Hierarchical Group-key Management for NoC-Based MPSoCs Protection

Johanna Sepulveda[1], Daniel Flórez[2], Vincent Immler[3], Guy Gogniat[2], Georg Sigl[1,3]

[1]Institute for Security in Information Technology, Technical University of Munich, Germany
[2]Lab-STICC, South Brittany University, France
[3] Fraunhofer Research Institution for Applied and Integrated Security (AISEC), Germany
johanna.sepulveda@tum.de

ABSTRACT

Group keys can be used in order to communicate secretly sensitive data among IP cores. However, the flexibility and dynamic nature of MPSoCs force reshaping the security zones at runtime. Members of a zone must be able to efficiently compute the new group key while former members must be prevented for data disclosure. Efficiently creating security zones for achieving sensitive traffic isolation in MPSoC environments is a challenging problem. In this work we present the implementation of hierarchical group-key management for NoC-based systems in order to efficiently perform the rekeying process. We implement three hierarchical protocols and we show that by decentralizing the security management of the rekeying process, it is possible to achieve an improvement of the performance when compared to the previous flat approaches.

Index Terms: MPSoCs, Security, Group-key, NoC

I. INTRODUCTION

Multi-Processors Systems on Chip (MPSoCs) integrate several computing and storing Intellectual Property (IP) cores, which exchange data through a Network-on-Chip (NoC). Their flexibility and computation power, have turned the MPSoC the key technology enabler of the new computation paradigm Internet-of-Things (IoT). As a result MPSoCs are not isolated anymore, but belong to a distributed computed network of devices interconnected through Internet. MPSoCs operating in the context of IoT promise to be a source of great benefits but this demands new requirements for their effective implementation. Among them, security has emerged as new and critical dimension of embedded system design. Flexibility in MPSoCs is expressed by their capability of supporting the execution of different applications downloaded to the device at runtime. Applications may require different security services which must be guaranteed by the system. High resource constraints and tight performance requirements make the security integration at MPSoCs challenging.

One of the most frequently used techniques to increase the MPSoC performance is to divide the applications into tasks and spread them among the computing IPs [1]. When applications are sensitive, the splitting strategy forces the sensitive data to exchange through the shared and insecure NoC. By infecting IP component, malicious applications may compromise the security of the system. Attacks at MPSoCs are able to extract sensitive information, modify the system behavior, or deny its operation.

Security-enhanced NoCs have appeared as an effective alternative to aid in the overall MPSoC protection. Sensitive data exchanged through the NoC can be isolated by means of security zones [2,3]. The main goal is to create an envelope around the sensitive IPs' communications. IPs that belong to the same security zone are considered trusted among each other. The size and the shape of the security zones are determined by the mapping of the sensitive application on the MPSoC. When such a mapping varies at runtime, dynamic security zones must be implemented.

Previous works propose the implementation of security zones by firewall-based traffic inspection (ACNoC) [2]. NoC traffic is filtered according to the security policy, able to be upgraded during runtime. Firewalls build a physical barrier that isolate sensitive traffic from possible malicious communications. Despite the good results, firewall-based approaches present two drawbacks: i) lack of plasticity, where the IP members of the security zones are located physically close; and ii) communication degradation, by avoid-

ing that communications outside the security zone take place inside the zone, firewalls may turn IPs unreachable. In order to overcome such difficulties, in our previous work [3] we proposed the implementation of group-key protocols for creating the security zones at the MPSoC. A secret shared among the IP members of the security zones can be established in order to exchange data securely. Our approach adopts a hybrid encryption technique that uses asymmetric cryptography for sharing the common secret and symmetric cryptography to encode the sensitive data. Only members of the security zone, and owners of the shared secret, are able to decrypt the information. The work implements two protocols: i) mapping-aware key pre-distribution scheme, based on pools of keys predefined at design time; and ii) Diffie-Hellman, a contributory approach that quantifies the shared secret as a function of the secrets of the members of the zone. Despite the high data protection, these approaches still present lack of scalability and efficiency.

In order to overcome these drawbacks, in this work we propose the implementation of a hierarchical group-key NoC-based architecture able to scale and efficiently create security zones at the MPSoC. By decentralizing the security management of the group-key protocols, our hierarchical approach improves the system's performance and scalability, while effectively handling the security policy changes. The architecture includes a global and several local managers distributed along the system, supporting several layered protocols. In this work we implement three hierarchical protocols: i) hierarchical mapping-aware pre-distribution (HMAPD), where the members of the zone must discover a group key within the own key pool; ii) Hierarchical Diffie-Hellman (HDH), where the iterative Diffie-Hellman approach is used to find a group Key; and iii) Hierarchical Tree-based Diffie-Hellman (HTDH), where by using the pair-wise keys among the members of the zone, the local manager is able to compute and distribute a group key. We test our architecture and compare the results with the previous firewall-based dynamic (ACNoC) [2] and the centralized flat approaches [3]. Each technique has a computation and communication overhead in the MPSoC. We show that our hierarchical technique is able to enhance the group key agreement performance up to 52% and that it is suitable for highly dynamic security zones.

The novelties of our work are:

- Hierarchical NoC-based group communicati on for efficient management of security zones.
- Implementation of two tree-based group-key management techniques
- Exploration of the performance, area and pow er trade-off of tree-based key management techniques.

This paper is divided into six sections. Section 2 presents the previous works in the area of NoC-based security in MPSoCs. Section 3 describes the Diffie-Hellman and mapping aware pre-distribution group-wise security protocols. Section 4 presents our architecture and its operation. Section 5 reports the experimental results. Finally, conclusions are presented in Section 6.

II. RELATED WORK

Security enhanced NoCs have been used to protect the MPSoC against software-based attacks [4-10]. Such approaches demonstrate that secure NoCs can be an effective solution to protect the MPSoC against attacks whose purpose is data modification [4-6], denial of service [7,8] and data extraction [9]. Firewall-based techniques are the most popular protection approaches at NoCs [4-7]. By matching the packet content with the security rules stored on the firewall, communications are allowed or blocked. NoC firewalls are used to implement access control [4-7], authentication [7] and integrity [2].

The security tables of the firewalls are able to change, allowing the implementation of dynamic security services. Security zones appeared as an attractive isolation technique, thus enabling the secure communication among the IP members of the zone [2,3]. In the work of [2] firewalls among the routers are employed to restrict the traffic around the security zone. By modifying the security tables, security zones can contract and expand. Despite the good results, this technique only protects continuous security zones, where all the IP members are closely mapped (physical location), and penalizes the network connectivity, thus degrading the performance. Group wise protocols can be used to implement security zones [3]. Fig. 1 shows the taxonomy of the group wise protocols. The work of [3] presents a NoC enhanced architecture able to support group-wise Mapping-aware key pre-distribution (static) and Diffie-Hellman (dynamic-central-flat) protocols. Despite the good results, these approaches present two

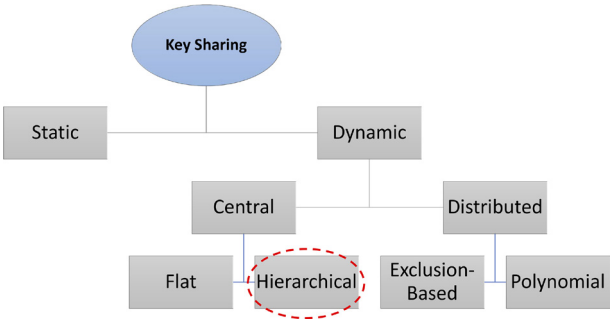

Figure 1. Taxonomy of the group-key sharing approaches.

drawbacks: i) performance highly depends on the size (amount of members) of the security zones. For highly dynamic systems, the rekeying process based on these techniques can be prohibitive; and ii) the existence of a central manager creates a communication hotspot in the system and turns it difficult to manage the security. In order to overcome these difficulties we propose in this work a hierarchical approach able to decentralize the security management and enhance the performance.

III. SECURITY PRELIMINARIES

This Section presents the description of the MPSoC security issues. The first subsection presents the MPSoC, targets of protection, attacker capabilities and assumptions regarding the attack and the system. The second subsection presents the security zone concept and their effect in the system security.

A. Threat model

MPSoCs are able to support several applications which may change during runtime. In order to increase the performance of the system, applications are split into tasks and spread on the MPSoC resources. Such a technique forces the peer interaction among the IP cores. Consequently, for MPSoCs that support critical information, sensitive data is exchanged among the different computation components through the shared NoC, thereby opening opportunities to attackers. The NoC route used to exchange sensitive information between two IP cores is known as sensitive path. Fig. 1 shows two sensitive paths from the source IP_{15} to the destination IP_2 and IP_3, composed by 5 and 4 routers, respectively.

In order to attack the system, the IP cores and NoC can be exploited. We consider that the attacker can infect the IPs and that the NoC contains Trojans. Code injection can be employed to install malwares in the IPs of the MPSoC, letting the attacker to manipulate the traffic injection of the infected IP. The content and destination of the malicious packets is controlled by the attacker. Malicious IPs are used to create packets able to activate the Trojans of the NoC as in [10]. The malicious NoC is then able to deviate sensitive traffic, spying and modify the sensitive packets.

The network interfaces, which are the components that perform the interconnection between the NoC routers and the IPs, are considered secures. This assumption is valid, once the interfaces are usually built in house [10]. In order to perform the attack, the following preconditions are required:

- The mapping of the sensitive cores is known.
- The NoC routing algorithm is known.
- Attacker is able to infect an IP of the MPSoC.
- Attacker can control the traffic generation of the infected IP.
- The NoC integrates a Trojan able to be trigge red by malicious packets.
- Malicious router (router with the Trojan) is inside the sensitive path.

We consider that sensitive (S) and malicious (M) processes are executed simultaneously at the sensitive IP cores (IP_{15}, IP_2, IP_3) and the infected IP cores (IP_{11}, IP_8), respectively. Note that the malicious IPs are used to trigger the NoC Trojans and turn the routers malicious, may be directly linked to the malicious router (R_{11}) or in any other location (R_7).

As protection mechanism we consider that secure network interfaces are able to provide confidentiality and integrity security services by means of cryptographic techniques. By encrypting the data, illegal modifications and data disclosure are avoided. In order to provide such services, a secret shared among the sensitive IPs is required.

B. MPSoC and security zones

Suppose an MPSoC composed of a set of IP cores which exchange data through the NoC composed of routers (R). Where $IP=\{IP_1,IP_2,\ldots,IP_n\}$ and $R=\{R_1,R_2,\ldots,R_n\}$. The link between R and IP cores are the network interfaces $NI=\{NI_1,NI_2,\ldots,NI_n\}$. NoCs should allow the exchange of sensitive and non-sensitive traffic inside the MPSoC while guaranteeing the data protection.

A sensitive application S which requires confidential data exchange is mapped on m IP_I such that IP_I IP, where $IP_I=\{ip_1,ip_2,\ldots,ip_m\}$. The sensitive path is composed by a subset of R. In order to protect S, a security zone can be created. A security zone Z is a physical or virtual space that integrates the IP_I and whose purpose is to isolate their communication from set $IP\text{-}IP_I$. The members of the security zone IP_I are considered trusted among each other. According to the shape of the security zones, it can be continuous, where IP_I are located physically adjacent, otherwise, it is called disrupted zone. Fig. 2 shows an MPSoC composed of 16 IP ($n=16$) interconnected by a 16-router mesh-based NoC. In order to increase the performance of the MPsoC, a single application is divided into tasks and mapped on different IPs of the MPSoC. The sensitive application S is mapped onto the $IP_I=\{ip_2, ip_3, ip_{15}\}$, thus forcing the communication of these IPs through the NoC. The sensitive communication uses two different paths inside the shared NoC. Different paths are the result of the routing algorithm implemented at each NoC router. In order to protect the sensitive data, a security zone of size 3 can be built around IP_I. The security zone of the example is disrupted, due the physical location of the IP_I.

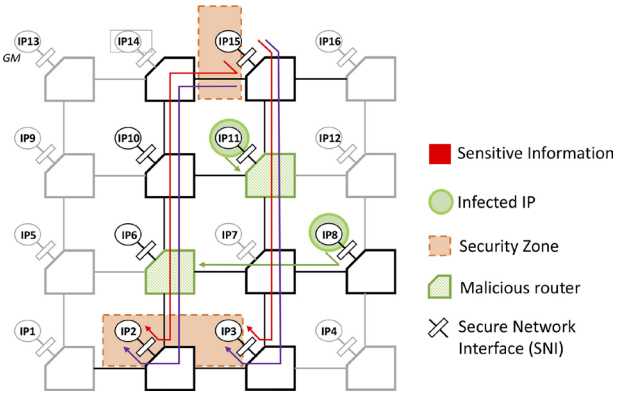

Figure 2. Security zones at MPSoCs.

Security zones are not static. Their number of members and their shape vary according to the runtime security requirements of the system. A security zone must be modified in one of the 3 scenarios: i) a new application is mapped on the system; ii) a task is migrated between IPs on the MPSoC; and iii) under special operation conditions of the MPSoC, for example if being under an attack.

The modification of the security zones is performed via three operations:

i) **Creation**, establishes the initial security zone composed by IP_I by computing the initial shared secret;

ii) **Modification**, changes the members of an existing zone. IP_I can be reduced (member leave) or increased (member join). As a result, it is possible that security zones are merged, in the case that the IP already has a membership of another zone. It requires that the secret key is recomputed; and

iii) **Elimination**, erases the shared secret among the members of the zone.

During these operations, the sensitive communication is blocked. As the security zones may change during runtime, it is desirable that the security zones operations are performed efficiently. The impact of the security must be kept low in order to allow that the final MPSoC satisfies the performance requirements of the different applications.

IV. Defining security zones using group key management

Group key management techniques can be used to implement continuous and disrupted security zones. By establishing a secret group key among the IP members of the security zone (IP_I), NoCs can isolate sensitive traffic and prevent data disclosure. Communication among the IP members of the security zone is performed in an encrypted way through the secret group key. Only IP_I members will be able to retrieve the plaintext information. Note that a single IP may belong to several security zones, thus it might integrate different group keys. The group-wise key management technique requires that NoCs also integrate authentication security in order that nodes can verify the authenticity of the source. Authentication services can be provided by the NoC as shown by [2,7].

Key management includes the set of procedures that support key distribution and maintenance of keying relationships between authorized IPs. Three stages can be identified in the group-wise security protocols: i) *start of security zone*, which sets up the parameters required for the initialization of the protocols; ii) *group key agreement*, to establish the secrecy of the group; and iii) *resume operation*, to allow the normal system operation. The key management can be performed at MPSoCs by centralized or by distributed techniques. Each alternative presents a trade-off between the security and the performance of the system. Centralized mechanisms have a root of trust but decrease the system performance due the rekeying process [11]. Totally distributed techniques decrease the impact on key management and increase the flexibility of the security zones. However, the security can be compromised due the possible attacks on infected IPs.

Hierarchical approaches arise as an alternative that combine the advantages of the central and distributed approaches to manage the complexity of the keying process. It ensures the existence of a root of trust (global security manager *GM*) while offering flexibility and high performance (local security manager ip_Z). A hierarchical key management scheme is composed by a GM and a set of distributed ip_Z. GM selects per security zone Z an IP member that is going to be the ip_Z, responsible for controlling and operating Z. Each IP of the MPSoC includes a secret key only known by the GM. It makes possible that GM can send data confidentially with each IP. Thus, GM sends encrypted to ip_Z, the address of the members of the security zone and a time-stamped certificate for creating the security zone. Such information will be used to securely creating and operating a security zone. The processing and communication behavior of the MPSoC components will depend on the adopted group-wise protocol. The security is locally controlled by ip_Z and globally by GM. Next subsections will present tree hierarchical protocols able to be adopted in MPSoC environments in order to create dynamic security zones. The reuse of well-known protocols from macro networks, such as Internet, is always a good practice. These robust protocols have been studied and developed during the last years, encapsulating good design practices as defined in [12]. As a result, designers avoid pitfalls which can be exploited by attackers.

A. Hierarchical Mapping-aware pre-distribution (HMAPD)

Mapping-aware pre-distribution (HMPAD) protocol is a pool-based technique which uses key pre-distribution and post-deployment key establishment. It ensures that two communicating IPs share at least one common key which will be computed and used for secure data communication. HMAPD is based on a pool of |P| keys and their identifiers. From this pool K pseudo-randomly selected keys are distributed on each IP_X, in order to guarantee that neighboring IPs have a higher probability to have more keys in common. Fig. 3 presents an example of the HMAPD key scheme for a 16-core MPSoC that creates a security zone among IP_{15}, IP_3 and IP_2. Each core is pre-loaded with a set of 4 keys (pool of keys). GM selects IP_{15} as the ip_Z, which is responsible of starting the zone creation. IPs which are expected to communicate have a high probability to share a common key (K_a) which will be a candidate as the group key (K_g). In case a common key is not found, ip_Z will be in charge of select and distribute the K_g. The ip_Z communicates securely to all the IP by means of public cryptography.

HMAPD forces the IPs of a security zone to undergo a discovery process for setting up shared keys. Such a secret allows establishing a secure communication. A basic scheme of key pre-distribution was proposed by [13]. The *group key agreement* is established via two steps:

i) ***Group key establishment***, whose goal is to find a common key among the IPs of the security zone. The ip_Z activates the IP_I in order to broadcast the list as in (1), where α is a challenge. If a pair of IPs shares a common key, the proper decryption of $EK_n(\alpha)$ at the destination IP will reveal the challenge α. Thus, the destination IP will be able to correctly answer the challenge, establishing a partial shared key K_p with the source IP.

$$\{\alpha, EK_n(\alpha), n=1,\ldots,K)\} \quad (1)$$

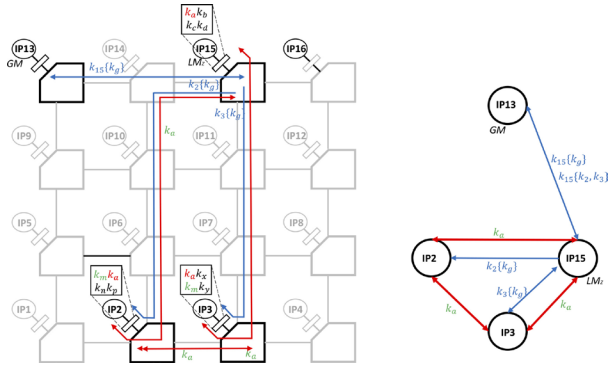

Figure 3. Mapping-aware pre-distribution (MAPD) key scheme.

ii) ***Group key propagation*** is performed when not all IPs of the security zone shared a common key K_p. After the group key discovery, if the IP_I do not share a common secret, the group can establish a secret through ip_Z. By selecting a key K_g of the pool of ip_Z and sharing it with the IP_Is by using the public cryptography, a group secret channel can be established.

At the end of the HMAPD scheme, all the IP members of the security zone will share the same K_g which allows the establishment of a secure communication. The encryption and decryption of the data can be performed by means of lightweight cryptographic techniques. Examples of such cyphers are PRESENT [14], HIGHT [15] and mCrypton [16].

B. Hierarchical Diffie-Hellman (HDH)

Diffie-Hellman (DH) is a contributory key agreement protocol proposed more than 40 years ago by [11]. During all this time, several studies have proven its security and tamper resistance property. DH defines a primitive root, also called generator g and a prime modulo p, which are public data known by all the IP_X. Hierarchical Diffie-Hellman (HDH) implements the DH protocol by means of ip_Z. GM sends the IP_I information according to the iteration relation among the IP_Is. The ip_Z activates each IP_I and controls the zone operation locally. The *group key agreement* is performed in three steps:

i) *Computation of the public partial key* c_i: Each IP_I defines a private and secret number s_i. IP_I can compute its contribution c_i (partial key), as in (2).

$$c_i = g^{si} \bmod p \quad (2)$$

ii) *Broadcast of the* c_i: The partial key c_i is then turned public and broadcasted among all the IP_X by means of the unprotected NoC.

iii) *Computation of the group secret key* K_g: Each IP_m, where $m \in I$, receives the set of c_i messages such that $i \neq m$. The group key K_g is computed as (3).

$$K_g = g^{Ci.Sm} \bmod p \quad (3)$$

At the end, the same K_g is obtained at each IP_I once the flipping of the exponent does not change the result. Note that s_i is always kept safe inside the IP_I and no secret is transmitted in order to quantify K_g. Even if an attacker intercepts and collects all the c_i messages, it is impossible to derive K_g, once the attacker does not have the private number s_i. The recovery of K_g, by the c_i messages is known for being a Discrete Logarithm Problem. So far, no efficient algorithm has been designed to break the Diffie-Hellman key exchange. Once the secret group key K_g is established, it is used for the members of the group to encode the data D_i of each IP_I. Fig. 4 shows

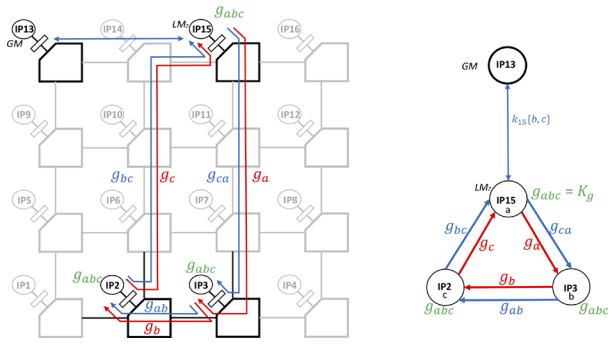

Figure 4. Hierarchical Diffie-Hellman (HDH) key scheme.

the employment of the HDH to establish the group key among IP_{15}, IP_3 and IP_2. The process is started when *GM* linked at IP_{13} selects IP_{15} as ip_Z. It activates IP_3 and IP_2 and controls the two rounds required to determine K_g for a zone of three members. The number of rounds will depend on the size of the zone.

C. Hierarchical Tree-based Diffie-Hellman (HTDH)

A tree-based Group Diffie-Hellman uses a hierarchical pair-wise DH approach to efficiently establish a group key. It organizes the members of the security zone in a binary tree structure *T* of nodes characterized by the pair (v,l). Where, v is the level and l identifier of each node in *T*. Each node has a secret key $K_{v,l}$ and a blind key $BK_{v,l}$ such that $BK_{v,l} = g^{Kv,l}$. Therefore, $K_{0,0}$ is the root of the tree, that is the ip_Z.

The structure of the zone is determined by the global manager *GM*, but organized by the ip_Z, as shown in Fig. 5. In order to create a security zone, the GM selects an ip_Z and sends the information about the IP_I members of the security zone together with the tree structure T. This approach allows the establishment of a hierarchy of keys, where each group member is secretly related with a key in the bottom of the hierarchy.

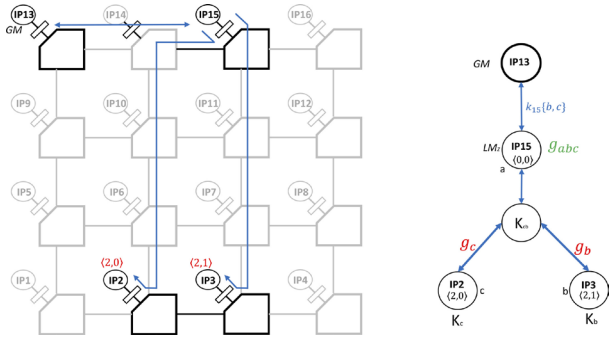

Figure 5. Hierarchical Tree-based Diffie-Hellman (HTDH) key scheme.

Each interior key is encrypted with all of its children keys and these encoded messages are broadcasted to the members of the security zone. Each IP can decrypt the keys along the path from the leaf to the root. The root key is used as K_g. Each modification of the security zone implies the modification of the key of the members linked to the branch of the tree from the modified leaf to the root. This characteristic turns the rekeying process efficient. The *group key agreement* is performed in two steps:

i) *Creation of pair-wise keys*: For each pair of members IP_a and IP_b, where a, b ϵ I, a pair-wise key K_{ab} is stablish by means of DH, as in (4).

$$K_{ab} = g^{KaKb} \quad (4)$$

ii) *Broadcasting of contribution:* Pair-wise keys are propagated through the tree. Each member of the zone is able to quantify the K_g.

In contrast to the HDH, each time a member is added or evicted, the key of the members linked to the branch of the modified leaf to the root is changed. Then the new K_g is quantified by ip_Z and broadcasted secretly for the remaining members of the zone. This technique avoids the inclusion of all the members in the rekeying process, saving several rounds of computation and communication resources.

V. SECURITY ARCHITECTURE

This Section presents the proposed architecture for supporting the hierarchical group key distribution in the NoC-based MPSoCs. The first subsection describes the three main components of the architecture: NoC, secure network interfaces and global manager. The second subsection describes the functionality of the proposed architecture.

A. Components

In this work we propose a NoC-based architecture able to support the hierarchical group key distribution schemes for MPSoC protection discussed in the Section 4: i) hierarchical mapping-aware pre-distribution (HMAPD); ii) Hierarchical Diffie-Hellman (HDH); and iii) Hierarchical Tree-based Diffie-Hellman (HTDH). Fig. 1 presents the proposed architecture. It is composed of three main components: i) NoC, which integrates the *routers* employed to exchange data required for the execution of the application and the establishment of the group key; ii) secure network interfaces (SNI) to implement the communication and security protocols. They also include a small FSM that allows each node to behave as local manager ip_Z and control the security zone configuration; and iii) global

manager (GM), which controls the configuration of the security parameters required to configure ip_Z and implement the security protocol.

NoC routers integrate five main components: i) input buffers, which store the data that request the router by one of its input ports; ii) routing algorithm, which selects the router output port to be employed for redirecting the incoming data; iii) arbitration logic, that grants the utilization of the crossbar switch to one of the input buffers; iv) crossbar switch, which links input to output ports of the router and v) mark control, which modifies the field of the packet used for authentication purposes. SNIs, shown in Fig. 2, are employed for implementing the communication and the security protocols. For the *communication protocol,* the SNI transforms the data into packets. This function is developed by the packing/unpacking module. Each packet is composed of five fields: i) *source*, to identify the IP which starts the communication; ii) destination, to identify the target IP; iii) operation, to identify the type of packet (control, data write, data read); iv) payload, that corresponds to the data generated by the IP *source;* and v) *path*, intended for authentication purposes by storing the marks of the router.

For the *security protocol*, the SNI performs three security services: *authentication*, *access control* and *confidentiality*. Authentication ensures the source integrity by matching the *source* field of the packet. Access control verifies that the transaction is authorized. Confidentiality ensures the secrecy of the information. Authentication and access control are implemented through the *security table* component, which stores the *source*, *destination*, *operation* and *path* contents that authorized transactions must satisfy [2,7,17]. Each time a packet arrives to the SNI, the control extracts the packet information for searching the content stored at the *security table*. When a match is produced, the transaction is allowed. Otherwise it is denied and the GM is notified.

Confidentiality service is implemented by three components: i) *Key_comp*, whose purpose is to calculate the secret group key at each IP; ii) *coding/decoding,* in charge of encrypting and decrypting the data with the group key K_g for transactions that belong to the security zone; and iii) *Local_man*, which stores the address of the IP members of the security zone and the security certificate. It is in charge of building the packets required to create and operate the security zone.

While *coding/decoding* and *Local_man* modules remain identical for all protocols employed for defining the group key, *Key_comp* architecture varies. For HMAPD, *Key_comp* calculates the key according to (1). *Key_comp* should store two pieces of information: i) pool of random keys *K* and their identifiers; ii) partial shared keys *kp* or *kr* and group key K_g. The computation of the *kp*. For HDH the key is calculated as (2) and (3) and for HTDH as (4).

Three types of information are stored at the *Key_comp*: i) public values g and p; *ii)* private number *s*; and iii) partial or pair-wise and group keys *ci,* K_{ab} and K_g.

The global manager (GM) is a light software task embedded into a secure processor [3]. It guarantees the implementation of the security policy of the system by controlling the security mechanism at the secure interface. The GM compiles the security requirements of the software tasks executed on the MPSoC and transforms them into a set of security rules. That is, the content that should be stored at the *security table* of the SI for the *authentication* and *access control* security services, as shown in our previous work [7]. For the confidentiality security service, GM determines the creation, modification and elimination of security zones at the MPSoC. It selects the ip_Z based on pre-defined selection (e.g. trust level, attacks exposure, accessibility) or security history of the module (e.g. number of attacks detected). Thus, GM is able to send securely to ip_Z the members' address of the security zone and a certificate (timed-stamped packet with the signature of GM as in [17, 18]). This technique guarantees that security zones are created by authorized ip_Z and avoids replay attacks [4]. The scope of this work is the hardware architecture, therefore the compilation process of GM is not addressed in this paper.

B. Functionality

Security zones are established at the MPSoC through the confidentiality service. A group-key K_g is established among the sensitive IPs, allowing the secure sensitive data exchange. Security zone key K_g can be obtained through one of the three hierarchical schemes discussed on Section 4. Fig. 6 shows the communication sequence for K_g negotiation among IP_a, IP_b and IP_d by means of HMAPD, HDH and HTDH protocols. Note that IP_c does not belong to the security zone. The example shows a portion of the MPSoC constituted by five IPs (*GM*, IP_a, IP_b, IP_c and IP_d), where IP_a is the ip_Z. Each IP is linked by the NoC interfaces (*GM*, SI_a, SI_b, SI_c and SI_d). The three stages of the protocol (*start of security zone*, *group key agreement* and *resume operation*) are identified.

The *start of the security zone* (activated for creation, modification or elimination of a zone), begins when GM selects the ip_Z and sets up the protocol parameters (address of IP members, tree structure for HTDH and certificate). The ip_Z then activates the members of the zone. It includes the events 1A, 1B and 1C of Fig. 6. As a result the events 2A (which wrap as packets the key pool and challenge data), 2B and 2C (which process the partial key c_i, called contribution) and are triggered, respectively.

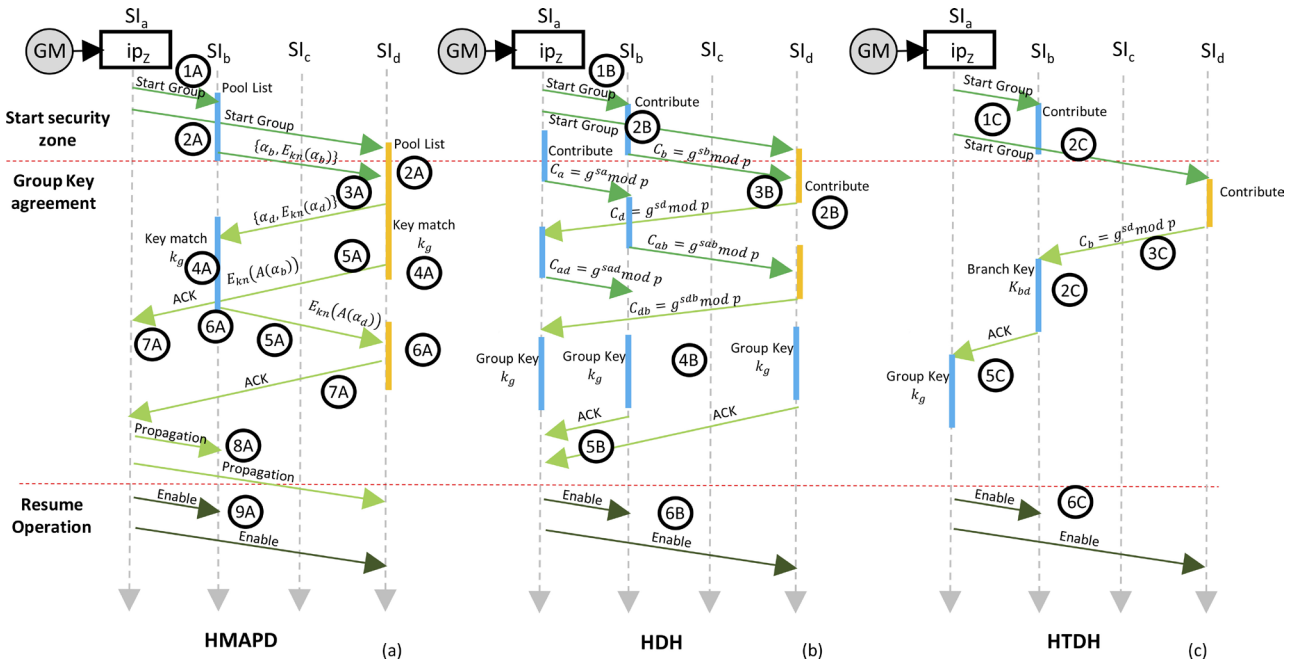

Figure 6. Hierarchical communication protocols.

The *group key agreement* is constituted by the events described at Sections 4.1, 4.2 and 4.3. For the IPs under HMAPD, the list of the key pool and the challenges are broadcasted (3A) in order to match the key (4A), answering the challenges of the members (5A) of the zone and testing the answers (6A). The matching is notified to the ip_Z (7A). If there is an IP without a match, ip_Z propagates the key (8A).

For the IPs under HDH, they calculate contribution c_i (2B), broadcast c_i (3B) and compute the partial keys after each round. The figure shows two rounds, required for 3 participants of HDH. When all the keys are received K_g is calculated (4B). When the computation is finished, the members send an acknowledgment signal to ip_Z (5B).

Finally, for the HTDH, the contribution of the right-child IP is quantified (2C) and sent to the left-child of the binary tree (3C). The right child quantifies the branch key (4C). Finally the contribution of the branch is sent to ip_Z (5C) in order to quantify K_g. This key is the broadcasted in the tree.

The *resume operation*, will release the communications (9A, 6B and 6C). All the transactions involved during the key agreement are authenticated by using the source and mark field of the packet.

6. EXPERIMENTAL WORK

This Section presents the characteristics of the conducted experiments, together with the security, performance and cost evaluation.

A. Experimental setup

The NoC-based group-wise security architecture has been modelled in SystemC-TLM and RTL-VHDL [3]. SHOC is a modular cycle accurate simulation environment which supports a wide variety of Instruction Set Architectures, traffic generators and all the components required for MPSoC simulation. This environment includes libraries of MPSoC attacks and tools for power and area estimation. All components are synthesized for 65nm technology at 500 MHz operating frequency and 25°C, by means of the Cadence Encounter RTL Compiler RC12.24 (v12.20-s034) tool. By integrating the model of the architecture proposed in Section 5, we model an 81-cores MPSoC that is interconnected though a 9x9 mesh-based NoC. For comparison reasons, we have implemented the dynamic elastic security zone architecture ACNoC of [2] and the flat group-wise protocols proposed in [3].

B. Security evaluation

Security of our solution has been performed under the seven attack scenarios described in Table 1. The results are expressed as the percentage of attacks detected by each configuration. It shows that all configurations that implement the group-wise protocols (flat and hierarchical) always protect the system. The succeeded attacks at the ACNoC were executed on the IPs that cannot be protected by the continuous zone because of their physical distance.

Table 1. Attack description and efficacy

Attack scenario	*ACNoC*	*Flat group-wise*		*Hierarchical group-wise*		
		MappingPre-distribution MP	*Diffie-Hellman DH*	*HMAPD*	*HDH*	*HTDH*
Overwrite memory data to modify the system behavior by corrupting memories	80%	100%	100%	100%	100%	100%
Read not allowed memory data	70%	100%	100%	100%	100%	100%
Inject packets without any valid destination to degrade system performance	100%	100%	100%	100%	100%	100%
Inject repeated packets to congest the communication and to create hot spots	100%	100%	100%	100%	100%	100%
Inject packets which destination is the same as the initiator to block degrade the system performance	100%	100%	100%	100%	100%	100%
Use the rights of a co-hosting process	80%	100%	100%	100%	100%	100%
Use rights of previous tasks allocated on the IP	90%	100%	100%	100%	100%	100%

C. Performance and cost evaluation

Fig. 7 shows the performance results of the stand-alone scenario as the normalized execution time for the key quantification for security zones of different sizes. It creates a single security zone by employing the flat (MP and DH) and the hierarchical (HMAPD, HDH and HTDH) protocols. The number of members varies from 1 to 64. The results show the high dependency of the execution time on the number of members. Flat protocols present the highest penalties. The hierarchical approaches overcome the flat protocol. The HMAPD and HTDH are the best approach for bigger security zones. This result can be explained because the DH and HDH require a heavy computation and larger amounts or rounds required to establish a secret key.

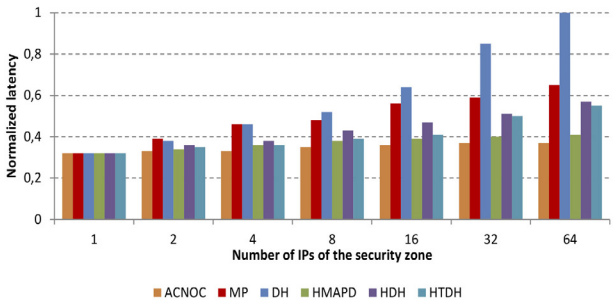

Figure 7. Normalized execution time for the stand-alone scenario.

The execution time for flat and hierarchical protocols under uniform traffic is shown in Fig. 8 and Fig. 9. In this experiment the MPSoC has 4 security zones of 3 members. The percentage of times that some of the security zones are updated are 20% and 40% of

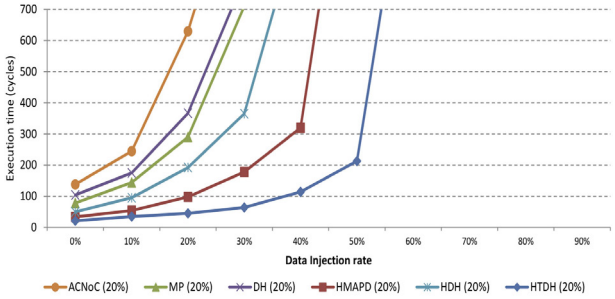

Figure 8. Execution time for Uniform traffic (20% dynamicity).

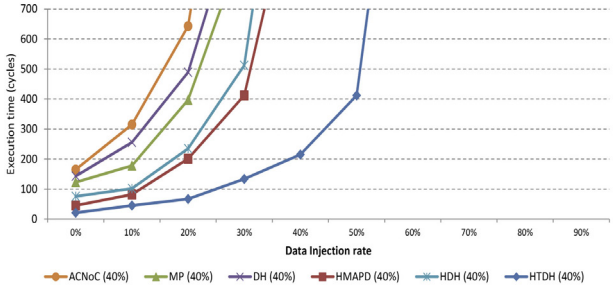

Figure 9. Execution time for Uniform traffic (40% dynamicity).

the simulation time. Hierarchical group-wise protocols overcome the performance of the flat group-wise. In our experiments, we observe that the distribution of security avoids the creation of hot spots and thus, reduces the routing time. For MPSoCs with higher dynamic behavior, the HTDH is the best option. The tree structure and the modification of just branches of the tree, favor the efficient security zone modification. The ACNoC presents the highest penalty due the long and huge amount of packets required to configure the new zones. Despite the good performance of HMAPD, its efficiency highly depends on the existence of common shared keys among the set of IPs. In our experiments we assume that the probability of sharing a common K_g is high, thus creating a memory overhead due the high amount of keys stored at the interfaces. If this probability is reduced, the system tends to behave like a central key distribution protocol.

Fig. 10 shows the normalized execution time of Splash2 benchmarks [19] on ARM processors linked to the NoC with flat and hierarchical protocols. In all the cases the hierarchical approaches' performance overcomes the flat approach.

The cost of the ACNoC, flat and hierarchical solutions are shown at Table 2 as a percentage of the penalty of each configuration when compared to the mesh NoC without security. The DH, HDF and HTDH approaches achieve the best results. The best area and power trade-off is achieved by the THDH configuration. Although the introduction of the Diffie-Hellman components, the reduction of area and power appears from the elimination of big memories included at the ACNoC, MP and HMAPD.

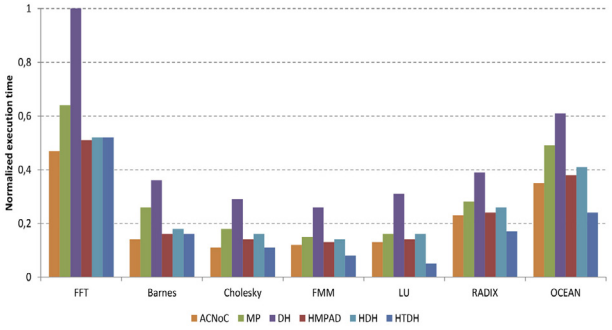

Figure 10. Execution time for SPLASH-2 benchmarks for security zones of different sizes.

Table 2. Penalty overhead compared with a NoC without security

Parameter	ACNoC	MP	DH	HMAPD	HDH	THDH
Area	12,0%	11,2	9,8%	11,6%	9,9%	9,9%
Power	10,1%	7,4	5,3%	6,9%	5,0%	4,6%

7. CONCLUSIONS

This work proposes for the first time the implementation of hierarchical group-wise security protocols for meeting the security requirements of MPSoCs. Sensitive traffic can be protected by creating security zones able to encode data with a shared secret among the trusty IP members. By decentralizing the security management, it is possible to efficiently establish group keys among the members of a security zone. Hierarchical group-wise security protocols introduce flexibility for mapping tasks which require confidentiality service. Each protocol presents a tradeoff between the computation and communication characteristics. While hierarchical Diffie-Hellman is computation intensive, the hierarchical pre-distribution mapping is a communication intensive scheme. The selection of one of the group-wise protocols will depend on the characteristics of the application and the area and power constraints of the MPSoC. As future work we plan to explore different NoC organizations for decreasing the performance penalty at the MPSoC.

REFERENCES

[1] E. Carara, N. Calazans, F. Moraes. Differentiated Communication Services for NoC-Based MPSoCs. In IEEE Transactions on Computers, vol. 63, 2014.

[2] J. Sepulveda, D. Florez, J. Diguet, M. Strum, C. Zeferino, G. Gogniat. Elastic Security Zones for NoC-Based 3D-MPSoCs. In In Proceedings of the 21st IEEE International Conference on Electronics, Circuits and Systems. ICECS 2014.

[3] J. Sepulveda, D. Florez, G. Gogniat. Reconfigurable group-wise Security Architecture for NoC−Based MPSoCs Protection. Proceedings of the 28th Symposium on Integrated Circuits and Systems Design, SBCCI 2015.

[4] J-P. Diguet, S. Evain, R. Vaslin, G. Gogniat, and E. Juin. Noc-centric security of reconfigurable SoC. In Proceedings of the First International Symposium on Networks-on-Chip, pages 223–232. IEEE Computer Society, 2007.

[5] L. Fiorin, G. Palermo, C. Silvano. A security monitoring service for NoCs. In Proceedings of the 6th IEEE/ACM/IFIP international conference on Hardware/Software codesign and system synthesis, pp. 197–202, 2008.

[6] L. Fiorin, S. Lukovic, G. Palermo. Implementation of a reconfigurable data protection module for noc-based mpsocs. In International Symposium on Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE, pp. 1–8. IEEE, 2008.

[7] J. Sepulveda, R. Pires, G. Gogniat, W. Chau, M. Strum. QoSS hierarchical NoC-based architecture for MPSoC dynamic protection. International Journal of Reconfigurable Computing. 2012.

[8] H. Yong et al. Automatic ILP-based Firewall Insertion for secure Application-specific Networks-on-Chip. In International Workshop on Interconnection Network Architectures: On-Chip, Multi-Chip (INA-OCMC), HIPEAC 2015.

[9] J. Sepulveda, J. Diguet, M. Strum. G. Gogniat. NoC-Based Protection for SoC Time-Driven Attacks. In. IEEE Embedded System Letters. Vol 7, pp. 7-10. 2015.

[10] D. M. Ancajas, K. Chakraborty, and S. Roy. Fort-NoCs: Mitigating the threat of a compromised noc. In Proceedings of the The 51st Annual Design Automation Conference on Design Automation Conference, DAC 14, pp. 158:1–158:6, USA, 2014.

[11] E. Rescorla. Diffie-hellman key agreement method. RFC 2631 (Proposed Standard), June 1999.

[12] R. Anderson, R. Needham. Robustness principles for public key protocols. In Proceedings of the 15th Annual International Cryptology Conference on Advances in Cryptology. Crypto 1995, pp 236-247.

[13] C. Haowen, A. Perrig, D. Song. Random key predistribution schemes for sensor networks. In Proceedings of IEEE Symposium on Security and Privacy, 2003.

[14] A. Bogdanov et al. “PRESENT: An Ultra-Lightweight Block Cipher”. In: Cryptographic Hardware and Embedded Systems CHES 2007, pp. 450–466. 2007

[15] D. Hong, J. Sung, S. Hong, J. Lim, S. Lee, B. Koo, C. Lee, D. Chang, J. Lee, K. Jeong, H. Kim, J. Kim, and S. Chee. HIGHT: A New Block Cipher Suitable for Low-Resource Device. In: Cryptographic Hardware and Embedded Systems CHES 2006, pp. 46–59.

[16] C. Lim and T. Korkishko. mCrypton - A Lightweight Block Cipher for Security of Low-cost RFID Tags and Sensors. In Workshop on Information Security Applications – WISA 2005, pp 243-258.

[17] J. Sepulveda, G. Gogniat, R. Pires, W. Chau, M. Strum. Dynamic NoC-Based architecture for MPSoC security implementation. In Proceedings of the 24th symposium on Integrated circuits and systems design. SBCCI 2011.

[18] P. Aumasson, J. Bernstein. SipHash: A Fast short-Input PRF. In Proc. INDOCRYPT, Vol 7668, p.p, 489-508. Springer, 2012.

[19] S. Cameron, M. Ohara, E. Torrie, J. Singh, A. Gupta. The splash-2 programs: Characterization and methodological considerations. In ACM SIGARCH Computer Architecture News vol 23, pp. 24–36. 1995.