ON THE POWER OF CASCADE CIPHERS

by

S. Even* and O. Goldreich

Technical Report #275

May 1983

* The research was supported by the Fund for the Promotion of Research at the Technion.
1. **Introduction**

A Cascade Cipher (CC) is defined as a concatenation of block cipher systems (hereafter referred to as its **stages**) such that the plaintext of the CC plays the role of the plaintext of the first stage, the ciphertext of the i-th stage is the plaintext of the (i+1)-st stage and the ciphertext of the last stage is the ciphertext of the CC. When operating in encryption [decryption] mode the plaintext [ciphertext] of the CC is fed into the first [last] stage which, as well as all other stages, operates in its encryption [decryption] mode. The output of the i-th stage is fed into the (i+1)-st [(i-1)-st] stage and the output of the last [first] stage is the ciphertext [plaintext] of the CC (see Fig. 1). Unless otherwise stated, we will denote by \( n \) the number of bits in the plaintext (ciphertext), by \( k \) the length of the key for each of the stages and by \( \ell \) the encryption mode (using the key \( K_1 \cdot K_2 \cdots K_{\ell-1} \cdot K_\ell \))

![Encryption Mode Diagram](image1)

The decryption mode (using the key \( K_1 \cdot K_2 \cdots K_{\ell-1} \cdot K_\ell \))

![Decryption Mode Diagram](image2)

**Figure 1:** The operation modes of a CC
number of stages. We will discuss both Cascades of Identical Ciphers (CIC), in which all stages are identical, and General Cascades of Ciphers (GCC), in which the stages may differ.

The problem of breaking a given cipher system, C, is defined as finding a permutation \( \sigma \) which can be performed by \( C \), such that \( \sigma(M_i) = C_i \) for \( 1 \leq i \leq t \), where \( \{(M_i, C_i)\}_{i=1}^{t} \) is given (i.e., known-plaintext attack).

Diffie and Hellman [1] showed that a CC consisting of two DES's [2] can be broken using \( 2^{56} \), 56-bit-long, words of memory and \( 2^{56} \) time units. Merkle and Hellman [3] showed that a restricted class of 3 stages CIC's, in which the same key is used for the first and last stage, can also be broken using \( 2^{56} \) space and \( 2^{56} \) time.

In Section 2 we present an algorithm, with a time-space tradeoff, for breaking a given CC. For every \( 2^{0.5k} \leq T \leq 2^k \) the algorithm can run in time \( T \) and space \( S \) such that \( \log_2 T \cdot S \approx k \). This algorithm establishes an exponential upper bound, in \( k \), for the time-complexity of breaking a CC. In Section 3 we show that a non-polynomial lower bound on the time-complexity (in \( k \)) for a certain class of instances of this problem will imply that \( P \neq NP \). Thus, we despair of proving that breaking a CC is exponentially hard (in \( k \)). We also prove a stronger result: the NP-Completeness of a restricted (class of instances of the) breaking problem. This result suggests that the use of randomized algorithms or probabilistic arguments, to solve this problem, might be more fruitful than the use of deterministic ones.

In Section 4, we prove that a 2-stage CC is not weaker than each of its stages.

The problem of determining a cipher, which consists of some known interconnection of a priori unknown cipher-boxes, is defined as acquiring
data which allows the efficient computation of the permutation, performed by the cipher when the key to the cipher (i.e. the keys to the boxes) is given. An experiment is defined as feeding known plaintext and key to the cipher and reading the output (i.e. the ciphertext).

An exhaustive experiment is defined as performing all (but one) possible experiments with a fixed (known) key. Obviously, any cipher can be determined by executing all possible exhaustive experiments, but in certain cases a subset of all experiments suffices to determine the cipher.

In Section 5 we show that $\lambda \cdot 2^k$ exhaustive experiments suffice to determine a GCC, the stages of which are random ciphers. We conclude that the conjectured strength (i.e. the exponential lower bound on the breaking problem) of the CC class cannot be based on the difficulty (i.e. the number of experiments) of determining its ciphers. It follows that an information-theoretic approach to prove the conjectures, is hopeless. We also show that the number of exhaustive experiments allowing the determination of a GCC does not exceed the number of exhaustive experiments which suffices to determine a CIC (which consists of copies of the same random cipher).

In Section 6 we prove that, with very high probability, a GCC consisting of random ciphers realizes $2^{\frac{\lambda k}{2}}$ different permutations. We also show that if $\{M_i\}_{i=1}^t$ is a given set of plaintexts and $t \cdot n > 2^\lambda \cdot k$, then, with very high probability, there is no pair $(\sigma_1, \sigma_2)$ of different permutations realizable by the CC such that for every $1 \leq i \leq t$ $\sigma_1(M_1) = \sigma_2(M_2)$.
2. Time-Space Trade-off for the Exhaustive Breaking of Cascade Ciphers

Let \( \sigma \) be a permutation that can be performed by a CC and
\( \{(M_i, \sigma(M_i))\}_{i=1}^{t} \) is known. We want to find a permutation \( \sigma' \) performable by the CC such that for every \( 1 \leq i \leq t \) \( \sigma'(M_i) = \sigma(M_i) \). In Section 6 we show that if the stages are random ciphers, and \( t \cdot n > 2^{t} \cdot k \), then, with very high probability, the only \( \sigma' \) which meets these terms is \( \sigma \). We will assume throughout Sections 2 and 3 that \( t \cdot n > 2^{t} \cdot k \).

It is straightforward to break a CC using time \( t \cdot n \cdot 2^{t} \cdot k \) and space \( \ell \cdot k \). (Just try all possible keys.) Diffie and Hellman [1] showed how to break a CC consisting of two stages (i.e. \( \ell = 2 \)) using time \( n \cdot k \cdot 2^{k} \) and space \( S = n \cdot 2^{k} \). The method uses a meet-in-the-middle technique (building one table of all encryptions performed on \( M_1 \) by the first stage, using all possible keys; building a second table of all decryptions performed on \( \sigma(M_1) \), by the second stage, using all possible keys; and then merging the tables (a merge by sort, requiring time \( S \log_2 S \), can be used) in order to find collisions, which are called candidate solutions. The solution is then obtained by testing all candidate solutions with additional plaintext-ciphertext pairs. This approach can be generalized to break a CC which consists of \( 2^{t} \) stages using time \( t \cdot n \cdot 2^{t} \cdot k \cdot 2^{2^{t} \cdot k} \) and space \( t \cdot n \cdot 2^{t} \cdot k \). In fact a more general result, which displays a time-space tradeoff is given in Theorem 1. (This result allows breaking a CC consisting of two DES's [2] using space \( 2^{41} \) and time \( 2^{71} \). Note that this is more likely to be feasible than either \( 2^{112} \) time, required in the straightforward method, or \( 2^{56} \) space, required in Diffie and Hellman's method.)
Theorem 1:

For every $q$, $0 \leq q \leq \ell' \cdot k$, any given CC which consists of $\ell \cdot 2\ell' \cdot k$ stages can be broken using time $T$ and space $S$ such that $\log_2 T \approx \ell' \cdot k + q$ and $\log_2 S \approx \ell' \cdot k - q$.

Proof:

Divide the CC into two portions consisting of $\ell'$ stages each. Denote by $FP$ the CC consisting of the first $\ell'$ stages and by $LP$ the CC which consists of the $\ell'$ last ones. Choose a q-vector $FV [LV]$ the elements of which are different and belong to $\{1, 2, \ldots, \ell' \cdot k\}$. Let $V_q$ denote the set of all binary q-vectors. Execute the following procedure:

for every $u, v \in V_q$ do begin
(1) Construct a table, the entries of which are pairs such that the first element of a pair is key $FK \in \{K \in V_{\ell' \cdot k}: (\forall i, 1 \leq i \leq q)(K(FV(i)) = u(i))\}$ and the second element is $(N_1^{(FK)}, N_2^{(FK)}, \ldots, N_t^{(FK)})$, where $N_j^{(FK)}$ is the encryption of $M_j$ by $FP$ using the key $FK$. (Note that there are $2^{\ell' \cdot k-q}$ entries in the table.) Sort the table according to the second element of its entries.

(2) Construct a table, the entries of which are pairs such that the first element of a pair is a key $LK \in \{K \in V_{\ell' \cdot k}: (\forall i, 1 \leq i \leq q)(K(LV(i)) = v(i))\}$ and the second element is $(L_1^{(LK)}, L_2^{(LK)}, \ldots, L_t^{(LK)})$, where $L_j^{(LK)}$ is the decryption of $\sigma(M_j)$ by $LP$ using the key $LK$. Sort the table according to the second element of its entries.
(3) Merge the tables constructed in (1) and (2).

If a collision occurs then:

print the entries which correspond to it and stop.

end

Remark: One does not have to use all \( t \) plaintext-ciphertext pairs available; about \( t' = (2^k/n) \) pairs suffice. Note that the procedure requires space \( 2 \cdot t' \cdot n \cdot 2^{k-q} \) and time bounded by

\[
(2^q)^2 \cdot 2 \cdot t' \cdot n \cdot (2^k-q+1) \cdot 2^{k-q} \leq 2 \cdot t' \cdot n (2^k+1) \cdot 2^{k+q}.
\]

Q.E.D.

Theorem 2:

For every \( q' \), \( 0 \leq q' \leq 2^k \), any given CC of \( \ell \leq 2^{k+1} \geq 3 \) stages can be broken using time \( T \) and space \( S \) such that

\[
\log_2 T \approx (2^{k+1})k + q' \quad \text{and} \quad \log_2 S \approx 2^k - q'.
\]

Proof:

Similar to the proof of Theorem 1, dividing the CC to a first portion of \( \ell' \) stages and a last portion of \( \ell' + 1 \) stages and choosing FV of length \( q' \) and LV of length \( q' + k \).

Q.E.D.
3. On the Unlikelihood of Proving Certain Lower Bounds for the CC Breaking Problem

The results of the previous section suggest that $O(2^{0.5 \ell k})$ is an upper bound on the time-complexity of breaking a CC. We believe that the problem has a non-polynomial lower bound (in $\ell$). To formulate our conjecture, we restrict our attention to a subset, $B_{p(\ell), f(\ell)}$, of instances of the CC-breaking problem, where $p(\ell)$ is a polynomial and $f(\ell)$ is a function which grows faster than any polynomial. An instance which consists of a CC, $F$, and a set of $t$ plaintext-ciphertext pairs, belongs to $B_{p(\ell), f(\ell)}$ if it satisfies the following four conditions:

1. $F$ is a GC the $\ell$ stages of which are random ciphers (each stage has $n$ input bits and $k$ key bits).
2. $2^k < p(\ell)$.
3. $(2^n)^t > f(\ell)$.
4. $(2k\ell/n) \leq t < p(\ell)$.

Let $B'_{p(\ell), f(\ell)}$ be a subset of instances of $B_{p(\ell), f(\ell)}$ which satisfy $2^n < p(\ell)$.

Conjecture A [A']:

There is a non-polynomial lower bound (in $\ell$) on the time-complexity of the CC-breaking problem, for instances which belong to $B'_{p(\ell), f(\ell)}$.

Note that Conjecture A [A'] can be formulated using any notion of time-complexity (e.g. worst case, average case, "most" cases etc.).
Let us explain the reasons we have restricted our attention to the \( B_p(o), f(o) \) class. First note that if \( 2^k \) is not bounded from above by a polynomial in \( \ell \) then the conjecture becomes trivial and uninteresting, since even breaking one of the GCC's stages cannot be done in polynomial time. On the other hand, note that if \( \ell^n \) is bounded from above by a polynomial in \( \ell \) then the conjecture is false. Also note that if \( t \) is not bounded from above by a polynomial in \( \ell \) then the conjecture becomes uninteresting, since even checking a solution is not polynomial in \( \ell \). The assumption \( 2k\ell \leq nt \) was made in Section 2.

**Theorem 3:**

Conjecture A', even when formulated using worst-case notion of time-complexity, yields \( P \neq NP \).

**Proof:**

Conjecture A' states that a specific problem is not polynomial in \( \ell \). However, the input to this problem is polynomial in \( \ell \) and a solution to it can be guessed and verified in polynomial time. Thus, the problem is in NP but not in \( P \).

Q.E.D.

**Corollary 1:**

Proving Conjecture A' is not easier than proving that \( P \neq NP \).

We believe that any proof method which proves Conjecture A is also adequate to prove Conjecture A'. Thus, we despair of proving any of these conjectures.
Next we strengthen the result of Theorem 3, by introducing a decision problem which corresponds to the breaking problem and proving that it is NP-complete. The problem, denoted DBP (Decision Breaking-Problem) is defined as follows:

Given an ordered set \( \{S_i\}_{i=1}^\ell \) of permutation sets over \( V_n \) (*) and a set of \( t \) pairs \( \{(M_i, C_i); M_i, C_i \in V_n\}_{i=1}^t \), determine whether there exists a permutation \( \sigma \in S_\ell \cdots S_2 \cdot S_1 \) (**), such that \( \sigma(M_i) = C_i \) for \( 1 \leq i \leq t \).

**Theorem 4:**

DBP is NP-Complete even if \( \ell \geq |S_i| = \ell \cdot 2^n \), \( 2\ell \leq t < 3\ell \) and \( \ell < 2^n < 5 \cdot \ell \).

The proof is given in Appendix A.

(Note that the DBP instances in Theorem 4 correspond to instances of the CC-breaking problem in which \( \ell < 2^n = 2^k < 5 \cdot \ell \) and \( 2\ell \leq t < 3\ell \); i.e. which are in \( B_{p_o(\cdot),f_o(\cdot)} \) where \( p_o(x) = 5 \cdot x \) and \( f_o(x) = x! \).)

**Corollary 2:**

If \( P \neq NP \) then both Conjectures A and A', when formulated using the worst-case complexity notion, are valid.

This corollary, however, has little cryptological significance, since worst-case resistance to cryptoanalysis is of little value.

---

(*) \( V_n \) is the set of binary vectors of length \( n \).

(**) \( A \triangleq \{\sigma \tau; \sigma \in A_n, \tau \in B\} \). By \( \sigma \tau \) we mean the permutation which results from the successive application of \( \tau \) and \( \sigma \).
4. A Cryptosystem which is Provably Not Easier to Break than Two Other Cryptosystems

In this section we will prove that a CC of two block-cipher systems is not easier to break than each of its stages. In order to make the result more interesting, we relax the definition of breaking as follows:

Given a cryptogram \( y \), which was produced from some unknown message \( x \) using some secret key \( K \), find \( x \), assuming one can get answers to the following queries:

(1) What is the cryptogram which corresponds to \( M \)? (Here \( M \) is arbitrary and the key is \( K \).)

(2) For \( C \neq y \), what is the message which corresponds to the cryptogram \( C \)? (Again the key is \( K \).)

We will refer to the above problem by \( \text{M-breaking} \).

Denotations:

Let \( F \) and \( G \) be two block-cipher systems, each with \( n \) input bits and \( k \) key bits. Denote by \((GF)\) the 2-stage CC, the first stage of which is \( F \) and the second is \( G \). The key \( K \) of the cryptosystem \((GF)\) consists of the key \( K' \) of \( F \) and the key \( K'' \) of \( G \) (i.e. \( K = (K', K'') \)). The key for \((GF)\) is chosen by independently choosing a key for \( F \) and a key for \( G \) (i.e. the probability that \( K \) is chosen is the product of the probability that \( K' \) is chosen by the probability that \( K'' \) is chosen). We denote by \( X_y(z) \) [\( X_y^{-1}(z) \)] the encryption [decryption] of \( z \) by the cryptosystem \( X \) using the key \( y \). Note that \((GF)_K(z) = G_{K''}(F_{K'}(z))\) and \((GF)_K^{-1}(z) = F_{K'}^{-1}(G_{K''}^{-1}(z))\)
Theorem 5:

(GF) is not easier to M-break than F [G].

Proof:

Let $A$ be an algorithm for M-breaking (GF) (i.e., obtaining $x$ from $(GF)_K(x)$), whose input-output behavior is as follows:

Input: The encryption of the message $x$ (i.e., $(GF)_K(x)$).

Queries: (Initiated by the algorithm during computation):

- What are the encryptions of $M_1, M_2, \ldots, M_r$ by (GF) using key $K$?
- What are the decryption of $C_1, C_2, \ldots, C_s$ by (GF) using key $K$, provided $C_i \neq (GF)_K(x)$ for $1 \leq i \leq s$?

Output: The message $x$ (i.e., the decryption of $(GF)_K(x)$ by (GF) using the key $K$).

We present an algorithm, denoted $A_F[A_G]$, for M-breaking the cryptosystems $F[G]$ (i.e., obtaining $x'$ from $F_{K'}(x')$ and $G_{K''}(x'')$). First let us specify the input-output behavior of $A_F[A_G]$.

Input: $F_{K'}(x')$ and $G_{K''}(x'')$.

Queries: What are the encryptions of $M_1, M_2, \ldots, M_r$ by $F[G]$ using the key $K'$?

- What are the decryptions of $C_1, C_2, \ldots, C_s$ by $F[G]$, using the key $K'[K'']$, provided $C_i \neq F_{K'}(x')$ and $C_i \neq G_{K''}(x'')$ for $1 \leq i \leq s$?

Output: The message $x'[x'']$.

$A_F[A_G]$ chooses, randomly, a key $K''[K']$ for the cryptosystem $G[F]$. It provokes $A$ to find the decryption of $G_{K''}(F_{K'}(x'))$ by the cryptosystem (GF), using the key $K = (K', K'')$, when $K'$ is the key to $F$ and $K''$ is the key to $G$. (Note that $K'[K'']$ is not known to $A_F[A_G]$ and both $K'$ and $K''$ are chosen randomly.)
and \( K'' \) are not known to \( A \).) To answer \( A \)'s query for the encryption of \( M_i \) by (GF) using key \( K \), \( A_F \) asks for the encryption of \( M_i \oplus M_i \) by \( F \) using key \( K' \) and applies \( G_{K''} \) to it \([A_G \text{ asks for the decryption of } M_i \oplus F_{K'}(M_i) \text{ by } G \text{ using key } K'']\). To answer \( A \)'s query for the decryption of \( C_i \neq G_{K''}(F_{K'}(x')) \) \([C_i \neq G_{K''}(x'')] \) by (GF) using key \( K \), \( A_F \) asks for the decryption of \( C_i \oplus G_{K''}^{-1}(C_i) \neq G_{K''}^{-1}(G_{K''}(F_{K'}(x'))) \) by \( F \) using key \( K' \) \([A_G \text{ asks for the decryption of } C_i \oplus C_i \neq G_{K''}(x'') \text{ by } G \text{ using key } K'' \) and applies \( F_{K'}^{-1} \) to it]. \( A \) outputs \( (GF)_K^{-1}(G_{K''}(F_{K'}(x'))) = x' \) and \( A_F \) terminates by outputting it \([A \text{ outputs } (GF)_K^{-1}(G_{K''}(x'')) = F_{K'}^{-1}(x''), A_G \text{ applies } F_{K'} \text{ to it and terminates by outputting the result, } x''\)].

Note that by any "reasonable" complexity measure, the complexity of \( A_F[A_G] \) is of the "order" of \( A \)'s complexity.

**Q.E.D.**

**Remarks:**

(1) One can use similar methods to prove that (GF) is not easier to break than either \( F \) or \( G \), (here breaking is as defined in Section 1).

(2) Define KM-breaking as obtaining a message given the corresponding cryptogram and plaintext-ciphertext pairs which correspond with the same key (i.e. know plaintext attack). (Note that this problem differs from breaking in which the corresponding permutation should be found.) One can use similar methods to prove that KM-breaking (GF) is not easier than KM-breaking either \( F \) or \( G \).

(3) Define PM-breaking a cryptosystem as applying a probabilistic method in order to M-break the cryptosystem such that the probability
that the method succeeds in finding the message is not negligible.

One can prove that PM-breaking (GF) is neither easier nor more
successful than PM-breaking either F or G. (Given a probabil-
istic algorithm, \( A \), for PM-breaking (GF) one can use the construction
presented in the proof of Theorem 5 to obtain an algorithm, \( A_F[A_G] \),
for PM-breaking \( F[G] \), such that \( A_F[A_G] \) has the same probabilistic
properties as \( A \).)

(4) Theorem 5 holds even if \( F \) and \( G \) are Public-Key Cryptosystems [4],
i.e. their decryption keys differ and are claimed to be hard to
compute from their encryption keys.

(5) Using similar techniques one can prove that the algorithm presented
by Asmuth and Blakley [5] produces cryptosystems which are not
easier to M-break [KM-break] than their components. Note that Rivest
and Sherman [6] considered the formal setting of such a statement
and its proof an open question. (The reason that Asmuth and Blakley
could not prove a similar statement rigorously is that they have not
defined rigorously the "cracking" problem.)
5. Determining a Cascade Cipher

[In this section, \( w_1 = w_2 \) means that the words \( w_1 \) and \( w_2 \) are equal (letter by letter); while \( w_1 \equiv w_2 \) means that the words \( w_1 \) and \( w_2 \) consist of sequences of operators which are equivalent (i.e., have the same effect)].

The problem of determining a CC, the stages of which are random ciphers, is introduced in Section 1. We show that although such a cipher realizes (as shown in Section 6) \( 2^k \cdot 2^k \) permutations, \( \lambda \cdot 2^k \) exhaustive experiments suffice to determine the CC.

Theorem 6:

Let \( S_\lambda \subseteq \text{Sym}(\mathbb{D}) \) be an ordered set of \( m_j \) permutations, \( 1 \leq j \leq \ell \). Let
\[
\sigma(i_\lambda, \ldots, i_2, i_1) = x_1^{(i_\lambda)} \circ \cdots \circ x_2^{(i_2)} \circ x_1^{(i_1)},
\]
where \( x^{(j)} \) is the \( i_j \)-th permutation of \( S_j \). There is a set of \( \left( \sum_{j=1}^{\ell} (m_j-1) + 1 \right) \sigma \)'s such that every \( \sigma \) can be computed from it.

Proof:

Note that for every \( i_1, i_2, \ldots, i_\lambda \) such that \( 1 \leq i_j \leq m_j \), for \( 1 \leq j \leq \lambda \),
\[
\sigma(i_\lambda, \ldots, i_2, i_1) = \sigma(i_\lambda, 1, \ldots, 1, 1) \circ \sigma(1, 1, \ldots, 1, 1) \circ \sigma(1, m_\lambda - 1, 1, \ldots, 1) \cdots
\]
\[
\sigma(1, \ldots, 1, 1, 1) \circ \sigma(1, \ldots, 1, i_2, 1) \circ \sigma(1, \ldots, 1, 1, 1) \circ \cdots \circ \sigma(1, \ldots, 1, i_1, 1, 1) \cdot
\]
[This can be easily verified by using the \( \sigma \)'s definitions.] Thus all the \( \sigma \)'s can be computed from
\[
\bigcup_{j=1}^{\ell} \{ \sigma(u_\lambda, \ldots, u_2, u_1) : 1 \leq u_j \leq m_j \text{ and } (\forall k, 1 \leq k \leq \lambda, \ k \neq j \) \ (u_k = 1) \}.
\]

Q.E.D.
Corollary 3:

Determining the permutations of a CC, which consists of \( \ell \) stages (each a random cipher with key length \( k \)), does not require more experiments than determining the permutations of a random cipher with key length \( k + \log_2 \ell \).

Corollary 3 implies that it is impossible to prove Conjecture A through information theoretical arguments. (Note that a polynomial (in \( \ell \)) number of exhaustive experiments suffices to determine a CC which satisfies the condition of the conjecture. On the other hand note that a non-polynomial lower bound (in \( \ell \)) for the number of exhaustive experiments needed to determine such a CC would have implied the validity of the conjecture.)

Denote by \( G(m, \ell) \) the following set of permutations
\[
\{ \sigma(i_{\ell}', \ldots, i_2, i_1) \cdot x_{i_{\ell}}^{(1)} \cdots x_{i_2}^{(2)} \cdot x_{i_1}^{(1)} : x_{i_{j}}^{(1)} \in S_j \text{ for } 1 \leq j \leq \ell \},
\]
where \( S_j \) is an ordered set of \( m \) permutations over the set \( D \). Denote by \( I(m, \ell) \) the following set of permutations
\[
\{ \sigma(i_{\ell}', \ldots, i_2, i_1) \cdot x_{i_{\ell}} \cdots x_{i_2} \cdot x_{i_1} : x_{i_{j}} \in S \text{ for } 1 \leq j \leq \ell \},
\]
where \( S \) is an ordered set of \( m \) permutations over \( D \).

Denote \( V \supset \{(i_{\ell}', \ldots, i_2, i_1) : 1 \leq i_j \leq m \text{ for } 1 \leq j \leq \ell \} \). We say that a subset, \( P \), of \( V \) allows the preset determination of all the \( \sigma \)'s of \( G(m, \ell) \) \( I(m, \ell) \), if for every \( u \in V \), \( \sigma_u \equiv \sigma_{u_{\ell}} \cdots \sigma_{u_2} \sigma_{u_1} \) where \( \sigma_j \in \{1, -1\} \) and \( v_j \in P \) for \( 1 \leq j \leq t \) (i.e. there is a fixed, predetermined expression for the calculation of each of the \( \sigma \)'s by \( \sigma \)'s with index-vector in \( P \).

Note that these expressions are independent of the value of the \( S_j \)'s and only depend on \( m \) and \( \ell \).

Note that the proof of Theorem 6 implies that:
Corollary 4:

There exists a set \( P \), \(|P| < m^\lambda \), which allows the preset determination of all \( \sigma \)'s defined by \( G(m, \lambda) \).

Clearly a set which allows the preset determination of all \( \sigma \)'s defined by \( G(m, \lambda) \), also allows the preset determination of all \( \sigma \)'s defined by \( I(m, \lambda) \). We will show that also the converse statement holds, namely:

Theorem 7:

Let \( P \) be a subset of \( V \) which allows the preset determination of all \( \sigma \)'s defined by \( I(m, \lambda) \). \( V \) (also) allows the preset determination of all \( \sigma \)'s defined by \( G(m, \lambda) \).

Proof:

Let \( E \) be a set of minimal length expressions for the computation of the \( \sigma \)'s, of \( I(m, \lambda) \), by the elements of \( \{ \sigma_v : v \in P \} \).

Consider the expression (in \( E \)) for the computation of \( \sigma_{(i_1, \ldots, i_2, i_1)} \), where \( 1 \leq i_1, i_2 \leq m \):

\[
\sigma_{(i_1, \ldots, i_2, i_1)} = \sigma_{(i_t, \ldots, i_t, 2, i_t, 1)} \cdot \sigma_{(i_2, \ldots, i_2, 2, i_2, 1)}
\]

where for every \( 1 \leq j \leq t \)

\[
(i_j, \ldots, i_j, 2, i_j, 1) \in P \quad \text{and} \quad \sigma_j \in \{ 1, -1 \}.
\]

By replacing in (1) each \( \sigma \) by (the r.h.s. of) its definition in \( I(m, \lambda) \), we get:

\[
{x_i}_1 \cdots {x_i}_2 \cdots {x_i}_1 \equiv (x_i)_{i_t, \ldots, i_t, 2, i_t, 1} \cdot (x_i)_{i_2, \ldots, i_2, 2, i_2, 1}.
\]
Note that equation (2) must hold for all values of the $x_i$'s. The r.h.s. of equation (2) can be simplified by replacing $(x_{i,j}^{l} \ldots x_{i,j}^{n} x_{i,j}^{n+1})^{\alpha_j}$ by $x_{i,j}^{l} \ldots x_{i,j}^{n} x_{i,j}^{n+1}$ if $\alpha_j = 1$, and by $x_{i,j}^{l}^{-1} x_{i,j}^{l} \ldots x_{i,j}^{l}^{-1}$ if $\alpha_j = -1$. Denote the simplified version of the r.h.s. of (2) by $R$ and the l.h.s. of (2) by $L$.

Note that $R$ and $L$ are words over the alphabet $\{x_i, x_i^{-1}: 1 \leq i \leq m\}$ and that this alphabet is free of any relation other than those implied by the following set of cancellation rules:

$\{x_i x_i^{-1} = \lambda, x_i^{-1} x_i = \lambda: 1 \leq i \leq m\}$,

because the $x_i$'s denote arbitrary permutations.

Since the $x_i$'s are operators, we can, uniquely, define the reduced form of a word $w$, hereafter denoted $\overline{w}$, as follows: $\overline{w}$ is the word which results from $w$ when successively applying cancellation rules to it until no cancellation rule can be applied. For a more detailed discussion of this point see [7].

Note that $L = R$ (since $L \equiv R$) and $\overline{L} = R$ (since no cancellation rule can be applied to $L$). Thus, $L = \overline{R}$.

To prove that $E$ allows the computation of the $\sigma$'s defined by $G(m, \ell)$ we show that the equation

$$(2') \quad (x_{i,j}^{l} \ldots x_{i,j}^{n} x_{i,j}^{n+1})^{\alpha_j} = (x_{i,j}^{l} \ldots x_{i,j}^{n} x_{i,j}^{n+1})^{\alpha_l}$$

which results from equation (1) by replacing each $\sigma$ by (the r.h.s. of) its definition in $G(m, \ell)$, is valid. Let $L'$ denote the l.h.s. of (2') and $R'$ denote the simplified r.h.s. of (2').
Note that $L'$ and $R'$ are words over the alphabet 
$$\{x_i^{(j)}, (x_i^{(j)})^{-1} : 1 \leq i \leq m, \ 1 \leq j \leq \ell \}$$
and that this alphabet is free of any relation other than those implied by the following set of cancellation rules:

$$\{x_i^{(j)} \cdot (x_i^{(j)})^{-1} \equiv \lambda \ , \ (x_i^{(j)})^{-1} \cdot x_i^{(j)} \equiv \lambda : 1 \leq i \leq m, \ 1 \leq j \leq \ell \}.$$

(Note that the reduced form of words over this alphabet is defined similarly to the reduced form of words over \(\{x_i, x_i^{-1} : 1 \leq i \leq m\}\).)

The following Lemmas (proven in Appendix B) yield that $L' \equiv R'$ and thus establish the validity of equation (2'):

Lemma 1:

If a cancellation rule is applied between variables in $R$ the indices of which are $p,q$ and $r,s$, then $q = s$.

(The Lemma implies that if the cancellation rule $x_i^{\alpha} \cdot x_i^{-\alpha} \equiv \lambda$ is applied in the reduction of $R$, then $(x_i^{(q)}) \cdot (x_i^{(s)}) \equiv \lambda$ can be applied to reduce $R'$.)

Lemma 2:

If $\overline{R} = x_{i_j}^{k_l} \cdots x_{i_j}^{k_2} \cdot x_{i_j}^{k_1}$ then $k_q = q$ for $1 \leq q \leq \ell$.

(Note that $L = \overline{R}$ implies that the condition of Lemma 2 is satisfied. Thus, Lemma 1 and 2 yield $\overline{R}' = x_{i_j}^{(q)} \cdots x_{i_j}^{(2)} \cdot x_{i_j}^{(1)}$. Since $L = \overline{R}$, $i_{j_{q'}} = i$ for $1 \leq q \leq \ell$. Thus $L' = \overline{R}'$ and $L' \equiv R'$ follow.)

Since equation (2') is valid, equation (1) is valid also for $\sigma$'s of $G(m,\ell)$. Thus $P$ allows the preset determination of all the $\sigma$'s of $G(m,\ell)$.

Q.E.D.
6. Enumeration Aspects of Cascade Ciphers

In Section 5 we stated that a GCC, of random ciphers (hereafter called: a RCC) realizes an exponential number, in $\ell$, of permutations. To prove this statement we present the following lemma, which is proven in Appendix C.

**Lemma 3:**

Let $D$ be an arbitrary set, $m \triangleq |D|!$ and $S$ an arbitrary set of $s$ permutations over $D$. If $\log_2 r \cdot s << \log_2 m$ then, with very high probability, a randomly chosen set, $R$, of $r$ permutations over $D$, satisfies $|R \cdot S| = r \cdot s$.

**Corollary 5:**

If $\ell k < 2^n$ then, with very high probability, a RCC realizes $2^{\ell k}$ different permutations. (We remind the reader that $\ell$ denotes the number of stages, while $k [n]$ denotes the number of plaintext [key] bits to each of the stages.)

In Section 2 we stated that if $t \cdot n > 2 \cdot \ell \cdot k$ then, with very high probability, $\sigma$ is the only permutation, realizable by a RCC, which transforms the plaintext $M_i$ to the ciphertext $\sigma(M_i)$, for $1 \leq i \leq t$. This statement follows from the following lemma, which is (also) proven in Appendix C.

**Lemma 4:**

Let $\{M_i\}_{i=1}^t$ be a set of plaintexts for a RCC. If $t \cdot n = 2 \cdot \ell \cdot k + \delta$, $\delta \gg 1$, $2^n! \gg 2^k$ and $t << 2^n$, then, with very high probability,
there are no two permutations, $\sigma$ and $\sigma'$, realizable by the RCC such that for every $1 \leq i \leq t$, $\sigma(M_i) = \sigma'(M_i)$.

Remark: Clearly, the Lemma holds even if $t \ll 2^n$ does not hold.
APPENDIX A: Proof of Theorem 4

The MGS (Minimum Generator Sequence) problem was defined and shown to be NP-hard by Even and Goldreich [8]. Its instance consists of a domain D, a set of permutations \( G \subset \text{Sym}(D) \), a target permutation \( P \in \text{Sym}(D) \) and an integer \( K \). The question is whether there exists a sequence of \( q \leq K \) permutations such that \( P = g_1 \cdots g_q \), where \( g_i \in G \), for \( 1 \leq j \leq q \).

We introduce a restricted version of this problem (hereafter denoted \( \text{RMGS} \)) and prove that it is NP-Complete. The RMGS is restricted by the assertions \( |G| = 2^\left[ \log_2 2.5 K \right] + 1 \) and \( C_1 \cdot |D| < K < C_2 \cdot |D| \), where \( C_1 \) and \( C_2 \) are integers such that \( 0 < 0.4 C_2 < 10 \). (For our purpose \( C_1 = 1 \) and \( C_2 = 3 \) will do.)

Lemma A1:

\( \text{RMGS} \) is NP-Complete.

Proof:

By reduction from the 3XC problem [9], along the lines of the proof of Theorem 1 in Even and Goldreich [8].

Given a 3XC instance \((U \Delta \{e_i\}_{i=1}^{3n}, S \Delta \{s_j\}_{j=1}^{m})\), we introduce the following RMGS instance:

Let \( n_1 = \left\lceil \sqrt{7C_1 n} \right\rceil + 2C_1 \), \( n_2 = n_1 + 1 \). (Note that \( n_1 \) and \( n_2 \) are relatively prime.)

Define \( D = \{1, 2, \ldots, 7n + n_1 + n_2\} \):

\[ G = \{g_j : \prod_{e_i \in S_j} (2i-1, 2i) : 1 \leq j \leq m \} \cup G' \cup \{g_0 : (7n + 1, \ldots, 7n + n_1) \cdot (7n' + n_1 + 1, \ldots, 7n + n_1 + n_2) \} \]

where \( G' \) is the set of all permutations that are not in \( G \) and \( G_0 \) is the set of all permutations that are not in \( G' \).

\[ G = \{g_j : \prod_{e_i \in S_j} (2i-1, 2i) : 1 \leq j \leq m \} \cup G' \cup \{g_0 : (7n + 1, \ldots, 7n + n_1) \cdot (7n' + n_1 + 1, \ldots, 7n + n_1 + n_2) \} \]
where $G'$ is a set of $(2^{\log_2 2.5K} - 1) - (m + 1)$ arbitrary permutations over $\text{Sym}(D)$, such that if $g \in G'$ and $g(i) \neq i$, then $6n < i \leq 7n$; (note that, for $1 \leq j \leq m$, $g_j$ permutes $2i-1$ with $2i$ iff $e_i \in S_j$. Also note that the order $^\star$ of $g_0$ is $n_1 \cdot n_2$.)

\[ P \triangleq g_0^{-1} \cdot \prod_{i=1}^{3n} (2i - 1, 2i); \]

\[ K \triangleq n + n_1 \cdot n_2 - 1. \]

Note that $C_1(7n + n_1 + n_2) < n + n_1 \cdot n_2 - 1$ and that if $7C_2 > 14C_1 + 1$ and $2C_2 > 4C_1 + 1$ (both follow from $C_1 \leq 0.4C_2$), then $n + n_1 \cdot n_2 - 1 < C_2(7n + n_1 + n_2)$.

The proof of the validity of the reduction resembles the original proof and is omitted. (Note that permutations of $G'$ are never used and that $g_0$ is used $n_1 \cdot n_2 - 1$ times.)

We complete the proof of Theorem 4 by reducing the RMGS problem to the DBP problem.

Given an RMGS (with $C_1 = 1$ and $C_2 = 3$) instance $(D, G, P, K)$, construct the following DBP instance:


define $\triangleq K$, $n \triangleq \lceil \log_2 2.5 \ell \rceil$ and $t \triangleq \lceil 2.5 \ell \rceil$. (Note that $|D| < K < 2^{\lceil \log_2 2.5K \rceil} = 2^n$.)

Let $D \triangleq \{0, 1, \ldots, |D| - 1\}$ and $\Pi$ be a permutation in $\text{Sym}(D)$, we simulate $\Pi$ by the permutation $\sigma_\Pi$ which is defined as follows:

\[ (*) \text{ } k \text{ is said to be the order of } \sigma \text{ if it is the minimum positive integer such that } \sigma^k \text{ is the identity permutation.} \]
\( \sigma_{\Pi} \) is a permutation in \( \text{Sym}(V_n) \) which transforms the binary vector 
\( \underline{x} = (x_{n-1}, \ldots, x_0) \) to the vector \( \underline{y} = (y_{n-1}, \ldots, y_0) \) if 
\( (\Pi(v(x)) = v(y) \quad \text{and} \quad v(x) < |D|) \quad \text{or} \quad (x = y \quad \text{and} \quad v(x) \geq |D|) \), where 
\( v((z_{n-1}, \ldots, z_0)) = \prod_{i=0}^{n-1} z_i^{2^i}. \) (The vector \( \underline{x} \) represents \( v(x) \in D \) if \( v(x) < |D| \) and is a dummy otherwise. \( \sigma_{\Pi} \) fixes all the dummies.)

\[ S_i = \{ \sigma : g \in G \} \cup \{ I \} \] for \( 1 \leq i \leq \ell \), where \( I \) denotes the identity permutation over \( V_n \). (Note that each of the \( S_i \)'s fully encodes \( G \).)

\[ H = \{ (x, \sigma_p(x)) : 0 \leq v(x) < t \}. \] (Note that 
\[ |D| < K < 2.5K = t \leq 2^\left\lfloor \log_2 2.5K \right\rfloor = 2^n. \] Also note that \( H \) fully encodes \( P \).)

Note that:

1. \( |S_i| = 2^\left\lfloor \log_2 2.5K \right\rfloor = 2^n \), for \( 1 \leq i \leq \ell \).
2. \( 2\ell \leq t < 3\ell \), since \( K = \ell \) and \( 2K \leq 2.5K = t < 3K \).
3. \( \ell < 2^\left\lfloor \log_2 2.5K \right\rfloor = 2^n < 5\ell \).

The validity of the reduction is straightforward. Note that the construction is polynomial in \( K \) and \( |G| \) and (since \( K < 3 \cdot |D| \)) is polynomial in the length of the input to the RMGS instance. Theorem 4 follows.
APPENDIX B: Proof of Lemmas 1 and 2

Definitions:

Given an indexed variable \( x_{i,j,k} \) in \( R \), define its \textit{origin} to be \( j \); its \textit{stage} to be \( k \); its \textit{type} to be \( \alpha_j \); its \textit{location} (in \( R \)) to be 
\[
(j - 1) \cdot \ell + k \text{ if } \alpha_j = 1 \text{ and } (j - 1) \cdot \ell + (\ell - k + 1) \text{ if } \alpha_j = -1.
\]
(Note that the location of a variable is its distance from the r.h.s. end of \( R \).)

A reduction (cancellation) process on a word, \( w \), is a sequence of pairs of indexed variables such that:

1. The first pair corresponds to variables which are adjacent and can be cancelled in \( w \).
2. The \( j \)-th pair corresponds to variables which are adjacent, and can be cancelled in the word which results from \( w \) by applying the first \((j - 1)\) cancellations.
3. The word which results from \( w \) by applying all the cancellations (of the sequence) is \( \overline{w} \).

Let \( C \) be a reduction process on \( R \). (Note that \( L \) results from \( R \) by applying \( C \) to \( R \).) Given an indexed variable in \( R \), which is reduced in \( C \), define its \textit{mate} to be the variable with which it is cancelled (i.e. they constitute a pair in \( C \)).
Denotation:

Let \( v \) be an indexed variable, denote \( v \)'s stage [type] by \( s_v[t_v] \).
Let \( 1 \leq i, j \leq \ell \), denote \( \text{add}(i,j) \triangleq (i + j - 1) \mod \ell + 1 \), where \( n \mod m \) denotes the reduction of \( n \) modulo \( m \).

Sublemma:

Let \( u \) and \( v \) (where \( u \)'s location is smaller than \( v \)'s location) be two indexed variables which become adjacent during (the reduction process) \( C \). If \( t_u = t_v \) then \( s_v = \text{add}(s_u,t_u) \), otherwise \( s_v = s_u \).

Proof:

Let \( C' \) be a prefix of \( C \), such that:

1. \( u \) and \( v \) are adjacent in the word which results from \( R \) by applying the cancellations of \( C' \).
2. \( C' \) is of minimal length with respect to (1).

The sublemma can be easily proven by induction on the length of \( C' \):
First note that if the length of \( C' \) is zero, then \( u \) and \( v \) are adjacent in \( R \) and the statement of the sublemma holds. To prove the induction step assume that \( C' = C''(u',v') \), where \( u \) is adjacent to \( u' \), \( u' \) is adjacent to \( v' \) and \( v' \) is adjacent to \( v \), in the string which results from \( R \) by applying the cancellations of \( C'' \). Note that \( t_u' \neq t_v' \). By the induction hypothesis (when applied to \( u' \) and \( v' \)), \( s_v' = s_u' \). Consider the case in which \( t_u = t_u' \), \( t_v' \neq t_v \). By the induction hypothesis, \( s_u' = \text{add}(s_u',t_u) \) and \( s_v' = s_v' \). Thus, \( s_v = s_u' = \text{add}(s_u',t_u) \) and \( t_u = t_v \) follow. The other (three) cases are handled similarly.
Note that Lemma 1 follows directly from the sublemma. To prove Lemma 2, apply the sublemma to each pair of adjacent variables in \( \overline{R} \), yielding \( k_{q+1} = \text{add}(k_q, 1) \) for \( 1 \leq q < \ell \). Also note that \( x_{i_1,1} \) is not reduced in \( C \) (otherwise the minimality of \( R \) is contradicted, since \( x_{i_1,1} \) must be cancelled by some \( x^{-1}_{i_p,1} \), which implies that)

\[
\sigma(i_t, \ldots i_t, 1) \cdots \sigma(i_2, \ldots i_2, 1) \sigma(i_1, \ldots i_1, 1) = \\
\sigma(i_t, \ldots i_t, 1) \cdots \sigma(i_{p+2}, \ldots i_{p+2}, 1) \sigma(i_{p+1}, \ldots i_{p+1}, 1).
\]

Thus, \( j_1 = k_1 = 1 \) and Lemma 2 follows.
APPENDIX C: Proof of Lemmas 3 and 4

Proof of Lemma 3:

Let $T$ denote the number of possibilities of choosing an ordered set $R$ such that $R \subseteq \text{Sym}(D)$ and $|R| = r$; and $N$ denote the number of possibilities of choosing such a set so that $|RS| < r \cdot s$.

Denote the elements of $S$ by $\sigma_i$ and those of $R$ by $\rho_i$ (i.e. $S \supseteq \{\sigma_i\}_{i=1}^s$, $R \supseteq \{\rho_i\}_{i=1}^r$). Denote by $N_{i,q}^j$ the number of possibilities of choosing $R$ such that $\rho_p \sigma_i = \rho_q \sigma_j$. Obviously:

1. $N \leq \sum_{1 \leq p \neq q \leq r} \sum_{1 \leq i \neq j \leq s} N_{i,q}^j \rho_p$.

2. $N_{i,q}^j = \binom{m}{r-1}(r-1)!$ if $i \neq j$ and $p \neq q$.

Combining (1) and (2) and using $\binom{a}{b} < \frac{b}{a-b} \binom{a}{b}$ we get

3. $N < r^2 \cdot s^2 \cdot \frac{r}{m-r} \cdot \frac{m}{r} \cdot (r-1)!$.

Since $T = \binom{m}{r} \cdot r!$ we get

$$\frac{N}{T} < \frac{r^2 \cdot s^2}{m-r} = 2^{2\log_2 r \cdot s - \log_2 (m-r)} \to 0,$$ since $\log_2 r \cdot s \ll \log_2 m$.

Denotation:

Denote $I \triangleq (i_1, i_2, \ldots, i_k)$ and $J \triangleq (i_{k+1}, i_{k+2}, \ldots, i_{2k})$.

(By $I \neq J$ we mean that there exists a $1 \leq q \leq 2k$ such that $i_q \neq i_{k+q}$.)

Proof of Lemma 4:

(Denote by $V_n$ the set of binary vectors of length $n$.)

Denote by $T$ the number of possibilities of choosing an ordered collection of $k$ ordered sets of $2^k$ permutations (each) over $V_n$. Denote
by $N$ the number of possibilities of choosing such a collection in which there exist two permutations, $\sigma$ and $\sigma'$, realizable by the GCC defined by this collection, so that $\sigma(M_j) = \sigma'(M_j)$ for every $1 \leq j \leq t$. Denote by $N^J_I$ the number of possibilities of choosing such a collection, 

$$\left\{ \left( \{ \sigma_i \} \right)_{i=1}^{2k} : 1 \leq j \leq k \right\},$$

so that

$$\sigma(\ell) \cdots \sigma(2) \cdot \sigma(1)(M_j) = \sigma_i(\ell) \cdots \sigma_i(2) \cdot \sigma_i(1)(M_j), \text{ for } 1 \leq j \leq t,$$

if $I \neq J$; and $N^J_J = 0$ if $I = J$.

Obviously:

$$N \leq \sum_{i_1=1}^{2k} \cdots \sum_{i_{2k}=1}^{2k} \frac{(i_{\ell+1}, i_{\ell+2}, \ldots, i_{2k})}{(i_1, i_2, \ldots, i_{2k})}$$

$$N^J_I = \frac{\left(\frac{n!}{2^k}\right) \cdot \left(\frac{n!}{2^k}\right)^{2-1} \cdot (2^k-1)! \cdot (2^n-t)!}{2^{2k}}$$

Combining (1) and (2), we get

$$N < (2^k)^2 \cdot \left(\frac{n!}{2^k}\right)^{2-1} \cdot \frac{2^k}{2^k-2^k} \cdot (2^k-1)! \cdot (2^n-t)!$$

Since $T = \frac{(2n!) \cdot (2^k)!}{2^{2k}}$ we get

$$N < \frac{2^k 2^{k \ell}}{2^{2k} \cdot (2^n-t)!} \approx 2^k 2^{k \ell} \cdot \frac{(2^n-t)!}{2^n!} \approx 2^{2k \ell - nt} = 2^{-\delta} \rightarrow 0,$$

since $2^n \gg 2^k$, $2^n \gg t$ and $\delta \gg 1$. 

\[\square\]
REFERENCES


