On the Gold Standard for Security of Universal Steganography

Sebastian Berndt[1] and Maciej Liśkiewicz[2]

[1] Department of Computer Science, Kiel University
seb@informatik.uni-kiel.de
[2] Institute for Theoretical Computer Science, University of Lübeck
liskiewi@tcs.uni-luebeck.de

Abstract. While symmetric-key steganography is quite well understood both in the information-theoretic and in the computational setting, many fundamental questions about its public-key counterpart resist persistent attempts to solve them. The computational model for public-key steganography was proposed by von Ahn and Hopper in EUROCRYPT 2004. At TCC 2005, Backes and Cachin gave the first universal public-key stegosystem – i. e. one that works on all channels – achieving security against replayable chosen-covertext attacks (SS-RCCA) and asked whether security against non-replayable chosen-covertext attacks (SS-CCA) is achievable. Later, Hopper (ICALP 2005) provided such a stegosystem for every efficiently sampleable channel, but did not achieve universality. He posed the question whether universality *and* SS-CCA-security can be achieved simultaneously. No progress on this question has been achieved since more than a decade. In our work we solve Hopper's problem in a somehow complete manner: As our main positive result we design an SS-CCA-secure stegosystem that works for *every* memoryless channel. On the other hand, we prove that this result is the best possible in the context of universal steganography. We provide a family of 0-*memoryless* channels – where the already sent documents have only marginal influence on the current distribution – and prove that no SS-CCA-secure steganography for this family exists in the standard non-look-ahead model.

1 Introduction

Steganography is the art of hiding the transmission of information to achieve secret communication without revealing its presence. In the basic setting, the aim of the steganographic encoder (often called Alice or the stegoencoder) is to hide a secret message in a document and to send it to the stegodecoder (Bob) via a public *channel* which is completely monitored by an *adversary* (Warden or steganalyst). The channel is modeled as a probability distribution of legal documents, called *covertexts*, and the adversary's task is to distinguish those from altered ones, called *stegotexts*. Although strongly connected with cryptographic encryption, steganography is not encryption: While encryption only tries to hide the content of the transmitted message, steganography aims to hide both the message and the fact that a message was transmitted at all.

As in the cryptographic setting, the security of the stegosystems should only rely on the secrecy of the keys used by the system. *Symmetric-key* steganography, which assumes that Alice and Bob share a secret-key, has been a subject of intensive study both in an information-theoretic [7,36,40] and in a computational setting [13,22,23,25,26,30]. A drawback of such an approach is that the encoder and the decoder must have shared a key in a secure way. This may be unhandy, e. g. if the encoder communicates with several parties.

In order to avoid this problem in cryptography, Diffie and Hellman provided the notion of a *public-key scenario* in their groundbreaking work [15]. This idea has proved to be very useful and is currently used in nearly every cryptographic application. Over time, the notion of security against so-called *chosen ciphertext attacks* (chosen-ciphertext attack (CCA)-security) has established itself as the "gold standard" for security in the public-key scenario [20,27]. In this setting, an attacker has also access to a decoding oracle that decodes every ciphertext different from the challenge-text. Dolev, Dwork and Naor [16] proved that the simplest assumption for public-key cryptography – the existence of trapdoor permutations – is sufficient to construct a CCA-secure public key cryptosystem.

Somewhat in contrast to the research in cryptographic encryption, only very little studies in steganography have been concerned so far within the public-key setting. Von Ahn and Hopper [38,39] were the first to give a formal framework and to prove that secure public-key steganography exists. They formalized security against a *passive* adversary in which Warden is allowed to provide challenge-hiddentexts to Alice in hopes of distinguishing covertexts from stegotexts encoding the hiddentext of his choice. For a restricted model, they also defined security against an active adversary; It is assumed, however, that Bob must know the identity of Alice, which deviates from the common bare public-key scenario.

Importantly, the schemes provided in [38,39] are *universal* (called also *black-box* in the literature). This property guarantees that the systems are secure with respect not only to a concrete channel $\mathcal{C}$ but to a broad range of channels. The importance of universality is based on the fact that typically no good description of the distribution of a channel is known.

In [3], Backes and Cachin provided a notion of security for public-key steganography with *active* attacks, called *steganographic chosen-covertext attacks (SS-CCAs)*. In this scenario the warden may provide a challenge-hiddentext to Alice and enforce the stegoencoder to send stegotexts encoding the hiddentext of his choice. The warden may then insert documents into the channel between Alice and Bob and observe Bob's responses in hope of detecting the steganographic communication. This is the steganographic equivalent of a chosen ciphertext attack against encryption and it seems to be the most general type of security for public-key steganography with active attacks similar to CCA-security in encryption. Backes and Cachin also gave a universal public-key stegosystem which, although not secure in the general SS-CCA-setting, satisfies a relaxed notion called *steganographic security against publicly-detectable replayable adaptive chosen-covertext attacks* (steganographic replayable chosen-covertext attack (SS-RCCA)) inspired by the work of Canetti et al. [8]. In this relaxed setting, the warden may still provide a hiddentext to Alice and is allowed to insert documents into the channel between Alice and Bob but with the restriction that the warden's document does not encode the chosen hiddentext. Backes and Cachin left as an open problem if secure public-key steganography exists at all in the SS-CCA-framework.

This question was answered by Hopper [21] in the affirmative in case Alice and Bob communicate via an efficiently sampleable channel $\mathcal{C}$. He proved (under the assumption of a CCA-secure cryptosystem) that for every such channel $\mathcal{C}$ there is an SS-CCA-secure stegosystem $\mathsf{PKStS}_{\mathcal{C}}$ on $\mathcal{C}$. The system cleverly "derandomizes" sampling documents by using the sampling-algorithm of the channel and using a pseudorandom generator to deterministically embed the encrypted message. Hence, $\mathsf{PKStS}_{\mathcal{C}}$ is only secure on the single channel $\mathcal{C}$ and is thus not universal. Hopper [21] posed as a challenging open problem to show the (non)existence of a universal SS-CCA-secure stegosystem. Since more than a decade, public key steganography has been used as a tool in different contexts (e. g. broadcast steganography [17] and private computation [9,11]), but this fundamental question remained open.

We solve Hopper's problem in a complete manner by proving (under the assumption of the existence of doubly-enhanced trapdoor permutations and collision-resistant hash functions) the existence of an SS-CCA-secure public key stegosystem that works for every *memoryless* channel, i. e. such that the documents are independently distributed (for a formal definition see next section). On the other hand, we also prove that the influence of the history – the already sent documents – dramatically limits the security of stegosystems in the realistic non-look-ahead model: We show that no stegosystem can be SS-CCA-secure against all 0-memoryless channels in the non-look-ahead model. In these channels, the influence of the history is minimal. We thereby demonstrate a clear dichotomy result for universal public-key steganography: While memoryless channels do exhibit an SS-CCA-secure stegosystem, the introduction of the history prevents this kind of security.

Our Contribution. As noted above, the stegosystem of Backes and Cachin has the drawback that it achieves a weaker security than SS-CCA-security while it works on every channel [3]. On the other hand, the stegosystem of Hopper achieves SS-CCA-security but is specialized to a single channel [21]. We prove (under the assumption of the existence of doubly-enhanced trapdoor permutations and collision-resistant hash functions) that there is a stegosystem that is SS-CCA-secure on a large class of channels (namely the memoryless ones). The main technical novelty is a method to generate covertexts for the message m such that finding a second sequence of covertexts that encodes m is hard. Hopper achieves this at the cost of the universality of his system, while we still allow a very large class of channels. We thereby answer the question of Hopper in the affirmative, in case of memoryless channels. Note that before this work, it was not even known whether an SS-CCA-secure stegosystem exists that works for some class of channels (Hopper's system only works on a single channel that is hard-wired into the system). Furthermore, we prove that SS-CCA-security for memoryless channels is the best possible in a very natural model: If the history influences the channel distribution in a minor way, i. e. only by its length, we prove that SS-CCA-security is not achievable in the standard non-look-ahead model of von Ahn and Hopper. In Table 1, we compare our results with previous works.

Table 1. Comparison of the public-key stegosystems

Paper	Security	Channels	Applicability
von Ahn and Hopper [38]	passive	universal	possible
Backes and Cachin [3]	SS-RCCA	universal	possible
Hopper [21]	SS-CCA	single constr. channel	possible
This work (Theorem 10)	SS-CCA	all memoryless channels	possible
This work (Theorem 12)	SS-CCA	universal	impossible*

* In the non-look-ahead model against non-uniform wardens.

Related Results. Anderson and Petitcolas [1] and Craver [12], have both, even before the publication of the work by von Ahn and Hopper [38,39], described ideas for public-key steganography, however, with only heuristic arguments for security. Van Le and Kurosawa [28] showed that every efficiently sampleable channel has an SS-CCA-secure public-key stegosystem. A description of the channel is built into the stegosystem and it makes use of a pseudo-random generator G that encoder and decoder share. But the authors make a strong assumption concerning changes of internal states of G each time the embedding operation is performed, which does not fit into the usual models of cryptography and steganography. Lysyanskaya and Meyerovich [32] investigated the influence of the sampling oracle on the security of public key stegosystems with passive attackers. They prove that the stegosystem of von Ahn and Hopper [39] becomes insecure if the approximation of the channel distribution by the sampling oracle deviates only slightly from the correct distribution. They also construct a channel, where no incorrect approximation of the channel yields a secure stegosystem. This strengthens the need for universal stegosystems, as even tiny approximation errors of the channel distribution may lead to huge changes with regard to the security of the system. Fazio, Nicolosi and Perera [17] extended public-key steganography to the multi-recipient setting, where a single sender communicates with a dynamically set of receivers. Their system is designed such that no outside party and no unauthorized user is able to detect the presence of these broadcast communication. Cho, Dachma-Soled and Jarecki [11] upgraded the covert multi-party computation model of Chandran et al. [9] to the concurrent case and gave protocols for several fundamental operations, e. g. string equality and set intersection. Their steganographic (or *covert*) protocols are based upon the decisional Diffie-Hellman problem.

The paper is organized as follows. Section 2 contains the basic definitions and notations. In Section 3, we give an example attack on the stegosystem of Backes and Cachin to highlight the differences between SS-RCCA-security and SS-CCA-security. The following Section 4 contains a high-level view of our construction. Section 5 uses the results of [21] to prove that one can construct cryptosystems with ciphertexts that are indistinguishable from a distribution on bitstrings related to the hypergeometric distribution, which we will need later on. The main core of our protocol is an algorithm to order the documents in an undetectable way that still allows us to transfer information. This ordering is described in Section 6. Our results concerning the existence of SS-CCA-secure steganography for every memoryless channel are then presented and proved in Section 7. Finally, Section 8 contains the impossibility result for SS-CCA-secure stegosystems in the non-look-ahead model on 0-memoryless channels.

In order to improve the presentation, we moved proofs of some technical statements to the appendix.

2 Definitions and Notation

If S is a finite set, we write $x \twoheadleftarrow S$ to denote the *random* assignment of a uniformly chosen element of S to x. If A is a probability distribution or a randomized algorithm, we write $x \leftarrow A$ to denote the assignment of the output of A, taken over the internal coin-flips of A.

As our cryptographic and steganographic primitives will be parameterized by the key length κ, we want that the ability of any polynomial algorithm to attack this primitives is lower than the inverse of all polynomials in κ. This is modeled by the definition of a negligible function. A function $\mathsf{negl}\colon \mathbb{N} \to [0,1]$ is called *negligible*, if for every polynomial p, there is an $N_0 \in \mathbb{N}$ such that $\mathsf{negl}(N) < p(N)^{-1}$ for every $N \geq N_0$. For a probability distribution D on support X, the *min-entropy* $H_\infty(D)$ is defined as $\inf_{x\in X}\{-\log D(x)\}$.

We also need the notion of a *strongly 2-universal hash function*, which is a set of functions G mapping bitstrings of length ℓ to bitstrings of length $\ell' < \ell$ such that for all $x, x' \in \{0,1\}^\ell$ with $x \neq x'$ and all (not necessarily different) $y, y' \in \{0,1\}^{\ell'}$, we have $|\{f \in G \mid f(x) = y \wedge f(x') = y'\}| = \frac{|G|}{2^{2\ell'}}$. If $\ell/\ell' \in \mathbb{N}$, a typical example of such a family is the set of functions

$$\{x \mapsto \left(\sum_{i=1}^{\ell/\ell'} a_i x_i + b\right) \bmod 2^{\ell'} \mid a_1, \ldots, a_{\ell/\ell'}, b \in \{0, \ldots, 2^{\ell'}-1\}\ \},$$

where x_i denotes the i-th block of length ℓ' of x and we implicitly use the canonical bijection between $\{0,1\}^n$ and the finite field $\{0, \ldots, 2^n - 1\}$. See e. g. the textbook of Mitzenmacher and Upfal [33] for more information on this. For two polynomials ℓ and ℓ', a *strongly 2-universal hash family* is a family $\mathcal{G} = \{G_\kappa\}_{\kappa\in\mathbb{N}}$ such that every G_κ is a strongly 2-universal hash function mapping strings of length $\ell(\kappa)$ to strings of length $\ell'(\kappa)$.

Channels and Stegosystems. In order to be able to embed messages into unsuspicious communication, we first need to provide a definition for this. We model the communication as an unidirectional transfer of *documents* that we will treat as strings of length n over a constant-size alphabet Σ. The communication is defined via the concept of a *channel* $\mathcal{C}$ on Σ: A function, that maps, for every $n \in \mathbb{N}$, a *history* $\mathsf{hist} \in (\Sigma^n)^*$ to a probability distribution on Σ^n. We denote this probability distribution by $\mathcal{C}_{\mathsf{hist},n}$ and its *min-entropy* $H_\infty(\mathcal{C}, n)$ as $\min_{\mathsf{hist}}\{H_\infty(\mathcal{C}_{\mathsf{hist},n})\}$.

Definition 1. *We say that a channel $\mathcal{C}$ is* memoryless*, if $\mathcal{C}_{\mathsf{hist},n} = \mathcal{C}_{\mathsf{hist}',n}$ for all* $\mathsf{hist}, \mathsf{hist}'$*, i. e. if the history has no effect on the channel distribution.*

Note the difference between memoryless and 0-*memoryless* channels of Lysyanskaya and Meyerovich [32], where only the *length* of the history has an influence on the channel, since the channel distributions are described by the use of *memoryless Markov chains*:

Definition 2 ([32]). *A channel $\mathcal{C}$ is* 0-memoryless*, if $\mathcal{C}_{\mathsf{hist},n} = \mathcal{C}_{\mathsf{hist}',n}$ for all* $\mathsf{hist}, \mathsf{hist}'$ *such that* $|\,\mathsf{hist}\,| = |\,\mathsf{hist}'\,|$.

A stegosystem PKStS tries to embed messages of length $\mathsf{PKStS.ml}$ into $\mathsf{PKStS.ol}$ documents of the channel $\mathcal{C}$ that each have size $\mathsf{PKStS.dl}$, such that this sequence is indistinguishable from a sequence of typical documents. A *public-key stegosystem* PKStS with message length $\mathsf{PKStS.ml}\colon \mathbb{N} \to \mathbb{N}$, document length $\mathsf{PKStS.dl}\colon \mathbb{N} \to \mathbb{N}$, and output length $\mathsf{PKStS.ol}\colon \mathbb{N} \to \mathbb{N}$ (all functions of the security parameter κ) is a triple of polynomial probabilistic Turing machines (PPTMs) $[\mathsf{PKStS.Gen}, \mathsf{PKStS.Enc}, \mathsf{PKStS.Dec}]$[3] with the functionalities:

- The *key generation* Gen on input 1^κ produces a pair (pk, sk) consisting of a *public key pk* and a *secret key sk* (we assume that *sk* also fully contains *pk*).
- The *encoding* algorithm Enc takes as input the public key *pk*, a message $m \in \{0,1\}^{\mathsf{ml}(\kappa)}$, a history $\mathsf{hist} \in (\Sigma^{\mathsf{dl}(\kappa)})^*$ and some state information $s \in \{0,1\}^*$ and produces a document $d \in \Sigma^{\mathsf{dl}(\kappa)}$ and state information $s' \in \{0,1\}^*$ by being able to sample from $\mathcal{C}_{\mathsf{hist},\mathsf{dl}(\kappa)}$. By $\mathsf{Enc}^{\mathcal{C}}(pk, m, \mathsf{hist})$, we denote the complete output of $\mathsf{ol}(\kappa)$ documents one by one. Note that generally, the encoder needs to decide upon document d_i before it is able to get samples for the $(i+1)$-th document, as in the secret-key model of Hopper et al. [23, Section 2, "channel access"] and the public-key model of von Ahn and Hopper [38,39, Section 3]. This captures the notion that an attacker should have as much information as possible while the stegosystem is not able to look-ahead into the future. To highlight this restriction, we call this model the *non-look-ahead model*. Note that this is no restriction for memoryless channels.
- The *decoding* algorithm Dec takes as input the secret key *sk*, a sequence of documents $d_1, \ldots, d_{\mathsf{ol}(\kappa)}$, history hist and outputs a message m'.

The following properties are essential for stegosystems PKStS with output length $\ell = \mathsf{PKStS.ol}(\kappa)$. It is *universal* (*black box*), if it works on every channel without

[3] We will drop the prefix PKStS if the context is clear.

prior knowledge of the probability distribution of the channel. Clearly channels with too small min-entropy (such as deterministic channels) are not suitable for steganographic purposes. We thus concentrate only on channels with sufficiently large min-entropy.

The system is *reliable* if the probability that the decoding fails is bounded by a negligible function. Formally, the *unreliability* $\mathbf{UnRel}_{\mathsf{PKStS},\mathcal{C}}(\kappa)$ is defined as probability that the decoding fails, i. e.

$$\max_{m,\mathsf{hist}}\{\Pr_{(pk,sk)\leftarrow\mathsf{PKStS.Gen}(1^\kappa)}[\mathsf{PKStS.Dec}(sk,\mathsf{PKStS.Enc}^{\mathcal{C}}(pk,m,\mathsf{hist}),\mathsf{hist})\neq m]\}.$$

The system PKStS is *secure*, if every polynomial attacker W (the *warden*) has only negligible success probability. W works in two phases: In the first phase (called $\mathsf{W.Find}$), the warden has access to the channel $\mathcal{C}$ and to a *decoding oracle* $\mathsf{Dec}_{sk}(\cdot)$, that returns upon input $d_1,\ldots,d_\ell$ and hist the same result as $\mathsf{PKStS.Dec}(sk,(d_1,\ldots,d_\ell),\mathsf{hist})$. At the end of the first phase, the warden chooses a message m^* and a history hist^*.

At the beginning of the second phase (called $\mathsf{W.Guess}$), the warden gets a sequence of documents $\boldsymbol{d}^*=d_1^*,\ldots,d_\ell^*$, which is with probability of 50% the result of $\mathsf{PKStS.Enc}^{\mathcal{C}}(pk,m^*,\mathsf{hist}^*)$ and with probability of 50% just the result of sampling ℓ random documents from $\mathcal{C}_{\mathsf{hist}^*,n}$, which we denote as $\mathcal{C}^{\ell}_{\mathsf{hist}^*,n}$. Speaking more precisely, this sampling is done as follows: $d_i^*\leftarrow\mathcal{C}_{\mathsf{hist}^*||d_1^*||\ldots||d_{i-1}^*,n}$ for $i=1,\ldots,\ell$. Next, the warden should distinguish between these two cases by having access to another *decoding oracle*. Depending on the oracle type, two definitions for security were proposed by Backes and Cachin in [3].

In the SS-CCA-model (chosen covertext attack), the decoding oracle, denoted as $\mathsf{Dec}_{sk,\boldsymbol{d}^*}(\cdot)$, works like $\mathsf{Dec}_{sk}(\cdot)$ on every input different from $\boldsymbol{d}^*$. If $\mathsf{Dec}_{sk,\boldsymbol{d}^*}(\cdot)$ is called with input $\boldsymbol{d}^*$, it simply returns $\perp$. In the weaker SS-RCCA-model (restricted chosen ciphertext attack), the decoding oracle, denoted as $\mathsf{Dec}_{sk,\boldsymbol{d}^*,m^*}(\cdot)$, works like $\mathsf{Dec}_{sk}(\cdot)$ on most inputs: If the input d equals $\boldsymbol{d}^*$ or is a valid encoding of m^* (a *replay* of $\boldsymbol{d}^*$), the oracle simply returns $\perp$.

Formally, SS-CCA-security is defined by the SS-CCA-security game given below and the advantage of $\mathsf{W}=[\mathsf{W.Find},\mathsf{W.Guess}]$ is defined as

$$\mathbf{Adv}^{\text{ss-cca}}_{\mathsf{W},\mathsf{PKStS},\mathcal{C}}(\kappa)=\left|\Pr[\mathsf{SS\text{-}CCA\text{-}Dist}(\mathsf{W},\mathsf{PKStS},\mathcal{C},\kappa)=1]-\frac{1}{2}\right|.$$

SS-CCA-security game: $\mathsf{SS\text{-}CCA\text{-}Dist}(\mathsf{W},\mathsf{PKStS},\mathcal{C},\kappa)$

```
Input: warden $\mathsf{W}$, stegosystem $\mathsf{PKStS}$, channel $\mathcal{C}$, security parameter $\kappa$
 1: $(pk,sk)\leftarrow\mathsf{PKStS.Gen}(1^\kappa)$; $(m^*,\mathsf{hist}^*,s)\leftarrow\mathsf{W.Find}^{\mathsf{Dec}_{sk},\mathcal{C}}(pk)$
 2: $b\leftarrow\{0,1\}$
 3: if $b=0$ then $\boldsymbol{d}^*\leftarrow\mathsf{PKStS.Enc}^{\mathcal{C}}(pk,m^*,\mathsf{hist}^*)$ else $\boldsymbol{d}^*\leftarrow\mathcal{C}^{\ell}_{\mathsf{hist}^*,n}$
 4: $b'\leftarrow\mathsf{W.Guess}^{\mathsf{Dec}_{sk,\boldsymbol{d}^*},\mathcal{C}}(pk,m^*,\mathsf{hist}^*,s,\boldsymbol{d}^*)$
 5: if $b'=b$ then return 1 else return 0
```

A stegosystem PKStS is called SS-CCA-secure against channel $\mathcal{C}$ if for some negligible function negl and all wardens W, we have $\mathbf{Adv}^{\text{ss-cca}}_{\mathsf{W},\mathsf{PKStS},\mathcal{C}}(\kappa) \leq \mathsf{negl}(\kappa)$. We define SS-RCCA-security analogously, where the Guess phase uses $\mathsf{Dec}_{sk,\boldsymbol{d}^*,m^*}$ as decoding oracle. Formally, a stegosystem is *universally SS-CCA-secure* (or just universal), if it is SS-CCA-secure against all channels of sufficiently large (i. e. super-logarithmic in κ) min-entropy.

Cryptographic Primitives. Due to space constraints, we only give informal definitions of the used cryptographic primitives and refer the reader to the textbook of Katz and Lindell [24] for complete definitions.

We will make use of different cryptographic primitives, namely hash functions, pseudorandom permutations and CCA-secure cryptosystems. A *collision-resistant hash function (CRHF)* $\mathsf{H} = (\mathsf{H.Gen}, \mathsf{H.Eval})$ is a pair of PPTMs such that $\mathsf{H.Gen}$ upon input 1^κ produces a key $k \in \{0,1\}^\kappa$. The keyed function $\mathsf{H.Eval}$ takes the key $k \leftarrow \mathsf{H.Gen}(1^\kappa)$ and a string $x \in \{0,1\}^{\mathsf{H.in}(\kappa)}$ and produces a string $\mathsf{H.Eval}_k(x)$ of length $\mathsf{H.out}(\kappa) < \mathsf{H.in}(\kappa)$. The probability of every PPTM Fi to find a collision – two strings $x \neq x'$ such that $\mathsf{H.Eval}_k(x) = \mathsf{H.Eval}_k(x')$ – upon random choice of k is negligible. For a set X, denote by $\mathsf{Perms}(X)$ the set of all permutations on X. A *pseudorandom permutation (PRP)* $\mathsf{P} = (\mathsf{P.Gen}, \mathsf{P.Eval})$ is a pair of PPTMs such that $\mathsf{P.Gen}$ upon input 1^κ produces a key $k \in \{0,1\}^\kappa$. The keyed function $\mathsf{P.Eval}$ takes the key $k \leftarrow \mathsf{P.Gen}(1^\kappa)$ and is a permutation on the set $\{0,1\}^{\mathsf{P.in}(\kappa)}$. An attacker Dist (the *distinguisher*) is given black-box access to $P \twoheadleftarrow \mathsf{Perms}(\{0,1\}^{\mathsf{P.in}(\kappa)})$ or to $\mathsf{P.Eval}_k$ for a randomly chosen k and should distinguish between those scenarios. The success probability of every Dist is negligible. A *public key encryption scheme (PKES)* $\mathsf{PKES} = (\mathsf{PKES.Gen}, \mathsf{PKES.Enc}, \mathsf{PKES.Dec})$ is a triple of PPTMs such that $\mathsf{PKES.Gen}(1^\kappa)$ produces a pair of keys (pk, sk) with $|pk| = \kappa$ and $|sk| = \kappa$. The key pk is called the *public key* and the key sk is called the *secret key* (or *private key*). The *encryption algorithm* $\mathsf{PKES.Enc}$ takes as input pk and a plaintext $m \in \{0,1\}^{\mathsf{PKES.ml}(\kappa)}$ of length $\mathsf{PKES.ml}(\kappa)$ and outputs a ciphertext $c \in \{0,1\}^{\mathsf{PKES.cl}(\kappa)}$ of length $\mathsf{PKES.cl}(\kappa)$. The *decryption algorithm* $\mathsf{PKES.Dec}$ takes as input sk and the ciphertext c and produces a plaintext $m \in \{0,1\}^{\mathsf{PKES.ml}(\kappa)}$. Informally, we will allow an attacker A to first choose a message m^* that should be encrypted and denote this by $\mathsf{A.Find}$. In the next step ($\mathsf{A.Guess}$), the attacker gets c^*, which is either $\mathsf{Enc}(pk, m^*)$ or a random bitstring. He is allowed to decrypt ciphertexts different from c^* and his task is to distinguish between these two cases. This security notion is known as security against chosen-ciphertext\$ attacks (CCA\$s). For an attacker A on cryptographic primitive $\Pi \in \{\text{hash}, \text{prp}, \text{pkes}\}$ with implementation X, we write $\mathbf{Adv}^{\Pi}_{\mathsf{A},X,\mathcal{C}}(\kappa)$ for the success probability of A against X relative to channel $\mathcal{C}$, i. e. the attacker A also has access to a sampling oracle of $\mathcal{C}$. In case of encryption schemes, the superscript cca\$ is used instead of pkes.

Due to the works [16,18,31,34] we know that CCA\$-secure cryptosystems and PRPs can be constructed from doubly-enhanced trapdoor permutations resp. one-way functions, while CRHFs can not be constructed from them in a *black-box* way, as Simon showed an oracle-separation in [37].

3 Detecting the Scheme of Backes and Cachin

In order to understand the difference between SS-CCA-security and the closely related, but weaker, SS-RCCA-security, we give a short presentation of the universal SS-RCCA-stegosystem of Backes and Cachin [3]. We also show that their system is not SS-CCA-secure, which was already noted by Hopper in [21]. The proof of insecurity nicely illustrates the difference between the security models. It also highlights the main difficulty of SS-CCA-security: One needs to prevent so called *replay attacks*, where the warden constructs upon stegotext c another stegotext c' – the *replay* of c – that embeds the same message as c.

Backes and Cachin [3] showed that there is a universal SS-RCCA-secure stegosystem under the assumption that a replayable chosen-covertext\$ attack (RCCA\$)-secure cryptosystem exists.[4] They make use of a technique called *rejection sampling.* Let $\{G_\kappa\}_{\kappa\in\mathbb{N}}$ be a strongly 2-universal hash function family, $f \in G_\kappa$ a function, $\mathcal{C}$ be a channel, hist be a history and $b \in \{0,1\}$ be a bit. The algorithm $\mathsf{rejsam}(f, \mathcal{C}, b, \mathsf{hist})$ samples documents $d \leftarrow \mathcal{C}_{\mathsf{hist},\mathsf{dl}(\kappa)}$ until it finds a document d^* such that $f(d^*) = b$ or until it has sampled κ documents. If PKES is an RCCA\$-secure cryptosystem, they define a stegosystem that computes $(b_1,\dots,b_\ell) \leftarrow \mathsf{PKES.Enc}(pk, m)$ and then sends $d_1, d_2, \dots, d_\ell$, where $d_i \leftarrow \mathsf{rejsam}(f, \mathcal{C}, b_i, \mathsf{hist}\,||d_1||\dots||d_{i-1})$. The function $f \in G_\kappa$ is also part of the public key. The system is universal as it does not assume any knowledge on $\mathcal{C}$.

They then prove that this stegosystem is SS-RCCA-secure. And indeed, one can show that their stegosystem is not SS-CCA-secure by constructing a generic warden W that works as follows: The first phase $\mathsf{W.Find}$ chooses as message $m^* = 00\cdots0$ and as hist^* the empty history $\varnothing$. The second phase $\mathsf{W.Guess}$ gets $\boldsymbol{d}^* = d_1^*, \dots, d_\ell^*$ which is either a sequence of random documents or the output of the stegosystem on pk, m^*, and hist^*. The warden W now computes another document d' via rejection sampling that embedds $f(d_\ell^*)$ (the *replay* of $\boldsymbol{d}^*$) and decodes $d_1^*, \dots, d_{\ell-1}^*, d'$ via the decoder of the rejection sampling stegosystem. It then returns 0 if the returned message m' consists only of zeroes. If $\boldsymbol{d}^*$ was a sequence of random documents, it is highly unlikely that $\boldsymbol{d}^*$ decodes to a message that only consists of zeroes. If $\boldsymbol{d}^*$ was produced by the stegosystem, the decoder only returns something different from the all-zero-message if $d' = d_\ell^*$ which is highly unlikely. The warden W has advantage of $1 - \mathsf{negl}(\kappa)$ and the stegosystem is thus not SS-CCA-secure. Backes and Cachin posed the question whether a universal SS-CCA-secure stegosystem exists.

4 An High-Level View of our Stegosystem

The stegosystem of Backes and Cachin only achieves SS-RCCA-security as a single ciphertext has many different possible encodings in terms of the documents used. Hopper achieves SS-CCA-security by limiting those encodings: Due to the sampleability of the channel, each ciphertext has exactly one deterministic

[4] The definition of a RCCA\$-secure cryptosystem is analogous to SS-RCCA-security given in Section 2.

encoding in terms of the documents. While Hopper achieves SS-CCA-security, he needs to give up the universality of the stegosystem, as a description of the channel is hard-wired into the stegosystem. In order to handle as many channels as possible, we will allow many different encodings of the same ciphertext, but make it hard to find them for anyone but the stegoencoder. To simplify the presentation, we focus on the case of embedding a single bit per document. Straightforward modifications allow embedding of $\log(\kappa)$ bits.

Our stegosystem, named PKStS^* will use the following approach to encode a message m: It first samples, for sufficiently large N, a set D of N documents from the channel $\mathcal{C}$ and uses a strongly 2-universal hash function $f \in G_\kappa$ to split these documents into documents D_0 that encode bit 0 (i. e. $D_0 = \{d \in D \mid f(d) = 0\}$) and D_1 that encode bit 1 (i. e. $D_1 = \{d \in D \mid f(d) = 1\}$). Now we encrypt the message m via a certain public-key encryption system, named PKES^* (described in the next section), and obtain a ciphertext $\boldsymbol{b} = b_1, \ldots, b_L$ of length $L = \lfloor N/8 \rfloor$. Next our goal is to order the documents in D into a sequence $\boldsymbol{d} = d_1, \ldots, d_N$ such that the first L documents $d_1, \ldots, d_L$ encode $\boldsymbol{b}$ (i. e. $f(d)_i = b_i$). This ordering is performed by the algorithm $\mathsf{generate}$. However, the attacker still has several possibilities for a replay attack on this scheme, for example:

– He could exchange some document d_i by another document d'_i with $f(d_i) = f(d'_i)$ (as f is publicly known) and the sequence $d_1, \ldots, d_{i-1}, d'_i, d_{i+1}, \ldots, d_N$ would be a replay of $\boldsymbol{d}$. Such attacks will be called *sampling attacks*. To prevent the attacker from exchanging a sampled document by a non-sampled one, we also encode a hash-value of all sampled documents D and transmit this hash value to Bob.
– The attacker can exchange documents d_i and d_j, with $i < j$ and $f(d_i) = f(d_j)$, and the resulting sequence $d_1, \ldots, d_{i-1}, d_j, d_{i+1}, \ldots, d_{j-1}, d_i, d_{j+1}, \ldots, d_N$ would be a replay of $\boldsymbol{d}$. Such attacks will be called *ordering attacks*. We thus need to prevent the attacker from exchanging the positions of sampled documents. We achieve this by making sure that the ordering of the documents generated by $\mathsf{generate}$ is deterministic, i. e. for each set of documents D and each ciphertext $\boldsymbol{b}$, the ordering $\boldsymbol{d}$ generated by $\mathsf{generate}$ is deterministic. This property is achieved by using PRPs to sort the sampled documents D. The corresponding keys of the PRPs are also transmitted to Bob and the stegodecoder can thus also compute this deterministic ordering.

In total, our stegoencoder $\mathsf{PKStS}^*.\mathsf{Enc}$ works on a secret message m and on a publicly known hash-function f as follows:

1. Sample N documents D from the channel;
2. Get a hash-key k_H and compute a hash-value $h = \mathsf{H.Eval}_{k_\mathsf{H}}(\mathrm{lex}(D))$ of the sampled documents, where $\mathrm{lex}(D)$ denotes the sequence of elements of D in lexicographic order. This prevents sampling attacks, where a sampled document is replaced by a non-sampled one;

3. Get two[5] PRP-keys k_{P} and k'_{P} that will be used to determine the unique ordering of the documents in D via $\mathsf{generate}$. This prevents ordering attacks, where the order of the sampled documents is switched;
4. Encrypt the concatenation of $m, k_{\mathsf{H}}, k_{\mathsf{P}}, k'_{\mathsf{P}}, h$ via a certain public key encryption scheme PKES^* and obtain the ciphertext $\boldsymbol{b}$ of length $L = \lfloor N/8 \rfloor$. As long as PKES^* is secure, the stegodecoder is thus able to verify whether all sampled documents were sent and can also verify the ordering of the documents.
5. Compute the ordering $\boldsymbol{d}$ of the documents D via $\mathsf{generate}$ that uses the PRP keys k_{P} and k'_{P} to determine the ordering of the documents. It also uses the ciphertext $\boldsymbol{b}$ to guarantee that the first L send documents encode the ciphertext $\boldsymbol{b}$, i. e. $b_1 \dots b_L = f(d_1) \dots f(d_L)$;
6. Send the ordering of the documents $\boldsymbol{d}$.

To decode a sequence of documents $\boldsymbol{d} = d_1, \dots, d_N$, the stegodecoder of PKStS^* computes the ciphertext $b_1 = f(d_1), \dots, b_L = f(d_L)$ encoded in the first L documents of $\boldsymbol{d}$. It then decodes this ciphertext $b_1 \dots b_L$ via PKES^* to obtain the message m, the PRP keys k_{P} and k'_{P}, the hash-key k_{H} and the hash-value h. First it verifies the hash-value by checking whether $\mathsf{H}.\mathsf{Eval}_{k_{\mathsf{H}}}(\mathrm{lex}(\{d_1, \dots, d_N\}))$ equals the hash-value h to prevent sampling attacks. It then uses the PRP keys k_{P} and $k'_{\mathsf{P}'}$ to compute an ordering of the received documents via $\mathsf{generate}$ to verify that no ordering attack was used. If these validations are successful, the decoder $\mathsf{PKStS}^*.\mathsf{Dec}$ returns m; Otherwise, it concludes that $\boldsymbol{d}$ is not a valid stegotext and returns $\perp$.

Intuitively, it is clear that a successful sampling attack on this scheme would break the collision-resistant hash function H, as it needs to create a collision of $\mathrm{lex}(D)$ in order to pass the first verification step. Furthermore, a successful ordering attack would need manipulate the ciphertext $\boldsymbol{b}$ and thus break the security of the public key encryption scheme PKES^*, as the PRP keys k_{P} and k'_{P} guarantee a deterministic ordering of the documents.

As explained above, our stegoencoder computes the ordering $\boldsymbol{d} = d_1, \dots, d_N$ of the documents $D = \{d_1, \dots, d_N\}$ via the deterministic algorithm $\mathsf{generate}$, that is given the following parameters: the set of documents D, the hash-function f and the ciphertext $\boldsymbol{b}$ to ensure that the first documents of the ordering encode $\boldsymbol{b}$. It has furthermore access to the PRP keys k_{P} and k'_{P} that guarantee a deterministic ordering of the documents in D and thus prevents ordering attacks. As the ordering $\boldsymbol{d}$ produced by $\mathsf{generate}$ is sent by the stegoencoder, this ordering must be indistinguishable from a random permutation on D (which equals the channel distribution) in order to be undetectable. As $f(d_1) = b_1, \dots, f(d_L) = b_L$, not every distribution upon the ciphertext $\boldsymbol{b}$ can be used to guarantee that $\boldsymbol{d}$ is indistinguishable from a uniformly random permutation. This indistinguishability is guaranteed by requiring that the ciphertext $\boldsymbol{b}$ is distributed according to a certain distribution corresponding to a random process modeled by drawing black

[5] We believe that one permutation suffices. But in order to improve the readability of the proof for security, we use two permutations in our stegosystem.

and white balls from an urn without replacement. In our setting, the documents in D will play the role of the balls and the coloring is given by the function f.

Section 5 describes this random process in detail and proves that we can indeed construct a public-key encryption system that produces ciphertexts that are indistinguishable from this process. Section 6 contains a formal description of generate, proves that no attacker can produce a replay of its output and shows that the generated permutation is indeed indistinguishable from a random permutation. Finally, Section 7 contains the complete description of the stegosystem.

5 Obtaining Biased Ciphertexts

We will now describe a probability distribution and show how one can derive a symmetric encryption scheme with ciphertexts that are indistinguishable from this distribution. In order to do this, we first define a channel that represents the required probability distribution together with appropriate parameters, use Theorem 3 to derive a stegosystem for this channel, and finally derive a cryptosystem from this stegosystem.

Based upon a CCA\$-secure public-key cryptosystem PKES, Hopper [21] constructs for every efficiently sampleable channel $\mathcal{C}$ an SS-CCA-secure stegosystem $\mathsf{PKStS}_{\mathcal{C}}$ by "derandomizing" the rejection sampling algorithm. The only requirement upon the channel $\mathcal{C}$ is the existence of the efficient sampling algorithm and that the stegoencoder and the stegodecoder use the same sampling algorithm. Importantly, due to the efficient sampleability of $\mathcal{C}$, the encoder of $\mathsf{PKStS}_{\mathcal{C}}$ does not need an access to the sample oracle. Thus, we get the following result.

Theorem 3 (Theorem 2 in [21]). *If $\mathcal{C}$ is an efficiently sampleable channel and PKES is a CCA\$-secure public-key cryptosystem (which can be constructed from doubly enhanced trapdoor permutations[6]) then there is a stegosystem $\mathsf{PKStS}_{\mathcal{C}}$ (without an access to the sample oracle) such that for all wardens W there is a negligible function negl such that*

$$\mathbf{Adv}^{\text{ss-cca}}_{\mathsf{W},\mathsf{PKStS}_{\mathcal{C}},\mathcal{C}}(\kappa) \le \mathsf{negl}(\kappa) + 2^{-H_\infty(\mathcal{C},\kappa)/2}.$$

Note that the system $\mathsf{PKStS}_{\mathcal{C}}$ is guaranteed to be secure (under the assumption that CCA\$-secure public-key cryptosystems exist), if the channel $\mathcal{C}$ is efficiently sampleable and has min-entropy $\omega(\log \kappa)$. We call such a channel *suitable*.

The probability distribution for the ciphertexts we are interested in is the distribution for the bitstrings $\boldsymbol{b}$ we announced in the the previous section. As we will see later, the required probability can be described equivalently as follows:

– We are given N elements: N_0 of them are labeled with 0 and the remaining $N - N_0$ elements are labeled with 1.
– We draw randomly a sequence of K elements from the set (drawing without replacements) and look at the generated bitstring $\boldsymbol{b} = b_1 \dots b_K$ of length K determined by the labels of the elements.

[6] See e. g. the work [18] of Goldreich and Rothblum.

We will assume that there are enough elements of both types, i. e. that $N_0 \geq K$ and $N - N_0 \geq K$. The resulting probability distribution, denoted as $D^*_{(N,N_0,K)}$, upon bitstrings of length K is then given as

$$\Pr[D^*_{(N,N_0,K)} = b_1 \dots b_K] \;=\; \frac{1}{\binom{K}{|\boldsymbol{b}|_0}} \cdot \frac{\binom{N_0}{|\boldsymbol{b}|_0} \cdot \binom{N-N_0}{K-|\boldsymbol{b}|_0}}{\binom{N}{K}} \;= \tag{1}$$
$$\Big(\prod_{j=0}^{K-1} \frac{1}{N-j}\Big) \cdot \Big(\prod_{j=0}^{|\boldsymbol{b}|_0-1} [N_0 - j]\Big) \cdot \Big(\prod_{j=0}^{|\boldsymbol{b}|_1-1} [N - N_0 - j]\Big),$$

where $|\boldsymbol{b}|_0$ denotes the number of zero bits in $\boldsymbol{b} = b_1, \dots, b_K$ and $|\boldsymbol{b}|_1$ the number of one bits in $\boldsymbol{b}$. Note that the distribution on the number of zeroes within such bitstrings is a hypergeometric distribution with parameters N, N_0, and K.

Now we will construct a channel $\mathcal{C}^*$ upon key parameter κ with document length $n = \mathsf{dl}(\kappa) = \kappa$. In the definition below, $\mathrm{bin}(x)_y$ denotes the binary representation of length exactly y for the integer x.

- For the empty history $\varnothing$, let $\mathcal{C}^*_{\varnothing,\kappa}$ be the uniform distribution on all strings $\mathrm{bin}(N)_{\lceil \kappa/2 \rceil}\,\mathrm{bin}(N_0)_{\lfloor \kappa/2 \rfloor}$ that range over all positive integers $N, N_0 \leq 2^{\lfloor \kappa/2 \rfloor}$ such that $N \geq 8\kappa$ and $1/3 \leq N_0/N \leq 2/3$ (in our construction we need initially a stronger condition than just $N_0 \geq \kappa$ and $N - N_0 \geq \kappa$).
- If the history is of the form $\mathsf{hist}' = \mathrm{bin}(N)_{\lceil \kappa/2 \rceil}\,\mathrm{bin}(N_0)_{\lfloor \kappa/2 \rfloor}\,\mathsf{hist}$ for some $\mathsf{hist} \in \{0,1\}^*$ then we consider two cases: if $|\,\mathsf{hist}\,| \leq \frac{1}{8}N$ then the distribution $\mathcal{C}^*_{\mathsf{hist}',\kappa}$ equals $D^*_{(N-|\,\mathsf{hist}\,|,N_0-|\,\mathsf{hist}\,|_0,\kappa)}$; Otherwise, i. e. if $|\,\mathsf{hist}\,| > \frac{1}{8}N$ then $\mathcal{C}^*_{\mathsf{hist}',\kappa}$ equals the uniform distribution over $\{0,1\}^\kappa$.

It is easy to see that the min-entropy $H_\infty(\mathcal{C}^*, n) = \min_{\mathsf{hist}'}\{H_\infty(\mathcal{C}^*_{\mathsf{hist}',n})\}$ of the channel $\mathcal{C}^*$ is obtained for the history $\mathsf{hist}' = \mathrm{bin}(N)_{\lceil \kappa/2 \rceil}\,\mathrm{bin}(N_0)_{\lfloor \kappa/2 \rfloor}\,\mathsf{hist}$, with $8\kappa \leq N \leq 2^{\lfloor \kappa/2 \rfloor}$ and such that (i) $N_0 = \frac{1}{3}N$ and $\mathsf{hist} = 00\dots0$ of length $\frac{1}{8}N - \kappa$ or (ii) $N_0 = \frac{2}{3}N$ and $\mathsf{hist} = 11\dots1$ of length $\frac{1}{8}N - \kappa$. In the first case we get that the min-entropy of the distribution $\mathcal{C}^*_{\mathsf{hist}',n}$ is achieved on the bitstring $11\dots1$ of length κ and in the second case on $00\dots0$ of length κ. By Eq. (1) the probabilities to get such strings are equal to each other and, since $\kappa \leq N/8$, they can be estimated as follows:

$$\prod_{j=0}^{\kappa-1} \frac{2N/3 - j}{7N/8 - \kappa - j} \;\leq\; \left(\frac{2N/3}{7N/8 - \kappa}\right)^\kappa \;\leq\; \left(\frac{2N/3}{6N/8}\right)^\kappa \;=\; (8/9)^\kappa.$$

Thus, we get that $H_\infty(\mathcal{C}^*, n) \geq \kappa \log(9/8)$.

Moreover one can efficiently simulate the choice of N, N_0, the sampling process of $D^*_{(N,N_0,\kappa)}$ and the uniform sampling in $\{0,1\}^\kappa$. Therefore we can conclude

Lemma 4. *The channel $\mathcal{C}^*$ is suitable, i. e. it is efficiently sampleable and has min-entropy $\omega(\log \kappa)$. Furthermore, for history* $\mathsf{hist} = \mathrm{bin}(N)_{\lceil \kappa/2 \rceil}\,\mathrm{bin}(N_0)_{\lfloor \kappa/2 \rfloor}$*, with $8\kappa \leq N \leq 2^{\lceil \kappa/2 \rceil}$ and $1/3 \leq N_0/N \leq 2/3$, and for any integer $\ell \leq \frac{N}{8\kappa}$, the* *bitstrings* $\boldsymbol{b} = b_1 \dots b_K$ *of length* $K = \kappa \cdot \ell \leq N/8$ *obtained by the concatenation of* ℓ *consecutive documents sampled from the channel with history* hist*, i. e.* $b_i \leftarrow \mathcal{C}^*_{\mathsf{hist}\, b_1 \dots b_{i-1}, n=\kappa}$*, have distribution* $D^*_{(N,N_0,K)}$.

A proof for the second statement of the lemma follows directly from the construction of the channel. Now, combining the first claim of the lemma with Theorem 3 we get the following corollary.

Corollary 5. *If doubly enhanced trapdoor permutations exists, there is a stegosystem* $\mathsf{PKStS}_{\mathcal{C}^*}$ *(without an access to the sample oracle) such that for all wardens* W *there is a negligible function* negl *such that* $\mathbf{Adv}^{\text{ss-cca}}_{\mathsf{W},\mathsf{PKStS}_{\mathcal{C}^*},\mathcal{C}^*}(\kappa) \leq \mathsf{negl}(\kappa)$.

Based upon this stegosystem $\mathsf{PKStS} = \mathsf{PKStS}_{\mathcal{C}^*}$, we construct a public-key cryptosystem PKES^*, with ciphertexts of length $\mathsf{PKES}^*.\mathsf{cl}(\kappa) = \kappa \cdot \mathsf{PKStS}.\mathsf{cl}(\kappa)$ such that PKES^* also has another algorithm, called $\mathsf{PKES}^*.\mathsf{Setup}$ that takes parameters: two integers N and N_0 which satisfy $8 \cdot \mathsf{PKES}^*.\mathsf{cl}(\kappa) \leq N \leq 2^{\lfloor \kappa/2 \rfloor}$ and $N_0/N \in [1/3, 2/3]$. Calling $\mathsf{PKES}^*.\mathsf{Setup}(N, N_0)$ stores the values N, N_0 such that $\mathsf{PKES}^*.\mathsf{Enc}$ and $\mathsf{PKES}^*.\mathsf{Dec}$ can use them.

- The key generation $\mathsf{PKES}^*.\mathsf{Gen}$ simply equals the key generation algorithm $\mathsf{PKStS}.\mathsf{Gen}$.
- The encoding algorithm $\mathsf{PKES}^*.\mathsf{Enc}$ takes as parameters the public key pk and a message m. It then simulates the encoder $\mathsf{PKStS}.\mathsf{Enc}$ on key pk, message m and history $\mathsf{hist} = \mathrm{bin}(N)_{\lceil \kappa/2 \rceil}\, \mathrm{bin}(N_0)_{\lfloor \kappa/2 \rfloor}$ and produces a bitstring of length $\mathsf{PKES}^*.\mathsf{cl}(\kappa) = \mathsf{PKStS}.\mathsf{ol}(\kappa) \cdot \kappa$.
- The decoder $\mathsf{PKES}^*.\mathsf{Dec}$ simply inverts this process by simulating the stegodecoder $\mathsf{PKStS}.\mathsf{Dec}$ on key sk and history $\mathsf{hist} = \mathrm{bin}(N)_{\lceil \kappa/2 \rceil}\, \mathrm{bin}(N_0)_{\lfloor \kappa/2 \rfloor}$.

Clearly, the ciphertexts of $\mathsf{PKES}^*.\mathsf{Enc}(pk, m)$ are indistinguishable from the distribution $D^*_{(N,N_0,\mathsf{PKES}^*.\mathsf{cl}(\kappa))}$ by the second statement of Lemma 4. This generalization of Theorem 3 yields the following corollary:

Corollary 6. *If doubly-enhanced trapdoor permutations exist, there is a secure public-key cryptosystem* PKES^**, equipped with the algorithm* $\mathsf{PKES}^*.\mathsf{Setup}$ *that takes two parameters* N *and* N_0*, such that its ciphertexts are indistinguishable from the probability distribution* $D^*_{(N,N_0,\mathsf{PKES}^*.\mathsf{cl}(\kappa)))}$ *whenever* N *and* N_0 *satisfy that* $8 \cdot \mathsf{PKES}^*.\mathsf{cl}(\kappa) \leq N \leq 2^{\lfloor \kappa/2 \rfloor}$ *and* $N_0/N \in [1/3, 2/3]$.

6 Ordering the Documents

As described before, to prevent replay attacks, we need to order the sampled documents. This is done via the algorithm $\mathsf{generate}$ described in this section. To improve the readability, we will abbreviate some terms and define $L = \mathsf{PKES}^*.\mathsf{cl}(\kappa)$ and $n = \mathsf{PKStS}^*.\mathsf{dl}(\kappa)$, where PKES^* is the public-key encryption scheme from the last section and PKStS^* is our target stegosystem that we will provide later on. We also define $N = 8L$.

To order the set of documents $D \subseteq \Sigma^n$, we use the algorithm $\mathsf{generate}$, presented below. It takes the set of documents D with $|D| = N$, a hash function $f\colon \Sigma^n \to \{0,1\}$ from G_κ, a bitstring $b_1, \ldots, b_L$, and two keys $k_\mathsf{P}, k'_\mathsf{P}$ for PRPs. It then uses the PRPs to find the right order of the documents.

Algorithm: $\mathsf{generate}(D, f, b_1, \ldots, b_L, k_\mathsf{P}, k'_\mathsf{P})$

```
Input: set $D$ with $|D| = N$, hash function $f$, bits $b_1, \ldots, b_L$, PRP-keys $k_\mathsf{P}, k'_\mathsf{P}$
 1: let $D_0 = \{d \in D \mid f(d) = 0\}$ and $D_1 = \{d \in D \mid f(d) = 1\}$ ▷ We assert that
    $|D| = N$, and furthermore $|D_0| \in [N/3, 2N/3]$
 2: for $i = 1$ to $L$ do
 3:     $d_i := \arg\min_{d \in D_{b_i}} \{\mathsf{P.Eval}_{k_\mathsf{P}}(d)\}$; $D_{b_i} := D_{b_i} \setminus \{d_i\}$
 4: let $D' = D_0 \cup D_1$                              ▷ collect remaining documents
 5: for $i = L+1, \ldots, N$ do
 6:     $d_i := \arg\min_{d \in D'} \{\mathsf{P.Eval}_{k'_\mathsf{P}}(d)\}$; $D' := D' \setminus \{d_i\}$
 7: return $d_1, d_2, \ldots, d_N$
```

Note that the permutation $\mathsf{P.Eval}_{k_\mathsf{P}}$ is a permutation upon the set $\{0,1\}^n$ (i. e. on the documents themselves) and the canonical ordering of $\{0,1\}^n$ thus implicitly gives us an ordering of the documents.

We note the following important property of $\mathsf{generate}$ that shows where the urn model of the previous section comes into play. For uniform random permutations P and P', we denote by $\mathsf{generate}(\cdots, P, P')$ the run of $\mathsf{generate}$, where the use of $\mathsf{P.Eval}_{k_\mathsf{P}}$ is replaced by P and the use of $\mathsf{P.Eval}_{k'_\mathsf{P}}$ is replaced by P'. If the bits $\boldsymbol{b} = b_1, \ldots, b_L$ are distributed according to $D^*_{(N,|D_0|,L)}$, the resulting distribution on the documents then equals the channel distribution.

Lemma 7. *Let $\mathcal{C}$ be any memoryless channel, f be some hash function and D be a set of $N = 8L$ documents of $\mathcal{C}$ such that $N/3 \leq |D_0| \leq 2N/3$, where $D_0 = \{d \in D \mid f(d) = 0\}$. If the permutations P, P' are uniformly random and the bitstring $\boldsymbol{b} = b_1, \ldots, b_L$ is distributed according to $D^*_{(N,|D_0|,L)}$, the output of* $\mathsf{generate}(D, f, \boldsymbol{b}, P, P')$ *is a uniformly random permutation of D.*

Proof. Fix any document set D of size $N = 8L$ and a function f that splits D into $D_0 \dot{\cup} D_1$, with $|D_0| \geq N/3$ and $|D_1| \geq N/3$. Let $\hat{\boldsymbol{d}} = \hat{d}_1, \ldots, \hat{d}_N$ be any permutation on D. We will prove that the probability (upon bits $\boldsymbol{b}$ and permutations P, P') that $\hat{\boldsymbol{d}}$ is produced, is $1/N!$ and thus establish the result. Let $\boldsymbol{d} = d_1, \ldots, d_N$ be the random variables that denote the outcome of $\mathsf{generate}(D, f, b_1, \ldots, b_L, P, P')$.

Note that if $\boldsymbol{d}[i]$ (resp. $\hat{\boldsymbol{d}}[i]$) denotes the prefix of length i of $\boldsymbol{d}$ (resp. $\hat{\boldsymbol{d}}$), then using the chain rule formula we get

$$\Pr_{\boldsymbol{b},P,P'}[d_1 d_2 \ldots d_N = \hat{d}_1 \hat{d}_2 \ldots \hat{d}_N] = \prod_{i=1}^{N} \Pr_{\boldsymbol{b},P,P'}[d_i = \hat{d}_i \mid \boldsymbol{d}[i-1] = \hat{\boldsymbol{d}}[i-1]].$$

To estimate each of the factors of the product, we consider two cases:

– Case $i \leq L$: Let $\hat{\boldsymbol{b}} = \hat{b}_1, \dots, \hat{b}_L$ be the bitstring such that $\hat{b}_i = f(\hat{d}_i)$ and let $\hat{\boldsymbol{b}}[i]$ be the prefix $\hat{b}_1, \dots, \hat{b}_i$ of $\hat{\boldsymbol{b}}$ of length i. Clearly, for $i \leq L$ it holds that the event $d_i = \hat{d}_i$ under the condition $\boldsymbol{d}[i-1] = \hat{\boldsymbol{d}}[i-1]$ occurs iff (A) $d_i \in D_{\hat{b}_i}$ and (B) d_i is put on position $|\hat{\boldsymbol{b}}[i]|_{\hat{b}_i}$ by the permutation P with respect to $D_{\hat{b}_i}$. Due to the distribution of bit b_i in the random bits $\boldsymbol{b}$, the event $d_i \in D_{\hat{b}_i}$ occurs with probability $(|D_{\hat{b}_i}| - |\hat{\boldsymbol{b}}[i-1]|_{\hat{b}_i})/(N-i+1)$ (under the above condition). As $\boldsymbol{d}[i-1] = \hat{\boldsymbol{d}}[i-1]$ holds, exactly $|\hat{\boldsymbol{b}}[i-1]|_{\hat{b}_i}$ documents from $D_{\hat{b}_i}$ are already used in the output. As P is a uniform random permutation, the probability that d_i is put on position $|\hat{\boldsymbol{b}}[i]|_{\hat{b}_i}$ by the permutation P (with respect to $D_{\hat{b}_i}$) is thus $1/(|D_{\hat{b}_i}| - |\hat{\boldsymbol{b}}[i-1]|_{\hat{b}_i})$. Since (A) and (B) are independent, we conclude for $i \leq L$ that the probability $\Pr_{\boldsymbol{b},P,P'}[d_i = \hat{d}_i \mid \boldsymbol{d}[i-1] = \hat{\boldsymbol{d}}[i-1]]$ is equal to

$$\begin{aligned}
&\Pr_{\boldsymbol{b}}[d_i \in D_{\hat{b}_i} \mid \boldsymbol{d}[i-1] = \hat{\boldsymbol{d}}[i-1]] \times \\
&\qquad \Pr_P[P \text{ puts } d_i \text{ on position } |\hat{\boldsymbol{b}}[i]|_{\hat{b}_i} \mid \boldsymbol{d}[i-1] = \hat{\boldsymbol{d}}[i-1]] \;=\\
&\frac{|D_{\hat{b}_i}| - |\hat{\boldsymbol{b}}[i-1]|_{\hat{b}_i}}{N-i+1} \cdot \frac{1}{|D_{\hat{b}_i}| - |\hat{\boldsymbol{b}}[i-1]|_{\hat{b}_i}} \;=\; \frac{1}{N-i+1}.
\end{aligned}$$

– Case $i > L$: As the choice of P' is independent from the choice of P, the remaining $2L$ items are ordered completely random. Hence, for $i > L$ we also have

$$\Pr_{\boldsymbol{b},P,P'}[d_i = \hat{d}_i \mid \boldsymbol{d}[i-1] = \hat{\boldsymbol{d}}[i-1]] \;=\; \frac{1}{N-i+1}.$$

Putting it together, we get

$$\Pr_{\boldsymbol{b},P,P'}[d_1 d_2 \dots d_N = \hat{d}_1 \hat{d}_2 \dots \hat{d}_N] \;=\; \prod_{i=1}^{N} \frac{1}{N-i+1} = \frac{1}{N!}. \qquad \square$$

As explained above, a second property that we need is that no attacker should be able to produce a "replay" of the output of $\mathsf{generate}$. Below, we formalize this notion and analyze the security of the algorithm. An attacker A on $\mathsf{generate}$ is a PPTM, that receives nearly the same input as $\mathsf{generate}$: a set D of N documents, a hash function $f\colon \Sigma^n \to \{0,1\}$ from the family G_κ, a sequence $b_1, \dots, b_L$ of L bits, and a key k_{H} for the CRHF H. Then A outputs a sequence $d'_1, \dots, d'_N$ of documents. We say that the algorithm A is *successful* if

1. $f(d_i) = f(d'_i)$ for all $i = 1, \dots, N$,
2. $d'_1, \dots, d'_N = \mathsf{generate}(D', f, b_1, \dots, b_L, k_{\mathsf{P}}, k'_{\mathsf{P}})$, and
3. $\mathsf{H.Eval}_{k_{\mathsf{H}}}(\mathrm{lex}(D')) = \mathsf{H.Eval}_{k_{\mathsf{H}}}(\mathrm{lex}(D))$,

where D' denotes the set $\{d'_1, \dots, d'_N\}$ and, recall, $\mathrm{lex}(X)$ denotes the sequence of elements of set X in lexicographic order. We can then conclude the following lemma.

Lemma 8 (Informal). *Let $D \subseteq \Sigma^n$ be a set of documents with $|D| = N$, let $b_1, \ldots, b_L$ be a bitstring, and $f \in G_\kappa$. For every attacker* A *on* $\mathsf{generate}$*, there is a collision finder* Fi *for the CRHF* H *such that the probability that* A *is successful on $D, f, b_1, \ldots, b_L, k_{\mathsf{H}}$ is bounded by* $\mathbf{Adv}^{\text{hash}}_{\mathsf{Fi},\mathsf{H},\mathcal{C}}(\kappa)$.

The formal definition of "A is successful" as well as a formal statement of the lemma can be found in the Appendix, Section A.

7 The Steganographic Protocol PKStS^*

We now have all of the ingredients of our stegosystem, namely the CCA-secure cryptosystem PKES^* from Section 5 and the ordering algorithm $\mathsf{generate}$ from Section 6. To improve the readability, we will abbreviate some terms and define $n = \mathsf{PKStS}^*.\mathsf{dl}(\kappa)$, $\ell = \mathsf{PKStS}^*.\mathsf{ol}(\kappa)$, and $L = \mathsf{PKES}^*.\mathsf{cl}(\kappa)$, where PKES^* is the public-key encryption scheme from Section 5 and PKStS^* is the stegosystem that we will define in this section. We also let $N = 8L$.

In the following, let $\mathcal{C}$ be a memoryless channel, P be a PRP relative to $\mathcal{C}$, H be a CRHF relative to $\mathcal{C}$ and $\mathcal{G} = \{G_\kappa\}_{\kappa \in \mathbb{N}}$ be a strongly 2-universal hash family. Remember, that PKES^* has the algorithm $\mathsf{PKES}^*.\mathsf{Setup}$ that takes the additional parameters $N, N_0 \leq 2^{\lceil \kappa/2 \rceil}$, such that if $N \geq 8 \cdot \mathsf{PKES}^*.\mathsf{cl}(\kappa)$ and $N_0/N \in [1/3, 2/3]$ then the output of $\mathsf{PKES}^*.\mathsf{Enc}(pk, m)$ is indistinguishable from $D^*_{(N,N_0,\mathsf{PKES}^*.\mathsf{cl}(\kappa))}$ (see Section 5 for a discussion). Furthermore, we assume that PKES^* has very sparse support, i. e. the ratio of valid ciphertexts compared to $\{0,1\}^{\mathsf{PKES}^*.\mathsf{cl}(\kappa)}$ is negligible: If $\mathsf{PKES}^*.\mathsf{Enc}(pk, m)$ is called, we first use some public key encryption scheme PKES with very sparse support to compute $c \leftarrow \mathsf{PKES}.\mathsf{Enc}(pk, m)$ and then encrypt c via PKES^*. This construction is due to Lindell [29] and also maintains the indistinguishability of the output of $\mathsf{PKES}^*.\mathsf{Enc}$ and the distribution D^*, as this properties hold for all fixed messages m. Now we are ready to provide our stegosystem named PKStS^*. Its main core is the ordering algorithm $\mathsf{generate}$.

- The key generating $\mathsf{PKStS}^*.\mathsf{Gen}$ queries $\mathsf{PKES}^*.\mathsf{Gen}$ for a key-pair (pk, sk) and chooses a hash-function $f \twoheadleftarrow G_\kappa$. The public key of the stegosystem will be $pk^* = (pk, f)$ and the secret key will be $sk^* = (sk, f)$.
- The encoding algorithm $\mathsf{PKStS}^*.\mathsf{Enc}$ presented below (as $\mathcal{C}_n$ is memoryless we skip hist in the description) works as described in Section 4: It chooses appropriate keys, samples documents D, computes a hash value of D, generates bitstring $\boldsymbol{b}$ via PKES^*, and finally orders the documents via $\mathsf{generate}$. [7]
- To decode a sequence of documents $d_1, \ldots, d_N$, the stegodecoder $\mathsf{PKStS}^*.\mathsf{Dec}$ first computes the bit string $b_1 = f(d_1), \ldots, b_N = f(d_N)$ and computes the number $N_0 = |\{d_i \colon f(d_i) = 0\}|$. In case $|\{d_1, \ldots, d_N\}| < N$ or $N_0/N \notin$

[7] That the number of produced documents is always divisible by 8 does not hurt the security: The warden always gets the same number of documents, whether steganography is used or not.

$[1/3, 2/3]$, the decoder $\mathsf{PKStS}^*.\mathsf{Dec}$ returns $\perp$ and halts. Otherwise, using $\mathsf{PKES}^*.\mathsf{Dec}$ with sk and parameters N, N_0, it decrypts from the ciphertext $b_1, b_2, \dots, b_L$ the message m, the keys $k_\mathsf{H}, k_\mathsf{P}, k'_\mathsf{P}$ and the hash-value h. It then checks whether the hash-value h is correct and whether $d_1, \dots, d_N = \mathsf{generate}(\{d_1, \dots, d_N\}, f, b_1, \dots, b_L, k_\mathsf{P}, k'_\mathsf{P})$. Only if this is the case, the message m is returned. Otherwise, $\mathsf{PKStS}^*.\mathsf{Dec}$ decides that it can not decode the documents and returns $\perp$.

The steganographic encoder: $\mathsf{PKStS}^*.\mathsf{Enc}(pk^*, m)$

```
Input: public key $pk^* = (pk, f)$, message $m$; access to channel $\mathcal{C}_n$
 1: let $L = \mathsf{PKES}^*.\mathsf{cl}(\kappa)$ and $N = 8L$; let $D_0 := \emptyset$ and $D_1 := \emptyset$
 2: for $j = 1$ to $N$ do
 3:     sample $d_j$ from $\mathcal{C}_n$; let $D_{f(d_j)} := D_{f(d_j)} \cup \{d_j\}$
 4: $N_0 = |D_0|$
 5: if $|D_0 \cup D_1| < N$ or $N_0/N \notin [1/3, 2/3]$ then return $d_1, \dots, d_N$ and halt
 6: choose hash key $k_\mathsf{H} \leftarrow \mathsf{H.Gen}(1^\kappa)$
 7: choose PRP keys $k_\mathsf{P}, k'_\mathsf{P} \leftarrow \mathsf{P.Gen}(1^\kappa)$
 8: let $h := \mathsf{H.Eval}_{k_\mathsf{H}}(\mathrm{lex}(D_0 \cup D_1))$        ▷ compute hash
 9: call $\mathsf{PKES}^*.\mathsf{Setup}(N, N_0)$        ▷ setup $N, N_0$
10: let $b_1, b_2, \dots, b_L \leftarrow \mathsf{PKES}^*.\mathsf{Enc}(pk, m \,||\, k_\mathsf{H} \,||\, k_\mathsf{P} \,||\, k'_\mathsf{P} \,||\, h)$
11: let $\boldsymbol{d} := \mathsf{generate}(D_0 \cup D_1, f, b_1, \dots, b_L, k_\mathsf{P}, k'_\mathsf{P})$
12: return $\boldsymbol{d}$
```

Proofs of Reliabiliy and Security. We will first concentrate on the reliability of the system PKStS^* and prove that its unreliability is negligible. This is due to the fact, that the decoding always works and the encoding can only fail if a document was drawn more than once or if the sampled documents are very imbalanced with regard to f.

Theorem 9. *The probability that a message is not correctly embedded by the encoder* $\mathsf{PKStS}^*.\mathsf{Enc}$ *is at most* $3N^2 \cdot 2^{-H_\infty(\mathcal{C},\kappa)} + 2\exp(-N/54)$.

If $1 < \lambda \leq \log(\kappa)$ bits per document are embedded, this probability is bounded by $2^{2\lambda} \cdot 3N^2 \cdot 2^{-H_\infty(\mathcal{C},\kappa)} + 2^{\lambda+1}\exp(-N/54)$, which is negligible in κ if $H_\infty(\mathcal{C},\kappa)$ sufficiently large. Now, it only remains to prove that our construction is secure. The proof proceeds similar to the security proof of Hopper [21]. But instead of showing that no other encoding of a message exists, we prove that finding any other encoding of the message is infeasible via Lemma 8.

Theorem 10. *Let* $\mathcal{C}$ *be a memoryless channel,* P *be a PRP relative to* $\mathcal{C}$, *the algorithm* H *be a CRHF relative to* $\mathcal{C}$, *the cryptosystem* PKES^* *be the cryptosystem designed in Section 5 with very sparse support relative to* $\mathcal{C}$, *and* $\mathcal{G}$ *be a strongly 2-universal hash family. The stegosystem* PKStS^* *is SS-CCA-secure against* every *memoryless channel.*

$\mathrm{H}_1 = \mathcal{C}_n^N$

```
1 : $pk^* = (pk, f) \leftarrow \mathsf{PKStS}^*.\mathsf{Gen}(1^\kappa)$
2 : for $j := 1, 2, \ldots, N$ :
3 :     $d_j \leftarrow \mathcal{C}_{\mathsf{dl}(\kappa)}$
4 : return $((d_1, \ldots, d_N), pk^*)$
```

H_2

```
$pk^* = (pk, f) \leftarrow \mathsf{PKStS}^*.\mathsf{Gen}(1^\kappa)$
Lines 1 to 4 in $\mathsf{PKStS}^*.\mathsf{Enc}$
5 : $P \twoheadleftarrow \mathsf{Perms}$
6 : return $((d_{P(1)}, \ldots, d_{P(N)}), pk^*)$
```

H_3

```
$pk^* = (pk, f) \leftarrow \mathsf{PKStS}^*.\mathsf{Gen}(1^\kappa)$
Lines 1 to 4 in $\mathsf{PKStS}^*.\mathsf{Enc}$
5 : $P \twoheadleftarrow \mathsf{Perms}; P' \twoheadleftarrow \mathsf{Perms}; k_\mathsf{H} \leftarrow \mathsf{H}.\mathsf{Gen}(1^\kappa)$
6 : $b_1, b_2, \ldots, b_L \leftarrow D^*_{(N, N_0, L)}$
7 : return $(\mathsf{generate}(D_0 \cup D_1, f, b_1, \ldots, b_L, P, P'), pk^*)$
// $\mathsf{generate}(\ldots, P, P')$ uses the permutations $P, P'$
```

H_4

```
$pk^* = (pk, f) \leftarrow \mathsf{PKStS}^*.\mathsf{Gen}(1^\kappa)$
Lines 1 to 4 in $\mathsf{PKStS}^*.\mathsf{Enc}$
5 : $k_\mathsf{P} \leftarrow \mathsf{P}.\mathsf{Gen}(1^\kappa); P' \twoheadleftarrow \mathsf{Perms}; k_\mathsf{H} \leftarrow \mathsf{H}.\mathsf{Gen}(1^\kappa)$
6 : $b_1, b_2, \ldots, b_L \twoheadleftarrow D^*_{(N, N_0, L)}$
7 : return $(\mathsf{generate}(D_0 \cup D_1, f, b_1, \ldots, b_L, k_\mathsf{P}, P'), pk^*)$
// $\mathsf{generate}(\ldots, P')$ uses the permutation $P'$
```

H_5

```
$pk^* = (pk, f) \leftarrow \mathsf{PKStS}^*.\mathsf{Gen}(1^\kappa)$
Lines 1 to 4 in $\mathsf{PKStS}^*.\mathsf{Enc}$
5 : $k_\mathsf{P} \leftarrow \mathsf{P}.\mathsf{Gen}(1^\kappa); k'_\mathsf{P} \leftarrow \mathsf{P}.\mathsf{Gen}(1^\kappa); k_\mathsf{H} \leftarrow \mathsf{H}.\mathsf{Gen}(1^\kappa)$
6 : $b_1, b_2, \ldots, b_L \twoheadleftarrow D^*_{(N, N_0, L)}$
7 : return $(\mathsf{generate}(D_0 \cup D_1, f, b_1, \ldots, b_L, k_\mathsf{P}, k'_\mathsf{P}), pk^*)$
```

$\mathrm{H}_6 = \mathsf{PKStS}^*.\mathsf{Enc}$

```
$pk^* = (pk, f) \leftarrow \mathsf{PKStS}^*.\mathsf{Gen}(1^\kappa)$
Lines 1 to 4 in $\mathsf{PKStS}^*.\mathsf{Enc}$
5 : $k_\mathsf{P} \leftarrow \mathsf{P}.\mathsf{Gen}(1^\kappa); k'_\mathsf{P} \leftarrow \mathsf{P}.\mathsf{Gen}(1^\kappa); k_\mathsf{H} \leftarrow \mathsf{H}.\mathsf{Gen}(1^\kappa)$
6 : $h := \mathsf{H}.\mathsf{Eval}_{k_\mathsf{H}}(\mathrm{lex}(D_0 \cup D_1))$
7 : $\mathsf{PKES}^*.\mathsf{Setup}(N, N_0)$
8 : $b_1, b_2, \ldots, b_L \leftarrow \mathsf{PKES}^*.\mathsf{Enc}(pk, m \,||\, k_\mathsf{H} \,||\, k_\mathsf{P} \,||\, k'_\mathsf{P} \,||\, h)$
9 : return $(\mathsf{generate}(D_0 \cup D_1, f, b_1, \ldots, b_L, k_\mathsf{P}, k'_\mathsf{P}), pk^*)$
```

Fig. 1. An overview of hybrids H_1 and H_6 used in the proof of Theorem 10. Changes between the hybrids are marked as shadowed.

Proof (Proof sketch). We prove that the above construction is secure via a *hybrid argument*. We thus define six distributions $H_1, \ldots, H_6$ shown in Figure 1.

We now proceed by proving that H_i and H_{i+1} are SS-CCA-indistinguishable (denoted by $H_i \sim H_{i+1}$). Informally, this means that we replace in $\mathsf{SS\text{-}CCA\text{-}Dist}$ the call to the stegosystem (if $b = 0$) by H_i and the call to the channel (if $b = 1$) by H_{i+1}. Denote by $\mathbf{Adv}_{\mathsf{W}}^{(i)}(\kappa)$ the advantage of a warden W in this situation. Clearly, the SS-CCA-advantage of W is bounded as $\mathbf{Adv}^{\text{ss-cca}}_{\mathsf{W},\mathsf{PKStS}^*,\mathcal{C}}(\kappa) \leq \mathbf{Adv}_{\mathsf{W}}^{(1)}(\kappa) + \mathbf{Adv}_{\mathsf{W}}^{(2)}(\kappa) + \mathbf{Adv}_{\mathsf{W}}^{(3)}(\kappa) + \mathbf{Adv}_{\mathsf{W}}^{(4)}(\kappa) + \mathbf{Adv}_{\mathsf{W}}^{(5)}(\kappa)$. This implies the theorem, as H_1 simply describes the channel and H_6 describes the stegosystem. Informally, we argue that:

1. $H_1 \sim H_2$ because a uniform random permutation on a memoryless channel does not change any probabilities;
2. $H_2 \sim H_3$ because our choice of $b_1, \ldots, b_L$ and random permutations equal the channel by Lemma 7;
3. $H_3 \sim H_4$ because P is a PRP;
4. $H_4 \sim H_5$ because P is a PRP;
5. $H_5 \sim H_6$ because PKES^* is secure due to Corollary 6 and because of Lemma 8.

□

8 An Impossibility Result

We first describe an argument for truly random channels using an infeasible assumption and then proceed to modify those channels to get rid of this. All channels will be 0-memoryless and we thus write $\mathcal{C}_{\eta,\mathsf{dl}}$ instead of $\mathcal{C}_{\mathsf{hist},\mathsf{dl}}$, if hist contains η document.

The main idea of our construction lies on the *unpredictability* of random channels. If $\mathcal{C}_\eta$ and $\mathcal{C}_{\eta+1}$ are independent and sufficiently random, we can not deduce anything about $\mathcal{C}_{\eta+1}$ before we have sampling access to it, which we only have *after* we sent the document of $\mathcal{C}_\eta$ in the standard non-look-ahead model. To be reliable, there must be enough documents in $\mathcal{C}_{\eta+1}$ continuing the already sent documents (call those documents *suitable*). To be SS-CCA-secure, the number of suitable documents in $\mathcal{C}_{\eta+1}$ must be very small to prevent replay attacks like those in Section 3. By replacing the random channels with pseudorandom ones, we can thus prove that *every* stegosystem is either unreliable or SS-CCA-insecure on one of those channels. To improve the readability, fix some stegosystem PKStS and let $n = \mathsf{PKStS.dl}(\kappa)$ and $\ell = \mathsf{PKStS.ol}(\kappa)$.

Lower Bound on Truly Random Channels. For $n \in \mathbb{N}$, we denote by $\mathcal{R}_n$ all subsets R of $\{0,1\}^n$ such that there is a negligible function negl with

- $|R| \geq \mathsf{negl}(n)^{-1}$ and
- $|R| \leq 2^{n/2}$.

This means each subset R has super-polynomial cardinality in n without being too large. For an infinite sequence $\boldsymbol{R} = R_0, R_1, \ldots$ with $R_i \in \mathcal{R}_n$, we construct a channel $\mathcal{C}(\boldsymbol{R})$ where the distribution $\mathcal{C}(\boldsymbol{R})_{i,n}$ is the uniform distribution on R_i. The family of all such channels is denoted by $\mathcal{F}(\mathcal{R}_n)$. We assume that a warden can test whether a document d belongs to the support of $\mathcal{C}(\boldsymbol{R})_{i,n}$ and denote this warden by $\mathsf{W}_{\boldsymbol{R}}$. In the next section, we replace the totally random channels by pseudorandom ones and will get rid of this infeasible assumption. For a stegosystem PKStS – like the system PKStS^* from the last section – we are now interested in two possible events that may occur during the run of $\mathsf{PKStS.Enc}$ on a channel $\mathcal{C}(\boldsymbol{R})$. The first event, denoted by $\mathcal{E}_{\mathrm{Nq}}$ (for *<u>N</u>onq<u>u</u>eried*), happens if $\mathsf{PKStS.Enc}$ outputs a document it has not seen due to sampling. We are also interested in the case that $\mathsf{PKStS.Enc}$ outputs something in the support of the channel, denoted by $\mathcal{E}_{\mathrm{InS}}$ for *<u>In</u> <u>S</u>upport*. Clearly, upon random choice of $\boldsymbol{R}$, η (the length of the history), m and pk we have

$$\Pr[\mathcal{E}_{\mathrm{InS}} \mid \mathcal{E}_{\mathrm{Nq}}] \;\leq \ell \cdot \frac{2^{n/2} - \mathsf{PKStS.query}(\kappa)}{2^n - \mathsf{PKStS.query}(\kappa)} \;\leq \ell \cdot 2^{-n/2},$$

where $\mathsf{PKStS.query}(\kappa)$ denotes the number of queries performed by PKStS. This is negligible in κ as n, query and ℓ are polynomials in κ. As warden $\mathsf{W}_{\boldsymbol{R}}$ can test whether a document belongs to the random sets, we have $\mathbf{Adv}^{\text{ss-cca}}_{\mathsf{W}_{\boldsymbol{R}},\mathsf{PKStS},\mathcal{C}(\boldsymbol{R})}(\kappa) \geq \Pr[\overline{\mathcal{E}_{\mathrm{InS}}}]$. Clearly, since we can assume $\overline{\mathcal{E}_{\mathrm{InS}}} \subseteq \mathcal{E}_{\mathrm{Nq}}$ we thus obtain

$$\Pr[\mathcal{E}_{\mathrm{Nq}}] \;=\; \frac{\Pr[\overline{\mathcal{E}_{\mathrm{InS}}} \wedge \mathcal{E}_{\mathrm{Nq}}]}{\Pr[\overline{\mathcal{E}_{\mathrm{InS}}} \mid \mathcal{E}_{\mathrm{Nq}}]} \;\leq\; \frac{\mathbf{Adv}^{\text{ss-cca}}_{\mathsf{W}_{\boldsymbol{R}},\mathsf{PKStS},\mathcal{C}(\boldsymbol{R})}(\kappa)}{1 - \ell \cdot 2^{-n/2}}.$$

Hence, if PKStS is SS-CCA-secure, the term $\Pr[\mathcal{E}_{\mathrm{Nq}}]$ must be negligible.

If PKStS is given a history of length η and it outputs documents $d_1, \ldots, d_\ell$, we note that $\mathsf{PKStS.Enc}$ only gets sampling access to $\mathcal{C}(\boldsymbol{R})_{\eta+\ell-1,n}$ after it sent $d_1, \ldots, d_{\ell-1}$ in the standard non-look-ahead model. Clearly, due to the random choice of $\boldsymbol{R}$, the set $R_{\eta+\ell}$ is independent of $R_\eta, R_{\eta+1}, \ldots, R_{\eta+\ell-1}$. The encoder $\mathsf{PKStS.Enc}$ thus needs to decide on the documents $d_1, \ldots, d_{\ell-1}$ without any knowledge of $R_{\eta+\ell}$. As $\mathsf{PKStS.Enc}$ draws a sample set D from $\mathcal{C}(\boldsymbol{R})_{\eta+\ell-1,n}$ with at most $q = \mathsf{PKStS.query}(\kappa)$ documents, we now look at the event $\mathcal{E}_{\mathrm{Nsui}}$ (for *<u>N</u>ot <u>sui</u>table*) that none of the documents in D are suitable for the encoding, i. e. if the sequence $d_1, d_2, \ldots, d_{\ell-1}, d$ is not a suitable encoding of the message m for all $d \in D$. Denote the unreliabiliy of the stegosystem by ρ. Clearly, if $\mathcal{E}_{\mathrm{Nsui}}$ occurs, there are two possibilities for the stegosystem: It either outputs something from D and thus increases the unreliability or it outputs something it has not queried. We thus have $\Pr[\mathcal{E}_{\mathrm{Nsui}}] \;\leq\; \max\{\rho, (1-\rho) \cdot \Pr[\mathcal{E}_{\mathrm{Nq}}]\}$. Note that ρ must be negligible if $\mathsf{PKStS.Enc}$ is reliable and, as discussed above, the term $\Pr[\mathcal{E}_{\mathrm{Nq}}]$ (and thus the term $(1-\rho) \cdot \Pr[\mathcal{E}_{\mathrm{Nq}}]$) must be negligible if $\mathsf{PKStS.Enc}$ is SS-CCA-secure. Hence, if $\mathsf{PKStS.Enc}$ is SS-CCA-secure and reliable, the probability $\Pr[\mathcal{E}_{\mathrm{Nsui}}]$ must be negligible. The insight, that $\Pr[\mathcal{E}_{\mathrm{Nsui}}]$ must be negligible directly leads us to the construction of a warden $\mathsf{W}_{\boldsymbol{R}}$ on the channel $\mathcal{C}(\boldsymbol{R})$. The warden chooses a random history of length η and a random message m and sends those to its challenging oracle. It then receives the document sequence $d_1, \ldots, d_\ell$. If $d_i \notin R_{\eta+i}$, the warden returns »Stego«. Else, it samples q documents D from $\mathcal{C}(\boldsymbol{R})_{\eta+\ell,n}$ and tests for all $d \in D$ via the decoding oracle $\mathsf{PKStS.Dec}_{sk}$ if the sequence $d_1, d_2, \ldots, d_{\ell-1}, d$ decodes to m. If we find such a d, return »Stego« and else return »Not Stego«. If the documents are randomly chosen from the channel, the probability to return »Stego« is at most $q/|2^{\mathsf{PKStS.ml}(\kappa)}|$, i. e. negligible. If the documents are chosen by the stegosystem, the probability of »Not Stego« is exactly $\Pr[\mathcal{E}_{\mathrm{Nsui}}]$. Hence, PKStS must be either unreliable or SS-CCA-insecure on some channel in $\mathcal{F}(\mathcal{R}_n)$.

Lower Bound on Pseudorandom Channels. To give a proof, we will replace the random channels $\mathcal{C}(\boldsymbol{R})$ by pseudorandom ones. The construction assumes existence of a CCA\$-secure cryptosystem PKES with $\mathsf{PKES.cl}(\kappa) \geq 2\,\mathsf{PKES.ml}(\kappa)$.

For $\omega = (pk, sk) \in \mathrm{supp}(\mathsf{PKES.Gen}(1^\kappa))$, let $\mathcal{C}(\omega)_{i,\mathsf{dl}(\kappa)}$ be the distribution $\mathsf{PKES.Enc}(pk, \mathrm{bin}(i)_{\mathsf{dl}(\kappa)})$, where $\mathrm{bin}(i)_{\mathsf{dl}(\kappa)}$ is the binary representation of the number i of length exactly $\mathsf{dl}(\kappa)$ modulo $2^{\mathsf{dl}(\kappa)}$. The family of channels $\boldsymbol{\mathcal{C}}_{\mathsf{PKES}} = \{\mathcal{C}(\omega)\}_\omega$ thus has the following properties:

1. There is a negligible function negl such that for each ω and each i, we have $2^{\mathsf{PKES.ml}(\kappa)/2} \geq |\,\mathcal{C}(\omega)_{i,\mathsf{dl}(\kappa)}| \geq \mathsf{negl}(\kappa)^{-1}$ if PKES is CCA\$-secure. This follows easily from the CCA\$-security of PKES: If $|\,\mathcal{C}(\omega)_{i,\mathsf{dl}(\kappa)}|$ would be polynomial, an attacker could simply store all corresponding ciphertexts.
2. An algorithm with the knowledge of ω can test in polynomial time, whether $d \in \mathrm{supp}(\mathcal{C}(\omega)_{i,\mathsf{dl}(\kappa)})$, i. e. whether d belongs to the support by simply testing whether $\mathsf{PKES.Dec}(sk, d)$ equals $\mathrm{bin}(i)_{\mathsf{dl}(\kappa)}$.
3. Every algorithm Q that tries to distinguish $\mathcal{C}(\omega)$ from a random channel $\mathcal{C}(\boldsymbol{R})$ fails: For every polynomial algorithm Q, we have that the term

$$\Big| \Pr_{\boldsymbol{R} \twoheadleftarrow \boldsymbol{\mathcal{R}}^*_{\mathsf{dl}(\kappa)}}[\mathsf{Q}^{\mathcal{C}(\boldsymbol{R})}(1^\kappa) = 1] - \Pr_{\omega \leftarrow \mathsf{PKES.Gen}(1^\kappa)}[\mathsf{Q}^{\mathcal{C}(\omega)}(1^\kappa) = 1] \Big|$$

   is negligible in κ if PKES is CCA\$-secure. This follows from the fact that no polynomial algorithm can distinguish $\mathcal{C}(\boldsymbol{R})$ upon random choice of $\boldsymbol{R}$ from the uniform distribution on $\{0,1\}^n$, as $|\,\mathcal{C}(\boldsymbol{R})_{i,n}|$ is super-polynomial. Furthermore, an attacker A on PKES can simulate Q for a successful attack.

Note that the third property directly implies that no polynomial algorithm can conclude anything about $\mathcal{C}(\omega)_{i,\mathsf{dl}(\kappa)}$ from samples of previous distributions $\mathcal{C}(\omega)_{0,\mathsf{dl}(\kappa)}, \ldots, \mathcal{C}(\omega)_{i-1,\mathsf{dl}(\kappa)}$, except for a negligible term. The second property directly implies that we can get rid of the infeasible assumption of the previous section concerning the ability of the warden to test whether a document belongs to the support of $\mathcal{C}(\omega)$: We simply equip the warden with the seed ω. Call the resulting warden W_ω. Note that this results in a non-uniform warden. As above, we are interested in the events that a stegosystem outputs a document that it has not seen ($\mathcal{E}_{\widehat{\mathrm{Nq}}}$), that a document is outputted which does not belong to the support ($\mathcal{E}_{\widehat{\mathrm{InS}}}$) and the event that a random set of q documents is not suitable to complete a given document prefix $d_1, d_2, \ldots, d_{\ell-1}$ ($\mathcal{E}_{\widehat{\mathrm{Nsui}}}$).

As $\mathcal{E}_{\widehat{\mathrm{InS}}}$ is a polynomially testable property (due to the second property of our construction), we can conclude a similar bound as above:

Lemma 11. *Let* PKStS *be an SS-CCA-secure universal stegosystem. For every warden* W *and every CCA\$-attacker* A, $\Pr[\mathcal{E}_{\widehat{\mathrm{Nq}}}] \leq \frac{\mathbf{Adv}^{\text{ss-cca}}_{\mathsf{W},\mathsf{PKStS},\mathcal{C}(\omega)}(\kappa)}{1-\ell\cdot 2^{-n/2}} + \mathbf{Adv}^{\text{pkes}}_{\mathsf{A},\mathsf{PKES}}(\kappa)$.

Hence, if the stegosystem PKStS is SS-CCA-secure and PKES is CCA\$-secure, the term $\Pr[\mathcal{E}_{\widehat{\mathrm{Nq}}}]$ must be negligible. As above, we can conclude that $\Pr[\mathcal{E}_{\widehat{\mathrm{Nsui}}}] \leq \max\{\rho, (1-\rho)\cdot\Pr[\mathcal{E}_{\widehat{\mathrm{Nq}}}]\}$ for unreliabiliy ρ. The warden W_ω similar to $\mathsf{W}_{\boldsymbol{R}}$ from the preceding section thus suceeds with very high probability. Hence, no SS-CCA-secure and reliable stegosystem can exist for the family $\boldsymbol{\mathcal{C}}_{\mathsf{PKES}}$:

Theorem 12. *If doubly-enhanced trapdoor permutations exist, for every stegosystem* PKStS *in the non-look-ahead model there is a* 0*-memoryless channel* $\mathcal{C}$ *such that* PKStS *is either unreliable or it is not SS-CCA-secure on* $\mathcal{C}$ *against non-uniform wardens.*

9 Discusssion

The work of Dedić et al. [13] shows that provable secure universal steganography needs a huge number of sample documents to embed long secret messages as high bandwidth stegosystems are needed for such messages. This limitation also transfers to the public-key scenario. However, such a limitation does not necessarily restrict applicability of steganography, especially in case of specific communication channels or if the length of secret messages is sufficiently short.

A prominent recent example for such applications is the use of steganography for channels determined by cryptographic primitives, like symmetric encryption schemes (SESs) or digital signature schemes. Bellare, Paterson, and Rogaway introduced in [5] so called *algorithm substitution attacks* against SESs, where an attacker replaces an honest implementation of the encryption algorithm by a modified version which allows to extract the secret key from the ciphertexts produced by the corrupted implementation. Several follow-up works have been done based on this paper, such as those by Bellare, Jaeger, and Kane [4], Ateniese, Magri, and Ventur [2], or Degabriele, Farshim, and Poettering [14]. These works strengthened the model proposed in [5] and presented new attacks against SESs or against other cryptographic primitives, e. g. against signature schemes. Surprisingly, as we show in [6], all such algorithm substitution attacks can be analyzed in the framework of computational secret-key steganography and in consequence, the attackers can be identified as stegosystems on certain channels determined by the primitives. In such scenarios, the secret message embedded by the stegosystem corresponds to a secret key of the cryptographic algorithm.

A similar approach was used by Pasquini, Schöttle, and Böhme [35] to show that so called *password decoy vaults* used for example by Chatterjee, Bonneau, Juels, and Ristenpart [10] and Golla, Beuscher, and Dürmuth [19] can also be interpreted as steganographic protocols.

References

1. Ross J Anderson and Fabien AP Petitcolas. On the limits of steganography. *Selected Areas in Communications, IEEE Journal on*, 16(4):474–481, 1998.
2. Giuseppe Ateniese, Bernardo Magri, and Daniele Venturi. Subversion-resilient signature schemes. In *Proc. CCS*, pages 364–375. ACM, 2015.
3. Michael Backes and Christian Cachin. Public-key steganography with active attacks. In *Proc. TCC*, volume 3378 of *Lecture Notes in Computer Science*, pages 210–226. Springer, 2005.
4. Mihir Bellare, Joseph Jaeger, and Daniel Kane. Mass-surveillance without the state: Strongly undetectable algorithm-substitution attacks. In *Proc. CCS 2015*, pages 1431–1440. ACM, 2015.
5. Mihir Bellare, Kenneth G. Paterson, and Phillip Rogaway. Security of symmetric encryption against mass surveillance. In *Proc. CRYPTO 2014*, volume 8616 of *Lecture Notes in Computer Science*, pages 1–19, 2014.
6. Sebastian Berndt and Maciej Liśkiewicz. Algorithm substitution attacks from a steganographic perspective. In *Proc. CCS*, pages 1649–1660, 2017. URL: `http://doi.acm.org/10.1145/3133956.3133981`, `doi:10.1145/3133956.3133981`.
7. Christian Cachin. An information-theoretic model for steganography. *Information and Computation*, 192(1):41–56, 2004.
8. Ran Canetti, Hugo Krawczyk, and Jesper Buus Nielsen. Relaxing chosen-ciphertext security. In *Proc. CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 565–582. Springer, 2003.
9. Nishanth Chandran, Vipul Goyal, Rafail Ostrovsky, and Amit Sahai. Covert multi-party computation. In *Proc. FOCS*, pages 238–248. IEEE Computer Society, 2007.
10. Rahul Chatterjee, Joseph Bonneau, Ari Juels, and Thomas Ristenpart. Cracking-resistant password vaults using natural language encoders. In *Proc. S&P*, pages 481–498, 2015. URL: `https://doi.org/10.1109/SP.2015.36`, `doi:10.1109/SP.2015.36`.
11. Chongwon Cho, Dana Dachman-Soled, and Stanislaw Jarecki. Efficient concurrent covert computation of string equality and set intersection. In *Proc. CT-RSA*, volume 9610 of *Lecture Notes in Computer Science*, pages 164–179. Springer, 2016.
12. Scott Craver. On public-key steganography in the presence of an active warden. In *Proc. Information Hiding*, pages 355–368. Springer, 1998.
13. Nenad Dedić, Gene Itkis, Leonid Reyzin, and Scott Russell. Upper and lower bounds on black-box steganography. *Journal of Cryptology*, 22(3):365–394, 2009.
14. Jean Paul Degabriele, Pooya Farshim, and Bertram Poettering. A more cautious approach to security against mass surveillance. In *Proc. FSE*, volume 9054 of *Lecture Notes in Computer Science*, pages 579–598. Springer, 2015.
15. Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *Information Theory, IEEE Transactions on*, 22(6):644–654, 1976.
16. Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000.
17. Nelly Fazio, Antonio Nicolosi, and Irippuge Milinda Perera. Broadcast steganography. In *Proc. CT-RSA*, volume 8366 of *Lecture Notes in Computer Science*, pages 64–84. Springer, 2014.
18. Oded Goldreich and Ron D. Rothblum. Enhancements of trapdoor permutations. *Journal of cryptology*, 26(3):484–512, 2013.

19. Maximilian Golla, Benedict Beuscher, and Markus Dürmuth. On the security of cracking-resistant password vaults. In *Proc. CCS*, pages 1230–1241, 2016. URL: `http://doi.acm.org/10.1145/2976749.2978416`, `doi:10.1145/2976749.2978416`.
20. Dennis Hofheinz, Vanishree Rao, and Daniel Wichs. Standard security does not imply indistinguishability under selective opening. In *Proc. TCC (B2)*, volume 9986 of *Lecture Notes in Computer Science*, pages 121–145, 2016.
21. Nicholas Hopper. On steganographic chosen covertext security. In *Proc. ICALP*, volume 3580 of *Lecture Notes in Computer Science*, pages 311–323. Springer, 2005.
22. Nicholas J. Hopper, John Langford, and Luis Ahn. Provably secure steganography. In *Proc. CRYPTO*, volume 2442 of *LNCS*, pages 77–92. Springer Berlin Heidelberg, 2002.
23. Nicholas J. Hopper, Luis von Ahn, and John Langford. Provably secure steganography. *Computers, IEEE Transactions on*, 58(5):662–676, 2009.
24. Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography, Second Edition*. CRC Press, 2014.
25. Stefan Katzenbeisser and Fabien A.P. Petitcolas. Defining security in steganographic systems. In *Proc. Electronic Imaging*, pages 50–56. SPIE, 2002.
26. Aggelos Kiayias, Yona Raekow, Alexander Russell, and Narasimha Shashidhar. A one-time stegosystem and applications to efficient covert communication. *Journal of Cryptology*, 27(1):23–44, 2014.
27. Eike Kiltz, Payman Mohassel, and Adam O'Neill. Adaptive trapdoor functions and chosen-ciphertext security. In *Proc. EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 673–692. Springer, 2010.
28. Tri Van Le and Kaoru Kurosawa. Bandwidth optimal steganography secure against adaptive chosen stegotext attacks. In *Proc. Information Hiding*, volume 4437 of *Lecture Notes in Computer Science*, pages 297–313. Springer, 2006.
29. Yehuda Lindell. A simpler construction of cca2-secure public-key encryption under general assumptions. In *Proc. EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*, pages 241–254. Springer, 2003.
30. Maciej Liśkiewicz, Rüdiger Reischuk, and Ulrich Wölfel. Grey-box steganography. *Theoretical Computer Science*, 505:27–41, 2013.
31. Michael Luby and Charles Rackoff. How to construct pseudo-random permutations from pseudo-random functions (abstract). In *Proc. CRYPTO*, volume 218 of *Lecture Notes in Computer Science*, page 447. Springer, 1985.
32. Anna Lysyanskaya and Mira Meyerovich. Provably secure steganography with imperfect sampling. In *Proc. PKC*, volume 3958 of *Lecture Notes in Computer Science*, pages 123–139. Springer, 2006.
33. Michael Mitzenmacher and Eli Upfal. *Probability and computing - randomized algorithms and probabilistic analysis*. Cambridge University Press, 2005.
34. Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *Proc. STOC*, pages 33–43. ACM, 1989.
35. Cecilia Pasquini, Pascal Schöttle, and Rainer Böhme. Decoy password vaults: At least as hard as steganography? In *Proc. SEC*, pages 356–370, 2017. `doi:10.1007/978-3-319-58469-0_24`.
36. Boris Ryabko and Daniil Ryabko. Constructing perfect steganographic systems. *Information and Computation*, 209(9):1223–1230, 2011.
37. Daniel R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In *Proc. EUROCRYPT*, volume 1403 of *Lecture Notes in Computer Science*, pages 334–345. Springer, 1998.

38. Luis von Ahn and Nicholas J. Hopper. Public key steganography. *IACR Cryptology ePrint Archive*, 2003/233, 2003.
39. Luis von Ahn and Nicholas J. Hopper. Public-key steganography. In *Proc. EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 323–341. Springer, 2004.
40. Ying Wang and Pierre Moulin. Perfectly secure steganography: Capacity, error exponents, and code constructions. *Information Theory, IEEE Transactions on*, 54(6):2706–2722, 2008.

A Remaining Proofs

To improve the readability, we will abbreviate some terms and define $n = \mathsf{PKStS}^*.\mathsf{dl}(\kappa)$, $\ell = \mathsf{PKStS}^*.\mathsf{ol}(\kappa)$ and $L = \mathsf{PKES}^*.\mathsf{cl}(\kappa)$, where PKStS^* is our stegosystem constructed in Section 7 and PKES^* is the public-key cryptosystem constructed in Section 5. We also define $N = 8L$.

A.1 Formal Statement of Lemma 8 and its Proof

We start with a formal definition for "A is successful on $D, f, b_1, \ldots, b_L, k_{\mathsf{H}}$".

Definition 13. *An attacker* A *on* generate *is a PPTM, that receives the following input:*
- *a sequence $d_1, \ldots, d_N$ of N pairwise different documents*
- *a hash function $f : \Sigma^n \to \{0,1\}$ from the family $\mathcal{G} = \{G_\kappa\}_{\kappa\in\mathbb{N}}$,*
- *a sequence $b_1, \ldots, b_L$ of L bits, and*
- *a hash-key k_{H} for H.*

The attacker A *then outputs a sequence $d'_1, \ldots, d'_N$ of documents. Note that the attacker knows the mapping function f and even the hash-key k_H for* H.

We say that A *is* successful *if the experiment* $\mathrm{Sgen}(\mathsf{A}, D, f, b_1, \ldots, b_L)$ *returns value* 1*:*

```
Security of generate: $\mathrm{Sgen}(\mathsf{A}, D, f, b_1, \ldots, b_L)$

Input: Attacker $\mathsf{A}$, set $D$, function $f$, bits $b_1, \ldots, b_L$
 1: $k_{\mathsf{P}}, k'_{\mathsf{P}} \leftarrow \mathsf{P.Gen}(1^\kappa)$
 2: $k_{\mathsf{H}} \leftarrow \mathsf{H.Gen}(1^\kappa)$
 3: $d_1, \ldots, d_N := \mathsf{generate}(D, f, b_1, \ldots, b_L, k_{\mathsf{P}}, k'_{\mathsf{P}})$
 4: $d'_1, \ldots, d'_N \leftarrow \mathsf{A}(d_1, \ldots, d_N, f, b_1, \ldots, b_L, k_{\mathsf{H}})$
 5: if $f(d'_i) = b_i$ for every $i = 1, \ldots L$ then
 6:     $D'_0 = \{d'_j \mid f(d'_j) = 0\}$; $D'_1 = \{d'_j \mid f(d'_j) = 1\}$
 7:     if $d'_1, \ldots, d'_N = \mathsf{generate}(D'_0 \cup D'_1, f, b_1, \ldots, b_L, k_{\mathsf{P}}, k'_{\mathsf{P}})$ then
 8:         if $\mathsf{H.Eval}_{k_{\mathsf{H}}}(\mathrm{lex}(D'_0 \cup D'_1)) = \mathsf{H.Eval}_{k_{\mathsf{H}}}(\mathrm{lex}(D_0 \cup D_1))$ then
 9:             if $d'_1, \ldots, d'_N \neq d_1, \ldots, d_N$ then
10:                 return 1 and halt
11: return 0
```

We are now ready to give the formal version of Lemma 8:

Lemma (formal version of Lemma 8). *Let $D \subsetneq \Sigma^n$ be a set of documents, with $|D| = N$, let $b_1, \dots, b_L$ be a bitstring, and $f \in G_\kappa$. For every attacker* A *on* generate*, there is a collision finder* Fi *for the CRHF* H *such that*

$$\Pr[\mathrm{Sgen}(\mathsf{A}, D, f, b_1, \dots, b_L) = 1] \leq \mathbf{Adv}^{\mathrm{hash}}_{\mathsf{Fi},\mathsf{H},\mathcal{C}}(\kappa),$$

where the probability is taken over the random choices made in experiment Sgen.

Proof. Let A be an attacker on generate with maximal success probability. Let $D = D_0 \dot{\cup} D_1$ be the input to generate, the sequence $d_1, \dots, d_N$ its output and $d'_1, \dots, d'_N$ be the output of A. Furthermore, let $D'_b = \{d'_j \mid f(d'_j) = b\}$ and $D' = D'_0 \cup D'_1$. We now distinguish three cases of the relation between D and D'. If $D' \subsetneq D$, the sequence $d'_1, \dots, d'_N$ must contain the same element on at least two positions, but generate does only accept sets of size exactly N. Hence, A was not successful in this case. If $D' = D$ and A was successful, it holds that $d'_1, \dots, d'_N \neq d_1, \dots, d_N$. Hence, there must be positions $i < j$ and $j' < i'$ such that $d_i = d_{i'}$ and $d_j = d_{j'}$. As k_P and k'_P define a total order, the sequence $d'_1, \dots, d'_N$ could not be produced by generate. Thus, A can not be successful in this case.

The only remaining case is $D' \setminus D \neq \emptyset$. If A was successful, it holds that $\mathsf{H}_{k_\mathsf{H}}(\mathrm{lex}(D')) = \mathsf{H}_{k_\mathsf{H}}(\mathrm{lex}(D))$, i.e. this is a collision with regard to H. We will now construct a finder Fi for H, such that $\mathbf{Adv}^{\mathrm{hash}}_{\mathsf{Fi},\mathsf{H},\mathcal{C}}(\kappa) \geq \Pr[\mathsf{A} \text{ succeeds}]$. The finder Fi receives a hash key k_H. It then chooses $f \twoheadleftarrow G_\kappa$, samples D documents of cardinality $|D| = N$ via rejection sampling and PRP-keys $k_\mathsf{P}, k'_\mathsf{P}$. The finder simulates A and receives

$$d'_1, \dots, d'_N \leftarrow \mathsf{A}(\mathsf{generate}(D, f, b_1, \dots, b_L, k_\mathsf{P}, k'_\mathsf{P}), f, b_1, \dots, b_L, k_\mathsf{H}).$$

Then, it returns D and $D' = \{d'_1, \dots, d'_N\}$. Whenever A succeeds, we have $D \neq D'$ by the discussion above and thus also $\mathsf{H}_{k_\mathsf{H}}(\mathrm{lex}(D)) = \mathsf{H}_{k_\mathsf{H}}(\mathrm{lex}(D'))$. Hence, Fi has successfully found a collision. This implies that $\mathbf{Adv}^{\mathrm{hash}}_{\mathsf{Fi},\mathsf{H},\mathcal{C}}(\kappa) \geq \Pr[\mathsf{A} \text{ succeeds}]$. □

A.2 Proof of Theorem 9

Recall the statement of the theorem:

Theorem (Theorem 9). *The probability that a message is not correctly embedded by* $\mathsf{PKStS}^*.\mathsf{Enc}$ *is at most* $3N^2 \cdot 2^{-H_\infty(\mathcal{C},\kappa)} + 2\exp(-N/54)$.

Proof. Note that $\mathsf{PKStS}^*.\mathsf{Enc}$ may not correctly embed a message m if (a) $|D_0 \cup D_1| < N$ i.e. a document sampled in line 3 was drawn twice, or (b) $N_0/N \notin [1/3, 2/3]$ i.e. the bias is too large, or (c) the number of elements of D_0 or D_1 is too small to embed $b_1, b_2, \dots, b_L$ by generate. The probability of (a) can be bounded similar to the birthday attack. It is roughly bounded by $3N^2 \cdot 2^{-H_\infty(\mathcal{C},\kappa)}$ as the probability of every document is bounded by $2^{-H_\infty(\mathcal{C},\kappa)}$.

A simple calculation shows that the probability of (b) and (c) is negligible. Note that the algorithm always correctly embeds a message, if $|D_0| \geq L$ and $|D_1| \geq L$. As $N_0/N = |D_0|/N$, this implies that $N_0/N \in [1/3, 2/3]$. We will thus estimate the probability for this. As f is drawn from a strongly 2-universal hash family, we note that the probability that a random document d is mapped to 1 is equal to 1/2. For $i = 1, \ldots, N$, let X_i be the indicator variable such that X_i equals 1 if the i-th element drawn from the channel maps to 1. The random variable $X = \sum_{i=1}^{N} X_i$ thus has the size of D_1. Clearly, its expected value is $N/2$. The probability that $|X - N/2| > L$ (and thus $|D_1| < L$ or $|D_0| < L$) is hence bounded by

$$\Pr[|X - N/2| > L] \leq 2\exp(-\frac{L \cdot (1/3)^2}{3}) = 2\exp(-N/54)$$

using a Chernoff-like bound. The probability that the message m is incorrectly embedded is thus bounded by $2^{-H_\infty(\mathcal{C},\kappa)} + 2\exp(-N/54)$. $\square$

A.3 Proof of Theorem 10

We recall:

Theorem (Theorem 10). *Let $\mathcal{C}$ be a memoryless channel,* P *be a PRP relative to $\mathcal{C}$, the algorithm* H *be a CRHF relative to $\mathcal{C}$, the cryptosystem* PKES^* *be the cryptosystem designed in Section 5 with very sparse support relative to $\mathcal{C}$, and $\mathcal{G}$ be a strongly 2-universal hash family. The stegosystem* PKStS^* *is SS-CCA-secure against* every *memoryless channel.*

Proof. We prove that the above construction is secure via a *hybrid argument*. We thus define six distributions $H_1, \ldots, H_6$ shown in Figure 1.

If P and Q are two probability distributions, denote by $\mathsf{SS\text{-}CCA\text{-}Dist}_{P,Q}$ the modification of the game $\mathsf{SS\text{-}CCA\text{-}Dist}$, where the call to the stegosystem (if $b = 0$) is replaced by a call to P and the call to the channel (if $b = 1$) is replaced by a call to Q. If W is some warden, denote by $\mathbf{Adv}^{\text{ss-cca}}_{\mathsf{W},P,Q}(\kappa)$ the winning probability of W in $\mathsf{SS\text{-}CCA\text{-}Dist}_{P,Q}$. If $\mathbf{Adv}^{\text{ss-cca}}_{\mathsf{W},P,Q}(\kappa) \leq \mathsf{negl}(\kappa)$ for a negligible function negl, we denote this situation as $P \sim Q$ and say that P and Q are *indistinguishable* with respect to $\mathsf{SS\text{-}CCA\text{-}Dist}$. Furthermore, we define $\mathbf{Adv}^{(i)}_{\mathsf{W}}(\kappa) = \mathbf{Adv}^{\text{ss-cca}}_{\mathsf{W},H_i,H_{i+1}}(\kappa)$. As the term $\mathbf{Adv}^{(i)}_{\mathsf{W}}(\kappa)$ can also be written as

$$\big|\Pr[\mathsf{W.Guess}\text{ outputs } b' = 0 \mid b = 0] - \Pr[\mathsf{W.Guess}\text{ outputs } b' = 0 \mid b = 1]\big|,$$

the triangle inequality implies that $\mathbf{Adv}^{\text{ss-cca}}_{\mathsf{W},\mathsf{PKStS}^*,\mathcal{C}}(\kappa) \leq \mathbf{Adv}^{(1)}_{\mathsf{W}}(\kappa) + \mathbf{Adv}^{(2)}_{\mathsf{W}}(\kappa) + \mathbf{Adv}^{(3)}_{\mathsf{W}}(\kappa) + \mathbf{Adv}^{(4)}_{\mathsf{W}}(\kappa) + \mathbf{Adv}^{(5)}_{\mathsf{W}}(\kappa)$.

Informally, we argue that:

1. $H_1 = H_2 \implies H_1 \sim H_2$ because a uniform random permutation on a memoryless channel does not change any probabilities;

2. $H_2 = H_3 \implies H_2 \sim H_3$ because our choice of $b_1, \ldots, b_L$ and random permutations equal the channel by Lemma 7;
3. $H_3 \sim H_4$ because P is a PRP;
4. $H_4 \sim H_5$ because P is a PRP;
5. $H_5 \sim H_6$ PKES^* is secure due to Corollary 6 and because of Lemma 8.

Distribution H_1 can be specified as follows:

```
H_1 = $\mathcal{C}_n^N$
1 : $pk^* = (pk, f) \leftarrow \mathsf{PKStS}^*.\mathsf{Gen}(1^\kappa)$
2 : for $j := 1, 2, \ldots, N$ :
3 :     $d_j \leftarrow \mathcal{C}_{\mathsf{dl}(\kappa)}$
4 : return $((d_1, \ldots, d_N), pk^*)$
```

Indistinguishability of H_1 and

```
H_2
$pk^* = (pk, f) \leftarrow \mathsf{PKStS}^*.\mathsf{Gen}(1^\kappa)$
Lines 1 to 4 in $\mathsf{PKStS}^*.\mathsf{Enc}$
5 : $P \twoheadleftarrow \mathsf{Perms}$
6 : return $((d_{P(1)}, \ldots, d_{P(N)}), pk^*)$
```

If $|D_0 \cup D_1| < N$, i. e. a document was sampled twice or $|D_0|/|D| \notin [1/3, 2/3]$, the system only outputs the sampled documents. Hence H_1 equals H_2 in this case. In the other case, we first permute the items before we output them. But, as P is a uniform random permutation and the documents are drawn independently from a memoryless channel, we have

$$\Pr_{H_1}[d_1, \ldots, d_N \text{ are drawn}] = \Pr_{H_1}[d_{P(1)}, \ldots, d_{P(N)} \text{ are drawn}].$$

As pk is not used in these hybrids, $H_1 = H_2$ follows.

Indistinguishability of H_2 and

```
H_3
$pk^* = (pk, f) \leftarrow \mathsf{PKStS}^*.\mathsf{Gen}(1^\kappa)$
Lines 1 to 4 in $\mathsf{PKStS}^*.\mathsf{Enc}$
5 : $P \twoheadleftarrow \mathsf{Perms}; P' \twoheadleftarrow \mathsf{Perms}; k_{\mathsf{H}} \leftarrow \mathsf{H.Gen}(1^\kappa)$
6 : $b_1, b_2, \ldots, b_L \leftarrow D^*_{(N, N_0, L)}$
7 : return $(\mathsf{generate}(D_0 \cup D_1, f, b_1, \ldots, b_L, P, P'), pk^*)$
// $\mathsf{generate}(\ldots, P, P')$ uses the permutations $P, P'$
```

If $|D_0 \cup D_1| < N$, i. e. a document was sampled twice or $|D_0|/|D| \notin [1/3, 2/3]$, the system only outputs the sampled documents. Hence H_2 equals H_3 in this case. If $|D_0 \cup D_1| = N$, Lemma 7 shows that H_2 equals H_3.

Indistinguishability of H_3 and

```
H_4
$pk^* = (pk, f) \leftarrow \mathsf{PKStS}^*.\mathsf{Gen}(1^\kappa)$
Lines 1 to 4 in $\mathsf{PKStS}^*.\mathsf{Enc}$
5 :  $k_\mathsf{P} \leftarrow \mathsf{P.Gen}(1^\kappa); P' \twoheadleftarrow \mathsf{Perms}; k_\mathsf{H} \leftarrow \mathsf{H.Gen}(1^\kappa)$
6 :  $b_1, b_2, \ldots, b_L \twoheadleftarrow D^*_{(N,N_0,L)}$
7 :  return $(\mathsf{generate}(D_0 \cup D_1, f, b_1, \ldots, b_L, k_\mathsf{P}, P'), pk^*)$
// $\mathsf{generate}(\ldots, P')$ uses the permutation $P'$
```

We will construct a distinguisher Dist on the PRP P with $\mathbf{Adv}^{\mathrm{prp}}_{\mathsf{Dist},\mathsf{P},\mathcal{C}}(\kappa) = \mathbf{Adv}^{(3)}_{\mathsf{W}}(\kappa)$. Note that such a distinguisher has access to an oracle that either corresponds to a truly random permutation or to $\mathsf{P.Eval}_k$ for a key $k \leftarrow \mathsf{P.Gen}(1^\kappa)$.
The PRP-distinguisher Dist simulates the run of W. It first chooses a key-pair $(pk, sk) \leftarrow \mathsf{PKStS}^*.\mathsf{Gen}(1^\kappa)$. It then simulates W. Whenever the warden W makes a call to its decoding-oracle $\mathsf{PKStS}^*.\mathsf{Dec}$, it computes $\mathsf{PKStS}^*.\mathsf{Dec}(sk, \cdot)$ (or $\perp$ if necessary). In order to generate the challenge sequence $\hat{d}$ upon the message m, it simulates the run of $\mathsf{PKStS}^*.\mathsf{Enc}$ and replaces every call to P or $\mathsf{P.Eval}_{k_\mathsf{P}}$ by a call to its oracle. Similarly, the bits output by $\mathsf{PKES}^*.\mathsf{Enc}(pk, m)$ are ignored and replaced by truly random bits distributed according to $D^*_{(N,|D_0|,L)}$. If the oracle is a truly random permutation, the simulation yields exactly H_3 and if the oracle equals $\mathsf{P.Eval}_k$ for a certain key k, the simulation yields H_4. The advantage of Dist is thus exactly $\mathbf{Adv}^{(3)}_{\mathsf{W}}(\kappa)$. As P is a secure PRP, this advantage is negligible and H_3 and H_4 are thus indistinguishable.

Indistinguishability of H_4 and

```
H_5
$pk^* = (pk, f) \leftarrow \mathsf{PKStS}^*.\mathsf{Gen}(1^\kappa)$
Lines 1 to 4 in $\mathsf{PKStS}^*.\mathsf{Enc}$
5 :  $k_\mathsf{P} \leftarrow \mathsf{P.Gen}(1^\kappa); k'_\mathsf{P} \leftarrow \mathsf{P.Gen}(1^\kappa); k_\mathsf{H} \leftarrow \mathsf{H.Gen}(1^\kappa)$
6 :  $b_1, b_2, \ldots, b_L \twoheadleftarrow D^*_{(N,N_0,L)}$
7 :  return $(\mathsf{generate}(D_0 \cup D_1, f, b_1, \ldots, b_L, k_\mathsf{P}, k'_\mathsf{P}), pk^*)$
```

We will construct a distinguisher Dist on the PRP P with $\mathbf{Adv}^{\mathrm{prp}}_{\mathsf{Dist},\mathsf{P},\mathcal{C}}(\kappa) = \mathbf{Adv}^{(4)}_{\mathsf{W}}(\kappa)$. Note that such a distinguisher has access to an oracle that either corresponds to a truly random permutation or to $\mathsf{P.Eval}_k$ for a key $k \leftarrow \mathsf{P.Gen}(1^\kappa)$.
The PRP-distinguisher Dist simulates the run of W. It first chooses a key-pair $(pk, sk) \leftarrow \mathsf{PKStS}^*.\mathsf{Gen}(1^\kappa)$ and a key $k_\mathsf{P} \leftarrow \mathsf{P.Gen}(1^\kappa)$ for the PRP P. It then simulates W. Whenever the warden W makes a call to its decoding-oracle $\mathsf{PKStS}^*.\mathsf{Dec}$, it computes $\mathsf{PKStS}^*.\mathsf{Dec}(sk, \cdot)$ (or $\perp$ if necessary). In order to

generate the challenge sequence $\hat{d}$ upon the message m, it simulates the run of $\mathsf{PKStS}^*.\mathsf{Enc}$ and replaces every call to P' or $\mathsf{P.Eval}_{k_\mathsf{P}}$ by a call to its oracle. Similarly, the bits output by $\mathsf{PKES}^*.\mathsf{Enc}(pk, m)$ are ignored and replaced by truly random bits distributed according to $D^*_{(N,|D_0|,L)}$. If the oracle is a truly random permutation, the simulation yields exactly H_4 and if the oracle equals $\mathsf{P.Eval}_k$ for a certain key k, the simulation yields H_5. The advantage of Dist is thus exactly $\mathbf{Adv}^{(4)}_{\mathsf{W}}(\kappa)$. As P is a secure PRP, this advantage is negligible and H_4 and H_5 are thus indistinguishable.

Indistinguishability of H_5 and

```
H_6 = $\mathsf{PKStS}^*.\mathsf{Enc}$
$pk^* = (pk, f) \leftarrow \mathsf{PKStS}^*.\mathsf{Gen}(1^\kappa)$
Lines 1 to 4 in $\mathsf{PKStS}^*.\mathsf{Enc}$
5 : $k_\mathsf{P} \leftarrow \mathsf{P.Gen}(1^\kappa); k'_\mathsf{P} \leftarrow \mathsf{P.Gen}(1^\kappa); k_\mathsf{H} \leftarrow \mathsf{H.Gen}(1^\kappa)$
6 : $h := \mathsf{H.Eval}_{k_\mathsf{H}}(\mathrm{lex}(D_0 \cup D_1))$
7 : $\mathsf{PKES}^*\mathsf{.Setup}(N, N_0)$
8 : $b_1, b_2, \ldots, b_L \leftarrow \mathsf{PKES}^*.\mathsf{Enc}(pk, m \,||\, k_\mathsf{H} \,||\, k_\mathsf{P} \,||\, k'_\mathsf{P} \,||\, h)$
9 : return $(\mathsf{generate}(D_0 \cup D_1, f, b_1, \ldots, b_L, k_\mathsf{P}, k'_\mathsf{P}), pk^*)$
```

We construct an attacker A on PKES^* such that there is a negligible function negl with $\mathbf{Adv}^{\mathrm{cca}}_{\mathsf{A},\mathsf{PKES}^*,\mathcal{C}}(\kappa) + \mathsf{negl}(\kappa) \geq \mathbf{Adv}^{(5)}_{\mathsf{W}}(\kappa)$. Note that such an attacker A has access to the decryption-oracle $\mathsf{PKES}^*.\mathsf{Dec}_{sk}(\cdot)$.
The attacker A simply simulates W. First, it chooses $f \twoheadleftarrow G_\kappa$. Whenever W uses its decryption-oracle to decrypt $d_1, \ldots, d_N$, the attacker A simulates $\mathsf{PKStS}^*.\mathsf{Dec}(d_1, \ldots, d_N)$ and uses its own decryption-oracle $\mathsf{PKES}^*.\mathsf{Dec}_{sk}(\cdot)$ in this. When W outputs the challenge m, the attacker A chooses all of the parameters $D_0, D_1, k_\mathsf{H}, k_\mathsf{P}, k'_\mathsf{P}$ as in $\mathsf{PKStS}^*.\mathsf{Enc}$ and chooses its own challenge $\widetilde{m} := m \,||\, k_\mathsf{H} \,||\, k_\mathsf{P} \,||\, k'_\mathsf{P} \,||\, h$, where $h = \mathsf{H.Eval}_{k_\mathsf{H}}(D_0 \cup D_1)$.
The attacker now either receives $\boldsymbol{b} \leftarrow \mathsf{PKES}^*.\mathsf{Enc}(pk, \widetilde{m})$ or L random bits $\boldsymbol{b}$ from $D^*_{(N,|D_0|,L)}$ and computes

$$d_1, \ldots, d_N = \mathsf{generate}(D_0 \cup D_1, f, b_1, \ldots, b_L, k_\mathsf{P}, k'_\mathsf{P}).$$

If the bits correspond to $\mathsf{PKES}^*.\mathsf{Enc}(pk, \widetilde{m})$, this simulates the stegosystem and thus H_6 perfectly. If the bits are random, this equals H_5.
After the challenge is determined, A continues to simulate W. Whenever W uses its decryption-oracle to decrypt $d_1, \ldots, d_N$, it behaves as above. There is now a significant difference to the pre-challenge situation: The attacker A is not allowed to decrypt the bits $\boldsymbol{b} = b_1, \ldots, b_L$. Hence, when W tries to decrypt documents $d_1, \ldots, d_N$ such that $f(d_i) = b_i$, it has no way to use its decryption-oracle and must simply return $\perp$. Suppose that this situation arises. Note that the decryption-oracle of W would only return a

message not equal to $\perp$ then iff $d_1, \ldots, d_N = \mathsf{generate}(D_0 \cup D_1, f, \boldsymbol{b}, k_{\mathsf{P}}, k'_{\mathsf{P}})$ and $\mathsf{H.Eval}_{k_{\mathsf{H}}}(\{d_1, \ldots, d_N\}) = h$.
If $\boldsymbol{b}$ is a truly random string from $D^*_{(N,|D_0|,L)}$, the sparsity of PKES^* implies that the probability that $\boldsymbol{b}$ is a valid encoding is negligible. Hence the probability that the decryption-oracle of W would return a message not equal to $\perp$ is negligible. It only remains to prove that the probability that the decryption-oracle of W returns a message not equal to $\perp$ is negligible if $\boldsymbol{b}$ is a valid encryption of a message. But Lemma 8 states just that. We thus have $\mathbf{Adv}^{\mathrm{cca}}_{\mathsf{A},\mathsf{PKES}^*,\mathcal{C}}(\kappa) + \mathsf{negl}(\kappa) \geq \mathbf{Adv}^{(5)}_{\mathsf{W}}(\kappa)$. As the system PKES^* is CCA-secure by Corollary 6, this advantage is negligible. Hence, H_5 and H_6 are indistinguishable.

Hence, the stegosystem PKStS^* is SS-CCA-secure on $\mathcal{C}$. $\square$