Advanced Encryption Standard

Draft Minimum Requirements and Evaluation Criteria

Abstract. This is the minute of a discussion held at the Fourth Fast Software Encryption Workshop, Haifa, Israel, on Monday January 20, 1997 from 15.30 to 16.30 on the NIST call for comments on the Advanced Encryption Standard proposal. The discussion was held in the presence of over 50 workshop participants from all over the world. These comments were collected during the discussion by Ross Anderson (the discussion chair), Bart Preneel, and Eli Biham, and then circulated by email to the participants who submitted a few further comments. The final draft was prepared by Ross Anderson.

General Comments

1. It was asked whether there should be a standard at all, or whether a diversity of algorithms might be safer and more adapted to applications. (This argument had been advanced by the NSA in opposition to the adoption of triple DES as a standard.) The counterarguments were
   (a) that a standard would be adopted whether we like it or not and we might as well help make it a good one
   (b) for due diligence reasons, many clients would only use an algorithm with a government seal of approval
   (c) that a new standard would give an opportunity for many existing systems to be redeveloped and serious vulnerabilities in protocols etc removed
   (d) that a new standard would concentrate cryptanalytic effort on a single target, which (if unsuccessful) would increase confidence in that target
   (e) that the AES initiative presented an opportunity to establish a standard supported from the outset by government, industry and the academy.
2. Public trust in the algorithm will be harder to build if the rationale behind design decisions is not made fully public, and if the public does not participate in the evaluation process. So the rationale behind all design decisions should be completely explicit.
3. It would be helpful if any S-boxes, constants etc should be chosen by some convincing method (such as at random from a sufficiently large space). There are two reasons for this. Firstly, if all the design choices are made by a single person or organization, then the algorithm will be less likely to be trusted; trapdoors will be suspected. On the other hand, we do not want a "committee" design. A customisable design is probably the best balance between these concerns. Secondly, there are users who will want to customise a standard algorithm (see 11 below).
4. We would favour a process in which the initial submissions are whittled down to a short list of perhaps 3-4 candidates. This would enable the community

to concentrate the analysis and evaluation effort on them rather than dispersing it on dozens of targets. (In this workshop alone about ten ciphers were suggested.)

5. NIST should clarify the role of non-US citizens. Clearly, a new US standard will (like DES) become widely used in other countries. Will non-US submissions be acceptable?
6. There is concern that the proposed timetable does not leave enough time for serious cryptanalysis.

General Requirements

7. It is not clear that one cipher can satisfy the requirements for all applications, and on all kinds of processors (or special hardware). The question arise whether we should have a family of ciphers, appropriate for different environments.

   For example, the majority of fielded DES implementations are on 8-bit processors such as smartcards and microcontrollers, and used in applications such as banking, power metering, pay-TV key management, door locks, road tolls and the like. In such applications, the main 'improvement' sought from a DES successor is a reduction in code size.

   On the other hand, the importance of intellectual property protection is growing and there is wide use of stream ciphers in, for example, pay-TV systems. Here, speed is a definite requirement and code size is relatively unimportant.

   So NIST should consider whether there should be two standards: a block cipher suitable for 8-bit processors, and a stream cipher optimised for speed.
8. There was wide condemnation of the draft proposal, that C source code be evaluated on a PC. Ideally, a survey of applications, both fielded and planned, should be undertaken so that the relative importance of different performance metrics (speed, code size, etc) could be evaluated and a realistic benchmark suite be specified. At the very least, NIST should be much more explicit about the performance requirements. We expand on this below.
9. NIST should also provide a ranking for the various evaluation criteria to clarify their relative importance.

Technical Requirements

10. There should be procedures agreed in advance for dealing with any weakness of the algorithm that arises later. This might be predictable, such as an advance in chip technology that makes a longer key necessary; unpredictable but minor, such as the discovery of a new but rare class of weak keys; or catastrophic, such as a new shortcut attack that forces a change to a completely different algorithm.

    Several mechanisms are thus likely to be necessary including a review body or process, a 'backup algorithm' and perhaps (as suggested by NIST) a means of increasing the keylength. There was no unanimity on this last point however; an alternative would be to adopt an algorithm with a keysize well beyond possible exhaustive search (e.g., 256 bits) and use part of the keyspace as appropriate.

One possible 'backup algorithm' is using the same algorithm with different parameters, such as with a different set of S boxes. This could provide a rapid and low-cost means of recovering from all but a total break.

11. There are other reasons to support customization by other means than the key. In addition to the building public confidence in the absence of trapdoors, as mentioned above, parametrisation will appeal to those users who want a compromise between a proprietary algorithm and a standard one - such as those who at present use DES with nonstandard S-boxes or other modifications to prevent keysearch. The successor to DES should be chosen so that it is not as difficult to choose strong values of the S-boxes or other constants as it is in the case of DES.
12. An increasing number of applications involve cryptographic authentication protocols (Kerberos being an example). Here, the 64-bit blocksize of DES is a disadvantage; the real requirement is to encrypt variable length blocks. Many implementers use DES-CBC but this can be vulnerable to cut and paste attacks. A block cipher of variable width would be ideal for such applications.
13. Some people felt that a 64 bit blocksize was inadequate for security reasons, as once large volumes of data start to be encrypted the volume limits set by the birthday paradox may be approached.
14. Given that the algorithm may be of variable width and may also have a variable key length, thought needs to be given on how such parameters will be securely expressed. The RC5 approach of packaging the key in a 'control block' with such parameters might provide inspiration here, as could the IBM approach of 'key control vectors' to enforce a functional partition of the keyspace where applications require this. We probably need an algorithm version number as well, and 'fields to be defined later'.
15. In the event that the standardized algorithm is simply another 64-bit block cipher, there is a need for a standard mode of operation that allows a variable length block to be encrypted with error extension in both directions. More generally, it is time to look not just at modes of operation but also at other supporting structures such as APIs and lower level interface definitions.
16. The algorithm should approximate to a random permutation as closely as possible, e.g. there should be no equivalent keys, no complementation properties, no related keys and no weak keys.
17. The bit naming convention should be explicitly defined.

Security Requirements

18. The types of attacks that the cipher must withstand must be made explicit (e.g., known plaintext, chosen plaintext, adaptive chosen plaintext/ciphertext, related-key).

19. The security targets must be quantified, e.g. '2^{10} related key queries, 2^{40} chosen plaintexts, 2^{50} storage, 2^{60} known plaintexts, 2^{80} effort'.
20. There must be minimum values set for security parameters, such as number of rounds, block size and key size, in order to prevent loss of confidence in the standard following a published attack on a legitimate implementation.

Efficiency requirements

21. As noted above, it was widely felt to be unwise to evaluate the candidate algorithms solely on a PC, as the majority of DES implementations are believed to run on 8-bit processors in embedded applications. It appears to be prudent engineering practice to optimise an algorithm for the slowest processor on which it will be widely used - which might mean the 8051 (although 4-bit processors are still used, and GOST appears to have been designed with these in mind). It should also run adequately in Java, as the commercial success of this language cannot be ignored.
    PCs will be important, but we do not know whether the typical PC CPU in five years time will be a RISC processor such as Alpha, a VLIW processor such as Philips' TriMedia, or a combination superscalar/SIMD such as Klamath. Similarly, hardware/firmware implementations (FPGA, ASIC, standard cell,...) should be considered.
22. Some applications, such as B-ISDN require fast key setup. The evaluation criteria should therefore define a maximum key scheduling delay; this might defined relative to encryption as a function of key length. A possible alternative would be ability to cache a number of round keys. However, while 1024 keys might be sufficient for current ATM switches, more keys might be needed by future equipment.
23. There should be targets for code size and memory size, especially for implementations on smartcards and other 8-bit processors. For hardware implementations, there should be a target gate count; and for power-critical applications (such as contactless smartcards) there should be a power target of microjoules per block encrypted.

Evaluation and interface requirements

24. The process of evaluation should involve bounties to attract serious and sustained attack. It is suggested that NIST offer a large sum (say $1m) for a significant shortcut attack. This should ensure that anyone outside the sigint community who discovers such an attack will report it rather than seek to exploit it. The shorter term evaluation procedure should be also clarified: what incentives will there be for outside contributors to invest effort in it?
25. When reducing a large number of candidates to a shortlist, one possible approach to the performance issue would be to define a minimum speed relative to known ciphers such as DES or triple-DES. However, some participants felt

that many people are unaware of, or have no access to, fast DES code for comparison.

26. In any case, a thorough examination of the performance aspects of shortlisted candidates should be carried out. As mentioned above, there would ideally be a study of existing and planned applications leading to the development of a benchmark suite. In the absence of such an exercise, then at the very least the following should be considered for each shortlisted candidate:
    (a) code and memory size, especially on common smartcards and microcontrollers
    (b) speed, not just on currently common chips such as 8051 and Pentium but also RISC and VLIW chips
    (c) gate count for simplest and fully pipelined hardware implementations. Tradeoffs between speed and gate/count should be considered, as well as the minimum number of microjoules per block encrypted
    (d) whether software implementations are significantly different (or more difficult) according to whether the processor is big endian or little endian
    (e) key agility, or round key memory requirements if cacheing is preferred for B-ISDN applications
    (f) whether there is a well understood tradeoff between number of rounds and attack effort
27. NIST should define a standard interface for the algorithms in order to facilitate validation by the wider crypto community.
28. Ease of validation is important. A single test vector is not enough: the algorithm designer should supply a full set of test vectors, plus a validation suite that exercises them via the standard interface mentioned above and performs any other tests required to check all single points of failure and thus ensure that an implementation is correct.
29. Submissions should include not just one or more implementations optimized for speed or memory size on various processors but also an easy-to-read endian-indifferent one, so that correspondence with the description of the cipher can be readily checked.
30. Finally, the evaluation criteria should be more carefully drafted. For example, criteria (b), (c) and (d) overlap, and it is not clear what exactly is meant by ‘simplicity’ and ‘flexibility’.