# Decryption Despite Errors

Combining FEC and encryption to achieve security against fuzzers and improve bandwidth efficiency.

Karolin Varner ~ karo@cupdev.net

https://github.com/koraa/decryption-despite-errors

# Use Case

e.g. media streaming, radio transmission

- Data is transmitted via **noisy channels**
- We reduce noise in cipher text using **FEC**
- Reject messages containing errors using **MAC**

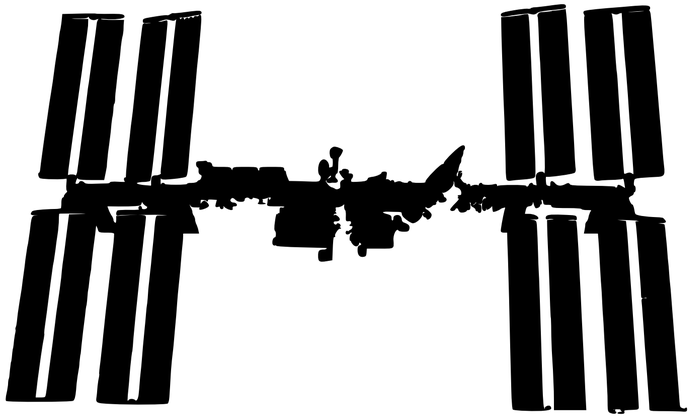
- Goal: Security against fuzzers!

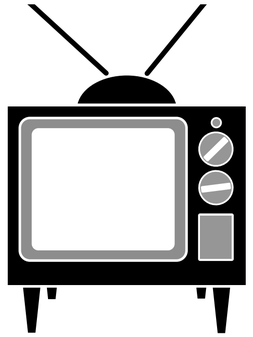
$\Longrightarrow$ Better reliability in shared media transmissions

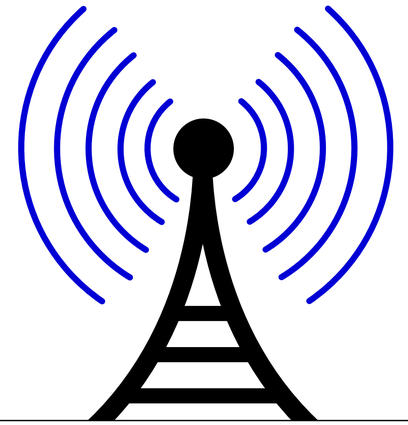
- Goal: Provable security under partial message recovery

$\Longrightarrow$ Improves transmission efficiency for media streaming & applications with sophisticated error correction

- Good work on AMAC[1] returning hard decision
- Limited work on encryption under noise[2]








[1] Tonien, D., Safavi-Naini, R., Nickolas, P., & Desmedt, Y. (2009, June). Unconditionally secure approximate message authentication.

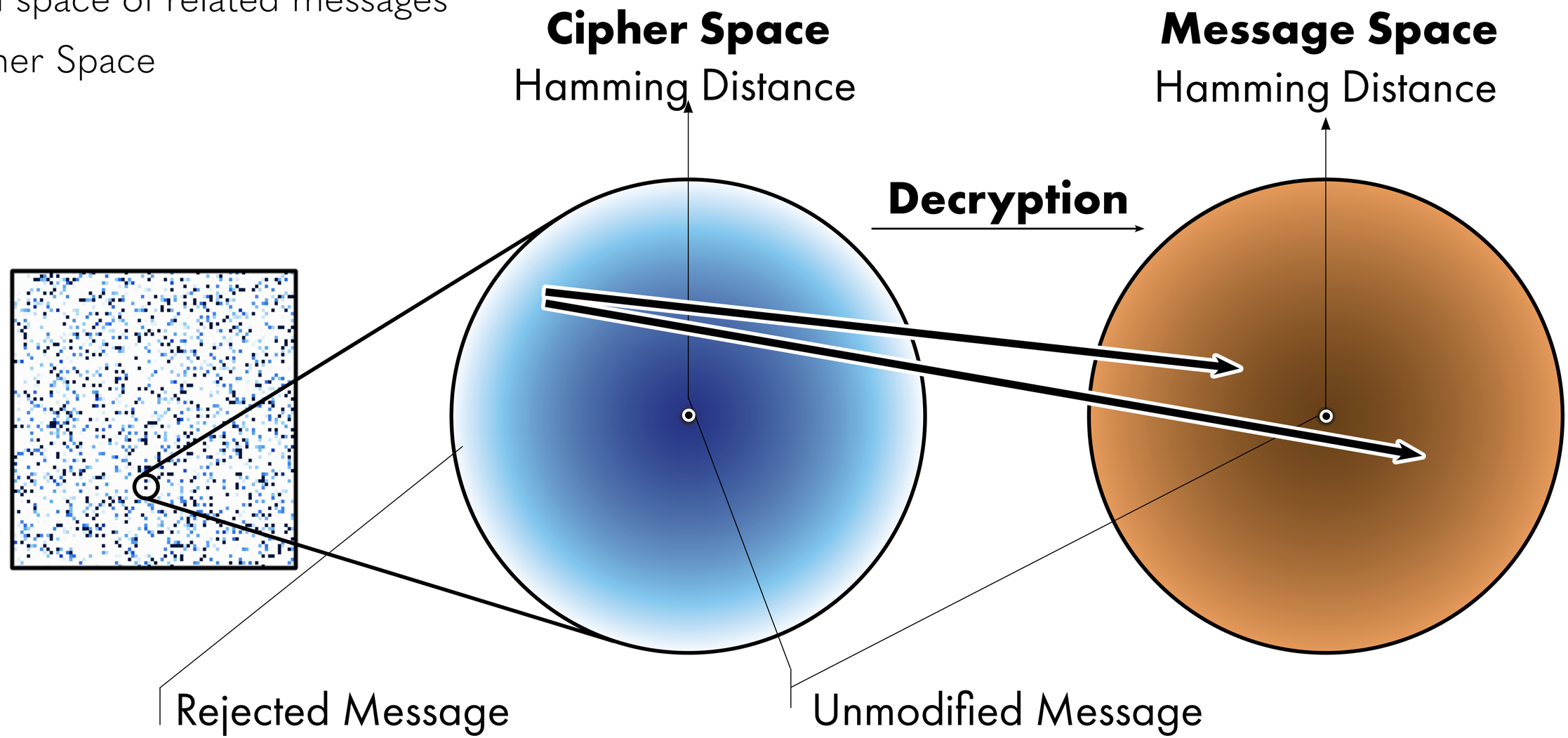
[2] Mathur, C. N., Narayan, K., & Subbalakshmi, K. P. (2006, June). High diffusion cipher: Encryption and error correction in a single cryptographic primitive.

Graphics: i2clipart.com /clipart-simple-television-0287, /clipart-radio-wireless-tower-cor-7b33, /clipart-international-space-station-cb77, /clipart-loudspeaker-b776 Open Clip Art Libary project

# Decryption Despite Errors

Embed space of related messages

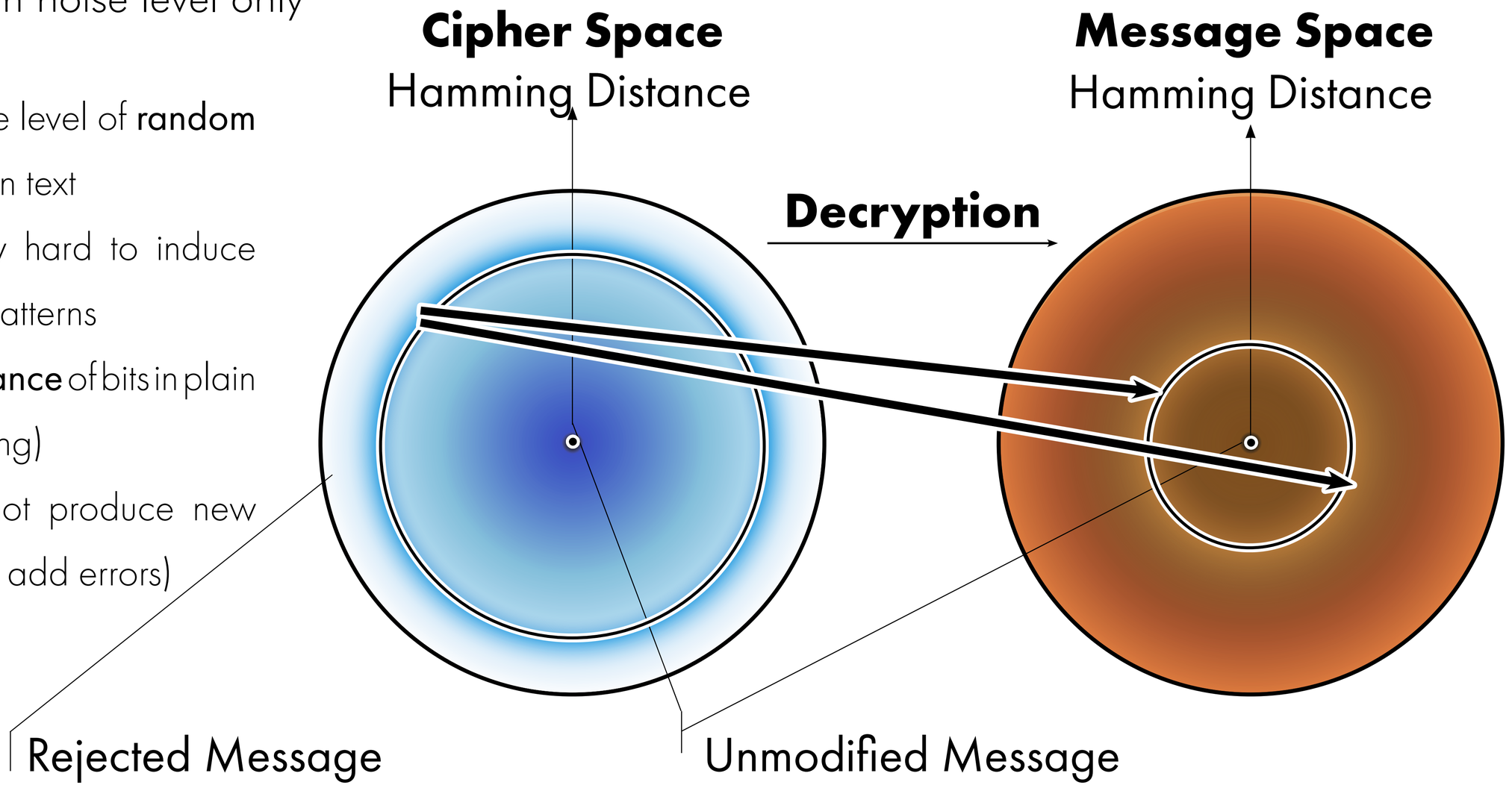
in Cipher Space



# Min. adversary capabilty

Controls random noise level only

- Let attacker raise level of **random** noise in the plain text
- Computationally hard to induce specific noise patterns
- **Uniform importance** of bits in plain text (for decoding)
- Adversary cannot produce new messages (only add errors)



# Standard Notions

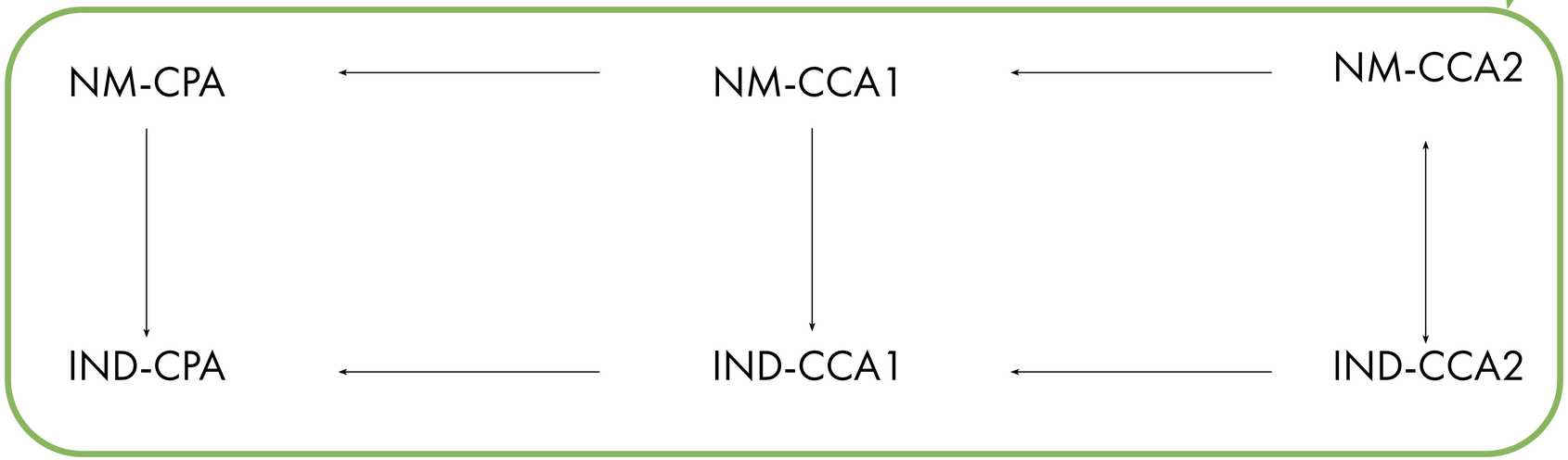
Security Against Fuzzing extends standard notions.

| | |
|---|---|
| $W(\cdot)$ | hamming weight |
| $Enc_{K,R}(k,n,x) = y$ | encryption |
| $Dec_{K,R}(k,n,y') = (x',w)$ | decryption |
| $K$ | security parameter |
| $k,n$ | key, nonce |
| $x,y$ | plain text, cipher text |
| $R$ | redundancy parameter |
| $x',y'$ | plain text, cipher text with errors |
| $w \approx W(x \oplus x')$ | error estimate |

**Security Against Fuzzing:**

Extends CCA2 security

(just set $w_{max}$=0)




Bellare, M., Desai, A., Pointcheval, D., & Rogaway, P. (1998, August). Relations among notions of security for public-key encryption schemes.

# Security Against Fuzzing

Attack model and security definition.

| | |
|---|---|
| $W(\cdot)$ | hamming weight |
| $Enc_{K,R}(k,n,x) = y$ | encryption |
| $Dec_{K,R}(k,n,y') = (x',w)$ | decryption |
| $K$ | security parameter |
| $k,n$ | key, nonce |
| $x,y$ | plain text, cipher text |
| $R$ | redundancy parameter |
| $x',y'$ | plain text, cipher text with errors |
| $w \approx W(x \oplus x')$ | error estimate |

## Attack Model

- Goal of a fuzzer is denial of service
- Defense is impossible if attacker can delete entire message; we allot the attacker to flip a limited number of bits.
- This models attacks in shared mediums, like radio transmissions.

## FEC-CCA1/2

"Larger redundancy $\Longrightarrow$ more errors corrected"

1. Adv. chooses messages $x_0$, $x_1$ & redundancy parameters $R_0$, $R_1$

3. A. chooses syndromes $s_0$, $s_1$, $W(s_1) = W(s_2)$

5. Adv. wins if msg with higher redundancy yields decryption with more errors

$\Longrightarrow$ For every error rate there exists some R that will recover the full message.

⚠ Definitions are very subtle because of soft decision, so no "guaranteed safe" parameters.

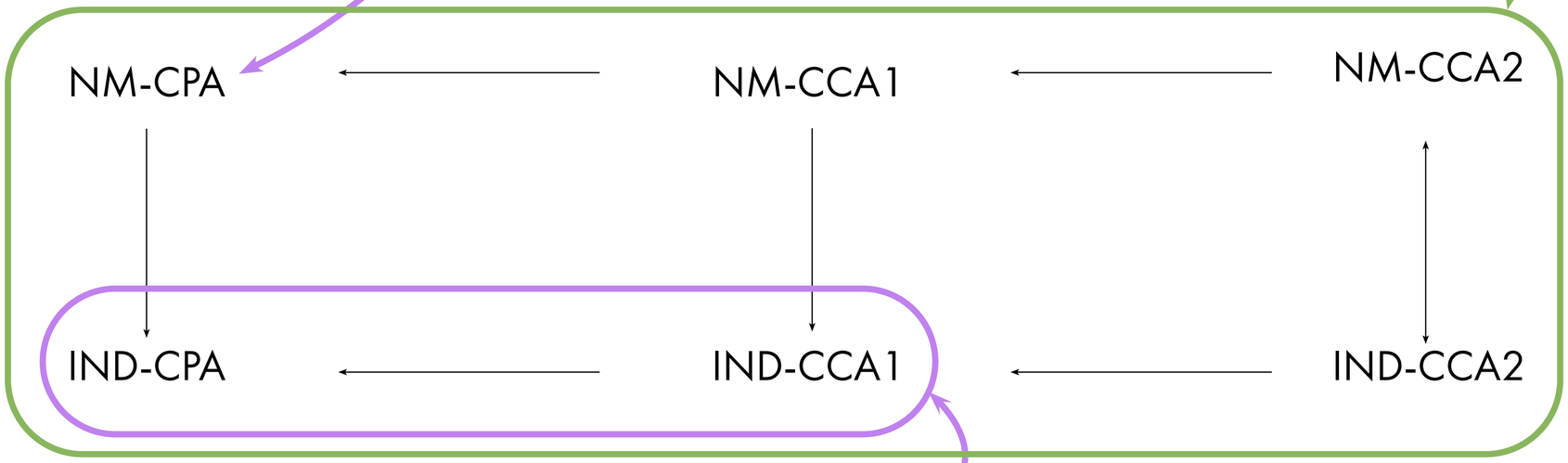
# Standard Notions

PMR is incompatible with NM.

| | |
|---|---|
| $W(\cdot)$ | hamming weight |
| $Enc_{K,R}(k,n,x) = y$ | encryption |
| $Dec_{K,R}(k,n,y') = (x',w)$ | decryption |
| $K$ | security parameter |
| $k,n$ | key, nonce |
| $x,y$ | plain text, cipher text |
| $R$ | redundancy parameter |
| $x',y'$ | plain text, cipher text with errors |
| $w \approx W(x \oplus x')$ | error estimate |

**Partial Message Recovery:**
No NM which implies no IND-CCA2: Adversary can just submit hamming distance from original plain text as the relation.

**Security Against Fuzzing:**
Extends CCA2 security (just set $w_{max}$=0)




**Partial Message Recovery:**
Encryption is randomized so IND-CCA1 is feasible.

Bellare, M., Desai, A., Pointcheval, D., & Rogaway, P. (1998, August). Relations among notions of security for public-key encryption schemes.

# Partial Message Recovery

CCA2 up to proportional loss.

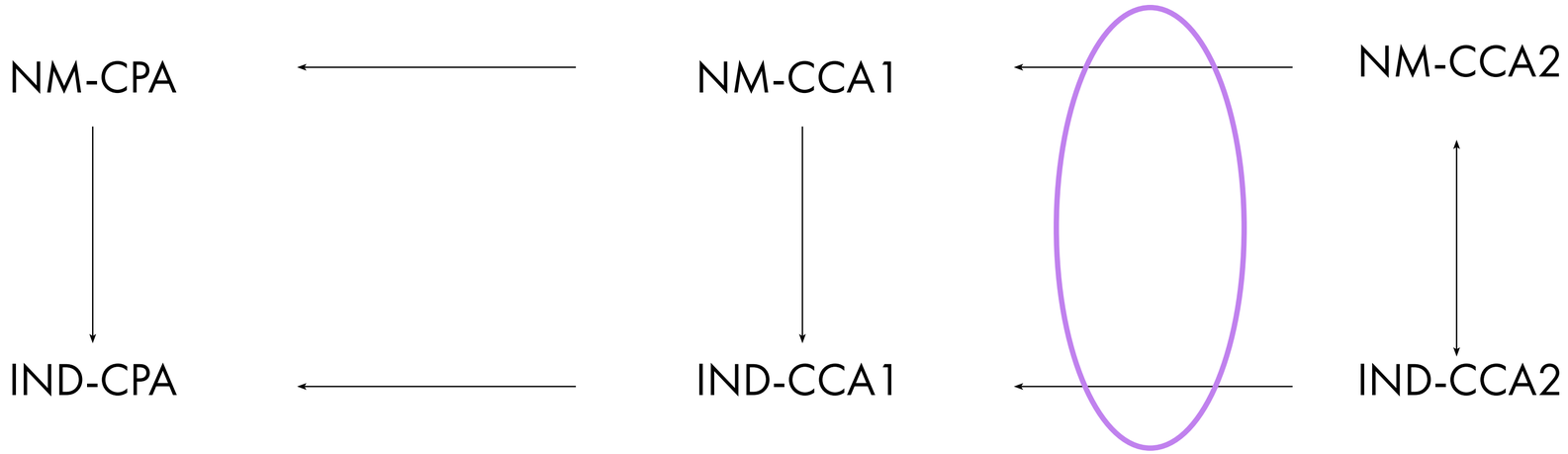
| | |
|---|---|
| $W(\cdot)$ | hamming weight |
| $Enc_{K,R}(k,n,x) = y$ | encryption |
| $Dec_{K,R}(k,n,y') = (x',w)$ | decryption |
| $K$ | security parameter |
| $k,n$ | key, nonce |
| $x,y$ | plain text, cipher text |
| $R$ | redundancy parameter |
| $x',y'$ | plain text, cipher text with errors |
| $w \approx W(x \oplus x')$ | error estimate |

**pl-IND-CCA2 "IND-CCA2 up to PL"**

1. Adv. chooses chooses msg $x$
3. Adv chooses syndromes $s_1$, $s_2$ such that $W(s_1) = W(s_2)$
4. Game chooses one at random $b \leftarrow^R \{0, 1\}$
6. Adv. winns if they can guess $b$ guess given $x'$

**pl-NM-CCA2 "NM-CCA2 up to PL"**

1. Adv. chooses chooses msg space M
2. Game chooses two msgs. $x_0$, $x_1 \leftarrow^R$ M
4. A. outputs rel. R & syndromes S of same weight
5. Game chooses rand. syndrome; same weight
7. Game chooses one $x$ at random $b \leftarrow^R \{0, 1\}$
8. Adv. wins if $R(x_b, Dec(y_1 \oplus S)) = b$



# Loss estimate unforegability

Unforgeability for the continuous domain

| | |
|---|---|
| $W(\cdot)$ | hamming weight |
| $Enc_{K,R}(k,n,x) = y$ | encryption |
| $Dec_{K,R}(k,n,y') = (x',w)$ | decryption |
| $K$ | security parameter |
| $k,n$ | key, nonce |
| $x,y$ | plain text, cipher text |
| $R$ | redundancy parameter |
| $x',y'$ | plain text, cipher text with errors |
| $w \approx W(x \oplus x')$ | error estimate |

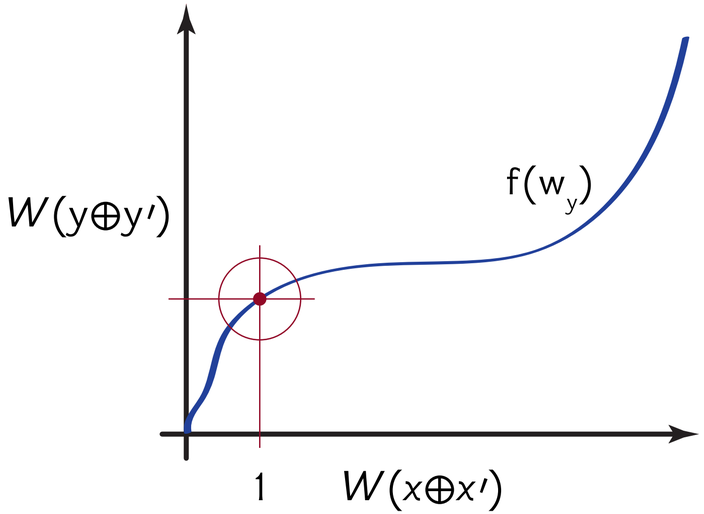
LEU-CCA2

"Producing large deviations of the error estimate from the true error level is computationally hard."

- Extra parameter: $\Delta w$
- Adapted unforgeability (SUF-CMA)
- No authentication separate from decryption, hence CCA
- Encryption oracle logs encryptions; game looks up closest ciphertext/plaintext pair
- Negligible advantage producing some y' such that $|w - W(x \oplus x')| > \Delta w_{max}$
- In K $\Delta w$

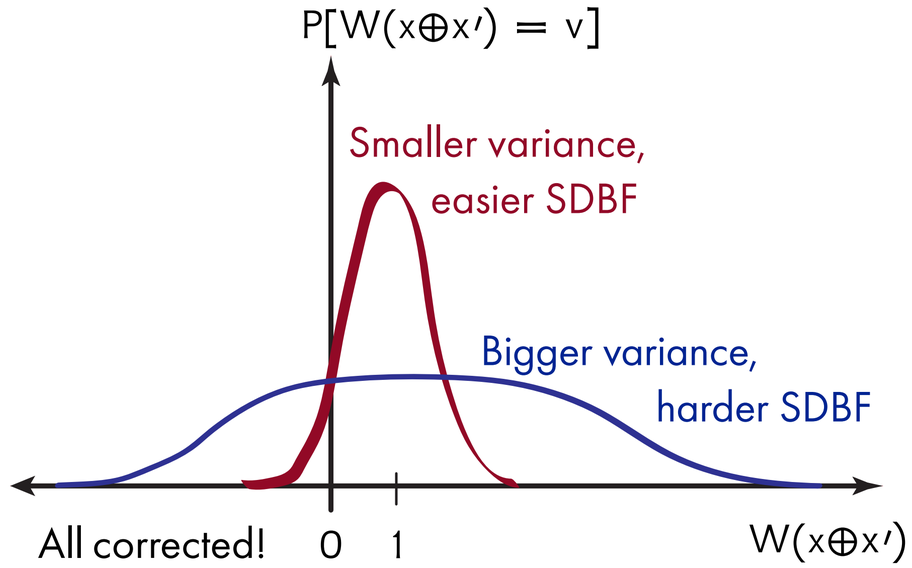
# Short Distance Brute Force

Real analysis for each use case is required to determine risk of allowing PMR.




- There is a function $f(w_y)$ from the no. of errors in cipher text to the average number in the plain text
- An adversary can determine the number of errors required to produce one error in the plaintext
- In a CCA2 scenario, an adversary can thus brute force a single, specific bit flip with probability $P \approx 1/|x|$ attempts; i.e. in $O(|x|)$ (linear complexity in the msg size) in a CCA2 setting

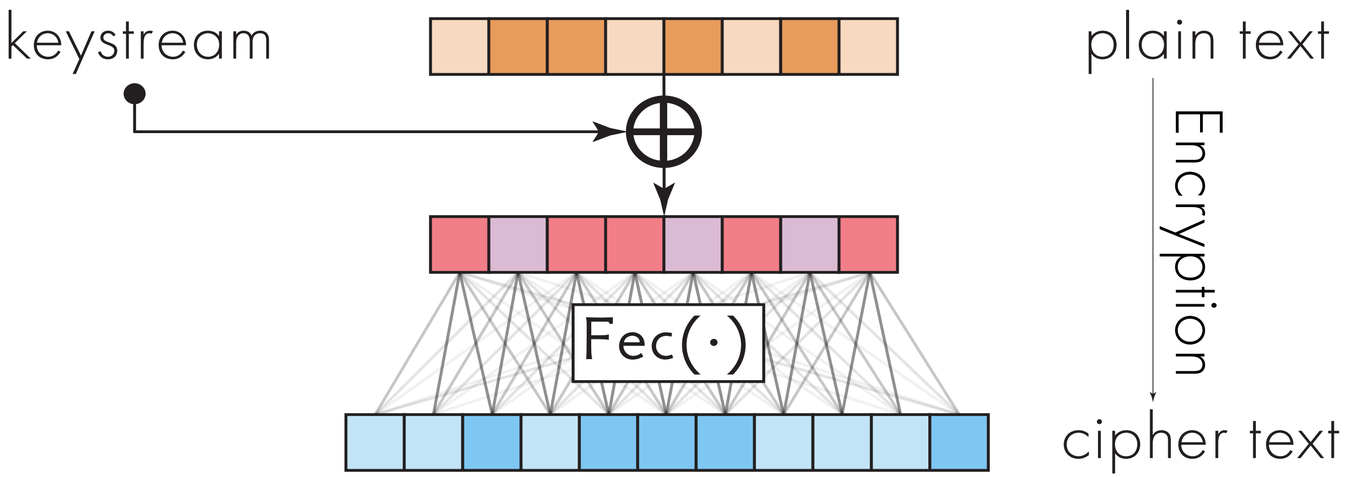
- Non-Polynomial attack complexity in message length; "search space" in the general case is given by $|x|! - (|x| - W(x \oplus x'))!$ which approaches $|x|^{W(x \oplus x')}$ or $|x|!$
- In practice the number of bit flips in $x$ is a distribution.
- We can use mitigations: increase the variance or modify the to some complex, non-gaussian distribution, but not achieve security against small plain text error vectors.



# Encryption in the random oracle model

## Unauthenticated stream cipher has great malleability but no security

- Not building a PRF or PRP from scratch
- Start with enc. in the random oracle model[1]
- Unauthenticated (auth. doesn't allow malleability)
- Should be IND-CPA secure!
- Gives us 1:1 malleability, just add FEC
- FEC fully exposed
- Will probably end up using a DECK[2] function as there is a decent amount of preexisting work on building modes based on deck functions

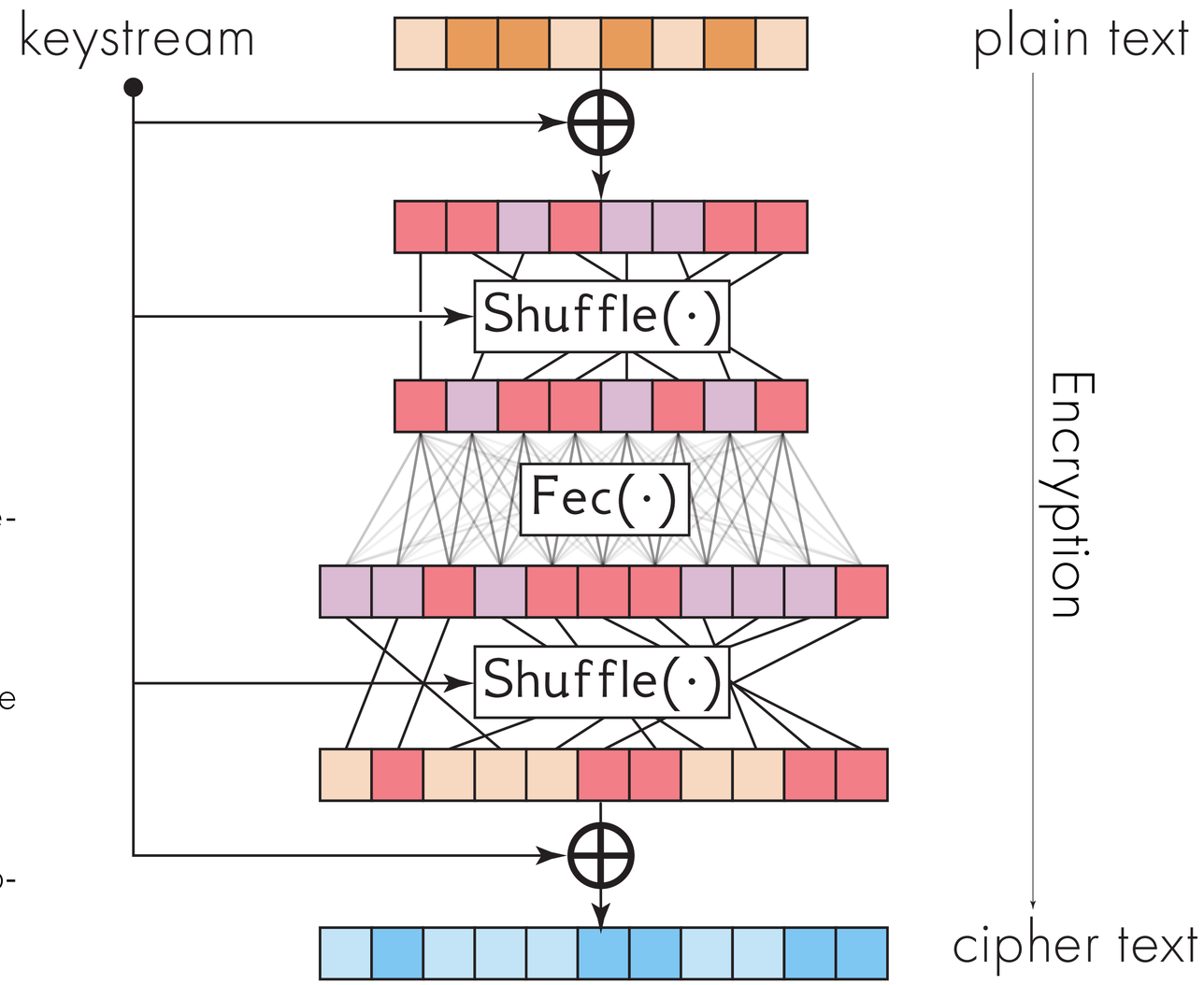



[1] Bellare, M., & Rogaway, P. (1993, December). Random oracles are practical: A paradigm for designing efficient protocols.
[2] Daemen, J., Hoffert, S., Van Assche, G., & Van Keer, R. (2018). Xoodoo cookbook

# CCA1 Setting

Shuffling the cipher text to protect location information

- FEC sandwiched in shuffles, sandwiched in XORs
- Simple transform, all security from random key stream
- XOR at end to protect against pathological cases (e.g. all zeros). One XOR might be sufficient.
- Structure easily probed in CCA2 setting; insecure if same nonce can be decrypted more than once
- Similarity with McEliece crypto system if instantiated with goppa codes & permutation matrix view is taken on shuffle

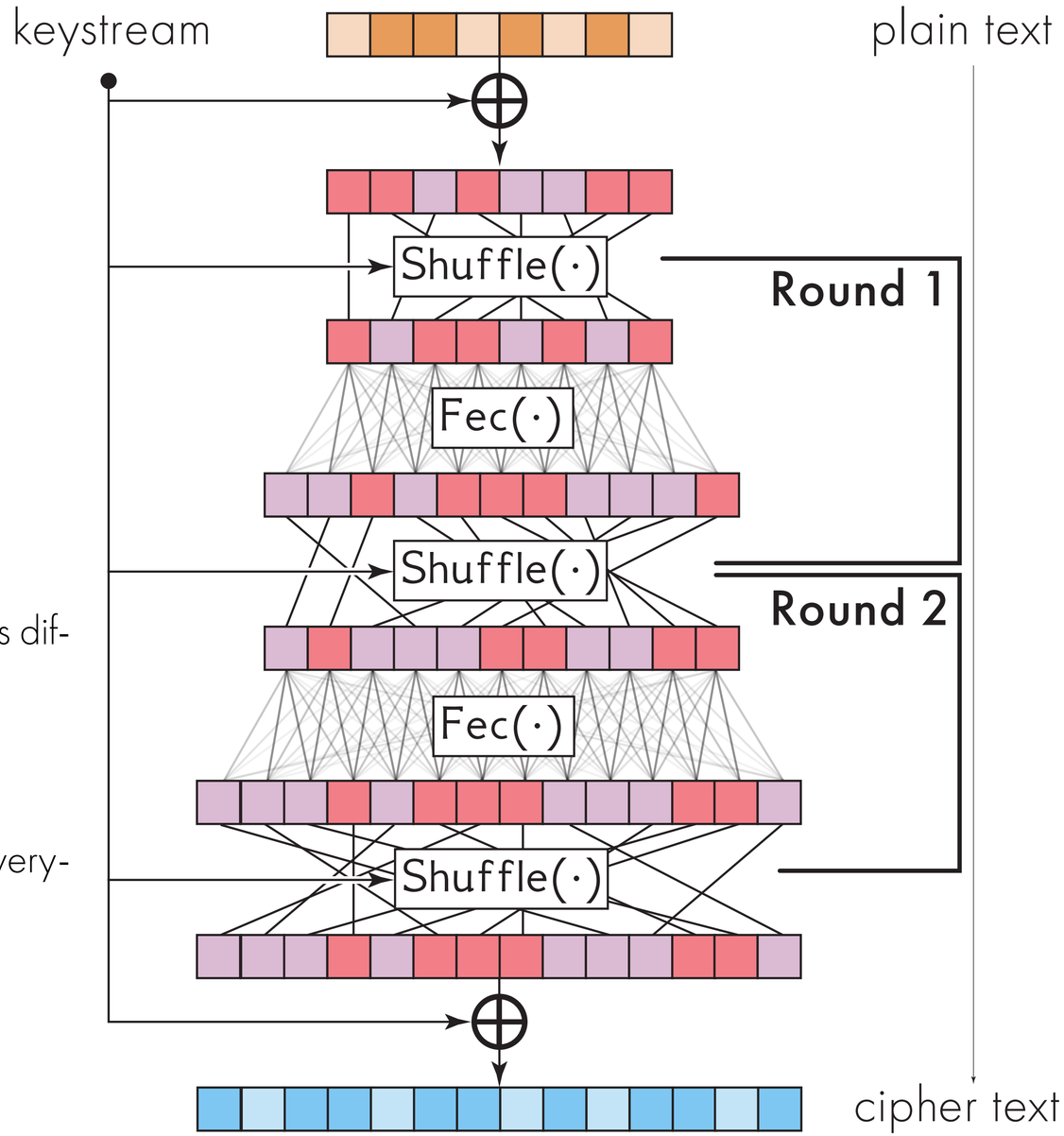


# CCA2 Setting

Protecting the internal structure of the shuffle with block-cipher-like structure

- Multiple rounds of shuffle/FEC
- Sandwich struct. is kept; two rounds can share a shuffle (linear op.)
- **FEC view:** Randomized interleaver, same FEC
- **Symmetric crypto view:** SP-Network – Shuffle for P-Box, FEC adds diffusion; block cipher with oracle key schedule
- Security from hardness of probing the shuffle pattern
- Attacker needs to probe large amount of information because everything is randomized

- Note: Security & Redundancy are proportional in this construction



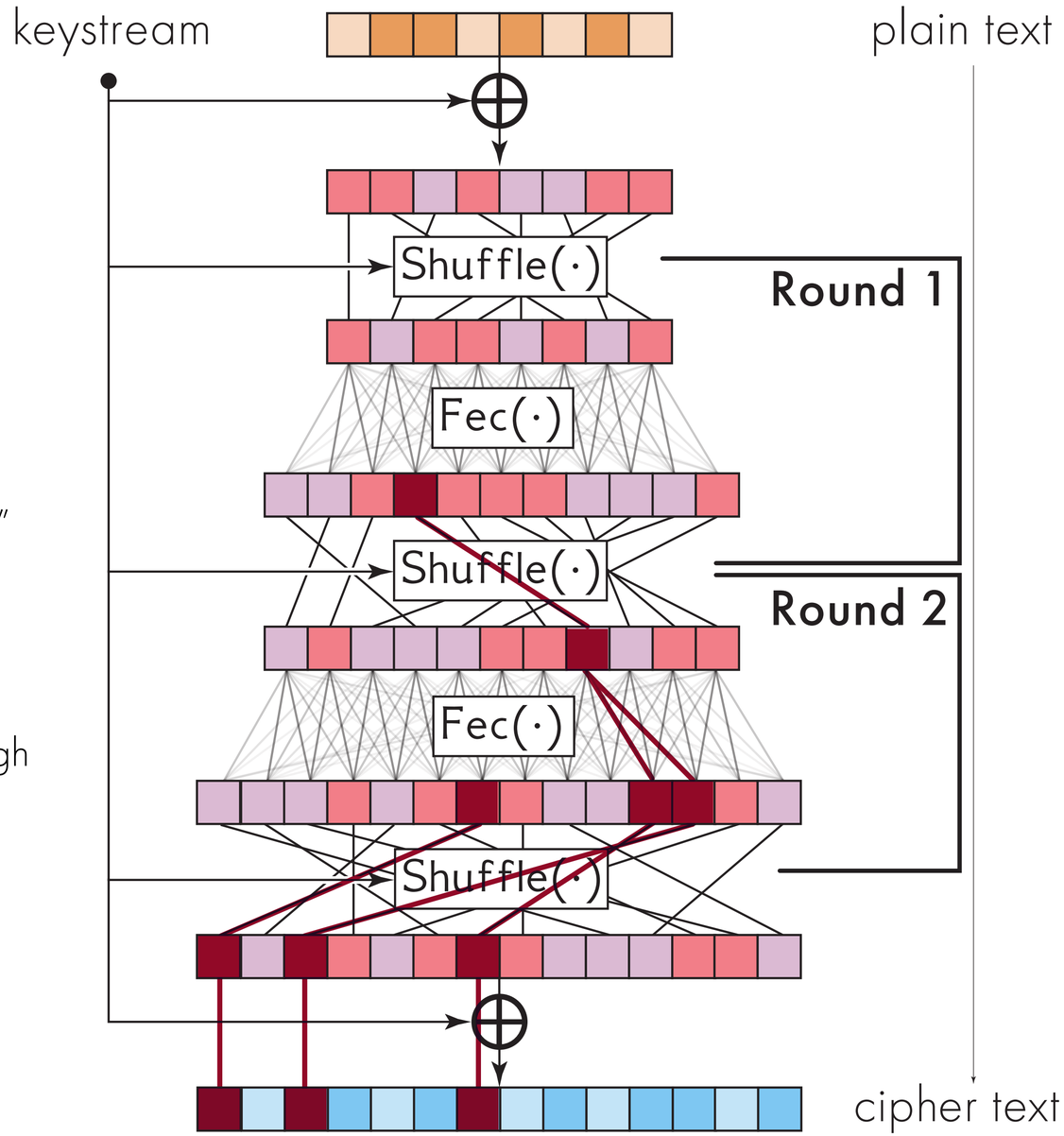
# Under CCA2 probing

## SP-Network in action!

- Flipping few bits cannot be probed (yields original plain text)
- Flipping more bits causes some bit flips to make it though
- Flipping even more bits yields unpredictable behavior;
  - Multiple close bit flips are required per area to make it “through”
  - Adding one may cause location change
  - Location change is amplified by shuffle
  - This may even decrease final number of errors that make it through

    e.g. because bit flips are more spread out
  - Final behavior depends of course on precise FEC used.

- Note: Security & Redundancy are proportional in this construction



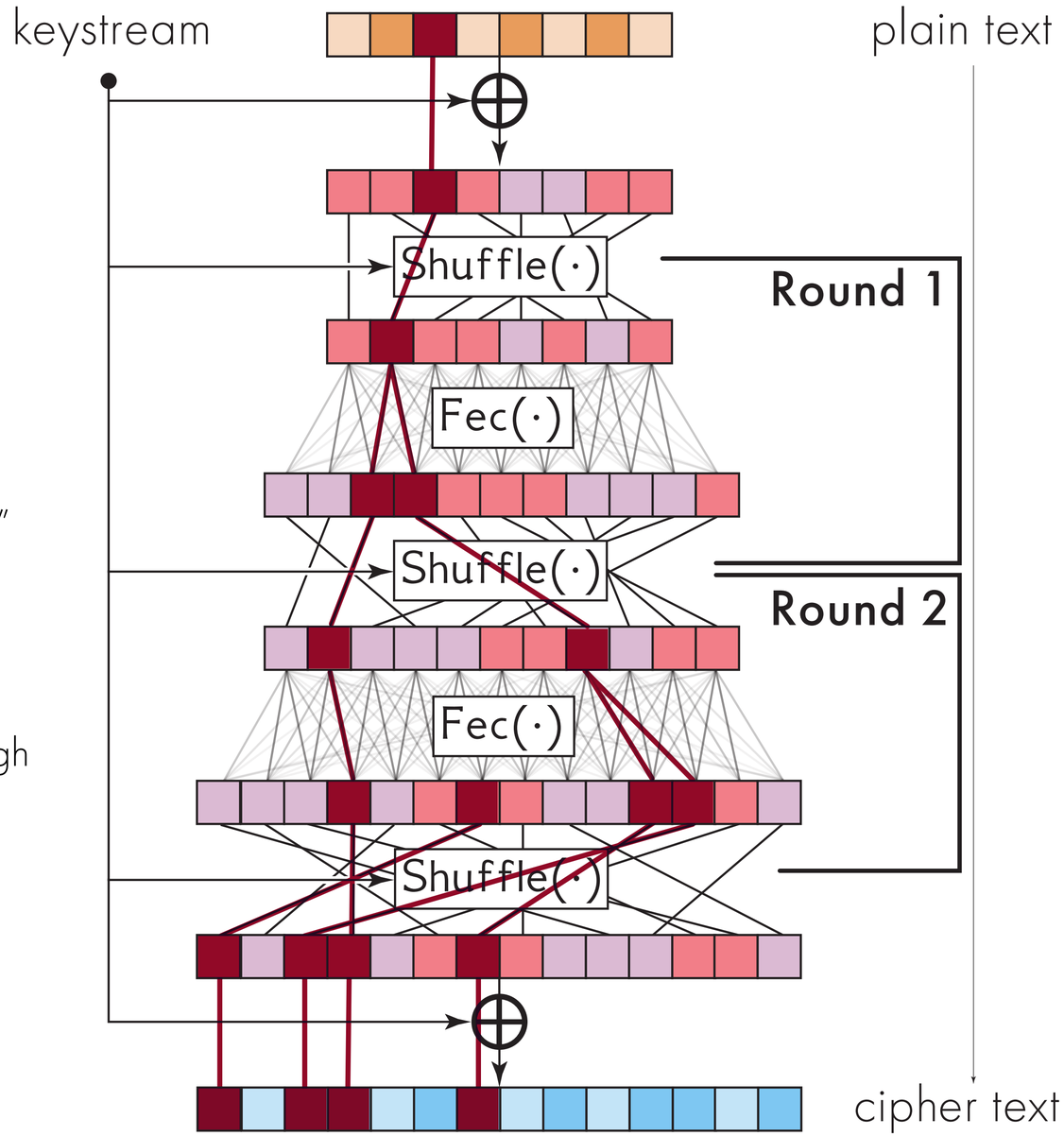
# Under CCA2 probing

## SP-Network in action!

- Flipping few bits cannot be probed (yields original plain text)
- Flipping more bits causes some bit flips to make it though
- Flipping even more bits yields unpredictable behavior;
  - Multiple close bit flips are required per area to make it “through”
  - Adding one may cause location change
  - Location change is amplified by shuffle
  - This may even decrease final number of errors that make it through

    e.g. because bit flips are more spread out
  - Final behavior depends of course on precise FEC used.

- Note: Security & Redundancy are proportional in this construction



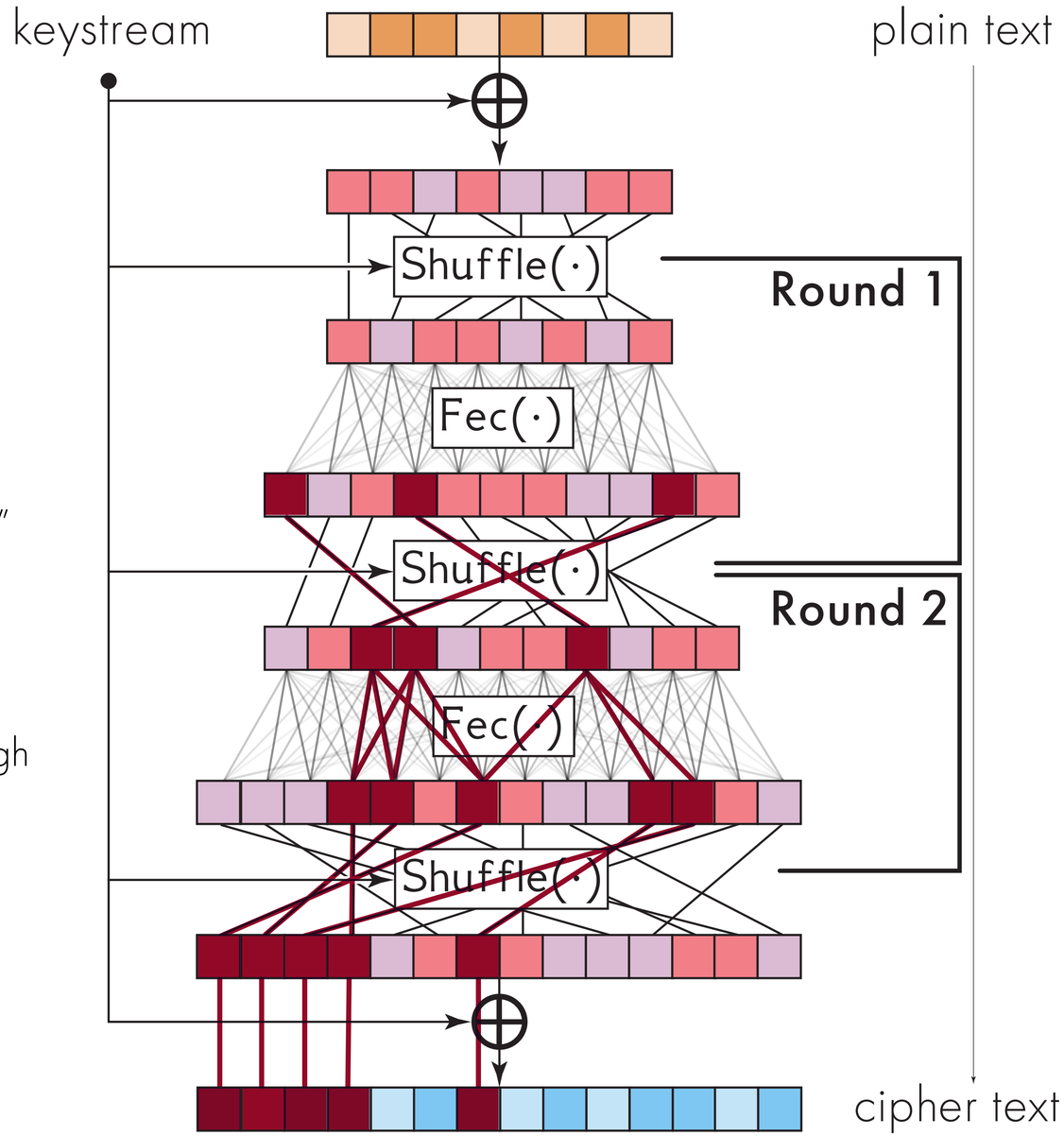
# Under CCA2 probing

## SP-Network in action!

- Flipping few bits cannot be probed (yields original plain text)
- Flipping more bits causes some bit flips to make it though
- Flipping even more bits yields unpredictable behavior;
  - Multiple close bit flips are required per area to make it “through”
  - Adding one may cause location change
  - Location change is amplified by shuffle
  - This may even decrease final number of errors that make it through

    e.g. because bit flips are more spread out
  - Final behavior depends of course on precise FEC used.

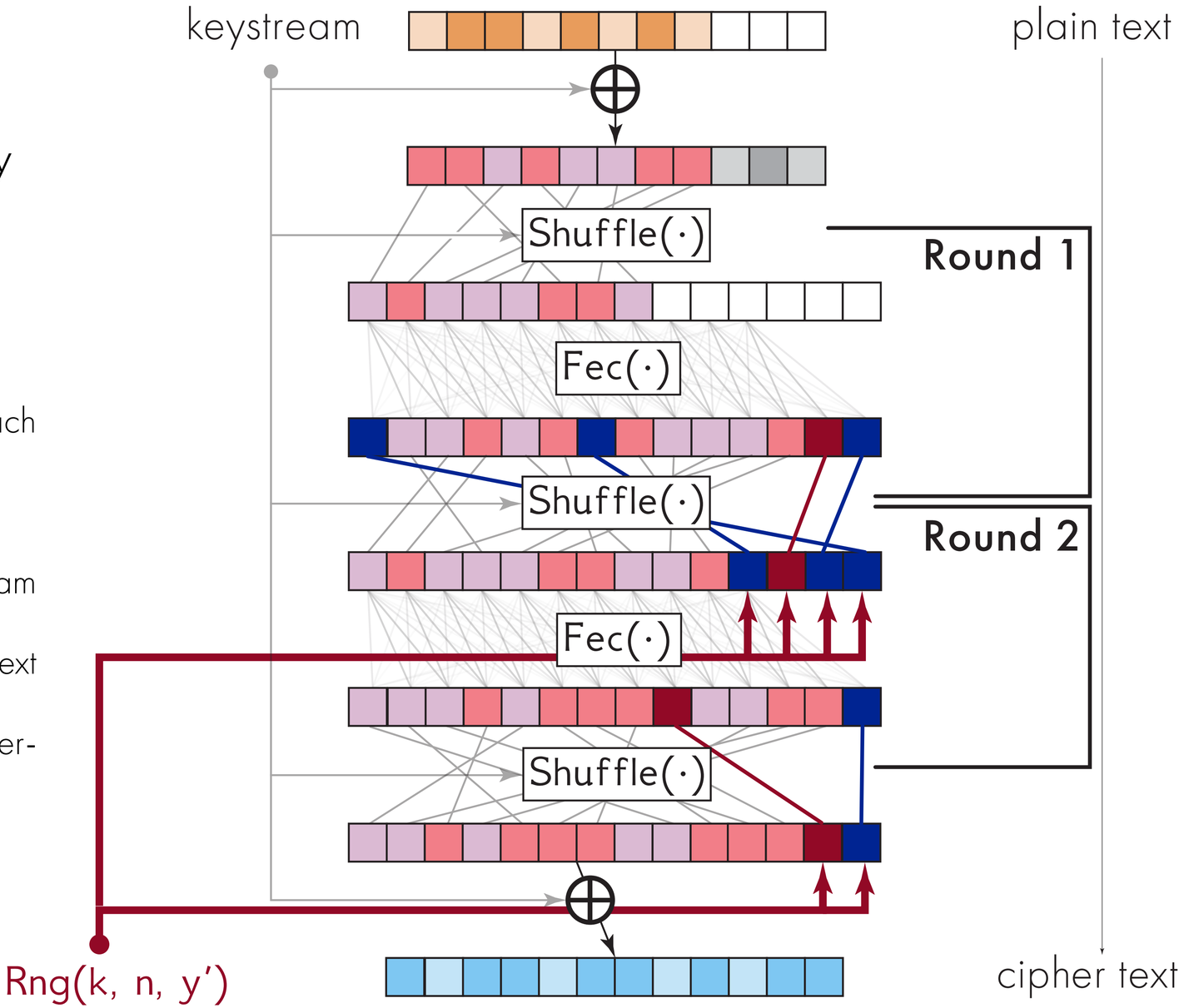
- Note: Security & Redundancy are proportional in this construction



# Decoder Randomization

## Puncturing the FEC & injecting entropy for increased efficiency!

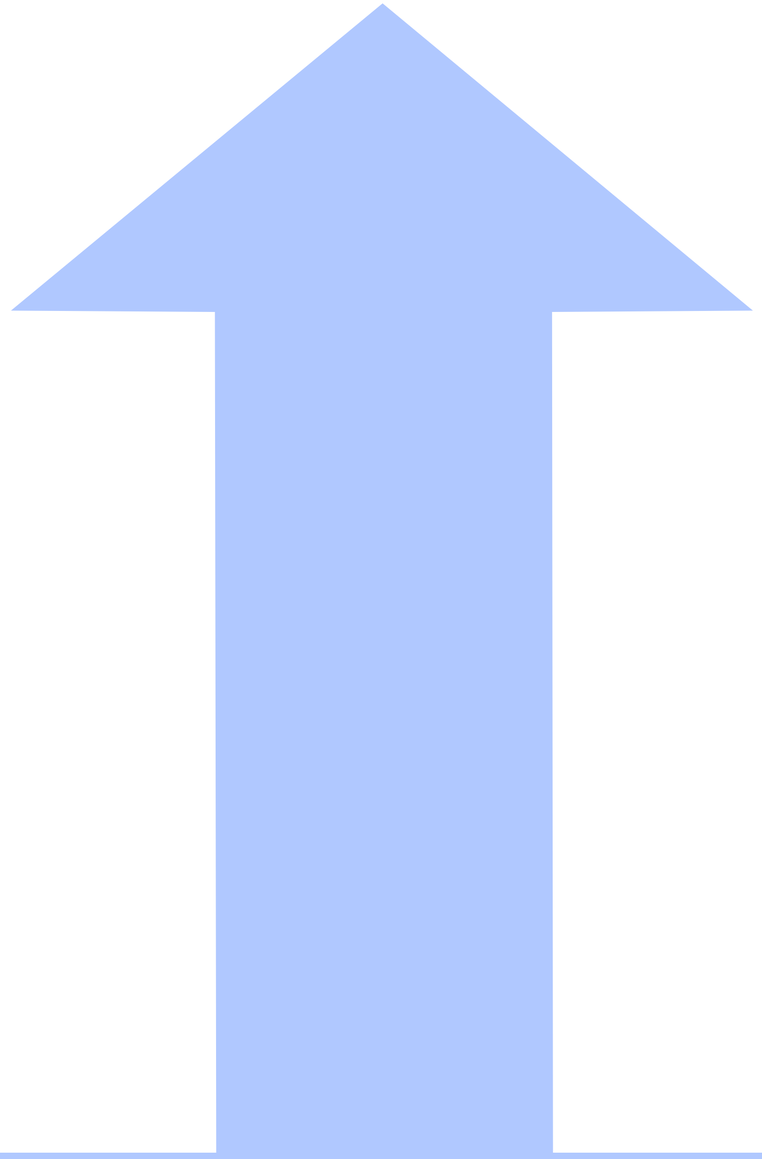
- Use puncturing to reduce redundancy
- XOF MAC: Assigns a secret decoding stream to each adversary chosen error pattern
- Fill in punctured values with values from decoding stream
- Defeats probing effect of small changes to the cipher text
- Puncturing/Padding requires careful design to avoid error floors and similar effects
- Reintroducing redundancy by using FEC as post processing (errors are now randomly distributed)



# Next Steps

Instantiation and formal security proof.

- Lots of small things to iron out (how to generate distance assessment, simplify security definitions, etc...)
- Instantiate the scheme (select RNG, shuffle, FEC)
- Create a proof of security (should be possible given reliance on PRF/XOF/DECK for oracle)
- Analysis of the properties of the FEC-permutation network in other settings (e.g. instantiate with goppa codes to see whether this provides an advantage in PQ-Crypto)
- Extensions to the scheme: CCA2 variant, long messages, as a FEC with random interleaver



# Decryption Despite Errors

Combining FEC and encryption to achieve security against fuzzers and improve band-width efficiency.

Karolin Varner ~ karo@cupdev.net

https://github.com/koraa/decryption-despite-errors

## Use Case

**Partial Message Recovery:** Improving transmission efficiency by exploiting inherent redundancy in data

**Security against Fuzzing:** Guarantees data can be transmitted as long as some bits make it through

## How it works

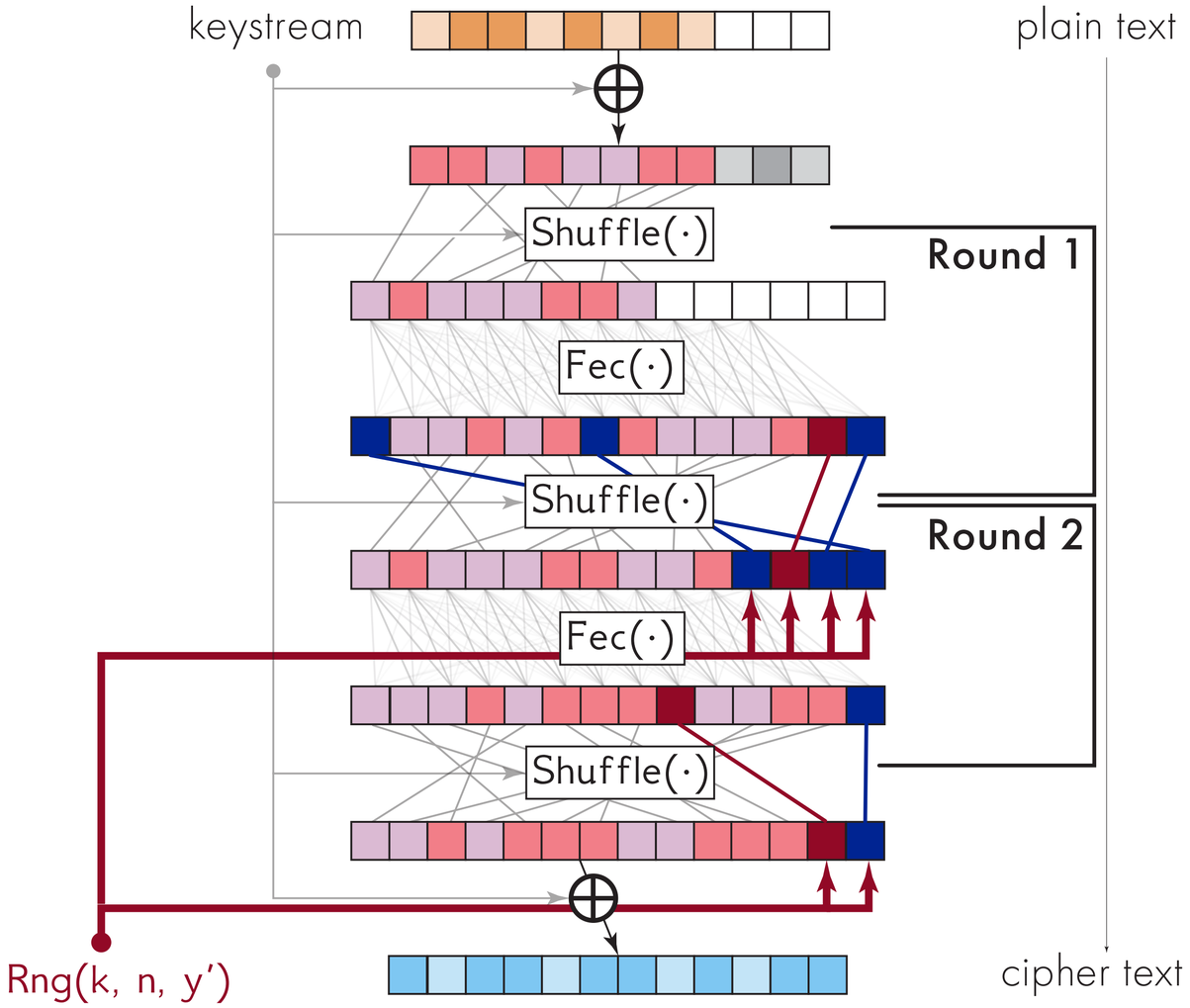
Shuffle cipher text before & after FEC to mask location. Multiple rounds to protect structure against CCA2 attacks. Randomization to increase efficiency.

## Formal Security Definitions

**FEC-CCA2:** Increasing the redundancy level guarantees more errors can be corrected.

**pl-IND-CCA2/pl-NM-CCA2:** CCA2 security up to proportional loss

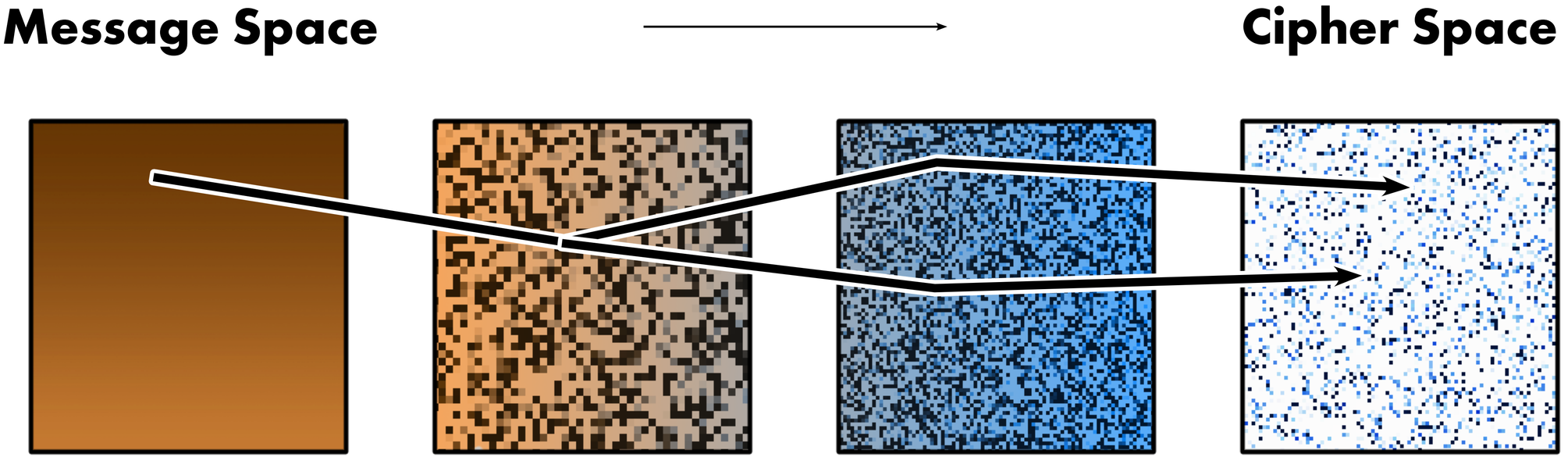
**LEU-CCA2:** Unforgeability of the loss estimate.




Thank you Benjamin Lipp

# Effects of Nonces and MACs on message space

Nonce for CPA, MAC for CCA security




**Permutation**
Randomizes message space

**Nonce**
CPA Security: Many possible cipher texts per message

**MAC**
CCA Security: Many invalid messages per valid message