An Efficient Implementation of a Threshold RSA Signature Scheme

Brian King
Purdue School of Engineering & Technology
Indiana Univ. Purdue Univ. at Indianapolis
briking@iupui.edu

Abstract. Several threshold RSA signature schemes have been proposed, in particular the schemes [4, 8] and [20]. Recent research has shown that the earlier schemes [4, 8] may be in some cases more “efficient” than these later schemes. Here we describe efficient implementations of threshold RSA schemes as well as further enhancements to improve performance of the Desmedt-Frankel scheme. Our conclusion is that if memory is not an issue there will be situations, for example if one can expect shareholders know who will be participating in the signature generation, that the Desmedt-Frankel scheme is very efficient.

1 Introduction

Digital signatures provide a means for binding a message and an identity. In many applications it is desirable that the secret key (signature key) does not reside on a single device, but rather that it is distributed via some sharing mechanism to several devices. For example, in high-level government setting, a single device may become compromised or disabled. If the secret key had been solely safeguarded on that device then authorized people may no longer able to generate signatures with that secret key and/or unauthorized people, that possess the device or key, may be able to sign. There are numerous other examples which illustrate the need for distributing the key to several devices. A t out of n threshold scheme is such that shares of secret k are distributed to n participants so that any set of t participants can compute k, and where any subset of t − 1 or less participants gain no information about k.

RSA [18] is a popular cryptographic primitive. The development of threshold RSA was problematic due to the fact that the modulus φ(N), as well any multiple, cannot be leaked to any of the shareholders. Threshold RSA has been examined in [5,6], then in [4,8,13], and most recently in [1,10,11,12,17,20]. The Desmedt-Frankel scheme [8] was the first secure threshold RSA sharing scheme. This is a zero-knowledge threshold scheme (for a formal definition of zero-knowledge threshold schemes see [3]). Further this scheme is a group independent scheme. That is, the shareholder’s reconstruction of the secret key is independent of the group. Group independent schemes provide a flexible method to achieve threshold secret sharing. However, there is a disadvantage when using
the Desmedt-Frankel scheme. The disadvantage is the amount of resources it requires, in the sense of memory (share size) and processing (computational time). The memory requirement is caused by share expansion. The share expansion is such that a single share will consist of $O(n)$ subshares (where $n$ is the number of shareholders) drawn from the keyspace. The processing cost comes from the computing requirements. Computations will need to be performed on these large shares. Moreover, computations will need to be performed in an extension ring. These resource requirements (and interest in development of robust, proactive, and/or verifiable threshold RSA) has led to searching for other schemes. In [20], Shoup described his Practical Threshold Signatures, which is widely regarded as the most efficient threshold RSA signature scheme. Further improvement are being developed, for examples schemes which include key generation [3].

Work has been initiated in realizing the precise computational requirements for the Desmedt-Frankel scheme. In [9], the authors established that within the Desmedt-Frankel scheme, the share size for each participant could be halved. In [14,15], algorithms were developed to reduce the number of required computations when using the Desmedt-Frankel scheme. Further, a comparison of the computations required by the Desmedt-Frankel signature scheme with Shoup’s Practical Threshold Signature scheme was made. It was pointed out that in many cases, it appeared that the Desmedt-Frankel scheme performed better than the signature scheme developed by Shoup (as long as it is assumed that the shareholders who wish to form the signature know all others who are willing to sign). The comparison was developed using complexity theory. In the following, we describe the results of a software implementation of [8]. This work illustrates that with application of algorithms developed in [14,15] significant performance improvement of the Desmedt-Frankel scheme (from now on we refer to it as the DF scheme) can be achieved. We also discuss further improvements. We provide a comparison between the work required by a shareholder of the DF scheme, compared to the work done by a RSA signer, as well as, compare it to the work done by a shareholder in Shoup’s scheme. Although our results are based on comparison of the DF scheme with the Shoup scheme, one would see similar results if they compare the DF scheme to any threshold RSA scheme which avoids using the Lenstra constant (see [8]).

2 Background: The Desmedt-Frankel Scheme

In [8], Desmedt and Frankel showed how to share with zero-knowledge a secret over any finite abelian group. In [4], Desmedt, De Santis, Frankel and Yung extended this to zero-knowledge sharing a function over any finite abelian group. Of course the motivation was to develop a zero-knowledge sharing of RSA keys.

---

1 The Lenstra constant is the cardinality of the largest set of units in $\mathbb{Z}[u]$ whose differences are also units (here $\mathbb{Z}[u]$ is the algebraic extension of the integers). The Lenstra constant maybe be used if one tries to generalize Shamir’s secret sharing, in this case one needs a unit for each participant. Hence, in this case the Lenstra constant needs to be $\geq n$. 