Evaluation of Security Level of Cryptography

Key-Sharing Schemes (Summary)

Phillip Rogaway

University of California at Davis
Chiang Mai University

January 16, 2001

1 Summary of Files

In support of Information-technology Promotion Agency, Japan, I assembled a team to look at at the four key-distribution protocols submitted to CRYPTREC. The team consisted of **Mihir Bellare** (UC San Diego), **Dan Boneh** (Stanford), and me. Our analysis is contained in the following files:

0		**ks-cover.pdf**	3 pages	Executive Summary
1	HDEF-ECDH	**ks-ecdh.pdf**	7 pages	Detailed Report
2	HIME-1	**ks-hime1.pdf**	8 pages	Detailed Report
3	ECDHS (of SEC 1)	**ks-ecdhs.pdf**	24 pages	Detailed Report
4	ECMQVS (of SEC 1)	**ks-ecmqv.pdf**	18 pages	Detailed Report
5		**ks.pdf**	60 pages	Concatenation of the above

On each of the detailed reports I have listed the authors in approximate order of contribution.

2 Summary Findings

We briefly summarize our findings.

1. **HDEF-ECDH**: This proposal is not (primarily) a key-sharing scheme; it is, instead, a method for choosing elliptic curve domain parameters. For that purpose it is a sensible method; in particular, the groups it finds will resist all currently known attacks. All the same, CRYPTREC had made no request for the functionality delivered by this proposal. Even if they had, we see no strong advantages over simply using the curves published in SEC 2, which likewise resist all currently known attacks. A section in the proposal describes using the resulting curves for key-sharing goals, but that section simply says to use the standard Diffie-Hellman approach. Analysis of that approach can be found in our report on ECDHS.

2. **HIME-1**: This isn't a key-sharing scheme at all; it is a trapdoor function. From such a function one can build an encryption scheme using standard methods. The intent was for the resulting encryption scheme to support faster decryption than alternative methods. But, comparing across a given level of security, we find that there are alternatives that have comparable or better speed for decryption, and superior speed for encryption. We see few advantages.

3. **ECDHS (of SEC 1)**: This is an open-ended family of key-sharing schemes. The methods here are efficient, and are already widely standardized. Still, most of these schemes fail to meet some significant security goals, and it is not carefully described in the spec what goals are achieved by what schemes. No provable security results are claimed in SEC 1. We sketch one security result, for one of the "strongest" of the instantiations possible. We find the schemes too open-ended, and having goals that are too ill-defined.

4. **ECMQVS (of SEC 1)**: This is again family of key-sharing schemes (that is, it supports multiple trust models). ECMQVS is particularly efficient, and is already widely standardized. For the trust model of primary interest (static/ephemeral keys for both parties) strong security results are already known to be impossible, by virtue of an attack described by Kaliski. This attack can be closed by various counter-measures (still within the scope of the standard). Still, one should not have to do such things, and the scheme appears rather ad hoc regardless. No provable-security results have appeared. We show that the scheme depends on an unusual assumption, that we call the LDH assumption. We do not now how this assumption is related to other, more standards assumptions.

Overall, we would be happier with key-sharing schemes that *are* key-sharing schemes, that are not too open-ended in instantiation possibilities, that claim to solve some clearly-specified problem, and that have verifiable proofs, under standard assumptions, to back up this claim. We are not there.

Detailed information is found in the four attached reports.

3 Where From Here?

The state-of-the-art for key-sharing schemes is not nearly at the level of the state-of-the-art for other cryptographic goals getting standardized by CRYPTREC (symmetric encryption, asymmetric encryption, digital signatures, and cryptographic hash functions). This is the only domain among those where: (1) there is no complete and agreed-to set of definitions; (2) there are no proven correct protocols for many of the problems one wishes to address; (3) the desired level of variability in security goals, and the round-cost in achieving them, motivates accepting a plurality of schemes; (4) the associated level of definitional complexity makes a mathematical treatment long and weighty; (5) even an informal description of requirements would be the subject of much debate. Thus it would seem to be harder to devise at a good standard for key-sharing than for any other problem considered. It is arguably a couple years premature to have a good standard in this domain.

In the face of this situation, natural options include:

- **Do nothing.** Do not pursue any new standard in this domain. Wait.

- **Copy.** Just take some existing standard, recognize it as flawed, refer to it, and consider it to be good enough. Revisit in a few years, with the hope of there being better options.

- **Copy-with-added-restrictions.** Slightly tweak some existing standard, getting rid of the worst parts. But don't invent much.
- **Invent.** Follow a different methodology than inviting people to submit proposals. Commission a well-chosen group of people to suggest a standard for a very specific purpose.

For a key-sharing scheme it is impossible to do well without a good understanding of the requirements. If the goal were asymmetric encryption, say, people could mostly agree about what is a good goal. The same can be said for digital signatures, block ciphers, and numerous other primitives. But for key-sharing the goals are not clear, and, more than with other primitives, there may be multiple goals, with their importance depending on the context. The difficulty of doing well in this domain should not be underestimated.

Sincerely,

Phil Rogaway