SecureNoC

(Team Colonel Panic): Xiaoming Guo, Sijia He, Amlan Nayak, Jay Zhang

October 9, 2015

1 Problem Statement

Networks on Chip (NoCs) have been steadily investigated from reliability, efficiency, and performance perspectives.However, little effort has been directed towards the security of complex on-chip networks. These networks are prone to attacks similar to those perpetrated against large scale networks such as datacenters and the internet. If a single core within an NoC is compromised, it can be used as a vehicle to contaminate other cores on the network, or even the entire network itself.

2 Importance

Datacenter security is of utmost importance given the explosive growth of big-data centric applications. The amount of data that a single user generates on a diurnal rhythm is staggering and thus high performance computing with large numbers of integrated cores are now a necessity. Since a vast portion of the data being generated needs to remain private, secure handling of data by data center applications and hardware is paramount. Though mechanisms exist to thwart attacks on a large scale network of servers, little attention has been paid to the security of on-die NoCs.

3 Solution

Two important security vulnerabilities of NoCs are Denial of Service (DoS) attacks and Extraction of Secret Information attacks. Now, in order to ensure secure communication between cores on an NoC, we propose a novel scheme. We augment each router within the network with a crypto engine and a traffic monitoring unit. These modules carry out the following objectives :

- Monitor transactions in order to detect DoS attacks and halt an attack in progress
- Encrypt data packets that are deemed critical for secure operations before sending them to destination router(s)
- Act as a standalone memory transaction encryption engine if used in a non-NoC system (such as a single CPU - like SGX)

4 Implementation Details

Two modules will be implemented in both C++ and SystemVerilog. Booksim will be used to simulate a 3×3 mesh.We will add our custom router with the aforementioned functionalities to the existing Booksim source. After completing a correct implementation in C++, we will port the two modules into SystemVerilog and synthesize using Some company's random library (TBD) to gauge area and power overheads.

5 Evaluation

We will verify our design with coverage based approach. Under this approach, we will ensure that our design performs as per specification. Coverage will exercise corner cases to validate the general robustness of the design. specifically, we intend to evaluate our design in the following manner:

- We will use the API provided by Booksim to simulate high packet injection from a single node which imitates a DoS attack and check whether our traffic monitoring unit (TMU) detects the attack
- We will check whether the TMU can successfully stop an attack once it detects that it has been compromised (through a threshold mechanism)
- We will test the correct operation of the secure key exchange and the following encryption/decryption mechanism within the source and destination routers
- We will randomly flip bits in the encrypted packets and check whether the decryption logic can detect tampering
- Introduce random heavy traffic patterns with injections from multiple nodes in Booksim
- Code corner case conditions and check whether random testing hit the particular corner case(s)

We will synthesize our design using TSMC 65nm library and report the area/power overheads of our proposed modules. Furthermore, the performance penalty of the crypto engine and traffic monitoring unit will be evaluated.

6 Timeline

Date	**Checkpoint**	**Task**
October 23	Checkpoint 1	Complete implementation of traffic monitoring unit
November 13	Checkpoint 2	Complete implementation of crypto engine
December 4	Checkpoint 3	Testing in progress and obtain area/power overhead
December 10	Checkpoint 4	Finish testing and finalize design