Anamorphic Encryption: Private Communication against a Dictator

Giuseppe Persiano* Duong Hieu Phan† Moti Yung‡

May 29, 2022

Abstract

Cryptosystems have been developed over the years under the typical prevalent setting which assumes that the receiver's key is kept secure from the adversary, and that the choice of the message to be sent is freely performed by the sender and is kept secure from the adversary as well. Under these fundamental and basic operational assumptions, modern Cryptography has flourished over the last half a century or so, with amazing achievements: New systems (including public-key Cryptography), beautiful and useful models (including security definitions such as semantic security), and new primitives (such as zero-knowledge proofs) have been developed. Furthermore, these fundamental achievements have been translated into actual working systems, and span many of the daily human activities over the Internet.

However, in recent years, there is an overgrowing pressure from many governments to allow the government itself access to keys and messages of encryption systems (under various names: escrow encryption, emergency access, communication decency acts, etc.). Numerous non-direct arguments against such policies have been raised, such as "the bad guys can utilize other encryption system" so all other cryptosystems have to be declared illegal, or that "allowing the government access is an ill-advised policy since it creates a natural weak systems security point, which may attract others (to masquerade as the government)." It has remained a fundamental open issue, though, to show directly that the above mentioned efforts by a government (called here "a dictator" for brevity) which mandate breaking of the basic operational assumption (and disallowing other cryptosystems), is, in fact, a futile exercise. This is a direct technical point which needs to be made and has not been made to date.

In this work, as a technical demonstration of the futility of the dictator's demands, we invent the notion of "*Anamorphic Encryption*" which shows that even if the dictator gets the keys and the messages used in the system (before anything is sent) and no other system is allowed, there is a covert way within the context of **well established public-key cryptosystems** for an entity to immediately (with no latency) send piggybacked secure messages which are, in spite of the stringent dictator conditions, hidden from the dictator itself! We feel that this may be an important direct technical argument against the nature of governments' attempts to police the use of strong cryptographic systems, and we hope to stimulate further works in this direction.

1 Introduction

Cryptography, like most scientific fields, has a profound impact on our Society, and even more so as it touches upon one of the most basic human rights, the right to privacy. The threats to privacy posed by the increased reliance on electronic forms of communication has been very well identified (see, for example,[DL]) and there has been a very vigorous debate between technologists and politicians about the ways of limiting the power of encryption as a safeguard to privacy (see the discussion on escrow systems below). One of the beneficial effects of the debate has been the increased awareness of the need to protect our privacy; this is witnessed by the growing use of end to end encryption (E2E encryption) in, for example, messaging apps [MP16, Wha20].

*Università di Salerno, Salerno, Italy. giuper@gmail.com

†Telecom Paris, Institut Polytechnique de Paris, France. hieu.phan@telecom-paris.fr

‡Google LLC, and Columbia University, USA. motiyung@gmail.com

Cryptography "rearranges power"[Rog15] and it obviously attracts the attention of the institutions and individuals that hold the same power that Cryptography threatens to rearrange. Among these power holding institutions, our main concerns involve the institutions that have the power to undermine the two assumptions on which Cryptography relies. As we argue below these assumptions, which have long gone overlooked (or implicit), are rather different in nature from the usual cryptographic assumptions (e.g., about the computing power of parties). More precisely, the security guarantees offered by Cryptography even for its most basic and classic setting, two-party private communication [Sha49], rely on two implicit and fundamental assumptions, one regarding the sender and one regarding the receiver, that can be challenged by exactly those parties whose power Cryptography threatens to limit. When these assumptions are challenged, all privacy guarantees are void (this holds for symmetric [Fei73] as well as for public key [DH76, RSA78, GM84] systems).

Concretely, it is assumed that the encryption and the decryption processes are conducted freely and privately by the sender and the receiver, respectively. Specifically, on the receiver's side, a message, once encrypted, is considered private based on the assumption that the receiver's (Bob's) private key (regardless of where it resides) is not compromised. In fact, relying solely on the key not being compromised and not on other obscurities is known as the Kerckhoffs principle and was already formulated in 1883 [Ker83]. On the sender's side, in turn, it is concretely assumed that the sender (Alice) is free to pick the message to be sent.

Indeed, if Alice is not free to pick the message to be sent, we can hardly talk of communication "from Alice to Bob." Similarly, if Bob's secret key is compromised (and the key is the only source of secrecy), we cannot consider the communication to be private. In other words, Cryptography currently implements *private communication* assuming that encryption takes as input a message freely chosen by Alice and that, among all other parties, only Bob has access to the key necessary for decryption. We call these two assumptions the *sender-freedom assumption* and the *receiver-privacy assumption*, respectively.

Both assumptions are realistic for normal settings, and thus, it is not a big surprise that the majority of symmetric and asymmetric encryption systems have been developed under them, and it is so natural that these are implicit in the modeling. Yet, these assumptions fail to exist in a dictator-led country where law enforcement agencies have the (legal) power to get the private key of citizens to be surrendered upon request (thus undermining the receiver-privacy assumption). Furthermore, in a dictatorship, individuals can be forced by the authorities to encrypt and send some adversary selected (e.g., wrong) statements (thus undermining the sender-freedom assumption). Note that while we characterize the above as a dictator-led country, it is quite prevalent around the world today for governments (of all types) to ask for at least some of the above powers when discussing encryption!

Let us now discuss the two assumptions more in depth.

The receiver-privacy assumption. Achieving security without having to rely on this assumption asks us to consider the classical cryptographic problem of two parties that wish to privately communicate in the presence of an adversary that has the ability to eavesdrop on all communication between the two parties *and* the power to request the receiver's secret key[1]. At first sight, achieving privacy against such an adversary seems like an impossible task since, by definition, the secret key allows for the decryption of any ciphertext and, once an adversary has gained access to the secret key, nothing can stop it from decrypting the ciphertexts.

Before giving an intuitive idea of how we plan to approach this seemingly impossible problem, we want to elaborate on the security threat that we address and how it puts further constraints on the solution space. As mentioned before, this setting models adversaries of the scale of a nation state whose government has the "legal" power to force citizens to reveal their secret keys. In such a setting, it is expected that a citizen would abide by the request as any refusal could be considered as evidence that the communication was unlawful (or be considered such in the eyes of the adversary/government); in fact, the citizens may wish (as an advantage in an hostile environment) to be able to prove the benign nature of their messages. As we shall see, addressing the needs of privacy in the presence of such a powerful adversary in an effective way cannot be achieved by introducing *new* constructions that would immediately be ruled as illegal, but rather by showing that *existing* constructions can be adapted to support the new need. Indeed, for its own nature,

[1]This is masterly described in `xkcd` comic 538 (see `https://xkcd.com/538`).

the deployment itself of countermeasures must be hidden from the adversary as a government has the legal means and the power to make it illegal.

The sender-freedom assumption. This assumption posits that the sender Alice is free to choose which message to encrypt; without this assumption, the meaning of *communication* originating from Alice is lost. It is not difficult to imagine a setting in which someone is under duress to send a ciphertext with a fake message. Specifically, the sender might be forced to produce a ciphertext $\mathtt{ct}$ for a given message m, the *forced* message, with respect to a given public key $\mathtt{fPK}$, the *forced* public key. In addition the adversary will want to see the coin tosses R used to produce $\mathtt{ct}$ so as to be able to check that $\mathtt{ct} = \mathsf{Enc}(\mathtt{fPK}, m; R)$. Is there a way for the sender to satisfy the adversary but still send a message other than m to some other party? Clearly, if the adversary has the power to pick the message m and the randomness to be used, then the sender has no maneuvering space. This would be equivalent to the adversary holding Alice's cellphone and using it to send messages on Alice's behalf; clearly (with no freedom to choose anything) nothing can be done to prevent this (which is a, de facto, impersonation of Alice). Instead, we do not consider impersonation, but rather consider the setting in which the more remote adversary does select the message to be sent, lets Alice compute the ciphertext $\mathtt{ct}$ and later on Alice is required to exhibit the coin tosses used to compute $\mathtt{ct}$; i.e., the adversary does not completely control the cryptographic device as in an impersonation attack, it allows strong cryptography as well, but forces a valid explanation for the ciphertext for messages it chooses, including possibly some one-way hash post-processing of truly random bits to produce the random bits used in encryption, to prevent, e.g., some meaningful message being part of the randomness used for systems that encrypt over a message and randomness, and issues of this sort.

Again, in authoritarian regimes, under threats, dissidents are often forced to send false statements to public and international newspapers. In this situation, an international human rights organization (HR) that is not under the control of the dictator can release its public key. When dissidents in a country ruled by a dictator are forced by the authority to send false statements to an international newspaper, say AP News under the public key of AP News, dissidents can obey that request while being able to add hidden messages to HR, for example saying that what they are saying is false, as an additional protection.

Normative Prescriptions. One may ask: most of Cryptography is based on computational assumptions and this has not slowed down its deployment in real-world applications; why should we be worried by two extra assumptions? Let us pause and ask ourselves why we believe in the assumptions we use to design cryptographic schemes. The assumed hardness of computational problems reflects our current intuition about and understanding of Nature (and thus, its representation within mathematical models): randomness, one-way functions, and trapdoor functions do exist in our current understanding of Nature. On the other hand, let us look at the receiver-privacy assumption. Why do we believe that users will not be forced to reveal their private key? Essentially because there is a *normative prescription*[2] by which forcing someone to do something against its will is a punishable crime. In other words, society attaches a punishment to anybody who violates the receiver-privacy assumption. The same holds for the sender-freedom assumption. However normative prescriptions are not laws of Nature (not inherent to the way things are!) and they can be modified by a new social/normative re-order. And this is exactly what a dictator will do. On the other hand, quite obviously, no dictator will ever be able to make a one-way function efficiently invertible or will be able to predict a random bit[3].

Indeed Cryptography can be seen as an effort to replace assumptions based on normative prescriptions with milder assumptions leveraging our understanding of Nature and, more specifically, of Computational Complexity (combined with Information Theory) and/or of Computing Architectures. A primary example of this effort is the concept of Secure Multiparty Computation (aka Private Function Evaluation). If we accept assumptions based on normative prescriptions, then we can identify some individual to act as *a trusted third party* (TTP); that is, the TTP receives the private inputs and returns the value of the function

[2]We do not use the term "law" to mark the conceptual difference from "law of Nature."

[3]In "The Game-Players of Titan," a novel by Philip K. Dick, randomness is the only weapon humans can use against silicon-based telepath aliens from Titan, called *vugs*.

to be computed. A law (a normative prescription) will describe the expected behavior of the TTP and identify the penalties for not following the prescription. On the other hand, the field of Secure multi-party computation [Yao86, GMW87] has shown how to achieve the same without relying on normative prescriptions regarding the TTP (replacing it with reliance on computational complexity assumptions and/or architectural separation and isolation of computing elements).

Our result in a nutshell: What is the point of being a dictator if you cannot dictate? A dictator is very likely going to declare Cryptography which is not available to the regime unlawful (it may outlaw Cryptography whatsoever, but in this case it will lose an advantage over the commons as *Cryptography rearranges power*! [Rog15]). This will stand against all normative assumptions and is the ultimate goal of an extreme "escrow encryption."

Our goal is to show that based on well established cryptosystems, under the dictator rules (that is, getting the receiver key and getting to dictate the messages of the sender, before the system starts sending any message), there is a way for a selected set of users (those who are trusted to not be agents of the regime) to use the system in a special way. Namely, there is always a way to use the system to send messages as requested by the dictator and to prove to the dictator the messages were sent, while at the same time using the system between the selected group in an "anamorphic fashion," *i.e.*, the selected receiver (with access to a special key) will get a distorted view of the ciphertext and will extract a covert message, yet any other receiver will get the dictated message (and it will be easy to prove to the dictator that this was indeed the message sent and that there is no other way to decrypt messages).

The above scenario (which we explain and formalize in the sequel) shows that with acceptable existing cryptosystems (which pre-existed our work, and were not designed with the above goal in mind) it is possible to nevertheless bypass the dictator. With these cryptosystems, any benefit of using them also allows the bypassing of the dictator decree with a built-in cover up. This seems to be a first clear indication that, directly and inherently (not due to potential weakening of the system by an increased external cyber-security attacks or software security weaknesses) Cryptography allows for bypassing the stringent dictator's rules and regulation. Therefore: If you want technology development and society to enjoy the fruits of Cryptography (as, for example, is given in [MY15] for the case of smartphones), then you have to know that it may not be controllable, and any attempt to do so is, in fact, an exercise in futility!

2 Related Works

Having presented the crux of our notion, let us examine related works and notions dealing with other aspects of violations of the two implicit normative assumptions above.

Key-Escrow. The availability and proliferation of E2E encryption for smartphone messaging applications (see [MP16, ACD19, Wha20] for two of the most widely used applications and for some of the formal treatment of the Cryptography on which they are based) has renewed the debate between law enforcement and government security agencies, and technologists. Actually, the debate goes back to at almost 30 years ago when the Internet became mainstream (early 1990s).The following quote is from [Dak96]:

Presently, anyone can obtain encryption devices for voice or data transmissions. Unfortunately, this group may include criminals, terrorists and drug dealers. Law enforcement groups believe this could soon create a devastating problem because these authorities commonly rely on electronic surveillance, also known as "wiretapping," as a tool for fighting crime. That is, if criminals can use advanced encryption technology in their transmissions, electronic surveillance techniques could be rendered useless because of law enforcement's inability to decode the message.

Much research in Cryptography has focused on methods to make the strong privacy guarantees offered by encryption ineffective under very specific and well identified situations. An early prototypical and pioneering example of this kind of work is the concept of a *Fair Cryptosystem* by Micali [Mic] that was one of the first to consider the possibility of targeted revocation of the privacy offered by encryption. Roughly speaking, this was achieved by sharing the secret key among a number of parties (the majority of which is assumed to be honest, say) and each would reveal its share, so as to allow reconstruction, only if a court order to that effect would be produced. In 1993 (essentially at the same time of Fair Cryptosystems), the US government put forth *The Clipper* proposal [Cli] by which all strong encryption systems had to retain a copy of keys necessary to decrypt information with a trusted third party who would turn over keys to law enforcement upon proper legal authorization. The Clipper chip was attempting to bind the ability to identify the key of the sender to facilitate government access and the ability to decrypt. The proposal had a flaws involving a too small authentication field [Bla94] and, actually, the binding was shown to be broken so as to bypass the mechanism [FY93]. All the above systems employed either trusted hardware or other trusted elements (trustees) added to the Public-Key Infrastructure setting. An open question at the time was to construct a "software only" escrow, employing the existing trusted point of the certification authority within the existing public key infrastructure as a handle (see [YY98]).

Given general security concerns, the report by Abelson et al. [AAB+97] (see also [AAB+15]) identifies the major threats of such a *key-escrow* system (primarily, the availability of an access method to the user's encrypted data which is not under the user own responsibility). The possibility that a key-escrow system could be abused by law-enforcement agency to violate the privacy of the users and to conduct large scale surveillance, which is our main concern in this paper, and was identified in the first report [AAB+97] that looked into the privacy issues arising from the Clipper chip and from other key-escrow mechanisms. We note that, obviously, the same concerns and more hold if weak Cryptography, which can be broken with enough feasible resources, is mandated in our modern time and with available public computational resources, such Cryptography simply does not work.

The main approach taken by cryptographers to address the above important point regarding escrow has been to construct systems relying on cryptographic tools (mathematical assumptions) that will make it impossible for governments to infringe on the privacy of the individuals and the guarantees made relying on a combination of normative assumptions as well as on cryptographic primitives. For example, the guarantee offered by a Fair Cryptosystem rely on the cryptographic strength of the secret-sharing primitives and on the normative prescription that the majority of shareholders will only act upon a request from the recognized authority, which, in turn, relies on the assumption that share holder are honest and uncoercible. From our point of view it is natural to ask: What if the authority is a dictator, and thus normative prescriptions have no effect?

Along these lines, the very recent work of Green et al. [GKVL] offers the most complete approach by formally defining the desired properties for a *law enforcement access system* and putting forth the notion of an *abuse-resistant law enforcement access system (ARLEAS).* In addition, they give a feasibility result using standard cryptographic techniques. Most systems, including ARLEAS [GKVL], consider three types of parties: *users*, that employ encryption to exchange messages; *law enforcement*, that need to access encrypted messages produced by users; *judiciary*, that grants or denies authorizations to the access requests of the law enforcement. Note that such a system does not offer any protection against dictatorial states. First of all, in most dictatorships the judiciary is not an independent power thus making the abuse-resistant property nothing more than a wish. Actually, a key-escrow system makes it even easier for a dictator to violate the privacy of the users as it is enough to nominate themselves, as dictators tend to do, to be the judiciary and law enforcement. Even more importantly, note that all such systems still rely on the receiver-privacy and sender-freedom assumptions.

The main technical objective of our work is to show that one can achieve privacy in communication using existing cryptographic systems even in the presence of a dictator without relying on any normative prescription.

Deniable Encryption. The concept of *Deniable Encryption*, put forth by Canetti et al. [CDNO97], might seem relevant to our setting. A deniable encryption scheme allows the sender to generate *fake* coin tosses that make the ciphertext look like an encryption of another, innocent, cleartext. So whenever the sender is requested to open a ciphertext by surrendering the coin tosses, he can just generate fake coin tosses and thus effectively conceal the real plaintext. However, an assumption in deniable encryption is that the adversary has the power to approach the sender *after* the ciphertext was transmitted. It was mentioned in [CDNO97] that deniability is impossible in the context of direct physical access, where "Eve [the adversary] approaches Alice [the sender] *before* the transmission and requires Alice [the sender] to send specific messages". It was clearly stated in the original paper [CDNO97] that "Certainly, if Alice [the sender] must hand Eve [the adversary] the *real* cleartext and random bits then no protection is possible". Hence, while deniability is an important property, it does not solve the issue of facing a dictator.

Kleptography. The concept of having a cryptosystem inside another cryptosystem was used in Kleptography [YYa, YY97, YYb]. However, the goal in Kleptography is to attack the cryptosystem owner by using inside an implementation (or a specification) a method to leak exclusively to an adversary. This goal is in fact, not to help against an adversary, but to covertly introduce an adversary, and is a tool to help a dictator!. In some sense it can be used as an implicit key escrow, and in some formal sense it negated the US government plan to distribute cryptography in black-box hardware devices. Due to Snowden revelations, in fact, an Elliptic Curve variant of the repeated DH kleptogram in the above works was deployed in the Dual-EC pseudorandom generator standard, as was verified in [CNE$^+$14]. As a response, nowadays, the cryptographic community is working on systems and architectures which can mitigate such system subversion attacks (see [BPR14, RTYZ16, RTYZ17]).

Steganography and "chaffing and winnowing". The other tool to send secret messages with is steganography, where a message is concealed within another message (exploiting some redundancy in the message structure and style). This is always possible, but systems of this nature are hard to deploy widely and systematically as part of an established large scale computing base in a situation it is not allowed (by the dictator).

In a response to the US government possibly restricting encryption systems while allowing authentication methods as part of its cryptographic export control, Rivest proposed in 1998 the "chaffing and winnowing" system [Riv]. This system shows how sharing a MAC key for authentication is turned into a method for sending concealed message (in some steganographic sense). This clearly demonstrated that the separation of cryptography for authentication and cryptography for confidentiality, proposed by the possible export regulation, is quite artificial, and gave a way to bypass key escrow using authentication keys which were not proposed to be part of escrow encryption schemes at the time. Note that in our scenario of the dictator, the dictator can certainly ask for a MAC (authentication) key as well and get the concealed message itself.

PublicKey Steganography and Subvertable Backdoored Encryption. von Ahn and Hopper [vAH04] were the first to study steganographic public key and key exchange. Their constructions rely on the existence of a public random string that cannot be tampered or chosen by the dictator. The work of Horel et al. [HPRV19] removes this assumptions by showing that steganographic key exchange can be achieved without resorting to public information. The key exchange is used then in conjunction with the *rejection sampling technique* (see Section 5.1).

Comparing our model with the one of [HPRV19], we note that our model assumes the parties have originally shared (private and thus un-tampered by dictator) information. This, importantly, allows for *zero-latency* communication hidden from the dictator. In Section 5.3 nevertheless, we show how to combine [HPRV19] with our construction thus sacrificing zero-latency and dispensing with the need of shared information. Comparing, in turn, one of our main solutions (see Section 5.3) with the one of Horel et al. [HPRV19] we note that, for a ciphertext carrying λ bits, rejection sampling allows to transmit $O(\log \lambda)$ extra bits whereas our construction carries λ thus achieving *bandwidth rate* of 1 (as opposed to $(\log \lambda)/\lambda$). Such a bandwidth rate is rare in steganographic systems in general. Our second result in Section 6 also achieves zero latency, and does not need the parties to share a private key.

3 Our approach

In this section, we would like to give a taste of our approach, and present a generic but limited solution that shows its feasibility. For concreteness, we consider the receiver's side.

As we have already observed, there is no hope for preventing an adversary that is in possession of the secret key from decrypting a ciphertext. In turn, one might think of ciphertexts that carry two messages and a different key is needed for each of them. When asked for the secret key, the receiver might release only one of the two keys, thus protecting one of the messages. But then, if the adversary knows that there are two keys, why should he be happy to receive just one key? Because we will make sure that the adversary *believes* that there is no second key! Roughly speaking, we would like to have an encryption scheme for which it is possible to generate a public key with one secret key or, alternatively, to generate a public key that has *two* associated secret keys. More precisely, the technical core of our proposed solution consists of the concept of an *Anamorphic Encryption* scheme, a special encryption scheme whose public keys can be generated in one of two possible modes: *normal* or *anamorphic*.

- A *Normal* public key is associated with a *Normal* secret key and it can be used for the normal encryption functionality: the sender encrypts the message m using the public key, and the receiver decrypts the ciphertext using the secret key. An adversarial authority (i.e., the dictator) that is in possession of a ciphertext can (legally) force the receiver to surrender the secret key and thus gain access to the message.

- An *Anamorphic*[4] public key instead is associated with two secret keys: a *normal* secret key and a *double* secret key. A ciphertext $\mathtt{ct}$ produced with an anamorphic public key carries two messages: the *normal* message m_0, that can be obtained by decrypting $\mathtt{ct}$ using the normal secret key; and the *anamorphic* message m_1, only visible to parties that have the *double* secret key. When requested to surrender his secret key to allow inspection of the ciphertexts, the owner of an anamorphic public key will pretend that the key is normal and reveal the normal secret key. Thus, the dictator will gain access only to the normal message m_0 and the special message m_1 is kept private.

Normal public keys can be used by receivers who do not expect their secret keys to be requested by the adversarial authority. Actually, these users need not even know about anamorphic public keys and their operations will not be affected in any way. Anamorphic public keys could instead be generated by someone who has reasons to believe that the dictator will want to get the information he has received; for example, an investigative reporter or an opposition leader. As an example, consider an investigative reporter Bob who wants to communicate in a private way with his informant, Alice. Bob sets up his public key as an anamorphic key and gives Alice the double secret key. Note that Bob publishes his anamorphic public key for everybody to use. However, when Alice has some sensitive information for Bob, she uses the double key obtained from Bob to produce an anamorphic ciphertext carrying two messages: m_0 is set equal to some innocent looking message (for example, a general question about the work of the reporter; recall that in a dictatorship Alice does not even have a free choice of this message), and message m_1 will instead contain the sensitive information that Alice wishes to communicate to Bob. Should Bob be requested to surrender his secret key, he will pretend that his public key is just a normal public key and will reveal the normal secret key. In this way the adversary will be able to read message m_0 (the innocent message containing no sensitive information) and will gain no access to the potentially incriminating message m_1.

Clearly, for this to work the following conditions must be satisfied:

- a pair of anamorphic public and secret keys must be indistinguishable from a normal corresponding pair; and

- the ciphertexts produced using an anamorphic public key must be indistinguishable from those produced by a normal public key.

[4]The adjective *Anamorphic* is used to denote a drawing with a distorted projection that appears normal when viewed from a particular point or with a suitable mirror or lens. Similarly, an anamorphic ciphertext will reveal a different plaintext when decrypted with a suitable key.

In addition, we are interested in performance parameters: We would like the system to have *zero-latency* in the sense that the anamorphic system is ready to be used whenever the normal system is ready. Additionally, we are interested in system where the anamorphic *bandwidth rate* (i.e, number of anomorphic bit transmitted divided by the number of normal bits transmitted) is high (whenever and as much as possible).

A simple solution that does not work. At first, one might think that constructing an Anamorphic Encryption scheme is not difficult and indeed it is possible to turn any encryption scheme $\mathcal{E}$ into a Anamorphic Encryption AME in the following rather straightforward way by adding redundancy to the ciphertext. The normal encryption process of AME consists of computing a ciphertext $\mathtt{ct}$ according to $\mathcal{E}$ and then outputting $(\mathtt{ct}, R)$, where R is a randomly selected string. During the decryption process of AME with the normal secret key, the random string R is ignored and the message is obtained by decrypting $\mathtt{ct}$ according to $\mathcal{E}$. Thus, a normal pair of public and secret key for AME is simply a pair of keys from $\mathcal{E}$. An anamorphic public key instead is associated also with the secret key K of a symmetric encryption scheme $\mathcal{E}'$ with pseudorandom ciphertexts (e.g., it is a pseudorandom permutation). During the anamorphic encryption process, ciphertext $\mathtt{ct}$ is an encryption of m_0 according to $\mathcal{E}$ and the random string R that is appended to $\mathtt{ct}$ is an encryption of m_1 with respect to $\mathcal{E}'$ computed using key K that, thus, plays the role of the double key. Note that the public keys are identical and, by the pseudorandomness properties of the ciphertexts of $\mathcal{E}'$, a normal ciphertext, in which R is truly random, is indistinguishable from an anamorphic ciphertext in which R is a ciphertext of $\mathcal{E}'$.

So, the question is: do we have a satisfactory solutions? Hardly so! The mere fact of using a standard encryption scheme and augmenting each ciphertext with a random string is suspicious and, moreover, a user not interested in sending hidden messages has no incentive to append a random string to the ciphertext just to attract the attention of the dictator. Rather we are interested in showing that *existing* encryption schemes can be used, in the form in which they have been originally designed, to provide a second channel that is secure also with respect to an adversary that has access to the secret key of the receiver. In practice, this will not raise any suspicion since all ciphertexts will be from a standard encryption scheme or will be indistinguishable from them. In addition, no extra action (like generating and appending a random string to each ciphertext) is required from the normal users who can actually be completely unaware of the special mode of operation.

In other words the question we ask is not

Can we construct an Anamorphic Encryption scheme?

but rather:

Is any of the existing *encryption schemes also Anamorphic?*

A positive answer to this question will constitute technical evidence of the futility of the efforts of governments around the world (dictatorships and democracies) to control encryption.

Rejection Sampling Encryption. We give a quick description of our approach by presenting the *rejection sampling encryption*, a simple cryptosystem that gives guarantees even if the secret-key assumption and the sender-freedom assumption do not hold. Rejection sampling encryption is inspired by the *biased-ciphertext attack* of [BPR14]. We note that the objective of the biased-ciphertext attack is to *subvert* an encryption scheme so that the private key can be leaked without the legitimate owner noticing any abnormal behaviour. It is interesting to note that this technique can be used to setup a dictator setting, as shown in [BPR14], as well as for the opposite of objective, as we show below.

Consider any public key encryption scheme $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ and a family of Pseudo-Random Functions $F(\cdot, \cdot)$. The normal pair of public and secret keys $(\mathtt{PK}, \mathtt{SK})$ is obtained by running the key-generation algorithm of $\mathcal{E}$ and encryption and decryption algorithms of $\mathcal{E}$ are used to create and decrypt ciphertexts. The anamorphic public key is generated in the same way but, in addition, a double key is selected by picking a random seed K for F. Now suppose that the sender wants to encrypt a normal plaintext m_0 and a one-bit hidden plaintext m_1. The sender proceeds by producing ciphertexts $\mathtt{ct}$ of m_0 w.r.t $\mathtt{PK}$ until one is obtained such that $F(K, \mathtt{ct}) = m_1$. Upon receiving $\mathtt{ct}$, m_0 is obtained by decrypting $\mathtt{ct}$ using the decryption algorithm of $\mathcal{E}$ and m_1 is obtained by computing $F(K, \mathtt{ct})$.

Why does this satisfy our needs? First, the anamorphic encryption of m_1 is a sound procedure due to the rejected sampling procedure on one-bit message where F is a pseudorandom function over random ciphertexts. Then, observe that in both cases the public key is simply a randomly generated public key for $\mathcal{E}$ and there is no indication that the double key has been selected. Given these facts, the receiver can plausibly deny the existence of a special secret key (no key is used to generate any field, it is rather via rejection sampling and a key not used in the process that the anamorphic message is generated). If requested by the adversary, the sender can give the random coin-tosses used to generate $\mathtt{ct}$ Note that this are not truly random as the ciphertext obtained has the property $F(K, \mathtt{ct}) = m_1$ but they are indistinguishable from truly random coin tosses by the properties of the PRF F. Note that the adversary does not have access to the special secret key K: it is not even sure that such a key exists! In other words, this simple construction allows secure two-party communication without having to rely on the secret-key assumption nor on the sender-freedom assumption. Moreover, it can be used without raising any suspicion as its operations can be plausibly described as obtained from using a standard encryption scheme and compliance with some requests about the randomness employed.

Still, we must point out two important limitations of the construction. First of all, the special plaintext is restricted to be 1 bit. It is not difficult to modify the construction so that m_1 can be of any length l, at the cost of making the running time (of the rejection sampling) exponential in l, which is clearly unsatisfactory. In addition, the sender and the receiver must agree on common special secret key K for this to work.

Our Main Technical Contributions. The rest of this paper consists of identifying encryption schemes that appear in the literature which address the two limitations we identified above.

- **Receiver's side.** We show that the Naor-Yung paradigm for CCA secure encryption from CPA secure encryption gives receiver-Anamorphic Encryption schemes (see Def. 1) with a *bandwidth rate* of 1. That is, each ciphertext for λ plaintext bits carries λ additional bits that are hidden from the dictator. Moreover, as in the public key model, communication hidden from the dictator can start with *zero latency.*

- **Sender's side.** We show that a lattice based cryptosystems from the literature [Reg05, GPV08] are sender-Anamorphic Encryption (see Def. 3) and they also achieve hidden communication with *zero latency.* In addition, they do not require the sender and receiver to share any secret (just that there is an additional receiver in the system), thus, enhancing their practicality.

Given these, one can conclude that the "Crypto Wars" which consists of attacks on the free use of strong cryptography (involving requirements to give keys of such cryptographic schemes to the dictator) are possibly futile. One may then conclude that dictators should mandate only weak cryptography! But, due to earlier battles, we are already know that disallowing strong Cryptography is totally unhelpful to and imposes limits on the development of advanced information technology systems, implying dire consequences to the economy and to society.

Roadmap. In Section 4, we put forth the concept of a Receiver-Anamorphic Encryption and give a formal notion of security. In Section 5.3, we show that the Naor-Yung paradigm [NY90] gives Receiver-Anamorphic Encryption with *zero latency and bandwidth rate 1.* In Section 6, we define the concept of an Sender-Anamorphic Encryption that can be used to obtain private communication when the sender-freedom assumption does not hold. We show that lattice-based cryptosystems have this property (and give some evidence that other public key cryptosystems are not Sender-Anamorphic Encryption). The scheme is *zero-latency* as well.

4 Receiver-Anamorphic Encryption

In this section we present the concept of a *Receiver-Anamorphic Encryption scheme* that provides private communication without relying on the receiver-privacy assumption (while in Section 6 we will present the concept of a *Sender-Anamorphic Encryption scheme.*)

A *Receiver-Anamorphic Encryption* scheme (a Receiver-AM scheme) consists of two encryption schemes: the *normal* scheme $(\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ and the *anamorphic* scheme $(\mathsf{aKG}, \mathsf{aEnc}, \mathsf{aDec})$. It can be deployed as normal scheme in which case Bob runs the (real) key generation algorithm KG to obtain a pair of keys $(\mathtt{PK}, \mathtt{SK})$ and, as usual, publishes $\mathtt{PK}$. When Alice wishes to send message m, she produces ciphertext $\mathtt{ct}$ by running the (real) encryption algorithm Enc by using $\mathtt{PK}$ and m. When $\mathtt{ct}$ is received by Bob, it is decrypted by running the (real) decryption algorithm Dec and using $\mathtt{SK}$. Thus, when deployed as normal a Receiver-AM is just a regular public-key encryption scheme. If the dictator comes for the secret key, Bob surrenders $\mathtt{SK}$.

Bob deploys the scheme as anamorphic when he wants to protect the confidentiality of the communication with Alice even in the event that he is forced to surrender his secret decryption key to the dictator. In this case, Bob runs the *anamorphic* key generation algorithm aKG that returns a pair of anamorphic public-secret keys $(\mathtt{aPK}, \mathtt{aSK})$ along with a special key, $\mathtt{dkey}$, called the *double key*. As usual, Bob publishes $\mathtt{aPK}$ and keeps $\mathtt{aSK}$ private but $\mathtt{dkey}$ is shared with Alice. If asked, Bob will surrender $\mathtt{aSK}$ to the dictator. The pair $(\mathtt{aPK}, \mathtt{aSK})$ is a fully functional pair of keys: if a message m is encrypted by using Enc and $\mathtt{aPK}$, it can be decrypted by Dec on input $\mathtt{aSK}$. Key $\mathtt{dkey}$ is instead used by Alice to send Bob messages that remain confidential even if $\mathtt{aSK}$ is compromised. Specifically, whenever Alice has a message m_1 that must remain confidential, she picks an innocent looking message m_0 and encrypts (m_0, m_1) using the anamorphic encryption algorithm aEnc with $\mathtt{dkey}$. The ciphertext $\mathtt{ct}$ produced by aEnc has the property that it returns m_0 when decrypted with the normal decryption algorithm Dec and with key $\mathtt{aSK}$; whereas it returns m_1 when decrypted by running the anamorphic decryption algorithm aDec on input the double key $\mathtt{dkey}$. In other words, the authority will obtain m_0 and Bob will obtain m_1. Clearly, the ciphertext produced by Alice must indistinguishable from a ciphertext of m_1 produced using Enc even to an adversary that has access to $\mathtt{aSK}$. We stress again that Alice and Bob share a key, $\mathtt{dkey}$, in order to achieve privacy without having to rely on the secret-key assumption.

We are interested in Receiver-AM schemes in which the normal scheme is an *established* and *already used* cryptosystem.

4.1 Syntax

In this section we formally define the concept of a Receiver-Anamorphic Encryption scheme. Then in Section 4.3 we will present the corresponding security notion.

Definition 1. *An encryption scheme* $\mathsf{AME} = (\mathsf{AME.KG}, \mathsf{AME.Enc}, \mathsf{AME.Dec})$ *is a* Receiver-Anamorphic Encryption *(or, simply, a Receiver-AM) if there exists an* anamorphic *triplet* $\mathsf{aAME} = (\mathsf{AME.aKG}, \mathsf{AME.aEnc}, \mathsf{AME.aDec})$ *with the following syntax*

- $\mathsf{AME.aKG}$ *takes as input the security parameter* 1^λ *and returns an* anamorphic *public key* $\mathtt{aPK}$*, an* anamorphic *secret key* $\mathtt{aSK}$ *and a* double key $\mathtt{dkey}$*.*

- $\mathsf{AME.aEnc}$ *takes as input the double key* $\mathtt{dkey}$ *and* two *messages* $\mathtt{m}_0$ *and* $\mathtt{m}_1$ *and returns an* anamorphic *ciphertext* $\mathtt{act}$*.*

- $\mathsf{AME.aDec}$ *takes as input the double key* $\mathtt{dkey}$ *and an anamorphic ciphertext* $\mathtt{act}$*, computed by running* $\mathsf{AME.aEnc}$ *on input* $\mathtt{dkey}$ *and messages* $\mathtt{m}_0$ *and* $\mathtt{m}_1$*, and returns message* $\mathtt{m}_1$*.*

As we have seen previously, a Receiver-AM scheme can be deployed as normal or as anamorphic. When Bob deploys the Receiver-AM scheme as anamorphic, Alice (or any other user that has received $\mathtt{dkey}$ from Bob) can use aEnc to produce ciphertexts. However, Bob cannot instruct all users to use aEnc as that would be like admitting that he has deployed the scheme as anamorphic; in other words users that are unaware that the scheme has been deployed as anamorphic will use Enc. It is thus crucial that the encryption-decryption functionality does not break for these users, despite using the normal encryption algorithm with an anamorphic public key. Indeed, the fact that ciphertexts generated by the normal encryption algorithm cannot be decrypted could be a strong evidence, if not a proof, that the public key is anamorphic.

Next, we formally identify the four different modes of operations that arise from mixing normal, and anamorphic algorithms and keys and then in Section 4.3 we present our security notion.

	Key Gen.	Encryption	Decryption
Fully Anamorphic	aKG	aEnc	aDec
Anamorphic with Normal Dec	aKG	aEnc	Dec
Anamorphic with Normal Enc	aKG	Enc	Dec
Normal	KG	Enc	Dec

Figure 1: The four modes of operation of an anamorphic encryption scheme. The *fully anamorphic mode* is used by Bob to communicate privately with Alice. The *anamorphic mode with normal decryption* is used by Bob when the dictator requests the decryption of an anamorphic ciphertext sent by Alice. The *anamorphic mode with normal encryption* is used by Charlie, unaware that Bob has an anamorphic key, to send a message to Bob. The *normal mode* is used by Charlie that sets up his key in normal mode to receive messages from other users. The normal mode offers no privacy guarantee against the dictator.

4.2 Modes of Operation

A Receiver-AM scheme AME naturally defines the following four modes of operation, each corresponding to a different scenario. Of these four modes, one uses the normal keys and three use anamorphic keys. The security definition (see Definition 4) will require that, roughly speaking, the the *normal* and the *fully anamorphic* mode be indistinguishable. This is sufficient to prove that all of them are indeed pairwise indistinguishable. Next, we formally define the modes of operations.

1. The *<u>f</u>ully <u>a</u>na<u>m</u>orphic <u>e</u>ncryption* mode is used when Bob deploys the scheme as anamorphic and Alice uses $\mathtt{dkey}$ to send Bob a private message. This mode is associated with the triplet of algorithms $\mathsf{fAME}_{\hat{m}} = (\mathsf{AME.aKG}_3, \mathsf{AME.aEnc}_{1,\hat{m}}, \mathsf{AME.aDec})$ where, for every message $\hat{m}$,

   - $\mathsf{AME.aKG}_3(1^\lambda)$ is the algorithm that runs $\mathsf{AME.aKG}(1^\lambda)$ obtaining $(\mathtt{aPK}, \mathtt{aSK}, \mathtt{dkey})$ and returns $\mathtt{dkey}$. Note that the index in aKG_3 denotes the components of the triplet generated by aKG that is selected to appear in the output.
   - $\mathsf{AME.aEnc}_{1,\hat{m}}(\mathtt{aPK}, m)$ is the algorithm that returns $\mathsf{AME.aEnc}(\mathtt{aPK}, \hat{m}, m)$. Note that index 1 in $\mathsf{AME.aEnc}_{1,\hat{m}}$ denotes that message $\hat{m}$ will be passed as first message to algorithm aEnc.

   Note that $\mathsf{fAME}\hat{m}$ is, for every $\hat{m}$, a symmetric encryption scheme.

2. The *<u>a</u>namorphic with <u>n</u>ormal <u>d</u>ecryption* mode is used when Bob deploys the scheme as anamorphic (and thus the public key is generated using algorithm $\mathsf{AME.aKG}$), the ciphertext $\mathtt{ct}$ is produced by Alice by running the anamorphic encryption algorithm $\mathsf{AME.aEnc}$ on input (m_0, m_1) and the double key $\mathtt{dkey}$, and the ciphertext is decrypted by the dictator by running the normal encryption algorithm $\mathsf{AME.Dec}$ on input $\mathtt{aSK}$.

   More formally, the mode is associated with the triplet $\mathsf{andAME}_{\hat{m}} = (\mathsf{aKG}_{1,2}, \mathsf{aEnc}_{2,\hat{m}}, \mathsf{Dec})$, where, for every $\hat{m}$,

   - $\mathsf{andAME.aEnc}_{2,\hat{m}}(\mathtt{aPK}, m)$ is the algorithm that returns $\mathsf{AME.aEnc}(\mathtt{aPK}, m, \hat{m})$. Note that the index 2 in $\mathsf{AME.aEnc}_{2,\hat{m}}$ denotes that message $\hat{m}$ will be passed as second message to algorithm aEnc.

3. The *<u>a</u>namorphic with <u>n</u>ormal <u>e</u>ncryption* mode is used when Bob deploys the scheme as anamorphic (and thus the anamorphic public key $\mathtt{aPK}$ is used as public key) and a sender encrypts messages using the normal encryption algorithm $\mathsf{AME.Enc}$. The ciphertexts produced in this way can be read by the dictator that has the secret key $\mathtt{aSK}$ associated with $\mathtt{aPK}$ by running the normal decryption algorithm $\mathsf{AME.Dec}$.

   More formally, the mode is associated with the triplet $\mathsf{aneAME} = (\mathsf{aKG}_{1,2}, \mathsf{Enc}, \mathsf{Dec})$, where

   - $\mathsf{aKG}_{1,2}(1^\lambda)$ is the algorithm that runs $\mathsf{AME.aKG}(1^\lambda)$, obtaining $(\mathtt{aPK}, \mathtt{aSK}, \mathtt{dkey})$, and returns the pair $(\mathtt{aPK}, \mathtt{aSK})$. Note that the indices in $\mathsf{aKG}_{1,2}$ denote the components of the triplet generated by aKG that are selected to appear in the output.

4. The *normal* mode of operation is associated to the triple of algorithms $\mathsf{nAME} = (\mathsf{AME.KG}, \mathsf{AME.Enc}, \mathsf{AME.Dec})$ and it corresponds to the scenario in which the scheme is deployed and used in normal mode; that is, the keys are generated by running KG, the ciphertexts are constructed by running Enc and decrypted by running Dec.

4.3 Security Notion

We are now ready to define the notion of a *Secure Receiver-AM* scheme. As we shall see, the security notion posits that no PPT dictator can distinguish whether the Receiver-AM scheme is in normal mode or in fully anamorphic mode. In other words, it is indistinguishable whether nAME or $\mathsf{fAME}\hat{m}$, for message $\hat{m}$, is being used. We first make two interesting remarks:

- No explicit security requirement is made for the security of the fully anamorphic mode; in other words, it is not explicitly required that Alice's secret message m_1 is actually secret.
- No explicit security and functional requirement regarding the partial anamorphic modes of operation, anamorphic with normal encryption and anamorphic with normal decryption, is made.

The requirements in both bullets above are clearly desirable: first, we would like Alice's secret message to be kept secure from the authorities; and we do not want the authority to be able to distinguish whether the normal mode or one of the other anamorphic modes is being used. As we shall see, the security notion of Definition 2 is sufficient for the requirements in the two bullets above. Let us now proceed more formally and define the following two games involving a dictator $\mathcal{D}$.

```
NormalGame_{AME,D}(λ)
    1. Set $(\mathtt{PK}, \mathtt{SK}) \leftarrow \mathsf{AME.KG}(1^\lambda)$ and send $(\mathtt{PK}, \mathtt{SK})$ to $\mathcal{D}$.
    2. For $i = 1, \ldots, \mathsf{poly}(\lambda)$:
        • $\mathcal{D}$ issues query $(m_0^i, m_1^i)$ and receives $\mathtt{ct} = \mathsf{AME.Enc}(\mathtt{PK}, m_0^i)$.
    3. Return $\mathcal{D}$'s output.
```

```
FullyAGame_{AME,D}(λ)
    1. Set $(\mathtt{aPK}, \mathtt{aSK}, \mathtt{dkey}) \leftarrow \mathsf{AME.aKG}(1^\lambda)$ and send $(\mathtt{aPK}, \mathtt{aSK})$ to $\mathcal{D}$.
    2. For $i = 1, \ldots, \mathsf{poly}(\lambda)$:
        • $\mathcal{D}$ issues query $(m_0^i, m_1^i)$ and receives $\mathtt{ct} = \mathsf{AME.aEnc}(\mathtt{dkey}, m_0^i, m_1^i)$.
    3. Return $\mathcal{D}$'s output.
```

Note that in $\mathsf{NormalGame}$ the key is normal (that is, output by KG) and the i-th ciphertext is an encryption of m_0^i computed using Enc. In other words, $\mathcal{D}$ interacts with the scheme in normal mode. On the other hand, in $\mathsf{FullyAGame}$ the key is output by aKG and the i-th ciphertext carries both messages m_0^i and m_1^i. In other words, $\mathcal{D}$ with the scheme in fully anamorphic mode. Note that in both cases, $\mathcal{D}$ is given the public key and the *associated secret key*.

More formally, we denote by $p_{\mathsf{AME},\mathcal{D}}^{\mathsf{NormalGame}}(\lambda)$ (respectively, $p_{\mathsf{AME},\mathcal{D}}^{\mathsf{FullyAGame}}(\lambda)$) the probability that $\mathsf{NormalGame}_{\mathsf{AME},\mathcal{D}}(\lambda)$ (respectively, $\mathsf{FullyAGame}_{\mathsf{AME},\mathcal{D}}(\lambda)$) outputs 1 and we introduce the following definition.

Definition 2. *A* Receiver-AM *scheme* $\mathsf{AME} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *with anamorphic triplet* $(\mathsf{aKG}, \mathsf{aEnc}, \mathsf{aDec})$ *is a Secure Receiver-AM scheme if*

1. AME *is an IND-CPA scheme;*
2. *for every message* $\hat{m}$*,* $\mathsf{fAME}_{\hat{m}}$ *is a symmetric encryption scheme;*
3. *for all PPT dictators* $\mathcal{D}$*,*
$$\left| p^{\mathsf{NormalGame}}_{\mathsf{AME},\mathcal{D}}(\lambda) - p^{\mathsf{FullyAGame}}_{\mathsf{AME},\mathcal{D}}(\lambda) \right| \leq \mathsf{negl}(\lambda).$$

In the rest of this section, we prove that the fully anamorphic mode (used by Alice and Bob to communicate privately) is an IND-CPA private key encryption scheme, thus addressing the observation in the first bullet above. We start in Section 4.4 by proving that the anamorphic modes with normal encryption is indistinguishable from the fully anamorphic mode and from the normal mode. We then build on this to prove the security of the fully anamorphic mode in Section 4.5. We will address the observation in the second bullet in Appendix B.

4.4 Properties of the Anamorphic Mode with Normal Encryption

We start by defining game $\mathsf{aneGame}_{\mathsf{AME},\mathcal{A}}$, for a Secure Receiver-AM AME and a PPT adversary $\mathcal{A}$. As usual, we denote by $p^{\mathsf{aneGame}}_{\mathsf{AME},\mathcal{A}}(\lambda)$ the probability that $\mathsf{aneGame}_{\mathsf{AME},\mathcal{A}}(\lambda)$ outputs 1. Game $\mathsf{aneGame}$ describes the Anamorphic Mode with Normal Encryption.

```
$\mathsf{aneGame}_{\mathsf{AME},\mathcal{A}}(\lambda)$
    1. Set $(\mathtt{aPK}, \mathtt{aSK}, \mathtt{dkey}) \leftarrow \mathsf{AME.aKG}(1^\lambda)$ and send $(\mathtt{aPK}, \mathtt{aSK})$ to $\mathcal{A}$.
    2. For $i = 1, \ldots, \mathsf{poly}(\lambda)$:
        • $\mathcal{A}$ issues query $(m_0^i, m_1^i)$ and receives $\mathtt{ct} = \mathsf{AME.Enc}(\mathtt{aPK}, m_0^i)$.
    3. Return $\mathcal{A}$'s output.
```

We note that $\mathcal{A}$ issues encryption queries (m_0^i, m_1^i) and that, clearly, $\mathcal{A}$ can also use the keys in its possession to encrypt and decrypt ciphertexts of its choice. The next lemma proves that $\mathsf{aneGame}$ is indistinguishable from $\mathsf{NormalGame}$ thus yielding that the Normal Mode is indistinguishable from the Anamorphic Mode with Normal Encryption.

Lemma 1. *Let* AME *be a Secure Receiver-AM scheme. Then for all PPT adversaries* $\mathcal{A}$*, we have*
$$\left| p^{\mathsf{NormalGame}}_{\mathsf{AME},\mathcal{A}}(\lambda) - p^{\mathsf{aneGame}}_{\mathsf{AME},\mathcal{A}}(\lambda) \right| \leq \mathsf{negl}(\lambda).$$

Proof. Suppose that there exists an adversary $\mathcal{A}$ that distinguishes $\mathsf{aneGame}$ from $\mathsf{NormalGame}$. We will use $\mathcal{A}$ to construct a dictator $\mathcal{D}$ that distinguishes $\mathsf{NormalGame}$ from $\mathsf{FullyAGame}$, thus violating the security of AME. $\mathcal{D}$ receives the challenge pair of keys $(\mathtt{PK}^\star, \mathtt{SK}^\star)$ and runs $\mathcal{A}$ on input $(\mathtt{PK}^\star, \mathtt{SK}^\star)$. For each query (m_0^i, m_1^i) issued by $\mathcal{A}$, $\mathcal{D}$ replies by returning $\mathsf{Enc}(\mathtt{PK}^\star, m_0^i)$. When $\mathcal{A}$ stops and returns b, $\mathcal{D}$ outputs b. This terminates the description of $\mathcal{D}$. Note that $\mathcal{D}$ issues no encryption query.

Now observe that if $\mathcal{D}$ is playing $\mathsf{FullyAGame}$ then $(\mathtt{PK}^\star, \mathtt{SK}^\star)$ are output by aKG and thus $\mathcal{D}$ is simulating $\mathsf{aneGame}$ for $\mathcal{A}$. On the other hand, if $\mathcal{D}$ is playing $\mathsf{NormalGame}$ then $(\mathtt{PK}^\star, \mathtt{SK}^\star)$ are output by KG and thus $\mathcal{D}$ is simulating $\mathsf{NormalGame}$ for $\mathcal{A}$. This concludes the proof. □

Transitivity of indistinguishability gives the following lemma.

Lemma 2. *Let* AME *be a Secure Receiver-AM scheme. Then for all PPT adversaries* $\mathcal{A}$*, we have*
$$\left| p^{\mathsf{FullyAGame}}_{\mathsf{AME},\mathcal{A}}(\lambda) - p^{\mathsf{aneGame}}_{\mathsf{AME},\mathcal{A}}(\lambda) \right| \leq \mathsf{negl}(\lambda).$$

4.5 Security of the Fully Anamorphic Mode

We are now ready to prove that the fully anamorphic mode of a *Secure Receiver-AM* scheme is IND-CPA even for an adversary that has access to $(\texttt{aPK}, \texttt{aSK})$ but not to $\texttt{dkey}$.

Let us start by formally defining the notion of IND-CPA security of the fully anamorphic mode of a *Secure Receiver-AM* scheme. For $\eta = 0, 1$, message $\hat{m}$, and stateful PPT adversary $\mathcal{A}$, we define game $\mathsf{IndCPA}^{\eta}_{\mathsf{AME},\mathcal{A},\hat{m}}$ as follows.

```
$\mathsf{IndCPA}^{\eta}_{\mathsf{AME},\mathcal{A},\hat{m}}(\lambda)$
    1. Set $(\texttt{aPK}, \texttt{aSK}, \texttt{dkey}) \leftarrow \mathsf{AME.aKG}(1^{\lambda})$.
    2. $(m_1^0, m_1^1) \leftarrow \mathcal{A}^{\mathsf{aEnc}(\texttt{dkey},\hat{m},\cdot)}(\texttt{aPK}, \texttt{aSK})$.
    3. Compute $\texttt{ct} \leftarrow \mathsf{aEnc}(\texttt{dkey}, \hat{m}, m_1^{\eta})$.
    4. $b \leftarrow \mathcal{A}^{\mathsf{aEnc}(\texttt{dkey},\hat{m},\cdot)}(\texttt{ct})$.
```

We denote by $\mathsf{pcpa}^{\eta}_{\mathsf{AME},\mathcal{A},\hat{m}}$ the probability that, in game $\mathsf{IndCPA}^{\eta}_{\mathsf{AME},\mathcal{A},\hat{m}}$, adversary $\mathcal{A}$ outputs 1 . We will prove the following theorem.

Theorem 1. *If* AME *is a Secure Receiver-AM scheme, then, for every message* $\hat{m}$*,* $\mathsf{fAME}_{\hat{m}}$ *is IND-CPA secure. That is, for all PPT adversaries* $\mathcal{A}$ *and every* $\hat{m}$

$$\left|\mathsf{pcpa}^{0}_{\mathsf{AME},\mathcal{A},\hat{m}}(1^{\lambda}) - \mathsf{pcpa}^{1}_{\mathsf{AME},\mathcal{A},\hat{m}}(1^{\lambda})\right| \leq \mathsf{negl}(\lambda).$$

To prove the theorem above, we consider, for $\eta = 0, 1$, the hybrid game $H^{\eta}_{\mathsf{AME},\mathcal{A},\hat{m}}$, in which all $\mathcal{A}$'s oracle calls to $\mathsf{aEnc}(\texttt{dkey}, \hat{m}, \cdot)$ in IndCPA^{η} are replaced with calls to $\mathsf{Enc}(\texttt{aPK}, \hat{m})$ and the challenge ciphertext $\texttt{ct}$ at Line 3 of IndCPA^{η} is computed as $\texttt{ct} \leftarrow \mathsf{Enc}(\texttt{aPK}, \hat{m})$. We denote by $\mathsf{hcpa}^{\eta}_{\mathsf{AME},\mathcal{A},\hat{m}}(1^{\lambda})$ the probability that $\mathcal{A}$ outputs 1 in $H^{\eta}_{\mathsf{AME},\mathcal{A},\hat{m}}$ with security parameter λ.

We have the following lemma whose proof relies on Lemma 2 above.

Lemma 3. *If* AME *is a Secure Receiver-AM then, for* $\eta = 0, 1$*, we have*

$$\left|\mathsf{pcpa}^{\eta}_{\mathsf{AME},\mathcal{A},\hat{m}}(1^{\lambda}) - \mathsf{hcpa}^{\eta}_{\mathsf{AME},\mathcal{A},\hat{m}}(1^{\lambda})\right| \leq \mathsf{negl}(\lambda).$$

Proof. For the sake of contradiction, assume there exists an adversary $\mathcal{A}$ that violates the lemma. We then construct an efficient dictator $\mathcal{D}$ that distinguishes $\mathsf{aneGame}$ and $\mathsf{FullyAGame}$ thus contradicting Lemma 2.

$\mathcal{D}$ receives the pair $(\texttt{aPK}, \texttt{aSK})$ computed by aKG and has access to an encryption oracle $\mathcal{O}(\cdot, \cdot)$. $\mathcal{D}$ runs $\mathcal{A}$ on input $(\texttt{aPK}, \texttt{aSK})$ and when $\mathcal{A}$ issues a query for m, $\mathcal{D}$ returns $\mathcal{O}(\hat{m}, m)$. Similarly, when $\mathcal{A}$ outputs the pair of messages (m_1^0, m_1^1), $\mathcal{D}$ returns $\mathcal{O}(\hat{m}, m_1^{\eta})$. At the end, $\mathcal{D}$ returns $\mathcal{A}$'s output.

If $\mathcal{D}$ is playing game $\mathsf{aneGame}$ then $\mathcal{O}(\hat{m}, m)$ returns $\mathsf{Enc}(\texttt{aPK}, \hat{m})$. Therefore $\texttt{ct}$ is computed as $\texttt{ct} \leftarrow \mathsf{Enc}(\texttt{aPK}, \hat{m})$. This implies that $\mathcal{D}$ simulates $H^{\eta}_{\mathcal{A},\mathsf{AME},\hat{m}}$ for $\mathcal{A}$. On the other hand, if $\mathcal{D}$ is playing $\mathsf{FullyAGame}$ then $\mathcal{O}(\hat{m}, m)$ returns $\mathsf{aEnc}(\texttt{dkey}, \hat{m}, m)$. Therefore $\texttt{ct}$ is computed as $\texttt{ct} \leftarrow \mathsf{Enc}(\texttt{aPK}, \hat{m}, m_1^{\eta})$ and $\mathcal{D}$ simulates $\mathsf{IndCPA}^{\eta}_{\mathcal{A},\mathsf{AME},\hat{m}}(1^{\lambda})$ for $\mathcal{A}$. This concludes the proof of the lemma. □

We are now ready to prove Theorem 1.

Proof of Theorem 1. The theorem follows from Lemma 3 and from the observation that game $H^{\eta}_{\mathcal{A},\mathsf{AME},\hat{m}}$ is independent from η and thus

$$\mathsf{hcpa}^{0}_{\mathcal{A},\mathsf{AME},\hat{m}}(1^{\lambda}) = \mathsf{hcpa}^{1}_{\mathcal{A},\mathsf{AME},\hat{m}}(1^{\lambda}).$$

□

5 Constructions

In this section we present two constructions of AM schemes. We first review the simple construction for a $(\log \lambda)$-bit AM scheme discussed in the Introduction and then, in Section 5.2, we present a construction for $\mathtt{poly}(\lambda)$ bits.

5.1 Rejection Sampling

In this section we present our first construction $\mathsf{1bit}$, a simple cryptosystem that gives guarantees even if the secret-key assumption and the sender-freedom assumption do not hold. It is based on the *rejection sampling* technique inspired by the *biased-ciphertext attack* of [BPR14] and it is used also by Horel et al. [HPRV19]. We review it here just to give a first example of a Receiver-Anamorphic Encryption. We note that the objective of the biased-ciphertext attack is to *subvert* an encryption scheme so that the private key can be leaked without the legitimate owner noticing any abnormal behaviour and it is interesting to note that this technique can be used to setup a dictator setting, as shown in [BPR14], as well as for the opposite of objective, as we show below.

The normal triplet of $\mathsf{1bit}$ is any IND-CPA secure encryption scheme $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ and the anamorphic triplet is defined as follows:

- $\mathsf{1bit.aKG}$, on input the security parameter 1^λ, runs the key generation algorithm KG of $\mathcal{E}$ obtaining $(\mathtt{PK}, \mathtt{SK})$ and random seed K for PRF F. The anamorphic public key is $\mathtt{aPK} = \mathtt{PK}$, the anamorphic secret key is $\mathtt{aSK} = \mathtt{SK}$ and the double key $\mathtt{dkey} = (\mathtt{PK}, K)$.
- $\mathsf{1bit.aEnc}$ takes as input the double key $\mathtt{dkey} = (\mathtt{PK}, K)$ and two messages m_0 and $m_1 \in \{0, 1\}$. Algorithm $\mathsf{1bit.aEnc}$ samples ciphertexts $\mathtt{ct}\mathsf{Enc}(\mathtt{PK}, m_0)$ and the anamorphic ciphertext $\mathtt{act}$ is set equal to the first $\mathtt{ct}$ such that $F(K, \mathtt{ct}) = m_1$.
- $\mathsf{1bit.aDec}$ takes as input $\mathtt{act}$ and $\mathtt{dkey}$ and returns $m_1 = F(K, \mathtt{act})$.

Note that $\mathsf{1bit}$ can be extended to l-bit of hidden plaintexts at the cost of pushing the expect encryption time to $O(2^l)$. Therefore $l = O(\log \lambda)$ hidden bits per ciphertext will keep encryption polynomial and it gives a bandwidth rate of $O((\log \lambda)/\lambda)$. To prove that $\mathsf{1bit}$ is a Secure Anamorphic Encryption we observe that the anamorphic keys have the same distribution as the normal keys. In addition the only difference between the normal ciphertext $\mathtt{ct}$ and the anamorphic ciphertext $\mathtt{act}$ is that the former is randomly distributed over the set of ciphertexts for $\mathtt{PK}$ and m_0 whereas $\mathtt{act}$ is randomly chosen over the set of ciphertexts for $\mathtt{aPK} = \mathtt{PK}$ and m_0 such that $F(K, \mathtt{act}) = m_1$. From the pseudorandomness of F and the fact that K is randomly chosen and hidden from the adversary $\mathcal{A}$, we can conclude that $\mathsf{1bit}$ is secure.

Theorem 2. *If F is a PRF and $\mathcal{E}$ is an IND-CPA Encryption scheme then* $\mathsf{1bit}$ *is a Secure Anamorphic Encryption.*

We also observe that the sender of $\mathsf{1bit}$, should she be forced to encrypt a given message $\mathtt{m}$ and reveal the coin tosses used, can still send a secret bit of her choice to Bob. In other words, the security of $\mathsf{1bit}$ does not rely on the sender-freedom assumption. In Section 6, we will show that this can be achieved by encryption schemes based on lattice hardness assumptions without the sender and receiver having to share a key beforehand.

5.2 The Naor-Yung transform

In this section we describe the Naor-Yung transform [NY90] (see also [Sah99]) that, when applied to an IND-CPA public-key cryptosystem $\mathcal{E}$ and a simulation sound NIZK Π for a specific polynomial-time relation $\mathsf{EqMsg}_{\mathcal{E}}$, gives a CCA public-key cryptosystem NYE. The formal definitions of the concepts used in the construction (IND-CPA, simulation sound NIZK) are found in Appendix A.

The polynomial time relation $\mathsf{EqMsg}_{\mathcal{E}}$ is defined by setting the witness for instance $((\mathtt{PK}_0, \mathtt{ct}_0), (\mathtt{PK}_1, \mathtt{ct}_1))$ to be the triplet (r_0, r_1, m) of two coin tosses and one message m such that

$$\mathtt{ct}_0 = \mathsf{Enc}(\mathtt{PK}_0, m; r_0) \text{ and } \mathtt{ct}_1 = \mathsf{Enc}(\mathtt{PK}_0, m; r_1).$$

The key generation algorithm $\mathsf{NYE.KG}$ constructs the public key $\mathsf{NYE.}\mathtt{PK} = (\mathtt{PK}_0, \mathtt{PK}_1, \Sigma)$ as consisting of two random and independently chosen public keys $\mathtt{PK}_0$ and $\mathtt{PK}_1$ of $\mathcal{E}$ and of a random string Σ. The secret key $\mathsf{NYE.}\mathtt{SK} = (\mathtt{SK}_0)$ associated with $\mathsf{NYE.}\mathtt{PK}$ consists solely of the secret key $\mathtt{SK}_0$ associated with $\mathtt{PK}_0$.

To encrypt message m, the encryption algorithm $\mathsf{NYE.Enc}$ first computes ciphertexts $\mathtt{ct}_0 = \mathsf{Enc}(\mathtt{PK}_0, m; r_0)$ and $\mathtt{ct}_1 = \mathsf{Enc}(\mathtt{PK}_1, m; r_1)$, using random and independent coin tosses r_0 and r_1. Then, it runs the prover's algorithm of Π to produce a proof π that $\mathtt{ct}_0$ and $\mathtt{ct}_1$ encrypt the same message. More precisely, the prover's algorithm of Π is run on input instance $((\mathtt{PK}_0, \mathtt{ct}_0), (\mathtt{PK}_1, \mathtt{ct}_1))$ and witness (r_0, r_1, m) using Σ found in $\mathsf{NYE.}\mathtt{PK}$ as reference string. The ciphertext $\mathtt{PK}$ is finally set to $(\mathtt{ct}_0, \mathtt{ct}_1, \pi)$.

The decryption algorithm $\mathsf{NYE.Dec}$, on input ciphertext $\mathtt{ct} = (\mathtt{ct}_0, \mathtt{ct}_1, \pi)$, runs the verifier algorithm of Π to check π and, if successful, outputs m obtained by decrypting $\mathtt{ct}_0$ using $\mathtt{SK}_0$.

5.3 The NY Transform Gives Receiver-AM Encryption

In this section we describe $Receiver - AM$ scheme NY with bandwidth rate 1 that is derived from NYE. The normal triplet of NY consists of the triple $(\mathsf{NYE.KG}, \mathsf{NYE.Enc}, \mathsf{NYE.Dec})$ described in the previous section. Next we define the anamorphic triplet
$(\mathsf{NYE.aKG}, \mathsf{NYE.aEnc}, \mathsf{NYE.aDec})$.

- The anamorphic key generation algorithm $\mathsf{NYE.aKG}$ constructs the anamorphic public key $\mathsf{NYE.a}\mathtt{PK} = (\mathtt{PK}_0, \mathtt{PK}_1, \Sigma)$ as consisting of two random and independently chosen public keys $\mathtt{PK}_0$ and $\mathtt{PK}_1$ of $\mathcal{E}$; the string Σ instead is obtained by running simulator $S_0(1^\lambda)$ that returns the pair (Σ, aux). The secret key $\mathtt{aSK} = (\mathtt{SK}_0)$ is set equal to the secret key associated with $\mathtt{PK}_0$ whereas the double key is set equal to $\mathtt{dkey} = (\mathtt{PK}_0, \mathtt{PK}_1, \mathtt{SK}_1, \mathsf{aux})$.

- The anamorphic encryption algorithm $\mathsf{NYE.aEnc}$ takes as input the $\mathtt{dkey}$ and two messages m_0 and m_1 and starts constructing the anamorphic ciphertext by computing $\mathtt{ct}_0 = \mathsf{Enc}(\mathtt{PK}_0, m_0)$ and $\mathtt{ct}_1 = \mathsf{Enc}(\mathtt{PK}_1, m_1)$. The proof π is constructed by running the simulator S_1 on input instance $((\mathtt{PK}_0, \mathtt{ct}_0), (\mathtt{PK}_1, \mathtt{ct}_1))$ and auxiliary information aux. Note that m_0 and m_1 are two messages of the same length and thus the construction has a bandwidth rate of 1

- The anamorphic decryption algorithm $\mathsf{NYE.aDec}$ takes an anamorphic ciphertext $\mathtt{act} = (\mathtt{ct}_0, \mathtt{ct}_1, \pi)$ and uses $\mathtt{SK}_1$ found in $\mathtt{dkey}$ to decrypt $\mathtt{ct}_1$ to obtain m_1.

We next prove that the anamorphic encryption scheme NY is secure according to Definition 4; that is, that games $\mathsf{NormalGame}$ and $\mathsf{FullyAGame}$ are indistinguishable. To do so, we consider the hybrid game aneG and will show that aneG and $\mathsf{NormalGame}$ are indistinguishable by the security properties of the NIZK and that aneG and $\mathsf{FullyAGame}$ are indistinguishable by the security of the encryption scheme $\mathcal{E}$. This implies that that $\mathsf{NormalGame}$ and $\mathsf{FullyAGame}$ are indistinguishable thus satisfying Definition 4. Let us now proceed more formally by instantiating game $\mathsf{NormalGame}_{\mathsf{NY}}$. We remind the reader that $p^{\mathsf{NormalGame}}_{\mathsf{NY},\mathcal{A}}(\lambda)$ denote the probability that adversary $\mathcal{A}$ outputs 1 in game $\mathsf{NormalGame}_{\mathsf{NY},\mathcal{A}}$. Next we define game $\mathsf{aneG}_{\mathsf{NY}}$ that differs from $\mathsf{NormalGame}_{\mathsf{NY}}$ in the way in which the public key and ciphertexts $\mathtt{ct}_i$, replies to $\mathcal{A}$'s queries, are computed. Specifically, the reference string Σ, that is part of the public key, and the proof π^i, that is part of the i-th ciphertext $\mathtt{ct}^i$, are computed by running the simulator. We define $p^{\mathsf{aneG}}_{\mathsf{NY},\mathcal{A}}(\lambda)$ to be the probability that adversary $\mathcal{A}$ outputs 1 in game $\mathsf{aneG}_{\mathsf{NY},\mathcal{A}}$.

$\mathsf{NormalGame}_{\mathsf{NY},\mathcal{A}}(\lambda)$

1. Set $(\mathtt{PK}_0, \mathtt{SK}_0), (\mathtt{PK}_1, \mathtt{SK}_1) \leftarrow \mathcal{E}.\mathsf{KG}(1^\lambda)$.
2. Set $\Sigma \leftarrow \{0,1\}^\lambda$.
3. Send $\mathtt{PK} = (\mathtt{PK}_0, \mathtt{PK}_1, \Sigma)$ and $\mathtt{SK} = (\mathtt{SK}_0)$ to $\mathcal{A}$.
4. For $i = 1, \ldots, \mathsf{poly}(\lambda)$:
   $\mathcal{A}$ issues query (m_0^i, m_1^i) and receives $\mathtt{ct}^i$ computed as follows:
   - Set $\mathtt{ct}_0^i = \mathsf{Enc}(\mathtt{PK}_0, m_0^i; r_0^i)$.
   - Set $\mathtt{ct}_1^i = \mathsf{Enc}(\mathtt{PK}_1, m_0^i; r_1^i)$.
   - Set $\pi^i \leftarrow \mathsf{Prover}(((\mathtt{PK}_0, \mathtt{ct}_0^i), (\mathtt{PK}_1, \mathtt{ct}_1^i)), (r_0^i, r_1^i), \Sigma)$.
   - Set $\mathtt{ct}^i = (\mathtt{ct}_0^i, \mathtt{ct}_1^i, \pi^i)$.
5. $\mathcal{A}$ outputs $b \in \{0,1\}$.

$\mathsf{aneG}_{\mathsf{NY},\mathcal{A}}(\lambda)$

1. Set $(\mathtt{PK}_0, \mathtt{SK}_0), (\mathtt{PK}_1, \mathtt{SK}_1) \leftarrow \mathcal{E}.\mathsf{KG}(1^\lambda)$.
2. Set $(\Sigma, \mathsf{aux}) \leftarrow S_0(1^\lambda)$ and $\mathtt{dkey} = (\mathtt{PK}_0, \mathtt{PK}_1, \mathtt{SK}_1, \mathsf{aux})$.
3. Send $\mathtt{PK} = (\mathtt{PK}_0, \mathtt{PK}_1, \Sigma)$ and $\mathtt{SK} = (\mathtt{SK}_0)$ to $\mathcal{A}$.
4. For $i = 1, \ldots, \mathsf{poly}(\lambda)$:
   $\mathcal{A}$ issues query (m_0^i, m_1^i) and receives $\mathtt{ct}^i$ computed as follows:
   - Set $\mathtt{ct}_0^i = \mathsf{Enc}(\mathtt{PK}_0, m_0^i; r_0^i)$.
   - Set $\mathtt{ct}_1^i = \mathsf{Enc}(\mathtt{PK}_1, m_0^i; r_1^i)$.
   - Set $\pi^i \leftarrow S_1(((\mathtt{PK}_0, \mathtt{ct}_0^i), (\mathtt{PK}_1, \mathtt{ct}_1^i)), \mathsf{aux})$.
   - Set $\mathtt{ct}^i = (\mathtt{ct}_0^i, \mathtt{ct}_1^i, \pi^i)$.
5. $\mathcal{A}$ outputs $b \in \{0,1\}$.

Lemma 4. *If $\Pi = (\mathsf{Prover}, \mathsf{Verifier})$ is a NIZK then for all PPT adversaries $\mathcal{A}$*

$$\left|p^{\mathsf{aneG}}_{\mathsf{NY},\mathcal{A}}(\lambda) - p^{\mathsf{NormalGame}}_{\mathsf{NY},\mathcal{A}}(\lambda)\right| < \mathsf{negl}(\lambda).$$

Proof. Suppose, for the sake of contradiction, that there exists a PPT $\mathcal{A}$ for which

$$\left|p^{\mathsf{aneG}}_{\mathsf{NY},\mathcal{A}}(\lambda) - p^{\mathsf{NormalGame}}_{\mathsf{NY},\mathcal{A}}(\lambda)\right| \geq 1/\mathsf{poly}(\lambda).$$

Then we construct an adversary $\mathcal{B}$ that receives a string Σ and has access to an oracle $\mathcal{O}$ that returns proofs and that breaks the zero-knowledge property of Π.

1. $\mathcal{B}(\Sigma)$ sets $(\mathtt{PK}_0, \mathtt{SK}_0), (\mathtt{PK}_1, \mathtt{SK}_1) \leftarrow \mathcal{E}.\mathsf{KG}(1^\lambda)$.
2. $\mathcal{B}$ receives Σ and starts $\mathcal{A}$ on input $\mathtt{PK} = (\mathtt{PK}_0, \mathtt{PK}_1, \Sigma)$ and $\mathtt{SK} = (\mathtt{SK}_0)$.
3. For $i = 1, \ldots, \mathsf{poly}(\lambda)$
   (a) $\mathcal{B}$ receives query (m_0^i, m_1^i) from $\mathcal{A}$.

(b) $\mathcal{B}$ sets $\texttt{ct}_0^i = \mathsf{Enc}(\texttt{PK}_0, m_0^i; r_0^i)$.

(c) $\mathcal{B}$ sets $\texttt{ct}_1^i = \mathsf{Enc}(\texttt{PK}_1, m_0^i; r_1^i)$.

(d) $\mathcal{B}$ invokes the oracle $\mathcal{O}$ on input instance $((\texttt{PK}_0, \texttt{ct}_0^i), (\texttt{PK}_1, \texttt{ct}_1^i))$ and witness (r_0^i, r_1^i) and receives proof π^i.

(e) $\mathcal{B}$ returns $\texttt{ct}^i = (\texttt{ct}_0^i, \texttt{ct}_1^i, \pi^i)$ to $\mathcal{A}$.

4. $\mathcal{B}$ returns $\mathcal{A}$'s output.

Now observe that if $\mathcal{B}$ is playing experiment RealZK, then Σ is randomly chosen from $\{0,1\}^\lambda$ and $\mathcal{O} = \mathsf{Prover}$ and therefore $\mathcal{A}$'s view is the same as in $\mathsf{NormalGame}$. On the other hand, if $\mathcal{B}$ is playing experiment $\mathsf{IdealZK}$ then Σ is output by S_0 and $\mathcal{O} = \mathsf{Oracle}$ and therefore $\mathcal{A}$'s view is the same as in aneG. Since we assumed that $\mathcal{A}$ distinguishes between $\mathsf{NormalGame}$ and aneG then we can conclude that $\mathcal{B}$ breaks the Zero-Knowledge property of Π. □

Next we instantiate the anamorphic security game $\mathsf{FullyAGame}_{\mathsf{NY}}$. We remind the reader that $p_{\mathsf{NY},\mathcal{A}}^{\mathsf{FullyAGame}}(\lambda)$ denote the probability that adversary $\mathcal{A}$ outputs 1 in game $\mathsf{FullyAGame}_{\mathsf{NY},\mathcal{A}}$.

$\mathsf{FullyAGame}_{\mathsf{NY},\mathcal{A}}(\lambda)$

1. Set $(\texttt{PK}_0, \texttt{SK}_0), (\texttt{PK}_1, \texttt{SK}_1) \leftarrow \mathcal{E}.\mathsf{KG}(1^\lambda)$.
2. Set $(\Sigma, \mathsf{aux}) \leftarrow S_0(1^\lambda)$ and $\texttt{dkey} = (\texttt{PK}_0, \texttt{PK}_1, \texttt{SK}_1, \mathsf{aux})$.
3. Send $\texttt{PK} = (\texttt{PK}_0, \texttt{PK}_1, \Sigma)$ and $\texttt{SK} = (\texttt{SK}_0)$ to $\mathcal{A}$.
4. For $i = 1, \ldots, \mathsf{poly}(\lambda)$:
   $\mathcal{A}$ issues query (m_0^i, m_1^i) and receives $\texttt{ct}^i$ computed as follows:
   - Set $\texttt{ct}_0^i = \mathsf{Enc}(\texttt{PK}_0, m_0^i; r_0^i)$.
   - Set $\texttt{ct}_1^i = \mathsf{Enc}(\texttt{PK}_1, m_1^i; r_1^i)$.
   - Set $\pi^i \leftarrow S_1(((\texttt{PK}_0, \texttt{ct}_0^i), (\texttt{PK}_1, \texttt{ct}_1^i)), \mathsf{aux})$.
   - Set $\texttt{ct}^i = (\texttt{ct}_0^i, \texttt{ct}_1^i, \pi^i)$.
5. $\mathcal{A}$ outputs $b \in \{0,1\}$.

We have the following Theorem.

Theorem 3. *If $\mathcal{E}$ is an IND-CPA secure encryption scheme then for all PPT adversaries $\mathcal{A}$*

$$\left| p_{\mathsf{NY},\mathcal{A}}^{\mathsf{aneG}}(\lambda) - p_{\mathsf{NY},\mathcal{A}}^{\mathsf{FullyAGame}}(\lambda) \right| < \mathsf{negl}(\lambda).$$

Proof. Suppose, for the sake of contradiction, that there exists a PPT $\mathcal{A}$ for which

$$\left| p_{\mathsf{NY},\mathcal{A}}^{\mathsf{aneG}}(\lambda) - p_{\mathsf{NY},\mathcal{A}}^{\mathsf{FullyAGame}}(\lambda) \right| \geq 1/\mathsf{poly}(\lambda).$$

Then we construct an adversary $\mathcal{B}$ that receives a public key $\texttt{PK}$ and has an encryption oracle $\mathcal{O}(\texttt{PK}, \cdot, \cdot)$ that returns a ciphertext.

1. $\mathcal{B}(\texttt{PK})$ sets $(\texttt{PK}_0, \texttt{SK}_0)$ and $\texttt{PK}_1 = \texttt{PK}$.
2. $\mathcal{B}$ sets $(\Sigma, \mathsf{aux}) \leftarrow S_0(1^\lambda)$ and runs $\mathcal{A}$ on input $\texttt{PK} = (\texttt{PK}_0, \texttt{PK}_1, \Sigma)$ and $\texttt{SK} = (\texttt{SK}_0)$.
3. For $i = 1, \ldots, \mathsf{poly}(\lambda)$:
   $\mathcal{A}$ issues query (m_0^i, m_1^i) and receives $\texttt{ct}^i$ computed as follows:

- $\mathcal{B}$ sets $\mathtt{ct}_0^i = \mathsf{Enc}(\mathtt{PK}_0, m_0^i; r_0^i)$;
- $\mathcal{B}$ sets $\mathtt{ct}_1^1 = \mathcal{O}(\mathtt{PK}, m_0^i, m_1^i)$;
- $\mathcal{B}$ sets $\pi^i \leftarrow S_1(((\mathtt{PK}_0, \mathtt{ct}_0^i), (\mathtt{PK}_1, \mathtt{ct}_1^i)), \mathsf{aux})$.
- $\mathcal{B}$ sets $\mathtt{ct}^i = (\mathtt{ct}_0^i, \mathtt{ct}_1^i, \pi^i)$.

4. $\mathcal{B}$ returns $\mathcal{A}$'s output.

Now observe that if $\mathcal{B}$ is playing the $\mathsf{IndCPA}_{\mathcal{E}}^0$ game then $\mathcal{O}(\mathtt{PK}, m_0, m_1) = \mathsf{Oracle}^0(\mathtt{PK}, m_0, m_1)$ returns an encryption of m_0 and therefore $\mathcal{A}$'s view is exactly as in $\mathsf{aneG}_{\mathsf{NY}}$. On the other hand, if $\mathcal{B}$ is playing the $\mathsf{IndCPA}_{\mathcal{E}}^0$ game then $\mathcal{O}(\mathtt{PK}, m_0, m_1) = \mathsf{Oracle}^1(\mathtt{PK}, m_0, m_1)$ returns an encryption of m_1 and therefore $\mathcal{A}$'s view is exactly as in $\mathsf{FullyAGame}_{\mathsf{NY}}$. Since we assumed that $\mathcal{A}$ distinguishes between $\mathsf{aneG}_{\mathsf{NY}}$ and $\mathsf{FullyAGame}_{\mathsf{NY}}$, we can conclude that $\mathcal{B}$ breaks the IND-CPA security of $\mathcal{E}$. $\square$

Bootstrapping the NY *construction.* In the description of the anamorphic triplet for the NY construction we have assumed that the sender and the receiver have a safe way to exchange `dkey`. This is a reasonable assumption for most applications as it is expected that the dictator is able to monitor online communication but it will be difficult to monitor a physical exchange. We next briefly note how to use the bootstrap technique of Horel et al. [HPRV19] in combination with the NY construction. Note that the result scheme does not enjoy the zero-latency property. Observe that `dkey` consists of $(\mathtt{PK}_0, \mathtt{PK}_1, \mathtt{SK1}, \mathsf{aux})$. However $\mathtt{PK}_0$ and $\mathtt{PK}_1$ are public and $\mathtt{SK}_1$ is not used by the sender and it is only used by the receiver to decrypt the message. Therefore, it is enough for the two parties to share the random string used by simulator S_0 to generate aux and this can be achieved by using the two-step bootstrap procedure of Horel et al. [HPRV19].

6 Sender-Anamorphic Encryption

In this section we address the sender-freedom assumption. In the introduction, and more formally in Section 5.1, we presented a simple cryptosystem whose security does not rely on the sender-freedom assumption. Specifically, Alice, fearing that she might be forced by the authorities to send a fake message, sets up a private shared key K with Bob that allows to produce, for every adversarially chosen message $\mathtt{m}_0$, a ciphertext `ct` carrying $\mathtt{m}_0$ and the coin tosses used to produce `ct` such that, when decrypted with K, `ct` gives a one-bit message $\mathtt{m}_1$ for Bob to receive privately. The main drawback of this setting is that Alice and Bob must interact in advance to share the key K and this is not always possible: Alice might not know of Bob when she first sets up her public key or might not have a way of securely sending K to Bob.

Next, we show that prior communication between Alice and Bob is not necessary and formalize the concept of a Sender-Anamorphic Encryption in this context. It seems again impossible (seems like decryption is not well defined?). However, the existence of other receivers and public channels save the day!

Specifically, when Alice is forced by the authorities to send the *forced* message $\mathtt{m}_0$ to Carol using Carol's public key `fPK` as the *forced* receiving public key, Alice might decide to embed a duplicate message $\mathtt{m}_1$ in the ciphertext that is revealed when the ciphertext is decrypted with Bob's key; that is, the secret key associated with the *duplicate* public key `dPK`. We stress again that `dPK`, Bob's public key, is generated by Bob without even knowing that one day he might receive a message from Alice and no secret is shared between Alice and Bob. To do so, we equip Alice with a special coin-toss faking algorithm $\mathsf{fRandom}$ that on input the *forced* public key `fPK`, the *duplicate* public key `dPK`, the *forced* message $\mathtt{m}_0$, and the *duplicate* message $\mathtt{m}_1$ outputs coin tosses to produce a ciphertext `ct` that gives $\mathtt{m}_0$ or $\mathtt{m}_1$ depending on whether it is decrypted with the secret key associated with the forced public key or with the secret key associated with the duplicate public key.

We next present formal definition for *Sender-Anamorphic Encryption.* Our definition is tailored for the no-shared secret setting that we have described. Similarly to the receiver side, it could be possible to define the notion of Sender-Anamorphic Encryption also for the case in which sender and receiver share a secret.

Definition 3. *We say that a public-key encryption scheme $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ is a* Sender-Anamorphic Encryption scheme *(Sender-AM) if there exists a* coin-toss faking *algorithm* $\mathsf{fRandom}$ *that, on input the* forced *public key* $\mathtt{fPK}$*, and the* forced message $\mathtt{m}_0$*, and the* duplicate *public key* $\mathtt{dPK}$ *and the duplicate message* $\mathtt{m}_1$*, outputs the* faking coin tosses $R^\star = \mathsf{fRandom}(\mathtt{fPK}, \mathtt{m}_0, \mathtt{dPK}, \mathtt{m}_1)$ *such that*

- *Let* $\mathtt{ct} = \mathsf{Enc}(\mathtt{fPK}, \mathtt{m}_0; R^\star)$ *be the ciphertext computed using the faking coin tosses; then* $\mathsf{Dec}(\mathtt{dSK}, \mathtt{ct}) = \mathtt{m}_1$*, except with negligible probability. The probability is taken over the coin tosses of* fRandom *and the coin tosses used to generate* $\mathtt{fPK}$ *and* $\mathtt{dPK}$*.*

To define the concept of a Secure Sender-Anamorphic Encryption we introduce two experiments. In both experiments, the adversary receives one public key $\mathtt{fPK}$ and gives polynomially many pairs of forced and duplicate message and for each pair receives as a reply a ciphertext for the first message and the random coin tosses used to produce it. In the real experiment, the ciphertext is produced by Alice that uses random coin tosses output by fRandom with respect to randomly chosen duplicate key. We require the real experiment to be indistinguishable from the ideal experiment in which there is only one public key and the forced message is encrypted by using truly random coin tosses.

```
$\mathsf{Ideal}_{\mathcal{E},\mathcal{A}}(\lambda)$
    1. Set $(\mathtt{fPK}, \mathtt{fSK}) \leftarrow \mathsf{KG}(1^\lambda)$ and send $\mathtt{fPK}$ to $\mathcal{A}$.
    2. For $i = 1, \ldots, \mathsf{poly}(\lambda)$:
        • $\mathcal{A}$ issues query $(\mathtt{m}_0^i, \mathtt{m}_1^i)$ and receives $\mathtt{ct} = \mathsf{Enc}(\mathtt{fPK}, \mathtt{m}_0^i; R^i)$,
          where $R_i$ are randomly chosen coin tosses.
    3. $\mathcal{A}$ outputs $b \in \{0, 1\}$.
```

```
$\mathsf{Real}_{\mathcal{E},\mathcal{A}}^{\mathsf{fRandom}}(\lambda)$
    1. Set $(\mathtt{fPK}, \mathtt{fSK}) \leftarrow \mathsf{KG}(1^\lambda)$ and send $\mathtt{fPK}$ to $\mathcal{A}$.
    2. Set $(\mathtt{dPK}, \mathtt{dSK}) \leftarrow \mathsf{KG}(1^\lambda)$.
    3. For $i = 1, \ldots, \mathsf{poly}(\lambda)$:
        • $\mathcal{A}$ issues query $(\mathtt{m}_0^i, \mathtt{m}_1^i)$ and receives $\mathtt{ct} = \mathsf{Enc}(\mathtt{fPK}, \mathtt{m}_0^i; R^i)$,
          where $R^i = \mathsf{fRandom}(\mathtt{fPK}, \mathtt{m}_0^i, \mathtt{dPK}, \mathtt{m}_1^i)$.
    4. $\mathcal{A}$ outputs $b \in \{0, 1\}$.
```

We denote by $p_{\mathcal{E},\mathcal{A}}^{\mathsf{Ideal}}(\lambda)$ (respectively, $p_{\mathcal{E},\mathcal{A}}^{\mathsf{Real}}(\lambda)$) the probability that $\mathsf{Ideal}_{\mathcal{E},\mathcal{A}}$ (respectively, $\mathsf{Real}_{\mathcal{E},\mathcal{A}}$) outputs 1 and present the following definition.

Definition 4. *A* Sender-AM *scheme* $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *with coin-toss faking algorithm* fRandom *is a SSender-AM scheme if*

1. *$\mathcal{E}$ is an IND-CPA scheme;*
2. *for all PPT adversaries $\mathcal{A}$,*
$$\left|p_{\mathcal{E},\mathcal{A}}^{\mathsf{Ideal}}(\lambda) - p_{\mathcal{E},\mathcal{A}}^{\mathsf{Real}}(\lambda)\right| \leq \mathsf{negl}(\lambda).$$

Deniable encryption comparison. We stress again that the guarantee offered by Anamorphic Encryption with no shared key is different from, and in a sense stronger than, the guarantees of Deniable Encryption. Specifically,

1. Deniable encryption helps the sender to deny having sent some plaintext. It allows Alice to send an encryption $\mathtt{ct}$ of a message m to Bob then deny it by presenting fake randomness and a fake message m' that corresponds to $\mathtt{ct}$, under the *same* public key of Bob. Because $\mathtt{ct}$ correspond to two different messages m and m' under the same public key of Bob, it was specified in [CDNO97] that no standard encryption guarantees deniability. This contradicts our objective to use standard encryptions to avoid any suspicion.

2. The security threat we consider is different. While deniable encryption considers the scenario where the adversaries approaches Alice *after* the ciphertext was transmitted, we consider the situation where the adversary approaches Alice *before* and forces Alice to send a *specific* message chosen by the adversary.

3. AM with no-shared key could serve for some kinds of deniability: To deny sending a message m to Bob, we do not need to present another message for Bob. In the 90's where communication was almost about point-to-point communication, Eve can be sure that Alice did send the ciphertext to Bob. Therefore, it makes sense to require Alice to present another message under Bob's public key. Nowadays, to send a message to Bob, Alice could simply broadcast the ciphertext over public channels or puts it in a cloud database/ blockchain and anyone can easily get it. Therefore, to deny sending a message m to Bob, Alice can say that she sends another message m' to Carol and prove that the suspected ciphertext corresponds indeed to an encryption of m' under the public key of Carol. AM with no-shared key can exactly serve this type of deniability by avoiding the restriction of presenting two different messages under the *same* public key, we can use standard encryptions for AM.

Difficulties in achieving Sender-AM with no shared key. Not every standard encryption can be used as AM with no shared key. In order to embed a secret message in a ciphertext for public key $\mathtt{fPK}$ a valid ciphertext for Bob's public key $\mathtt{PK}$, it must be the case that the ciphertext is a valid ciphertext for both keys. From this point of view, the schemes with redundancy (in particular, with validity check in the decryption) seem difficult to employ because a ciphertext for Bob should probably be an invalid ciphertext to Carlos. Almost all known CCA encryption with redundancy (like the one based on the Naor-Yung transform) cannot be used in this context.

But the lack of redundancy is not sufficient for a scheme to be used as AM with no shared key. For the schemes that do not have the property of common randomness, we do not know how to add secret messages to ciphertexts. For example, in Goldwasser-Micali cryptosystem [GM84], the very first semantically secure encryption, two users generate two independent moduli N_0 and N_1 and the secret key of one system is independent of the other one. Therefore, given a ciphertext $\mathtt{ct}$ for N_0, without the knowledge of the factorization of N_1, it is hard to decide whether this ciphertext is a quadratic residue modulo N_1. As a result, Alice does not know the underlying plaintext for the ciphertext $\mathtt{ct}$ with respect to Bob's key. Consequently, this gives evidence that it seems impossible for Alice, in the no-shared-key model, to embed a secret message to a ciphertext in the Goldwasser-Micali cryptosystem. We next identify a set of sufficient conditions for AM with no shared key and then we describe some encryption schemes from the literature that satisfy the conditions.

6.1 Sufficient Conditions for Sender-AM with no shared key

In this section, we introduce three sufficient conditions for a one-bit cryptosystem $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ to be Sender-AM with no shared key.

1. *Common randomness property.* Public key encryption scheme $\mathcal{E}$ satisfies the common randomness property if:

   for every two public keys $\mathtt{PK}_0$ and $\mathtt{PK}_1$ output by KG and for every ciphertext $\mathtt{ct}$ produced using public key $\mathtt{PK}_0$ and coin tosses R there exists a message m such that $\mathtt{ct} = \mathsf{Enc}(\mathtt{PK}_1, m; R)$; that is, $\mathtt{ct}$ is a ciphertext also for public key $\mathtt{PK}_1$ with the *same* coin tosses R.

   For example, the Elgamal encryption-s over a public group satisfies this common randomness property.

2. *Message recovery from randomness.* A public key encryption scheme $\mathcal{E}$ satisfies the common randomness property if:

   it is possible to recover the plaintext carried by a ciphertext $\mathtt{ct}$ from the randomness R used to produce it and the public key $\mathtt{PK}$.

3. *Equal Distribution of Plaintexts.* A one-bit public key encryption scheme satisfies the property of equal distribution of plaintexts if the following properties are verified:

   - all public keys share the same ciphertext space;
   - for a ciphertext $\mathtt{ct}$ in the common ciphertext space and for a random key pair $(\mathtt{PK}, \mathtt{SK})$ as sampled by the key generation algorithm, $\mathtt{ct}$ is the ciphertext of the bit 0 with a probability $\frac{1}{2}$.

We now show that, if a one-bit public key encryption satisfies the above three properties, it is a Sender-AM with no shared key.

Theorem 4. *If an IND-CPA secure one-bit public key encryption scheme satisfies the three properties of common randomness, message recovery from randomness, and equal distribution of plaintexts, then it is a Sender-AM scheme with no shared key.*

Proof. Let $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ be an IND-CPA secure scheme and define $\mathsf{fRandom}(\mathtt{fPK}, m_0, \mathtt{dPK}, m_1)$ as follows:

1. Randomly choose coin tosses R;
2. Compute $\mathtt{ct} = \mathsf{Enc}(\mathtt{fPK}, m_0; R)$;
3. By the property of common randomness, $\mathtt{ct}$ is also a ciphertext according to $\mathtt{dPK}$ and it is computed using the same coin tosses R.
4. By the property of message recovery from randomness, message m' carried by $\mathtt{ct}$ w.r.t. to $\mathtt{dPK}$ can be computed; that is $\mathsf{Enc}(\mathtt{dPK}, m'; R)$ is equal to $\mathtt{ct}$.
5. If $m' = m_1$ then return $(\mathtt{ct}, R)$ and halt; otherwise, go back to Step 1.

The $\mathsf{fRandom}$ defined above satisfies Definition 3. Indeed, the correctness is directly verified. Now observe that the coin tosses R selected at step 1 of $\mathsf{fRandom}$ have probability 1/2 of being the output in the last step, over the choices of random $\mathtt{dPK}$, by the equal distribution of plaintexts. As $\mathtt{dPK}$ is uniformly distributed (and unknown) to the adversary, the adversary cannot distinguish the outputted coin tosses with a real randomness.

□

6.2 Constructions based on LWE Encryption schemes

Theorem 5. *The LWE Encryption and the Dual LWE Encryption are each a Sender-AM scheme with no shared key.*

Proof. A detailed exposition of LWE Encryption, Dual LWE Encryption can be found in [Reg05, GPV08]. We recall here a quick description of these schemes and their properties assuring that they are AM schemes with no shared key.

We first recall the LWE encryption [Reg05] in Figure 2. It is parameterized by some $r \leq \omega(\sqrt{\log m}))$, which specifies the discrete Gaussian distribution $D_{\mathbb{Z}^m,r}$ over the integer lattice $\mathbb{Z}^m$ from which the secret keys are chosen.

- From the ciphertext $(\mathbf{a}, b)$, we see that, for any user, the implicit random input is the same. This satisfies the property of common randomness.

Setup: The system is characterized by m, q and a probability distribution χ on $\mathbb{Z}_q^n$. All users share a common matrix $\mathbf{A} \in \mathbb{Z}_q^{nm}$ chosen uniformly at random, which is the index of the function $f_{\mathbf{A}}(\mathbf{e}) = \mathbf{A}\mathbf{e} \mod q$.

Key Generation:

- Choose $s \in \mathbb{Z}_q^n$ uniformly at random. The private key is s.
- Choose error vector $\mathbf{e} \leftarrow D_{\mathbb{Z}^m,r}$.The public key consists of $(\mathbf{A}, \mathbf{b} = f_{\mathbf{A}}(\mathbf{s}) + \mathbf{e})$

Encryption: The encryption of a bit $x \in \{0,1\}$ is done by choosing a random subset S of $[m]$ and then defining the ciphertext of x as $(\sum_{i\in S} \mathbf{a}_i, x\lfloor \frac{q}{2} \rfloor + \sum_{i\in S} b_i)$

Decryption: The decryption of $(\mathbf{a}, b)$ is 0 if $b - \langle \mathbf{a}, \mathbf{s} \rangle$ is closer to 0 than to $\lfloor \frac{q}{2} \rfloor$ modulo q, and 1 otherwise.

Figure 2: LWE public-key encryption

- From the random input and the public-key, one can get the plaintext. This satisfies the property of message recovery from randomness.
- For a ciphertext $(\mathbf{a}, b)$, with a public key that is returned from the key generation, the vector $\vec{b}$ is indistinguishable from a randomly chosen vector in $\mathbb{Z}_q^n$, the underlying plaintext is thus 0 with probability $\frac{1}{2}$. This satisfies the property of equal distribution of plaintexts.

We now recall the Dual LWE encryption [GPV08] in Figure 3. It is parameterized by two integers m, q and a probability distribution χ on $\mathbb{Z}_q$. We show that it supports AM in the similar way as in the case of the LWE encryption:

- From the ciphertext $(\mathbf{p}, c)$, we see that, for any user, the implicit random input is the same. This gives the property of common randomness.
- From the random input and the public-key, one can get the plaintext. This satisfies the property of message recovery from randomness.
- For a ciphertext $(\mathbf{p}, c)$, with a public key that is returned from the key generation, the vector $\mathbf{u}^T\mathbf{s}$ is indistinguishable from a randomly chosen vector in $\mathbb{Z}_q$, the underlying plaintext is thus 0 with probability $\frac{1}{2}$. This satisfies the property of equal distribution of plaintexts.

$\square$

7 Conclusion

This is the first and most likely not the last work on Anamorphic Encryption as our framework does not cover all the schemes. In this first work, we have shown the anamorphic property for some standard encryptions with specific properties. This does not mean that it is impossible for schemes without these properties. It is also reasonable to wonder what would happen if the dictator can ban these schemes from being used. However, in this way, the dictator will always need to run after our research and this will force the dictator to assure that his children will get a PhD in Cryptography.

The open natural question of this work is whether the systems are unique examples or this is a prevalent phenomenon. In a coming work we introduce numerous new techniques for anamorphic cryptography, and we

Setup: all users share a common matrix $\mathbf{A} \in \mathbb{Z}_q^{nm}$ chosen uniformly at random, which is the index of the function $f_{\mathbf{A}}(\mathbf{e}) = \mathbf{A}\mathbf{e} \mod q$.

Key Generation:

- Choose an error vector $\mathbf{e} \leftarrow D_{\mathbb{Z}^m,r}$ which is the secret key.
- The public key is the syndrome $\mathbf{u} = f_{\mathbf{A}}(\mathbf{e})$.

Encryption: to encrypt a bit $b \in \{0,1\}$, choose $\mathbf{s} \in \mathbb{Z}_q^n$ uniformly and $\mathbf{p} = \mathbf{A}^T\mathbf{s} + \mathbf{x} \in \mathbb{Z}_q^m$, where $\mathbf{x} \in \chi^m$. Output the ciphertext $(\mathbf{p}; c = \mathbf{u}^T\mathbf{s} + x + b\lfloor \frac{q}{2} \rfloor) \in \mathbb{Z}_q^m \times \mathbb{Z}_q$, where $x \leftarrow \chi$.

Decryption: Compute $z = c - \mathbf{e}^T\mathbf{p} \in \mathbb{Z}_q$. Output 0 if z is closer to 0 than to $\lfloor \frac{q}{2} \rfloor \mod q$, otherwise output 1.

Figure 3: Dual LWE public-key encryption

further demonstrate how they cover a wide area of known cryptosystems, making anamorphic cryptography a much more viable tool for bypassing dictators and a possible stronger answer to the futility of crypto wars.

Of course, we leave it to others to determine, based on policy, law, and other societal aspects beyond pure technology, whether our results are aiming toward being the final nail in the coffin of governments control of the use of strong cryptographic systems. Our meta-conjecture (and induced policy implication) is that *for any standard scheme, there is a technical demonstration (perhaps employing Anamorphism+stego+klepto) of the futility of the dictator's demands.*

Acknowledgments

We thank all anonymous reviewers for insightful feedback. Duong Hieu Phan was partially supported by the ANR ALAMBIC (ANR16-CE39-0006). Part of this work was done while Giuseppe Persiano was visiting Google NY.

References

[AAB+97] H. Abelson, R. Anderson, S. M. Bellovin, J. Benaloh, M. Blaze, W. Diffie, J. Gilmore, P. G. Neumann, R. L. Rivest, J. I. Schiller, and B. Schneier. The risks of key recovery, key escrow, and trusted third-party encryption, 1997. `https://doi.org/10.7916/D8GM8F2W`.

[AAB+15] H. Abelson, R. Anderson, S. M. Bellovin, J. Benaloh, M. Blaze, W. Diffie, J. Gilmore, M. Green, S. Landau, P. G. Neumann, R. L. Rivest, J. I. Schiller, B. Schneier, M. A. Specter, and D. J. Weitzner. Keys under doormats: mandating insecurity by requiring government access to all data and communications, 2015. `https://doi.org/10.7916/D8H41R9K`.

[ACD19] Joël Alwen, Sandro Coretti, and Yevgeniy Dodis. The double ratchet: Security notions, proofs, and modularization for the Signal protocol. In *EUROCRYPT 2019*, pages 129–158, 2019.

[BDSMP91] Manuel Blum, Alfredo De Santis, Silvio Micali, and Giuseppe Persiano. Noninteractive zero-knowledge. *SIAM J. Comput.*, 20(6):1084–1118, December 1991.

[Bla94] Matt Blaze. Protocol Failure in the Escrowed Encryption Standard. In *CCS '94*, pages 59–67, 1994.

[BPR14] Mihir Bellare, Kenneth G. Paterson, and Phillip Rogaway. Security of symmetric encryption against mass surveillance. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO '14*, pages 1–19, 2014.

[CDNO97] Ran Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable encryption. In *CRYPTO '97*, pages 90–104, 1997.

[Cli] *Statement by the Press Secretary, The White House, April 16, 1993.* Reprinted in David Banisar (ed.), 1994, Cryptography and Privacy Sourcebook.

[CNE$^+$14] S. Checkoway, R. Niederhagen, A. Everspaugh, M. Green, T. Lange, T. Ristenpart, D. J. Bernstein, J. Maskiewicz, H. Shacham, and M. Fredrikson. On the Practical Exploitability of Dual EC in TLS Implementations. In *USENIX Security Symposium*, pages 319–335, 2014.

[Dak96] Howard S. Dakoff. The clipper chip proposal: Deciphering the unfounded fears that are wrongfully derailing its implementation. *J. Marshall L. Rev.*, 29, 1996.

[DH76] Whitfield Diffie and Martin E. Hellman. New Directions in Cryptography. *IEEE Trans. on Information Theory*, 22:644–654, 1976.

[DL] Whitfield Diffie and Susan Landau. *Privacy on the Line. The Politics of Wiretapping and Encryption.*

[Fei73] Horst Feistel. Cryptography and Computer Privacy. *Scientific American*, 228(5), 1973.

[FLS99] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple noninteractive zero knowledge proofs under general assumptions. *SIAM Journal on Computing*, 29(1):1–28, 1999.

[FY93] Yair Frankel and Moti Yung. Escrow Encryption Systems Visited: Attacks, Analysis and Designs. In *CRYPTO 94*, pages 222–235, 1993.

[GKVL] Matthew Green, Gabriel Kaptchuk, and Gijs Van Laer. Abuse resistant law enforcement access systems. In *EuroCrypt 2021*, pages 553–583.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, (2):270–299, 1984.

[GMW87] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *STOC '87*, page 218–229, 1987.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC '08*, pages 197–206, 2008.

[HPRV19] Thibaut Horel, Sunoo Park, Silas Richelson, and Vinod Vaikuntanathan. How to subvert backdoored encryption: Security against adversaries that decrypt all ciphertexts. In *10th Innovations in Theoretical Computer Science Conference, ITCS*, volume 124 of *LIPIcs*, pages 42:1–42:20, 2019.

[Ker83] Auguste Kerckhoffs. La Cryptographie Militaire. *Journal des sciences militaires*, August 1883.

[Mic] Silvio Micali. Fair public-key cryptosystems. In *CRYPTO' 92*, pages 113–138.

[MP16] M. Marlinspike and T. Perrin. The double ratchet algorithm, 11, 2016. `https://whispersystems.org/docs/specifications/doubleratchet/doubleratchet.pdf`.

[MY15] Daisuke Moriyama and Moti Yung. The bright side arguments for the coming smartphones crypto war: The added value of device encryption. In *2015 IEEE Conference on Communications and Network Security, CNS*, pages 65–73, 2015.

[NY90] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *STOC '90*, pages 427–437, 1990.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC '05*, pages 84–93, 2005.

[Riv] Ronald L. Rivest. Chaffing and Winnowing: Confidentiality without Encryption. MIT Lab for Computer Science, March 18, 1998 (rev. July 1, 1998), http://people.csail.mit.edu/rivest/chaffing-980701.txt.

[Rog15] Phillip Rogaway. The Moral Character of Cryptographic Work. ePrint 2015/1162, 2015. `https://ia.cr/2015/1162`.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Commun. ACM*, 21, 1978.

[RTYZ16] Alexander Russell, Qiang Tang, Moti Yung, and Hong-Sheng Zhou. Cliptography: Clipping the power of kleptographic attacks. In *ASIACRYPT 2016*, pages 34–64, 2016.

[RTYZ17] Alexander Russell, Qiang Tang, Moti Yung, and Hong-Sheng Zhou. Generic semantic security against a kleptographic adversary. In *CCS '17*, page 907–922, 2017.

[Sah99] Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *FOCS '99*, page 543, 1999.

[Sha49] Claude E. Shannon. Communication theory of secrecy systems. *Bell Systems Technical Journal*, 28(4):656–715, 1949.

[vAH04] Luis von Ahn and Nicholas J. Hopper. Public-key steganography. In *EUROCRYPT 2004*, pages 323–341, 2004.

[Wha20] WhatsApp. WhatsApp Encryption Overview, Oct. 2020.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets. In *FOCS '86*, pages 162–167, 1986.

[YYa] Adam Young and Moti Yung. The dark side of "black-box" cryptography, or: Should we trust capstone? In *CRYPTO '96*, pages 89–103.

[YYb] Adam Young and Moti Yung. The prevalence of kleptographic attacks on discrete-log based cryptosystems. In *CRYPTO 97*, pages 264–276.

[YY97] Adam Young and Moti Yung. Kleptography: Using cryptography against cryptography. In *EuroCrypt '97*, pages 62–74, 1997.

[YY98] Adam L. Young and Moti Yung. Auto-Recoverable Auto-Certifiable Cryptosystems. In *EuroCrypt '98*, pages 17–31, 1998.

A Preliminaries

In this section we review some of the concepts that we use in our constructions (with a deeper technical emphasize than in the introduction).

A.1 Encryption schemes

We start by defining the notion of a public-key encryption scheme.

Definition 5. *A* public-key encryption scheme $\mathcal{E}$ *is a triplet* $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *of probabilistic algorithms with the following syntax*

1. *the* key-generator algorithm KG *takes as input the security parameter* 1^λ *and returns the* public key $\mathtt{PK}$ *and the* secret key $\mathtt{SK}$*.*

2. *the* encryption algorithm Enc *takes as input the public key* $\mathtt{PK}$ *and a message* $\mathtt{m}$ *and returns a* ciphertext$\mathtt{ct} \leftarrow \mathsf{Enc}(\mathtt{PK}, \mathtt{m})$*.*

3. *the* decryption algorithm Dec *takes as input the secret key* $\mathtt{SK}$ *and a ciphertext* $\mathtt{ct}$ *and returns a message* $\mathtt{m}$*.*

that enjoys the following correctness property:

- *for every* $\mathtt{m}$ *and every* λ*,*

$$\text{Prob}\left[(\mathtt{PK}, \mathtt{SK}) \leftarrow \mathsf{KG}(1^\lambda); \mathtt{ct} = \mathsf{Enc}(\mathtt{PK}, \mathtt{m}) : \mathsf{Dec}(\mathtt{SK}, \mathtt{ct}) \neq \mathtt{m}\right] \leq \mathsf{negl}(\lambda).$$

When we wish to stress the random coin tosses $\mathcal{R}$ used by the encryption algorithm we will write $\mathtt{ct} \leftarrow \mathsf{Enc}(\mathtt{PK}, \mathtt{m}; \mathcal{R})$.

Let us now review the security notion for public-key encryption schemes by presenting the IndCPA games. More precisely, for public-key encryption scheme $\mathcal{E}$, $\eta = 0, 1$ and PPT adversary $\mathcal{A}$, we consider the following game $\mathsf{IndCPA}^{\eta}_{\mathcal{E},\mathcal{A}}$.

$\mathsf{IndCPA}^{\eta}_{\mathcal{E},\mathcal{A}}(\lambda)$

1. $(\mathtt{PK}, \mathtt{SK}) \leftarrow \mathcal{E}.\mathsf{KG}(1^\lambda)$;
2. Return
$$\mathcal{A}^{\mathsf{Oracle}^{\eta}(\mathtt{PK},\cdot,\cdot)}(\mathtt{PK}),$$
where $\mathsf{Oracle}^{\eta}(\mathtt{PK}, m_0, m_1) = \mathsf{Enc}(\mathtt{PK}, m_\eta)$.

We denote by $p^{\mathsf{IndCPA}^{\eta}}_{\mathcal{E},\mathcal{A}}(\lambda)$ the probability that $\mathsf{IndCPA}^{\eta}_{\mathcal{E},\mathcal{A}}(\lambda)$ outputs 1.

Definition 6. *An encryption scheme* $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *is IND-CPA secure if for all PPT adversary* $\mathcal{A}$

$$\left|p^{\mathsf{IndCPA}^{0}}_{\mathcal{E},\mathcal{A}}(\lambda) - p^{\mathsf{IndCPA}^{1}}_{\mathcal{E},\mathcal{A}}(\lambda)\right| \leq \mathsf{negl}(\lambda).$$

For symmetric encryption schemes instead we have the following definitions.

Definition 7. *A* symmetric encryption scheme $\mathcal{E}$ *is a triplet* $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *of probabilistic algorithms with the following syntax*

1. *the* key-generator algorithm KG *takes as input the security parameter* 1^λ *returns the* secret key $\mathtt{SK}$*.*

2. *the* encryption algorithm Enc *takes as input the secret key* $\mathtt{SK}$ *and a message* $\mathtt{m}$ *and returns a* ciphertext $\mathtt{ct} \leftarrow \mathsf{Enc}(\mathtt{SK}, \mathtt{m})$.

3. *the* decryption algorithm Dec *takes as input the secret key* $\mathtt{SK}$ *and a ciphertext* $\mathtt{ct}$ *and returns a message* $\mathtt{m}$.

that enjoys the following correctness property:

- *for every* $\mathtt{m}$ *and every* λ,

$$\text{Prob}\left[\mathtt{SK} \leftarrow \mathsf{KG}(1^\lambda); \mathtt{ct} = \mathsf{Enc}(\mathtt{SK}, \mathtt{m}) : \mathsf{Dec}(\mathtt{SK}, \mathtt{ct}) \neq \mathtt{m}\right] \leq \mathsf{negl}(\lambda).$$

When we wish to stress the random coin tosses $\mathcal{R}$ used by the encryption algorithm we will write $\mathtt{ct} \leftarrow \mathsf{Enc}(\mathtt{SK}, \mathtt{m}; \mathcal{R})$.

Let us now review the security notion for symmetric encryption schemes by presenting the IndCPA games. More precisely, for an encryption scheme $\mathcal{E}$, $\eta = 0, 1$, and PPT adversary $\mathcal{A}$, we consider the following security games that, with a slight abuse of notation, we call $\mathsf{IndCPA}^{\eta}_{\mathcal{E},\mathcal{A}}$.

$\mathsf{IndCPA}^{\eta}_{\mathcal{E},\mathcal{A}}(\lambda)$

1. $\mathtt{SK} \leftarrow \mathcal{E}.\mathsf{KG}(1^\lambda)$;
2. Return

$$\mathcal{A}^{\mathsf{Oracle}^{\eta}_{\mathsf{c}}(\mathtt{SK},\cdot,\cdot), \mathsf{Oracle}_{\mathsf{e}}(\mathtt{SK},\cdot),}(1^\lambda),$$

where $\mathsf{Oracle}^{\eta}_{\mathsf{c}}(\mathtt{SK}, m_0, m_1) = \mathsf{Enc}(\mathtt{SK}, m_\eta)$ and $\mathsf{Oracle}_{\mathsf{e}}(\mathtt{SK}, m) = \mathsf{Enc}(\mathtt{SK}, m)$.

We denote by $p^{\mathsf{IndCPA}^{\eta}}_{\mathcal{E},\mathcal{A}}(\lambda)$ the probability that $\mathsf{IndCPA}^{\eta}_{\mathcal{E},\mathcal{A}}(\lambda)$ outputs 1.

Definition 8. *A symmetric scheme* $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *is IND-CPA secure if for all PPT adversary* $\mathcal{A}$

$$\left|p^{\mathsf{IndCPA}^{0}}_{\mathcal{E},\mathcal{A}}(\lambda) - p^{\mathsf{IndCPA}^{1}}_{\mathcal{E},\mathcal{A}}(\lambda)\right| \leq \mathsf{negl}(\lambda).$$

A.2 Simulation Sound NIZK

In this section we review the notion of a *Simulation Sound NIZK* [BDSMP91, FLS99, Sah99].

Definition 9. *A* polynomial-time relation Rel *is a subset* $\mathsf{Rel} \subseteq \{0,1\}^\star \times \{0,1\}^\star$ *of pairs* (x, w) *of* instances *and* witnesses *for which there exists a* verification algorithm Ver *running in time polynomial in the length of its first input such that* $\mathsf{Ver}(x, w) = 1$ *iff* $(x, w) \in \mathsf{Rel}$.

The language L_{Rel} *associated with relation* Rel *is defined as* $L_{\mathsf{Rel}} = \{x | \exists w : (x, w) \in \mathsf{Rel}\}$.

Definition 10 (NIZK [BDSMP91, FLS99])**.** *A NIZK* $\Pi = (\mathsf{Prover}, \mathsf{Verifier})$ *for polynomial-time relation* Rel *consists of a pair of PPT algorithms* $(\mathsf{Prover}, \mathsf{Verifier})$ *such that, for some polynomial* l,

1. **Completeness.** *For all* $(x, w) \in \mathsf{Rel}$, *with* $|x| = \mathsf{poly}(\lambda)$, *and all* $\Sigma \in \{0,1\}^\lambda$, $\mathsf{Verifier}(x, \mathsf{Prover}(x, w, \Sigma), \Sigma) = 1$.

2. **Soundness.** *For all adversaries* $\mathcal{A}$, *the probability over* Σ *chosen at random from* $\{0,1\}^\lambda$ *that* $\mathcal{A}$ *outputs* (x, π) *with* $|x| = \mathsf{poly}(\lambda)$, $x \notin L_{\mathsf{Rel}}$, *and* $\mathsf{Verifier}(x, \pi, \Sigma) = 1$ *is negligible in* λ.

3. **Zero Knowledge.** *There exists a* simulator $S = (S_0, S_1)$ *consisting of two PPT algorithms such that for all stateful PPT adversaries* $\mathcal{A}$ *we have that*

$$\left|\text{Prob}\left[\mathsf{RealZK}_{\mathcal{A}}(\lambda) = 1\right] - \text{Prob}\left[\mathsf{IdealZK}^{S}_{\mathcal{A}}(\lambda) = 1\right]\right| \mathsf{negl}(\lambda),$$

where

```
$\mathsf{RealZK}_{\mathcal{A}}(\lambda)$;
    $\Sigma \leftarrow \{0,1\}^{\lambda}$;
    return $\mathcal{A}^{\mathsf{Prover}(\cdot,\cdot,\Sigma)}(\Sigma)$.
```

```
$\mathsf{IdealZK}^{S}_{\mathcal{A}}(\lambda)$;
    $(\Sigma, \mathsf{aux}) \leftarrow S_0(1^n)$;
    return $\mathcal{A}^{\mathsf{Oracle}_{\mathsf{aux}}(\cdot,\cdot,\Sigma)}(\Sigma)$;
```

where $\mathsf{Oracle}_{\mathsf{aux}}(x, w, \Sigma)$ *returns* $S_1(x, \mathsf{aux})$.

In the definition above, the verifier is guaranteed that no prover can make the verifier accept $x \notin L_{\mathsf{Rel}}$ if the reference string Σ is randomly chosen. In a simulation sound NIZK this guarantee holds even if Σ is constructed by the simulator and the adversary is allowed to see the proof for an x of his choice. More formally, we have the following definition.

Definition 11 (Simulation Sound NIZK [Sah99])**.** *A* NIZK $(\mathsf{Prover}, \mathsf{Verifier})$ *for polynomial-time relation* Rel *with simulator S is* Simulation Sound *if for all PPT adversaries* $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ *we have that* $\text{Prob}\left[\mathsf{ExpSS}_{\mathcal{A}}(n) = 1\right]$ *is negligible, where*

```
$\mathsf{ExpSS}_{\mathcal{A}}(n)$;
    $(\Sigma, \mathsf{aux}) \leftarrow S_0(1^n)$;
    $(x, \tau) \leftarrow \mathcal{A}_0(\Sigma)$;
    $\pi \leftarrow S_1(x, \Sigma, \mathsf{aux})$;
    $(x', \pi') \leftarrow \mathcal{A}_1(x, \pi, \Sigma, \tau)$;
    if $\pi \neq \pi'$ and $x' \notin L_{\mathsf{Rel}}$ and $\mathsf{Verifier}(x', \pi', \Sigma) = 1$ then
        return 1;
    else
        return 0;
```

B Functionality and Security of the Partial Anamorphic Modes

In this section, we show that the two *Partial* Anamorphic modes, that is the Anamorphic Mode with Normal Decryption and the Anamorphic Mode with Normal Encryption, are functional and secure. Specifically, the decryption algorithm is the functional inverse of the encryption algorithm and that security is preserved. We will used $\mathsf{aneGame}$ defined in Section 4.4.

B.1 Properties of the Anamorphic Mode with Normal Encryption

In this section, we use Lemma 1 to show that the encryption scheme aneAME associated with the Anamorphic mode with Normal Encryption is indeed an encryption scheme and that it is secure.

Lemma 5. *Let* AME *be a Secure Receiver-AM scheme. Then* $\mathsf{aneAME} = (\mathsf{aKG}_{1,2}, \mathsf{Enc}, \mathsf{Dec})$ *is an encryption scheme.*

Proof. For the sake of contradiction, suppose there exist message $\hat{m}$ and $\mathsf{poly}(\cdot)$ such that

$$\text{Prob}\left[(\mathtt{aPK}, \mathtt{aSK}) \leftarrow \mathsf{aKG}_{1,2}(1^{\lambda}); \mathtt{ct} \leftarrow \mathsf{Enc}(\mathtt{aPK}, \hat{m}) : \mathsf{Dec}(\mathtt{aSK}, \mathtt{ct}) \neq \hat{m}\right] \geq 1/\mathsf{poly}(\lambda). \tag{1}$$

This can be used to distinguish $\mathsf{NormalGame}$ from aneG by an adversary $\mathcal{A}$ that receives the pair of keys $(\mathtt{PK}^\star, \mathtt{SK}^\star)$, uses $\mathtt{PK}^\star$ to encrypt $\hat{m}$ in ciphertext $\mathtt{ct}$ and then outputs 1 if and only if decryption of $\mathtt{ct}$ with $\mathtt{SK}^\star$ is successful. In $\mathsf{NormalGame}$, $\mathcal{A}$, by the correctness of the normal encryption scheme, will output 1 will probability $1 - \mathsf{negl}(\lambda)$. On the other hand, in aneG, by Eq. 1, $\mathcal{A}$ outputs 1 with probability at most $1 - 1/\mathsf{poly}(\lambda)$. This contradicts Lemma 1. □

The next lemma establishes the security properties of aneAME.

Lemma 6. *Let* AME *be an IND-CPA secure* Secure Receiver-AM. *Then* aneAME *is IND-CPA secure.*

Proof. Suppose there exists a successful IND-CPA adversary $\mathcal{A}$ for aneAME. Then we construct the following adversary $\mathcal{B}$ that contradicts Lemma 1.

$\mathcal{B}$ receives $(\mathtt{PK}^\star, \mathtt{SK}^\star)$ and passes $\mathtt{PK}^\star$ to $\mathcal{A}$. When $\mathcal{A}$ issues the challenge query (m_0, m_1), $\mathcal{B}$ randomly selects $\eta \leftarrow \{0, 1\}$, encrypts m_η by using $\mathtt{fPK}^\star$ and Enc and returns the ciphertexts to $\mathcal{A}$. Finally, when $\mathcal{A}$ outputs its guess η', $\mathcal{B}$ outputs 1 iff $\eta = \eta'$. Let us now bound the probability that $\mathcal{B}$ outputs 1 when playing $\mathsf{NormalGame}$ and aneG.

If $\mathcal{B}$ is playing $\mathsf{NormalGame}$ then $(\mathtt{PK}^\star, \mathtt{SK}^\star)$ is output by KG and, by the security of AME, $\mathcal{A}$ has probability at most $1/2 + \mathsf{negl}(\lambda)$ of guessing η. On the other hand, if $\mathcal{B}$ is playing aneG then $(\mathtt{PK}^\star, \mathtt{SK}^\star)$ is output by $\mathsf{aKG}_{1,2}$ and thus, by assumption, $\mathcal{A}$ has probability at least $1/2 + 1/\mathsf{poly}(\lambda)$ of guessing η. This concludes the proof. □

We note that the above proof can be used to show that if AME is IND-CCA then so is aneAME. Roughly speaking, we can modify $\mathcal{B}$ so that, whenever $\mathcal{A}$ issues a decryption query, $\mathcal{B}$ replies by decrypting the ciphertext received by using $\mathtt{SK}^\star$ and Dec.

B.2 Properties of the Anamorphic Mode with Normal Decryption

In this section we consider the properties of the encryption scheme andAME associated with the Anamorphic Mode with Normal Decryption. We remind the reader that this corresponds to the case the adversarial authority $\mathcal{A}$, that has access to the anamorphic keys $(\mathtt{aPK}, \mathtt{aSK})$, tries to decrypt an anamorphic ciphertexts one using Dec and $\mathtt{aSK}$ (instead of aDec and $\mathtt{dkey}$).

We prove that for every $\hat{m}$, the scheme $\mathsf{andAME}_{\hat{m}}$ associated with Anamorphic Mode with Normal Decryption is indeed an encryption scheme according to Definition 5.

Lemma 7. *Let* AME *be a* Secure Receiver-AM *scheme. Then, for every* $\hat{m}$, $\mathsf{andAME}_{\hat{m}}$ *is an encryption scheme according to Definition 5.*

Proof. For the sake of contradiction, let $\hat{m}_0$ and $\hat{m}_1$ be two messages such that

$$\text{Prob}\left[(\mathtt{aPK}, \mathtt{aSK}, \mathtt{dkey}) \leftarrow \mathsf{aKG}(1^\lambda); \mathtt{ct} \leftarrow \mathsf{aEnc}(\mathtt{dkey}, \hat{m}_0, \hat{m}_1) : \mathsf{Dec}(\mathtt{aSK}, \mathtt{ct}) \neq \hat{m}_0\right] \geq 1/\mathsf{poly}(\lambda). \quad (2)$$

Consider the following adversary $\mathcal{A}$ that distinguishes aneG from $\mathsf{FullyAGame}$. $\mathcal{A}$ receives $(\mathtt{aPK}^\star, \mathtt{aSK}^\star)$ and sends the pair of messages $(\hat{m}_0, \hat{m}_1)$ to the challenger. Then $\mathcal{A}$ tries to decrypt the ciphertext $\mathtt{ct}$ received by running Dec and by using key $\mathtt{fSK}$ and it outputs 1 iff the decryption returns $\hat{m}_0$.

If $\mathcal{A}$ is playing game $\mathsf{FullyAGame}$ then the ciphertext is computed aEnc and thus, by Eq. 2, the probability that $\mathcal{A}$ returns 1 is at most $1 - 1/\mathsf{poly}(\lambda)$. On the other hand if $\mathcal{A}$ is playing aneG then $\mathtt{ct}$ is computed by running algorithm Enc with $\mathtt{aPK}$ and, by Corollary 5, the probability that $\hat{m}_0$ is returned by Dec, and thus $\mathcal{A}$ returns 1, is at least $1 - \mathsf{negl}(\lambda)$. This contradicts Lemma 2. □

As far as the IND-CPA security of andAME we observe that andAME and fAME use the same key-generation algorithm and the same encryption algorithm and therefore IND-CPA security of andAME follows from the IND-CPA security of fAME established in Theorem 1.

Lemma 8. *Let* AME *be an IND-CPA* Secure Receiver-AM *scheme. Then, for every* $\hat{m}$, $\mathsf{andAME}_{\hat{m}}$ *is IND-CPA.*