Towards Version 1.0 of the ZKProof Community Reference

Daniel Benarroch (QEDIT), Luís Brandão (NIST), Eran Tromer (Columbia & TAU)
editors@zkproof.org

Presented at 3rd ZKProof Workshop - Home Edition
May 18, 2020

A quick Survey

https://bit.ly/2X4LSsQ

Outline

1. Alice in ZK-Wonderland
2. The Upbringing
3. Recent Changes
4. Editorial Process
5. Survey Results
6. Open Discussion

Goals

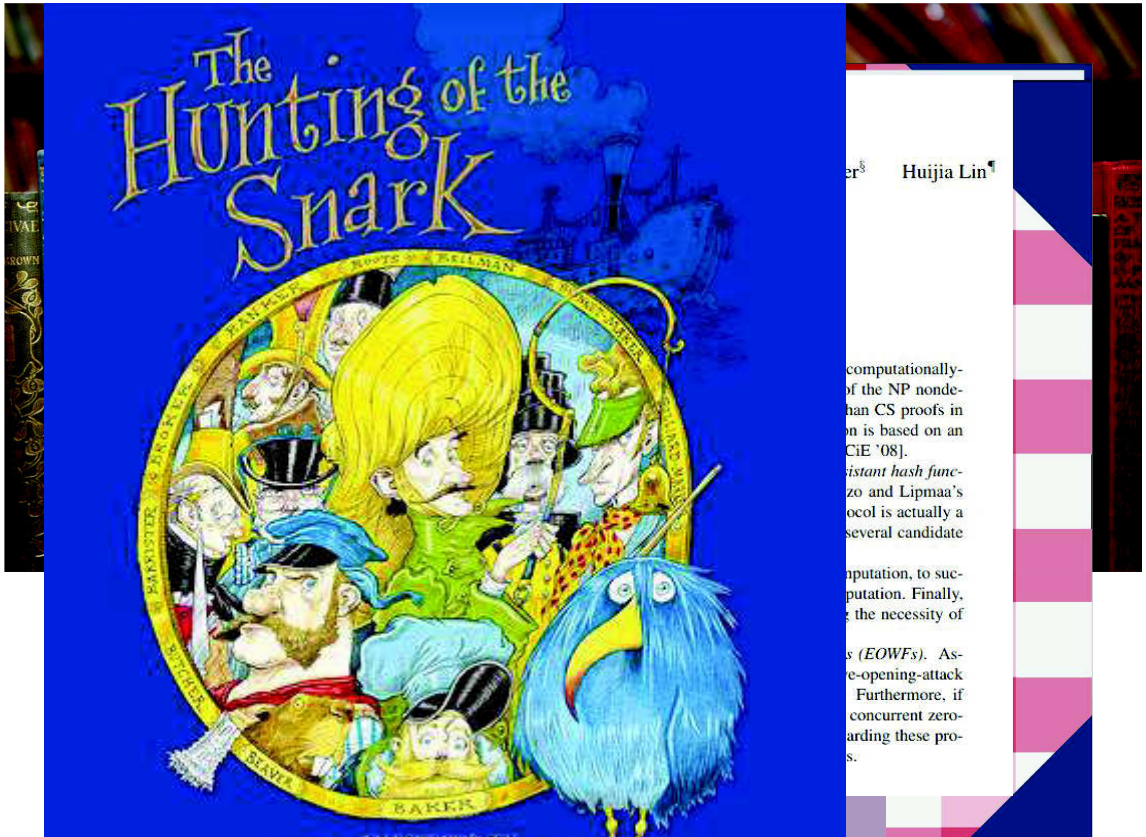
- ➔ **Present the ZKProof Community Reference**
- ➔ **Motivate collaboration for an improved version**

1. Alice in ZK-Wonderland

Disclaimer

"All characters appearing in this short story are fictitious. Any resemblance to real persons, living or dead, is purely coincidental"

Once upon a time



In a future world

Business

May 16th 2048 edition ›

NEW YORK

Eye of the hurricane

America Inc faces a wave of bankruptcies which failed to prove solvency in zero knowledge

But some firms will use their SMPC to prevent from going broke

Editor's note: The Economist *is making some of its most important coverage of the covid-19 pandemic freely available to readers of* The Economist Today, *our daily newsletter. To receive it, register here. For our coronavirus tracker and more coverage, see our hub*

"YOU WILL get business failures on a grand scale." So declared James Bullard, president of the Federal Reserve Bank of St Louis, on May 12th. Peter Orszag, a former official in Barack Obama's White House and now with Lazard, an

As she became older

THE KNOWLEDGE COMPLEXITY OF
INTERACTIVE PROOF SYSTEMS*

ASSER‡, SILVIO MICALI‡, AND

A Framework for Practical Universally
Composable Zero-Knowledge Protocols*

nocchio: Nearly Practical Verifiable Computation

ctor Shoup[3]

tzerland

Switzerland, and

niversity, USA

Bryan Parno
Jon Howell
Microsoft Research

Craig Gentry
Mariana Raykova
IBM Research

dence in computations outsourced to
uld be able to *verify* the correctness
. To this end, we introduce Pinoc-
efficiently verifying general computa-
on cryptographic assumptions. With
reates a public evaluation key to de-
; this setup is proportional to evalu-
once. The worker then evaluates the
ular input and uses the evaluation key
correctness. The proof is only 288
computation performed or the size of
Anyone can use a public verification

We consider cryptograpl
combinatorial puzzle. We d
party, the verifier, that the
to the verifier. The questio
to the given puzzle, and (ii)

lon
in many
graphic
prover

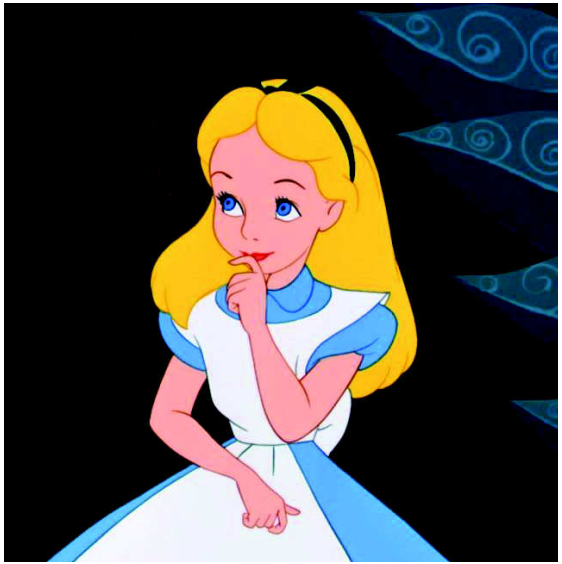
Computing [9–11] or other secure hardware [12–15] assume that physical protections cannot be defeated. Finally, the theory community has produced a number of beautiful, general-purpose protocols [16–23] that offer compelling asymptotics. In practice however, because they rely on complex Probabilistically Checkable Proofs (PCPs) [17] or fully-homomorphic encryption (FHE) [24], the performance is unacceptable – verifying small instances would take hundreds to trillions of years (§5.2). Very recent work [25–28] has improved these protocols considerably, but efficiency is still problematic, and the protocols lack features like public verification.

In contrast, we describe Pinocchio, a concrete system for efficiently verifying general computations while making only cryptographic assumptions. In particular, Pinocchio supports public verifiable computation [22, 29], which allows an in-

e logarithms and related pr
ot 2009, Camenisch, Kiayi
ch protocols, which allows
esigners just need to spec
t an efficient proof protoco
ation was in the CKY-lang
-called Σ-protocols, the re
of knowledge, which *not* re
when used as building blo
he Universal Composabilit
rotocols and, in particular
ists generic transformation
notion, these transformat

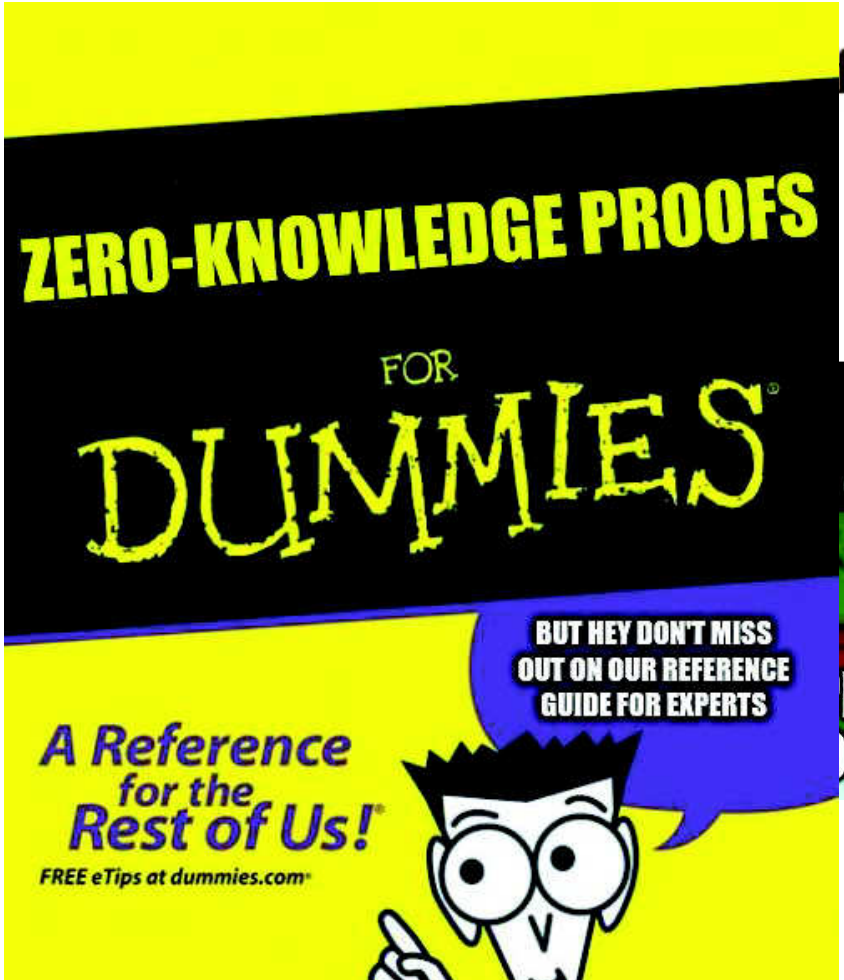
ie UC-framework by intro
a specification language akin to the CKY-language and a compiler such that protocols spec
our language are UC-secure and efficient. To this end we propose an extension of the UC-fra
addressing the problem that UC-secure zero-knowledge proofs are always proofs *of knowled*
state a special composition theorem which allows one to use the weaker – but more efficient an
sufficient – notion of proofs *of existence* in the UC-framework for the first time. We believe th
contributions enable the design of practical protocols that are UC-secure and thus themselves
used as building blocks.

All she wanted was

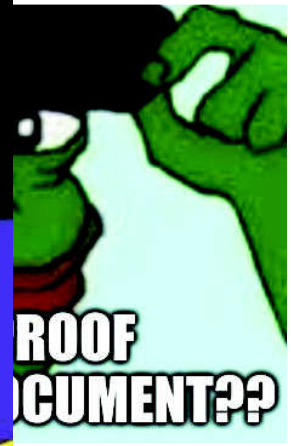


- ***Educational*** tutorial to enter the field
- Agreed-upon ***guidelines*** built around an inclusive ***community***
- ***Dynamic*** document to ensure up-to-date and able to update

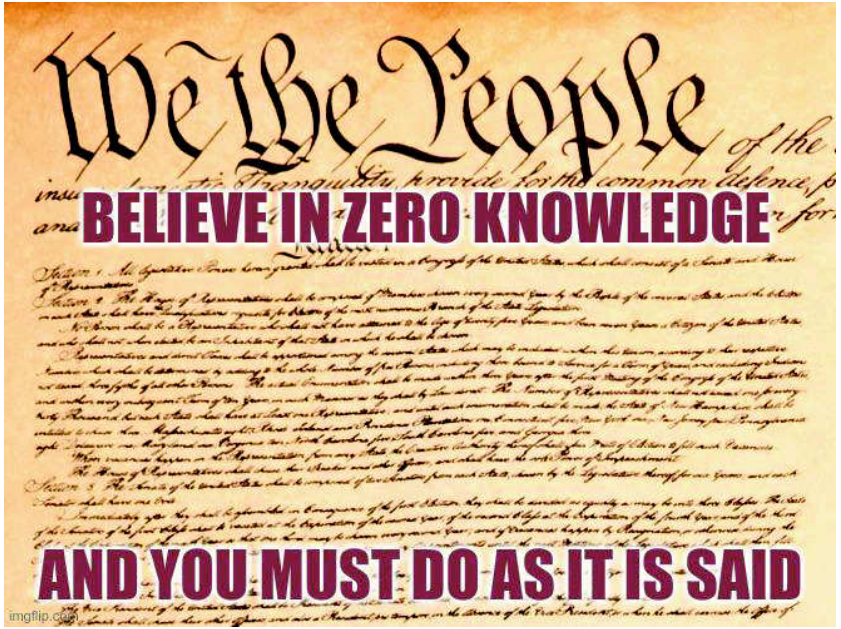
Mr Pepes' Tip



ference



An industry playground

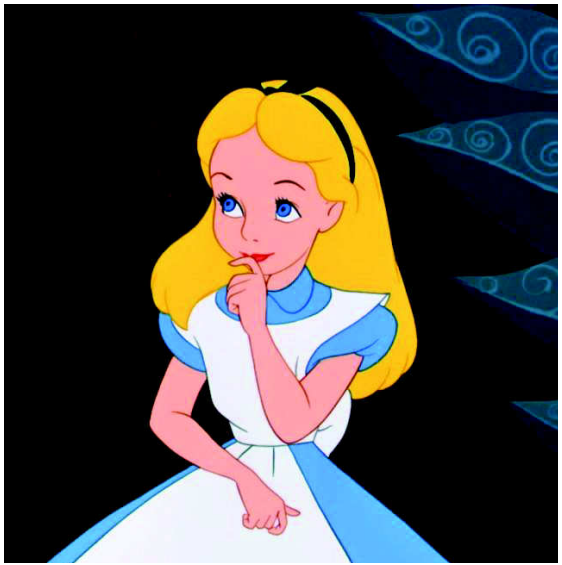


The sage in her

WIKIPEDIA
The Free Encyclopedia


The End
A ZKProof Reference Editors Production

Who are you?



CONSUMER

CONTRIBUTOR

Who are you?

CONSUMER

- ***Learner:*** onboarding, usage in applications, start research
- ***Mentor:*** professor / manager suggest material w/t guidance
- ***Evaluator:*** check a system is secure and state-of-the-art

CONTRIBUTOR

- ***Innovator:*** integrate own research / experience
- ***Sage:*** convey knowledge, recomm. & warnings

Disclaimer #2

WELL MAYBE NOT ALL THAT COINCIDENTAL

A quick Survey

https://bit.ly/2X4LSsQ

2. The Upbringing

The Upbringing

ZKProof Standards
Security Track Proceedings

ZKProof Standards
Implementation Track Proceedings

ZKProof Standards
Applications Track Proceedings

1 August 2018 + subsequent revisions

This document is an ongoing work in progress.
Feedback and contributions are encouraged.

Track Chairs:
Daniel Benarroch, Ran Canetti and Andrew Miller

Track Participants:
Shashank Agrawal, Tony Arcieri, Vipin Bharathan, Josh Cincinnati, Joshua Daniel, Anuj Das Gupta, Angelo De Caro, Michael Dixon, Maria Dubovitskaya, Nathan George, Brett Hemenway Falk, Hugo Krawczyk, Jason Law, Anna Lysyanskaya, Zaki Manian, Eduardo Morais, Neha Narula, Gavin Pacini, Jonathan Rouach, Kartheek Solipuram, Mayank Varia, Douglas Wikstrom and Aviv Zohar

ZKProof Community Reference

Version 0.1

(Draft 2019-04-11)

ZKProof Community Reference

Version 0.2

December 31, 2019

This document is an ongoing work.
Feedback and contributions are encouraged.
Find the latest version at https://zkproof.org.
Send your comments to editors@zkproof.org.

ZKPROOF

The Upbringing: over 70 people

Track chairs:
Jens Groth, Yael Kalai, Muthu Venkitasubramaniam

Track participants:
Nir Bitansky, Ran Canetti, Henry Corrigan-Gibbs, Shafi Goldwasser, Charanjit Jutla, Yuval Ishai, Rafail Ostrovsky, Omer Paneth, Tal Rabin, Maryana Raykova, Ron Rothblum, Alessandra Scafuro, Eran Tromer, Douglas Wikström

Track Chairs:
Daniel Benarroch, Ran Canetti and Andrew Miller

Track Participants:
Shashank Agrawal, Tony Arcieri, Vipin Bharathan, Josh Cincinnati, Joshua Daniel, Anuj Das Gupta, Angelo De Caro, Michael Dixon, Maria Dubovitskaya, Nathan George, Brett Hemenway Falk, Hugo Krawczyk, Jason Law, Anna Lysyanskaya, Zaki Manian, Eduardo Morais, Neha Narula, Gavin Pacini, Jonathan Rouach, Kartheek Solipuram, Mayank Varia, Douglas Wikstrom and Aviv Zohar

Track chairs:
Sean Bowe, Kobi Gurkan, Eran Tromer

Track participants:
Benedikt Bünz, Konstantinos Chalkias, Daniel Genkin, Jack Grigg, Daira Hopwood, Jason Law, Andrew Poelstra, abhi shelat, Muthu Venkitasubramaniam, Madars Virza, Riad S. Wahby, Pieter Wuille

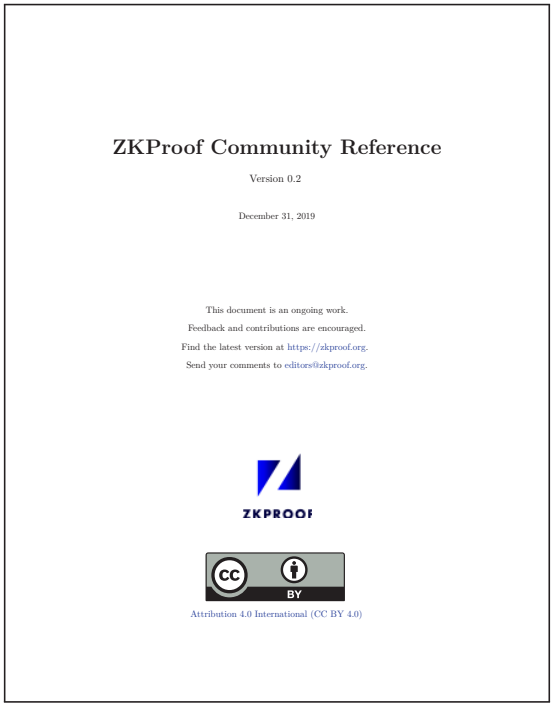
Contributors 0.1 → 0.2:
Daniel Benarroch, Luís Brandão, Yu Hang, Eduardo Morais, René Peralta, Angela Robinson, Justin Thaler, Eran Tromer, Ivan Visconti, Riad Wahby, Yupeng Zhang.

And thanks to all who participated in the discussions at the 2nd ZKProof Workshop!!

3. Recent Changes

Outline 2

New in ZkpComRef 0.2



ZKProof Community Reference

Version 0.2

December 31, 2019

This document is an ongoing work.
Feedback and contributions are encouraged.
Find the latest version at https://zkproof.org.
Send your comments to editors@zkproof.org.

ZKPROOF

https://github.com/zkpstandard/zkreference/

Some highlights:

- New chapter: Paradigms
- Some revised portions
- IP “expectations”
- Various editorial improvements

Goal: motivate next phase of changes

Why refining a ZkpComRef?

Isn't the basic knowledge already out there in papers?

- Value in community/collaborative result $\rightarrow$ credibility, referenceability
- A basis for seeking further consensus (future standards?)

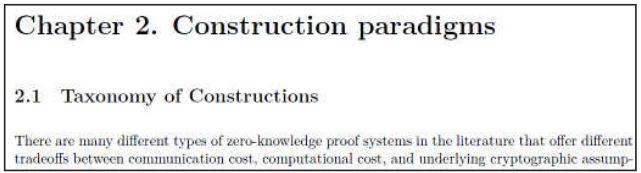
To achieve that, the ZkpComRef needs to evolve more!

It's a process. Since v0.1 we got:

1. Suggestions from review comments
2. Suggestions from 2nd ZKProof workshop (breakout sessions)
3. Diverse contributions

Paradigms (new chapter)

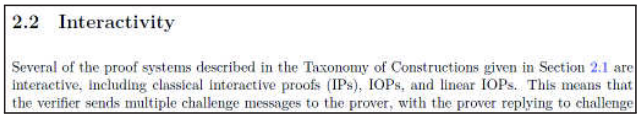
Goal: Give intuition on how to achieve ZKPs



Chapter 2. Construction paradigms

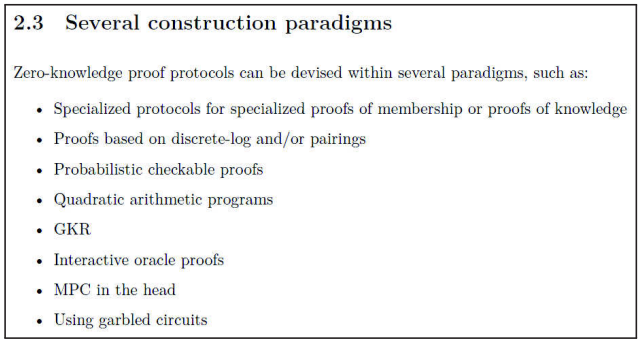
2.1 Taxonomy of Constructions

There are many different types of zero-knowledge proof systems in the literature that offer different tradeoffs between communication cost, computational cost, and underlying cryptographic assump-



2.2 Interactivity

Several of the proof systems described in the Taxonomy of Constructions given in Section 2.1 are interactive, including classical interactive proofs (IPs), IOPs, and linear IOPs. This means that the verifier sends multiple challenge messages to the prover, with the prover replying to challenge



2.3 Several construction paradigms

Zero-knowledge proof protocols can be devised within several paradigms, such as:

- Specialized protocols for specialized proofs of membership or proofs of knowledge
- Proofs based on discrete-log and/or pairings
- Probabilistic checkable proofs
- Quadratic arithmetic programs
- GKR
- Interactive oracle proofs
- MPC in the head
- Using garbled circuits

- "Paradigms" chapter
  - Old section "Taxonomy" (from v0.1)
  - New section "Interactivity" (following a 2nd workshop "breakout" session)
  - Missing sections: explain diverse approaches

ZK Proofs of Knowledge vs. of Membership?

A ZKP proves that a *statement* is truthful and reveals nothing else

But what kind of *statement*? (About a public *instance* x.)

- Statement of membership: $x \in L$
- Statement of knowledge: I known witness w such that $(x, w) \in R$

#	Elements / Scenarios	**Statement** being proven	**Instance** used as substrate	**Witness** treated as confidential
1	**Legal age for purchase**	I am an adult	Tamper-resistant identification chip	Birthdate and personal data (signed by a certification authority)
4	**Chessboard configuration**	<This configuration> can be reached	The rules of Chess	A sequence of valid chess moves

Done, missing:

- Some clarification provided in v0.2 ...
- Still needs improvement, e.g., missing definition of proof of knowledge

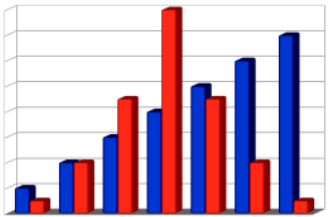
Security parameters for benchmarking

Version 0.2 recommends concrete parameters for benchmarking:

- **Computational:** 128 bits, 256 bits, ...
- **Statistical:** 64 bits, 40 bits, ...
- Compare efficiency & security for at least two parametrizations

Possible improvements:

- A few examples of concrete tradeoffs
- A graphic illustrating tradeoffs
- Concrete benchmark use-cases



Gadgets

Table 4.1: List of gadgets

#	Gadget name	English description of the initial gadget (before adding ZKP)	Table with examples
G1	Commitment	Envelope	Table 4.2
G2	Signatures	Signature authorization letter	Table 4.3
G3	Encryption	Envelope with a receiver stamp	Table 4.4
G4	Distributed decryption	Envelope with a receiver stamp that requires multiple people to open	Table 4.5
G5	Random function	Lottery machine	Table 4.6
G6	Set membership	Whitelist/blacklist	Table 4.7
G7	Mix-net	Ballot box	Table 4.8
G8	Generic circuits, TMs, or RAM programs	General calculations	Table 4.9

- **Done in v0.2:** Improved explanation of some gadgets.
- **Possible improvement:** Use the gadget analogies (e.g., commitment = sealed envelope) to devise a diagram for each application use-case

"Expectations" on IP disclosure and licensing

How it appeared?

- Review-comment about v0.1 → Contribution to v0.2
- Text inspired on NIST-ITL patent policy

Some goals:

- Prevent abuse of the ZKProof process
- Promote wide availability of ZKP technology
- Ease collaboration

Main concepts:

1. **Open source** implementations: encouraged
2. **IP disclosure**: strong encouragement / requirement when contributing
3. **IP licensing**: reasonable and non-discriminatory terms
4. **Improvable documentation**: creative-commons license (as per Charter)

IP expectations

ZKProof is an open initiative that seeks to promote the secure and interoperable use of zero-knowledge proofs. To foster open development and wide adoption, it is valuable to promote technologies with open-source implementations, unencumbered by royalty-bearing patents. However, some useful technologies may fall within the scope of patent claims. Since ZKProof seeks to represent the technology, research and community in an inclusive manner, it is valuable to set expectations about the disclosure of intellectual property and the handling of patent claims.

The members of the ZKProof community are hereby strongly encouraged to provide information on known patent claims (their own and those from others) potentially applicable to the guidance, requirements, recommendations, proposals and examples provided in ZKProof documentation, including by disclosing known pending patent applications or any relevant unexpired patent. Particularly, such disclosure is promptly required from the patent holders, or those acting on their behalf, as a condition for providing content contributions to the "Community Reference" and to "Proposals" submitted to ZKProof for consideration by the community. The ZKProof documentation will be updated based on received disclosures about pertinent patent claims.

ZKProof aims to produce documents that are open for all and free to use. As such, the content produced for publication within the context of the ZKProof Standardization effort should be made available under a Creative Commons Attribution 4.0 International license. Furthermore, any technology that is promoted in said ZKProof documentation and that falls within patent claims should be made available under licensing terms that are reasonable, and demonstrably free of unfair discrimination, preferably allowing free open-source implementations.

Please email relevant information to editors@zkproof.org.

We'd like to know what you think.

All changes were annotated

https://github.com/zkpstandard/zkreference/

List of contribution topics:

C1: Implement editorial structural changes
C2: Set expectations on intellectual property disclosure
C3: Add an executive summary
C4: Clarify proofs of knowledge
C5: Explain the computational security parameter
C6: Clarify the public vs. non-public aspect of "common" in CRS enhancement
C7: Discuss transferability and deniability
C8: Explain the statistical security parameter
C9: Clarify the (implicit) scope of some use-cases
C10: Compare circuits vs. R1CS
C11: Add introduction to interactive zero-knowledge proofs
C12: Improve description of applications and predicates
C13: Improve motivation in the application chapter
C14: Improve the table of gadgets
C15: Include references in Application chapter

Example:

#	Item id	Location	**Contribution topic C5**: Explain the computational security parameter	Related	Incorporated changes	Edit id
39	C5.1	Chapter 2 ("Implementation"), mostly in Section 2.5.	– **Context:** Proposed in the item 18 of the "NIST comments on the initial ZKProof documentation" (April 06, 2019). – **Proposed contribution:** Add text about possible computational security parameters, and the different security properties they may apply to (e.g., soundness, ZK, short-term vs. long-term, ...). In section 2.5, replace occurrences of "120" by "128".	GI3	– **Contributors:** NIST-PEC team – **Changed:** See items below.	E44
40	C5.2	Section 1.5			Wrt to required (approximate) level of security, change 120 to 128	E45, E46
41	C5.3	Section 1.7.1			In benchmarks, characterize different security properties	E33
42	C5.4	Section 1.7.2			Computational security levels for benchmarks	E34, E35

1 **ZKProof Community Reference**

2 ~~Version 0.1~~ Version 0.2 E1: C1.2

3 ~~(Draft 2019-04-11)~~ December 31, 2019

4 ~~A compilation of documents available at~~ This document is an ongoing work.

5 Feedback and contributions are encouraged.

6 Find the latest version at https://zkproof.org. E2: C1.2

7 Send your comments to editors@zkproof.org.

Outline 3

Editorial process

Cycle 2020:

1. **Public reviews per chapter** (till early July)
2. **Public contributions per section** (till October'ish)
3. **Public integration — compile v0.3** (till Dec 31, by editors)

   We hope a good advance in 2020 get us closer to a version 1.0.

Cycle 2021: revise the process @ 4th ZKProof workshop

Public reviews per chapter

Starting today: gather volunteers ... contact us at editors@zkproof.org

Version in review:	https://github.com/zkpstandard/zkreference/raw/master/changes-v0.2-from-v0.1.pdf		
Chapter	**Reviewer 1**	**Reviewer 2**	**Reviewer 3**
1. Security			
2. Construction Paradigms			
3. Implementation			
4. Applications			

https://drive.google.com/open?id=14_W-6203M8J0kzaODt4kLU9wGu9L1Vom2xk1pGsSxkQ

- Intended reviewers: ZKP experts + students + industry ... anyone
- Wanted comments: high-level and detailed ... even "annoying" comments
- Refer to the "Annotated Changes version" with line numbers

More on review comments

- What is not clear?
- Missing content or explanations?
- Technical accuracy
- Use-cases of interest
- What **examples** would make concepts easier to understand?
- Where would **diagrams or illustrations** improve understanding?
- General text revision

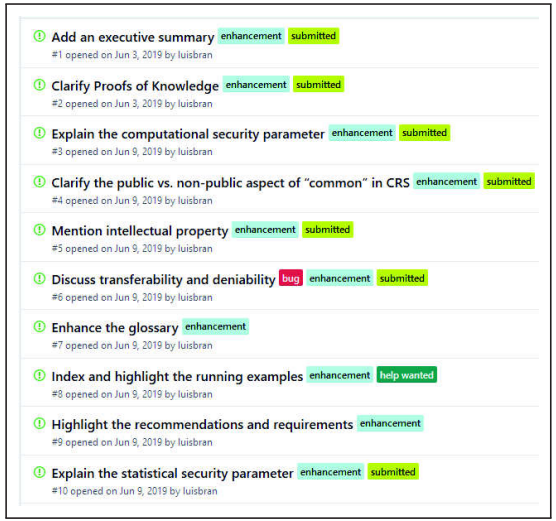
Example topics of review comments: NIST-PEC comments on the ZkpComRef 0.2

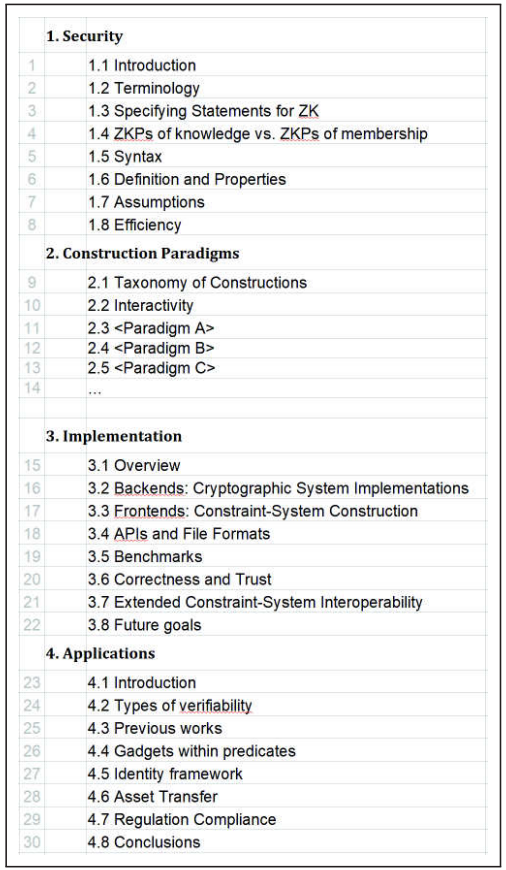
Public contributions per section

Editors will:

1. Organize a "call for contributions" (based on review comments)
2. Organize GitHub issues
3. Receive contributions per section

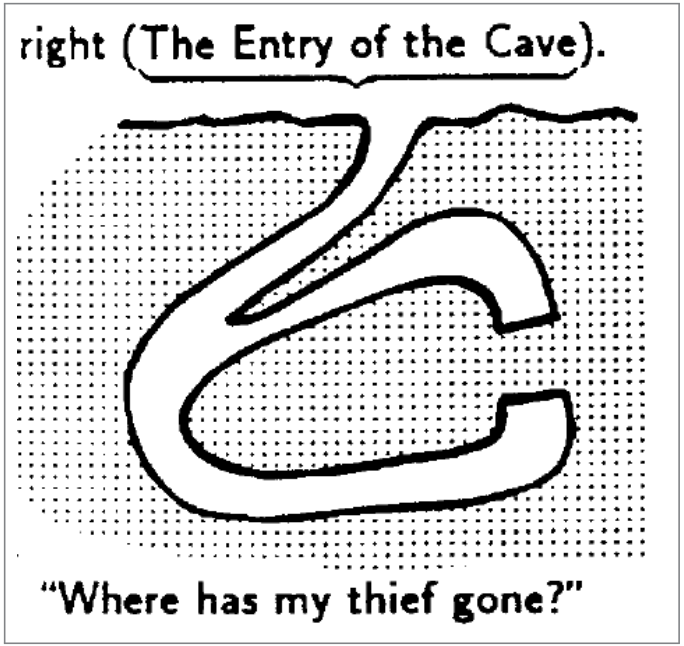
https://github.com/zkpstandard/zkreference/issues





Visual intuition: 1 image = 1000 words

One type of wanted contribution: figures, diagrams, ...



How to Explain Zero-Knowledge Protocols to Your Children
Quisquater et al. DOI:10.1007/0-387-34805-0_60

The ZkpComRef is missing about 20 figures/diagrams.

Can you pictorially describe, in one page, an approach for ZKP?

Consider suggesting or creating an illustration or diagram to explain a concept in the document

Integration by editors

The editors will integrate the contributions into the main document

We hope to get version 0.3 by the end of 2020

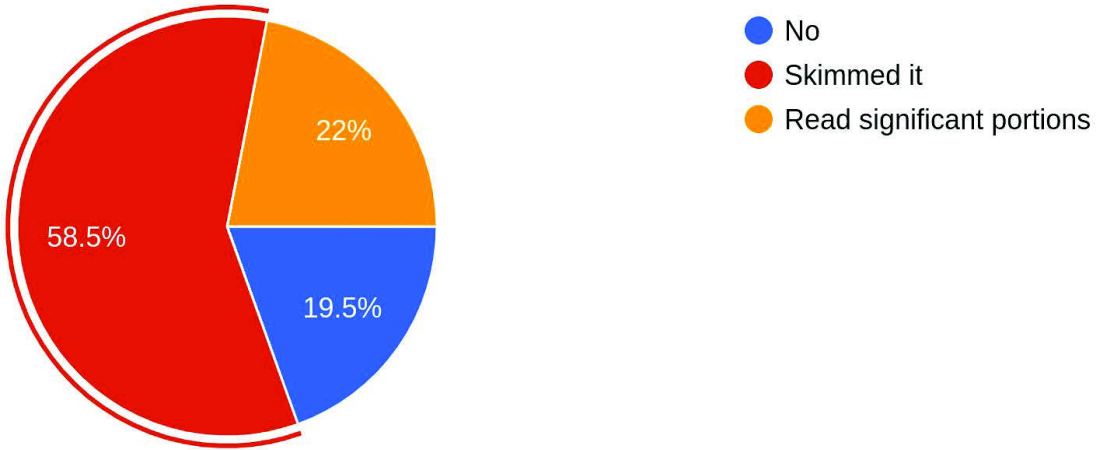
Thank you for the attention! → next section open discussion

Outline 4

Outline 5

5. Survey Results

1. Have you looked at the ZKProof Community Reference before today?



41 first participants by 18 May 2020

2. In what role do/would you use (or contribute to) the Community Reference?

Learner: I wish to learn about ZK, use it in an application, or get involved in research

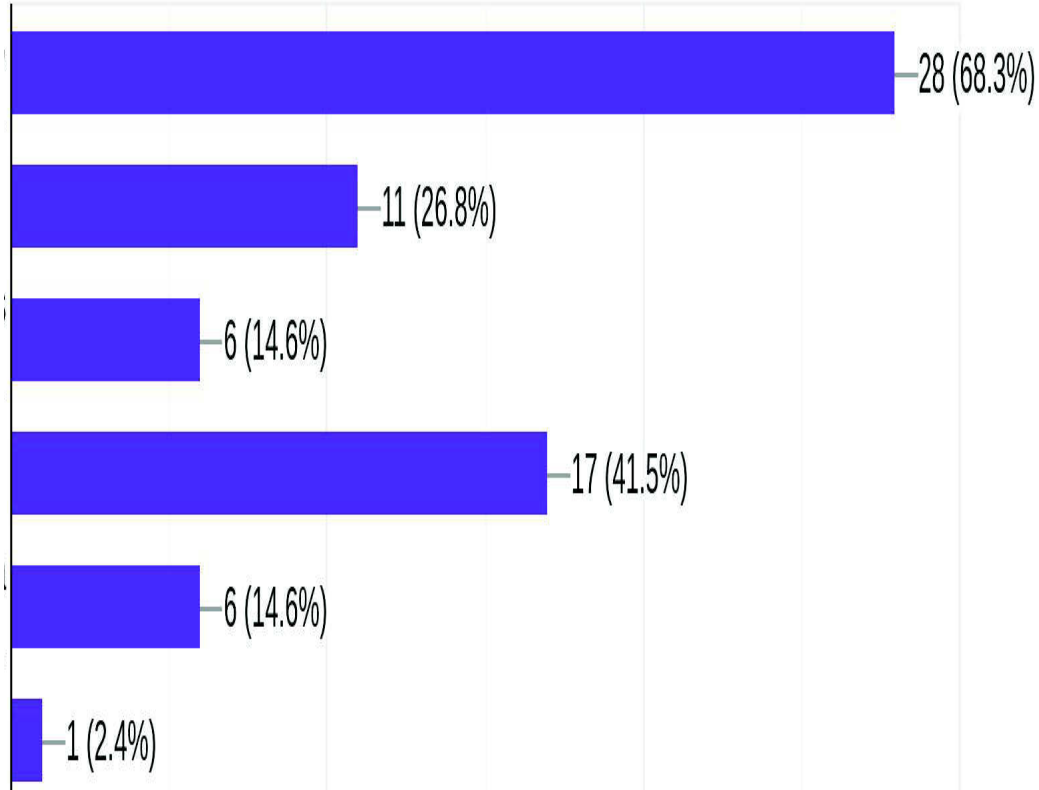
Mentor: I'm a professor/manager suggesting the reference as reading/reference material, with guidance and context

Sage: I'm an expert who wishes to convey knowledge, recommendations and warnings

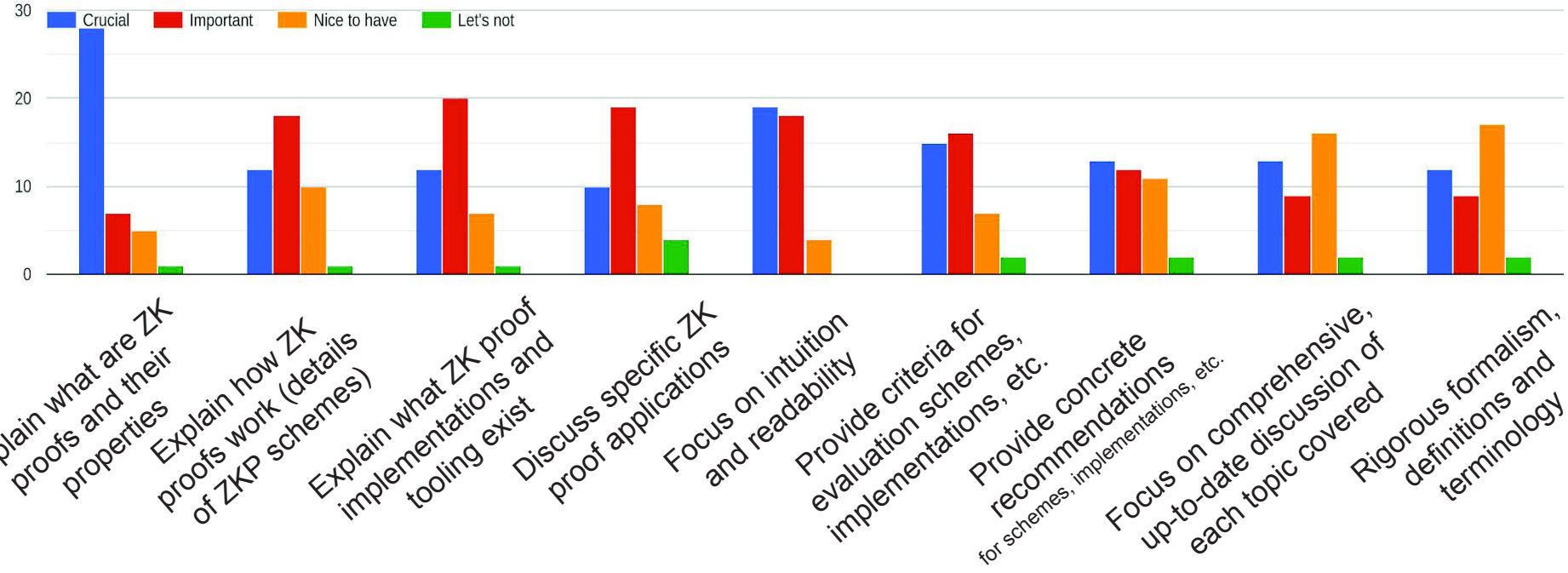
Innovator: I wish to integrate my own research/experience into the document

Evaluator: checks whether a given system is secure and state-of-the-art

Other



3. How important is it, for you, that the Community Reference will achieve each of the following?



4. What topics/aspects should be improved?

1. Schemes:
   a. Discuss polynomial commitments and ZKP schemes based on them *(x3)*
   b. MPC, “How ZKP and MPC work together'”
   c. Explain how ZK proofs work (details of ZKP schemes)
   d. Post-quantum lattice-based SNARG/Ks
2. More references for further reading *(x2)*
3. Survey of software libraries which have implemented ZKP schemes
4. Standards for all levels of the implementations stack
   Field/group API, ..., proof system API
5. Gadgets:
   a. Define gadgets and their composition in formal PL style
   b. What can be standardized in order to facilitate interoperability?

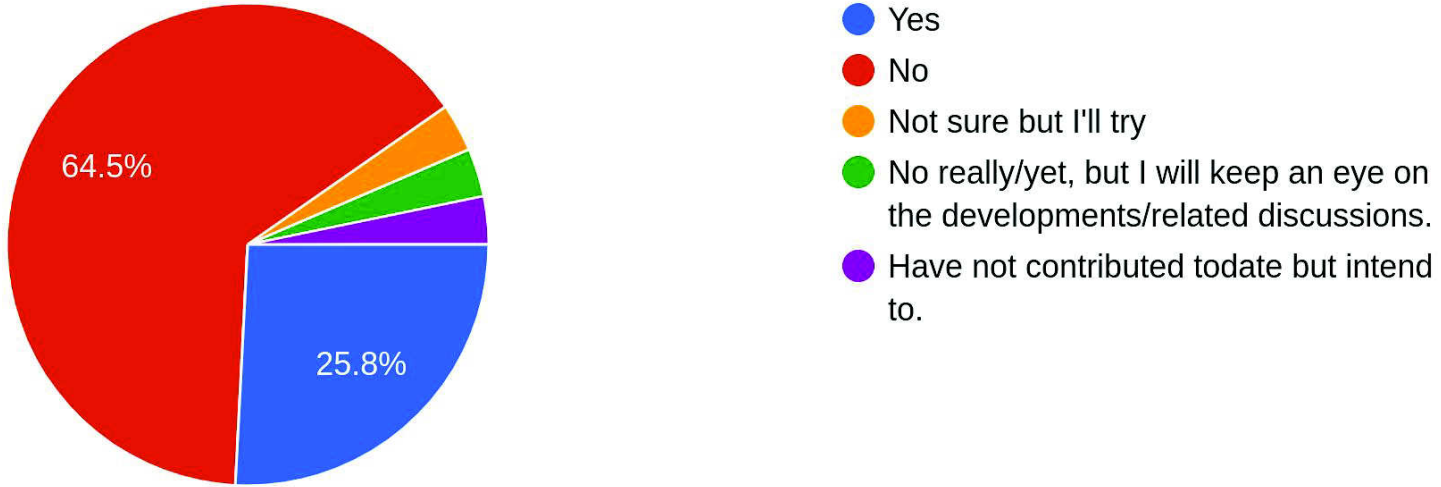
4. What topics/aspects should be improved? (cont.)

1. Cover ZK-friendly primitives
   inside/outside the constraint system?
2. Composability is crucial
   (should not be a single sub-sub-section of "advanced techniques")
3. Constraint systems:
   a. Improve description of R1CS *(x2)*
   b. Beyond R1CS
4. Explicitly explain what's not covered
5. More applications and explain how to use ZKP there *(x2)*
6. Interoperability /integrations contracts (?)
7. Amazing document already. Eventually looked for slightly more mathematical content; mostly definitions (in addition to the current intuition-based defs), say in 1.6, but also the protocols in 2.1. Chapter 3 presents the frontend/backend paradigm; I found Chapter 4 was somewhat ignoring this; being concerned with applications, I was expecting it to be mostly concerned with the construction of an IR, but as I read it further it wasn't so clear. Definition of predicate on p.47 is somewhat inconsistent with 1.2 Terminology, which says statement = relation + instance. Still unclear whether predicate is the same as relation. Are gadgets involved with the backend?

5. Which of the above can YOU improve?

1. Discussion of structured reference string and how to generate them. *(x2)*
2. Applications discussion *(x3)*
    a. e.g., coersion freeness
3. ZKP and MPC integration. Provide some real life examples for the same. *(x2)*
4. Post-quantum lattice-based SNARG/Ks
5. PL-style gadget definitions
6. Composability of ZKP
7. General review
8. Science communication / education / business development
9. Description of primitives used in Zcash

6. Do you know what to do in order to contribute to the Community Reference?



7. What existing parts did you like best?

1. The construction paradigms section particularly interesting.
2. I liked the section on transferable proof/deniability.
   It explain extremely well those subtleties.
3. All parts were excellent
4. General structure, high-level presentation (of definitions, security, etc.)
5. Gadgets discussion
6. Use cases
7. Security discussion *(x2)*
8. All was great, but I liked Chapter 1 and Chapter 3 best, Chapter 4 was helpful but felt a bit disconnected from the previous ones. Presentation of 3.4.2 R1CS File Format interesting, but isn't it a bit too committing, and what's its relationship with zkinterface?

8. Other comments?

1. Unclear what goes into a standardization and what not
2. The field is moving very quickly. This is both a challenge and an opportunity, i.e. a document that keeps pace with the rate of change and that strives to capture an unbiased consensus (to the extend possible) is extremely valuable.
3. The at home version worked well, would recommend in future holding another at home version. Thank you for organising and pulling this through in a challenging time.
4. Thank You for everything and please send me any info that i can lean from..Thank you
5. **It is very impressive. I regret not doing more up to now, but I am motivated to contribute!**

6. Open Discussion of the ZKProof Community Reference

Real-time discussion notes linked from

zkproof.org/workshop3-links