# Protecting Network-on-Chip Intellectual Property using Timing Channel Fingerprinting

ARNAB KUMAR BISWAS, Indian Institute of Information Technology Raichur, India
BIPLAB SIKDAR, National University of Singapore, Singapore

The theft of Intellectual property (IP) is a serious security threat for all businesses that are involved in the creation of IP. In this paper we consider such attacks against IP for Network-on-Chip (NoC) that are commonly used as a popular on-chip scalable communication medium for Multiprocessor System-on-Chip (MPSoC). As a protection mechanism, we propose a timing channel fingerprinting method and show its effectiveness by implementing five different solutions using this method. We also provide a formal proof of security of the proposed method. We show that the proposed technique provides better security and requires much lower hardware overhead (64% to 74% less) compared to an existing NoC IP security solution without affecting the normal packet latency or degrading the NoC performance.

CCS Concepts: • **Networks** → **Network on chip**; • **Security and privacy** → *Malicious design modifications*; • **Applied computing** → Computer-aided design.

Additional Key Words and Phrases: Intellectual property protection, NoC IP protection, fingerprinting technique, timing channel.

**ACM Reference Format:**
Arnab Kumar Biswas and Biplab Sikdar. 2021. Protecting Network-on-Chip Intellectual Property using Timing Channel Fingerprinting. *ACM Trans. Embedd. Comput. Syst.* 0, 0, Article 0 (November 2021), 21 pages. https://doi.org/10.1145/3495565

## 1 INTRODUCTION

Integrating intellectual property (IP) from different vendors into a single design is a common practice in modern Multiprocessor System-on-Chip (MPSoC) design. Network-on-Chip (NoC) is also used as an IP in MPSoC design process. This calls for built-in NoC IP security mechanisms to enable the widespread adoption of the technology without any security risks such as IP theft. Enabling NoC IP security can also provide security to the whole MPSoC chip that contains this IP. To the best of our knowledge, there are no specific reports of NoC IP theft. However, NoC IP is a specific case of semiconductor based IP and theft of such IP is a well reported matter. According to a recent report [18], multicore technology is becoming dominant in the consumer electronics sector and NoC IP is one of the main components of such technology. The report also indicates IP theft as one of the main challenges faced by this market sector. Also, the World Semiconductor Council (WSC), which comprises of companies in the semiconductor industry from around the world, has a dedicated task force [11] for IP theft prevention because of the serious nature of this threat.

Authors' addresses: Arnab Kumar Biswas, Indian Institute of Information Technology Raichur, Raichur, Karnataka, India, arnab@iiitr.ac.in; Biplab Sikdar, National University of Singapore, Singapore.


1539-9087/2021/11-ART0 $15.00
https://doi.org/10.1145/3495565

In this paper, we focus on the IP security threat for NoC. In this threat model, we consider adversaries that perform IP theft and engage in counterfeiting and cloning. Such adversaries may be individuals or even foundries that perform over-building and cloning [16]. The objective of this paper is to develop solutions that allow the owner of the NoC IP to prove the ownership of the IP from the over-produced or cloned products.

Watermarking is one of promising and commonly used techniques for establishing ownership of IP. There are many hardware IP watermarking methods available in literature. However, only [17] and [6] have specifically considered NoC IP security. Existing watermarking techniques for MPSoC IP security use features such as power consumption patterns, records of state variables, solutions to constraint based optimization problems, etc. These techniques require considerable amounts of storage or computational power, can be slow in extracting fingerprints, and are susceptible to noise.

In this paper, we propose a timing channel based NoC IP fingerprinting method. The term "timing channel based fingerprinting" is used to indicate that a timing channel is used to transfer the fingerprint. In our proposed method, the routers in the NoC modify the timing behavior of passing packets according to the embedded fingerprint. We propose five types of timing channels to show the effectiveness of the timing channel based fingerprinting. These timing channels produce five types of fingerprinting solutions: on-off timing channel based fingerprinting (OTF), distinct interval based timing channel fingerprinting (DITF), reordering based timing channel fingerprinting (RTF), reordering based on-off timing channel fingerprinting (ROTF), and reordering based distinct interval timing channel fingerprinting (RDITF).

The proposed timing channel based fingerprinting method ensures uniqueness of the NoC IP because non-fingerprinted NoC routers will never modify timing behavior of authentication packets. These packets' timing behavior do not get affected by traffic because of absence of any interfering traffic during authentication phase. The uniqueness property helps the IP provider to trace individual IP buyer and prevent forgery because each NoC IP becomes unique with fingerprint. In addition, our proposed method provides cryptographic security to any attack that tries to obtain owner's watermark, though the fingerprint can be easily obtained from the NoC. We believe that the owner's watermark needs to be kept secret from any malicious attacker. Later we will discuss how the SSP method in [17] does not provide cryptographic security to protect the owner's watermark though the fingerprint is not easily obtainable. Please note that the watermark of an owner is like his/her private or secret key. If this information is known to an attacker, the attacker may use this information in different situations to impersonate the owner's identity. Even if a watermarking method does not allow the attacker to use this known information into a forged design but the attacker can still use this information in interaction with a trusted third party which is required for ownership claim. If the attacker also can provide the same watermark to the trusted third party as the owner, it will not be possible to prove the ownership. So, we believe that the owner's watermark should always remain hidden from an attacker. Also, our proposed method allows easy and fast extraction of fingerprint from NoC which reduces the authentication time. In [6], we have proposed an NoC IP protection method called Circular path based fingerprinting (CPF) using fingerprint embedding. The main basis of that previous work is the observation that normal NoC prefers to avoid circular path creation when transferring packets. We have used polyomino theory to get the number of distinct fingerprints in an NoC using our CPF solution. In our current work, we consider the same threat model and provide similar level of security against those attacks like [6] but with lower area overhead (for most of our current solutions). We also use fingerprints to identify the owner of the IP but our current proposed solutions utilize timing channels which is not present in any existing NOC IP security solutions including our earlier work.

### 1.1 List of our contributions

The main contributions of this work are as follows:

(1) We propose a new NoC IP fingerprinting technique using timing channel based fingerprinting.
(2) We provide a detailed analysis of the proposed method by considering five new timing channels which result in five timing channel based fingerprinting methods: OTF, DITF, RTF, ROTF, and RDITF.
(3) We provide detailed description of router architectural modifications that need to be done to implement each of our proposed solutions.
(4) We provide a security analysis that formally proves the security provided by our proposed method.
(5) We show that our proposed method provides better security and requires less router area compared to existing work on NoC IP fingerprinting.

The rest of the paper organization is as follows. Related works are described in Section 2. Section 3 gives background information about Integrated circuit (IC) design and fabrication stages showing different types of IP usage and also the threat model considered in this paper. We describe our proposed technique in Section 4 and provide the security analysis, and simulation and synthesis results in Section 5. Section 6 concludes the paper.

## 2 RELATED WORKS

Different types of hardware IP watermarking techniques are discussed in this section.

In [12], the authors have used a watermark memory to store a watermark characterizing the IP and a watermark signal generator to generate a watermark signal detectable on a power supply line of the electronic circuit. The main disadvantage of this technique is that the watermark cannot be successfully detected in the presence of noise at the power supply line. In [3], a similar method that incorporates an authentication identification generating circuit in the IP is proposed. In this method, the power consumption waveform of the IP under a predetermined condition is initially stored. Later, the power waveform of a suspected IP under the same condition is checked against the previously stored waveform. Matching of waveforms authenticates the IP. The main disadvantage of this technique is that the storage required to store all waveforms is huge. Another disadvantage is that the predetermined condition needs to be exactly same and any noise at the power supply line has to be eliminated for exact waveform matching.

A watermark generating circuit (WGC) and a test circuit are proposed to embed into the IP core at the behavioral design level in [13]. Authors in [10] have proposed a blind fingerprinting technique by recording the state variables of a sequential circuit to embed a test machine. Constraint based watermarking is proposed in [14, 15]. A design is said to have a watermark embedded in it if a solution of a given optimization instance satisfies a particular set of constraints, hence providing a probabilistic proof of authorship. A key-less public watermarking method is proposed in [19] where the watermark is embedded into the original problem as a special (mutually exclusive) constraint. A generic fingerprinting technique is proposed in [8] by applying iterative optimization in an incremental fashion to solve a fingerprinted instance. The technique mainly considers a constraint based optimization solution that embeds a watermark as a seed solution. Then, different fingerprinted optimization solutions are obtained by considering each buyer's fingerprint and the seed solution. Thus, even though the fingerprinting optimization solution requires less effort than the seed solution, the designer still needs to do a thorough search in the solution space.

Authors in [17] have proposed a square spiral routing algorithm in NoC to embed the watermark. Each router in the selected square spiral path (SSP) stores a part of the watermark. At the examination phase, a trusted third party needs to check all possible SSPs in the NoC and gather information

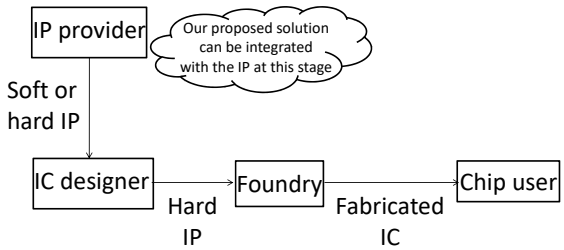


Fig. 1. Integrated circuit (IC) design and fabrication stages showing different types of IP usage. It also shows the stage where our proposed solution needs to be integrated with the IP.

to check against any possible match with the set of data from the designer. This requires a lot of searching and processing at the detection phase. Later it will be shown that the SSP method does not provide cryptographic security like the proposed method.

In [6], we have proposed an NoC IP protection technique called circular path–based fingerprinting (CPF) using fingerprint embedding. We have also provided a theoretical model using polyomino theory to get the number of distinct fingerprints in an NoC. This theory is applicable because of unique structure of NoC circular paths in any size of NoC. An authentication packet always travels through the previously set circular path(s) and extracts embedded fingerprint while passing through each router. This extracted fingerprint is validated to uniquely identify the owner.

## 3 PRELIMINARIES AND THREAT MODEL

This section presents a discussion on different types of hardware IPs and the threat model considered in this paper.

### 3.1 Preliminaries

In general, two types of IPs are used in the IC design flow: soft IP and hard IP. Synthesizable register-transfer level (RTL) code and gate level netlist are used as soft IP which allows synthesis, placement and route design flow. The IPs that are offered in layout format or GDSII (Graphic Design System II) format are known as hard IP. These IPs must obey the target foundry's process design rules. Figure 1 shows different stages of IC design and fabrication flow and use of different types of IPs. This discussion about IPs is applicable to NoC IP also. Please note that the chip design and fabrication flow is much more complicated and Figure 1 does not show the exhaustive details of the flow. Our main aim is to show the main entities and how they share and use different IPs and also to show where our solution is applicable.

### 3.2 Threat model

In this paper we consider IP stealing attacks against hard IP or fabricated ICs. We target the outsider adversary who performs IP theft, counterfeiting and cloning, as well as foundries that perform over-building and cloning [16]. Over-building and cloning refer to scenarios where the foundry produces the chip in excess numbers without the owner's permission, and probably with different labels to hide any claim from the owner. With our proposed solution, the owner can easily prove the ownership of the IP from these over-produced or cloned products. Counterfeiting refers to the scenario where a similar looking but fake product is made with the owner's label which does not cost much to produce. This type of sub-standard or poorly manufactured products cause reputation loss of the owner in the market, and hence, loss of revenue due to decrease of its product sales. Using the proposed solution, the IP owner can easily prove that the product is fake and it was not

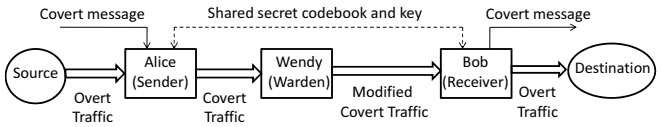


Fig. 2. A generic model of a covert timing channel showing various entities.

created using the owner's IP and can claim damages from the adversary. Usually, rogue business entities are driven only by profit. They are mainly interested in IP theft or misuse rather than tampering or changing the IP design [16]. This implies that our solution will also be useful against these types of adversaries. Also, note that in this work we assume that the owner is responsible for authenticating the IP in case of any doubt about IP ownership. In case there is a dispute between the IP owner and the IP user, a trusted third party is involved who checks and verifies the ownership of the IP and provide a judgment. In this case, the owner will provide all information including the watermark to help the trusted third party to make the ownership decision.

### 3.3 Basics of Timing Channels

A timing channel is a communication channel that can transfer information to a receiver/decoder by modulating the timing behavior of an entity. In our work, the entity is the inter-packet delay, the ordering of packets, and different combinations of both. Among different types of timing channels [7], we only consider the covert timing channel whose generic model is shown in Figure 2. Here, overt traffic denotes the actual data payload the packets are carrying and covert traffic denotes the hidden information that is transmitted using the timing channel. Alice is the sender of the covert message, Bob is the covert message's receiver, and they may or may not be the source and destination of the overt message, respectively. In our case, Alice (the covert sender) is different from the overt sender because the NoC routers together play the role of Alice. Covert receiver Bob and overt destination are located in the same place. There is another entity called Wendy (warden) whose task is to identify the presence of the timing channel and also to decode the covert message after detection. In our case, this is the attacker and is located at the destination of packets. The attacker tries to identify the presence of the timing channel and also tries to decode the covert information (in our case, the embedded fingerprint). Note that the attacker's main aim is to obtain the owner's watermark from the fingerprint which will allow the attacker to claim the IP ownership resulting in IP theft.

## 4 PROPOSED FINGERPRINTING TECHNIQUES

In this section, we describe the proposed fingerprint generation process and the five types of timing channel based NoC IP fingerprinting methods.

### 4.1 Overview of Proposed Technique

Figure 3 gives a brief overview of our work using two flowcharts. Figure 3(a) shows the NoC IP buying phase and Figure 3(b) shows the authentication phase. Note that the authenticator can be the IP owner herself or a Trusted Third Party. Further details of each step are given in later sections.

### 4.2 Fingerprint Generation

Figure 4(a) shows the fingerprint generation process. A designer or owner of the IP has a set of watermarks $\{W_0, \ldots, W_m\}$ and the aim of the fingerprint generation process is to generate a

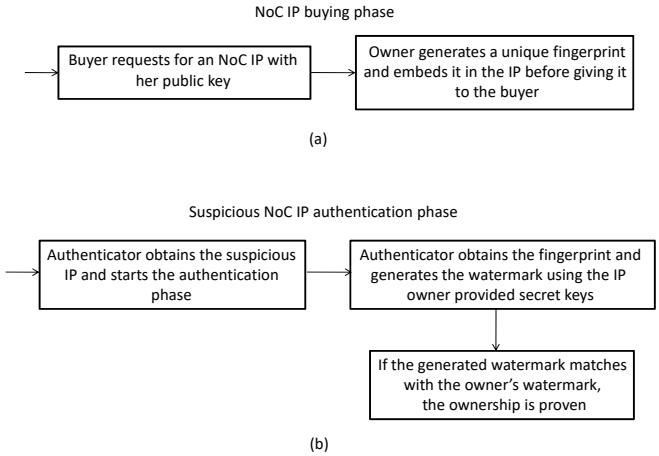


Fig. 3. Flowcharts showing brief overview of our work with (a) NoC IP buying phase, and (b) suspicious NoC IP authentication phase.

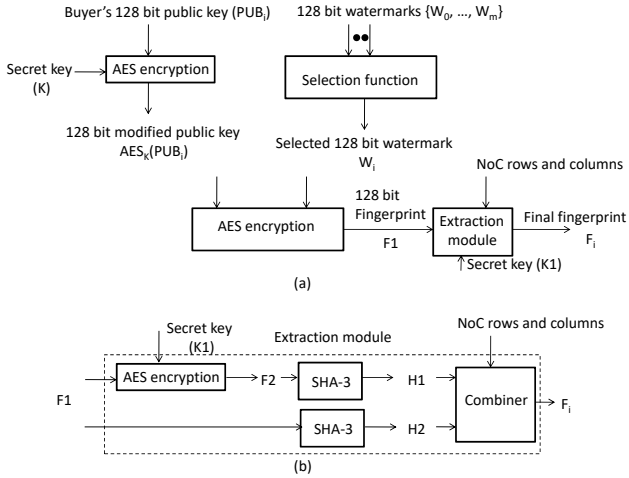


Fig. 4. Fingerprint generation process. The step to select one secret key from a set of keys is omitted.

fingerprint $F_i$ specific to a buyer $i$ using a randomly chosen watermark $W_i$ from this set. Here, we assume that the watermark is 128 bit long. When a buyer requests to buy an IP, the buyer's public key ($PUB_i$) is passed to an Advanced Encryption Standard (AES) encryption engine that uses an 128 bit secret key ($K$) to generate 128 bit output $AES_K(PUB_i)$. The AES output is obtained by performing encryption of the buyer's public key ($PUB_i$), considering it as the plaintext message. Note that AES is chosen for encryption because it provides sufficient security against an attacker who knows the plaintext message ($PUB_i$) but does not know the secret key ($K$) used for encryption (i.e., security against the known plaintext attack) [2]. Thus, the attacker does not know the encrypted (modified) public key $AES_K(PUB_i)$ that is obtained after encryption. The Selection Function randomly selects one watermark $W_i$ from the set of watermarks available to the IP owner. Then, a fingerprint $F1 = AES_{AES_K(PUB_i)}W_i$ is obtained by encrypting (using AES) this watermark $W_i$ using the modified

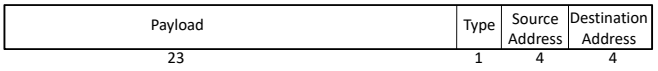


Fig. 5. Normal packet with different field widths (given in bits).

public key $AES_K(PUB_i)$. Next, the Extraction Module gives the final fingerprint $F_i$ after taking the target NoC size (number of rows and columns) and the generated fingerprint $F1$ as inputs. The details of the Extraction Module are shown in Figure 4(b). Inside the Extraction Module, fingerprint $F1$ is encrypted (using AES) using a secret key to generate fingerprint $F2$. Next, two hash values $H1$ and $H2$ are obtained using SHA3 from the two fingerprints $F1$ and $F2$, respectively. The length of the hash values are randomly chosen such that it is ensured that if both are combined together, they produce the final fingerprint ($F_i = H1||H2$) which will be embedded in the NoC as embedded fingerprint. Note that the encryption of $F1$ to generate $F2$ and generation of $H1$ and $H2$ from these fingerprints are mainly done to provide cryptographic security against NoC IP stealing attack. These steps provide necessary security against an attacker who wants to falsely prove to a trusted third party that he/she owns that particular NoC IP. The security implications of these steps are further discussed in Section 5.

In all of our proposed fingerprinting techniques, the fingerprint is embedded at the NoC design stage so that it can be extracted later from the IP and prove the ownership. That means it is the owner's responsibility to maintain the set of watermarks and secret keys well protected and to generate an unique fingerprint after receiving the buyer's public key along with the NoC size. After design and integration into an MPSoC, the NoC IP has two working phases: normal working phase and authentication phase. During the normal working phase, the NoC transfers regular packets between different processing elements of the MPSoC according to the running applications. Figure 5 shows normal packet structure with a 1 bit type field. This bit is 0 for normal packets. If there is any doubt about ownership, the authentication phase is started. During this phase, authentication packets with 1 in the type field are used for extraction of the embedded fingerprint from the NoC IP.

### 4.3 Proposed Timing Channel based Fingerprinting Technique

The fingerprint is generated by the routers in the NoC. First, we need to select (source, destination) pairs that will decide the path of authentication packets. In general, any (source, destination) pair that ensures that there is no impact of one authentication traffic flow on another can be selected. This prevents unintentional modification of timing channel information. Our design uses the rows of the NoC as authentication packet paths as shown in Figure 6. As shown in the figure, the authentication packets are injected at the left end of each row (e.g., (0,0), (0,1), etc.) and packets are ejected from the right end of the same row (e.g., (3,0), (3,1), etc.). Once the paths are decided, routers in those paths are set (using their routing tables) so that the authentication packets flow through them. Next, the embedded fingerprint is divided among all the timing channel encoders of all the routers in those paths. All these steps of fingerprint embedding method are shown in the flowchart of Figure 7. Note that different timing channel encoders are used for different timing channel methods, like on-off timing channel encoders are used in OTF method, reordering based timing channel encoders are used for DITF method, and so on. During the authentication phase, authentication packets are sent through every row. Multiple packets need to be sent in each row so that all the embedded information in that row can be extracted. Routers modulate the arrival time of passing packets to encode the embedded information using the timing channel according to the method used (i.e., OTF, DITF, RTF, ROTF, and RDITF). The details of each method are described later

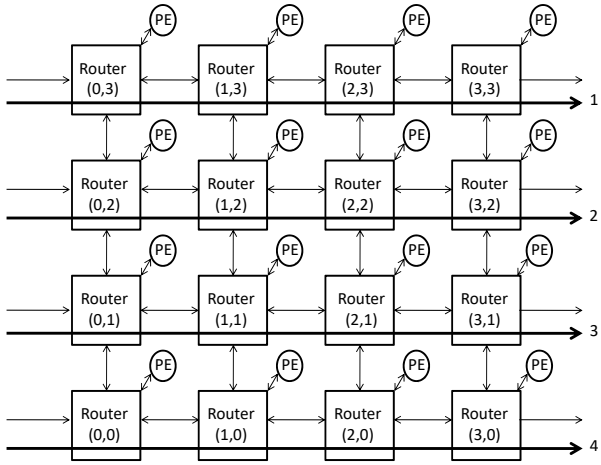


Fig. 6. A $4 \times 4$ NoC with authentication packets traversing in every row. In this figure PE stands for processing element.

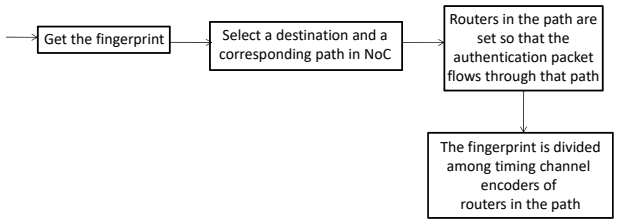


Fig. 7. Flowchart of fingerprint embedding method using timing channel.

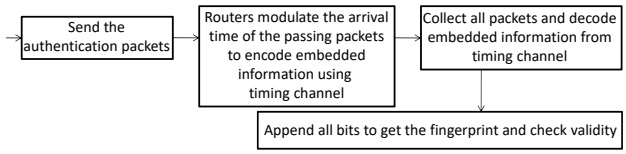


Fig. 8. Flowchart of fingerprint decoding method from NoC routers.

using router architecture. After receiving all packets, the embedded timing channel information are decoded to get the embedded fingerprint and this is then used to check the validity. The different steps of the decoding method from NoC are shown in the flowchart of Figure 8.

### 4.4 Router Architecture and Authentication Packet Routing

Figure 9 shows the proposed router architecture to implement different timing channel fingerprinting techniques. Note that the packet storage is only required for RTF, ROTF and RDITF methods. The router has five input and five output ports: North (N), South (S), East (E), West (W) and Eject (I). Only two input and two output ports are shown in the figure. The type checker module checks the input packet's type field when it enters the input first-in-first-out (FIFO) queue of an input port

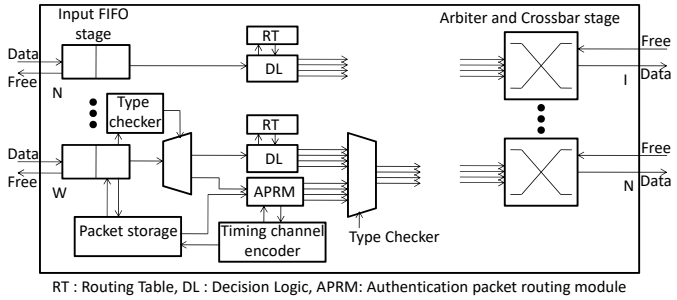


Fig. 9. Proposed router architecture where the packet storage is present only for RTF, ROTF and RDITF methods.

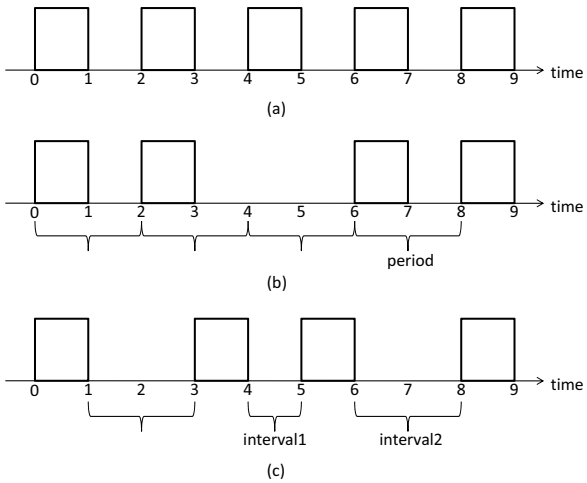


Fig. 10. Authentication packet flow (a) without any timing channel, (b) with on-off timing channel using a period of two clock cycles and (c) with distinct interval based timing channel using intervals of one and two clock cycles.

and transfers it to decision logic (for normal packets) or to authentication packet routing module (APRM) (for authentication packets). The decision logic routes normal packets using a routing table (RT) based deterministic Y-X routing algorithm. The timing channel encoder embeds the fingerprint bit in the authentication packet by modulating the packet's timing behavior and the APRM sends the packet to the correct output port using the same RT based deterministic Y-X routing algorithm. The actual timing modification depends on the specific methods that are described next. Note that in this design the APRM and related modules are only present at the West input port as we have selected the rows (West in and East out) as the target paths.

*4.4.1 OTF and DITF methods.* The flow of authentication packets without any timing channel is shown in Figure 10(a). In the OTF technique, presence or absence of a packet within a period indicates an encoded bit. Accordingly, the example shown in Figure 10(b) uses a period of two clock

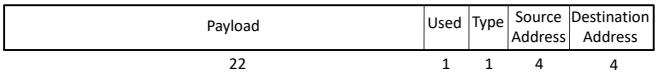


Fig. 11. Packet structure used for OTF and DITF methods.

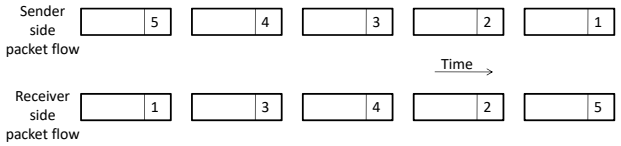


Fig. 12. Authentication packet flow (a) without any timing channel and (b) with reordering based timing channel.

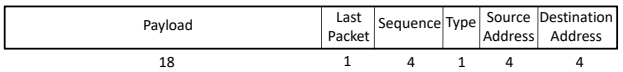


Fig. 13. Packet structure used for RTF method.

cycles and encodes 1101. The DITF technique uses two distinct inter-packet delays to convey the covert bits 1 and 0. The example shown in Figure 10(c) uses interval1 to encode 0 and transmits 101.

The packet structure used for OTF and DITF techniques is shown in Figure 11. In case of OTF, routers send packets to encode 1 and do not send to denote 0. This means that a router needs to inform the next routers if the transmitted packet has already been used to encode a bit 1 or if the next router can use the packet. If the next router uses an already used packet, the timing channel information associated with that packet will be destroyed. Thus, the packet structure has a 'Used' field which is set by routers if that packet has been used to encode a bit 1. In case of DITF, routers need to send packets both for 1 and 0 with distinct intervals. Consequently, the 'Used' field is set for both 1 and 0 in the DITF method.

*4.4.2 RTF method.* Figure 12 shows the authentication packet flow without and with reordering based timing channel. Every packet has a sequence number field that helps to identify the order of packets. Note that the receiver side packet sequence is different than the sender side packet flow sequence. This is because a particular bit sequence is encoded by a particular order of the packets. An inter-packet delay of 1 clock cycle is maintained in the flow.

Each router in the RTF method uses the 'packet storage' element to store incoming packets so that they can be sent following a particular order which encodes a set of bits. Once a group of packets is correctly ordered, the next routers do not store or reorder them again. The RTF method uses a triggering method to inform the routers when to start reordered packets. Once this triggering packet is received, the router starts sending packets in an order according to the preset embedded information. The 'Last Packet' field in the packet structure as shown in Figure 13 is specifically used by the last packet of the group to indicate the end of incoming packets. Figure 13 also shows that the packet has a field called 'Sequence' that is used to carry the sequence number.

*4.4.3 ROTF and RDITF methods.* The ROTF method is a combination of RTF and OTF methods where a particular packet order encodes a specific set of bits and simultaneously, the inter-packet

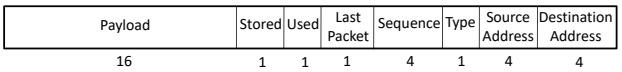


Fig. 14. Packet structure used for ROTF and RDITF methods.

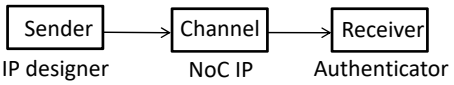


Fig. 15. NoC IP fingerprinting channel model.

delay is modulated using on-off timing channel to encode extra fingerprint bits. Similarly, the RDITF method is a combination of RTF and DITF methods where the inter-packet delay is modulated using a distinct interval based timing channel.

The packet structure used for ROTF and RDITF methods is shown in Figure 14. This is mainly a combination of packet structures for OTF/DITF and RTF methods. One additional field called 'Stored' is present to help the ROTF method. This bit indicates that the packet has already been used in a reordering operation and can now be used by the on-off timing channel. The 'Used' bit indicates that the packet has already been used to encode 1. So if the 'Stored' bit is set and 'Used' bit is not set, a router can use the packet for the on-off timing channel. This may happen because in the ROTF method, absence of a packet encodes bit 0. Thus, a stored and reordered packet may not contribute to an on-off timing channel fingerprint bit when it is sent from a router to the next one. The same packet structure is used for RDITF method but in this case, both 'Used' and 'Stored' bits mean same. This is because a packet needs to be sent with different inter-packet delay to encode both 0 or 1 using distinct interval based timing channel and both fields are set in both cases.

### 4.5 NoC IP Fingerprinting Channel Capacity

This section analyzes the proposed fingerprinting technique using an information channel capacity model. The concept of fingerprinting an IP for subsequent checking can be considered as a message transmission over a channel. Figure 15 shows the NoC IP fingerprinting channel model. Here, the start of the message is the selling of the IP to the buyer after embedding the fingerprint in the IP. In our case, NoC IP is the channel that carries the message from sender to receiver by modulating some property of the IP. This model is applicable to all IP watermarking and fingerprint techniques and rigorous mathematical models are available in literature ([4, 20]). In this paper, we assume that the channel is noiseless because we can obtain the embedded fingerprint without any error. Thus, the channel capacity for our proposed methods can be obtained by considering the maximum information embedding capacity and the time required to obtain the information from the NoC IP by the authenticator. This is because the channel capacity is the upper bound on the rate at which information can be transmitted over a channel from the sender (IP designer) to the receiver (authenticator). Note that the authenticator is the one who obtains and authenticates the NoC IP and it can be same as the IP owner or a trusted third party (TTP). In general, we consider TTP as the authenticator because the IP owner can also authenticate the IP by playing the role of TTP as a special case.

Let us assume that for an $n \times n$ NoC each router can encode $m$ bits of embedded information using an on-off or distinct interval based timing channel. Then, in the case of OTF and DITF methods,

the channel capacity ($CC_{OTF/DITF}$) is

$$CC_{OTF/DITF} = \frac{n^2 m}{T_1}, \tag{1}$$

where $T_1$ is the fingerprint extraction and authentication time.

For the RTF method, $P$ packets can encode $\log_2 P!$ bits of information when passing through one row. When considering all rows in the NoC, the channel capacity ($CC_{RTF}$) is

$$CC_{RTF} = \frac{n \log_2 P!}{T_1}. \tag{2}$$

In the ROTF method, the channel capacity ($CC_{ROTF}$) is the total of RTF and OTF methods as given by

$$CC_{ROTF} = \frac{n(nm + \log_2 P!)}{T_1}. \tag{3}$$

In case of RDITF method also, the channel capacity ($CC_{RDITF}$) is obtained from the RTF and DITF methods. However, the routers in a row cannot use all the reordered packets because the first router needs to send the first packet only to indicate the start of distinct interval based timing channel. This means a loss of 1 bit because each packet is used to encode a 1 or 0. Thus, the channel capacity ($CC_{RDITF}$) is

$$CC_{RDITF} = \frac{n(nm - 1 + \log_2 P!)}{T_1}. \tag{4}$$

Note that $T_1$ is $O(1)$ because the authentication paths are fixed after design and the authenticator doesn't need to search for it.

For comparison purposes, if we consider the case of the existing SSP method [17], at most $(n(n-1)+1)b$ bits can be embedded in an $n \times n$ NoC where $b$ bits are stored in each router. The first term is the number of routers in the path and it is less than $n^2$ because the shape of the square spiral path does not include the first column/row of routers, except the source router. If we assume $T_2$ as the fingerprint extraction and authentication time, the channel capacity of SSP ($CC_{SSP}$) is

$$CC_{SSP} = \frac{(n^2 - n + 1)b}{T_2}. \tag{5}$$

Here $T_2$ is $O(n^2(n^2-1))$ because of the searching time of the authentication path itself among all possible paths. Again, the evaluation time for each path includes the packet traversal time to flow through that path.

Note that we assume that the packets are able to flow through the NoC without facing any fault. Regarding the presence of noise in the on-chip interconnection, it can be modeled using bit error rate ($\epsilon$) which indicates the average number of bits received in error. This bit error rate depends on various on chip parameters like voltage noise and channel cut off frequency (refer to [21] for detailed analysis). In our previous analysis, in the presence of noise, every expression of channel capacity for different cases will be reduced by $\epsilon$.

## 5 SECURITY ANALYSIS, SIMULATION AND SYNTHESIS RESULTS

This section present the security analysis of our proposed fingerprinting technique and provides simulation and synthesis results.

### 5.1 Formal Security Analysis

This section provides a formal analysis of the security properties of the proposed fingerprinting mechanism. The objective of the analysis is to show that it is difficult for an adversary to devise its own watermark that results in the same fingerprint as that embedded by the real owner of the IP, in order to claim the IP as its own. The semantic security of the proposed mechanism is based on an adaptation of the Real-or-Random (RoR) model [1]. In the RoR model, an adversary $\mathcal{A}$ interacts with the $k$-th participant instance $\Pi^k$. In the proposed fingerprinting mechanism, we denote the IP owner and the trusted third-party's instances by $U$ and $S$, receptively. The RoR model is based on the use of Execute, Send, Reveal, CorruptDevice, and Test queries to simulate attack scenarios and these queries are defined as:

- Execute($\Pi_U^{k_1}, \Pi_S^{k_2}$): this query models an eavesdropping attack where $\mathcal{A}$ can obtain the messages communicated between $U$ and $S$. In our scenario, this models the adversary's ability to observe the fingerprints generated by the NoC.
- Send($\Pi^k, m$): this query models active attacks where $\mathcal{A}$ can send a message $m$ to participant instance $\Pi^k$, and then receive a response from $\Pi^k$.
- Reveal($\Pi^k$): this query allows $\mathcal{A}$ to obtain the watermark $w$ registered between $\Pi^k$ and its partner.
- CorruptDevice($\Pi^k$): this query models an active attack where $\mathcal{A}$ can obtain all the sensitive information, such as any infomation stored on the NoC.
- Test($\Pi^k$): in this query, $\mathcal{A}$ requests $\Pi^k$ for watermark $w$ and receives a probabilistic output based on an unbiased coin $c$. If no watermark for instance $\Pi^k$ is defined or if a Reveal query was asked to either $\Pi^k$ or its partner, then the query returns the undefined symbol $\perp$. Otherwise, the query returns the watermark $w$ if $c = 1$, and returns a random number of same length if $c = 0$.

Definition 1. *Semantic security of the watermark:* In our adaptation of the RoR model to evaluate the security of the watermark, the adversary is challenged in an experiment to distinguish between an instance's real watermark and a random number. The adversary may ask multiple Test queries to the trusted third-party instance. The output of the Test query must be consistent with regard to the random bit $c$. The objective of the adversary is to guess the hidden bit $c$ used by the Test oracle. In the end, $\mathcal{A}$ returns a guess bit $c'$ and wins the game if $c' = c$.

Let SUCC denote the event where the adversary wins the game. The advantage of $\mathcal{A}$ with time complexity $t$, in breaking the semantic security of the fingerprint generation protocol (*fgp*) $P$ when the AES passwords are drawn from a dictionary $D$ is defined as

$$Adv_{P,D}^{fgp}(\mathcal{A}, t) = \mid 2\Pr[SUCC] - 1 \mid . \tag{6}$$

The protocol $P$ is said to be secure in the ROR sense if $Adv_{P,D}^{fgp}(\mathcal{A}, t)$ is negligible.

The security proof uses a collision resistant one-way hash function (denoted by $h(\cdot)$), which, along with AES (denoted by function $E$), are modeled as pseudo-random number generator based oracles.

Definition 2. *Secure pseudo-random function:* Assume that $f(\cdot)$ is a polynomial-time computable function and consider a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ that wishes to distinguish $f(\cdot)$ from any other function. $\mathcal{A}$ may request polynomialy bounded queries with its selected inputs and obtain the outputs computed by $f(\cdot)$ for training. Once the training phase is complete, $\mathcal{A}$ is provided with a function that is either $f(\cdot)$ or a truly random function. $f(\cdot)$ is defined as a secure pseudorandom function if it is indistinguishable from a truly random function under $\mathcal{A}$, i.e., $\mathcal{A}$

is given either $f(\cdot)$ or a truly random function according to a random bit {0, 1} and it can only distinguish $f(\cdot)$ with a probability of $\frac{1}{2} + \varepsilon$.

**Theorem 1:** Let $\mathcal{A}$ be a polynomial time adversary running in time $t$ against the fingerprinting protocol $P$. Then, the advantage of $\mathcal{A}$ in breaking the semantic security of the proposed fingerprinting mechanism can be represented as

$$Adv_{P,D}^{fgp}(\mathcal{A}, t) \leq \frac{q_h^2}{|Hash|} + \frac{2q_{send}}{|D|}$$

where $q_h$ and $q_{send}$ denote the number of hash and Send queries, respectively. Also, $|Hash|$ and $|D|$ denote the range space of the hash output and the size of the AES password dictionary $D$, respectively.

**Proof.** The proof is based on a sequence of games denoted by $G_i$, where $i = 0, 1, 2, 3$. Let $SUCC_{\mathcal{A}}^{G_i}$ denote the event that $\mathcal{A}$ correctly guesses the random bit $c$ in game $G_i$. The advantage of the adversary to win game $G_i$ is then given by

$$Adv_{\mathcal{A},G_i}^{fgp} = \Pr\left[SUCC_{\mathcal{A}}^{G_i}\right]$$

and corresponds to the case where the adversary correctly guesses the hidden bit $c$ involved in the Test queries. The proof is based on modeling the adversary's advantage in a sequence of games, starting with the real attack on the fingerprinting mechanism (considered in game $G_0$) and ending in a game in which the adversary's advantage is 0, and by bounding the difference in the adversary's advantage between any two consecutive games.

**Game** $G_0$: This game models the real attack by the adversary on the fingerprinting mechanism $P$ proposed in this paper. Since the bit $c$ was chosen randomly at the start of game $G_0$, we have

$$Adv_{P,D}^{fgp}(\mathcal{A}, t) = \left|2Adv_{\mathcal{A},G_0}^{fgp} - 1\right|. \tag{7}$$

**Game** $G_1$: This game simulates the adversary's ability to observe the information used for verifying the fingerprints by $U$ and $S$ (e.g., the fingerprint generated by the NoC) by using the Execute query. At the end of the game, the adversary uses the Reveal and Test queries and has to decide whether the output of Test is the real watermark or a random number. In the proposed fingerprinting scheme, the fingerprint is created through a sequence of AES encryptions and hash functions. Thus, the adversary needs to know the secret keys shared by the IP owner and the trusted third-party in order to verify the watermark corresponding to a fingerprint. Knowledge of the fingerprint resulting from the Execute query does not help in increasing the adversary's probability of winning the game $G_1$. Thus, $\Pr\left[SUCC_{\mathcal{A}}^{G_1}\right] = \Pr\left[SUCC_{\mathcal{A}}^{G_0}\right]$ and we have

$$Adv_{\mathcal{A},G_1}^{fgp} = Adv_{\mathcal{A},G_0}^{fgp}. \tag{8}$$

**Game** $G_2$: This game differs from $G_1$ by simulating active attacks using Send and hash queries. The adversay needs to find a hash collision that results in the same fingerprint in order to deceive the trusted third-party. Thus, $\mathcal{A}$ may employ multiple hash queries to find a digest collision. Recall that we assume that the hash function can be modeled as secure pseudo-random functions and their collision probability is thus negligible. From the birthday paradox of hash functions, we then have:

$$\left|Adv_{\mathcal{A},G_1}^{fgp} - Adv_{\mathcal{A},G_2}^{fgp}\right| \leq \frac{q_h^2}{2|Hash|}. \tag{9}$$

**Game** $G_3$: In this game, the adversary is also allowed to use the CorruptDevice query. Using this query, the adversary may obtain any information that is stored on the NoC. In the proposed fingerprinting mechanism, the NoC is not required to store any secret information (such as keys)

in its memory. Hence, $\mathcal{A}$ does not obtain any secret information by using the CorruptDevice query and there is no additional gain in its advantage by simulating this game. However, the adversary may use guessing or online dictionary attacks to guess the passwords involved in the protocol $P$. Thus, we have:

$$\left|Adv^{fgp}_{\mathcal{A},G_2} - Adv^{fgp}_{\mathcal{A},G_3}\right| \leq \frac{q_{send}}{|D|}. \tag{10}$$

All available queries/oracles were simulated in the last game. Once the adversary has tried all available mechanisms for breaking the security of the protocol $P$ without any success, as a last resort, it can resort to a guess for the random bit $c$. Thus, the adversary's probability of success in game $G_3$ equals the probability of correctly guessing the bit $c$ and we have

$$Adv^{fgp}_{\mathcal{A},G_3} = \frac{1}{2}. \tag{11}$$

From games $G_0$, $G_1$, $G_2$ and $G_3$, we have $Adv^{fgp}_{\mathcal{A},G_0} = Adv^{fgp}_{\mathcal{A},G_1}$, $\left|Adv^{fgp}_{\mathcal{A},G_1} - Adv^{fgp}_{\mathcal{A},G_2}\right| \leq \frac{q_h^2}{2|Hash|}$, $\left|Adv^{fgp}_{\mathcal{A},G_2} - Adv^{fgp}_{\mathcal{A},G_3}\right| \leq \frac{q_{send}}{|D|}$, and $Adv^{fgp}_{\mathcal{A},G_3} = \frac{1}{2}$. Using these in (7) along with the triangular inequality, we get

$$\begin{aligned}
\frac{1}{2}Adv^{fgp}_{P,D}(\mathcal{A},t) &= \left|Adv^{fgp}_{\mathcal{A},G_0} - \frac{1}{2}\right| \\
&= \left|Adv^{fgp}_{\mathcal{A},G_1} - Adv^{fgp}_{\mathcal{A},G_3}\right| \\
&\leq \left|Adv^{fgp}_{\mathcal{A},G_1} - Adv^{fgp}_{\mathcal{A},G_2}\right| + \left|Adv^{fgp}_{\mathcal{A},G_2} - Adv^{fgp}_{\mathcal{A},G_3}\right| \\
&\leq \frac{q_h^2}{2|Hash|} + \frac{q_{send}}{|D|}.
\end{aligned}$$

Multiplying both sides by 2, we have:

$$Adv^{fgp}_{P,D}(\mathcal{A},t) \leq \frac{q_h^2}{|Hash|} + \frac{2q_{send}}{|D|}. \tag{12}$$

■

### 5.2 Informal Security analysis

One of the main objectives during the design of the fingerprinting mechanism is that a non-watermarked NoC should not be detected as an watermarked one. The probability of occurrence (false detection) of such events should be very low. This property is known as the uniqueness property. In our case, the false detection probability is 0. This is because a non-fingerprinted NoC will not cause any change in packet timing behavior. In the absence of any cross-traffic to impact packet delays, routers simply route packets to the destination as and when packets arrive at the input port. The uniqueness property helps to prevent forgery of owner's watermark.

Note that in the proposed technique, the embedded fingerprint cannot be extracted easily from the NoC because the attacker first needs to detect the presence of a timing channel before it can decode the information. Only then she can extract the final/embedded fingerprint. If she wants to claim ownership of the IP, she needs to obtain the two hash values from the final fingerprint and also obtain the two fingerprints from the hash values such that one fingerprint can be obtained by encrypting the other fingerprint using AES. If the attacker wants to claim ownership of more than one IP, she also needs to obtain the watermark but this is prevented by AES. Note that both types of attacks are prevented by the standard security provided by AES and SHA3. Ultimately, the watermark which denotes the owner's identity will remain hidden.

The fingerprint generation process shown in Figure 4 provides cryptographic security against any attempt to obtain the watermark from the final fingerprint. This ensures that the attacker cannot obtain the watermark of the owner and cannot misuse this information. This is in contrast to the existing work like [17] where the owner's watermark is not hidden or kept secret from the attacker. Even though they have used AES to encrypt the watermark, in their case, the path information is exactly same as the watermark. Thus, a successful extraction of the embedded encrypted information is enough for an attacker to know the watermark of the owner without any need to decrypt it. It appears that the encryption is simply done to increase the number of fingerprints in [17] and not to provide any cryptographic security. This implies that while our solution provides cryptographic security, the work in [17] only provides security which is due to the effort required to find the exact path. This is much weaker than the security obtained due to the use of AES in our case. Our previous solution in [6] provides similar level of cryptographic security compared to our current solution because we have used similar method to generate the fingerprint from owner's watermark.

Masking and removal attacks can be used to change the IP design to remove or hide the fingerprint without changing the functionality of the IP. This attack can be prevented by hiding the IP details (specifically the router architecture) using hardware obfuscation techniques. We assume that layout level obfuscation techniques such as those mentioned in [5] are used to obfuscate the NoC IP. Note that we consider only hard IP or fabricated chips in this paper as mentioned in the threat model in Section 3.2. Note that, although expensive, an obfuscation method by itself can also protect from reverse engineering and IP piracy. If an owner wants to prove his/her ownership, he/she has to remove each layer of the chip to extract the layout. In this paper, we are proposing a much simpler solution which doesn't require this costly procedure to prove the ownership. The proposed solution only requires access to the chip to check the ownership. Thus, in addition to an obfuscation method, our solution can also provide full protection. Additionally, it is much simpler to prove ownership with the proposed solution as compared to gate level extraction and it is also robust at the gate level because of the obfuscation method. Since the main target of the obfuscation in our case is to hide the details of hardware implementation of each router, we believe that layout level obfuscation methods and camouflaging will be effective. Please note that the existing SSP method in [17] also uses an obfuscation method as mentioned in that paper.

Additionally, it is mentioned in [17] that the routers with SSP are different from other routers as per hardware implementation, which is not true in our proposed technique. Thus, the security attributed by the hidden nature of a path in SSP gets nullified because an attacker can easily distinguish SSP routers from the non-SSP routers that are different from hardware implementation point of view.

### 5.3 Simulation and Synthesis Results

In this section, we first simulate and analyze the NoC performance for different scenarios and later we provide the synthesis results to show the hardware resource overhead.

*5.3.1 Impact on NoC IP performance.* We implemented the routers for the proposed techniques and also for [17] using Bluespec System Verilog and simulated different NoC sizes to check if the no-load latency of normal packets gets modified due to addition of security measures in routers. We only change the routers for every simulation case and use the packet structure shown in Figure 5. Our proposed routers are single cycle routers and the router in [17] is 2 cycle router where one additional output stage is present due to security additions as mentioned in [17]. From the architecture point of view, there will not be any effect on normal packet's latency during normal working phase of the NoC because of the modifications. We use artificial traffic generators and

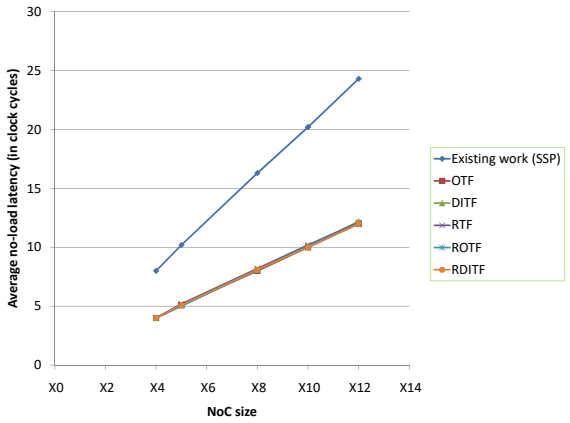


Fig. 16. Average no-load latency values (in clock cycles) for different NoC sizes.

sinks for traffic generation and measurements. The no-load latency measurement is performed by keeping the source-destination pair fixed for 1000 clock cycles and changing the pair after all packets are received. In this way, we ensure that there is no traffic from any other packet flow which will impact the packet latency. This method is followed while keeping the injection rate at the source very low (0.1) which ensures that there is no back pressure from the earlier packets sent from the source. Note that the aim of no-load latency measurement is to find if there is any effect of the modified router on the packet latency and this is done by eliminating the impact of traffic on the latency. This also ensures fair comparison with authentication packet latency where no other packet flow is present which can affect the authentication packet latency. Figure 16 shows the average no-load latency values in clock cycles for different cases. The no-load latency values of [6] are identical to our current solutions and that is why they are not included in Figure 16 separately. We also simulate the authentication phase for different cases and the average latency values are exactly the same as shown in Figure 16. That shows that there is no difference between normal no-load latency and authentication packet latency due to the security additions.

If we consider the change in latency as compared to a non-secure router which does not contain the modifications due to proposed security measures, the proposed technique does not increase the packet latency. However, the router architecture in [17] has an additional output stage due to the addition of security measures, as compared to a non-secure router. That means in case of [17], the latency increases two times compared to a non-secure router, for both normal packets and authentication packets.

We also modify the Noxim simulator [9] to simulate normal packet behavior in the NoC with our proposed solution using OTF. We only provide simulation results for OTF because all of our proposed solutions behave exactly similarly for normal packets as shown in Figure 16. A $4 \times 4$ NoC is considered for our evaluation and all the simulations are done for 15000 clock cycles. The warm-up period is 1000 clock cycles, measurement period is from 1000 to 10000 clock cycles, and remaining clock cycles are the drainage period. Figure 17 shows the average packet latency for different routing algorithms for different packet injection rates under various traffic distributions. Figures 17(a) and 17(b) show that Dyad routing results in saturation much earlier and XY routing results in saturation much later than other routing mechanisms, for random traffic with and without hotspots, respectively. Also, Figures 17(c) and 17(d) show much lower latency values for transpose1 and transpose2 traffic, respectively, compared to random traffic. This is because both transpose1

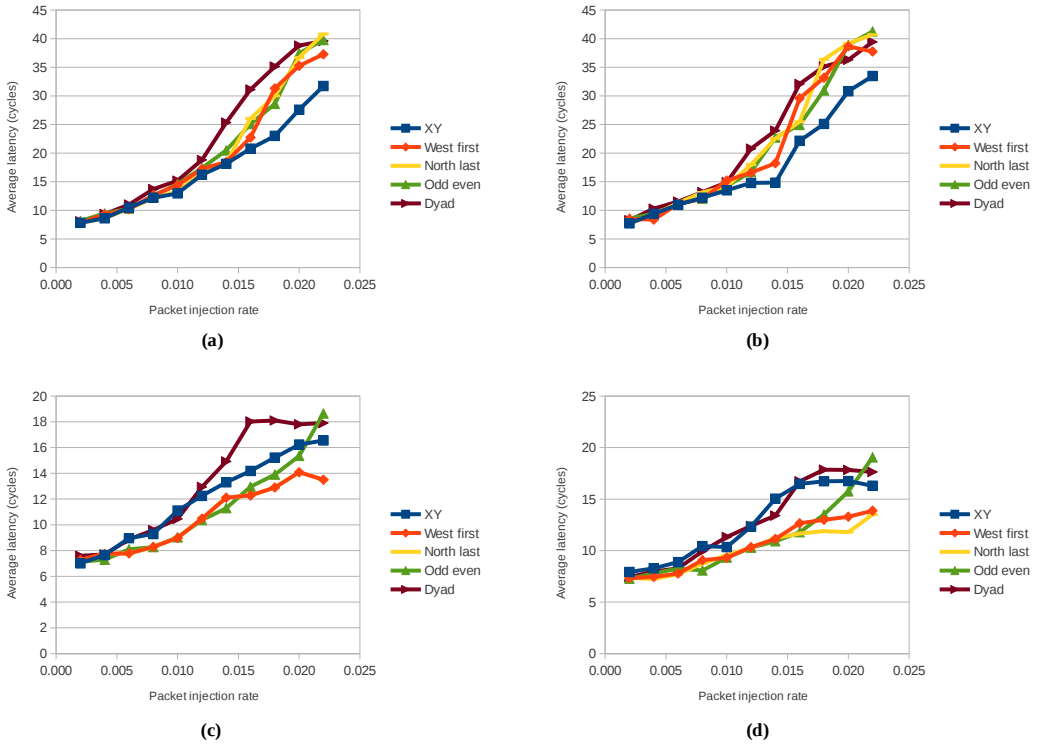


Fig. 17. Average latency values (in clock cycles) for different packet injection rates for (a) random traffic, (b) hotspot at (2,2) with 6% probability, (c) transpose1 traffic and (d) transpose2 traffic.

Table 1. Area and power results for different routers.

| | Area (in $\mu m^2$) | Power (in mW) |
|---|---|---|
| OTF | 605.24 | 0.350 |
| DITF | 608.93 | 0.352 |
| RTF | 823.92 | 0.544 |
| ROTF | 842.37 | 0.547 |
| RDITF | 843.83 | 0.547 |
| SSP [17] | 2353.96 | 1.46 |
| CPF [6] | 841.97 | 0.547 |
| Non secure | 590.76 | 0.337 |

and transpose2 traffic have a fixed destination for every source and cause less congestion than random traffic. All of these results clearly show that there is no change in normal packet flow behavior in our solution.

In addition to synthetic traffic, we also use real benchmark programs to evaluate and compare OTF with SSP. We select 3 programs from Splash2 (fft, cholesky and raytrace) and 2 from parsec (blackscholes and bodytrack) for this evaluation as shown in Figure 18. It can be observed that the global average delay values for OTF are much lower than SSP for all programs.

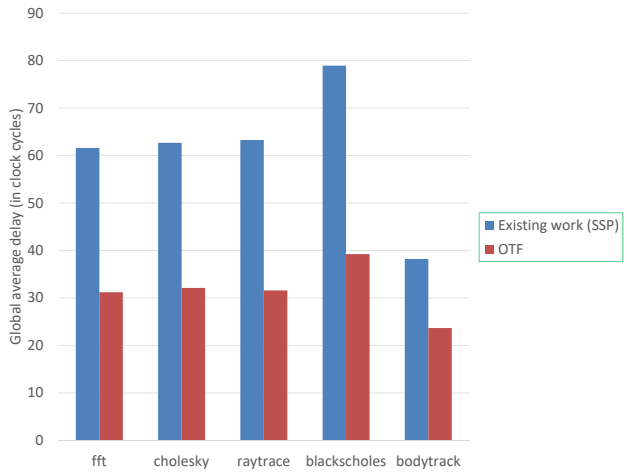


Fig. 18. Global average delay values (in clock cycles) for different benchmark programs.

*5.3.2 Hardware resource overhead.* Synopsis Design Compiler is used for synthesis purposes using 22 nm Globalfoundries fully depleted Silicon-on-Insulator (FD-SOI) technology library. Area and power results for different routers are given in Table 1 at a frequency of 770 MHz which is the maximum frequency of operation. The router areas for OTF, DITF, RTF, ROTF and RDITF are 74.29%, 74.13%, 65%, 64.21%, and 64.15% lower, respectively, compared to the router in [17]. When comparing to [6], most of our current solutions (OTF, DITF, and RTF) require less area overhead and ROTF and RDITF require almost similar area. The power results also show similar trends. This shows that our proposed techniques require less hardware overhead compared to the existing works. Among different proposed routers, RTF area is larger than OTF and DITF because of the additional packet storage module. The area of ROTF and RDITF are larger than other proposed routers because they are a combination of both RTF and OTF/DITF. We have also mentioned the non-secure router area in Table 1 which lacks any IP protection mechanism in it. The router area increments of OTF, DITF, RTF, ROTF, RDITF, and the work in [17] compared to the non-secure router are 2.45%, 3.08%, 39.47%, 42.59%, 42.84%, and 298.46%, respectively. The security implementation in [17] requires an extra router output stage (area increment) which is absent in the non-secure router.

Note that while the ROTF and RDITF methods require larger area compared to OTF, DITF and RTF methods, they also provide higher fingerprinting channel capacity. Also, the embedded fingerprint extraction time required is lowest for the RTF method among all proposed methods because the inter-packet delay is fixed to 1 clock cycle. In all other cases, the inter-packet delay is modulated and hence the extraction time of all packets will be higher. Thus, a designer needs to consider these trade-offs between different proposed techniques before choosing a particular security solution.

We also provide the main disadvantages of existing methods and comparative advantages of our proposed solution in Table 2.

## 6 CONCLUSION

NoC IP stealing attack is a serious security threat. In this paper, we have proposed a timing channel based fingerprinting method and have described five different methods (OTF, DITF, RTF, ROTF, and RDITF) to show the effectiveness of timing channel based fingerprinting. We have also provided

Table 2. Comparison with existing solutions.

| | Main disadvantage of existing solutions | Advantage of our solution |
|---|---|---|
| Square spiral path (SSP) [17] | This solution doesn't provide cryptographic security, and requires larger area and performance overhead. | Our solution provides cryptographic security, requires less area, and provides better performance. |
| Circular path based fingerprinting (CPF) [6] | This solution requires larger area overhead. | Our solution requires less area. |
| Constraint based watermarking [14, 15] | This solution may produce a design with degraded performance. | Our solution does not have any impact on the design performance. |
| Watermark generating circuit (WGC) [13] | WGC is a separate entity from the design and can be removed or disabled by an adversary. | Our solution is not a completely separate entity and it is present throughout the NoC. |

a formal proof of security for our proposed method. We have shown that all of our proposed techniques require much less hardware overhead compared to existing NoC IP security solutions and also provide better security. It is shown that our proposed techniques require between 64% to 74% less router area compared to an existing solution. We have also shown that our solutions do not affect the normal packet latency and hence do not degrade the NoC performance.

## ACKNOWLEDGMENTS

This research is supported by the National Research Foundation, Singapore, under grant NRF2018NCR-NCR002-0001.

## REFERENCES

[1] Michel Abdalla, Pierre-Alain Fouque, and David Pointcheval. 2005. Password-Based Authenticated Key Exchange in the Three-Party Setting. In *Proceedings of the 8th International Conference on Theory and Practice in Public Key Cryptography* (Les Diablerets, Switzerland). Springer-Verlag, 65—-84.

[2] AES 2001. Announcing the Advanced Encryption Standard (AES).

[3] Toshihiro KATASHITA Akashi SATOH. 2013. ELECTRONIC CIRCUIT COMPONENT AUTHENTICITY DETERMINATION METHOD.

[4] Mauro Barni, Franco Bartolini, Alessia De Rosa, and Alessandro Piva. 1999. Capacity of the watermark channel: how many bits can be hidden within a digital image?. In *Proc.SPIE*, Vol. 3657. 3657 – 3657.

[5] Georg T. Becker, Marc Fyrbiak, and Christian Kison. 2017. *Hardware Obfuscation: Techniques and Open Challenges.* Springer International Publishing, 105–123.

[6] Arnab Kumar Biswas. 2020. Network-on-Chip Intellectual Property Protection Using Circular Path–Based Fingerprinting. *J. Emerg. Technol. Comput. Syst.* 17, 1, Article 4 (Sept. 2020), 22 pages.

[7] Arnab Kumar Biswas, Dipak Ghosal, and Shishir Nagaraja. 2017. A Survey of Timing Channels and Countermeasures. *ACM Comput. Surv.* 50, 1, Article 6 (March 2017), 39 pages.

[8] A. E. Caldwell, Hyun-Jin Choi, A. B. Kahng, S. Mantik, M. Potkonjak, Gang Qu, and J. L. Wong. 2004. Effective iterative techniques for fingerprinting design IP. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 23, 2 (Feb 2004), 208–215.

[9] Vincenzo Catania, Andrea Mineo, Salvatore Monteleone, Maurizio Palesi, and Davide Patti. 2017. Improving Energy Efficiency in Wireless Network-on-Chip Architectures. *J. Emerg. Technol. Comput. Syst.* 14, 1, Article 9 (2017).

[10] C. H. Chang and L. Zhang. 2014. A Blind Dynamic Fingerprinting Technique for Sequential Circuit Intellectual Property Protection. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 33, 1 (Jan 2014), 76–89.

[11] World Semiconductor Council. 2018. Winning the Battle Against Counterfeit Semiconductor Products. White Paper.

[12] Juergen Teich Daniel Ziener. 2007. WATERMARKING APPARATUS, SOFTWARE ENABLING AN IMPLEMENTATION OF AN ELECTRONIC CIRCUIT COMPRISING A WATERMARK, METHOD FOR DETECTING A WATERMARK AND APPARATUS FOR DETECTING A WATERMARK.

[13] Y. C. Fan. 2008. Testing-Based Watermarking Techniques for Intellectual-Property Identification in SOC Design. *IEEE Transactions on Instrumentation and Measurement* 57, 3 (March 2008), 467–479.

[14] A. B. Kahng, J. Lach, W. H. Mangione-Smith, S. Mantik, I. L. Markov, M. Potkonjak, P. Tucker, H. Wang, and G. Wolfe. 2001. Constraint-based watermarking techniques for design IP protection. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 20, 10 (Oct 2001), 1236–1252.

[15] Andrew B. Kahng, Stefanus Mantik, Igor L. Markov, Miodrag Potkonjak, Paul Tucker, Huijuan Wang, and Gregory Wolfe. 1998. Robust IP Watermarking Methodologies for Physical Design. In *Proceedings of the 35th Annual Design Automation Conference* (San Francisco, California, USA) *(DAC '98)*. ACM, New York, NY, USA, 782–787.

[16] Bao Liu and Gang Qu. 2016. VLSI supply chain security risks and mitigation techniques: A survey. *Integration, the VLSI Journal* 55 (2016), 438 – 448.

[17] Q. Liu, W. Ji, Q. Chen, and T. Mak. 2016. IP Protection of Mesh NoCs Using Square Spiral Routing. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 24, 4 (April 2016), 1560–1573.

[18] MarketsandMarkets 2019. Semiconductor Intellectual Property (IP) Market by Design IP (Processor IP, Interface IP, Memory IP), IP Source (Royalty and Licensing), Vertical (Consumer Electronics, Telecom, Industrial, Automotive, Commercial), and Region - Global Forecast to 2024. https://www.marketsandmarkets.com/Market-Reports/semiconductor-silicon-intellectual-property-ip-market-651.html

[19] Gang Qu. 2001. Keyless Public Watermarking for Intellectual Property Authentication. In *Information Hiding*, Ira S. Moskowitz (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 96–111.

[20] Ryo Sugihara. 2001. Practical Capacity of Digital Watermarks. In *Information Hiding*, Ira S. Moskowitz (Ed.). Springer Berlin Heidelberg, 316–330.

[21] F. Worm, P. Ienne, P. Thiran, and G. de micheli. 2002. An adaptive low-power transmission scheme for on-chip networks. In *15th International Symposium on System Synthesis, 2002.* 92–100.