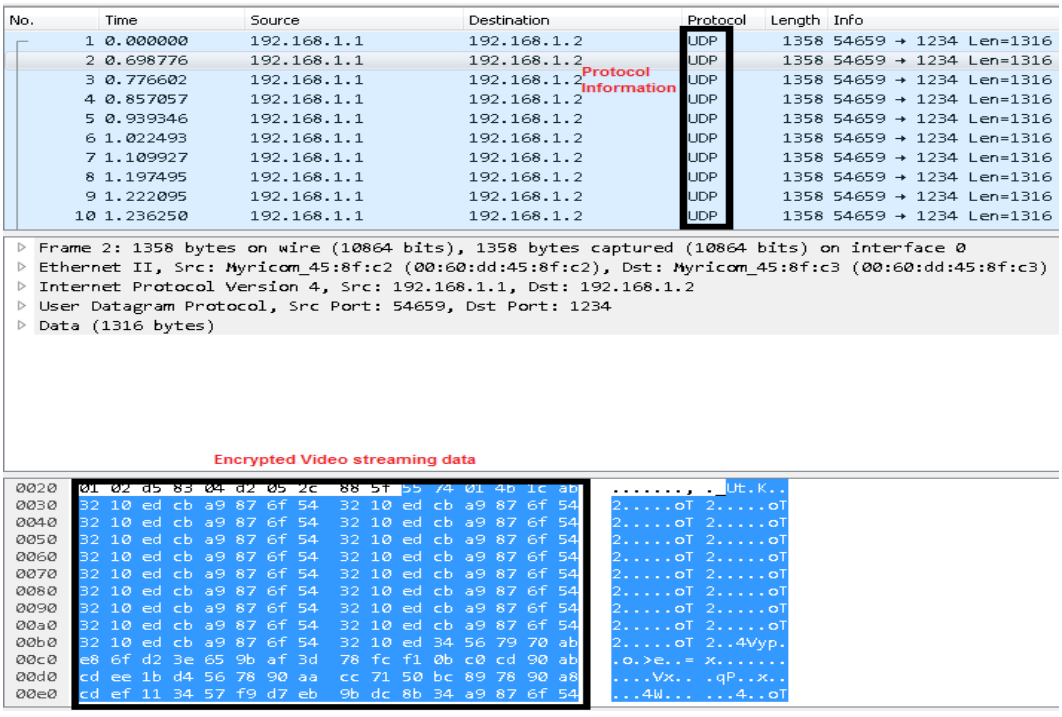
To incorporate the security function in the OF-Switch architecture, we have modified the following components:

- Flow tables are extended to provide per-flow keys. Encryption and decryption use the symmetrical security key,
- A per-flow security enable flag is introduced. This flag is used by a software controller to enable or disable the security function for any specific flow as per user/application requirement,
- Security Action is defined and is implemented using the PRESENT algorithm.

The resource utilization of the PRESENT implementation in NetFPGA-10G OF-switch is presented in Tab. 1.



(a): Unencrypted video streams



(b): Encrypted video data stream

Figure 3: Functional verification

Table 1: NetFPGA-10G implementation: Resource usage

Resource	Utilization	Utilization (%)
LUT	158	0.30
FF	213	0.20
IO	129	65.50
BUFG	1	1.07

For data traffic tracing, we deploy the Wireshark network analyzer at both the client and the server sites. This allows us to monitor and compare the data and network latency. Fig. 3 presents the data captured using the Wireshark packet analyzer with and without enabling encryption functionality respectively. It is observed that without encryption, the protocol type MPEG for video streaming is clearly visible in the captured data packets as shown in Fig. 3(a), whereas the encrypted data packets declare only UDP protocol in Fig. 3(b).

4 Performance evaluation and analysis

In order to evaluate QoE for the security mechanism in SDN architecture, dedicated security functions are desired. In this section, we first describe the experimental configurations and afterwards, we investigate and compare the performance of hardware and software implementations of the PRESENT algorithm.

4.1 Performance evaluation: PRESENT enabled NetFPGA-10G

To evaluate the performance of the hardware enabled security, we consider a Cloud Multimedia Secure Delivery setup. For investigation, we configure the multimedia setup using video streaming services. Video contents of various qualities are stored on a streaming server. We integrate the NetFPGA-10G as hardware accelerator near the video streaming server. We configure the video traffic to route through the NetFPGA-10G platform.

The aim of this testbed is to evaluate the performance of streaming video of various video quality standards with/without security feature. In the video streaming server, a VLC program [VideoLan (2015)] is configured to stream the video traffic via NetFPGA-10G to the client. The client is a typical user computer with an Ubuntu 14 operating system. All measurements are conducted using the video streaming platform VLC and Wireshark. Fig. 4 shows the measured video streaming latency with/without encryption in our SDN Testbed, for streaming videos of various video quality standards. The x-axis indicates various video quality standards, for the two types of each video sent directly (unencrypted) and securely (encrypted). The y-axis shows the latency, represented using bar levels for both types of transmission. The measured transmission latency of encrypted video streaming show approximately 8 to 10 percentiles increase compared with the latency of streaming the corresponding unencrypted videos. The slight increase in latency is not perceived in the decrypted video quality at the receiver end. This latency increase is so small that interacting services such as video conferencing would be unaffected.

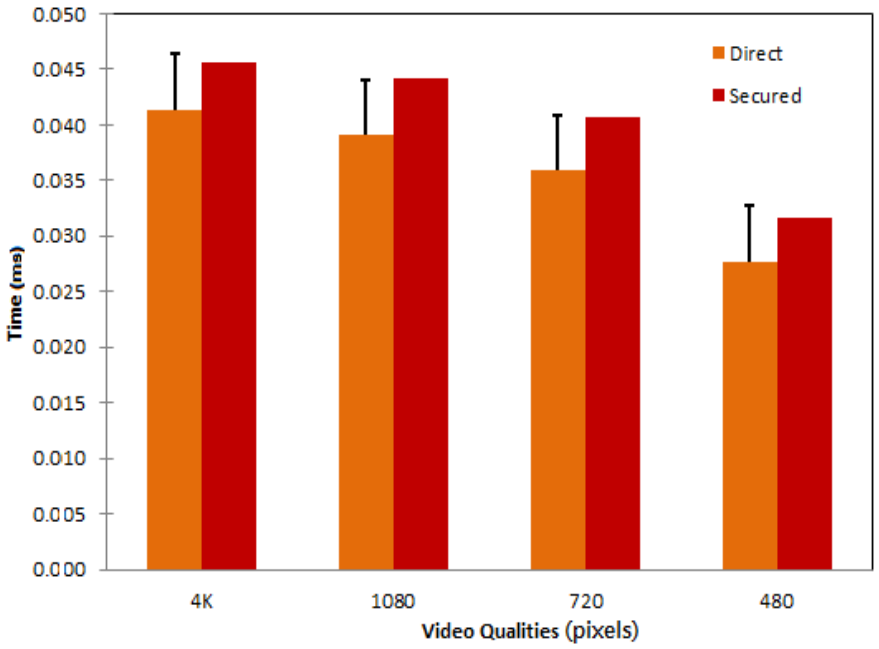


Figure 4: Video streaming latency in SDN testbed

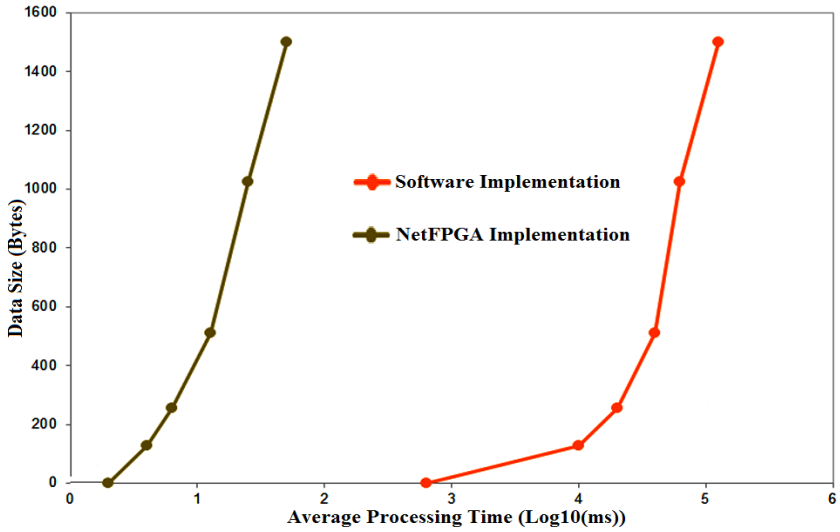
4.2 *Comparison of hardware and software implementation of PRESENT*

The PRESENT cipher is implemented in both software and hardware and is tested on two separate devices. A standard laptop for software implementation and a Xilinx NetFPGA-10G for hardware implementation are setup for this comparative study. The results are shown in Tab. 2. The software-based implementation was written using C and runs on a desktop server. The hardware implementation is deployed using Verilog on NetFPGA-10G.

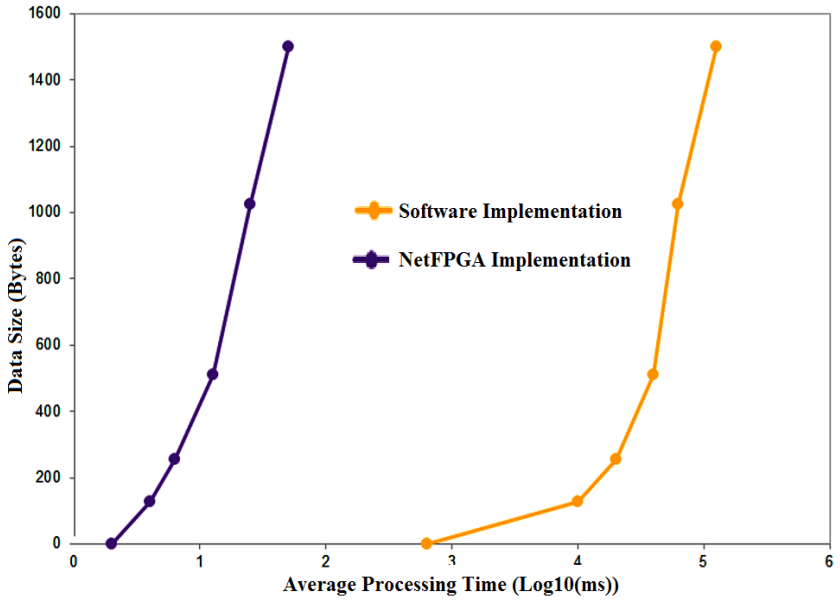
Table 2: Hardware/Software implementation platform comparison

Device	OS	Processor	Clock	RAM
Desktop	Ubuntu 14.1	Intel I5	2.4 GHz	8 GB
NetFPGA-10G	Custom Linux	Virtex-5	160 MHz	256 KB

In Fig. 5, a comparative study of average processing time for various data sizes is presented. Fig. 5 shows that the hardware implementation was approximately a thousand times faster at encrypting and decrypting information then the software implementation.



(a): Encryption time



(b): Decryption time

Figure 5: Performance evaluation: Hardware/Software PRESENT implementations

5 Conclusion

This paper presented an FPGA implementation of the lightweight PRESENT cipher for secure video streaming in a SDN Testbed. In this work, the PRESENT cipher is implemented in both software and hardware for performance comparison. The FPGA implementation proved to be more efficient and faster with fewer resources by taking

advantage of Verilog coding. This implementation executes one round per clock cycle to support both encryption and decryption at a minimal cost. A video application testbed outcome shows that only a negligible latency is observed for various video quality standards.

Acknowledgement: This work was supported by the National Natural Science Foundation of China under grant number 61471055, and European Horizon 2020 INPUT project "In-Network Programmability for next-generation personal Cloud service support", www.input-project.eu, under grant agreement number 644672.

References

Anwer, M. B.; Feamster, N. (2009): Building a fast, virtualized data plane with programmable hardware. *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 1, pp. 75-82.

Biryukov, A.; Perrin, L. (2017): State of the art in lightweight symmetric cryptography. *IACR Cryptology ePrint Archive*, vol. 2017, pp. 511.

Bogdanov, A.; Knudsen, L. R.; Leander, G.; Paar, C.; Poschmann, A. et al. (2007): Present: An ultra-lightweight block cipher. *Cryptographic Hardware and Embedded Systems*, vol. 4727, pp. 450-466.

Bruneau-Queyreix, J.; Lacaud, M.; Negru, D.; Batalla, J. M.; Borcoci, E. (2018): Adding a new dimension to http adaptive streaming through multiple-source capabilities. *IEEE MultiMedia*, vol. 1, no. 1, pp. 1-1.

Cho, J. Y. (2010): Linear cryptanalysis of reduced-round PRESENT. *Topics in Cryptology-CT-RSA 2010*, vol. 5985, pp. 302-317.

Cisco (2016): *Global VNI*. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/white-paper-c11-738085.html.

Feldhofer, M.; Dominikus, S.; Wolkerstorfer, J. (2004): Strong authentication for RFID systems using the AES algorithm. *Cryptographic Hardware and Embedded Systems*, vol. 3156, pp. 357-370.

Hong, D.; Sung, J.; Hong, S.; Lim, J.; Lee, S. et al. (2006): Hight: A new block cipher suitable for lowresource device. *Cryptographic Hardware and Embedded Systems*, vol. 4249, pp. 46-59.

Tepsic, G. (2018): *Use SSL/HTTPS to rank & look better on google.* https://fourdots.com/blog/why-you-need-ssl-to-rank-better-in-2016-and-how-to-set-it-2169.

Lim, C. H.; Korkishko, T. (2006): Mcrypton-lightweight block cipher for security of low-cost rfid tags and sensors. *Proceedings of the 6th International Conference on Information Security Applications*, vol. 3786, pp. 243-258.

Liu, G.; Jin, C.; Kong, Z. (2016): Key recovery attack for PRESENT using slender-set linear cryptanalysis. *Science China Information Sciences*, vol. 59, no. 3, pp. 1-14.

NetFPGA (2007): *NetFPGA*. http://NetFPGA.org/.

NetFPGA-10G (2009): *NetFPGA-10G*. http://NetFPGA.org/10Gspecs.html.

NetFPGA-1G (2007): *NetFPGA-1G*. http://NetFPGA.org/1Gspecs.html.

Standaert, F. X.; Piret, G.; Gershenfeld, N.; Quisquater, J. J. (2006): Sea: A scalable encryption algorithm for small embedded applications. *Smart Card Research and Advanced Applications*, vol. 3928, pp. 222-236.

Standard, A. E. (2001): *FIPS 197: Advanced Encryption Standard*. National Institute of Standards and Technology.

Unnikrishnan, D.; Vadlamani, R.; Liao, Y.; Dwaraki, A.; Crenne, J. et al. (2010): Scalable network virtualization using fpgas. *Proceedings of the 18th Annual ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, pp. 219-228.

VideoLan (2015): *VideoLan*. https://www.videolan.org/index.html.

Wang, M. (2008): Differential cryptanalysis of reduced-round PRESENT. *Progress in Cryptology-AFRICACRYPT 2008*, vol. 5023, pp. 40-49.