Dynamic Random Probing Expansion with Quasi Linear Asymptotic Complexity

Sonia Belaïd \(^1\), Matthieu Rivain \(^1\),
Abdul Rahman Taleb \(^{1,2}\) and Damien Vergnaud \(^{2,3}\)

\(^1\) CryptoExperts, France
\(^2\) Sorbonne Université, CNRS, LIP6, F-75005 Paris, France
\(^3\) Institut Universitaire de France, France

December 7, 2021
Side-Channel Attacks & Masking

Security against **side-channel attacks**
Side-Channel Attacks & Masking

Security against side-channel attacks

Masking countermeasure (sensitive variable $x$ over field $\mathbb{K}$)
Side-Channel Attacks & Masking

Security against **side-channel attacks**

Masking countermeasure (sensitive variable $x$ over field $\mathbb{K}$)

\[
x \longrightarrow (x_1, \ldots, x_n) \in \mathbb{K}^n
\]
Side-Channel Attacks & Masking

Security against side-channel attacks

Masking countermeasure (sensitive variable $x$ over field $\mathbb{K}$)

$$x \rightarrow (x_1, \ldots, x_n) \in \mathbb{K}^n$$

shares of $x$

$$x_1 + \ldots + x_n = x$$
Security against **side-channel attacks**

**Masking countermeasure (sensitive variable $x$ over field $\mathbb{K}$)**

$$x \longrightarrow (x_1, \ldots, x_n) \in \mathbb{K}^n$$

shares of $x$

$x_1 + \ldots + x_n = x$

$(+, \times, ||)$ operations over $\mathbb{K}$
Security against side-channel attacks

Masking countermeasure (sensitive variable $x$ over field $\mathbb{K}$)

$$
\begin{align*}
x & \rightarrow \left( x_1, \ldots, x_n \right) \in \mathbb{K}^n \\
\text{shares of } x \\
x_1 + \ldots + x_n & = x
\end{align*}
$$

$(+, \times, ||)$ operations over $\mathbb{K}$ $\rightarrow (G_{\text{add}}, G_{\text{mult}}, G_{\text{copy}}, G_{\text{refresh}})$ $n$-share circuits over $\mathbb{K}$
Side-Channel Attacks & Masking

Security against side-channel attacks

Masking countermeasure (sensitive variable $x$ over field $\mathbb{K}$)

$$x \longrightarrow (x_1, \ldots, x_n) \in \mathbb{K}^n$$

shares of $x$

$$x_1 + \ldots + x_n = x$$

$(+, \times, ||)$ operations over $\mathbb{K} \longrightarrow (G_{\text{add}}, G_{\text{mult}}, G_{\text{copy}}, G_{\text{refresh}})$ $n$-share circuits over $\mathbb{K}$

Example $G_{\text{add}}(a, b) = c$ with $n = 2$
Side-Channel Attacks & Masking

Security against **side-channel attacks**

Masking countermeasure (sensitive variable $x$ over field $\mathbb{K}$)

$$
    x \longrightarrow (x_1, \ldots, x_n) \in \mathbb{K}^n
$$

shares of $x$

$$
    x_1 + \ldots + x_n = x
$$

$(+, \times, ||)$ operations over $\mathbb{K}$

$$
    \longrightarrow (G_{\text{add}}, G_{\text{mult}}, G_{\text{copy}}, G_{\text{refresh}}) \ n\text{-share circuits over } \mathbb{K}
$$

Example $G_{\text{add}}(a, b) = c$ with $n = 2$
Leakage Models

- Realistic
- Convenient
- Random probing model
  each variable leaks with proba. $p$
- Noisy Leakage model
  noisy leakage of all the variables
Leakage Models

Convenient

$t$-probing model
$t$ leaking variables

Realistic
Leakage Models

Convenient

Realistic

Random probing model

\[ \text{each variable leaks with proba. } p \]

$t$-probing model

\[ t \text{ leaking variables} \]
Leakage Models

- **t-probing model**
  - $t$ leaking variables

- Random probing model
  - each variable leaks with proba. $p$

- Noisy Leakage model
  - noisy leakage of all the variables
Leakage Models

Convenient

$t$-probing model
$t$ leaking variables

Random probing model
each variable leaks with proba. $p$

Noisy Leakage model
noisy leakage of all the variables

Realistic
Leakage Models

- **$t$-probing model**
  - $t$ leaking variables

- **Random probing model**
  - each variable leaks with proba. $p$

- **Noisy Leakage model**
  - noisy leakage of all the variables

Convenient → Realistic
Prior Works

Prior Works


- Security of masking in the Random Probing (RP) Model
Prior Works


- Security of masking in the Random Probing (RP) Model
- RP-secure gadgets composition (RP composition)
Prior Works


- Security of masking in the Random Probing (RP) Model
- RP-secure gadgets composition (RP composition)
- RP-secure security level amplification (RP expansion)
Prior Works


- Security of masking in the Random Probing (RP) Model
- RP-secure gadgets composition (RP composition)
- RP-secure security level amplification (RP expansion)

Prior Works


- Security of masking in the **Random Probing (RP) Model**
- RP-secure gadgets composition (RP composition)
- RP-secure security level amplification (RP expansion)


- In-depth analysis of RP expansion
Prior Works


- Security of masking in the Random Probing (RP) Model
- RP-secure gadgets composition (RP composition)
- RP-secure security level amplification (RP expansion)


- In-depth analysis of RP expansion
- Generic constructions for RP expansion with improved complexities
Prior Works


- Security of masking in the **Random Probing (RP) Model**
- RP-secure gadgets composition (RP composition)
- RP-secure security level amplification (RP expansion)


- In-depth analysis of RP expansion
- Generic constructions for RP expansion with improved complexities
- Concrete instantiations for RP expansion tolerating a leakage rate of \( p \approx 2^{-7.5} \)
Contributions

- Introduction of **Dynamic** Random Probing Expansion (RPE)

Generalization of RPE to support any basic operations (e.g. multiplication by a constant $G_{cmult}$)

Construction of $n$-share RPE-secure $G_{add}$, $G_{copy}$, $G_{cmult}$ with $O(n \log n)$ complexity (using $G_{refresh}$ by Battistello et al. - CHES 2016)

Construction of RPE-secure $G_{mult}$ with $O(n \log n)$ randomness and $O(n)$ multiplications between variables, from:

- Extension of sub-multiplication gadget $G_{submult}$: $K_n \times K_n \rightarrow K_{2n+1}$ by Belaïd et al. - Crypto 2017
- New compression gadget $G_{compress}$: $K_{2n+1} \rightarrow K_n$
Contributions

- Introduction of **Dynamic** Random Probing Expansion (RPE)

- Generalization of RPE to support any basic operations (e.g. multiplication by a constant $G_{\text{cmult}}$)
Contributions

- Introduction of **Dynamic** Random Probing Expansion (RPE)

- Generalization of RPE to support any basic operations (e.g. multiplication by a constant $G_{cmult}$)

- Construction of $n$-share RPE-secure $G_{add}$, $G_{copy}$, $G_{cmult}$ with $O(n \log n)$ complexity (using $G_{refresh}$ by Battistello et al. - CHES 2016)
Contributions

- Introduction of **Dynamic** Random Probing Expansion (RPE)

- Generalization of RPE to support any basic operations (e.g. multiplication by a constant $G_{cmult}$)

- Construction of $n$-share RPE-secure $G_{add}$, $G_{copy}$, $G_{cmult}$ with $O(n \log n)$ complexity (using $G_{refresh}$ by Battistello et al. - CHES 2016)

- Construction of RPE-secure $G_{mult}$ with $O(n \log n)$ **randomness** and $O(n)$ **multiplications** between variables, from:
Contributions

- Introduction of **Dynamic** Random Probing Expansion (RPE)

- Generalization of RPE to support any basic operations (e.g. multiplication by a constant $G_{cmult}$)

- Construction of $n$-share RPE-secure $G_{add}$, $G_{copy}$, $G_{cmult}$ with $O(n \log n)$ complexity (using $G_{refresh}$ by Battistello et al. - CHES 2016)

- Construction of RPE-secure $G_{mult}$ with $O(n \log n)$ randomness and $O(n)$ multiplications between variables, from:
  - extension of sub-multiplication gadget $G_{submult} : \mathbb{K}^n \times \mathbb{K}^n \rightarrow \mathbb{K}^{2n+1}$ by Belaïd et al. - Crypto 2017
Contributions

- Introduction of **Dynamic** Random Probing Expansion (RPE)
- Generalization of RPE to support any basic operations (e.g. multiplication by a constant $G_{cmult}$)
- Construction of $n$-share RPE-secure $G_{add}$, $G_{copy}$, $G_{cmult}$ with $O(n \log n)$ complexity (using $G_{refresh}$ by Battistello et al. - CHES 2016)
- Construction of RPE-secure $G_{mult}$ with $O(n \log n)$ **randomness** and $O(n)$ **multiplications** between variables, from:
  - extension of sub-multiplication gadget $G_{submult} : \mathbb{K}^n \times \mathbb{K}^n \rightarrow \mathbb{K}^{2n+1}$ by Belaïd et al. - Crypto 2017
  - new compression gadget $G_{compress} : \mathbb{K}^{2n+1} \rightarrow \mathbb{K}^n$
$(p, \varepsilon)$-RP Security

\[ \begin{array}{c}
\oplus \quad \text{Add} \\
\otimes \quad \text{Mult.}
\end{array} \]

\[ \begin{array}{c}
| \quad \text{Copy} \\
r \quad \text{Random}
\end{array} \]

\[ \begin{array}{c}
\otimes_c \quad \text{Mult. by constant}
\end{array} \]
$(p, \varepsilon)$-RP Security

$W$ set of wires

Add  Mult.  Copy  Random

Mult. by constant
\((p, \varepsilon)\)-RP Security

\(\mathcal{W}\) set of wires

Independent from secret inputs?

\begin{itemize}
  \item Add \hspace{1cm} \times \text{ Mult.}
  \item Copy \hspace{1cm} \circ \text{ Random}
  \item \times_c \text{ Mult. by constant}
\end{itemize}
(\(p, \varepsilon\))-RP Security

\(W\) set of wires

Independent from secret inputs?

- yes
- no
(p, ε)-RP Security

**W** set of wires

Independent from secret inputs?

- yes
- no

Simulation Success

Add | Mult.
---|---
Copy | Random
×c | Mult. by constant
(\(p, \varepsilon\))-RP Security

\(W\) set of wires

Independent from secret inputs?

- yes
- no

### Simulation Success

Add \(\oplus\) \hspace{1cm} \times\) Mult.

Copy \(\llbracket\llbracket\) \hspace{1cm} Random \(r\)

Mult. by constant \(\times_c\)
$(p, \varepsilon)$-RP Security

$W$ set of wires

Independent from secret inputs?

- yes
- no

Simulation Success

Simulation Failure

Failure Probability $\varepsilon$

Add $\oplus$
Mult. $\times$
Copy $\|$
Random $r$
Mult. by constant $\times_c$

S. Belaid, M. Rivain, A. Taleb, D. Vergnaud
Using \( n \)-share gadgets \( G_1, \ldots, G_\beta \)
Using $n$-share gadgets $G_1, \ldots, G_\beta$
Using $n$-share gadgets $G_1, \ldots, G_\beta$
Using $n$-share gadgets $G_1, \ldots, G_\beta$
Using $n$-share gadgets $G_1, \ldots, G_\beta$

Leakage probability $p$
Using $n$-share gadgets $G_1, \ldots, G_\beta$
Using $n$-share gadgets $G_1, \ldots, G_\beta$

Leakage probability $p$

Simulation Failure $\varepsilon$
Using $n$-share gadgets $G_1, \ldots, G_\beta$

Leakage probability $p$

Simulation Failure $\epsilon$

First Expansion
$\rightarrow$

Second Expansion
$\rightarrow$

$\epsilon_{k}$ first exp

$\epsilon < p$ (tolerated leakage rate)
Using $n$-share gadgets $G_1, \ldots, G_\beta$

Leakage probability $p$

Simulation Failure $\varepsilon$

**First Expansion**

$n=2$

$G_i \rightarrow G_j \rightarrow G_k$

**Second Expansion**

$n^2=4$

$G_i^{(2)} \rightarrow G_j^{(2)} \rightarrow G_k^{(2)}$
Using $n$-share gadgets $G_1, \ldots, G_\beta$
Using $n$-share gadgets $G_1, \ldots, G_\beta$
Using $n$-share gadgets $G_1, \ldots, G_\beta$

Leakage probability $p$

Simulation Failure $\varepsilon$

$n^k \rightarrow \varepsilon^k$
Using $n$-share gadgets $G_1, \ldots, G_\beta$

**Condition**: $\varepsilon < p$ (tolerated leakage rate)
**RP Expansion**

**Definition**

$(t, p, \varepsilon)$-**RP expandability** (RPE) of gadget $G$ guarantees:

Independent failure probability on each input sharing $G_1, \ldots, G_\beta$ are $(t, p, \varepsilon)$-RPE $\Rightarrow$ compiled circuit $C$ is $(p, 2^{|C|} \cdot \varepsilon^k)$-RP Secure.
RP Expansion

Definition

$(t, p, \varepsilon)$-RP expandability (RPE) of gadget $G$ guarantees:

- $(p, \varepsilon)$-RP security of $G$ (RPE is stronger than RP)
RP Expansion

Definition

(t, p, ε)-RP expandability (RPE) of gadget G guarantees:

- (p, ε)-RP security of G (RPE is stronger than RP)

- composition of G with other RP secure gadgets: ability to simulate any set W of internal wires and t output shares using t input shares
(t, p, ε)-RP expandability (RPE) of gadget $G$ guarantees:

- $(p, ε)$-RP security of $G$ (RPE is stronger than RP)

- composition of $G$ with other RP secure gadgets: ability to simulate any set $W$ of internal wires and $t$ output shares using $t$ input shares
RP Expansion

Definition

\((t, p, \varepsilon)\)-RP expandability (RPE) of gadget \(G\) guarantees:

- \((p, \varepsilon)\)-RP security of \(G\) (RPE is stronger than RP)

- composition of \(G\) with other RP secure gadgets: ability to simulate any set \(W\) of internal wires and \(t\) output shares using \(t\) input shares

\[ G_1 \quad \quad G_3 \quad \quad G_4 \]

\[ G_2 \]

\(t = 2\)
(t, p, ε)-RP expandability (RPE) of gadget G guarantees:

- (p, ε)-RP security of G (RPE is stronger than RP)
- composition of G with other RP secure gadgets: ability to simulate any set W of internal wires and t output shares using t input shares

\[ t = 2 \]
\((t, p, \varepsilon)\)-RP expandability (RPE) of gadget G guarantees:

- \((p, \varepsilon)\)-RP security of G (RPE is stronger than RP)
- \textbf{composition} of G with other RP secure gadgets: ability to simulate any set \(W\) of internal wires and \(t\) output shares using \(t\) input shares
**RP Expansion**

**Definition**

(t, p, ε)-RP **expandability** (RPE) of gadget G guarantees:

- (p, ε)-RP security of G (RPE is stronger than RP)
- **composition** of G with other RP secure gadgets: ability to simulate any set \( W \) of internal wires and \( t \) output shares using \( t \) input shares

![Diagram showing composition of three gadgets](image)

\( t = 2 \)
**(t, p, ε)-RP expandability (RPE)** of gadget $G$ guarantees:

- $(p, ε)$-RP security of $G$ (RPE is stronger than RP)

- **composition** of $G$ with other RP secure gadgets: ability to simulate any set $W$ of internal wires and $t$ output shares using $t$ input shares

\[
\begin{aligned}
&G_1 \quad W_1 \\
&G_2 \quad W_2 \\
&G_3 \\
&G_4 \quad W_4
\end{aligned}
\]
RP Expansion

Definition

\((t, p, \varepsilon)\)-RP expandability (RPE) of gadget \(G\) guarantees:

- \((p, \varepsilon)\)-RP security of \(G\) (RPE is stronger than RP)
- composition of \(G\) with other RP secure gadgets: ability to simulate any set \(W\) of internal wires and \(t\) output shares using \(t\) input shares
- Independent failure probability on each input sharing
(t, p, ε)-RP expandability (RPE) of gadget G guarantees:

- (p, ε)-RP security of G (RPE is stronger than RP)
- **Composition** of G with other RP secure gadgets: ability to simulate any set W of internal wires and t output shares using t input shares

- Independent failure probability on each input sharing

$G_1, \ldots, G_\beta$ are (t, p, ε)-RPE $\implies$ compiled circuit C is $(p, 2.|C|.\varepsilon^k)$-RP Secure
Complexity of expanded circuit $C$ of security parameter $\kappa$: 

\[ O(|C| \cdot \kappa^e) \]

\[ e = \log(N_{\text{max}}) \log(d) \]

\[ N_{\text{max}} \approx \max(\# \times \text{in} \ G_{\text{mult}}, \#(+| |) \text{in} \ G_{\text{add}}, G_{\text{copy}}, \# \times c \text{in} \ G_{\text{cmult}}) \]

$d$: amplification order (i.e. smallest failure set of internal wires)

\[ a_1b_1 + a_2b_2 + c_1c_2 \]

Example $t = 1, n = 2$

Output $c_1 = a_1 + b_1$, set $W = \{b_2\}$

Simulation needs $a_1(\leq t)$ and $b_1, b_2(> t)$

Failure on $b = \Rightarrow d = |W| = 1$
Complexity of expanded circuit $C$ of security parameter $\kappa$:

$$\mathcal{O}(|C| \cdot \kappa^e), \quad e = \frac{\log(N_{\text{max}})}{\log(d)}$$
Complexity of expanded circuit $C$ of security parameter $\kappa$:

$$\mathcal{O}(|C| \cdot \kappa^e), \quad e = \frac{\log(N_{\text{max}})}{\log(d)}$$

$$N_{\text{max}} \approx \max(\# \times \text{ in } G_{\text{mult}}, \#(+, ||) \text{ in } G_{\text{add}}, G_{\text{copy}}, \# \times c \text{ in } G_{\text{cmult}})$$
Complexity of expanded circuit $C$ of security parameter $\kappa$:

$$O(|C| \cdot \kappa^e), \quad e = \frac{\log(N_{\text{max}})}{\log(d)}$$

$$N_{\text{max}} \approx \max(\# \times\text{ in } G_{\text{mult}}, \#(+, ||) \text{ in } G_{\text{add}}, G_{\text{copy}}, \# \times_c \text{ in } G_{\text{cmult}})$$

$d$: amplification order (i.e. smallest failure set of internal wires)
Complexity of expanded circuit $C$ of security parameter $\kappa$:

$$O(|C| \cdot \kappa^e), \quad e = \frac{\log(N_{\text{max}})}{\log(d)}$$

$$N_{\text{max}} \approx \max(\# \times \text{ in } G_{\text{mult}}, \#(+,||) \text{ in } G_{\text{add}}, G_{\text{copy}}, \# \times c \text{ in } G_{\text{cmult}})$$

$d$: amplification order (i.e. smallest failure set of internal wires)

Example $t = 1, n = 2$
Complexity of expanded circuit $C$ of security parameter $\kappa$:

$$O(|C| \cdot \kappa^e), \quad e = \frac{\log(N_{\text{max}})}{\log(d)}$$

$$N_{\text{max}} \approx \max(\# \times \text{ in } G_{\text{mult}}, \#(+, ||) \text{ in } G_{\text{add}}, G_{\text{copy}}, \# \times c \text{ in } G_{\text{cmult}})$$

$d$: amplification order ($i.e.$ smallest failure set of internal wires)

Example $t = 1, n = 2$

Output $c_1 = a_1 + b_1$,
Complexity of expanded circuit $C$ of security parameter $\kappa$:

$$O(|C| \cdot \kappa^e), \quad e = \frac{\log(N_{\text{max}})}{\log(d)}$$

$$N_{\text{max}} \approx \max(\# \times \text{ in } G_{\text{mult}}, \#(+, ||) \text{ in } G_{\text{add}}, G_{\text{copy}}, \# \times c \text{ in } G_{\text{cmult}})$$

$d$: amplification order (i.e. smallest failure set of internal wires)

Example $t = 1, n = 2$

Output $c_1 = a_1 + b_1$, set $W = \{b_2\}$
Complexity of expanded circuit $C$ of security parameter $\kappa$:

$$O(|C| \cdot \kappa^e), \quad e = \frac{\log(N_{\text{max}})}{\log(d)}$$

$N_{\text{max}} \approx \max(\# \times \text{ in } G_{\text{mult}}, \#(+, ||) \text{ in } G_{\text{add}}, G_{\text{copy}}, \# \times c \text{ in } G_{\text{cmult}})$

$d$: amplification order (i.e. smallest failure set of internal wires)

Example $t = 1, n = 2$

Output $c_1 = a_1 + b_1$, set $W = \{b_2\}$

Simulation needs $a_1 (\leq t)$ and $b_1, b_2 (\geq t)$
Complexity of expanded circuit $C$ of security parameter $\kappa$:

$$\mathcal{O}(|C| \cdot \kappa^e), \quad e = \frac{\log(N_{\text{max}})}{\log(d)}$$

$N_{\text{max}} \approx \max(\# \times \text{ in } G_{\text{mult}}, \#(+, ||) \text{ in } G_{\text{add}}, G_{\text{copy}}, \# \times c \text{ in } G_{\text{cmult}})$

$d$: amplification order (i.e. smallest failure set of internal wires)

Example $t = 1$, $n = 2$

Output $c_1 = a_1 + b_1$, set $W = \{b_2\}$

Simulation needs $a_1 \leq t$ and $b_1, b_2 > t$

Failure on $b$
Complexity of expanded circuit $C$ of security parameter $\kappa$:

$$\mathcal{O}(|C|.\kappa^e), \quad e = \frac{\log(N_{\text{max}})}{\log(d)}$$

$$N_{\text{max}} \approx \max(\# \times \text{ in } G_{\text{mult}}, \# (+, ||) \text{ in } G_{\text{add}}, G_{\text{copy}}, \# \times c \text{ in } G_{\text{cmult}})$$

$d$: amplification order (i.e. smallest failure set of internal wires)

Example $t = 1, n = 2$

Output $c_1 = a_1 + b_1$, set $W = \{b_2\}$

Simulation needs $a_1 (\leq t)$ and $b_1, b_2 (> t)$

Failure on $b \implies d = |W| = 1$
Complexity of expanded circuit $C$ of security parameter $\kappa$:

$$O(|C|\kappa^e), \quad e = \frac{\log(N_{\text{max}})}{\log(d)}$$

$N_{\text{max}} \approx \max(\# \times \text{ in } G_{\text{mult}}, \#(+,||) \text{ in } G_{\text{add}}, G_{\text{copy}}, \# \times c \text{ in } G_{\text{cmult}})$

$d$: amplification order (i.e. smallest failure set of internal wires)

$$\varepsilon = f(p) = c_d \cdot p^d + O(p^{d+1})$$
Complexity of expanded circuit $C$ of security parameter $\kappa$:

$$O(|C|\cdot\kappa^e), \quad e = \frac{\log(N_{\text{max}})}{\log(d)}$$

$N_{\text{max}} \approx \max(\# \times \text{ in } G_{\text{mult}}, \#(+,||) \text{ in } G_{\text{add}}, G_{\text{copy}}, \# \times c \text{ in } G_{\text{cmult}})$

d: amplification order (i.e. smallest failure set of internal wires)

$$\varepsilon = f(p) = c_d \cdot p^d + O(p^{d+1})$$

- during expansion: $\varepsilon^k = f^{(k)}(p) = f(f(\ldots f(f(p))\ldots))$
Parameters

Complexity of expanded circuit $C$ of security parameter $\kappa$:

\[ \mathcal{O}(|C|\kappa^e), \quad e = \frac{\log(N_{\text{max}})}{\log(d)} \]

$N_{\text{max}} \approx \max(\# \times \text{ in } G_{\text{mult}}, \#(+,||) \text{ in } G_{\text{add}}, G_{\text{copy}}, \# \times c \text{ in } G_{\text{cmult}})$

$d$: amplification order \textit{(i.e.} smallest failure set of internal wires\textit{)}

\[ \varepsilon = f(p) = c_d \cdot p^d + \mathcal{O}(p^{d+1}) \]

- during expansion: $\varepsilon^k = f^{(k)}(p) = f(f(\ldots f(f(p))\ldots))$

- higher $d \implies$ faster decrease in failure probability ($d_{\text{max}} = \frac{n + 1}{2}$)
Dynamic RP Expansion

Idea

Using RPE compilers $CC_1, \ldots, CC_\ell$ with numbers of shares $n_1, \ldots, n_\ell$

\[
C \xrightarrow{\text{$k_1$ times}} CC_1
\]

Leakage rate $p$

Conditions:

$\varepsilon_1 < p$, $\varepsilon_2 < \varepsilon_{k_1}$, $\ldots$, $\varepsilon_\ell < \varepsilon_{k_\ell - 1}$

Why?
Dynamic RP Expansion

Idea

Using RPE compilers $CC_1, \ldots, CC_\ell$ with numbers of shares $n_1, \ldots, n_\ell$

\[ C \xrightarrow{k_1 \text{ times}} CC_1 \]

Leakage rate $p$

$\varepsilon_{1}^{k_1} = f_1^{(k_1)}(p)$

Conditions:

$\varepsilon_1 < p$, $\varepsilon_2 < \varepsilon_1^{k_1}$, \ldots, $\varepsilon_\ell < \varepsilon_{\ell-1}^{k_\ell}$
Dynamic RP Expansion

Idea

Using RPE compilers $CC_1, \ldots, CC_\ell$ with numbers of shares $n_1, \ldots, n_\ell$

\[ C \xrightarrow[k_1]{CC_1} \hat{C}_1 \xrightarrow[k_2]{CC_2} \ldots \xrightarrow[k_\ell]{CC_\ell} \hat{C}_\ell \]

Leakage rate $p$

\[ \varepsilon_{k_1}^{n_1} = f_1^{(k_1)}(p) \]

Conditions:

\[ \varepsilon_1 < p, \varepsilon_2 < \varepsilon_{k_1}, \ldots, \varepsilon_\ell < \varepsilon_{k_\ell-1} \]

Why?

S. Belaid, M. Rivain, A. Taleb, D. Vergnaud
Dynamic RP Expansion

Idea

Using RPE compilers $CC_1, \ldots, CC_\ell$ with numbers of shares $n_1, \ldots, n_\ell$

\[
\begin{align*}
C & \xrightarrow{CC_1 \ k_1 \ \text{times}} \hat{C}_1 & \xrightarrow{CC_2 \ k_2 \ \text{times}} \hat{C}_2 \\
\text{Leakage rate } p & \\
& n_1^{k_1} \ \text{shares} & n_2^{k_2} \cdot n_1^{k_1} \ \text{shares} \\
& \varepsilon_1^{k_1} = f_1^{(k_1)}(p) & \varepsilon_2^{k_2} = f_2^{(k_2)}(f_1^{(k_1)}(p))
\end{align*}
\]
Dynamic RP Expansion

Idea

Using RPE compilers $CC_1, \ldots, CC_\ell$ with numbers of shares $n_1, \ldots, n_\ell$

\[ C \xrightarrow{CC_1 \text{ times}} \hat{C}_1 \xrightarrow{CC_2 \text{ times}} \hat{C}_2 \xrightarrow{\text{\ldots}} \ldots \]

Leakage rate $p$

- $n_1^{k_1}$ shares
  \[ \varepsilon_1^{k_1} = f_1^{(k_1)}(p) \]
- $n_2^{k_2} \cdot n_1^{k_1}$ shares
  \[ \varepsilon_2^{k_2} = f_2^{(k_2)}(f_1^{(k_1)}(p)) \]

Conditions:

- $\varepsilon_1 < p$
- $\varepsilon_2 < \varepsilon_1$
- $\ldots$
- $\varepsilon_\ell < \varepsilon_{\ell-1}$
Dynamic RP Expansion

Idea

Using RPE compilers $CC_1, \ldots, CC_\ell$ with numbers of shares $n_1, \ldots, n_\ell$

\[ C \xrightarrow{CC_1 \text{ times}} \hat{C}_1 \xrightarrow{CC_2 \text{ times}} \hat{C}_2 \xrightarrow{\cdots \text{ times}} \cdots \xrightarrow{CC_\ell \text{ times}} \]

Leakage rate $p$

- $n_1^{k_1}$ shares
  \[ \varepsilon_1^{k_1} = f_1^{(k_1)}(p) \]
- $n_2^{k_2} \cdot n_1^{k_1}$ shares
  \[ \varepsilon_2^{k_2} = f_2^{(k_2)}(f_1^{(k_1)}(p)) \]
Dynamic RP Expansion

Idea

Using RPE compilers $CC_1, \ldots, CC_\ell$ with numbers of shares $n_1, \ldots, n_\ell$

\[
\begin{align*}
C & \xrightarrow{CC_1 \text{ times}} \hat{C}_1 & \xrightarrow{CC_2 \text{ times}} \hat{C}_2 & \xrightarrow{\cdots} \cdots & \xrightarrow{CC_\ell \text{ times}} \hat{C}_\ell \\
& \quad n_1^{k_1} \text{ shares} & \quad n_2^{k_2} \cdot n_1^{k_1} \text{ shares} & \quad n_\ell^{k_\ell} \cdots n_1^{k_1} \text{ shares} \\
& \quad \varepsilon_1^{k_1} = f_1^{(k_1)}(p) & \quad \varepsilon_2^{k_2} = f_2^{(k_2)}(f_1^{(k_1)}(p)) & \quad \varepsilon_\ell^{k_\ell} = f_\ell^{(k_\ell)}(\cdots(f_1^{(k_1)}(p)\cdots)
\end{align*}
\]

Conditions:

$\varepsilon_1 < p$, $\varepsilon_2 < \varepsilon_1^{k_1}$, $\ldots$, $\varepsilon_\ell < \varepsilon_{\ell-1}^{k_{\ell-1}}$

Why?

S. Belaid, M. Rivain, A. Taleb, D. Vergnaud
Dynamic RP Expansion

Idea

Using RPE compilers $CC_1, \ldots, CC_\ell$ with numbers of shares $n_1, \ldots, n_\ell$

\[
\begin{align*}
C \quad \xrightarrow{k_1 \text{ times}} \quad CC_1 \quad \xrightarrow{k_2 \text{ times}} \quad CC_2 \quad \xrightarrow{\ldots} \quad \ldots \quad \xrightarrow{k_\ell \text{ times}} \quad CC_\ell \\
C_1 \quad \xrightarrow{k_1 \text{ times}} \quad CC_1 \quad \xrightarrow{k_2 \text{ times}} \quad CC_2 \quad \xrightarrow{\ldots} \quad \ldots \quad \xrightarrow{k_\ell \text{ times}} \quad CC_\ell \\
\end{align*}
\]

Leakage rate $p$

\[
\begin{align*}
\varepsilon_1^{k_1} &= f_1^{(k_1)}(p) \\
\varepsilon_2^{k_2} &= f_2^{(k_2)}(f_1^{(k_1)}(p)) \\
\varepsilon_\ell^{k_\ell} &= f_\ell^{(k_\ell)}(\ldots(f_1^{(k_1)}(p))\ldots)
\end{align*}
\]

Conditions: $\varepsilon_1 < p$, $\varepsilon_2 < \varepsilon_1^{k_1}$, \ldots, $\varepsilon_\ell < \varepsilon_{\ell-1}^{k_\ell-1}$
Using RPE compilers $CC_1, \ldots, CC_\ell$ with numbers of shares $n_1, \ldots, n_\ell$

\[
\begin{align*}
C \xrightarrow[\times k_1]{CC_1} \hat{C}_1 \xrightarrow[\times k_2]{CC_2} \hat{C}_2 \rightarrow \cdots \rightarrow \cdots \xrightarrow[\times k_\ell]{CC_\ell} \hat{C}_\ell
\end{align*}
\]

Leakage rate $p$

\[
\begin{align*}
\varepsilon^{k_1}_1 &= f^{(k_1)}_1(p) \\
\varepsilon^{k_2}_2 &= f^{(k_2)}_2(f^{(k_1)}_1(p)) \\
\varepsilon^{k_\ell}_{\ell} &= f^{(k_\ell)}_{\ell}(\cdots(f^{(k_1)}_1(p))\cdots)
\end{align*}
\]

**Conditions:** $\varepsilon_1 < p$, $\varepsilon_2 < \varepsilon^{k_1}_1$, $\ldots$, $\varepsilon_\ell < \varepsilon^{k_\ell-1}_{\ell-1}$

**Why?**
Dynamic RP Expansion

Motivation

\(n\)-share RPE compilers:

\[d_{\text{max}} = n + 1\]

\[\text{Complexity and security level of RP AES starting from tolerated leakage of } p = 2^{-7.6} \text{ using 3-share CC3 and 5-share CC5 by Belaïd et al. - EuroCrypt 2021}\]
$n$-share RPE compilers:

- **small** $n$: fewer sets of probes that reveal the secret $\Rightarrow$ tolerate better leakage rate $p$
Motivation

\( n \)-share RPE compilers:

- **small** \( n \): fewer sets of probes that reveal the secret \( \Rightarrow \) tolerate better leakage rate \( p \)
- **big** \( n \): have higher amp. order \( d_{\text{max}} = \frac{n + 1}{2} \) \( \Rightarrow \) have better asymptotic complexity
Dynamic RP Expansion

Motivation

- **small** $n$: fewer sets of probes that reveal the secret $\implies$ tolerate better leakage rate $p$
- **big** $n$: have higher amp. order $d_{\text{max}} = \frac{n+1}{2} \implies$ have better asymptotic complexity

Complexity and security level of RP AES starting from tolerated leakage of $p = 2^{-7.6}$ using 3-share $CC_3$ and 5-share $CC_5$ by Belaïd et al. - EuroCrypt 2021
n-share RPE compilers:

- **small** \( n \): fewer sets of probes that reveal the secret \( \Rightarrow \) tolerate better leakage rate \( p \)
- **big** \( n \): have higher amp. order \( d_{\text{max}} = \frac{n + 1}{2} \) \( \Rightarrow \) have better asymptotic complexity

Complexity and security level of RP AES starting from tolerated leakage of \( p = 2^{-7.6} \) using 3-share \( CC_3 \) and 5-share \( CC_5 \) by Belaïd et al. - EuroCrypt 2021
Dynamic RP Expansion

Motivation

\( n \)-share RPE compilers:

- **small** \( n \): fewer sets of probes that reveal the secret \( \Rightarrow \) tolerate better leakage rate \( p \)
- **big** \( n \): have higher amp. order \( d_{\text{max}} = \frac{n + 1}{2} \) \( \Rightarrow \) have better asymptotic complexity

Complexity and security level of RP AES starting from tolerated leakage of \( p = 2^{-7.6} \) using 3-share \( CC_3 \) and 5-share \( CC_5 \) by Belaïd et al. - EuroCrypt 2021
2 possible directions:
2 possible directions:

- look for gadgets with **small** number of shares tolerating the best leakage rate (eventually with high complexity)
2 possible directions:

- look for gadgets with **small** number of shares tolerating the best leakage rate (eventually with high complexity)

- look for gadgets which achieve maximal amp. order for **any** number shares with low asymptotic complexity
2 possible directions:

- look for gadgets with **small** number of shares tolerating the best leakage rate (eventually with high complexity)

- look for gadgets which achieve maximal amp. order for **any** number shares with low asymptotic complexity

In this work:
2 possible directions:

- look for gadgets with small number of shares tolerating the best leakage rate (eventually with high complexity)
- look for gadgets which achieve maximal amp. order for any number shares with low asymptotic complexity

In this work:

- construction of \( n \)-share linear \( G_{\text{add}}, G_{\text{copy}}, G_{\text{cmult}} \) with \( O(n \log n) \) asymptotic complexity and maximal amp. order
2 possible directions:

- look for gadgets with **small** number of shares tolerating the best leakage rate (eventually with high complexity)

- look for gadgets which achieve maximal amp. order for **any** number shares with low asymptotic complexity

In this work:

- construction of $n$-share linear $G_{\text{add}}$, $G_{\text{copy}}$, $G_{\text{cmult}}$ with $O(n \log n)$ asymptotic complexity and maximal amp. order

- construction of $n$-share $G_{\text{mult}}$ with $O(n \log n)$ **randomness** and $O(n)$ **multiplications** between variables
Linear Gadgets
Building Block

$O(n \log n)$ refresh gadget $G_{\text{refresh}}$ by Battistello et al. - CHES 2016:

\[
\begin{pmatrix}
a_1 \\
\vdots \\
a_n \\
\end{pmatrix}
\begin{pmatrix}
a_1 \\
\vdots \\
a_n \\
\end{pmatrix}
\]
$O(n \log n)$ refresh gadget $G_{\text{refresh}}$ by Battistello et al. - CHES 2016:

\[ \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix} \xrightarrow{L_1} \begin{pmatrix} b_1 \\ \vdots \\ b_{n/2} \\ a_{n/2+1} \\ \vdots \\ a_n \end{pmatrix} \]

\[
\begin{align*}
    b_i & \leftarrow a_i + r_i \\
    b_{n/2+i} & \leftarrow a_{n/2+i} + r_i
\end{align*}
\]
$O(n \log n)$ refresh gadget $G_{\text{refresh}}$ by Battistello et al. - CHES 2016:

\[
\begin{pmatrix}
  a_1 \\
  \vdots \\
  a_n
\end{pmatrix}
\xrightarrow{L_1}
\begin{pmatrix}
  b_1 \\
  \vdots \\
  b_{n/2}
\end{pmatrix}
\]

\[
\begin{pmatrix}
  b_{n/2+1} \\
  \vdots \\
  b_n
\end{pmatrix}
\]

\[
\begin{pmatrix}
  b_1 \\
  \vdots \\
  b_{n/2}
\end{pmatrix} + \begin{pmatrix}
  r_1 \\
  \vdots \\
  r_{n/2}
\end{pmatrix}
\]

\[
\begin{pmatrix}
  b_{n/2+1} \\
  \vdots \\
  b_n
\end{pmatrix} + \begin{pmatrix}
  r_{n/2+1} \\
  \vdots \\
  r_n
\end{pmatrix}
\]

\[
\begin{pmatrix}
  b_i \\ b_{n/2+i}
\end{pmatrix} = \begin{pmatrix}
  a_i \\ a_{n/2+i}
\end{pmatrix} + \begin{pmatrix}
  r_i \\ r_{n/2+i}
\end{pmatrix}
\]

\[
\begin{pmatrix}
  b_{n/2+1} \\
  \vdots \\
  b_n
\end{pmatrix} = \begin{pmatrix}
  a_{n/2+1} \\
  a_{n+1}
\end{pmatrix} + \begin{pmatrix}
  r_{n/2+1} \\
  r_{n+1}
\end{pmatrix}
\]
$O(n \log n)$ refresh gadget $G_{\text{refresh}}$ by Battistello et al. - CHES 2016:

\[
\begin{pmatrix}
a_1 \\ \vdots \\ a_n
\end{pmatrix}
\rightarrow
\begin{pmatrix}
b_1 \\ \vdots \\ b_{n/2}
\end{pmatrix}
\rightarrow
R_1
\rightarrow
R_2
\rightarrow
\begin{pmatrix}
b_{n/2+1} \\ \vdots \\ b_n
\end{pmatrix}
\]

Recursive call

\[
b_i \leftarrow a_i + r_i
\]

$n/2$ randoms

\[
b_{n/2+i} \leftarrow a_{n/2+i} + r_i
\]
$\mathcal{O}(n \log n)$ refresh gadget $G_{\text{refresh}}$ by Battistello et al. - CHES 2016:

\[
\begin{pmatrix}
    a_1 \\
    \vdots \\
    a_n
\end{pmatrix}
\rightarrow
\begin{pmatrix}
    b_1 \\
    \vdots \\
    b_{n/2}
\end{pmatrix}
\text{recursive call}
\rightarrow
\begin{pmatrix}
    c_1 \\
    \vdots \\
    c_{n/2}
\end{pmatrix}
\]

\[
\begin{pmatrix}
    b_{n/2+1} \\
    \vdots \\
    b_n
\end{pmatrix}
\text{recursive call}
\rightarrow
\begin{pmatrix}
    c_{n/2+1} \\
    \vdots \\
    c_n
\end{pmatrix}
\]

$n/2$ randoms

\[
b_i \leftarrow a_i + r_i
\]

\[
b_{n/2+i} \leftarrow a_{n/2+i} + r_i
\]
$\mathcal{O}(n \log n)$ refresh gadget $G_{\text{refresh}}$ by Battistello et al. - CHES 2016:
$O(n \log n)$ refresh gadget $G_{\text{refresh}}$ by Battistello et al. - CHES 2016:

\[
\begin{align*}
L_1 & \quad \begin{pmatrix}
  a_1 \\
  \vdots \\
  a_n
\end{pmatrix} \quad \begin{pmatrix}
  b_1 \\
  \vdots \\
  b_{n/2}
\end{pmatrix} \\
R_1 & \quad \begin{pmatrix}
  b_{n/2+1} \\
  \vdots \\
  b_n
\end{pmatrix} \quad \begin{pmatrix}
  c_1 \\
  \vdots \\
  c_{n/2}
\end{pmatrix} \\
R_2 & \quad \begin{pmatrix}
  c_{n/2+1} \\
  \vdots \\
  c_n
\end{pmatrix} \\
L_0 & \quad \begin{pmatrix}
  d_1 \\
  \vdots \\
  d_n
\end{pmatrix}
\end{align*}
\]

Recursive call

\[
\begin{align*}
& n/2 \text{ randoms} \quad b_i \leftarrow a_i + r_i \\
& b_{n/2+i} \leftarrow a_{n/2+i} + r_i \\
& n/2 \text{ randoms} \quad d_i \leftarrow c_i + r'_i \\
& d_{n/2+i} \leftarrow c_{n/2+i} + r'_i
\end{align*}
\]
Example (4 shares):

\[
\begin{align*}
    d_1 &\leftarrow (a_1 + r_1) + r_3 + r_5 \\
    d_2 &\leftarrow (a_2 + r_2) + r_3 + r_6 \\
    d_3 &\leftarrow (a_3 + r_1) + r_4 + r_5 \\
    d_4 &\leftarrow (a_4 + r_2) + r_4 + r_6
\end{align*}
\]
Example (4 shares):

\[
\begin{align*}
    d_1 &\leftarrow (a_1 + r_1) + r_3 + r_5 \\
    d_2 &\leftarrow (a_2 + r_2) + r_3 + r_6 \\
    d_3 &\leftarrow (a_3 + r_1) + r_4 + r_5 \\
    d_4 &\leftarrow (a_4 + r_2) + r_4 + r_6
\end{align*}
\]

- proven by Battistello et al. to be \((n-1)\)-SNI in the probing model
Linear Gadgets

Building Block

Example (4 shares):

\[ d_1 \leftarrow (a_1 + r_1) + r_3 + r_5 \]
\[ d_2 \leftarrow (a_2 + r_2) + r_3 + r_6 \]
\[ d_3 \leftarrow (a_3 + r_1) + r_4 + r_5 \]
\[ d_4 \leftarrow (a_4 + r_2) + r_4 + r_6 \]

- proven by Battistello et al. to be \((n - 1)\)-SNI in the probing model

- proven in our work to satisfy stronger requirements to be used as a building block for RPE secure constructions (extension of requirements proposed by Belaïd et al. - EuroCrypt 2021)
Using $\mathcal{O}(n \log n)$ $G_{\text{refresh}}$
Using $O(n \log n)$ $G_{\text{refresh}}$

$G_{\text{add}}$

\[ a_1, \ldots, a_n \quad b_1, \ldots, b_n \]

\[ G_{\text{refresh}} \quad G_{\text{refresh}} \]

\[ e_1, \ldots, e_n \quad f_1, \ldots, f_n \]

\[ c_i = e_i + f_i \]
Using $\mathcal{O}(n \log n)$ $G_{\text{refresh}}$

**$G_{\text{add}}$**

\[
a_1, \ldots, a_n \quad b_1, \ldots, b_n
\]

\[
G_{\text{refresh}} \quad G_{\text{refresh}}
\]

\[
e_1, \ldots, e_n \quad f_1, \ldots, f_n
\]

\[
\begin{array}{c}
+ \\
\end{array}
\]

\[
c_i = e_i + f_i
\]

**$G_{\text{copy}}$**

\[
a_1, \ldots, a_n
\]

\[
G_{\text{refresh}} \quad G_{\text{refresh}}
\]

\[
e_1, \ldots, e_n \quad f_1, \ldots, f_n
\]
Using $O(n \log n)$ $G_{\text{refresh}}$

\[ G_{\text{add}} \]
\[ a_1, \ldots, a_n \quad b_1, \ldots, b_n \]
\[ G_{\text{refresh}} \quad G_{\text{refresh}} \]
\[ e_1, \ldots, e_n \quad f_1, \ldots, f_n \]
\[ + \]
\[ c_i = e_i + f_i \]

\[ G_{\text{copy}} \]
\[ a_1, \ldots, a_n \]
\[ G_{\text{refresh}} \quad G_{\text{refresh}} \]
\[ e_1, \ldots, e_n \quad f_1, \ldots, f_n \]

\[ G_{\text{cmult}} \]
\[ a_1, \ldots, a_n \]
\[ \times_c \]
\[ c \cdot a_1, \ldots, c \cdot a_n \]
\[ G_{\text{refresh}} \]
\[ e_1, \ldots, e_n \]
Linear Gadgets

Constructions

Using $O(n \log n)$ $G_{\text{refresh}}$

$G_{\text{add}}$

\[ a_1, \ldots, a_n \quad b_1, \ldots, b_n \]

\[ G_{\text{refresh}} \quad G_{\text{refresh}} \]

\[ e_1, \ldots, e_n \quad f_1, \ldots, f_n \]

$\quad +$

\[ c_i = e_i + f_i \]

\[ G_{\text{copy}} \]

\[ a_1, \ldots, a_n \]

\[ G_{\text{refresh}} \quad G_{\text{refresh}} \]

\[ e_1, \ldots, e_n \quad f_1, \ldots, f_n \]

\[ G_{\text{cmult}} \]

\[ a_1, \ldots, a_n \]

\[ \times c \]

\[ c \cdot a_1, \ldots, c \cdot a_n \]

\[ G_{\text{refresh}} \]

\[ e_1, \ldots, e_n \]

- Complexity in $O(n \log n)$
Linear Gadgets
Constructions

Using $\mathcal{O}(n \log n)$ $G_{\text{refresh}}$

- $G_{\text{add}}$
  \[ a_1, \ldots, a_n \quad b_1, \ldots, b_n \]
  \[ G_{\text{refresh}} \quad G_{\text{refresh}} \]
  \[ e_1, \ldots, e_n \quad f_1, \ldots, f_n \]
  \[ \text{+} \]
  \[ c_i = e_i + f_i \]

- $G_{\text{copy}}$
  \[ a_1, \ldots, a_n \]
  \[ G_{\text{refresh}} \quad G_{\text{refresh}} \]
  \[ e_1, \ldots, e_n \quad f_1, \ldots, f_n \]

- $G_{\text{cmult}}$
  \[ a_1, \ldots, a_n \]
  \[ \times_c \]
  \[ c \cdot a_1, \ldots, c \cdot a_n \]
  \[ G_{\text{refresh}} \]
  \[ e_1, \ldots, e_n \]

- Complexity in $\mathcal{O}(n \log n)$
- RPE secure with $d = d_{\text{max}} = \frac{n + 1}{2}$
$G_{\text{mult}}$ (over $\mathbb{K}$) construction from 2 subgadgets
Multiplication Gadget

Construction from $G_{\text{submult}}, G_{\text{compress}}$

$G_{\text{mult}}$ (over $\mathbb{K}$) construction from 2 subgadgets
Multiplication Gadget
Construction from $G_{\text{submult}}, G_{\text{compress}}$

$G_{\text{mult}}$ (over $\mathbb{K}$) construction from 2 subgadgets

- In classical constructions, $m = \mathcal{O}(n^2)$
**Multiplication Gadget**

**Construction from** $G_{\text{submult}}, G_{\text{compress}}$

$G_{\text{mult}}$ (over $\mathbb{K}$) construction from 2 subgadgets

- In classical constructions, $m = \mathcal{O}(n^2)$
- $G_{\text{mult}}$ must be RPE secure $\iff$ composition of $G_{\text{submult}}$ and $G_{\text{compress}}$ must be RPE secure
Multiplication Gadget
Construction from $G_{\text{submult}}, G_{\text{compress}}$

$G_{\text{mult}}$ (over $\mathbb{K}$) construction from 2 subgadgets

- In classical constructions, $m = \mathcal{O}(n^2)$
- $G_{\text{mult}}$ must be RPE secure $\implies$ composition of $G_{\text{submult}}$ and $G_{\text{compress}}$ must be RPE secure
- Extension of $G_{\text{submult}}$ by Belaïd et al. - Crypto 2017 with $m = 2n + 1$
Multiplication Gadget
Construction from $G_{\text{submult}}, G_{\text{compress}}$

$G_{\text{mult}}$ (over $\mathbb{K}$) construction from 2 subgadgets

- In classical constructions, $m = \mathcal{O}(n^2)$
- $G_{\text{mult}}$ must be RPE secure $\implies$ composition of $G_{\text{submult}}$ and $G_{\text{compress}}$ must be RPE secure
- Extension of $G_{\text{submult}}$ by Belaïd et al. - Crypto 2017 with $m = 2n + 1$
- New $G_{\text{compress}}$ with complexity in $\mathcal{O}(m \log m)$
Inputs $a,b$ (illustration with 3 shares), field $\mathbb{K}$
Multiplication Gadget
Extension of $G_{\text{submult}}$ by Belaïd et al. - Crypto 2017

Inputs $a, b$ (illustration with 3 shares), field $\mathbb{K}$

\[
\gamma = \begin{pmatrix}
\gamma_{1,1} & \gamma_{1,2} & \gamma_{1,3} \\
\gamma_{2,1} & \gamma_{2,2} & \gamma_{2,3} \\
\gamma_{3,1} & \gamma_{3,2} & \gamma_{3,3}
\end{pmatrix}
\]

\[
\delta = \begin{pmatrix}
1 - \gamma_{1,1} & 1 - \gamma_{2,1} & 1 - \gamma_{3,1} \\
1 - \gamma_{1,2} & 1 - \gamma_{2,2} & 1 - \gamma_{3,2} \\
1 - \gamma_{1,3} & 1 - \gamma_{2,3} & 1 - \gamma_{3,3}
\end{pmatrix}
\]
Multiplication Gadget

Extension of $G_{\text{submult}}$ by Belaïd et al. - Crypto 2017

Inputs $a, b$ (illustration with 3 shares), field $\mathbb{K}$

$$\gamma = \begin{pmatrix}
\gamma_{1,1} & \gamma_{1,2} & \gamma_{1,3} \\
\gamma_{2,1} & \gamma_{2,2} & \gamma_{2,3} \\
\gamma_{3,1} & \gamma_{3,2} & \gamma_{3,3}
\end{pmatrix}$$

$$\delta = \begin{pmatrix}
1 - \gamma_{1,1} & 1 - \gamma_{2,1} & 1 - \gamma_{3,1} \\
1 - \gamma_{1,2} & 1 - \gamma_{2,2} & 1 - \gamma_{3,2} \\
1 - \gamma_{1,3} & 1 - \gamma_{2,3} & 1 - \gamma_{3,3}
\end{pmatrix}$$

$$c_1 \leftarrow ((r_1 + a_1) + (r_2 + a_2) + (r_3 + a_3)) \cdot ((s_1 + b_1) + (s_2 + b_2) + (s_3 + b_3))$$
Inputs $a, b$ (illustration with 3 shares), field $\mathbb{K}$

$$\gamma = \begin{pmatrix}
\gamma_{1,1} & \gamma_{1,2} & \gamma_{1,3} \\
\gamma_{2,1} & \gamma_{2,2} & \gamma_{2,3} \\
\gamma_{3,1} & \gamma_{3,2} & \gamma_{3,3}
\end{pmatrix}$$

$$\delta = \begin{pmatrix}
1 - \gamma_{1,1} & 1 - \gamma_{2,1} & 1 - \gamma_{3,1} \\
1 - \gamma_{1,2} & 1 - \gamma_{2,2} & 1 - \gamma_{3,2} \\
1 - \gamma_{1,3} & 1 - \gamma_{2,3} & 1 - \gamma_{3,3}
\end{pmatrix}$$

$$c_1 \leftarrow ((r_1 + a_1) + (r_2 + a_2) + (r_3 + a_3)) \cdot ((s_1 + b_1) + (s_2 + b_2) + (s_3 + b_3))$$

$$c_2 \leftarrow -r_1 \cdot ((\delta_{1,1} \cdot s_1 + b_1) + (\delta_{1,2} \cdot s_2 + b_2) + (\delta_{1,3} \cdot s_3 + b_3))$$

$$c_3 \leftarrow -r_2 \cdot ((\delta_{2,1} \cdot s_1 + b_1) + (\delta_{2,2} \cdot s_2 + b_2) + (\delta_{2,3} \cdot s_3 + b_3))$$

$$c_4 \leftarrow -r_3 \cdot ((\delta_{3,1} \cdot s_1 + b_1) + (\delta_{3,2} \cdot s_2 + b_2) + (\delta_{3,3} \cdot s_3 + b_3))$$
Multiplication Gadget
Extension of $G_{\text{submult}}$ by Belaïd et al. - Crypto 2017

Inputs $a, b$ (illustration with 3 shares), field $\mathbb{K}$

\[
\gamma = \begin{pmatrix}
\gamma_{1,1} & \gamma_{1,2} & \gamma_{1,3} \\
\gamma_{2,1} & \gamma_{2,2} & \gamma_{2,3} \\
\gamma_{3,1} & \gamma_{3,2} & \gamma_{3,3}
\end{pmatrix}
\]

\[
\delta = \begin{pmatrix}
1 - \gamma_{1,1} & 1 - \gamma_{2,1} & 1 - \gamma_{3,1} \\
1 - \gamma_{1,2} & 1 - \gamma_{2,2} & 1 - \gamma_{3,2} \\
1 - \gamma_{1,3} & 1 - \gamma_{2,3} & 1 - \gamma_{3,3}
\end{pmatrix}
\]

\[
c_1 \leftarrow ((r_1 + a_1) + (r_2 + a_2) + (r_3 + a_3)) \cdot ((s_1 + b_1) + (s_2 + b_2) + (s_3 + b_3))
\]

\[
c_2 \leftarrow -r_1 \cdot ((\delta_{1,1} \cdot s_1 + b_1) + (\delta_{1,2} \cdot s_2 + b_2) + (\delta_{1,3} \cdot s_3 + b_3))
\]

\[
c_3 \leftarrow -r_2 \cdot ((\delta_{2,1} \cdot s_1 + b_1) + (\delta_{2,2} \cdot s_2 + b_2) + (\delta_{2,3} \cdot s_3 + b_3))
\]

\[
c_4 \leftarrow -r_3 \cdot ((\delta_{3,1} \cdot s_1 + b_1) + (\delta_{3,2} \cdot s_2 + b_2) + (\delta_{3,3} \cdot s_3 + b_3))
\]

\[
c_5 \leftarrow -s_1 \cdot ((\gamma_{1,1} \cdot r_1 + a_1) + (\gamma_{1,2} \cdot r_2 + a_2) + (\gamma_{1,3} \cdot r_3 + a_3))
\]

\[
c_6 \leftarrow -s_2 \cdot ((\gamma_{2,1} \cdot r_1 + a_1) + (\gamma_{2,2} \cdot r_2 + a_2) + (\gamma_{2,3} \cdot r_3 + a_3))
\]

\[
c_7 \leftarrow -s_3 \cdot ((\gamma_{3,1} \cdot r_1 + a_1) + (\gamma_{3,2} \cdot r_2 + a_2) + (\gamma_{3,3} \cdot r_3 + a_3))
\]
$G_{\text{submult}}$ uses $2^n$ random values, outputs $2^n + 1$ shares, performs $2^n + 1$ multiplications, performs $2^n 2$ multiplications by a constant. It is proven to be secure for $G_{\text{mult}} RPE secure construction, for the right choice of constants in $\gamma$ (can be chosen uniformly at random if the field is large enough).
Multiplication Gadget
Extension of $G_{\text{submult}}$ by Belaïd et al. - Crypto 2017

$G_{\text{submult}}$
- uses $2n$ random values
Multiplication Gadget

Extension of $G_{\text{submult}}$ by Belaïd et al. - Crypto 2017

$G_{\text{submult}}$

- uses $2n$ random values
- outputs $2n + 1$ shares
Multiplication Gadget
Extension of $G_{\text{submult}}$ by Belaïd et al. - Crypto 2017

$G_{\text{submult}}$

- uses $2n$ random values
- outputs $2n + 1$ shares
- performs $2n + 1$ multiplications operations
Multiplication Gadget
Extension of $G_{\text{submult}}$ by Belaïd et al. - Crypto 2017

$G_{\text{submult}}$
- uses $2n$ random values
- outputs $2n + 1$ shares
- performs $2n + 1$ multiplications operations
- performs $2n^2$ multiplications by a constant
Multiplication Gadget

Extension of $G_{\text{submult}}$ by Belaïd et al. - Crypto 2017

$G_{\text{submult}}$

- uses $2n$ random values
- outputs $2n + 1$ shares
- performs $2n + 1$ multiplications operations
- performs $2n^2$ multiplications by a constant
- is proven to be secure for $G_{\text{mult}}$ RPE secure construction, for the right choice of constants in $\gamma$ (can be chosen uniformly at random if the field is large enough)
The \([m : n]\)-compression gadget proposed by Belaïd et al. - Crypto 2017 is not secure as claimed.
The \([m : n]\)-compression gadget proposed by Belaïd et al. - Crypto 2017 is not secure as claimed.

New Compression gadget

\[
\begin{align*}
a_1, \ldots, a_m \\
\text{G}_{\text{refresh}} \\
c_1, \ldots, c_n, c_{n+1}, \ldots, c_{2n}, \ldots, c_{Kn+1}, \ldots, c_m \\
\ldots \\
\mathbf{d}_i = c_i + c_{n+i} + \ldots + c_{Kn+i}
\end{align*}
\]
New Construction of $G_{\text{compress}}$

New $G_{\text{compress}}$
New $G_{\text{compress}}$

- is of size $\mathcal{O}(|G_{\text{refresh}}| + m)$
New \( G_{\text{compress}} \)

- is of size \( \mathcal{O}(|G_{\text{refresh}}| + m) \)

- using \( \mathcal{O}(n \log n) \) \( G_{\text{refresh}} \), has complexity \( \mathcal{O}(m \log m) \)
Multiplication Gadget
New Construction of $G_{\text{compress}}$

New $G_{\text{compress}}$
- is of size $O(|G_{\text{refresh}}| + m)$
- using $O(n \log n)$ $G_{\text{refresh}}$, has complexity $O(m \log m)$
- With $m = O(n)$ (from $G_{\text{submult}}$), has complexity $O(n \log n)$
New $G_{\text{compress}}$

- is of size $\mathcal{O}(|G_{\text{refresh}}| + m)$

- using $\mathcal{O}(n \log n)$ $G_{\text{refresh}}$, has complexity $\mathcal{O}(m \log m)$

- With $m = \mathcal{O}(n)$ (from $G_{\text{submult}}$), has complexity $\mathcal{O}(n \log n)$

- is proven secure for $G_{\text{mult}}$ RPE secure construction
New $G_{\text{compress}}$

- is of size $\mathcal{O}(|G_{\text{refresh}}| + m)$
- using $\mathcal{O}(n \log n)$ $G_{\text{refresh}}$, has complexity $\mathcal{O}(m \log m)$
- With $m = \mathcal{O}(n)$ (from $G_{\text{submult}}$), has complexity $\mathcal{O}(n \log n)$
- is proven secure for $G_{\text{mult}}$ RPE secure construction

Using $G_{\text{submult}}$ described earlier, and new $G_{\text{compress}}$, we get $G_{\text{mult}}$: 
Multiplication Gadget
New Construction of \( G_{\text{compress}} \)

New \( G_{\text{compress}} \)
- is of size \( \mathcal{O}(|G_{\text{refresh}}| + m) \)
- using \( \mathcal{O}(n \log n) \) \( G_{\text{refresh}} \), has complexity \( \mathcal{O}(m \log m) \)
- With \( m = \mathcal{O}(n) \) (from \( G_{\text{submult}} \)), has complexity \( \mathcal{O}(n \log n) \)
- is proven secure for \( G_{\text{mult}} \) RPE secure construction

Using \( G_{\text{submult}} \) described earlier, and new \( G_{\text{compress}} \), we get \( G_{\text{mult}} \):
- performs \( \mathcal{O}(n) \) multiplications between variables
New $G_{\text{compress}}$

- is of size $\mathcal{O}(|G_{\text{refresh}}| + m)$
- using $\mathcal{O}(n \log n)$ $G_{\text{refresh}}$, has complexity $\mathcal{O}(m \log m)$
- With $m = \mathcal{O}(n)$ (from $G_{\text{submult}}$), has complexity $\mathcal{O}(n \log n)$
- is proven secure for $G_{\text{mult}}$ RPE secure construction

Using $G_{\text{submult}}$ described earlier, and new $G_{\text{compress}}$, we get $G_{\text{mult}}$:

- performs $\mathcal{O}(n)$ multiplications between variables
- uses $\mathcal{O}(n \log n)$ random values
New $G_{\text{compress}}$
- is of size $O(|G_{\text{refresh}}| + m)$
- using $O(n \log n)$ $G_{\text{refresh}}$, has complexity $O(m \log m)$
- With $m = O(n)$ (from $G_{\text{submult}}$), has complexity $O(n \log n)$
- is proven secure for $G_{\text{mult}}$ RPE secure construction

Using $G_{\text{submult}}$ described earlier, and new $G_{\text{compress}}$, we get $G_{\text{mult}}$:
- performs $O(n)$ multiplications between variables
- uses $O(n \log n)$ random values
- is RPE secure with amplification order $d = d_{\text{max}} = \frac{n + 1}{2}$
New RPE Compiler
With Quasi-Linear Asymptotic Complexity

New Linear gadgets $G_{\text{add}}$, $G_{\text{copy}}$, $G_{\text{cmult}}$ with $\mathcal{O}(n \log n)$ complexity
New RPE Compiler
With Quasi-Linear Asymptotic Complexity

New Linear gadgets $G_{\text{add}}$, $G_{\text{copy}}$, $G_{\text{cmult}}$ with $O(n \log n)$ complexity

New $G_{\text{mult}}$ with $O(n)$ multiplications between variables
New RPE Compiler

With Quasi-Linear Asymptotic Complexity

New Linear gadgets $G_{\text{add}}$, $G_{\text{copy}}$, $G_{\text{cmult}}$ with $O(n \log n)$ complexity

New $G_{\text{mult}}$ with $O(n)$ multiplications between variables

All gadgets of amplification order $d = \frac{n + 1}{2}$
New RPE Compiler
With Quasi-Linear Asymptotic Complexity

New Linear gadgets $G_{\text{add}}$, $G_{\text{copy}}$, $G_{\text{cmult}}$ with $O(n \log n)$ complexity

New $G_{\text{mult}}$ with $O(n)$ multiplications between variables

All gadgets of amplification order $d = \frac{n + 1}{2}$

Complexity of expansion of a circuit $C$:

$$O(|C|.\kappa^e), \quad e = \frac{\log(N_{\text{max}})}{\log(d)}$$
New RPE Compiler
With Quasi-Linear Asymptotic Complexity

New Linear gadgets $G_{\text{add}}$, $G_{\text{copy}}$, $G_{\text{cmult}}$ with $O(n \log n)$ complexity

New $G_{\text{mult}}$ with $O(n)$ multiplications between variables

All gadgets of amplification order $d = \frac{n + 1}{2}$

Complexity of expansion of a circuit $C$:

$$O(|C| \cdot \kappa^e), \quad e = \frac{\log(N_{\text{max}})}{\log(d)}$$

$N_{\text{max}} \approx \max(\# \times \text{ in } G_{\text{mult}}, \#(+, ||) \text{ in } G_{\text{add}}, G_{\text{copy}}, \# \times c \text{ in } G_{\text{cmult}}) = O(n \log n)$
New RPE Compiler
With Quasi-Linear Asymptotic Complexity

\[ O(|C| \cdot \kappa^e), \quad e = \frac{\log(N_{\max})}{\log(d)} \]

Previously best compiler with \( N_{\max} = O(n^2), \quad d = (n + 1)/2 \)
New RPE compiler with \( N_{\max} = O(n \log n), \quad d = (n + 1)/2 \)
Conclusion

- Construction of new RPE compiler with quasilinear complexity from
Construction of new RPE compiler with quasilinear complexity from

- $n$-share $G_{\text{add}}$, $G_{\text{copy}}$, $G_{\text{cmult}}$ with $\mathcal{O}(n \log n)$ complexity
Conclusion

- Construction of new RPE compiler with quasilinear complexity from
  - $n$-share $G_{\text{add}}, G_{\text{copy}}, G_{\text{cmult}}$ with $O(n \log n)$ complexity
  - $n$-share $G_{\text{mult}}$ with $O(n \log n)$ randomness and $O(n)$ multiplications between variables

Dynamic RPE (different compilers) is more interesting than static RPE (single compiler)
- Start with RPE compiler with small nb. of shares tolerating the best leakage rate
- Continue with RPE compiler with best asymptotic complexity (e.g. our new RPE compiler)

Future work: Find gadgets with small nb. of shares (e.g. 3 shares) which tolerate the best possible leakage rate
Conclusion

- Construction of new RPE compiler with quasilinear complexity from
  - $n$-share $G_{\text{add}}$, $G_{\text{copy}}$, $G_{\text{cmult}}$ with $O(n \log n)$ complexity
  - $n$-share $G_{\text{mult}}$ with $O(n \log n)$ randomness and $O(n)$ multiplications between variables

- Dynamic RPE (different compilers) is more interesting than static RPE (single compiler)
Conclusion

- Construction of new RPE compiler with quasilinear complexity from
  - $n$-share $G_{\text{add}}, G_{\text{copy}}, G_{\text{cmult}}$ with $O(n \log n)$ complexity
  - $n$-share $G_{\text{mult}}$ with $O(n \log n)$ randomness and $O(n)$ multiplications between variables

- Dynamic RPE (different compilers) is more interesting than static RPE (single compiler)
  - start with RPE compiler with small nb. of shares tolerating the best leakage rate
Conclusion

- Construction of new RPE compiler with quasilinear complexity from
  - $n$-share $G_{\text{add}}, G_{\text{copy}}, G_{\text{cmult}}$ with $O(n \log n)$ complexity
  - $n$-share $G_{\text{mult}}$ with $O(n \log n)$ randomness and $O(n)$ multiplications between variables

- Dynamic RPE (different compilers) is more interesting than static RPE (single compiler)
  - start with RPE compiler with small nb. of shares tolerating the best leakage rate
  - continue with RPE compiler with best asymptotic complexity (e.g. our new RPE compiler)
Conclusion

- Construction of new RPE compiler with quasilinear complexity from
  - $n$-share $G_{\text{add}}, G_{\text{copy}}, G_{\text{cmult}}$ with $O(n \log n)$ complexity
  - $n$-share $G_{\text{mult}}$ with $O(n \log n)$ randomness and $O(n)$ multiplications between variables

- Dynamic RPE (different compilers) is more interesting than static RPE (single compiler)
  - start with RPE compiler with small nb. of shares tolerating the best leakage rate
  - continue with RPE compiler with best asymptotic complexity (e.g. our new RPE compiler)

- Future work: Find gadgets with small nb. of shares (e.g. 3 shares) which tolerate the best possible leakage rate