Secure Conversion Between Boolean and Arithmetic Masking of Any Order

Jean-Sébastien Coron Johann Großschädl Praveen Kumar Vadnala

University of Luxembourg, Luxembourg

CHES, 2014. Busan, Korea.

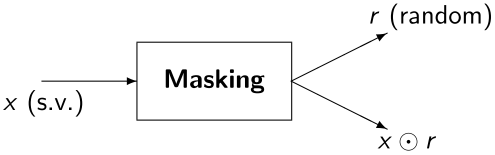
Countermeasures against side-channel attacks

- Hiding
  - Shuffling, Dummy instructions, $\cdots$
  - Efficient but ad-hoc
- Masking
  - Each sensitive variable is masked with a random value

  - Second and higher order masking
  - Higher the number of masks used, better the security
  - Security can be proved

Countermeasures against side-channel attacks

- Hiding
  - Shuffling, Dummy instructions, $\cdots$
  - Efficient but ad-hoc
- Masking
  - Each sensitive variable is masked with a random value



  - Second and higher order masking
  - Higher the number of masks used, better the security
  - Security can be proved

Masking types

- Boolean masking
  - Masked using XOR operation
  - Compatible with: XOR, shift etc.
- Arithmetic masking
- Multiplicative masking
- Conversion problem
- Applications: IDEA, HMAC-SHA1, ARX based ciphers, GOST, ...
- This talk : Conversion between Boolean and arithmetic masking

Masking types

- Boolean masking
  - Masked using XOR operation
  - Compatible with: XOR, shift etc.
- Arithmetic masking
- Multiplicative masking
- Conversion problem
- Applications: IDEA, HMAC-SHA1, ARX based ciphers, GOST, ...
- This talk : Conversion between Boolean and arithmetic masking

Masking types

- Boolean masking
  - Masked using XOR operation
  - Compatible with: XOR, shift etc.
- Arithmetic masking
- Multiplicative masking
- Conversion problem
- Applications: IDEA, HMAC-SHA1, ARX based ciphers, GOST, ...
- This talk : Conversion between Boolean and arithmetic masking

Masking types

- Boolean masking
  - Masked using XOR operation
  - Compatible with: XOR, shift etc.
- Arithmetic masking
- Multiplicative masking
- Conversion problem
- Applications: IDEA, HMAC-SHA1, ARX based ciphers, GOST, ...
- This talk : Conversion between Boolean and arithmetic masking

First order secure conversion

- Goubin solution (CHES, 2001)
  - B→A: Constant number of operations
  - A→B: Number of operations proportional to the size of the masked data
- Improved A→B solution by Coron and Tchulkine (CHES, 2003)
- Blandine Debraize's solution (CHES, 2012)
- Karroumi *et al.* secure addition (COSADE, 2014)

Higher order conversion

- No higher order conversion algorithms to date
- Genralizing Goubin's solution to higher order?
- Second order secure conversion by Vadnala-Großschädl at SPACE-2013
  - First step but inefficient in practice
  - No generalization for any order

Higher order conversion

- No higher order conversion algorithms to date
- Genralizing Goubin's solution to higher order?
- Second order secure conversion by Vadnala-Großschädl at SPACE-2013
    - First step but inefficient in practice
    - No generalization for any order

Our contributions

- First higher order secure conversion - Two approaches
  - Perform addition directly on Boolean shares
  - Convert from one form to the other
- Security proof in Ishai-Sahai and Wagner (ISW) framework
- Application to HMAC-SHA-1

Our contributions

- First higher order secure conversion - Two approaches
  - Perform addition directly on Boolean shares
  - Convert from one form to the other
- Security proof in Ishai-Sahai and Wagner (ISW) framework
- Application to HMAC-SHA-1

Our contributions

- First higher order secure conversion - Two approaches
  - Perform addition directly on Boolean shares
  - Convert from one form to the other
- Security proof in Ishai-Sahai and Wagner (ISW) framework
- Application to HMAC-SHA-1

Proving masking schemes

- Classical model
- Limitations of classical model
- ISW framework

Security model: ISW framework

- Visualize the implementation of cryptosystem in terms of Boolean circuit (C)

Theorem (Ishai, Sahai, Wagner)

There exists a perfectly t-private stateless transformer (T, I, O) such that T maps any stateless circuit C of size $|C|$ and depth d to a randomized stateless circuit of size $\mathcal{O}(n^2 \cdot |C|)$ and depth $\mathcal{O}(d \log t)$, where $n = 2t + 1$.

- Represent the circuit C using only NOT and AND gates
- Converting NOT gate is easy: if $x = x_1 \oplus x_2 \oplus \cdots \oplus x_n$ then $\mathrm{NOT}(x) = \mathrm{NOT}(x_1) \oplus x_2 \oplus \cdots \oplus x_n$
- How to convert AND gate? SecAnd function

Security model: ISW framework

- Visualize the implementation of cryptosystem in terms of Boolean circuit (C)

Theorem (Ishai, Sahai, Wagner)

There exists a perfectly t-private stateless transformer (T, I, O) *such that* T *maps any stateless circuit* C *of size* $|C|$ *and depth* d *to a randomized stateless circuit of size* $\mathcal{O}(n^2 \cdot |C|)$ *and depth* $\mathcal{O}(d \log t)$*, where* $n = 2t + 1$.

- Represent the circuit C using only NOT and AND gates
- Converting NOT gate is easy: if $x = x_1 \oplus x_2 \oplus \cdots \oplus x_n$ then $\mathrm{NOT}(x) = \mathrm{NOT}(x_1) \oplus x_2 \oplus \cdots \oplus x_n$
- How to convert AND gate? SecAnd function

Security model: ISW framework

- Visualize the implementation of cryptosystem in terms of Boolean circuit (C)

Theorem (Ishai, Sahai, Wagner)

There exists a perfectly t-private stateless transformer (T, I, O) *such that* T *maps any stateless circuit* C *of size* $|C|$ *and depth* d *to a randomized stateless circuit of size* $\mathcal{O}(n^2 \cdot |C|)$ *and depth* $\mathcal{O}(d \log t)$, *where* $n = 2t + 1$.

- Represent the circuit C using only NOT and AND gates
- Converting NOT gate is easy: if $x = x_1 \oplus x_2 \oplus \cdots \oplus x_n$ then $\mathrm{NOT}(x) = \mathrm{NOT}(x_1) \oplus x_2 \oplus \cdots \oplus x_n$
- How to convert AND gate? SecAnd function

Higher order secure addition

- Represent modular addition as Boolean circuit
- Apply ISW method
- Two approaches
  - A modular addition of two k-bit variables x and y can be defined recursively as $(x+y)^{(i)} = x^{(i)} \oplus y^{(i)} \oplus c^{(i)}$, where

$$\begin{cases} c^{(0)} = 0 \\ \forall i \geq 1, c^{(i)} = (x^{(i-1)} \wedge y^{(i-1)}) \oplus (x^{(i-1)} \wedge c^{(i-1)}) \oplus (c^{(i-1)} \wedge y^{(i-1)}) \end{cases}$$

  - Use Goubin's formula: $x + y = x \oplus y \oplus u_{k-1}$, where u_{k-1} is obtained from the following recursion formula:

$$\begin{cases} u_0 = 0 \\ \forall i \geq 0, u_{i+1} = 2[u_i \wedge (x \oplus y) \oplus (x \wedge y)] \end{cases}$$

Higher order secure addition

- Represent modular addition as Boolean circuit
- Apply ISW method
- Two approaches
  - A modular addition of two k-bit variables x and y can be defined recursively as $(x+y)^{(i)} = x^{(i)} \oplus y^{(i)} \oplus c^{(i)}$, where

$$\begin{cases} c^{(0)} = 0 \\ \forall i \geq 1, c^{(i)} = (x^{(i-1)} \wedge y^{(i-1)}) \oplus (x^{(i-1)} \wedge c^{(i-1)}) \oplus (c^{(i-1)} \wedge y^{(i-1)}) \end{cases}$$

  - Use Goubin's formula: $x + y = x \oplus y \oplus u_{k-1}$, where u_{k-1} is obtained from the following recursion formula:

$$\begin{cases} u_0 = 0 \\ \forall i \geq 0, u_{i+1} = 2[u_i \wedge (x \oplus y) \oplus (x \wedge y)] \end{cases}$$

First variant

Algorithm 1 SecAdd

Input: (x_i) and (y_i) for $1 \le i \le n$

Output: (z_i) for $1 \le i \le n$, with $\bigoplus_{i=1}^{n} z_i = \bigoplus_{i=1}^{n} x_i + \bigoplus_{i=1}^{n} y_i$

$(c_i^{(0)})_{1 \le i \le n} \leftarrow 0$ $\quad\triangleright$ Initially carry is zero
for $j = 0$ to $k-2$ **do** $\quad\triangleright$ Compute carry bit by bit
  $(xy_i^{(j)})_{1 \le i \le n} \leftarrow \mathsf{SecAnd}((x_i^{(j)})_{1 \le i \le n}, (y_i^{(j)})_{1 \le i \le n})$ $\quad\triangleright\ x^{(j)} \wedge y^{(j)}$
  $(xc_i^{(j)})_{1 \le i \le n} \leftarrow \mathsf{SecAnd}((x_i^{(j)})_{1 \le i \le n}, (c_i^{(j)})_{1 \le i \le n})$ $\quad\triangleright\ x^{(j)} \wedge c^{(j)}$
  $(yc_i^{(j)})_{1 \le i \le n} \leftarrow \mathsf{SecAnd}((y_i^{(j)})_{1 \le i \le n}, (c_i^{(j)})_{1 \le i \le n})$ $\quad\triangleright\ y^{(j)} \wedge c^{(j)}$
  $(c_i^{(j+1)})_{1 \le i \le n} \leftarrow (xy_i^{(j)})_{1 \le i \le n} \oplus (xc_i^{(j)})_{1 \le i \le n} \oplus (yc_i^{(j)})_{1 \le i \le n}$
end for
$(z_i)_{1 \le i \le n} \leftarrow (x_i)_{1 \le i \le n} \oplus (y_i)_{1 \le i \le n} \oplus (c_i)_{1 \le i \le n}$ $\quad\triangleright\ z = x + y = x \oplus y \oplus c$
return $(z_i)_{1 \le i \le n}$

Second variant

Algorithm 2 SecAddGoubin

Input: (x_i) and (y_i) for $1 \le i \le n$

Output: (z_i) for $1 \le i \le n$, with $\bigoplus_{i=1}^{n} z_i = \bigoplus_{i=1}^{n} x_i + \bigoplus_{i=1}^{n} y_i$

$(w_i)_{1 \le i \le n} \leftarrow \mathrm{SecAnd}((x_i)_{1 \le i \le n}, (y_i)_{1 \le i \le n})$ ▷ $\omega = x \wedge y$

$(u_i)_{1 \le i \le n} \leftarrow 0$ ▷ Initialize shares of u to zero

$(a_i)_{1 \le i \le n} \leftarrow (x_i)_{1 \le i \le n} \oplus (y_i)_{1 \le i \le n}$ ▷ $a = x \oplus y$

for $j = 1$ to $k - 1$ **do**

    $(ua_i)_{1 \le i \le n} \leftarrow \mathrm{SecAnd}\big((u_i)_{1 \le i \le n}, (a_i)_{1 \le i \le n}\big)$

    $(u_i)_{1 \le i \le n} \leftarrow (ua_i)_{1 \le i \le n} \oplus (w_i)_{1 \le i \le n}$

    $(u_i)_{1 \le i \le n} \leftarrow 2(u_i)_{1 \le i \le n}$ ▷ $u \leftarrow 2(u \wedge a \oplus \omega)$

end for

$(z_i)_{1 \le i \le n} \leftarrow (x_i)_{1 \le i \le n} \oplus (y_i)_{1 \le i \le n} \oplus (u_i)_{1 \le i \le n}$ ▷ $z = x + y = x \oplus y \oplus u$

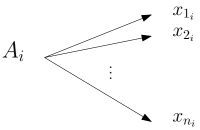
return $(z_i)_{1 \le i \le n}$

Analysis

- Both algorithms have running time in $\mathcal{O}(n^2 k)$
  - SecAnd: $\mathcal{O}(n^2)$
  - k size of the shares
- In practice, second variant is more efficient
  - Less calls to SecAnd function
  - No need to perform bit manipulations

Secure conversion from arithmetic to Boolean masking: Simple solution

- Assume $x = A_1 + \cdots + A_n$
- Re-share each of the arithmetic shares A_i $(1 \leq i \leq n)$ into n random Boolean shares $x_{i,j}$ $(1 \leq j \leq n)$ so that $A_i = x_{i,1} \oplus \cdots \oplus x_{i,n}$

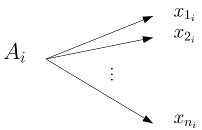


- The sensitive variable x is now given as:

$$x = (x_{1,1} \oplus \cdots \oplus x_{1,n}) + \cdots + (x_{n,1} \oplus \cdots \oplus x_{n,n})$$

Secure conversion from arithmetic to Boolean masking: Simple solution

- Assume $x = A_1 + \cdots + A_n$
- Re-share each of the arithmetic shares A_i $(1 \leq i \leq n)$ into n random Boolean shares $x_{i,j}$ $(1 \leq j \leq n)$ so that $A_i = x_{i,1} \oplus \cdots \oplus x_{i,n}$

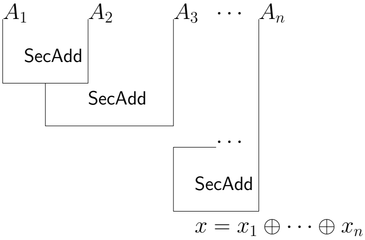


- The sensitive variable x is now given as:

$$x = (x_{1,1} \oplus \cdots \oplus x_{1,n}) + \cdots + (x_{n,1} \oplus \cdots \oplus x_{n,n})$$

Secure conversion from arithmetic to Boolean masking: Simple solution

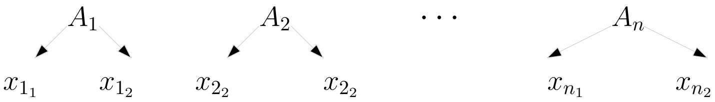
- Now perform secure addition using one of the variants



- Time complexity: $\mathcal{O}(n^3 k)$

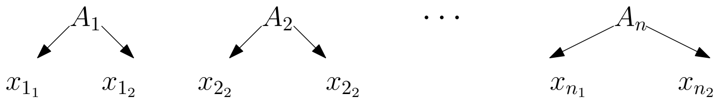
Improved conversion from Arithmetic to Boolean masking

- Use lesser shares at every step instead of n^2 shares
- Build a bottom-up solution
- Start with two shares for every A_i



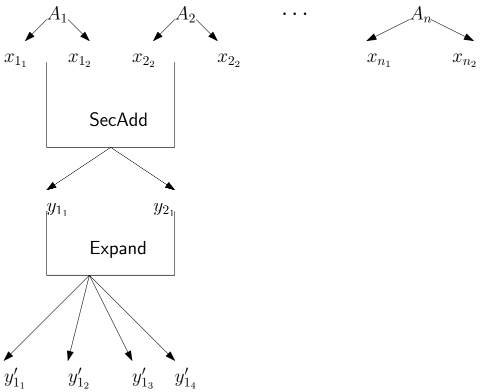
Improved conversion from Arithmetic to Boolean masking

- Use lesser shares at every step instead of n^2 shares
- Build a bottom-up solution
- Start with two shares for every A_i



Improved conversion from Arithmetic to Boolean masking

- At every step, halve the number of additive shares and double the number of Boolean shares (Binary tree)



- Number of shares $\leq 2n$ at every level $\implies \mathcal{O}(n^2 k)$ complexity

Putting it altogether...

Algorithm 3 ConvertA→B

Input: (A_i) for $1 \le i \le n$

Output: (z_i) for $1 \le i \le n$, with $\bigoplus_{i=1}^{n} z_i = \sum_{i=1}^{n} A_i$

1: If $n = 1$ then **return** A_1

2: $(x_i)_{1 \le i \le n/2} \leftarrow$ ConvertA→B $\left((A_i)_{1 \le i \le n/2}\right)$

3: $(x'_i)_{1 \le i \le n} \leftarrow$ Expand $\left((x_i)_{1 \le i \le n/2}\right)$ $\quad \triangleright \bigoplus_{i=1}^{n} x'_i = \bigoplus_{i=1}^{n/2} x_i = \sum_{i=1}^{n/2} A_i$

4: $(y_i)_{1 \le i \le n/2} \leftarrow$ ConvertA→B $\left((A_i)_{n/2+1 \le i \le n}\right)$

5: $(y'_i)_{1 \le i \le n} \leftarrow$ Expand $\left((y_i)_{1 \le i \le n/2}\right)$ $\quad \triangleright \bigoplus_{i=1}^{n} y'_i = \bigoplus_{i=1}^{n/2} y_i = \sum_{i=n/2+1}^{n} A_i$

6: $(z_i)_{1 \le i \le n} \leftarrow$ SecAdd $\left((x'_i)_{1 \le i \le n},\ (y'_i)_{1 \le i \le n}\right)$

7: **return** $(z_i)_{1 \le i \le n}$ $\quad \triangleright \bigoplus_{i=1}^{n} z_i = \bigoplus_{i=1}^{n} x'_i + \bigoplus_{i=1}^{n} y'_i = \sum_{i=1}^{n} A_i$

Secure conversion from Boolean to arithmetic masking

- Given $x = x_1 \oplus \cdots \oplus x_n$ compute $A_1, \cdots, A_n$ so that $x = A_1 + \cdots + A_n$
- Idea: Take advantage of ConvertA$\rightarrow$B and SecAdd
- Generate $(A_i)_{1 \leq i \leq n-1}$ randomly
- Compute $A_n = x - (A_1 + \cdots + A_{n-1}) = x + (-A_1 - \cdots - A_{n-1})$
- Complexity : $\mathcal{O}(n^2 k)$, but inefficient compared to ConvertA$\rightarrow$B

Secure conversion from Boolean to arithmetic masking

- Given $x = x_1 \oplus \cdots \oplus x_n$ compute $A_1, \cdots, A_n$ so that $x = A_1 + \cdots + A_n$
- Idea: Take advantage of ConvertA$\rightarrow$B and SecAdd
- Generate $(A_i)_{1 \leq i \leq n-1}$ randomly
- Compute $A_n = x - (A_1 + \cdots + A_{n-1}) = x + (-A_1 - \cdots - A_{n-1})$
- Complexity : $\mathcal{O}(n^2 k)$, but inefficient compared to ConvertA$\rightarrow$B

Experimental results

Algorithm	Time	rand
second-order addition		
Algorithm 1	87	1240
Algorithm 2	26	320
second-order conversion		
Algorithm 3	54	484
Algorithm B→A	81	822
third-order addition		
Algorithm 1	156	2604
Algorithm 2	46	672
third-order conversion		
Algorithm 3	121	1288
Algorithm B→A	162	1997

Table : Execution times of all algorithms (in thousands of clock cycles) for $t = 2, 3$ and the number of calls to the rand function

Application to HMAC-SHA-1

Algorithm	Time	Penalty
HMAC-SHA-1	104	1
second-order addition		
Algorithm 1	57172	549
Algorithm 2	17847	171
second-order conversion		
Algorithm 3, B$\rightarrow$A	62669	602
third-order addition		
Algorithm 1	106292	987
Algorithm 2	31195	299
third-order conversion		
Algorithm 3, B$\rightarrow$A	127348	1224

Table : Execution times of second and third-order secure masking (in thousands of clock cycles) and performance penalty compared to an unmasked implementation of HMAC-SHA-1

Conclusion

- First higher order secure B→A and A→B conversion
- Proofs in ISW model
- Generic solution: Applicable to number of cryptosystems
- Future work
  - Improved solution for B→A?
  - Improved solutions for $n \geq 3$?

Conclusion

- First higher order secure B$\rightarrow$A and A$\rightarrow$B conversion
- Proofs in ISW model
- Generic solution: Applicable to number of cryptosystems
- Future work
  - Improved solution for B$\rightarrow$A?
  - Improved solutions for $n \geq 3$?