RSAConference2020

San Francisco | February 24 – 28 | Moscone Center

HUMAN ELEMENT

SESSION ID: CRYP-R02

Efficient FPGA Implementations of LowMC and Picnic

Roman Walch

PhD Student
IAIK / Know-Center GmbH, Graz University of Technology
@rw0x0
Joint work with: Daniel Kales, Sebastian Ramacher, Christian Rechberger and Mario Werner

Post-Quantum Digital Signatures

- Shor‘s algorithm for factoring and discrete logarithm
- Quantum computer breaks:
  - Most asymmetric cryptography
  - **RSA**, **DSA**, **ECDSA, …**
- NIST Standardization Project for PQ Signatures
  - Currently second round
  - **Picnic** [Cha+17; Cha+19] (using **LowMC** [Alb+15])
  - Performance optimized implementations required

Contribution

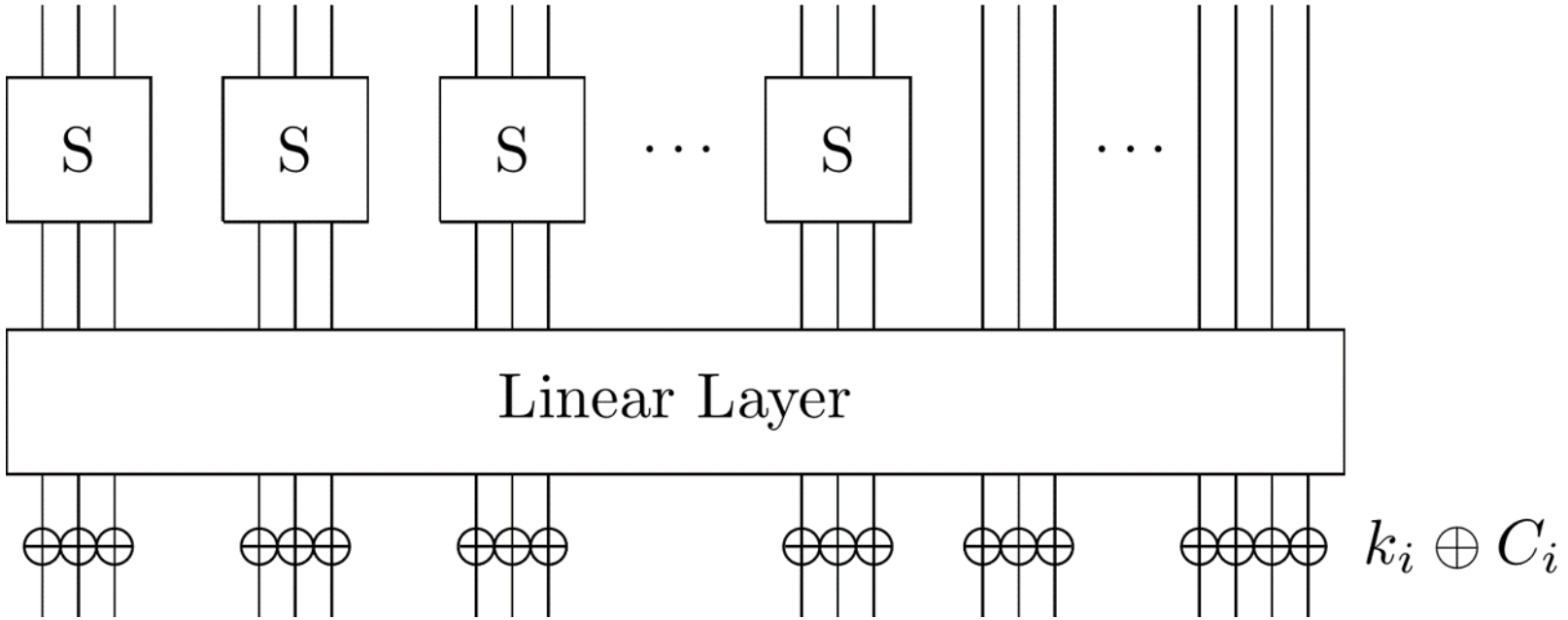
- First efficient VHDL implementation of **LowMC**
- First VHDL implementation of **Picnic**
  - **Picnic1-L1-FS:** 128 (64) bit security (PQ)
  - **Picnic1-L5-FS:** 256 (128) bit security (PQ)
- Coprocessors accessible via PCIe interface

RSAConference2020

The LowMC Block Cipher

LowMC – Round

- Substitution-Permutation Network (SPN) with reduced SboxLayer:



LowMC – Details

- Designed to minimize AND gates (3 ANDs / Sbox)
  - $S(a,b,c) = (a \oplus (b \wedge c), a \oplus b \oplus (a \wedge c), a \oplus b \oplus c \oplus (a \wedge b))$
- Linear Layer:
  - State multiplied with matrix over GF(2)
  - $n \times n$ matrix per round
- Roundkey schedule
  - Key multiplied with matrix over GF(2)
  - $n \times k$ matrix per round + inital key whitening

n … blocksize
k … keysize

LowMC – Constants per Instance

- Naive implementaion:
  - L1: ~82 KiB
  - L5: ~617 KiB

- Impact on hardware utilization

nr.	LowMC				without opt.	
	n	k	m	r	LUTs	% LUTs
L1	128	128	10	20	42 395	**20.80%**
L5	256	256	10	38	209 348	**102.72%**

LowMC – Constants per Instance

- Naive implementaion:
  - L1: ~82 KiB
  - L5: ~617 KiB
- Optimizations by [Din+19]:
  - L1: ~29 KiB
  - L5: ~117 KiB

- Impact on hardware utilization

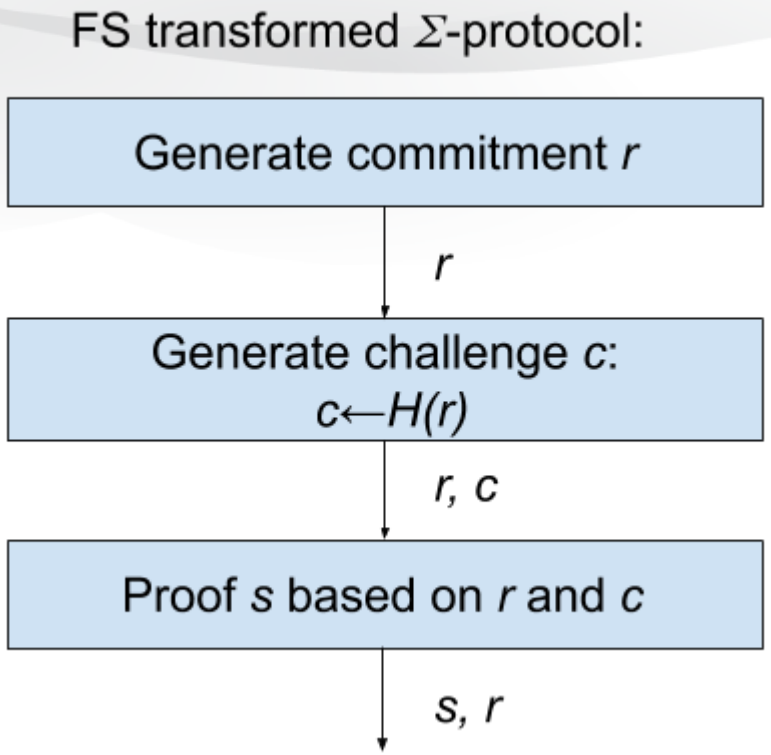
nr.	LowMC				without opt.		with opt.		Improv.
	n	k	m	r	LUTs	% LUTs	LUTs	% LUTs	%
L1	128	128	10	20	42 395	**20.80%**	13 558	**6.65%**	68.02%
L5	256	256	10	38	209 348	**102.72%**	44 431	**21.8 %**	78.78%

RSAConference2020

The Picnic Signature Scheme

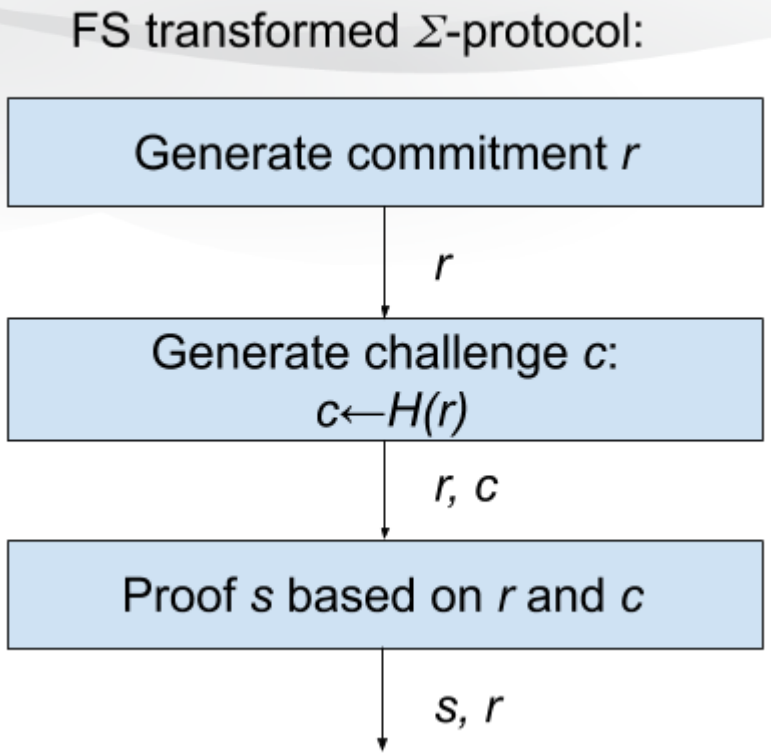
Picnic – Building Blocks

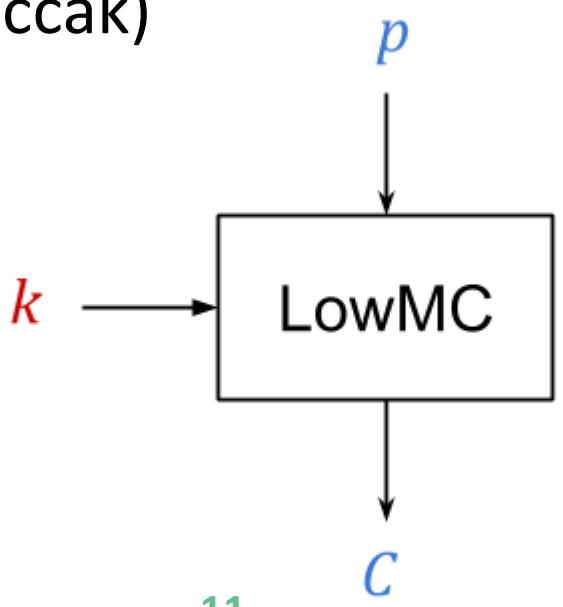
- FS transformed Σ-protocol
- Σ-protocol: **ZKB++** or **KKW**



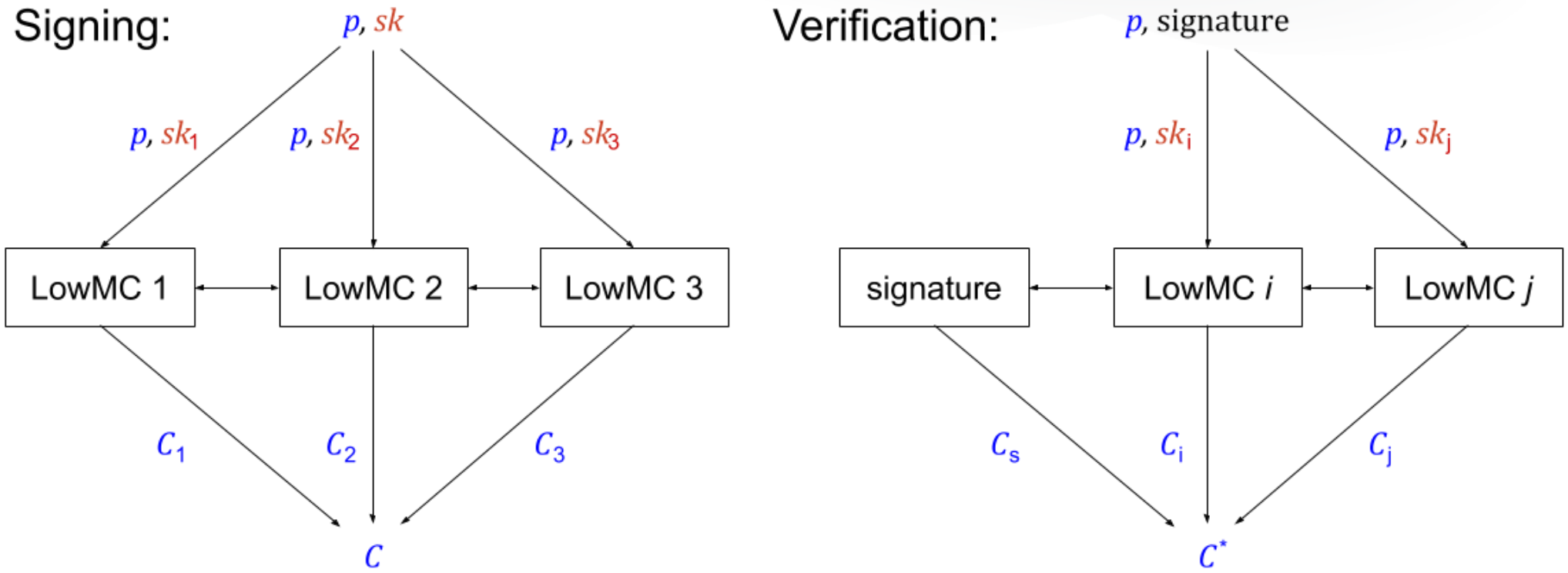
Picnic – Building Blocks

- FS transformed Σ-protocol
- Σ-protocol: ZKB++ or **KKW**
- Proof system:
  - Multi-party computation (MPC) of **LowMC**
  - Random oracle: **SHAKE** (Keccak)
- Keys:
  - Public Key: $pk = (C, p)$
  - Secret Key: $sk = k$





Picnic – Proof System



- Communication per AND gate
- Publish 2 players in signature (based on challenge)

Picnic – MPC contd.

- MPC repeated T times
  - Reduce probability to cheat
  - **Picnic1-L1-FS:** $T = 219$
  - **Picnic1–L5–FS:** $T = 438$
- **Picnic** signature:
  - Challenge
  - Published Players (based on challenge)
  - MPC Communication (**LowMC** vs. **AES**)

Picnic – MPC Implementation

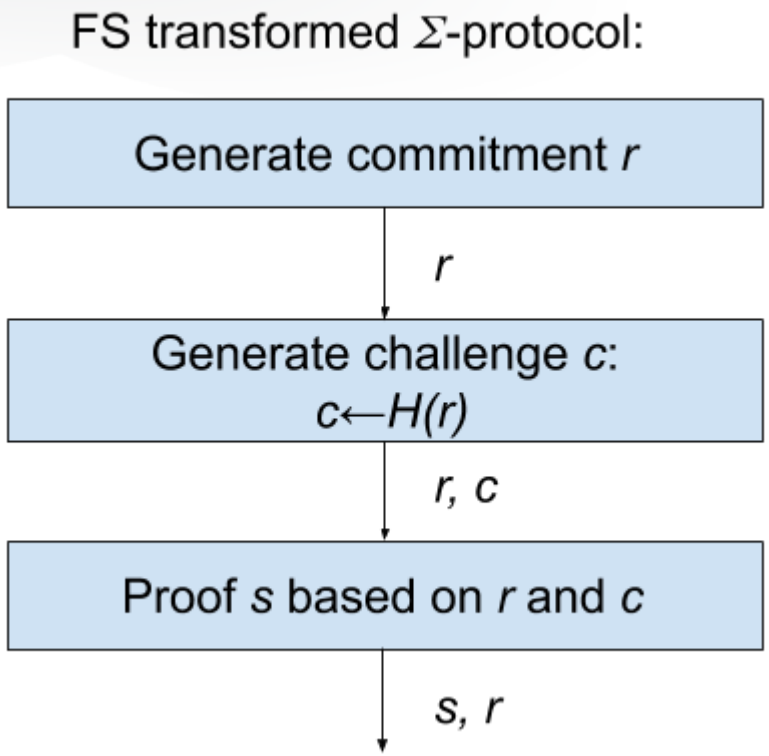
- Optimized for speed:
  - 3 players calculated in parallel
- Further improvement
  - Precomputation of one share
  - Only 2 **LowMC** instances on FPGA
- Sign / Verify use same LUTs for matrices

Picnic – Other Submodulues

- Pseudorandomness for MPC
- Commitments
  - MPC Players commit to results
- Challenge creation (Random Oracle)

⇒ All using **SHAKE**

  - … different configurations



Picnic – Implementation

- Custom **SHAKE** implementation
- 3 players parallel per MPC run t
- BRAM for intermediate values
  - ~ 400 KiB for **Picnic1-L5-FS**
- **Picnic1-L1-FS** and **Picnic1-L5-FS** implementations for
  - Sign / Verify only
  - Sign and Verify combined

RSAConference2020

Practical Evaluation

FPGA and PCIe

- Xilinx Kintex-7 FPGA KC705 Evaluation Kit
- PCIe Wrapper
  - Manages FPGA/PC interface
- Developed C-Library for PC/FPGA communication

Hardware Utilization

- Lookup tables (LUTs) and BRAM utilization (% available)

Design Part	LUTs	%	BRAM	%
Picnic1-L1	90 037	**44.18 %**	52.5	11.80 %
Picnic1-L1-Sign	76 472	**37.52 %**	52.5	11.80 %
Picnic1-L1-Verify	68 614	**33.67 %**	33.5	7.53 %
Picnic1-L5	167 530	**82.20 %**	98.5	22.13 %
Picnic1-L5-Sign	149 456	**73.33 %**	98.5	22.13 %
Picnic1-L5-Verify	138 547	**67.98 %**	62.5	14.04 %
PCIe Wrapper	22 216	**10.90 %**	42.5	9.55 %

Runtime Comparison

- Software platform:
  - Ubuntu 18.04.1, GCC 7.3.0, 16 GB RAM
  - CPU: Intel i7-4790, 3.6 GHz

Coprocessor	clock frequency	clock cycles	FPGA runtime	C-Access runtime	Software	
					SIMD	No SIMD
	MHz	k cycles	ms	ms	ms	ms
Picnic1-L1-Sign	125	~31.3	**0.25**	**0.35**	1.44	2.82
Picnic1-L1-Verify	125	~29.6	**0.24**	**0.40**	1.15	2.34
Picnic1-L5-Sign	125	~154.5	**1.24**	**1.38**	5.87	12.37
Picnic1-L5-Verify	125	~146.6	**1.17**	**2.13**	4.92	10.59

Comparison of FPGA implementations

Scheme	Security		FPGA	Area			f	t
	Classic	PQ		LUT	FF	BRAM	MHz	ms
Picnic1-L1-FS	**128**	**64**	**K7**	**90 037**	**23 105**	**52.5**	**125**	**0.25**
SPHINCS+-128	128	64	V7	11 438	3 335	?	100	9.38
Picnic1-L5-FS	**256**	**128**	**K7**	**167 530**	**33 164**	**98.5**	**125**	**1.24**
SPHINCS-256	256	128	K7	19 067	38 132	36	525	1.53
ECDSA-256	128	X	V7	6 816	4 442	0	225	1.49
ECDSA-256	128	X	V4	34 869	32 430	176	375	0.04
RSA-2048	112	X	V7	3 558 slices		0	399	5.68

Reducing LUT Utilization

- Implementation is optimized for speed
- **LowMC** matrices encoded in LUTs
  - 1 multiplication per clock cycle
  - High LUT utilization
- Reduce LUT utilization:
  - Store **LowMC** matrices in BRAM

  ... reduces performance
  - **LowMC** same matrix each round?
  - Alternatives to **LowMC**?

Conclusion

- First efficient VHDL implementation **LowMC**
- First VHDL implementation of **Picnic**
  - **Picnic1-L1-FS** and **Picnic1-L5-FS**
  - Extended to FPGA-based coprocessor (PCIe Interface)
- Good runtime
  - Trade off with high hardware utilization

RSAConference2020

Efficient FPGA Implementations of LowMC and Picnic

Questions?

Bibliography I

[Alb+15] Martin R. Albrecht, Christian Rechberger, Thomas Schneider, Tyge Tiessen, and Michael Zohner. **Ciphers for MPC and FHE.** EUROCRYPT (1). Vol. 9056. LNCS. Springer, 2015, pp. 430–454.

[Cha19] André Chailloux. **Quantum security of the fiat-shamir transform of commit and open protocols.** ePrint, 2019:699, 2019.

[Cha+17] Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, and Greg Zaverucha. **Post-Quantum Zero-Knowledge and Signatures from Symmetric-Key Primitives.** ACM CCS. ACM, 2017, pp. 1825-1842.

Bibliography II

[Cha+19] Melissa Chase et al. **The Picnic Signature Scheme Design Document (version 2).** 2019. URL: https://github.com/microsoft/Picnic/blob/master/spec/design-v2.0.pdf.

[Din+19] Itai Dinur, Daniel Kales, Angela Promitzer, Sebastian Ramacher, and Christian Rechberger. **Linear Equivalence of Block Ciphers with Partial Non-Linear Layers: Application to LowMC.** EUROCRYPT (1). Vol. 11476. LNCS. Springer, 2019, pp. 343–372.