Broadcast Steganography
or
How to Broadcast a Secret *Covertly*

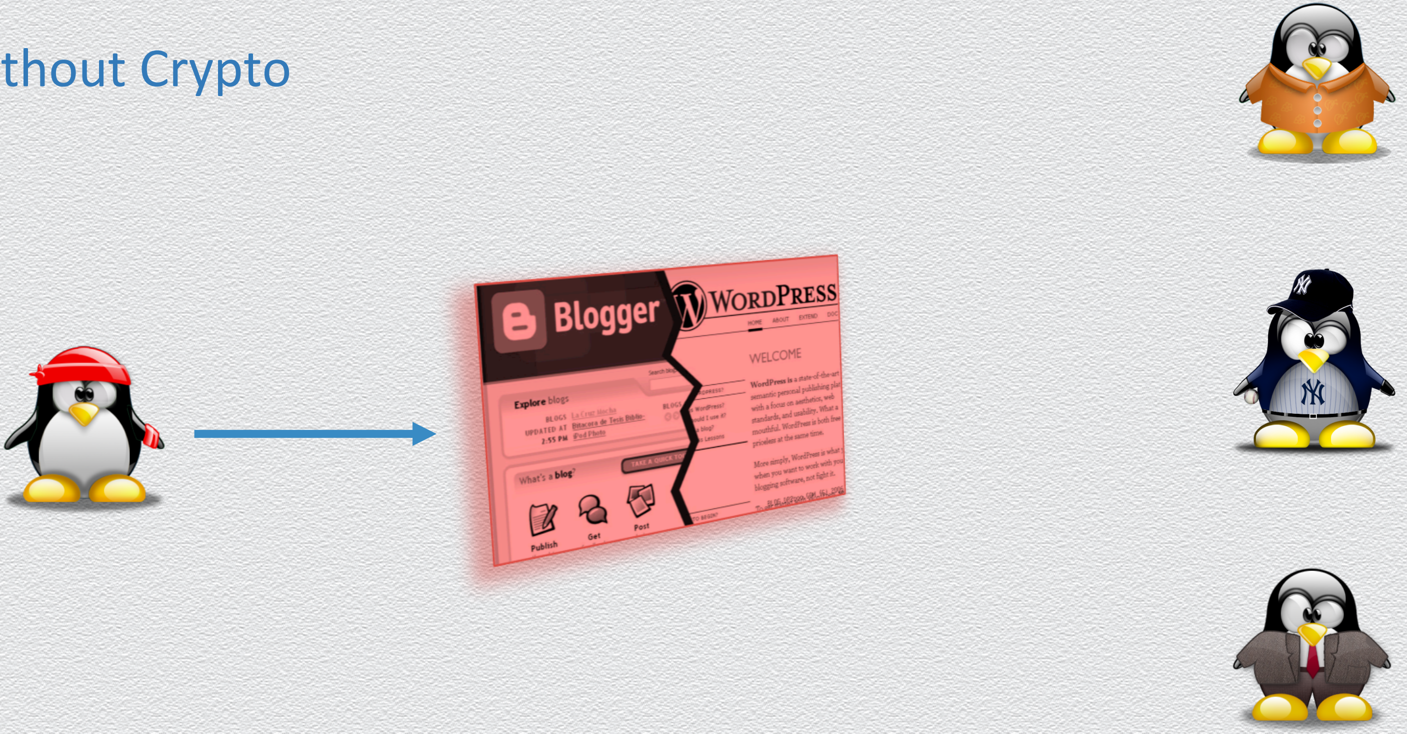
Nelly Fazio
The City College of CUNY
fazio@cs.ccny.cuny.edu

Antonio R. Nicolosi
Stevens Institute of Technology
nicolosi@cs.stevens.edu

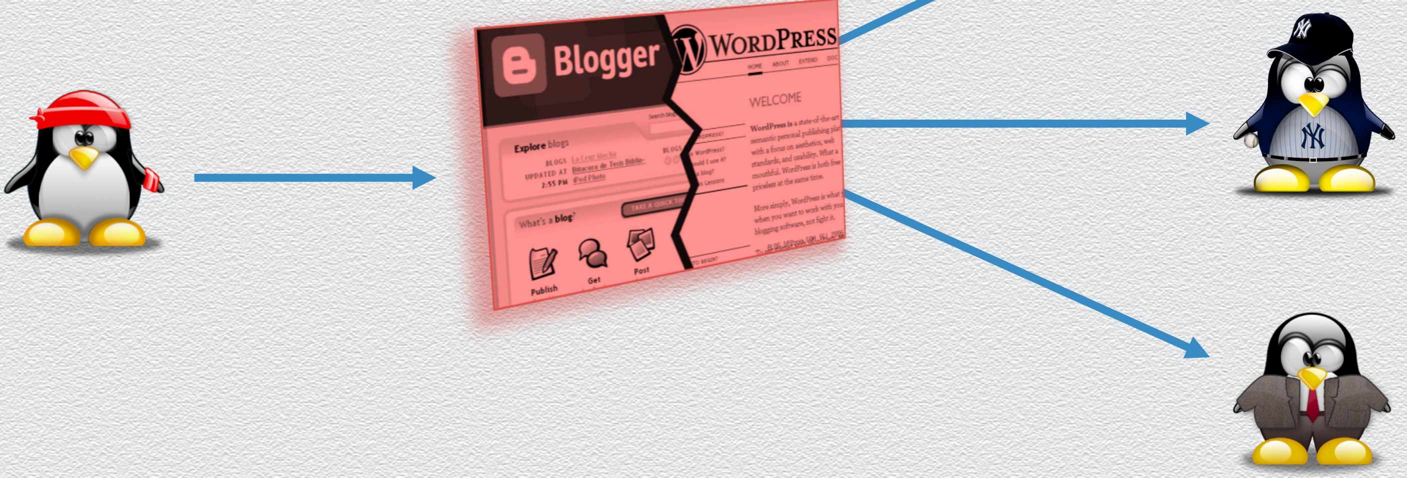
Irippuge Milinda Perera
The Graduate Center of CUNY
iperera@gc.cuny.edu

NY
NY

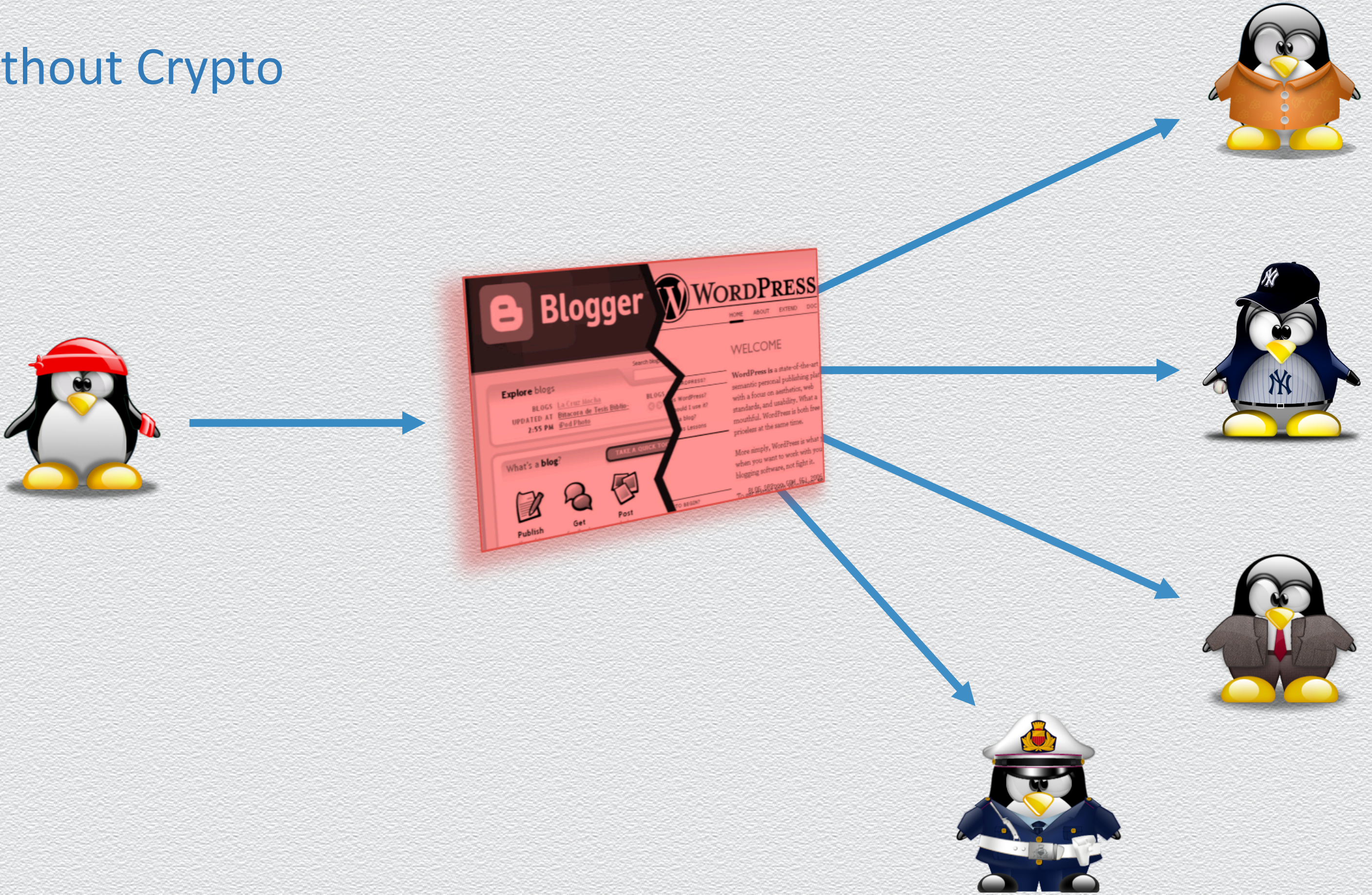
Without Crypto



Without Crypto



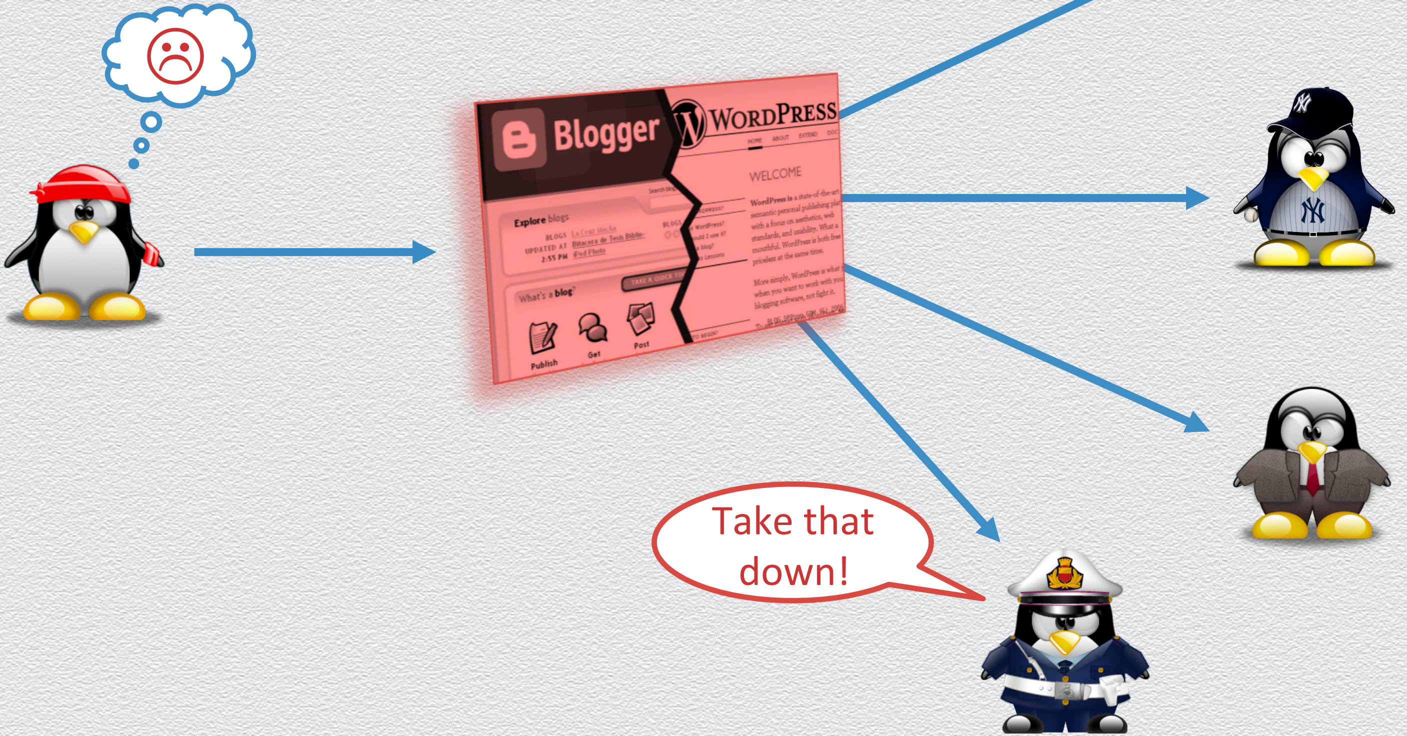
Without Crypto



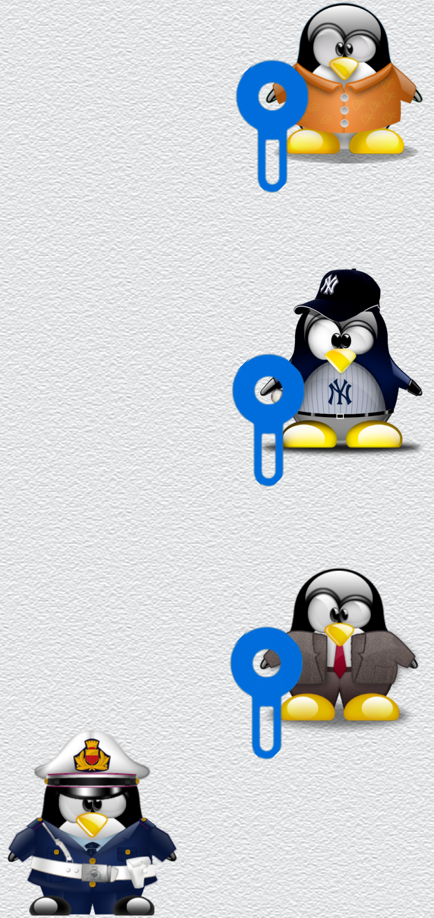
WordPress is a state-of-the-art semantic personal publishing plat with a focus on aesthetics, web standards, and usability. What a mouthful. WordPress is both free priceless at the same time.

More simply, WordPress is what y when you want to work with you blogging software, not fight it.

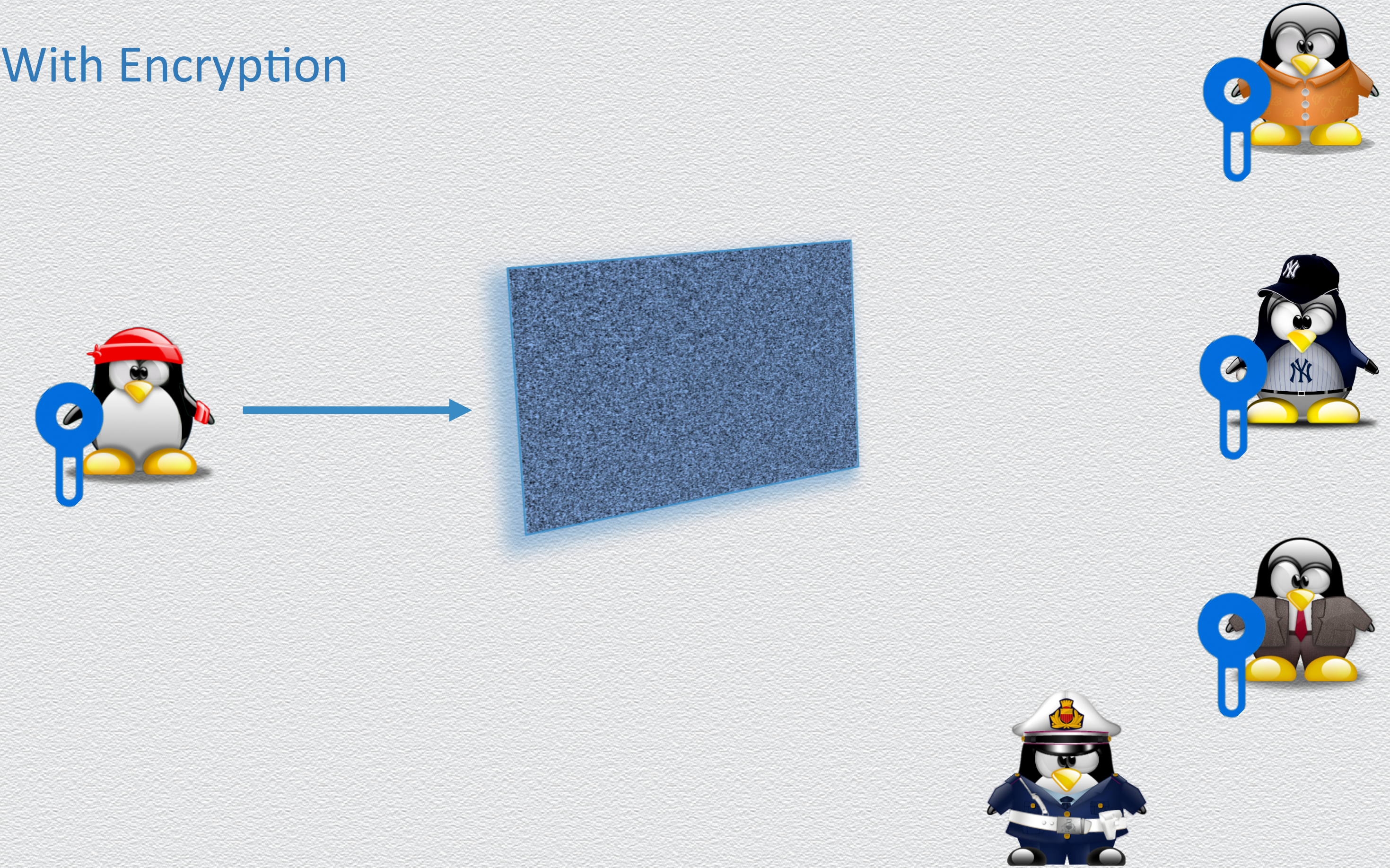
Without Crypto



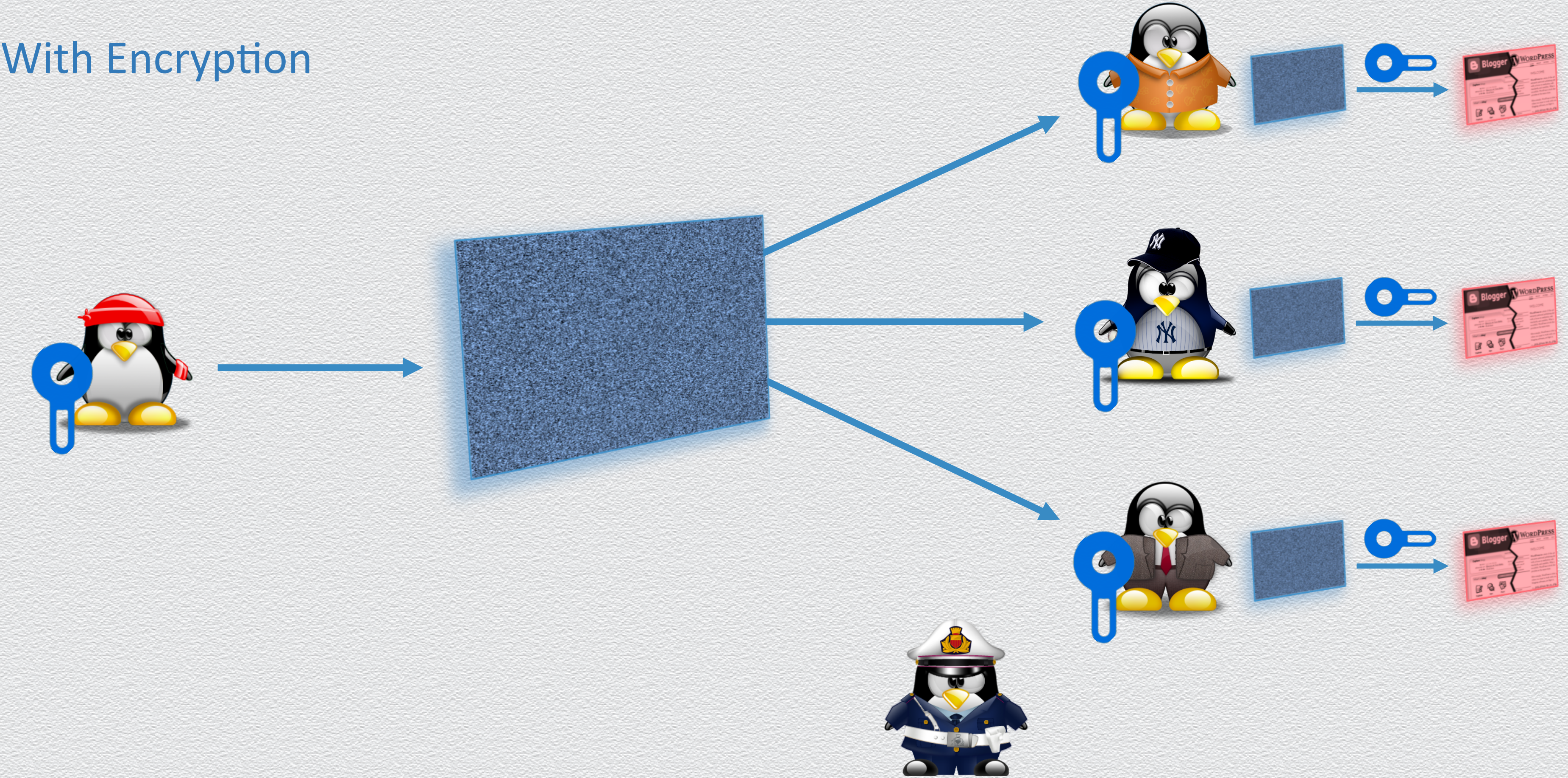
With Encryption



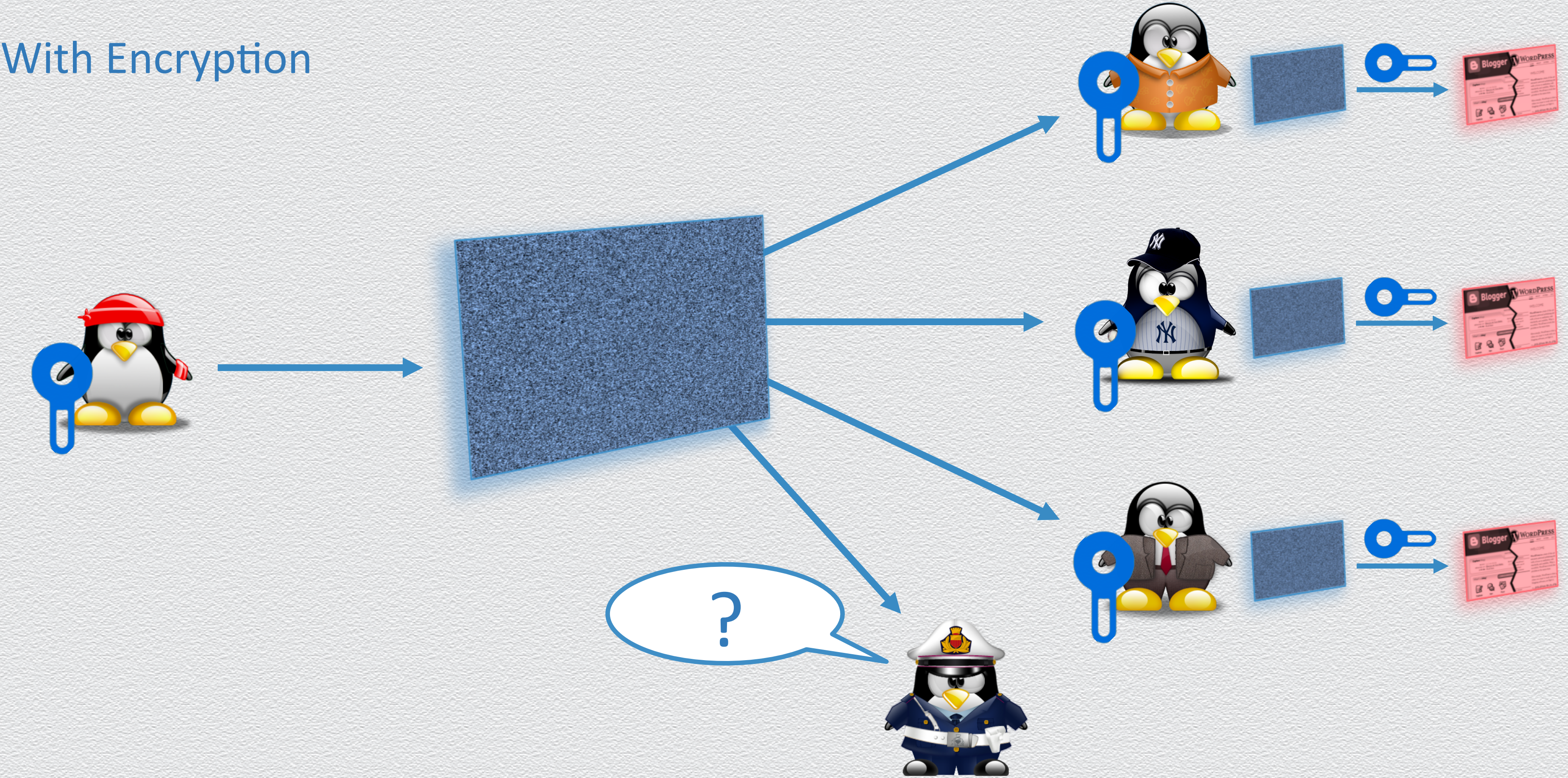
With Encryption



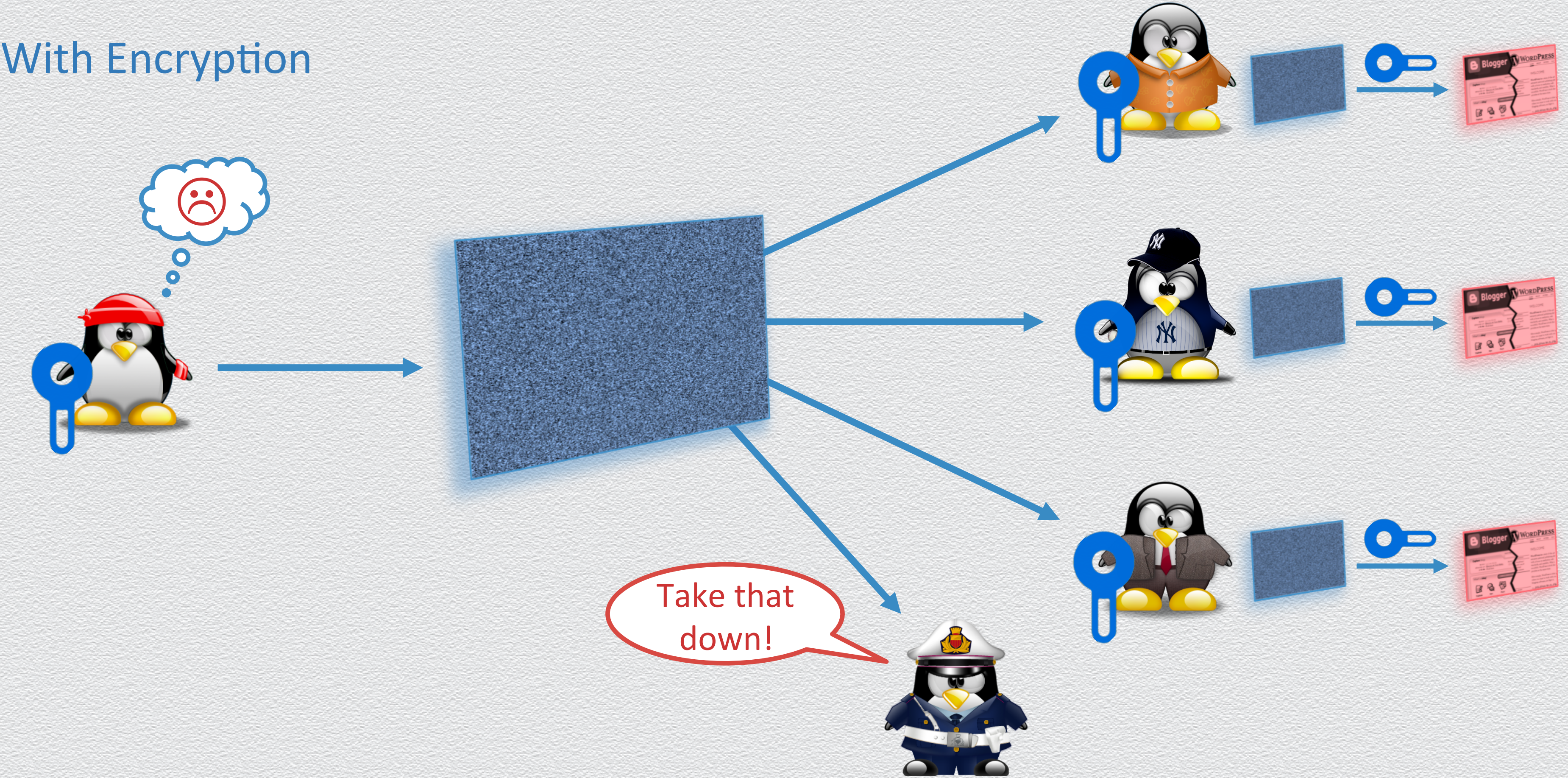
With Encryption



With Encryption

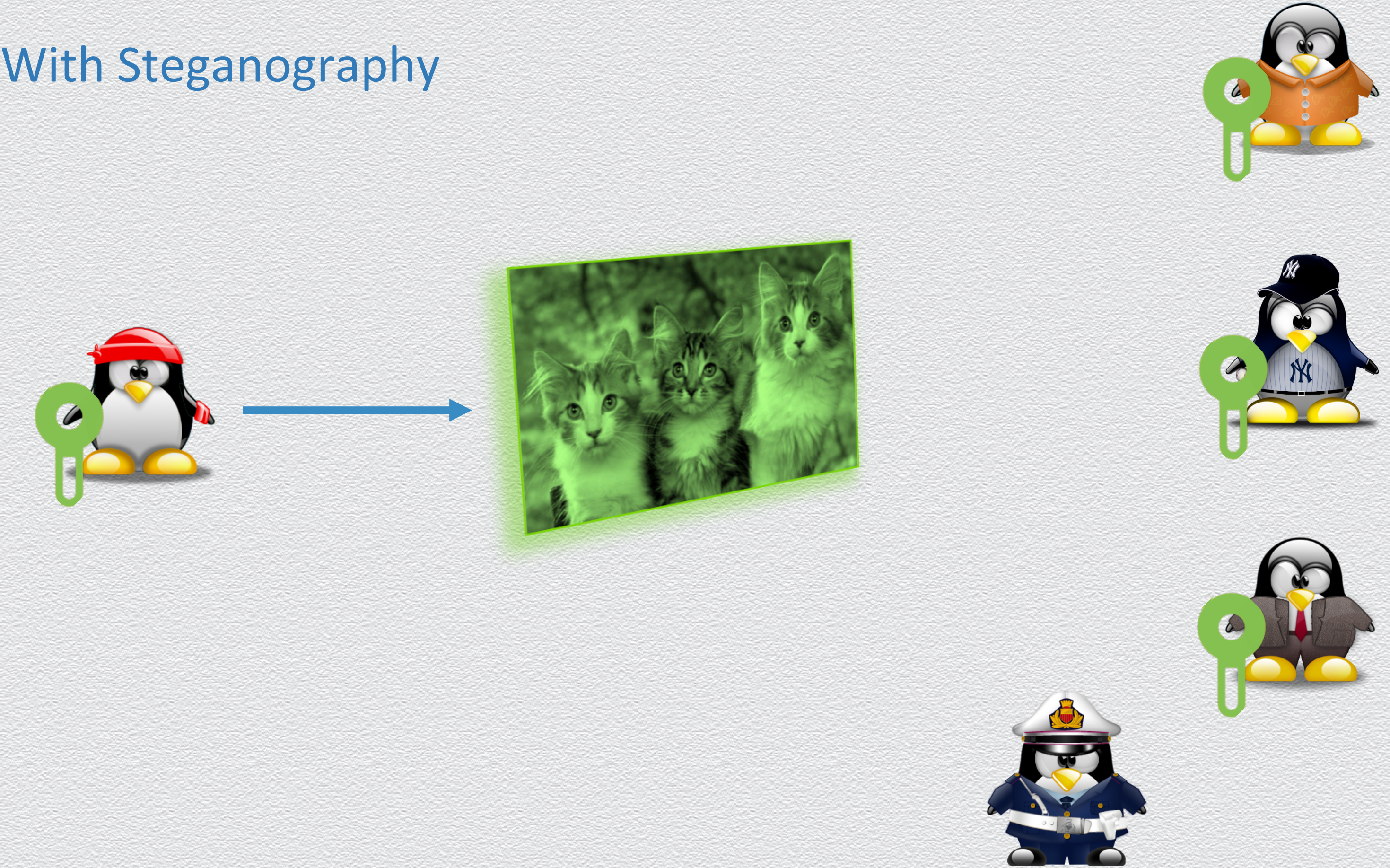


With Encryption

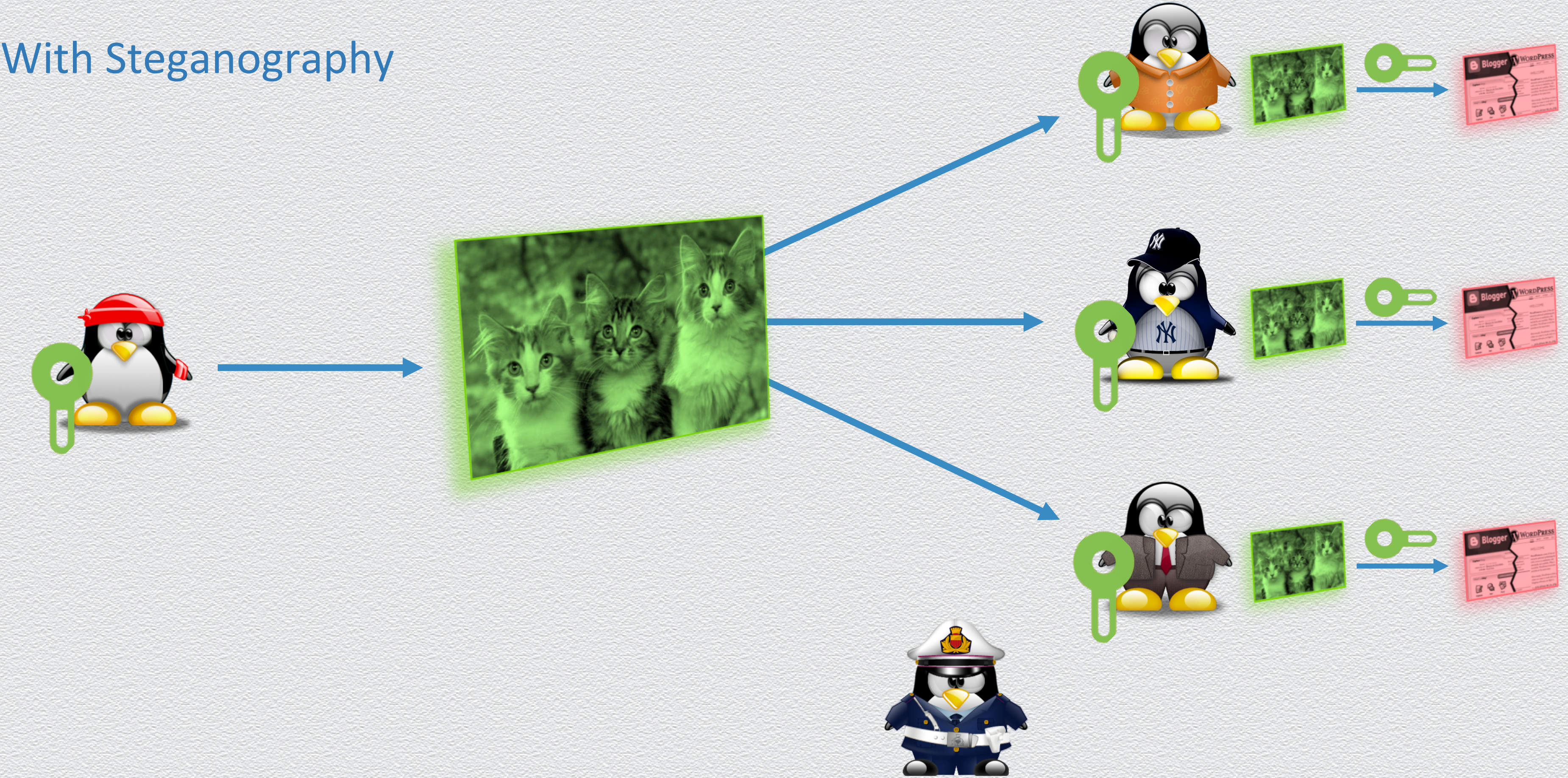


With Steganography

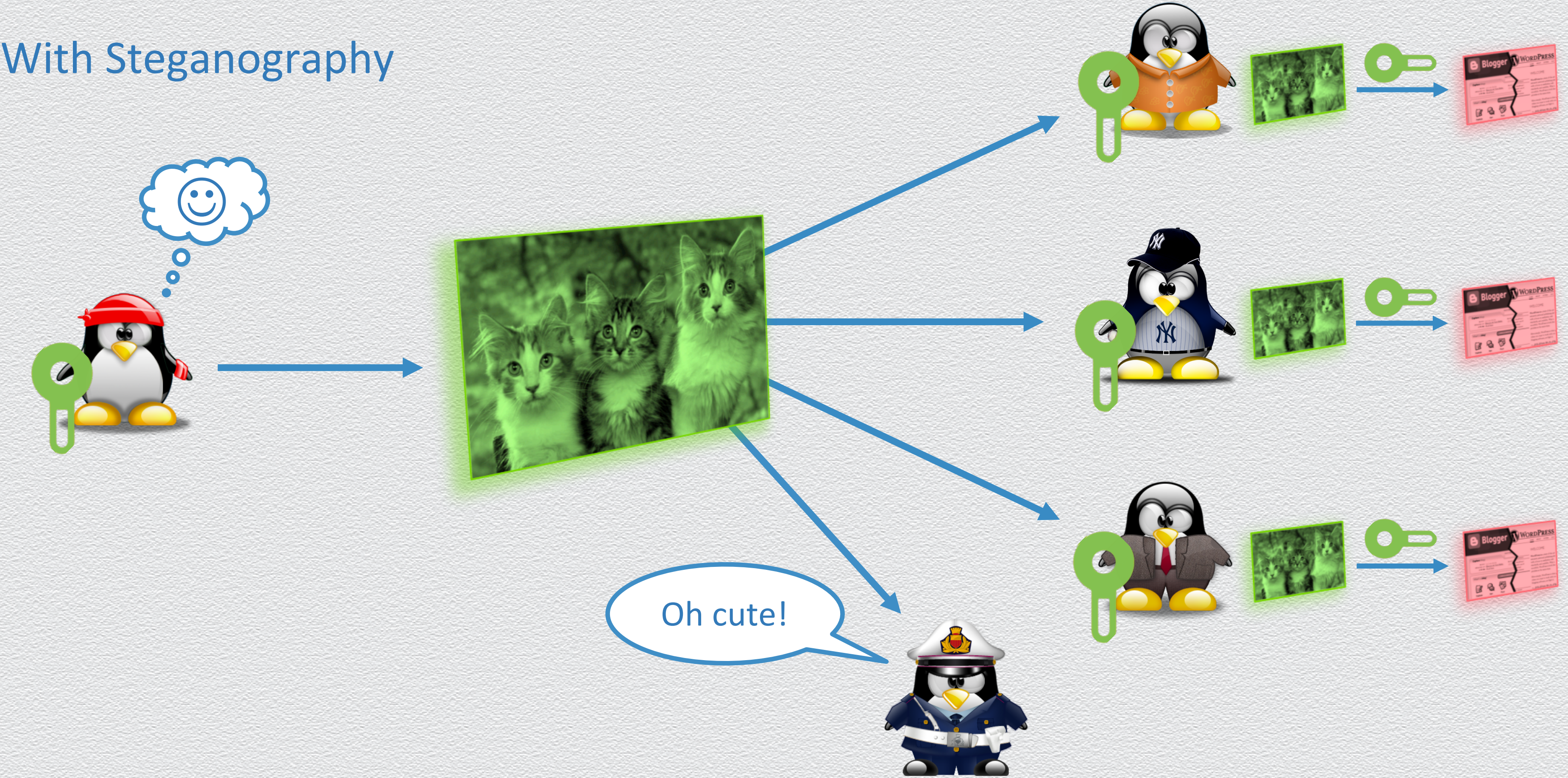
With Steganography



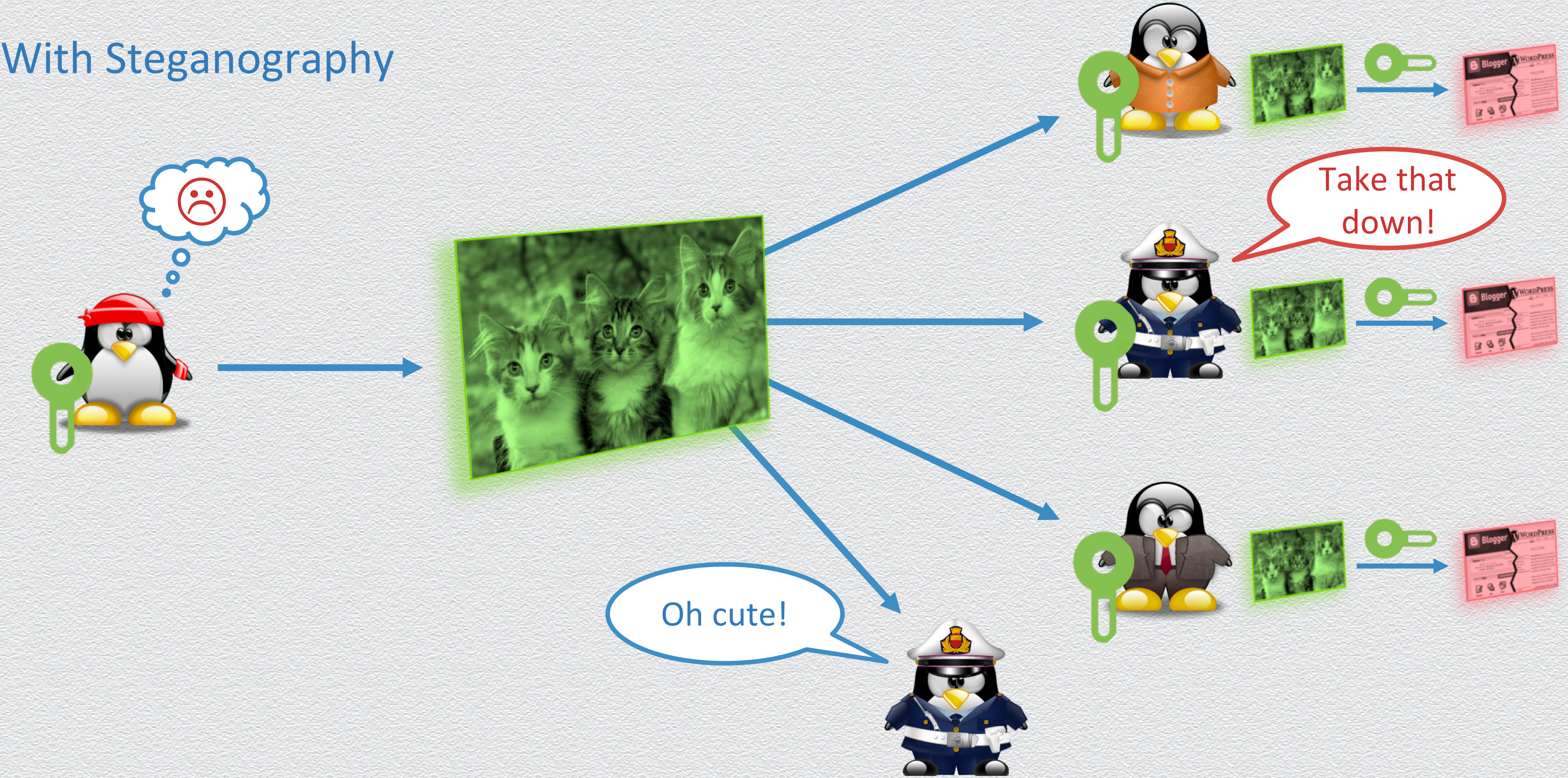
With Steganography



With Steganography

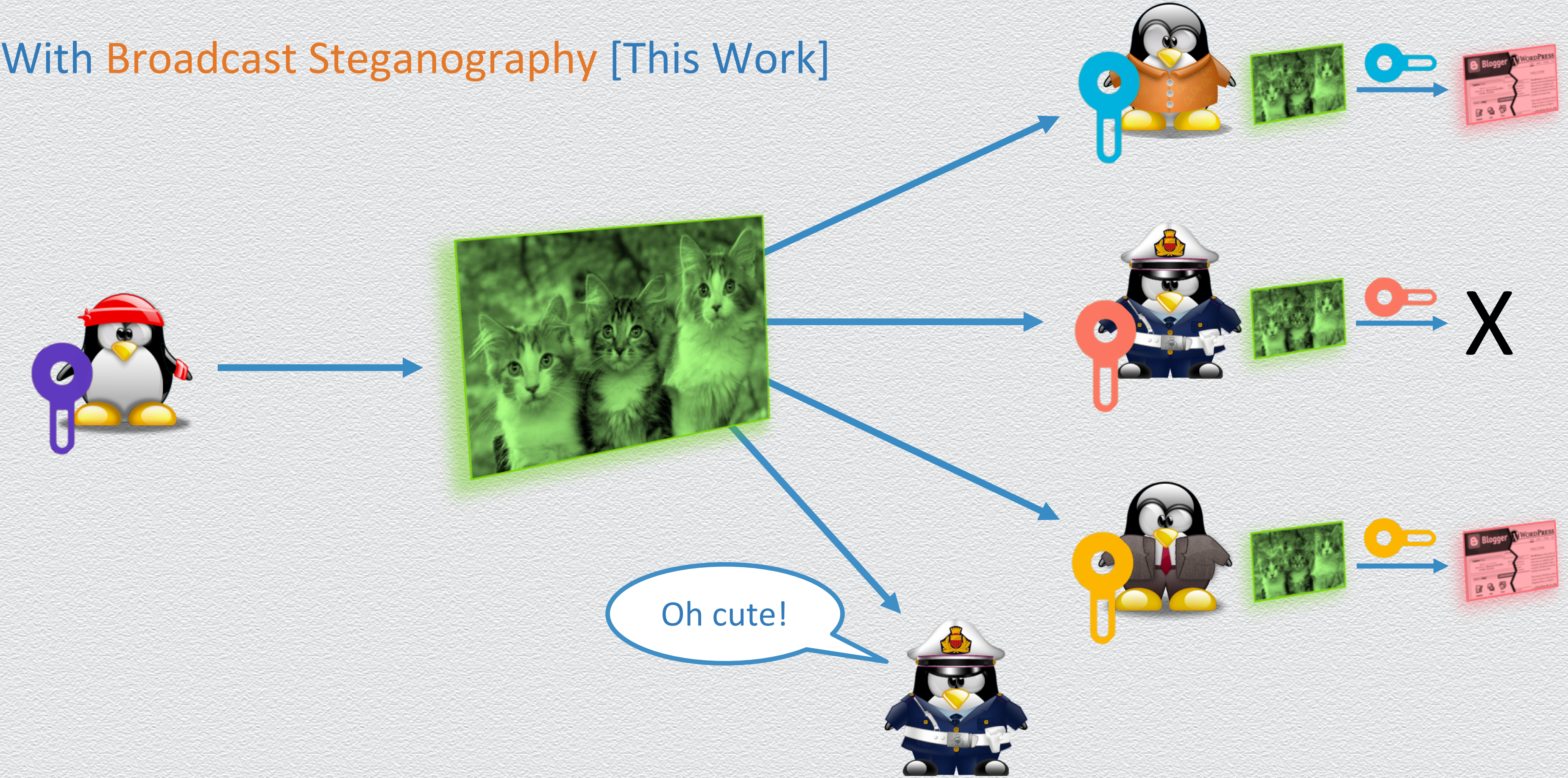


With Steganography

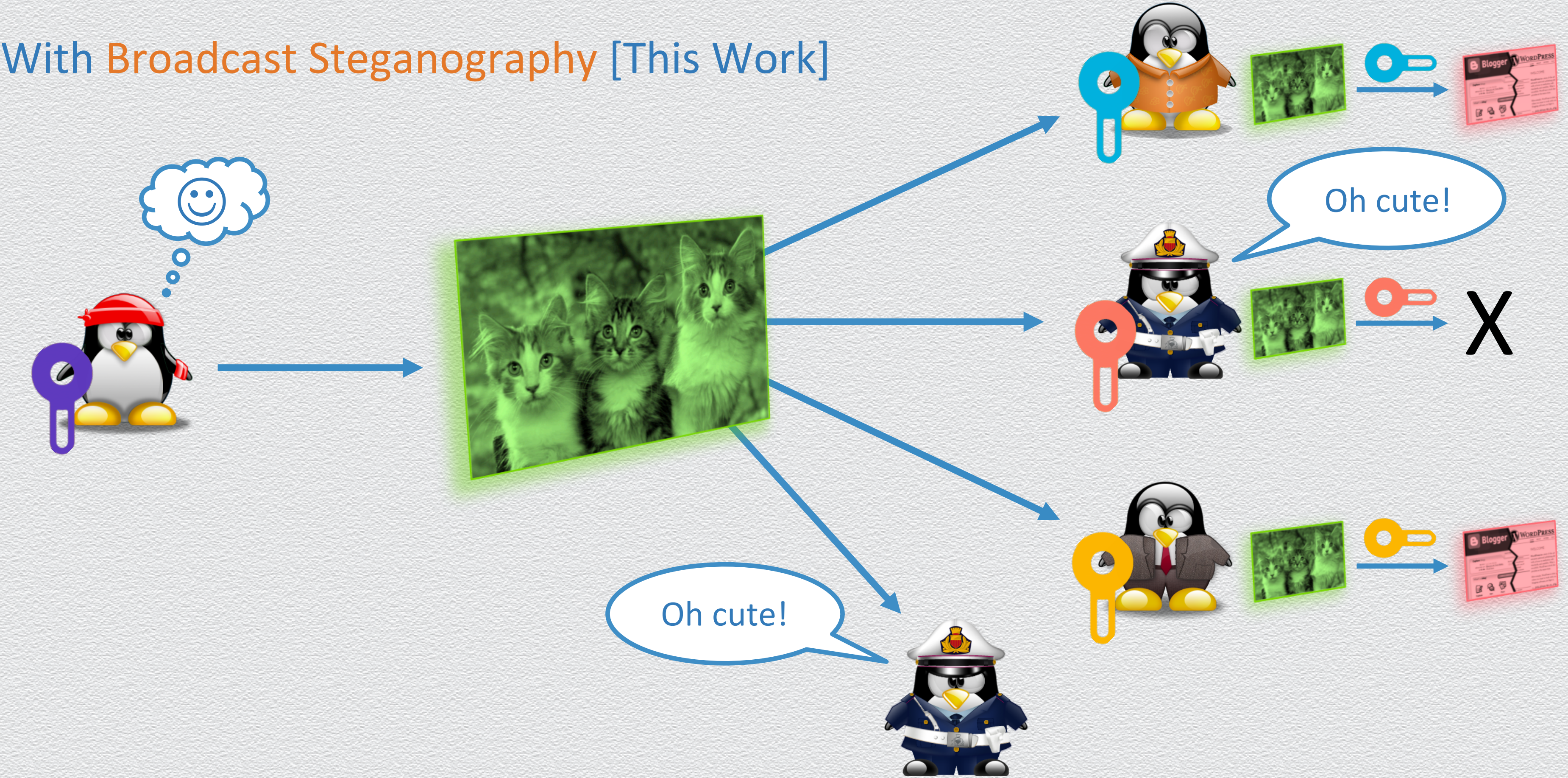


With Broadcast Steganography [This Work]

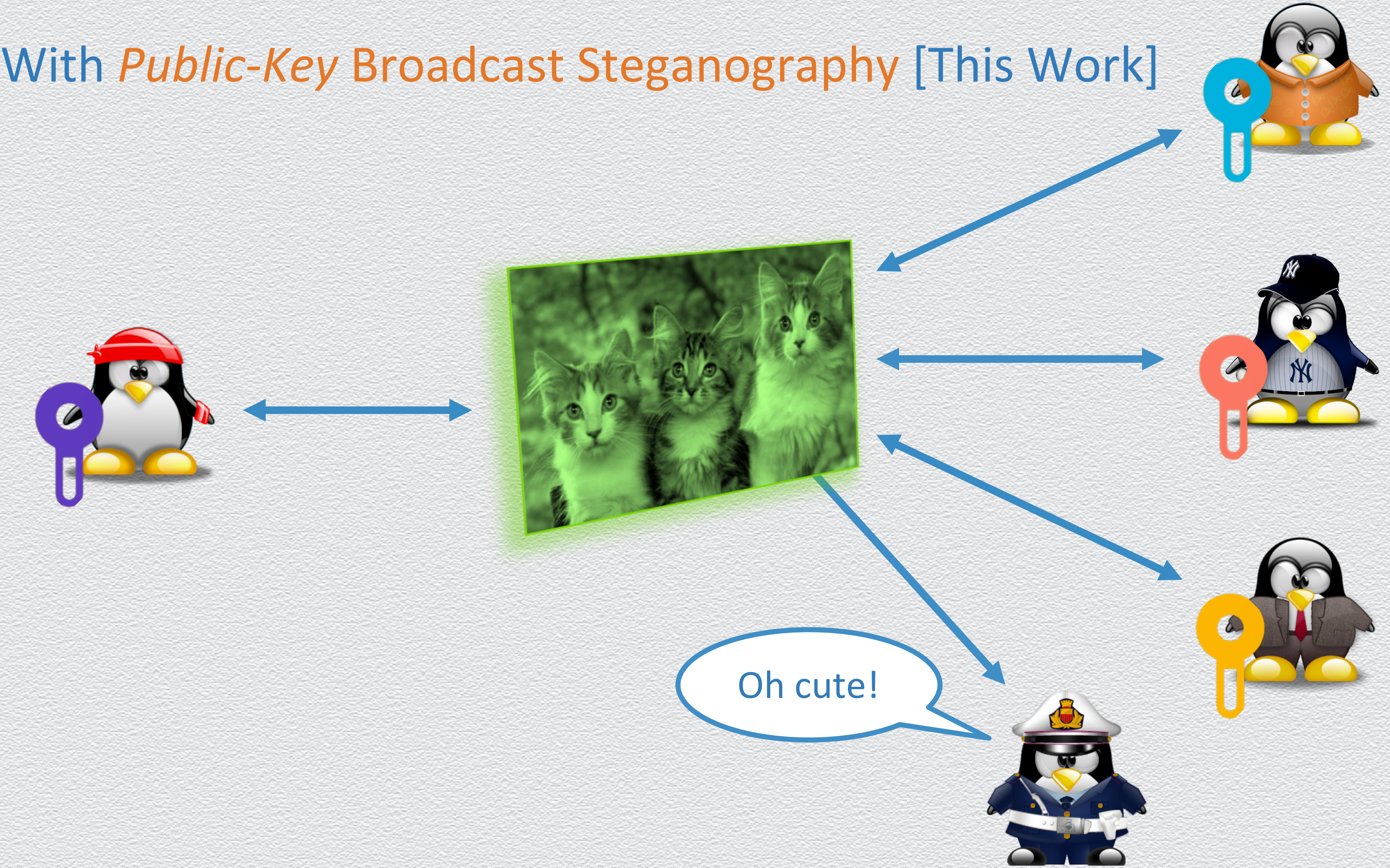
With Broadcast Steganography [This Work]



With Broadcast Steganography [This Work]



With *Public-Key* Broadcast Steganography [This Work]



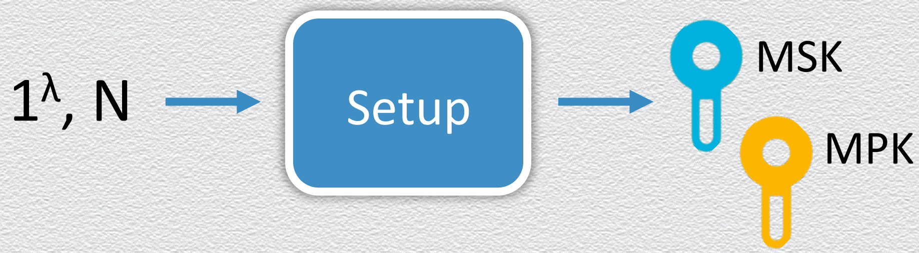
- Broadcast Steganography (BS)
- Constructions
- Summary

- ◉ Broadcast Steganography (BS)
- ○ Constructions
- ○ Summary

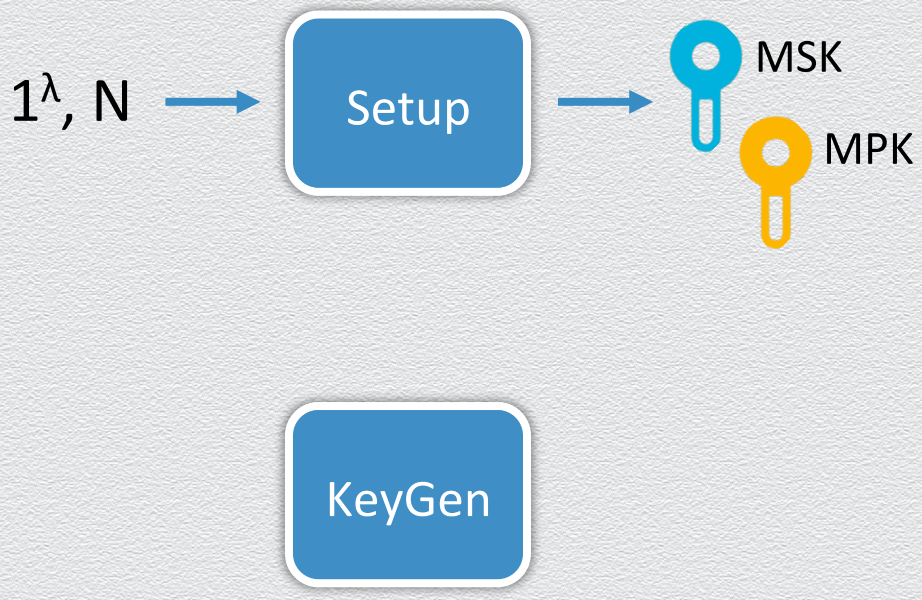
The Setting

Setup

The Setting



The Setting



1^λ, N

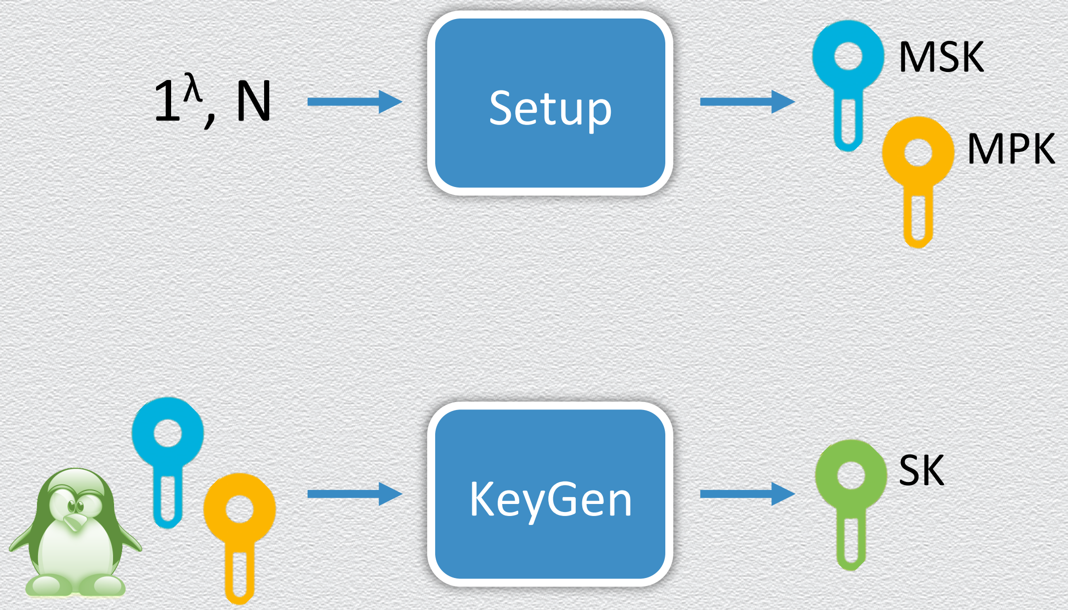
Setup

MSK

MPK

KeyGen

The Setting



1^λ, N

Setup

MSK

MPK

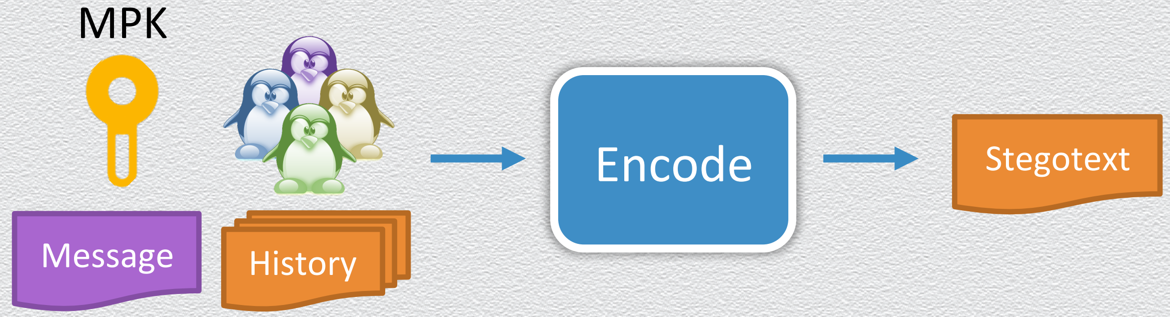
KeyGen

SK

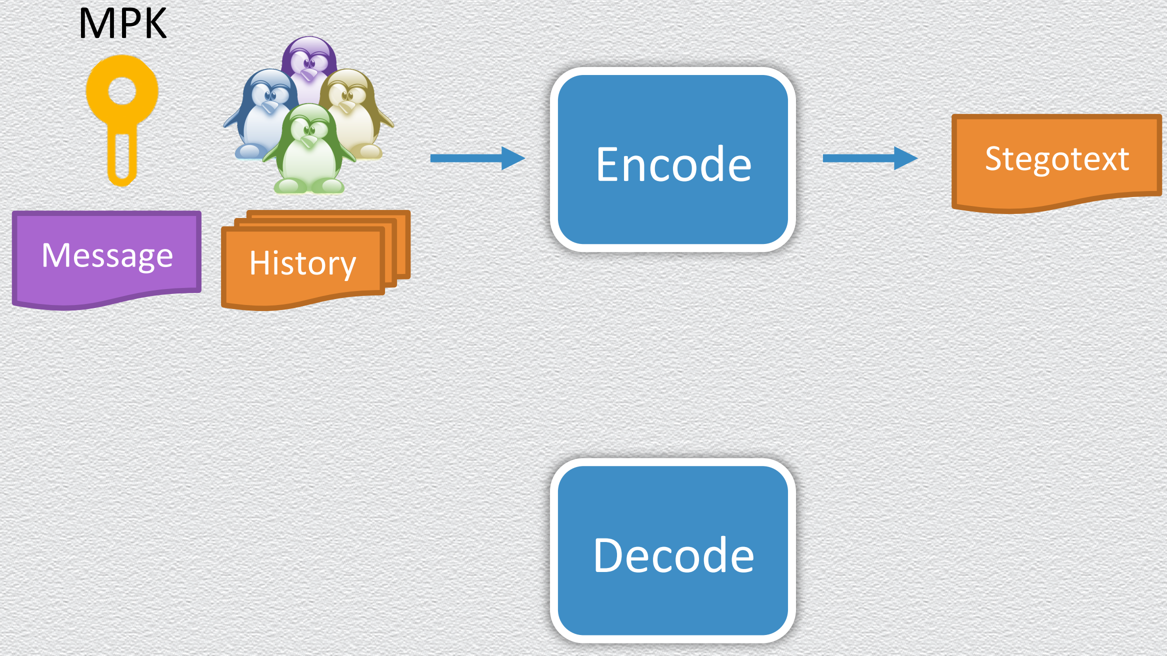
The Setting

Encode

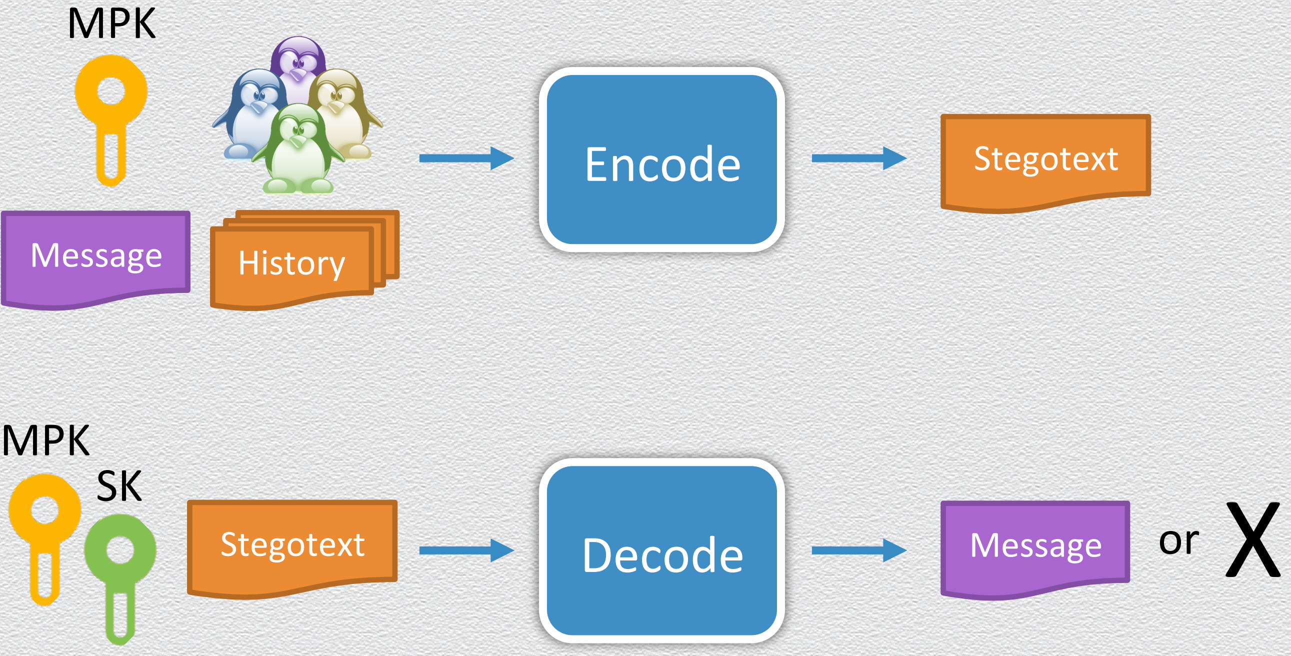
The Setting



The Setting



The Setting


MPK
Message
History
Encode
Stegotext
MPK
SK
Stegotext
Decode
Message
or
X

The Security Model

1. Chosen-Covertext Attack (BS-IND-CCA)
    - Analogous to BE-IND-CCA model
    - Adversary is allowed to corrupt users
    - Adversary is also given access to a decoding oracle
2. Publicly-Detectable Replayable Chosen Covertext Attack (BS-IND-PDR-CCA)
    - Similar to BS-IND-CCA, but with stricter restrictions on allowable decoding queries
3. Chosen-Hiddentext Attack (BS-IND-CHA)
    - Analogous to BE-IND-CPA model
    - Adversary is only allowed to corrupt users
    - No decoding queries

The Security Model

1. Chosen-Covertext Attack (BS-IND-CCA)
   - Analogous to BE-IND-CCA model
   - Adversary is allowed to corrupt users
   - Adversary is also given access to a decoding oracle
2. Publicly-Detectable Replayable Chosen Covertext Attack (BS-IND-PDR-CCA)
   - Similar to BS-IND-CCA, but with stricter restrictions on allowable decoding queries
3. Chosen-Hiddentext Attack (BS-IND-CHA)
   - Analogous to BE-IND-CPA model
   - Adversary is only allowed to corrupt users
   - No decoding queries

The Security Model

BS-IND-CCA Game

① Setup

② Oracle Phase 1

③ Challenge

④ Oracle Phase 2

⑤ Guess

The Security Model

① Setup

The Security Model

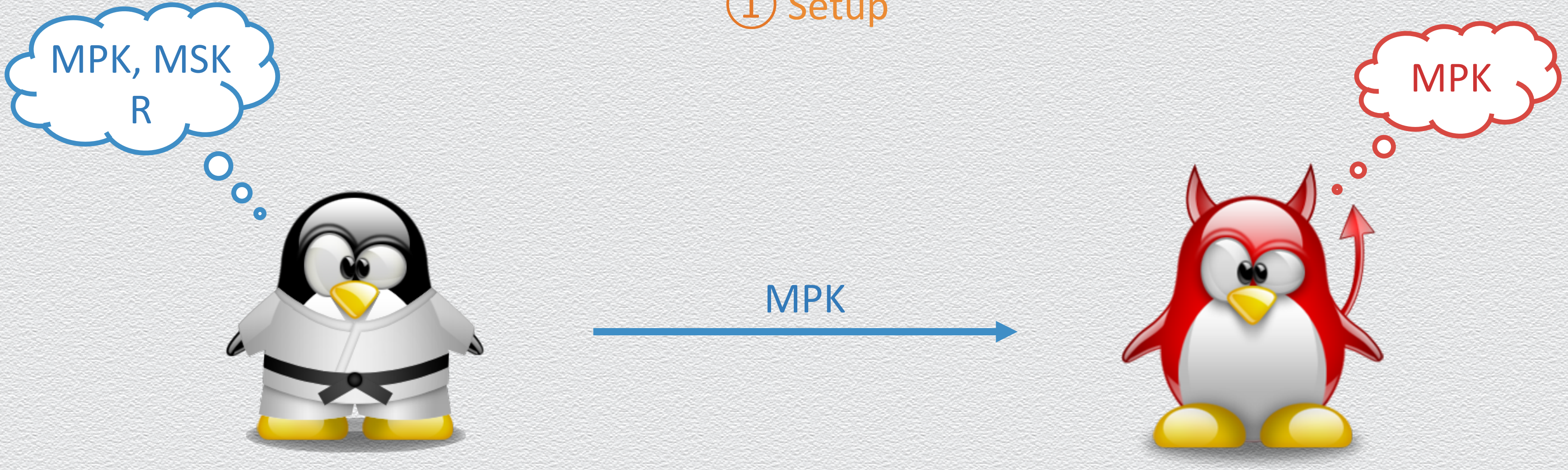
① Setup

$(\mathrm{MPK}, \mathrm{MSK}) \leftarrow \mathrm{Setup}(1^{\lambda}, N)$

$\mathrm{R} := \emptyset$

The Security Model

① Setup



MPK, MSK
R

MPK

MPK

$(\mathrm{MPK}, \mathrm{MSK}) \leftarrow \mathrm{Setup}(1^\lambda, N)$

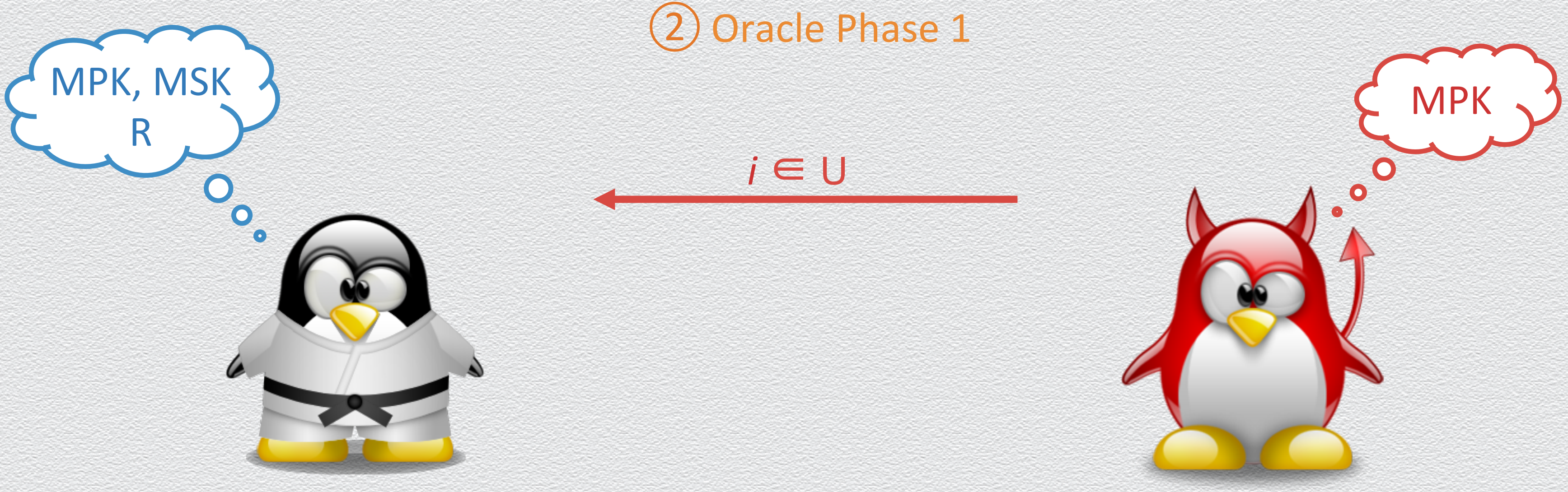
$\mathrm{R} \coloneqq \emptyset$

The Security Model

② Oracle Phase 1

The Security Model

② Oracle Phase 1



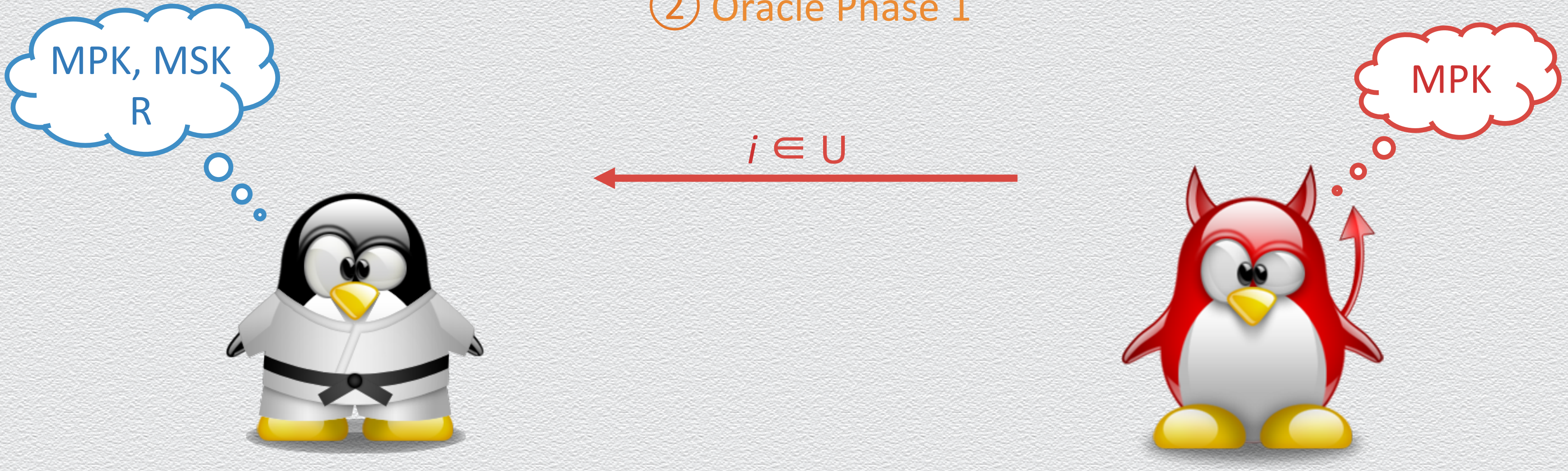
MPK, MSK
R

$i \in U$

MPK

The Security Model

② Oracle Phase 1



MPK, MSK

R

$i \in$ U

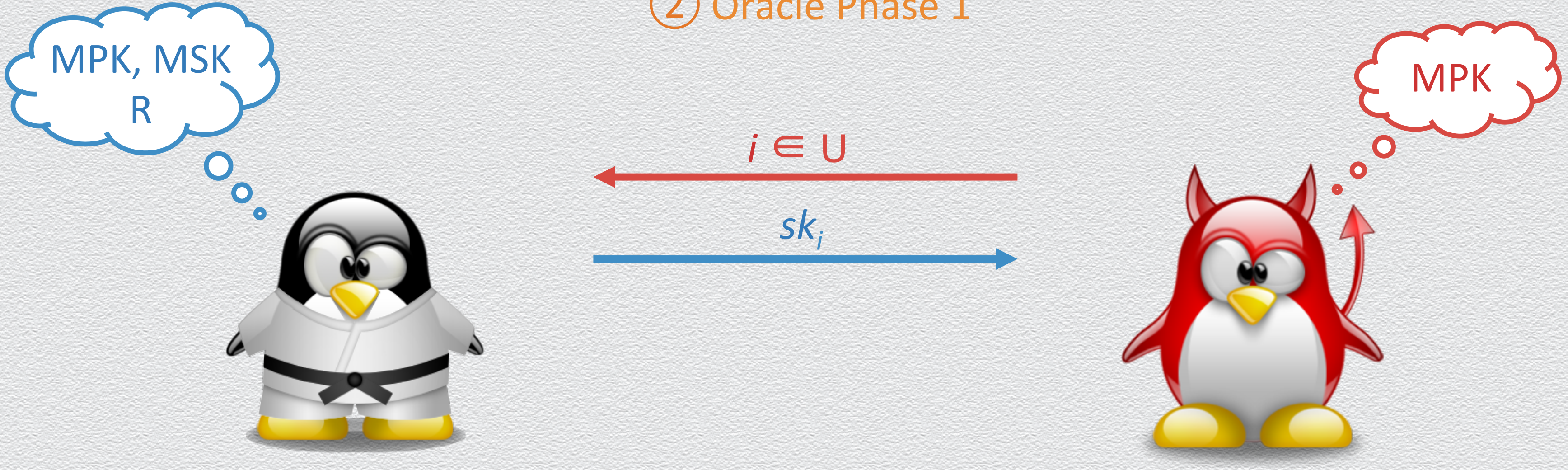
MPK

$\mathrm{R} := \mathrm{R} \cup \{i\}$

$sk_i \leftarrow \mathrm{KeyGen}(\mathrm{MPK}, \mathrm{MSK}, i)$

The Security Model

② Oracle Phase 1



MPK, MSK
R

$i \in$ U

sk_i

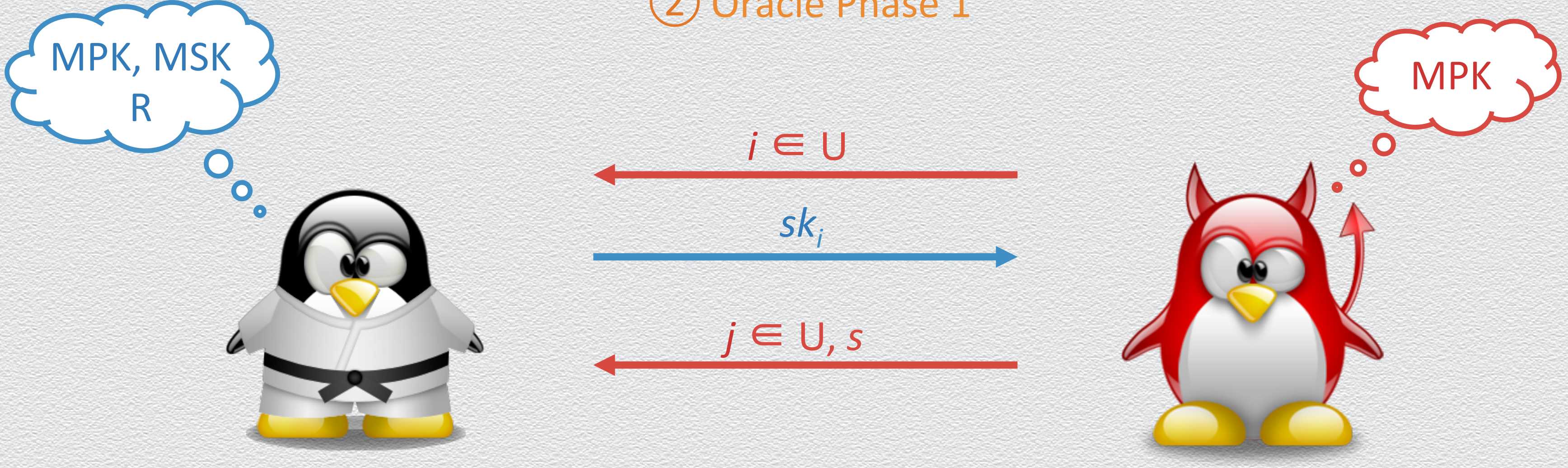
MPK

R := R ∪ { i }

$sk_i \leftarrow$ KeyGen(MPK, MSK, i)

The Security Model

② Oracle Phase 1



MPK, MSK
R

$i \in U$

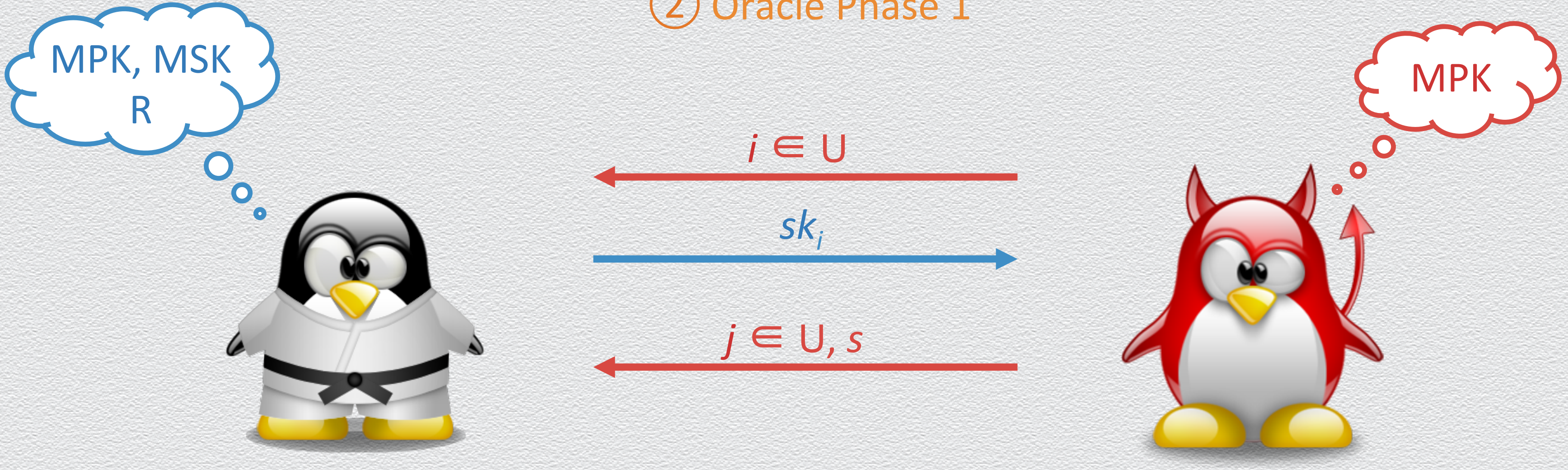
sk_i

$j \in U, s$

MPK

The Security Model

② Oracle Phase 1



MPK, MSK
R

$i \in \mathrm{U}$

sk_i

$j \in \mathrm{U}, s$

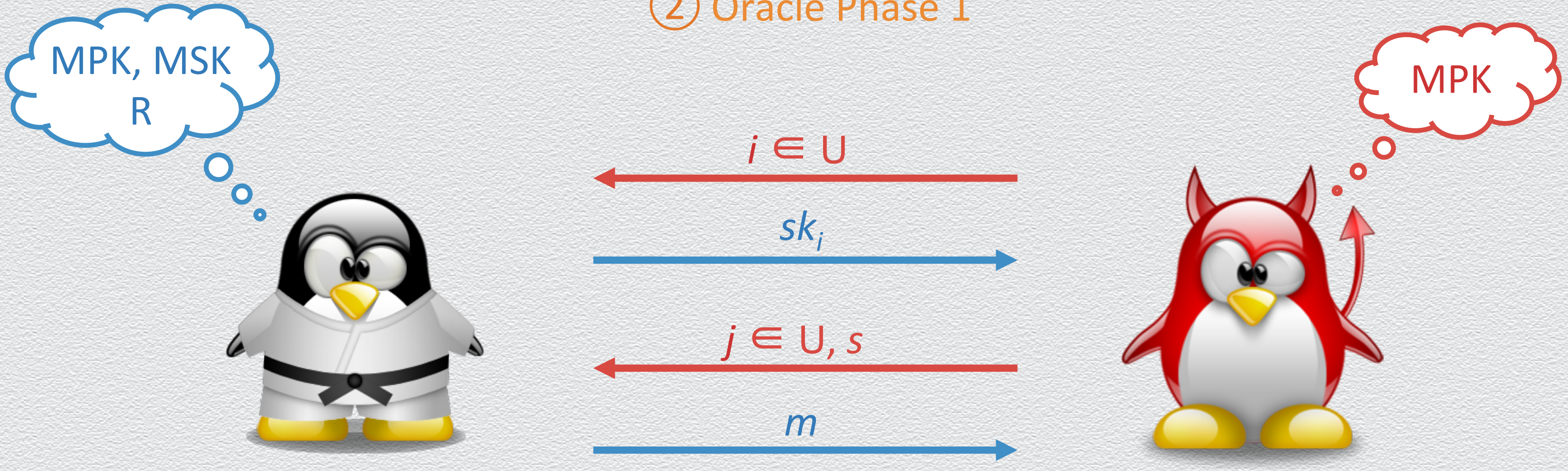
MPK

$sk_j \leftarrow \mathrm{KeyGen}(\mathrm{MPK}, \mathrm{MSK}, j)$

$m := \mathrm{Decode}(\mathrm{MPK}, sk_j, s)$

The Security Model

② Oracle Phase 1



MPK, MSK
R

MPK

$i \in$ U

sk_i

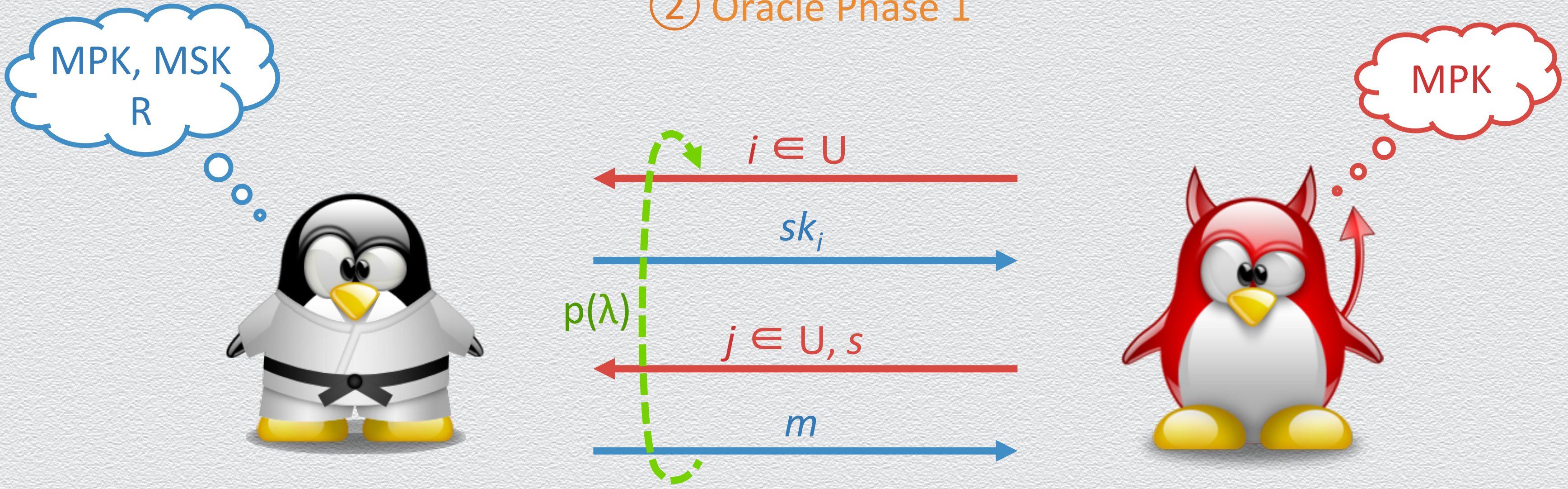
$j \in$ U, s

m

$sk_j \leftarrow$ KeyGen(MPK, MSK, j)
$m :=$ Decode(MPK, sk_j, s)

The Security Model

② Oracle Phase 1

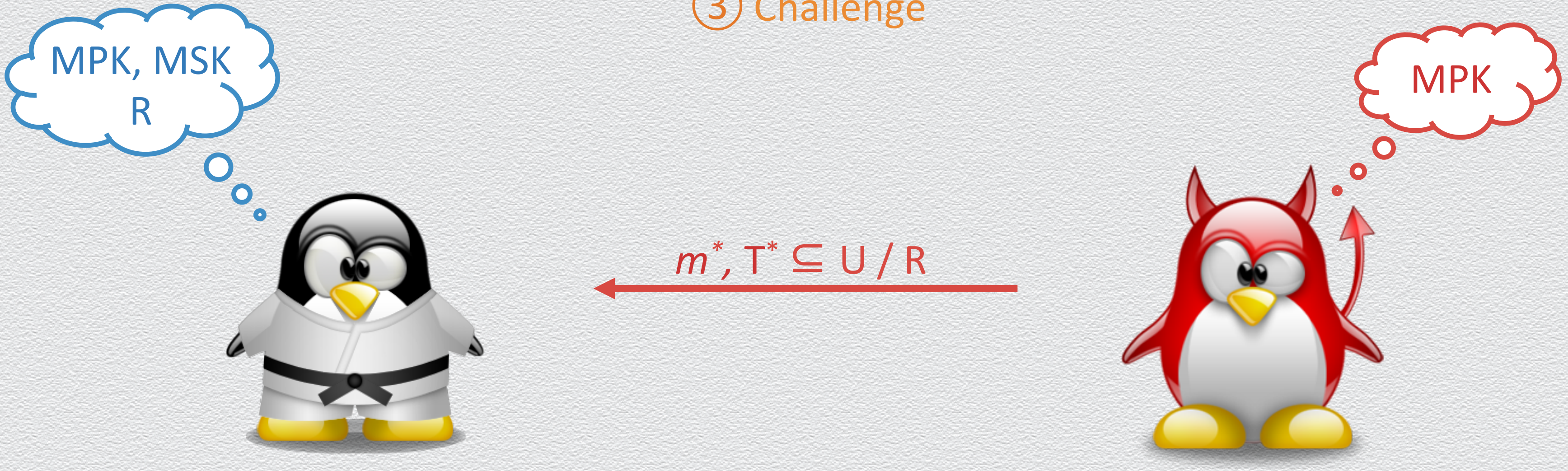


The Security Model

③ Challenge

The Security Model

③ Challenge



MPK, MSK
R

$m^*, T^* \subseteq U / R$

MPK

The Security Model

③ Challenge

MPK, MSK
R

$m^*, \mathrm{T}^* \subseteq \mathrm{U} / \mathrm{R}$ ←

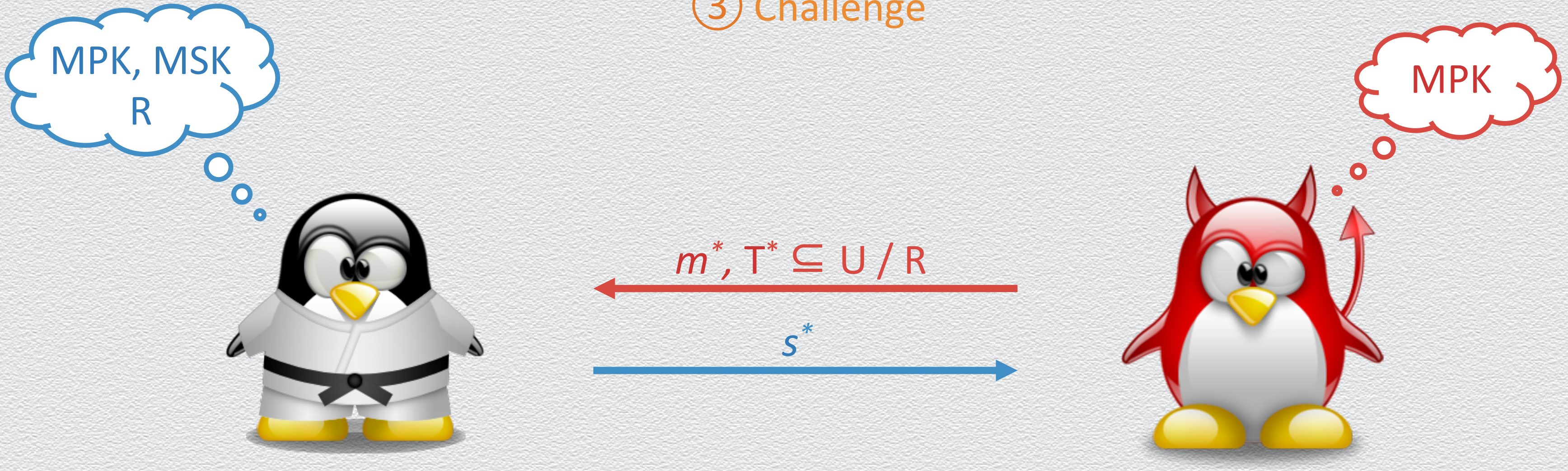
MPK

$b^* \leftarrow \{0,1\}$

$$s^* \leftarrow \begin{cases} \text{Encode}(\text{MPK}, \text{T}^*, m^*) & \text{if } b^* = 0 \\ \text{a random covertext} & \text{otherwise} \end{cases}$$

The Security Model

③ Challenge



MPK, MSK
R

m^*, $\mathrm{T}^* \subseteq \mathrm{U} / \mathrm{R}$

s^*

MPK

$b^* \leftarrow \{0,1\}$

$$s^* \leftarrow \begin{cases} \text{Encode}(\text{MPK}, \text{T}^*, m^*) & \text{if } b^* = 0 \\ \text{a random covertext} & \text{otherwise} \end{cases}$$

The Security Model

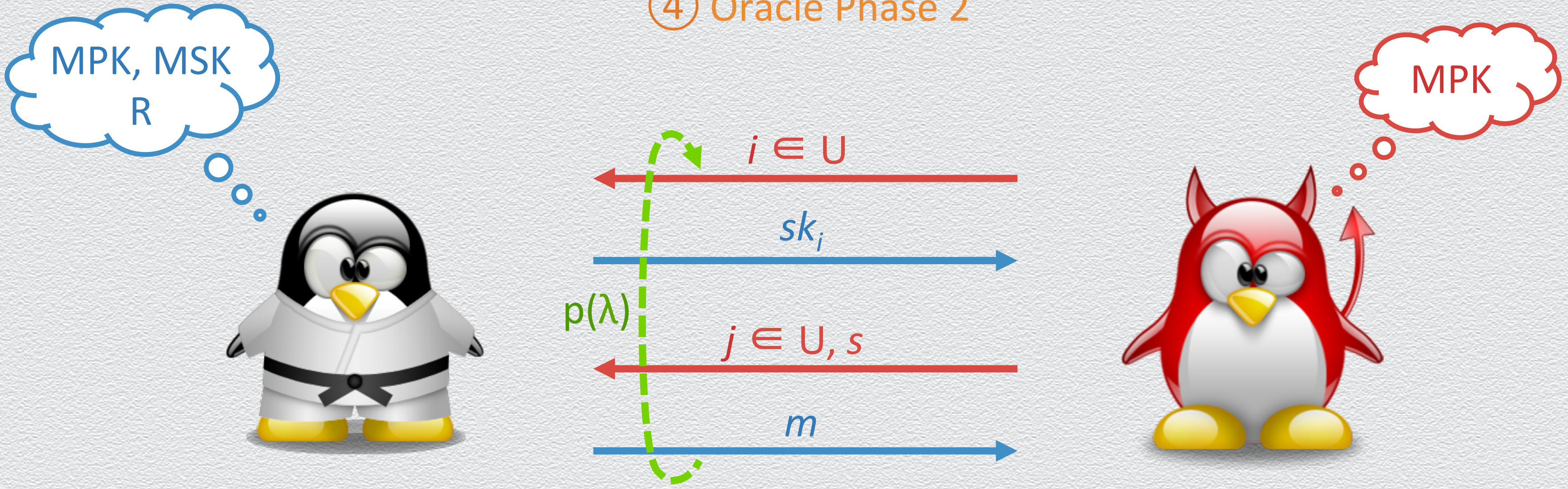
④ Oracle Phase 2

MPK, MSK
R

MPK

The Security Model

④ Oracle Phase 2

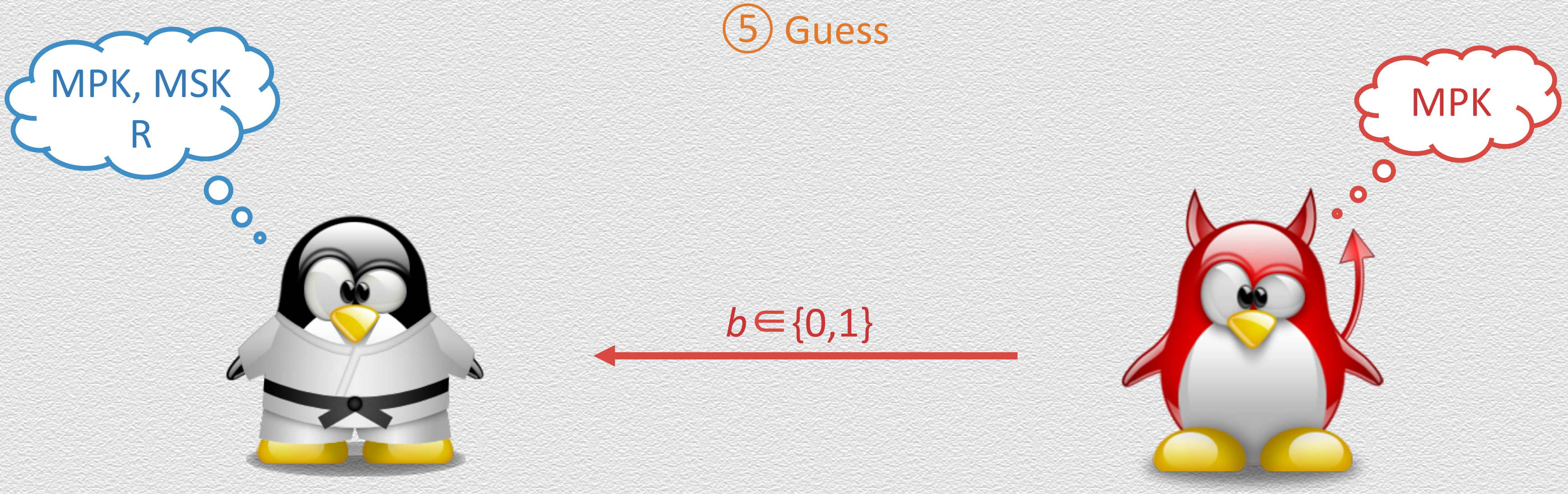


1) $i \notin T^*$
2) if $j \in T^*$, then $s \neq s^*$

The Security Model

⑤ Guess

The Security Model



⑤ Guess

MPK, MSK
R

MPK

$b \in \{0,1\}$

$\mathrm{Adv}_A = |\Pr[b = b^*] - ½|$

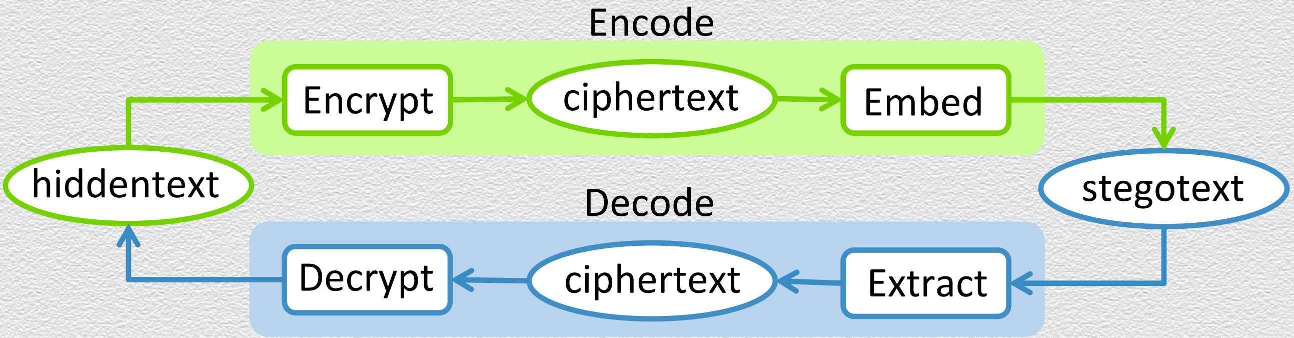
◉ Broadcast Steganography (BS)

◉ Constructions

○ Summary

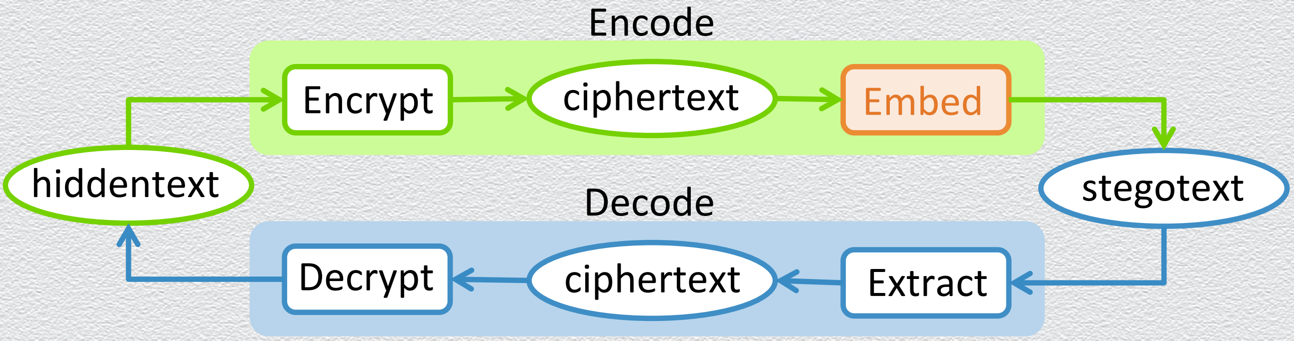
Realizing Broadcast Steganography

- Encrypt-then-Embed Paradigm [HLvA02, BaCa05]



Realizing Broadcast Steganography

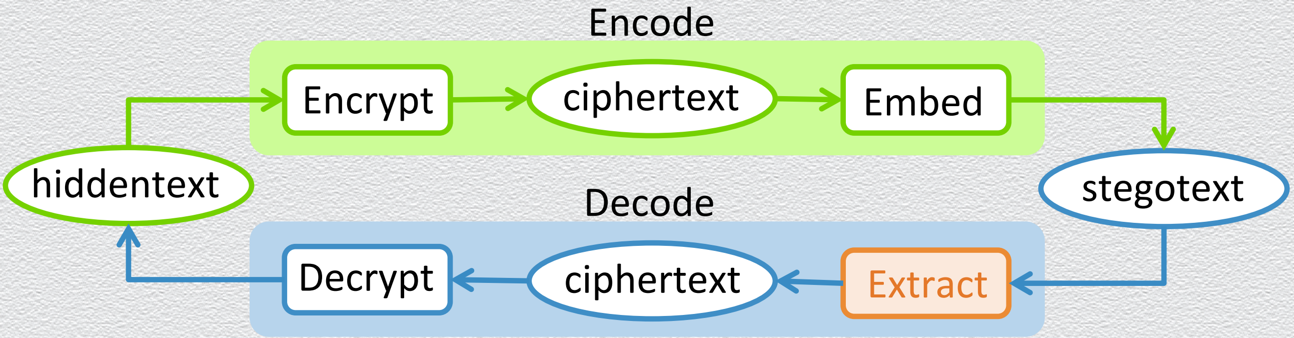
- Encrypt-then-Embed Paradigm [HLvA02, BaCa05]



- Embed (rejection-sampling)
  1. Let H be a strongly universal hash function
  2. Break the ciphertext c into bits $c_1,c_2,\ldots,c_l$
  3. To embed c_i, sample s_i from the channel until $\mathrm{H}(s_i) = c_i$
  4. Output $s = s_1||s_2|| \cdots ||s_l$

Realizing Broadcast Steganography

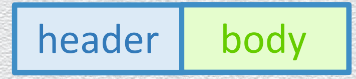
- Encrypt-then-Embed Paradigm [HLvA02, BaCa05]



- Extract
  1. Break the stegotext s into documents $s_1,s_2,\ldots,s_l$
  2. Set $c_i = \mathrm{H}(s_i)$
  3. Output $c = c_1||c_2||\cdots||c_l$

Broadcast Encryption + Encrypt-then-Embed = Broadcast Steganography?

- Encrypt-then-Embed requires pseudorandom ciphertexts …
- … but, Broadcast ciphertexts have structure



broadcast ciphertext format

- Neither header nor body is pseudorandom

Outsider-Anonymous Broadcast Encryption [FaPe12]

- Motivation: Anonymous Broadcast Encryption with short ciphertexts
  - A fully anonymous ciphertext length is subject to a linear lower bound [KiSa12]
  - In some applications, content may give recipient set away
    - ⇒ Suffices to protect anonymity of receivers from outsiders

- Outsider-Anonymity in Broadcast Encryption
  - Trades some degree of anonymity for better efficiency
  - Allows constructions with sub-linear ciphertext length

oABE Encryption in [FaPe12]

- Encrypt(S, m)
    1. Group users in S into S’, a set of disjoint subsets
        - |S’| is sub-linear in |S|
    2. Generate a ciphertext c_i for each s_i in S’ (using anonymous IBE)
    3. Attach a tag t_i to each c_i (for efficient decryption at the receivers)
    4. Bundle all (t_i, c_i) components using one-time signature

oABE Encryption in [FaPe12]

- Encrypt(S, m)
  1. Group users in S into S’, a set of disjoint subsets
     - |S’| is sub-linear in |S|
  2. Generate a ciphertext c_i for each s_i in S’ (using anonymous IBE)
  3. Attach a tag t_i to each c_i (for efficient decryption at the receivers)
  4. Bundle all (t_i, c_i) components using one-time signature

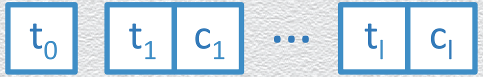
oABE Encryption in [FaPe12]

- Encrypt(S, m)
  1. Group users in S into S’, a set of disjoint subsets
     - |S’| is sub-linear in |S|
  2. Generate a ciphertext c_i for each s_i in S’ (using anonymous IBE)
  3. Attach a tag t_i to each c_i (for efficient decryption at the receivers)
  4. Bundle all (t_i, c_i) components using one-time signature

c_1 … c_l

oABE Encryption in [FaPe12]

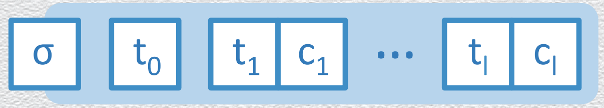
- Encrypt(S, m)
  1. Group users in S into S’, a set of disjoint subsets
     - |S’| is sub-linear in |S|
  2. Generate a ciphertext c_i for each s_i in S’ (using anonymous IBE)
  3. Attach a tag t_i to each c_i (for efficient decryption at the receivers)
  4. Bundle all (t_i, c_i) components using one-time signature



t_0 t_1 c_1 … t_l c_l

oABE Encryption in [FaPe12]

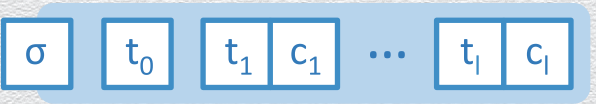
- Encrypt(S, m)
  1. Group users in S into S’, a set of disjoint subsets
     - |S’| is sub-linear in |S|
  2. Generate a ciphertext c_i for each s_i in S’ (using anonymous IBE)
  3. Attach a tag t_i to each c_i (for efficient decryption at the receivers)
  4. Bundle all (t_i, c_i) components using one-time signature



σ | t_0 | t_1 c_1 | … | t_l c_l

oABE Encryption in [FaPe12]

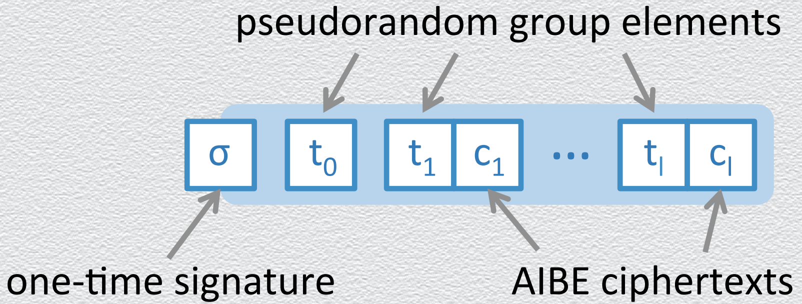
- Encrypt(S, m)
  1. Group users in S into S’, a set of disjoint subsets
     - |S’| is sub-linear in |S|
  2. Generate a ciphertext c_i for each s_i in S’ (using anonymous IBE)
  3. Attach a tag t_i to each c_i (for efficient decryption at the receivers)
  4. Bundle all (t_i, c_i) components using one-time signature



σ t_0 t_1 c_1 … t_l c_l

- Notice that ciphertexts have no header …
- … but still exhibit structure due to tags and signature
- Idea: Toward a BS construction, make these components pseudorandom

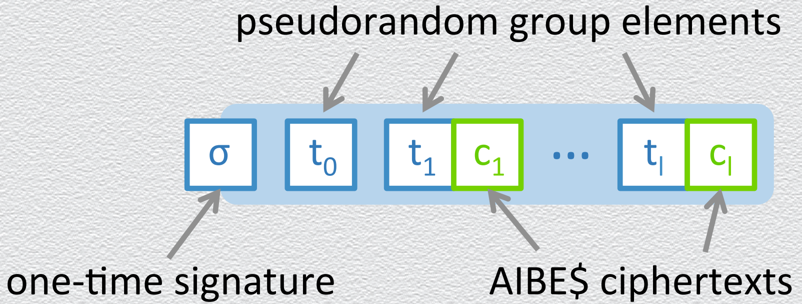
oABE with Pseudorandom Ciphertexts (oABE$) [This Work]



- How to make oABE ciphertexts pseudorandom?

1. Replace the underlying AIBE with AIBE$ [AgBo09]
2. Apply an entropy smoothing hash to group elements
3. Replace one-time signature with a MAC (implemented via PRF)

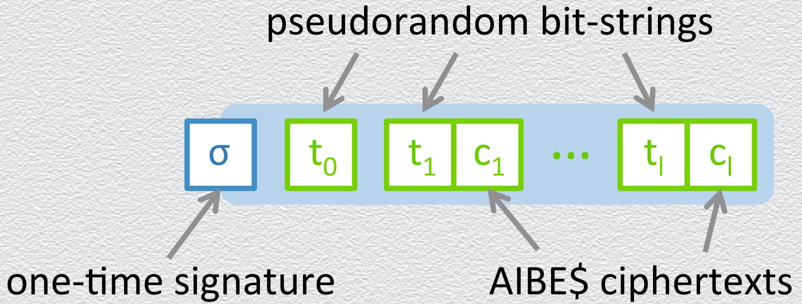
oABE with Pseudorandom Ciphertexts (oABE$) [This Work]



- How to make oABE ciphertexts pseudorandom?

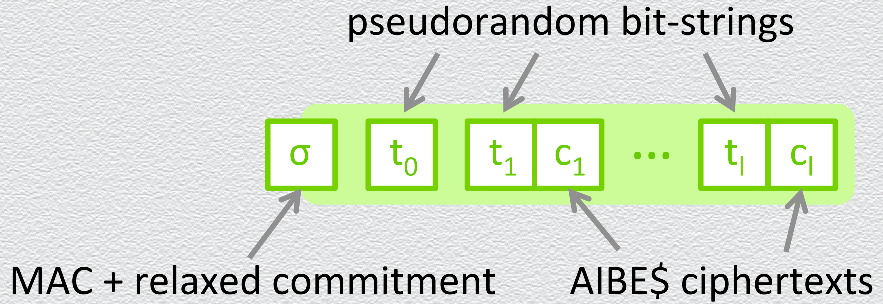
1. Replace the underlying AIBE with AIBE$ [AgBo09]
2. Apply an entropy smoothing hash to group elements
3. Replace one-time signature with a MAC (implemented via PRF)

oABE with Pseudorandom Ciphertexts (oABE$) [This Work]



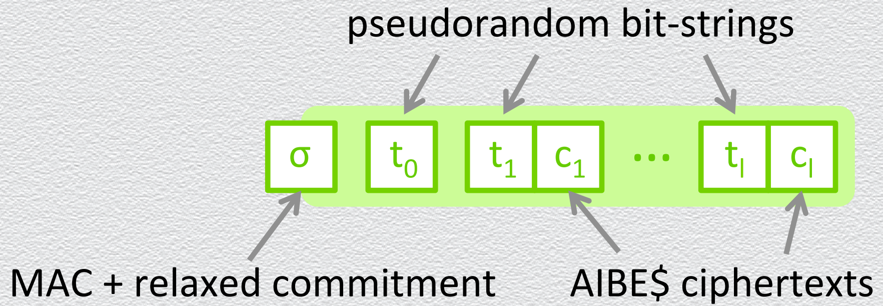
- How to make oABE ciphertexts pseudorandom?
  1. Replace the underlying AIBE with AIBE$ [AgBo09]
  2. Apply an entropy smoothing hash to group elements
  3. Replace one-time signature with a MAC (implemented via PRF)

oABE with Pseudorandom Ciphertexts (oABE$) [This Work]



- How to make oABE ciphertexts pseudorandom?
  1. Replace the underlying AIBE with AIBE$ [AgBo09]
  2. Apply an entropy smoothing hash to group elements
  3. Replace one-time signature with a MAC (implemented via PRF)

oABE with Pseudorandom Ciphertexts (oABE$) [This Work]



- How to make oABE ciphertexts pseudorandom?
  1. Replace the underlying AIBE with AIBE$ [AgBo09]
  2. Apply an entropy smoothing hash to group elements
  3. Replace one-time signature with a MAC (implemented via PRF)

Question: How to embed the MAC key in c_i's and still obtain CCA security?

Solution: Construct an encapsulation mechanism [DoKa05, BoKa05] with pseudorandom commitments

Comparison of BE Schemes with Anonymity Properties

Scheme	|PK|	|sk|	|c|	Security Model	Anonymity
BBW06	O(N)	O(1)	O(N-r)	Static, RO	Full
LPQ12	O(N)	O(1)	O(N-r)	Adaptive, Standard	Full
FaPe12a	O(N)	O(log N)	O(r log (N/r))	Adaptive, Standard	Outsider
FaPe12b	O(N log N)	O(N)	O(r)	Adaptive, Standard	Outsider
This Work	O(N)	O(log N)	O(r log (N/r))	Adaptive, Standard	Outsider

N: total number of users, r: number of revoked users

- Only oABE$ provides pseudorandom ciphertexts

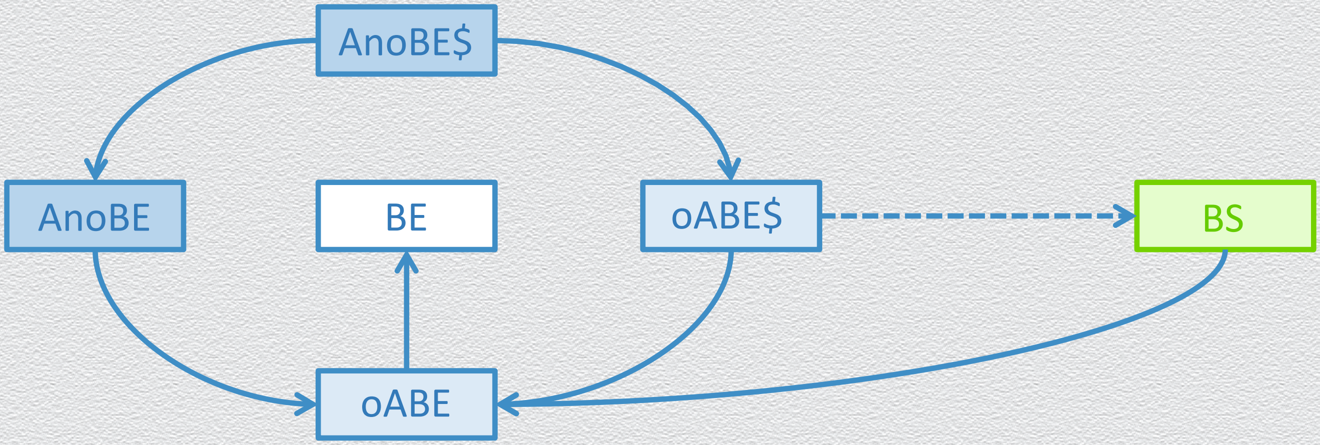
Our Construction of Broadcast Steganography

- Highlights
  - oABE$ + Encrypt-then-Embed = Broadcast Steganography
  - Our constructions have sub-linear stegotext length
  - For CCA security, requires stateless channel

- Constructions:
  1. BS-CHA
  2. BS-PDR-CCA
  3. BS-CCA

- ⦿ Broadcast Steganography (BS)
- ⦿ Constructions
- ⦿ Summary

BE and Friends



Summary

- Initiated the study of Broadcast Steganography
  - A multi-recipient communication tool to plant undetectable messages in innocent-looking conversations
- Put forth sublinear constructions of broadcast steganography under a range of security notions
- In the process, devised efficient broadcast encryption schemes with pseudorandom ciphertexts and anonymity properties
  - Implementing CCA checks without imposing structure on broadcast ciphertexts required overcoming multiple technical hurdles