The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs

Mihir Bellare[1] and Phillip Rogaway[2]

[1] Dept. of Computer Science & Engineering University of California at San Diego, 9500 Gilman Drive, La Jolla, California 92093 USA.
[2] Dept. of Computer Science, University of California, Davis, California 95616, USA

Abstract. We show that, in the ideal-cipher model, triple encryption (the cascade of three independently-keyed blockciphers) is more secure than single or double encryption, thereby resolving a long-standing open problem. Our result demonstrates that for DES parameters (56-bit keys and 64-bit plaintexts) an adversary's maximal advantage against triple encryption is small until it asks about 2^{78} queries. Our proof uses code-based game-playing in an integral way, and is facilitated by a framework for such proofs that we provide.

1 Introduction

TRIPLE ENCRYPTION. Given a blockcipher $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ with inverse D consider blockciphers $\mathsf{Cascade}_E^{\mathrm{eee}}(K_0K_1K_2,\, X) = E_{K_2}(E_{K_1}(E_{K_0}(X)))$ and $\mathsf{Cascade}_E^{\mathrm{ede}}(K_0K_1K_2,\, X) = E_{K_2}(D_{K_1}(E_{K_0}(X)))$. Our results are the same for both constructions. Following [14, 6, 9], we model E as a family of random permutations, one for each key, and we provide the adversary with oracle access to the blockcipher $E(\cdot,\cdot)$ and its inverse $E^{-1}(\cdot,\cdot)$. Given such oracles, the adversary is asked to distinguish between (a) $\mathsf{Cascade}_E^{\mathrm{eee}}(K_0K_1K_2,\,\cdot\,)$ and its inverse, for a random key $K_0K_1K_2$, and (b) a random permutation on n bits and its inverse. We show that the adversary's advantage in making this determination, $\mathbf{Adv}_{k,n}^{\mathrm{eee}}(q)$, remains small until it asks about $q = 2^{k+0.5\min\{k,n\}}$ queries (the actual expression is more complex). The bound we get is plotted as the rightmost curve of Fig. 1 for DES parameters $k = 56$ and $n = 64$. In this case an adversary must ask more than $2^{78.5}$ queries to get advantage 0.5. Also plotted are the security curves for single and double encryption, where the adversary must ask 2^{55} and $2^{55.5}$ queries to get advantage 0.5. For a blockcipher with $k = n = 64$, the adversary must ask more than 2^{89} queries to get advantage 0.5. As there are matching attacks and security bounds for single and double encryption [4, 1] our result proves that, in the ideal-cipher model, triple encryption is much more secure than single or double encryption.

As background for the above, note that the security of the cascade construction, where two or more independently keyed blockciphers are composed with one another, is a long-standing open problem [4, 12]. Even and Goldreich refer to it as a "critical question" in cryptography [5, p. 109]. They showed that the

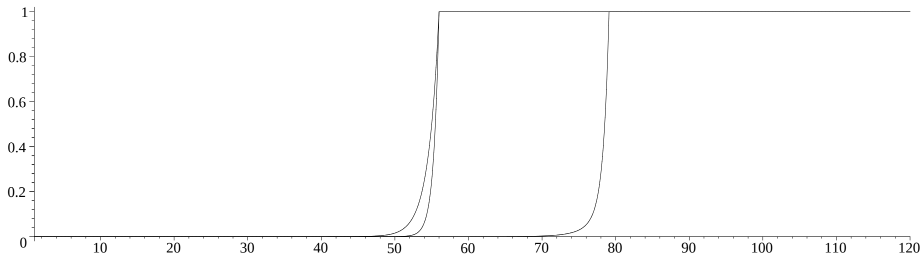

Fig. 1. Upper bound on adversarial advantage (proven security) verses $\log_2 q$ (where q=number of queries) for the cascade construction, assuming key length $k = 56$ and block length $n = 64$. Single encryption is the leftmost curve, double encryption is the middle curve [1], and triple encryption in the rightmost curve, as given by Theorem 1.

cascade of ciphers is at least as strong as the *weakest* cipher in the chain [5], while Maurer and Massey showed that, in a weaker attack model, it is at least as strong as the *first* cipher in the chain [11]. Aiello, Bellare, Di Crescenzo, and Venkatesan [1] prove the security of double-encryption (the two-stage cascade) in the same model as we use in this paper, showing that the maximal adversary advantage in q queries is $q^2/2^{2k}$. The meet-in-the-middle attack [4] implies that this is the best possible. So the adversary's advantage for double-encryption is only small until $q \approx 2^k$, just as for single encryption (although it grows as a slower rate). Thus triple encryption is the shortest potentially "good" cascade. And, indeed, triple DES is the cascade that is widely standardized and used [13].

The best published attack on three-key triple-encryption is due to Lucks [10]. He does not work out an explicit lower bound for $\mathbf{Adv}^{\mathrm{eee}}_{k,n}(q)$, but in the case of triple-DES, the adversary's advantage becomes large at around $q = 2^{90}$ queries (using enormous time and memory, too). We prove security to about 2^{78} queries, so there is no contradiction.

As for the cascade of $\ell \geq 4$ blockciphers, the maximal advantage in our attack model is no worse than it is for triple encryption, so our result proves that cascade "works" for all $\ell \geq 3$. It is open if security increases with increasing ℓ.

Game-playing framework. Our proof for triple-encryption uses game-playing in an integral way, first to recast the advantage we wish to bound via a simpler game, and later to analyze that game via others. Ultimately one is left with a game where conventional probabilistic reasoning can be applied.

What constitutes a game-playing proof is a matter of perspective. To some, a game-playing proof in cryptography is any proof where one conceptualizes the adversary's interaction with its environment as a kind of game, the proof proceeding by constructing a "chain" of such games. Viewed in this way, game-playing proofs have their origin in the earliest hybrid arguments, which began with Goldwasser and Micali [7] and Yao [16]. Bellare and Goldwasser [2] provide an early example of the use of a game-chain to prove security of a construction that uses multiple different cryptographic primitives.

In our treatment, games are code (ie, programs), not abstract environments; as we develop it, game-playing centers around making disciplined transformations to code. This approach begins with Kilian and Rogaway [9], and was used by Rogaway in many subsequent works. The framework of Section 2 develops this approach, and in particular our Fundamental Lemma (Lemma 1) is about the probability that an adversary can distinguish between games (programs) that differ in a certain syntactic way.

Shoup has independently and contemporaneously prepared a manuscript on game playing [15], advocating the use of game chains to make proofs more accessible. Shoup has often used game-playing over the years. His approach is not code-based. Shoup's [15, Lemma 1] functions like our Fundamental Lemma, but the former is cast in terms of conditional probabilities while the latter talks of programs that differ only after the setting of a flag *bad*.

Following the web distribution of this paper, Halevi argues for the creation of an automated tool to help write and verify game-based proofs [8]. We agree. The possibility for such tools has always been one of our motivations, and one of the reasons we focus on code-based games.

Our broader paper [3] contains further illustrations of game-based proofs (the PRP/PRF Switching Lemma, the CBC MAC, and OAEP) and a discussion of general techniques for code-based game-playing.

2 The Game-Playing Framework

Games are programs, written in pseudocode or in some formalized programming language. We describe some elements of the language we use. The semantics of a boolean variable, which we will also call a *flag*, is that once true it stays true. A *random-assignment* statement has the form $s \xleftarrow{\$} S$ where S is a finite set. This is the only source of randomness in programs. A game consists of an *initialization procedure* (Initialize), a *finalization procedure* (Finalize), and named *oracles* (each a procedure). The adversary, which we also regard as code, makes calls to the oracles, passing in values from some finite domain associated to each oracle. The initialization or finalization procedures may be absent, and often are, and there may be any number of oracles, including none. All variables in a game are global, and they are not visible to the adversary.

We can *run* a game G with an adversary A. To begin, variables are given initial values. Integer variables are initialized to 0; boolean variables are initialized to false; string variables are initialized to the empty string ε; set variables are initialized to the empty set $\emptyset$; and array variables hold the value $\mathsf{undefined}$ at every point. These conventions often enable omitting explicit initialization code. When used in a boolean expression, undefined values are regarded as false.

The Initialize procedure is the first to execute, possibly producing an output *inp*. This is provided as input to the Adversary procedure A, which now runs. The adversary code can make oracle queries via statements of the form $y \leftarrow P(\cdots)$ for any oracle P that has been defined in the game. The result is to assign to y the value returned by the procedure call. We assume that the game and adversary match syntactically, meaning that all the oracle calls made by the adversary are to oracles specified in the game, and with arguments that match in quantity and type. The semantics of a call is call-by-value; the only way for an oracle to return a value to the adversary is via a return statement. When adversary A halts, possibly with some *adversary output*, we call Finalize, providing it any such output. The Finalize procedure returns a string that we call the *game output*. If we omit specifying Initialize or Finalize, or their return-statements, it means that the procedure returns its input. The game output and adversary output are often the same, because Finalize (or a return-statement for it) is unspecified.

The adversary and game outputs can be regarded as random variables. We write $\Pr[A^G \Rightarrow 1]$ for the probability that the adversary output is 1 when we run game G with adversary A, and $\Pr[G^A \Rightarrow 1]$ for the probability that the game output is 1 when we run game G with adversary A.

ADVANTAGES. If G and H are games and A is an adversary, let $\mathbf{Adv}(A^G, A^H) = \Pr[A^G \Rightarrow 1] - \Pr[A^H \Rightarrow 1]$ and $\mathbf{Adv}(G^A, H^A) = \Pr[G^A \Rightarrow 1] - \Pr[H^A \Rightarrow 1]$. These represent the advantage of the adversary in distinguishing the games, the first measured via adversary output and the second via game output. We refer to the first as the *adversarial advantage* and the second as the *game advantage*. We say that G, H are *adversarially indistinguishable* if for any adversary A it is the case that $\mathbf{Adv}(A^G, A^H) = 0$, and *equivalent* if for any adversary A it is the case that $\mathbf{Adv}(G^A, H^A) = 0$. We will often use the fact that

$$\mathbf{Adv}(A^G, A^I) = \mathbf{Adv}(A^G, A^H) + \mathbf{Adv}(A^H, A^I) \tag{1}$$

$$\mathbf{Adv}(G^A, I^A) = \mathbf{Adv}(G^A, H^A) + \mathbf{Adv}(H^A, I^A) \tag{2}$$

for any games G, H, I and any adversary A. These follow simply from the fact that $(a-b)+(b-c) = a-c$. These will be referred to as the *triangle equalities*.

THE FUNDAMENTAL LEMMA. Let G and H be games and let *bad* be a flag that occurs in both of them. Then we say that G and H are *identical-until-bad* if their code is the same except that there might be places where G has a statement $\underline{bad \leftarrow \mathsf{true}, S}$ while game H has a corresponding statement $\underline{bad \leftarrow \mathsf{true}, T}$ for some T that may be different from S. (One could also say that G and H are are identical-until-*bad* if one has the statement $\underline{\text{if } bad \text{ then } S}$ where the other has the empty statement, for this can be rewritten in the form above.) The identical-until-*bad* predicate is an equivalence relation.

We write $\Pr[A^G \text{ sets } bad\,]$ or $\Pr[G^A \text{ sets } bad\,]$ to refer to the probability that the flag *bad* is true at the end of the execution of the adversary A with game G (that is, when Finalize terminates). The fundamental lemma says that the advantage that an adversary can obtain in distinguishing a pair of identical-until-*bad* games is at most the probability that its execution sets *bad* in one of them.

Lemma 1. [Fundamental lemma of game-playing] *Let G and H be identical-until-bad games and let A be an adversary. Then*

$$\mathbf{Adv}(A^G, A^H) \le \Pr[A^G \text{ sets } bad\,] \text{ and } \mathbf{Adv}(G^A, H^A) \le \Pr[G^A \text{ sets } bad\,]\,.$$

More generally, if G, H, and I are identical-until-bad games then

$$\left|\mathbf{Adv}(A^G, A^H)\right| \le \Pr[A^I \text{ sets } bad\,] \text{ and } \left|\mathbf{Adv}(G^A, H^A)\right| \le \Pr[I^A \text{ sets } bad\,]. \blacksquare$$

One of the most common manipulations of games along a game chain is to change what happens after *bad* gets set to true. Any modification following the setting of *bad* leaves unchanged the probability of setting *bad*.

Proposition 1. [After *bad* is set, nothing matters] *Let G and H be identical-until-bad, and A an adversary. Then* $\Pr[G^A \text{ sets } bad\,] = \Pr[H^A \text{ sets } bad\,]$. ∎

3 The Security of Three-Key Triple-Encryption

Definitions. Let $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher with key length k and block length n. For $K \in \{0,1\}^k$ and $X \in \{0,1\}^n$ let $E_K(X) = E(K,X)$. Let $E^{-1}\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be the blockcipher that is the inverse of E. We associate to E two blockciphers formed by composition; denoted $\mathsf{Cascade}_E^{\mathrm{eee}}, \mathsf{Cascade}_E^{\mathrm{ede}}\colon \{0,1\}^{3k} \times \{0,1\}^n \to \{0,1\}^n$, these were defined in Section 1. These blockciphers have key length $3k$ and block length n and are sometimes referred to as the *three-key* forms of triple encryption. We will call the two methods EEE and EDE, respectively. There is also a *two-key* variant of triple encryption, obtained by setting $K_0 = K_2$, but we do not investigate it since the method admits comparatively efficient attacks [12].

We will be working in the ideal-blockcipher model, as in works like [6, 9, 1]. Let $\mathrm{Bloc}(k,n)$ be the set of all blockciphers $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$. Thus $E \xleftarrow{\$} \mathrm{Bloc}(k,n)$ means that $E_K\colon \{0,1\}^n \to \{0,1\}^n$ is a random permutation on n-bit strings for each $K \in \{0,1\}^k$. We consider an adversary A that can make four types of oracle queries: $\boldsymbol{T}(X)$, $\boldsymbol{T^{-1}}(Y)$, $\boldsymbol{E}(K,X)$, and $\boldsymbol{E^{-1}}(K,Y)$, where $X, Y \in \{0,1\}^n$ and $K \in \{0,1\}^k$. (As for our syntax, $\boldsymbol{T}$, $\boldsymbol{T^{-1}}$, $\boldsymbol{E}$, $\boldsymbol{E^{-1}}$ are formal symbols, not specific functions.) The *advantage of A against EEE* and the *maximal advantage against EEE obtainable using q queries* are defined as

$$\mathbf{Adv}_{k,n}^{\mathrm{eee}}(A) \;=\; \mathbf{Adv}(A^{C_0}, A^{R_0}) \quad \text{and} \quad \mathbf{Adv}_{k,n}^{\mathrm{eee}}(q) \;=\; \max_{A}\left\{\, \mathbf{Adv}_{k,n}^{\mathrm{eee}}(A) \,\right\}$$

where the games C_0, R_0 are shown in Fig. 2 and the maximum is over all adversaries A that make at most q oracle queries (that is, a total of q across all four oracles). The advantage of A measures its ability to tell whether $\boldsymbol{T}(\cdot)$ is a random permutation or is $\mathsf{Cascade}_E^{\mathrm{eee}}(K_0K_1K_2,\,\cdot\,)$ for $K_0K_1K_2$ chosen independently at random from $\{0,1\}^{3k}$ and where $\boldsymbol{E}$ realizes a random blockcipher and $\boldsymbol{T^{-1}}, \boldsymbol{E^{-1}}$ realize inverses of $\boldsymbol{T}, \boldsymbol{E}$, respectively.

Define the *query threshold* $\mathbf{QTh}_{1/2}^{\mathrm{eee}}(k,n)$ as the largest integer q for which $\mathbf{Adv}_{k,n}^{\mathrm{eee}}(q) \le 1/2$. We will regard EEE as being secure up to $\mathbf{QTh}_{1/2}^{\mathrm{eee}}(k,n)$ queries. Let $\mathbf{Adv}_{k,n}^{\mathrm{ede}}(A), \mathbf{Adv}_{k,n}^{\mathrm{ede}}(q)$, and $\mathbf{QTh}_{1/2}^{\mathrm{ede}}(k,n)$ be defined in the analogous way.

Results. We are now ready to state our result about the security of triple encryption.

Theorem 1. [Security of triple-encryption] *Let* $k, n \geq 2$. *Let* $\alpha = \max(2e2^{k-n}, 2n+k)$. *Then*

$$\mathbf{Adv}^{\mathrm{eee}}_{k,n}(q) \;\leq\; 4\alpha\,\frac{q^2}{2^{3k}} + 10.7\left(\frac{q}{2^{k+n/2}}\right)^{2/3} + \;\frac{12}{2^k}\,. \quad \blacksquare \qquad (3)$$

We display the result graphically in Fig. 1 for DES parameters $k = 56$ and $n = 64$. Our bound implies that $\mathbf{QTh}^{\mathrm{eee}}_{1/2}(k,n)$ is, very roughly, about $2^{k+\min(k,n)/2}$, meaning that EEE is secure up to this many queries.

For EDE the result is the same, meaning that $\mathbf{Adv}^{\mathrm{ede}}_{k,n}(q)$ is also bounded by the quantity on the right-hand-side of (3). This can be shown by mostly-notational modifications to the proof of Theorem 1.

4 Proof of Theorem 1

Overview. The first step in our proof reduces the problem of bounding the advantage of an adversary A against EEE to bounding certain quantities that relate to a different, *simplified* adversary B. By a simplified adversary we mean one that makes no $\boldsymbol{T}(\cdot)$, $\boldsymbol{T}^{-1}(\cdot)$ queries, meaning it only has oracles $\boldsymbol{E}(\cdot,\cdot)$ and $\boldsymbol{E}^{-1}(\cdot,\cdot)$. We will consider two games, both involving random, distinct keys K_0, K_1, K_2. In one game (R_3) E_{K_2} is random, while in the other (D_S), it is correlated to E_{K_0}, E_{K_1}. The quantities we will need to bound are the ability of our simplified adversary to either distinguish these games without extending a 2-chain, or to extend a 2-chain in one of the games, where what it means to extend a 2-chain is explained below. We will be able to provide these two bounds via two lemmas. The first considers a simplified game in which an adversary has only three permutation oracles, either all random or one correlated to the rest, and has to distinguish them without extending a 2-chain. The second bounds the probability that the adversary can extend a 2-chain in R_3.

Conventions. We begin with some conventions. An adversary A against EEE can make oracle queries $\boldsymbol{T}(X)$, $\boldsymbol{T}^{-1}(Y)$, $\boldsymbol{E}(K,X)$, or $\boldsymbol{E}^{-1}(K,Y)$ for any $X, Y \in \{0,1\}^n$ and $K \in \{0,1\}^k$. We will assume that any adversary against EEE is deterministic and never makes a *redundant* query. A query is redundant if it has been made before; a query $\boldsymbol{T}^{-1}(Y)$ is redundant if A has previously received Y in answer to a query $\boldsymbol{T}(X)$; a query $\boldsymbol{T}(X)$ is redundant if A has previously received X in answer to a query $\boldsymbol{T}^{-1}(Y)$; a query $\boldsymbol{E}^{-1}(K,Y)$ is redundant if A has previously received Y in answer to a query $\boldsymbol{E}(K,X)$; a query $\boldsymbol{E}(K,X)$ is redundant if A has previously received X in answer to a query $\boldsymbol{E}^{-1}(K,Y)$. Assuming A to be deterministic and not to ask redundant queries is without loss of generality in the sense that for any A that asks q queries there is an A' asking at most q queries that satisfies these assumptions and achieves the same advantage as A. Our general conventions about games imply that A never asks a query with arguments outside of the intended domain, meaning $\{0,1\}^k$ for keys and $\{0,1\}^n$ for messages.

procedure Initialize
$K_0, K_1, K_2 \xleftarrow{\$} \{0,1\}^k$, $E \xleftarrow{\$} \text{Bloc}(k,n)$, $T \xleftarrow{\$} \text{Perm}(n)$, $\boxed{T \leftarrow E_{K_2} \circ E_{K_1} \circ E_{K_0}}$

procedure $\boldsymbol{T}(P)$
return $T[P]$

procedure $\boldsymbol{T^{-1}}(S)$
return $T^{-1}[S]$

procedure $\boldsymbol{E}(K, X)$
return $E_K[X]$

procedure $\boldsymbol{E^{-1}}(K, Y)$
return $E_K^{-1}[Y]$

Game R_0 $\boxed{\text{Game } C_0}$

---

procedure Initialize
$(K_0, K_1, K_2) \xleftarrow{\$} \text{Distinct}_3^k$, $E \xleftarrow{\$} \text{Bloc}(k,n)$, $T \xleftarrow{\$} \text{Perm}(n)$, $\boxed{E_{K_2} \leftarrow T \circ E_{K_0}^{-1} \circ E_{K_1}^{-1}}$

procedure $\boldsymbol{T}(P)$
return $T[P]$

procedure $\boldsymbol{T^{-1}}(S)$
return $T^{-1}[S]$

procedure $\boldsymbol{E}(K, X)$
return $E_K[X]$

procedure $\boldsymbol{E^{-1}}(K, Y)$
return $E_K^{-1}[Y]$

Game R_1 $\boxed{\text{Game } C_1}$

---

procedure Initialize
$(K_0, K_1, K_2) \xleftarrow{\$} \text{Distinct}_3^k$, $E \xleftarrow{\$} \text{Bloc}(k,n)$, $\boxed{E_{K_2} \leftarrow T \circ E_{K_0}^{-1} \circ E_{K_1}^{-1}}$

procedure $\boldsymbol{E}(K, X)$
return $E_K[X]$

procedure $\boldsymbol{E^{-1}}(K, Y)$
return $E_K^{-1}[Y]$

Game R_2 $\boxed{\text{Game } C_T}$

---

procedure Initialize
$(K_0, K_1, K_2) \xleftarrow{\$} \text{Distinct}_3^k$, $E \xleftarrow{\$} \text{Bloc}(k,n)$, $\boxed{E_{K_2} \leftarrow S \circ E_{K_0}^{-1} \circ E_{K_1}^{-1}}$

Game R_3 $\boxed{\text{Game } D_S}$

procedure $\boldsymbol{E}(K, X)$
if $\exists\, i \in \{0,1,2\}$ such that $K = K_i$ then
  $Q \leftarrow E_{K_{i+2}}^{-1}[X]$, $P \leftarrow E_{K_{i+1}}^{-1}[Q]$
  if $P \xrightarrow{i+1} Q \xrightarrow{i+2} X$ then *x2ch* $\leftarrow$ `true`
  Add arc $X \xrightarrow{i} E_K[X]$
return $E_K[X]$

procedure $\boldsymbol{E^{-1}}(K, Y)$
if $\exists\, i \in \{0,1,2\}$ such that $K = K_i$ then
  $Q \leftarrow E_{K_{i+1}}[Y]$, $R \leftarrow E_{K_{i+2}}[Q]$
  if $Y \xrightarrow{i+1} Q \xrightarrow{i+2} R$ then *x2ch* $\leftarrow$ `true`
  Add arc $E_K^{-1}[Y] \xrightarrow{i} Y$
return $E_K^{-1}[Y]$

procedure Finalize(out)
if *x2ch* then return 1 else return out

Fig. 2. The C_X or D_X games include the boxed statements while the R_i games do not.

4.1 Reduction to simplified adversary

Consider the games in Fig. 2. The R-games (where R stands for random) omit the boxed assignment statements while the C-games and D-game include them. Distinct_3^k denotes the set of all triples $(K_0, K_1, K_2) \in (\{0,1\}^k)^3$ such that $K_0 \neq K_1$ and $K_1 \neq K_2$ and $K_0 \neq K_2$. Games R_0, R_1, C_0, C_1 will be run with an adversary against EEE. The rest of the games will be run with a simplified adversary. Game C_T is parameterized by a permutation $T \in \mathrm{Perm}(n)$, meaning we are effectively defining one such game for every T, and similarly D_S is parameterized by a permutation $S \in \mathrm{Perm}(n)$. Game D_S grows an (initially without edges) edge-labeled directed graph with vertex set $\{0,1\}^n$. An arc $X \xrightarrow{i} Y$ is created when a query $\mathbf{E}_{K_i}(X)$ returns the value Y or a query $E^{-1}_{K_i}(Y)$ returns the value X. The boolean flag *x2ch* is set if the adversary extends a 2-chain, meaning that a path $P \xrightarrow{i+1} Q \xrightarrow{i+2} R$ exists in the graph and the adversary asks either $E_{K_i}(R)$ or $E^{-1}_{K_i}(P)$, where the indicated addition is modulo 3. Note that D_S has an explicit Finalize procedure, indicating we will be interested in the game output rather than the adversary output.

Lemma 2. *Let A be an adversary that makes at most q queries. Then there is a permutation $S \in \mathrm{Perm}(n)$ and a simplified adversary B making at most q queries such that $\mathbf{Adv}^{\mathrm{eee}}_{k,n}(A)$ is at most*

$$\mathbf{Adv}(D_S^B, R_3^B) + \Pr\left[\, D_S^B \text{ sets x2ch} \,\right] + \Pr\left[\, R_3^B \text{ sets x2ch} \,\right] + \frac{6}{2^k} \,. \qquad \blacksquare$$

Proof (Lemma 2). Game C_0 defines T as $E_2 \circ E_1 \circ E_0$ for random E_0, E_1, E_2, while game C_1 defines E_2 as $T \circ E^{-1}_{K_0} \circ E^{-1}_{K_1}$ for random T, E_{K_0}, E_{K_1}. However, these processes are identical. With this factored out, the difference between C_1 and C_0 is that the former draws the keys K_0, K_1, K_2 from $\mathrm{Distinct}^k_3$ while the latter draws them from $(\{0,1\}^k)^3$. Games R_1 and R_0 differ in only the latter way. So using (1) we have

$$\mathbf{Adv}^{\mathrm{eee}}_{k,n}(A) = \mathbf{Adv}(A^{C_0}, A^{R_0}) \le \mathbf{Adv}(A^{C_1}, A^{R_1}) + \frac{6}{2^k} \,.$$

Game C_T is parameterized by a permutation $T \in \mathrm{Perm}(n)$. For any such T we consider an adversary A_T that has T hardwired in its code and is simplified, meaning can make queries $\boldsymbol{E}(K,X)$ and $\boldsymbol{E^{-1}}(K,Y)$ only. This adversary runs A, answering the latter's $\boldsymbol{E}(K,X)$ and $\boldsymbol{E^{-1}}(K,Y)$ queries via its own oracles, and answering $\boldsymbol{T}(X)$ and $\boldsymbol{T^{-1}}(Y)$ queries using T. Note that A_T makes at most q oracle queries. Choose $S \in \mathrm{Perm}(n)$ such that $\mathbf{Adv}(A_S^{C_S}, A_S^{R_2})$ is the maximum over all $T \in \mathrm{Perm}(n)$ of $\mathbf{Adv}(A_T^{C_T}, A_T^{R_2})$ and let $B = A_S$. We now have $\mathbf{Adv}(A^{C_1}, A^{R_1}) \le \mathbf{Adv}(B^{C_S}, B^{R_2})$. Now by (2) we have

$$\mathbf{Adv}(B^{C_S}, B^{R_2}) \le \mathbf{Adv}(C_S^B, D_S^B) + \mathbf{Adv}(D_S^B, R_3^B) + \mathbf{Adv}(R_3^B, R_2^B) \,.$$

Game C_S (resp. game R_2) can be easily transformed into an equivalent game such that this game and game D_S (resp. R_3) are identical-until-*x2ch*, so by the Fundamental Lemma we have $\mathbf{Adv}(C_S^B, D_S^B) \le \Pr[D_S^B \text{ sets } x2ch]$ and $\mathbf{Adv}(R_3^B, R_2^B) \le \Pr[R_3^B \text{ sets } x2ch]$. Putting all this together completes the lemma's proof. $\blacksquare$

Letting $p = \Pr\left[\, R_3^B \text{ sets } x2ch \,\right]$, we now need to bound

$$\mathbf{Adv}(D_S^B, R_3^B) + (\Pr\left[\, D_S^B \text{ sets } x2ch \,\right] - p) + 2p \,. \tag{4}$$

We will be able to bound the first two terms by bounding the advantages of a pair B_1, B_2 of adversaries, related to B, in distinguishing between a pair of games that involve only three permutation oracles, the first two random, and the third either random or correlated to the first two. We will bound p separately via a combinatorial argument. We now state the lemmas we need, conclude the proof of Theorem 1 using them in Section 4.4, and then return to provide the proofs of the two lemmas.

4.2 Pseudorandomness of three correlated permutations

We posit a new problem. Consider games G and H defined in Fig. 3. Game G grows an edge-labeled graph, which we shall describe shortly. An adversary may make queries $\boldsymbol{\Pi}(i, X)$ or $\boldsymbol{\Pi}^{-1}(i, Y)$ where $i \in \{0, 1, 2\}$ and $X, Y \in \{0, 1\}^n$. The oracles realize three permutations and their inverses, the function realized by $\boldsymbol{\Pi}^{-1}(i, \cdot)$ being the inverse of that realized by $\boldsymbol{\Pi}(i, \cdot)$. In both games permutations π_0, π_1 underlying $\boldsymbol{\Pi}(0, \cdot)$ and $\boldsymbol{\Pi}(1, \cdot)$ are uniform and independent. In game G the permutation π_2 underlying $\boldsymbol{\Pi}(2, \cdot)$ is also uniform and independent of π_0 and π_1, but in game H it is equal to $\pi_1^{-1} \circ \pi_0^{-1}$.

Notice that it is easy for an adversary to distinguish between games G and H by making queries that form a "chain" of length three: for any $P \in \{0, 1\}^n$, let the adversary ask and be given $Q \leftarrow \pi_0(P)$, then $R \leftarrow \pi_1(Q)$, then $P' \leftarrow \pi_2(R)$, and then have the adversary output 1 if $P = P'$ (a "triangle" has been found) or 0 if $P \neq P'$ (the "three-chain" is not in fact a triangle). What we will establish is that, apart from such behavior—extending a known "2-chain"—the adversary is not able to gain much advantage. To capture this, as the adversary A makes its queries and gets replies, the games grow an (initially without edges) edge-labeled directed graph $\mathcal{G}$ with vertex set. An arc $X \xrightarrow{i} Y$ is created when a query $\boldsymbol{\Pi}(i, X)$ returns the value Y or a query $\boldsymbol{\Pi}^{-1}(i, Y)$ returns the value X. The boolean flag *x2ch* is set in the games if the adversary extends a 2-chain, meaning that a path $P \xrightarrow{i+1} Q \xrightarrow{i+2} R$ exists in the graph and the adversary asks either $\boldsymbol{\Pi}(i, R)$ or $\boldsymbol{\Pi}^{-1}(i, P)$, where the indicated addition is modulo 3. We will be interested in the game outputs rather than the adversary outputs. Again using a game-based proof, we prove the following in Section 4.5:

Lemma 3. *If* $\Pr\left[B^G \text{ makes} \geq h \text{ oracle queries}\right] \leq \delta$ *then* $\mathbf{Adv}(H^B, G^B) \leq 2.5\, h^2/2^n + \delta$. ∎

We remark that the lemma makes no (explicit) assumption about the probability that B^H makes h or more oracle queries.

4.3 The improbability of forming a 3-chain

Consider an adversary B that can make $\boldsymbol{E}(K, X)$ or $\boldsymbol{E}^{-1}(K, Y)$ queries. Game L of Fig. 3 implements the oracles as a random blockcipher and its inverse, respectively, but samples these lazily, defining points as they are needed. Write

procedure Initialize	Game G
$\pi_0, \pi_1, \pi_2 \xleftarrow{\$} \mathrm{Perm}(n)$, $\boxed{\pi_2 \leftarrow \pi_1^{-1} \circ \pi_0^{-1}}$	$\boxed{\text{Game } H}$
procedure $\boldsymbol{\Pi}(i, X)$	**procedure** $\boldsymbol{\Pi^{-1}}(i, Y)$
if $\exists\, P \xrightarrow{i+1} Q \xrightarrow{i+2} X \in \mathcal{G}$ then $x2ch \leftarrow \mathsf{true}$	if $\exists\, Y \xrightarrow{i+1} Q \xrightarrow{i+2} R \in \mathcal{G}$ then $x2ch \leftarrow \mathsf{true}$
add $X \xrightarrow{i} \pi_i[X]$ to $\mathcal{G}$	add $\pi_i^{-1}[Y] \xrightarrow{i} Y$ to $\mathcal{G}$
return $\pi_i[X]$	return $\pi_i^{-1}[Y]$
procedure Finalize(out)	
if $x2ch$ then return 1 else return out	

Fig. 3. Game H includes the boxed statement, game G does not.

procedure $\boldsymbol{E}(K, X)$	**procedure** $\boldsymbol{E^{-1}}(K, Y)$	Game L
return $E_K[X] \xleftarrow{\$} \overline{\mathrm{image}}(E_K)$	$E_K^{-1}[Y] \xleftarrow{\$} \overline{\mathrm{domain}}(E_K)$	
procedure Finalize		
$K_0, K_1, K_2 \xleftarrow{\$} \{0,1\}^k$		
if $(\exists P)$ $[E_{K_2}[E_{K_1}[E_{K_0}[P]]]]$ then $bad \leftarrow \mathsf{true}$		

Fig. 4. Game L captures improbability of making three chains.

$X \xrightarrow{K} Y$ to mean that that B has made query $\boldsymbol{E}(K, X)$ and obtained Y as a result, or made query $\boldsymbol{E^{-1}}(K, Y)$ and obtained X as a result, for $K \in \{0,1\}^k$ and $X, Y \in \{0,1\}^n$. The Finalize procedure picks keys K_0, K_1, K_2 at random, and sets bad if the adversary's queries have formed a three chain, meaning that there exist points $P, Q, R, S \in \{0,1\}^n$ such that $P \xrightarrow{K_0} Q \xrightarrow{K_1} R \xrightarrow{K_2} S$: the conditional which is the last line of Finalize means that there is a P for which $E_{K_0}[P]$ is defined and $E_{K_1}[E_{K_0}[P]]$ is defined and $E_{K_2}[E_{K_1}[E_{K_0}[P]]]$ is defined. Our next lemma bounds the probability of this happening. The proof is in Section 4.6.

Lemma 4. *Let $k, n \geq 1$. Let B be an adversary that asks at most q queries. Let $\alpha = \max(2e\, 2^{k-n}, 2n + k)$. Then $\Pr[B^L \text{ sets } bad\,] < 2\alpha\, q^2/2^{3k}$.* ∎

4.4 Putting together the pieces to conclude Theorem 1

Let B be a simplified adversary and $S \in \mathrm{Perm}(n)$ a permutation. We associate to B, S a pair of adversaries $B_{S,1}$ and $B_{S,2}$ that make $\boldsymbol{\Pi}(i, X)$ or $\boldsymbol{\Pi^{-1}}(i, Y)$ queries, where $i \in \{0, 1, 2\}$ and $X, Y \in \{0,1\}^n$, as follows. For $b \in \{1, 2\}$, adversary $B_{S,b}$ picks (K_0, K_1, K_2) at random from $\mathrm{Distinct}_3^k$ and picks E at random from $\mathrm{Bloc}(k, n)$. It then runs B, replying to its oracle queries as follows. If B makes a query $\boldsymbol{E}(K, X)$, adversary $B_{S,b}$ returns $E_K(X)$ if $K \notin \{K_0, K_1, K_2\}$; returns $\boldsymbol{\Pi}(i, X)$ if $K = K_i$ for $i \in \{0, 1\}$; and returns $S \circ \boldsymbol{\Pi}(2, X)$ if $K = K_2$. Similarly, if B makes a query $\boldsymbol{E^{-1}}(K, Y)$, adversary $B_{S,b}$ returns $E_K^{-1}(Y)$ if $K \notin \{K_0, K_1, K_2\}$; returns $\boldsymbol{\Pi^{-1}}(i, Y)$ if $K = K_i$ for $i \in \{0, 1\}$; and returns $\boldsymbol{\Pi^{-1}}(2, Y) \circ S^{-1}$ if $K = K_2$. Adversaries $B_{S,1}, B_{S,2}$ differ only in their output, the first always returning 0 and the second returning the output out of B.

Lemma 5. *Let B be a simplified adversary that makes at most q oracle queries, and let $S \in \mathrm{Perm}(n)$. Let $B_{S,1}, B_{S,2}$ be defined as above. Let $K = 2^k$. Then for $b \in \{1, 2\}$ and any $c > 0$, $\Pr[B^G_{S,b}$ makes $\geq 3cq/K$ oracle queries$\,] \leq 1/c$.* ▮

Proof (Lemma 5). The oracles B sees when it is run by $B_{S,b}$ are exactly a random block cipher and its inverse. (A random permutation composed with a fixed one is still random so the composition by S does not change anything.) Now let X be the random variable that is the number of queries by B that involve keys K_0, K_1, or K_2 in the experiment where we first run B with oracles E, E^{-1} for $E \stackrel{\$}{\leftarrow} \mathrm{Bloc}(k, n)$ and then pick $(K_0, K_1, K_2) \stackrel{\$}{\leftarrow} \mathrm{Distinct}^k_3$. Then the probability that $B^G_{S,b}$ makes $\geq 3cq/K$ oracle queries is exactly the probability that $X \geq 3cq/K$. Now assume wlog that B always makes exactly q distinct oracle queries rather than at most q. Then

$$\begin{aligned}
\mathbf{E}[X] &= q \cdot \left[1 - \left(1 - \frac{1}{K}\right)\left(1 - \frac{1}{K-1}\right)\left(1 - \frac{1}{K-2}\right)\right] \\
&= q \cdot \left[1 - \frac{K-1}{K}\frac{K-2}{K-1}\frac{K-3}{K-2}\right] \;=\; q \cdot \left[1 - \frac{K-3}{K}\right] \;=\; \frac{3q}{K}\,.
\end{aligned}$$

We can conclude via Markov's inequality. ▮

Proof (Theorem 1). Let A be an adversary against EEE that makes at most q oracle queries. Let B be the simplified adversary, and S the permutation, given by Lemma 2, and let $p = \Pr\left[\, R^B_3 \text{ sets } \mathit{x2ch} \,\right]$. Let $B_{S,1}, B_{S,2}$ be the adversaries associated to B as described above. Note that

$$\begin{array}{rcrl}
\Pr[D^B_S \text{ sets } \mathit{x2ch}]{=}\Pr[H^{B_{S,1}} \Rightarrow 1] & \text{and} & \Pr[R^B_3 \text{ sets } \mathit{x2ch}]{=}\Pr[G^{B_{S,1}} \Rightarrow 1] & \\
\Pr[D^B_S \Rightarrow 1]{=}\Pr[H^{B_{S,2}} \Rightarrow 1] & \text{and} & \Pr[R^B_3 \Rightarrow 1]{=}\Pr[G^{B_{S,2}} \Rightarrow 1]\,. &
\end{array} \tag{5}$$

Combining (4) and (5) we have:

$$\mathbf{Adv}^{\mathrm{eee}}_{k,n}(A) \leq 2p + \mathbf{Adv}(H^{B_{S,1}}, G^{B_{S,1}}) + \mathbf{Adv}(H^{B_{S,2}}, G^{B_{S,2}}) + \frac{6}{2^k}\,. \tag{6}$$

Let $\alpha = \max(2e2^{k-n}, 2n + k)$ and let c be any positive real number. Since the probability that R^B_3 extends a 2-chain is at most the probability that L^B forms a 3-chain we have $p \leq 3 \cdot 2^{-k} + \Pr[B^L \text{ sets } \mathit{bad}\,]$. (The extra term is because L picks the keys K_0, K_1, K_2 independently at random while R_3 picks them from $\mathrm{Distinct}^k_3$.) Applying Lemma 4 we get $p \leq 3{\cdot}2^{-k}{+}2\alpha\, q^2{\cdot}2^{-3k}$. Applying Lemma 3 in conjunction with Lemma 5 we have

$$\mathbf{Adv}(H^{B_{S,b}}, G^{B_{S,b}}) \leq \frac{2.5}{2^n}\left(\frac{3cq}{2^k}\right)^2 + \frac{1}{c}$$

for both $b = 1$ and $b = 2$. Putting everything together we have

$$\mathbf{Adv}^{\mathrm{eee}}_{k,n}(A) \leq 2\left(\frac{3}{2^k} + 2\alpha\,\frac{q^2}{2^{3k}}\right) + \frac{5}{2^n}\left(\frac{3cq}{2^k}\right)^2 + \frac{2}{c} + \frac{6}{2^k}\,.$$

Now, since the above is true for any $c > 0$, we pick a particular one that minimizes the function $f(c) = 45\, c^2 q^2\, 2^{-n-2k} + 2c^{-1}$. The derivative is $f'(c) = 90\, cq^2\, 2^{-n-2k} - 2c^{-2}$, and the only real root of the equation $f'(c) = 0$ is $c = (2^{n+2k}/45q^2)^{1/3}$, for which we have $f(c) = 3(45q^2/2^{n+2k})^{1/3}$. Plugging this into the above yields (3) and concludes the proof of Theorem 1. ∎

4.5 Proof of Lemma 3

We prove Lemma 3 as a corollary of:

Lemma 6. *If A asks at most q queries then* $\left|\mathbf{Adv}(G^A, H^A)\right| \le 2.5\, q^2/2^n$. ∎

Proof (Lemma 3). We construct an adversary A that has the same oracles as B. Adversary A runs B, answering B's oracle queries via its own oracles. It also keeps track of the number of oracle queries that B makes. If this number hits h, it stops and outputs 1; else it outputs whatever B outputs. Then we note that $\Pr[H^B \Rightarrow 1] \le \Pr[H^A \Rightarrow 1]$ and $\Pr[G^A \Rightarrow 1] \le \Pr[G^B \Rightarrow 1] + \delta$. Thus we have

$$\begin{aligned}\mathbf{Adv}(H^B, G^B) &= \Pr[H^B \Rightarrow 1] - \Pr[G^B \Rightarrow 1] \\ &\le \Pr[H^A \Rightarrow 1] - \big(\Pr[G^A \Rightarrow 1] - \delta\big) \;\; = \;\; \mathbf{Adv}(H^A, G^A) + \delta\,.\end{aligned}$$

As A makes $\le h$ queries, conclude by applying Lemma 6 to A with $q = h$. ∎

Proof (Lemma 6). We assume that the adversary A never repeats a query, never asks a query $\boldsymbol{\Pi}^{-1}(i, Y)$ having asked some $\boldsymbol{\Pi}(i, X)$ that returned Y, and never asks a query $\boldsymbol{\Pi}(i, X)$ having asked some $\boldsymbol{\Pi}^{-1}(i, Y)$ that returned X. Call an adversary *valid* if it never extends a two-chain.

We begin by noting that to bound A's advantage in distinguishing games G and H we may assume that A is valid. Why? Because for any adversary A^* making at most q_0 queries there exists a valid A that makes at most q_0 queries and the advantage of A is at least that of A^*. Adversary A runs A^*, answering A^*'s oracle queries via its own oracles, but at any point that A^* would extend a two chain, adversary A simply halts and outputs 1. So now assuming A's validity, our task is to show that $|\mathbf{Adv}(A^{G_1}, A^{H_1})| \le 2.5\, q^2/2^n$ where the games G_1, H_1 are shown in Fig. 5. We show that games G_1 and H_1 are close by showing that both are close to game G_3 (defined in the same figure). First, we claim that $\left|\mathbf{Adv}(A^{G_1}, A^{G_3})\right| \le 0.5\, q^2/N$ where, here and in the rest of this proof, $N = 2^n$. Rewrite game G_1 to game $G_{1.5}$ (not shown) by lazily growing π_0, π_1, π_2, setting the flag *bad* whenever there is a collision; that is, game $G_{1.5}$ is identical to game G_2 except, after setting *bad* at line 211, set $Y \xleftarrow{\$} \overline{\mathrm{image}}(\pi_i)$, and after setting *bad* at line 221, set $X \xleftarrow{\$} \overline{\mathrm{domain}}(\pi_i)$. Then modify game $G_{1.5}$ to *not* re-sample after setting *bad*, obtaining game G_2. Now $\left|\mathbf{Adv}(A^{G_1}, A^{G_3})\right| = \left|\mathbf{Adv}(A^{G_{1.5}}, A^{G_3})\right| = \left|\mathbf{Adv}(A^{G_{1.5}}, A^{G_2})\right| \le \Pr[A^{G_2} \text{ sets } bad\,]$. Then note that on the i^{th} query the probability that *bad* will be set in game G_2 is at most $(i-1)/N$ since the size of $\mathrm{domain}(\pi_j)$ and $\mathrm{image}(\pi_j)$ will be at most $i - 1$ for each $j \in \{0, 1, 2\}$. So

procedure Initialize Game G_1

$\pi_0, \pi_1, \pi_2 \xleftarrow{\$} \mathrm{Perm}(n)$, $\boxed{\pi_2 \leftarrow \pi_1^{-1} \circ \pi_0^{-1}}$ $\boxed{\text{Game } H_1}$

procedure $\boldsymbol{\Pi}(i, X)$

return $\pi_i[X]$

procedure $\boldsymbol{\Pi^{-1}}(i, Y)$

return $\pi_i^{-1}[Y]$

---

procedure $\boldsymbol{\Pi}(i, X)$

$Y \xleftarrow{\$} \{0,1\}^n$

if $Y \in \mathrm{image}(\pi_i)$ then $bad \leftarrow \mathsf{true}$

$\pi[X] \leftarrow Y$

return Y

procedure $\boldsymbol{\Pi^{-1}}(i, Y)$ Game G_2

$X \xleftarrow{\$} \{0,1\}^n$

if $X \in \mathrm{domain}(\pi_i)$ then $bad \leftarrow \mathsf{true}$

$\pi[X] \leftarrow Y$

return X

---

procedure $\boldsymbol{\Pi}(i, X)$

return $Y \xleftarrow{\$} \{0,1\}^n$

procedure $\boldsymbol{\Pi^{-1}}(i, Y)$ Game G_3

return $X \xleftarrow{\$} \{0,1\}^n$

---

procedure $\boldsymbol{\Pi}(i, X)$ Game G_4

if $\exists\, (i, X, Y) \in \mathcal{C}$ then return Y

$X_i \leftarrow X$

$X_{i+1} \xleftarrow{\$} \{0,1\}^n$, if $X_{i+1} \in S_{i+1}$ then $bad \leftarrow \mathsf{true}$, $X_{i+1} \xleftarrow{\$} \{0,1\}^n \setminus S_{i+1}$

$X_{i+2} \xleftarrow{\$} \{0,1\}^n$, if $X_{i+2} \in S_{i+2}$ then $bad \leftarrow \mathsf{true}$, $X_{i+2} \xleftarrow{\$} \{0,1\}^n \setminus S_{i+2}$

$S_i \leftarrow S_i \cup \{X_i\}$, $S_{i+1} \leftarrow S_{i+1} \cup \{X_{i+1}\}$, $S_{i+2} \leftarrow S_{i+2} \cup \{X_{i+2}\}$

$\mathcal{C} \leftarrow \mathcal{C} \cup \{(i, X_i, X_{i+1}),\ (i+1, X_{i+1}, X_{i+2}),\ (i+2, X_{i+2}, X_i)\}$

return X_{i+1}

procedure $\boldsymbol{\Pi^{-1}}(i, Y)$

if $\exists\, (i, X, Y) \in \mathcal{C}$ then return X

$X_{i+1} \leftarrow Y$

$X_i \xleftarrow{\$} \{0,1\}^n$, if $X_i \in S_{i+1}$ then $bad \leftarrow \mathsf{true}$, $X_i \xleftarrow{\$} \{0,1\}^n \setminus S_{i+1}$

$X_{i+2} \xleftarrow{\$} \{0,1\}^n$, if $X_{i+2} \in S_{i+2}$ then $bad \leftarrow \mathsf{true}$, $X_{i+2} \xleftarrow{\$} \{0,1\}^n \setminus S_{i+2}$

$S_i \leftarrow S_i \cup \{X_i\}$, $S_{i+1} \leftarrow S_{i+1} \cup \{X_{i+1}\}$, $S_{i+2} \leftarrow S_{i+2} \cup \{X_{i+2}\}$

$\mathcal{C} \leftarrow \mathcal{C} \cup \{(i, X_i, X_{i+1}),\ (i+1, X_{i+1}, X_{i+2}),\ (i+2, X_{i+2}, X_i)\}$

return X_i

Fig. 5. Games for bounding the probability of distinguishing (π_0, π_1, π_2) and $(\pi_0, \pi_1, \pi_1^{-1} \circ \pi_0^{-1})$ by an adversary that never extends a two-chain.

over all q queries, the probability that *bad* ever gets set in game G_2 is at most $0.5q(q-1)/N \le 0.5q^2/N$. To establish Lemma 6 we now claim that

$$\left|\mathbf{Adv}(A^{H_1}, A^{G_3})\right| \le 2\, q^2/N\ . \tag{7}$$

First rewrite game H_1 as game G_4 (again in Fig. 5). Addition (+1 and +2) is again understood to be modulo 3. Game G_4 uses a form of lazy sampling, but it is not maximally lazy; on each query, not only is its answer chosen, but answers for some related queries are chosen and stored. In particular, the game maintains a set $\mathcal{C}$ of *commitments*. Initially there are no commitments, but every time a query $\boldsymbol{\Pi}(i, X)$ or $\boldsymbol{\Pi^{-1}}(i, Y)$ is asked, one of two things happens: if a commitment has already been made specifying how to answer this query, we answer according to that commitment; else we not only answer the query asked, but commit ourselves to all of the queries in a "triangle" containing the queried point. In greater detail, $(i, X, Y) \in \mathcal{C}$ (for $i \in \{0,1,2\}$ and $X, Y \in \{0,1\}^n$) means that it has already been decided that $\pi_i(X) = Y$, so a forward query $\boldsymbol{\Pi}(i, X)$ will need to be answered by Y and a backward query $\boldsymbol{\Pi^{-1}}(i, Y)$ will need to be answered by X. In effect, we grow permutations π_0, π_1, and π_2 but store their values in $\mathcal{C}$ and their domains in S_0, S_1, and S_2.

We claim that games H_1 and G_4 are adversarially indistinguishable even by an adversary that is *not* valid and asks all $6N$ possible queries. From this we know that $\Pr[A^{G_4} \Rightarrow 1] = \Pr[A^{H_1} \Rightarrow 1]$. To show this equivalence we claim that whether the queries are answered by game G_4 or by game H_1 the adversary gets the same view: any of $(N!)^2$ possible outcomes, each with probability $1/(N!)^2$, the answers correspond to a pair of permutations π_0, π_1 along with $\pi_2 = \pi_1^{-1} \circ \pi_0^{-1}$. This is obviously the case when playing game H_1; we must show it is so for game G_4. Note that sets S_0, S_1, S_2, and $\mathcal{C}$ begin with no points in them, then they grow to 1, 1, 1, and 3 points; then to 2, 2, 2, and 6 points; and so forth, until they have N, N, N, and $3N$ points. Not every query changes the sizes of these sets; it either leaves the sets unaltered or changes them as indicated. The first query that augments $\mathcal{C}$ extends the partial functions (π_0, π_1, π_2) in any of N^2 different ways, each with the same probability; the second query that augments $\mathcal{C}$ extends (π_0, π_1, π_2) in any of $(N-1)^2$ different ways, each with the same probability; and so forth, until we have extended (π_0, π_1, π_2) in any of $(N!)^2$ different ways, each with the same probability. This establishes the claim.

Now let us go back to assuming that the adversary is valid. We make a change to game G_4 to arrive at game G_5, shown in Fig. 6. In the transition, we drop the first commitment from each group of three, since our assumptions about the adversary's behavior mean that these queries cannot be asked. We also drop the sequels to *bad* getting set at lines 412, 413, 422, and 423. More interestingly, in game G_5 we maintain a set of "poisoned" queries $\mathcal{P}$. As with game G_4, when the adversary asks $\boldsymbol{\Pi}(i, X_i)$ we return a random X_{i+1}, and when the adversary asks $\boldsymbol{\Pi^{-1}}(i, X_{i+1})$ we return a random X_i, and in either case we choose a random X_{i+2} and "complete the triangle" using this point. We don't expect the adversary to ask about X_{i+2}, and, what is more, his asking will cause problems. So we record the unlikely but problematic queries involving X_{i_2} in $\mathcal{P}$. If the adversary makes a poisoned query then we set *bad*. The changes we have made can only increase the probability that *bad* gets set: $\Pr[A^{G_4} \text{ sets } bad\,] \le \Pr[A^{G_5} \text{ sets } bad\,]$.

We claim that game G_5 is adversarially indistinguishable from game G_3. Remember that our adversary is valid: it does not ask queries whose answers are trivially known and it does not ask to extend any 2-chain. Suppose first that the adversary asks a query whose answer has not been memoized in a commitment. Then for a forward query, we choose a uniform value X_{i+1} at line 514 and return it at line 519. Likewise for a backward query, we choose a uniform value X_i at line 524 and return it at line 529. So consider instead a query for which a commitment has been memoized. The code executes at lines 511–512 or lines 521–522. If the memoized query was poisoned—added to set $\mathcal{P}$ by an earlier execution of lines 518 or 528—then we return a random string (at

procedure $\boldsymbol{\Pi}(i, X)$ Game G_5
510 if $\exists\,(i, X, Y) \in \mathcal{C}$ then
511 if $(+1, i, X) \in \mathcal{P}$ then $bad \leftarrow \mathsf{true}$, $Y \xleftarrow{\$} \{0,1\}^n$
512 return Y
513 $X_i \leftarrow X$
514 $X_{i+1} \xleftarrow{\$} \{0,1\}^n$, if $X_{i+1} \in S_{i+1}$ then $bad \leftarrow \mathsf{true}$
515 $X_{i+2} \xleftarrow{\$} \{0,1\}^n$, if $X_{i+2} \in S_{i+2}$ then $bad \leftarrow \mathsf{true}$
516 $S_i \leftarrow S_i \cup \{X_i\}$, $S_{i+1} \leftarrow S_{i+1} \cup \{X_{i+1}\}$, $S_{i+2} \leftarrow S_{i+2} \cup \{X_{i+2}\}$
517 $\mathcal{C} \leftarrow \mathcal{C} \cup \{(i+1, X_{i+1}, X_{i+2}),\ (i+2, X_{i+2}, X_i)\}$
518 $\mathcal{P} \leftarrow \mathcal{P} \cup \{(1, i+2, X_{i+2}), (-1, i+1, X_{i+2})\}$
519 return X_{i+1}

procedure $\boldsymbol{\Pi}^{-1}(i, Y)$
520 if $\exists\,(i, X, Y) \in \mathcal{C}$ then
521 if $\exists\,(-1, i, Y) \in \mathcal{P}$ then $bad \leftarrow \mathsf{true}$, $X \xleftarrow{\$} \{0,1\}^n$
522 return X
523 $X_{i+1} \leftarrow Y$
524 $X_i \xleftarrow{\$} \{0,1\}^n$, if $X_i \in S_{i+1}$ then $bad \leftarrow \mathsf{true}$
525 $X_{i+2} \xleftarrow{\$} \{0,1\}^n$, if $X_{i+2} \in S_{i+2}$ then $bad \leftarrow \mathsf{true}$
526 $S_i \leftarrow S_i \cup \{X_i\}$, $S_{i+1} \leftarrow S_{i+1} \cup \{X_{i+1}\}$, $S_{i+2} \leftarrow S_{i+2} \cup \{X_{i+2}\}$
527 $\mathcal{C} \leftarrow \mathcal{C} \cup \{(i+1, X_{i+1}, X_{i+2}),\ (i+2, X_{i+2}, X_i)\}$
528 $\mathcal{P} \leftarrow \mathcal{P} \cup \{(1, i+2, X_{i+2}), (-1, i+1, X_{i+2})\}$
529 return X_i

Fig. 6. Game G_5.

line 511 or 521). If the memoized query was not poisoned, then we are extending a 1-chain, providing a value X_{i+2} that was selected uniformly from $\{0,1\}^n$ at an earlier execution of line 515 or 525, with this value not yet having influenced the run. Thus we return a uniform random value, independent of all oracle responses so far, and $\Pr[A^{G_5} \Rightarrow 1] = \Pr[A^{G_3} \Rightarrow 1]$.

Finally, we must bound the probability that *bad* gets set in game G_5. The probability that *bad* ever gets set at any of lines 514, 515, 524, or 525 is at most $2(1 + 2 + \cdots + (q-1))/N \le q^2/N$. The probability that it gets set at lines 511 or 521 is at most $2(1 + 2 + \cdots + (q-1))/N$ because no information about the poisoned query is surfaced to the adversary. Overall we have that $\Pr[A^{G_5}$ sets $bad\,] \le 2q^2/N$. Putting everything together we have (7) and the proof of the lemma is complete. ▮

4.6 Proof of Lemma 4

To prove this lemma we can assume without loss of generality that B is deterministic. For any particular blockcipher $E \in \mathrm{Bloc}(k, n)$ we consider the game in which B is executed with oracles E, E^{-1}, which it queries, adaptively, until it halts. Note that there is no randomness involved in this game, since E is fixed and B is deterministic. Recall that $X \xrightarrow{K} Y$ means that B has either made query $E(K,X)$ and obtained Y as a result, or it has made query $E^{-1}(K,Y)$ and obtained X as a result, for $K \in \{0,1\}^k$ and $X,Y \in \{0,1\}^n$. Now we let

$$\mathsf{Ch}_3^{E,B} = \left| \{ (K_0,K_1,K_2,P) \;:\; \exists\, Q,R,S\, [\, P \overset{K_0}{\to} Q \overset{K_1}{\to} R \overset{K_2}{\to} S \,]\, \} \right| .$$

This is the number of *3-chains* created by B's queries. Here $K_0,K_1,K_2 \in \{0,1\}^k$ are keys, and $P,Q,R,S \in \{0,1\}^n$. As the notation indicates, $\mathsf{Ch}_3^{E,B}$ is a number that depends on E and B. Regarding it as a random variable over the choice of E we have the following lemma, from which Lemma 4 will follow.

Lemma 7. *Let* $\alpha = \max(2e2^{k-n}, 2n+k)$. *Then* $\mathbf{E}[\mathsf{Ch}_3^{E,B}] < 2\alpha \cdot q^2$, *the expectation over* $E \xleftarrow{\$} \mathrm{Bloc}(k,n)$. ▮

Proof (Lemma 4). Consider the following game L^E parameterized by a blockcipher $E \in \mathrm{Bloc}(k,n)$: adversary B is executed with oracles E, E^{-1} until it halts, then K_0,K_1,K_2 are chosen at random from $\{0,1\}^k$, and flag *bad* is set if there exist P,Q,R,S such that $P \overset{K_0}{\to} Q \overset{K_1}{\to} R \overset{K_2}{\to} S$. Let $p^{E,B} = \Pr[L_B^E \text{ sets } bad\,]$, the probability being over the random choices of K_0,K_1,K_2. Then for any $E \in \mathrm{Bloc}(k,n)$ we have

$$\begin{aligned} p^{E,B} &= \Pr\left[\exists\, P,Q,R,S \;:\; P \overset{K_0}{\to} Q \overset{K_1}{\to} R \overset{K_2}{\to} S\right] \\ &= \frac{|\{ (K_0,K_1,K_2) \;:\; \exists\, P,Q,R,S \;:\; P \overset{K_0}{\to} Q \overset{K_1}{\to} R \overset{K_2}{\to} S \}|}{2^{3k}} \\ &\le \frac{\sum_P |\{ (K_0,K_1,K_2) \;:\; \exists\, Q,R,S \;:\; P \overset{K_0}{\to} Q \overset{K_1}{\to} R \overset{K_2}{\to} S \}|}{2^{3k}} \;=\; \frac{\mathsf{Ch}_3^{E,B}}{2^{3k}} . \end{aligned}$$

By Lemma 7 we have $\Pr[B^L \text{ sets } bad\,] = \mathbf{E}[p^{E,B}] \le \mathbf{E}[\mathsf{Ch}_3^{E,B}]\cdot 2^{-3k} < 2\alpha\, q^2\, 2^{-3k}$ where $\alpha = \max(2e2^{k-n}, 2n+k)$ and the expectation is over $E \xleftarrow{\$} \mathrm{Bloc}(k,n)$. ▮

Towards the proof of Lemma 7, for $E \in \mathrm{Bloc}(k,n)$ and $Q,R \in \{0,1\}^n$ we let

$$\mathsf{Keys}^E(Q,R) \;=\; |\{ K \;:\; E(K,Q) = R \}| \quad \text{and} \quad \mathsf{Keys}^E \;=\; \max_{Q,R}\{\mathsf{Keys}^E(Q,R)\} .$$

The first is the number of keys for which Q maps to R under E, and the second is the maximum value of $\mathsf{Keys}^E(Q,R)$ over all $Q,R \in \{0,1\}^n$. No adversary is involved in this definition; Keys^E is simply a number associated to a given blockcipher. Viewing it as a random variable over the choice of blockcipher we have the following.

Lemma 8. *Suppose* $\beta \ge 2e2^{k-n}$. *Then* $\Pr\left[\mathsf{Keys}^E \ge \beta\right] \;<\; 2^{2n+1-\beta}$, *where the probability is over* $E \xleftarrow{\$} \mathrm{Bloc}(k,n)$. ▮

Proof (Lemma 8). We claim that for any $Q, R \in \{0,1\}^n$

$$\Pr\left[\,\mathsf{Keys}^E(Q,R) \geq \beta\,\right] \;<\; 2^{1-\beta}\;. \tag{8}$$

The lemma follows via the union bound. We prove (8) using an occupancy-problem approach. Let $b = \lceil \beta \rceil$. Then

$$\begin{aligned}\Pr\left[\,\mathsf{Keys}^E(Q,R) \geq \beta\,\right] &= \textstyle\sum_{i=b}^{2^k} \binom{2^k}{i} \left(\frac{1}{2^n}\right)^i \left(1 - \frac{1}{2^n}\right)^{2^k - i} \\ &\leq \textstyle\sum_{i=b}^{2^k} \left(\frac{2^k e}{i}\right)^i \left(\frac{1}{2^n}\right)^i \;\leq\; \sum_{i=b}^{2^k} \left(\frac{2^k e}{2^n b}\right)^i\;.\end{aligned}$$

Let $x = (e/b)2^{k-n}$. The assumption $\beta \geq 2e2^{k-n}$ gives $x \leq 1/2$. So the above is

$$=\; \textstyle\sum_{i=b}^{2^k} x^i \;<\; x^b \cdot \sum_{i=0}^{\infty} x^i \;=\; \frac{x^b}{1-x} \;\leq\; \frac{2^{-b}}{1-1/2} \;=\; 2^{1-b} \;\leq\; 2^{1-\beta}$$

as desired. ∎

Proof (Lemma 7). For any $Q, R \in \{0,1\}^n$ we let

$$\begin{aligned}\mathsf{Ch}_2^{E,B}(R) &= |\{\,(K_0,K_1,P) \;:\; \exists\, Q\,[P \xrightarrow{K_0} Q \xrightarrow{K_1} R\,]\,\}| \\ \mathsf{Ch}_1^{E,B}(Q) &= |\{\,(K_0,P) \;:\; P \xrightarrow{K_0} Q\,\}| \\ \mathsf{Ch}_0^{E,B}(R) &= |\{\,K_2 \;:\; \exists\, S\,[R \xrightarrow{K_2} S\,]\,\}|\;.\end{aligned}$$

Then for any $E \in \mathrm{Bloc}(k,n)$ we have

$$\begin{aligned}\mathsf{Ch}_3^{E,B} &= \textstyle\sum_R \mathsf{Ch}_2^{E,B}(R) \cdot \mathsf{Ch}_0^{E,B}(R) \\ &\leq \textstyle\sum_R \left(\sum_Q \mathsf{Ch}_1^{E,B}(Q) \cdot \mathsf{Keys}^E(Q,R)\right) \cdot \mathsf{Ch}_0^{E,B}(R) \\ &\leq \textstyle\sum_R \left(\sum_Q \mathsf{Ch}_1^{E,B}(Q) \cdot \mathsf{Keys}^E\right) \cdot \mathsf{Ch}_0^{E,B}(R) \\ &= \mathsf{Keys}^E \cdot \left(\textstyle\sum_Q \mathsf{Ch}_1^{E,B}(Q)\right) \cdot \left(\textstyle\sum_R \mathsf{Ch}_0^{E,B}(R)\right) \\ &\leq \mathsf{Keys}^E \cdot q \cdot q \;=\; q^2 \cdot \mathsf{Keys}^E\;.\end{aligned}$$

Using the above and Lemma 8, we have the following, where the probability and expectation are both over $E \stackrel{\$}{\leftarrow} \mathrm{Bloc}(k,n)$:

$$\begin{aligned}\mathbf{E}[\mathsf{Ch}_3^{E,B}] &< \mathbf{E}\left[\mathsf{Ch}_3^{E,B} \;\mid\; \mathsf{Keys}^E < \alpha\right] + \mathbf{E}\left[\mathsf{Ch}_3^{E,B} \;\mid\; \mathsf{Keys}^E \geq \alpha\right] \cdot 2^{2n+1-\alpha} \\ &\leq q^2 \cdot \alpha + q^2 \cdot 2^k \cdot 2^{2n+1-\alpha}\;.\end{aligned}$$

The last inequality above used the fact that Keys^E is always at most 2^k. Since $\alpha = \max(2e2^{k-n}, 2n+k) > 2$ we get $\mathbf{E}[\mathsf{Ch}_3^{E,B}] < q^2\alpha + q^2 \cdot 2 < 2\alpha \cdot q^2$ as desired. ∎

Acknowledgments

We thank the Eurocrypt 2006 PC for their comments. Mihir Bellare was supported by NSF grants CCR-0208842 and CNS-0524765. Phil Rogaway was supported by NSF 0208842 and a gift from Intel Corp. Much of the work on this paper was carried out while Phil was hosted by Chiang Mai University, Thailand.

References

1. W. Aiello, M. Bellare, G. Di Crescenzo, and R. Venkatesan. Security amplification by composition: the case of doubly-iterated, ideal ciphers. *Advances in Cryptology — CRYPTO '98*, Lecture Notes in Computer Science, vol 1462, Springer, pp. 390–407, 1998.
2. M. Bellare and S. Goldwasser. New paradigms for digital signatures and message authentication based on non-interactive zero knowledge proofs. *Advances in Cryptology — CRYPTO '89*, Lecture Notes in Computer Science, vol. 435, Springer, pp. 194–211, 1990.
3. M. Bellare and P. Rogaway. Code-based game-playing proofs and the security of triple encryption. Cryptology ePrint archive report 2004/331, 2006.
4. W. Diffie and M. Hellman. Exhaustive cryptanalysis of the data encryption standard. *Computer*, vol. 10, pp. 74–84, 1977.
5. S. Even and O. Goldreich. On the power of cascade ciphers. *ACM Transactions on Computer Systems*, vol. 3, no. 2, pp. 108–116, 1985.
6. S. Even and Y. Mansour. A construction of a cipher from a single pseudorandom permutation. *Advances in Cryptology — ASIACRYPT '91*, Lecture Notes in Computer Science, vol.739, Springer, pp. 210–224, 1993.
7. S. Goldwasser and S. Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, vol. 28, no. 2, pp. 270–299, 1984. Earlier version in *STOC '82*.
8. S. Halevi. A plausible approach to computer-aided cryptographic proofs. Cryptology ePrint archive report 2005/181, 2005.
9. J. Kilian and P. Rogaway. How to protect DES against exhaustive key search (an analysis of DESX). *J. of Cryptology*, vol. 14, no. 1, pp. 17–35, 2001. Earlier version in *Crypto '96*.
10. S. Lucks. Attacking triple encryption. *Fast Software Encryption (FSE '98)*, Lecture Notes in Computer Science, vol. 1372, Springer, pp. 239–253, 1998.
11. U. Maurer and J. Massey. Cascade ciphers: the importance of being first. *J. of Cryptology*, vol. 6, no. 1, pp. 55–61, 1993.
12. R. Merkle and M. Hellman. On the security of multiple encryption. *Communications of the ACM*, vol. 24, pp. 465–467, 1981.
13. National Institute of Standards and Technology. FIPS PUB 46-3, Data Encryption Standard (DES), 1999. Also ANSI X9.52, Triple Data Encryption Algorithm modes of operation, 1998, and other standards.
14. C. Shannon. Communication theory of secrecy systems. *Bell Systems Technical Journal*, vol. 28, no. 4, pp. 656–715, 1949.
15. V. Shoup. Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint archive report 2004/332, 2006.
16. A. Yao. Theory and applications of trapdoor functions. *IEEE Symposium on the Foundations of Computer Science (FOCS 1982)*, IEEE Press, pp. 80–91, 1982.