2nd ZKProof Workshop Notes

Interactive Zero Knowledge

<https://community.zkproof.org/t/breakout-session-interactive-zero-knowledge/>

https://community.zkproof.org/c/breakout-sessions

LaTeX: https://drive.google.com/drive/u/2/folders/1HWZYMH-6Mx8wcX8geium506L0KRxcgPe

Moderators: Yael Kalai and Justin Thaler

Scribes: Sean Coughlin

Notes Delivery Owner (post notes by April 19th):

Participants (as sign up sheet) -
https://docs.google.com/spreadsheets/d/1bVx3sekpR23wMigGTBuUa0s73LaNk5JI4UZPk2ru61A/edit?usp=sharing

General Notes

The goal of the session was to identify advantages and disadvantages of interactive zero-knowledge protocols, and, upon weighing them, identify settings or applications where interactive protocols may be particularly suitable or relevant.

Some relevant background: Zero-knowledge protocols are often constructed in a two-step process. First, an information-theoretically secure protocol is constructed (e.g., an interactive proof, an interactive oracle proof, a PCP, a linear PCP, etc. As their names suggest, interactive proofs and interactive oracle proofs are interactive, while PCPs and linear PCPs are not). Second, the information-theoretically secure protocol is “compiled” via cryptographic techniques into a zero-knowledge argument, which may or may not be interactive. See Yuval Ishai’s talk at this workshop for more information on this two-step approach to constructing zero-knowledge protocols.

Below is a list of advantages (or non-disadvantages) discussed:

*Efficiency, Simplicity, and Setup Assumptions: Interactive protocols can often be simpler than non-interactive ones, and may be more efficient as a result. They also may rely on simpler or weaker trusted setup assumptions (i.e., they typically do not require a structured reference string (SRS), the way many linear PCP-based arguments do.).

Yet, if an interactive protocol is public coin, it can be rendered non-interactive in most settings via the Fiat-Shamir heuristic (secure in the Random Oracle Model), often with little loss in efficiency. This means that protocol designers have the freedom to leverage interactivity as a “resource” to simplify protocol design, improve efficiency, or weaken or remove trusted setup, and still have the option of rendering the protocol non-interactive via Fiat-Shamir.

> A concrete example of how interactive protocols can be simpler and accordingly more efficient than non-interactive variants: researchers attempted to render PCPs practical (e.g., BCGT, STOC 2013), but the resulting PCPs were still complicated, and fell short of practicality. More recent work (SCI, STARKs, Aurora, etc.) turned to interactive oracle proofs rather than PCPs, leveraging interaction to achieve simpler and more efficient protocols.
>
> Space efficiency of the prover was also discussed. PCP- and linear-PCP (and even interactive oracle proof based arguments) often have large space requirements and require the prover to perform FFTs on big vectors (see efforts in the DIZK work that Howard talked about to distribute the prover due to these issues). Interactive proof based protocols currently seem more space efficient (e.g., GKR protocol works one circuit layer at a time, no FFTs), easier to distribute.

Note: in some specialized settings, the Fiat-Shamir heuristic is not known to be able to render a public-coin protocol totally non-interactive (see, e.g., https://eprint.iacr.org/2019/188)

*Interactive protocols can be based on weaker cryptographic assumptions than non-interactive ones (e.g., interactive proofs are information-theoretically secure, interactive oracle proofs can be compiled into interactive arguments based only on string commitments). In comparison, zkSNARKs are often based on knowledge assumptions.

*Many applications are inherently interactive (e.g., real-world networking protocols involve multiple messages just to initiate a connection). If an application is inherently interactive, why not leverage interaction as a resource to make a protocol simpler, more efficient, remove trusted setup, or use weaker crypto assumptions?

*Interactive protocols can potentially be run with fewer bits of security and hence be more efficient (adversaries may only have one try to break things in an interactive setting). Can also impose a time limit for the protocol to terminate, limiting the runtime of attackers, and thereby get away with fewer bits of security.

*Interactive protocols may be non-transferrable/deniable: the verifier cannot turn around and convince someone else of the validity of the statement. This can be essential in many applications. However, subtleties may arise if messages are signed by the prover (having the prover sign messages of an interactive protocol can make it transferrable).

*Zero-knowledge protocols are often combined with other cryptographic primitives in applications (e.g., oblivious transfer). If the other primitives are interactive, then the final cryptographic protocol will be interactive regardless of whether the zero-knowledge protocol is non-interactive.

Below is a list of disadvantages discussed:

*Currently, the zero-knowledge protocols with the shortest known proofs are based on linear PCPs, which are non-interactive. These proofs are just a few group elements. While (public-coin) zero-knowledge protocols based on interactive proofs or interactive oracle proofs can be rendered non-interactive with the Fiat-Shamir heuristic, they currently produce longer proofs (log or polylog field elements). The longer proofs may render these protocols unsuitable for some applications (e.g., public blockchain), but they may still be suitable for other applications (even related ones, like enterprise blockchain applications).

*Applying the Fiat-Shamir heuristic to an interactive protocol may increase soundness error.

*Network latency can make interactive protocols slow.

*Interactive protocols must occur online, i.e., the proof cannot simply be published or posted and checked later at the verifier's convenience.

*Also discussed was whether interactive protocols are more vulnerable to concurrency attacks on zero-knowledge (i.e., multiple malicious verifiers may interactive with a single prover and coordinate and interleave their messages to try to learn information from the prover).

*Many applications require non-interactivity.

*Because interactive protocols require the prover to send multiple messages, there may be more vulnerability to side channel or timing attacks (timing attacks will only affect zero-knowledge, not soundness, for public-coin protocols, since the verifier's messages are just random coins and timing attacks shouldn't leak information to the prover in this case. In private coin protocols, both zero-knowledge and soundness may be affected by these attacks).

Other topics that came up:

* Luis raised a possible error in the documents produced from the last workshop, regarding whether in public verifiable case, schemes where proofs are transferrable must be non-interactive. Luis will look into this.

*There was brief discussion about techniques other than Fiat-Shamir to remove interaction. Some exist but are not necessarily practical (e.g., Kalai, Paneth, Yang 2019).

*Verifiable Delay Functions

Suggested Contributions

A

Name	Email	Specific Contribution / Action Point
Ivan Visconti		Deniability
Luís T. A. N. Brandão		Statistical Security
Yupeng Zhang		Efficiency and trusted setup
Yael and Justin		Intro