NIST-PEC comments on the ZkpComRef 0.2

Luís T. A. N. Brandão, René Peralta, Angela Robinson

National Institute of Standards and Technology*

April 17, 2020, Gaithersburg USA

The Privacy Enhancing Cryptography (PEC) project at the National Institute of Standards and Technology (NIST) encourages the development of reference material about zero-knowledge proofs, such as that being developed in the scope of the ZKProof initiative.

This document provides comments on the ZKProof Community Reference (ZkpComRef), version 0.2, publicly available on GitHub since December 31, 2019. This is in follow-up to:

(i) our comments (April 2019) on the initial ZKProof documentation; and

(ii) our contributions (Sep/Oct 2019) to advance the ZkpComRef from version 0.1 to version 0.2.

Index

1 Generic comment **2**

2 Development context **3**

3 New and revised comments **4**

3.1 On chapter 1 (Security) 4

F1.1. Clearer "Introduction" (Sec. 1.1) 4

F1.2. Terminology example (Sec. 1.2) . 4

F1.3. Statement representations (Sec. 1.3) 4

F1.4. Definition of Proof of knowledge 5

F1.5. Concurrency [old C8] 5

3.2 On chapter 2 (Paradigms) 5

F2.1. Clarify how a PCP works 5

F2.2. Explain the several paradigms . 5

3.3 On chapter 3 (Implementation) . . 5

F3.1. Backend choice NIZK-R1CS [old C17] 5

F3.2. Backends and frontends 6

F3.3. APIs and file formats 6

F3.4. Side-channels [old C20] 6

F3.5. Validation [old C21] 6

3.4 On chapter 4 (Applications) 6

F4.1. References on existing applications 6

F4.2. Illustrative diagram per application 6

F4.3. Shorter structured descriptions . 6

F4.4. More use-cases 6

3.5 On transversal editorial aspects . . 7

F5.1. Recommendations [based on old C2] 7

F5.2. Glossary [based on old C4] 7

F5.3. Examples [old C6] 7

F5.4. References [based on old C16] 7

* Opinions expressed in this paper are from the authors and are not to be construed as official or as views of the U.S. Department of Commerce. Certain commercial entities, equipment, or materials may be identified in this document in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by NIST, nor is it intended to imply that the entities, materials, or equipment are necessarily the best available for the purpose.

1. Generic comment

The version 0.2 of the ZKProof Community Reference (ZkpComRef) [ZKP19a] consolidates a draft reference about zero-knowledge proof (ZKP) technology, congregating content contributed by the community. It is useful in highlighting various aspects relevant to the development of secure, practical and interoperable ZKP implementation and applications.

We find positive the emphasis placed on community engagement and openness, as expressed in the charter, in the expectations on intellectual property, and in the outlined editorial process. The process of congregating contributions from various stakeholders poses inherent challenges, but also opportunities for promoting consensus and credibility, and for paving a basis for possible standards.

The document could benefit from a major text revision towards more clarity and rigor of concepts, and a more precise description of options and concrete examples. It is important to keep aiming at a wide audience, to better enable that readers from different backgrounds can easily navigate the document, and find and understand the content pertinent to them. Ideally, various track-record experts in ZK proofs would first provide voluntary detailed feedback, *à la* peer reviewing. Additional contributors could then address the received comments, editing existing content and complementing with identified missing elements, such as some descriptions, examples, references and figures. It may be relevant that in a revision step the contributors be given the role to revise the text of at least a full section, to promote a more uniform and careful style in several places.

We start with five high-level comments/suggestions:

G1. Editorial. Per chapter of the ZkpComRef-0.2, promote that two (or more) volunteer ZKP-experts commit to carefully **read the chapter and provide detailed public feedback about editorial aspects and content**. The resulting comments, referring to the line numbers in the "Annotated Changes" version, can then be an additional basis for development of content by other contributors and integration by the editors. See Section 5.

G2. Chapter Security. Some rewriting could make the material more accessible to readers that have a general computer science background but no background on zero-knowledge. It would be useful for the introduction to explain ZKPs and ZKPs of knowledge in separate paragraphs. Examples to go with the explanations would be useful. See Section 1.

G3. Chapter Paradigms. The chapter presents a useful systematization / organization structure of different paradigms. However, it does not yet provide an easy intuition of how and why each paradigm works. It would be useful to provide intuitive explanations of how a technique can actually be achieved within each paradigm, preferably with complementary diagrams (for visual intuition), simple examples of instantiation, and relevant references. See Section 2.

G4. Chapter Implementations. The chapter is not entirely self contained, leaving the reader to locate details on a few external web links. It would be useful to include a few examples of backend and frontend options with perhaps some performance comparisons. See Section 3.

G5. Chapter Applications. Some application use-cases could be described more succinctly, while still adding one diagram for a visual interpretation. Widening the variety of use-cases may also enhance the motivation support for ZKPs. See Section 4.

2. Development context

Initial comments. In reply to a request for feedback from ZKProof in early 2019, we elaborated the "NIST comments on the initial ZKProof documentation" [PEC19a], dated April 6, 2019, with 22 comments (C1–C22) about the content of the initial documentation, and seven comments (D1–D7) promoting the compilation of a community reference. These comments were presented during the 2nd ZKProof workshop [ZKP19b] and served as a basis for some of the subsequent contributions.

From version 0.1 to 02. After the 2nd ZKProof Workshop there was a call for contributions, in response to which we produced the "NIST-PEC contributions to advance the draft ZKProof Community Reference from version 0.1 to 0.2" [PEC19b]. The initial version, dated September 10, 2019, was followed by subsequent corrections. The document elaborated contributions on seven of the initial comments (C5, C7, C9, C11, C18, C19 and C22). Other ten of those initial comments (C1, C3, D1–D7) were directly used as contributions integrated by the editors. External contributions also addressed six other initial comments (C10, C12–C16). Various mentioned comments/contributions were integrated in the ZkpComRef 0.2 (December 31, 2019). Table 1 enumerates the mentioned initial comments that were addressed. Further contributions were possible by other contributors.

Table 1: Old comments [PEC19a] that have been resolved or partially addressed in ZkpComRef-0.2

#	Title	Brief description
C1	Reference Document	Created a consolidated reference document
C3	Scope of Creative Commons License	Widened the license to cover the new reference
C5	Executive summary	Created one
C7	Proofs of knowledge	Explains PoK/PoM, but needs more formalization
C9	Transferability	Now discussed in §1.6.6 and §2.2
C10	Circuits vs. R1CS	There is now an intro description to R1CS
C11	Common vs. public	More clear distinction in some points
C12	Motivation	Improved intro of chapter Applications
C13	Gadgets	The table of gadgets was improved
C14	Interactivity vs. transferability	Now discussed in new chapter Interactivity
C15	Implicit scope of use-cases	Some updates in the text
C16	References	Added some references
C18	Computational security parameter	Propose parameters for benchmarking
C18	Statistical security	Explain the concept and propose parameters
C22	Intellectual property	Added expectations about disclosure and licensing

New comments on version 0.2, towards version 03. This document updates a list of comments, intending to serve as a basis for consideration of new contributions towards an improved version (0.3?) of the ZkpComRef. Section 3 presents the list of comments — one set per each chapter of the ZkpComRef, and then some additional editorial comments. Several comments are new; others are updates from still-applicable previous comments.

3. New and revised comments

3.1. On chapter 1 (Security)

The chapter contains the most relevant concepts before implementation. The chapter can benefit from a re-write more considerate of a reader not yet acquainted with zero-knowledge. There should be paragraphs that intend to explain at an introductory level, and different paragraphs that intend to specify and/or explain at a detailed level. In general, consider revising which details can be left outside of the introductory explanations, and should be included in subsequent paragraphs that "specify" concepts in more detail.

F1.1. **Clearer "Introduction" (Sec. 1.1).** If a reader does not know about zero-knowledge already, part of the current description is difficult to follow. Some examples:

- **Secrecy from the point of view of the prover.** The meaning of "secret" and "secrecy" is fairly intuitive and usual when dealing with ZKPs (and cryptography in general). It may however get confusing when explaining that the secret can already be known to the verifier (is it then really a secret?). It is still important to make the point, since the security properties should hold regardless of apriori knowledge by the verifier. Consider improving the text wrt this.
- **Common (known both by the prover and verifier).** Version 0.2 of the ZkpComRef revised the text to use the word "commonly" was a way to address a needed distinction between what is "public" vs. what is "common" input (to both prover and verifier, e.g., as also used in the "C" of CRS). However, the the current use of "commonly known by the prover and verifier" may be confused with "usually" or "typically". Consider replacing by (or explaining that it means) "known both by the prover and verifier".
- **ZKP vs. ZKPoK.** After the initial explanation of zero-knowledge proofs, consider making more explicit the distinction between ZKPoK and ZKP. The distinction is not yet sufficiently clear throughout the document, so it is helpful to make it explicit from the start. There is some challenge (worth tackling) in writing such distinction in a clear way right in the intro. It could be considered while doing a careful revision of the text.
- **Types of requirements.** Section 1.1.2 proposes five "specification requirements" for a ZK proof system. However, only the first three (syntax; setup; algorithms) seem to be about actual specification of the proof system. Consider differentiating better those and the other two (security definitions; security analysis), which may be requirements for acceptability of use and/or of standardization, but are not about specifying the proof system taking place between a prover and a verifier.

F1.2. **Terminology example (Sec. 1.2).** Section 1.2 introduces important terminology (relation R, instance w, witness w, language L). It would be useful to start with an example of a zero-knowledge proof statement (maybe graph colorability). Then say an instance x is "a graph". A witness w is "a three-coloring of the graph". The language L is the set of three-colorable graphs. The relation R is the set of pairs $\{(x, w)$ where (x is an instance) and (w is a witness,) and (w is a three-coloring of x) $\}$.

F1.3. **Statement representations (Sec. 1.3).**

Consider improving the explanations of statement representations. Same portions could have simpler descriptions, visual examples, and a better justification of some restrictions applied to the definitions. For example:

- **Types of circuits.** In general a circuit may have several outputs, but the current description states that it is only allowed to have one output. This may be reasonable when in a clear context of verifying a Boolean predicate. But a reader may also be inclined to think in terms of proving/verifying that the input of a circuit (e.g., for multiplication) is equal to a certain output (e.g., composite number). Consider clarifying / justifying the context to make explicit why some restrictions may be being applied. Consider also adding a figure of a circuit to highlight the mentioned components.
- **R1CS.** Version 0.2 of the ZCRef improved by explaining R1CS. R1CS is not such a difficult concept, but Section 1.3.2 is somewhat difficult to read. Consider revising towards a simpler description. Also, as mentioned for the explanation of circuits, consider distinguishing a general definition of R1CS from particular choices/restrictions (e.g., large field) that are being tailored due to the subsequent steps towards succinct ZKPs. Consider also adding a concrete simple example (preferable within a figure) of translation between a Boolean circuit and an R1CS.

F1.4. **Definition of Proof of knowledge.** Consider completing Section 1.6.3, which currently has a placeholder ("To improve. A future version of this document should include here ...") a formal description of "proof of knowledge" — to be in style similar to the game-based definitions given for soundness and zero-knowledge.

F1.5. **Concurrency** [old C8]**.** Aspects of concurrency could be addressed more explicitly. Do the prover and verifier know in which session they are interacting? In Section 1.6, consider mentioning the need for session ids.

3.2. On chapter 2 (Paradigms)

F2.1. **Clarify how a PCP works.** While section 2.1 is focused a lot on PCPs (and that is okay), it could provide a stronger intuition on how they can be achieved. It talks about certain types of queries, but a reader outside of the area might not understand how is the proof string with respect to which these queries will be answered, or why this approach enables proving the validity of a proof. Some terms (e.g., MA) are used but not explained. Consider adding a sub-section whose goal is to provide a sketch of how and why a PCP works. This can then also serve as a running example enabling the reader to appreciate the enhancements that are possible across various proof systems.

F2.2. **Explain the several paradigms.** The current Section 2.3 is a simple bulleted list identifying several distinct paradigms of how to achieve ZKPs. It would be useful to have one subsection (e.g., one page long) per paradigm, explaining a basis to understand the main techniques. It could be specially useful to have one diagram per technique, to enable a visual intuition of the protocol flow. Several relevant references could be added to each description.

3.3. On chapter 3 (Implementation)

F3.1. **Backend choice NIZK-R1CS** [old C17]**.** Consider providing more rationale for the choice of NIZK and R1CS. Section 3.2 could benefit from a comparative overview of the various low-level backend options for representing relations. Comparing the advantages and disadvantages of interactive vs. non-interactive, and of several representations (e.g., including arithmetic circuits), may open more room for future document contributions on the cases that have not yet been explored in the existing documentation.

F3.2. **Backends and frontends.** Sections 3.2 and 3.3 mention that numerous choices for backends and frontends exist, including many implementations, but the given references are too vague (previous chapter, external websites) and the reader is left too wonder about concrete examples. It would be useful to name a few concrete examples, so that the document is better self-contained and the reader does not need not rely on external links.

F3.3. **APIs and file formats.** Subsection 3.4 could benefit from being moved to its own section and ensuring the contents are self-contained. The goal of designing a file format for encoding R1CS and its assignments to promote interoperability is declared, and a preliminary design is mentioned on line 1414, but the result is not given. Perhaps the goal of this section could be modified to discuss API and file format considerations and a more general level.

F3.4. **Side-channels** [old C20]**.** Consider exemplifying conceivable cases where side-channels are problematic.

F3.5. **Validation** [old C21]**.** Consider including some discussion on testing and validation of implementations.

3.4. On chapter 4 (Applications)

F4.1. **References on existing applications.** Section 4.3 "Previous works" is proposing to include an overview of works and applications existing in the ZK world. This section needs expansion, and a short description for each of the several references. The section may be organized into a few subsections, each covering a type of application, and including an enumerated list of references. Each reference could get a short description (no more than 5 lines of text). Consider also moving to here the references mentioned in section 4.1. In contrast to the rest of the document, this section is a place where it is specifically useful to let the reader learn about existing work, with a corresponding citation that the interested reader can follow.

F4.2. **Illustrative diagram per application.** Chapter 4 discusses three examples at length, across sections 4.5, 4.6 and 4.7. For each of the three applications, consider adding a page sized figure, containing a diagram depicting the parties involved, the flow of information, and the requirements about said flow. Each figure should serve to: enable an initial intuition of the detailed explanation that follows; a come-back-to point for sanity check of the understanding that the reader gets after reading through the section.

F4.3. **Shorter structured descriptions.** Some of the descriptions are too long, namely section 4.5, remaining abstract for the most part. For each application (a section 4.x), consider handling separately two goals: (i) convey an idea of the capability brought by ZK to an application (with goals, roles, requirements, etc.); (ii) give a more-simplified but more-concrete example (possibly toy-example), showing the actual values, names, identifiers and relations, and their flow in an application, instead of always keeping it abstract (some service, some claim, some value, some commitment, ...). The current text conveys does attempt to show concrete use-cases, but we think their descriptions is still too difficult to grasp by a reader trying to gain a sense of how ZKPs can be used in practice.

F4.4. **More use-cases.** Without increasing the size of the chapter, consider which other application use-cases could be relevant to include in order to widen the motivation for ZKPs and facilitate the understanding of the ZKP capabilities. It may be beneficial to decrease the size of description of some of the currently present applications, in favor of (within the same overall space) describing a few more applications, overall covering a wider area of interest. With

more use-cases described, it may also become clear what are the basic concepts and tools (gadgets, etc.) that can form the basis to support a simple description of all (or most) use-cases. One example of application, with major privacy considerations, that recently became of obvious interest is *contact tracing*, where information exchanged in encounters, possibly including coordinates (time and geolocation), may enable measurements useful to determine a risk of infection during a pandemic. Within which time-frame are zero-knowledge proofs for deployment as an essential tool at play in this kind of applications?

3.5. On transversal editorial aspects

The following comments pertain to editorial aspects (adjustments and indexing) or to new generic content (e.g., recommendations and examples) that apply to all chapters.

F5.1. **Recommendations** **[based on old C2].** To highlight suggested and essential practices, consider enhancing the identifiability and organization (e.g., indexing) of "recommendations" throughout the document. The reference document could then add a "List of Recommendations" similarly to how it contains a "List of Tables". Note: the old comment also mentioned "requirements"; now we are simplifying the suggestion to focus on the useful starting step of identification of recommendations.

F5.2. **Glossary** **[based on old C4].** Consider making the glossary more comprehensive, listing all technical terms and providing corresponding links to where they are defined and/or used in the document. Revise some of the definitions in the glossary for better consistency with those given in the main text.

F5.3. **Examples** **[old C6].** For better accessibility to a broad audience, consider enhancing the document with indexed examples that illustrate concepts that may be unfamiliar to some target audience. Each example can be highlighted with a caption (e.g., "Example 5: ZK proof setup with a CRS with trapdoor"), an explanation (possibly an illustration) within a boxed environment, and a footnote identifying the included concepts (e.g., "setup, trapdoor, CRS, prover and verifier").

F5.4. **References** **[based on old C16].** While bearing in mind that the ZkpComRef is not positioned as a survey of all prior work on ZKPs, consider adding supporting bibliographic references in numerous places where the text mentions specific prior results, definitions, claims, etc. This should aim at being helpful to the reader that may want to fact-check and do further reading, as well as to ensure proper attribution.

References

[PEC19a] L.B., R.P., A.R. *NIST comments on the initial ZKProof documentation.* Apr. 2019.

[PEC19b] L.B., R.P., A.R. *NIST-PEC contributions to advance the draft ZKProof Community Reference from version 0.1 to 0.2.* Sept. 2019. Revised on Oct. 2019.

[ZKP19a] ZKProof. *ZKproof Community Reference. Version 0.2 (Draft).* Ed. by D. Benarroch, L. T. A. N. Brandão, and E. Tromer. Pub. by *zkproof.org*, Dec. 2019. Check latest version at https://github.com/zkpstandard/zkreference/.

[ZKP19b] ZKProof. *2nd ZKProof Workshop.* zkproof.org, Apr. 2019. Berkeley, USA.