Public Key Encription with Pattern Matching

Élie Bouscatié

December 6, 2021

Many interesting applications of pattern matching (*e.g.* deep-packet inspection or medical data analysis) target very sensitive data. In particular, spotting illegal behaviour in internet traffic conflicts with legitimate privacy requirements, which usually forces users (*e.g.* children, employees) to blindly trust an entity that fully decrypts their traffic in the name of security.

The compromise between traffic analysis and privacy can be achieved through searchable encryption. However, as the traffic data is a stream and as the patterns to search are bound to evolve over time (*e.g.* new virus signatures), these applications require a kind of searchable encryption that provides more flexibility than the classical schemes. We indeed need to be able to search for patterns of variable sizes in an arbitrary long stream that has potentially been encrypted prior to pattern identification. To stress these specificities, we call such a scheme a stream encryption supporting pattern matching.

Recent papers use bilinear groups to provide public key constructions supporting these features. These solutions are lighter than more generic ones (*e.g.* fully homomorphic encryption) while retaining the adequate expressivity to support pattern matching without harming privacy more than needed. However, all existing solutions in this family have weaknesses with respect to efficiency and security that need to be addressed. Regarding efficiency, their public key has a size linear in the size of the alphabet, which can be quite large, in particular for applications that naturally process data as bytestrings. Regarding security, they all rely on a very strong computational assumption that is both interactive and specially tailored for this kind of scheme.

In this paper, we tackle these problems by providing two new constructions using bilinear groups to support pattern matching on encrypted streams. Our first construction shares the same strong assumption but dramatically reduces the size of the public key by removing the dependency on the size of the alphabet, while nearly halving the size of the ciphertext. On a typical application with large patterns, our public key is two order of magnitude smaller than the one of previous schemes, which demonstrates the practicality of our approach. Our second construction manages to retain most of the good features of the first one while exclusively relying on a simple (static) variant of DDH, which solves the security problem of previous works.