G22.3220-001/G63.2180 Advanced Cryptography October 14, 2009

Lecture 6

Lecturer: Yevgeniy Dodis *Scribe: Bianca De Souza*

Topics to be covered in this lecture:

- Zero-knowledge proofs/arguments
- Challenge space and soundness of ZK of Σ-protocols
- Sequential and parallel compositions
- 4/5-round ZK argument/proof with commitments
- ZK argument with trapdoor commitments

Zero-knowledge proofs and arguments

Let's recall the definition of zero-knowledge (ZK).

Definition 1 *$P(x) \to V(y, aux)$ is a (auxiliary-input black-box) ZK with soundness s protocol for a NP relation R if:*

(a) *Completeness: for all $(x, y) \in R$ (i.e., $x \in L_R$), $\mathtt{Pr}(P \leftrightarrow V$ accepts$) = 1$*

(b) *Soundness: for all $y \notin L_R$ and any $P^*(\forall x', R(x', y) = 0)$, we have*
$\mathtt{Pr}(P^*(y) \leftrightarrow V(y, aux)$ accepts$) \leq s$
Note that for now, this holds even for unbounded P^* and we call ZK proof. *If only holds for* PPT*(*AI*)* P^* *it is called* argument.
If $s = \mathsf{negl}(\lambda)$, we omit s and simply say ZK.

(c) *Black-box auxiliary input (*AI*) ZK: there exists a* PPT *simulator Sim such that for all* PPT *V^*, $\mathtt{View}_{V^*}(P(x) \leftrightarrow V^*(y, aux)) \approx Sim^{V^*}(y, aux)$*[1]

Remarks:

1. Non-black-box (non-BB) is when for all V^* there exists unbounded Sim^* (we will not study this).
2. aux could contain any information about x (this ZK property is leakage-resilient).

[1] $\approx$ means computational indistinguishability called computational ZK ($\equiv$ is called perfect ZK and we will not cover it)

3. ZK implies two other properties: a) *deniability* that intuitively means once the prover has run the protocol to proof knowledge he or she cannot deny such a proof and *non-transferability* means that once the verifier has been convinced of the proof of knowledge by the prover, he or she cannot use the transcript to convince somebody else.

4. Typically, we want to insist on PPT prover P. Aside,

   Theorem 2 *If $L \in ZK \cap \mathcal{NK}$ has ZK proof with inefficient P then it has another ZK proof with a* PPT *prover P'.*

5. It is important that both P and V be probabilistic.

   - Result 1: If V is deterministic then $L \in$ BPP.
   - Result 2: If P is deterministic and π is AI-ZK then $L \in$ BPP.

6. In general, ZK implies WI (because *Sim* does not depend on witness).

7. HVZK is ZK where $V^* = V$.

ZK of Σ-protocols

Question: Are Σ-protocols ZK?

The good news is they satisfy (a) and (b) with soundness $s = 1/N$ where N is the challenge space. And the bad news is that it is hard to prove ZK if N is $\lambda^{\omega(1)}$.

Example: (Schnorr protocol)

$\mathbf{P}(\mathbf{x})$		$\mathbf{V}^*(\mathbf{y})$
$r \leftarrow \$$		
$a = g^r$		
	$\xrightarrow{a}$	
		$c^* = H(a)$
		where H is a hash function
	$\xleftarrow{c^*}$	
$z^* = r - c^* x$		
	$\xrightarrow{z^*}$	

How this can be simulated?

The job of *Sim* is to output the transcript (a, c^*, z^*) such that $c^* = H(a)$ and $g^{z^*} = a y^{c^*}$.

Choice 1: *Sim* chooses z^* and c^*, receives a and hopes that $H(a) = c^*$.

Choice 2: *Sim* chooses a and determine c^* which implies solving DL.

Either cases are very hard since the simulator has to break the hash function property or solve DL.

In fact, if H is ROM then any *Sim* must fail (else forge Fiat-Shamir-signature of empty message).

Challenge space and soundness of ZK Σ-protocols

Results:

1. (Dwork et all) If Σ-protocols with large challenge space N are ZK then Fiat-Shamir are insecure for any real H (and limit converse is true, as well).
2. (Goldreich-Krawczyk) If three-round protocol is ZK (with negligible soundness) then $L \in \mathsf{BPP}$.
3. If constant round public-coin protocol is ZK (with negligible soundness) then $L \in \mathsf{BPP}$.

So what is the essence of the problem? Answer: large challenge space N.

Theorem 3 *If Σ-protocols π has $N = poly(\lambda)$ then π is ZK with soundness $1/N$ (non-negligible)*

Proof: (Sketch) (a) and (b) are easy. For (c), we construct a simulator whose running-time is proportional to $N \cdot poly(\lambda)$.

Let *HSim* be a HVZK simulator:

$Sim(\mathbf{y}, \mathbf{aux})$		$\mathbf{V}^*(\mathbf{y}, \mathbf{aux})$
step 0: choose random tape r of V^*		
	$\xrightarrow{r}$	
let $t = N \cdot \lambda$ repeat for t steps		
samples $(a_i, c_i, z_i) \leftarrow HSim(y)$		
	$\xrightarrow{a_i}$	
	$\xleftarrow{c_i^*}$	
If $c_i = c_i^*$, stop and output $[r, (a_i, c_i, z_i)]$, else rewind V^* to first round and go to next iteration		

Intuitively, a_i gives no information about c_i because V^* generates c_i randomly(output of a hash function), so $\texttt{Pr}(c_i = c_i^*) = 1/N$. Hence, $\texttt{Pr}(Sim \text{ fails iteration}) \leq 1 - 1/N \Rightarrow \texttt{Pr}(Sim \text{ fails } N \cdot \lambda \text{ iterations}) \leq (1 - 1/N)^{N \cdot \lambda} \leq e^{-\lambda}$. If succeeds then this is the correct distribution. □

If we have Σ-protocol with large message space then we can choose small challenge space.

Now how to reduce soundness and have ZK with negligible soundness?
There are some techniques that allow us to do that:

1. Parallel repetition: run the same proof $\omega(\log \lambda)$ times.

   It reduces soundness and maintains the number of rounds but, unfortunately, it does not guarantee the ZK property [see GK' 90]. In this case, the verifier can learn all the first flows and use them to coordinate the challenges.

2. Sequential repetition: run the proof, wait until done and run again repeating the security parameter.

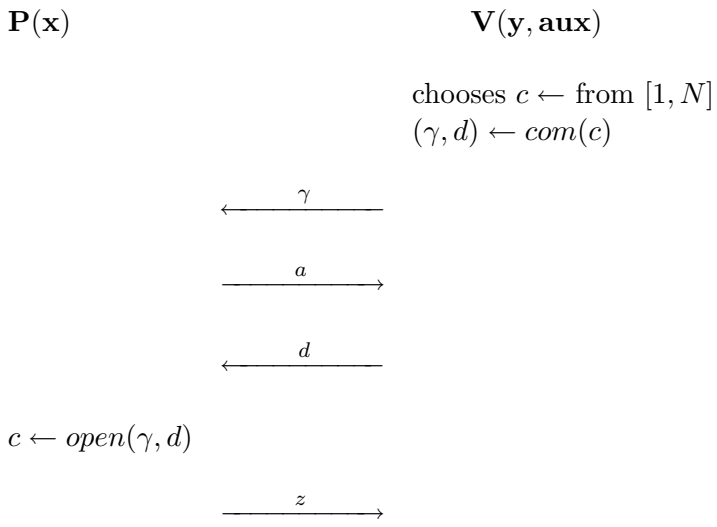
   The advantage of this is that soundness amplifies (s^λ). In this case, the verifier can not coordinate the second flow. Also this technique keeps the ZK property. Simply run one simulator after another with longer N and longer aux information (need AI-ZK).

   The bad aspect of this is that the number of rounds is $\omega(\log \lambda)$.

Constant-round negligible soundness ZK

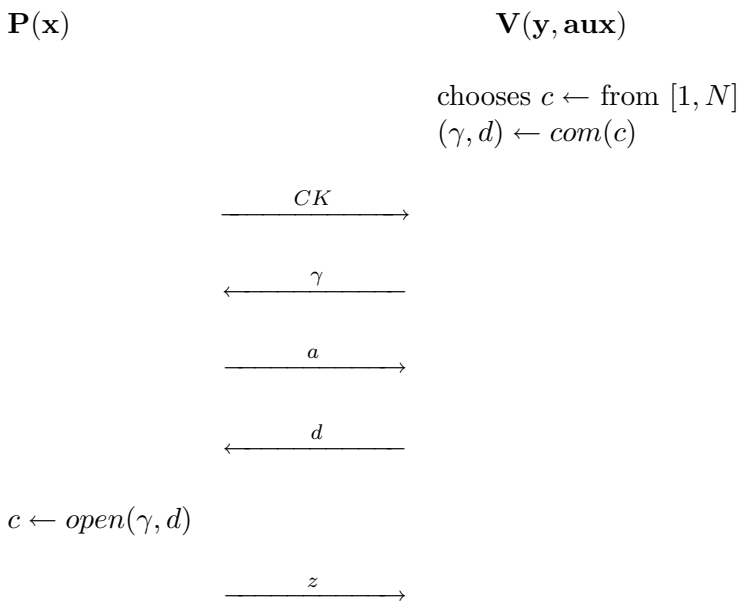
We will get 4 or 5 rounds.

The idea is to start with a Σ-protocol with large challenge space N but let V commit to c.



Note that commitments often need CK (commitment key).
We have three options:

1. Relax since definition of ZK assume trusted setup (note there is no trapdoors).
2. For computationally hiding commitments often there is no need for CK. No problem but get ZK argument.
3. For most commitment constructions (including ours from Σ-protocols) one can let receiver choose CK (optional). Thus, instead of four rounds, it has five rounds. This is shown below.



Punchline:

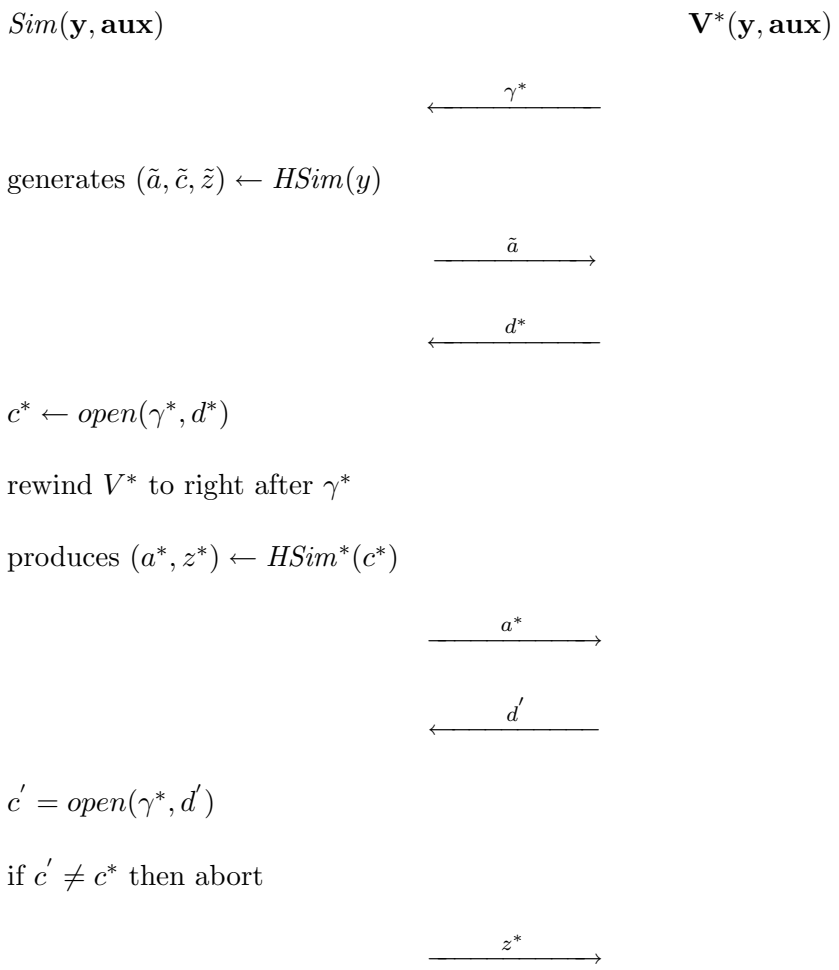
Trusted setup implies four round proof. And no setup implies either four round argument or five round proof.

Note that four round proofs are only possible for "easy" languages. [Katz, TCC' 08]

Theorem 4 *The above protocol is ZK with negligible soundness.*

Proof: For concreteness, let's just do trusted setup perfectly hiding commitment. The soundness is $s = 1/N$ since γ gives no information about c (inherits from Σ-protocol and hiding of commitment). For the ZK property,

Consider the initial simulator:



Simulator outputs [coins of V^*, $(\gamma^*, a^*, d^{'}, z^*)$]

For the ZK property, there are two ways for V^* to fool the simulator: a) opening γ^* to c^* first and then to $c^{'} \neq c^*$ but this is impossible by binding, and b) openning γ^* (to c^*) for $\tilde{a}$ but refuses to open for a^*.

Intuitively, if open first time with probability $\geq \epsilon$ then must open a second time with probability $\geq \epsilon$ (negligible) since $\tilde{a} \equiv a^*$.

The full *Sim* has to do two more steps:

1. it needs to estimate $\texttt{Pr}(V^* \text{ opens } \gamma^*)$ after it opens the first.
2. once known, it knows how long to repeat second step of the *Sim* initial

□

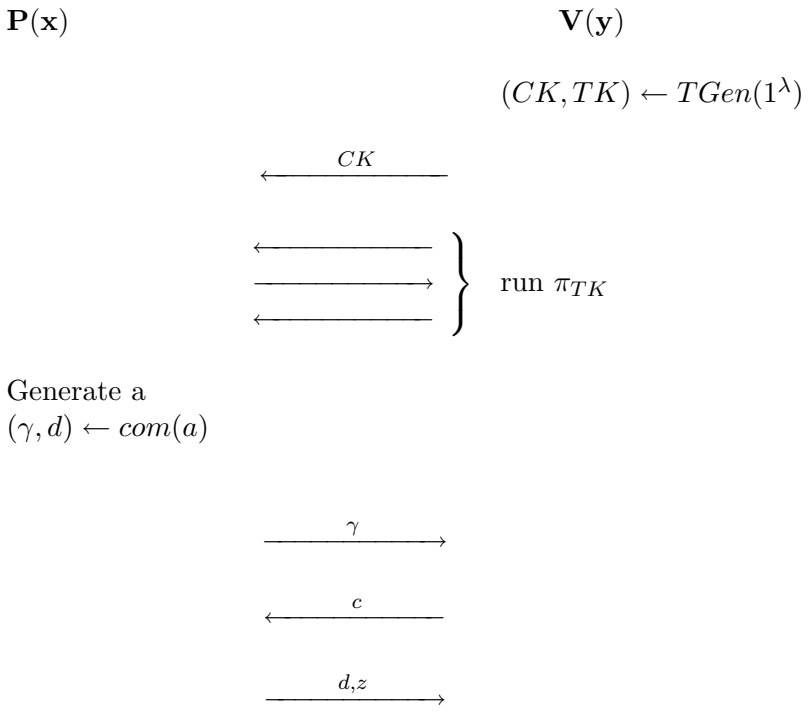
Alternative construction of ZK argument

We will do a complimentary idea which will later help us to build concurrent ZK.

Assume there exists a trapdoor commitment (TC) obtained from a Σ-protocol π_{TK} (Recall, $CK = \bar{y}$, $TK = \bar{x}$). Such TC's have the following property:

Property: There exists a Σ-protocol for proving knowledge of TK.

Construction of the protocol:



In this protocol, instead of V committing, P commits.

Theorem 5 *"For appropriate" Σ-protocol π_{TK}, the above protocol is ZK argument for L.*

Proof: We'll prove this next lecture.

The intuition is that the simulator extracts TK from V because V proves that it knows TK. Once the simulator figures it out, it commits to garbage and then when c is known . it knows how to open "garbage" to c.

This trick is going to be very useful for constructing concurrent ZK. □