Anamorphic Signatures

Secrecy From a Dictator Who Only Permits Authentication!

Giuseppe Persiano

Google + U. Salerno

Joint work with Miroslaw Kutyłowski, Duong Hieu Phan, Moti Yung, Marcin Zawada

Privacy as a Human Right

UDHR, Article 12: (1948)

No one shall be subjected to arbitrary interference with his privacy, family, home or **correspondence**,...

Privacy as a Human Right

UDHR, Article 12: (1948)

No one shall be subjected to arbitrary interference with his privacy, family, home or ***correspondence****,...*

End to End Encryption

- Cryptography has been very successful in providing tools for encrypting communication
  - The Signal protocol and app

The receiver-privacy assumption

Encryption guarantees message confidentiality only with respect to parties that do not have access to the receiver's private key

The receiver-privacy assumption

The receiver keeps his secret key in a private location

Receiver privacy

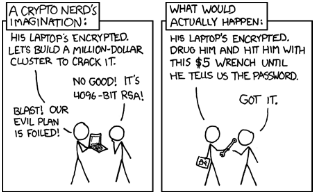
- Realistic for "normal" settings
- No wonder Encryption has been developed under these assumptions
    - with no explicit mention

Receiver privacy

- Realistic for "normal" settings
- No wonder Encryption has been developed under these assumptions
  - with no explicit mention
- In a dictatorship, instead

Receiver privacy

- Realistic for "normal" settings
- No wonder Encryption has been developed under these assumptions
  - with no explicit mention
- In a dictatorship, instead
  - No receiver privacy: citizens might be "*invited*" to surrender their private keys



Democracies attempt to regulate encryption

The Clipper Chip

Presently, anyone can obtain encryption devices for voice or data transmissions. [...] if criminals can use advanced encryption technology in their transmissions, electronic surveillance techniques could be rendered useless because of law enforcement's inability to decode the message.

Howard S. Dakoff
The Clipper Chip Proposal, J. Marshall L. Rev., 29, 1996.

Ban of E2E encryption

In our country, do we want to allow a means of communication between people which even in extremis, with a signed warrant from the Home Secretary personally, that we cannot read?

David Cameron
UK Prime Minister
January 2015

Crypto Wars

Arguments against restricting encryptions:

- *the bad guys can utilize other encryption systems*
- *all other encryption schemes must be declared illegal*
  - what qualifies as an encryption scheme? e.g., *chaffing and winnowing*
- *it creates a natural weak system security point*
  - Frankel and Yung showed how to frame legitimate users in the Clipper Chip [Crypto 95]

Crypto Wars

Arguments against restricting encryptions:

- *the bad guys can utilize other encryption systems*
- *all other encryption schemes must be declared illegal*
  - what qualifies as an encryption scheme? e.g., *chaffing and winnowing*
- *it creates a natural weak system security point*
  - Frankel and Yung showed how to frame legitimate users in the Clipper Chip [Crypto 95]

indirect and non-technical

A Dictator's Paradise

A thought experiment

- Let us have it the Dictator's way
  - Encryption still allowed
  - Users must surrender the secret keys associated with their public keys
- Did the dictator achieve their goal?
  - Access to all communication

Enter Anamorphic Encryption

[P, Phan, Yung – Eurocrypt 22]

The anamorphic approach

- A ciphertext is associated with two secret keys $\mathtt{sk}, \mathtt{dkey}$
- share $\mathtt{dkey}$ with your friend
- A ciphertext $\mathtt{ct}$ carries two plaintexts m_0, m_1, one for each key
  - $m_0 = \mathsf{Dec}(\mathtt{ct}, \mathtt{sk})$, the regular decryption algorithm
  - $m_1 = \mathsf{aDec}(\mathtt{ct}, \mathtt{dkey})$, the anamorphic decryption algorithm
- ...and there is **no** second key
  - at least, that's what the dictator thinks
  - when dictator asks for keys, give him $\mathtt{sk}$ because *there is only one key...*

The anamorphic thesis

The anamorphic thesis

Regulating/crippling encryption is technically **futile**

- Because Anamorphic Encryption exists
  - Feasibility of Anamorphic Encryption [P, Phan, Yung – EC22]
    - The Naor-Yung CCA encryption scheme
- Because Anamorphic Encryption is being used
  - Prevalence of Anamorphic Encryption [Kutilowski, P, Phan, Yung, Zawada – PETS 23]
    - Paillier, RSA-OAEP, Goldwasser-Micali
    - ElGamal, Cramer-Shoup, Smooth Projective Hash Functions

Resistance is futile

A Frequent Objection

The dictator can hit you with the wrench until you surrender the second key

and if there is no second key, too bad...

A Frequent Objection

The dictator can hit you with the wrench until you surrender the second key

and if there is no second key, too bad...

The objective of the research is not to defeat the dictator

sorry, can't help you with that...

A Frequent Objection

The dictator can hit you with the wrench until you surrender the second key

and if there is no second key, too bad...

The objective of the research is not to defeat the dictator

sorry, can't help you with that...

Rather to tell democracies how low they must go to effectively control private communication

and then ask if it is worth it

Futile, you said?

Encryption is declared illegal

- the dictator mandates that all communication happens through a central hub
- messages can only be digitally signed
  - so that we know whom we are talking to

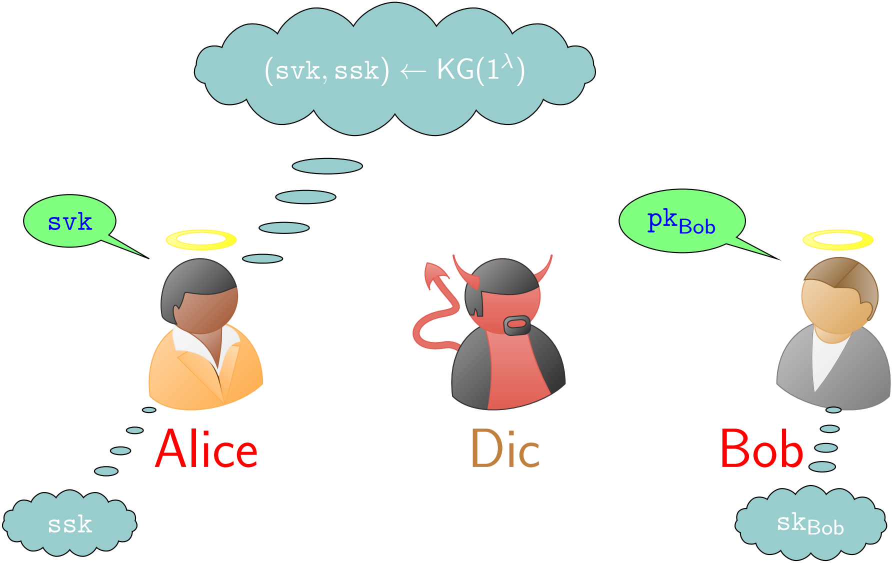
Futile, you said?

Encryption is declared illegal

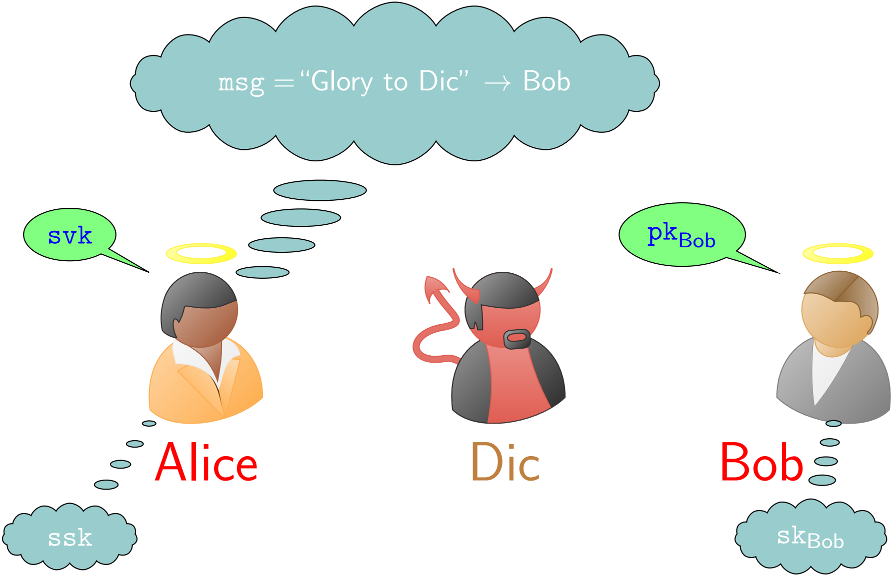
- the dictator mandates that all communication happens through a central hub
- messages can only be digitally signed
  - so that we know whom we are talking to

Do not annoy your dictator!

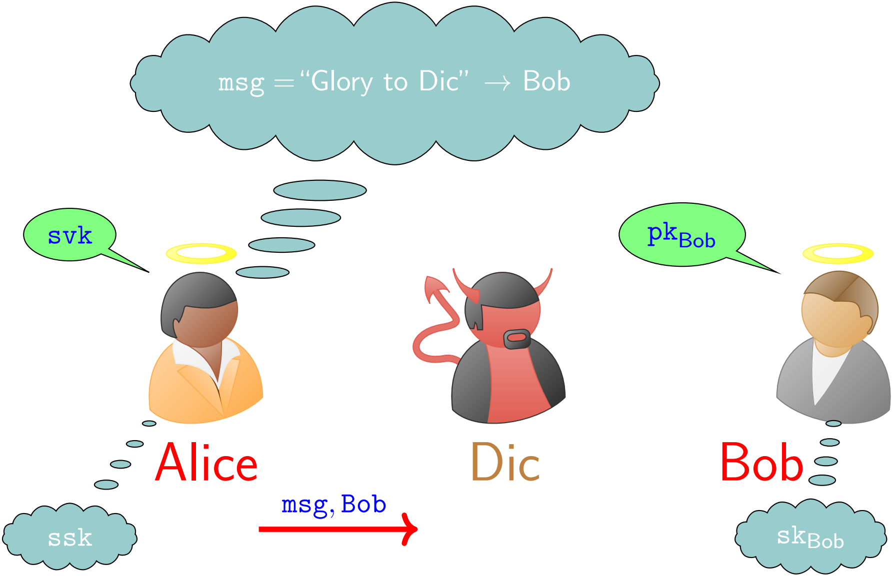
The new dictatorial setting



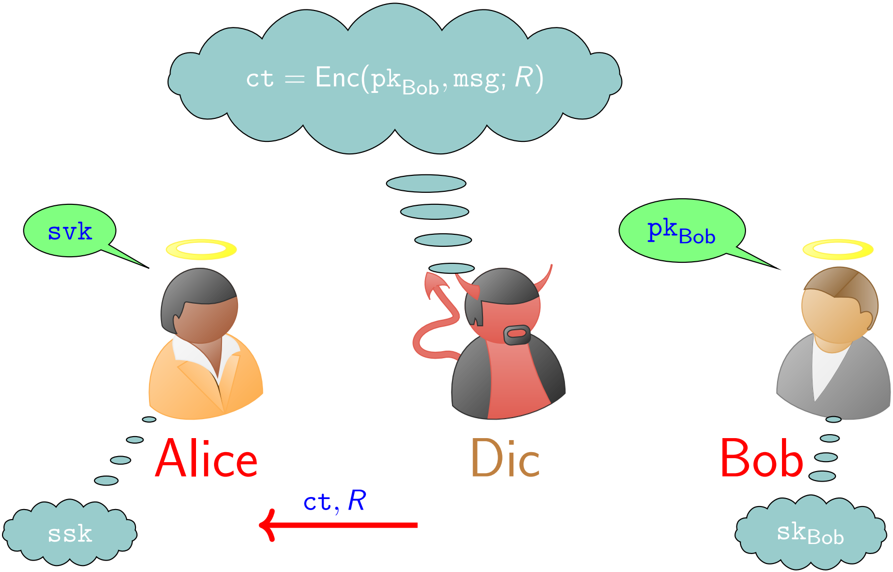
The new dictatorial setting



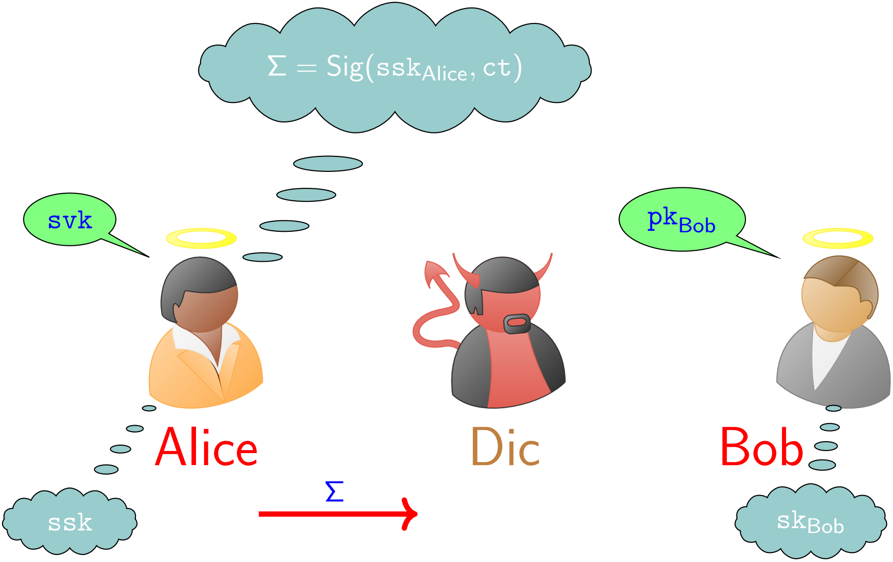
The new dictatorial setting



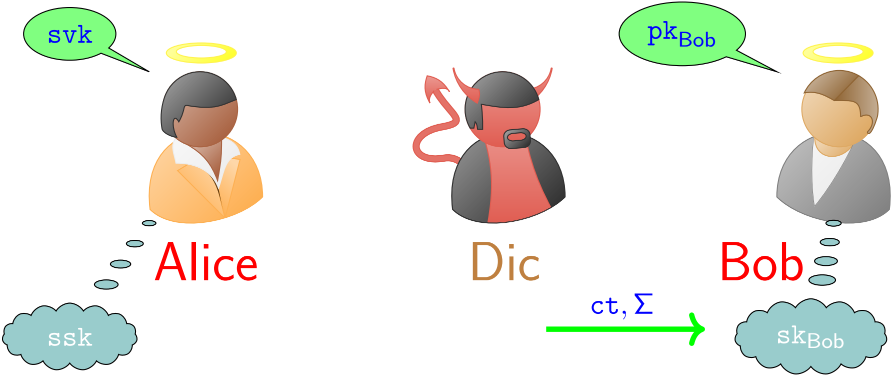
The new dictatorial setting



The new dictatorial setting



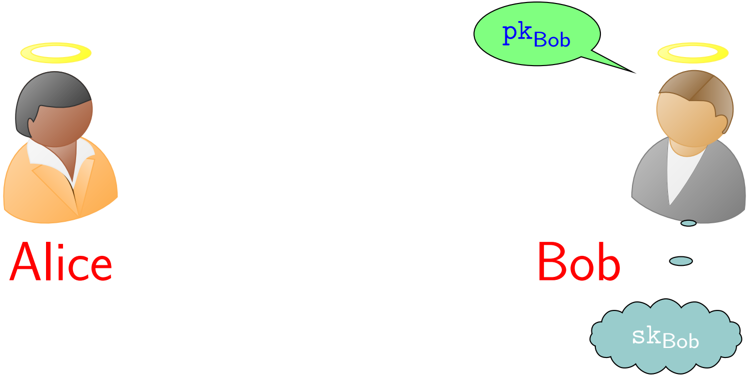
The new dictatorial setting



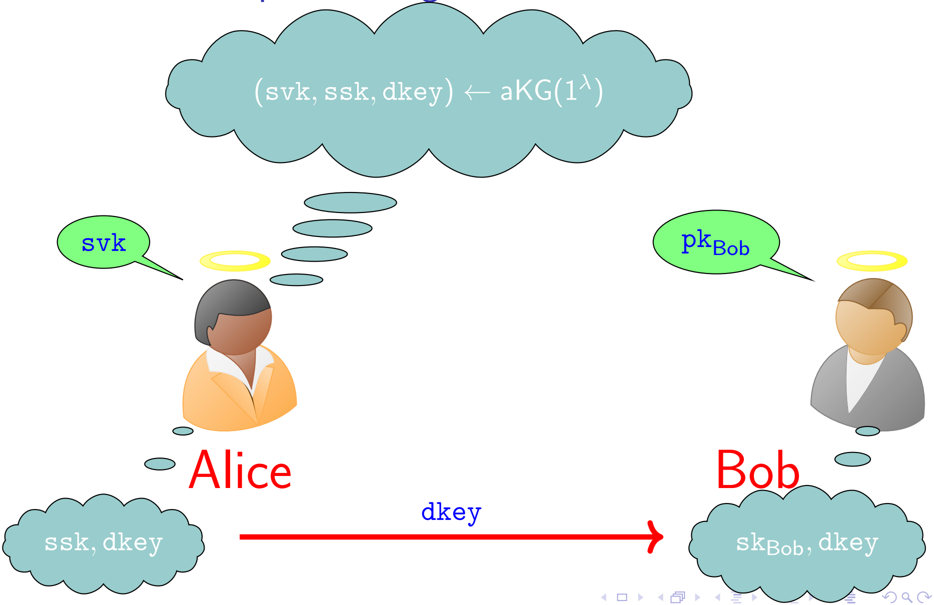
Dictator's thinking

- Every user has a private channel to the Dic
- Every user has a public and secret encryption key
- Every user has a verification and signing key
- Dic is the only allowed to use encryption on a public channel

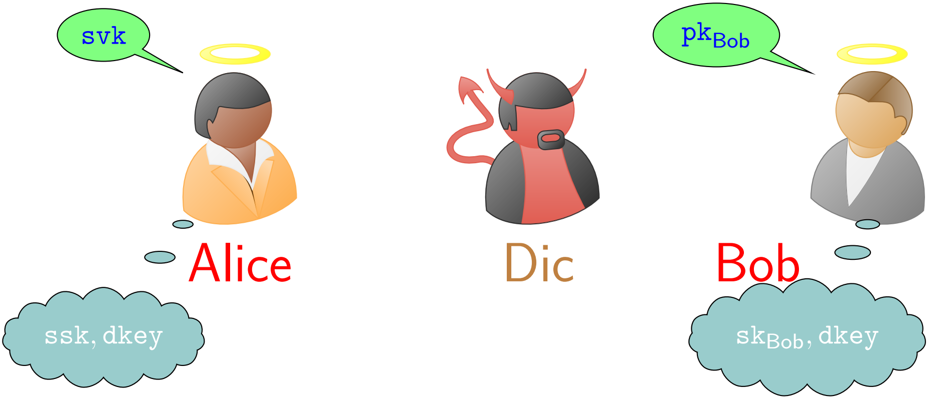
The new anamorphic setting



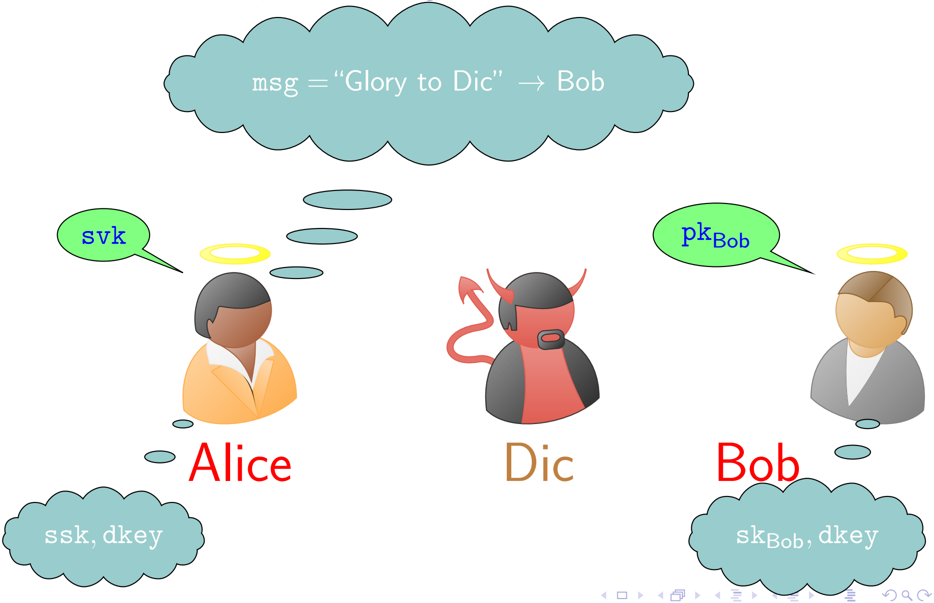
The new anamorphic setting



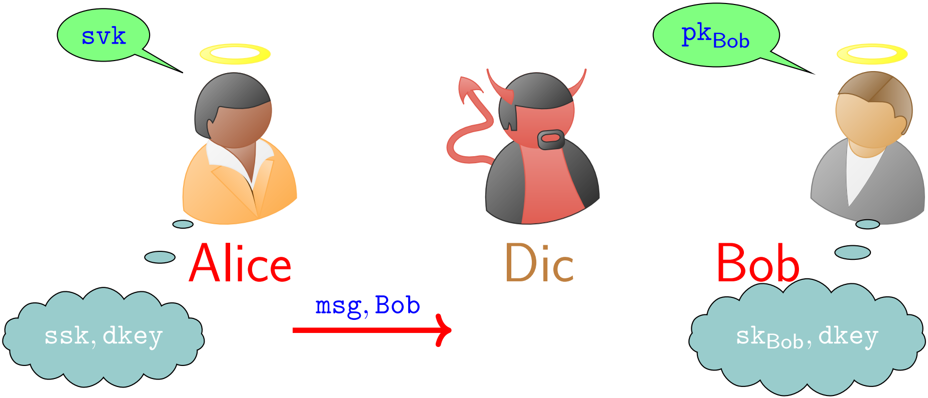
The new anamorphic setting



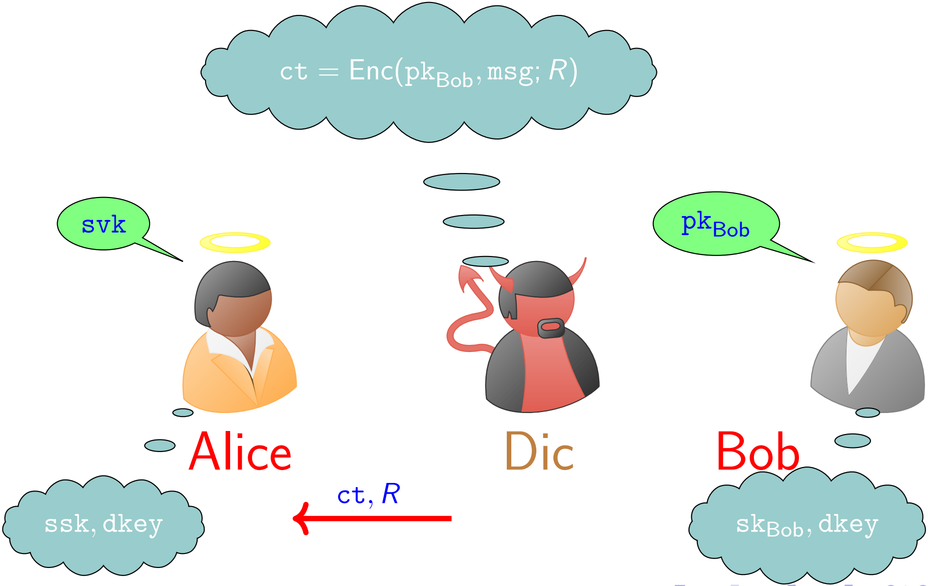
The new anamorphic setting



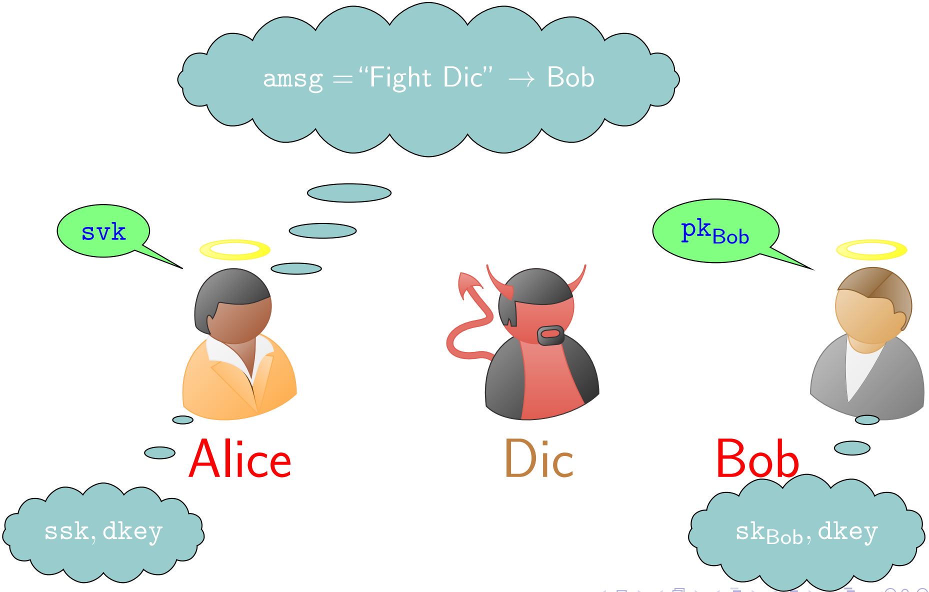
The new anamorphic setting



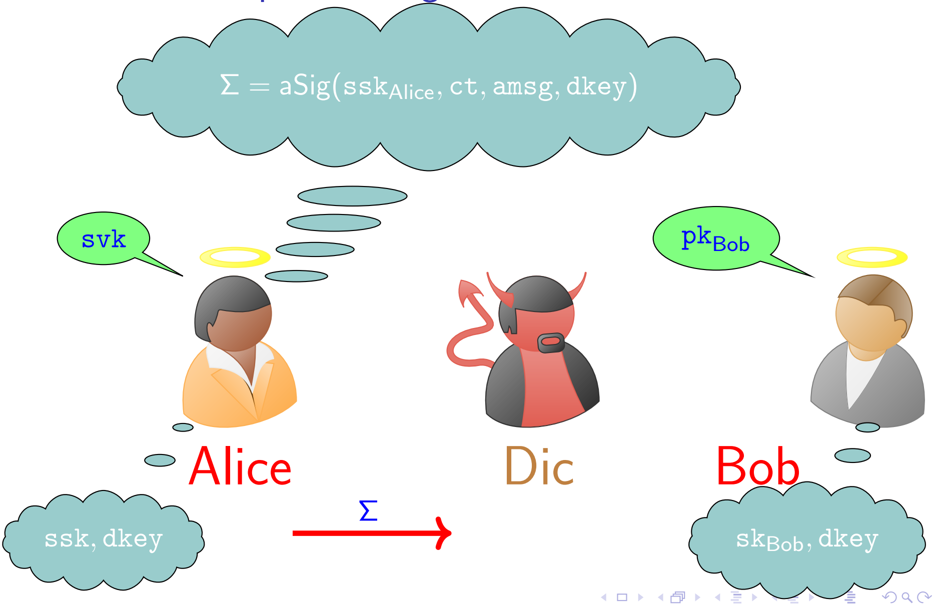
The new anamorphic setting



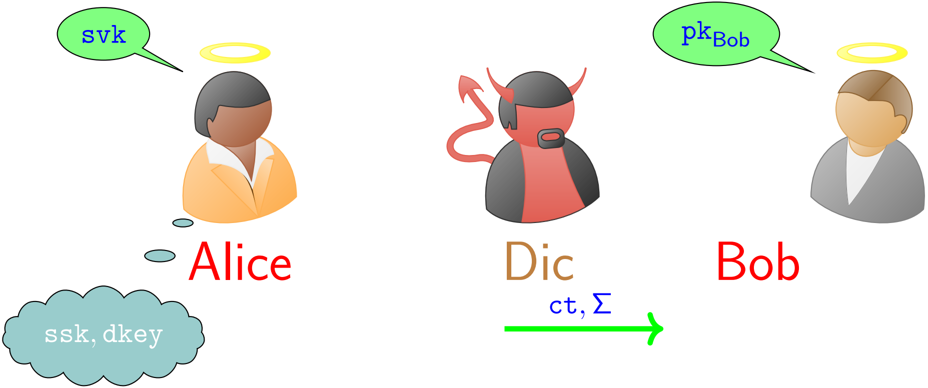
The new anamorphic setting



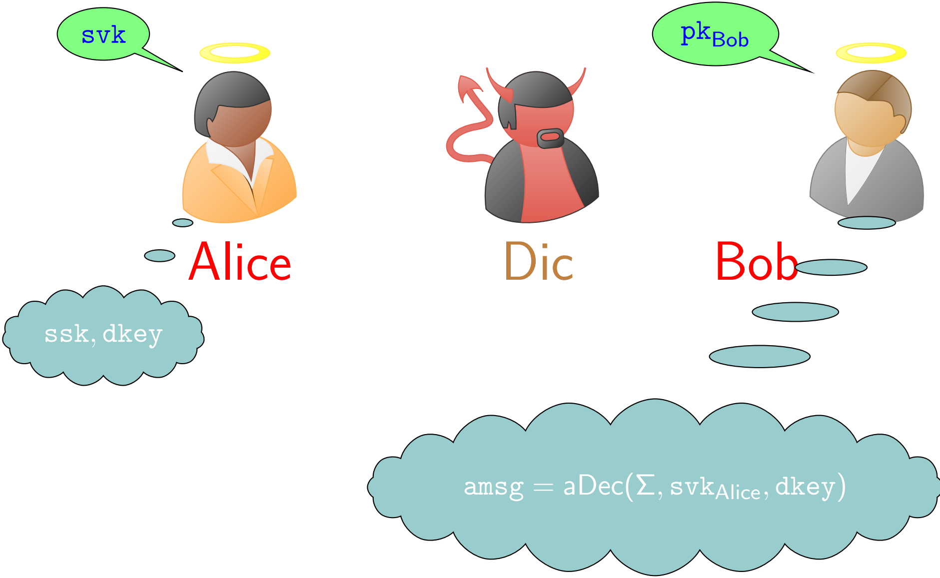
The new anamorphic setting



The new anamorphic setting



The new anamorphic setting



Signature Schemes

- the *key-generation* algorithm $\mathsf{KG}(1^\lambda)$
  - $(\mathtt{svk}, \mathtt{ssk})$, a public *verification* key and secret *signing* key;
- the *signing* algorithm $\mathsf{Sig}(\mathtt{msg}, \mathtt{ssk})$
  - *signature* Σ;
- the *verification* algorithm $\mathsf{Verify}(\Sigma, \mathtt{msg}, \mathtt{svk})$
  - accepts or rejects Σ as a signature of $\mathtt{msg}$.

Anamorphic Triplet

- the *anamorphic key-generation* algorithm $\mathsf{aKG}(1^\lambda)$
  - $(\mathtt{svk}, \mathtt{ssk}, \mathtt{dkey})$, a public *verification* key, a secret *signing* key, and a *double* key;
- the *anamorphic signing* algorithm $\mathsf{aSig}(\mathtt{msg}, \mathtt{amsg}, \mathtt{ssk}, \mathtt{dkey})$
  - *anamorphic signature* Σ;
- the *anamorphic decryption* algorithm $\mathsf{aDec}(\Sigma, \mathtt{svk}, \mathtt{dkey})$
  - $\mathtt{amsg}$.

Security Notion for Anamorphic Signatures

$\mathsf{RealG}_{\mathsf{S},\mathcal{D}}(\lambda)$

1. $(\mathtt{svk}, \mathtt{ssk}) \leftarrow \mathsf{KG}(1^\lambda)$;
2. return $\mathcal{D}^{\mathsf{Os}(\cdot,\cdot,\mathtt{ssk})}(\mathtt{svk}, \mathtt{ssk})$, where
   $\mathsf{Os}(\mathtt{msg}, \mathtt{amsg}, \mathtt{ssk}) = \mathsf{Sig}(\mathtt{msg}, \mathtt{ssk})$.

$\mathsf{AnamorphicG}_{\mathsf{T},\mathcal{D}}(\lambda)$

1. $(\mathtt{asvk}, \mathtt{assk}, \mathtt{dkey}) \leftarrow \mathsf{aKG}(1^\lambda)$;
2. return $\mathcal{D}^{\mathsf{Oa}(\cdot,\cdot,\mathtt{assk},\mathtt{dkey})}(\mathtt{asvk}, \mathtt{assk})$, where
   $\mathsf{Oa}(\mathtt{msg}, \mathtt{amsg}, \mathtt{assk}, \mathtt{dkey}) = \mathsf{aSig}(\mathtt{msg}, \mathtt{amsg}, \mathtt{assk}, \mathtt{dkey})$.

Two good points raised by the audience

- Going back to our motivating scenario:
  - `msg` is the ciphertext `ct` produced by $\mathcal{D}$ that carries the innocent looking message "Glory to Dic".
  - `amsg` is the message "Fight Dic" that Alice wants to secretely send to Bob.

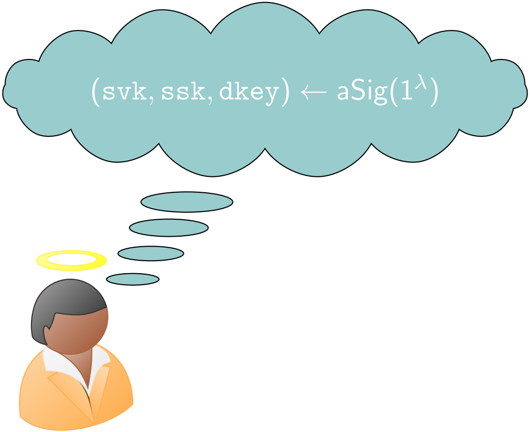
  To get a stonger notion of security, we require indistinguishability to hold even if $\mathcal{D}$ picks the two messages

- If $\mathcal{D}$ has access to `ssk`, what is the point of having Alice sign the ciphertext?
  - in the motivating scenario (see previous slides), $\mathcal{D}$ does not have access to `ssk`
  - still we require indistinguishability to hold in case $\mathcal{D}$ has some doubt and requests Alice's signing key

Anamorphic Channels

- `dkey` can be used to establish an anamorphic channel between signers and verifiers that have access to `dkey`
- The channel can be One-to-Many
  - `dkey` does not give you the ability to sign
  - only the signer can send anamorphic messages
- The channel can be Many-to-Many
  - `dkey` does give you the ability to sign
  - everybody is a signer and can send anamorphic messages

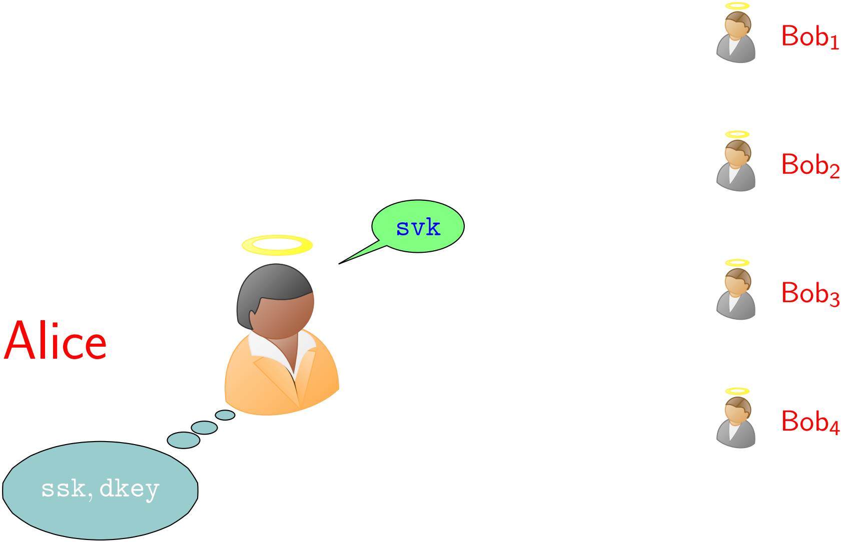
Many-To-Many



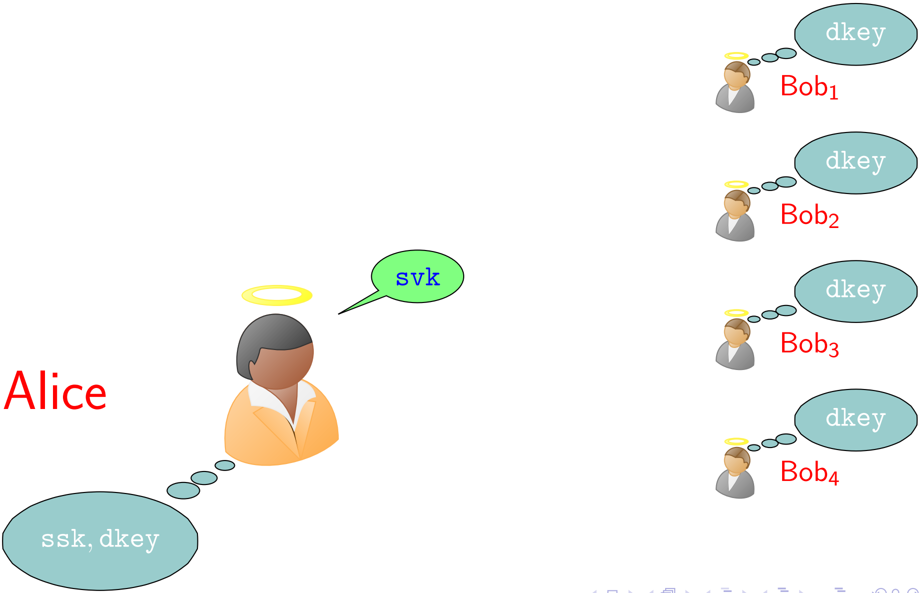
Alice

Many-To-Many

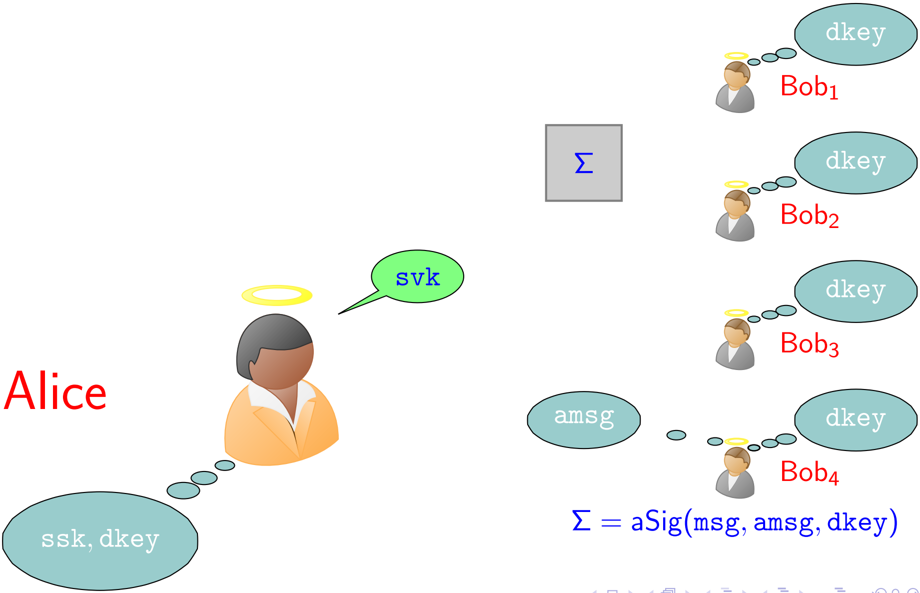
Many-To-Many



Many-To-Many

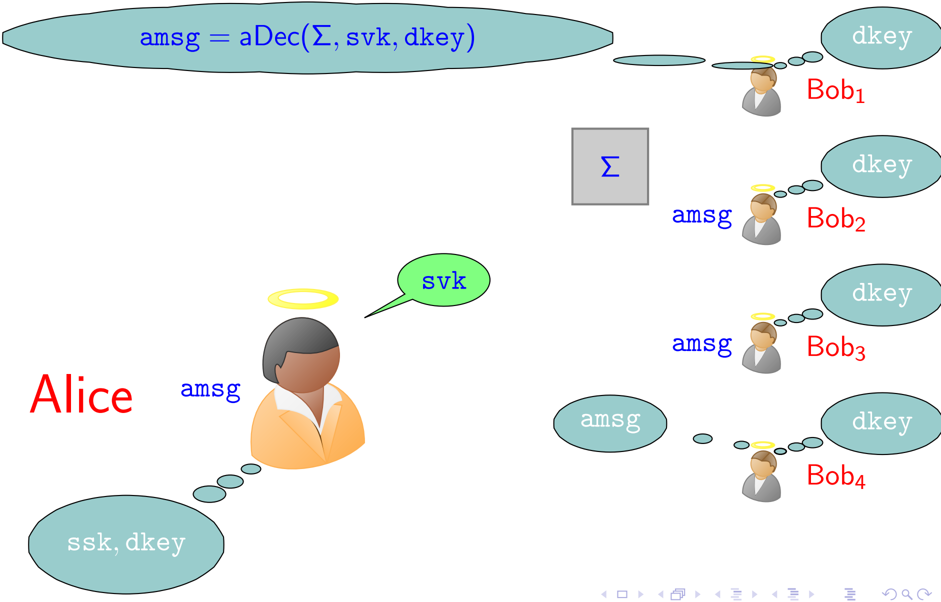


Many-To-Many



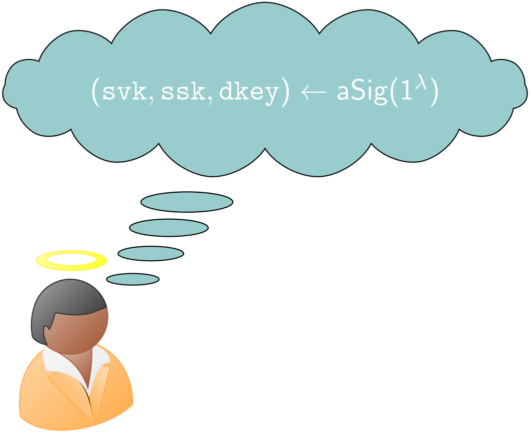
$\Sigma = \mathsf{aSig}(\mathtt{msg}, \mathtt{amsg}, \mathtt{dkey})$

Many-To-Many



$\mathtt{amsg} = \mathsf{aDec}(\Sigma, \mathtt{svk}, \mathtt{dkey})$
dkey
Bob$_1$
Σ
dkey
amsg
Bob$_2$
svk
dkey
amsg
Bob$_3$
Alice
amsg
amsg
dkey
Bob$_4$
ssk, dkey

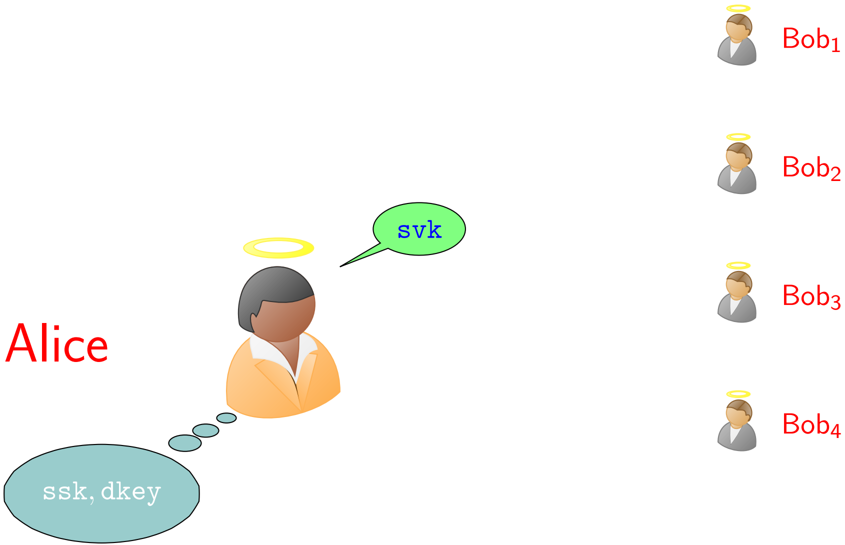
One-To-Many



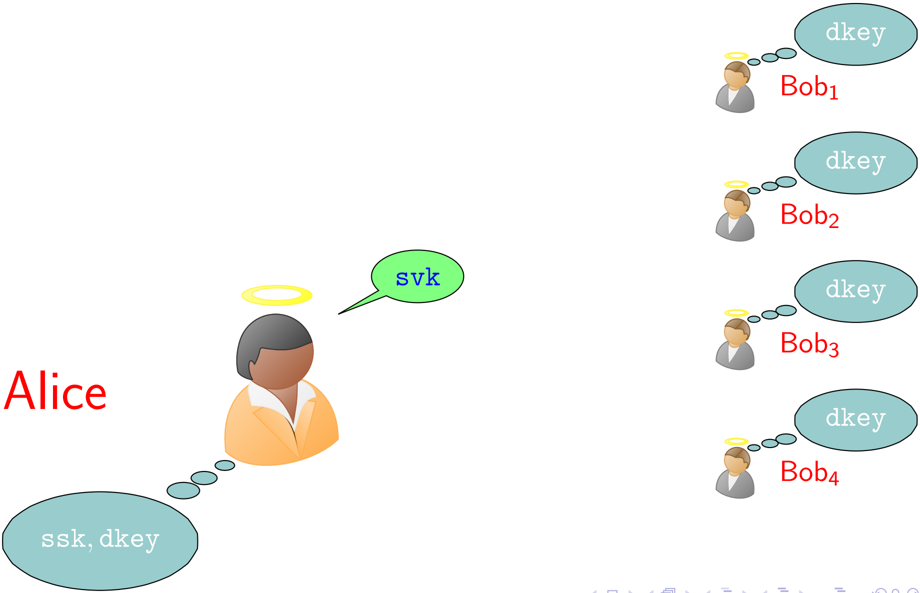
Alice

One-To-Many

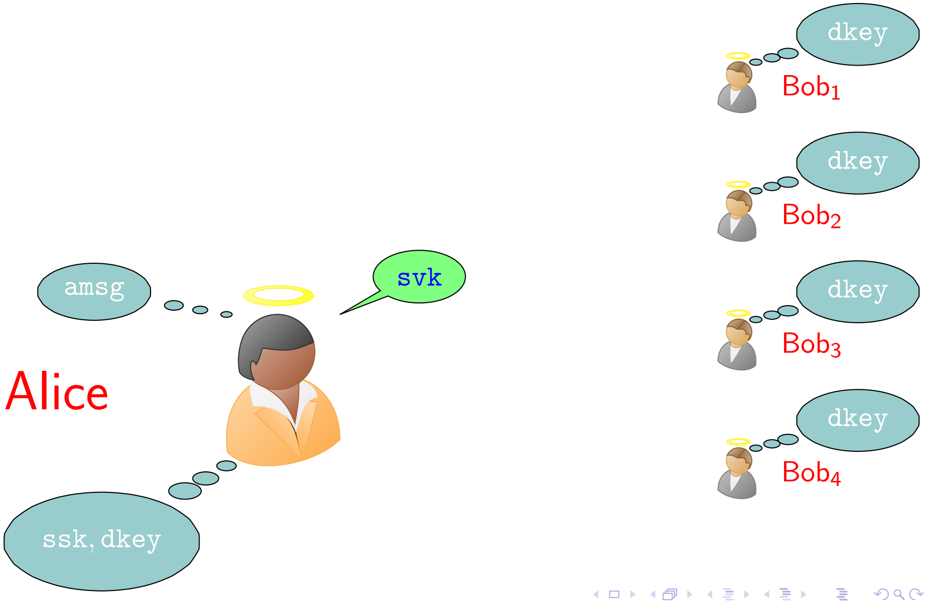
One-To-Many



One-To-Many



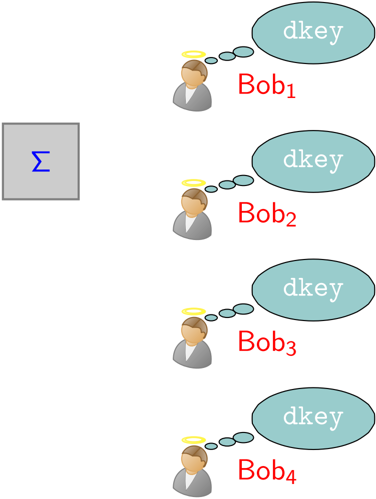
One-To-Many



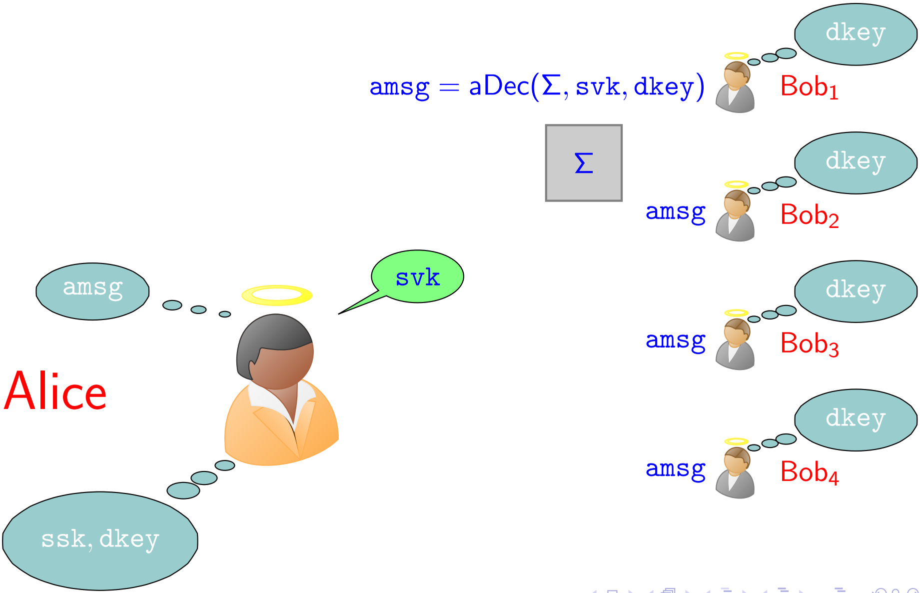
dkey
Bob$_1$
dkey
Bob$_2$
amsg
svk
dkey
Bob$_3$
Alice
dkey
Bob$_4$
ssk, dkey

One-To-Many

$\Sigma = \mathsf{aSig}(\mathtt{msg}, \mathtt{amsg}, \mathtt{ssk}, \mathtt{dkey})$



One-To-Many



amsg = aDec(Σ, svk, dkey)
Σ
dkey
Bob1
dkey
amsg
Bob2
dkey
amsg
Bob3
dkey
amsg
Bob4
amsg
svk
Alice
ssk, dkey

One-to-Many Anamorphic Signatures

$\mathsf{DsigG}^{\mathcal{A}}_{\mathsf{S},\mathsf{T}}(\lambda)$

1. $(\mathtt{asvk}, \mathtt{assk}, \mathtt{dkey}) \leftarrow \mathsf{aKG}(1^\lambda)$;
2. $(\mathtt{msg}, \Sigma) \leftarrow \mathcal{A}^{\mathsf{Os}(\cdot, \mathtt{assk})}(\mathtt{asvk}, \mathtt{dkey})$, where $\mathsf{Os}(m, \mathtt{assk}) = (m, \mathsf{Sig}(m, \mathtt{assk}))$;
3. if $\mathsf{Verify}(\Sigma, \mathtt{msg}, \mathtt{asvk}) = 1$ and $(\mathtt{msg}, \Sigma)$ has not been returned by Os then return 1; else return 0.

A General Technique

- `dkey` includes the key K of a symmetric encryption scheme
- A two step procedure
  - identify *extractable* randomness from the signature
  - replace randomness with ciphertext encrypted using K

ciphertexts must be indistinguishable from random

Symmetric Encryption with PseudoRandom Ciphertexts

prEnc returns $\ell(\lambda)$-bit ciphertexts for encrypting $n(\ell)$-bit messages with a key with security parameter λ

$\mathsf{PRCtG}^{\beta}_{\mathsf{prE},\mathcal{A}}(\lambda)$

1. Set $K \leftarrow \mathsf{prKG}(1^\lambda)$
2. Return $\mathcal{A}^{\mathsf{OPr}^{\beta}(K,\cdot)}()$, where, for $n(\lambda)$-bit plaintext $\mathtt{msg}$,
   $\mathsf{OPr}^0(K,\mathtt{msg})$ returns a randomly selected $\ell(\lambda)$-bit string;
   $\mathsf{OPr}^1(K,\mathtt{msg}) = \mathsf{prEnc}(K,\mathtt{msg})$.

The Boneh-Boyen signature scheme

- The Key Generation algorithm $\mathrm{bbKG}(1^\lambda)$
  - $(G_1, G_2, G_T, \mathrm{e}, p) \leftarrow \mathcal{G}(1^\lambda)$
  - Generators $g_1 \in G_1, g_2 \in G_2$
  - $x, y \leftarrow \mathbb{Z}_p$
  - $z = \mathrm{e}(g_1, g_2), u = g_2^x, v = g_2^y$.
  - $\mathtt{svk} = (g_1, g_2, u, v, z)$ and $\mathtt{ssk} = (g_1, x, y)$.
- The Signing algorithm $\mathrm{bbSig}(\mathtt{ssk} = (g_1, x, y), \mathtt{msg} \in \mathbb{Z}_p)$
  - randomly selects $r \leftarrow \mathbb{Z}_p$.
  - If $r = -(x + \mathtt{msg})/y$ then $\perp$.
  - return $(r, \sigma = g_1^{1/(x+\mathtt{msg}+yr)})$.
- The Verification algorithm $\mathrm{bbVerify}(\Sigma = (r, \sigma))$
  - check

$$\mathrm{e}(\sigma, u \cdot g_2^m \cdot v^r) = z.$$

Anamorphic Triplet for BB

- The *anamorphic key generation* algorithm $\mathsf{abbKG}(1^\lambda)$
  - $(\mathtt{svk}, \mathtt{ssk}) \leftarrow \mathsf{bbKG}(1^\lambda)$
  - $K \leftarrow \mathsf{prKG}(1^\lambda)$.
  - return
    - anamorphic verification key $\mathtt{asvk} := \mathtt{svk}$,
    - anamorphic signing key $\mathtt{assk} := \mathtt{ssk}$,
    - double key $\mathtt{dkey} := K$
- The *anamorphic signing* algorithm $\mathsf{abbSig}(\mathtt{msg}, \mathtt{amsg}, \mathtt{ssk}, \mathtt{dkey})$
  - $\mathsf{act} = \mathsf{prEnc}(K, \mathtt{amsg})$
  - $r = \mathsf{act}$ and if $r = -(x+m)/y$ then $\perp$
  - return $\mathsf{a}\Sigma = (r, \sigma = g_1^{1/(x+m+yr)})$.
- The *anamorphic decryption* algorithm $\mathsf{aDec}(\mathsf{a}\Sigma = (r, \sigma), \mathtt{dkey} = K)$
  - return $\mathtt{amsg} = \mathsf{prDec}(K, r)$.

The Fiat-Shamir Heuristics

The Schnorr Signature Scheme

- The *Key Generation* algorithm $\mathsf{ScKG}(1^\lambda)$
  - $\mathbb{G}$, a cyclic group of prime order q
  - a generator $g \in \mathbb{G}$
  - hash function $H : \{0,1\}^\star \times \mathbb{G} \rightarrow \mathbb{Z}_q$.
  - $x \leftarrow \mathbb{Z}_q$ and sets $y = g^x$

  $\mathtt{ssk} := (\mathbb{G}, g, H, x)$ and $\mathtt{svk} := (\mathbb{G}, g, H, y)$.

- The *Signing* algorithm $\mathsf{ScSig}(\mathtt{ssk}, \mathtt{msg})$
  - $\kappa \leftarrow \mathbb{Z}_q$,
  - $r = g^\kappa$, $c = H(\mathtt{msg}, r)$ and $s = \kappa + c \cdot x$
  - Return $\Sigma = (r, s)$.

- The *Verification* algorithm $\mathsf{ScVerify}(\Sigma, \mathtt{msg}, \mathtt{svk})$
  - checks that

$$r = g^s \cdot y^{-H(\mathtt{msg}, r)}$$

Anamorphic Schnorr

- The *Signing* algorithm $\mathrm{ScSig}(\mathtt{ssk}, \mathtt{msg})$
  - $\kappa \leftarrow \mathbb{Z}_q$,
  - $r = g^{\kappa}$, $c = H(\mathtt{msg}, r)$ and $s = \kappa + c \cdot x$
  - Return $\Sigma = (r, s)$

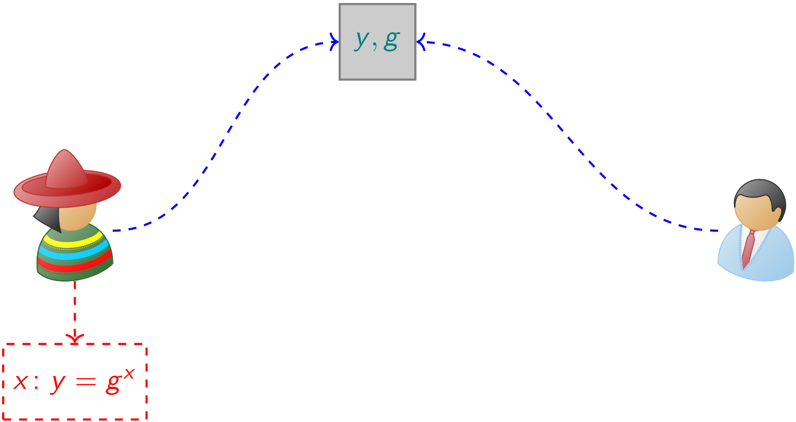
Anamorphic Schnorr

- The *Signing* algorithm $\mathsf{ScSig}(\mathtt{ssk}, \mathtt{msg})$
  - $\kappa \leftarrow \mathbb{Z}_q$,
  - $r = g^\kappa$, $c = H(\mathtt{msg}, r)$ and $s = \kappa + c \cdot x$
  - Return $\Sigma = (r, s)$

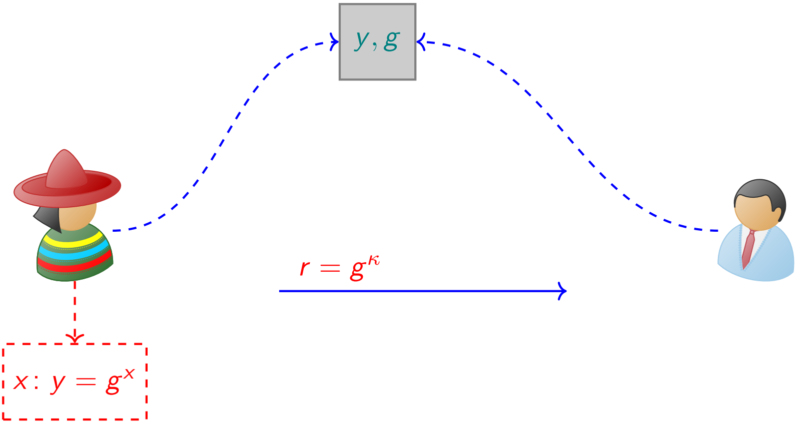
Fishing for randomness

- Set $r = \mathsf{prEnc}(K, \mathtt{amsg})$
  - need κ to compute s
- Set $\kappa = \mathsf{prEnc}(K, \mathtt{amsg})$
  - cannot recover κ during verification
    - add x to $\mathtt{dkey}$
    - a many-to-many anamorphic channel

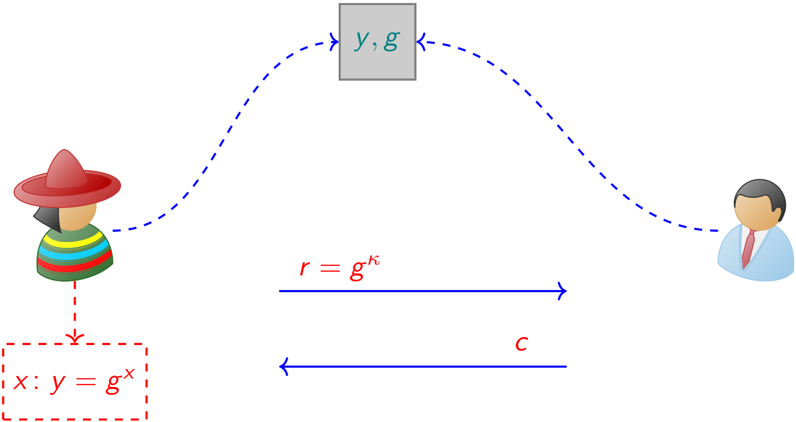
Schnorr's Proof of Knowledge



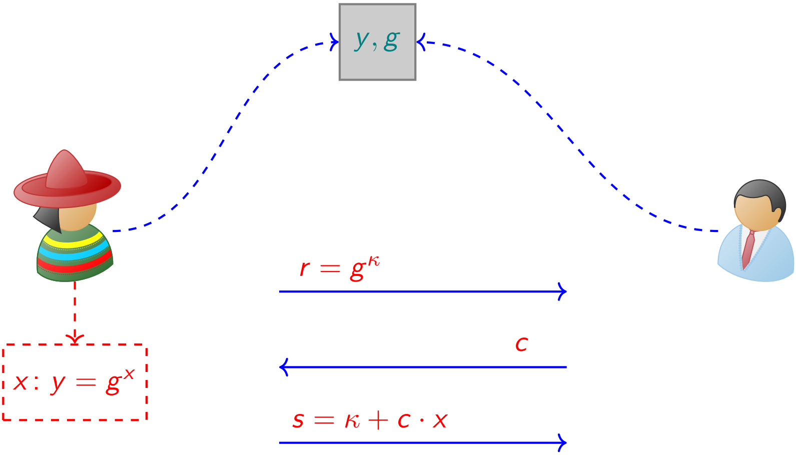
Schnorr's Proof of Knowledge



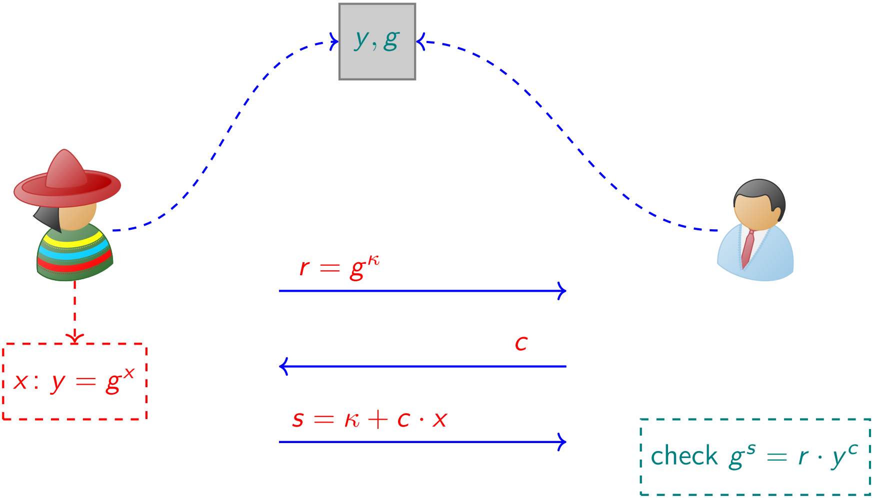
Schnorr's Proof of Knowledge



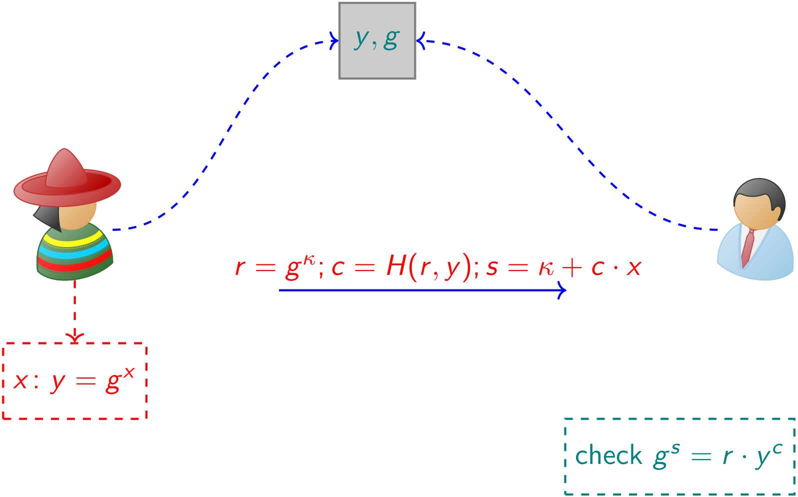
Schnorr's Proof of Knowledge



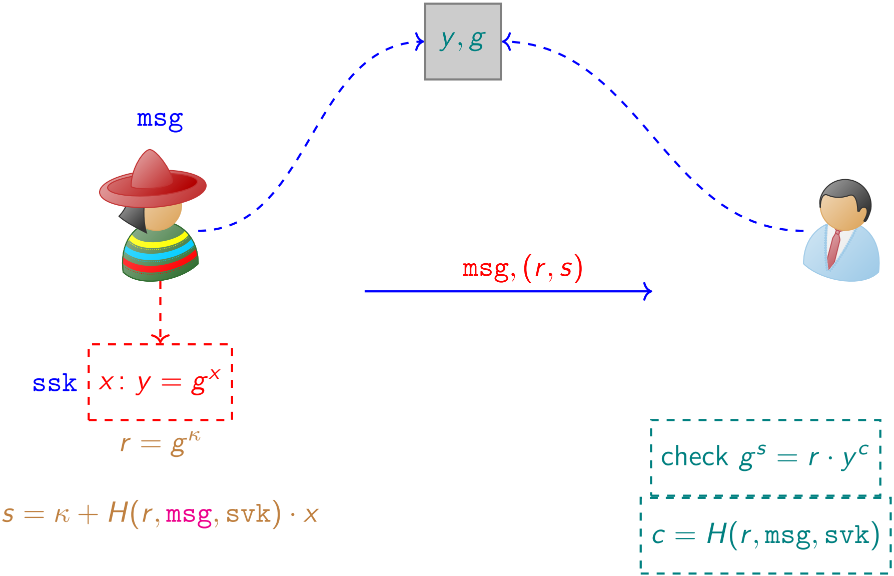
Schnorr's Proof of Knowledge



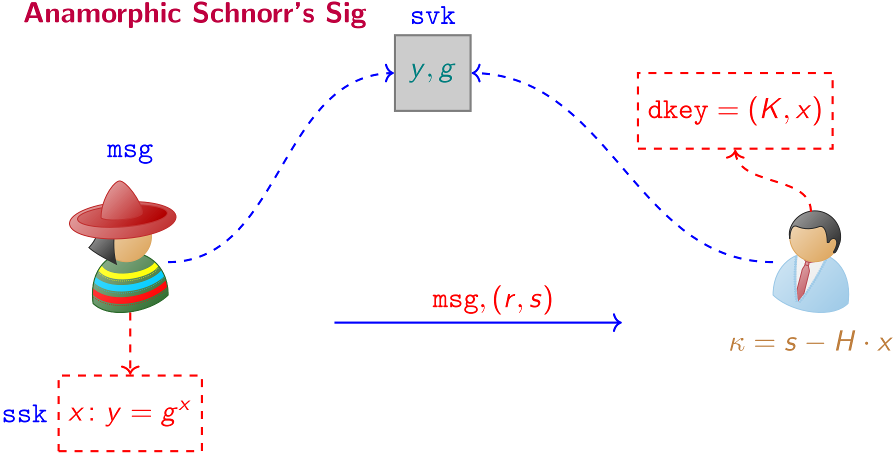
Fiat-Shamir



Schnorr's Sig



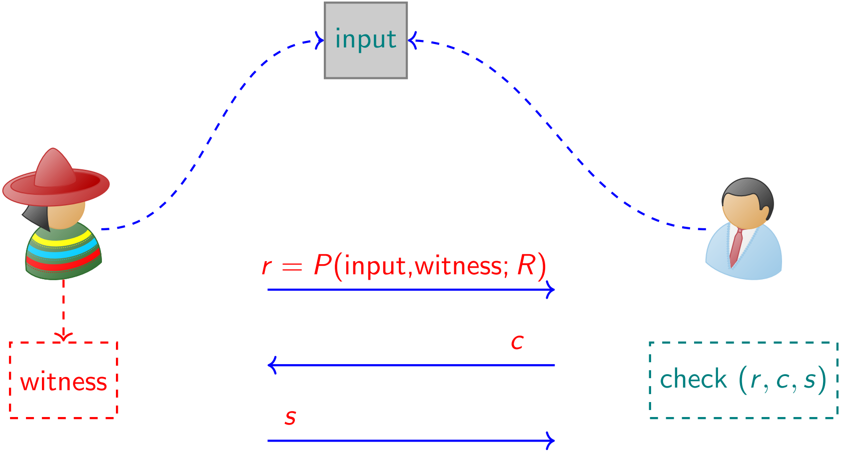
Anamorphic Schnorr's Sig



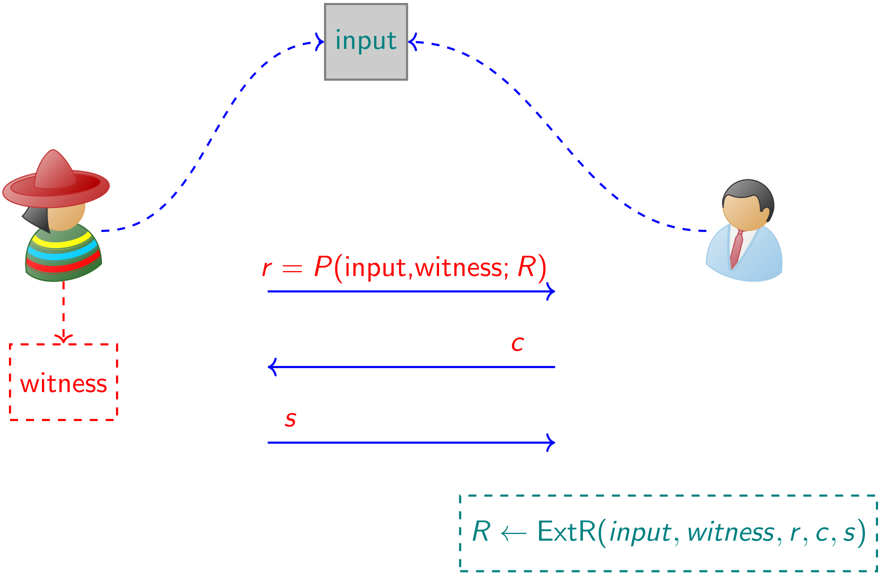
$$\kappa = \mathrm{prEnc}(K, \mathtt{amsg})$$

$$r = g^{\kappa}$$

$$s = \kappa + H(r, \mathtt{msg}, \mathtt{svk}) \cdot x$$


input
$r = P(\text{input,witness}; R)$
c
s
witness
check (r, c, s)

Sufficient condition for Anamorphism



Fiat-Shamir preserves Anamorphism

Fiat-Shamir Heuristics

How to construct a signature scheme from a 3-round interactive proof

Fiat-Shamir preserves Anamorphism

If 3-round interactive proof is anamorphic the resulting signature scheme is also anamorphic

The Naor-Yung Transformation

Lamport's Tagging Scheme [1979]

- *Key Generation* algorithm $\mathsf{LKG}(1^\lambda, 1^\ell)$
  - for $j = 1, \ldots, \ell$.
    - ★ $x_{0,j}, x_{1,j} \leftarrow \{0,1\}^\lambda$
    - ★ $y_{0,j} = f(x_{0,j})$ and $y_{1,j} = f(x_{1,j})$

  $\mathtt{Lvk} = ((y_{0,j}, y_{1,j}))_{j=1}^{\ell}$ and $\mathtt{Lsk} = ((x_{0,j}, x_{1,j}))_{j=1}^{\ell}$.

- *Signing* algorithm $\mathsf{LSig}(m_1, \ldots, m_\ell, \mathtt{Lsk})$
  - $\Sigma = (x_{m_j,j})_{j=1}^{\ell}$.

- *Verification* algorithm $\mathsf{LVerify}(\Sigma, m_1, \ldots, m_\ell, \mathtt{Lvk})$
  - check $f(s_j) = y_{m_j,j}$ for $j = 1, \ldots, \ell$.

We have a problem

- **Signing is deterministic**
  - no randomness to be extracted by the verifier
- There is randomness in the verification key: the $x_{b,j}$'s
- We can embed the anamorphic message $\mathtt{amsg}$ as $x_{0,1} = \mathrm{prEnc}(K, \mathtt{amsg})$ and $x_{1,1} = \mathrm{prEnc}(K, \mathtt{amsg})$
  - anamorphic message to be determined at key generation time
  - **weakly anamorphic**
- It is a one-time signature
- We are going to be fine
  - for one-time signatures key generation time "*coincides*" with signing time

The Naor-Yung transform

NY Lifting

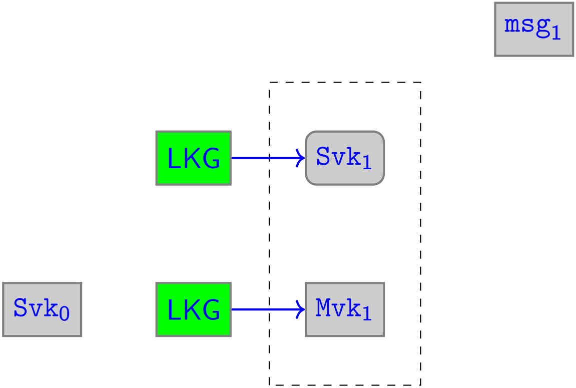
- one-time to multi-time signature
- weakly anamorphic to fully anamorphic

The Naor-Yung transform

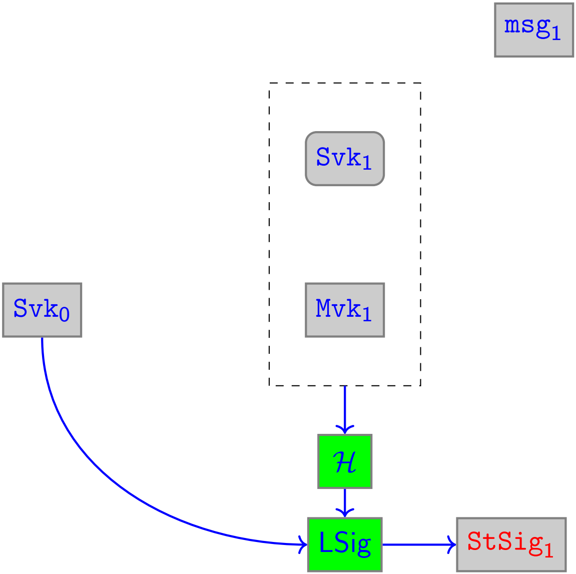
$\mathtt{msg}_1$

$\mathtt{Svk}_0$

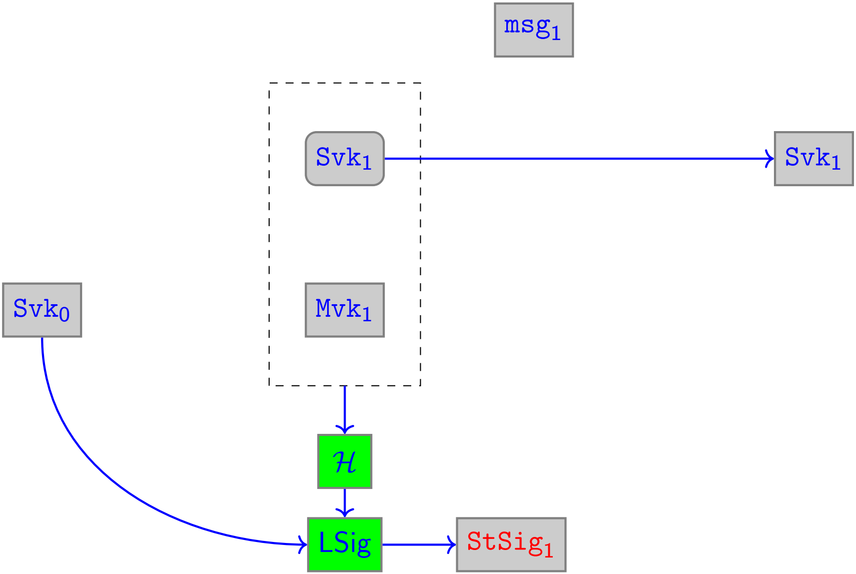
The Naor-Yung transform



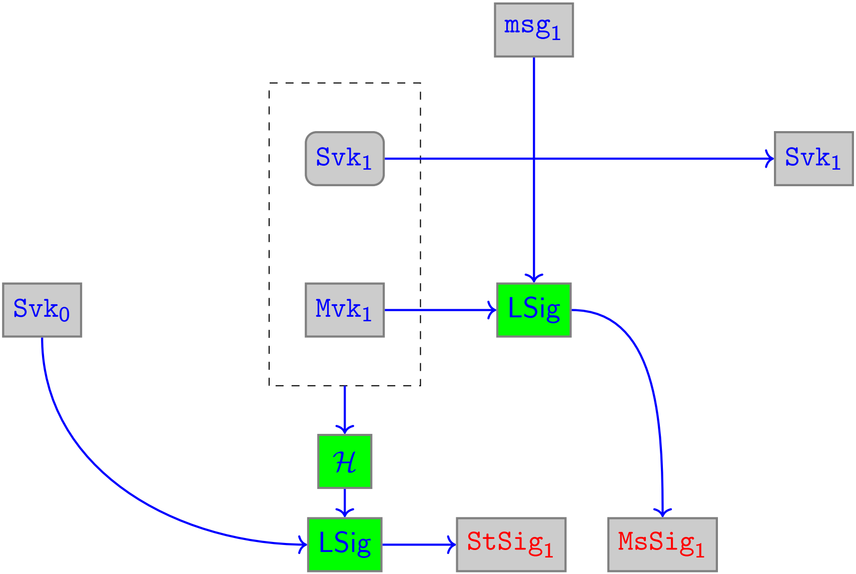
The Naor-Yung transform



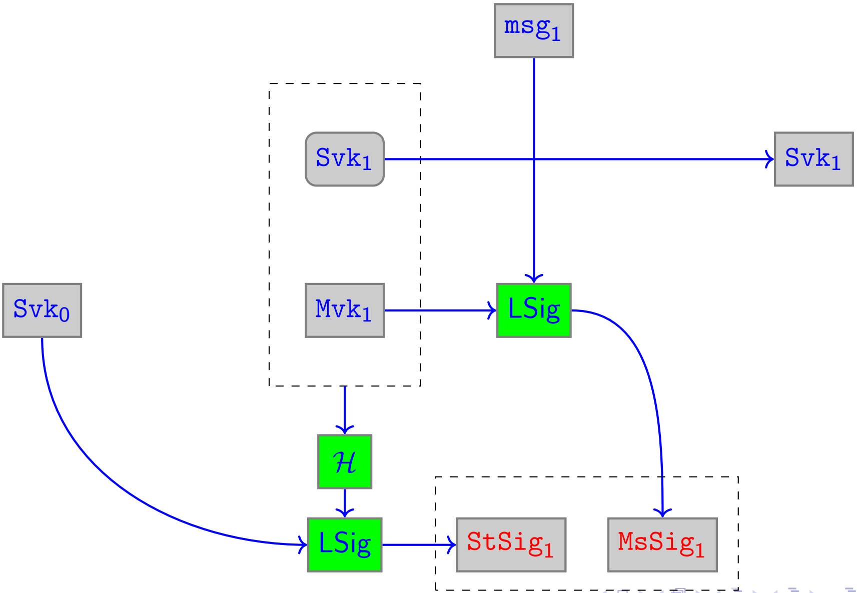
The Naor-Yung transform



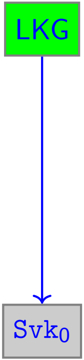
The Naor-Yung transform



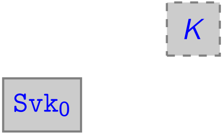
The Naor-Yung transform



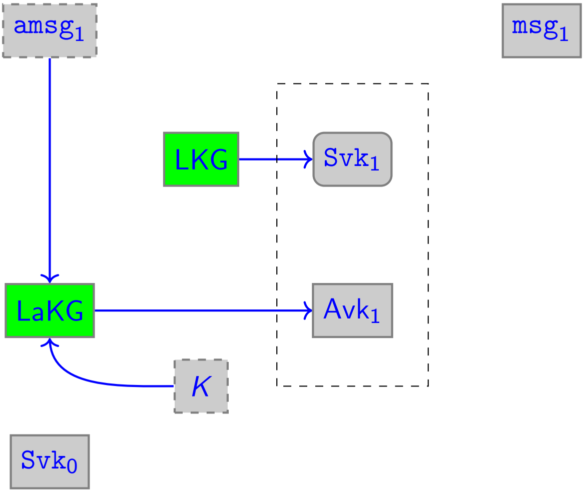
Anamorphic Lifting via the Naor-Yung transform



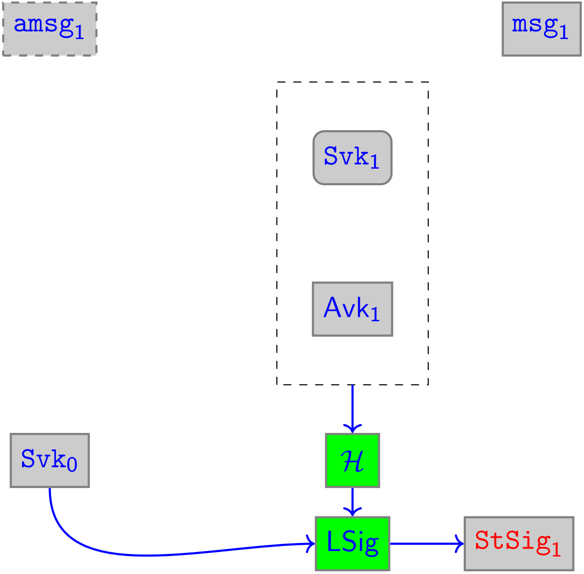
Anamorphic Lifting via the Naor-Yung transform



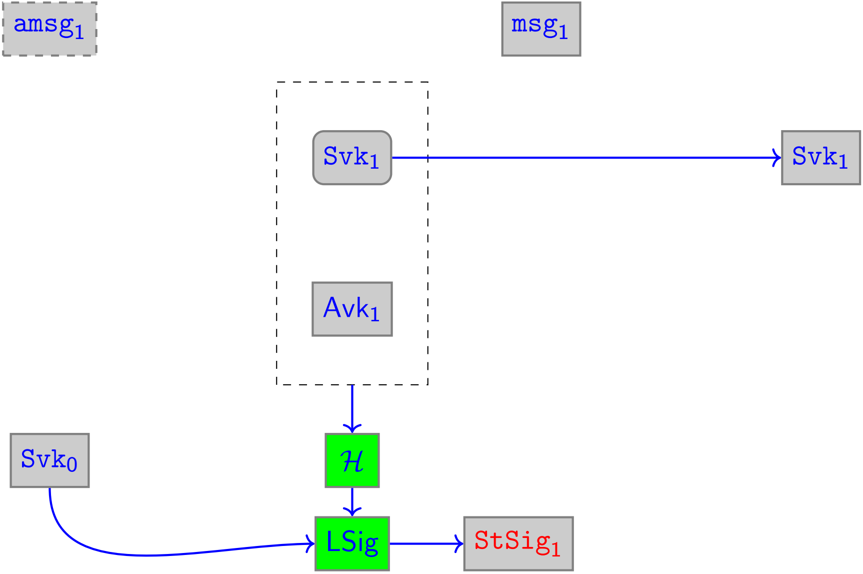
Anamorphic Lifting via the Naor-Yung transform



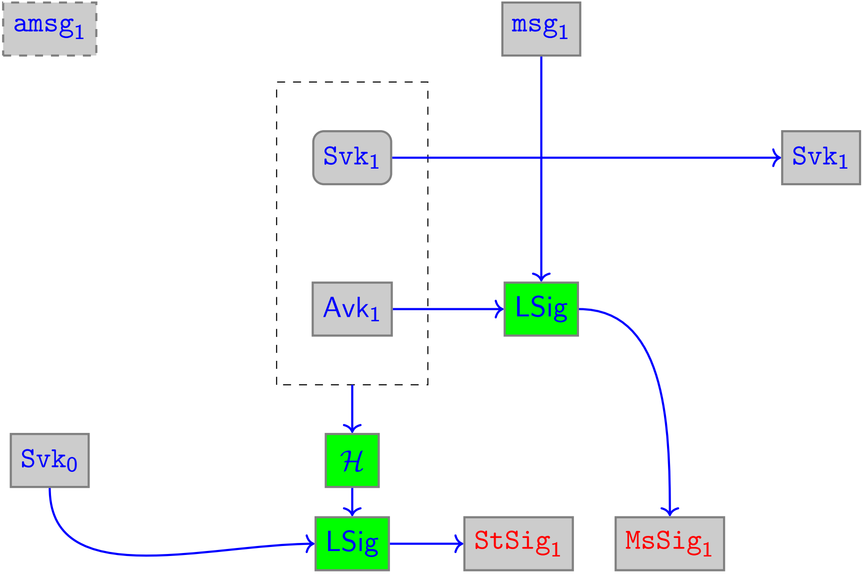
Anamorphic Lifting via the Naor-Yung transform



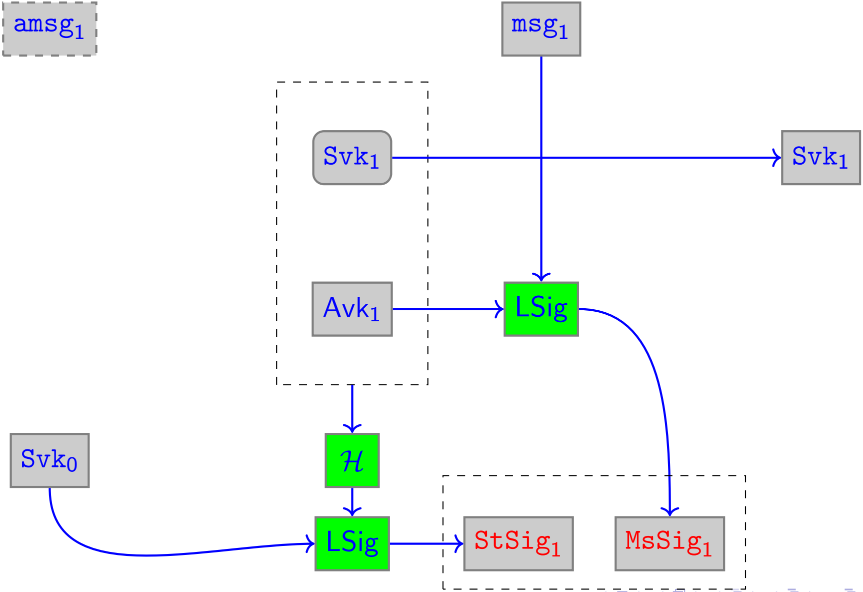
Anamorphic Lifting via the Naor-Yung transform



Anamorphic Lifting via the Naor-Yung transform



Anamorphic Lifting via the Naor-Yung transform



Anamorphism of NY

Lifting

If universal one-way hash functions exist, any weakly anamorphic one-time signature can be lifted to a fully anamorphic multi-time signature.

Naor-Yung-Lamport-Rompel

If one-functions exist then there exists a fully anamorphic multi-time signature scheme.

One-to-Many Anamorphism of NY

Separability of K

- K and $(\mathtt{Svk}_0, \mathtt{Ssk}_0)$ are independent
- only need K to extract $\mathtt{amsg}$
- K will not help produce signatures

Technical summary

- the new notion of anamorphic signature
- theoretical properties
- two flavors:
  - *one-to-many anamorphic* communication
    - ★ `dkey` allows decryption but not signature
  - *many-to-many anamorphic* communication
    - ★ `dkey` allows decryption and signature
- one general technique
  - extract randomness and replace with ciphertext

Technical Summary

- two design paradigms for signatures preserve anamorphism
  - Fiat-Shamir turns 3-round protocols into signatures in the ROM
    - ★ If prover randomness can be extracted, then resulting signature is anamorphic
    - ★ Schnorr, [Beth 88], [Guillou+Quisquater90], [Ong+Schnorr90, [Brickell+McCurley91], [Girault91], [Okamoto93],[Pointcheval95],[Stern94]
    - ★ Need the witness to extract (i.e., the signing key).
    - ★ Many-to-Many Anamorphism
  - Naor-Yung turns one-time signatures into many-time signatures in the standard model (assume one-way functions)
    - ★ If one-time signature eenjoys a weak form of anamorphism, resulting many-time is fully anamorphic
    - ★ Lamport, BC, HORS
    - ★ One-to-Many Anamorphism
- Applications to schemes using digital signatures
  - Canetti-Halevi-Katz CCA encryptions scheme uses a signature scheme
  - the transformation preserves anamorphism

Conclusions

- disallowing encryption is not sufficient
- must disallow message authentication too
  - complete disruption of communication
  - not clear who is talking to whom
- or disallow randomized signatures
  - more to come about this...
- dictator will not care
  - just give me `dkey` or else...
  - if no `dkey` then can't surrender it...
- technical evidence that a democracy cannot actually control communication
  - unless, that is, it ceases to be a democracy

- Giuseppe Persiano, Duong Hieu Phan, Moti Yung: *Anamorphic Encryption: Private Communication against a Dictator.* IACR Cryptol. ePrint Arch. 2022: 639 (2022). Eurocrypt '22

- Mirek Kutylowski, Giuseppe Persiano, Duong Hieu Phan, Moti Yung, Marcin Zawada: *The Self-Anti-Censorship Nature of Encryption: On the Prevalence of Anamorphic Cryptography.* IACR Cryptol. ePrint Arch. 2023: 434 (2023). PETS '23

- Mirek Kutylowski, Giuseppe Persiano, Duong Hieu Phan, Moti Yung, Marcin Zawada: *Anamorphic Signatures: Secrecy From a Dictator Who Only Permits Authentication!* IACR Cryptol. ePrint Arch. 2023: 356 (2023). CRYPTO '23