Scalability of the Parallelized Pollard Rho Method

Faisal Nawab (nawab@cs.ucsb.edu)

October 30, 2015

Abstract

The integrity of elliptic curve cryptography (ECC) stems from the presumed complexity of the elliptic curve discrete logarithm problem (ECDLP). Thus, the study of algorithms to solve the ECDLP is essential for two reasons: (1) to establish the integrity of ECC, and (2) to guide designers in the choice of ECC parameters, such as the length of the parameters and the characteristics of the curve.

The Pollard-Rho method [1] is one of the most effective general algorithms for computing elliptic curve discrete logarithms. A plethora of work have been proposed to improve the efficiency of the Pollard-Rho method. In this work, we focus on the parallelized Pollard-Rho (PPR) method proposed by Oorschot and Wiener [2]. It is timely to revisit this direction in light of the emergence of large-scale many-core systems and cloud computing. Attackers, now more than ever, have access to larger resources in terms of computation and storage.

The main question that I will be investigating is: *how scalable is the PPR method?*. In the original work, Oorschot and Wiener showed that their PPR method leads to a linear speedup with the number of processors. This, however, is a theoretical result, which does not take into account the practical challenges of distributed computing. These challenges are caused by: (1) the need for coordination between cores/machines and, (2) the need to store large amounts of data which might not fit in main-memory and thus causes an I/O bottleneck. PPR, unfortunately, needs both extensive coordination (to detect collisions), and generates large amounts of data (to remember the generated sequence).

In this project, I will implement the PPR and study its scalability in many-core and cloud settings. This empirical study will provide us with insights on the scalability limitations of PPR. Understanding these limitations will lead to more efficient designs of the PPR method that are suitable for a many-core, cloud era.

References

[1] Pollard, John M. "Monte Carlo methods for index computation" Mathematics of computation 32.143 (1978): 918-924.

[2] Van Oorschot, Paul C., and Michael J. Wiener. "Parallel collision search with cryptanalytic applications." Journal of cryptology 12.1 (1999): 1-28.