Leakage-Resilient Public-Key Cryptography in the Bounded-Retrieval Model

Joël Alwen, Yevgeniy Dodis, Daniel Wichs

New York University

CRYPTO '09

Speaker: Daniel Wichs

Goal of Leakage-Resilient Crypto

- Crypto: schemes w/ proofs of security in *idealized model*.
  - Assume secret keys can be stored securely.
- Reality: schemes broken using “key-leakage attacks”.
  - Timing attacks, Power consumption attacks, Cold boot attacks
  - But also… Hackers, Malware, Viruses
- Usual Crypto Response: Not our problem.
  - Blame the Electrical Engineers, OS programmers…
- Leakage-Resilient Crypto: Let’s try to help.
  - Primitives that provably allow some leakage of secret key.
  - Assume leakage is arbitrary but incomplete.

Model of Leakage Resilience

- Adversary can learn *any efficiently computable function* $f : \{0,1\}^* \rightarrow \{0,1\}^L$ of the secret key. L = Leakage Bound.

- **Relative Leakage** [..., AGV09, DKL09, NS09].
  - "Standard" cryptosystem with small keys (e.g. 1,024 bits).
  - Leakage L is a large portion of key size. (e.g. 50% of key size).
- **Bounded Retrieval Model** [Dzi06, CLW06,...]
  - Leakage L is a parameter. Can be large. (e.g. few bits or many Gigabytes).
  - Increase sk size to allow L bits of leakage.
    (e.g. set |sk| = 2L).
  - System remains efficient as L grows.
    PK size, comm., comp. are independent of L.

sk

leak

50% of |sk|

Why have schemes in the BRM?

- Security against Hackers/Malware/Viruses:
  - Hacker/Malware/Virus downloads arbitrary information from compromised system, but bounded in length (< 10 GB).
    - Bandwidth too low, Cost too high, System security may detect.
  - Protect against such attacks by making secret key large (20 GB).
    - But everything else efficient.

- Security against side-channel attacks:
  - Adversary gets some "physical output" of computation (e.g. timing/power consumptions).
  - Many physical measurements => leakage can be large.
  - Still, may be reasonable to assume that it is bounded overall.
  - How "bounded" is it? Varies! (few Kb – few Mb).

Crypto Primitives with Leakage

- **Limitations** to leakage-resilience in **non-interactive** primitives.
  - Encryption Schemes: Leakage cannot depend on the ciphertext.
  - Existentially Unforgeable Signatures: Leakage must be smaller than signature size.
    - Impractical in BRM.
- Can have ***qualitatively stronger*** security with **interaction:**
  - Main goal: Authenticated Key Agreement.
    - Allows for interactive Encryption/Authentication.
  - Leakage before and after, but not during, protocol execution.

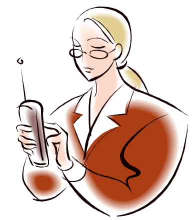
Private Communication

Non-interactive (Encryption)

pk_{Alice}

$Enc(m; pk_{Alice})$ →

(pk_{Alice}, sk_{Alice})



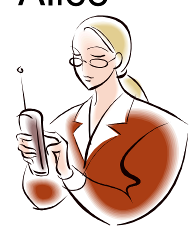
Prior to Communication | After Communication

Timeline: | Partial Leakage | No Leakage

AKA

pk_{Alice}

(pk_{Alice}, sk_{Alice})



Prior to Communication | Protocol Run | After Communication

Timeline: | Partial Leakage | No Leakage | Full Leakage

Two Goals

- GOAL 1(BRM): Schemes that allow arbitrarily large leakage bounds L, by increasing |sk|, but without increasing public key size, computation, communication.

- GOAL 2: Ensure privacy/authenticity of communication even if leakage occurs both before and after the communication takes place.

Prior Work

- Much prior and recent work on restricted classes of leakage functions [CDH+00, MR04, DP08, Pie08…].
  - Not applicable to e.g. hacking/malware attacks.
- **Relative Leakage.**
    - Symmetric-Key Authenticated Encryption [DKL09]
    - Public-Key Encryption [AGV09, NS09, KV09]
  - Problems: 1) non-BRM 2) no leakage after ciphertext.
- **Bounded Retrieval Model [Dzi06,CLW06].**
    - Symmetric-Key Identification [Dzi06]
    - Symmetric-Key Authenticated Key Agreement [Dzi06,CDD+07]
  - Main Problem: So far, only symmetric key.
    - Key distribution of huge keys is even more difficult in the BRM than usual.
- **This Work: Public-Key *Authenticated Key Agreement* in BRM.**

Roadmap of Construction

- Authenticated Key Agreement
  - Based on “Entropically Unforgeable Signatures”

- Entropcially Unforgeable Signatures
  - Based on “Identification Schemes”

- Identification Schemes:
  - Scheme 1: Relative Leakage
  - Scheme 2: “Direct product” extension to BRM
  - Scheme 3: Compressing Communication

Authenticated Key Agreement (AKA)

$(vk_{Bob}, sigk_{Bob}), vk_{Alice}$

$(vk_{Alice}, sigk_{Alice}), vk_{Bob}$

b

a

g^a

g^b, $Sign((g^a, g^b), sigk_{Bob})$

$Sign((g^a, g^b), sigk_{Alice})$

Key = g^{ab}

Key = g^{ab}

Entropically Unforgeable Signatures:

Adversary cannot forge signatures of random messages from high-entropy dist.

(even after leakage)

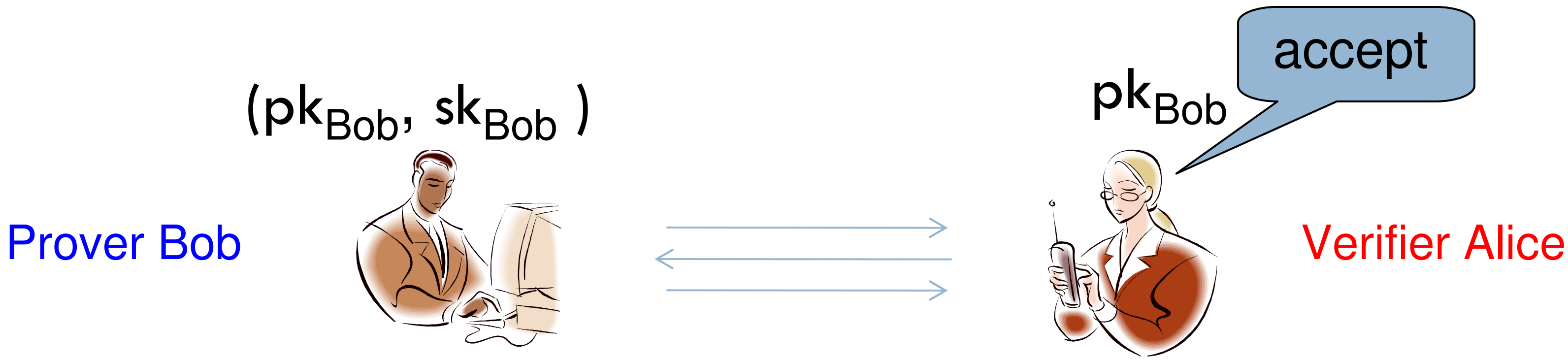
Roadmap of Construction

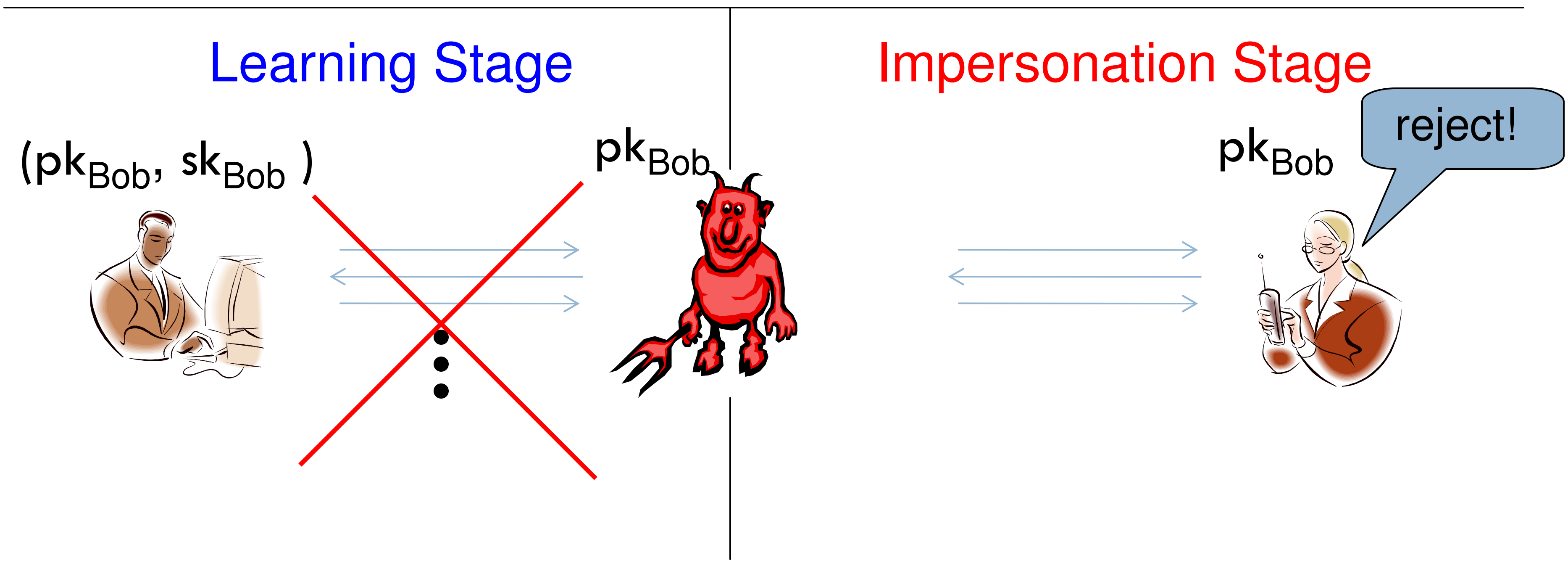
- ☑ Authenticated Key Agreement
  - Based on "Entropically Unforgeable Signatures"

- Entropcially Unforgeable Signatures
  - Based on "Identification Schemes"

- Identification Schemes:
  - Scheme 1: Relative Leakage
  - Scheme 2: "Direct product" extension to BRM
  - Scheme 3: Compressing Communication

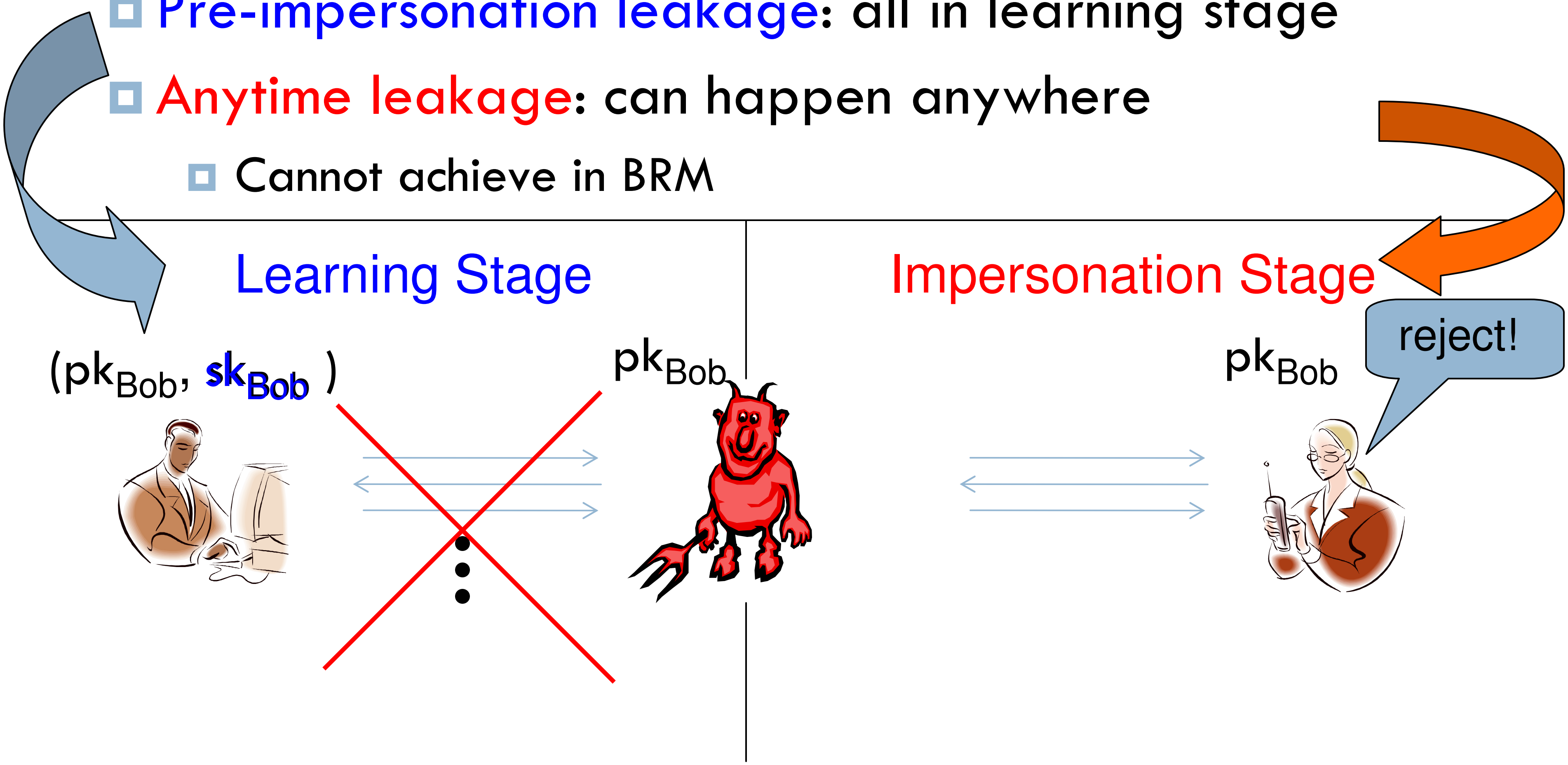
Definition: Identification Schemes



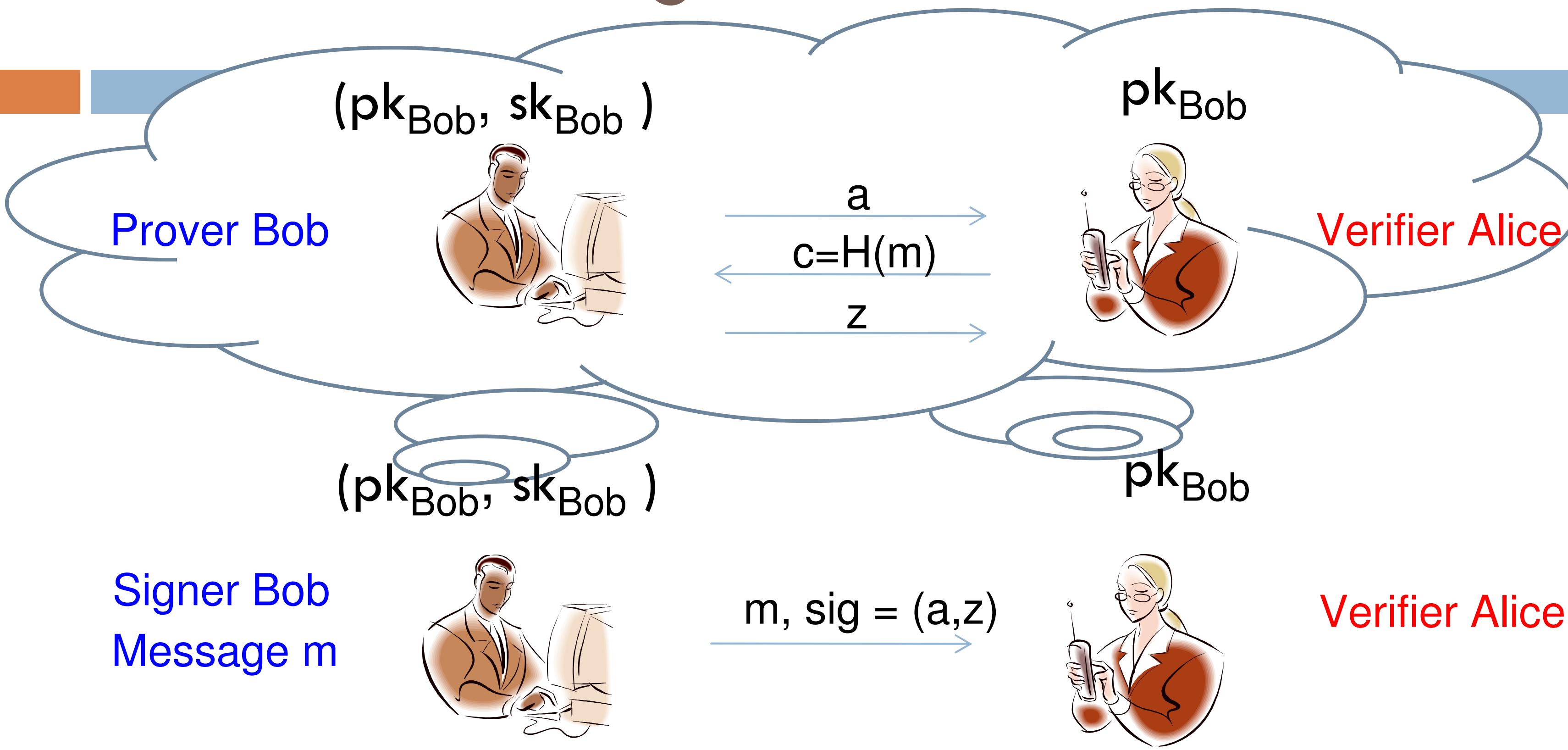


Leakage-Resilient Identification

- Bob's key can leak !!!
  - Pre-impersonation leakage: all in learning stage
  - Anytime leakage: can happen anywhere
    - Cannot achieve in BRM



Fiat-Shamir: Signatures from ID



- 3 round (public-coin) ID scheme => Signature.
  - Only works in the Random Oracle Model.

From ID to Signatures

- Theorem: Applying Fiat-Shamir
  - Anytime Leakage ID $\Rightarrow$ Existentially Unforgeable Sig.
  - Pre-imperson. Leakage ID $\Rightarrow$ Entropically Unforgeable Sig.
- Fiat-Shamir preserves leakage bound L, public/secret key sizes, communication, computation.
- New Goal: Construct efficient ID schemes with "pre-impersonation leakage" in the BRM.

Roadmap of Construction

- ☑ Authenticated Key Agreement
  - Based on "Entropically Unforgeable Signatures"
- ☑ Entropcially Unforgeable Signatures
  - Based on "Identification Schemes"
- Identification Schemes:
  - Scheme 1: Relative Leakage
  - Scheme 2: "Direct product" extension to BRM
  - Scheme 3: Compressing Communication

Okamoto's ID Scheme

Prover Bob | Verifier Alice

PK = $h = g_1^{x_1} \cdot g_2^{x_2}$,
SK = (x_1, x_2)

$a = g_1^{r_1} \cdot g_2^{r_2}$ →

c ←

$z_1 = r_1 - c \cdot x_1$, $z_2 = r_2 - c \cdot x_2$ →

PK

Check: $a = g_1^{z_1} \cdot g_2^{z_2} \cdot h^c$

- Properties of Protocol:
  - Many possible SK's (x_1, x_2) consistent with fixed PK h.
  - WI: proof *perfectly hides* which (x_1, x_2) is used.
  - Can extract a valid SK' = (x'_1, x'_2) from adv. prover.
  - DL ⇒ given one secret key, hard to find another.

Leakage Resilience of Okamoto ID

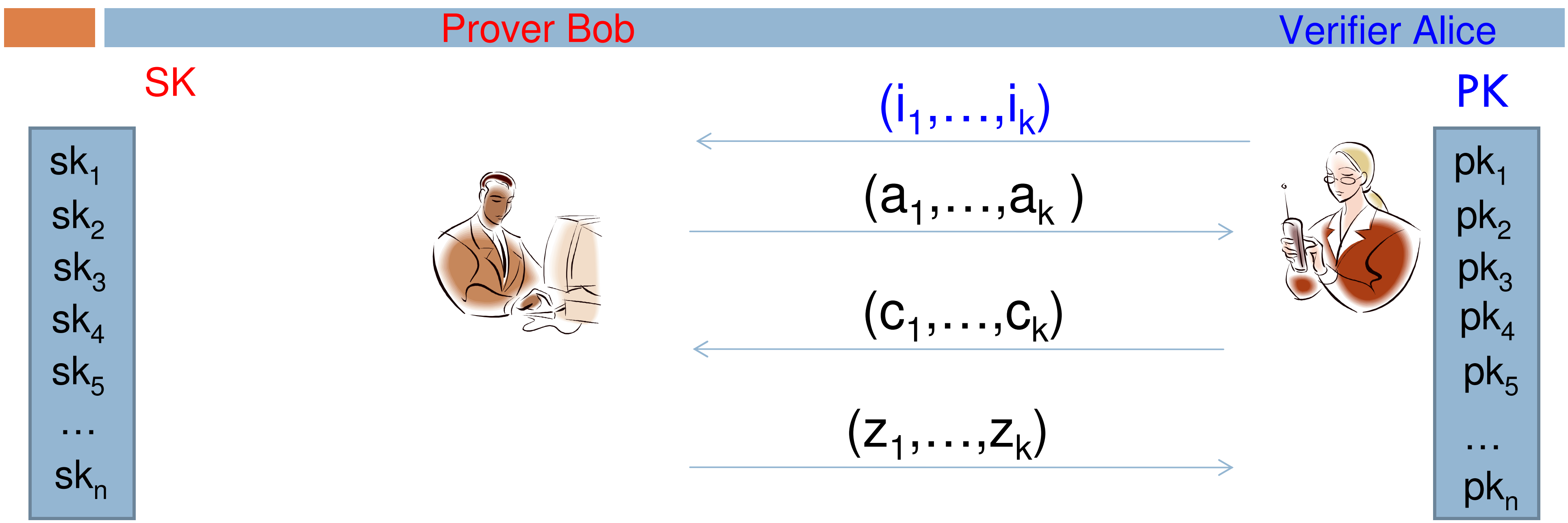
- Many possible SK's (x_1, x_2) consistent with fixed PK h.
  - $\Rightarrow$ Bob's SK has entropy, even if adv. gets PK+ "some" leakage.
- WI: proof *perfectly hides* which (x_1, x_2) is used.
  - $\Rightarrow$ "proofs" do not reduce entropy in SK.
- Can extract a valid SK' = (x'_1, x'_2) from adv. prover.
  - $\Rightarrow$ Adv. prover yields SK' $\neq$ SK.
- Contradict: DL $\Rightarrow$ given one secret key, hard to find another.

- Leakage:
  - As Is: Pre-imper. leakage $|SK|/2$, anytime leakage $|SK|/4$.
  - More generators: Pre-imper. $(1 - \varepsilon)\cdot|SK|$, anytime $(½ - \varepsilon)\cdot|SK|$.

Roadmap of Construction

- ☑ Authenticated Key Agreement
  - Based on “Entropically Unforgeable Signatures”
- ☑ Entropcially Unforgeable Signatures
  - Based on “Identification Schemes”
- Identification Schemes:
  - ☑ Scheme 1: Relative Leakage
  - Scheme 2: “Direct product” extension to BRM
  - Scheme 3: Compressing Communication

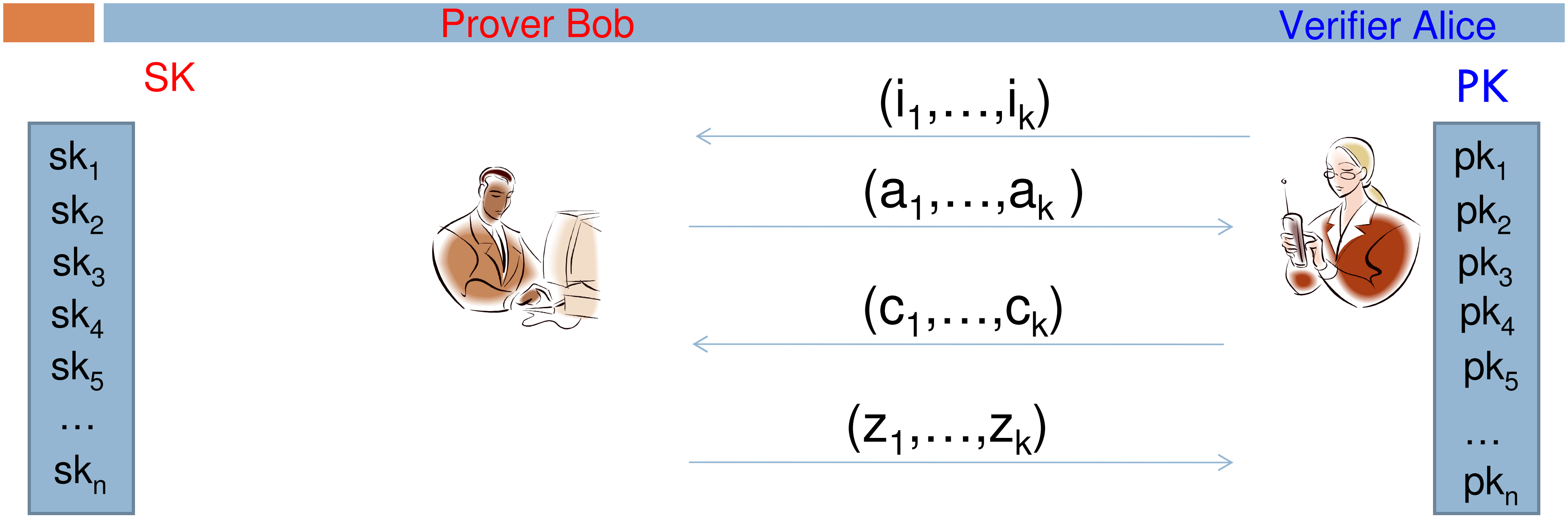
Direct-Product ID Scheme



- Bob's SK is a database of n Okamoto keys sk_i
- Alice chooses random k indices in $\{1,\ldots,n\}$.
- Alice and Bob execute k copies of Okamoto ID in parallel.
- Hope: Basic scheme allows L bits of pre-impersonation leakage

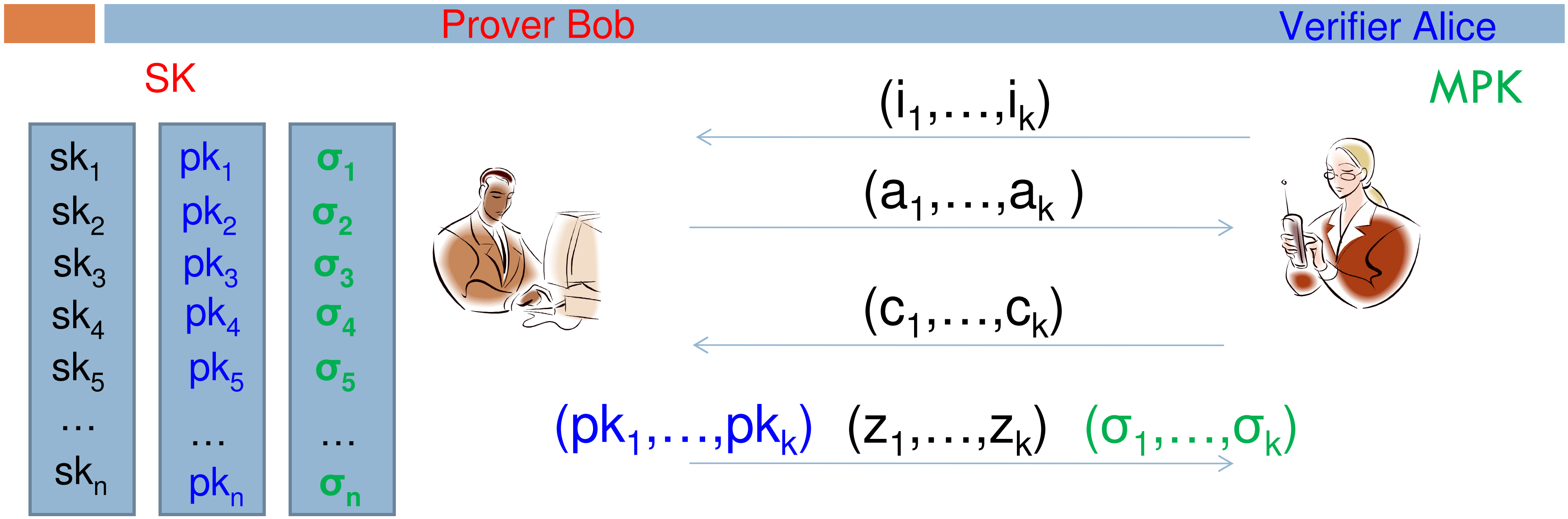
  => Direct-Product allows $\approx nL$ pre-impersonation leakage.

Direct-Product ID Scheme



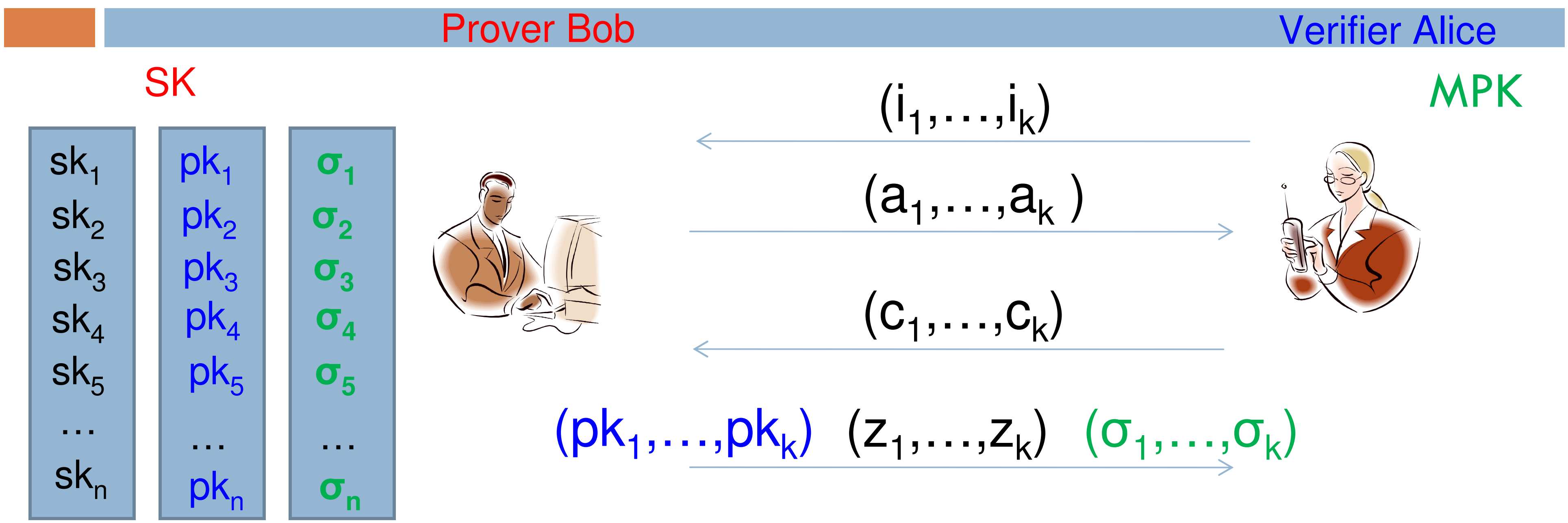
- Problem: Public-Key PK is huge!

Direct-Product ID Scheme



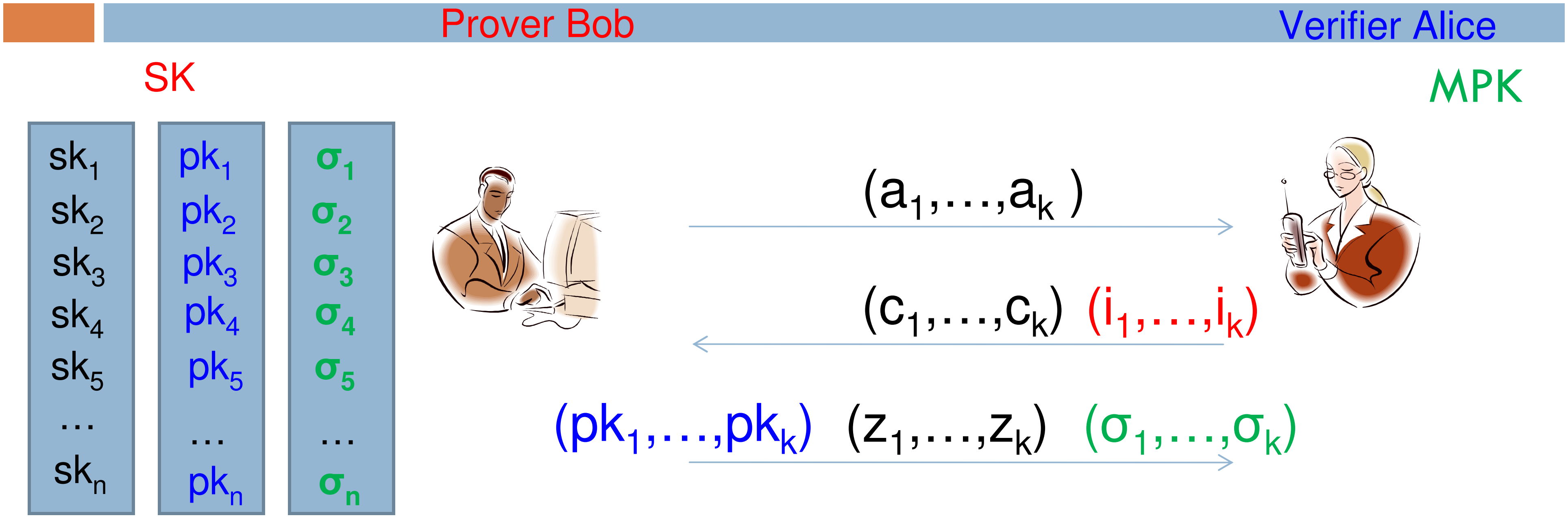
- Problem: Public-Key PK is huge!
- Solution: Bob stores all pk_i himself. Gives relevant keys to Alice during protocol execution.
- Bob signs individual public keys pk_i with a master signing key (which is deleted). Alice stores master verification key.

Direct-Product ID Scheme



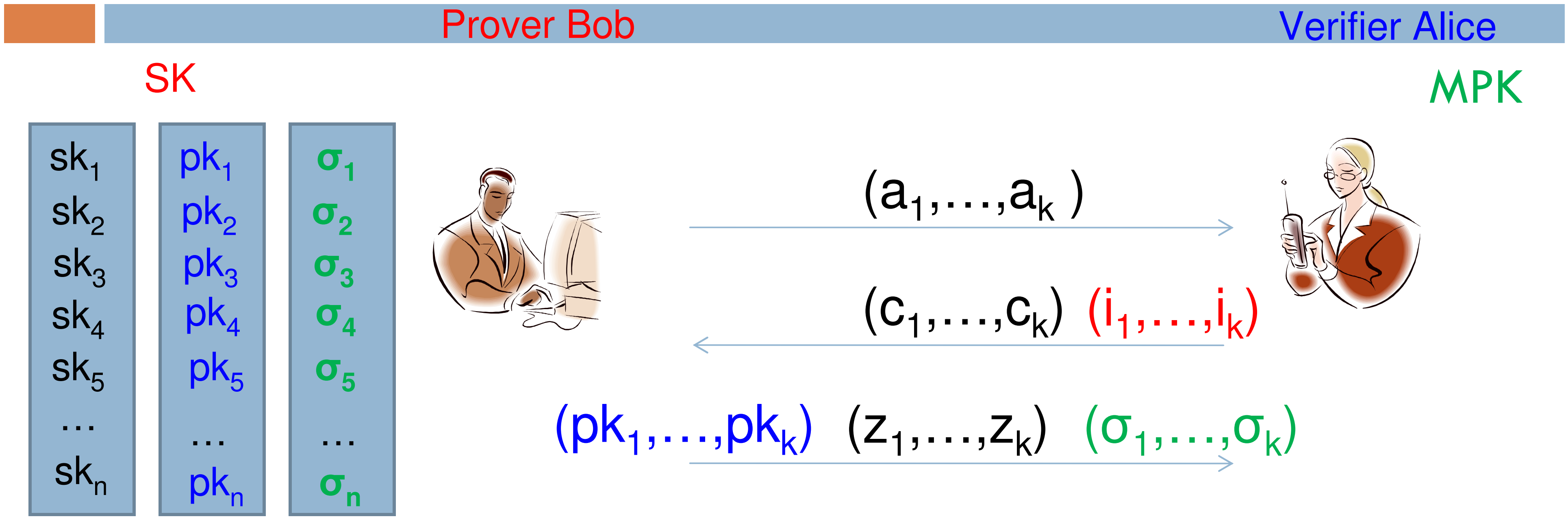
- <u>Problem:</u> 4 rounds instead of 3 (need 3 for Fiat-Shamir).

Direct-Product ID Scheme



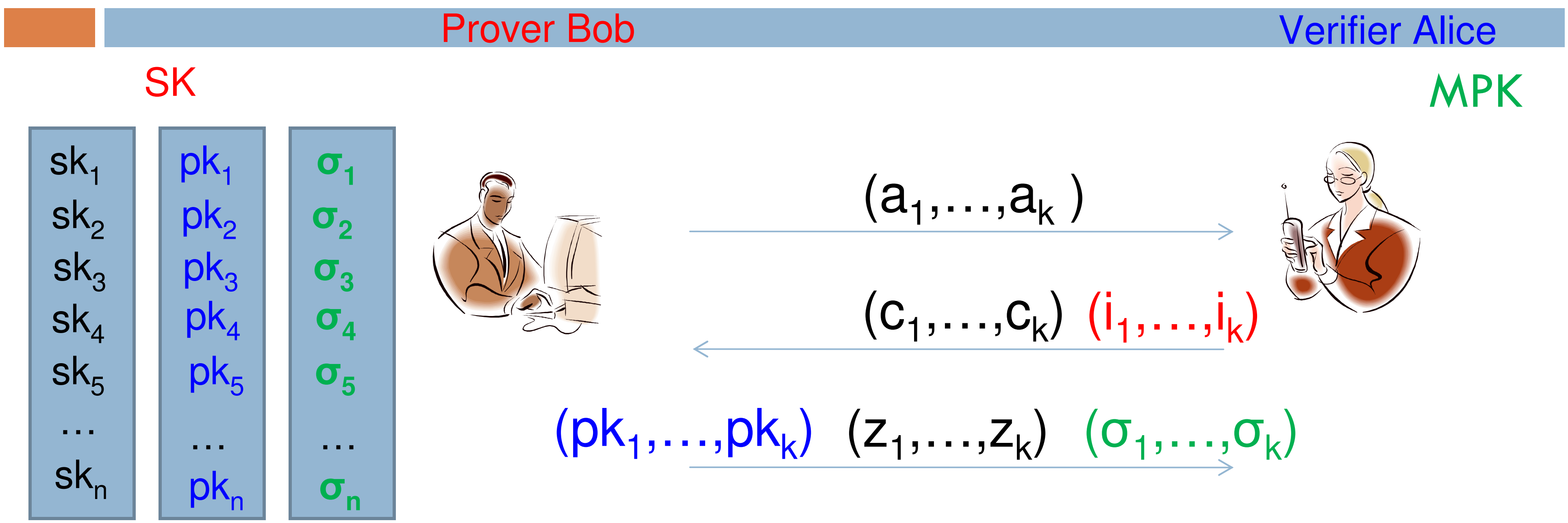
- Problem: 4 rounds instead of 3 (need 3 for Fiat-Shamir).
- Solution: Alice chooses indices during challenge round.
  - Okamoto has nice property that first round does not depend on pk.

Direct-Product ID Scheme



- <u>Question:</u> Can we prove that direct-product scheme allows n times as much leakage as small scheme?
- <u>Answer 1:</u> Interestingly, not in general. (counter-example)
- <u>Answer 2:</u> Works for Okamoto…

Direct-Product ID Scheme



- Efficiency Concern: Communication complexity has multiplied by k (essentially security parameter).

Roadmap of Construction

- ☑ Authenticated Key Agreement
  - Based on "Entropically Unforgeable Signatures"
- ☑ En
- Iden
  - ☑ Schem
  - ☑ Sch 2: "Direct product" extension to BRM
  - Scheme 3: Compressing Communication

No Time!
Bottom Line: communication is O(1) group elements, same as original Okamoto scheme.

Summary of Results

- Construct efficient ID schemes, entropic signatures, Authenticated Key Agreement protocols in BRM.
  - Secret key size: $L(1+\varepsilon)$. Leakage bound L.
  - Public key, Communication: constant # of group elements.
  - Data Accessed: O(sec. parameter) group elements.
  - Computation: O(sec. parameter) exponentiations.

- Existentially-UF sigs. with relative leakage of ½ of |sk|.
  - Independently discovered by [KV09]. Also possible without RO.

- Key Updates: Can "refresh" secret key to allow more leakage over the long-run.

- Future Work: Public-key encryption, IBE in BRM [ADN+ 09].

THANK YOU!

Questions?