SecureNoC

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

October 21, 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 :

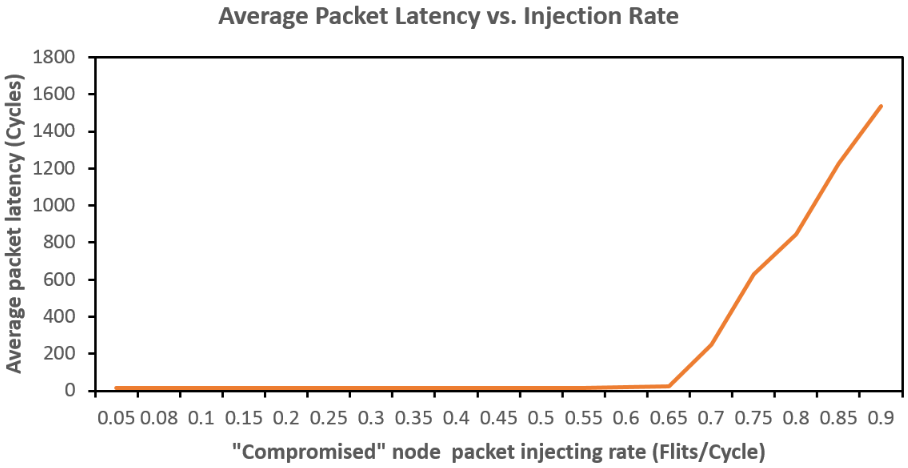
- Monitor transactions in order to detect DoS attacks and halt an attack in progress
- Encrypt data packets that are deemed critical for secure operations before sending them to destination router(s)[1]

[1] Need to discuss whether router should encrypt or Core should

- Use packet privilege classes and time division multiplexing to support secure communications

4 Progress

We have implemented a threshold based DoS attack detection mechanism in Booksim. The system can detect an abnormally high packet injection rate from any node in a given epoch (determined heuristically). Once this abnormal behavior is detected, the system will stop accepting more packets from the compromised node. We are using the Average Packet Latency to determine when/if a potential DoS attack is occurring. With a highly saturated network, the average packet latency increases significantly. We carried out simulations with a 3 × 3 mesh with uniform traffic, a flit injection rate of 0.05 for all nodes except for one, and successively increased the injection rate for the "compromised" node. We found that with an injection rate of 0.7, the average packet latency increases by an order of magnitude, as shown in the figure below:



Using this result, we are able to determine a threshold for the flit injection rate that might be due to a DoS attack. Once the attack is detected, our current mechanism simply stops accepting any more flits from the compromised node. We would like to explore other recovery mechanisms that avoid potential false positives - for instance, by only shutting out a node for a finite number of cycles and then allowing it to inject again.

5 Issues/Showstoppers

The following are a list of issues we currently face:

1. **Crypto-Engine vs. Privilege Level Time Division Multiplexing** : We are debating between two approaches to communication security on the NoC. The first approach involves an encryption engine embedded into the router logic which only encrypts packets if a particular bit is set by the source node in the said packet. The second approach involves Time Division Multiplexing (TDM) that allots specific cycles to packets of different privilege levels.

2. **DoS Detection HW** : In order to do the verilog implementation of our DoS detection and recovery mechanism, we need to decide which hardware component within an NoC needs to be extended; the router or the network interface.

3. **Verilog Implementation** : We are not certain about how extensive our Verilog models need to be. Should we model a single router with 5 ports and a network interface with our additional security mechanisms or should we create an entire 3×3 mesh that is analogous to our Booksim model?