# Non-malleable Randomness Encoders and their Applications

Bhavana Kanukurthi    Sai Lakshmi Bhavana Obbattu
**Sruthi Sekar**

Indian Institute of Science, Bangalore

3rd May 2018

A coding scheme $(Enc, Dec)$ satisfying

# Non-malleable Codes: Dziembowski, Pietrzak and Wichs [ICS 2010]

A coding scheme $(Enc, Dec)$ satisfying

- Correctness

# Non-malleable Codes: Dziembowski, Pietrzak and Wichs [ICS 2010]

A coding scheme $(Enc, Dec)$ satisfying

- Correctness: $\forall m,\ \Pr[Dec(Enc(m)) = m] = 1$

# Non-malleable Codes: Dziembowski, Pietrzak and Wichs [ICS 2010]

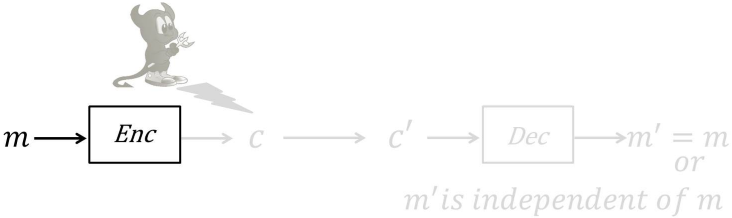
A coding scheme $(Enc, Dec)$ satisfying

- Correctness: $\forall m,\ \Pr[Dec(Enc(m)) = m] = 1$
- Non-malleability:

# Non-malleable Codes: Dziembowski, Pietrzak and Wichs [ICS 2010]

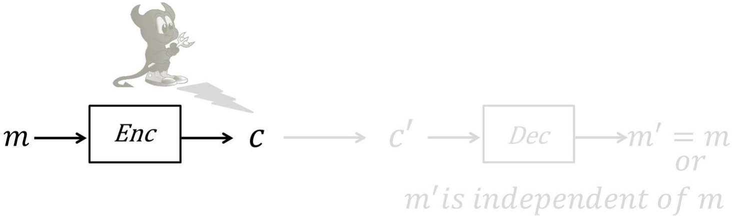
A coding scheme $(Enc, Dec)$ satisfying

- Correctness: $\forall m,\ \Pr[Dec(Enc(m)) = m] = 1$
- Non-malleability:



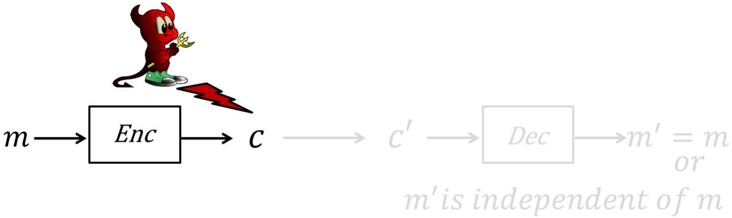
A coding scheme $(Enc, Dec)$ satisfying

- Correctness: $\forall m,\ \Pr[Dec(Enc(m)) = m] = 1$
- Non-malleability:



A coding scheme $(Enc, Dec)$ satisfying

- Correctness: $\forall m,\ \Pr[Dec(Enc(m)) = m] = 1$
- Non-malleability:



# Non-malleable Codes: Dziembowski, Pietrzak and Wichs [ICS 2010]

A coding scheme $(Enc, Dec)$ satisfying

- Correctness: $\forall m,\ \Pr[Dec(Enc(m)) = m] = 1$
- Non-malleability:

$$m \longrightarrow \boxed{Enc} \longrightarrow c \xrightarrow{f \in \mathcal{F}} c' \longrightarrow \boxed{Dec} \longrightarrow m' = m$$

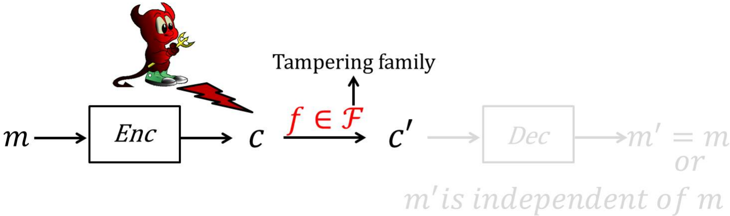
or

$m'$ is independent of $m$

# Non-malleable Codes: Dziembowski, Pietrzak and Wichs [ICS 2010]

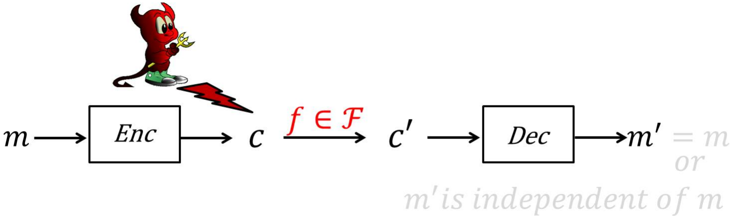
A coding scheme $(Enc, Dec)$ satisfying

- Correctness: $\forall m,\ \Pr[Dec(Enc(m)) = m] = 1$
- Non-malleability:



A coding scheme $(Enc, Dec)$ satisfying

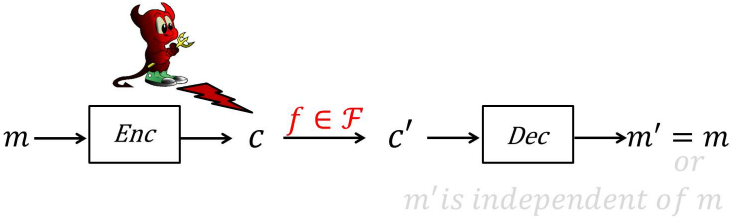
- Correctness: $\forall m,\ \Pr[Dec(Enc(m)) = m] = 1$
- Non-malleability:



# Non-malleable Codes: Dziembowski, Pietrzak and Wichs [ICS 2010]

A coding scheme $(Enc, Dec)$ satisfying

- Correctness: $\forall m,\ \Pr[Dec(Enc(m)) = m] = 1$
- Non-malleability:



# Non-malleable Codes: Dziembowski, Pietrzak and Wichs [ICS 2010]

A coding scheme $(Enc, Dec)$ satisfying

- Correctness: $\forall m,\ \Pr[Dec(Enc(m)) = m] = 1$
- Non-malleability:

$$m \longrightarrow \boxed{Enc} \longrightarrow c \xrightarrow{f \in \mathcal{F}} c' \longrightarrow \boxed{Dec} \longrightarrow m' = m$$

or

$m'$ is independent of $m$

# Non-malleable Codes: Dziembowski, Pietrzak and Wichs [ICS 2010]

- Non-malleability:

$$m \longrightarrow \boxed{Enc} \longrightarrow c \xrightarrow{f \in \mathcal{F}} c' \longrightarrow \boxed{Dec} \longrightarrow m'$$

# Non-malleable Codes: Dziembowski, Pietrzak and Wichs [ICS 2010]

- Non-malleability:

$$m \longrightarrow \boxed{Enc} \longrightarrow c \xrightarrow{f \in \mathcal{F}} c' \longrightarrow \boxed{Dec} \longrightarrow m'$$

$(Enc, Dec)$ is $\epsilon$-non-malleable with respect to $\mathcal{F}$ if

# Non-malleable Codes: Dziembowski, Pietrzak and Wichs [ICS 2010]

- Non-malleability:

$$m \longrightarrow \boxed{Enc} \longrightarrow c \xrightarrow{f \in \mathcal{F}} c' \longrightarrow \boxed{Dec} \longrightarrow m'$$

$(Enc, Dec)$ is $\epsilon$-non-malleable with respect to $\mathcal{F}$ if

$$\forall\, m,\ Tamper_f^m \approx_\epsilon Sim_f$$

| $Tamper_f^m$ | $Sim_f$ |
|---|---|
| $m' := Dec(f(Enc(m)))$<br>O/P: $m'$ | $m' \leftarrow Sim_f$<br>O/P: $m'$ |

# Non-malleable Codes: Dziembowski, Pietrzak and Wichs [ICS 2010]

- Non-malleability:

$$m \longrightarrow \boxed{Enc} \longrightarrow c \xrightarrow{f \in \mathcal{F}} c' \longrightarrow \boxed{Dec} \longrightarrow m'$$

$(Enc, Dec)$ is $\epsilon$-non-malleable with respect to $\mathcal{F}$ if

$$\forall\, m,\ Tamper_f^m \approx_\epsilon Sim_f$$

$$\boxed{\begin{array}{c} m' := Dec(f(Enc(m))) \\ \text{O/P: } m' \end{array}} \qquad \boxed{\begin{array}{c} m' \leftarrow Sim_f \\ \text{O/P: } m' \end{array}}$$

# Non-malleable Codes: Dziembowski, Pietrzak and Wichs [ICS 2010]

- Non-malleability:

$$m \longrightarrow \boxed{Enc} \longrightarrow c \xrightarrow{f \in \mathcal{F}} c' \longrightarrow \boxed{Dec} \longrightarrow m'$$

$(Enc, Dec)$ is $\epsilon$-non-malleable with respect to $\mathcal{F}$ if

$\forall f \in \mathcal{F}, \exists\, Sim_f$ such that

$$\forall\, m,\ Tamper_f^m \approx_\epsilon Sim_f$$

$$\boxed{\begin{array}{c} m' := Dec(f(Enc(m))) \\ \text{O/P: } m' \end{array}} \qquad \boxed{\begin{array}{c} m' \leftarrow Sim_f \\ \text{O/P: } m' \end{array}}$$

- Non-malleability:

$$m \longrightarrow \boxed{Enc} \longrightarrow c \xrightarrow{f \in \mathcal{F}} c' \longrightarrow \boxed{Dec} \longrightarrow m'$$

$(Enc, Dec)$ is $\epsilon$-non-malleable with respect to $\mathcal{F}$ if

$\forall f \in \mathcal{F}, \exists\, Sim_f$ such that

$$\forall\, m,\ Tamper_f^m \approx_\epsilon Sim_f$$

| $Tamper_f^m$ | $Sim_f$ |
|---|---|
| $m' := Dec(f(Enc(m)))$<br>O/P: $m'$ | $m' \leftarrow Sim_f$<br>O/P: $m'$ |

# Non-malleable Codes: Dziembowski, Pietrzak and Wichs [ICS 2010]

- Non-malleability:

$$m \longrightarrow \boxed{Enc} \longrightarrow c \xrightarrow{f \in \mathcal{F}} c' \longrightarrow \boxed{Dec} \longrightarrow m'$$

$(Enc, Dec)$ is $\epsilon$-non-malleable with respect to $\mathcal{F}$ if

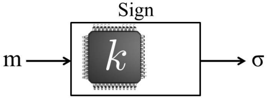
$\forall f \in \mathcal{F}, \exists\, Sim_f$ such that

$$\forall\, m,\ Tamper_f^m \approx_\epsilon Sim_f$$

$$\boxed{\begin{array}{c} m' := Dec(f(Enc(m))) \\ \text{O/P: } m' \end{array}} \qquad \boxed{\begin{array}{c} m' \leftarrow Sim_f \\ \text{O/P: } m' \end{array}}$$

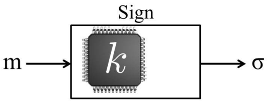
# An Application: Related Key Attacks

**Digital Signature Scheme**



# An Application: Related Key Attacks

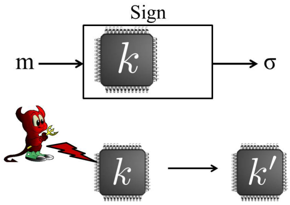
**Digital Signature Scheme**




- Standard security: Given oracle access to “Sign", adversary can’t forge signature on a new message.

# An Application: Related Key Attacks

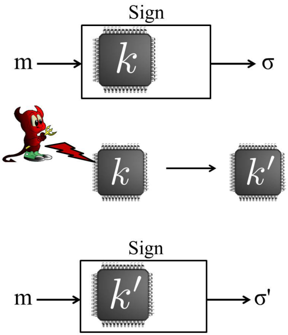
**Digital Signature Scheme**




- Standard security: Given oracle access to “Sign", adversary can’t forge signature on a new message.
- But what if adversary tampers the device and modifies $k$?

# An Application: Related Key Attacks

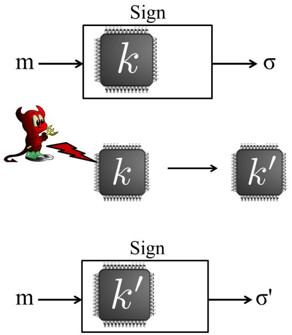
**Digital Signature Scheme**




- Standard security: Given oracle access to "Sign", adversary can't forge signature on a new message.
- But what if adversary tampers the device and modifies $k$?
- Adversary sees a signature on a related key.

# An Application: Related Key Attacks

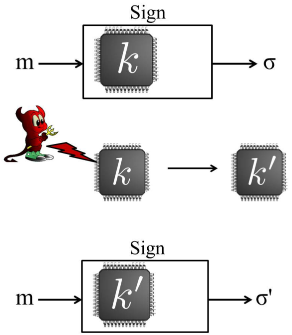
**Digital Signature Scheme**




- Standard security: Given oracle access to “Sign", adversary can’t forge signature on a new message.
- But what if adversary tampers the device and modifies $k$?
- Adversary sees a signature on a related key.
- Security of signature scheme not guaranteed!

# An Application: Related Key Attacks

**Digital Signature Scheme**

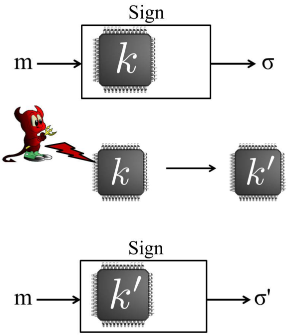



- Standard security: Given oracle access to “Sign", adversary can’t forge signature on a new message.
- But what if adversary tampers the device and modifies $k$?
- Adversary sees a signature on a related key.
- Security of signature scheme not guaranteed!

How to get security against this?

# An Application: Related Key Attacks

**Digital Signature Scheme**




- Standard security: Given oracle access to "Sign", adversary can't forge signature on a new message.
- But what if adversary tampers the device and modifies $k$?
- Adversary sees a signature on a related key.
- Security of signature scheme not guaranteed!

How to get security against this?

**Non-malleable codes**

# Non-malleable Codes: Parameters

- **Tampering family**:

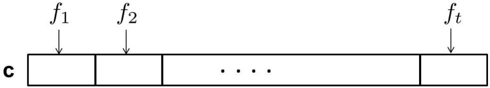
# Non-malleable Codes: Parameters

- **Tampering family**:

- **Rate**:

# Non-malleable Codes: Parameters

- **Tampering family**: Commonly studied tampering family is the *t-split-state family*:

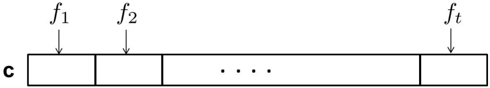



$$\mathcal{F}_t = \{(f_1, \cdots, f_t) : f_i : \{0,1\}^{n/t} \to \{0,1\}^{n/t} \text{ for each } i\}$$

- **Rate**:

# Non-malleable Codes: Parameters

- **Tampering family**:

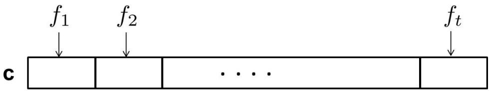



Lower the value of $t \rightarrow$ More powerful Adversary

- **Rate**:

# Non-malleable Codes: Parameters

- **Tampering family**:

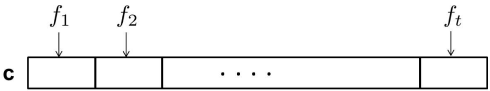



Lower the value of $t$ $\rightarrow$ More powerful Adversary

- **Rate**: $\dfrac{\text{message length}}{\text{codeword length}}$

# Non-malleable Codes: Parameters

- **Tampering family**:

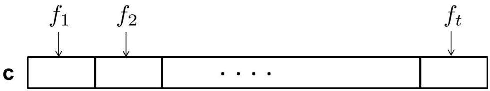



Lower the value of $t \rightarrow$ More powerful Adversary

- **Rate**: Higher rate $\rightarrow$ Lower redundancy

# Non-malleable Codes: Parameters

- **Tampering family**:

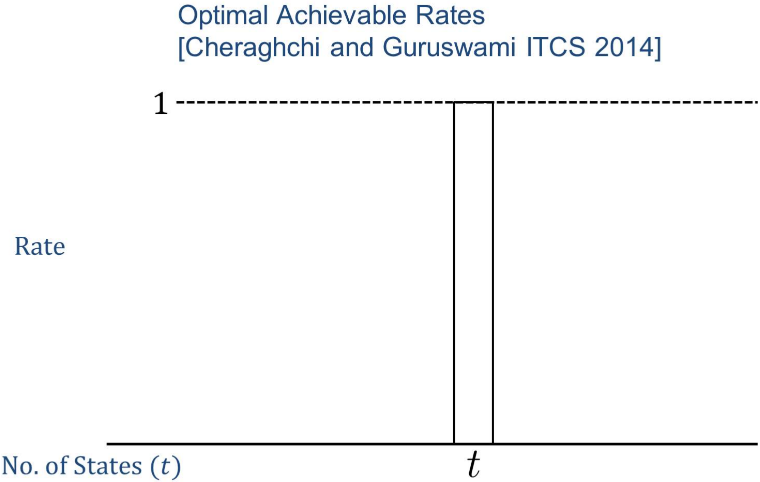



Lower the value of $t \rightarrow$ More powerful Adversary

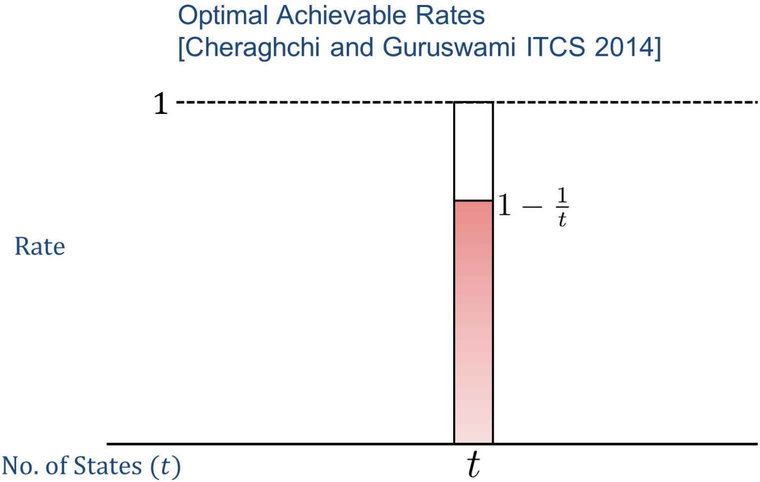
- **Rate**: Higher rate $\rightarrow$ Lower redundancy

*Holy Grail*: Build **optimal rate** NMCs for $\mathcal{F}_2$

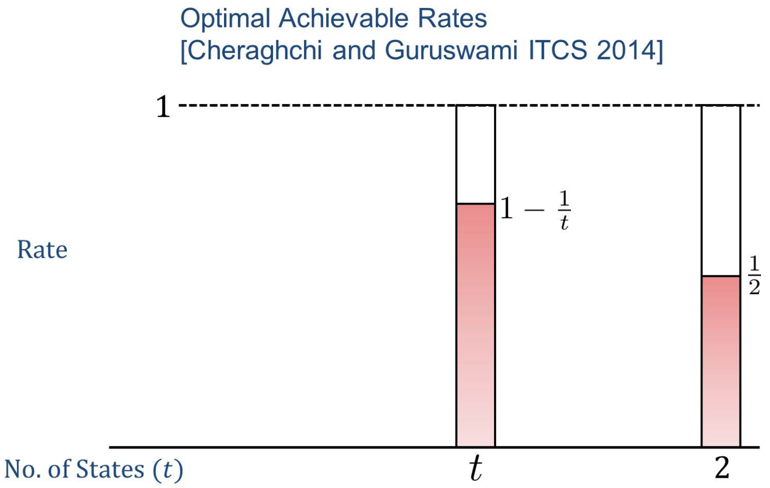
# Motivating NMREs



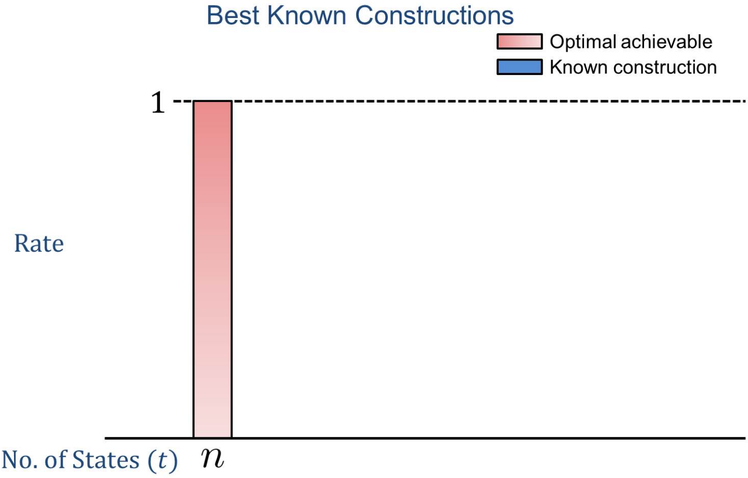
# Motivating NMREs



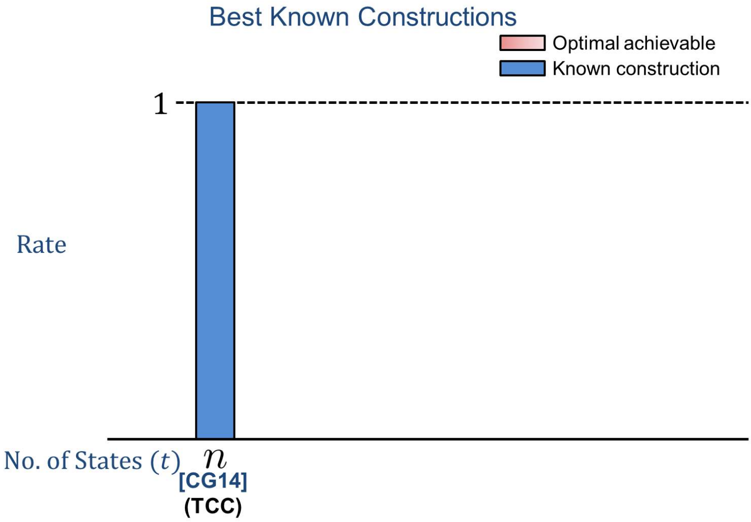
# Motivating NMREs



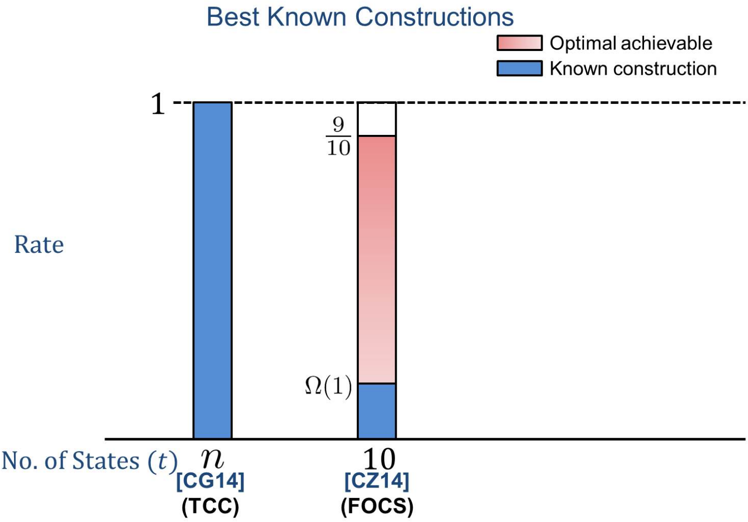
# Motivating NMREs



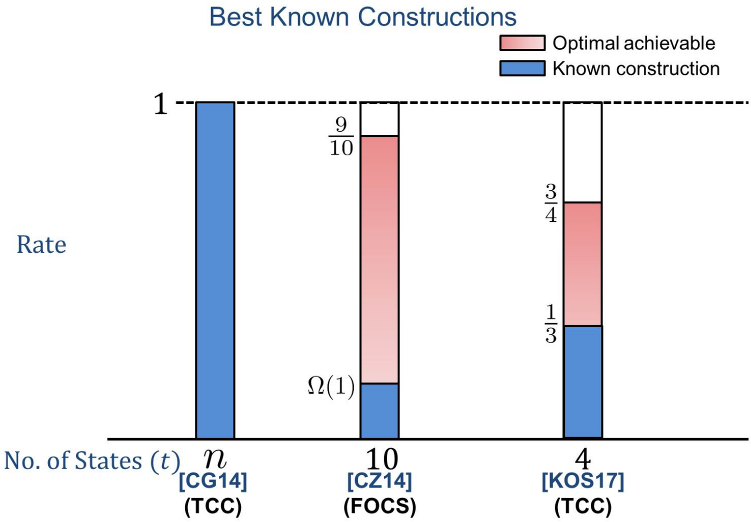
# Motivating NMREs



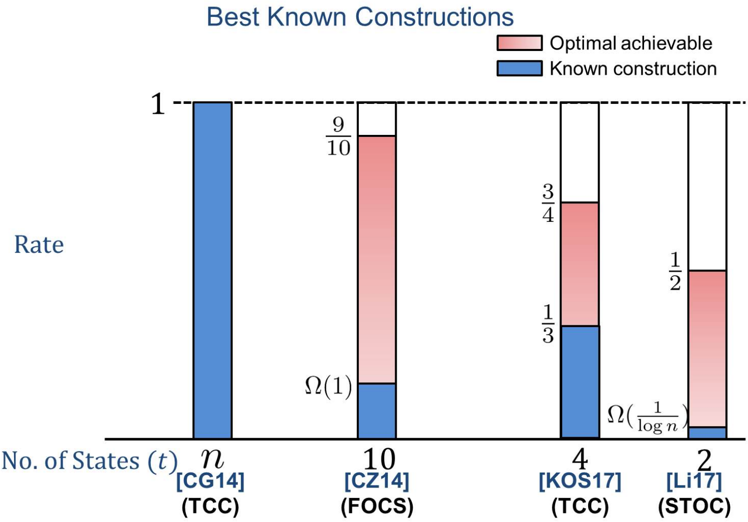
# Motivating NMREs



# Motivating NMREs



# Motivating NMREs



# Motivating NMREs

- No constant rate NMCs for $t < 4$.

# Motivating NMREs

- No constant rate NMCs for $t < 4$.
- NMCs give strong guarantee of non-malleability for every message.

# Motivating NMREs

- No constant rate NMCs for $t < 4$.
- NMCs give strong guarantee of non-malleability for every message.

Question: Can we do better for random messages?

# Motivating NMREs

- No constant rate NMCs for $t < 4$.
- NMCs give strong guarantee of non-malleability for every message.

Question: Can we do better for random messages?

*This work*: Non-malleable Randomness Encoders (NMREs)

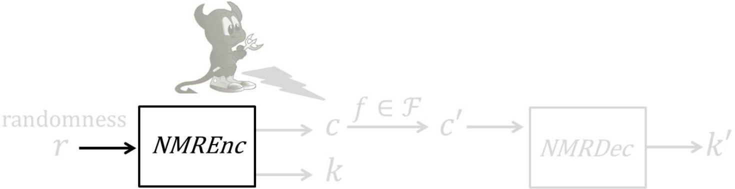
# Motivating NMREs

- No constant rate NMCs for $t < 4$.
- NMCs give strong guarantee of non-malleability for every message.

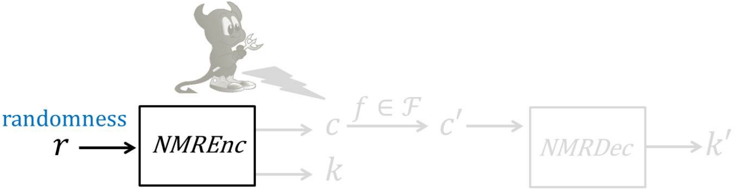
Question: Can we do better for random messages?

*This work*: 2-state, $1/2$-rate NMRE

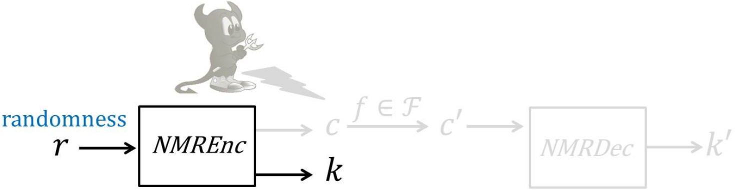
# Non-malleable Randomness Encoders (NMREs)



# Non-malleable Randomness Encoders (NMREs)

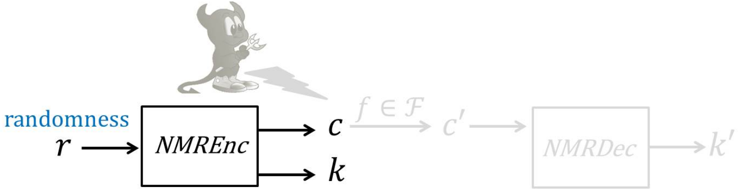


# Non-malleable Randomness Encoders (NMREs)



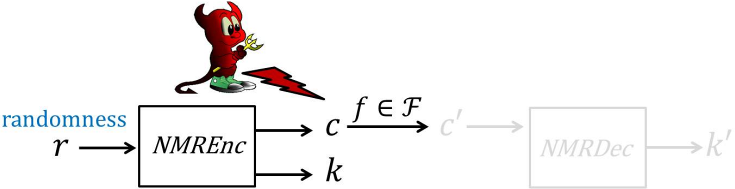

- A *random message* $k$ is generated

# Non-malleable Randomness Encoders (NMREs)



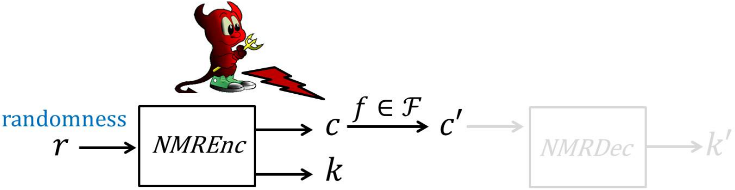

- A *random message* $k$ is generated along with its corresponding *non-malleable encoding* $c$.

# Non-malleable Randomness Encoders (NMREs)



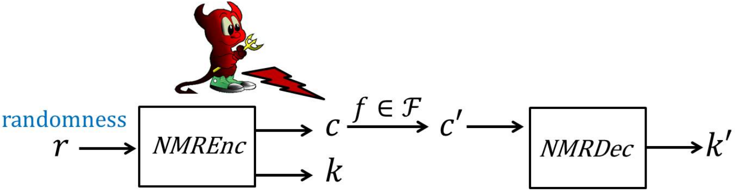

- A *random message* $k$ is generated along with its corresponding *non-malleable encoding* $c$.

# Non-malleable Randomness Encoders (NMREs)



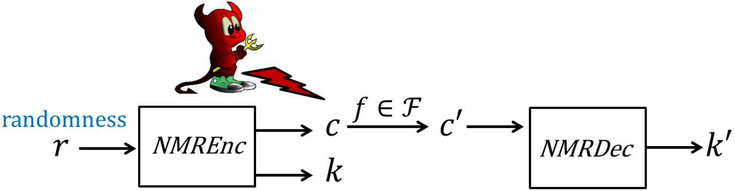

- A *random message* $k$ is generated along with its corresponding *non-malleable encoding* $c$.

# Non-malleable Randomness Encoders (NMREs)



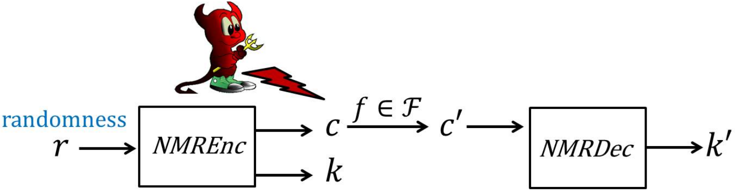

- A *random message* $k$ is generated along with its corresponding *non-malleable encoding* $c$.

# Non-malleable Randomness Encoders (NMREs)



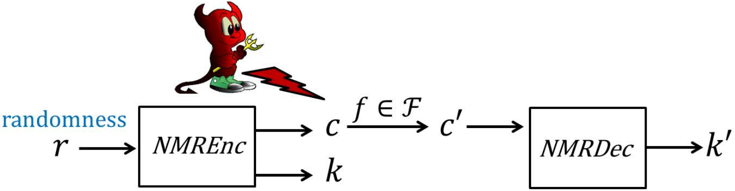

- A *random message* $k$ is generated along with its corresponding *non-malleable encoding* $c$.
- *Informal definition*: If $c$ is tampered by $f \in \mathcal{F}$ to $c'$, then

# Non-malleable Randomness Encoders (NMREs)



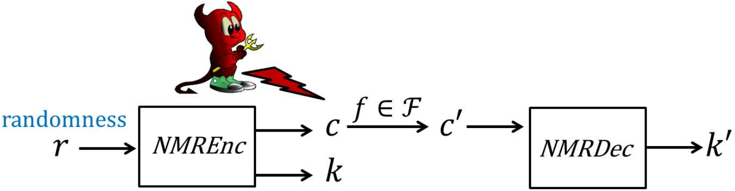

- A *random message* $k$ is generated along with its corresponding *non-malleable encoding* $c$.
- *Informal definition*: If $c$ is tampered by $f \in \mathcal{F}$ to $c'$, then
  - either $k' = k$

# Non-malleable Randomness Encoders (NMREs)




- A *random message* $k$ is generated along with its corresponding *non-malleable encoding* $c$.
- *Informal definition*: If $c$ is tampered by $f \in \mathcal{F}$ to $c'$, then
    - either $k' = k$
    - or $k$ looks uniform, even given $k'$.

# Non-malleable Randomness Encoders (NMREs)

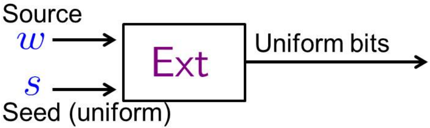



- A *random message* $k$ is generated along with its corresponding *non-malleable encoding* $c$.
- *Informal definition*: If $c$ is tampered by $f \in \mathcal{F}$ to $c'$, then
  - either $k' = k$
  - or $k$ looks uniform, even given $k'$.
- Any NMC is by default a secure NMRE.

# Journey Ahead

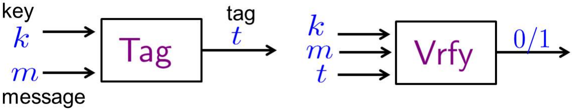
- Building blocks
- Motivating the construction
- Our construction
- Security proof

Converts non-uniform source string to a uniform string




$$S, \mathsf{Ext}(W; S) \approx S, U$$

MAC is composed algorithms $(\mathsf{Tag}, \mathsf{Vrfy})$:



# Building Blocks

## Two-state NMC

A Non-malleable code $(\mathsf{NMEnc}, \mathsf{NMDec})$ w.r.t. to $\mathcal{F}_2$

# Building Blocks

## Two-state NMC

A Non-malleable code (NMEnc, NMDec) w.r.t. to $\mathcal{F}_2$

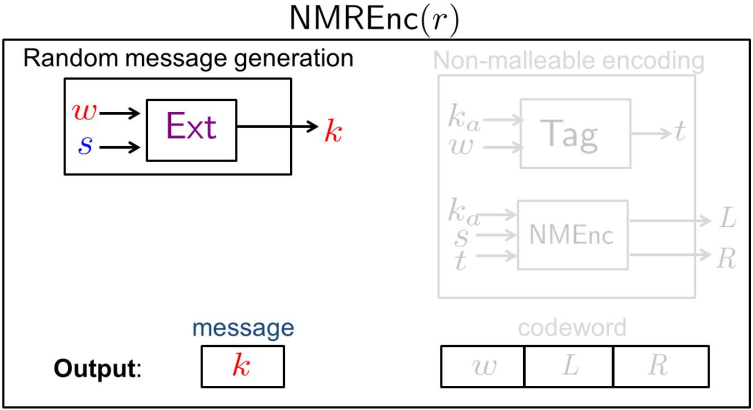

- Can be any 2-state NMC.
  Specific instantiation: [Li17]

# Building Blocks
## Two-state NMC

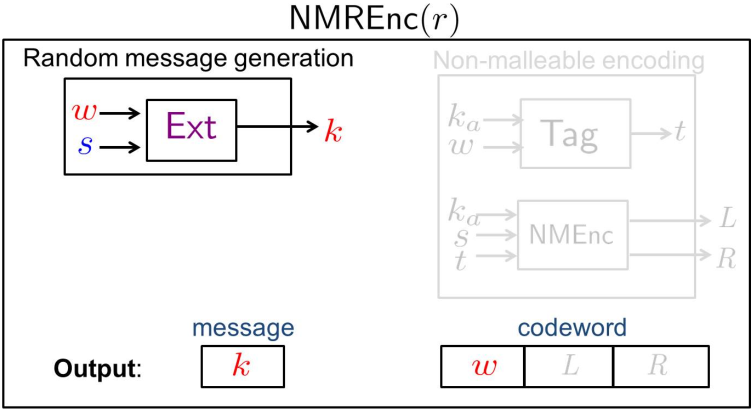
A Non-malleable code $(\mathsf{NMEnc}, \mathsf{NMDec})$ w.r.t. to $\mathcal{F}_2$


- Can be any 2-state NMC.
  Specific instantiation: [Li17]
- Used to encode short messages only

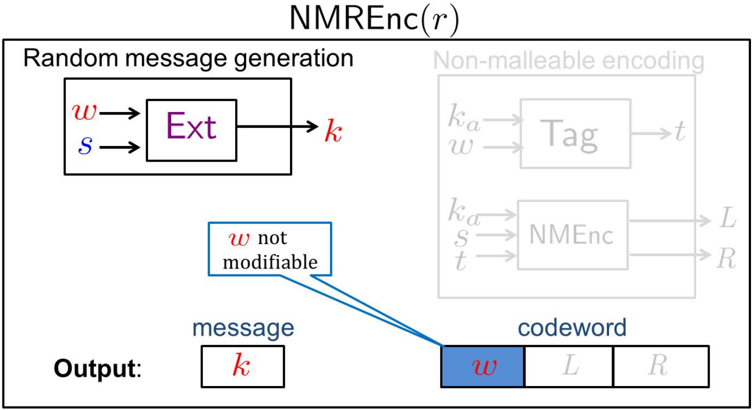
# Motivating our construction



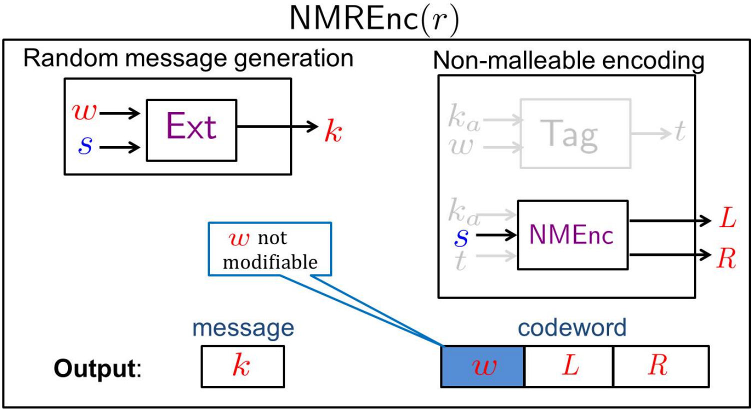
# Motivating our construction



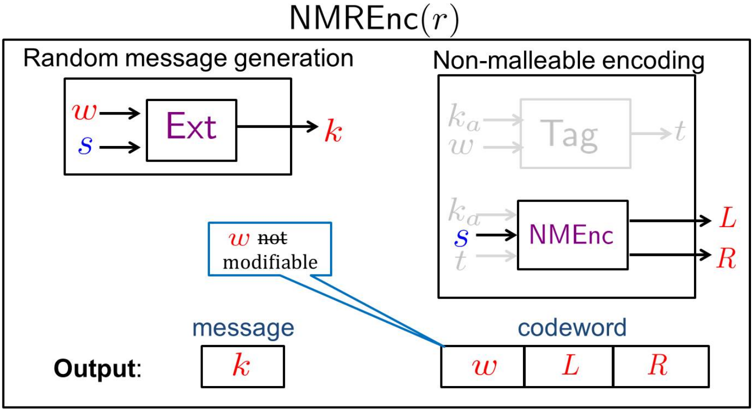
# Motivating our construction



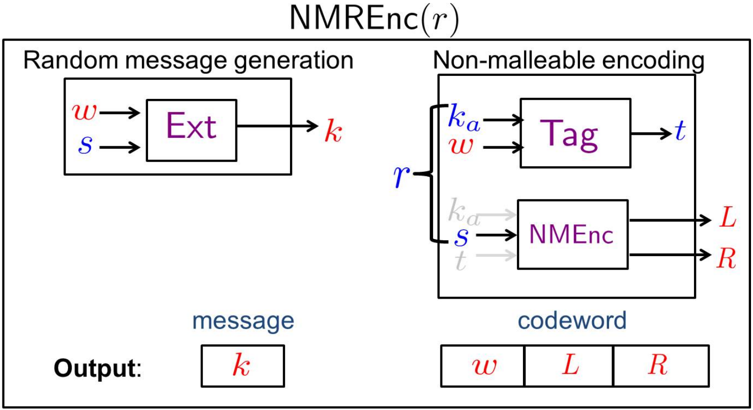
# Motivating our construction



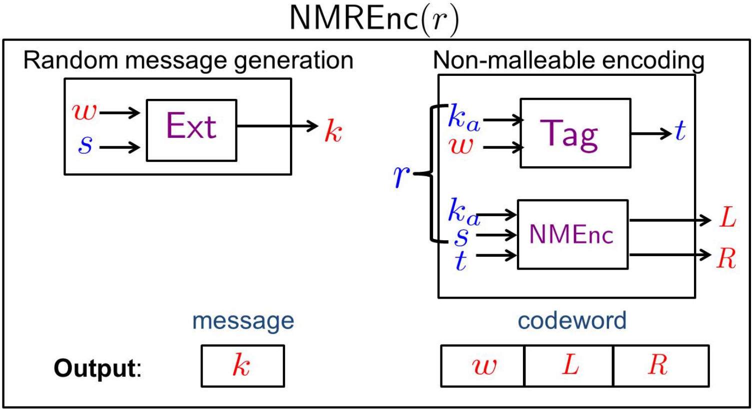
# Motivating our construction



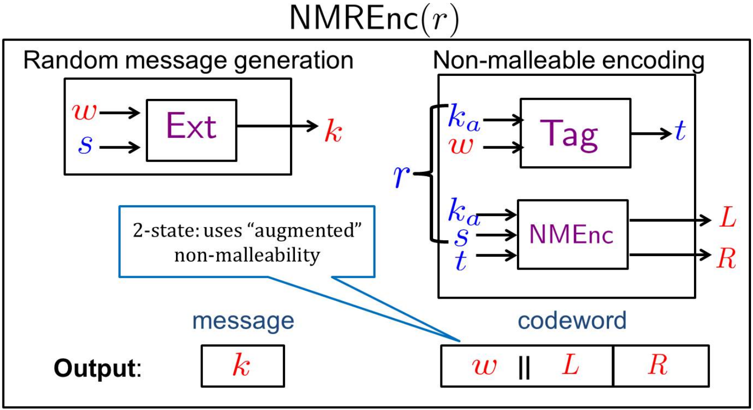
# Motivating our construction



# Motivating our construction



# Motivating our construction



# Security Proof

Goal: Build a simulator $\mathsf{NMRSim}_{f,g}$, similar to NMCs.

$\mathsf{NMRSim}_{f,g}$

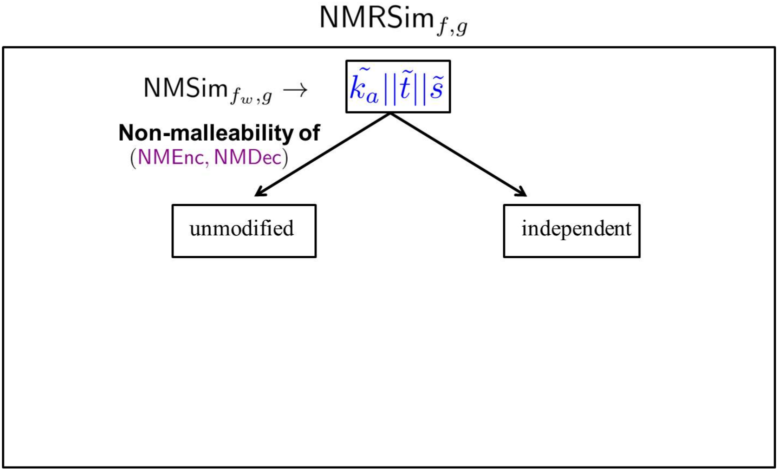
# Security Proof

To do this, we use the simulator for NMC, $\mathsf{NMSim}$ in black box.

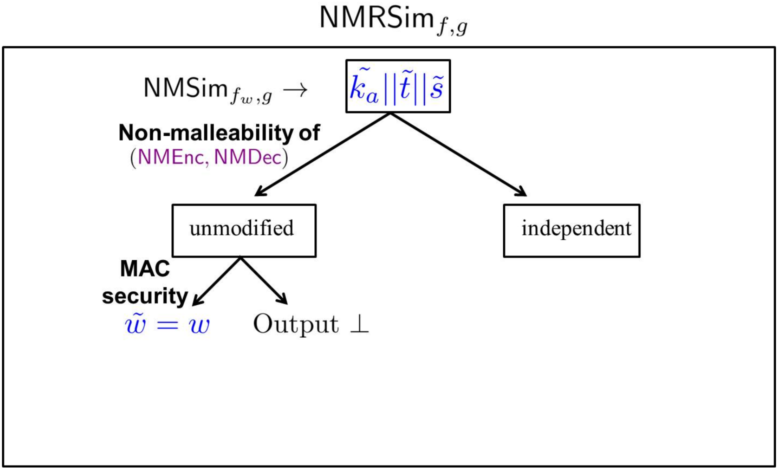
$\mathsf{NMRSim}_{f,g}$

$\mathsf{NMSim}_{f_w,g} \rightarrow \boxed{\tilde{k_a}||\tilde{t}||\tilde{s}}$

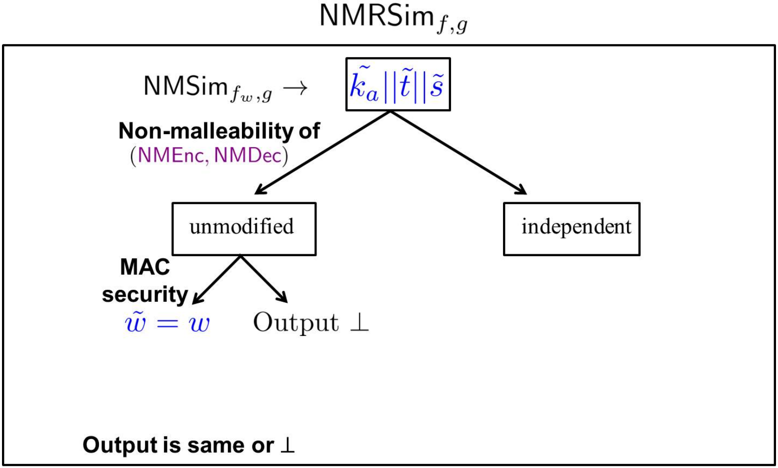
# Security Proof



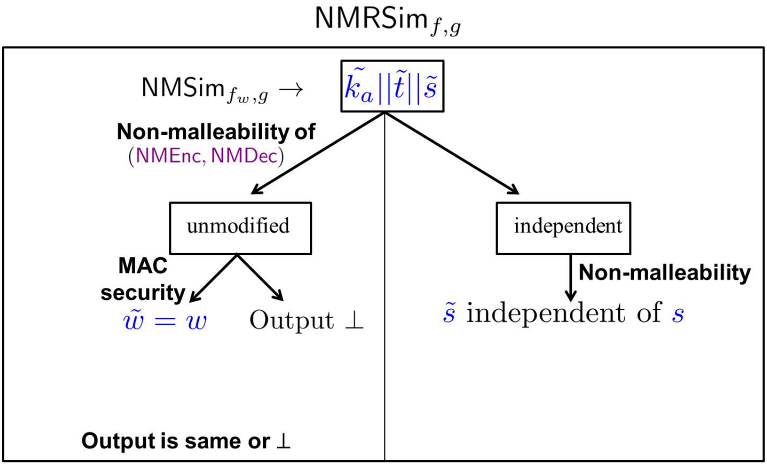
# Security Proof



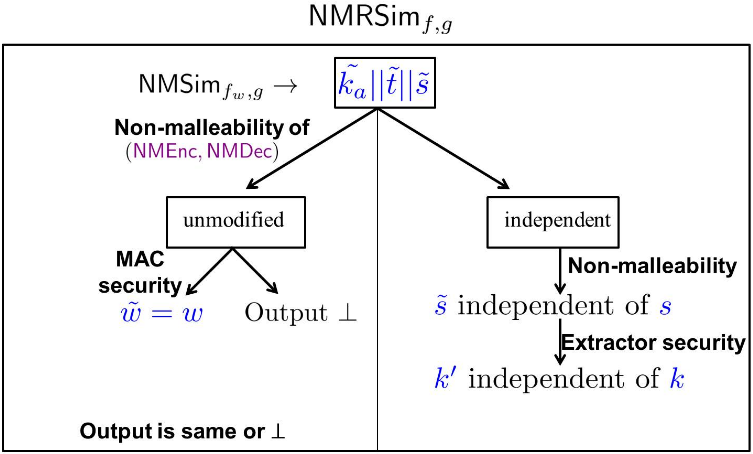
# Security Proof



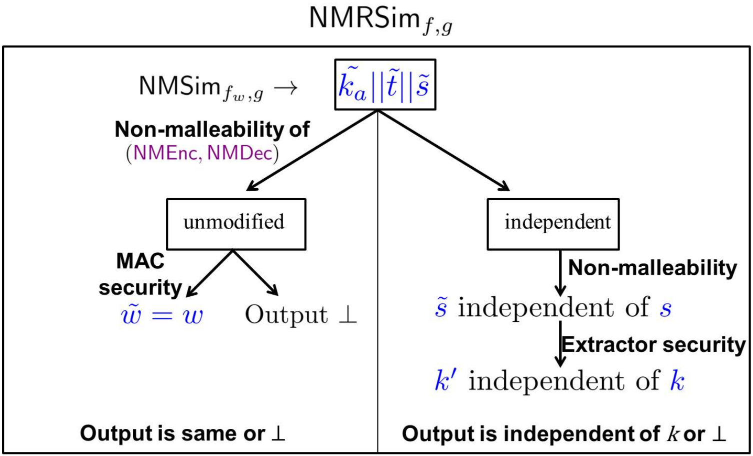
# Security Proof



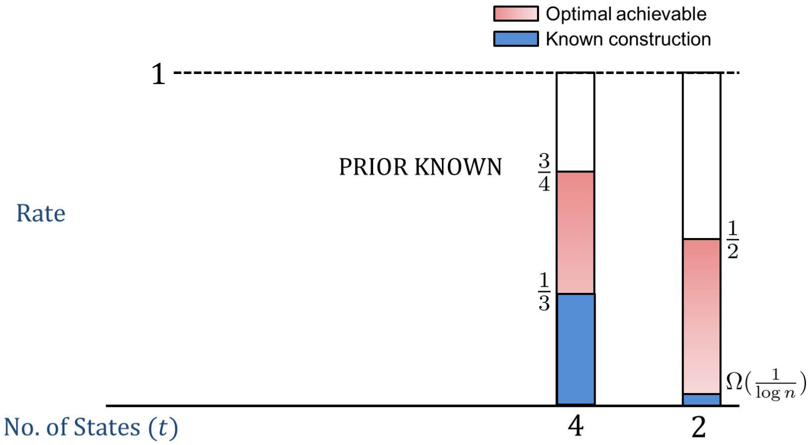
# Security Proof



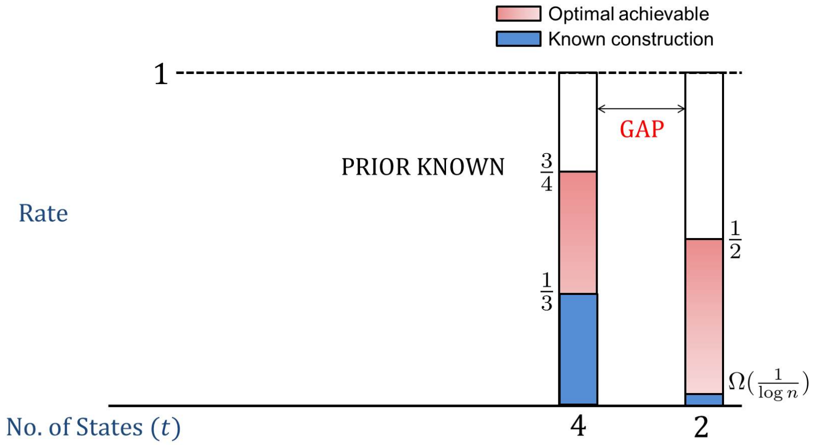
# Security Proof



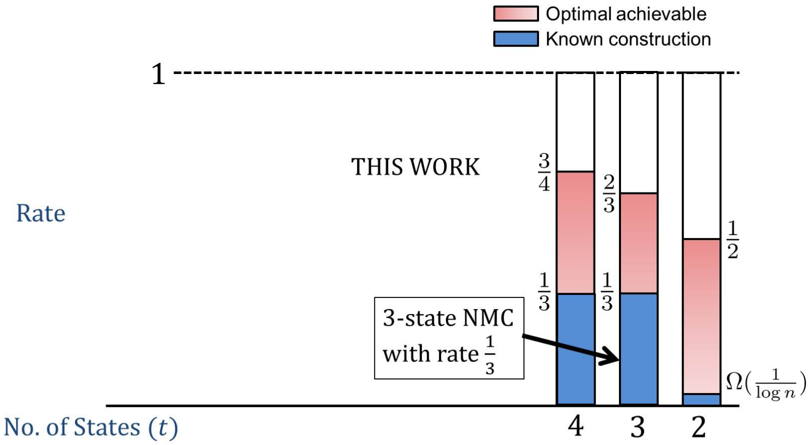
# Application of NMRE: Constant Rate 3-state NMCs



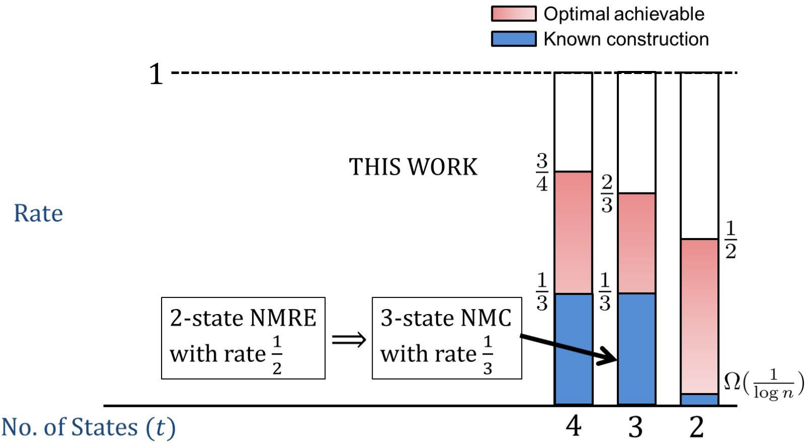
# Application of NMRE: Constant Rate 3-state NMCs



# Application of NMRE: Constant Rate 3-state NMCs



# Application of NMRE: Constant Rate 3-state NMCs



# Conclusion

**Summarizing**:

- Introduced NMREs as an alternative for non-malleable encoding of random messages.

# Conclusion

**Summarizing**:

- Introduced NMREs as an alternative for non-malleable encoding of random messages.
- Built 2-state $1/2$-rate NMRE.

# Conclusion

**Summarizing**:

- Introduced NMREs as an alternative for non-malleable encoding of random messages.
- Built 2-state $1/2$-rate NMRE.
- Built 3-state $1/3$-rate NMC.

# Conclusion

**Summarizing**:

- Introduced NMREs as an alternative for non-malleable encoding of random messages.
- Built 2-state $1/2$-rate NMRE.
- Built 3-state $1/3$-rate NMC.

**Open problems**:

# Conclusion

**Summarizing**:

- Introduced NMREs as an alternative for non-malleable encoding of random messages.
- Built 2-state $1/2$-rate NMRE.
- Built 3-state $1/3$-rate NMC.

**Open problems**:

- Is $1/2$ the optimal achievable rate for 2-state NMRE?

# Conclusion

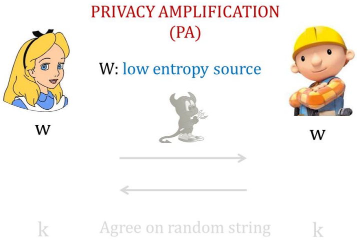
**Summarizing**:

- Introduced NMREs as an alternative for non-malleable encoding of random messages.
- Built 2-state $1/2$-rate NMRE.
- Built 3-state $1/3$-rate NMC.

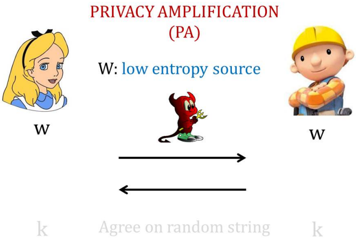
**Open problems**:

- Is $1/2$ the optimal achievable rate for 2-state NMRE?
- Other applications of NMREs

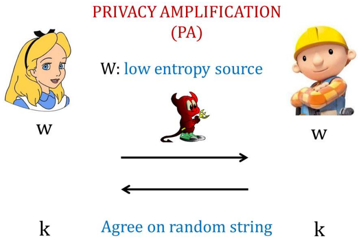
# Subsequent Work



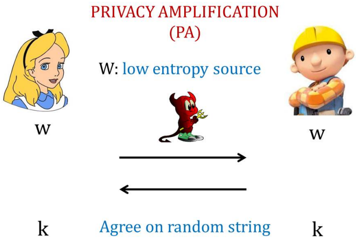
# Subsequent Work



# Subsequent Work

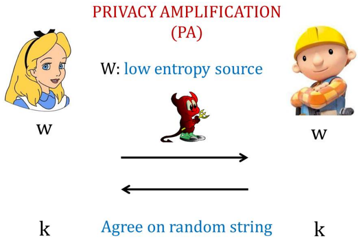


# Subsequent Work




Holy Grail for PA: Build 2-round protocol with entropy loss $\Theta(\lambda)$ and requiring a min-entropy of $\mathcal{O}(\lambda + \log n)$

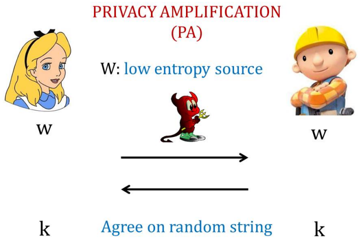
# Subsequent Work




Holy Grail for PA: Build 2-round protocol with entropy loss $\Theta(\lambda)$ and requiring a min-entropy of $\mathcal{O}(\lambda + \log n)$

Our Result: An "augmented" 2-state constant rate NMRE with optimal error $\implies$ 8-round PA protocol with optimal entropy loss and min-entropy requirement.

# Subsequent Work




Holy Grail for PA: Build 2-round protocol with entropy loss $\Theta(\lambda)$ and requiring a min-entropy of $\mathcal{O}(\lambda + \log n)$

Our Result: An “augmented" 2-state constant rate NMRE with optimal error $\implies$ 8-round PA protocol with optimal entropy loss and min-entropy requirement.

(Joint work with: Eshan Chattopadhyay, Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu)
(https://eprint.iacr.org/2018/293)

THANK YOU!!