How to Securely Prolong the Computational Bindingness of Pedersen Commitments

Denise Demirel[1] and Jean Lancrenon[2]

[1] Technische Universität Darmstadt ddemirel@cdc.informatik.tu-darmstadt.de
[2] University of Luxembourg jean.lancrenon@uni.lu

Abstract. Pedersen commitments are important cryptographic primitives. They allow a prover to commit to a certain value without revealing any information about it and without the prover being able to change its mind later on. Since the first property holds unconditionally this is an essential primitive for many schemes providing long-term confidentiality. However, the second property only holds computationally. Hence, in the long run bindingness is lost, making the primitive improper for long-lived systems. Thus in this paper, we describe a protocol that, in a sense, *prolongs the bindingness* of a given Pedersen commitment. More precisely, we demonstrate how to prove in perfect zero-knowledge that a new Pedersen commitment - generated with a *larger* security parameter - and a corresponding old commitment both commit to the same value. We stress that this is a non-trivial procedure. Up until now the only known perfect zero-knowledge proof techniques for proving message equivalence of two commitments work when both commitments use isomorphic message spaces. However, as we will show in this work, to prolong the security of Pedersen commitments we cannot tolerate this restriction. Our prolonging technique works for non-isomorphic message spaces, is efficient, can be repeated an arbitrary number of times, maintains unconditional confidentiality, and allows to preserve the format of the Pedersen commitments. This makes the construction presented here an important contribution to long-lived systems. Finally, we illustrate this by discussing how commitments with prolongable bindingness can be used to allow for archiving solutions that provide not only integrity but also confidentiality in the long-term.

Keywords: unconditionally hiding commitments, long-term security, perfect zero-knowledge proofs, Pedersen commitments

1 Introduction

Problem statement In cryptography, a commitment scheme allows a prover to commit to a certain value while keeping it hidden from the public (confidentiality) and without being able to change its mind later on (bindingness). A well-known and widely used commitment scheme is the one proposed by Pedersen [39].

One practical application of commitment schemes (among others) is the storage of digital documents. In future IT environments digital archives are needed that efficiently and securely preserve electronic documents for a long period of time. Examples include scientific data, medical records, and land registries. Hospitals in the United Kingdom [15] and land registries in Estonia [10], for instance, have to generate and store an ever-growing amount of data. Data archived must attain several protection goals to remain useful. It should, for instance, be possible to prove that a document existed at a certain point in time, *proof of existence*, and that it was not changed since, *integrity*. In addition, in many scenarios, e.g. medical records, the data stored contains private information, hence the storage solution must provide some form of *confidentiality*.

Assume, for instance, the following setting. The owner of a sensitive document wants to store it such that long-term confidentiality is preserved. The only known approach to do this is to use proactive secret sharing, see e.g. [29, 26]. Using this technique the document is split into shares, that are distributed to different shareholders, and are renewed from time to time. To ensure that the document cannot be modified by the shareholders during the share renewal process the owner also sends a Pedersen commitment to the document to each shareholder. Assume now the owner wants to generate a proof of existence and integrity of that document, and wishes to do so without revealing any information on its contents to a notary or the public. In this case, the owner might simply get the Pedersen commitment timestamped, notarially attested, and published. Since the commitment scheme is binding the document's owner can prove to a verifier that the attestation belongs to the document. Furthermore, since the commitment is hiding only the verifier gets to see the document.

However, if one is interested in the *long-term* archiving of *sensitive* documents, it is important that the integrity and confidentiality properties of the above scheme hold *unconditionally*. Computationally secure cryptography (such as public-key encryption) is not suitable in this case, as it only guarantees security for a certain period of time, given a fixed set of security parameters. Fortunately, in the example above Pedersen commitments already solve part of this issue, since they happen to be unconditionally hiding: confidentiality is assured no matter how “weak” the security parameters become in the future. Unfortunately, they are only computationally binding. It is well-known that commitment schemes cannot enjoy both properties simultaneously. Note that generating an attestation to a commitment that is only computationally binding leads to a proof of existence and integrity that is only valid for a certain period of time. As soon as the commitment can be opened to another value the existence and integrity of the document can no longer be proven by the owner. In addition, the computational bindingness of Pedersen commitments also limits the runtime of the proactive secret sharing protocol. Since this data is crucial for verifiable share renewal and share reconstruction, security can only be provided as long as the computational bindingness of the commitment scheme holds. (A more detailed description of the problem statement with respect to proactive secret sharing and long-term archiving will be provided in Section 5). One might argue that such primitives can be instantiated using large security parameters. However, the period of time this instantiation would be secure can only be estimated and it can not be excluded that advances in hardware or improved attacks allow to break the security earlier than expected. Thus, such a limited and approximate lifetime is not an option when dealing with long-lived systems. It follows that if the initially chosen parameters' security is close to fading out, a new commitment scheme with renewed parameters needs to be generated. This must be done without 1) giving the owner of the document the chance to change the data committed to, or 2) revealing it to unauthorized parties. Note that to ensure unconditional confidentiality the commitment scheme used must provide this property right from the beginning. If this is not the case an attacker may capture commitments and store them until the underlying computational problem can be solved for the parameters chosen. Thus the only approach is to renew the binding property, which is lost over time, all the while preserving the hiding property, which always holds.

In this paper, we demonstrate a method to achieve this for Pedersen commitments.

More precisely, let p_1 and q_1 be large primes such that $q_1 | p_1 - 1$, and g_1 and h_1 be random generators of the unique q_1^{th}-order subgroup G_1 of $\mathbb{Z}_{p_1}^*$. Using the Pedersen commitment scheme, a commitment to a message $m \in \mathbb{Z}_{q_1}$ is generated by computing $c_1 = \mathsf{Com}_1(m, r_1) = g_1^m h_1^{r_1} \in G_1 \subset \mathbb{Z}_{p_1}^*$, where $r_1 \in \mathbb{Z}_{q_1}$ is a random value. To prove that c_1 is a commitment to m the prover only has to open the commitment by revealing m and r_1. This commitment is indeed only computationally binding because the prover's inability to find two values m' and r', with $m' \neq m$, and such that $\mathsf{Com}_1(m', r') = c_1$, rests on its inability to solve the discrete logarithm problem in G_1. The challenge is to devise a method by which, given a new set of parameters (p_2, q_2, g_2, h_2) (with the same cryptographic roles as p_1, q_1, g_1, and h_1), the prover can "transfer" the commitment $c_1 \in \mathbb{Z}_{p_1}^*$ to $m \in \mathbb{Z}_{q_1}$ to a commitment $c_2 \in \mathbb{Z}_{p_2}^*$ to $m \in \mathbb{Z}_{q_1} \subset \mathbb{Z}_{q_2}$, in a way that is both secure for the prover and potential verifiers. Security for the prover means that the unconditional confidentiality is maintained even taking into account leakage of the data generated by the transfer procedure. Otherwise, using unconditionally hiding commitments to begin with is meaningless. Security for the verifiers means that the owner of the document cannot use the transfer to change the document. Later, once the old security parameters become stale, the new parameters will prevent this from occurring. Indeed, the fact that the old commitment may then be opened by the user to an arbitrary value is no longer important, since only the new commitment is to be considered valid. This effectively "refreshes" the computational bindingness of the overall scheme, without damaging its unconditional confidentiality.

Some naïve approaches If the prover simply generates a new commitment c_2 to m by computing $c_2 = \mathsf{Com}_2(m, r_2) = g_2^m h_2^{r_2} \in \mathbb{Z}_{p_2}^*$, a potential verifier has no way of knowing whether the underlying message m is the same as that committed to in c_1. On the other end of the spectrum, the prover cannot simply open the old and new commitments to the public at the time when the new commitment is generated to provide evidence; this clearly violates confidentiality. Note that when generating c_2 the security of c_1 is about to fade out. Thus, the prover can use c_2 to commit to another message m' and at a later point in time, when it is asked to prove integrity, simply opens both commitments c_2 and c_1 to message m'. This is possible since c_1 is unconditionally hiding and can therefore be a commitment to any message. It follows that solely generating new commitments and storing all commitments for evidence does not guarantee that the message initially committed to cannot be changed. Thus, these trivially insecure approaches can be discarded and a proof is needed showing that c_2 has been generated correctly while c_1 was still secure.

Another attempt exploits existing methods by which the prover can demonstrate to the public in perfect zero-knowledge that it did not change message m from one commitment to the other. These are known as proofs of message equivalence (see e.g. [38]). However, these methods have the restriction that the message space of both commitments must be isomorphic, and to the best of our knowledge, they are the only ones that have been discovered so far. Thus, one could try generating a new commitment with the same message space but just a larger randomization space. Unfortunately, this approach fails as well because the message space stays small, allowing computational bindingness to be violated over time. Furthermore, the commitment is no longer a standard Pedersen commitment because the message and randomization spaces defined by the generators are no longer of equal size.

This shows that the problem is not a trivial one.

Our contribution In this paper we show how the message and randomization space of a commitment can be enlarged, such that the correct generation of the new commitment can be proven in perfect zero-knowledge. To do this, we essentially exhibit what can be viewed as a perfect zero-knowledge proof of message equivalence for Pedersen commitments *with non-isomorphic message spaces*. On a high-level, we propose the following approach. Given an old commitment $c_1 = \mathsf{Com}_1(m, r_1) = g_1^m h_1^{r_1} \in \mathbb{Z}_{p_1}^*$ and a new commitment $c_2 = \mathsf{Com}_2(m, r_2) = g_2^m h_2^{r_2} \in \mathbb{Z}_{p_2}^*$ we generate a third group $\mathbb{Z}_{p'}^*$ that contains one subgroup G'_1 that is isomorphic to G_1 and one subgroup G'_2 that is isomorphic to G_2. Using this third group as a connection between the old scheme and the new scheme, we can show that both commitments commit to the same message using a combination of perfect zero-knowledge range proofs and message-space-isomorphic equivalence proofs. To show that this is an important contribution to long-lived systems we discuss how our construction can be used to improve long-term archiving schemes. More precisely, we discuss how commitments with prolongable bindingness can be used to generate a proof of existence and integrity for secretly shared data and thus allow to build long-term archiving systems providing long-term confidentiality.

Remarks and organization Note that our methods rely - as the Pedersen commitment - on the hardness of the discrete logarithm problem. We are aware that in some decades it might be possible to build quantum computers that are capable of computing certain discrete logarithms. Prolonging the security of a given Pedersen commitment does not protect against this threat. Nevertheless, since Pedersen commitments are still widely used we will first provide a solution for this primitive and plan to consider developing corresponding solutions for post-quantum secure commitment schemes for future work.

The main application scenario of our contribution is that of long-term archiving, but it may very be useful in other settings as well. Furthermore, this perfect zero-knowledge proof of message equivalence for commitments with non-isomorphic message statements appears to be the first of its kind, and may be of independent interest.

The remainder of this paper is structured as follows: After an overview of the preliminaries in Section 2 we provide a detailed description of the protocol allowing to prolong the security of Pedersen commitments in Section 3. In Section 4 we point out some nice features of our protocol, and prove that it provides certain properties such as correctness and unconditional confidentiality. Finally, in Section 5 we discuss how commitments with prolongable bindingness can be used to provide integrity and confidentiality in long-term archiving solutions, and we conclude in Section 6.

2 Preliminaries

In the remainder of the paper, for any integer $n \geq 1$ we shall use $\mathbb{Z}_n$ or $[0, n-1]$ to designate the set $\{0, ..., n-1\} \subset \mathbb{N}$.

2.1 Pedersen commitments

The Pedersen commitment scheme [39] works as follows: Let κ be the security parameter. Then, the algorithm $\mathsf{GenCom}(1^\kappa)$ sets up the scheme by generating a set of common parameters. More precisely, it chooses two primes p and q where p and q are sufficiently large and q divides $p-1$. Then it randomly selects two generators g and h of the q^{th}-order subgroup G of $\mathbb{Z}_p^*$.

To commit to a secret $m \in \mathbb{Z}_q$ the committer chooses a random value $r \in \mathbb{Z}_q$ - the so called decommitment value - and computes commitment $c \in \mathbb{Z}_p^*$ by $c = \mathsf{Com}(m, r) = g^m h^r \bmod p$. Note that under the assumption that $\log_g h$ is unknown to the committer, it needs to have knowledge of the opening values m and r to open commitment c.

The algorithm $\mathsf{Unv}(c, m, r)$ takes as input a commitment c, message $m \in \mathbb{Z}_q$, and decommitment value $r \in \mathbb{Z}_q$, and returns m, if c is a valid commitment to m, with decommitment value r, i.e. $c = \mathsf{Com}(m, r)$, and $\perp$ if not. The commitment scheme is:

Computationally Binding Given a commitment $c = \mathsf{Com}(m, r)$, for any probabilistic polynomial-time adversary P the probability to find a second pair of opening values (m', r') with $m \neq m'$ such that $\mathsf{Com}(m, r) = \mathsf{Com}(m', r')$ is negligible in the security parameter κ.

Unconditionally Hiding For any pair $m, m' \in \mathbb{Z}_q$ the distribution of the randomized values $\mathsf{Com}(m, r)$ and $\mathsf{Com}(m', r')$ is identical when $r, r' \in \mathbb{Z}_q$ are chosen uniformly at random.

Additively Homomorphic For all $m, m' \in \mathbb{Z}_q$ and $r, r' \in \mathbb{Z}_q$ a multiplication of two commitments leads to an addition of their opening values, i.a. $\mathsf{Com}(m, r) \cdot \mathsf{Com}(m'r') = \mathsf{Com}(m + m', r + r')$.

The first property is the one we are seeking to prolong. The second is vital in our setting; indeed, it would not make sense to try prolonging computational hiding commitments since ancient copies of these commitments may have been compromised anyway. As for the third, we shall see that it comes in handy for our construction, and is anyway heavily exploited in the zero-knowledge proofs we employ.

2.2 Perfect zero-knowledge proofs

In this section we provide an overview of the zero-knowledge proofs[3] needed within our protocol. These are all honest-verifier and public-coin zero-knowledge proofs of knowledge, terms we informally define below. To ensure that the verifier is honest these protocols should be performed publicly, e.g. using a bulletin board, where the random values and coins are taken from a random beacon. Furthermore, we will describe the proofs as interactive protocols between a prover and a verifier. Note that they can easily be transferred (in the random oracle model) to non-interactive proofs using the Fiat-Shamir heuristic [21]. There might be proofs that are better suited to our construction than those described here. Nevertheless, in this paper we will concentrate on the security of Pedersen commitments and consider to find more efficient proofs for future work.

Honest-verifier zero-knowledge Proofs of knowledge We first informally recall the main properties of honest-verifier, (public coin,) zero-knowledge proofs of knowledge (abbreviated HVZKP). An HVZKP is an interactive proof system between a polynomial-time prover P and a verifier V. The prover wishes to convince the verifier V of the truth of the statement that it *knows a certain secret possessing a certain property.* In our case, the common input to both P and V will be mostly one or more commitments, and the secret input to P will be openings of these commitments. At the end of the interaction between P and V, V outputs a bit, 1 if the proof is accepted and 0 if not.

HVZKPs have the following properties.

[3] Note that since Pedersen commitments are only computationally binding the proofs are only computationally sound. Thus, to be precise, we use zero-knowledge arguments instead of proofs. However, since in the literature the authors mostly refer to proofs we will continue using this term.

1. **Completeness:** If both P and V are behaving honestly and P indeed knows the secret, V outputs 1. This is a basic correctness requirement.
2. **Soundness:** If some prover P convinces V with non-negligible probability, there exists a polynomial-time extractor χ that runs P and V as subroutines and outputs P's knowledge. This requirement ensures that if the proof is convincing to V, P really "knows" the underlying secret.
3. **Perfect Honest-Verifier Zero-Knowledge:** Let tr be the random variable consisting of the transcript of exchanged messages produced by P and V. We require that there exists a polynomial-time machine that, on the common input to P and V, is able to produce a transcript identically distributed to tr. This guarantees that the interaction transcript will information-theoretically leak no information on the secret to the *honest* verifier. Note that this is true even if the verifier has unbounded computational power.

That the interaction is public coin simply means that the verifier's randomness consists of uniformly distributed public challenges to the prover. Precise formal definitions can be found in, e.g. [22, 3].

Proof of knowledge Here, a prover P simply convinces a verifier V that it has knowledge of the opening values m and r of a public commitment $c = \mathsf{Com}(m, r)$. This can be done similarly to Schnorr's proof of knowledge [41] by performing the following steps:

1. P picks a random message $m' \in \mathbb{Z}_q$, and decommitment value $r' \in \mathbb{Z}_q$ and sends commitment $c' = \mathsf{Com}(m', r')$ to V
2. V sends a random challenge $e \in \mathbb{Z}_q$ to P
3. P sends $u = m' + em \bmod q$ and $v = r' + er \bmod q$ to V
4. V accepts if $\mathsf{Com}(u, v) = c'c^e$.

The security properties of this protocol are similar to those of Schnorr's protocol. One might argue that if the prover does not know the correct opening values, it might simply guess the correct answer. However, if the verifier is truly choosing its challenge values at random from $\mathbb{Z}_q$, then the probability of extraction is only $\frac{1}{q}$. If q is too small the success probability can be further reduced by repeating the protocol several times. In addition, this protocol hides m information-theoretically: even given c', e, u, and v, an attacker with infinite computation power is unable to learn anything about m. For any m^* there exists a message m'', such that $u = m' + em = m'' + em^*$, i.e. $m'' = m + e(m - m^*) \bmod q$.

Range proof In a range proof, a prover P convinces a verifier V that it has committed to a message m within a certain range, say $[0, H]$. There are mainly three families of zero-knowledge range proofs: using some mathematical properties of positive integers [6, 36, 24, 49], using the decomposition of the secret in a (multi)-base [4, 14, 42, 37], and using public signatures on the elements within the range [8, 11, 9].

The first family of range proofs is not suitable for our protocol since for these approaches the underlying group has to be of unknown order. The second family uses the idea to decompose the secret m in its binary representation $m_0, \ldots, m_{l-1}$ such that $\sum_{i=0}^{l-1} 2^i m_i = m$ for $m_i \in \{0,1\}$ and $l = log_2(H+1)$. Assume P knows the opening values m and r of a public commitment $c = \mathsf{Com}(m,r)$ and wants to convince V that $m \in [0, 2^l - 1]$ for $l = log_2(H+1)$. It performs the following steps:

1. For all $i \in [0, l-1]$ prover P picks a random value $r_i \in \mathbb{Z}_q$, commits to m_i by computing $c_i = \mathsf{Com}(m_i, r_i) = g^{m_i} h^{r_i}$ and sends the commitments $c_0, \ldots, c_{l-1}$ to V
2. V computes $c^* = \prod_{i=0}^{l-1} c_i^{2^i} = \mathsf{Com}(m, \prod_{i=0}^{l-1} 2^i r_i)$
3. P convinces V in a zero-knowledge that
   a) for each $i \in [0, l-1]$ either $c_i = h_i^r$ or $c_i/g = h_i^r$ (showing that $m_i \in \{0,1\}$) and that
   b) c^* is a commitment to m

If there is no l such that $2^l - 1 = H$ then two range proofs have to be executed to show that both m and $H - m$ belong to the interval $[0, 2^{\lfloor log_2 H \rfloor + 1} - 1]$. Lipmaa, Asokan, and Niemi [37] improved this result by showing that for some well chosen coefficients G_i, there exists a set of $m_i \in \{0,1\}$ such that $m = \sum_{i=0}^{\lfloor log_2 H \rfloor} G_i m_i$. This allows to prove that $m \in [0, H]$ using only one range proof even for an arbitrary H. Note that it must be ensured that in Step 3.a) and Step 3.b) of the range proof shown above no information about m is revealed (not even for a computationally unbounded attacker). This can be achieved, for instance, by using a perfect witness indistinguishable proof (e.g. [13, 4]) to show that $m_i \in \{0,1\}$ and a perfect zero-knowledge proof of message equality (see Paragraph 2.2) to show that c^* commits to m. Depending on the application it might also be sufficient to require the proofs to be only statistically hiding instead of perfectly hiding. In this case the statistical zero-knowledge proofs presented in [37] can be used in both steps 3.a) and 3.b).

Another interesting approach using a similar idea is the (perfect) "zero-knowledge proof that a committed value is in $\mathbb{Z}_{2^k}$" proposed by Moran and Naor [38]. The basic idea behind this protocol is that P convinces V that the binary representation of m has only k bits using a proof of valid shuffle and message equality. They combine the standard cut-and-choose technique together with perfectly hiding commitments to gain a protocol ensuring perfect hidingness for the value committed to.

The third family of range proofs works also in the non-binary case. The basic idea is that the verifier first publishes signatures on all integers in the range $[0, H]$. Following this, P provides a proof of knowledge on the signatures of the elements committed to. This proves that it knows the representation of the secret with respect to the basis chosen. However, all publications embracing this approach concentrate on computational zero-knowledge range proofs, making them unsuitable for our purpose. Furthermore, if a signed value must be provided on all integers in the range $[0, H]$, then H cannot be too large; this is incompatible with our setting as well.

Proof of message equivalence In a proof of message equivalence, a prover P convinces a verifier V that two commitments $c_1 = \mathsf{Com}_1(m, r_1)$ and $c_2 = \mathsf{Com}_2(m, r_2)$ generated using the schemes Com_1 and Com_2 can be decommitted to the same message m. As long as the message spaces of both commitment schemes are isomorphic this can be done using the standard cut-and-choose technique as proposed by [12, 38]. We denote $\mathbb{Z}_{q_1}$ and $\mathbb{Z}_{q_2}$ the randomness spaces of Com_1 and Com_2 respectively, and, for simplicity, we assume that the schemes share message space $\mathbb{Z}_q$. The following steps are performed:

```
1. $P$ picks a random message $m' \in \mathbb{Z}_q$, and two decommitment values $r'_1 \in \mathbb{Z}_{q_1}$
   and $r'_2 \in \mathbb{Z}_{q_2}$ and sends the two commitments $c'_1 = \mathsf{Com}_1(m + m', r_1 + r'_1)$
   and $c'_2 = \mathsf{Com}_2(m + m', r_2 + r'_2)$ to $V$
2. $V$ sends a random bit $b \in \{0, 1\}$ to $P$
3. if b=0
   $P$ sends $m'$,$r'_1$, and $r'_2$ to $V$
   $V$ accepts if $c'_1 = c_1\mathsf{Com}_1(m', r'_1)$ and $c'_2 = c_2\mathsf{Com}_2(m', r'_2)$
   else
   $P$ sends $m + m'$,$r_1 + r'_1$, and $r_2 + r'_2$ to $V$
   $V$ accepts if $c'_1 = \mathsf{Com}_1(m + m', r_1 + r'_1)$ and $c'_2 = \mathsf{Com}_2(m + m', r_2 + r'_2)$
```

If the commitments c_1 and c_2 commit to distinct messages, then this will be detected with probability $\frac{1}{2}$. Thus, this protocol must be repeated several times until the probability of not being detected is below a certain threshold value.

3 Prolonging the security of Pedersen commitments

We now present our protocol for prolonging the computational binding property of Pedersen commitments. Essentially, the protocol is an honest-verifier, public-coin, zero-knowledge proof of message equivalence between two Pedersen commitments with non-isomorphic message spaces. Thus, we describe it in terms of a prover P and a verifier V.

The prover P is the owner of the document, and the entity that creates commitment(s) to it. In our context, the verifier's role is to aid P in passing its message(s) on from one set of parameters to the next. We assume that parameter generation is not an issue, i.e. P and V receive both the old and new sets of parameters, and trust their validity. (A parameter-generating authority could be used for this, or perhaps P and V can run an interactive protocol to jointly agree on this data.) On one hand, V's role is to make sure that a malicious P does not change its document during the procedure. On the other hand, an honest P wants the guarantee that during the procedure, no information on the document is leaked to an honest V. We will assume for now that V is not fully malicious, since the zero-knowledge proofs we employ are honest-verifier. However, in some scenarios it might be of interest to publish all data needed for verification on a secure and online-accessible bulletin board, so as to also allow any third party to check correctness. In this case, under the assumption that the bulletin board is secure and that all challenges come from a trusted random beacon, a cheating P will be detected with overwhelming probability even in the presence of a fully malicious V. Thus V's role in this scenario is mainly to ensure robustness by checking that the proof published on the bulletin board is indeed valid. Furthermore, this also protects an honest P against a cheating V. Since the correctness of the proofs published by P can by publicly verified by any third party, a cheating V cannot reject a valid proof without being detected.

Assume P generated and published commitment $c_1 = \mathsf{Com}_1(m, r_1) = {g_1}^m h_1^{r_1} \bmod p_1$ and still has knowledge of the opening values m and r_1. Furthermore, we assume that the old parameters are still secure. Now, P wants to use V to prolong the security of its commitment to m such that it is able to prove in perfect zero-knowledge that m did not change. We denote κ_1 the old security parameter.

1 Selecting new parameters Let κ_2 be the new security parameter.

a) Two new primes p_2 and q_2 are chosen, where p_2 and q_2 are sufficiently large and q_2 divides $p_2 - 1$.
b) Two generators g_2 and h_2 of subgroup G_2 of $\mathbb{Z}_{p_2}^*$ are randomly selected.
c) A prime p' such that $q_1 q_2$ divides $p' - 1$ is determined.
d) The new cyclic group $\mathbb{Z}_{p'}^*$ has three unique subgroups G'_1, G'_2, and G'_{12} of orders q_1, q_2, and $q_1 q_2$ respectively. For each group a new generator is randomly selected, i.e. $< g'_1 >= G'_1, < g'_2 >= G'_2$, and $< h' >= G'_{12}$.
e) The full set of new parameters is given to P and V.

2 Generating the new commitment P selects a new random value r_2 and generates a new commitment $c_2 = \mathsf{Com}_2(m, r_2) = g_2^m h_2^{r_2} \bmod p_2$ to message m.

3 Proving message equality What follows is the technical core of the protocol, the zero-knowledge proof itself.

- **Proof parameters:** $Params = (p_1, q_1, g_1, h_1, p_2, q_2, g_2, h_2, p', g'_1, g'_2, h')$
- **Common input to P and V:** $(Params, c_1, c_2)$
- **Private input to P:** (m, r_1, r_2)
- **Statement to prove:** P knows (m, r_1, r_2) such that $c_1 = \mathsf{Com}_1(m, r_1)$ and $c_2 = \mathsf{Com}_2(m, r_2)$
- **Protocol:**

a) P generates two new commitments to m in group $\mathbb{Z}_{p'}^*$: one that uses the message space $\mathbb{Z}_{q_1}$ (like c_1) and is computed as $c'_1 = \mathsf{Com}'_1(m, r'_1) = {g'_1}^m {h'}^{r'_1} \bmod p'$, and one that uses the message space $\mathbb{Z}_{q_2}$ (like c_2) and is computed as $c'_2 = \mathsf{Com}'_2(m, r'_2) = {g'_2}^m {h'}^{r'_2} \bmod p'$. Here, r'_1 and r'_2 are selected randomly from $\mathbb{Z}_{q_1 q_2}$.

b) P proves to V that it can open the commitments c_1 and c'_1 to the same message.
c) Similarly, P proves to V that it can open the commitments c_2 and c'_2 to the same message.
d) V multiplies the commitments c'_1 and c'_2, i.e. computes $c'_3 = c'_1 c'_2 = {g'_1}^m {h'}^{r'_1} {g'_2}^m {h'}^{r'_2} = (g'_1 g'_2)^m {h'}^{(r'_1+r'_2)} \bmod p'$.
e) P proves to V that it can open c'_3 *to a message in* $\mathbb{Z}_{q_1}$.

The proofs that 1) two commitments can be opened to one message and 2) a commitment can be opened to a value in $\mathbb{Z}_q$ are done using honest-verifier, perfect zero-knowledge proof protocols (see for instance "proof of message equality" and "range proof" presented in Paragraph 2.2).

4 Properties and proofs

4.1 Convenience

We emphasize here two important practical properties that this scheme enjoys.

First, we find remarkable that the degree of freedom in choosing the new parameters is essentially optimal. Indeed, aside from the fact that q_1, q_2, and p' have to be determined in such a way that $q_1 q_2$ divides $p' - 1$, there are absolutely no constraints on the parameters for the process to be carried out. This leaves protocol designers the ability to select parameters with other interesting properties at their leisure. It also avoids potentially having additional algebraic structure with which to attack the protocol.

Secondly, it is worth noting that the commitment scheme itself is not changed. This implies directly that the construct presented here can be easily integrated into other protocols using this scheme. (Some of our previous attempts at solving this problem yielded schemes that would have altered the Pedersen protocol.)

4.2 An overview of how the transfer procedure works

We call the transfer procedure the steps taken in the "proving message equality" paragraph above. We briefly go through how these all come together. Prover P's inputs to the transfer procedure are the previously generated commitments c_1 and c_2.

In Step 3a), two commitments are created in the transfer group $\mathbb{Z}^*_{p'}$: $c'_1 \in G'_1$ and $c'_2 \in G'_2$. In Step 3b), P proves on one hand that it can open c_1 and c'_1 to some message $m_1 \in \mathbb{Z}_{q_1}$, and on the other hand that it can open c_2 and c'_2 to some message $m_2 \in \mathbb{Z}_{q_2}$. Next, P demonstrates in steps 3d) and 3e) that it can open *the product* $c'_1 c'_2$ to some message $n \in \mathbb{Z}_{q_1}$. Now, simply using the openings of c'_1 to m_1, c'_2 to m_2, and $c'_1 c'_2$ to $n \in \mathbb{Z}_{q_1}$, P can form a specific algebraic equation, and it turns out that P can have $m_1 \neq m_2$ in this case *if and only if it can compute certain discrete logarithms.* Thus, a *computationally limited* P can only pass this check if $m_1 = m_2$, *provided we prohibit it from trying to cheat by using values outside of* $\mathbb{Z}_{q_1}$. This is the role of the range proof on the product.

Finally, V still needs to be convinced that the m_1 and m_2 values considered above cannot be different from those used to compute the inputs c_1 and c_2. But this is ensured by the computational bindingness of Com_1 and Com_2.

It will be apparent that several of the discrete logarithm puzzles used in this construction depend solely on security parameter κ_1, but this is not a problem. Indeed, the transfer procedure's data only needs to be secured by κ_1 during the procedure itself, which should happen while κ_1 is still large enough. After the message has been passed on to the new set of parameters, only κ_2's size remains important in the long term.

4.3 Security

In this section, we provide a proof sketch that the protocol above achieves its goal of prolonging the security of the binding property of a Pedersen commitment, when the verifier honestly executes the protocol. It is clear that to achieve this, we only need to prove that the core of our construction is in fact an HVZKP that two Pedersen commitments with non-isomorphic message spaces can be opened to the same message. Indeed, the correctness of our protocol follows from the underlying proof being complete, the fact that binding is prolonged will follow from soundness, and information-theoretic confidentiality will follow from the honest-verifier zero-knowledge property.

Correctness Completeness of the protocol follows directly from the completeness of the standard message equivalence and range proofs. Therefore, the construction is correct.

Prolonged binding The aim of this section is to prove the following result:

Theorem 1 *The protocol described in Section 3 proves that with overwhelming probability, P can open $c_1 = \mathsf{Com}_1(m, r_1) = g_1^m h_1^{r_1} \bmod p_1$ and $c_2 = \mathsf{Com}_2(m, r_2) = g_2^m h_2^{r_2} \bmod p_2$ to one unique message in $\mathbb{Z}_{q_1}$.*

Proof of the theorem: We begin by establishing that our interactive proof described in paragraph 3 is sound. For this, we first need a technical lemma.

Lemma 1 *A probabilistic polynomial-time P cannot find triples $(m, m^*, n) \in \mathbb{Z}_{q_1} \times \mathbb{Z}_{q_2} \times \mathbb{Z}_{q_1}$ with $m \neq m^*$ and $(r'_1, r'_2, r^*) \in \mathbb{Z}^3_{q_1 q_2}$ such that*

$$g_1'^{\,m} h'^{r'_1} g_2'^{\,m^*} h'^{r'_2} = (g'_1 g'_2)^n h'^{r^*}$$

Proof of the lemma: Suppose that P was able to find such values. We set $r := r'_1 + r'_2 \bmod q_1 q_2$. Applying the Chinese Remainder Theorem (CRT), let t be the unique integer in $[0, q_1 q_2 - 1]$ such that $g_1'^{\,m} g_2'^{\,m^*} = (g'_1 g'_2)^t$. We thus have

$$g_1'^{\,t} g_2'^{\,t} h'^{r} = (g'_1 g'_2)^t h'^{r} = g_1'^{\,m} g_2'^{\,m^*} h'^{r} = (g'_1 g'_2)^n h'^{r^*} = g_1'^{\,n} g_2'^{\,n} h'^{r^*} \qquad (*)$$

We will need the following

Claim ***a)*** *If* $t \equiv n \bmod q_1$*, then* $n = m$ *and* $t \not\equiv n \bmod q_2$*.*
b) *If* $t \equiv n \bmod q_2$*, then* $n = m^*$ *and* $t \not\equiv n \bmod q_1$*.*

Proof of the claim: If $t \equiv n \bmod q_1$, then we have $n \equiv t \equiv m \bmod q_1$. Since both n and m are in $[0, q_1 - 1]$, this yields $n = m$. Similarly, if $t \equiv n \bmod q_2$, this implies that $n \equiv t \equiv m^* \bmod q_2$, and since both n and m^* are in $[0, q_2 - 1]$, this means $n = m^*$. Therefore, if $t \equiv n \bmod q_1$ and $t \equiv n \bmod q_2$, we have $m = n = m^*$, a contradiction to the requirement $m \neq m^*$. ■

We return to the proof of the lemma. Suppose that $t \equiv n \bmod q_1$; thus, $t \not\equiv n \bmod q_2$ by claim a). We can rewrite equation $(*)$ as follows:

$$g_1'^{\,n} g_2'^{\,t} h'^{r} = g_1'^{\,n} g_2'^{\,n} h'^{r^*}$$

or equivalently

$$g_2'^{\,t-n} = h'^{r^*-r}$$

Again applying the lemma, we have that $t - n$ is invertible modulo q_2; letting $u \in [0, q_2 - 1]$ be the unique inverse, we get

$$g_2' = h'^{u(r^*-r)}$$

This shows that if $t \equiv n \bmod q_1$ and P is able to open the commitment using the opening values (n, r^*), then it can find the discrete logarithm of g_2' to the base h'. Since this computation should be infeasible (because the new security parameter κ_2 is large enough), this shows that $t \not\equiv n \bmod q_1$.

Now, suppose that $t \equiv n \bmod q_2$, implying $t \not\equiv n \bmod q_1$ by claim b). In this case, $(*)$ becomes:

$$g_1'^{\,t-n} = h'^{r^*-r}$$

and re-applying the lemma gives us this time that $t - n$ is invertible modulo q_1, so similarly to above this shows that if $t \equiv n \bmod q_2$, P can compute the discrete logarithm of g_1' to the base h'. Since this should be infeasible (because the old security parameter κ_1 is *still* large enough), we conclude that $t \not\equiv n \bmod q_2$.

So far, we have proved that if P is able to find some $(m, m^*, n, r_1', r_2', r^*)$ with the desired properties, then it must have found n such that $n - t$ is invertible both modulo q_1 and modulo q_2. But now, using the CRT and letting v be the unique inverse modulo $q_1 q_2$ of $n - t$, we have that $g_1' g_2' = h'^{v(r^*-r)}$, so P is able to find the discrete logarithm of $g_1' g_2'$ to the base h', which is also infeasible. This concludes the proof of the lemma. ■

We now return to dealing with the soundness of our protocol.

Proposition 1 *The interactive proof described in paragraph 3 is sound.*

Sketch of proof: We need to show that a suitable polynomial-time extractor χ exists in case P succeeds in proving its claim to V. We build χ roughly as follows.

P proves to V that it knows an $m_1 \in \mathbb{Z}_{q_1}$ and $m_2 \in \mathbb{Z}_{q_2}$ such that c_1 and c'_1 can be opened to m_1 and c_2 and c'_2 can be opened to m_2. Thus, using the soundness of the message equivalence proof, χ extract triples (m_1, r_1, r'_1) and (m_2, r_2, r'_2) such that $c_1 = \mathsf{Com}_1(m_1, r_1)$, $c'_1 = \mathsf{Com}'_1(m_1, r'_1)$, $c_2 = \mathsf{Com}_2(m_2, r_2)$, and $c'_2 = \mathsf{Com}'_2(m_2, r'_2)$. P also proves that it can open the product $c'_1 c'_2$ to some message $n \in \mathbb{Z}_{q_1}$. Therefore, by the soundness of the range proof, this time χ can extract (n, r^*), with $r^* \in \mathbb{Z}_{q_1 q_2}$, such that $c'_1 c'_2 = (g'_1 g'_2)^n h'^{r^*}$. Now, this leads to the equality ${g'_1}^{m_1} {h'}^{r'_1} {g'_2}^{m_2} {h'}^{r'_2} = (g'_1 g'_2)^n h'^{r^*}$, and by Lemma 1, since χ is a polynomial-time machine, we obtain necessarily that $m_1 = m_2 = n$. The triple we were looking to extract can thus be set to (n, r_1, r_2). This completes the proof sketch. ■

We can now complete the proof of our theorem. The proposition above shows that if V is convinced by its interaction with P, then P must be able to open $c_1 = \mathsf{Com}_1(m, r_1) = g_1^m h_1^{r_1} \bmod p_1$ and $c_2 = \mathsf{Com}_2(m, r_2) = g_2^m h_2^{r_2} \bmod p_2$ to some message m in $\mathbb{Z}_{q_1}$. Since the commitments Com_1 and Com_2 are computationally binding, this message is unique. ■

A remark concerning the range proof on c'_3: The proof of Lemma 1 shows how a malicious P could cheat the system if instead of requiring that a range proof be conducted on c'_3 in Step 3e), we simply require that P demonstrate it can open c'_3 to some value in $\mathbb{Z}_{q_1 q_2}$.

Suppose that P - instead of using $m \in \mathbb{Z}_{q_1}$ as required - selects some different message $m^* \in \mathbb{Z}_{q_2}$ to commit to in c_2 and c'_2. Passing Step 3c) of the protocol is certainly not a problem. Now, P can very well compute the unique value $n \in \mathbb{Z}_{q_1 q_2}$ such that $n \equiv m \bmod q_1$ and $n \equiv m^* \bmod q_2$, and hence demonstrate in perfect zero-knowledge that it can open c_3 even though $m \neq m^*$. However, it is the fact that m and m^* are *distinct* - even in the case that $m^* \in \mathbb{Z}_{q_1}$ - that guarantees by the CRT isomorphism that n *must be greater or equal to* q_1. This precisely gives the clue to keep an eye out for: We can thwart malicious behavior by checking that $n \in [0, q_1 - 1]$.

A remark concerning the intermediate commitments Com'_1 and Com'_2: It may seem strange that the proofs given above do not make use of the fact that the intermediate commitments Com'_1 and Com'_2 employed in step 3a) are computationally binding. In reality, this argument is indeed present; it is just very well buried in the proof of Lemma 1. Evidence of this can be seen by observing that to prove the computational bindingness of Com'_1 and Com'_2, one reduces it to computing discrete logarithms that depend on parameters κ_1 and κ_2 respectively. This is done in very much the same way Lemma 1 relies on both κ_1 and κ_2 (in fact, on the smaller of the two).

Unconditional confidentiality Finally, we prove that our scheme maintains information-theoretic confidentiality. Of course, this relies on the perfect honest-verifier zero-knowledge property of our interactive proof.

Theorem 2 *The protocol described in Section 3 information-theoretically hides P's message from an honest V.*

Proof of the theorem: We know that both commitments Com_1 and Com_2 are unconditionally hiding, so the only issue is showing that the transfer is perfectly honest-verifier zero-knowledge. Now, since the proofs of message equivalence and range are HVZKP, their transcripts leak nothing to V, and it remains to study the commitments c_1' and c_2'. So the only claim to show is that these are unconditionally hiding. Therefore, we prove the following:

Lemma 2 *A commitment of the form $c_1' = \mathsf{Com}_1'(m, r_1') = {g_1'}^m {h'}^{r_1'} \bmod p'$ is unconditionally hiding.*

Proof of the Lemma: Recall that g_1' and h' are generators of the q_1^{th}-order and $q_1q_2^{th}$-order subgroups G_1' and G_{12}' of $\mathbb{Z}_{p'}^*$ respectively. It suffices to show that for every fixed arbitrary $c_1' \in G_{q_1q_2}'$ and $m \in \mathbb{Z}_{q_1}$ there exists a unique $r_1' \in \mathbb{Z}_{q_1q_2}$ such that $c_1' = {g_1'}^m {h'}^{r_1'} \bmod p'$, since this implies that any element in $G_{q_1q_2}'$ can be uniquely opened as a commitment to any message in $\mathbb{Z}_{q_1}$ with respect to the preset bases. But because $G_{q_1q_2}'$ is cyclic, this fact is trivial: the group element ${g_1'}^{-m} c_1'$ is in $G_{q_1q_2}'$, so there exists a unique $r_1' \in \mathbb{Z}_{q_1q_2}$ such that ${g_1'}^{-m} c_1' = {h'}^{r_1'}$. This is indeed equivalent to $c_1' = {g_1'}^m {h'}^{r_1'}$. ■

It can be shown analogously that commitment Com_2' is unconditionally hiding. This completes the proof of the theorem. ■

4.4 Performance

Besides proving that important security requirements are fulfilled it is also necessary to show that the protocol proposed works in practice. One essential assumption for our construction is that it is always possible to build a transfer group between the parameters used for the old and the new commitments. More precisely, having the old primes p_1 and q_1 and the new primes p_2 and q_2 the prover can always find a new prime p', such that $q_1q_2 | p' - 1$ (Section 3 Step 1c). To show this we use Dirichlet's theorem. In number theory, his theorem, also called the Dirichlet prime number theorem, states that for any two positive coprime integers a and b, there are infinitely many primes of the form $a + nb$, where $0 \leq n$. Since in our construction $a = 1$ and $b = q_1q_2$ it follows that $gcd(b, 1) = 1$, which is why there should be infinitely many primes p' allowing to build a transfer group between the old and the new parameters.

However, the distribution of prime numbers is irregular, because for large numbers they occur less frequently. To further analyse this, we implemented a prototype using *Mathematica 9* that searches for new parameters. We started with a randomly chosen q_1 of at least 190 bits and a p_1 of at least 2049 bits and simulated the process of selecting new parameters for q_2 of size at least 248 bits and p_2 of size at least 4093 bits, q_3 of size at least 326 bits and p_3 of size at least 8204 bits, and q_4 of size at least 427 bits and p_4 of size at least 16443 bits. Using the predictions based on the model presented by Lenstra in [35], these parameters would allow to prolong the security of the original commitment for around 100 years. Note that we are aware that within the next 100 years quantum computers might be available that allow to efficiently attack Pedersen commitments. For now we do not consider this threat, but plan to work on post-quantum solutions in the future.

We ran 1000 iterations using a 4 x Quad-Core AMD Opteron(tm) Processor 8356 with 64 GB RAM[4] and determined the average time needed to find new parameters and their average bit length. Table 1, Table 2, and Table 3 summarize our results for the first, second, and third commitment renewal. (The values p'', g_1'', g_2'', h'', p''', etc. play the same roles as p', g_1', g_2', etc. in a way that should be clear to the reader.)

	p_2	q_2	g_2	h_2	p'	g_1'	g_2'	h'
average time (in sec)	5.6632	0.0195	0.0596	0.0596	4.8955	0.0599	0.0596	0.0595
average length (in bit)	4093	248	4091	4091	4093	4091	4091	4091

Table 1. First renewal using κ_1

	p_3	q_3	g_3	h_3	p''	g_2''	g_3''	h''
average time (in sec)	68.5535	0.0425	0.3819	0.3820	54.0756	0.3821	0.3820	0.3819
average length (in bit)	8204	326	8202	8202	8204	8202	8202	8202

Table 2. Second renewal using κ_2

	p_4	q_4	g_4	h_4	p'''	g_3'''	g_4'''	h'''
average time (in sec)	670.1844	0.2203	2.2934	2.2936	621.9710	2.2889	2.2906	2.2905
average length (in bit)	16443	427	16441	16441	16443	16441	16441	16441

Table 3. Third renewal using κ_3

Our tests show that in total it takes only $\sim$ 11 seconds to find parameters for the first renewal. To determine the parameters for the second and third renewal the software had to run significantly longer, $\sim$ 2 minutes and $\sim$ 22

[4] Note that during the tests the RAM was never fully allocated.

minutes respectively. However, the parameters for the Pedersen commitments can be chosen large enough such that they ensure computationally bindingness for several decades. So this process does not have to be run often. Furthermore, until commitments generated with the parameters q_2 and p_2 have to be renewed (around 2083 [35]) more efficient hardware will be available.

4.5 An Attack on a Naïve Approach

We conclude the section by showing how an attempt at just increasing the randomness space's size fails to adequately refresh the binding property of the commitment.

The old parameters are still given by (p_1, q_1, g_1, h_1), working in the q_1^{th}-order subgroup G_1 of $\mathbb{Z}_{p_1}^*$. The refreshed parameters are generated first by selecting a larger prime p_2 such that q_1 divides $p_2 - 1$, and then picking two random generators g_2 and h_2 of the q_1^{th}-order subgroup G_2 of $\mathbb{Z}_{p_2}^*$ and of $\mathbb{Z}_{p_2}^*$ itself, respectively. The idea is that the message space stays the same - it will be $\mathbb{Z}_{q_1}$ - but the randomness space is increased from q_1 to $p_2 - 1$.

Committing to message $m \in \mathbb{Z}_{q_1}^*$ with the new parameters is done by selecting a random r_2 in $\mathbb{Z}_{p_2-1}$ and setting $c_2 = \mathsf{Com}_2(m, r_2) = g_2^m h_2^{r_2}$. Since the message space is not changed, proving that the old and new commitments can be opened by the message owner to the same message is easy to do in perfect zero-knowledge, see Paragraph 2.2.

Unfortunately, not increasing the message space leads to the following long-term attack. Suppose the old parameters "run out" of security, and in particular the malicious prover P can perform q_1 operations. We set $u := (p_2 - 1)/q_1$.

By assumption, P can successively compute all group elements h_2^{ut} for $t \in \mathbb{Z}_{q_1}$. It does so, thereby recording a list which equal to G_2, since for all $t \in \mathbb{Z}_{q_1}$ we have $(h_2^{ut})^{q_1} = 1$. Thus, g_2 must have collided with some $h_2^{ut'}$, so P now has in fact computed $\ell := \log_{h_2} g_2 = ut'$.

Suppose P committed previously to message $m \in \mathbb{Z}_{q_1}$ using randomness $r_2 \in \mathbb{Z}_{p_2-1}$. By definition, its commitment c_2 is $\mathsf{Com}_2(m, r_2) = g_2^m h_2^{r_2}$. Now, suppose P wants to open c_2 to a different message $m^* \in \mathbb{Z}_{q_1}$, i.e. find r^* such that $g_2^{m^*} h_2^{r^*} = g_2^m h_2^{r_2}$. P can simply compute

$$r^* := \ell(m - m^*) + r_2 \bmod p_2 - 1$$

and open c_2 to (m^*, r^*). A straightforward calculation shows this works:

$$\begin{aligned} g_2^{m^*} h_2^{r^*} &= h_2^{\ell m^* + r^*} \\ &= h_2^{\ell m^* + \ell(m - m^*) + r_2} \\ &= h_2^{\ell m} h_2^{r_2} \\ &= g_2^m h_2^{r_2} \\ &= c_2 \end{aligned}$$

5 Integrity and confidentiality preserving long-term archiving

In this section we discuss in more detail how Pedersen commitments can be used for long-term archiving. We will first provide an overview of the current solutions and point out their weaknesses. Afterwards we show how proactive secret sharing works, how it can be improved using the prolongable bindingness of Pedersen commitments, and how this approach can be combined with existing archiving solutions to allow for long-term confidentiality of archived documents.

5.1 State of the art of long-term archiving and storage solutions

To our knowledge the only approach to provide long-term confidentiality for stored data is to use proactive secret sharing. The basic idea is to split the documents into shares that are renewed from time to time. This approach has been introduced by Herzberg et al. in 1995 [29]. Their scheme uses an information theoretically secure and verifiable secret sharing scheme, such as Shamir's Secret Sharing [43], combined with Pedersen's [39] or Feldman's [20] commitments. However, a disadvantage of their proposal is that the structure of the shareholders must be maintained. Desmedt and Jajodia [16] improved this approach by showing how to redistribute a secret without recovering it, making the dynamic addition and renewal of shareholders possible. This especially allows to reboot or reinstall compromised servers. Then, Wong et al. [48] further developed this scheme by combining it with verifiable secret sharing. It follows that assuming the majority of the old shareholders to be trustworthy, the new shareholders can verify the validity of the shares received. This not only protects against mobile adversaries, but also against active (or Byzantine [33]) adversaries that are able to alter or replace messages. Nevertheless, correctness of the new shares is only guaranteed if all new shareholders are trustworthy and verify the data received. Therefore, Gupta and Gopinath [25] further improved this protocol by adding a complaint mechanism requiring only a majority of the new shareholders to be reliable. Since this approach use Feldman's commitments the public audit information is only computationally hidden. Therefore, Gupta and Gopinath also published an information theoretic secure protocol named $\mathrm{G}^2_{\mathrm{Its}}$ using Pedersen commitments [26]. This protocol provides verifiability, information theoretic security, and allows for the dynamic addition and removal of shareholders. However, a closer look at this protocol reveals that the verifiability relies on a witness in the form of a Pedersen commitment to the document stored. This data is crucial to ensure integrity during the share renewal and reconstruction process. However, the authors do not answer the question of how to proceed, if the bindingness of this commitment is about to fade out. It follows that this process can only be run as long as the commitment scheme is secure for the parameters chosen.

In addition, to provide long-term integrity of data the information stored must be authenticated, e.g. by using widely visible media [2] or timestamps [28]. Based on either approach many technical specifications for long-term archives have been proposed, e.g. Cumulative Notarization [34], Content Integrity Service [27], Advanced Electronic Signatures [19,18], Evidence Record Syntax [23,5], and Attested Certificates [45]. A more detailed description of archiving schemes providing a proof of existence can be found in [46].

The setup of all these approaches is quite similar. On a high level, three types of entities are considered: *users, archives,* and *trusted third parties.* A *user* owns a document that it wants archived. An *archive* is responsible for archiving the documents of one or more users. The *trusted third party* (TTP), upon request, issues a proof of existence. This proof allows to verify that a document existed at a certain point in time and has not been changed since. To archive a document the user sends it to an archive which requests a proof of existence from a TTP and stores this proof together with the document. However, none of these solutions provide confidentiality.

Thus, solutions have been proposed using asymmetric encryption schemes [32,17,31] or symmetric encryption schemes [1,47,7]. Nevertheless, if data is encrypted using a standard encryption scheme, it remains confidential only for a few decades. In some scenarios this might be sufficient, but not when dealing with highly sensitive data, for instance military and government secrets, industrial secrets, lawsuit records, election data, and medical records. This information requires long-term or even everlasting confidentiality.

Therefore several archiving schemes have been proposed that addresses proof of existence, integrity, and confidentiality. However, they either use only secret sharing and are therefore not secure in the long-term [44], assume a trusted party that is allowed to see all shares [30], or grow exponentially in the number of shareholders and shares [40].

Thus, in this section we show how the proactive secret sharing protocol by Gupta and Gopinath [26] can be improved using the prolongable bindingness of Pedersen commitments, and how this allows to combine it with long-term archiving.

5.2 Long-term storage

Before we discuss how to combine long-term archiving with proactive secret sharing we provide a brief overview of Gupta and Gopinath's verifiable secret redistribution process G^2_{Its}. For details regarding the protocol, such as the technical details, the complaint mechanism, and the security analysis, we refer to the publication by Gupta and Gopinath [26]. Their protocol consists of three subprotocols: initialization, share renewal, and reconstruction.

Initialization During initialization a new document $d \in \mathcal{M}$ is stored over n shareholders such that $m \leq n$ shareholders are needed to reconstruct the document, while no information about the document can be obtained from any $m-1$ shares. More precisely, the owner C of document $d \in \mathcal{M}$ chooses a random number $r \in \mathcal{R}$ and computes commitment $c_0 = \mathsf{Com}(d,r)$. Then C splits both opening values d and r of the commitment into shares and distributes them over the n shareholder using Shamir's Secret Sharing [43].

In addition, C computes some audit data and broadcasts them together with the commitment $c_0 = \mathsf{Com}(d, r)$ to document d. Each shareholder i then verifies its input by using its share pair and the data broadcast. If the shareholder found wrongly generated shares, then this is resolved using the complaint mechanism. If not then it stores the witness c_0 together with the shares to the opening values, that is document share d_i and random value share r_i.

Share renewal To renew and redistribute the shares to a new set of n' shareholders such that $m' \leq n'$ shares are needed to reconstruct document d, each old shareholder i computes new shares and sends one pair of new shares $(d'_{i,j}, r'_{i,j})$ to each shareholder j. In addition, each old shareholder i computes some audit data and broadcasts this together with witness $c_0 = \mathsf{Com}(d, r)$, and the commitment to its own shares $c'_{i,0} = \mathsf{Com}(d_i, r_i)$. Afterwards, each old shareholder deletes its share pair. Each new shareholder j then verifies the correctness of all received shares and checks whether the set of old shares was correct using the broadcast commitments $c'_{i,0}$ and witness c_0. Each new shareholder j takes a subset of shares for which both tests were valid, generates its pair of shares (d'_j, r'_j), and stores its share pair together with witness c_0.

Reconstruction To reconstruct a document d the retriever R, e.g. the owner of the document or another authorized person, requests the share pair (d_i, r_i) from each shareholder i, for $i \in [1, n]$ together with witness $c_0 = \mathsf{Com}(d, r)$ and checks for each m-subset of shareholders whether the shares are correct using witness c_0. As soon as R finds a set for which this test succeeds it reconstructs d.

Under the assumption that (1) all communication channels used guarantee reliable delivery of messages, (2) any two shareholders can communicate via a private channel, (3) all shareholders can receive messages sent over a broadcast channel, (4) any shareholder can declare and no shareholder can spoof its identity, and (5) a majority of the shareholders is trustworthy, the verifiable secret redistribution protocol ensures correctness of the shares generated and provides long-term confidentiality for the data stored.

5.3 Long-term archiving of secretly shared data

The protocol described in the last subsection allows storing documents while preserving long-term confidentiality under the assumption that a majority of shareholders act honestly and do not reveal their shares stored. This is a standard assumption for secret sharing inherent to the approach. However, using proactive secret sharing this is actually not only an assumption for confidentiality, but also for integrity. In $\mathrm{G}^2_{\mathrm{Its}}$ each old shareholder sends witness $\mathsf{Com}(d, r)$ to d to each new shareholder. Thus, if a majority of old shareholders is dishonest and sends a commitment to d' instead of a commitment to d the document can be replaced without being noticed. In practise this might not be acceptable when, for instance, proving the existence and integrity of an industrial secret to a court. There are trust assumptions made by many long-term archives, e.g. there exists a trusted third party (TTP) that generates timestamps. However, here a suitable trusted third party can be chosen, e.g. a civil law notary or a notary for patents. This is a significant difference to solutions relying on the trustworthiness of some archiving servers run by a company. Furthermore, this approach does not provide a proof of existence. Note that, the witness needed during the share renewal and reconstruction phase is a Pedersen commitment whose security will fade out if this protocol runs for a long period of time. Thus, if we do not assume that a majority of shareholders is trustworthy, combining this time-bounded approach with long-term archiving is meaningless without a solution to prolong the bindingness of Pedersen commitments.

Thus in this section we show how to obtain an archiving scheme providing both long-term integrity and confidentiality by combining long-term archiving solutions with proactive secret sharing. As a basis we use the verifiable secret redistribution process $\mathrm{G}^2_{\mathrm{Its}}$ described in the last section and extend the protocol such that it provides two additional properties. First, it provides a proof of existence without assuming the trustworthiness of a majority of shareholders and, second, the process is adapted such that it can be run for a long period of time using the finding that Pedersen commitments have prolongable bindingness.

With respect to the first task the challenge is to identify which data needs to be authenticated by the TTP to provide proofs of existence and integrity. Note that this data should not reveal any information about the content of the document shared. Thus, we cannot simply sign the shares since for everlasting confidentiality they must be deleted during the share renewal process. Furthermore, the TTP and everyone else that gets to see the authenticated data learn some information about the shares which violates long-term confidentiality. Thus, we authenticate, using any archiving scheme providing long-term integrity, witness c_0 to the document. The binding property of the Pedersen commitment allows for long-term integrity while the hiding property ensures everlasting confidentiality. We see in the protocol described above that during share renewal and during reconstruction integrity is provided with the help of c_0. In fact c_0 can only be opened, if the opening values d and r were reconstructed correctly using the shares received from the shareholders (this is also the statement the security of $\mathrm{G}^2_{\mathrm{Its}}$ relies on). This can easily be shown by contradiction, because if an attacker is able to modify document d to d', where $d \neq d'$ and to adapt r' without being detected, then it found an efficient algorithm violating the bindingness property of the commitment scheme used. Thus, proof of existence and integrity can be gained by authenticating witness c_0. Note that this ensures integrity even in the presence of a majority of untrusted shareholders, since the witness is authenticated by a TTP and can therefore not be modified by the shareholders. In addition, the Pedersen commitments $\mathsf{Com}(d_i, r_i)$ to the latest shares can also be authenticated. This allows R to identify a cheating shareholder i by checking whether the shares d_i and r_i the shareholder sent are the correct opening values to the authenticated commitment $\mathsf{Com}(d_i, r_i)$. Note that the ability to delete the shares and all information about the shares is essential to ensure long-term confidentiality. However, this is still possible, since Pedersen commitments are unconditionally hiding and therefore the authenticated commitments $\mathsf{Com}(d_i, r_i)$ do not reveal any information about replaced shares.

On the other hand Pedersen commitments are only computationally binding, which leads us to the second task. When the document is archived for a long period of time the witness $c_0 = \mathsf{Com}(d, r)$ generated and used within the $\mathrm{G}^2_{\mathrm{Its}}$ protocol will become insecure for the parameters chosen. This affects the security of the archived document, because the verifiable secret redistribution process uses c_0 to ensure integrity during the share renewal and reconstruction process. Thus, the $\mathrm{G}^2_{\mathrm{Its}}$ protocol has to be adapted such that it deals with the aging of Pedersen commitments. This can be achieved by adding an additional procedure to the protocol in which the witness is renewed. More precisely, first the owner requests the shares, generates a new commitment, and proves correctness to the TTP. The TTP then generates a new witness by authenticating the old and the new commitment together with the transcript of the proof. This data can be used to extend the lifetime of the proof of existence and integrity for document beyond the lifetime of the initially generated commitment. Afterwards, the prover generates a new set of shares and sends these shares together with the new authenticated witness to the shareholders. Finally, the old shareholders delete their old share pair.

Enhancing the proactive secret sharing protocol by Gupta and Gopinath with authenticated commitments also further improves the proactive secret sharing protocol itself. The shareholders, for instance, can make use of the authenticated commitments while less information must be broadcast for verifiability. This also shortens the time the retriever R of document d needs for reconstruction. Instead of finding a correct subset of shares, R can simply reconstruct d and decommitment value r using a randomly selected subset of shares and check whether the values open the authenticated commitment $\mathsf{Com}(d, r)$.

6 Conclusion and Future work

In this paper, we showed how to efficiently and securely transfer a Pedersen commitment to a message m from one set of ambient parameters to another, thereby renewing the security of the computational bindingness. Our construction essentially amounts to a perfect zero-knowledge proof of message equivalence for commitments with non-isomorphic message spaces, in particular allowing us to maintain information-theoretic confidentiality. This makes the method suitable for long-lived systems that process confidential data. To illustrate this we showed how our construction can be used to store a document in a proactive secret sharing fashion while preserving long-term integrity. Since this can be provided even in the presence of untrusted shareholders we could use this as a basis to develop an archiving scheme providing not only proof of existence and long-term integrity but also long-term confidentiality for the data archived. While we pointed out long-term archiving as a potential application for this procedure, it is worth noting that our techniques do not alter the Pedersen commitment protocol itself - indeed, only the parameters are enlarged. Hence, they may be easily integrated into other constructs that use these commitments. Also, our new perfect zero-knowledge proof may be of independent interest.

For future work, it could be first of all interesting to see if our procedure can be simplified or optimized. An indication that this might be possible is that some of the zero-knowledge proofs perform overlapping tasks. It could also be interesting to investigate which concrete protocols for these proofs are the most suited to this construction. Secondly, we would like to see if a similar protocol can be devised for other instantiations of commitment schemes, or even generically. As mentioned before prolonging the security of Pedersen commitments does not protect against attacks from quantum computers. Therefore, for future work we plan to further investigate how the security of post-quantum secure commitments can be prolonged. Furthermore, we would like to examine how a prover could show that two commitments generated with two different schemes, e.g. a discrete logarithm-based commitment scheme and a lattice-based commitment scheme, commit to the same message. Finally, the idea of securely renewing the security level of deployed schemes should be examined for other cryptographic services.

Acknowledgments

This work has been co-funded by the DFG as part of project Long-Term Secure Archiving within the CRC 1119 CROSSING. This project has also received funding from the European Union's Horizon 2020 research and innovation program under Grant Agreement No 644962. Finally, this work was partially funded by the Fonds National de la Recherche Luxembourg.

References

1. Adya, A., Bolosky, W.J., Castro, M., Cermak, G., Chaiken, R., Douceur, J.R., Howell, J., Lorch, J.R., Theimer, M., Wattenhofer, R.: Farsite: Federated, available, and reliable storage for an incompletely trusted environment. In: OSDI (2002)
2. Bayer, D., Haber, S., Stornetta, W.S.: Improving the efficiency and reliability of digital time-stamping. In: Sequences II: Methods in Communication, Security and Computer Science. pp. 329–334. Springer-Verlag (1993)
3. Bellare, M., Goldreich, O.: On defining proofs of knowledge. In: Brickell, E. (ed.) Advances in Cryptology - CRYPTO '92, Lecture Notes in Computer Science, vol. 740, pp. 390–420. Springer Berlin Heidelberg (1993), `http://dx.doi.org/10.1007/3-540-48071-4_28`
4. Bellare, M., Goldwasser, S.: Verifiable partial key escrow. In: Proceedings of the 4th ACM Conference on Computer and Communications Security. pp. 78–91. CCS '97, ACM, New York, NY, USA (1997)
5. Blazic, A.J., Saljic, S., Gondrom, T.: Extensible Markup Language Evidence Record Syntax (XMLERS). RFC 6283 (Proposed Standard) (Jul 2011), `http://www.ietf.org/rfc/rfc6283.txt`
6. Boudot, F.: Efficient proofs that a committed number lies in an interval. In: Advances in Cryptology - EUROCRYPT 2000, Bruges, Belgium, May 14-18, 2000, Proceeding. pp. 431–444 (2000)

7. Braun, J., Wiesmaier, A., Buchmann, J.: On the security of encrypted secret sharing. In: HICSS. pp. 4966–4975 (2013)
8. Camenisch, J., Chaabouni, R., Shelat, A.: Efficient protocols for set membership and range proofs. In: Advances in Cryptology - ASIACRYPT 2008, Melbourne, Australia, December 7-11, 2008. Proceedings. pp. 234–252 (2008)
9. Canard, S., Coisel, I., Jambert, A., Traoré, J.: New results for the practical use of range proofs. In: Public Key Infrastructures, Services and Applications - 10th European Workshop, EuroPKI 2013, Egham, UK, September 12-13, 2013, Revised Selected Papers. pp. 47–64 (2013)
10. Centre of Registers and Information Systems: e-land register, `http://www.egov-estonia.eu/e-land-register`
11. Chaabouni, R., Lipmaa, H., Shelat, A.: Additive combinatorics and discrete logarithm based range protocols. In: Information Security and Privacy - 15th Australasian Conference, ACISP 2010, Sydney, Australia, July 5-7, 2010. Proceedings. pp. 336–351 (2010)
12. Chaum, D., Pedersen, T.P.: Wallet databases with observers. In: Advances in Cryptology - CRYPTO '92, Santa Barbara, California, USA, August 16-20, 1992, Proceedings. pp. 89–105 (1992)
13. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of partial knowledge and simplified design of witness hiding protocols. In: Advances in Cryptology - CRYPTO '94, Santa Barbara, California, USA, August 21-25, 1994, Proceedings. pp. 174–187 (1994)
14. Damgård, I., Jurik, M.: A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In: proceedings of PKC 2001, LNCS series. pp. 119–136. Springer-Verlag (2001)
15. Department of Health: Annex d1: Health records retention schedule, `http://www.dh.gov.uk/prod_consum_dh/groups/dh_digitalassets/documents/digitalasset/dh_093027.pdf`
16. Desmedt, Y., Jajodia, S.: Redistributing secret shares to new access structures and its applications (1997)
17. Druschel, P., Rowstron, A.I.T.: Past: A large-scale, persistent peer-to-peer storage utility. In: HotOS. pp. 75–80 (2001)
18. ETSI: CMS Advanced Electronic Signatures (CAdES). No. TS 101 733, 1.7.4 edn. (Jul 2010)
19. ETSI: XML Advanced Electronic Signatures (XAdES). No. TS 101 903, 1.8.3 edn. (jan 2010)
20. Feldman, P.: A practical scheme for non-interactive verifiable secret sharing. In: FOCS. pp. 427–437 (1987)
21. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings. pp. 186–194 (1986)
22. Goldreich, O.: Foundations of Cryptography: Volume 1. Cambridge University Press, New York, NY, USA (2006)
23. Gondrom, T., Brandner, R., Pordesch, U.: Evidence Record Syntax (ERS) (2007), `http://www.ietf.org/rfc/rfc4998.txt`
24. Groth, J.: Non-interactive zero-knowledge arguments for voting. In: Applied Cryptography and Network Security, Third International Conference, ACNS 2005, New York, NY, USA, June 7-10, 2005, Proceedings. pp. 467–482 (2005)
25. Gupta, V.H., Gopinath, K.: An extended verifiable secret redistribution protocol for archival systems. In: ARES. pp. 100–107 (2006)

26. Gupta, V.H., Gopinath, K.: G^2_{its} vsr: An information theoretical secure verifiable secret redistribution protocol for long-term archival storage. In: Proceedings of the Fourth International IEEE Security in Storage Workshop. pp. 22–33. SISW '07, IEEE Computer Society, Washington, DC, USA (2007)
27. Haber, S.: Content Integrity Service for Long-Term Digital Archives. In: Archiving 2006. pp. 159–164. IS&T, Ottawa, Canada (2006)
28. Haber, S., Stornetta, W.S.: How to time-stamp a digital document. J. Cryptology 3(2), 99–111 (1991)
29. Herzberg, A., Jarecki, S., Krawczyk, H., Yung, M.: Proactive secret sharing or: How to cope with perpetual leakage. In: CRYPTO. pp. 339–352 (1995)
30. Hühnlein, D., Korte, U., Langer, L., Wiesmaier, A.: A comprehensive reference architecture for trustworthy long-term archiving of sensitive data. In: NTMS. pp. 1–5 (2009)
31. Kotla, R., Alvisi, L., Dahlin, M.: Safestore: A durable and practical storage system. In: USENIX Annual Technical Conference. pp. 129–142 (2007)
32. Kubiatowicz, J., Bindel, D., Chen, Y., Czerwinski, S., Eaton, P., Geels, D., Gummadi, R., Rhea, S., Weatherspoon, H., Weimer, W., Wells, C., Zhao, B.: Oceanstore: An architecture for global-scale persistent storage. SIGPLAN Not. 35(11), 190–201 (Nov 2000)
33. Lamport, L., Shostak, R.E., Pease, M.C.: The byzantine generals problem. ACM Trans. Program. Lang. Syst. 4(3), 382–401 (1982)
34. Lekkas, D., Gritzalis, D.: Cumulative notarization for long-term preservation of digital signatures. Computers & Security 23(5), 413–424 (2004)
35. Lenstra, A.K.: Key length (2004)
36. Lipmaa, H.: On diophantine complexity and statistical zero-knowledge arguments. In: Advances in Cryptology - ASIACRYPT 2003, Taipei, Taiwan, November 30 - December 4, 2003, Proceedings. pp. 398–415 (2003)
37. Lipmaa, H., Asokan, N., Niemi, V.: Secure vickrey auctions without threshold trust. In: Financial Cryptography, 6th International Conference, FC 2002, Southampton, Bermuda, March 11-14, 2002, Revised Papers. pp. 87–101 (2002)
38. Moran, T., Naor, M.: Split-ballot voting: Everlasting privacy with distributed trust. ACM Trans. Inf. Syst. Secur. 13(2) (2010)
39. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: CRYPTO. pp. 129–140 (1991)
40. Ramos, T.A., da Silva, N., Lung, L.C., Kohler, J.G., Custódio, R.F.: An infrastructure for long-term archiving of authenticated and sensitive electronic documents. In: EuroPKI. pp. 193–207 (2010)
41. Schnorr, C.: Efficient signature generation by smart cards. J. Cryptology 4(3), 161–174 (1991)
42. Schoenmakers, B.: Some efficient zero-knowledge proof techniques. In: Workshop on Cryptographic Protocols (2001)
43. Shamir, A.: How to share a secret. Commun. ACM 22(11), 612–613 (1979)
44. Storer, M.W., Greenan, K.M., Miller, E.L., Voruganti, K.: Potshards - a secure, recoverable, long-term archival storage system. TOS 5(2) (2009)
45. Vigil, M.A.G., Cabarcas, D., Buchmann, J., Huang, J.: Assessing trust in the long-term protection of documents. In: ISCC. pp. 185–191 (2013)
46. Vigil, M., Buchmann, J., Cabarcas, D., Weinert, C., Wiesmaier, A.: Integrity, authenticity, non-repudiation, and proof of existence for long-term archiving: a survey. Tech. rep., TU Darmstadt (2015), to appear in Computers & Security
47. Wang, E.K., Yau, J.C.K., Hui, L.C.K., Jiang, Z.L., Yiu, S.M.: A key-recovery system for long-term encrypted documents. In: EDOC Workshops. p. 52 (2006)

48. Wong, T.M., Wang, C., Wing, J.M.: Verifiable secret redistribution for archive system. In: IEEE Security in Storage Workshop. pp. 94–106 (2002)
49. Yuen, T.H., Huang, Q., Mu, Y., Susilo, W., Wong, D.S., Yang, G.: Efficient non-interactive range proof. In: Computing and Combinatorics, 15th Annual International Conference, COCOON 2009, Niagara Falls, NY, USA, July 13-15, 2009, Proceedings. pp. 138–147 (2009)