The Case for Serpent

Ross Anderson, Eli Biham and Lars Knudsen

24th March 2000

Summary

Serpent should be chosen because it is the most secure of the AES finalists. Not only does it have ample safety margin, but its simple structure enables us to be sure that none of the currently known attacks will work. It is also simple to check that an implementation is correct. Although Serpent is not as fast as the other finalists on the 200 MHz Pentium machine used for round 1 benchmarking, this disadvantage largely disappears when we consider the likely platforms and applications of the 21st century. In hardware, for example, Serpent has easily the best performance, while on IA64 it's second.

1 Security

The most important requirement is stated succinctly in the AES announcement [7]: *'The security provided by an algorithm is the most important factor in the evaluation.'*

From the day in September 1997 when we started designing Serpent, we asked ourselves what protection requirements we were trying to meet. We concluded that AES needed to last for a useful service lifetime plus a human lifetime after that. That means at least a century. So we like the AES motto of a *'crypto algorithm for the twenty-first century'*. Also, if Moore's Law runs out sometime this century, then the AES might never be replaced. So the selectors should consider how their choice will look in the twenty-second century and beyond.

1.1 Advances in mathematics

An algorithm may break if someone comes up with a powerful new theory. We do not believe that the history of cryptanalysis is over. Although we have no real idea what the next hundred (or five hundred) years of mathematics will bring, there are three things we can do to future-proof a design.

First, a block cipher should be simple and easy to analyse. The DES algorithm had such a complex description that until the late 1980's no-one appears to have tried seriously to attack it. When they did, differential [5] and then linear [9] attacks were found – both of which can now be explained to bright students in a single 50-minute lecture.

Second, a block cipher should have more rounds than are needed to block today's attacks. Improvements in cryptanalysis usually increase the number of rounds required.

Third, a block cipher should use only well understood primitives. S-boxes and SP-networks have been around for over a quarter of a century, so it is less likely that surprising new attacks will be found on them.

Serpent was designed with all these considerations firmly in mind.

1.2 Engineering issues

Moore's Law may be the most obvious interaction between crypto security and engineering. But assurance is at least as important. If Moore's law continues, then 128-bit keys will be vulnerable in about a century; but many systems fail right now because of design and implementation errors.

Complicated algorithms are hard to implement correctly, and it is harder still to prove implementations to be correct. Serpent's simple design makes verification easier. It is so simple that it can be optimised in high level languages such and C and Ada. So a developer can avoid many of the errors that creep into assembly language routines.

Many secure systems are also vulnerable because of poor random number generators, memory remanence or other engineering failures (e.g., [3]). These risks provide an even more compelling argument for 256-bit keys than either Moore's Law or quantum computers. It would be nice if implementation failures became less common over time, but experience suggests the contrary. As systems get more complex, there are more things to go wrong.

1.3 Public confidence

Ciphers can also be damaged through erosion of public confidence.

Recall the effect which the invention of differential and then linear cryptanalysis had on the standing of DES. Neither of these attacks is practical: there are no DES applications known to us where an opponent might get hold of 2^{40} texts. Indeed, a prudent designer would normally never use any key for a 64-bit block cipher to encipher more than 2^{32} texts. Yet despite the discoverers' strenuous efforts to keep the story straight, differential and linear attacks became translated in the public mind to 'DES has been broken'. It's imprudent to expect the public to distinguish between practical attacks and 'certificational' attacks – attacks which require infeasibly large amounts of data or effort.

We have often been asked why, given that Serpent is secure today with at most 16 rounds, we do not allow 16 rounds – at least for 128-bit keys. The answer is this. Having experienced what happened to DES, we are concerned that, in perhaps 50 years' time, advances in mathematics will lead to a certificational attack on 16-round Serpent. As the other AES finalists have no more margin of safety than 16-round Serpent, they run a similar risk. (That is why we believe that they should have more rounds, rather than Serpent having less.) We think such an attack on Serpent is unlikely. But 'unlikely' isn't enough; the AES algorithm should have the highest achievable level of design assurance.

So we believe that the Advanced Encryption Standard should be 32-round Serpent with 256-bit keys. If people want to use less than 256 bits, or less than

32 rounds, then they should do so only with good reason, and understand that the two issues are orthogonal. The threats against 128-bit 32-round Serpent and 16-round 256-bit Serpent are different.

2 Performance

Many superficial analyses of the AES finalists have concluded that Serpent is half the speed of the other candidates, because we used twice as many rounds as we needed to. This is not accurate.

The three most important aspects of performance are hardware complexity, software speed and memory cost. We have already discussed memory usage extensively in [2]; this is the critical parameter for embedded and smartcard applications. Serpent does extremely well here. We will spend the rest of this section discussing hardware and software.

First, Serpent is the best of the AES finalists in hardware – even with the full 32 rounds. An independent team produced implementations for the Xilinx XCV1000 FPGA of RC6, Rijndael, Serpent and Twofish[1]. Serpent was the only finalist for which a fully pipelined implementation could be fitted into a single chip. Serpent was also by far the fastest, achieving a throughput of 5.04 Gbit/sec, versus 2.40 Gbit/sec for RC6, 1.94 Gbit/sec for Rijndael and 1.71 Gbit/sec for Twofish [6]. An NSA study of ASIC costs predicts 8.03 Gbit/sec for Serpent versus 5.163 for Rijndael, 2.171 for RC6 and 1.445 for Twofish [12].

Second, several AES finalists are heavily optimised for encrypting very large files on the Pentium II. But in most applications, key agility matters more, and this isn't likely to change any time soon.

Gigabit networks already demand encryption of ATM cell streams. This often won't be done in the end systems, as people rely increasingly on boundary control devices such as firewalls or guards to create virtual private networks. This is likely to mean changing the key every three blocks.

In low cost embedded systems, key changes are already common. In [2] we described a typical fielded electronic purse system where each transaction involved ten key set-ups and fourteen block cipher operations.

So we believe that most real applications will have one key change every 1–5 encryptions, and suggest for simplicity's sake that the benchmark should be the one natural in ATM networks, namely the average cost of one key change plus three block cipher operations. On this benchmark, Serpent does not badly across a wide range of platforms, especially the IA-64 architecture which will almost certainly be the standard for the next generation of PCs. According to engineers from Hewlett Packard, the relevant figures are [13]:

	MARS	RC6	Rijndael	Serpent	Twofish	Serpent is:
IA64	2965	3051	504	**2269**	2991	**2nd**
PA-RISC	3409	2686	666	**2415**	3453	**2nd**

[1] Although this team did not implement MARS, there seems no reason to suppose that MARS would do any better than RC6

The above figures are the average clock cycle costs, over encryption and decryption, of one key setup plus three block cipher operations. Even on Pentium, using this benchmark, Serpent is the third fastest algorithm when one combines the published cycle count figures from Gladman [8] and Osvik [11], and fourth fastest combining Worley et al [13] and Osvik. It's second and third respectively with Osvik's latest figures (2531 cycles on a K7). We hope to have stable and comparable figures by the May 15th deadline. NIST's results also show Serpent doing well on Ultrasparc II [4] (though unfortunately without clock cycle counts).

One of the main things to emerge from the extensive testing of round 2 finalists is that some algorithms achieve high throughput at the cost of slow key setup, while others are reasonably key agile. We believe that very many application designers will prefer the latter.

Another point is that some algorithms achieve high software throughput at the cost of high hardware complexity. We believe that the AES should have a simple hardware implementation.

We are not trying to claim that Serpent is the fastest algorithm. Speed was not the primary goal of the AES competition, and we designed Serpent according to the specification from NIST. What we do say is that Serpent's security was not bought at an unacceptable price in speed.

3 Miscellaneous

Much has been written recently about power analysis. One of us is currently doing an implementation of all five finalists on an 8051-based smartcard with no specific power analysis defences. As the bulk of the work is being done by students, full results aren't expected until the end of the academic year. But from what's known so far, we don't expect that any one finalist will be much superior to any other: just that the attack techniques will differ.

The likely solution to power analysis is hardware engineering, and a strong contender is dual-rail logic in which the current drawn is independent of the data. One of us is involved in such a project [10]. Dual-rail design is easier where one only has to worry about the simple logical operations used in Serpent, rather than operations with carry, and especially multiplications. So choosing Serpent as the AES will make the smartcard designer's job easier.

Finally, the claim that Serpent's whole key schedule has to be worked out in advance for decryption is incorrect. It is not necessary to apply the S-boxes during the forward computation.

4 Conclusion

Serpent should be chosen as the Advanced Encryption Standard. It's the fastest algorithm in hardware, and the second fastest in software on the IA-64 architecture. Above all, Serpent should be chosen because it's the most secure of the candidates.

References

1. RJ Anderson, E Biham, LR Knudsen, "Serpent: A Proposal for the Advanced Encryption Standard', submitted to NIST as an AES candidate. A short version of the paper appeared at the AES conference, August 1998; both papers are available at `http://www.cl.cam.ac.uk/~rja14/serpent.html`
2. RJ Anderson, E Biham, LR Knudsen, "Serpent and Smartcards" in *Cardis 98*, Springer Verlag (2000) pp 257–264; also available at `http://www.cl.cam.ac.uk/~rja14/serpent.html`
3. RJ Anderson, MG Kuhn, "Low Cost Attacks on Tamper Resistant Devices" in *Security Protocols – Proceedings of the 5th International Workshop* (1997) Springer LNCS vol 1361 pp 125–136
4. LE Bassham III, "Efficiency Testing of ANSI C implementations of Round 2 Candidate Algorithms for the Advanced Encryption Standard", to appear in the proceedings of the 3rd AES Candidate Conference
5. E Biham, A Shamir, *'Differential Cryptanalysis of the Data Encryption Standard'* (Springer 1993)
6. AJ Elbirt, W Yip, B Chetwynd, C Paar, "An FPGA-Based Performance Evaluation of the AES Block Cipher Candidate Algorithm Finalists", to appear in the proceedings of the 3rd AES Candidate Conference
7. "Announcing Request for Candidate Algorithm Nominations for the Advanced Encryption Standard (AES)", in *Federal Register* September 12, 1997 (Volume 62, Number 177), pp 48051–48058
8. B Gladman, "Implementation Experience with AES Candidate Algorithms", in *Proceedings of the 2nd AES Candidate Conference* (NIST, 1999) pp 7–14
9. M Matsui, "Linear Cryptanalysis Method for DES Cipher", in *Advances in Cryptology — Eurocrypt 93*, Springer LNCS v 765 pp 386–397
10. SW Moore, RJ Anderson, MG Kuhn, "Improving Smartcard Security using Self-timed Circuit Technology", Fourth ACiD-WG Workshop, Grenoble, ISBN 2-913329-44-6, 2000
11. DA Osvik, "Speeding Up Serpent", to appear in the proceedings of the 3rd AES Candidate Conference
12. B Weeks, M Bean, T Rozylowicz, C Ficke, "Hardware Performance Simulations of Round 2 Advanced Encryption Standard Algorithms", to appear in the proceedings of the 3rd AES Candidate Conference
13. J Worley, B Worley, T Christian, C Worley, "AES Finalists on PA-RISC and IA64: Implementations and Performance", to appear in the proceedings of the 3rd AES Candidate Conference