End-to-end Design of a PUF-based Privacy Preserving Authentication Protocol

Aydin Aysu[1], Ege Gulcan, Daisuke Moriyama[2], Patrick Schaumont, and Moti Yung

Navid Vafaei
Sepand Haghighi

Fall - 2016

Hardware Security & Trust - Dr.Bayat

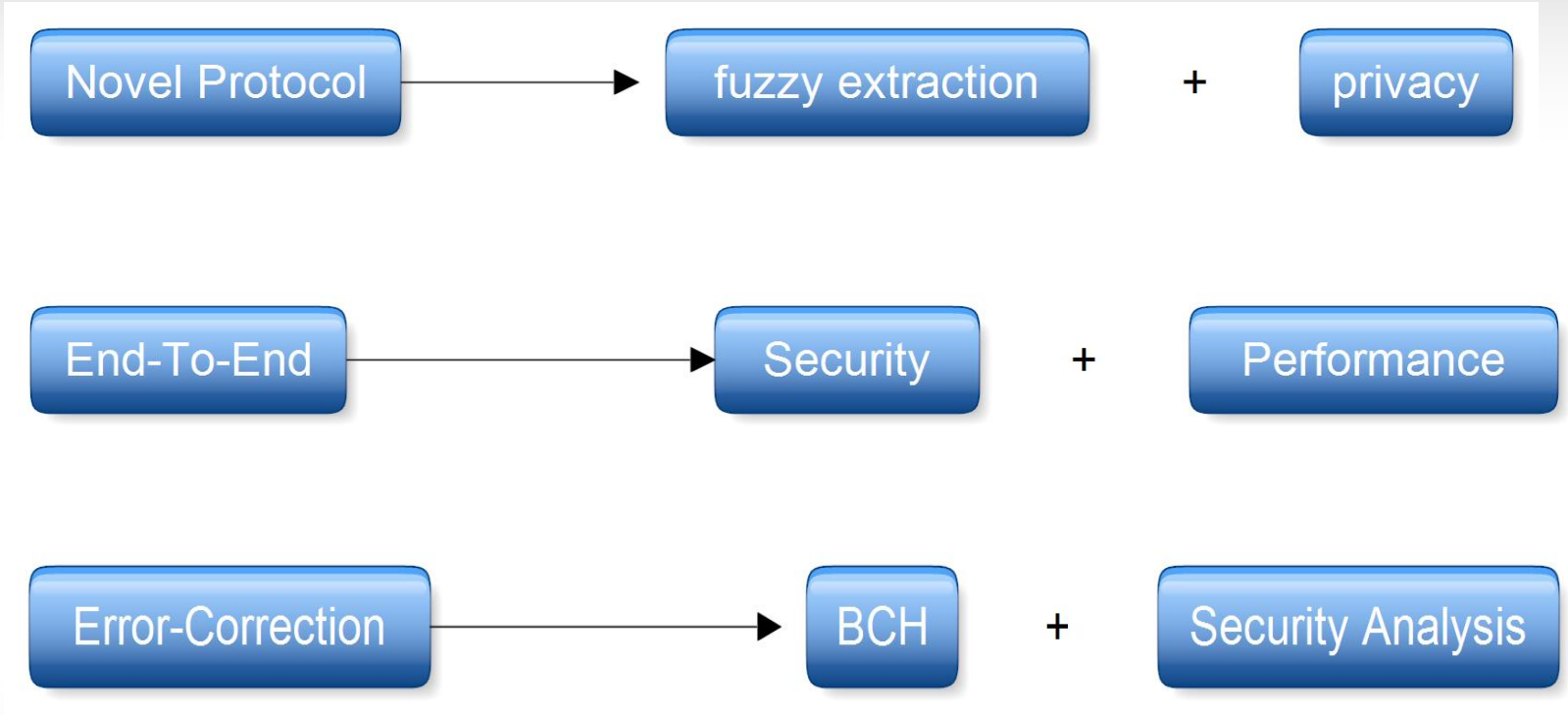
Outline

- Introduction
- Notations
- Protocol
- Design of SRAM-PUF/TRNG
- Design of Fuzzy Extractor
- Architecture Design
- Conclusion

Introduction

- Prototype Implementation of a provably secure protocol
- Support privacy-preserving mutual authentication
- Based on PUF
- Implemented on SASEBO-GII using onboard SRAM

Innovative Features



Notations

- **Truly Random Number Generator**

  TRNG derives a truly random number sequence.

- **Physically Unclonable Functions**

  $f: K \times D \rightarrow R$ which takes as input a physical characteristic $x \in K$ and message $y \in D$ and outputs $z \in R$

Notations

- **Symmetric Key Encryption**

  $SKE := (SKE.Enc; SKE.Dec)$ denotes the symmetric key encryption. $SKE.Enc$ takes as input secret key sk and plaintext m and outputs ciphertext c. $SKE.Dec$ decrypts the ciphertext c using the same secret key sk to generate plaintext m

- **Pseudorandom Function**

$PRF, PRF' : K' \times D' \rightarrow R'$ takes as input secret key $sk \in K'$ and message $m \in D'$ and provides an output which is indistinguishable from random.

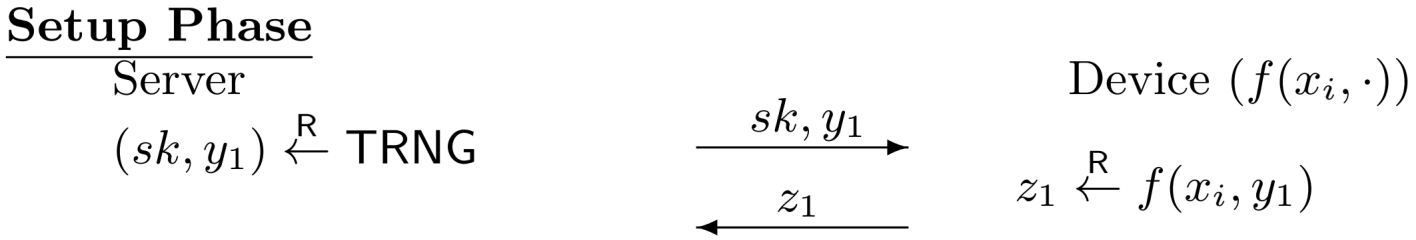
Notations

- **Fuzzy Extractor**

$FE := (FE.Gen, FE.\mathrm{Rec})$ denotes a fuzzy extractor. The $FE.Gen$ algorithm takes as input a variable $\underline{\mathbf{z}}$ and outputs randomness $\underline{\mathbf{r}}$ and helper data $\mathbf{hd}$.

The $FE.\mathrm{Rec}$ algorithm recovers $\mathbf{r}$ with input variable $\underline{\mathbf{z}'}$ and $\underline{\mathbf{hd}}$.

Protocol
Setup Phase



Protocol
Authentication Phase

Authentication Phase

Server $\{(z_1, sk, z_{old}, sk_{old})\}_i$		Device $(f(x_i, \cdot), sk, y_1)$
$y_1' \xleftarrow{R} \mathsf{TRNG}$	$\xrightarrow{y_1'}$	$z_1' \xleftarrow{R} f(x_i, y_1)$
		$(r_1, hd) \xleftarrow{R} \mathsf{FE.Gen}(z_1')$
		$c := \mathsf{SKE.Enc}(sk, hd)$
		$y_2' \xleftarrow{R} \mathsf{TRNG}$
		$(t_1, \ldots, t_5) := \mathsf{PRF}(r_1, y_1' \| y_2')$
		$y_2 \xleftarrow{R} \mathsf{TRNG}$
		$z_2' \xleftarrow{R} f(x_i, y_2)$
		$u_1 := z_2' \oplus t_2$
$hd := \mathsf{SKE.Dec}(sk, c)$	$\xleftarrow{c, y_2', t_1, u_1, v_1}$	$v_1 := \mathsf{PRF}'(t_3, c \| u_1)$

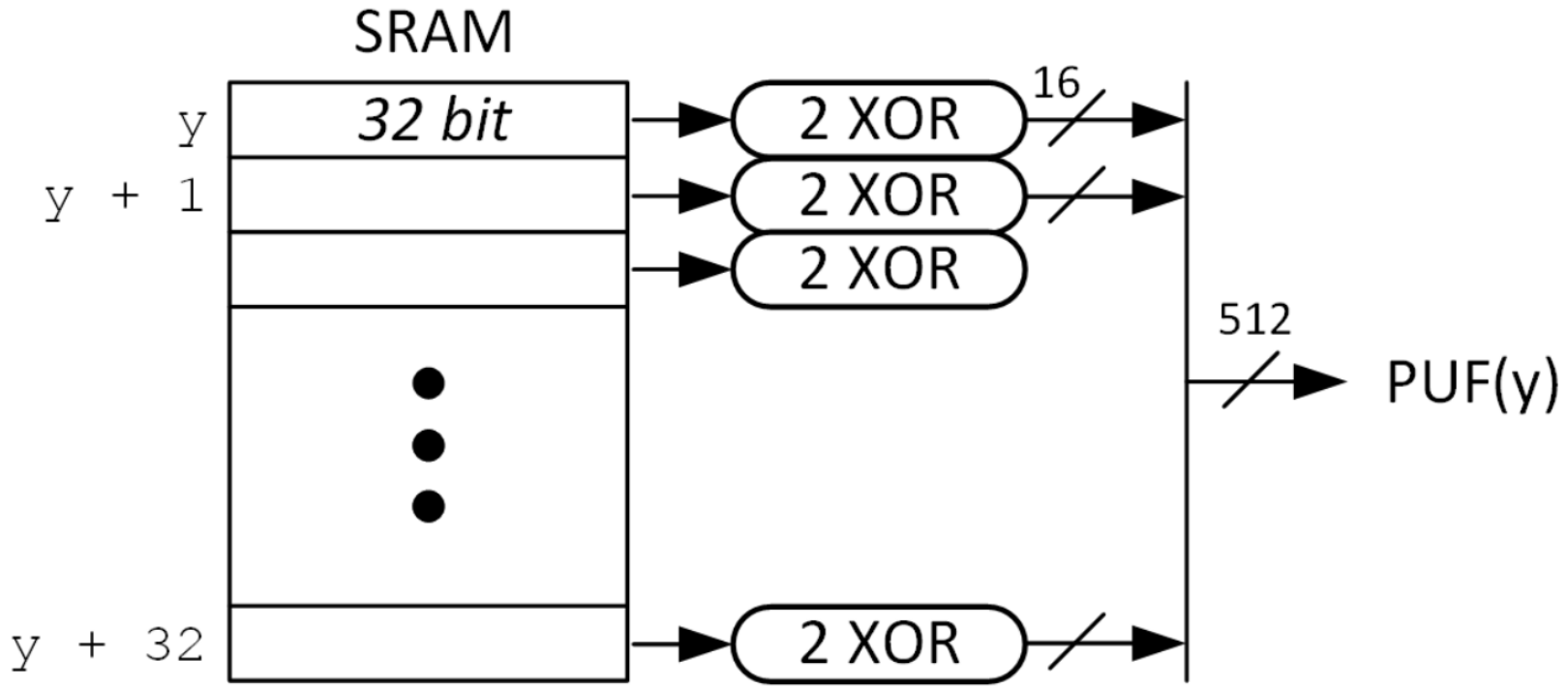
Protocol Authentication Phase

$r_1 := \mathsf{FE.Rec}(z_1, hd)$
$(t'_1, \ldots, t'_5) := \mathsf{PRF}(r_1, y'_1 \| y'_2)$
If $t'_1 = t_1$ in $1 \leq i \leq \mathsf{num}$,
  If $v_1 = \mathsf{PRF}'(t'_3, c \| u_1)$,
    $z'_2 := u_1 \oplus t'_2$
    Update $(z_1, sk, z_{old}, sk_{old})$ to
      (z'_2, t_5, z_1, sk)
Else, $hd_1 := \mathsf{SKE.Dec}(sk_{old}, c)$
$r_1 := \mathsf{FE.Rec}(z_{old}, hd_1)$
$(t'_1, \ldots, t'_5) := \mathcal{G}(r_1, y'_1 \| y'_2)$
$\vdots$
Else, $t'_4 \xleftarrow{\mathsf{R}} \mathsf{TRNG}$

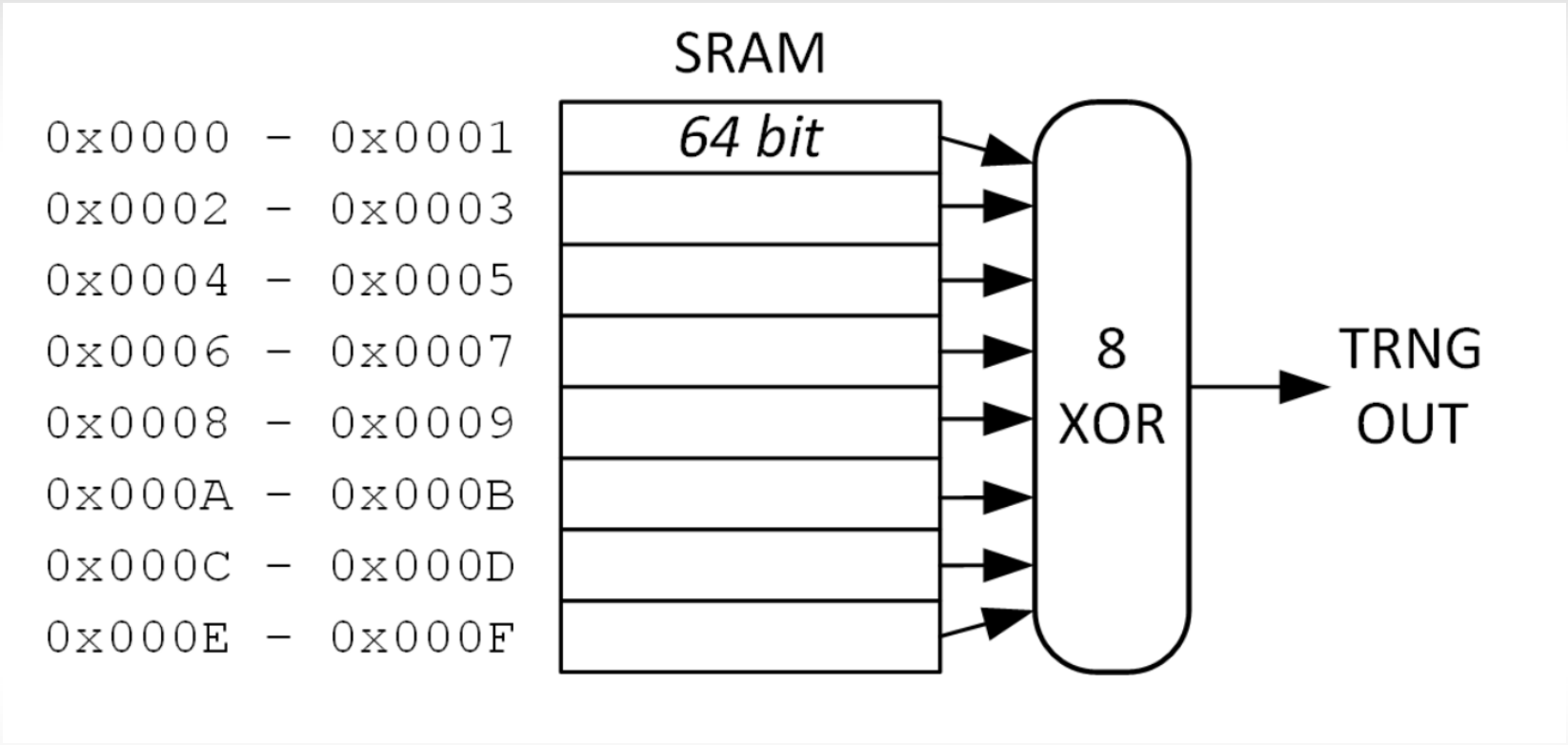
$\xrightarrow{t'_4}$

If $t'_4 = t_4$,
  $(y_1, sk) := (y_2, t_5)$

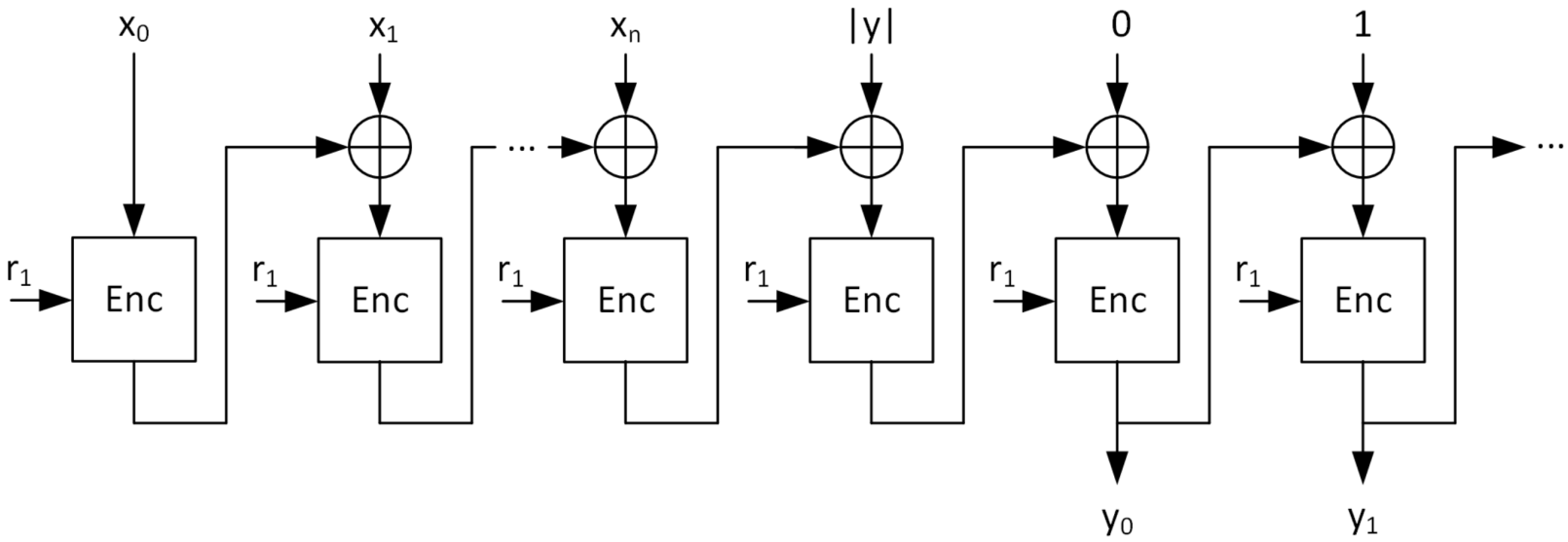
SRAM - PUF



SRAM - TRNG


SRAM
0x0000 – 0x0001
0x0002 – 0x0003
0x0004 – 0x0005
0x0006 – 0x0007
0x0008 – 0x0009
0x000A – 0x000B
0x000C – 0x000D
0x000E – 0x000F
64 bit
8
XOR
TRNG
OUT

PRF based on CBC



Fuzzy Extractor Error Correction

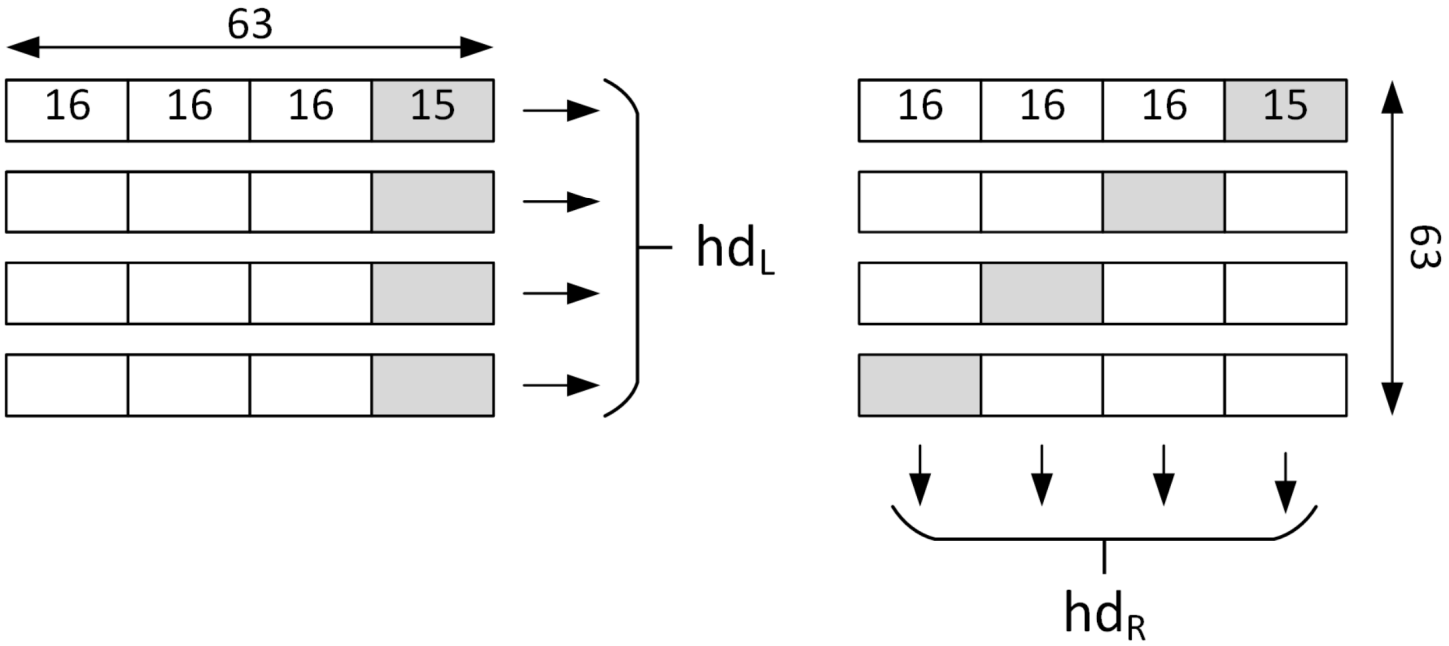
$\mathsf{Encode}(a)$: $\delta \xleftarrow{\mathsf{R}} \mathsf{TRNG} \in \{0,1\}^{k_1}, cw := \mathsf{BCH.Gen}(\delta) \in \{0,1\}^{n_1}, hd := a \oplus cw$

$z_1' \xleftarrow{\mathsf{R}} f(x_i, y_1)$
$(r_1, hd) \xleftarrow{\mathsf{R}} \mathsf{FE.Gen}(z_1')$

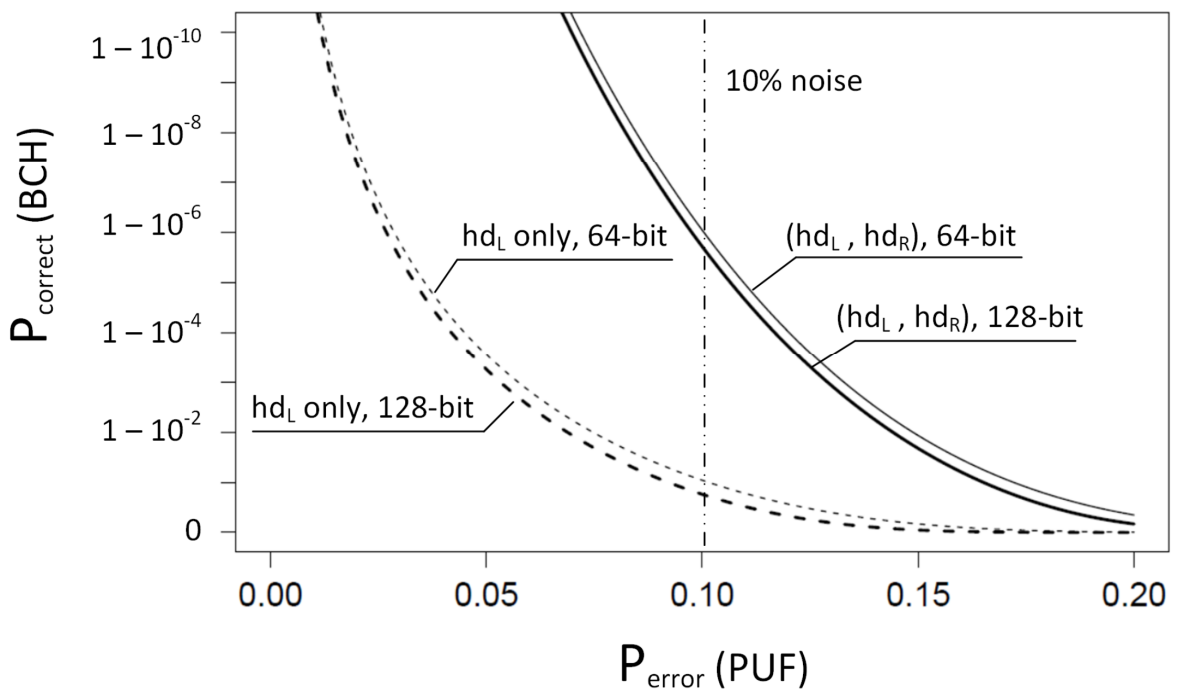
$\mathsf{Decode}(a', hd)$: $cw' := a' \oplus hd, cw := \mathsf{BCH.Dec}(cw'), a := cw \oplus hd$

$hd := \mathsf{SKE.Dec}(sk, c)$
$r_1 := \mathsf{FE.Rec}(z_1, hd)$

Helper Data



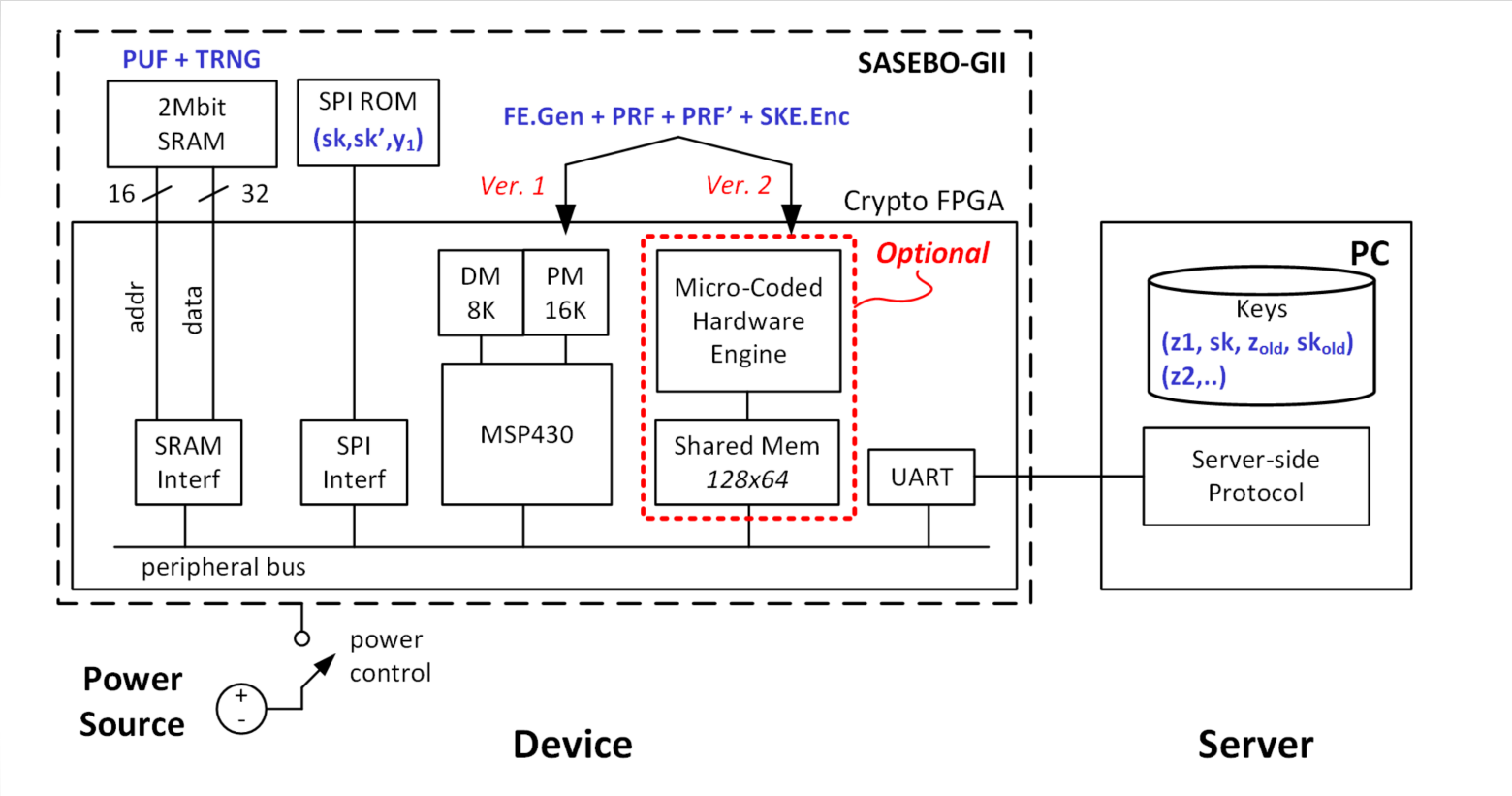
Faulty PUF Output



Key Length and Data Sizes

Category	**Purpose**	**Variables**	**64-bit security**	**128-bit security**
Setup Phase	Input address	y_1	12	12
	PUF's output	z_1	252	504
	Stored key	sk, sk'	64	128
Authentication Phase	PUF's output	z'_1, z'_2	252	504
	Nonce	y'_1, y'_2	64	128
	Randonmess for FE	δ, rnd	128	256
	Secret key for PRF	r_1	64	128
	Helper data	hd (includes rnd)	632	1,264
	Ciphertext	c	640	1,280
	PUF's input	y_2	12	12
	Mutual authentication	t_1, t_4	64	128
	XORed element	t_2	252	504
	Secret key for PRF$'$ and MAC	t_3, s_1	64	128
	Updated stored key	t_5	128	256
Communication	First message (from server)	y'_1	64	128
	Second message (from device)	(c, y'_2, t_1, u_1, s_1)	1,084	2,168
	Third message (from server)	t'_4	64	128
Memory	Persistent State (NVM)	(sk, sk', y_1)	140	268
	SRAM area for PUF		504	1,008
	SRAM area for RNG		2,656	5,216

System Architecture of the Device and Server



Data Flows

Seq	**Authentication Step**	MSP430 *Fig. 6 Ver. 1*	MSP430 + HW Engine *Fig. 6 Ver. 2*
1	Receive y_1'	`UART.Receive` $\rightarrow$ `MSP430.DM`	`UART.Receive` $\rightarrow$ `MSP430.DM`
2	Read sk, sk', y_1	`SPIROM.Read` $\rightarrow$ `MSP430.DM`	`SPIROM.Read` $\rightarrow$ `MSP430.DM`
3	$z_1' \xleftarrow{R} f(x_i, y_1)$	`SRAM.PUF` $\rightarrow$ `MSP430.DM`	`SRAM.PUF` $\rightarrow$ `MSP430.DM`
4	$(r_1, hd_1) \xleftarrow{R} \mathsf{FE.Gen}(z_1')$	`MS430.run(PRF)` `MS430.run(BCH.Enc)`	
5	$m_2 \xleftarrow{R} \mathsf{TRNG}$	`SRAM.TRNG` $\rightarrow$ `MSP430.DM`	`SRAM.TRNG` $\rightarrow$ `MSP430.DM`
6	$y_2 \xleftarrow{R} \mathsf{TRNG}$ $(t_1, \ldots, t_5) := \mathsf{PRF}(r_1, y_1' \| y_2')$	`MS430.run(PRF)`	
7	$c := \mathsf{SKE.Enc}(sk, hd_1)$	`MS430.run(Enc)`	
8	$z_2' \xleftarrow{R} f(x_i, y_2)$	`SRAM.PUF` $\rightarrow$ `MSP430.DM`	`SRAM.PUF` $\rightarrow$ `MSP430.DM`
9	$u_1 := z_2' \oplus t_2$	`MSP430.run(xor)`	
10	$v_1 := \mathsf{PRF}'(t_3, c \| u_1)$	`MS430.run(PRF)`	

11	*HW Execution Step*		`MSP430.DM` $\rightarrow$ `HW.SharedMem`
			`HW.run`
			`HW.SharedMem` $\rightarrow$ `MS430.DM`

12	Send c, m_2, t_1, u_1, v_1	`MSP430.DM` $\rightarrow$ `UART.Send`	`MSP430.DM` $\rightarrow$ `UART.Send`
13	Receive t_4'	`UART.Receive` $\rightarrow$ `MSP430.DM`	`UART.Receive` $\rightarrow$ `MSP430.DM`
14	Write y_2, t_5	`MSP430.DM` $\rightarrow$ `SPIROM.Write`	`MSP430.DM`$\rightarrow$ `SPIROM.Write`

Conclusion

Reference	**PUFKY** [27]	**Slender** [29]	**Reverse-FE** [38]	**This work**
Operation	Key generation	Protocol	Protocol	Protocol
Privacy	No	No	No	Yes
Security flaws	No	Major [10]	Minor [10]	No
Implemented Parties	N/A	Device	Device	Device, Server
Communication Interface	Yes: Bus	No	No	Yes: Bus, UART
Flexibility	Low	Low	Low	High
Reconfiguration Method	Redesign Hardware	Redesign Hardware	Redesign Hardware	Modify Software, Update Microcode
Demonstrator	FPGA	FPGA	FPGA	FPGA + PC
Security-level	128-bits	128-bits	128-bits	64,128-bits
Execution time (clock cycles)	55,310	-	-	18,597
Logic Cost (w/o PUF)	210 Slices	144 LUT, 274 Register	658 LUT, 496 Register	1221 LUT, 441 Register
PUF-type	Strong-PUF	Strong-PUF	-	Weak-PUF
PUF-instance	RO-PUF	XOR-Arbiter	-	SRAM
Hardware Platform	XC6SLX45	XC5VLX110T	XC5VLX50	XC5VLX30


Thank you!