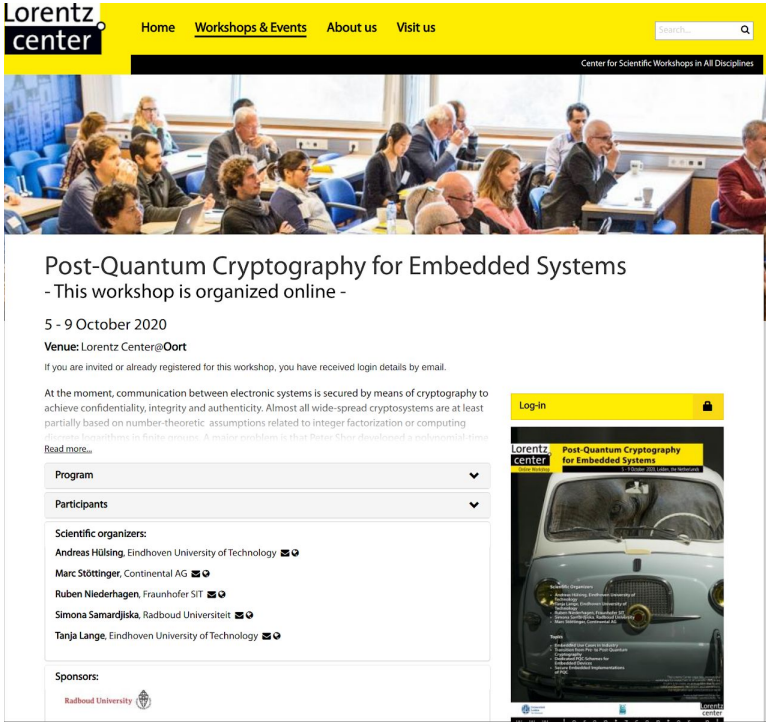
Verifying Post-Quantum Signatures in 8 kB of RAM

Ruben Gonzalez[1], Andreas Hülsing[2], Matthias J. Kannwischer[3], Juliane Krämer[4], Tanja Lange[2], Marc Stöttinger[5], Elisabeth Waitz[6], Thom Wiggers[7], and Bo-Yin Yang[8]

PQCrypto 2021, 20. - 22. July 2021
Presenter: Ruben Anthony Gonzalez

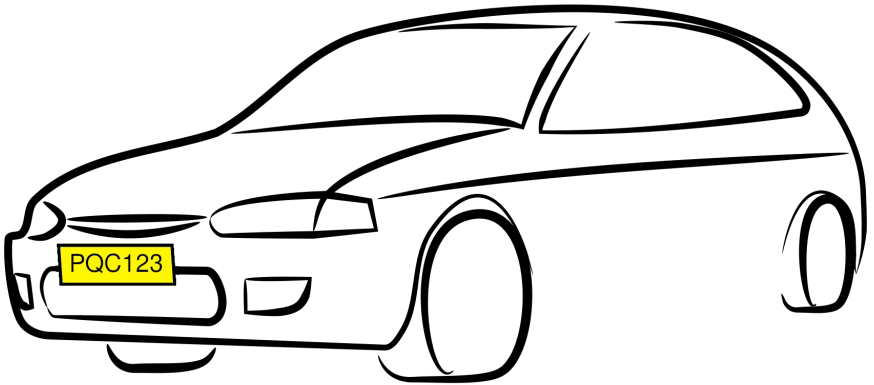
Background

- We're at Round 3
  - Let's look at the real world
- PQC for Embedded Systems Workshop
  - Bringing together industry and academia



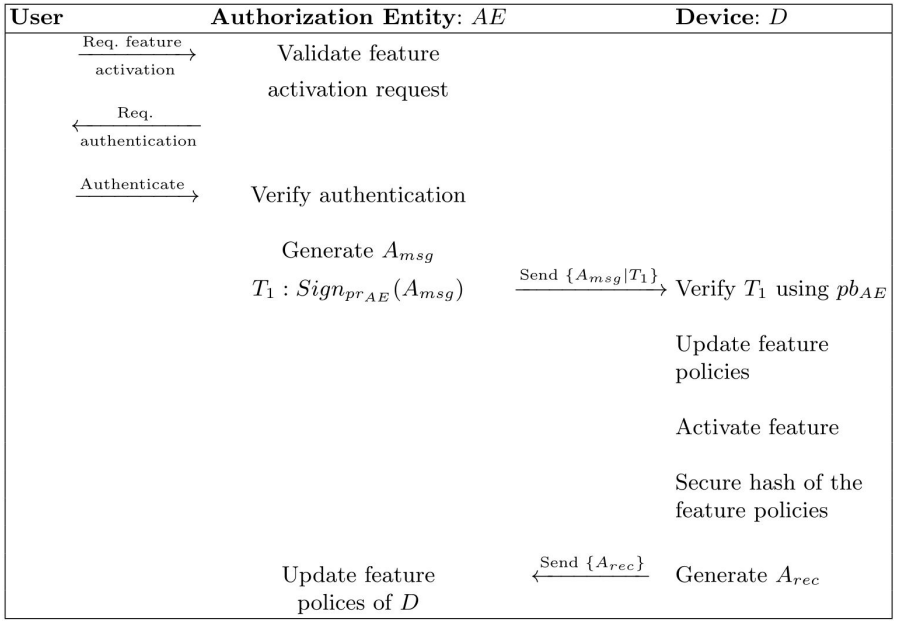
Use Case

- **Feature Activation in Cars**
  - Short signed messages



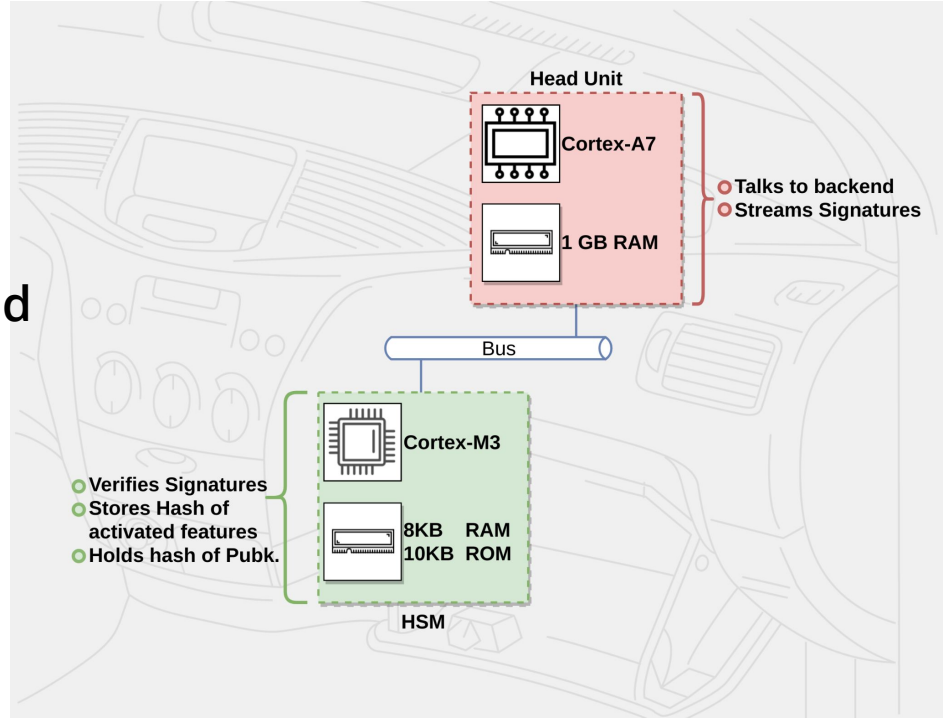
Use Case

- **Feature Activation in Cars**
  - Short signed messages
- **Protocol already exists**
  - Uses ECC



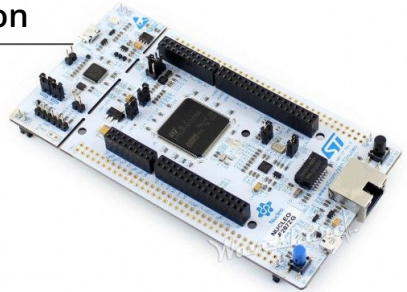
Use Case

- Feature Activation in Cars
  - Short signed messages
- Protocol already exists
  - Uses ECC
- HSM has to verify signatures and Pubkey
  - Is resource constrained
  - Holds hash of public key
  - Stores activated features in secure memory
  - Simulated on a STM32-F207ZG



Experimental Setup

Serial Communication



Host Computer

STM32-F207ZG

↪ Cortex M3

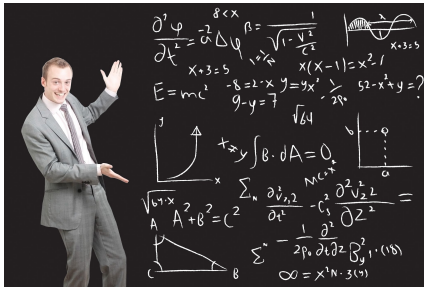
Investigated Schemes

SPHINCS+

Hash Based

GeMSS

Rainbow



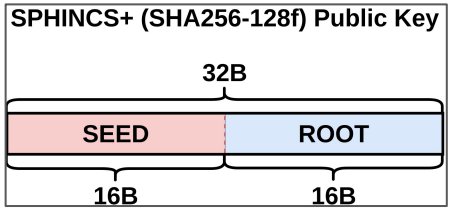
Multivariate

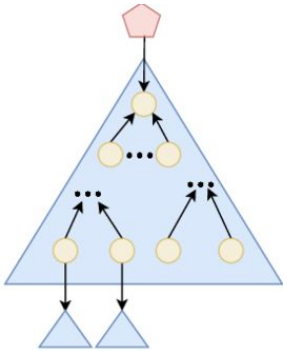
Dilithium

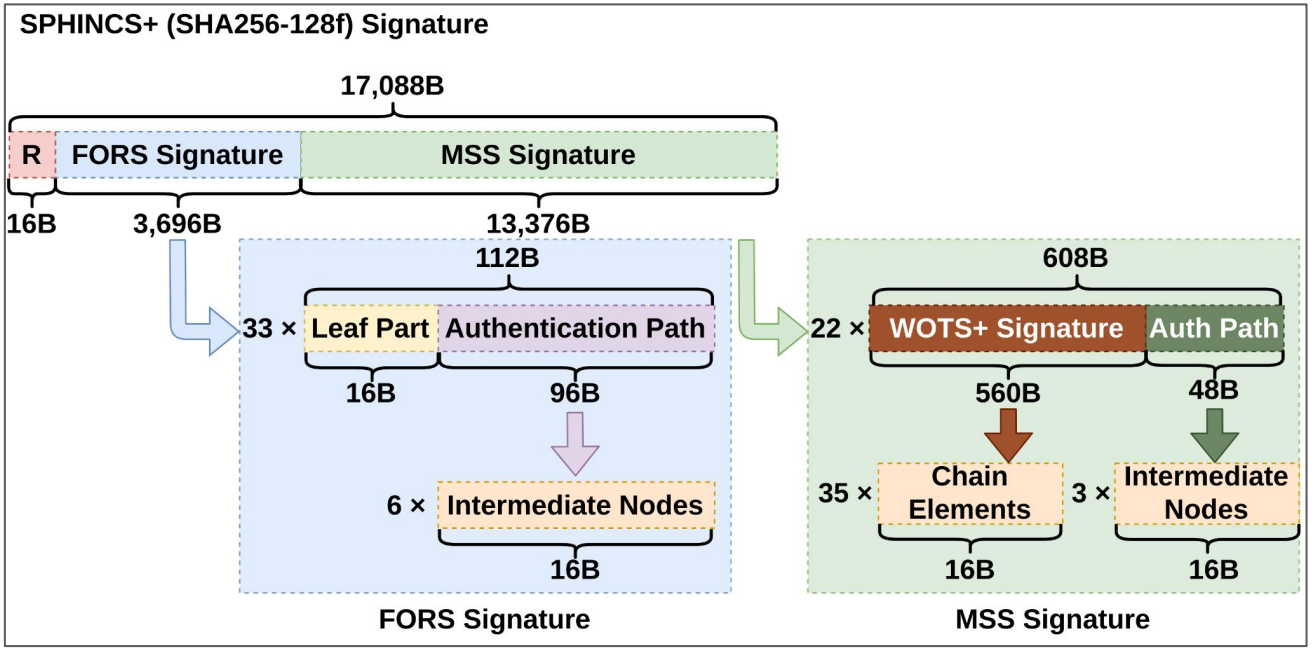
Falcon

Lattice Based

SPHINCS+ (SHA256-128)

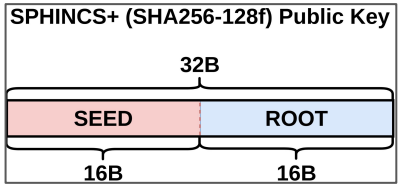


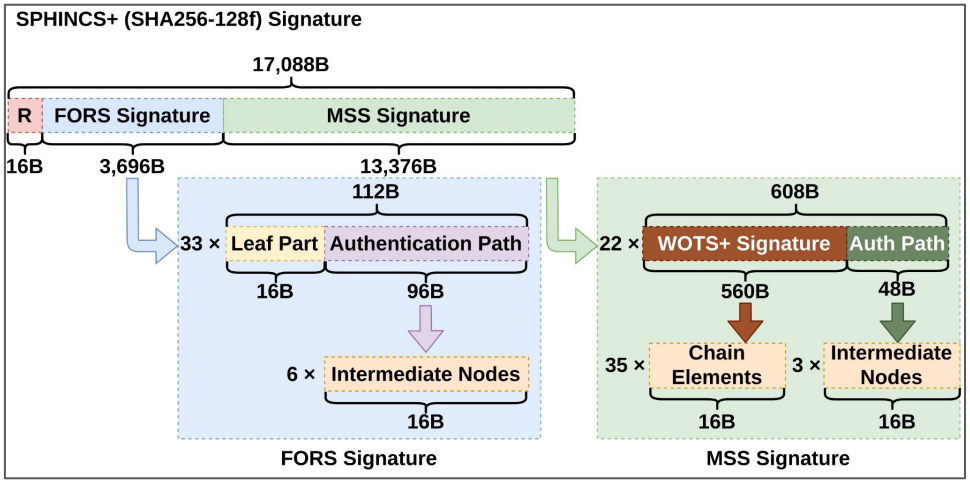




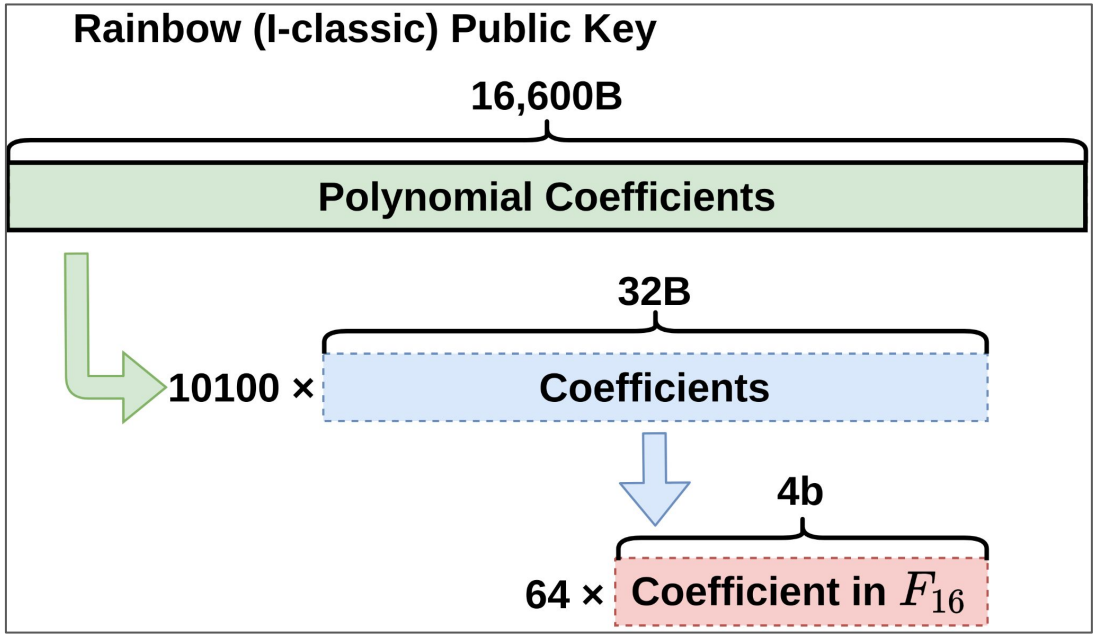
SPHINCS+ (SHA256-128)

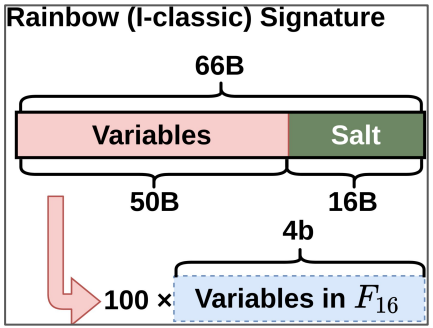
- Hold **pubkey in memory**
- **Stream signature**
  - in chunks of n × 16B
- Used reference code
  - Majority of time is spend in hash function



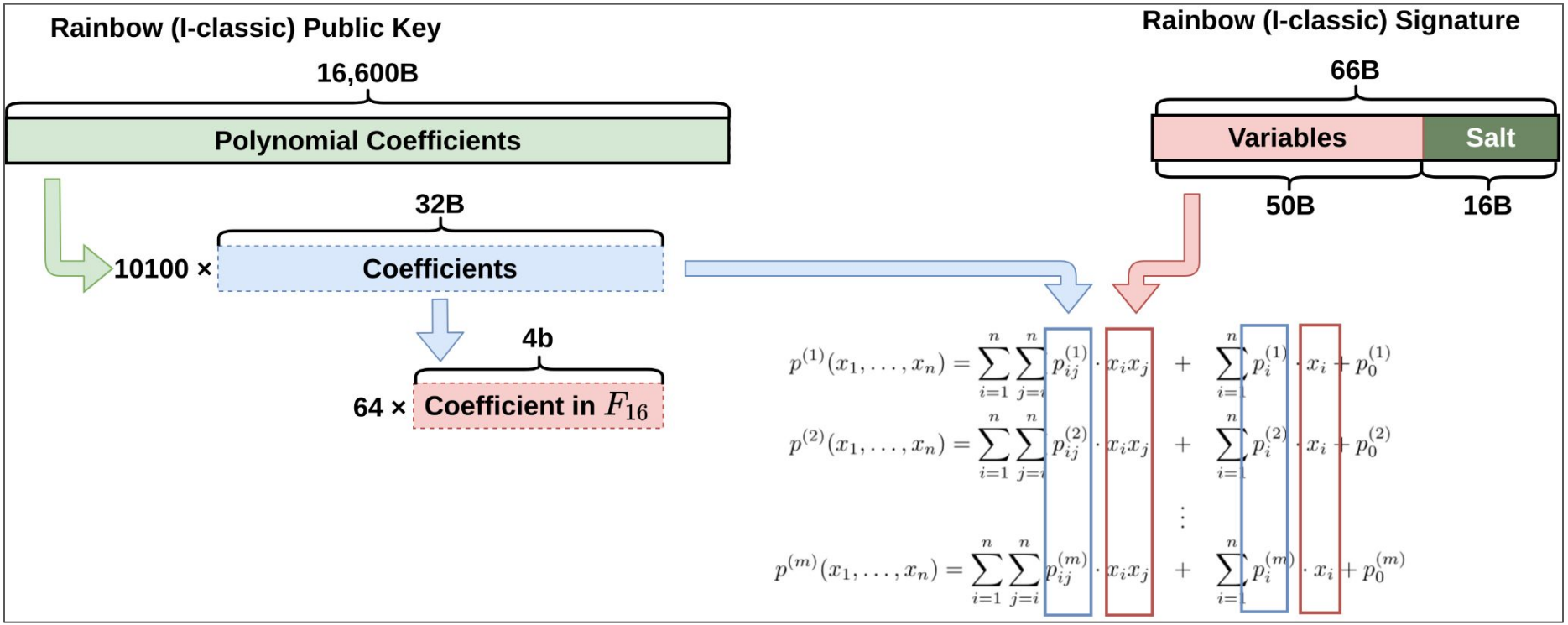


Rainbow (I-classic)



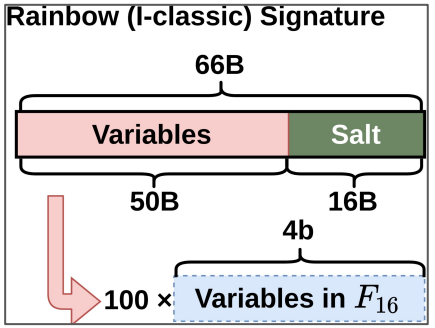


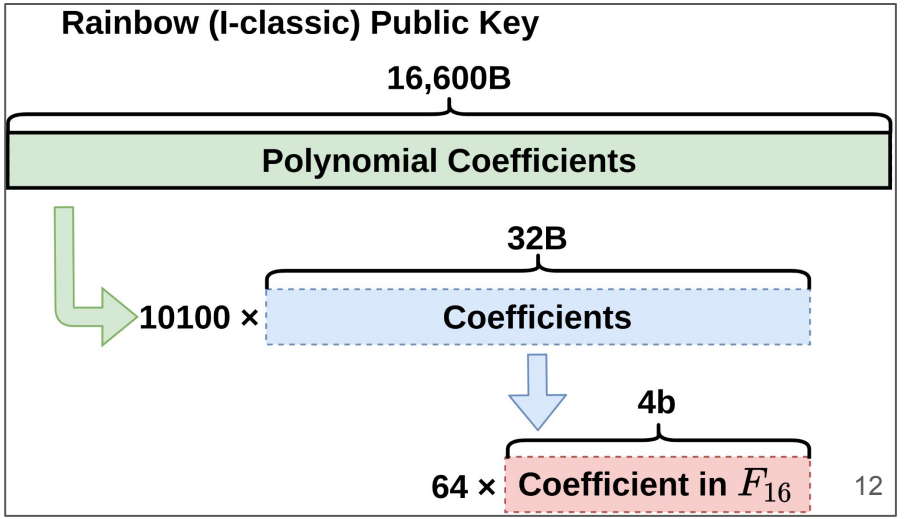
Rainbow (I-classic)



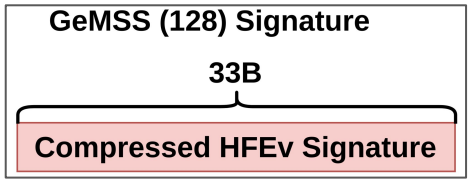
Rainbow (I-classic)

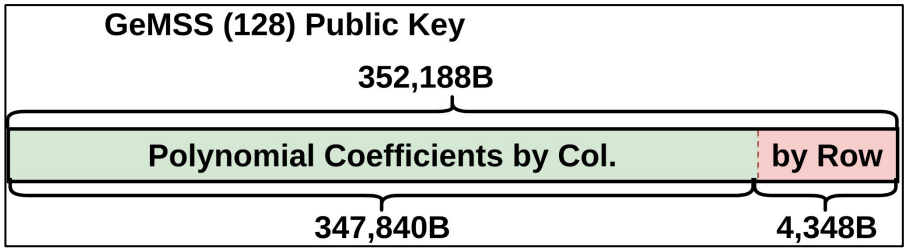
- Hold **signature in memory**
- **Stream public key**
  - In chunks of n × 32B
- Based on bitslice implementation





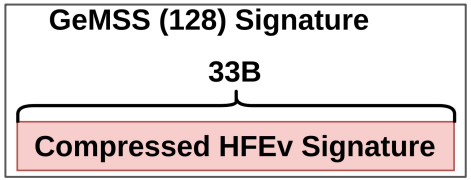
GeMSS (128)

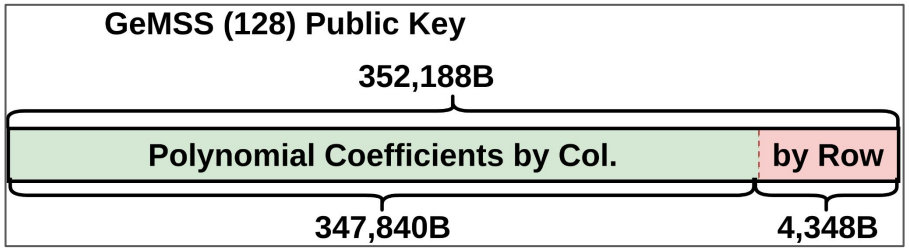

GeMSS (128) Signature
33B
Compressed HFEv Signature


GeMSS (128) Public Key
352,188B
Polynomial Coefficients by Col.
by Row
347,840B
4,348B

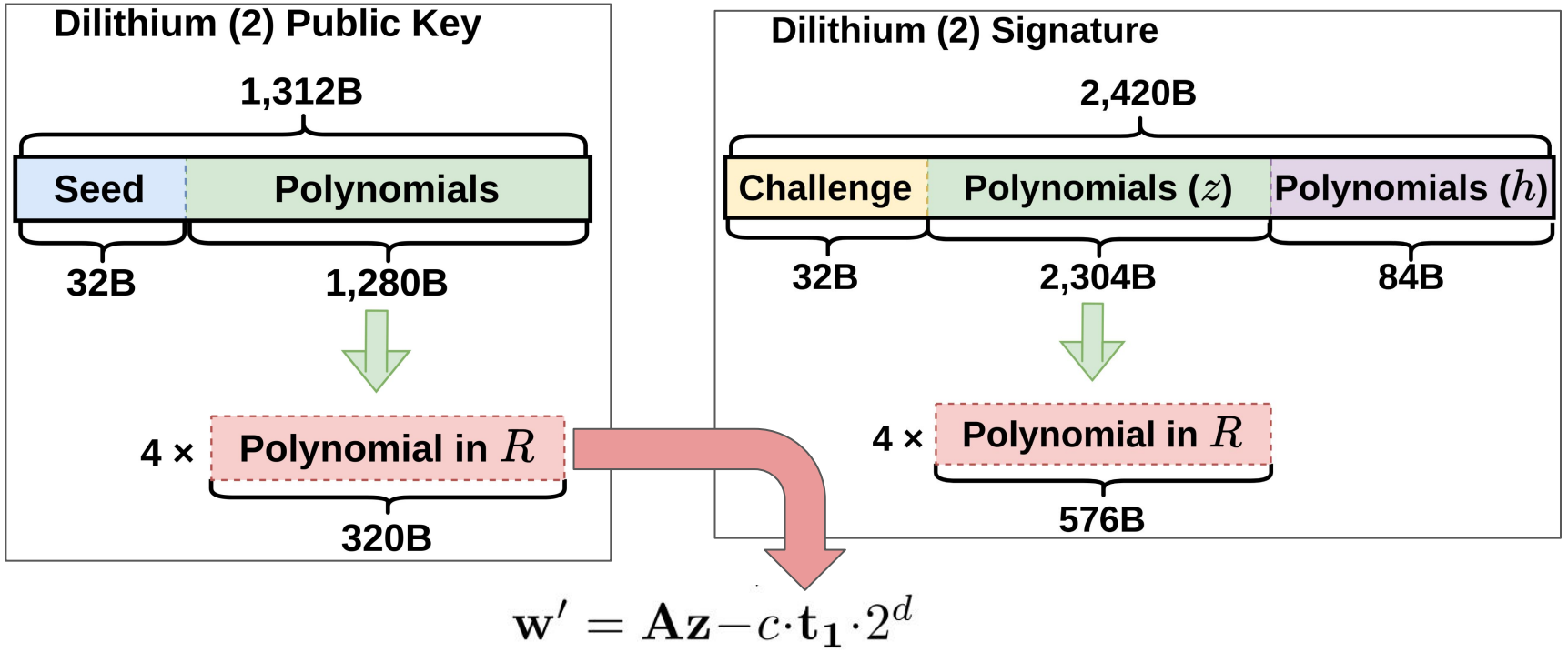
GeMSS (128)

- Hold **signature in memory**
- **Stream public key 4 times**
  - Verification has 4 iterations
  - in chunks of n × 2174B
- Based on reference code
  - No embedded implementation available





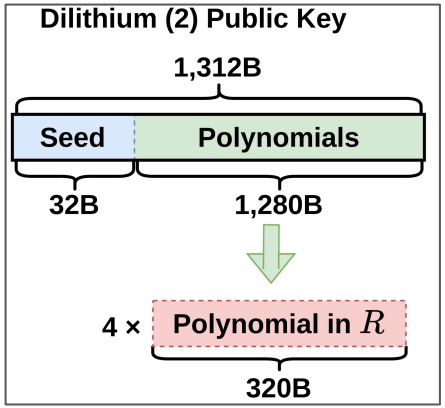
Dilithium (2)

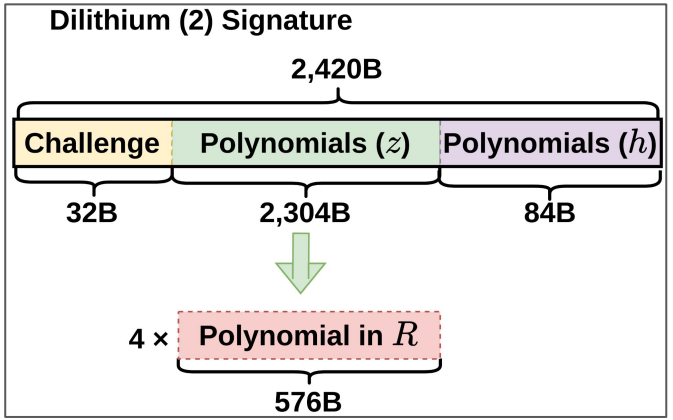


$$\mathbf{w}' = \mathbf{A}\mathbf{z} - c \cdot \mathbf{t_1} \cdot 2^d$$

Dilithium (2)

- Hold **signature in memory**
- **Stream public key**
  - One polynomials at a time
  - in chunks of 320B
- Based on optimized round 2 implementation for Cortex-M3/4
  - Updated for round 3 parameters





Falcon

- **Everything fits in memory**
- **No streaming required**
- Cortex-M3 compatible optimized implementation available
  - **No FPU** on M3, has to be **emulated**
  - Only tiny modification needed for memory constraints

Results[1]

	streaming data			streaming time	
	$\|pk\|$	$\|sig\|$	total	500 kbit/s	20 Mbit/s
sphincs-s[a]	32	7 856	7 888	126.2 ms	3.2 ms
sphincs-f[b]	32	17 088	17 120	273.9 ms	6.9 ms
rainbowI-classic	161 600	66	161 666	2 586.7 ms	64.7 ms
gemss-128	352 188	33	1 408 785[c]	22 540.6 ms	563.5 ms
dilithium2	1 312	2 420	3 732	59.7 ms	1.5 ms
falcon-512	897	690	1 587	25.4 ms	0.6 ms

[a] `-sha256-128s-simple` [b] `-sha256-128f-simple` [c] $4 \cdot |pk| + |sig|$

Data Volume

	w/o pk vrf.	w/ pk verification			w/ streaming
		pk vrf.	total	time[e]	20 Mbit/s
sphincs-s[a]	8 741k	0	8 741k	87.4 ms	90.6 ms
sphincs-f[b]	26 186k	0	26 186k	261.9 ms	268.7 ms
rainbowI-classic	333k	6 850k[d]	7 182k	71.8 ms	136.5 ms
gemss-128	1 619k	109 938k[c]	111 557k	1 115.6 ms	1 679.1 ms
dilithium2	1 990k	133k[c]	2 123k	21.2 ms	21.8 ms
falcon-512	581k	91k[c]	672k	6.7 ms	8.2 ms

[a] `-sha256-128s-simple` [b] `-sha256-128f-simple` [c] SHA-3/SHAKE
[d] SHA-256 [e] At 100 MHz (no wait states)

Cycle Counts

	memory				code
	total	buffer	.bss	stack	.text
sphincs-s[a]	6 904	4 928	780	1 196	2 724
sphincs-f[b]	7 536	4 864	780	1 892	2 586
rainbowI-classic	8 168	6 848	724	596	2 194
gemss-128	8 176	4 560	496	3 120	4 740
dilithium2	8 048	40	6 352	1 656	7 940
falcon-512	6 552	897	5 255	400	5 784

[a] `-sha256-128s-simple` [b] `-sha256-128f-simple`

Memory Usage

[1]for 1000 iterations

Resources

Paper: https://ia.cr/2021/662

Code: https://git.fslab.de/pqc/streaming-pq-sigs/

Hyptertree visualization by Abid Khan.