Anamorphic Cryptography: Current Developments

Private Communication against a Dictator

Giuseppe Persiano

Summer School on Real-World Crypto and Privacy – June 2024

Joint work with:
Duong Hieu Phan, Moti Yung and Miroslaw Kutylowski, Marcin Zawada.

Content

- Receiver-Privacy and Sender-Freedom: Dictators and Crypto Wars
- Our Approach for Receiver-Privacy: No New Constructions
- Receiver-Privacy: Formal Definitions
- Receiver-Privacy: Constructions
  - General result with low rate
  - Randomness Recoverable Encryption
  - CCA secure Encryption
- Signatures

Eurocrypt 2022 https://ia.cr/2022/639
PETS 2023 https://ia.cr/2023/434
Crypto 2023 https://ia.cr/2023/356
Crypto 2024

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

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

But its success relies on two assumptions that might be challenged in dictatorial states

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

The sender-freedom assumption

A ciphertext carries the message that was provided as an input, not the one that the sender wishes to encrypt

The sender-freedom assumption

The sender is free to pick the message to be encrypted

Receiver privacy and Sender freedom

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

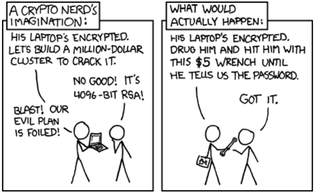
Receiver privacy and Sender freedom

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

Receiver privacy and Sender freedom

- Both assumptions are 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



  - No sender freedom: citizens might be invited to send messages to international newspapers to make the dictator look good

OK...two more assumptions

Why is this a problem?

Theorem

Assume existence of one-way functions and receiver privacy. Then, there exist secure symmetric encryption schemes.

Two assumptions

- Existence of one-way functions
- Ability to hide my key

Law of Nature vs Normative Prescription

- Assumption of the existence of one-way functions comes from *our current scientific understanding of Nature*
  - if true, it is enforced by Nature
  - it might be false but then it is false for all

- Receiver privacy is a *norm*:
  - it is enforced by political power
  - it can be changed by law, decree, force
  - it could change for some but not for all

Not only dictators...

Crypto Wars

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

- Combination of cryptographic tools and normative prescription
- From [Micali 1992] to [Green-Kaptchuk-van Laer 2021]
  - Rely on the existence of an independent judiciary system (missing in a Dictatorship!)
- Several related concepts
  - Kleptography [Young-Yung 97]
  - Subvertable encryption
  - Steganography (see later)

Crypto Wars

Several arguments have been made 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 systems security point*

Crypto Wars

Several arguments have been made 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 systems security point*

All these arguments are indirect and non-technical

Crypto Wars

Several arguments have been made 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 systems security point*

All these arguments are indirect and non-technical

We wish to give technical evidence that it is *futile* to try to restrict encryption

Resistance is futile


RESISTANCE IS FUTILE

Our approach for receiver privacy

Our approach for receiver privacy

Constraints

- If the dictator has the secret key $\mathtt{sk}$, it can decrypt and read messages
- But only messages encrypted with respect to $\mathtt{sk}$ can be decrypted

Our approach for receiver privacy

Constraints

- If the dictator has the secret key $\mathtt{sk}$, it can decrypt and read messages
- But only messages encrypted with respect to $\mathtt{sk}$ can be decrypted

Our approach

- A ciphertext is associated with two secret keys $\mathtt{sk}_0, \mathtt{sk}_1$
- share $\mathtt{sk}_1$ with your friend
- A ciphertext carries two plaintexts m_0, m_1, one for each key
- ...and there is **no** second key
  - at least, that's what the dictator thinks
  - when dictator asks for keys, give him $\mathtt{sk}_0$ because *there is only one key...*

Anamorphic Encryption

$\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ can be used

- in normal mode.
  - one public key $\mathtt{PK}$, one secret key $\mathtt{sk}$
  - one ciphertext $\mathtt{ct}$, one plaintext m

Anamorphic Encryption

$\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ can be used

- in normal mode.
  - one public key $\mathtt{PK}$, one secret key $\mathtt{sk}$
  - one ciphertext $\mathtt{ct}$, one plaintext m
- or in *anamorphic* mode: $(\mathsf{aKG}, \mathsf{aEnc}, \mathsf{aDec})$
  - one public key $\mathtt{PK}$, two secret keys $\mathtt{sk}_0, \mathtt{sk}_1$
  - one ciphertext $\mathtt{ct}$, two plaintexts m_0, m_1
  - $\mathtt{sk}_0$ decrypts $\mathtt{ct}$ to m_0 and $\mathtt{sk}_1$ decrypts $\mathtt{ct}$ to m_1

Anamorphic Encryption

$\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ can be used

- in normal mode.
  - one public key $\mathtt{PK}$, one secret key $\mathtt{sk}$
  - one ciphertext $\mathtt{ct}$, one plaintext m
- or in *anamorphic* mode: $(\mathsf{aKG}, \mathsf{aEnc}, \mathsf{aDec})$
  - one public key $\mathtt{PK}$, two secret keys $\mathtt{sk}_0, \mathtt{sk}_1$
  - one ciphertext $\mathtt{ct}$, two plaintexts m_0, m_1
  - $\mathtt{sk}_0$ decrypts $\mathtt{ct}$ to m_0 and $\mathtt{sk}_1$ decrypts $\mathtt{ct}$ to m_1

When in anamorphic mode and dictator asks for secret key

- $\mathtt{sk}_0$ is released
- dictator has no reason to believe that $\mathtt{sk}_1$ exists
- dictator can only read m_0

Implementing Our Approach

Normal mode:

- modify Enc to append a string τ of ℓ random bits
- ciphertext $\mathtt{ct} = (\mathtt{ct}_0, \tau)$
- one secret key $\mathtt{sk}$ output by KG

Implementing Our Approach

Normal mode:

- modify Enc to append a string τ of ℓ random bits
- ciphertext $\mathtt{ct} = (\mathtt{ct}_0, \tau)$
- one secret key $\mathtt{sk}$ output by KG

Anamorphic mode:

- generate a $\mathtt{sk}_1$ for $(\mathsf{KG}', \mathsf{Enc}', \mathsf{Dec}')$ encryption scheme with pseudo-random ciphertexts
- to encrypt m_0 and m_1
  - Encrypt m_0 by running Enc and obtain $\mathtt{ct}_0$
  - Encrypt m_1 by running Enc' and obtain $\mathtt{ct}_1$
  - Output $\mathtt{ct} = (\mathtt{ct}_0, \tau)$ with $\tau := \mathtt{ct}_1$

Note: In anamorphic mode there is a secret key generated by KG' shared behind the dictator's back.

This does not work!!

- We just designed an encryption scheme that is secure without assuming receiver privacy and/or sender freedom
- What is the dictator going to do?
  - It will be considered illegal
  - The simple act of using the new scheme will be self accusatory
  - The encryption scheme and its use will be seen as provocations

This does not work!!

- We just designed an encryption scheme that is secure without assuming receiver privacy and/or sender freedom
- What is the dictator going to do?
  - It will be considered illegal
  - The simple act of using the new scheme will be self accusatory
  - The encryption scheme and its use will be seen as provocations

Rather, we should look at existing *schemes to see if they can be used to defeat the dictator*

This does not work!!

- We just designed an encryption scheme that is secure without assuming receiver privacy and/or sender freedom
- What is the dictator going to do?
  - It will be considered illegal
  - The simple act of using the new scheme will be self accusatory
  - The encryption scheme and its use will be seen as provocations

Rather, we should look at existing *schemes to see if they can be used to defeat the dictator*

Existing schemes cannot be disallowed as there are legitimate uses for them. Legitimate, even for the dictator.

Our thesis

Our thesis

- Regulating/crippling encryption is technically futile
  - Not because we can construct Anamorphic Encryption
  - But because Anamorphic Encryption is already among us
- The more schemes are found to be anamorphic, the stronger our thesis

Rejection Sampling Encryption

Hopper, Langford, von Ahn [CRYPTO02]
Bellare, Paterson, Rogaway [CRYPTO14]

Normal mode

- $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ any encryption scheme
- Bob has $(\mathtt{PK}, \mathtt{sk})$ and makes $\mathtt{PK}$ public
- Alice computes $\mathtt{ct} = \mathsf{Enc}(\mathtt{PK},$ "Glory to our Leader"$)$
- Dictator decrypts $\mathtt{ct}$ using $\mathtt{sk}$

Rejection Sampling Encryption

Hopper, Langford, von Ahn [CRYPTO02]
Bellare, Paterson, Rogaway [CRYPTO14]

Normal mode

- $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ any encryption scheme
- Bob has $(\texttt{PK}, \texttt{sk})$ and makes $\texttt{PK}$ public
- Alice computes $\texttt{ct} = \mathsf{Enc}(\texttt{PK}, \text{“Glory to our Leader”})$
- Dictator decrypts $\texttt{ct}$ using $\texttt{sk}$

Anamorphic mode

- Alice and Bob share a randomly chosen seed K for a PRF $\mathcal{F}$
- Alice wants to send a bit b to Bob
  - samples $\texttt{ct} = \mathsf{Enc}(\texttt{PK}, \text{“Glory to our Leader”})$
  - until $\mathcal{F}(K, \texttt{ct}) = b$

Receiver Anamorphic Encryption Schemes: Syntax

- A receiver anamorphic scheme AME consists of schemes:
  - the *normal* scheme (AME.KG, AME.Enc, AME.Dec);
  - the *anamorphic* scheme (AME.aKG, AME.aEnc, AME.aDec);

Bob deploys AME

Normal: use (AME.KG, AME.Enc, AME.Dec) as a regular public-key encryption scheme

Anamorphic Deployment of AME for Alice

- Bob runs $(\texttt{aPK}, \texttt{ask}, \texttt{dkey}) \leftarrow \text{AME.aKG}$
- $\texttt{aPK}$ is public, $\texttt{ask}$ is given to $\mathcal{D}$, and *double key* is $\texttt{dkey}$ shared with Alice.
- Normal users use AME.Enc and $\texttt{aPK}$ to send messages to Bob.
- Alice wants to send confidential message m_1
  - Alice sets $m_0 =$ "Glory to our Leader"
  - Alice computes $\texttt{act} \leftarrow \text{AME.aEnc}(\texttt{dkey}, m_0, m_1)$
  - $\mathcal{D}$ computes $m_0 \leftarrow \text{AME.Dec}(\texttt{act}, \texttt{ask})$
  - Bob gets $m_1 \leftarrow \text{AME.aDec}(\texttt{act}, \texttt{dkey})$

Note: Alice and Bob share $\texttt{dkey}$

Rejection Sampling as AME

$\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ is the Normal scheme

The Anamorphic scheme

Key Generation: $\mathsf{aKG}(1^\lambda)$

$(\mathtt{PK}, \mathtt{sk}) \leftarrow \mathsf{KG}(1^\lambda)$ and $K \leftarrow \{0,1\}^\lambda$

$\mathtt{aPK} := \mathtt{PK}, \mathtt{ask} := \mathtt{sk}, \mathtt{dkey} := (K, \mathtt{PK})$

Anamorphic Encryption: $\mathsf{aEnc}(\mathtt{dkey}, m, b)$

sample $\mathtt{ct} \leftarrow \mathsf{Enc}(\mathtt{aPK}, m)$ until $\mathcal{F}(K, \mathtt{ct}) = b$

Anamorphic Decryption: $\mathsf{aDec}(\mathtt{ask}, \mathtt{dkey}, \mathtt{ct})$

compute $m := \mathsf{Dec}(\mathtt{ask}, \mathtt{ct})$

compute $b := \mathcal{F}(K, \mathtt{ct})$

Modes of Operations

	Key Gen.	Encryption	Decryption
Fully Anamorphic	aKG	aEnc	aDec
Anamorphic with Normal Dec	aKG	aEnc	Dec
Anamorphic with Normal Enc	aKG	Enc	Dec
Normal	KG	Enc	Dec

- The *fully anamorphic mode* to communicate privately with Alice.
- The *anamorphic mode with normal decryption* is used by $\mathcal{D}$ to decrypt an anamorphic ciphertext sent by Alice.
- The *anamorphic mode with normal encryption* is used by Charlie, unaware that Bob has an anamorphic key, to send a message to Bob.
- The *normal mode* no privacy guarantee against $\mathcal{D}$

Security notion

Normal game and **Fully Anamorphic game** are indistinguishable to $\mathcal{D}$

$\mathsf{NormalG}_{\mathsf{AME},\mathcal{D}}(\lambda)$

- Set $(\mathtt{PK}, \mathtt{sk}) \leftarrow \mathsf{AME.KG}(1^\lambda)$ and send $(\mathtt{PK}, \mathtt{sk})$ to $\mathcal{D}$.
- For $i = 1, \ldots, \mathsf{poly}(\lambda)$:
  - $\mathcal{D}$ issues query (m_0^i, m_1^i) and receives $\mathtt{ct} = \mathsf{AME.Enc}(\mathtt{PK}, m_0^i)$.
- Return $\mathcal{D}$'s output.

$\mathsf{FullyAG}_{\mathsf{AME},\mathcal{D}}(\lambda)$

- Set $(\mathtt{aPK}, \mathtt{ask}, \mathtt{dkey}) \leftarrow \mathsf{AME.aKG}(1^\lambda)$ and send $(\mathtt{aPK}, \mathtt{ask})$ to $\mathcal{D}$.
- For $i = 1, \ldots, \mathsf{poly}(\lambda)$:
  - $\mathcal{D}$ issues query (m_0^i, m_1^i) and receives $\mathtt{ct} = \mathsf{AME.aEnc}(\mathtt{dkey}, m_0^i, m_1^i)$.
- Return $\mathcal{D}$'s output.

Anamorphic Encryption Schemes

Definition

$\mathsf{AME} = ((\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec}), (\mathsf{aKG}, \mathsf{aEnc}, \mathsf{aDec}))$ is Receiver Anamorphic if

- $(\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ is a secure encryption scheme
- For any PPT $\mathcal{D}$,

$$\left|\mathrm{Prob}\left[\mathsf{NormalG}_{\mathsf{AME},\mathcal{D}}(\lambda) = 1\right] - \mathrm{Prob}\left[\mathsf{FullyAG}_{\mathsf{AME},\mathcal{D}}(\lambda) = 1\right]\right|$$

  is negligible in λ.

Anamorphic Encryption Schemes

Definition

$\mathsf{AME} = ((\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec}), (\mathsf{aKG}, \mathsf{aEnc}, \mathsf{aDec}))$ is Receiver Anamorphic if

- $(\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ is a secure encryption scheme
- For any PPT $\mathcal{D}$,

$$\left|\mathrm{Prob}\left[\mathsf{NormalG}_{\mathsf{AME},\mathcal{D}}(\lambda) = 1\right] - \mathrm{Prob}\left[\mathsf{FullyAG}_{\mathsf{AME},\mathcal{D}}(\lambda) = 1\right]\right|$$

is negligible in λ.

$(\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ is *anamorphic* if there exists $(\mathsf{aKG}, \mathsf{aEnc}, \mathsf{aDec})$ such that

$$((\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec}), (\mathsf{aKG}, \mathsf{aEnc}, \mathsf{aDec}))$$

is *anamorphic*.

Steganography

Steganography enables two parties to embed a secret conversation in a *channel conversation*. *Hopper, Langford, von Ahn [CRYPTO02]*

Stego vs Anamorphic

- Steganography works for every distribution over *channel conversations*
  - Anamorphic Encryption is Steganography for *channel conversation* consisting of ciphertexts of a secure encryption scheme for which the dictator has decryption keys.
- In Anamorphic Encryption the dictator has access to the secret keys corresponding to all public keys
  - The dictator can break the public-key steganography by von Ahn, Hopper [Eurocrypt 04]

Receiver privacy

Feasibility result

Rejection sampling encryption gives a one-bit symmetric encryption scheme whose secure does not rely on the receiver-privacy assumption.

Rate

- *Rejection Sampling* can be extended to any length ℓ
- Expected encryption time is exponential in ℓ
- If you want encryption to be polynomial, each ciphertext carries $\Theta(\log \lambda)$ hidden bits

Exploiting randomness

The Goldwasser-Micali Encryption

- Key Generation: $\mathsf{GM.KG}(1^\lambda)$
  $N = p \cdot q$,
  y, a non-square with Jacobi symbol $+1$
  $\mathtt{PK} = (N, y)$, $\mathtt{sk} = (p, q)$
- Encryption of $b \in \{0, 1\}$: $\mathsf{GM.Enc}$
  randomly select $r \leftarrow Z_N^\star$ and output $\mathtt{ct} = r^2 \cdot y^b$
- Decryption of $\mathtt{ct}$: $\mathsf{GM.Dec}$
  if $\mathtt{ct}$ is a square, output 0; else output 1

How to make GM anamorphic

Let $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ any encryption scheme with pseudorandom ciphertexts.

Key Generation: $\mathsf{aKG}(1^\lambda)$

$(\mathsf{GM}.\mathtt{PK}, \mathsf{GM}.\mathtt{sk}) \leftarrow \mathsf{GM}.\mathsf{KG}(1^\lambda)$ and $\mathtt{sk} \leftarrow \mathsf{KG}(1^\lambda)$.

$\mathtt{aPK} := \mathsf{GM}.\mathtt{PK}, \mathtt{ask} := \mathsf{GM}.\mathtt{sk}, \mathtt{dkey} := (\mathtt{sk})$

Anamorphic Encryption: $\mathsf{aEnc}(\mathtt{dkey}, b, m)$

use $\mathtt{ct} \leftarrow \mathsf{Enc}(\mathtt{sk}, m)$ as randomness r in the $\mathsf{GM}.\mathsf{Enc}$ algorithm encrypting b.

Anamorphic Decryption: $\mathsf{aDec}(\mathsf{GM}.\mathtt{sk}, \mathtt{dkey}, \mathtt{ct})$

recover r from $\mathtt{ct}$ using $\mathsf{GM}.\mathtt{sk}$ and decrypt it using $\mathtt{sk}$

Why did it work?

Randomness Recoverable Encryption

- the decryption key sk gives the plaintext and (part of) the randomness used to produce the ciphertext
- Paillier, OAEP, OAEP+, NTRU, McEliece are randomness recoverable encryption schemes

The Naor-Yung Encryption Scheme

Normal Mode

- Let $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ any encryption scheme
- Alice runs KG twice, randomly selects Σ and sets $\mathtt{PK} = (\mathtt{PK}_0, \mathtt{PK}_1, \Sigma)$ and $\mathtt{sk} = \mathtt{sk}_0$
- If Bob wants to send "Glory to our Leader" to Alice
  - Compute $\mathtt{ct}_0 = \mathsf{Enc}(\mathtt{PK}_0,$ "Glory to our Leader"$)$
  - Compute $\mathtt{ct}_1 = \mathsf{Enc}(\mathtt{PK}_1,$ "Glory to our Leader"$)$
  - Compute NIZK proof Π that $\mathtt{ct}_0$ and $\mathtt{ct}_1$ carry the same plaintext
  - Set $\mathtt{ct} = (\mathtt{ct}_0, \mathtt{ct}_1, \Pi)$
- To decrypt $\mathtt{ct}$, Alice
  - Checks Π is a valid proof
  - If valid decrypts $\mathtt{ct}_0$ using $\mathtt{sk}$

The Naor-Yung Encryption Scheme

Anamorphic Mode

- Alice runs KG twice, runs the **simulator** to get $(\Sigma, \mathtt{aux})$ and sets $\mathtt{PK} = (\mathtt{PK}_0, \mathtt{PK}_1, \Sigma)$ and $\mathtt{sk} = (\mathtt{sk}_0, \mathtt{sk}_1)$
- $\mathtt{dkey} := \mathtt{aux}$ is shared with Bob
- If Bob wants to send $m_0 =$ "Glory to our Leader" to the dictator and $m_1 =$ "F*** our Leader" to Alice
  - Compute $\mathtt{ct}_0 = \mathsf{Enc}(\mathtt{PK}_0, \text{"Glory to our Leader"})$
  - Compute $\mathtt{ct}_1 = \mathsf{Enc}(\mathtt{PK}_1, \text{"F*** our Leader"})$
  - Simulate NIZK proof Π that $\mathtt{ct}_0$ and $\mathtt{ct}_1$ carry the same plaintext
  - Set $\mathtt{ct} = (\mathtt{ct}_0, \mathtt{ct}_1, \Pi)$
- To decrypt $\mathtt{ct}$, Alice uses $\mathtt{sk}_1$ to decrypt $\mathtt{ct}_1$
- If asked to surrender her secret key, Alice gives $\mathtt{sk}_0$
  - The dictator verifies Π, decrypts $\mathtt{ct}_0$ and reads $m_0 =$ "Glory to our Leader"

Why does this work?

Informal

- NIZK implies that the anamorphic and the normal public keys are indistinguishable
- NIZK+IND CPA imply ciphertexts are indistinguishable
- If asked to surrender secret key, Alice gives $\mathtt{sk}_0$
  - $\mathtt{PK}_1$ could be generated without the associated secret key (e.g., El Gamal has this property)
- $(\mathtt{PK}_0, \mathtt{PK}_1, \Sigma, \mathtt{aux})$ is a symmetric encryption key

Same reasoning applies to [DDN91] and [Sahai99]

The Koppula-Waters Encryption Scheme CRYPTO '19

- Key Generation: $\mathsf{kw.KG}(1^\lambda)$
  - Generate 2λ pairs $(\mathtt{PK}_{bi}, \mathtt{sk}_{bi})$, $b \in \{0,1\}, i \in \{1,\ldots,n\}$
  - Randomly select $a_1,\ldots,a_n \leftarrow \{0,1\}^\lambda$ and $B \leftarrow \{0,1\}^\lambda$
  - Set $\mathsf{kw}.\mathtt{PK} = \left(B, (a_i, \mathtt{PK}_{0i}, \mathtt{PK}_{1i})_{i=1}^\lambda\right)$ and $\mathsf{kw}.\mathtt{sk} = (\mathtt{sk}_{0i})_{i=1}^\lambda$
- Encryption: $\mathsf{kw.Enc}(\mathsf{kw}.\mathtt{PK}, m)$
  - randomly select $K \leftarrow \{0,1\}^\lambda$ and $(\mathtt{sigK}, \mathtt{vK}) \leftarrow \mathsf{Sign.KG}(1^\lambda)$
  - set $c = \mathcal{F}(K,0) \oplus m$
  - for $i = 1,\ldots,\lambda$
    - ★ $\tilde{r}_i = \mathcal{F}(K,i)$ and $v_i \leftarrow \{0,1\}^{\lambda-1}$
    - ★ if $K_i = 0$
      $c_{0,i} = \mathsf{Enc}(\mathtt{PK}_{0i}, 1|v_i;\tilde{r}_i)$, $c_{1,i} = \mathsf{Enc}(\mathtt{PK}_{1i}, 0^\lambda)$, $c_{2,i} = G(v_i)$
    - ★ if $K_i = 1$
      $c_{0,i} = \mathsf{Enc}(\mathtt{PK}_{0i}, 0^\lambda)$, $c_{1,i} = \mathsf{Enc}(\mathtt{PK}_{1i}, 1|v_i;\tilde{r}_i)$,
      $c_{2,i} = G(v_i) + a_i + B \cdot \mathtt{vK}$
  - Sign $(c, (c_{0,i}, c_{1,i}, c_{2,i}))$ using $\mathtt{sigK}$

The Koppula-Waters Encryption Scheme CRYPTO '19

- Key Generation: $\mathsf{kw.KG}(1^\lambda)$
  - Generate 2λ pairs $(\mathtt{PK}_{bi}, \mathtt{sk}_{bi})$, $b \in \{0,1\}, i \in \{1,\ldots,n\}$
  - Randomly select $a_1,\ldots,a_n \leftarrow \{0,1\}^\lambda$ and $B \leftarrow \{0,1\}^\lambda$
  - Set $\mathsf{kw}.\mathtt{PK} = \left(B, (a_i, \mathtt{PK}_{0i}, \mathtt{PK}_{1i})_{i=1}^{\lambda}\right)$ and $\mathsf{kw}.\mathtt{sk} = (\mathtt{sk}_{0i})_{i=1}^{\lambda}$
- Encryption: $\mathsf{kw.Enc}(\mathsf{kw}.\mathtt{PK}, m)$
  - randomly select $K \leftarrow \{0,1\}^\lambda$ and $(\mathtt{sigK}, \mathtt{vK}) \leftarrow \mathsf{Sign.KG}(1^\lambda)$
  - set $c = \mathcal{F}(K, 0) \oplus m$
  - for $i = 1,\ldots,\lambda$
    - ★ $\tilde{r}_i = \mathcal{F}(K, i)$ and $v_i \leftarrow \{0,1\}^{\lambda-1}$
    - ★ if $K_i = 0$
      $c_{0,i} = \mathsf{Enc}(\mathtt{PK}_{0i}, 1|v_i; \tilde{r}_i)$, $c_{1,i} = \mathsf{Enc}(\mathtt{PK}_{1i}, 0^\lambda)$, $c_{2,i} = G(v_i)$
    - ★ if $K_i = 1$
      $c_{0,i} = \mathsf{Enc}(\mathtt{PK}_{0i}, 0^\lambda)$, $c_{1,i} = \mathsf{Enc}(\mathtt{PK}_{1i}, 1|v_i; \tilde{r}_i)$,
      $c_{2,i} = G(v_i) + a_i + B \cdot \mathtt{vK}$
  - Sign $(c, (c_{0,i}, c_{1,i}, c_{2,i}))$ using $\mathtt{sigK}$

Obs0: there are 2λ public keys

The Koppula-Waters Encryption Scheme CRYPTO '19

- Key Generation: $\mathsf{kw.KG}(1^\lambda)$
  - Generate 2λ pairs $(\mathtt{PK}_{bi}, \mathtt{sk}_{bi})$, $b \in \{0,1\}, i \in \{1,\ldots,n\}$
  - Randomly select $a_1,\ldots,a_n \leftarrow \{0,1\}^\lambda$ and $B \leftarrow \{0,1\}^\lambda$
  - Set $\mathsf{kw}.\mathtt{PK} = \left(B, (a_i, \mathtt{PK}_{0i}, \mathtt{PK}_{1i})_{i=1}^\lambda\right)$ and $\mathsf{kw}.\mathtt{sk} = (\mathtt{sk}_{0i})_{i=1}^\lambda$
- Encryption: $\mathsf{kw.Enc}(\mathsf{kw}.\mathtt{PK}, m)$
  - randomly select $K \leftarrow \{0,1\}^\lambda$ and $(\mathtt{sigK}, \mathtt{vK}) \leftarrow \mathsf{Sign.KG}(1^\lambda)$
  - set $c = \mathcal{F}(K,0) \oplus m$
  - for $i = 1,\ldots,\lambda$
    - ★ $\tilde{r}_i = \mathcal{F}(K,i)$ and $v_i \leftarrow \{0,1\}^{\lambda-1}$
    - ★ if $K_i = 0$
      $c_{0,i} = \mathsf{Enc}(\mathtt{PK}_{0i}, 1|v_i; \tilde{r}_i)$, $c_{1,i} = \mathsf{Enc}(\mathtt{PK}_{1i}, 0^\lambda)$, $c_{2,i} = G(v_i)$
    - ★ if $K_i = 1$
      $c_{0,i} = \mathsf{Enc}(\mathtt{PK}_{0i}, 0^\lambda)$, $c_{1,i} = \mathsf{Enc}(\mathtt{PK}_{1i}, 1|v_i; \tilde{r}_i)$,
      $c_{2,i} = G(v_i) + a_i + B \cdot \mathtt{vK}$
  - Sign $(c, (c_{0,i}, c_{1,i}, c_{2,i}))$ using $\mathtt{sigK}$

Obs1: dictator has only λ secret keys $\mathtt{sk}_{0i}$

The Koppula-Waters Encryption Scheme CRYPTO '19

- Key Generation: $\mathsf{kw.KG}(1^\lambda)$
  - Generate 2λ pairs $(\mathtt{PK}_{bi}, \mathtt{sk}_{bi})$, $b \in \{0,1\}, i \in \{1,\ldots,n\}$
  - Randomly select $a_1,\ldots,a_n \leftarrow \{0,1\}^\lambda$ and $B \leftarrow \{0,1\}^\lambda$
  - Set $\mathsf{kw}.\mathtt{PK} = \left(B, (a_i, \mathtt{PK}_{0i}, \mathtt{PK}_{1i})_{i=1}^{\lambda}\right)$ and $\mathsf{kw}.\mathtt{sk} = (\mathtt{sk}_{0i})_{i=1}^{\lambda}$
- Encryption: $\mathsf{kw.Enc}(\mathsf{kw}.\mathtt{PK}, m)$
  - randomly select $K \leftarrow \{0,1\}^\lambda$ and $(\mathtt{sigK}, \mathtt{vK}) \leftarrow \mathsf{Sign.KG}(1^\lambda)$
  - set $c = \mathcal{F}(K, 0) \oplus m$
  - for $i = 1,\ldots,\lambda$
    - ★ $\tilde{r}_i = \mathcal{F}(K, i)$ and $v_i \leftarrow \{0,1\}^{\lambda-1}$
    - ★ if $K_i = 0$
      $c_{0,i} = \mathsf{Enc}(\mathtt{PK}_{0i}, 1|v_i; \tilde{r}_i)$, $c_{1,i} = \mathsf{Enc}(\mathtt{PK}_{1i}, 0^\lambda)$, $c_{2,i} = G(v_i)$
    - ★ if $K_i = 1$
      $c_{0,i} = \mathsf{Enc}(\mathtt{PK}_{0i}, 0^\lambda)$, $c_{1,i} = \mathsf{Enc}(\mathtt{PK}_{1i}, 1|v_i; \tilde{r}_i)$, $c_{2,i} = G(v_i) + a_i + B \cdot \mathtt{vK}$
  - Sign $(c, (c_{0,i}, c_{1,i}, c_{2,i}))$ using $\mathtt{sigK}$

Obs2: dictator can decrypt all the $c_{0,i}$ and learn K

The Koppula-Waters Encryption Scheme CRYPTO '19

- Key Generation: $\mathsf{kw.KG}(1^\lambda)$
  - Generate 2λ pairs $(\mathtt{PK}_{bi}, \mathtt{sk}_{bi})$, $b \in \{0,1\}, i \in \{1,\ldots,n\}$
  - Randomly select $a_1,\ldots,a_n \leftarrow \{0,1\}^\lambda$ and $B \leftarrow \{0,1\}^\lambda$
  - Set $\mathsf{kw}.\mathtt{PK} = \left(B, (a_i, \mathtt{PK}_{0i}, \mathtt{PK}_{1i})_{i=1}^{\lambda}\right)$ and $\mathsf{kw}.\mathtt{sk} = (\mathtt{sk}_{0i})_{i=1}^{\lambda}$
- Encryption: $\mathsf{kw.Enc}(\mathsf{kw}.\mathtt{PK}, m)$
  - randomly select $K \leftarrow \{0,1\}^\lambda$ and $(\mathtt{sigK}, \mathtt{vK}) \leftarrow \mathsf{Sign.KG}(1^\lambda)$
  - set $c = \mathcal{F}(K, 0) \oplus m$
  - for $i = 1, \ldots, \lambda$
    - ★ $\tilde{r}_i = \mathcal{F}(K, i)$ and $v_i \leftarrow \{0,1\}^{\lambda-1}$
    - ★ if $K_i = 0$
      $c_{0,i} = \mathsf{Enc}(\mathtt{PK}_{0i}, 1|v_i; \tilde{r}_i)$, $c_{1,i} = \mathsf{Enc}(\mathtt{PK}_{1i}, 0^\lambda)$, $c_{2,i} = G(v_i)$
    - ★ if $K_i = 1$
      $c_{0,i} = \mathsf{Enc}(\mathtt{PK}_{0i}, 0^\lambda)$, $c_{1,i} = \mathsf{Enc}(\mathtt{PK}_{1i}, 1|v_i; \tilde{r}_i)$, $c_{2,i} = G(v_i) + a_i + B \cdot \mathtt{vK}$
  - Sign $(c, (c_{0,i}, c_{1,i}, c_{2,i}))$ using $\mathtt{sigK}$

Obs3: dictator can obtain all the $\tilde{r}_i$

The Koppula-Waters Encryption Scheme CRYPTO '19

- Key Generation: kw.KG(1^λ)
  - Generate 2λ pairs $(\mathtt{PK}_{bi}, \mathtt{sk}_{bi})$, $b \in \{0,1\}, i \in \{1,\ldots,n\}$
  - Randomly select $a_1,\ldots,a_n \leftarrow \{0,1\}^\lambda$ and $B \leftarrow \{0,1\}^\lambda$
  - Set kw.PK $= \left(B, (a_i, \mathtt{PK}_{0i}, \mathtt{PK}_{1i})_{i=1}^{\lambda}\right)$ and kw.sk $= (\mathtt{sk}_{0i})_{i=1}^{\lambda}$
- Encryption: kw.Enc(kw.PK, m)
  - randomly select $K \leftarrow \{0,1\}^\lambda$ and $(\mathtt{sigK}, \mathtt{vK}) \leftarrow$ Sign.KG(1^λ)
  - set $c = \mathcal{F}(K, 0) \oplus m$
  - for $i = 1,\ldots,\lambda$
    - ⋆ $\tilde{r}_i = \mathcal{F}(K,i)$ and $v_i \leftarrow \{0,1\}^{\lambda-1}$
    - ⋆ if $K_i = 0$
      $c_{0,i} = \mathrm{Enc}(\mathtt{PK}_{0i}, 1|v_i;\tilde{r}_i)$, $c_{1,i} = \mathrm{Enc}(\mathtt{PK}_{1i}, 0^\lambda)$, $c_{2,i} = G(v_i)$
    - ⋆ if $K_i = 1$
      $c_{0,i} = \mathrm{Enc}(\mathtt{PK}_{0i}, 0^\lambda)$, $c_{1,i} = \mathrm{Enc}(\mathtt{PK}_{1i}, 1|v_i;\tilde{r}_i)$, $c_{2,i} = G(v_i) + a_i + B \cdot \mathtt{vK}$
  - Sign $(c, (c_{0,i}, c_{1,i}, c_{2,i}))$ using sigK

Obs4: these are semantically secure w.r.t. dictator

Making KW19 Anamorphic

Anamorphic key generation

- keep all $\mathtt{sk}_{1i}$

Anamorphic Encryption

How to encrypt:

- $m_0 =$ "Glory to our Leader"
- $m_1 =$ "F*** our Leader"

1. Use kw.Enc to encrypt m_0
2. Let i be such that $K_i = 0$
   - set $c_{1,i} = \mathsf{Enc}(\mathtt{PK}_{1i}, m_1)$

Making KW19 Anamorphic

Anamorphic key generation

- keep all $\mathtt{sk}_{1i}$

Anamorphic Encryption

How to encrypt:

- $m_0 =$ "Glory to our Leader"
- $m_1 =$ "F*** our Leader"

1. Use kw.Enc to encrypt m_0
2. Let i be such that $K_i = 0$
   - set $c_{1,i} = \mathsf{Enc}(\mathtt{PK}_{1i}, m_1)$

Note1: $\Theta(\lambda)$ messages can be sent with v.h.p.

Note2: No shared information!!!

Receiver-Privacy Assumption

- If sender and receiver have a shared secret
  - every encryption scheme can be made anamorphic with logarithmic rate
  - every Randomness Recoverable Encryption can be made anamorphic with rate depending on the amount of randomess recovered ElGamal, Cramer-Shoup, GM, RSA-OAEP
  - the NIZK based CCA secure encryption schemes à la Naor-Yung can be made anamorphic with constant rate
- If sender and receiver have no shared secret
  - the Koppula-Waters encryption scheme can be made anamorphic with rate greater > 1.

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

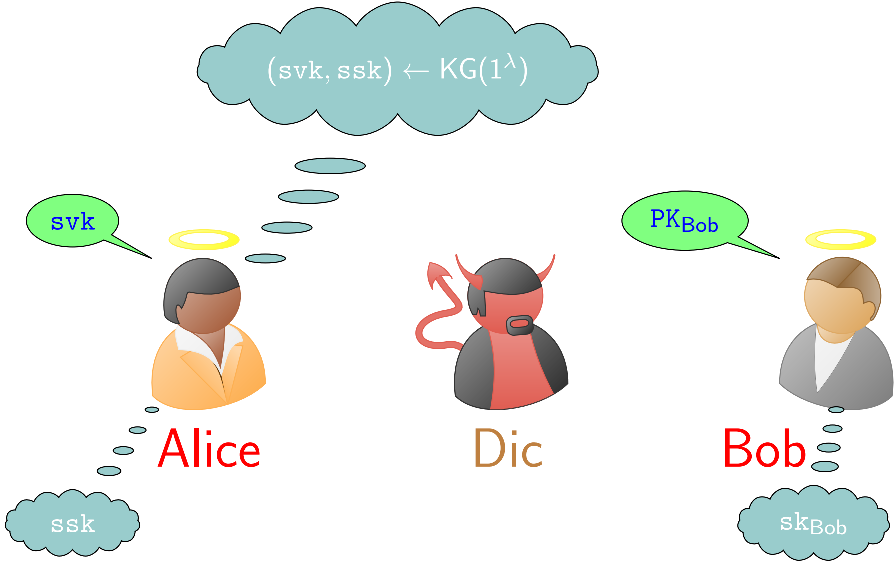
Futile, you said?

Encryption is declared illegal

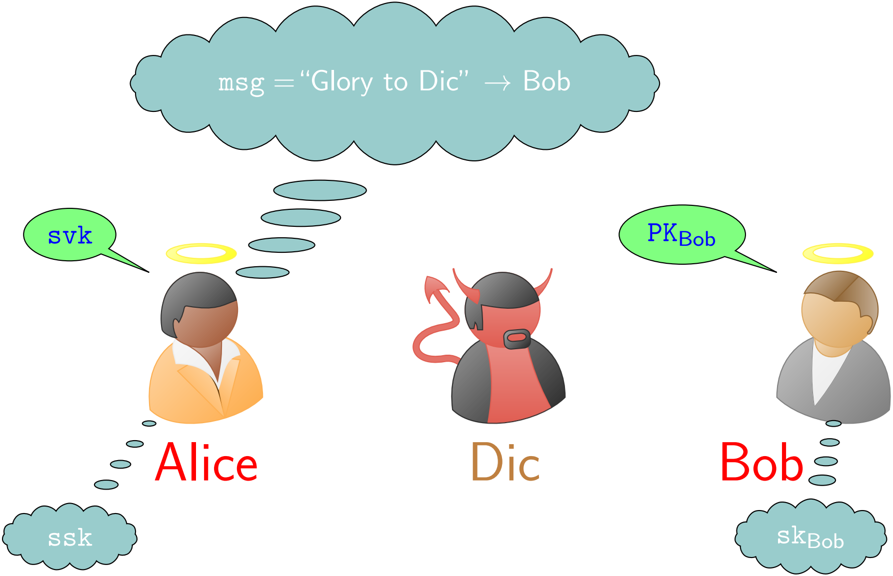
- the dictator mandates that all communication happens through a central hub
- messages can only be digitally signed
  - so that we know whom we are talking to

Do not annoy your dictator!

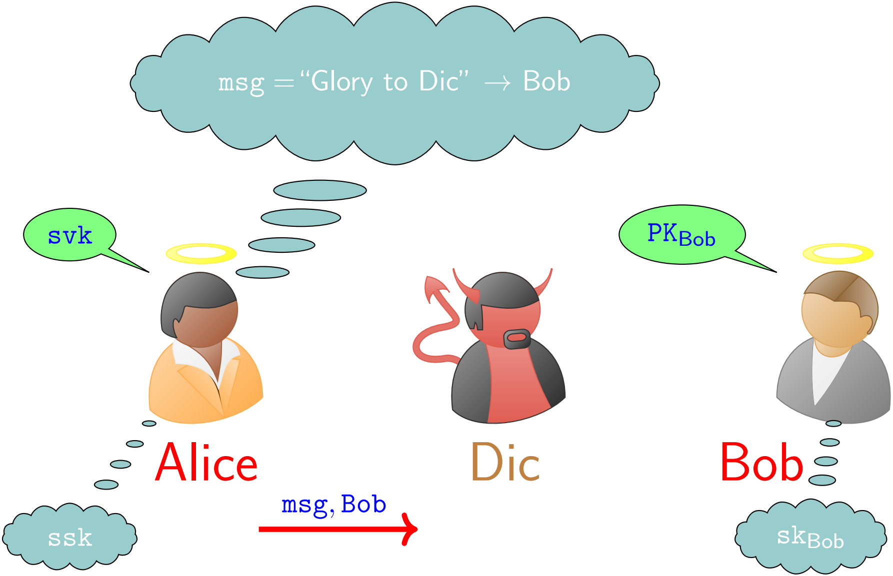
The new dictatorial setting



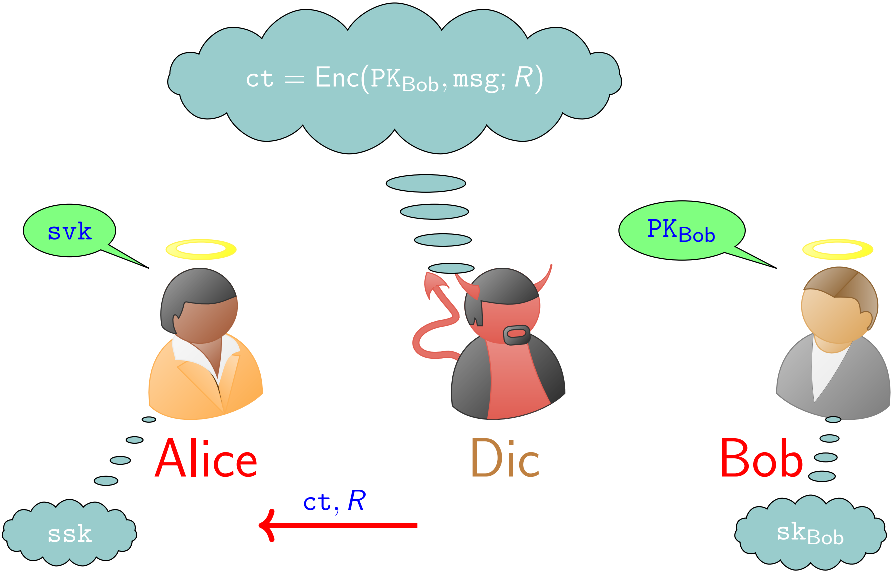
The new dictatorial setting



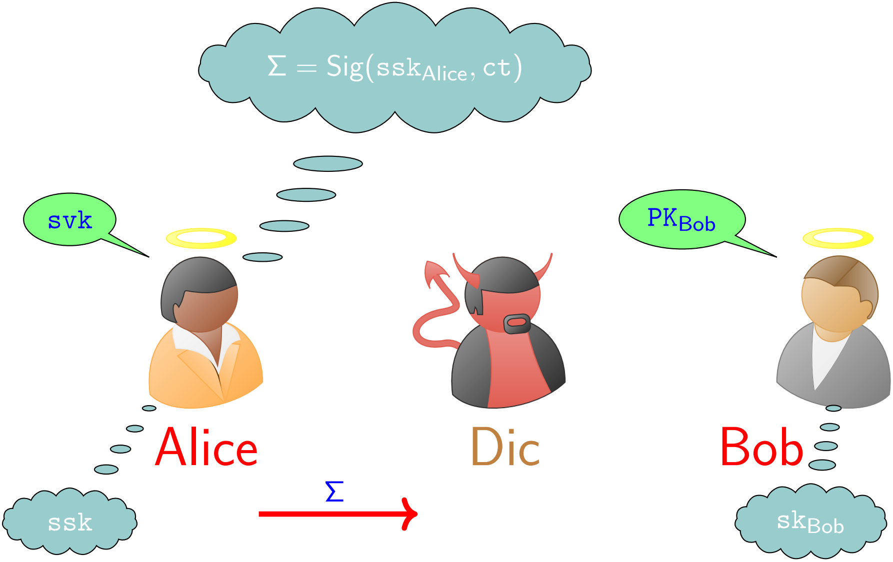
The new dictatorial setting



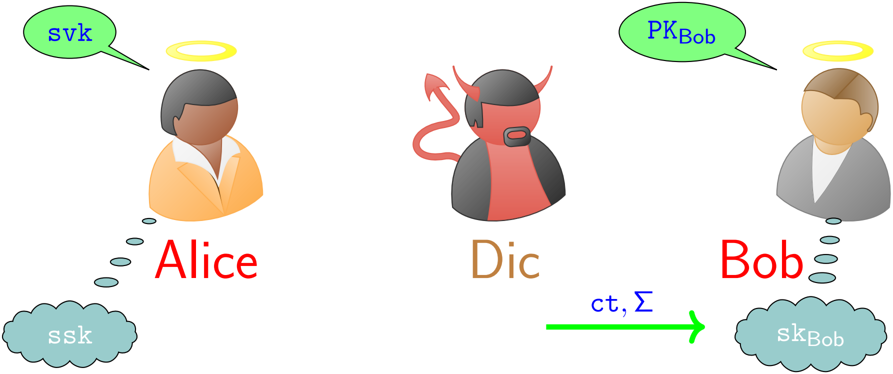
The new dictatorial setting



The new dictatorial setting



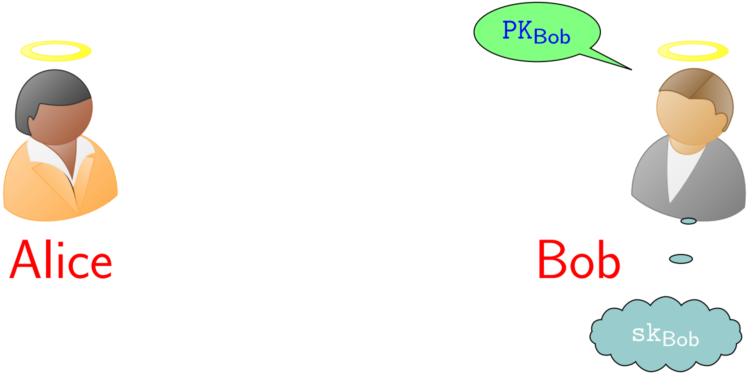
The new dictatorial setting



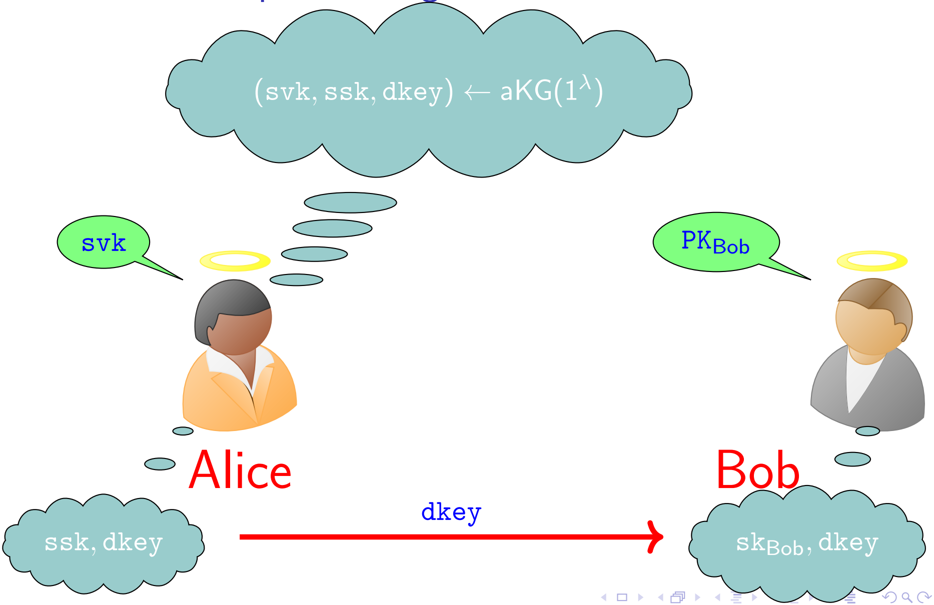
Dictator's thinking

- 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

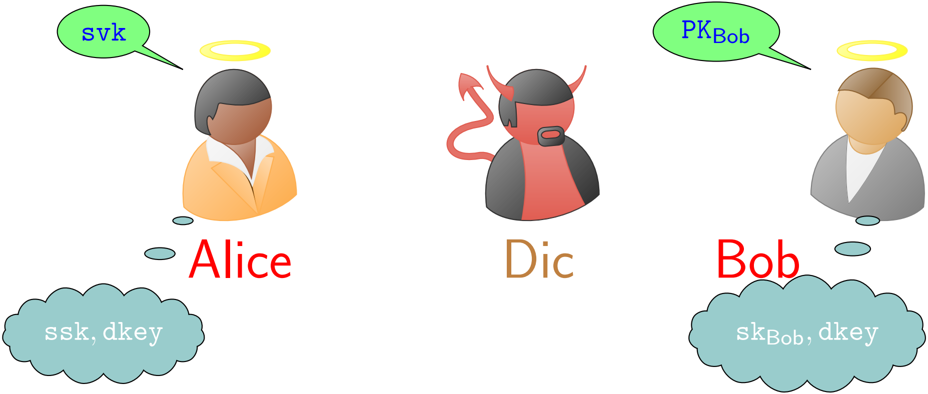
The new anamorphic setting



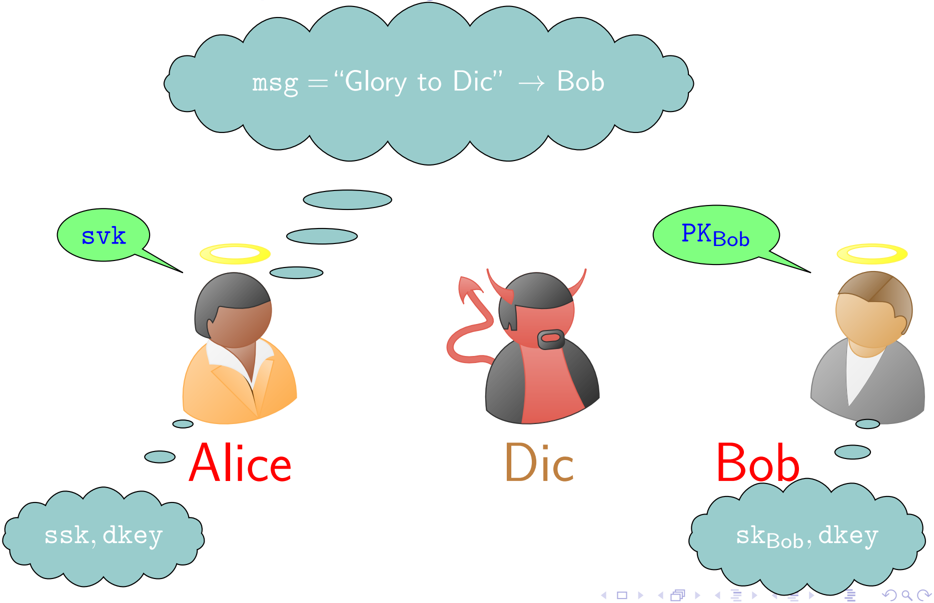
The new anamorphic setting



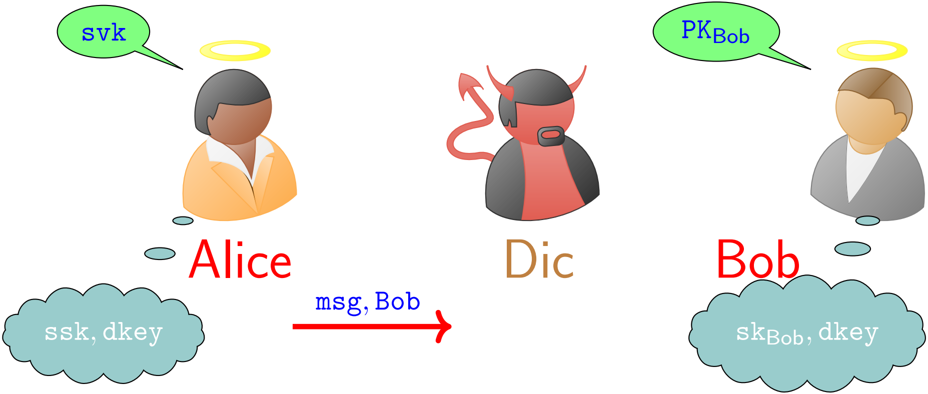
The new anamorphic setting



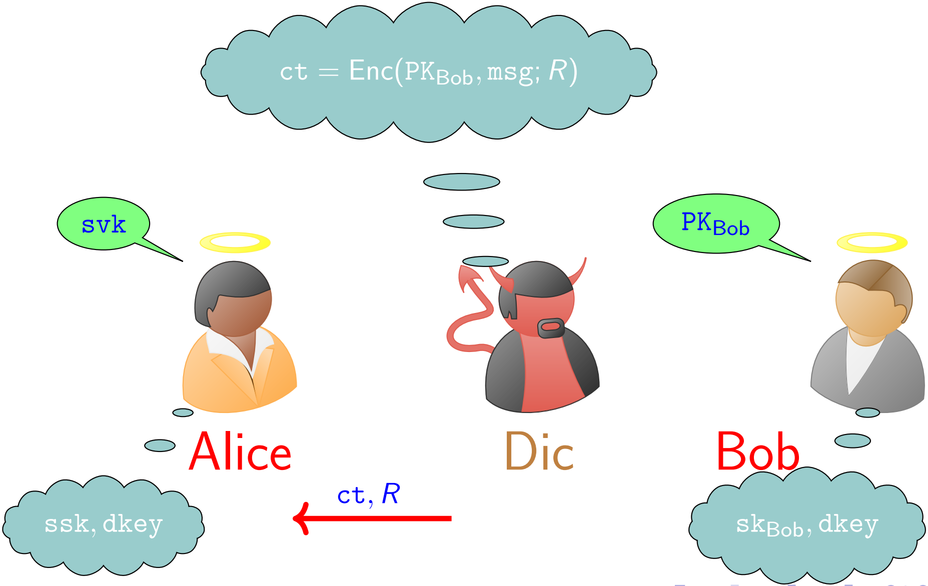
The new anamorphic setting



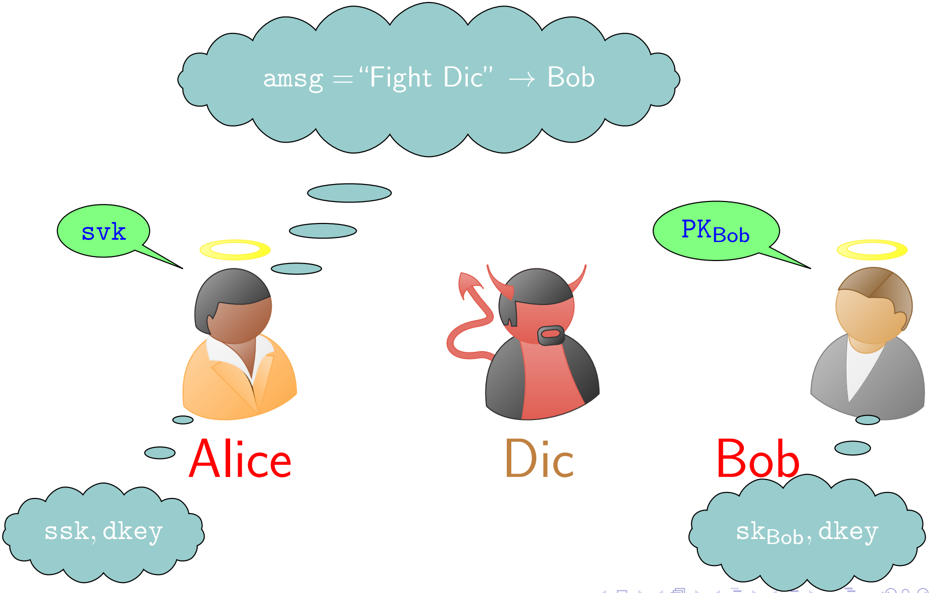
The new anamorphic setting



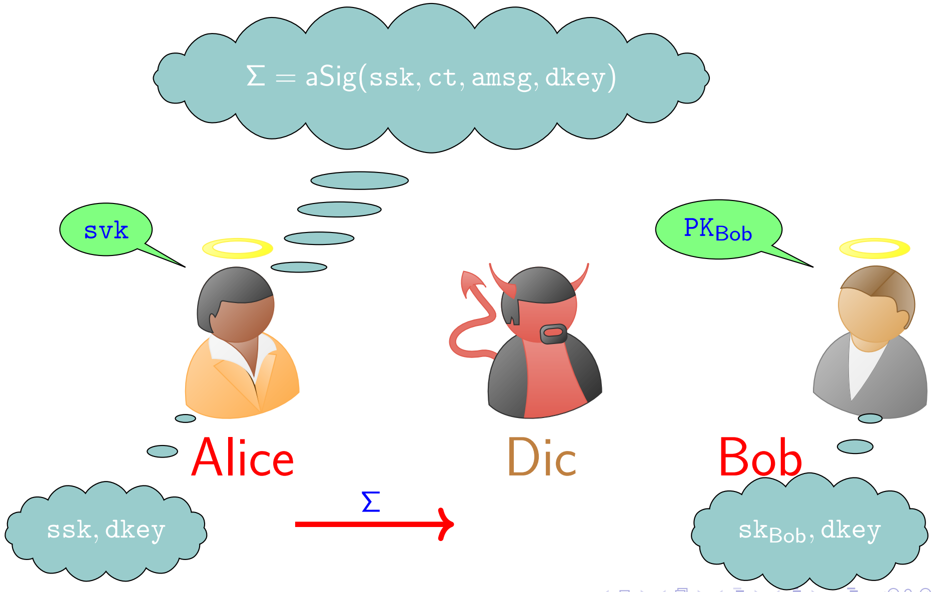
The new anamorphic setting



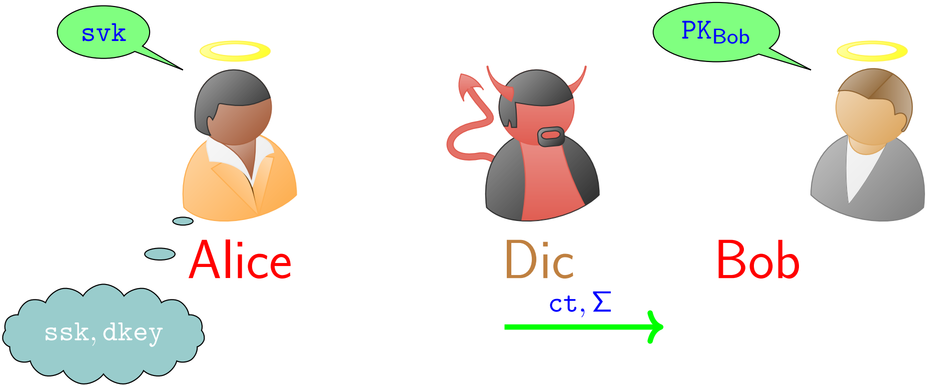
The new anamorphic setting



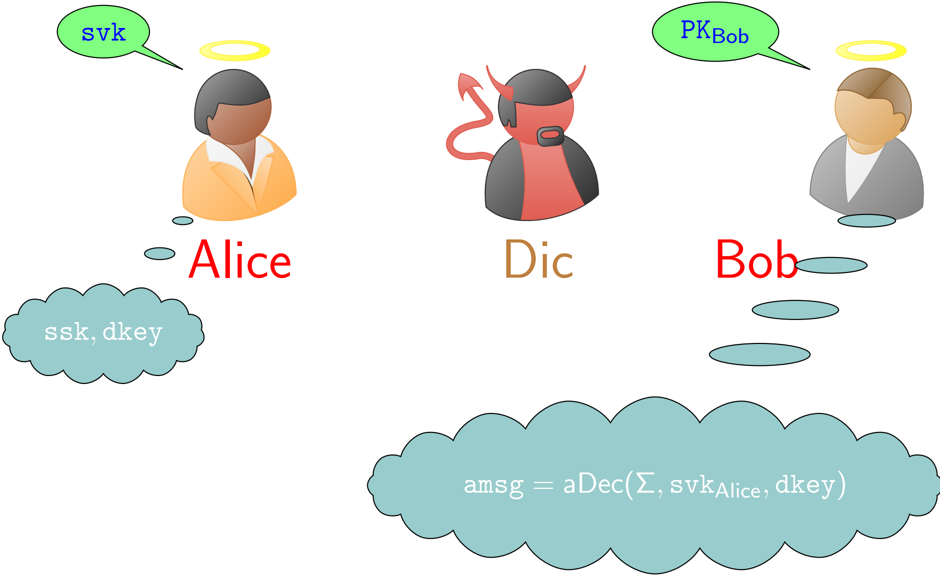
The new anamorphic setting



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 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 $\mathsf{aDec}(\Sigma, \mathtt{svk}, \mathtt{dkey})$
  - $\mathtt{amsg}$.

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

Sender Anamorphic Encryption

The story of Oscar and John

- Oscar, an opposition leader, is "asked" by the Leader to send the following message to some media outlet

  $m_0 =$ "I am fine and in good health"

  to a forced public key fPK

Sender Anamorphic Encryption

The story of Oscar and John

- Oscar, an opposition leader, is "asked" by the Leader to send the following message to some media outlet

  $$m_0 = \text{"I am fine and in good health"}$$

  to a forced public key `fPK`
- Oscar wants also to send message

  $$m_1 = \text{"I am in prison"}$$

  to the public key `dPK` of a journalist John

Sender Anamorphic Encryption

The story of Oscar and John

- Oscar, an opposition leader, is "asked" by the Leader to send the following message to some media outlet
  $$m_0 = \text{"I am fine and in good health"}$$
  to a forced public key $\texttt{fPK}$
- Oscar wants also to send message
  $$m_1 = \text{"I am in prison"}$$
  to the public key $\texttt{dPK}$ of a journalist John
- Oscar computes special coin tosses $R^\star$ such that by setting $\texttt{ct} = \mathsf{Enc}(\texttt{fPK}, m_0; R^\star)$ it holds that
  $$m_1 = \mathsf{Dec}(\texttt{dsk}, \texttt{ct})$$

Sender Anamorphic Encryption

The story of Oscar and John

- Oscar, an opposition leader, is "asked" by the Leader to send the following message to some media outlet
$$m_0 = \text{"I am fine and in good health"}$$
to a forced public key $\mathtt{fPK}$
- Oscar wants also to send message
$$m_1 = \text{"I am in prison"}$$
to the public key $\mathtt{dPK}$ of a journalist John
- Oscar computes special coin tosses $R^\star$ such that by setting $\mathtt{ct} = \mathsf{Enc}(\mathtt{fPK}, m_0; R^\star)$ it holds that
$$m_1 = \mathsf{Dec}(\mathtt{dsk}, \mathtt{ct})$$

Sender Anamorphic Encryption

The story of Oscar and John

- Oscar, an opposition leader, is "asked" by the Leader to send the following message to some media outlet

  $$m_0 = \text{"I am fine and in good health"}$$

  to a forced public key $\texttt{fPK}$

- Oscar wants also to send message

  $$m_1 = \text{"I am in prison"}$$

  to the public key $\texttt{dPK}$ of a journalist John

- Oscar computes special coin tosses $R^\star$ such that by setting $\texttt{ct} = \mathsf{Enc}(\texttt{fPK}, m_0; R^\star)$ it holds that

  $$m_1 = \mathsf{Dec}(\texttt{dsk}, \texttt{ct})$$

No prior shared knowledge is needed between Oscar and John

Sender Anamorphic vs Deniable Encryption

Deniable encryption:

- applies to the *same* public key
- is not suitable for dictator setting: It was mentioned in [CDNO97] that deniability is impossible where "*Eve [the adversary] approaches Alice [the sender]* before *the transmission and requires Alice [the sender] to send specific messages*".
- is impossible for a standard encryptions [CDNO97] (This contradicts our objective to use standard encryptions).

Sender Anamorphic vs Deniable Encryption

Deniable encryption:

- applies to the *same* public key
- is not suitable for dictator setting: It was mentioned in [CDNO97] that deniability is impossible where "*Eve [the adversary] approaches Alice [the sender]* before *the transmission and requires Alice [the sender] to send specific messages*".
- is impossible for a standard encryptions [CDNO97] (This contradicts our objective to use standard encryptions).

Sender Anamorphic Encryption can be used to provide some form of deniability

- ciphertext is now broadcast over a public channel and not sent on a point to point channel
- denying having sent a message m to John under the ciphertext $\mathtt{ct}$, by proving that $\mathtt{ct}$ corresponds to a message m' sent to Carol.

Sufficient conditions for Sender Anamorphic with no shared key

Any PKE satisfying the 3 following conditions is sender anamorphic.

1. *Common randomness property*.
   For any $c = \text{Enc}(\texttt{PK}, m, r)$ and any $\texttt{PK}'$, there is a m' such that $c = \text{Enc}(\texttt{PK}', m', r)$
2. *Message recovery from randomness*.
   Given the ciphertext and the used randomness, one can recover the corresponding message.
3. *Equal Distribution of Plaintexts*.
   Given any c in the ciphertext space, for a randomly generated secret key sk: $Pr[\text{Dec}(sk, c) = 0] = Pr[\text{Dec}(sk, c) = 1]$

Sufficient conditions for Sender Anamorphic with no shared key

Any PKE satisfying the 3 following conditions is sender anamorphic.

1. *Common randomness property*.
   For any $c = \mathsf{Enc}(\mathtt{PK}, m, r)$ and any $\mathtt{PK}'$, there is a m' such that $c = \mathsf{Enc}(\mathtt{PK}', m', r)$
2. *Message recovery from randomness*.
   Given the ciphertext and the used randomness, one can recover the corresponding message.
3. *Equal Distribution of Plaintexts*.
   Given any c in the ciphertext space, for a randomly generated secret key sk: $Pr[\mathsf{Dec}(sk, c) = 0] = Pr[\mathsf{Dec}(sk, c) = 1]$

Consequently:

- LWE encryption by Regev, 2005
- Dual LWE encryption by Gentry, Peikert, and Vaikuntanathan, 2008

are sender anamorphic encryption schemes.

Conclusions

- We introduced two new concepts:
  - receiver anamorphic encryption
    the receiver of a communication is under the dictator's control
  - sender anamorphic encryption
    the sender of a message is under the dictator's control
- Anamorphic encryption is not an isolated phenomenon.
- Our results gives technical evidence of the futility of the Crypto Wars
  - the dictator doomed to read Crypto papers and outlaw schemes as they are shown to be *anamorphic*
- How this is going to affect policy, law and other societal aspects is beyond the scope of this work

Thank You