Subversion Resilient Hashing: Efficient Constructions and Modular Proofs for Crooked Indifferentiability

Rishiraj Bhattacharyya[1], Mridul Nandi[2], and Anik Raychaudhuri[2]

[1] University of Birmingham, UK, rishiraj.bhattacharyya@gmail.com
[2] Indian Statistical Institute, Kolkata, India, mridul.nandi@gmail.com,anikrc1@gmail.com

Abstract. We consider the problem of constructing secure cryptographic hash functions from *subverted* ideal primitives. Hash functions are used to instantiate Random Oracles in cryptographic protocols. The indifferentiability security notion is a popular tool to certify the structural soundness of a hash design for such instantiations. In CRYPTO 2018, Russell, Tang, Yung, and Zhou introduced the notion of crooked-indifferentiability to extend this paradigm even when the underlying primitive of the hashing mode is subverted. They showed that an n-to-n-bit function implemented using Enveloped XOR construction (EXor) with $3n+1$ many independent n-to-n-bit functions and $3n^2$-bit random seed can be proven secure asymptotically in the crooked-indifferentiability setting. Unfortunately, known techniques to prove crooked-indifferentiability are extremely complicated, and no practical hashing mode has been analyzed in this setting.

- We introduce new techniques to prove crooked-indifferentiability. We establish that upper bounding the subversion probability of a chaining query is sufficient to argue subversion resistance of a standard indifferentiable mode of operation. Our technique links standard indifferentiability and crooked-indifferentiability and circumvents the complications of proving the consistency of the simulator in the crooked setting.
- We prove crooked-indifferentiability of the sponge construction when the underlying primitive is modelled as an n-to-n-bit random function. Our proofs only require n-bit randomly chosen but fixed IV and do not mandate any independent function requirement. The result naturally extends to the Merkle-Damgård domain extension with prefix-free padding. Our results minimize required randomness and solve the main open problem raised by Russell, Tang, Yung, and Zhou.

1 Introduction

We consider the problem of designing Cryptographic Hash Functions from subverted primitives. Traditionally cryptographic hash functions are designed via applying a domain extension algorithm on suitable primitives of a smaller domain. Security of the hash functions is often derived via information-theoretic arguments assuming the underlying primitives behave as ideal where the adversary is permitted only to query the primitives. In practice, however, the implementations of the primitives may leak more information to the adversary and possibly even allow malicious tampering. A good example is the Dual-EC tampering attack [15] which led to the withdrawal of a standardized PRG due to a potential backdoor in the implementation.

The framework of Kleptography, introduced by Young and Yung [31,32] more than twenty years ago, allows a "proud but curious" adversary to replace a cryptographic implementation with a crooked version intending to subvert its security without getting caught. Bellare, Paterson, and Rogaway [6] revitalized the framework under the name of Algorithmic Substitution Attack (ASA). They showed that it is possible to mount an algorithm substitution attack against almost all known symmetric key encryption schemes to the extent that the attacker learns the secret key. A series of work has been done in recent years formalizing approaches to resist algorithm substitution attacks [21,5,26,19,20,28,29,2,4].

Indifferentiability of Hash Functions and Security against ASA. Hash functions are ubiquitous in modern cryptography. Hash functions are widely popular as the drop-in replacements of Random Oracles (RO) in cryptographic schemes and protocols. To facilitate this application, the notion of *indifferentiability* from a Random Oracle, introduced by Maurer, Renner, and Holenstein [24], has been established as a mainstream security criterion. Indifferentiability from a Random Oracle implies all security guarantees (like collision resistance) satisfied by a Random Oracle in a single-stage game up to the indifferentiability bound. Starting from the work of Coron, Dodis, Malinaud, and Puniya [18], a plethora of results [14,9,11,23,1,12,10,25,27] have been proven to show indifferentiability of different constructions based on different ideal primitives.

Surprisingly, analysis of secure hash functions against ASA has been scarce. In CRYPTO 2018, Russel, Tang, Yung and Zhou [30] studied the problem of correcting subverted Random Oracles. They introduced the notion of Crooked-Indifferentiability as a replacement for classical indifferentiability for the kleptographic setting. They showed that the Enveloped XOR construction could be proven secure in this framework.

Like classical indifferentiability, the game of crooked-indifferentiability challenges the adversary to distinguish between two worlds. In the real world, the adversary has access to the underlying ideal primitive f, and the construction C, which has subroutine access to $\tilde{f}$, the subverted implementation of f.[3] The implementation $\tilde{f}$ on input an element x queries the function (possibly adaptively) at maximum $\tilde{q}$ many points and, based on the transcript, decides the evaluation of x. As the adversary likes the subversion to go undetected, it is assumed that $\tilde{f}$ differs from f only on some negligible fraction (ϵ) of the domain.

In the ideal world, the construction is replaced by a Random Oracle $\mathcal{F}$. The role of f is played by a simulator with oracle access to $\mathcal{F}$ and the subverted implementation $\tilde{f}$. The job of the simulator is to simulate f in such a way that $(C^{\tilde{f}}, f)$ is indistinguishable from $(\mathcal{F}, S^{\mathcal{F},\tilde{f}})$. In order to avoid trivial attacks, the framework allows a *public* random string R to be used as the salt in the construction. The string R is fixed after the adversary publishes the implementation but stays the same throughout the interaction. All the parties, including the simulator and the adversary, get R as part of the initialization input. We note that even in the weaker setting of Random Oracles with auxiliary input, a random salt is required to prove security [17,22].

The notion of crooked-indifferentiability from a Random Oracle and the composition theorem proved in [30] guarantees that a construction proved secure in this framework can be used to replace a Random Oracle in any single-stage game in the kleptographic setting. While popular hash functions are the most natural choice for instantiating the Random Oracle, their suitability is still unknown. We ask, *can the popular hashing modes, for some parameters, achieve this many-fold stronger security notion?* Given the surge of new constructions in the ASA setting [16,2,3,4], the importance of the question cannot be overstated.

Proving a construction secure in the crooked-indifferentiability setting is an immensely challenging task. Unlike the classical setting where the adversary is passive, the crooked-indifferentiability adversary is active and could subvert any algorithm. The only known crooked-indifferentiability bound is for the construction called Enveloped XOR (EXor), depicted in Figure 1. In [30], the authors using the rejection-sampling technique showed the security of EXor construction. The instantiation requires $3n + 1$ many independent functions and n^2 many random bits. We note, however, that the Enveloped XOR construction produces an n-bit to n-bit random function. Instantiating a hash function would require applying domain extension techniques on top of it, implying more function calls and possibly more independent random bits. *Minimizing randomness and reducing the number of function calls while still achieving crooked-indifferentiability was left as the main challenge in [30].*

Finally, the technique of [30], though ingenious, is very complex. It is difficult to give an intuitive justification for why the construction and the approach work. The alternative proof of [13] is also quite involved. Given that we have established tools to prove indifferentiability in the classical setting, it is natural to ask whether we can leverage those tools to prove crooked-indifferentiability.

[3] The domain extension algorithms are simple, and the correctness of their implementations are easy to verify.

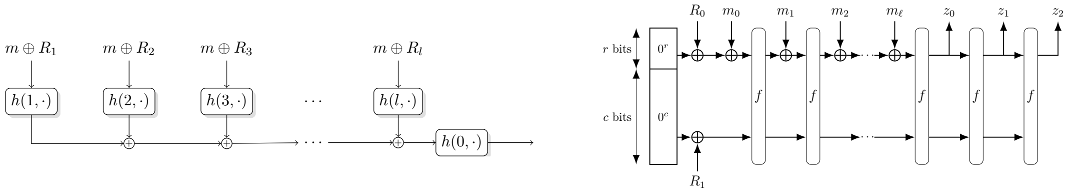

Fig. 1. EXor construction (left) and Sponge Construction with random IV (right).

1.1 Our Contributions

We introduce new techniques to prove crooked-indifferentiability and establish security bounds for popular hashing modes, the sponge construction and the ubiquitous Merkle-Damgård construction. We elaborate on our contributions below.

New Techniques for Crooked-Indifferentiability. We present new techniques to prove Crooked-Indifferentiability. We introduce a new security game called Force-Crook, where the challenge to the adversary is to produce a message for which the construction makes a primitive query on a subverted input. We show that bounding the advantage of the adversary in the Force-Crook game is sufficient to prove Crooked-Indifferentiability of constructions secure under the classical indifferentiability paradigm.

Crooked-Indifferentiability of Popular Hashing Modes. We apply our techniques to prove the security of popular hashing modes. Our main contribution is to show that the sponge construction, instantiated with a random function and a randomized initial value, is crooked-indifferentiable from a Random Oracle. The construction uses the same function at every iteration. The design is identical to the one proven indifferentiable in [9]. This result positively answers our quest for a practical crooked-indifferentiable hashing mode. Moreover, the proof requires only a linear (in terms of the security parameter) number of random bits and thus answers the *main open question* raised by RTYZ [30].

We show that the technique with a minor modification is sufficient to prove the security of the classical Merkle-Damgård construction with prefix-free padding. The hash function uses an $n+1$-to-n-bit compression function.

1.2 Overview of Our Techniques

Technical Challenges in Crooked-Indifferentiability. The main challenge in the crooked setting is to prove the randomness of the construction's output. As the underlying primitives are subverted, the adversary may have full information about the function on some points without querying the oracles. Consider the following example. We are given an n-to-n-bit random function f. By definition, f is classically indifferentiable from a random oracle. Now consider a simple subverted implementation $\tilde{f}$ of f. The program $\tilde{f}$ honestly implements f everywhere except at point 0, where it outputs $\tilde{f}(0) = 0$. Such an $\tilde{f}$ can be easily distinguished from a random oracle.

The established technique to correct the situation would be the random-masking technique, but that does not work either. Consider, for example, simple input masking with a random string R obtained by the function $g_R(M) \overset{\text{def}}{=} f(M \oplus R)$. As the string R is fixed at the start of the game (after the adversary submits the subverted implementation), the distinguisher can indeed choose the message $M = R$, resulting in a distinguishing condition $g_R(R) = 0$. From the above two examples, one can abstract out the first challenge of proving crooked-indifferentiability. *The output distribution of the underlying primitive, conditioned on the adversary's view, is not uniform for every point.* The challenge becomes even more daunting when we consider an implementation that can subvert a point based on the function evaluations at that and possibly some other points. We can no longer assume function values are independently distributed. Thus the tools and techniques developed for classical indifferentiability seem to be useless here.

The Intermediate Game Force-Crook. We found a seemingly obvious but powerful technique to handle subversions. The difference between the real world in the crooked-indifferentiability and the real world in the classical indifferentiability setting is only in the oracle of the construction C. In the crooked setting, C is given oracle access to $\tilde{f}$ whereas, in the classical setting, C queries the primitive f itself. As long as no chaining value results in querying f on a crooked point, the output distributions of these two worlds are identical! In other words, if for every message M submitted by the adversary to C, it holds with a high probability that $C^f(M) = C^{\tilde{f}}(M)$, then (C^f, f) and $(C^{\tilde{f}}, f)$ are indistinguishable. If C is indifferentiable in the classical setting, then that simulator would work perfectly as the simulator in the crooked setting.

In Section 3, we introduce a security game Force-Crook where the adversary is challenged to find a message where $C^f(M) \neq C^{\tilde{f}}(M)$. We show that for a construction proven indifferentiable from a random oracle in the classical setting (with security bound δ_i), the crooked-indifferentiability advantage is bounded by the advantage of winning the Force-Crook game plus δ_i.

Bounding Winning Advantage of Force-Crook To bound the adversary's success probability of winning the game Force-Crook, we focus on ensuring all the chaining inputs remain uncrooked with high probability. Our intuition is to argue that if a chaining query is uncrooked, the output is uniform. Given that only a negligible fraction of points are crooked, when we use random iv, the first chaining inputs are random and, thus, with high probability, uncrooked. Suppose only a few bits of the message are injected at every iteration. Then the following chaining query input is close to being uniform and, thus with high probability, uncrooked as well. Now we can repeat this argument throughout the computation of C. For the sponge and Merkle-Damgård constructions, this idea in itself is sufficient for handling simple subversion.

We explain it in more detail for the following simplified setting. Suppose the subverted implementation $\tilde{f}$ is such that on input a point x, the output of $\tilde{f}(x)$ depends only on the value of $f(x)$, and it is independent of $f(y)$ for all $y \neq x$. Consider the sponge construction based on a random function $f : \{0,1\}^n \to \{0,1\}^n$. By definition of worst-case subversion by a proud but curious adversary, for *all* choices for the function f, at most ϵ fraction of the inputs are crooked $(\tilde{f}(x) \neq f(x))$. In addition, there are at most q_1 many points queried by the implementor before producing the subverted implementation. Hence for every function f, there is a set S_f of size at least $(1-\epsilon)2^n - q_1$ whose members are neither fixed by the implementor nor subverted. For a randomly chosen function f and a random string, with overwhelming probability, the random string will be a member of S_f. If we set the rate part of the sponge construction to be 1, for both the choice of $m_0 \in \{0,1\}$, the first chaining query to f will be a member of S_f with probability $(1 - 2\epsilon - \frac{2q_1}{2^n})$.

We can repeat the above argument inductively. Consider the lazy sampling framework of random functions. We say a chaining query x_i is good if, for all choices of $m_{i+1} \in \{0,1\}$, the next chaining query $x_{i+1} = f(x_i) \oplus m_{i+1}$ is subverted with low probability (say $\epsilon^{\frac{1}{2}}$). In other words, x_{i+1} is a member of S_f with high probability. One can show that a randomly chosen point is good with high probability. As $f(x_i)$ is uniformly distributed, x_{i+1} would also be a good chaining query. For the base case of the induction argument, we recall that the first chaining query is generated from the initial random string. For all values of $m_0 \in \{0,1\}$, it is a good chaining query with high probability. Thus we get all the chaining queries would be good, and by extension, all the chaining queries will be uncrooked with overwhelming probability.

The matter gets complicated when we consider a general $\tilde{f}$ whose output can depend on adaptively chosen multiple points. With careful analysis, *we extend our arguments to this general case.* In Section 4, we present the analysis in detail.

1.3 Impact of Our Results

Subversion Agnostic Indifferentiability. We achieve a strong form of crooked-indifferentiability where the simulator is subversion agnostic. When we establish crooked-indifferentiability via the

Force-Crook game, S does not even need access to subverted implementation $\tilde{f}$. While we show sponge and Merkle-Damgård attain such security, not all constructions achieve such strong crooked-indifferentiability. One notable example is the Enveloped Xor construction, where the simulator must have access to $\tilde{f}$ to achieve crooked-indifferentiability as formulated in [30]. Thus our modular proof technique illustrates a simple condition for a classical indifferentiable construction to achieve crooked-indifferentiability.

Crooked vs Classical. A learned reader may observe that a crooked-indifferentiable construction's efficiency and security parameters are worse than what can be proven in the classical indifferentiability setting. One can wonder about the crooked-indifferentiability framework's significance and our results' impact. In particular, for the sponge construction with n bit function, we prove crooked indifferentiability security of asymptotically $n/4$ bits when at each round, *one* bit of message is injected and $\epsilon \leq 1/2^{n/2}$. In contrast, SHA3, with each iteration consuming r bits of messages, achieves $(n-r)/2$ bits of security in the classical indifferentiability setting.

However, comparing bit-security without considering the adversary's power leads to misleading impressions. While proving indifferentiability, we aim to achieve independent and uniformly sampled hash output for *every* point. The classical indifferentiability assumes that an adversary is passive and is content with only black-box access to the underlying primitive. Thus, the primitive could be modelled as ideal. In particular, each point is mapped independently following a high-entropy probability distribution.

In comparison, the adversary in the kleptographic setting is *active*. The implementation of the primitive is subverted. The points are not mapped independently, and for some "small" yet non-zero fraction of the inputs, the adversary has carefully *chosen* the function. We can no longer directly leverage the randomness of the underlying primitive. It is natural that the security-efficiency tradeoff achieved in the crooked setting against such an active adversary is somewhat weaker than what is accomplished against the passive adversary of the classical indifferentiability paradigm.

2 Notations and Preliminaries

Notations. Let $\mathbb{N} = \{0, 1, \ldots\}$ be the set of natural numbers and $\{0,1\}^*$ be the set of all binary strings. For a positive integer n, the term $\{0,1\}^n$ denotes the set of all n-bit binary strings. If x and y are two strings, xy denotes the concatenated string. We write $x \overset{\$}{\leftarrow} S$ to denote the process of choosing x uniformly at random from a set S and independently from all other random variables defined so far. For a positive integer l, we use $(l]$ and $[l]$ to denote the set $\{1, \ldots, k\}$ and $\{0, 1, \ldots, k\}$ respectively.
Class of Functions. $\mathsf{H}_{\mathcal{D},\mathcal{R}}$ denotes the set of all functions from $\mathcal{D}$ to $\mathcal{R}$. $\mathsf{F}_{m,n}$ denotes the set of all functions from $\{0,1\}^m$ to $\{0,1\}^n$. $f : (k] \times \mathcal{D}_f \to \mathcal{R}_f$ denotes a family of k many functions from $\mathcal{D}_f$ to $\mathcal{R}_f$. We often use the shorthand f to denote the family $\{f_1 := f(1,\cdot), \ldots, f_k := f(k,\cdot)\}$ when the function family is given as oracles.

For any tuples of pairs $\tau = ((x_1, y_1), \ldots, (x_{|\tau|}, y_{|\tau|}))$ we write $\mathcal{D}(\tau)$ (called domain of τ) to denote the set $\{x_i : 1 \leq i \leq |\tau|\}$. We write $\tau_j = ((x_1, y_1), \ldots, (x_j, y_j))$. We say a function f agrees with τ if for all $(x, y) \in \tau$, $f(x) = y$. For every $x \in \mathcal{D}_f$, $\alpha \in \mathcal{R}_f$, we use $f_{x\to\alpha}$ to denote the following function:

$$f_{x\to\alpha}(y) = \begin{cases} f(y) & \text{if } x \neq y \\ \alpha & \text{if } x = y \end{cases}.$$

Security Games. The results are proven in the framework of code-based games [7]. A game G consists of a `main` oracle and zero or more stateful oracles $O_1, O_2, \ldots, O_n$. If a game G is implemented using a function f, we write $G[f]$ to denote the game. The success probability of algorithm $\mathcal{A}$ in the game G is defined by $\mathbf{Succ}_{\mathcal{A},G} \overset{\text{def}}{=} \Pr\left[G^{\mathcal{A}} = 1\right]$. The query complexity of $\mathcal{A}$ is the number of queries made by $\mathcal{A}$ to its oracles.

Definition 1 (Domain Extension). *Let $\mathcal{D} \supseteq \mathcal{D}_f$. A domain extender C with oracle access to a family of functions $f : (k] \times \mathcal{D}_f \to \mathcal{R}$ is an algorithm that implements the function $\mathtt{H} = C^f : \mathcal{D} \to \mathcal{R}$.*

During the computation of $C^f(M)$, the f query inputs made by C are called the *chaining queries.* **Adversaries and Distinguishing Advantage.** An *adversary* $\mathcal{A}$ is an algorithm possibly with access to oracles $\mathcal{O}_1, \ldots, \mathcal{O}_k$ denoted by $\mathcal{A}^{\mathcal{O}_1,\ldots,\mathcal{O}_k}$. The adversaries considered in this paper are computationally unbounded. The complexities of these algorithms are measured solely on the number of queries they make. An algorithm $\mathcal{A}$ having access to an oracle is called a q-query algorithm if it makes at most q queries to its oracle. Similarly, an oracle algorithm having access to two oracles is called a (q_1, q_2)-query algorithm if it makes at most q_1 and q_2 queries to its first and second oracles, respectively. Adversarial queries and the corresponding responses are stored in a transcript τ. $\mathcal{D}(\tau)$ denotes the list of inputs (queries) in the transcript.

Definition 2 (Distinguishing Advantage). *Let F^k and G^k be two k-tuples of probabilistic oracle algorithms for some positive integer k. We define the* advantage *of an adversary $\mathcal{A}$ at distinguishing F^k from G^k as*

$$\varDelta_{\mathcal{A}}(F^k \ ; \ G^k) = \left|\Pr[\mathcal{A}^{F_1,F_2,\ldots,F_k} = 1] - \Pr[\mathcal{A}^{G_1,G_2,\ldots,G_k} = 1]\right|.$$

2.1 Classical Indifferentiability

An oracle construction $C^{\mathcal{O}}(\cdot,\cdot)$ with a randomized *initial value* (IV) first fixes the IV R (chosen randomly from an initial value space). Afterwards, on input M, the construction C interacts with the oracle $\mathcal{O}$, and finally, it returns an output, denoted as $C^{\mathcal{O}}(R, M)$. When the initial value space is a singleton (i.e., degenerated), we call C an oracle construction. An (IV-based) oracle construction C is called *$\mathcal{F}$-compatible* if the domains and ranges of C and $\mathcal{F}$ (an ideal primitive) are the same. Now we state the definition of indifferentiability of an oracle construction as stated in [18,24] in our terminologies. In the following definition, adversary $\mathcal{A}$ and simulator S have independent, private random coins. Construction C has the random initial vector R, sampled at the start and fixed throughout the game. The adversary $\mathcal{A}$ and the simulator S receive R as input.

Definition 3 (Indifferentiability). *Let $\mathcal{F}$ be an ideal primitive and C^P be an $\mathcal{F}$-compatible oracle construction. C is said to be* $((q_P, q_C, q_{\text{sim}}), \varepsilon)$-indifferentiable *from an ideal primitive $\mathcal{F}$ if there exists a q_{sim}-query algorithm $S^{\mathcal{F}}$ (called simulator) such that for any (q_P, q_C)-query algorithm $\mathcal{A}$, it holds that*

$$\varDelta_{\mathcal{A}(R)}\left((P, C^P(R,\cdot)) \ ; \ (S^{\mathcal{F}}(R,\cdot), \mathcal{F})\right) < \varepsilon.$$

where R is the random initial vector of the construction C, chosen uniformly from the initial coin space and provided to the adversary $\mathcal{A}$, simulator S.

In the above definition, one may include the adversary and simulator's complexity (time, query etc.). However, for information-theoretic security analysis, we may ignore their time complexities.[4] A popular indifferentiability treatment for hash functions considers $\mathcal{F}$ to be an n-bit random oracle that returns independent and uniform n-bit strings for every distinct query. However, the hash function C^P can be defined through different types of primitives P (a random oracle, or a random permutation π_n, chosen uniformly from the set of all permutations over $\{0,1\}^n$).

[4] one can easily extend the concrete setup to an asymptotic setup. Let $\langle \mathcal{F}_n, P_n \rangle_{n \in \mathbb{N}}$ be a sequence of primitives and $C(n)$ be a polynomial time $\mathcal{F}_n$-compatible oracle algorithm. $C^{P_n}(n)$ is said to be (computationally) indifferentiable from $\mathcal{F}_n$ if there exists a polynomial-time simulator $S^{\mathcal{F}_n}$ such that for all polynomial-time oracle algorithm $\mathcal{A}$, $\varDelta_{\mathcal{A}}\left((P_n, C^{P_n}(n)) \ ; \ (S^{\mathcal{F}_n}, \mathcal{F}_n)\right) = \mathsf{negl}(n)$.

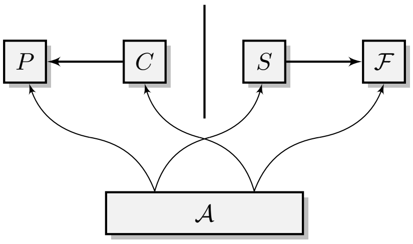

Fig. 2. The indifferentiability security notion. The real world consists of the construction C and the underlying ideal primitive P. The ideal world consists of the ideal primitive $\mathcal{F}$ and the simulator S. The construction C has oracle access to the underlying primitive P. The simulator S has oracle access to $\mathcal{F}$. When C has a random IV R, the distinguisher $\mathcal{A}$ and the simulator S receive R as input. The distinguisher $\mathcal{A}$ interacts either with the real world or with the ideal world.

2.2 Modeling Subversion Algorithms and Crooked-Indifferentiability

We recall the related terms and notations introduced in [30] in our terminologies.

Implementer. A $(q, \tilde{q})$ *implementer* is a q-query oracle algorithm $\mathcal{A}^{\mathcal{O}}$. $\mathcal{A}$ outputs the description of another oracle algorithm $\tilde{F}^{\mathcal{O}}$. The algorithm $\tilde{F}^{\mathcal{O}}$ makes at most $\tilde{q}$ many queries to its oracle. We call $\tilde{F}$ the *implementation.* We let $\tilde{\tau}$ denote the transcript of oracle queries of $\mathcal{A}$. The transcript $\tilde{\tau}$ is hardwired in $\tilde{F}$, and all the $\tilde{q}$ queries made by $\tilde{F}$ are different from $\mathcal{D}(\tilde{\tau})$.

The implementation $\tilde{F}$ is *correct* if for all $f \in \mathsf{H}_{\mathcal{D}_f,\mathcal{R}_f}$ and for all $x \in \mathcal{D}_f$, $\tilde{f}(x) \stackrel{\text{def}}{=} \tilde{F}^f(x) = f(x)$.

A subverted implementation $\tilde{f}$ on input x queries $\alpha_1^{(x)}, \alpha_2^{(x)}, \ldots, \alpha_{\tilde{q}}^{(x)}$, and based on the query-responses outputs $\tilde{f}(x)$. Without loss of generality, we assume $\alpha_1^{(x)} = x$, that is the first query of $\tilde{f}(x)$ is $f(x)$. We use $\alpha \twoheadrightarrow_f \alpha'$ to denote that $\tilde{f}(\alpha)$ queries $f(\alpha')$. Similarly, $\alpha \not\twoheadrightarrow_f \alpha'$, denotes that $\tilde{f}(\alpha)$ does not query $f(\alpha')$. We define the following two sets: (1) $\tilde{Q}_f(x) \stackrel{\text{def}}{=} \{y \mid x \twoheadrightarrow_f y\}$ and (2) $\overrightarrow{Q}_f(x) \stackrel{\text{def}}{=} \{y \mid y \twoheadrightarrow_f x\}$. Specifically, $\tilde{Q}_f(x)$ denotes the set $\{\alpha_1^{(x)}, \alpha_2^{(x)}, \ldots, \alpha_{\tilde{q}}^{(x)}\}$. $\overrightarrow{Q}_f(x)$ denotes the set of all points whose (subverted) evaluation queries the point x.

Definition 4 (Crooked Implementer). *A $(q, \tilde{q})$ implementer $\mathcal{A}_1$ is called ϵ-*crooked *for a function family $\mathsf{H}_{\mathcal{D}_f,\mathcal{R}_f}$, if for every $f \in H_{\mathcal{D}_f,\mathcal{R}_f}$, it holds that*

$$\Pr_{\alpha \xleftarrow{\$} \mathcal{D}_f}[\tilde{f}(\alpha) \neq f(\alpha)] \leq \epsilon$$

where $\tilde{f} \leftarrow \mathcal{A}_1^f$.

Summary. A (crooked) implementation $\tilde{f}$, to compute $\tilde{f}(x)$, queries $f(\alpha_1^{(x)}), \ldots, f(\alpha_{\tilde{q}}^{(x)})$ on $\tilde{q}$ many distinct points ($\alpha_1 = x$) and its decision of whether to subvert $f(\alpha)$ depends on this transcript and the hardwired string $\tilde{\tau}$. For an ϵ-crooked implementation, for each $f \in \mathsf{H}_{\mathcal{D}_f,\mathcal{R}_f}$, for at most ϵ fraction of $x \in \mathcal{D}_f$, $f(x)$ is subverted.

Detection Algorithm. Given an implementation, one may check the algorithm's correctness by comparing the implementation's outputs with a known correct algorithm. More precisely, we sample $\alpha_1, \ldots, \alpha_t \xleftarrow{\$} \{0,1\}^m$ and then for all $0 \leq i \leq l$, we check whether $\tilde{f}(\alpha_i) = f(\alpha_i)$ holds. If it does not hold, the implementation will be discarded. It is easy to see that for an ϵ-crooked implementation; the subversion would be detected with a probability of at most $t\epsilon$. So for negligible ϵ, this probability would be negligible for all polynomial function t, and the implementation can survive for further use.

Crooked Distinguisher. A crooked distinguisher is a two-stage adversary; the first stage is a crooked implementer and the second stage is a distinguisher.

Definition 5 (Crooked Distinguisher). *We say that a pair $\mathcal{A} := (\mathcal{A}_1, \mathcal{A}_2)$ of probabilistic algorithms $((q_1, \tilde{q}, \epsilon), q_2)$-*crooked distinguisher *for $\mathsf{H}_{\mathcal{D}_f,\mathcal{R}_f}$ if* *(i) $\mathcal{A}_1(r)$ is a ϵ-crooked $(q_1, \tilde{q})$ implementer for $\mathsf{H}_{\mathcal{D}_f,\mathcal{R}_f}$ and*

(ii) $\mathcal{A}_2(r, \tilde{\tau}, R)$ is a q_2-query distinguisher where r is the random coin of $\mathcal{A}$, $\tilde{\tau}$ is the advice-string, the transcript of the interaction of $\mathcal{A}_1$ with f, and R is the (randomized) initial vector of the target construction. The random string r and the advice-string $\tilde{\tau}$ are hardwired to $\mathcal{A}_2$, and the random IV R is provided as input.

Crooked-Indifferentiability. Now, we state the crooked-indifferentiable security definition (as introduced in [30]) in our notation and terminology. The definition is based on the following two-stage distinguishing game. The ideal primitives f and $\mathcal{F}$ are sampled. The crooked-distinguisher $\mathcal{A}$ (with random string r as the random coins) runs the first phase $\mathcal{A}_1$. The crooked implementer $\mathcal{A}_1$, with oracle access to f, produces a subverted implementation $\tilde{F}$. Then, a uniformly random string R is sampled and published as the IV of the construction C. Finally, $\mathcal{A}_2$ is invoked with an internal random string r, the advice-string $\tilde{\tau}$, and the random IV R as input. In the real world, $\mathcal{A}_2$ interacts with the f (same from the first stage) and the construction $C^{\tilde{f}}(R, \cdot)$. In the ideal world, the simulator S gets the advice-string $\tilde{\tau}$, the initial value R and blackbox access to the subverted implementation $\tilde{F}$ as inputs, along with oracle access of a random oracle $\mathcal{F}$. The simulator is aimed to simulate f so that the behaviour of $(f, C^{\tilde{f}})$ is as close as $(S, \mathcal{F})$ to the distinguisher $\mathcal{A}_2$.

Definition 6 (Crooked-Indifferentiability [30]). *Let $\mathcal{F}$ be an ideal primitive and C^f be an IV-based $\mathcal{F}$-compatible oracle construction. The construction C is said to be* $((q_1, \tilde{q}), (q_2, q_{\text{sim}}), \epsilon, \delta)$-**crooked-indifferentiable from** $\mathcal{F}$ *if there is a q_{sim}-query algorithm S (called simulator) such that for all $((\epsilon, q_1, \tilde{q}), q_2)$-crooked distinguisher $(\mathcal{A}_1(r), \mathcal{A}_2(r, \cdot, \cdot))$ for $\mathsf{H}_{\mathcal{D}_f,\mathcal{R}_f}$, we have*

$$\Delta_{\mathcal{A}_2(r,\tilde{\tau},R)}\big((f, C^{\tilde{f}}(R, \cdot)) \ ; \ (S^{\mathcal{F},\tilde{F}}(\tilde{\tau}, R), \mathcal{F})\big) \leq \delta \tag{1}$$

where $\tilde{\tau}$ is the advice string of $\mathcal{A}_1^f$. R is the random initial value of the construction sampled after the subverted implementation is set.

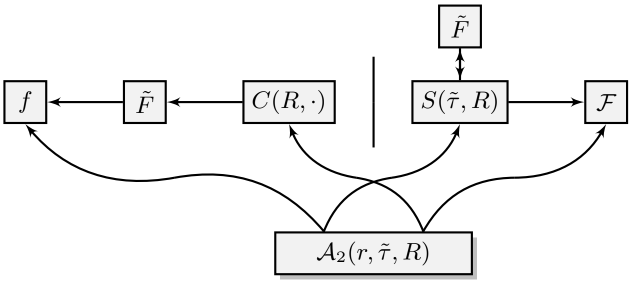

Fig. 3. The crooked-indifferentiability notion. In the first phase of the real world, $\mathcal{A}_1$ interacts with f and returns an oracle algorithm $\tilde{F}$ (which would be accessed by the construction C in the second phase). In the second phase, the random initial value R will be sampled and given to construction C and also to $\mathcal{A}_2$. In the ideal world, the simulator $S^{\mathcal{F}}$ gets the transcript of the first phase as an advice string, blackbox access to the subverted implementation $\tilde{F}$ and the initial value R.

Remark 1. The simulator S gets a blackbox subroutine access to the algorithm $\tilde{F}$. The simulator can compute $\tilde{F}(x)$ by invoking $\tilde{F}$ with input x and responding to the oracle queries made by $\tilde{F}$.

Convention on Crooked Distinguishers. Note that there is no loss in assuming that both $\mathcal{A}_1$ and $\mathcal{A}_2$ are deterministic (so we skip the notation r) when we consider a computationally unbounded adversary. $\mathcal{A}$ can fix the best internal random coin r for which the distinguishing advantage of $\mathcal{A}_2$ is maximum. As the randomness of $f, \mathcal{F}$, the public IV R and the internal random coins of S are independently sampled from r, the maximum distinguishing advantage would follow from an averaging argument.

We also assume that $\mathcal{A}_2$ makes all distinct queries distinct from those made by $\mathcal{A}_1$. We skip the notation $\tilde{\tau}$ as an input of $\mathcal{A}_2$ as it is fixed throughout the game. As the advice string is fixed, we consider it part of the transcript. Specifically, the transcript τ_0, view of $\mathcal{A}_2$ at the start of the second stage, is set as the advice string $\tilde{\tau}$.

2.3 Markov Inequality

Lemma 1. *Let X be a non-negative random variable and $a > 0$ be a real number. Then it holds that*

$$\Pr[X \geq a] \leq \frac{\mathbb{E}(X)}{a}.$$

A simple application of Markov inequality (which is used repeatedly in this paper) is the following. Consider a joint distribution of random variables X and Y. Suppose E is an event for which $\Pr[(X,Y) \in E] \leq \epsilon$. Let $f(x) := \Pr[(X,Y) \in E | X = x]$ and $E_1 := \{x : f(x) \geq \delta\}$. It follows from the definition that $\mathbb{E}(f(X)) = \Pr[E]$. Now, we use Markov's inequality

$$\begin{aligned}\Pr[E_1] &= \Pr[f(X) \geq \delta] \\ &\leq \mathbb{E}(f(X))/\delta \\ &= \epsilon/\delta.\end{aligned}$$

Note that when X and Y are independent, $f(x) = \Pr[(x, Y) \in E]$.

2.4 Suitable Functions and Sets

Let $f \colon \mathcal{D}_f \to \mathcal{R}_f$ be a function. For a transcript τ, we define $C_{f,\tau}$ to be the union of the set of subverted points for the function f and the points fixed by τ.

Definition 7. $C_{f,\tau} = \{x \mid x \in \mathcal{D}(\tau) \vee \tilde{f}(x) \neq f(x)\}$.

By the definition of ϵ-crooked,

$$\frac{|C_{f,\tau}|}{|\mathcal{D}_f|} \leq \epsilon_\tau := \epsilon + \frac{|\tau|}{|\mathcal{D}_f|}.$$

At the beginning of the second stage of the crooked-indifferentiability game, the transcript contains the interaction of the q_1 many queries made by the implementer. We define

$$\epsilon_1 = \epsilon + \frac{q_1}{2^n}.$$

Let τ be a (partial) transcript. Recall, we say a function g agrees on a transcript τ when the transcript holds for the function g.

$$\mathsf{F}_{n,n|\tau} \overset{\text{def}}{=} \{g \in \mathsf{F}_{n,n} \mid g \text{ agrees on } \tau\}.$$

3 From Classical Indifferentiability to Crooked-Indifferentiability

In this section, we establish sufficient conditions to lift the classical indifferentiability results to the crooked indifferentiability setting. Let $f \colon \mathcal{D}_f \to \mathcal{R}$ and $\mathcal{F} \colon \mathcal{D} \to \mathcal{R}$ be two random oracles where $\mathcal{D} \supseteq \mathcal{D}_f$. Let C^f be an $\mathcal{F}$-compatible construction. We consider a crooked distinguisher $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$.

```
Game Force-Crook$(C)$
$f \xleftarrow{\$} \mathcal{F}_{n,n}$
$(\tilde{\tau}, \langle \tilde{f} \rangle) \leftarrow \mathcal{A}_1^f$
$M \leftarrow \mathcal{A}_2^{(C^f(\cdot,R),f)}(\tilde{\tau}, R)$
if $C^f(M) \neq C^{\tilde{f}}(M)$
  return 1
else
  return 0
```

Fig. 4. The Force-Crook game

3.1 Force-Crook game

In this section, we introduce the security game Force-Crook. Formally the game is defined in Figure 4. The force-crook advantage of an adversary is defined as

$$\mathbf{Adv}^{\textsf{force-crook}}_{\mathcal{A},C} \overset{\text{def}}{=} \mathbf{Succ}_{\mathcal{A},\textsf{force-crook[C]}}.$$

Given a construction C, we define

$$\mathbf{Insec}^{\textsf{force-crook}}_{C,(q_1,\tilde{q},\epsilon),q_2} \overset{\text{def}}{=} \max_{\mathcal{A}} \mathbf{Adv}^{\textsf{force-crook}}_{\mathcal{A},C}.$$

where the maximum is taken over all $((q_1, \tilde{q}, \epsilon), q_2)$-crooked-distinguishers.

3.2 Achieving Crooked-Indifferentiability

Our main technique to prove the security of sponge and prefix-free Merkle-Damgård constructions results from Theorem 1. The idea is depicted in Figure 5. Suppose C is indifferentiable from $\mathcal{F}$ (the advantage of distinguishing middle and rightmost worlds is small). If the Force-Crook advantage is small, then the advantage of distinguishing between the leftmost and the middle-world is small. Then the classical simulator S successfully acts as the simulator in the real world of the crooked setting.

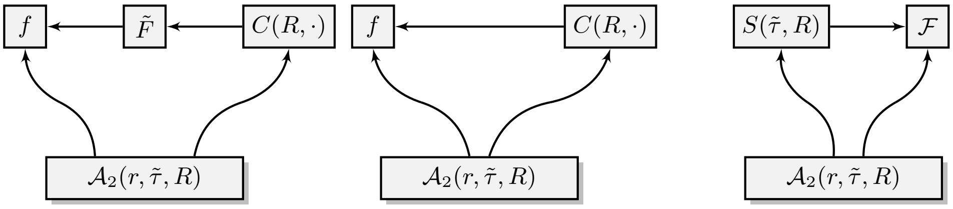
f
$\tilde{F}$
$C(R,\cdot)$
f
$C(R,\cdot)$
$S(\tilde{\tau},R)$
$\mathcal{F}$
$\mathcal{A}_2(r,\tilde{\tau},R)$
$\mathcal{A}_2(r,\tilde{\tau},R)$
$\mathcal{A}_2(r,\tilde{\tau},R)$

Fig. 5. The hybrid: The leftmost picture is the real world of the crooked setting. The middle picture is the real world in the classical setting. The rightmost picture is the ideal world in the classical setting.

Theorem 1. *Let $C^f : \mathcal{D} \to \mathcal{R}$ be a hash function built on primitive $f : \mathcal{D}_f \to \mathcal{R}$. Let C^f be $((q_P, q_C, q_{\text{sim}}), \delta_i)$-indifferentiable from a random oracle $\mathcal{F}$. C^f is $((q_1, \tilde{q}),\ (q_2, q_{\text{sim}}),\ \epsilon, \delta_c)$-crooked-indifferentiable from $\mathcal{F}$ where*

$$\delta_c \leq \delta_i + \mathbf{Insec}^{\textsf{force-crook}}_{C,(q_1,\tilde{q},\epsilon),q_2}$$

and $q_1 + q_2 \leq q_P$.

Proof. From the definitions and using triangle inequality, we get

$$\delta_c \leq \delta_i + \Delta_{\mathcal{A}_2(r,\tilde{\tau},R)}\big((f, C^{\tilde{f}}(R,\cdot)) \ ; \ (f, C^{f}(R,\cdot))\big).$$

In order to prove the theorem, we need to show

$$\Delta_{\mathcal{A}_2(r,\tilde{\tau},R)}\big((f, C^{\tilde{f}}(R,\cdot)) \ ; \ (f, C^{f}(R,\cdot))\big) \leq \mathbf{Insec}^{\textsf{force-crook}}_{C,(q_1,\tilde{q}),(q_2,q_s)}.$$

Let $\textsc{Bad}$ denote the event $\mathcal{A}_2(r,\tilde{\tau},R)$ makes a query to $C^{\tilde{f}}$ (or C^f) oracle with input M such that

$$C^f(R,M) \neq C^{\tilde{f}}(R,M).$$

Now unless $\textsc{Bad}$ is set, the outputs of the oracles in both the world $(f, C^{\tilde{f}}(R,\cdot))$ and $(f, C^{f}(R,\cdot))$ are exactly the same. Thus we get

$$\Pr[\mathcal{A}_2^{(f,C^{\tilde{f}}(R,\cdot))}(r,\tilde{\tau},R) = 1 \cap \neg\textsc{Bad}] = \Pr[\mathcal{A}_2^{(f,C^{f}(R,\cdot))}(r,\tilde{\tau},R) = 1 \cap \neg\textsc{Bad}]. \tag{2}$$

We derive, using Definition 2, triangle inequality, and Equation 2

$$\begin{aligned}
&\Delta_{\mathcal{A}_2(r,\tilde{\tau},R)}\big((f, C^{\tilde{f}}(R,\cdot)) \ ; \ (f, C^{f}(R,\cdot))\big)\\
&= \left|\Pr[\mathcal{A}_2^{(f,C^{\tilde{f}}(R,\cdot))}(r,\tilde{\tau},R) = 1] - \Pr[\mathcal{A}_2^{(f,C^{f}(R,\cdot))}(r,\tilde{\tau},R) = 1]\right|\\
&\leq \left|\Pr[\mathcal{A}_2^{(f,C^{\tilde{f}}(R,\cdot))}(r,\tilde{\tau},R) = 1 \cap \textsc{Bad}] - \Pr[\mathcal{A}_2^{(f,C^{f}(R,\cdot))}(r,\tilde{\tau},R) = 1 \cap \textsc{Bad}]\right| +\\
&\qquad \left|\Pr[\mathcal{A}_2^{(f,C^{\tilde{f}}(R,\cdot))}(r,\tilde{\tau},R) = 1 \cap \neg\textsc{Bad}] - \Pr[\mathcal{A}_2^{(f,C^{f}(R,\cdot))}(r,\tilde{\tau},R) = 1 \cap \neg\textsc{Bad}]\right|\\
&= \left|\Pr[\mathcal{A}_2^{(f,C^{\tilde{f}}(R,\cdot))}(r,\tilde{\tau},R) = 1 \cap \textsc{Bad}] - \Pr[\mathcal{A}_2^{(f,C^{f}(R,\cdot))}(r,\tilde{\tau},R) = 1 \cap \textsc{Bad}]\right|\\
&\leq \Pr[\textsc{Bad}]\,.
\end{aligned}$$

The last inequality follows as both $\Pr[\mathcal{A}_2^{(f,C^{\tilde{f}}(R,\cdot))}(r,\tilde{\tau},R) = 1 \cap \textsc{Bad}]$ and $\Pr[\mathcal{A}_2^{(f,C^{f}(R,\cdot))}(r,\tilde{\tau},R) = 1 \cap \textsc{Bad}]$ are numbers between 0 and $\Pr[\textsc{Bad}]$. Finally, if $\textsc{Bad}$ happens then $\mathcal{A}_2(r,\tilde{\tau},R)$ wins the game $\textsf{Force-Crook}$. Thus

$$\Pr[\textsc{Bad}] \leq \mathbf{Insec}^{\textsf{force-crook}}_{C,(q_1,\tilde{q}),(q_2,q_s)}.$$

The theorem follows. □

4 Crooked-Indifferentiability of Sponge Construction

4.1 Sponge Construction based on Random Functions

In this section, we show that the sponge construction [8] based on an n-to-n-bit random function can be proved crooked-indifferentiable from a random oracle when initialized with a random IV.

Sponge Construction. The details of the parameters of the sponge construction we consider are listed below.

TARGET HASH FUNCTION. The construction implements a FIL-hash function $H : \{0,1\}^\ell \to \{0,1\}^s$.

PRIMITIVES. The underlying primitive of the construction is an n-to-n bit function $f : \{0,1\}^n \to \{0,1\}^n$. In the security proof, f is modelled as a random oracle.

PUBLIC RANDOMNESS. The public randomness is $R \xleftarrow{\$} \{0,1\}^n$.

PADDING. We use the same padding scheme as the original sponge construction, where it is required

```
Procedure Sponge
// random string $R$, Message $m \in \{0,1\}^\ell$
$x = (x_a, x_c) = R$
for $i = 0$ to $\left\lceil \frac{\ell}{r} \right\rceil - 1$ do
    $(x_a, x_c) = f(x_a \oplus m_i, x_c)$
endfor
for $i = 0$ to $\left\lceil \frac{s}{r} \right\rceil - 1$ do
    Append $x_a$ to output
    $(x_a, x_c) = f(x_a, x_c)$
endfor
```

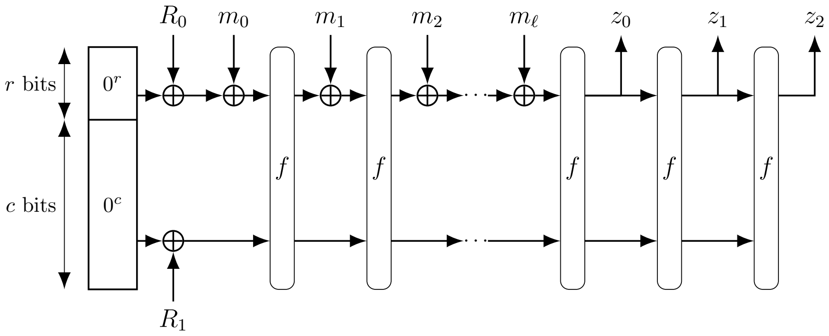

Fig. 6. Crooked-Indifferentiable Sponge Construction

that the last message block is non-zero.

CONSTRUCTION. The chaining value of the sponge construction is divided into two parts, *rate* (length denoted by r) and *capacity* (length denoted by c). The message is divided into r-bit blocks. The construction works in two phases, absorbing and squeezing. In one round of the absorbing phase, one r-bit message block is xored with the rate part of the chaining value. The function f is then applied to the result (of the xor) to get the chaining value of the next round. The construction enters the squeezing phase once all the input message blocks are processed. At each round, the rate part of the chaining value is stored as the output block, followed by the application of f on the whole chaining value. The algorithm stops once we have s bits of output. The construction is described in Figure 6.

q_1	Number of f queries made by the implementor $\mathcal{A}_1$
$\tilde{q}$	Number of f queries made by the subverted implementation $\tilde{f}$
q_2	Total number of queries made by the distinguisher $\mathcal{A}_2$
q_{sim}	Total number of $\mathcal{F}$ queries made by the simulator S
ϵ	Fraction of subverted points under $\tilde{f}$

Fig. 7. Recalling the notations

Our main result in this section is Theorem 2. We recall the notations in Figure 7.

Theorem 2. *Let $f : \{0,1\}^n \to \{0,1\}^n$ be a random function and $C^f : \{0,1\}^\ell \to \{0,1\}^s$ be the sponge construction. Let r be the rate part, and $c = n - r$ be the capacity part of the chain. Then there exists a simulator S such that for all $((q_1, \tilde{q}, \epsilon), q_2)$-crooked distinguishers $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, it holds that*

$$\mathbf{Adv}^{\mathsf{crooked\text{-}indiff}}_{\mathcal{A},(C,f)} \leq \mathcal{O}\left(2^r \times \sigma \times \left(\epsilon_1^{\frac{1}{2}} + \frac{\tilde{q}}{2^{\frac{n}{4}}} + \epsilon_1^{\frac{1}{2}} + \frac{\sigma}{2^n} + \frac{\sigma}{2^{\frac{n}{2}}}\right)\right)$$

where $\epsilon_1 = \epsilon + \frac{q_1}{2^n}$, σ is the total number of blocks in the queries made by $\mathcal{A}_2$. The simulator makes $\mathcal{O}(\sigma)$ queries.

The rest of the section is dedicated to proving Theorem 2. First, we recall the result of Bertoni, Daemen, Peeters, and Van Assche [9] to find the classical Indifferentiability bound of the sponge construction. Then we shall bound the $\mathbf{Insec}^{\mathsf{force\text{-}crook}}_{C,(q_1,\tilde{q}),(q_2,q_s)}$, the advantage of any distinguisher against our construction in the Force-Crook game. Finally, using Theorem 1, we shall get Theorem 2.

Classical Indifferentiability of Sponge with Random Function. We recall the classical indifferentiability result of sponge mode from [9] in our notations and parameters.

Theorem 3 (Theorem 1 in [9]). *Let $f : \{0,1\}^n \to \{0,1\}^n$ be a random function. The sponge construction instantiating $C^f : \{0,1\}^\ell \to \{0,1\}^s$ is (q, q_{sim}, δ_i)-indifferentiable from a random oracle for $q_{sim} = \mathcal{O}(\sigma)$ and $\delta_i = \mathcal{O}(\frac{\sigma^2}{2^c})$ where σ is the total number of queries made by the distinguisher.*

We note that in [9], the above theorem is proved to hold for any fixed IV. Thus we can conclude that the theorem holds for a randomly chosen and then fixed IV, as required in our case.

4.2 Bounding Probability of Winning $\mathsf{Force\text{-}Crook}$: Sponge on Random Functions

Now we bound $\mathbf{Insec}^{\mathsf{force\text{-}crook}}_{C,(q_1,\tilde{q}),(q_2,q_s)}$. We shall prove the following lemma which summarizes the main findings of this section. We recall the notations in Figure 7.

Lemma 2. *Let C be the sponge construction with randomized IV. Let r be the rate part, and $c = n - r$ be the capacity part of the chain. It holds that*

$$\mathbf{Insec}^{\mathsf{force\text{-}crook}}_{C,(q_1,\tilde{q}),(q_2,q_s)} \leq \mathcal{O}\left(2^r \times \sigma \times \left(\epsilon_1^{\frac{1}{2}} + \frac{\tilde{q}}{2^{\frac{n}{4}}} + \epsilon_1^{\frac{1}{2}} + \frac{\sigma}{2^n} + \frac{\sigma}{2^{\frac{n}{2}}}\right)\right)$$

where $\sigma = q_2(\ell + s) + q_S$.

The Setup of Bounding Adversary's Advantage. The main idea of our proof is to bound the probability that the adversary can produce a message such that a chaining query is subverted. We need the following definition.

Definition 8 (Robust Point). *A point $x \in \{0,1\}^n$ is said to be a (r, ϵ_1)-robust point with respect to a transcript τ, if*

1. *$x \notin \mathcal{D}(\tau)$.*
2. *Define $y_\zeta = f(x) \oplus \zeta 0^{n-r}$ for $\zeta \in \{0,1\}^r$. It holds that*

$$\Pr_{f \leftarrow \mathsf{F}_{n,n|\tau}}\left[\bigvee_{\zeta \in \{0,1\}^r} y_\zeta \in C_{f,\tau}\right] \leq 2^r \left(\epsilon_1^{\frac{1}{2}} + \frac{|\tau|}{2^n} + \frac{|\tau|}{2^{\frac{n}{2}}}\right).$$

Popular Points. Consider a point $x \in \mathcal{D}_f \setminus \mathcal{D}(\tau)$. x is called favourite of y with respect to τ if

$$\Pr_{f \overset{\$}{\leftarrow} \mathsf{F}_{n,n|\tau}}[y \twoheadrightarrow_f x] \geq \frac{1}{2^{\frac{n}{2}}}.$$

Definition 9. *x is* popular *with respect to τ if*

$$\Pr_y[x \textit{ is favourite of } y] > \frac{1}{2^{\frac{n}{4}}}.$$

Recall that the subversion algorithm $\tilde{f}$ makes at most $\tilde{q}$ many queries; for all $y \in \mathcal{D}_f$, it holds that $|\tilde{f}(y)| \leq \tilde{q}$. Using an averaging argument, we get the following lemma.

Lemma 3. *For all transcript τ, it holds that the number of popular points is at most $\tilde{q}2^{\frac{3n}{4}}$.*

Definition 10 (Good Point). *A point x is (r, ϵ_1)-good with respect to τ if it is (r, ϵ_1) robust and not popular with respect to τ.*

The following lemma is a corollary of Lemma 3 and the definition of the ϵ-crooked implementor. It says a random point is good with high probability.

Lemma 4. *Let τ be a transcript. It holds that*

$$\Pr_{x \overset{\$}{\leftarrow} \mathcal{D}_f}[x \textit{ is not } (r, \epsilon_1) \textit{ good with respect to } \tau] \leq \epsilon_1^{\frac{1}{2}} + \frac{|\tau|}{2^n} + \frac{\tilde{q}}{2^{\frac{n}{4}}}.$$

Proof. Define $y_\zeta = f(x) \oplus \zeta 0^{n-r}$ for $\zeta \in \{0,1\}^r$. From the definition of ϵ crooked implementor,

$$\Pr_{x \xleftarrow{\$} \mathcal{D}_f, f \leftarrow \mathsf{F}_{n,n|\tau}} \left[\bigvee_{\zeta \in \{0,1\}^r} y_\zeta \in C_{f,\tau} \right] \leq 2^r \epsilon_1.$$

By an averaging argument,

$$\Pr_{x \xleftarrow{\$} \mathcal{D}_f} \left[\Pr_{f \leftarrow \mathsf{F}_{n,n|\tau}} \left[\bigvee_{\zeta \in \{0,1\}^r} y_\zeta \in C_{f,\tau} \right] > 2^r \epsilon_1^{\frac{1}{2}} \right] \leq \epsilon_1^{\frac{1}{2}}.$$

We derive,

$$\begin{aligned}
&\Pr_{x \xleftarrow{\$} \mathcal{D}_f} [x \text{ is not } (r, \epsilon_1) \text{ good with respect to } \tau] \\
&= \Pr_{x \xleftarrow{\$} \mathcal{D}_f} [x \text{ is not } (r, \epsilon_1) \text{ robust with respect to } \tau] + \Pr_{x \xleftarrow{\$} \mathcal{D}_f} [x \text{ is popular with respect to } \tau] \\
&\leq \epsilon_1^{\frac{1}{2}} + \frac{|\tau|}{2^n} + \frac{\tilde{q}}{2^{\frac{n}{4}}}.
\end{aligned}$$

Next, we wish to ensure that all possible chaining values generated from a good point also become good points. We need the following definition.

Definition 11. *Let x be an (r, ϵ_1)-good point with respect to τ. We say y is eligible for (τ, x) if*

1. *y is an (r, ϵ_1)-good point with respect to τ.*
2. *for $\tau' = \tau \cup (x, y)$, it holds that y is (r, ϵ_1)-good point with respect to τ'.*

Now we are ready to state our main tool.

Proposition 4 *Let x be ϵ_1-good point with respect to τ.*

$$\Pr_{y \xleftarrow{\$} \mathcal{D}_f} [y \textit{ is not eligible with respect to } (\tau, x)] \leq \epsilon_1^{\frac{1}{2}} + \frac{|\tau|}{2^n} + \frac{\tilde{q} + 2^r}{2^{\frac{n}{4}}}.$$

Proof. The idea of the proof is to show that if we sample a point uniformly at random from $\mathcal{D}_f$, then by Lemma 4, with high probability, the point is (r, ϵ_1)-good with respect to τ. That means

$$\Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}} \left[\bigvee_{b' \in \{0,1\}^r} (f(y) \oplus b' 0^{n-r} \in C_{f,\tau'}) \right] \leq 2^r \left(\epsilon_1^{\frac{1}{2}} + \frac{|\tau|}{2^n} + \frac{|\tau|}{2^{\frac{n}{2}}} \right).$$

Now, if it also holds that $(f(y) \oplus b'0^{n-r}) \not\twoheadrightarrow x$ for any $b' \in \{0,1\}^r$, the point y will remain ϵ_1-good with respect to $\tau \cup (x, y)$. To prove it formally, we consider the following events.

1. **y-is-bad**: y is not (r, ϵ_1)-good with respect to τ.
2. **x-is-queried**: $\Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}} [(f(y) \oplus b'0^{n-r}) \twoheadrightarrow x] \geq \frac{1}{2^{\frac{n}{2}}}$ for some $b' \in \{0,1\}^r$.

The following lemma (to be proved in Section 4.3) says that if the above two events do not occur, then $f(y)$ is an (r, ϵ_1)-good point with respect to τ'.

Lemma 5. *Suppose y is such that the event* $\neg$**y-is-bad** $\bigwedge \neg$**x-is-queried** *holds. Then it holds that y is (r, ϵ_1)-good with respect to $\tau' = \tau \cup (x, y)$.*

$$\Pr_{f \xleftarrow{\$} \Gamma_{\tau'}} \left[\bigvee_{b' \in \{0,1\}^r} (f(y) \oplus b' 0^{n-r} \in C_{f,\tau'}) \right] \leq 2^r \left(\epsilon_1^{\frac{1}{2}} + \frac{|\tau| + 1}{2^n} + \frac{|\tau| + 1}{2^{\frac{n}{2}}} \right).$$

Thus to prove Proposition 4, we need to bound the probability of the events **y-is-bad** and **x-is-queried**. By Lemma 4,

$$\Pr_{y \xleftarrow{\$} \mathcal{D}_f}[\textbf{y-is-bad}] \leq \epsilon_1^{\frac{1}{2}} + \frac{|\tau|}{2^n} + \frac{\tilde{q}}{2^{\frac{n}{4}}}.$$

Finally, by the definition of popular points,

$$\Pr_{y \xleftarrow{\$} \mathcal{D}_f}[\textbf{x-is-queried}] = 2^r \Pr_{z \xleftarrow{\$} \mathcal{D}_f}[x \text{ is favourite of } z] \leq \frac{2^r}{2^{\frac{n}{4}}}.$$

This finishes the proof of Proposition 4. □

Bounding Probability of Winning Force-Crook. We are ready to bound the success probability of any adversary in the Force-Crook game against the sponge construction when the underlying primitive is a random function $f : \{0,1\}^n \to \{0,1\}^n$. Specifically, we shall show that the adversary can not force a crooked chaining input for any query made by C.

Bad events. Recall that the adversary makes at most q_2 many queries to the oracle C^f. Each such query leads to $\ell + s$ many calls (referred to as chaining queries) to f made by C. We consider these chaining queries to be a sequence of $\sigma = q_2(\ell + s)$ many queries. By saying i^{th} query, we denote the i^{th} chaining query from this sequence. We consider the following bad events. The first bad event (**CrookedFirstInput**) occurs if, for any message, the *first* chaining value is crooked. We set the second bad event (**BadChain**) if for some message queried by the distinguisher, we get a chaining value that is not (r, ϵ_1)-good as defined in Definition 10.

1. **CrookedFirstInput**. We say a bad event occurs if for the initial random R, for some $m_0 \in \{0,1\}^r$,
$$\Pr_{f \xleftarrow{\$} \varGamma_{\tilde{\tau}}}[R \oplus m_0 0^{n-r} \in C_{f,\tilde{\tau}}] \geq 2^r \left(\epsilon_1^{\frac{1}{2}} + \frac{q_1}{2^n}\right).$$

2. **BadChain**. We say the i^{th} chaining query x_i raises bad event (denoted by $\textbf{BadChain}_i$) if x_i is not (r, ϵ_1)-good with respect to the (up to that query) transcript τ. We define $\textbf{BadChain} \stackrel{\text{def}}{=} \cup_{i=1}^{\sigma} \textbf{BadChain}_i$.

Note that, for standard indifferentiability proofs, we usually consider a bad event when a chaining query input collides with some unchained query (made by the adversary to the oracle f) input. In our setting, such unchained queries are part of the transcript, and the definition of good points ensures that the chaining query does not result in such collisions.

Bounding Probabilities of Bad Events. First, we bound the probability of **CrookedFirstInput**. From the definition of ϵ-subversion, when the probabilities are taken over $f \xleftarrow{\$} \varGamma_{\tilde{\tau}}$ and $x \xleftarrow{\$} \mathcal{D}_f$

$$\Pr[x \in C_{f,\tilde{\tau}}] \leq \epsilon_1.$$

By an averaging argument, we get that

$$\Pr_{R \xleftarrow{\$} \mathcal{D}_f}\left[\Pr_{f \xleftarrow{\$} \varGamma_{\tilde{\tau}}}[R \in C_{f,\tilde{\tau}}] > \epsilon_1^{\frac{1}{2}}\right] \leq \epsilon_1^{\frac{1}{2}}.$$

Thus we bound

$$\Pr_{R \xleftarrow{\$} \mathcal{D}_f}[\textbf{CrookedFirstInput}] \leq 2^r \epsilon_1^{\frac{1}{2}}. \qquad (3)$$

Next, we bound Pr[**BadChain**]. For this case, we derive

$$\Pr[\mathbf{BadChain}] = \Pr[\mathbf{BadChain}_1] + \sum_{j=2}^{\sigma} \Pr[\mathbf{BadChain}_j \mid \bigwedge_{j'=1}^{j-1} \neg\mathbf{BadChain}_{j'}].$$

We start with bounding Pr[**BadChain**$_1$]. As R is uniformly chosen, from Lemma 4

$$\Pr[\mathbf{BadChain}_1] = \Pr_{R \xleftarrow{\$} \mathcal{D}_f} [R \text{ is not } (r, \epsilon_1) \text{ -good with respect to. } \tilde{\tau}] \le \epsilon_1^{\frac{1}{2}} + \frac{q_1}{2^n} + \frac{\tilde{q}}{2^{\frac{n}{4}}}.$$

Consider the i^{th} chaining query x_i where $i > 1$. Let τ_i denote the transcript up to i^{th} query. We find the chaining query x_k, queried before x_i $(k < i)$ such that

$$x_i = f(x_k) \oplus b0^{n-r} \qquad \text{for some } b \in \{0,1\}^r.$$

Given $\bigwedge_{j'=1}^{i-1} \neg\mathbf{BadChain}_{j'}$, we conclude x_k is (r, ϵ_1)-good. If $f(x_k) \oplus b0^{n-r}$ is not (r, ϵ_1)-good with respect to τ_{k+1}, this means $f(x_k) \oplus b0^{n-r}$ was not eligible with respect to (τ_k, x_k) for some $b \in \{0,1\}^r$. Using Proposition 4,

$$\Pr_{f \xleftarrow{\$} \Gamma_{\tau_k}} \left[\bigvee_{b \in \{0,1\}^r} (f(x_k) \oplus b0^{n-r}) \text{ is not eligible w.r.t. } (\tau_k, x_k) \right] \le 2^r \left(\epsilon_1^{\frac{1}{2}} + \frac{\tilde{q} + 2^r}{2^{\frac{n}{4}}} + \frac{k}{2^n} \right).$$

Thus we get

$$\Pr[\mathbf{BadChain}_j \mid \bigwedge_{j'=1}^{j-1} \neg\mathbf{BadChain}_{j'}] \le 2^r \left(\epsilon_1^{\frac{1}{2}} + \frac{\tilde{q} + 2^r}{2^{\frac{n}{4}}} + \frac{j}{2^n} \right).$$

Taking sum over all j we get

$$\Pr[\mathbf{BadChain}] \le 2^r \left(\sigma\epsilon_1^{\frac{1}{2}} + \frac{\sigma(\tilde{q} + 2^r)}{2^{\frac{n}{4}}} + \frac{\sigma^2}{2^n} \right). \tag{4}$$

Bounding the Force-Crook Advantage. Let W_i denote the event that the input to the i^{th} query is crooked.

$$\Pr[\mathcal{A} \text{ wins the game Force-Crook}] \le \Pr[\mathbf{CrookedFirstInput}] + \Pr[\mathbf{BadChain}] + \sum_{i=1}^{\sigma} \Pr[W_i \mid \neg\mathbf{CrookedFirstInput} \bigwedge \neg\mathbf{BadChain}].$$

As we already have the bound on the probabilities of the bad events, we need to bound

$$\Pr\left[W_i \mid \neg\mathbf{CrookedFirstInput} \bigwedge \neg\mathbf{BadChain}\right].$$

Consider the i^{th} chaining query x_i where $i > 1$. We find the chaining query x_k previous to x_i $(k < i)$. As $\neg$**BadChain** holds, x_k is (r, ϵ')-good with respect to the partial transcript τ_k.

$$\Pr_{f \xleftarrow{\$} \Gamma_{\tau_k}} \left[\bigvee_{b \in \{0,1\}^r} (f(x_k) \oplus b0^{n-r} \in C_{f,\tau_k}) \right] \le 2^r \left(\epsilon_1^{\frac{1}{2}} + \frac{k}{2^n} + \frac{k}{2^{\frac{n}{2}}} \right).$$

This implies

$$\Pr_{f \xleftarrow{\$} \Gamma_{\tau_k}}[x_i \in C_{f,\tau_k}] \le 2^r \left(\epsilon_1^{\frac{1}{2}} + \frac{k}{2^n} + \frac{k}{2^{\frac{n}{2}}}\right) \le 2^r \left(\epsilon_1^{\frac{1}{2}} + \frac{i}{2^n} + \frac{i}{2^{\frac{n}{2}}}\right).$$

As the responses of all the f queries are answered truthfully, for a $f \xleftarrow{\$} \Gamma_{\tau_k}$, $f \cup \tau_k$ is a uniform random element of $\Gamma_{\tilde{\tau}}$. Thus we get

$$\Pr_{f \xleftarrow{\$} \Gamma_{\tilde{\tau}}}[x_i \in C_{f,\tilde{\tau}} \mid \neg\textbf{CrookedFirstInput} \wedge \neg\textbf{BadChain}] \le 2^r \left(\epsilon_1^{\frac{1}{2}} + \frac{i}{2^n} + \frac{i}{2^{\frac{n}{2}}}\right).$$

Taking the sum over all i, we get

$$\sum_{i=1}^{\sigma} \Pr_{f \xleftarrow{\$} \Gamma_{\tilde{\tau}}}[W_i \mid \neg\textbf{CrookedFirstInput} \wedge \neg\textbf{BadChain}] \le \sum_{i=1}^{\sigma} 2^r \left(\epsilon_1^{\frac{1}{2}} + \frac{i}{2^n} + \frac{i}{2^{\frac{n}{2}}}\right) \le 2^r \left(\sigma\epsilon_1^{\frac{1}{2}} + \frac{\sigma^2}{2^n} + \frac{\sigma^2}{2^{\frac{n}{2}}}\right). \tag{5}$$

Finally, adding Inequalities (3),(4), and (5) we get

$$\Pr[\mathcal{A} \text{ wins the game } \mathsf{Force\text{-}Crook}] \le \mathcal{O}\left(2^r \times \sigma \times \left(\epsilon_1^{\frac{1}{2}} + \frac{(\tilde{q} + 2^r)}{2^{\frac{n}{4}}} + \epsilon_1^{\frac{1}{2}} + \frac{\sigma}{2^n} + \frac{\sigma}{2^{\frac{n}{2}}}\right)\right).$$

This finishes the proof of Lemma 2 and thus the proof of Theorem 2. □

4.3 Proof of Lemma 5

Proof. Lemma 5 considers a transcript τ and points $x, y \in \{0,1\}^n$. Suppose y is such that the condition ($\neg$**y-is-bad** $\bigwedge \neg$**x-is-queried**) holds. The condition ($\neg$**y-is-bad**) implies that y is a (r, ϵ_1)-good point with respect to τ. The lemma says that y is a (r, ϵ_1)-good point with respect to $\tau' = \tau \cup (x, y)$. Given the conditions and following Definition 10, we get that y is (r, ϵ_1)-robust with respect to τ and y is not popular . By Definition 8 we have

$$\Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}}\left[\bigvee_{b' \in \{0,1\}^r} (f(y) \oplus b'0^{n-r} \in C_{f,\tau})\right] \le 2^r \left(\epsilon_1^{\frac{1}{2}} + \frac{|\tau|}{2^n} + \frac{|\tau|}{2^{\frac{n}{2}}}\right).$$

Our target is to bound the probability that y is not (r, ϵ_1)-good with respect to τ'. Let $Y_{b'}$ denote $f(y) \oplus b'0^{n-r}$. First, we bound the probability (over $f \xleftarrow{\$} \Gamma_{\tau'}$) that $Y_{b'}$ is not a (r, ϵ_1)-robust point with respect to $\tau' = \tau \cup (x, y)$. We have two cases: *a*) $Y_{b'} = x$ for some $b' \in \{0,1\}^r$, *b*) $Y_{b'} \in C_{f,\tau}$ for some $b' \in \{0,1\}^r$. By union bound

$$\Pr\left[\bigvee_{b' \in \{0,1\}^r} (Y_{b'} \in C_{f,\tau'})\right] \le \Pr\left[\bigvee_{b' \in \{0,1\}^r} (Y_{b'} = x)] + \Pr[\bigvee_{b' \in \{0,1\}^r} (Y_{b'} \in C_{f,\tau})\right]. \tag{6}$$

The term $\Pr[\bigvee_{b' \in \{0,1\}^r}(Y_{b'} = x)]$ is bounded above by $\frac{2^r}{2^n}$. For the second term, we can bound the probability (over $f \xleftarrow{\$} \Gamma_{\tau'}$) as

$$\Pr\left[\bigvee_{b' \in \{0,1\}^r} (Y_{b'} \in C_{f,\tau})\right] \le \Pr\left[\bigvee_{b' \in \{0,1\}^r} (Y_{b'} \twoheadrightarrow_f x)\right] + \tag{7}$$

$$\Pr\left[\left(\bigvee_{b' \in \{0,1\}^r} (Y_{b'} \in C_{f,\tau})\right) \bigwedge \bigwedge_{b' \in \{0,1\}^r} (Y_{b'} \not\twoheadrightarrow_f x)\right]. \tag{8}$$

Bounding $\Pr\left[\bigvee_{b'\in\{0,1\}^r}(Y_{b'} \twoheadrightarrow_f x)\right]$. We first show that the probability that $Y_{b'}$ queries x is the same for all the transcripts irrespective of where the value of $f(x)$ is set. In other words, we shall establish that the probability that $Y_{b'}$ queries x is the same in both transcripts τ and τ'.

$$\begin{aligned}
\Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}}\left[\bigwedge_{b'\in\{0,1\}^r} (Y_{b'} \not\twoheadrightarrow_f x)\right] &= \sum_z \Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}}\left[\bigwedge_{b'\in\{0,1\}^r} (Y_{b'} \not\twoheadrightarrow_f x) \bigwedge f(x) = z\right] \\
&= 2^n \Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}}\left[\bigwedge_{b'\in\{0,1\}^r} (Y_{b'} \not\twoheadrightarrow_f x) \bigwedge f(x) = y\right] \\
&= \Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}}\left[\left(\bigwedge_{b'\in\{0,1\}^r} (Y_{b'} \not\twoheadrightarrow_f x)\right) \mid f(x) = y\right] \\
&= \Pr_{f \xleftarrow{\$} \varGamma_{\tau'}}\left[\bigwedge_{b'\in\{0,1\}^r} (Y_{b'} \not\twoheadrightarrow_f x)\right].
\end{aligned}$$

Now taking the complement

$$\begin{aligned}
\Pr_{f \xleftarrow{\$} \varGamma_{\tau'}}\left[\bigvee_{b'\in\{0,1\}^r} (Y_{b'} \twoheadrightarrow_f x)\right] &= 1 - \Pr_{f \xleftarrow{\$} \varGamma_{\tau'}}\left[\bigwedge_{b'\in\{0,1\}^r} (Y_{b'} \twoheadrightarrow_f x)\right] \\
&= 1 - \Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}}\left[\bigwedge_{b'\in\{0,1\}^r} (Y_{b'} \twoheadrightarrow_f x)\right] \\
&= \Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}}\left[\bigvee_{b'\in\{0,1\}^r} (Y_{b'} \twoheadrightarrow_f x)\right].
\end{aligned}$$

Bounding $\Pr\left[\left(\bigvee_{b'\in\{0,1\}^r}(Y_{b'} \in C_{f,\tau})\right) \wedge \bigwedge_{b'\in\{0,1\}^r}(Y_{b'} \not\twoheadrightarrow_f x)\right]$**.** Similar to the first case, we show the probability is identical for both the transcripts.

$$\begin{aligned}
&\Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}}\left[\left(\bigvee_{b'\in\{0,1\}^r} (Y_{b'} \in C_{f,\tau})\right) \bigwedge \bigwedge_{b'\in\{0,1\}^r} (Y_{b'} \not\twoheadrightarrow_f x)\right] \\
&= \sum_z \Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}}\left[\left(\bigvee_{b'\in\{0,1\}^r} (Y_{b'} \in C_{f,\tau})\right) \bigwedge \bigwedge_{b'\in\{0,1\}^r} (Y_{b'} \not\twoheadrightarrow_f x) \bigwedge f(x) = z\right] \\
&= 2^n \Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}}\left[\left(\bigvee_{b'\in\{0,1\}^r} (Y_{b'} \in C_{f,\tau})\right) \bigwedge \bigwedge_{b'\in\{0,1\}^r} (Y_{b'} \not\twoheadrightarrow_f x) \bigwedge f(x) = y\right] \\
&= \Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}}\left[\left(\bigvee_{b'\in\{0,1\}^r} (Y_{b'} \in C_{f,\tau})\right) \bigwedge \bigwedge_{b'\in\{0,1\}^r} (Y_{b'} \not\twoheadrightarrow_f x) \mid f(x) = y\right] \\
&= \Pr_{f \xleftarrow{\$} \varGamma_{\tau'}}\left[\left(\bigvee_{b'\in\{0,1\}^r} (Y_{b'} \in C_{f,\tau})\right) \bigwedge \bigwedge_{b'\in\{0,1\}^r} (Y_{b'} \not\twoheadrightarrow_f x)\right].
\end{aligned}$$

The Final Derivation. Now we are ready to bound $\Pr_{f \xleftarrow{\$} \varGamma_{\tau'}}[\bigvee_{b'\in\{0,1\}^r}(Y_{b'} \in C_{f,\tau'})]$. In the following derivation, we use inequality 6 in the first step, inequality 7 in the second step, and the above two cases in the third step.

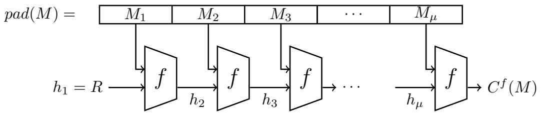

Fig. 8. Merkle-Damgård mode of operation with random IV

$$
\begin{aligned}
&\Pr_{f \xleftarrow{\$} \Gamma_{\tau'}}\left[\bigvee_{b' \in \{0,1\}^r} (Y_{b'} \in C_{f,\tau'})\right] \\
&\leq \Pr_{f \xleftarrow{\$} \Gamma_{\tau'}}\left[\bigvee_{b' \in \{0,1\}^r} (Y_{b'} = x)\right] + \Pr_{f \xleftarrow{\$} \Gamma_{\tau'}}\left[\bigvee_{b' \in \{0,1\}^r} (Y_{b'} \in C_{f,\tau})\right] \\
&\leq \frac{2^r}{2^n} + \Pr_{f \xleftarrow{\$} \Gamma_{\tau'}}\left[\bigvee_{b' \in \{0,1\}^r} (Y_{b'} \twoheadrightarrow_f x)\right] + \Pr_{f \xleftarrow{\$} \Gamma_{\tau'}}\left[\left(\bigvee_{b' \in \{0,1\}^r} (Y_{b'} \in C_{f,\tau})\right) \bigwedge \bigwedge_{b' \in \{0,1\}^r} (Y_{b'} \not\twoheadrightarrow_f x)\right] \\
&= \frac{2^r}{2^n} + \Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}}\left[\bigvee_{b' \in \{0,1\}^r} (Y_{b'} \twoheadrightarrow_f x)\right] + \Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}}\left[\left(\bigvee_{b' \in \{0,1\}^r} (Y_{b'} \in C_{f,\tau})\right) \bigwedge \bigwedge_{b' \in \{0,1\}^r} (Y_{b'} \not\twoheadrightarrow_f x)\right] \\
&\leq \frac{2^r}{2^n} + \Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}}\left[\bigvee_{b' \in \{0,1\}^r} (Y_{b'} \twoheadrightarrow_f x)\right] + \Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}}\left[\left(\bigvee_{b' \in \{0,1\}^r} (Y_{b'} \in C_{f,\tau})\right)\right] \\
&\leq \frac{2^r}{2^n} + \frac{2^r}{2^{\frac{n}{2}}} + 2^r\left(\epsilon_1^{\frac{1}{2}} + \frac{|\tau|}{2^n} + \frac{|\tau|}{2^{\frac{n}{2}}}\right).
\end{aligned}
$$

In the last line we used, as the event (¬**y-is-bad**) holds,

$$\Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}}\left[\left(\bigvee_{b' \in \{0,1\}^r} (Y_{b'} \in C_{f,\tau})\right)\right] \leq 2^r\left(\epsilon_1^{\frac{1}{2}} + \frac{|\tau|}{2^n} + \frac{|\tau|}{2^{\frac{n}{2}}}\right).$$

and as the event (¬**x-is-queried**) holds

$$\Pr_{f \xleftarrow{\$} \mathsf{F}_{n,n|\tau}}\left[\bigvee_{b' \in \{0,1\}^r} (Y_{b'} \twoheadrightarrow_f x)\right] \leq \frac{2^r}{2^{\frac{n}{2}}}.$$

□

5 Crooked-Indifferentiability of Merkle-Damgård

In this section, we show that the classical Merkle-Damgård construction using $n+1$-to-n-bit compression function f and instantiated with a random initialization vector is crooked-indifferentiable from a random oracle.

Merkle-Damgård Construction. The details of the parameters of Merkle-Damgård construction are listed below. The construction is shown in Figure 8

<u>Target Hash Function.</u> The construction implements a hash function $H : \{0,1\}^\mu \to \{0,1\}^n$.

<u>Primitives.</u> The underlying primitive of the construction is an $n+1$-to-n bit function $f : \{0,1\}^{n+1} \to \{0,1\}^n$.

<u>Public Randomness.</u> The public randomness is $R \xleftarrow{\$} \{0,1\}^n$.

Message Preprocessing. The indifferentiability of Merkle-Damgård requires the message space to be prefix-free. We assume the same. Note if we consider the fixed input length hash function, we do not need any prefix-free padding. The input message $M \in \{0,1\}^{\mu}$ is parsed as bits $M_1 M_2 \dots M_{\mu}$. Our main result in this section is Theorem 5.

Theorem 5. *Let $f : \{0,1\}^{n+1} \to \{0,1\}^n$ be a random function and $C^f : \{0,1\}^{\mu} \to \{0,1\}^n$ be the Merkle-Damgård construction. There exists a simulator S such that for all $((q_1, \tilde{q}, \epsilon), q_2)$-crooked distinguisher $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*

$$\mathbf{Adv}^{\mathsf{crooked\text{-}indiff}}_{\mathcal{A},(C,f)} \leq \mathcal{O}\left(\sigma \times \left(\epsilon_1^{\frac{1}{2}} + \frac{\tilde{q}}{2^{\frac{n}{4}}} + \epsilon_1^{\frac{1}{2}} + \frac{\sigma}{2^n} + \frac{\sigma}{2^{\frac{n}{2}}}\right)\right)$$

where $\epsilon_1 = \epsilon + \frac{q_1}{2^n}$ q_2 is the total number of construction queries made by $\mathcal{A}_2$ and σ is the total number of blocks in the queries made by $\mathcal{A}_2$.

The Theorem follows from Theorem 6 and Lemma 6.

Classical Indifferentiability of Merkle-Damgård Construction. We recall the classical indifferentiability result of Merkle-Damgård mode from [18] in our notations.

Theorem 6 (Theorem 3.1 in [18]). *Prefix-free Merkle-Damgård mode instantiating $C^f : \{0,1\}^{\mu} \to \{0,1\}^n$ is (q_2, q_{sim}, δ_i)-indifferentiable from a random oracle for $q_{sim} = \mathcal{O}(\sigma^2)$ and $\delta_i = \mathcal{O}(\frac{\sigma^2}{2^n})$ where σ is the total number of blocks in the queries made by the distinguisher.*

Bounding Probability of Winning Force-Crook**.**

Lemma 6. *Let C be the Merkle-Damgård construction considered in this section.*

$$\mathbf{Insec}^{\mathsf{force-crook}}_{C,(q_1,\tilde{q}),(q_2,q_s)} \leq \mathcal{O}\left(\sigma \times \left(\epsilon_1^{\frac{1}{2}} + \frac{\tilde{q}}{2^{\frac{n}{4}}} + \epsilon_1^{\frac{1}{2}} + \frac{\sigma}{2^n} + \frac{\sigma}{2^{\frac{n}{2}}}\right)\right)$$

where $\sigma = q_2\mu + q_S$.

The proof of the lemma works exactly as the proof of Lemma 2. The only difference is in the parameters of the definitions. We skip the proof.

6 Concluding Discussion

We wish to finish the paper with some discussion on the possibility and challenges of extending our proof to sponge construction with permutations. Finally, we present some research directions we find interesting.

6.1 Sponge Construction Based on Permutation

The reader may note that the sponge construction in practice is based on a fixed *permutation* where the adversary is allowed to make inverse queries. We attempted to extend our proof for the permutations as well but could not solve one key issue. One main step (Proposition 4) in our proof was to show that a good point y with respect to a partial transcript τ remains a good point if another good point x is mapped to y. In order to prove that we argued that the queries of $\tilde{f}(y)$ and $\tilde{f}(f(y))$ are independent from the preimage of y. Thus we could include a good point and extend the transcript without invoking bad.

This argument does not hold when f is a permutation. In that case $\tilde{f}$ can indeed make f^{-1} queries. Extending the transcript with good points and simultaneously handling inverse queries seem to require a different technique. One could try adding additional ingredients like xoring independent random strings in each iteration. But that would increase the number of random strings to be linear with the message length, and the resulting construction would not be practical.

6.2 Conclusion and Future Research Directions

Subversion Resistance of hash function is an important security property when used to replace random oracles in the kleptographic setting. This work is the first to analyze the security of practically used hashing modes in the crooked indifferentiability framework. Our techniques show how to prove crooked indifferentiability when the underlying primitive is modelled as a random function. The first natural research problem would be to consider the crooked indifferentiability of sponge construction in the random permutation model. It would also be interesting to consider proving crooked indifferentiability of the ideal cipher constructions like the Feistel Network. Finally, extending crooked indifferentiability to the multi-stage setting like reset indifferentiability would also be interesting.

Acknowledgements. We sincerely thank the reviewers of this and the previous versions of the work for their insightful comments on the content and the presentation of the paper. Their inputs significantly improved the work. Part of this work was done when RB was in NISER, India. RB is supported by EPSRC EP/R007128/1. For the purpose of open access, the authors has applied a Creative Commons Attribution (CC BY) license to any accepted manuscript version arising.

References

1. Andreeva, E., Mennink, B., Preneel, B.: On the indifferentiability of the Grøstl hash function. In: Garay, J.A., Prisco, R.D. (eds.) SCN 10. LNCS, vol. 6280, pp. 88–105. Springer, Heidelberg (Sep 2010). `https://doi.org/10.1007/978-3-642-15317-4_7` 1
2. Ateniese, G., Francati, D., Magri, B., Venturi, D.: Public immunization against complete subversion without random oracles. In: Deng, R.H., Gauthier-Umaña, V., Ochoa, M., Yung, M. (eds.) ACNS 19. LNCS, vol. 11464, pp. 465–485. Springer, Heidelberg (Jun 2019). `https://doi.org/10.1007/978-3-030-21568-2_23` 1, 1
3. Ateniese, G., Kiayias, A., Magri, B., Tselekounis, Y., Venturi, D.: Secure outsourcing of cryptographic circuits manufacturing. In: Provable Security - 12th International Conference, ProvSec 2018, Jeju, South Korea, October 25-28, 2018, Proceedings. pp. 75–93 (2018), `https://doi.org/10.1007/978-3-030-01446-9_5` 1
4. Ateniese, G., Magri, B., Venturi, D.: Subversion-resilient signatures: Definitions, constructions and applications. Theor. Comput. Sci. **820**, 91–122 (2020), `https://doi.org/10.1016/j.tcs.2020.03.021` 1, 1
5. Bellare, M., Hoang, V.T.: Resisting randomness subversion: Fast deterministic and hedged public-key encryption in the standard model. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part II. LNCS, vol. 9057, pp. 627–656. Springer, Heidelberg (Apr 2015). `https://doi.org/10.1007/978-3-662-46803-6_21` 1
6. Bellare, M., Paterson, K.G., Rogaway, P.: Security of symmetric encryption against mass surveillance. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 1–19. Springer, Heidelberg (Aug 2014). `https://doi.org/10.1007/978-3-662-44371-2_1` 1
7. Bellare, M., Rogaway, P.: The security of triple encryption and a framework for code-based game-playing proofs. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 409–426. Springer, Heidelberg (May / Jun 2006). `https://doi.org/10.1007/11761679_25` 2
8. Bertoni, G., Daemen, J., Peeters, M., Assche, G.: Sponge functions. ECRYPT Hash Workshop 2007 (01 2007) 4.1
9. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: On the indifferentiability of the sponge construction. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 181–197. Springer, Heidelberg (Apr 2008). `https://doi.org/10.1007/978-3-540-78967-3_11` 1, 1.1, 4.1, 3, 4.1
10. Bhattacharyya, R., Mandal, A.: On the indifferentiability of fugue and luffa. In: Applied Cryptography and Network Security - 9th International Conference, ACNS 2011, Nerja, Spain, June 7-10, 2011. Proceedings. pp. 479–497 (2011), `https://doi.org/10.1007/978-3-642-21554-4_28` 1
11. Bhattacharyya, R., Mandal, A., Nandi, M.: Indifferentiability characterization of hash functions and optimal bounds of popular domain extensions. In: Progress in Cryptology - INDOCRYPT 2009, 10th International Conference on Cryptology in India, New Delhi, India, December 13-16, 2009. Proceedings. pp. 199–218 (2009), `https://doi.org/10.1007/978-3-642-10628-6_14` 1
12. Bhattacharyya, R., Mandal, A., Nandi, M.: Security analysis of the mode of JH hash function. In: Hong, S., Iwata, T. (eds.) Fast Software Encryption, 17th International Workshop, FSE 2010,. Lecture Notes in Computer Science, vol. 6147, pp. 168–191. Springer (2010) 1
13. Bhattacharyya, R., Nandi, M., Raychaudhuri, A.: Crooked indifferentiability of enveloped XOR revisited. In: Progress in Cryptology - INDOCRYPT 2021 - 22nd International Conference on Cryptology in India, Jaipur, India, December 12-15, 2021, Proceedings. pp. 73–92 (2021), `https://doi.org/10.1007/978-3-030-92518-5_4` 1
14. Chang, D., Nandi, M.: Improved indifferentiability security analysis of chopMD hash function. In: Nyberg, K. (ed.) FSE 2008. LNCS, vol. 5086, pp. 429–443. Springer, Heidelberg (Feb 2008). `https://doi.org/10.1007/978-3-540-71039-4_27` 1

15. Checkoway, S., Niederhagen, R., Everspaugh, A., Green, M., Lange, T., Ristenpart, T., Bernstein, D.J., Maskiewicz, J., Shacham, H., Fredrikson, M.: On the practical exploitability of dual EC in TLS implementations. In: Proceedings of the 23rd USENIX Security Symposium, San Diego, CA, USA, August 20-22, 2014. pp. 319–335 (2014), https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/checkoway 1
16. Chow, S.S.M., Russell, A., Tang, Q., Yung, M., Zhao, Y., Zhou, H.: Let a non-barking watchdog bite: Cliptographic signatures with an offline watchdog. In: Public-Key Cryptography - PKC 2019 - 22nd IACR International Conference on Practice and Theory of Public-Key Cryptography, Beijing, China, April 14-17, 2019, Proceedings, Part I. pp. 221–251 (2019), https://doi.org/10.1007/978-3-030-17253-4_8 1
17. Coretti, S., Dodis, Y., Guo, S., Steinberger, J.P.: Random oracles and non-uniformity. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part I. LNCS, vol. 10820, pp. 227–258. Springer, Heidelberg (Apr / May 2018). https://doi.org/10.1007/978-3-319-78381-9_9 1
18. Coron, J.S., Dodis, Y., Malinaud, C., Puniya, P.: Merkle-Damgård revisited: How to construct a hash function. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 430–448. Springer, Heidelberg (Aug 2005). https://doi.org/10.1007/11535218_26 1, 2.1, 5, 6
19. Degabriele, J.P., Farshim, P., Poettering, B.: A more cautious approach to security against mass surveillance. In: Leander, G. (ed.) FSE 2015. LNCS, vol. 9054, pp. 579–598. Springer, Heidelberg (Mar 2015). https://doi.org/10.1007/978-3-662-48116-5_28 1
20. Degabriele, J.P., Paterson, K.G., Schuldt, J.C.N., Woodage, J.: Backdoors in pseudorandom number generators: Possibility and impossibility results. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part I. LNCS, vol. 9814, pp. 403–432. Springer, Heidelberg (Aug 2016). https://doi.org/10.1007/978-3-662-53018-4_15 1
21. Dodis, Y., Ganesh, C., Golovnev, A., Juels, A., Ristenpart, T.: A formal treatment of backdoored pseudorandom generators. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part I. LNCS, vol. 9056, pp. 101–126. Springer, Heidelberg (Apr 2015). https://doi.org/10.1007/978-3-662-46800-5_5 1
22. Dodis, Y., Guo, S., Katz, J.: Fixing cracks in the concrete: Random oracles with auxiliary input, revisited. In: Coron, J.S., Nielsen, J.B. (eds.) EUROCRYPT 2017, Part II. LNCS, vol. 10211, pp. 473–495. Springer, Heidelberg (Apr / May 2017). https://doi.org/10.1007/978-3-319-56614-6_16 1
23. Dodis, Y., Reyzin, L., Rivest, R.L., Shen, E.: Indifferentiability of permutation-based compression functions and tree-based modes of operation, with applications to MD6. In: Dunkelman, O. (ed.) FSE 2009. LNCS, vol. 5665, pp. 104–121. Springer, Heidelberg (Feb 2009). https://doi.org/10.1007/978-3-642-03317-9_7 1
24. Maurer, U.M., Renner, R., Holenstein, C.: Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 21–39. Springer, Heidelberg (Feb 2004). https://doi.org/10.1007/978-3-540-24638-1_2 1, 2.1
25. Mennink, B.: Indifferentiability of double length compression functions. In: Stam, M. (ed.) 14th IMA International Conference on Cryptography and Coding. LNCS, vol. 8308, pp. 232–251. Springer, Heidelberg (Dec 2013). https://doi.org/10.1007/978-3-642-45239-0_14 1
26. Mironov, I., Stephens-Davidowitz, N.: Cryptographic reverse firewalls. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part II. LNCS, vol. 9057, pp. 657–686. Springer, Heidelberg (Apr 2015). https://doi.org/10.1007/978-3-662-46803-6_22 1
27. Naito, Y.: Indifferentiability of double-block-length hash function without feed-forward operations. In: Pieprzyk, J., Suriadi, S. (eds.) ACISP 17, Part II. LNCS, vol. 10343, pp. 38–57. Springer, Heidelberg (Jul 2017) 1
28. Russell, A., Tang, Q., Yung, M., Zhou, H.S.: Cliptography: Clipping the power of kleptographic attacks. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016, Part II. LNCS, vol. 10032, pp. 34–64. Springer, Heidelberg (Dec 2016). https://doi.org/10.1007/978-3-662-53890-6_2 1
29. Russell, A., Tang, Q., Yung, M., Zhou, H.S.: Generic semantic security against a kleptographic adversary. In: Thuraisingham, B.M., Evans, D., Malkin, T., Xu, D. (eds.) ACM CCS 2017. pp. 907–922. ACM Press (Oct / Nov 2017). https://doi.org/10.1145/3133956.3133993 1
30. Russell, A., Tang, Q., Yung, M., Zhou, H.S.: Correcting subverted random oracles. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part II. LNCS, vol. 10992, pp. 241–271. Springer, Heidelberg (Aug 2018). https://doi.org/10.1007/978-3-319-96881-0_9 1, 1.1, 1.3, 2.2, 2.2, 6
31. Young, A., Yung, M.: The dark side of "black-box" cryptography, or: Should we trust capstone? In: Koblitz, N. (ed.) CRYPTO'96. LNCS, vol. 1109, pp. 89–103. Springer, Heidelberg (Aug 1996). https://doi.org/10.1007/3-540-68697-5_8 1
32. Young, A., Yung, M.: Kleptography: Using cryptography against cryptography. In: Fumy, W. (ed.) EUROCRYPT'97. LNCS, vol. 1233, pp. 62–74. Springer, Heidelberg (May 1997). https://doi.org/10.1007/3-540-69053-0_6 1