Online Ciphers and the Hash-CBC Construction

Mihir Bellare[1], Alexandra Boldyreva[1], Lars Knudsen[2], and Chanathip Namprempre[1]

[1] Department of Computer Science & Engineering
University of California, San Diego
La Jolla, California 92093
{mihir,aboldyre,meaw}@cs.ucsd.edu
http://www-cse.ucsd.edu/users/{mihir,aboldyre,cnamprem}
[2] Department of Informatics
PB 7800, N-5020 Bergen, Norway
lars@ramkilde.com
http://www.ramkilde.com

Abstract. We initiate a study of on-line ciphers. These are ciphers that can take input plaintexts of large and varying lengths and will output the ith block of the ciphertext after having processed only the first i blocks of the plaintext. Such ciphers permit length-preserving encryption of a data stream with only a single pass through the data. We provide security definitions for this primitive and study its basic properties. We then provide attacks on some possible candidates, including CBC with fixed IV. Finally we provide a construction called HCBC which is based on a given block cipher E and a family of AXU functions. HCBC is proven secure against chosen-plaintext attacks assuming that E is a PRP secure against chosen-plaintext attacks.

1 Introduction

We begin by saying what we mean by on-line ciphers. We then describe a notion of security for them, and discuss constructions and analyses. Finally, we discuss usage, applications, and related work.

1.1 Online Ciphers

A *cipher* over domain D is a function F: $\{0,1\}^k \times D \to D$ such that for each key K the map $F(K,\cdot)$ is a length-preserving permutation on D, and possession of K enables one to both compute and invert $F(K,\cdot)$. The most popular examples are block ciphers, where $D = \{0,1\}^n$ for some n called the block length; these are fundamental tools in cryptographic protocol design. However, one might want to encipher data of large size, in which case one needs a cipher whose domain D is appropriately large. (A common choice, which we make, is to set the domain to $D_{d,n}$, the set of all strings having a length that is at most some large value d, and is also divisible by n.) Matyas and Meyer refer to these as "general" ciphers [10].

J. Kilian (Ed.): CRYPTO 2001, LNCS 2139, pp. 292–309, 2001.

In this paper, we are interested in general ciphers that are computable in an on-line manner. Specifically, cipher F is said to be *on-line* if the following is true. View the input plaintext $M = M[1] \dots M[l]$ to an instance $F(K, \cdot)$ of the cipher as a sequence of n-bit blocks, and similarly for the output ciphertext $F(K, M) = C[1] \dots C[l]$. Then, given the key K, for all i, it should be possible to compute output block $C[i]$ after having seen input blocks $M[1] \dots M[i]$. That is, $C[i]$ does not depend on blocks $i + 1, \dots, l$ of the plaintext.

An on-line cipher permits real-time, length-preserving encryption of a data stream without recourse to buffering, which can be attractive in some practical settings.

The intent of this paper is to find efficient, proven secure constructions of on-line ciphers and to further explore the applications. Let us now present the relevant security notions and our results.

1.2 A Notion of Security for Online Ciphers

A commonly accepted notion of security to target for a cipher is that it be a pseudorandom permutation (PRP), as defined by Luby and Rackoff [9]. Namely, for a cipher F to be a PRP, it should be computationally infeasible, given an oracle g, to have non-negligible advantage in distinguishing between the case where g is a random instance of F and the case where g is a randomly-chosen, length-preserving permutation on the domain of the cipher. However, if a cipher is on-line, then the ith block of the ciphertext does not depend on blocks $i + 1, i + 2, \dots$ of the plaintext. This is necessary, since otherwise it would not be possible to output the ith ciphertext block having seen only the first i plaintext blocks. Unfortunately, this condition impacts security, since a cipher with this property certainly cannot be a PRP. An easy distinguishing test is to ask the given oracle g the two-block queries AB and AC, getting back outputs WX and YZ respectively, and if $W = Y$ then bet that g is an instance of the cipher. This test has a very high advantage since the condition being tested fails with high probability for a random length-preserving permutation.

For an on-line cipher, then, we must give up on the requirement that it meet the security property of being a PRP. Instead, we define and target an appropriate alternative notion of security. This is quite natural; we simply ask that the cipher behave "as randomly as possible" subject to the constraint of being on-line. We say that a length-preserving permutation π is *on-line* if for all i the ith output block of π depends only on the first i input blocks to π, and let $\mathsf{OPerm}_{d,n}$ denote the set of all length-preserving permutations π on domain $D_{d,n}$. The rest is like for a PRP, with members of this new set playing the role of the "ideal" objects to which cipher instances are compared: it should be computationally infeasible, given an oracle g, to have non-negligible advantage in distinguishing between the case where g is a random instance of F and the case where g is a random member of $\mathsf{OPerm}_{d,n}$. A cipher secure in this sense is called an on-line-PRP.

The fact that an on-line-PRP meets a notion of security that is relatively weak compared to a PRP might at first lead one to question the introduction

of such a notion. However, finding appropriate balances between security and practical constraints is an impactful and active research endeavor where the goal is not necessarily to achieve some strong notion of security but to have the "best possible" security under given practical constraints, so that weaker notions of security are useful. Furthermore, we will see that in this case, even this weak primitive, if properly used, can provide strong security.

1.3 Candidates for Online Ciphers

To the best of our knowledge, the problem of designing on-line ciphers with security properties as strong as those required by our definition has not been explicitly addressed before. When one comes to consider this problem, however, it is natural to test first some existing candidate ciphers or natural constructions from the literature. We consider some of them and present attacks that are helpful to gather intuition about the kinds of security properties we are seeking.

It is natural to begin with standard modes of operation of a block cipher, such as CBC. However, CBC is an encryption scheme, not a cipher; each invocation chooses a new random initial vector as a starting point and makes this part of the ciphertext. In particular, it is not length-preserving. The natural way to modify it to be a cipher is to fix the initial vector. There are a couple of choices: make it a known public value, or, hopefully better for security, make it a key that will be part of the secret key of the cipher. The resulting ciphers are certainly on-line, but they do not meet the notion of security we have defined. In other words, the CBC cipher with fixed IV, whether public or private, can be easily distinguished from a random on-line permutation. Attacks demonstrating this are provided in Section 4.

We then consider the Accumulated Block Chaining (ABC) mode proposed by Knudsen in [7], which is a generalization of the Infinite Garble Extension mode proposed by Campbell [5]. It was designed to have "infinite error propagation," a property that intuitively seems necessary for a secure on-line cipher but which, as we will see, is not sufficient. In Section 4, we present attacks demonstrating that this is not a secure on-line cipher.

1.4 The HCBC Online Cipher and Its Security

We seek a construction of a secure on-line cipher based on a given block cipher $E\colon\ \{0,1\}^{ek}\times\{0,1\}^n \to \{0,1\}^n$. We provide a construction called HCBC that uses a family $H\colon\ \{0,1\}^{hk}\times\{0,1\}^n \to \{0,1\}^n$ of Almost-XOR-Universal (AXU) hash functions [8]. The key $eK\|hK$ for an instance $\mathsf{HCBC}(eK\|hK,\cdot)$ of the cipher consists of a key eK for the block cipher and a key hK specifying a member $H(hK,\cdot)$ of the family H. The construction is just like CBC, except that a ciphertext block is first hashed via $H(hK,\cdot)$ before being XORed with the next plaintext block. (The initial vector is fixed to 0^n.) A picture is in Figure 3, and a full description of the construction is in Section 6. It is easy to see that this cipher is on-line.

We stress that the hash functions map n bits to n bits, meaning work on inputs of the block length, as does the given block cipher. Numerous designs of fast AXU families are known, so that our construction is quite efficient. For an overview of the state-of-the-art of AXU families refer to [12].

We prove that HCBC meets the notion of security for an on-line cipher that we discussed above, assuming that the underlying block cipher E is a PRP. The proof involves finding and exploiting a way of looking at an on-line cipher as a 2^n-ary tree of permutations on n bits, and then going through a hybrid argument involving a sequence of different games that "move" from $\mathsf{OPerm}_{d,n}$ to HCBC.

1.5 Security against Chosen Ciphertext Attacks

The notions of PRPs and on-line PRPs that we have discussed above represent security under chosen-plaintext attack. A stronger requirement is security under chosen-ciphertext attack. For a PRP this means that the adversary has an oracle not just for the challenge permutation, but also for its inverse. (An object secure in this sense was called a strong PRP in [11] and a super-PRP in [9].) This notion is easily adapted to yield a notion of on-line PRPs secure against chosen-ciphertext attack. We provide an attack showing that HCBC is not secure against chosen-ciphertext attack. The question of finding a construction of an on-line PRP secure against chosen ciphertext attack, based on a block cipher assumed to be a PRP secure against chosen-ciphertext attack, is open. In the full version of this paper [1] we report on some efforts to this end.

1.6 Usage and Application of Online Ciphers

There are settings in which the input plaintext is being streamed to a device that has limited memory for buffering and wants to produce output at the same rate at which it is getting input. The on-line property becomes desirable in these settings. The most direct usage of an on-line cipher will be in settings where, additionally, there is a constraint requiring the length of the ciphertext to equal the length of the plaintext. (Otherwise, one can use a standard mode of encryption like CBC, since it has the on-line property. But it is length expanding in the sense that the length of the ciphertext exceeds that of the plaintext, due to the changing initial vector.) This type of constraint occurs when one is dealing with fixed packet formats or legacy code.

However, an on-line cipher is more generally useful, via the "encode-then-encipher" paradigm discussed in [4]. This paradigm was presented for ciphers that are PRPs, and says that enciphering yields an IND-CPA secure encryption scheme if the message space has enough entropy, and provides integrity (meaning achieves INT-CTXT) if the message space contains enough redundancy. (The privacy requires that the PRP be secure against chosen-plaintext attack, while the integrity requires security against chosen-ciphertext attack.) Entropy and redundancy might be present in the data, as often happens when enciphering structured data like packets, which have fixed formats and often contain counters. Or, entropy and redundancy can be explicitly added, for example by inserting a

random value and a constant string in the message. (This will of course increase the size of the plaintext, so is only possible when data expansion is permitted.)

Claims similar to those made in [4] remain true even if the cipher is an on-line-PRP rather than a PRP. Specifically, the requirement on the message space must be strengthened to require not just that entropy be present, but that it be in the first blocks of the message; and similarly, that redundancy not just be present, but be at the end of the data. Again, one might already have data of such structure, in which case the encryption will be length preserving yet provide semantic security and integrity, or one can prepend a random number and append a constant to the message, getting the same properties but at the cost of data expansion.

1.7 Related Work

The problem addressed by our Hash-CBC construction is that of building a general cipher from a block cipher. Naor and Reingold [11] consider this problem for the case where the general cipher is to be a PRP or strong PRP, while we want the general cipher to be an on-line-PRP or strong-on-line-PRP. The constructions of [11, Section 7] are not on-line; indeed, they cannot be, since they achieve the stronger security notion of a PRP. Our construction, however, follows that of [11] in using hash functions in combination with block ciphers. A problem that has received a lot of attention is to take a PRP and produce another having twice the input block length of the original [9,11]. We are, however, interested in allowing inputs of varying and very large size, not merely twice the block size.

2 Definitions

We recall basic definitions of families of functions and ciphers following [2].

NOTATION. A *string* is a member of $\{0,1\}^*$. If x is a string, then $|x|$ denotes its length. The empty string is denoted ε. If $x, y \in \{0,1\}^*$ are strings, then we denote by $\mathsf{LCP}_n(x,y)$ the *longest common n-prefix* of x, y. This is the longest string s such that $|s|$ is a multiple of n, and s is a prefix of both x and y. A map $f\colon\ D \to R$ is a *permutation* if $D = R$ and f is a bijection (i.e. one-to-one and onto). A map $f\colon\ D \to R$ is *length-preserving* if $|f(x)| = |x|$ for all $x \in D$. If $n \geq 1, d \geq 1$ are integers, then $D_{d,n}$ denotes the set of all strings whose length is a positive multiple of n bits and at most dn bits. If $P \in D_{d,n}$, then $P[i]$ denotes its ith block, meaning $P = P[1] \ldots P[l]$ where $l = |P|/n$ and $|P[i]| = n$ for all $i = 1, \ldots, l$. We will typically consider functions whose inputs and outputs are in $D_{d,n}$, so that both are viewed as sequences of blocks where each block is n bits long. We let $f^{(i)}$ denote the function which on input M returns the ith block of $f(M)$. (Or ε if $|f(M)| < ni$.)

FUNCTION FAMILIES AND CIPHERS. A *family of functions* is a map $F\colon\ \mathit{Keys}(F) \times \mathit{Dom}(F) \to \mathit{Ran}(F)$ where $\mathit{Keys}(F)$ is the *key space* of F; $\mathit{Dom}(F)$ is the *domain* of F; and $\mathit{Ran}(F)$ is the *range* of F. If $\mathit{Keys}(F) = \{0,1\}^k$, then we refer to k as

the key-length. The two-input function F takes a key $K \in \mathit{Keys}(F)$ and an input $x \in \mathit{Dom}(F)$ to return a point $F(K,x) \in \mathit{Ran}(F)$. For each key $K \in \mathit{Keys}(F)$, we define the map $F_K\colon \mathit{Dom}(F) \to \mathit{Ran}(F)$ by $F(K,x)$ for all $x \in \mathit{Dom}(F)$. Thus, F specifies a collection of maps from $\mathit{Dom}(F)$ to $\mathit{Ran}(F)$, each map being associated with a key. (That is why F is called a family of functions.) We refer to $F(K,\cdot)$ as an *instance* of F. The operation of choosing a key at random from the key space is denoted $K \xleftarrow{R} \mathit{Keys}(F)$. We write $f \xleftarrow{R} F$ for the operation $K \xleftarrow{R} \mathit{Keys}(F)\,;\ f \leftarrow F(K,\cdot)$. That is, $f \xleftarrow{R} F$ denotes the operation of selecting at random a function from the family F. When f is so selected it is called a *random instance* of F. Let $\mathsf{Rand}_{n,n}$ be the family of all functions mapping $\{0,1\}^n$ to $\{0,1\}^n$ so that $f \xleftarrow{R} \mathsf{Rand}_{n,n}$ denotes the operation of selecting at random a function from $\{0,1\}^n$ to $\{0,1\}^n$. Similarly, let Perm_n be the family of all permutations mapping $\{0,1\}^n$ to $\{0,1\}^n$ so that $\pi \xleftarrow{R} \mathsf{Perm}_n$ denotes the operation of selecting at random a permutation on $\{0,1\}^n$. We say that F is a *cipher* if $\mathit{Dom}(F) = \mathit{Ran}(F)$ and each instance $F(K,\cdot)$ of F is a length-preserving permutation. A *block cipher* is a cipher whose domain and range equal $\{0,1\}^n$ for some integer n called the *block size*. (For example, the AES has block size 128.) If F is a cipher, then F^{-1} is the *inverse cipher*, defined by $F^{-1}(K,x) = F(K,\cdot)^{-1}(x)$ for all $K \in \mathit{Keys}(F)$ and $x \in \mathit{Dom}(F)$.

Pseudorandomness of ciphers. A "secure" cipher is one that approximates a family of random permutations; the "better" the approximation, the more secure the cipher. This is formalized following [6,9]. A *distinguisher* is an algorithm that has access to one or more oracles and outputs a bit. Let $F\colon\ \mathit{Keys}(F) \times \{0,1\}^n \to \{0,1\}^n$ be a family of functions with domain and range $\{0,1\}^n$. Let A_1 be a distinguisher with one oracle and A_2 a distinguisher with two oracles. Let

$$\mathbf{Adv}^{\text{prp-cpa}}_F(A_1) = \Pr\left[\, g \xleftarrow{R} F \ :\ A_1^g = 1 \,\right] - \Pr\left[\, g \xleftarrow{R} \mathsf{Perm}_n \ :\ A_1^g = 1 \,\right] .$$

If $F\colon \mathit{Keys}(F) \times \{0,1\}^n \to \{0,1\}^n$ is a cipher, then we also let

$$\mathbf{Adv}^{\text{prp-cca}}_F(A_2) = \Pr\left[\, g \xleftarrow{R} F \ :\ A_2^{g,g^{-1}} = 1 \,\right] - \Pr\left[\, g \xleftarrow{R} \mathsf{Perm}_n \ :\ A_2^{g,g^{-1}} = 1 \,\right] .$$

These capture the *advantage* of the distinguisher in question in the task of distinguishing a random instance of F from a random permutation on D. In the first case, the distinguisher gets to query the challenge instance. In the second, it also gets to query the inverse of the challenge instance. For any integers $t, q_e, q_d, \mu_e, \mu_d$, we now let

$$\mathbf{Adv}^{\text{prp-cpa}}_F(t, q_e, \mu_e) = \max_{A_1} \left\{\, \mathbf{Adv}^{\text{prp-cpa}}_F(A_1) \,\right\}$$

$$\mathbf{Adv}^{\text{prp-cca}}_F(t, q_e, \mu_e, q_d, \mu_d) = \max_{A_2} \left\{\, \mathbf{Adv}^{\text{prp-cca}}_F(A_2) \,\right\} .$$

The maximum is over all distinguishers having time-complexity t, making to the g oracle at most q_e queries totaling at most μ_e bits, and, in the second case, also making to the g^{-1} oracle at most q_d queries totaling at most μ_d bits. We say that a PRP F is *secure against chosen-plaintext attacks* if the function $\mathbf{Adv}^{\text{prp-cpa}}_F(t, q_e)$ grows "slowly." Similarly, we say that a PRP F is *se-*

cure against chosen-ciphertext attacks if the function $\mathbf{Adv}_F^{\text{prp-cca}}(t, q_e, q_d)$ grows "slowly." Time complexity includes the time to reply to oracle calls by computation of $F(K,\cdot)$ or $F(K,\cdot)^{-1}$.

3 Online Ciphers and Their Basic Properties

We say that a function $f\colon D_{d,n} \to D_{d,n}$ is n-on-line if the i-th block of the output is determined completely by the first i blocks of the input. A more formal definition follows. We refer the reader to Section 2 for the definition of $f^{(i)}$.

Definition 1. Let $n, d \geq 1$ be integers, and let $f\colon D_{d,n} \to D_{d,n}$ be a length-preserving function. We say that f is *n-on-line* if there exists a function $X\colon D_{d,n} \to \{0,1\}^n$ such that for every $M \in D_{d,n}$ and every $i \in \{1,\ldots,|M|/n\}$ it is the case that

$$f^{(i)}(M) \;=\; X(M[1]\ldots M[i])\,.$$

A cipher F having domain and range a subset of $D_{d,n}$ is said to be n-on-line if for every $K \in \mathit{Keys}(F)$ the function $F(K,\cdot)$ is on-line. ∎

Definition 2. Let f be an n-on-line function. Let $i \geq 1$. Fix $M[1],\ldots,M[i-1] \in \{0,1\}^n$. Define the function $\Pi^f_{M[1]\ldots M[i-1]}\colon \{0,1\}^n \to \{0,1\}^n$ by

$$\Pi^f_{M[1]\ldots M[i-1]}(x) = f^{(i)}(M[1]\ldots M[i-1]x)$$

for all $x \in \{0,1\}^n$. ∎

Proposition 1. *If f is an n-on-line permutation, $i \geq 1$ and $M[1],\ldots,M[i-1] \in \{0,1\}^n$, then the map $\Pi^f_{M[1]\ldots M[i-1]}$ is a permutation on $\{0,1\}^n$.*

The proof of proposition 1 is in the full version of this paper [1].

Pseudorandomness of on-line ciphers. Let $\mathsf{OPerm}_{d,n}$ denote the family of all n-on-line, length-preserving permutations on $D_{d,n}$. A "secure" on-line cipher is one that closely approximates $\mathsf{OPerm}_{d,n}$; the "better" the approximation, the more "secure" the on-line cipher. This formalization is analogous to the previously presented formalization of the pseudorandomness of ciphers. Let $F\colon \mathit{Keys}(F) \times D_{d,n} \to D_{d,n}$ be a family of functions with domain and range $D_{d,n}$. Let A_1 be a distinguisher with one oracle and A_2 a distinguisher with two oracles. Let

$$\mathbf{Adv}_F^{\text{oprp-cpa}}(A_1) = \Pr\left[\, g \xleftarrow{R} F \;:\; A_1^g = 1 \,\right] - \Pr\left[\, g \xleftarrow{R} \mathsf{OPerm}_{d,n} \;:\; A_1^g = 1 \,\right].$$

If $F\colon \mathit{Keys}(F) \times \{0,1\}^n \to \{0,1\}^n$ is a cipher, then we also let

$$\mathbf{Adv}_F^{\text{oprp-cca}}(A_2) = \Pr\left[g \xleftarrow{R} F \;:\; A_2^{g,g^{-1}} = 1\right] - \Pr\left[g \xleftarrow{R} \mathsf{OPerm}_{d,n} \;:\; A_2^{g,g^{-1}} = 1\right].$$

These capture the *advantage* of the distinguisher in question in the task of distinguishing a random instance of F from a random, length-preserving, n-on-line

permutation on $D_{d,n}$. In the first case, the distinguisher gets to query the challenge instance. In the second, it also gets to query the inverse of the challenge instance. For any integers $t, q_e, \mu_e, q_d, \mu_d$, we now let

$$\mathbf{Adv}^{\text{oprp-cpa}}_F(t, q_e, \mu_e) = \max_{A_1} \left\{ \mathbf{Adv}^{\text{oprp-cpa}}_F(A_1) \right\}$$

$$\mathbf{Adv}^{\text{oprp-cca}}_F(t, q_e, \mu_e, q_d, \mu_d) = \max_{A_2} \left\{ \mathbf{Adv}^{\text{oprp-cca}}_F(A_2) \right\} .$$

The maximum is over all distinguishers having time-complexity t, making to the oracle g at most q_e queries totaling at most μ_e bits, and, in the second case, also making to the g^{-1} oracle at most q_d queries totaling at most μ_d bits. We say that an online PRP (OPRP) F is secure against chosen plaintext attacks if the function $\mathbf{Adv}^{\text{oprp-cpa}}_F(t, q_e, \mu_e)$ grows "slowly." Similarly, we say that an OPRP F is secure against chosen ciphertext attacks if the function $\mathbf{Adv}^{\text{oprp-cca}}_F(t, q_e, \mu_e, q_d, \mu_d)$ grows "slowly." Time complexity includes the time to reply to oracle calls by computation of $F(K, \cdot)$ or $F(K, \cdot)^{-1}$.

Tree-based characterization. We present a tree-based characterization of n-on-line ciphers that is useful to gain intuition and to analyze constructs. Let $N = 2^n$. An N-ary tree of functions is an N-ary tree T each node of which is labeled by a function mapping $\{0,1\}^n$ to $\{0,1\}^n$. We label each edge in the tree in a natural way via a string in $\{0,1\}^n$. Then, each node in the tree is described by a sequence of edge labels defining the path from the root to the node in question. The function labeling node x in the tree, where x is a string of length ni for some $0 \le i \le d$, is then denoted T_x. A tree defines a function T from $D_{d,n}$ to $D_{d,n}$ as described below. If the nodes in the tree are labeled with permutations, then the tree also defines an inverse function T^{-1}.

$T(M[1] \dots M[l])$	$T^{-1}(C[1] \dots C[l])$
$x \leftarrow \varepsilon$	$x \leftarrow \varepsilon$
For $i = 1, \dots, l$ do	For $i = 1, \dots, l$ do
$\quad C[i] \leftarrow T_x(M[i])$	$\quad M[i] \leftarrow T_x^{-1}(C[i])$
$\quad x \leftarrow x \| C[i]$	$\quad x \leftarrow x \| C[i]$
EndFor	EndFor
Return $C[1] \dots C[l]$	Return $M[1] \dots M[l]$

Here, $1 \le l \le d$. Let $G : \mathit{Keys}(G) \times \{0,1\}^n \to \{0,1\}^n$ be a function family. (We are most interested in the case where G is Perm_n or $\mathsf{Rand}_{n,n}$.) We let $\mathsf{Tree}(n, G, d)$ denote the set of all 2^n-ary trees of functions in which each function is an instance of G and the depth of the tree is d. This set is viewed as equipped with a distribution under which each node of the tree is assigned a random instance of G, and the assignments to the different nodes are independent. We claim that a tree-based construction defined above is a valid characterization of on-line ciphers, as stated in the following proposition and proven in [1].

Proposition 2. *There is a bijection between* $\mathsf{Tree}(n, \mathsf{Perm}_n, d)$ *and* $\mathsf{OPerm}_{d,n}$.

Inversion. It turns out that the inverse of an on-line permutation is itself on-line, as stated below and proven in [1].

Proposition 3. *Let $f\colon D_{d,n} \to D_{d,n}$ be an n-on-line permutation, and let $g = f^{-1}$. Then g is an n-on-line permutation.*

We note that the proof does not tell us anything about the computational complexity of function f^{-1}, meaning it could be the case that f is efficiently computable, but the f^{-1} given by Proposition 3 is not. However, whenever we design a cipher F, we will make sure that both $F(K,\cdot)$ and $F^{-1}(K,\cdot)$ are efficiently computable given K, and will explicitly specify F^{-1} in order to make this clear.

4 Analysis of Some Candidate Ciphers

We consider several candidates for on-line ciphers. First, we consider one based on the basic CBC mode. Then, we consider the Accumulated Block Chaining (ABC) proposed by Knudsen in [7], which is a generalization of the Infinite Garble Extension mode proposed by Campbell [5]. In this section, we let $E\colon \{0,1\}^{ek} \times \{0,1\}^n \to \{0,1\}^n$ be a given block cipher with key size ek and block size n.

CBC AS AN ON-LINE CIPHER. In CBC encryption based on E, one usually uses a new, random IV for every message. This does not yield a cipher, let alone an on-line one. To get an on-line cipher, we fix the IV. We can, however, make it secret; this can only increase security. In more detail, the *CBC cipher associated to E*, denoted OCBC, has key space $\{0,1\}^{ek+n}$. For $M, C \in D_{d,n}$, $eK \in \{0,1\}^{ek}$ and $C[0] \in \{0,1\}^n$, we define

$\mathsf{OCBC}(eK \| C[0], M)$	$\mathsf{OCBC}^{-1}(eK \| C[0], C)$
Parse M as $M[1] \ldots M[l]$ with $l \geq 1$	Parse C as $C[1] \ldots C[l]$ with $l \geq 1$
For $i = 1, \ldots, l$ do	For $i = 1, \ldots, l$ do
$\quad C[i] \leftarrow E(eK, M[i] \oplus C[i-1])$	$\quad M[i] \leftarrow E^{-1}(eK, C[i]) \oplus C[i-1]$
Return $C[1] \ldots C[l]$	Return $M[1] \ldots M[l]$

Here, $C[0]$ is the IV. The key is the pair $eK \| C[0]$, consisting of a key eK for the block cipher, and the IV. It is easy to check that the above cipher is on-line. For clarity, we have also shown the inverse cipher. We now present the attack. The adversary A shown in Figure 1 gets an oracle g where g is either an instance of OCBC or an instance of $\mathsf{OPerm}_{d,n}$. We claim that

$$\mathbf{Adv}^{\text{oprp-cpa}}_{\mathsf{OCBC}}(A) \geq 1 - 2^{-n} \; . \tag{1}$$

We justify Equation (1) in the full version of this paper [1]. Since A made only 3 oracle queries, this shows that the CBC mode with a fixed IV is not a secure on-line cipher.

The idea of the attack is to gather some input-output pairs for the cipher. Then we use these values to construct a new sequence of input blocks so that one of the input blocks to E collides with one of the previous input blocks to E. This enables us to predict an output block of the cipher. If our prediction is correct, then we know that the oracle is an instance of OCBC with overwhelming probability.

Distinguisher A^g
Let $M[2], \ldots, M[l]$ be any n-bit strings
Let $M_1 = 0^n M[2] \ldots M[l]$ and let $M_2 = 1^n M[2] \ldots M[l]$
Let $C_1[1] \ldots C_1[l] \leftarrow g(M_1)$ and let $C_2[1] \ldots C_2[l] \leftarrow g(M_2)$
Let $M_3[2] = M[2] \oplus C_1[1] \oplus C_2[1]$ and let $M_3 = 1^n M_3[2] M[3] \ldots M[l]$
Let $C_3[1] \ldots C_3[l] \leftarrow g(M_3)$
If $C_3[2] = C_1[2]$ then return 1 else return 0

Fig. 1. Attack on the CBC based on-line cipher.

ABC AS AN ON-LINE CIPHER. Knudsen in [7] proposes the Accumulated Block Chaining (ABC) mode of operation for block ciphers. This is an on-line cipher that is a natural starting point in the problem of finding a secure on-line cipher because it has the property of "infinite error propagation." We formalize and analyze ABC with regard to meeting our security requirements.

The mode is parameterized by initial values $P[0], C[0] \in \{0,1\}^n$ and also by a public function h: $\{0,1\}^n \to \{0,1\}^n$. (Instantiations for h suggested in [7] include the identity function, the constant function always returning 0^n, and the function which rotates its input by one bit.) We are interested in the security of the mode across various settings and choices of these parameters. (In particular, we want to consider the case where the initial values are public and also the case where they are secret, and see how the choice of h impacts security in either case.) Accordingly, it is convenient to first introduce auxiliary functions EABC and DABC. For $M, C \in D_{d,n}$ and $eK \in \{0,1\}^k$, we define

$\mathrm{EABC}(eK, P[0], C[0], M)$	$\mathrm{DABC}(eK, P[0], C[0], C)$
Parse M as $M[1] \ldots M[l]$ with $l \geq 1$	Parse C as $C[1] \ldots C[l]$ with $l \geq 1$
For $i = 1, \ldots, l$ do	For $i = 1, \ldots, l$ do
$\quad P[i] \leftarrow M[i] \oplus h(P[i-1])$	$\quad P[i] \leftarrow E^{-1}(eK, C[i] \oplus P[i-1])$
$\quad C[i] \leftarrow E(eK, P[i] \oplus C[i-1])$	$\qquad \oplus C[i-1]$
$\qquad \oplus P[i-1]$	$\quad M[i] \leftarrow P[i] \oplus h(P[i-1])$
EndFor	EndFor
Return $C[1] \ldots C[l]$	Return $M[1] \ldots M[l]$

We now define two versions of the ABC cipher. The first uses public initial values, while the second uses secret initial values. The *ABC cipher with public initial values* associated to E, denoted PABC, has key space $\{0,1\}^k$ and domain and range $D_{d,n}$. We fix values $P[0], C[0] \in \{0,1\}^n$ which are known to all parties including the adversary. We then define the cipher and the inverse cipher as follows:

$\mathsf{PABC}(eK, M)$	$\mathsf{PABC}^{-1}(eK, C)$
Return $\mathrm{EABC}(eK, P[0], C[0], M)$	Return $\mathrm{DABC}(eK, P[0], C[0], C)$

The *ABC cipher with secret initial values* associated to E, denoted SABC, has key space $\{0,1\}^{k+2n}$ and domain and range $D_{d,n}$. The key is $eK \| P[0] \| C[0]$. We then define the cipher and the inverse cipher as follows:

Distinguisher A^g
Let $M[2], \ldots, M[l]$ be any n-bit strings
Let $M_1 = 0^n M[2] \ldots M[l]$ and let $M_2 = 1^n M[2] \ldots M[l]$
Let $C_1[1] \ldots C_1[l] \leftarrow g(M_1)$ and let $C_2[1] \ldots C_2[l] \leftarrow g(M_2)$
Let $M_3[2] = M[2] \oplus C_1[1] \oplus C_2[1] \oplus h(0^n \oplus h(P[0])) \oplus h(1^n \oplus h(P[0]))$
Let $M_3 = 1^n M_3[2] M[3] \ldots M[l]$
Let $C_3[1] \ldots C_3[l] \leftarrow g(M_3)$
If $C_3[2] = C_1[2] \oplus 1^n$, then return 1 else return 0

Fig. 2. Attack on the ABC based on-line cipher.

$\mathsf{SABC}(eK \| P[0] \| C[0], M)$	$\mathsf{SABC}^{-1}(eK \| P[0] \| C[0], C)$
Return $\mathrm{EABC}(eK, P[0], C[0], M)$	Return $\mathrm{DABC}(eK, P[0], C[0], C)$

It is easy to check that both the above ciphers are n-on-line.

We show that the ABC cipher with public initial values is not a secure OPRP for *all* choices of the function h. The attack is shown in Figure 2. The adversary A gets an oracle g where g is either an instance of PABC or an instance of $\mathsf{OPerm}_{d,n}$. The adversary can mount this attack because the function h as well as the value $P[0]$ are public. We claim that

$$\mathbf{Adv}^{\text{oprp-cpa}}_{\mathsf{PABC}}(A) \geq 1 - 2 \cdot 2^{-n} \; . \tag{2}$$

Since A made only three oracle queries, this means that PABC is not a secure on-line cipher.

We show that the ABC cipher with secret initial values is not a secure OPRP for a class of functions h that includes the ones suggested in [7]. Specifically, let us say that a function h: $\{0,1\}^n \to \{0,1\}^n$ is *linear* if $h(x \oplus y) = h(x) \oplus h(y)$ for all $x, y \in \{0,1\}^n$. (Notice that the identity function, the constant function always returning 0^n, and the function which rotates its input by one bit are all linear.) For any linear hash function h, we simply note that the above attack applies. This is because the fourth line of the adversary's code can be replaced by

Let $M_3[2] = M[2] \oplus C_1[1] \oplus C_2[1] \oplus h(0^n) \oplus h(1^n)$

The adversary can compute $M_3[2]$ because h is public. The fact that h is linear means that the value $M_3[2]$ is the same as before, so the attack has the same success probability. The analysis for the attacks against both PABC and SABC appear in the full version of this paper [1].

5 Lemmas about AXU Families

Our constructions of on-line ciphers will use the families of AXU (Almost Xor Universal) functions as defined by Krawczyk [8]. We recall the definition, and then prove some lemmas that will be helpful in our analyses.

Definition 3. Let $n, hk \geq 1$ be integers, and let $H\colon \{0,1\}^{hk} \times \{0,1\}^n \to \{0,1\}^n$ be a family of functions. Let

$$\mathbf{Adv}^{\mathrm{axu}}_H = \max_{x_1,x_2,y} \left\{ \Pr\left[K \xleftarrow{R} \{0,1\}^{hk} \ : \ H(K,x_1) \oplus H(K,x_2) = y \right] \right\}$$

where the maximum is over all *distinct* $x_1, x_2 \in \{0,1\}^n$ and all $y \in \{0,1\}^n$. ∎

The "advantage function" based notation we are introducing is novel: previous works used instead the term "ϵ-AXU" family to refer to a family H that, in our notation, has $\mathbf{Adv}^{\mathrm{axu}}_H \leq \epsilon$. We find the "advantage function" based notation more convenient, and more consistent with the rest of our security definitions.

The definition is information-theoretic, talking of the maximum value of some probability. We will find it convenient to think in terms of an adversary attacking the scheme, and will use the following lemma. We stress that below there are no limits on the running time of the adversary. This lemma is standard, and follows easily from Definition 3, so we omit the proof.

Lemma 1. *Let $n, hk \geq 1$ be integers, and let $H\colon \{0,1\}^{hk} \times \{0,1\}^n \to \{0,1\}^n$ be a family of functions. Let A be any possibly probabilistic algorithm that takes no inputs and returns a triple (x_1, x_2, y) of n-bit strings. Then*

$$\Pr\left[(x_1,x_2,y) \xleftarrow{R} A\,;\ K \xleftarrow{R} \{0,1\}^{hk} \ : \ H(K,x_1) \oplus H(K,x_2) = y \right] \ \leq \ \mathbf{Adv}^{\mathrm{axu}}_H.$$

In the formulation of Lemma 1, it is important that the adversary is constrained to pick x_1, x_2, y before the K is chosen. In our upcoming analyses, we will, in contrast, be considering an adversary that obtains some partial information regarding $H(K,\cdot)$ in the course of its search for a certain kind of "collision," and uses this to guide its search. Specifically, our adversary B can be viewed as having access to an oracle that knows a key K. The adversary functions in stages. In stage i, it produces a pair (x_i, y_i) of values which it submits to the oracle. The latter responds with a bit indicating whether or not there exists some $j \in \{1, \ldots, i-1\}$ such that $H(K,x_j) \oplus H(K,x_i) = y_j \oplus y_i$. (The oracle is stateful because it has to remember the adversary queries from previous stages in order to be able to answer the current query.) We wish to argue that the partial information about $H(K,\cdot)$ that is obtained by the adversary via this process is not too large. Specifically, we argue that the probability that the adversary ever gets back a positive response from the oracle is $O(q^2) \cdot \mathbf{Adv}^{\mathrm{axu}}_H$.

In the formal definition that follows, we first describe an algorithm that serves as a stateful oracle discussed above. Then, we describe an experiment in which the adversary B with oracle access to the algorithm is executed.

Definition 4. Let $H\colon \{0,1\}^{hk} \times \{0,1\}^n \to \{0,1\}^n$ be a family of hash functions, and let hK be a string of length hk. We define the following stateful algorithm D. It maintains a counter i and arrays X, Y, and takes n-bit strings x, y as inputs. Then, we let B be an adversary with oracle access to D_{hK} and define an experiment in which B executes.

Algorithm $D_{hK}(x,y)$
$i \leftarrow i+1\,;\ r \leftarrow 0\,;\ X[i] \leftarrow x\,;\ Y[i] \leftarrow y$

For $j = 1, \ldots, i-1$ do
If $(H(hK, X[j]) \oplus Y[j] = H(hK, X[i]) \oplus Y[i])$ and $(X[j] \neq X[i])$ then $r \leftarrow j$
EndFor
Return r

Experiment $\mathbf{Exp}_H^{\text{axu-cr}}(B)$
$hK \xleftarrow{R} \{0,1\}^{hk}$
Initialize D_{hK} with $i = 0$ and X, Y empty
Run $B^{D_{hk}(\cdot,\cdot)}$ until it halts
If B made some oracle query that received a non-zero response,
then return 1, else return 0.

We define the *advantage* of the adversary B and the *AXU-Collision advantage function* of H as follows. For any integer q,

$$\mathbf{Adv}_H^{\text{axu-cr}}(B) = \Pr[\,\mathbf{Exp}_H^{\text{axu-cr}}(B) = 1\,]$$
$$\mathbf{Adv}_H^{\text{axu-cr}}(q) = \max_B \{\, \mathbf{Adv}_H^{\text{axu-cr}}(B) \,\}$$

where the maximum is taken over all adversaries making q queries. ∎

The following lemma states the relationship between Definition 3 and Definition 4. The proof is presented in the full version of this paper [1].

Lemma 2. *Let* $H\colon \{0,1\}^{hk} \times \{0,1\}^n \to \{0,1\}^n$ *be a family of hash functions. Then,*

$$\mathbf{Adv}_H^{\text{axu-cr}}(q) \;\leq\; q(q-1) \cdot \mathbf{Adv}_H^{\text{axu}} \;.$$

6 The HCBC Cipher

In this section, we suggest a construction of an on-line cipher. We call it HCBC and prove its security against chosen-plaintext attacks. This construction is similar to the CBC mode of encryption. The only difference is that each output block passes through a keyed hash function before getting exclusive-or-ed with the next input block. The key of the hash function is kept secret.

Construction 1. Let $n, d \geq 1$ be integers, and let $E\colon \{0,1\}^{ek} \times \{0,1\}^n \to \{0,1\}^n$ be a block cipher. Let $H\colon \{0,1\}^{hk} \times \{0,1\}^n \to \{0,1\}^n$ be a family of hash functions. We associate to them a cipher $\mathsf{HCBC}\colon \{0,1\}^{ek+hk} \times D_{d,n} \to D_{d,n}$. A key for it is a pair $eK \| hK$ where eK is a key for E and hK is a key for H. The cipher and its inverse are defined as follows for $M, C \in D_{d,n}$. Figure 3 illustrates the cipher.

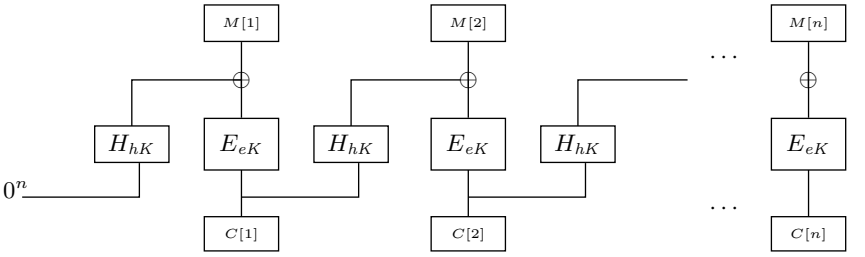

Fig. 3. The HCBC cipher.

$\mathsf{HCBC}(eK\|hK, M)$	$\mathsf{HCBC}^{-1}(eK\|hK, C)$
Parse M as $M[1]\dots M[l]$ with $l \geq 1$	Parse C as $C[1]\dots C[l]$ with $l \geq 1$
$C[0] \leftarrow 0^n$	$C[0] \leftarrow 0^n$
For $i = 1, \dots, l$ do	For $i = 1, \dots, l$ do
$\quad P[i] \leftarrow H(hK, C[i-1]) \oplus M[i]$	$\quad P[i] \leftarrow E^{-1}(eK, C[i])$
$\quad C[i] \leftarrow E(eK, P[i])$	$\quad M[i] \leftarrow H(hK, C[i-1]) \oplus P[i]$
EndFor	EndFor
Return $C[1]\dots C[l]$	Return $M[1]\dots M[l]$ ▮

The following theorem implies that, if E is a PRP secure against chosen-plaintext attacks and H is an AXU family of hash functions, then HCBC is an OPRP secure against chosen-plaintext attacks.

Theorem 1. *Let* $E\colon \{0,1\}^{ek} \times \{0,1\}^n \to \{0,1\}^n$ *be a block cipher, and let* $H\colon \{0,1\}^{hk} \times \{0,1\}^n \to \{0,1\}^n$ *be a family of hash functions. Let* HCBC *be the* n*-on-line cipher associated to them as per Construction 1. Then, for any integers* $t, q_e, \mu_e \geq 0$ *such that* $\mu_e/n \leq 2^{n-1}$*, we have*

$$\mathbf{Adv}^{\text{oprp-cpa}}_{\mathsf{HCBC}}(t, q_e, \mu_e) \;\leq$$

$$\mathbf{Adv}^{\text{prp-cpa}}_{E}(t, \mu_e/n, \mu_e) + \left(\frac{\mu_e^2 - n\mu_e}{n^2}\right) \cdot \mathbf{Adv}^{\text{axu}}_{H} + \frac{\mu_e^2 + 2n(q_e+1)\mu_e}{n^2 \cdot 2^n} \,.$$

HCBC is not secure against chosen-ciphertext attacks. We present an attack in the full version of this paper [1].

A complete proof of Theorem 1 can be found in the full version of this paper [1]. In the rest of this section, we provide an overview of this proof.

We introduce the notation $\mathsf{HCBC}_\pi(hK, \cdot)$ to denote an instance of a cipher defined by Construction 1 where a permutation π and π^{-1} are used in place of a permutation from the family E and its inverse, respectively. The proof looks at an on-line cipher as a 2^n-ary tree of permutations on $\{0,1\}^n$, and goes through a hybrid argument involving a sequence of different games that "move" from $\mathsf{OPerm}_{d,n}$ to HCBC. Let A be an adversary that has oracle access to a length-preserving function $f\colon D_{d,n} \to D_{d,n}$. We assume that A makes at most q_e oracle

queries the sum of whose lengths is at most μ_e bits. We define three games associated with the adversary A as follows.

Game 1. Choose a tree of random permutations $T \xleftarrow{R} \mathsf{Tree}(n, \mathsf{Perm}_n, d)$. Run A, replying to its oracle queries via T as described in Section 3. Let P_1 be the probability that A returns 1.

Game 2. Choose a random permutation, $\pi \xleftarrow{R} \mathsf{Perm}_n$, and choose a random key for H via $hK \xleftarrow{R} \{0,1\}^{hk}$. Run A, replying to its oracle queries via $\mathsf{HCBC}_\pi(hK, \cdot)$. Let P_2 be the probability that A returns 1.

Game 3. Choose random keys for E and H via $eK \xleftarrow{R} \{0,1\}^{ek}$ and $hK \xleftarrow{R} \{0,1\}^{hk}$, respectively. Run A, replying to its oracle queries via $\mathsf{HCBC}(eK \| hK, \cdot)$. Let P_3 be the probability that A returns 1.

By the definition of $\mathbf{Adv}^{\text{oprp-cpa}}_{\mathsf{HCBC}}(A)$, we have

$$\mathbf{Adv}^{\text{oprp-cpa}}_{\mathsf{HCBC}}(A) = P_3 - P_1 = (P_3 - P_2) + (P_2 - P_1)\ . \tag{3}$$

We bound the difference terms via the following lemmas:

Lemma 3. $P_3 - P_2 \ \leq\ \mathbf{Adv}^{\text{prp-cpa}}_{E}(t, \mu_e/n, \mu_e)$

Lemma 4. $P_2 - P_1 \ \leq\ \dfrac{\mu_e^2 + 2n(q_e+1)\mu_e}{n^2 \cdot 2^n} + \mathbf{Adv}^{\text{axu-cr}}_{H}(\mu_e/n)$

Equation (3), Lemma 2, and the above lemmas imply the statement of the theorem. We proceed to discuss the proofs of the lemmas.

The proof of Lemma 3 is a standard simulation argument, detailed in [1]. The rest of this section is devoted to an overview of the proof of Lemma 4. We let $M_1, \ldots, M_{q_e}$ denote A's queries, where $M_j = M_j[1] \ldots M_j[l_j]$ for $j = 1, \ldots, q_e$. Let hK denote the key of the hash function, and π the choice of permutation from Perm_n, that underly Game 2. Then we introduce the following notation in this game:

For each $j = 1, \ldots, q_e$
  Let $C_j[0] = 0^n$
  For $i = 1, \ldots, l_j$
    Let $P_j[i] = H(hK, C_j[i-1]) \oplus M_j[i]$ and let $C_j[i] = \pi(P_j[i])$

We now define some events in Game 2:

Event ZO_2 : There exist (i, j) such that $1 \leq j \leq q_e$, $1 \leq i \leq l_j$ and $C_j[i] = 0^n$

Event HC : There exist $(i, j), (i', j')$ such that $1 \leq j < j' \leq q_e$, $1 \leq i \leq l_j$, $1 \leq i' \leq l_{j'}$ and $P_j[i] = P_{j'}[i']$, but $C_j[i-1] \neq C_{j'}[i'-1]$

Event B_2 : $\mathsf{ZO}_2 \vee \mathsf{HC}$.

Now let T denote the random choice of tree from $\mathsf{Tree}(n, \mathsf{Perm}_n, d)$ that underlies Game 1 and introduce the following notation in this game:

For each $j = 1, \ldots, q_e$
  Let $x_j[0] = \varepsilon$
  For $i = 1, \ldots, l_j$
    Let $C_j[i] = T_{x_j[i-1]}(M_j[i])$ and let $x_j[i] = x_j[i-1] \| C_j[i]$

We now define some events in Game 1:

Event ZO_1 : There exist (i, j) such that $1 \le j \le q_e$, $1 \le i \le l_j$ and $C_j[i] = 0^n$

Event OC : There exist $(i, j), (i', j')$ such that $1 \le j < j' \le q_e$, $1 \le i \le l_j$, $1 \le i' \le l_{j'}$ and $x_j[i-1] \ne x_{j'}[i'-1]$ but $C_j[i] = C_{j'}[i']$

Event $\mathsf{B}_1 : \mathsf{ZO}_1 \vee \mathsf{OC}$

Let $\Pr_1[\cdot]$ denote the probability function underlying Game 1, namely that created by the random choice $T \xleftarrow{R} \mathsf{Tree}(n, \mathsf{Perm}_n, d)$, and let $\Pr_2[\cdot]$ denote the probability function underlying Game 2, namely that created by the random choices of π and hK. Let F denote $\mathsf{HCBC}_\pi(hK, \cdot)$.

Claim. $\Pr_2[A^F = 1 \mid \overline{\mathsf{B}}_2] = \Pr_1[A^T = 1 \mid \overline{\mathsf{B}}_1]$

Given this claim, a conditioning argument can be used to show that

$$P_2 - P_1 \le \Pr_2[\mathsf{HC}] + \Pr_2[\mathsf{ZO}_2] + \Pr_1[\mathsf{B}_1] .$$

The terms are bounded via the following claims:

Claim. $\Pr_2[\mathsf{HC}] \le \mathbf{Adv}^{\text{axu-cr}}_H(\mu_e/n)$ ∎

Claim. $\Pr_2[\mathsf{ZO}_2] \le \dfrac{2\mu_e}{n \cdot 2^n}$ ∎

Claim. $\Pr_1[\mathsf{B}_1] \le \dfrac{\mu_e^2 + 2nq_e\mu_e}{n^2 \cdot 2^n}$ ∎

The proofs of the four claims above can be found in [1]. We conclude this sketch by providing some intuition regarding the choice of the "bad" events, beginning with the following definition.

Definition. Suppose $1 \le j, j' \le q$, $1 \le i \le l_j$ and $1 \le i' \le l_{j'}$. We say that $(i, j) \prec (i', j')$ if: either $j = j'$ and $i < i'$, or $j < j'$. We say that (i', j') is *trivial* if there exists some $j < j'$ such that $ni' \le |\mathsf{LCP}_n(M_j, M_{j'})|$.

We claim that the bad event B_2 has been chosen so that, in its absence, the following is true for every non-trivial (i', j'): If $(i, j) \prec (i', j')$ then $P_j[i] \ne P_{j'}[i']$. In other words, any two input points to the function π are unequal unless they are equal for the trivial reason that the corresponding message prefixes are equal. This means that in the absence of the bad event, ciphertext blocks whose value is not "forced" by message prefix conditions are random but distinct, being outputs of a random permutation. We have choosen event B_1 in Game 1 so that the output distribution here, conditioned on the absence of this event, is the same.

7 Usage of Online Ciphers

The use of an on-line ciphers can provide strong privacy and authenticity properties, even though the cipher itself is weak compared to a standard one, if the plaintext space has appropriate properties. This follows via the the encode-then-encipher paradigm of [4], under which we imagine an explicit encoding step applied to the raw data before enciphering. While [4] say that randomness and redundancy anywhere in the message suffices, we have to be more constrained: we *prepend* randomness and *append* redundancy.

Construction 2. Let n, d be integers, and let F: $Keys(F) \times D_{d,n} \to D_{d,n}$ be a cipher. We associate to them the following symmetric encryption scheme $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$:

Algorithm $\mathcal{K}$	Algorithm $\mathcal{E}(K, M)$	Algorithm $\mathcal{D}(K, C)$
$K \xleftarrow{R} Keys(F)$	$r \xleftarrow{R} \{0,1\}^n$	$x \leftarrow F^{-1}(K, C)$
Return K	$x \leftarrow r \Vert M \Vert 0^n$	If $\lvert x \rvert < 3n$ then return $\perp$
	$C \leftarrow F(K, x)$	Parse x as $r \Vert M \Vert \tau$ with $\lvert r \rvert = \lvert \tau \rvert = n$
	Return C	If $\tau = 0^n$ then return M
		Else return $\perp$ ▌

We want to show that this scheme provides privacy, when F is an n-on-line cipher secure against chosen-plaintext attacks, and authenticity, when F is an n-on-line cipher secure against chosen-ciphertext attacks. Definitions for these privacy and authenticity notions are standard (see for example [3]). Briefly, the symmetric encryption scheme achieves privacy and is called IND-CPA-secure if no polynomial time adversary, which gets to see ciphertexts for plaintexts of its choice and is given a challenge ciphertext, can get "any" information about the underlying plaintext. The symmetric encryption scheme achieves integrity and is called INT-CTXT-secure if no polynomial time adversary, which gets to see ciphertexts of plaintexts of its choice, can create a "new" valid ciphertext. The following claims state our results.

Proposition 4. *Let F: $Keys(F) \times D_{d,n} \to D_{d,n}$ be an n-on-line cipher, and let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be the symmetric encryption scheme defined in Construction 2. Then, for any integers $t, q_e, \mu_e \geq 0$,*

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}}(t, q_e, \mu_e) \leq 2\mathbf{Adv}^{\text{oprp-cpa}}_{F}(t, q_e, \mu_e) + \frac{q_e^2}{2^n} .$$

Also, for any integers $t, q_e, q_d, \mu_e, \mu_d \geq 0$,

$$\mathbf{Adv}^{\text{int-ctxt}}_{\mathcal{SE}}(t, q_e, q_d, \mu_e, \mu_d) \leq 2\mathbf{Adv}^{\text{oprp-cca}}_{F}(t, q_e, \mu_e, q_d, \mu_d) + \frac{q_d}{2^n} .$$

That is, if F is an n-on-line cipher secure against chosen-plaintext attacks, then $\mathcal{SE}$ is IND-CPA secure, and if F is also secure against chosen-ciphertext attacks, then $\mathcal{SE}$ is INT-CTXT secure.

The proof of Proposition 4 is simple and follows [4]. We present it in [1]. Note that if n-on-line ciphers are used to encrypt messages which by their nature start

with at least n random bits and end with some fixed sequence of n bits than we get a symmetric encryption scheme that achieves privacy and integrity and, moreover, is length-preserving.

Acknowledgments. The UCSD authors are supported in part by Bellare's 1996 Packard Foundation Fellowship in Science and Engineering. We thank Anand Desai, Bogdan Warinschi, and the Crypto 2001 program committee for their helpful comments.

References

1. M. Bellare, A. Boldyreva, L. Knudsen, C. Namprempre. On-line ciphers and the Hash-CBC construction. Full version of this paper, available via `http://www-cse.ucsd.edu/users/mihir`.
2. M. Bellare, J. Kilian, and P. Rogaway. The security of the cipher block chaining message authentication code. In *Journal of Computer and System Sciences*, volume 61, No. 3, pages 362-399, Dec 2000. Academic Press.
3. M. Bellare and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In T. Okamoto, editor, *Advances in Cryptology — ASIACRYPT ' 00*, volume 1976 of *Lecture Notes in Computer Science*, pages 531–545, Berlin, Germany, Dec. 2000. Springer-Verlag.
4. M. Bellare and P. Rogaway. Encode-then-encipher encryption: How to exploit nonces or redundancy in plaintexts for efficient cryptography. In T. Okamoto, editor, *Advances in Cryptology — ASIACRYPT ' 00*, volume 1976 of *Lecture Notes in Computer Science*, pages 317–330, Berlin, Germany, Dec. 2000. Springer-Verlag.
5. C. Campbell. Design and specification of cryptographic capabilities. In D. Brandstad, editor, *Computer Security and the Data Encryption Standard,* National Bureau of Standards Special Publications 500-27, U.S. Department of Commerce, pages 54-66, February 1978.
6. O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions, *Journal of the ACM,* Vol. 33, No. 4, 1986, pp. 210–217.
7. L. Knudsen. Block chaining modes of operation. Reports in Informatics, Report 207, Dept. of Informatics, University of Bergen, October 2000.
8. H. Krawczyk. LFSR-based hashing and authenticating. In Y. Desmedt, editor, *Advances in Cryptology — CRYPTO '94*, volume 839 of *Lecture Notes in Computer Science*, pages 129–139, Berlin, Germany, 1994. Springer-Verlag.
9. M. Luby and C. Rackoff. How to construct pseudo-random permutations from pseudo-random functions. *SIAM Journal of Computing,* Vol. 17, No. 2, pp. 373–386, April 1988.
10. C. Meyer and Matyas. *A new direction in Computer Data Security.* John Wiley & Sons, 1982.
11. M. Naor and O.Reingold. On the construction of pseudorandom permutations: Luby-Rackoff Revisited. In J. Feigenbaum, editor, *Journal of Cryptology*, Volume 12, Number 1, Winter 1999. Springer-Verlag.
12. W. Nevelsteen and B. Preneel. Software performance of universal hash functions. In J. Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 24–41, Berlin, Germany, 1999. Springer-Verlag.