The future of integer factorization

Andrew M. Odlyzko
AT&T Bell Laboratories
Murray Hill, NJ 07974
amo@research.att.com

June 30, 1995

1. Introduction

How large should be the moduli in public key cryptosystems such as RSA and DSS (the US Digital Signature Standard)? The answer depends on the anticipated threats. Even if those are known, there is no way to provide a definitive answer, since progress in integer factorization and discrete logarithm algorithms is not predictable. (Furthermore, there is still the possibility that RSA and DSS could be broken by other methods than factoring or discrete log approaches.) However, since choices of moduli have to be made, it is necessary to make some estimates, and this note attempts to do so, taking into account both the increase in available computing power and future algorithmic developments. The projections made below suggest that RSA and DSS moduli might have to be unpleasantly large. In particular, the 512-bit moduli that have been adopted in a variety of cryptosystems are already unsafe for all applications except those with very modest security requirements.

I would like to stress that while the assertion about insecurity of 512-bit moduli is easy to support with solid technical evidence, the projections about the future are much less certain, and should not be treated as firm forecasts, but as possible ways that computational number theory might develop.

Only conventional algorithms and standard integrated circuit technologies will be considered. If either quantum computers (à la Shor) or DNA computers (à la Adleman) become practical, the outlook might change, and even larger moduli might become necessary.

Computing power will be measured in units of MY, or mips-years. By convention, a 1 mips machine is equivalent to the DEC VAX 11/780 in computing power, and so 1 MY is one year on a VAX 11/780. This measure has many defects, and nowadays a wide variety of other benchmarks are used in preference to mips measures. Still, given the uncertainty in any projection far into the future, this measure seems adequate.

43D will refer to an integer of 43 decimal digits.

Discussion will be restricted to integer factorization. Discrete logarithms are, with the present state of knowledge, slightly more difficult to compute modulo an appropriately chosen prime than it is to factor a "hard" integer of the same size, but the difference is not large. Therefore to be on the safe side in designing cryptosystems, one should assume that all the projections about sizes of integers that it will be possible to factor will also apply to sizes of primes modulo which one can compute discrete logarithms.

Table 1: Historical records in integer factorization (see Appendix B).

year	record factorizations
1964	20D
1974	45D
1984	71D
1994	129D

Table 2: Computing power used to achieve record factorizations (see Appendix C)

year	MY
1974	0.001
1984	0.1
1994	5000

2. Factorization records and historical estimates

There is a long record (see Appendix A) of estimates of sizes of integers that could be factored. They have uniformly turned out to be too low, primarily because of unanticipated algorithmic improvements. Faster than expected growth in available computing resources also played a role, though. Table 1 summarizes the progress that has occurred in the last few decades. Table 2 shows how much the computing power available for integer factorizations has increased.

3. Projections of computing power available in the future

The dramatic increase in computing power used for factorization between 1984 and 1994 resulted largely from the introduction of distributed computing, using the idle time on a network of workstations. This trend was started by Bob Silverman. (An earlier instance was Richard Schroeppel's attempt to factor F_8, but that work did not attract much public attention.) It was fully developed by Arjen Lenstra and Mark Manasse. The RSA129 factorization used idle time on around 1600 computers around the world during an 8 month period. Most modern factoring methods do lend themselves to distributed implementations.

In the remainder of this discussion, we will only consider factoring attempts similar to that on RSA129, namely ones that use idle time on networks of computers. Unlike attacks on DES, where effective attacks appear to require a special purpose machine (see Appendix J), these attacks do not require any major commitment of financial or technical resources, and can even be mounted surreptitiously.

Another projection of the future of integer factorization has been made by Ron Rivest [Rivest]. Rivest's expectations for progress in computer technology and algorithms do not differ much from mine, but he considers what can be done with a fixed amount of money using machines bought especially for factoring.

We note that even without an extensive effort, the organizers of the RSA129 project were able to obtain about 0.03% of the total computing power of the Internet. They also estimated

Table 3: 1994 computing power (see Appendix D)

	mips rating
RSA129 project	10^4
Internet	$3 \cdot 10^7$
the world	$3 \cdot 10^8$

[AGLL] that without extraordinary effort they could have organized a project with 100 times as much power, or about 3% of the capacity of the Internet.

We should also note that (and this is relevant for cryptosystem security) there are relatively small organizations that have large amounts of computing power all by themselves. Silicon Graphics, for example, has about 5,000 employees and 10,000 workstations, for total computing power of perhaps 10^5 mips, about 10 times the computing power of the RSA129 project. Thus it is conceivable that a few individuals, such as systems administrators at a large corporation, could organize a covert effort at factoring that would dwarf that of the RSA129 project. In particular, using the current implementations of the number field sieve, they could easily factor a 512-bit integer in under a year of elapsed time, and could do so now.

What about the future? Moore's "Law" says that microprocessor processing speed doubles every 18 months. In 10 years, that means an increase by a factor of about 100. Let us assume that this "law" will hold for the next 10 years (Appendix E), and that a further increase in processing power of between 10 and 100 will be achieved in the following 10 years. We then find that the typical processor in 2004 might be rated at 10^3 mips, and in 2014 at $10^4 - 10^5$ mips.

Since there are already over 10^8 computers in the world, it seems safe to assume there will be at least $2 \cdot 10^9$ by 2004. By the year 2014, we might have $10^{10} - 10^{11}$ (Appendix F). Further, by that time almost all are likely to be networked together. However, it is uncertain what fraction might be available for a factoring experiment. Let us consider two scenarios:

(a) Widely known collaborative effort to break some challenge cipher. It does not seem out of the question that up to 0.1% of the world's computing power might be made available for a year. (The RSA129 project organizers estimated they could have obtained access to 3% of the computing power of the Internet, but this 3% factor might not scale as the networks grow.) This yields $2 \cdot 10^9$ MY available in 2004, and $10^{11} - 10^{13}$ MY in 2014.

(b) Surreptitious effort arranged by a handful of people at an organization such as a corporation or university. They might have available to them 10^5 computers in 2004, and up to 10^6 in 2014. Thus they might have 10^8 MY at their disposal in 2004, and $10^{10} - 10^{11}$ in 2014. (Would they attempt to attack a single cryptographic system, or would they attempt to attack 1000 different systems? This would likely depend on what moduli are used, and on the potential payoff.)

We summarize the projections above in Table 4.

4. Factorization using current algorithms

Of the methods that are currently known and apply to generally hard integers, the general number field sieve (gnfs) has the best asymptotic running time estimate. It is also practical, and runs faster than previous algorithms on generally hard integers of more than about 115D.

Table 4: Computing power available for integer factorization (in MY)

year	covert attack	open project
2004	10^8	$2 \cdot 10^9$
2014	$10^{10} - 10^{11}$	$10^{11} - 10^{13}$

Table 5: Computing required to factor integers with current version of gnfs

bits of n	MY required
512	$3 \cdot 10^4$
768	$2 \cdot 10^8$
1024	$3 \cdot 10^{11}$
1280	$1 \cdot 10^{14}$
1536	$3 \cdot 10^{16}$
2048	$3 \cdot 10^{20}$

Since a 119D integer was factored recently using gnfs with 250 MY (see Appendix C), we can project, using standard methods (see Appendix H), the amount of computing that is likely to be required to factor various integers. The results are shown in Table 5.

Thus, based on tables 4 and 5, moduli of 1280 bits are likely to be safe for well over 20 years, and even 1024-bit moduli are are not likely to be vulnerable, unless they conceal extremely valuable information. However, Table 5 assumes that the current version of gnfs will remain the best algorithm. This would be an extremely imprudent assumption, as history shows that algorithmic improvements are often crucial.

It is important to note that 512-bit integers, which are used in a variety of commercial implementations of RSA, can already be factored with the available computing power. There is no need for new algorithms or faster or more computers, just the effort to find enough people willing to let their workstations be used in their idle time. The machines at Silicon Graphics alone could factor a single 512-bit integer in about half a year total time (under the assumption that they are idle about two thirds of the time).

5. Algorithmic improvements

The special number field sieve (snfs) applies to integers such as the Fermat numbers. Based on the recent factorization of a 162D special integer by Boender et al. in about 200 MY, we can estimate how long snfs takes to factor various integers, and the results are presented in Table 6.

In particular, it appears that it might be possible to factor $F_{10} = 2^{1024} + 1$ by the year 2000 (continuing the tradition in which F_7 was factored in 1970, F_8 in 1980, and F_9 in 1990).

Is it reasonable to expect a breakthrough that would enable generally hard integers to be factored in about the same time that the snfs factors integers of comparable size? I feel that it is prudent to expect even larger improvements. It is impossible to predict scientific breakthroughs. However, just as in other disciplines (cf. "Moore's Law"), one can observe a steady progress in integer factoring. Every few years a new algorithm appears that allows for

Table 6: Computing required to factor integers with the snfs

bits of n	MY required
768	$1 \cdot 10^5$
1024	$3 \cdot 10^7$
1280	$3 \cdot 10^9$
1536	$2 \cdot 10^{11}$
2048	$4 \cdot 10^{14}$

factoring much larger integers, and then there is a steady stream of incremental improvements. As one example, there were wide concerns that the linear algebra step that plays a crucial role in most of the fast integer factorization algorithms might become a serious bottleneck when factoring large integers, since it could not be executed easily in a distributed way. However, new algorithms were developed in the last decade that have allayed these concerns.

To factor a 129D integer with the continued fraction method that was used in the 1970s would have required about $6 \cdot 10^{11}$ times more computing power than to factor a 45D integer, but the factorization took only about $5 \cdot 10^6$ times as much because a much better algorithm was used. Thus here the algorithmic improvement was comparable (on a logarithmic scale) to the hardware one. This is similar to what has been reported in other areas, such as numerical analysis, where again better mathematical ideas have contributed about as much as faster computers.

Let us assume that algorithmic improvements will continue to be comparable to those from increasing computing power. Then we can expect that even a covert attack in 2004, which with present algorithms could only factor about a 768-bit integer, will instead be able to factor a 1024-bit one. For the year 2014, the threshold of what might be achievable rises to 1500 bits or even more.

6. Conclusions

For many applications, the conjectured progress in factorization is not a serious threat. For example, for digital signatures, time-stamping (à la Haber and Stornetta) provides a way to maintain their validity (although in a somewhat cumbersome way, requiring recourse to a document trail) even as secret moduli are factored. (This assumes, or course, that there are no totally unexpected breakthroughs, so that it is possible to estimate at any given time what are the largest integers that might be factorable in the next year, say.) Also, for many records, the security requirements are not serious, in that loss of secrecy after 10 years or sometimes even 10 days is acceptable. Hardware improvements favor the cipher designer, since a 100-fold speedup in commonly used processors allows for a 10-fold increase in the modulus in DSS (assuming, as seems reasonable, that the auxiliary prime q stays at 160 bits, or is increased to at most 240 bits), for example. However, for some records, even 20-year protection is not sufficient. In those cases extremely large moduli appear to be required for many of the most popular public key systems, such as RSA and the various ElGamal-type algorithms, such as the DSA. For extremely sensitive information, it might sometimes be prudent to use 10,000-bit moduli.

The main reason that the projections for lengths of safe moduli are growing so fast is that the asymptotic running time estimates for the latest factoring algorithms are subexponential. In particular, the growth rate for the running time of the number field sieve is not fast. By comparison, there are problems where the only known algorithms have exponential running times, such as DES and related ciphers, and also many public key schemes on elliptic curves (see Appendix I). It might therefore be prudent to consider even more seriously elliptic curve cryptosystems.

ACKNOWLEDGEMENTS: I thank Len Adleman, John Brillhart, Dan Bernstein, Marije Huizing, Hugo Krawczyk, Arjen Lenstra, Paul Leyland, Mark Manasse, Carl Pomerance, Ron Rivest, and Michael Wiener for their helpful comments.

References

[AGLL] D. Atkins, M. Graff, A. K. Lenstra, and P. C. Leyland, The magic words are squeamish ossifrage, pp. 263-277 in *Advances in Cryptology - ASIACRYPT '94*, J. Pieprzyk and R. Safavi-Naini, eds., Lecture Notes in Comp. Sci. 917, Springer, 1995.

[BS] J. Brillhart and J. L. Selfridge, Some factorizations of $2^n \pm 1$ and related results, *Math. Comp. 21* (1967), 87-96.

[DL] B. Dodson and A. K. Lenstra, NFS with four large primes: An explosive experiment, *Advances in Cryptology - CRYPTO '95*, D. Coppersmith, ed., Lecture Notes in Comp. Sci., Springer, 1995, to appear.

[DongarraMS] J. J. Dongarra, H. W. Meuer, and E. Strohmaier, TOP500 supercomputer sites, report available electronically at URL http://www.netlib.org/benchmark/top500.html.

[Gardner] M. Gardner, Mathematical games, *Sci. Amer.*, August 1977, 120-124.

[Guy] R. K. Guy, How to factor a number, in "Proc. Fifth Manitoba Conf. Num. Math.," *Congressus Num. 16* (1976), 49-89.

[Jevons] W. Stanley Jevons, *The Principles of Science,* 1874.

[LL] A. K. Lenstra and H. W. Lenstra, Jr., eds., *The Development of the Number Field Sieve,* Lecture Notes in Math. 1554, Springer, 1993.

[Menezes] A. J. Menezes, *Elliptic Curve Public Key Cryptosystems,* Kluwer, 1993.

[Pomerance] C. Pomerance, The number field sieve, pp. 465-480 in *Mathematics of Computation 1943-1993: A Half-Century of Computational Mathematics,* W. Gautschi, ed., Amer. Math. Soc., 1994.

[PST] C. Pomerance, J. W. Smith, and R. Tuler, A pipeline architecture for factoring large integers with the quadratic sieve algorithm, *SIAM J. Comput. 17* (1988), 387-403.

[Rivest] R. L. Rivest, Dr. Ron Rivest on the difficulty of factoring, first published in *Ciphertext: The RSA Newsletter*, vol. 1, no. 1, fall 1993, and reprinted, in an updated form, in an appendix on pp. 361-364 in S. Garfinkel, *PGP: Pretty Good Privacy,* O'Reilly & Associates, 1995.

[Wiener] M. J. Wiener, Efficient DES key search, TR-244, May 1994, School of Computer Science, Carleton University, Ottawa, Canada. Presented at the Rump Session of Crypto '93. Available through anonymous ftp from ripem.msu.edu:/pub/crypt/docs/des-key-search.ps.

APPENDICES:

Appendix A: Historical estimates of the difficulty of factoring

The problem of factoring integers has fascinated mathematicians for a long time, and there is a famous quote of Gauss on what a fundamental question this is. A concrete estimate of how hard it might be was provided in 1874 by W. Stanley Jevons, the English economist and logician. He conjectured [Jevons] that nobody but he would ever know the factors of the 10D integer 8616460799. However, he was proved wrong by Bancroft Brown around 1925, and perhaps by others even earlier. In an unpublished manuscript, a copy of which was kindly provided by John Brillhart, Brown explained how he obtained the factorization

$$8616460799 = 96079 \cdot 89681.$$

In 1967, John Brillhart and John Selfridge [BS] stated that "... in general nothing but frustration can be expected to come from an attack on a number of 25 or more digits, even with the speeds available in modern computers." By 1970 their estimate was out of date because of a new factoring method that allowed Mike Morrison and John Brillhart to factor an integer of 39D. In 1976, Richard Guy stated "I shall be surprised if anyone regularly factors numbers of size 10^{80} without special form during the present century." He was also shown to be too cautious in a few years. The choice of a 129D integer for the RSA challenge number in 1977 [Gardner] was apparently motivated by an estimate of Ron Rivest that to factor such an integer would require more than "40 quadrillion years" on a computer much faster than any that exist even today. However, this challenge integer was factored in 1994.

Appendix B: Historical factorization records

In 1964, there was practically no organized activity in factoring integers, and so this entry is based on the remarks in [BS].

The largest generally hard integer that had been factored by 1974 was F_7, the 7-th Fermat number, $2^{128} + 1$, which is 39D. (Today it would not be classified as "generally hard," since the special number field sieve handles numbers of this type much more efficiently than general ones. Also, as another technical point, the integer that was factored was $257 \cdot F_7$, since the use of a multiplier speeded up the algorithm.) F_7 had been factored in 1970 by Mike Morrison and John Brillhart, but their paper describing this work was only published in the 1975 D. H. Lehmer special issue of *Math.Comp.* In those days, integer factorization was not fashionable, and there was not much interest in going after records. Therefore inspection of the published literature is not adequate to assess the state of the art. Experts who were active in this area in the 1970s say that they thought that numbers of 45D were doable with the algorithms and machines available then, and so 45D is the figure used here.

1984: This is the Sandia factorization of Jim Davis, Diane Holdridge, and Gus Simmons.

1994: This is RSA129, the RSA challenge integer of 1977, which was factored by a worldwide collaborative effort organized and led by Derek Atkins, Michael Graff, Arjen Lenstra, and Paul Leyland [AGLL].

If we also consider integers of a special form, then the 162D integer $(12^{151} - 1)/11$ provides another record. It was factored in 1994 by Henk Boender, Joe Buhler, Scott Huddleston, Marije Huizing, Walter Lioen, Peter Montgomery, Herman te Riele, Robby Robson, Russell Ruby, and Dik Winter.

Appendix C. Computing power of historical factorizations

According to John Brillhart, integer factorizations in the early 1970s usually used up only about an hour of cpu time on the mainframes of those days, which are typically rated at 1-10 mips.

The 1984 Sandia factorization used 9.5 cpu hours on a Cray X-MP, which is on the order of 100 mips.

1994: This is the Atkins et al. estimate [AGLL] for the computation, which used the ppmpqs algorithm. Since then a 119D integer has been factored by Scott Contini, Bruce Dodson, Arjen Lenstra, and Peter Montgomery in about 250 MY using gnfs (the general number field sieve), suggesting that today the RSA129 factorization could be carried out in about 1000 MY instead of the 5000 MY that was used. See [DL] for details.

Appendix D. Current computing power

The oft-quoted figure of 30M users of the Internet is questionable, as it is obtained by assuming there are 10 users per computer. However, the estimate that about 3M machines of one kind or another are hooked up to the Internet seems much more solid. Since many of them are old, an average 10 mips rating seems reasonable for them.

There are well over 10^8 PCs in the world. Since several tens of millions of them already have the 486 chips, which are usually rated at tens of mips, the estimate of $3 \cdot 10^8$ mips is conservative. On the other hand, most of these machines are not easily usable for factoring, since they are not networked, do not have enough memory, do not have operating systems that allow for background jobs to run easily in idle time, etc. All these factors will change in a few years, but now it would be hard to harness a large fraction of all the PCs in a factoring project.

We might note that all the supercomputers in the world (about 10^3 in total) have a computing power of about $3 \cdot 10^6$ mips, with several sites having between 5% and 10% of of that capacity. (This estimate equates 1 megaflop with 1 mips, which is not very accurate, and is based on the June 1995 version of the TOP500 report [DongarraMS].) IBM mainframes shipped in 1994 (which was a record year in terms of mainframe computing power, although not in dollar volume of sales) amounted to only $2 \cdot 10^5$ mips.

Appendix E. Moore's "Law"

There is some skepticism whether this law will continue to hold much longer. Line widths will eventually be so small that entire new technologies might be needed, architectural improvements (speculative execution, etc.) might run into the exponential complexity blowup, and so on. Even if the bare technologies are not a barrier, economics might present one, since the costs of state of the art fabrication facilities are also escalating. However, such concerns are not new, and were already present a decade ago, and yet progress has continued unhindered. Thus it seems imprudent to assume Moore's "Law" will be violated any time soon. Since state of the art microprocessors are already running close to 500 mips, the projection that by 2004 the typical processor will have a rating of 1000 mips (compared to around 10 mips today) seems safe. Beyond that, there are bigger question marks. Even if raw processor speed continues to increase, memories might be more of a bottleneck. Since most fast factorization algorithms rely on substantial memories to perform sieving operations, they are often already limited by memory system performance more than by the processor. However, this is a problem that is afflicting an increasing range of problems. Therefore there are strong incentives towards dealing with it, and again it seems imprudent to assume it will form an insurmountable barrier.

Appendix F. Number of computers

Since there will be about 10^{10} people on Earth in 2014, a projection of 10^{11} computers implies that there will be over 10 computers per person. This may seem fanciful, but we should remember that there are many embedded microprocessors in everyday appliances such as cars, dishwashers, etc., and they will be getting increasingly powerful. To provide voice recognition capability for the coffee pot will require considerable processing power. (The TV set-top boxes being designed today have more computing power than the Cray-1, the first supercomputer. The new 32-bit video game machines, of which tens of millions are expected to be sold each year, will have similar power.) We might note, as a forerunner of what will be common, that there were two fax machines among the approximately 1600 processors in the RSA129 project.

There is still a question, even if we assume that there will be many computers around, of whether they will all be networked together, and whether their computing power will be easily accessible. Will there be computing-power brokers, selling time on millions of machines? Will people be willing to tolerate somebody else running jobs on their machines, even in spare time?

Appendix G. Running time of algorithms

In a variant of the standard notation, we define

$$L[n, v, a] = \exp(a \cdot (\log n)^v \cdot (\log \log n)^{(1-v)}),$$

where $\log n$ refers to the natural logarithm of n.

The heuristic running time (there are no rigorous proofs, but experience and heuristics support the calculations) of the gnfs to factor an integer n is

$$L[n, 1/3, c_0 + o(1)] \quad \text{as} \quad n \to \infty,$$

where $c_0 = (64/9)^{1/3} = 1.9229\ldots$. There is also a variant, due to Don Coppersmith, which does not seem to be practical, that allows the replacement of c_0 by $c_1 = 1.9018\ldots$. See [LL, Pomerance] for presentations of the gnfs.

The special number field sieve, which factors efficiently integers of the form $a^k \pm b$, for example, where a and b are small, and k large (k can also be small, but then has to fall into certain ranges) has running time

$$L[n, 1/3, c_2 + o(1)] \quad \text{as} \quad n \to \infty,$$

where $c_2 = (32/9)^{1/3} = 1.5262\ldots$.

Previous methods, such as variants of the quadratic sieve algorithm, have running times of the form

$$L[n, 1/2, 1 + o(1)] \quad \text{as} \quad n \to \infty.$$

The continued fraction method, which was the most widely used method in the 1970s, appears (at least for the most common variant) to have running time

$$L[n, 1/2, c_3 + o(1)] \quad \text{as} \quad n \to \infty,$$

where $c_3 = 2^{1/2} = 1.4142\ldots$.

Appendix H. Comparison of running times of algorithms

The $o(1)$ terms in the estimates of running times of factorization algorithms are usually not computed explicitly. Instead, to estimate how long an algorithm with asymptotic running time $L[n, v, a + o(1)]$ should take to factor an integer n, one takes the observed running time X on an integer m and computes $X \cdot L[n, v, a]/L[m, v, a]$. This has worked well in practice. This method does ignore various important practical aspects, such as the need for memory and communication capacity as well as cpu cycles, but those have been overcome in the past either through better algorithms or general technological improvements, so it seems reasonable to continue to ignore them.

Appendix I. Exponential algorithms

Public key cryptosystems such as RSA and DSA require use of large moduli because known general algorithms for factoring integers and computing discrete logarithms are subexponential, and so hardware improvements by themselves lead to substantial progress. In addition, there have been steady and rapid improvements in algorithms. On the other hand, there are some problems in which the best algorithms known are exponential, and where there has been no recent algorithmic improvement. We cite here as an example attacks on DSA, the US Digital Signature Algorithm, that do not use the structure of the multiplicative group of integers modulo the large basic prime p. The DSA uses discrete exponentiation modulo a prime p, but the exponents are computed modulo a prime q such that q divides $p - 1$. (This is the Schnorr method that speeds up the algorithm.) Therefore all the integers are inside an abelian group of order q. To break DSA, one can either solve the general discrete log problem modulo p, which can be done using a variant of the number field sieve, or else work inside the group of order q. The best algorithms for the second attack are those of Dan Shanks and John Pollard, both of which take about $\sqrt{q}$ operations. Since these $\sqrt{q}$ operations involve multiplications modulo p, though, even for q of 160 bits and p of 1024 bits or more, it would take at least 10^{14} MY on general purpose computers to implement either the Shanks or the Pollard algorithm. Hence we can conclude that 160 bits for q is likely to be safe for at least 20 years, and 200 bits (which would require at least 10^6 as much computing power to break) for much longer. (As with all the other projections about algorithms, this one could turn out to be faulty if a breakthrough occurs.)

As another example where only exponential attacks are known, we can cite elliptic curve cryptosystems [Menezes], proposed initially by Neal Koblitz and Victor Miller. If the elliptic curve is chosen carefully, only the Shanks and Pollard methods are known for computing the analog of discrete logs on these curves. There is still some reluctance to use elliptic curve cryptosystems, though, since they have not been scrutinized as carefully as integer factorization and ordinary discrete logs.

Appendix J. Attacks on DES:

For comparison, we present some estimates of the computing power needed to break DES. We consider known plaintext attacks on cipher codebook mode. We assume that only exhaustive key search will be tried, so that on average 2^{55} keys will have to be tested to find the right one. The best software implementations (written in C) appear to achieve rates of up to about 200 KB/sec on 25 mips machines (such as 50 MHz 80486 PCs), which (since each iteration of DES involves encrypting 8 bytes) corresponds to 25,000 encryptions per second, or about 1,000 encryptions per second on a 1 mips computer. Hence 1 MY allows us to test about $3 \cdot 10^{10}$

encryptions. Therefore to find a single DES key will on average take $1.2 \cdot 10^6$ MY, or about 300 times as much as the factorization of RSA129 required.

DES was made to run fast in hardware, and special purpose machines can provide substantial assistance in breaking it. Michael Wiener [Wiener] has proposed a pipelined parallel computer that could be built for about $1.5M (both development and construction cost) and would find a single DES key in about 4 hours. What this shows is that special purpose hardware can be of great help in breaking DES. On the other hand, general integer factorization and discrete logarithm algorithms do not benefit that much from special designs, and it is the threat of the free computing power on the Internet that seems most serious. Special designs for sieve processors for factoring have been proposed (see [PST]), but the economics of the electronics industry favors general purpose computers. (Fast parallel modular multiplication units could be of use in the implementation of the Pollard and Shanks exponential-time algorithms, though, or of the subexponential-time elliptic curve method.)