PRF Domain Extension using DAGs

Charanjit S. Jutla

IBM T. J. Watson Research Center
Yorktown Heights, NY 10598

Abstract. We prove a general domain extension theorem for pseudo-random functions (PRFs). Given a PRF F from n bits to n bits, it is well known that employing F in a chaining mode (CBC-MAC) yields a PRF on a bigger domain of mn bits. One can view each application of F in this chaining mode to be a node in a graph, and the chaining as edges between the node. The resulting graph is just a line graph. In this paper, we show that the underlying graph can be an arbitrary directed acyclic graph (DAG), and the resulting function on the larger domain is still a PRF. The only requirement on the graph is that it have unique source and sink nodes, and no two nodes have the same set of incident nodes. A new highly parallelizable MAC construction follows which has a critical path of only $3 + \log^* m$ applications of F.

If we allow Galois field arithmetic, we can consider edge-colored DAGs, where the colors represent multiplication in the field by the color. We prove an even more general theorem, where the only restriction on the colored DAGs is that if two nodes (u and v) have the same set of incident nodes W, then at least one w in W is incident on u and v with a different colored edge. PMAC (Parallelizable Message Authentication [6]) is a simple example of such graphs. Finally, to handle variable length domain extension, we extend our theorem to a collection of DAGs. The general theorem allows one to have further optimizations over PMAC, and many modes which deal with variable lengths.

Keywords: PRF, MAC, DAG, Partial Order, Galois Field

1 Introduction

There is often a need to extend the domain of a given pseudo-random function (PRF). One of the most popular and well-known such schemes is the CBC-MAC [1]. In [3] it was shown that if F is a pseudo-random function from n bits to n bits, then the CBC (cipher block chaining) construction yields a PRF from mn bits to n bits. Although the construction is called a MAC (message authentication code), which is a strictly weaker notion than PRF [9], the above shows that it is indeed a PRF domain extension method. Other domain extension schemes are known as well, for example, the cascade construction [2] and the protected counter sum construction [5]. Recently, a scheme PMAC (or Parallelizable Message Authentication) [6] (also see XECB [11]) was also shown to be a domain extension scheme.

Despite all these results, there is no unifying theme in these results. In this paper, we attempt to remedy this situation, by proving a general theorem for domain extension. In essence, we show that arbitrary acyclic networks of the same pseudo-random function can be used to build a pseudo-random function on a larger domain. To illustrate this paradigm, consider the CBC-MAC scheme. Let F be a PRF from n bits to n bits (and which takes k bits of secret key). For example, DES [10] is usually assumed to be such a PRF on 64 bits, with 56 bits of secret key. A PRF $\tilde{F}$ from mn bits to n bits is defined as follows. The mn bit input is divided into m blocks $P_1, P_2, ..., P_m$. The function F_k (i.e. F with key k) is applied to the first block P_1 to yield an intermediate value C_1. The function F_k is next invoked on the xor of the next block P_2 and previous intermediate value C_1, to yield C_2. This chaining process is continued, and the output of $\tilde{F}_k$ is just C_m. The chaining process defines an underlying directed graph of m nodes $V_1, V_2, ..., V_m$, with an edge from V_i to V_{i+1}.

Now, consider an arbitrary directed acyclic graph (DAG) $G = (V, E)$, with m nodes V, and edges E. Assume that G has only one source node V_1, and only one sink node V_m. Given a PRF F from n bits to n bits, a composite PRF $\tilde{F}$ from mn bits to n bits is defined as follows. As before, assume that the input is a sequence $P_1, ..., P_m$. The first intermediate value is just $C_1 = F_k(P_1)$. Inductively assume that we have computed the intermediate values of all predecessors of a node V_i. Then, the intermediate value C_i for the node V_i is

$$C_i = F_k(P_i \oplus \bigoplus_{(i,j) \in E} C_j)$$

The output of the composite function $\tilde{F}_k$ is just C_m. See Figure 1 for an example.

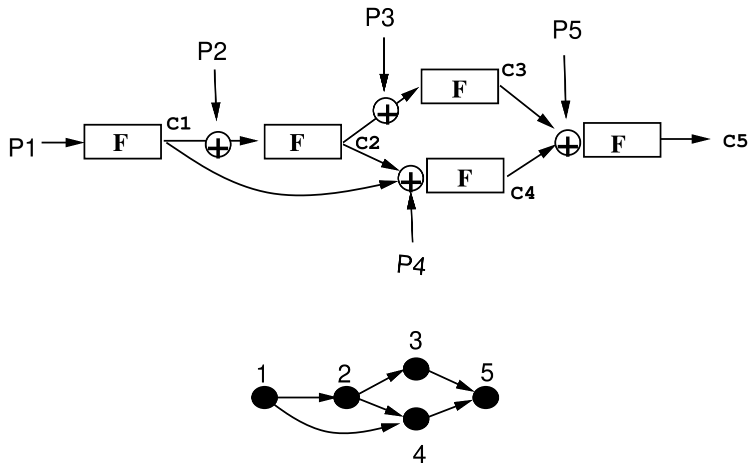

Fig. 1. A PRF Domain Extension Mode and its DAG

Of course, not all DAGs are expected to yield a PRF. However, consider DAGs with the restriction that no two nodes have the same set of incident nodes (u is said to be incident on v if there is an edge from u to v), and that they have unique source and sink nodes. In this paper we show that given a PRF F from n bits to n bits, the composite $\tilde{F}$ defined as above on such DAGs, is a PRF from mn bits to n bits.

An immediate application is that if a party has access to parallel hardware, then instead of simple chaining as in CBC-MAC, it can compute the PRF in parallel. For instance, if it has four processors, then it can employ the method given by the graph in Figure 2. A parallel mode with critical path of length only $3 + \log^* m$ also follows. Unlike PMAC [6], this mode does not use any Galois arithmetic.

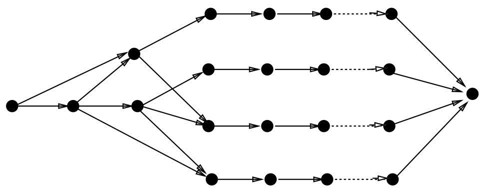

Fig. 2. A Parallel Mode for four processors

If we allow Galois Field arithmetic (in particular, fields $\mathrm{GF}(2^n)$), we can consider edge-colored DAGs. The colors on the edges represent multiplication in the field by the color (assume that each color is mapped to a unique element in the field). For example, going back to figure 1, suppose we employ three colors, col1, col2 , and col3. Let w be a primitive element in the field. We map col1 to unity in the field, col2 to w, and col3 to w^2. Then, if we color the edge $(1, 4)$ by col2, then in the definition of the composite function, we multiply the intermediate result C_1 with w in the field, before xoring it with the plaintext P_4 and C_2, and applying F_k.

The main result of the paper can be stated as follows. Consider an edge colored DAG G with unique source and sink nodes and m total nodes, and with the condition that if two nodes (say u and v) have the same set of incident nodes (say W), then for at least one node w in W, the color on the edge (w, u) is different from the color on the edge (w, v). Given a PRF F from n bits to n bits, the composite $\tilde{F}$ built using the graph G as above, is a PRF from mn bits to n bits. The result is proven under the adaptive adversary model, which is of course the difficult case. Our proof technique is novel, and even when considered as just a proof for CBC-MAC it offers a simpler and novel proof in the adaptive adversary model. It is well known that the difficulty in analyzing the security of such schemes stems from the fact that we need to model the underlying oracle as a function, i.e. an oracle replying consistently with earlier queries. The key advance is an identity (lemma 4) which reduces the analysis to a scheme where the oracle replies randomly. The adversary remains adaptive, but the analysis in this "random game" becomes much easier.

Using the new theorem, the mode in fig 2 can now be parallelized further as in fig 3(a). The additional cost is a few $\mathrm{GF}(2^n)$ operations. Security of PMAC [6] follows (see fig 3(b)), as it is a simple example of such a colored DAG. Further, we obtain an additional optimization over PMAC, because unlike PMAC, we do not even need to compute F_k on the all zero word (i.e. $F_k(0^n)$).

In Section 5 we extend our results to *variable length domain extension.*

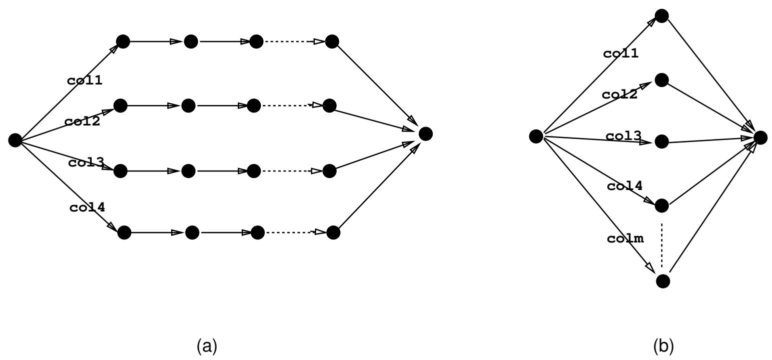

Fig. 3. Modes using $\mathrm{GF}(2^n)$

2 Definitions

Definition 1. For positive integers n, l, let $\mathcal{F}(n \rightarrow l)$ be the set of all functions from n bits to l bits.

Definition 2. (PRF) A *pseudo-random function* has signature

$$F: \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^l.$$

Define $\mathrm{Sec}_F(q,T)$ to be the maximum advantage an adaptive adversary can obtain when trying to distinguish between $F_K(\cdot)$ (with K chosen uniformly at random) and a function chosen uniformly at random from $\mathcal{F}(n \rightarrow l)$, when given q queries and time T.

3 Domain Extension using arbitrary acyclic graphs

Definition 3. Let $G = (V, E)$, be a directed acyclic graph (DAG) [13] with a finite vertex set V and edges E. A node u is said to be **incident** on a node v, if there is an edge from u to v, i.e $E(u, v)$. Such an edge will sometimes be denoted $\langle u, v\rangle$. Define a DAG to be **non-redundant** if for every pair of nodes, the set of their incident nodes is different. For two vertices u and v, we say that $u \prec v$ if there is a directed path from u to v. Since G is a finite DAG, the relation $\prec$ is a finite partial order.

Definition 4. Given a function f from n bits to n bits, and a non-redundant DAG $G = (V, E)$ with only one source node and only one sink node, and a total of m nodes, define $f^G : \{0,1\}^{nm} \to \{0,1\}^n$ as follows:

- Let the input to f^G be an mn bit string P, which is divided into m n-bit strings $P_1, P_2, ..., P_m$.
- Since $|V| = m$, let V_1,.....,V_m be an enumeration of the nodes. When it is clear from context, we will identify the index of a vertex with the vertex itself. Let the unique source node be V_1, and the unique sink node be V_m.
- For the unique source node, define $M_1 = P_1$.
- For every non-source node V_j, $j > 1$, inductively (over $\prec$) define $M_j = P_j \oplus_{u:E(u,j)} f(M_u)$
- For notational convenience, for every node V_j, let C_j denote $f(M_j)$.
- The output of the function f^G is just C_m.

It is clear that the restriction of one sink node is crucial, for if there was another sink node other than V_m, then the plaintext fed into this other sink node has no influence on C_m. It is possible that there are instances of DAGs G with *two source nodes* such that F^G is a PRF; however, a more stringent requirement than non-redundancy will definitely be required. Consider a DAG G, with two source nodes V_1 and V_2, both with only one outgoing edge and that too to the same vertex. Then, the resulting function is clearly not a PRF. A similar situation motivates the requirement of non-redundancy.

One may be tempted to weaken the non-redundancy requirement. For instance, one idea is to have a condition on the DAG that it have no non-trivial automorphism. However, such a DAG may not yield a secure PRF, as illustrated in Figure 4. The two queries $\langle p1, p2, p2, p4, p5, p6\rangle$ and $\langle p1, p2, p2, p5, p4, p6\rangle$ yield the same result.

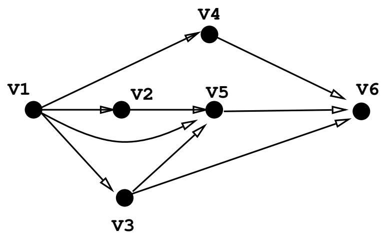

Fig. 4. A non-automorphic DAG

Theorem 1. *For a non-redundant DAG $G = (V, E)$ with unique source and sink nodes, and m total nodes, let f^G be as above. Then, no adaptive adversary, with q queries, can distinguish between (a) f^G where f is chosen uniformly at random from $\mathcal{F}(n \to n)$, (b) and a function chosen uniformly at random from $\mathcal{F}(nm \to n)$, with probability more than $(mq)^2 2^{-(n+1)}$.*

In the next section, we state and prove a more general theorem.

4 Domain Extension using colored DAGs and GF(2^n)

If we allow Galois field arithmetic, we get an even more general construction, and a corresponding PRF domain extension theorem. Assuming that the underlying function F has an n-bit output, we will use the Galois field GF(2^n). Such fields have the property that they have exactly 2^n elements. Moreover, each element can be represented as a n bit vector, with addition in the field being just the bitwise exclusive-or ($\oplus$). Since multiplication distributes over addition in a field, it follows that if a, b and c are three elements in the field then $a * (b \oplus c) = a * (b + c) = (a * b) + (a * c) = (a * b) \oplus (a * c)$. A further useful property of finite fields is that for a fixed non-zero a in the field, if b is picked uniformly at random from the field, then $a * b$ is also uniformly distributed in the field.

Definition 5. Let $G = (V, E)$, be a directed acyclic graph (DAG). Let $|V| = m$. A coloring χ of the edges of the graph is a map $\chi : E \to [1..m]$. The triple (V, E, χ) will be called an **edge-colored** DAG. Define an edge-colored DAG to be **non-singular** if for every pair of nodes u, v, if the set of their incident nodes is same (say W), then at least for one $w \in W$, $\chi(\langle w, u\rangle) \neq \chi(\langle w, v\rangle)$. For two vertices u and v, we say that $u \prec v$ if there is a directed path from u to v. Since G is a finite DAG, the relation $\prec$ is a finite partial order.

Definition 6. Given a function f from n bits to n bits, and a non-singular edge-colored DAG $G = (V, E, \chi)$ with only one source node and only one sink node and a total of $m < 2^n$ nodes, define $f^G : \{0, 1\}^{nm} \to \{0, 1\}^n$ as follows:

- Since $m < 2^n$, we can view χ as a map from E to GF$(2^n)^*$, i.e. the non- zero elements of the field.
- Let the input to f^G be mn bit string P, which is divided into m n-bit strings $P_1, P_2, ..., P_m$.
- Since $|V| = m$, let V_1,.....,V_m be an enumeration of the nodes. When it is clear from context, we will identify the index of a vertex with the vertex itself. Let the unique source node be V_1, and the unique sink node be V_m.
- For the unique source node, define $M_1 = P_1$.
- For every non-source node V_j, $j > 1$, inductively (over $\prec$) define $M_j = P_j + \sum_{u:E(u,j)} \chi(\langle u, j\rangle) * f(M_u)$, where $f(M_u)$, which is an n-bit quantity, is viewed as an element of GF(2^n). The summation is addition in the field, which is the same as n-bit exclusive-or.
- For notational convenience, for every j, we denote $f(M_j)$ by C_j.

– The output of the function f^G is just C_m.

Theorem 2. *: (Main Theorem) For a non-singular edge-colored DAG $G = (V, E, \chi)$ with unique source and sink nodes, and $m < 2^n$ total nodes, let f^G be as above. Then, no adaptive adversary, with q queries, can distinguish between (a) f^G where f is chosen uniformly at random from $\mathcal{F}(n \to n)$, (b) and a function chosen uniformly at random from $\mathcal{F}(nm \to n)$, with probability more than $(mq)^2 2^{-(n+1)}$.*

Theorem 3. *Given a PRF $F : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, and a non-singular edge-colored DAG $G = (V, E, \chi)$ with unique source and sink nodes, and $m < 2^n$ total nodes, a function $F^G : \{0,1\}^k \times \{0,1\}^{mn} \to \{0,1\}^n$ can be defined be letting for each K, $(F^G)_K$ to be $(F_K)^G$ (as in definition 6). Then,*

$$Sec_{F^G}(q,T) \leq Sec_F(q,T) + (mq)^2 2^{-(n+1)}$$

The proof follows from Theorem 2 by standard techniques.

4.1 Background

Most theorems in cryptography involving PRF [3, 2, 5, 6, 11] and PRP [21, 22, 15, 12] constructions, as well as modes of operations of block ciphers [4, 16, 11, 24, 19][1], from other primitives must tackle the issue of *collisions* in the oracle calls to the smaller primitive. Fortunately, these collision probabilities are usually low, and conditioning on distinctness of oracle calls, the target construct can be shown to behave like a random function or permutation.

Upper bounding the collision probability requires different techniques in many of these theorems, and the difficulty in the proof can depend on issues like whether there are two independent oracles (as in [22, 16]) or if a fresh initial vector is used in each invocation (as in [4, 16, 24, 19]). In particular, the proof of security of CBC-MAC [3] is more involved than that of CBC [4] for precisely this reason, i.e. in the former there is no fresh initial vector in each invocation. As our theorem generalizes CBC-MAC, we expect similar intricacies in our proof. However, as mentioned in the introduction, we prove a novel technical lemma which precisely captures this nuance of CBC-MAC.

4.2 Notation

Before we prove theorem 2, we need to fix more notation and give a general idea of the proof. We first note that we allow arbitrary functions as adversaries and not just computable functions. Then without loss of generality, we can assume that the adversary is deterministic, as every probabilistic adversary is just a probability distribution over all deterministic adversaries [18].

Fix an adaptive adversary. Since the adversary is deterministic, the first query's plaintext (say $P^1 = \langle P_1^1, ..., P_m^1 \rangle$) is fixed for that adversary. Thus, the

[1] These references are not meant to be exhaustive.

first query's output, say C_m^1 is only a function of f. The adversary being adaptive, its second query is a function of C_m^1. But, since C_m^1 is only a function of f, the second query's plaintext can also be written just as a function of f. Thus, C_m^2 is only a function of f, and so forth.

We will denote probabilities under the first scenario, i.e. (a) in the theorem 2 statement, as Pr, and the probabilities in the second scenario, i.e. (b) in the theorem 2 statement, as $\Pr_{(b)}$. Most of the analysis will be devoted to the first scenario. So, unless otherwise mentioned, all random variables from now on are in the first scenario.

For all variables corresponding to a query, we will use superscripts to denote the query number. Subscripts will be used to denote blocks within a query. The variables will be as in Definition 6, i.e. P standing for plaintext input, M standing for the variable on which the f function is applied, and C standing for the output of f.

Thus, by the convention above C_j^i is the output of f in the i^{th} query's j^{th} block. We will use C to denote the whole **transcript** $\{C_j^i\}_{i\in[q],j\in[m]}$ of f outputs. There will often be a need to just refer to the sequence of last blocks of each query; we will use C_m^* to denote the sequence $C_m^1,\ldots,C_m^q$, i.e. the mth block from all the queries. More precisely, as argued earlier, these variables should be written as a function of f, e.g. $C(f)$, but we will drop the argument when it is clear from context.

Let c denote a constant mqn-bit transcript, i.e. a prospective value for $C(f)$. For a fixed c, P_j^i and M_j^i *can be viewed as functions of only* c (see definition 6), and we will write them as $P_j^i(c)$ and $M_j^i(c)$. Just as for C, we will use $P(c)$ to denote the whole sequence.

Definition 7. Given a constant mqn-bit transcript c, let the plaintext chosen by the adversary be $p = P(c)$. For any vertices j, j', and query indices i, i' we say that $(i,j) \equiv_c (i',j')$ if

$$(j = j') \text{ and } \forall k \preceq j : p_k^i = p_k^{i'}$$

Define

$$\mu_c(i,j) = \min\{i' | (i',j) \equiv_c (i,j)\}.$$

Not every mqn-bit constant c can be a real transcript $C(f)$ for some f. So, we define a notion of consistent c. We call c **consistent** ($\text{con}(c)$) if

$$\forall j \in [1..m], \forall i : \; c_j^i = c_j^{\mu_c(i,j)}$$

Define the following "correcting" function ρ from mq n-bit blocks to mq n-bit blocks:

$$\rho(c) = \overline{c}, \text{ where } \overline{c}_j^i = c_j^{\mu_c(i,j)}.$$

The above definition of a correcting function is similar to the one used in the proof of the Luby-Rackoff theorem (see [20]).

Define the *core* index set of c to be $I = \{(i,j) | \mu_c(i,j) = i\}$. Informally, I is the set of indices which are not required to be "consistent" with smaller indices. Thus, $\rho(c)$ retains the values at core indices, and corrects them otherwise.

Consider the following condition **PD** (*pairwise different*).

Definition 8. For any constant c, define PD(c) to be

$$\forall i, i' \in [1..q], \forall j, j' \in [1..m],\ j \neq j'\ :\ M_j^i(c) \neq M_{j'}^{i'}(c),$$

$$\text{and}\quad \forall i, i' \in [1..q], \forall j \in [1..m]\ :\ (i,j) \not\equiv_c (i',j) \Rightarrow M_j^i(c) \neq M_j^{i'}(c).$$

4.3 Proof of Main Theorem

Lemma 4. (PRF Technical Lemma) *For every qn-bit constant $r = \langle r^i \rangle_{i \in [1..q]}$*

$$\begin{aligned} &Pr_{c \in_U \{0,1\}^{mqn}, f \in_U \mathcal{F}(n \to n)}[\, C(f) = c\ \wedge\ \ PD(c) \mid c_m^* = r] \\ &= 2^{-mqn} * Pr_{c \in_U \{0,1\}^{mqn}}[\, PD(\rho(c)) \mid c_m^* = r] \end{aligned}$$

In the left hand side of the lemma, we have that c is consistent, as it is not difficult to see that $C(f)$ is consistent (as proven below in lemma 5(i)). Now for consistent c, it is also easy to see that "correcting" it leaves it unchanged, i.e. $\rho(c) = c$ (see lemma 5(f)). Hence, we can replace PD(c) by PD$(\rho(c))$ in the left hand side above. Thus, the lemma can be restated as

$$\Pr_{c,f}[\, C(f) = c \mid\ \mathrm{PD}(\rho(c))\ \wedge\ c_m^* = r] = 2^{-mqn}$$

To prove this, we can try to see what constraints are imposed on f and c by $C(f) = c$. For $C(f)$ to be same as c, the transcript c must be consistent, as $C(f)$ is consistent. Let I be the *core* index set of c. Let $l = |I|$. Then for c to be consistent, there are exactly $(mq - l)$ n-bit linear constraints on c. We will also see that for every consistent c (in which case $c = \rho(c)$), such that $PD(c)$ holds, a function f_c can be defined using the M and the c values at core indices I such that $C(f_c) = c$ (see lemma 6). Moreover, such an f_c is *unique* on these l input values M (lemma 5(f)). Thus, there are exactly l n-bit constraints on f such that $C(f) = c$. Thus, there are a total of mq n-bit constraints on f and c. However, we have not addressed the issue of whether the condition PD$(\rho(c))$ perhaps influenced this count of mq total constraints. We show below rigorously that even under the condition PD$(\rho(c))$ the number of n-bit constraints on f and c is exactly mq.

So to start with, for each consistent c, we would like to define a function f_c such that $C(f_c) = c$. We also show below (lemma 5(h)) that for consistent c, $M_j^i(c) = M_j^{\mu_c(i,j)}(c)$. Thus, if we define f_c at core indices, i.e. define $f_c(M_j^i(c)) = c_j^i$ for all i in I, we might have $C(f_c) = c$. There is a slight problem however, i.e. f_c may not be well-defined, as the $M(c)$ values at core indices may not be distinct. In fact, we will need an even stronger distinctness condition, i.e. PD defined above, than just being distinct at core indices.

Definition 9. For each c, such that PD(c) holds, define f_c as follows. Let $I = \{(i,j) \mid \mu_c(i,j) = i\}$ be the *core* index set. For $(i,j) \in I$, define $f_c(M_j^i(c)) = c_j^i$. This is well defined as PD(c) holds. We will not need to define f_c on other values.

In Lemma 6 below we show that for every consistent c such that $\mathrm{PD}(c)$ holds it is indeed the case that $C(f_c) = c$.

We collect all simple statements about μ, ρ and consistency and their relationships to each other in the following lemma.

Lemma 5. *For all $i, i' \in [1..q]$, $i \neq i'$, for all $j \in [1..m]$ and mqn bit constant transcript c:*
(a) $(i,m) \not\equiv_c (i',m)$, i.e. $\mu_c(i,m) = i$,
(b) $\equiv_c$ is an equivalence relation,
(c) $\mu_c(\mu_c(i,j),j) = \mu_c(i,j)$,
(d) $\mu_c = \mu_{\rho(c)}$,
(e) $\rho(c)$ is consistent,
(f) Let c be consistent, and let b be such that for all i s.t. $\mu_c(i,j) = i$, $b^i_j = c^i_j$. Then $\rho(b) = c$. Also, for consistent c, $\rho(c) = c$
(g) For $u \preceq j$, $\mu_c(i,u) = \mu_c(\mu_c(i,j),u)$.
(h) For consistent c, $M^i_j(c) = M^{\mu_c(i,j)}_j(c)$
(i) $C(f)$ is consistent,
(j) For the transcript c let $p = P(c)$ be its corresponding plaintext. If for all u s.t. $E(u,j)$, $\mu_c(i,u) = \mu_c(i',u)$, and $p^i_j = p^{i'}_j$, then $\mu_c(i,j) = \mu_c(i',j)$.

Proof: (a) As we have assumed, wlog, that the adversary does not repeat queries, it follows that i and i' $(i \neq i')$ can never be equivalent over all vertices V. In particular, it is not the case that $(i,m) \equiv_c (i',m)$. To see this, note that we have assumed that the graph has only one sink node, i.e. V_m. It follows that for every node j, $j \preceq m$, hence the claim.
(b) & (c) straightforward.
(d) Note that the adversary's choice of $p = P(c)$ depends only on c^*_m. So we first show that for all i, $\rho(c)^i_m = c^i_m$. This follows as $\mu_c(i,m) = i$ by (a). Thus p remains same for $\rho(c)$.
(e) We just note that for all i, i', $(i,j) \equiv_c (i',j)$ implies $\mu_c(i,j) = \mu_c(i',j)$. Thus, by definition of ρ, we have $\rho(c)^i_j = \rho(c)^{i'}_j$.
(f) We first note that, since by (a), $\mu_c(i,m) = i$, we have $b^i_m = c^i_m$. Thus, as in proof of (d) above, $\mu_b = \mu_c$. Now, $\rho(b)^i_j = b^{\mu_b(i,j)}_j = b^{\mu_{c,j}(i)}_j = c^{\mu_{c,j}(i)}_j$, the last equality following from (c) and condition on b. For consistent c, this is same as c^i_j.
(g) For $u \preceq j$, $(i,j) \equiv_c (i',j)$ implies $(i,u) \equiv_c (i',u)$. Now, $(i,j) \equiv_c (\mu_c(i,j),j)$. Thus, $(i,u) \equiv_c (\mu_c(i,j),u)$.
(h) $M^i_j(c) = P^i_j(c) + \sum_{u:E(u,j)} \chi(\langle u,j\rangle) * c^i_u$. First note that $P^i_j(c) = P^{\mu_c(i,j)}_j(c)$. Also, for consistent c and $u \preceq j$, $c^i_u = c^{\mu_c(i,u)}_u = c^{\mu_c(\mu_c(i,j),u)}_u$ by (g). Again by consistency of c, the latter is same as $c^{\mu_c(i,j)}_u$. This shows that $M^i_j(c) = M^{\mu_c(i,j)}_j(c)$.
(i) by induction on the finite partial order $\prec$.
(j) We just need to show that $(i,j) \equiv_c (i',j)$. But $\mu_c(i,u) = \mu_c(i',u)$ implies $(i,u) \equiv_c (i',u)$. This along with $p^i_j = p^{i'}_j$ shows that p agrees in queries i and i' over all blocks $j' \preceq j$. $\square$

Lemma 6. *For any consistent c such that $PD(c)$ holds:*

$$C(f_c) = c$$

Proof: Follows by induction. See the appendix for a full prove. $\square$

Lemma 4 (PRF Technical lemma restated) For every qn-bit constant $r = \langle r^i \rangle_{i \in [1..q]}$

$$\begin{aligned} &\Pr_{c \in_U \{0,1\}^{mqn}, f \in_U \mathcal{F}(n \to n)}[C(f) = c \ \wedge \ \ \mathrm{PD}(c) \mid c^*_m = r] \\ &= 2^{-mqn} * \Pr_{c \in_U \{0,1\}^{mqn}}[\mathrm{PD}(\rho(c)) \mid c^*_m = r] \end{aligned}$$

Proof: We first show that the LHS above is same as

$$\Gamma = \Pr_{c, b \in_U \{0,1\}^{mqn}}[b^i_j = c^i_j|_{(i,j): \mu_c(i,j) = i} \ \wedge \ \mathrm{con}(c) \ \wedge \ \ \mathrm{PD}(c) \mid c^*_m = r]$$

By lemma 5(i), the conjunct $\mathrm{con}(c)$ can be added to the LHS of the lemma. We show that the two probabilities are same for every constant c. So, fix a c. As before, let $I = \{(i,j) \mid \mu_c(i,j) = i\}$ be the core index set of c. Let $S = \{M^i_j(c) \mid (i,j) \in I\}$. Since PD(c) holds, $|S| = |I|$. Let S' be an arbitrary set of n bit strings, disjoint from S, and $|S'| = mq - |I|$. Thus, $|S \cup S'| = mq$.

By lemma 6, $C(f_c) = c$. Thus, for each b agreeing with c on I, we have a function f_c defined on $|I|$ inputs S, such that $C(f_c) = c$. We can use the remaining $mq - |I|$ values of b (i.e. from indices which are not in I) to extend f_c to be defined on $S \cup S'$. This map from b to the extended f_c is 1-1.

Similarly, for any function f defined on $S \cup S'$, such that $C(f) = c$ (note that f need only be defined on S for $C(f)$ to be well defined), we can define an mqn-bit long b which agrees with c on I. For indices in $(i,j) \in I$, use $f(M^i_j(c))$ to define b^i_j, and use $f(s)$, $s \in S'$, to define the remaining part of b. This map from f to b is also 1-1. This shows that the LHS of the statement of the lemma is same as Γ.

We next show that, the RHS of the statement of the lemma is same as Γ. To this end, we show that the following two sets are equinumerous, i.e. we show a bijection between the two sets. The two sets are

$$\mathcal{C} = \{c | c \in \{0,1\}^{mqn}, \mathrm{PD}(\rho(c)), \ \text{and} \ \ c^*_m = r\}$$

$$\mathcal{D} = \{(c,b) | c, b \in \{0,1\}^{mqn}, b^i_j = c^i_j|_{(i,j) \in I}, \ \mathrm{con}(c), \ \ PD(c), \ \ \text{and} \ \ c^*_m = r\}$$

That they are equinumerous follows easily from lemma 5(e,f,a,d), but to be rigorous consider the following extension of ρ to a function $\hat{\rho}$ from $\mathcal{C}$ to $\mathcal{D}$.

$$\hat{\rho}(c) = (\rho(c), c)$$

It needs to be shown that the function has $\mathcal{D}$ as its range, is 1-1 and onto. The function is obviously 1-1. To prove that its range is $\mathcal{D}$, we need to prove three things:

1. $\rho(c)$ is consistent: follows by lemma 5(e).

2. $c^i_j = \rho(c)^i_j|_{\mu_{\rho(c)}(i,j)=i}$: follows directly from definition of ρ and lemma 5(d).
3. $\forall i$, $\rho(c)^i_m = r^i$: by lemma 5(a) and definition of ρ we have $\rho(c)^i_m = c^i_m$; and hence $c^i_m = r^i$ implies $\rho(c)^i_m = r^i$.

To prove that it is onto, for any (c, b) in $\mathcal{D}$, we show that b is in $\mathcal{C}$ and $\hat{\rho}(b) = (c, b)$. But for any (c, b) in $\mathcal{D}$, by lemma 5(f), $\rho(b) = c$. Thus, $\hat{\rho}(b) = (c, b)$. It also follows that $\mathrm{PD}(\rho(b))$ holds. Moreover, by lemma 5(a), $b^*_m = c^*_m$. Thus b is in $\mathcal{C}$.

The lemma follows by noting that $\Gamma = |\mathcal{D}|/2^{2mqn} = 2^{-mqn} * (|\mathcal{C}|/2^{mqn})$. □

Lower bounding the right hand side of the above lemma is a much easier task, as there is no function f involved.

We will denote by Δ the quantity $(mq)^2 2^{-(n+1)}$.

Lemma 7. *For every qn bit constant r,*

$$Pr_{c \in_{\mathcal{U}} \{0,1\}^{mqn}}[\ PD(\rho(c)) \mid c^*_m = r]\ \geq 1 - \Delta$$

Proof: First note that for all i, $c^i_m = \rho(c)^i_m$, by lemma 5(a) and definition of ρ. Thus, once c^*_m is fixed (and hence $\rho(c)^*_m$) to r, the plaintext $p = P(c)$ is fixed, independent of other c^i_j ($i \in [1..q]$, $j < m$). We will prove the lemma by upper bounding the probability of $\neg$PD by union bound.

For each vertex j, let V_j be its set of incident vertices, i.e. $V_j = \{u | E(u, j)\}$. Recall,

$$M^i_j(\rho(c)) = p^i_j + \sum_{u:E(u,j)} \chi(\langle u, j\rangle) * c_u^{\mu_c(i,u)}$$

If $j \neq j'$, and $V_j \neq V_{j'}$, wlog let $w \in V_j$ and $w \notin V_{j'}$. Then $M^i_j(\rho(c)) = M^{i'}_{j'}(\rho(c))$ iff

$$\begin{aligned}
&\chi(\langle w, j\rangle) * c_w^{\mu_c(i,w)} \\
&\quad = p^i_j + p^{i'}_{j'} + \sum_{u:E(u,j),u\neq w} \chi(\langle u, j\rangle) * c_u^{\mu_c(i,u)} + \sum_{u:E(u,j')} \chi(\langle u, j'\rangle) * c_u^{\mu_c(i',u)}
\end{aligned}$$

Since, $c_w^{\mu_c(i,w)}$ does not appear on the RHS, and $w < m$, and $\chi(\langle w, j\rangle) \neq 0$, the probability of above is 2^{-n}.

If $j \neq j'$, and $V_j = V_{j'}$, then for some $w \in V_j$, $\chi(\langle w, j\rangle) \neq \chi(\langle w, j'\rangle)$, as the underlying graph G is non-singular. Thus, similarly to the argument above, $M^i_j = M^{i'}_{j'}$ happens with probability 2^{-n}.

When j equals j' (and $i \neq i'$), we have three cases. If for some u incident on j ($E(u, j)$), $\mu_c(i, u) \neq \mu_c(i', u)$, then the probability of the two Ms being equal is at most 2^{-n}. Otherwise, if $p^i_j \neq p^{i'}_j$, then the probability is zero. If $p^i_j = p^{i'}_j$, we have $\mu_c(i, j) = \mu_c(i', j)$ by lemma 5(j), and hence the corresponding disjunct in $\neg$PD is false.

Since all the probabilities are 2^{-n} or zero, the bound in the lemma follows. □

Lemma 8.

$$Pr_f[\ PD(C(f))]\ \geq\ 1-\Delta$$

Proof:

$$\begin{aligned}
&\mathrm{Pr}_f[\ \mathrm{PD}(C(f))] \\
&= \sum_r \sum_c \mathrm{Pr}_f[C(f) = c\ \wedge\ \mathrm{PD}(c)\ \wedge\ c^*_m = r] \\
&= \sum_r \mathrm{Pr}_{c,f}[C(f) = c\ \wedge\ \mathrm{PD}(c)\ \wedge\ c^*_m = r] * 2^{mqn} \\
&= \sum_r \mathrm{Pr}_{c,f}[C(f) = c\ \wedge\ \mathrm{PD}(c) \mid c^*_m = r] * 2^{-qn} * 2^{mqn} \\
&= \sum_r 2^{-qn} * \mathrm{Pr}_c[\mathrm{PD}(\rho(c)) \mid c^*_m = r] \qquad \text{(by lemma 4)} \\
&\geq 1 - \Delta \quad \text{(by lemma 7)}
\end{aligned}$$

□

Since the adversary A decides 0 or 1 based on the oracle replies, say $O = \langle O^1, O^2, ..., O^q \rangle$, we can write its output as $\mathrm{A}(O)$. In scenario (a), O is really $C^*_m(f)$, with f chosen randomly. Since in scenario (b), the oracle is a random function with range n bits, O is just a uniformly random string of length qn.

Lemma 9.

$$Pr_{(b)}[A(O) = 0] \geq Pr_f[A(C^*_m) = 0\ \wedge\ PD(C(f))] \geq (1-\Delta)Pr_{(b)}[A(O) = 0]$$

Proof: To begin with, we have

$$\begin{aligned}
&\mathrm{Pr}_f[A(C^*_m) = 0\ \wedge\ \mathrm{PD}(C(f))\ ] = \sum_c \mathrm{Pr}_f[A(c^*_m) = 0\ \wedge C(f) = c\ \wedge\ \mathrm{PD}(c)] \\
&= 2^{mqn} * \mathrm{Pr}_{c \in_{\mathcal{U}} \{0,1\}^{mqn}, f}[A(c^*_m) = 0\ \wedge C(f) = c\ \wedge\ \mathrm{PD}(c)] \\
&= 2^{mqn} * \mathrm{Pr}_{c \in_{\mathcal{U}} \{0,1\}^{mqn}, f}[C(f) = c\ \wedge\ \mathrm{PD}(c) \mid A(c^*_m) = 0]\ *\ \mathrm{Pr}_{(b)}[A(O) = 0]
\end{aligned}$$

The above is at least $(1-\Delta)\mathrm{Pr}_{(b)}[A(O) = 0]$ by lemma 4 and lemma 7, and at most $\mathrm{Pr}_{(b)}[A(O) = 0]$. □

Proof of Theorem 2 (Main Theorem): By lemma 9 and lemma 8 it follows that

$$|\ \mathrm{Pr}_f[A(C^*_m(f)) = 0] - \mathrm{Pr}_{(b)}[A(O) = 0]\ |\ \leq\ \Delta$$

□

5 Variable Length Domain Extension and Family of Graphs

The previous constructions were devoted to extending the domain of a function from n bits to mn bits, for a fixed m. In other words, the plaintext queries of the adversary were restricted to be exactly mn bits. We could fix m to be large enough, say $m = 2^n$, and use a canonical encoding of smaller sized plaintexts into length mn bit strings. Such an encoding exists for all plaintexts of size less than mn by appending plaintexts of size q bits, by 10^i, where $i = mn - q - 1$. In other words, 10^i acts as an end marker. However, smaller sized plaintexts have to undergo $m = 2^n$ applications of F, which is very inefficient. This problem of a really long end marker was resolved by [23] (also see [7]) by noting that the end marker can actually be of length zero, if it can be authenticated.

The simplest way to achieve this is to have two independent PRFs $F1$ and $F2$. Use $F1$ when the plaintext is not a multiple of the block size n, and use $F2$ when the plaintext is a multiple of n. In the former case, append an end marker of the kind 10^i, but now i need only be of length at most $n - 2$.

So, given a function F_k on n bits, consider a collection of graphs, one graph G_q in the family for each (plaintext) bit-length q. Then if we define $\tilde{F}_k^{G_q}$ similarly to as before, we have a composite function from all strings to n bits. We know that individually each $\tilde{F}^{G_q}$ is a PRF given F is a PRF. As explained in the previous paragraph, we need to assure that these different functions are (almost) independent. We prove that if the family of graphs satisfy certain constraints then this is indeed the case.

We consider a fixed n throughout the rest of this section. We will assume that we are only interested in domain extension up to length $2^n * n$ bits, as theorem 2 is ineffective beyond that length (this restriction is only for sake of simplicity). Each query of the adversary will be a string p of length q bits, $(0 < q < 2^n * n)$. We let the composite function answers the query as follows: If q is a multiple of n, then it returns $f^{G_q}(p)$. Otherwise, let p' be p appended with 10^i, where i is the smallest positive number to make $|p'|$ a multiple of n. The composite function then returns $f^{G_{|p'|}}(p')$.

For every $0 \leq l < 2^n$, since strings of length $ln + 1$ to $ln + n - 1$ bits get canonically encoded in the above method, we can use the same graph for all these lengths. Thus, for each l, we really need only two graphs ([7]), one for lengths $ln + 1$ to $ln + n - 1$, and one for length $ln + n$. From now on, we will assume that all plaintexts are of bit length multiples of n. Each adversarial query will be a pair: (p, z), where p is a bit string of length multiple of n, and z is in $\{0, 1\}$ (z signifies if the plaintext was of length a multiple of n or if it was padded to make it so).

Definition 10. Let S be the set of all binary strings of length non-zero multiples of n, but less than $2^n * n$. Let $\mathcal{F}$ be the set of all functions:

$$S \times \{0,1\} \rightarrow \{0,1\}^n$$

Let $\tilde{F}$ be a function with signature:

$$\{0,1\}^k \times S \ \times \{0,1\} \rightarrow \{0,1\}^n$$

Given a PRF F from n bits to n bits, we need to define $\tilde{F}$ such that no adaptive adversary can distinguish between $\tilde{F}_K$, with K chosen randomly, and a function chosen uniformly at random from $\mathcal{F}$. As in the previous sections, given a function f from n bits to n bits, and given a collection of graphs $\mathcal{G}$, we first define a function $f^{\mathcal{G}}$ in $\mathcal{F}$.

Definition 11. Let $\mathcal{G}$ be a collection of edge-colored DAGs $G(l)$ (see definition 5), $l \leq (2^n - 1) * 2$. Each $G(l)$ is required to have unique source and sink nodes. Further, each $G(l)$ is required to have at least $\lceil \frac{l}{2} \rceil$ nodes. Define a function $f^{\mathcal{G}}$ as follows:

$$f^{\mathcal{G}}(p, z) = f^{G(2*|p|-z)}(p)$$

where f^G is as in definition 6. If the graph has more nodes than the length of the plaintext, then append enough zeroes to the plaintext. Usually, graphs will have exactly the required number of nodes. However, at the base cases, i.e. small length plaintexts, it may be necessary to have extra nodes. For an example, see Section 5.1.

For a theorem similar to theorem 2 to hold, we need further restrictions on $\mathcal{G}$. In particular, it will not be enough that individual graphs in $\mathcal{G}$ be non-singular. Since, we will need to extend the notion of non-singularity to the whole collection of graphs, it is best to fix a set of vertices V, and just define the edges and colorings for the individual graphs. Thus, we will define $E(l)$, and $\chi(l)$. The partial order $\prec_l$ is, as before, the transitive closure of $E(l)$.

To motivate the generalized definition of non-singularity, we first consider an example where it is not enough for individual graphs to be non-singular. Let $V = [1..4]$. The graphs are identical (see fig 5), except that the second graph $G(2)$ has an extra edge from 3 to 4. The first graph $G(1)$ is used to answer queries of length 3 blocks, and the second to answer queries of length 4. Clearly, both graphs are individually non-singular. Consider two queries, one of length three, and another of length four, the latter being just an extension of the first. However, the first graph's output is C_3, and is accessible to the adversary. Thus, during the second query the internal state C_3 is available to the adversary, and it can force M_4 to be any value of its choice.

This suggests that for each graph $G(i)$, the graph $G(i)$ itself cannot be allowed to be an induced subgraph of another graph $G(i')$. We prove that this condition is sufficient for the composite function to be a PRF.

Definition 12. For any vertex j in V, let U_j^l be the set of incident vertices of j in $G(l)$.
For any vertex j in V, we say $(l, j) \cong (l', j)$ if either $(j = 1)$ or

- $U_j^l = U_j^{l'}$, and
- for all $u \in U_j^l$: $\chi_l(\langle u, j \rangle) = \chi_{l'}(\langle u, j \rangle)$, and inductively $(l, u) \cong (l', u)$.

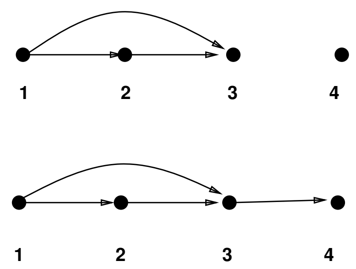

Fig. 5. An Incorrect Construction

Essentially, (l, j) is congruent to (l', j) if the two graphs $G(l)$ and $G(l')$ are identical till j.

Definition 13. Let $\mathcal{G} = \langle G(l) \rangle$, where each $G(l) = (V, E(l), \chi(l))$ is an edge-colored DAG, be a collection of graphs.

- With each $G(l)$ we associate its size $m(l)$ to be the largest numbered node in V such that there is an edge directed to it in $G(l)$.
- For each $G(l)$ we define the graph $\tilde{G}(l) = ([1..m(l)], E(l), \chi(l))$, to be the induced subgraph of $G(l)$ on vertices $[1..m(l)]$.

The collection $\mathcal{G}$ is called **PRF-preserving** if

- each $\tilde{G}(l)$ has only one source node, one sink node, has at least $\lceil \frac{l}{2} \rceil$ nodes, and
- if for any pair of nodes u, v ($u \neq v$), and graphs $G(l)$ and $G(l')$, the set of incident nodes of u in $G(l)$, and the set of incident nodes of v in $G(l')$ are same (say W), then for at least one $w \in W$, $\chi_l(\langle w, u \rangle) \neq \chi_{l'}(\langle w, v \rangle)$.
- for each graph $G(l)$, it is not the case that there is another graph $G(l')$, $l' \neq l$, s.t. $(l, m(l')) \cong (l', m(l'))$

Basically, the second condition above has extended the non-singularity requirement to be over all graphs.

Theorem 10. *: For a PRF-preserving collection of $2 * (2^n - 1)$ DAGs $\mathcal{G}$, let $f^{\mathcal{G}}$ be as in definition 11. Then, no adaptive adversary, with q adaptive queries $\langle (p^i, z^i) \rangle$ ($i \in [1..q]$, and $|p^i| \leq 2^n - 1$), can distinguish between (a) $f^{\mathcal{G}}$ where f is chosen uniformly at random from $\mathcal{F}(n \to n)$, (b) and a function chosen uniformly at random from $\mathcal{F}$, with probability more than $(\sum_{i \in [1..q]} |p^i|)^2 2^{-(n+1)}$.*

Proof: To adapt the proof of theorem 2, we first need to redefine the notion of consistent transcripts c. First note that, on a fixed transcript c, the queries of the adversary are fixed, say $\langle p^i, z^i \rangle_{i \in [1..q]}$. Recall, by definition of $f^{\mathcal{G}}$, on input p^i, z^i the graph $G(2 * |p^i| - z^i)$ is used. We just denote this graph by G^i. The corresponding edge relation, coloring and partial order will be denoted E^i, χ^i, and $\prec^i$ resp. Also, for the graph G^i, its induced subgraph as per definition 13, will be denoted $\tilde{G}^i$. Similarly, the size of the graph $\tilde{G}^i$ will be denoted by m^i. Note that $m^i = |c^i| \geq |p^i|$.

Definition 14. For any vertex j in V, let V^i_j be the set of incident vertices of j in G^i.
For any vertex j in V, we say $(i, j) \cong_c (i', j)$ if either $(j = 1)$ or
- $V^i_j = V^{i'}_j$, and
- for all $u \in V^i_j$: $\chi^i(\langle u, j\rangle) = \chi^{i'}(\langle u, j\rangle)$, and inductively $(i, u) \cong_c (i', u)$.

Essentially, (i, j) is congruent (w.r.t. c) to (i', j) if the two graphs G^i and $G^{i'}$ are identical till j.

Once we generalize the definition of $\equiv_c$, rest of the definitions and proofs remain almost the same.

Definition 15. For any vertices j, j', and query indices i, i' we say that $(i, j) \equiv_c (i', j')$ if

$$(j = j') \text{ and } (i, j) \cong_c (i', j) \text{ and } \forall k \preceq^i j : p^i_k = p^{i'}_k$$

As before, define

$$\mu_c(i, j) = \min\{i' | (i', j) \equiv_c (i, j)\}.$$

We call c **consistent** (con(c)) if

$$\forall j \in [1..2^n - 1], \forall i \ : \ c^i_j = c^{\mu_c(i,j)}_j$$

Define the following "correcting" function ρ:

$$\rho(c) = \overline{c}, \text{ where } \overline{c}^i_j = c^{\mu_c(i,j)}_j, \text{ for } j \in [1..m^i]$$

Since the proof of theorem 10 will be adapted from the proof of theorem 2, we will denote all lemmas for theorem 10 corresponding to lemmas for theorem 2 by the prime symbol. In the proof of lemma 5(a)$'$, if $m^i \neq m^{i'}$, then $(i, m) \not\cong_c (i', m')$. Otherwise, if the plaintexts p^i and $p^{i'}$ are different, then again $(i, m^i) \not\equiv_c (i', m^i)$. If the plaintexts are also same, then as the adversary does not repeat queries, wlog let $G^i = G(2 * m^i - 1)$, and $G^{i'} = G(2 * m^i)$. But $(i, m^i) \cong_c (i', m^i)$ is not allowed in $\mathcal{G}$ which is PRF-preserving. That proves lemma 5(a)$'$.

Proof of rest of lemma $5'$ is similar to proof of lemma 5. In the statement and proof of lemma 5(f)$'$, j must be restricted to be $[1..m^i]$. Similar restrictions apply in the definition of PD (definition 8) and definition of f_c (definition 9). Proof of lemma $6'$ is similar to proof of lemma 6.

Lemma 4 is now restated as (recall S from definition 10):

Lemma 4$'$. For every qn bit constant $\langle r^i \rangle$ $(i \in [1..q])$

$$\begin{aligned}&\Pr_{c \in_U S^q, f}[C(f) = c \ \wedge \ \text{PD}(c) \,|\, \forall i \ : c^i_{m^i} = r^i] \\ &= \ 2^{-mqn} * \Pr_{c \in_U S^q}[\text{PD}(\rho(c)) \,|\, \forall i \ : c^i_{m^i} = r^i]\end{aligned}$$

Proof Sketch: The proof is similar to proof of lemma 4, if we notice that we fix c in the first part of the proof. For a fixed c, let $I = \{(i, j) \,|\, \mu_c(i, j) = (i, j), j \in$

$[1..m^i]\}$. Let $T = \{M^i_j(c) \mid (i,j) \in I\}$. Since PD(c) holds, $|T| = |I|$. Let T' be an arbitrary set of n bit strings, disjoint from T, and $|T'| = \sum_{i\in[1..q]} m^i - |I|$. Thus, $|T \cup T'| = \sum_{i\in[1..q]} m^i$.

By, lemma 6′, $C(f_c) = c$. Thus, for each b agreeing with c on I, we have a function f_c defined on $|I|$ inputs T, such that $C(f_c) = c$. We can use the remaining $\sum_{i\in[1..q]} m^i - |I|$ values of b (i.e. from indices which are not in I) to extend f_c to be defined on $T \cup T'$. This map from b to the extended f_c is 1-1.

The reverse direction is done as in lemma 4.

Rest of the proof is also as in proof of lemma 4. □

Let Δ denote $(\sum_{i\in[1..q]} m^i)^2 * 2^{-(n+1)}$.

Lemma 7′. For every qn bit constant $\langle r^i \rangle$ $(i \in [1..q])$,

$$\Pr_{c\in_U S^q}[\ \mathrm{PD}(\rho(c)) \mid c^i_{m^i} = r^i]\ \geq 1 - \Delta$$

Proof: First note that for all i, $c^i_{m^i} = \rho(c)^i_{m^i}$, by lemma 5(a)′ and definition of ρ. As opposed to lemma 7, we need to show that it is not the case that a $\rho(c)^i_j$, with $j \neq m^i$, can be defined to be a $c^{i'}_j$, such that $j = m^{i'}$. Suppose, there is indeed an $(i',j) \equiv_c (i,j)$, such that $j = m^{i'}$. Since, $(i',j) \equiv_c (i,j)$, we have $(i',j) \cong_c (i,j)$. Thus the graphs G^i and $G^{i'}$ are identical till $j = m^{i'}$. Thus, unless they are the same graph, this is not allowed by the condition on PRF-preserving $\mathcal{G}$. If they are the same graph, then $j = m^i$, a contradiction.

Rest of the proof is similar to proof of lemma 7. □

Rest of the proof of theorem 10 is identical to that of theorem 2.

5.1 Applications to Variable Length Domain Extension

As an application of theorem 10, we get the variable length domain extension scheme as described in figure 6. In the figure, for each plaintext block length two graphs are given as required in definition 11. The number on the left of the graphs denotes the plaintext block lengths for which those graphs are to be employed. The 0/1 bit signifies if the plaintext was padded to make its bit-length a multiple of n. We have only illustrated graphs up to length five, as for larger lengths, we follow similar methods as for length four and five. Note that for plaintext block length one, we have graphs which have two nodes. As remarked at the end of definition 11, this requires that plaintexts of length one block must be appended with a zero block, before employing graphs “ONE-0” or “ONE-1”.

This mode has an advantage over XCBC [7], and OMAC [14] that it does not even need to employ the initial F on a constant like 0^n. Moreover, the scheme shows that if the plaintexts are restricted to be more than three blocks in length, then no Galois field arithmetic is required.

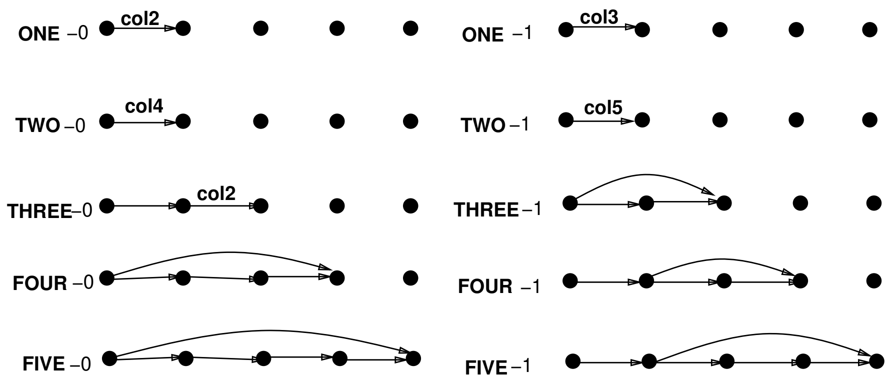

Fig. 6. A Variable Length Mode

References

1. ANSI X3.106, "American National Standard for Information Systems - Data Encryption Algorithm - Modes of Operation", *American National Standards Institute, 1983.*
2. M . Bellare, R. Canetti, H. Krawczyk, " Pseudorandom Functions Revisited: The Cascade Construction and its Concrete Security", Proc. IEEE FOCS 1996.
3. M. Bellare, J. Kilian, P. Rogaway, "The Security of Cipher Block Chaining", *JCSS*, Vol. 61, No. 3, Dec 2000, pp. 362-399
4. M. Bellare, A. Desai, E. Jokiph, P. Rogaway, "A Concrete Security Treatment of Symmetric Encryption: Analysis of the DES Modes of OPeration", 38th IEEE FOCS, 1997
5. D. Bernstein, " How to Stretch Random Functions: The security of Protected Counter Sums", J. of Cryptology, Vol 12,No. 3, (1999).
6. J. Black, P. Rogaway, " A Block Cipher Mode of Operation for Parallelizable Message Authentication", Proc. Eurocrypt 2002.
7. J. Black, P. Rogaway, "CBC MACs for arbitrary length messages: The three key constructions". CRYPTO 2000, LNCS 1880.
8. J. Carter, M. Wegman, "Universal Classes of Hash Functions", *JCSS*, Vol. 18, 1979, pp 143-154.
9. O. Goldreich, S. Goldwasser, and S. Micali, " How to construct random functions", J. ACM, vol. 33, no. 4, 1986.
10. National Bureau of Standards, Data Encryption Standard, U.S. Department of Commerce, FIPS 46 (1977)
11. V.D. Gligor, P. Donescu, "Fast Encryption Authentication: XCBC Encryption and XECB Authentication Modes", `http://csrc.nist.gov/encryption/modes/workshop1`
12. S. Halevi and P. Rogaway, "A Tweakable Enciphering Mode", CRYPTO 2003, LNCS 2729.
13. F. Harary, *Graph Theory*, Addison-Wesley 1969.
14. T. Iwata, K. Kurosawa, " OMAC: One -key CBC-MAC", FSE 2003, LNCS 2887.
15. C. S. Jutla, "Generalized Birthday Attacks on Unbalanced Feistel Networks", CRYPTO 1998, LNCS 1462.

16. C. S. Jutla, " Encryption Modes with Almost Free Message Integrity", *Proc. Eurocrypt 2001*, LNCS 2045, 2001.
17. Hugo Krawczyk, "LFSR-based Hashing and Authentication", *Proc. Crypto 94*, LNCS 839, 1994
18. H.W. Kuhn, "Extensive games and the problem of information" in *Contributions to the Theory of Games II*, H.W. Kuhn and A. W. Tucker eds., Annals of Mathematical Studies No. 28, Princeton Univ. Press, 1950.
19. M. Liskov, R. Rivest and D. Wagner, "Tweakable Block Ciphers", CRYPTO 2002, LNCS 2442.
20. M. Luby, "Pseudorandomness and Cryptographic Applications", *Princeton Computer Science Notes*, Princeton Univ. Press, 1996
21. M. Luby and C. Rackoff, "How to Construct Pseudorandom Permutations From Pseudorandom Functions", SIAM J. on Computing, Vol. 17, 1988, pp. 373-386.
22. M. Naor and O. Reingold, "On the construction of pseudo-random permutations: Luby-Rackoff revisited", *Proc. 29th ACM STOC*, 1997, pp 189-199.
23. E. Petrank, C. Rackoff, "CBC-MAC for real-time data sources", J. of Cryptology, vol 13, no. 3, nov 2000.
24. P. Rogaway, M. Bellare, J. Black and T. Krovetz, "OCB: A block-cipher mode of operation for efficient authenticated encryption", *Proc. 8th ACM Conf. Comp. and Comm. Security* (CCS), ACM, 2001.

APPENDIX

Lemma 6 For any consistent c such that $\mathrm{PD}(c)$ holds:

$$C(f_c) = c$$

Proof: Let $p = P(c)$ and $M = M(c)$ be shorthands. Also, we will use $\bar{c}$ as shorthand for $C(f_c)$. Similarly, let $\bar{M} = M(\bar{c})$, $\bar{p} = P(\bar{c})$.

We do induction over the query index.

Base Case: Since the adversary is fixed, the first plaintext message is the same, i.e. $\bar{p}^1 = p^1$. Since $\bar{M}_1^1 = p_1^1$, $\bar{c}_1^1 = f_c(\bar{M}_1^1) = f_c(M_1^1) = c_1^1$, as $(1,1)$ is trivially in I. For $j > 1$, $\bar{M}_j^1 = p_j^1 + \sum_{u:E(u,j)} \chi(\langle u,j\rangle) * \bar{c}_u^1$ But, by induction over the partial order $\prec$, $\bar{c}_u^1 = c_u^1$, hence $\bar{M}_j^1 = M_j^1$. Moreover, $(1,j)$ is trivially in I, and hence $\bar{c}_j^1 = c_j^1$.

So, assume that for all $i' < i$, and all j, $\bar{c}_j^{i'} = c_j^{i'}$. Thus, $\bar{p}^i = p^i$. Again, $\bar{M}_1^i = p_1^i = M_1^i$. Thus, $\bar{c}_1^i = f_c(M_1^i) = f_c(M_1^{\mu_c(i,1)})$ by lemma 5(h). By definition of f_c, this is same as $c_1^{\mu_c(i,1)} = c_1^i$. For $j > 1$, $\bar{M}_j^i = p_j^i + \sum_{u:E(u,j)} \chi(\langle u,j\rangle) * \bar{c}_u^i$. But, by induction over the partial order $\prec$, $\bar{c}_u^i = c_u^i$, thus $\bar{M}_j^i = M_j^i$. As before, using lemma 5(h), we are done. □