Witness-Hiding Proofs of Knowledge for Cable Locks

Chen-Da Liu Zhang, Ueli Maurer, Martin Raszyk, Daniel Tschudi
Department of Computer Science, ETH Zurich, Switzerland
{lichen, maurer, tschudid}@inf.ethz.ch, mraszyk@student.ethz.ch

***Abstract*—We consider the general setting where users need to provide a secret code c to a verifying entity V in order to obtain access to a resource. More generally, the right to access the resource could, for example, be granted if one knows one of two codes c_1 and c_2. For privacy reasons, a party P may want to hide which of the two codes it knows and only prove that it knows at least one of them. For example, if the knowledge of a code corresponds to membership in a certain society, one may want to hide which society one belongs to. In cryptography, such a proof is called a witness-hiding proof of knowledge. How can P prove such a statement to V?**

This paper is concerned with witness-hiding proofs of knowledge using simple mechanical tools. Specifically, we consider cable (or bicycle) locks, where the codes of the locks correspond to the secret codes. The above example of proving knowledge of either c_1 or c_2 in a witness-hiding fashion can be achieved simply as follows. When given the two locks closed and unlinked (by V), P presents the configuration of the two locks interlocked, which can be generated if and only if P knows at least one of the codes.

In the most general case with n codes $c_1, \ldots, c_n$, the access right is characterized by a so-called knowledge structure $\Gamma \subseteq \mathcal{P}(\{1, \ldots, n\})$, a subset of the power set of $\{1, \ldots, n\}$. Access is granted if a user knows the codes corresponding to any of the subsets of Γ. We present lock-based protocols for witness-hiding proofs of knowledge for any such monotone knowledge structure, and investigate the efficiency (i.e., in particular, the number of lock configurations that P must present) in several settings such as the availability of solid rings or the availability of multiple locks for a given code.

The topic of this paper is similar in spirit to other works, such as the picture hanging puzzles by Demaine et al., which explore connections between topology and real-world applications, where the motivation arises also, or even primarily, from mathematical curiosity.

I. Introduction

A similar protocol to the one in the abstract works in the case where P wants to prove that he knows either[1] a code c_1, or both codes c_2 and c_3. In this case, V gives P the three locks closed and unlinked, and P presents the configuration of the first lock interlocked with the other two. Since such a configuration can be generated if and only if P either knows c_1, or both c_2 and c_3, V is convinced. However, V does not learn which set of locks P can actually open.

An interesting question for this type of protocols is: how can P prove that he knows the code to open at least one out of three locks? And k out of n? In this paper, we settle these questions in the general case. We consider a set of n locks $\mathcal{L} = \{L_1, \ldots, L_n\}$ and a knowledge structure $\Gamma \subseteq \mathcal{P}(\mathcal{L})$. We provide protocols that allow P to convince V that he can open all the locks in some set $\mathcal{W} \in \Gamma$ without revealing $\mathcal{W}$ itself.

[1]note that *either* is considered to be non-exclusive

A. Related Work

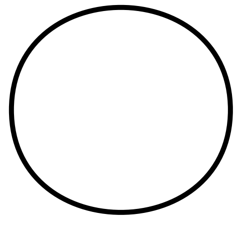
An interactive proof [GMR89] is a protocol which allows a prover P to convince a verifier V of some statement. An interactive proof has to be complete and sound. Completeness means that an honest prover succeeds in convincing an honest verifier, and soundness means that a dishonest prover does not succeed in convincing V of a false statement. A proof of knowledge is a special type of interactive proof where the prover shows knowledge of a witness (e.g., keys for locks) for some statement. Such a protocol is called witness-hiding if the verifier does not learn the witness from the protocol execution [FS90]. A stronger notion is zero-knowledge. Here, the interactive proof is performed in such a way that the protocol transfers only the fact that the claimed statement is true but does not leak any further information. More precisely, in a zero-knowledge proof the verifier is able to simulate the entire protocol transcript by himself without interacting with the prover. In particular, this implies that the transcript is not convincing for any other party [GMR89], [GMW91].

In our proofs of knowledge, V first gives P a configuration of locks. Then, P proves his knowledge by altering this configuration in a specific way. Similar ideas where used in [DDM$^+$12] for the picture hanging problem. There, the goal is to wrap a rope around n nails so that a picture hung at the rope falls whenever certain subsets of the nails get removed.

B. Contributions and Outline

In this work, we define an efficiency notion and provide witness-hiding proofs of knowledge for cable locks in the described setting. Our protocols operate under various assumptions such as availability of solid rings (locks where the key is unknown), or ability to clone locks (to obtain locks that can be opened using the same key).

We proceed as follows. We first introduce the basic concepts and techniques necessary for the rest of the paper. Then, we formalize the notion of a witness-hiding proof of knowledge and define a complexity notion. In the second part, we present four witness-hiding proofs of knowledge. The first protocol assumes that locks can be copied and that solid rings are available. The second protocol assumes that locks can be copied. The third protocol assumes the availability of solid rings. The last protocol assumes neither the possibility to copy


(a) Unknot

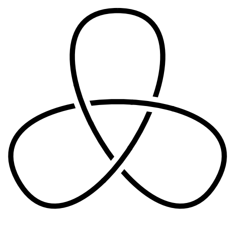

(b) Trefoil Knot

Figure 1: Basic Knots

locks, nor the availability of solid rings. Finally, we compare the described protocols in terms of the complexity notion and discuss briefly a stronger definition for the proof of knowledge.

II. Preliminaries

A. Knot Theory

A knot is a closed curve (a closed tangled string) in the three-dimensional Euclidean space $\mathbb{R}^3$. A trivial knot (or an unknot) is a closed curve without a knot in it. This means it is a (possibly deformed) closed circle. A trefoil knot is the simplest example of a nontrivial knot. Both unknot and trefoil knot are depicted in Figure 1. One cannot obtain the trefoil knot from the unknot without opening it. A link is the union of some mutually disjoint knots. A Brunnian link is a nontrivial link which becomes a collection of unlinked circles if any component is removed. We denote a Brunnian link that consists of n unknots as an n-component Brunnian link.

Theorem 1 ([Bru92]). *There exists an n-component Brunnian link for any $n \geq 2$.*

B. Boolean Formulas

A Boolean formula over atomic formulas $A_1, \ldots, A_n$ is *monotone* if it does not contain any negation. More formally, *monotone Boolean formulas* are defined inductively as follows. Any atomic formula A_i is a monotone Boolean formula. If $F_1, \ldots, F_k$ are monotone Boolean formulas, then also the conjunction $\bigwedge_{i=1}^{k} F_i$ and disjunction $\bigvee_{i=1}^{k} F_i$ are monotone Boolean formulas. A monotone Boolean formula can be seen as a rooted tree, where every leaf corresponds to an atomic formula, and each of the inner nodes corresponds to the conjunction or the disjunction of the children nodes.

A monotone Boolean formula is in conjunctive normal form (CNF) if it is a conjunction of clauses, where a clause is a disjunction of atomic formulas. Similarly, a monotone Boolean formula is in disjunctive normal form (DNF) if it is a disjunction of clauses, where each clause is a conjunction of atomic formulas.

For a monotone Boolean formula F we denote by $|F|$ the number of atomic formulas in F and by $\|F\|$ the number of distinct atomic formulas in F. For example, a formula F with $|F| = 3$ and $\|F\| = 2$ is $F = (A_1 \wedge A_2) \vee A_2$. If F is in CNF, then $\gamma_{\mathsf{CNF}}(F)$ stands for the number of disjunctive clauses of F and if F is in DNF, then $\gamma_{\mathsf{DNF}}(F)$ stands for the number of conjunctive clauses of F.

III. Locks and Proof of Knowledge

In this work we consider a finite set $\mathcal{L} = \{L_1, \ldots, L_n\}$ of idealized cable *locks*. We assume that such locks can be bent or stretched arbitrarily but are otherwise unbreakable. In particular, a lock L_i can only be opened using the corresponding key k_i. A lock is called a *clone* or *copy* of another lock if they can be opened using the same key. We assume that one can determine whether two locks are clones without having access to their keys. For instance, locks with different keys could be colored differently. Furthermore, we consider *solid rings* which one can bend or stretch arbitrarily but are otherwise unbreakable. They correspond to locks where the corresponding keys are unknown. We assume that solid rings come in the form of unknots.

Let $\Gamma \subseteq \mathcal{P}(\mathcal{L})$ be a collection of subsets of $\mathcal{L}$. The goal of *prover* P is to convince *verifier* V that he can open all the locks in some set $\mathcal{W} \in \Gamma$, i.e., that he knows all the keys for the locks in $\mathcal{W}$. However, P does not want V to learn $\mathcal{W}$ itself.

Definition 1. *A protocol π between a prover P and a verifier V is a* witness-hiding proof of knowledge *for $(\mathcal{L}, \Gamma)$ if the following three conditions are satisfied:*

Completeness: *If P knows all the keys for some $\mathcal{W} \in \Gamma$ and follows the protocol, then P can convince the verifier V.*

Soundness: *Any prover who does not know all the keys for any set in Γ does not succeed in convincing the verifier.*

Witness-Hiding: *Let $\mathcal{W}, \mathcal{W}' \in \Gamma$ and $\mathcal{W} \neq \mathcal{W}'$. The view of V in a protocol execution where P knows the keys for $\mathcal{W}$ is indistinguishable from the view of V in a protocol execution where P knows the keys for $\mathcal{W}'$.*

The first condition ensures that an honest P can convince V that he knows the keys of some set $\mathcal{W} \in \Gamma$. The second condition ensures that a dishonest P who does not know all the keys of any $\mathcal{W}$ does not succeed in convincing V. The third condition ensures that a dishonest V cannot extract any information about P's set of keys from the protocol execution except what is given by $(\mathcal{L}, \Gamma)$.

Observe that a proof of knowledge for Γ is also a proof of knowledge for the set $\{\mathcal{W}' \subseteq \mathcal{L} \mid \exists \mathcal{W} \in \Gamma : \mathcal{W} \subseteq \mathcal{W}'\}$ which is its *monotone closure* with respect to taking supersets. That comes from the fact that if P knows all the keys of locks in $\mathcal{W}'$, then he also knows all the keys of locks in the set $\mathcal{W} \subseteq \mathcal{W}'$. Given a collection Γ, it is thus enough to prove knowledge for its *minimal sets* $\overline{\Gamma} := \{\mathcal{W} \in \Gamma \mid \nexists \mathcal{W}' \in \Gamma : \mathcal{W}' \subsetneq \mathcal{W}\}$. In the following we assume that $\Gamma = \overline{\Gamma}$, i.e., that Γ does not contain any unnecessary sets. To compare different proofs of knowledge, we use the following complexity notion.

Definition 2. *A witness-hiding proof of knowledge π for $(\mathcal{L}, \Gamma)$ has complexity (ℓ, s, r) if it requires ℓ locks, s solid rings, and runs for r rounds.*

In each round the verifier gives the prover a link. The prover alters the link and gives it back to the verifier. This means that the number of rounds equals the total number of links which are *successively* given to the verifier. To express the complexity of our protocols, we associate Γ with a monotone Boolean formula. We define for every lock $L_i \in \mathcal{L}$ the atomic formula A_i which stands for the statement "P knows the key of lock L_i". This allows to represent the statement "P knows all keys of some set $\mathcal{W}$ in Γ" as the following monotone Boolean formula

$$F_\Gamma := \bigvee_{\mathcal{W}\in\Gamma} \left(\bigwedge_{L_i \in \mathcal{W}} A_i \right). \tag{1}$$

IV. Proof of Knowledge Protocols

In this section, we present witness-hiding proofs of knowledge for some given $(\mathcal{L}, \Gamma)$ under different assumptions such as the availability of lock copies or solid rings.

A. Multiple Lock Copies and Solid Rings

We show a witness-hiding proof of knowledge under the assumption that P and V may use an arbitrary amount of copies of each lock in $\mathcal{L}$. It is also assumed that both parties have access to a link that consists of two interlocked solid rings. P and V both consider a single monotone Boolean formula F which is equivalent to F_Γ. First, V constructs[2] the link L_F using Algorithm 1 and gives the link to P. The link L_F has the shape of an unknot and can be opened if and only if a party has knowledge of a set of keys $\mathcal{W} \in \Gamma$. P then proves his knowledge by transforming the link into a trefoil shape.

Algorithm 1: Link L_F for F

```
Require: Monotone Boolean formula $F$ (seen as a tree).
  $L_F$ is initialized as the set of unlinked and unknotted cable locks corresponding to the individual occurrences of atomic formulas in $F$.
  for each level $i$ of $F$ starting from the leaves to the root do
      for each conjunction at level $i$ do
          Place all children links in parallel in any order.
      for each disjunction at level $i$ do
          Link all children links as a chain in any order.
  Form a closed unknot-shaped structure with $L_F$ by connecting each extreme of $L_F$ with one of the two interlocked solid rings, respectively.
  return $L_F$
```

The resulting link L_F has the shape of an unknot[3]. Observe that by construction, if P knows a set of keys that correspond to a satisfying assignment of F, then P is able to open L_F (the solid rings cannot be opened). An example of the link L_F for a given formula is depicted in Figures 2 and 3.

[2]This assumes that V can open all locks in $\mathcal{L}$.

[3]Observe that the two solid rings ensure that L_F is a link with the structure of an unknot.

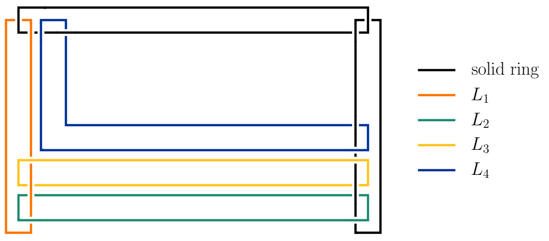


Figure 2: The link created by Vic in protocol **Trefoil** for the formula $(A_1 \vee (A_2 \wedge A_3)) \wedge A_4$.

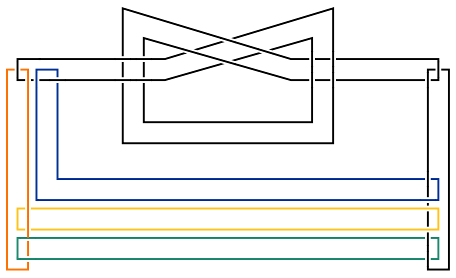


Figure 3: The link transformed by Peggy

Protocol Trefoil$(\mathcal{L}, F)$

Require: F is a monotone Boolean formula over $\mathcal{L}$.

1: V creates link L_F according to Algorithm 1, which has the shape of an unknot, and gives it to P.
2: P transforms (in the absence of V) L_F into a trefoil-shaped structure, and gives the resulting link to V.
3: V checks if the received link is a trefoil-shaped modification of the L_F. If so, V accepts. Otherwise, V rejects.

Lemma 1. *Let $\mathcal{L}$ be a nonempty set of locks and $\Gamma \subseteq \mathcal{P}(\mathcal{L})$. Let F be a monotone Boolean formula which is equivalent to F_Γ from* (1). *Then* **Trefoil**$(\mathcal{L}, F)$ *is a witness-hiding proof of knowledge for* $(\mathcal{L}, \Gamma)$ *with complexity* $(|F|, 2, 1)$.

Proof. Completeness: Assume that P knows all the keys for some $\mathcal{W} \in \Gamma$. This implies that the set of keys P knows corresponds to a satisfying assignment of F_Γ. As F is equivalent to F_Γ, P knows the keys to open L_F, and hence he is able to construct a trefoil-shaped knot with it. *Soundness:* Assume that a (dishonest) P does not know all the keys for any $\mathcal{W} \in \Gamma$. In this case he cannot open L_F. Thus he cannot construct a trefoil-shaped structure with it and fails to convince V. *Witness-Hiding:* The protocol is witness-hiding since the link constructed by P is fully defined by F which is known to V in advance. It does not depend on the actual witness $\mathcal{W}$ of P. *Complexity:* The protocol has complexity $(|F|, 2, 1)$ since it uses $|F|$ locks, two solid rings, and it is executed in one round. $\square$

B. Multiple Lock Copies

In this section, we assume the availability of an arbitrary amount of copies of each lock in $\mathcal{L}$. In the protocol

BrunnianLinkDNF, P and V consider a single monotone Boolean formula F in DNF which is equivalent to F_Γ. For instance, they could use $F = F_\Gamma$. If P is honest, he is able to open all locks corresponding to a clause of F. P proves his knowledge by constructing a Brunnian link which contains a component for each clause of F. An example for such a link is depicted in Figure (4).

Protocol BrunnianLinkDNF$(\mathcal{L}, F)$

Require: $F = \bigvee_i \bigwedge_j A_{i_j}$ is a monotone Boolean formula in DNF over $\mathcal{L}$.

1: For each individual occurrence of an atomic formula A_{i_j} in the formula F, V gives P a copy of the lock L_{i_j}. The locks P receives are closed, unlinked, and unknotted.

2: P constructs (in the absence of V) a Brunnian link with $\gamma_{\mathsf{DNF}}(F)$ components, i.e., one component per conjunctive clause. The component for a clause $A_{i_1} \wedge \cdots \wedge A_{i_k}$ consists of the locks $L_{i_1}, \ldots, L_{i_k}$ placed parallel to each other. Finally, P gives the constructed link to V.

3: If the received link is constructed as required by the protocol, V accepts. Otherwise, V rejects.

Lemma 2. *Let $\mathcal{L}$ be a nonempty set of locks and $\Gamma \subseteq \mathcal{P}(\mathcal{L})$. Let F be a monotone Boolean formula in DNF which is equivalent to F_Γ from* (1). *Then* **BrunnianLinkDNF**$(\mathcal{L}, F)$ *is a witness-hiding proof of knowledge for* $(\mathcal{L}, \Gamma)$ *with complexity* $(|F|, 0, 1)$.

Proof. Completeness: Assume that P knows all the keys for some $\mathcal{W} \in \Gamma$. Then P can open all locks corresponding to at least one conjunctive clause of F, i.e., all locks of one component of the considered Brunnian link. This is enough to construct the Brunnian link from the unlinked components and thus to convince V. *Soundness:* Assume that a (dishonest) P does not know all the keys for any $\mathcal{W} \in \Gamma$. In this case he cannot open all the locks corresponding to any conjunctive clause of F. Thus he cannot build the required Brunnian link and fails to convince V. *Witness-Hiding:* The protocol is witness-hiding since the Brunnian link constructed by P is fully defined by F which is known to V in advance. It does not depend on the actual witness $\mathcal{W}$ of P. *Complexity:* The link contains $|F|$ locks, and it does not contain solid rings. □

C. Solid Rings

In this section, we assume that each lock is unique, i.e., there are no copies of locks available. Instead, both parties have access to arbitrarily many solid rings. For the following protocol, P and V use a monotone Boolean formula F in CNF which is equivalent to F_Γ. The idea is to construct a link which contains a Brunnian link for each disjunctive clause in F. An example construction is depicted in Figure (5).

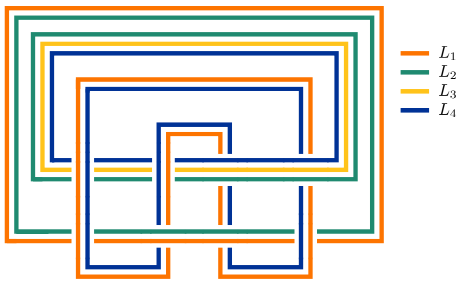


Figure 4: The link constructed in **BrunnianLinkDNF** for the formula $(A_1 \wedge A_2) \vee (A_2 \wedge A_3 \wedge A_4) \vee (A_1 \wedge A_4)$.

Protocol BrunnianLinkCNF$(\mathcal{L}, F)$

Require: $F = \bigwedge_i \bigvee_j A_{i_j}$ is a monotone Boolean formula in CNF over $\mathcal{L}$.

1: For each atomic formula A_k which occurs at least once in F, V gives P the lock L_k. The locks P receives are closed, unlinked, and unknotted. For each disjunctive clause in F, V gives P a solid ring.

2: P arranges the locks concentrically and constructs (in the absence of V) a Brunnian link for each disjunctive clause in F using solid rings. More specifically, the Brunnian link for a clause of the form $A_{i_1} \vee A_{i_2} \vee \cdots \vee A_{i_r}$ is a Brunnian link of the locks $L_{i_1}, L_{i_2}, \ldots, L_{i_r}$ and a fresh solid ring. Finally, P gives the constructed link to V.

3: If the received link is constructed as required by the protocol, V accepts. Otherwise, V rejects.

Lemma 3. *Let $\mathcal{L}$ be a nonempty set of locks and $\Gamma \subseteq \mathcal{P}(\mathcal{L})$. Let F be a monotone Boolean formula in CNF which is equivalent to F_Γ from* (1). *Then* **BrunnianLinkCNF**$(\mathcal{L}, F)$ *is a witness-hiding proof of knowledge for* $(\mathcal{L}, \Gamma)$ *with complexity* $(\|F\|, \gamma_{\mathsf{CNF}}(F), 1)$.

Proof. Completeness: Assume that P can open all the locks in some $\mathcal{W} \in \Gamma$. This implies that the set of keys P knows corresponds to a satisfying assignment of F_Γ. As F is equivalent to F_Γ, it follows that for each clause in F, P is able to open at least one lock. This is enough to construct all Brunnian links, and thus to convince V. *Soundness:* Assume that a (dishonest) P cannot open all locks of any $\mathcal{W} \in \Gamma$. This implies that there exists a clause in F where he cannot open any lock. Thus he is not able to construct the corresponding Brunnian link, and thus fails to convince V. *Witness-hiding:* The constructed link only depends on F which is known to V in advance. It does not depend on the actual witness $\mathcal{W}$ of P. *Complexity:* The link consists of $\|F\|$ locks and $\gamma_{\mathsf{CNF}}(F)$ solid rings. □

D. Multiple rounds

In this setting, we assume that solid rings and copies of locks are unavailable. This means that the locks are

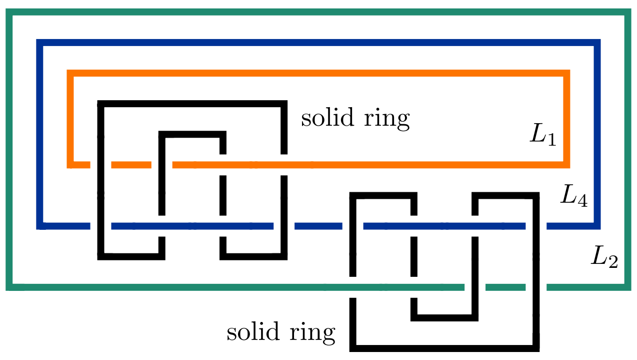


Figure 5: The link constructed in **BrunnianLinkCNF** for the formula $(A_1 \vee A_4) \wedge (A_2 \vee A_4)$. The solid rings are drawn in black.

unique. The following protocol is similar to the protocol **BrunnianLinkCNF**. As there are no solid rings available, the Brunnian links for the clauses in F are shown in successive rounds.

Protocol BrunnianLinkCNF2$(\mathcal{L}, F)$

Require: $F = \bigwedge_i \bigvee_j A_{i_j}$ is a monotone Boolean formula in CNF over $\mathcal{L}$.

1: For each atomic formula A_k which occurs at least once in F, V gives P the lock L_k. The locks P receives are closed, unlinked, and unknotted.

2: In successive rounds P constructs for each disjunctive clause $A_{i_1} \vee \cdots \vee A_{i_r}$ of F a Brunnian link consisting of locks $L_{i_1}, \dots, L_{i_r}$, and shows it to V.

3: If all the shown links have been constructed as required by the protocol, V accepts. Otherwise, V rejects.

Lemma 4. *Let $\mathcal{L}$ be a nonempty set of locks and $\Gamma \subseteq \mathcal{P}(\mathcal{L})$. Let F be a monotone Boolean formula in CNF which is equivalent to F_Γ from* (1)*. Then* **BrunnianLinkCNF2**$(\mathcal{L}, F)$ *is a witness-hiding proof of knowledge for* $(\mathcal{L}, \Gamma)$ *with complexity* $(\|F\|, 0, \gamma_{\mathsf{CNF}}(F))$.

Proof. Completeness, Soundness, Witness-hiding: The proof is similar to that of Lemma 3. *Complexity:* The number of locks that are used is equal to $\|F\|$, i.e., the number of distinct atomic formulas in F. Also, for each clause in F, P shows a link to V. Hence, the number of rounds is $\gamma_{\mathsf{CNF}}(F)$. □

E. Summary of Complexity

To conclude this section, we summarize the complexity of our protocols in the following table.

	no. locks	no. solid rings	no. rounds
Lemma 1	$\lvert F\rvert$	2	1
Lemma 2	$\lvert F_{DNF}\rvert$	0	1
Lemma 3	$\|F_{CNF}\|$	$\gamma_{\mathsf{CNF}}(F_{CNF})$	1
Lemma 4	$\|F_{CNF}\|$	0	$\gamma_{\mathsf{CNF}}(F_{CNF})$

In the above table, F denotes a monotone Boolean formula which is equivalent to F_Γ from (1), F_{DNF} denotes a monotone Boolean formula in DNF which is equivalent to F_Γ from (1), and F_{CNF} denotes a monotone Boolean formula in CNF which is equivalent to F_Γ from (1).

V. Designated Verifier Zero-Knowledge

The protocols from the previous section ensure that a dishonest V does not obtain information about the set of locks that P actually knows beyond what is given by the knowledge P wants to prove, i.e., $(\mathcal{L}, \Gamma)$. This witness-hiding property is a natural definition for security against a dishonest V in our physical setting. In some contexts one could require an even stronger notion of security. For example, one might want to ensure that V cannot secretly film the protocol execution and use the recording to convince other people about P's knowledge of opening the locks (cf. [QQQ+89]). This type of security is called *designated verifier zero-knowledge* and ensures that the proof of knowledge only convinces the designated verifier even if the verifier is dishonest [JSI96]. We can easily modify our protocols to satisfy this property. We assume that there is a designated lock $L_V \notin \mathcal{L}$ which can be opened by V, but not by (honest) P. The idea is that P shows that he can open all locks of some $\mathcal{W} \in \Gamma$ or that he can open L_V. This will still convince an honest V since he is sure that P cannot open his lock. However, any other party will not be convinced as P and V can easily simulate a protocol execution even if they cannot open any lock in $\mathcal{L}$. In the protocols that assume solid rings, the solid rings can trivially be replaced by (copies of) V's private lock L_V, whereas in the other protocols P can directly prove $(\mathcal{L} \cup \{L_V\}, \Gamma \cup \{\{L_V\}\})$.

References

[Bru92] Hermann Brunn. Über Verkettung. In *Sitzungsbericht der Bayerischen Akademie der Wissenschaft, Mathematisch Naturwissenschaftliche Abteilung*, pages 77–99, 1892.

[DDM+12] Erik D. Demaine, Martin L. Demaine, Yair N. Minsky, Joseph S. B. Mitchell, Ronald L. Rivest, and Mihai Patrascu. Picture-hanging puzzles. *CoRR*, abs/1203.3602, 2012.

[FS90] Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 416–426. ACM, 1990.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on computing*, 18(1):186–208, 1989.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems. *Journal of the ACM (JACM)*, 38(3):690–728, 1991.

[JSI96] Markus Jakobsson, Kazue Sako, and Russell Impagliazzo. Designated verifier proofs and their applications. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 143–154. Springer, 1996.

[QQQ+89] Jean-Jacques Quisquater, Myriam Quisquater, Muriel Quisquater, Michaël Quisquater, Louis Guillou, Marie Annick Guillou, Gaïd Guillou, Anna Guillou, Gwenolé Guillou, and Soazig Guillou. How to explain zero-knowledge protocols to your children. In *Conference on the Theory and Application of Cryptology*, pages 628–631. Springer, 1989.