Virtual PUF: Built-in Model for Highly Reliable and Secure PUF

Presenter: Neelam NASIR,
Wei Cheng, Ulrich KUHNE, Tarik GRABA
and Jean-Luc DANGER

April 1st, 2026

CASCADE 2026

Outline

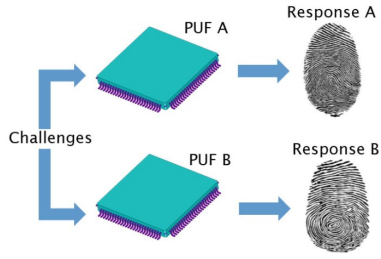
Physically Unclonable Functions - Overview

How it works

- Challenge $\rightarrow$ Unique physical response
- Response comes from manufacturing variations
- Difficult to clone or reproduce

Why it is useful

- Device Authentication
- Cryptographic Keys
- Anti-counterfeiting

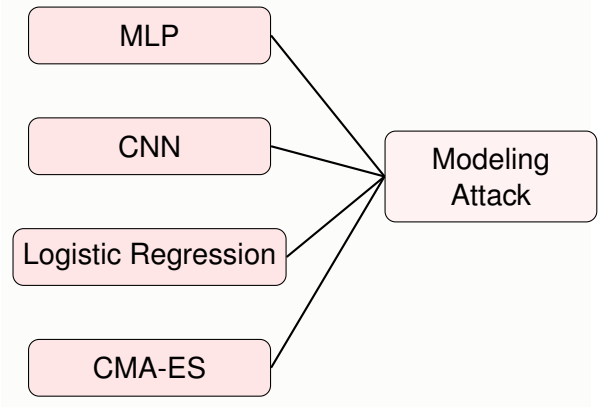


Key challenge for CRP PUF

Balancing resistance to modeling attacks, reliability, and low hardware cost.

Related Work: ML Attacks on PUFs

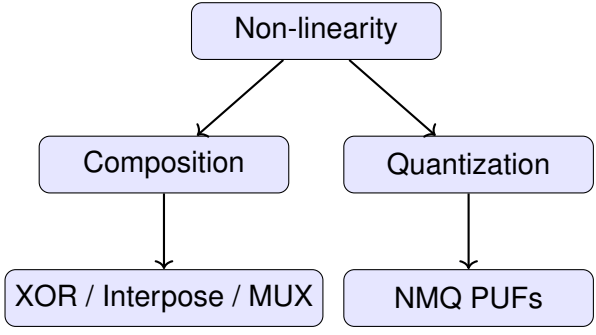
Typical attack models



Observation

- PUFs based on delay chains have an inherent linear behavior.
- Vulnerable to ML modeling attacks.
- Non-linearity is the key to security.

Related Work: Defenses Based on Non-Linearity



Key idea

- Stronger ML resistance comes from **non-linearity**
- For Arbiter-family PUFs: composition
- For Multi-bin PUFs: non-monotonic quantization

Key Point

This work exploits on NMQ as the non-linearity source.

Non-Monotonic Quantization (NMQ)

- **Applies to multi-bin PUFs** (e.g. RO-PUF, Loop PUF)
- **NMQ**: adjacent quantization intervals are mapped as 0/1/0/1
- **Effect**: harder ML prediction, but lower reliability when Q is high

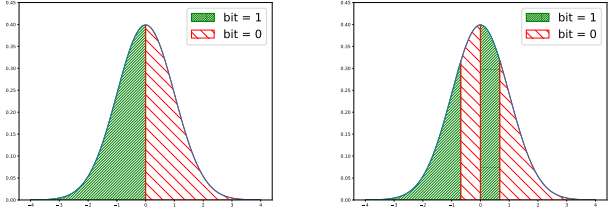
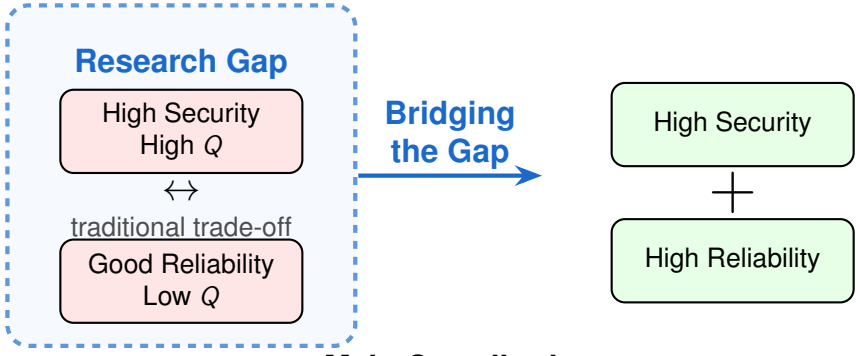


Figure: Quantization ($Q = 2$) and NMQ with ($Q = 4$)

[1] Stangherlin, K., Wu, Z., Patel, H., & Sachdev, M. (2022). Enhancing strong PUF security with nonmonotonic response quantization. *IEEE Trans. VLSI Syst.*, 31(1), 55-64.

Bridging the Gap between Security and Reliability



Main Contribution

Virtual PUF
Perfect reliable, Built-in Model, ML-resistant PUF architecture

Security Analysis: Modeling attacks against NMQ PUF

Simulation Results with no noise

- **Logistic Regression**: Resistant even at small Q values ($Q = 4$)
- **CNN/MLP Attacks**: Require higher Q for effective protection ($Q = 16, 32$)
- **CMA-ES**: No noise, no reliability attacks

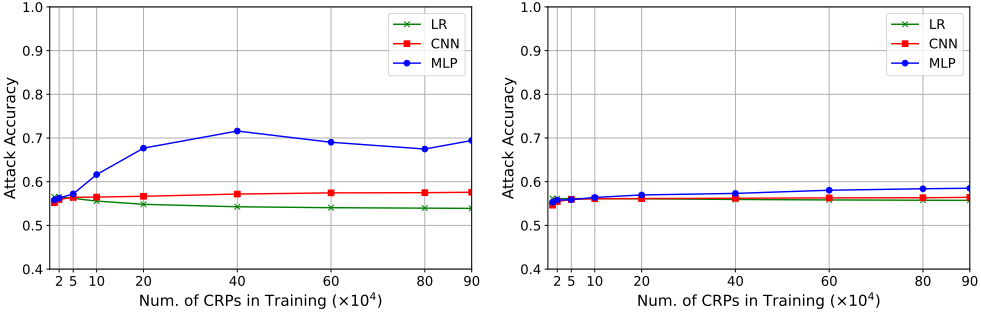


Figure: Experimental results of Modeling attacks at Q = 16,32

Outline

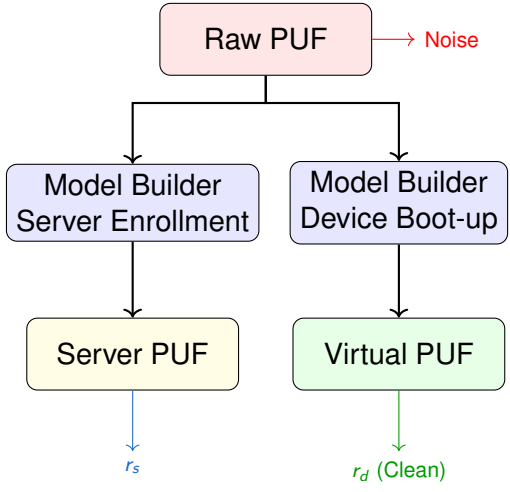
Virtual PUF
Core Concept

Core Concept

- At runtime, **digital built-in model** replaces physical PUF
- **Eliminates:** Environmental noise (perfect reliability)
- **Enables:** Higher Q with security

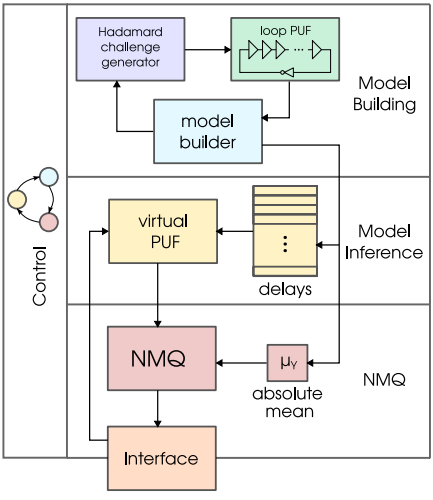
Is it feasible?

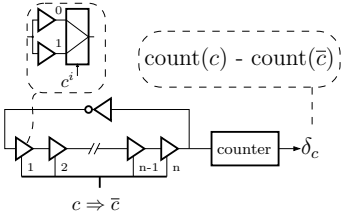
- Lightweight hardware implementation.
- Mismatch between device and server models.



Virtual PUF

Hardware Architecture





Loop PUF raw response

$$\delta_c = \hat{c} \cdot \hat{D}$$

- $\hat{c}$: challenge vector with $0 \rightarrow -1$
- $\hat{D}$: vector of delay differences

Lightweight Loop PUF Model

From physical PUF to model

$$D \approx H^{-1}\Delta_H$$

- H: Hadamard matrix
- $\Delta_H = (\delta_{H_0}, \delta_{H_1}, \ldots, \delta_{H_{n-1}})$

Hardware-friendly simplification

$$H^{-1} = \frac{1}{n}H^T$$

and for symmetric Hadamard: $H^T = H$

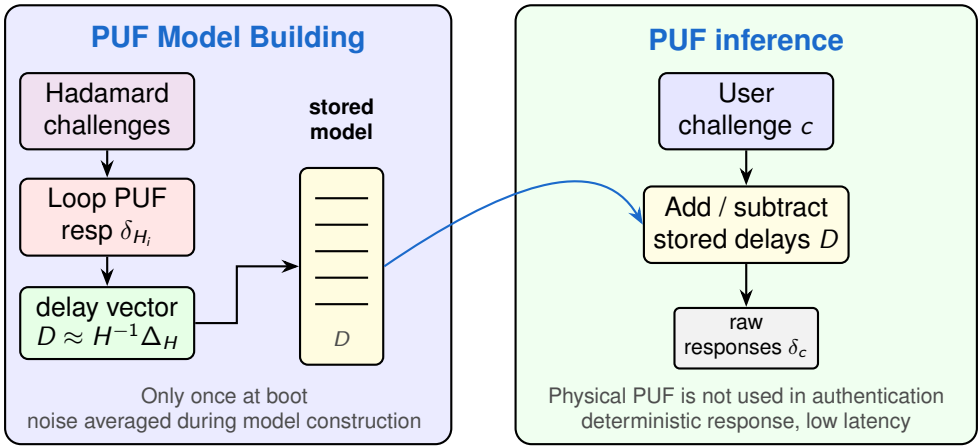
$$D_i = \frac{1}{n}\sum_{j=1}^{n} \begin{cases} +\delta_{H_j}, & H_{ij} = +1 \\ -\delta_{H_j}, & H_{ij} = -1 \end{cases}$$

Key Points

Hadamard challenges turn model building into **add/subtract + shift**, which is hardware friendly. The division by $n = 2^k$ is right-shift in hardware.

Virtual PUF

Model Building & Inference



Key idea: replace repeated noisy measurements with a stored internal model.

NMQ: Threshold Calculation

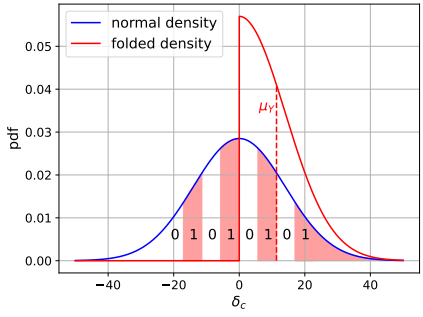


Figure: Thresholds for $Q = 8$ derived from μ_Y

$$\mu_Y = \frac{1}{n}\sum_{i=0}^{n-1} |\delta_{H_i}|$$

Folded mean from Hadamard responses

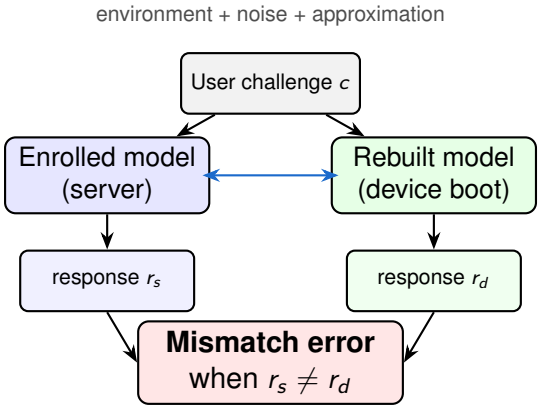
step size $= \mu_Y \gg (\log_2 Q - 2)$

Compare $|\delta_c|$ with thresholds

Toggle output across zones $\Longrightarrow$ final bit

Outline

What is Mismatch Error?



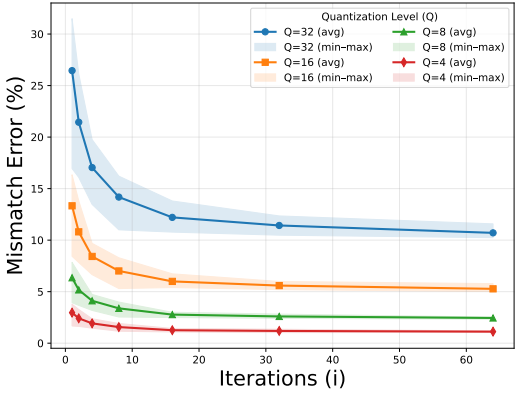
Average mismatch error: $e = \Pr\left[r_s(c) \neq r_d(c)\right]$

Main causes

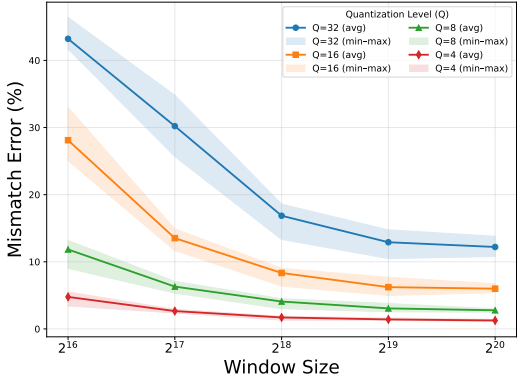
- **Higher** Q: more thresholds, more sensitivity
- **Low iterations** i: less averaging of noise
- **Small window** w: noisier model construction
- **Low precision** p: larger rounding error
- **Temperature shift** T: server and device models diverge

Mismatch Error Analysis

Iterations & Window Size



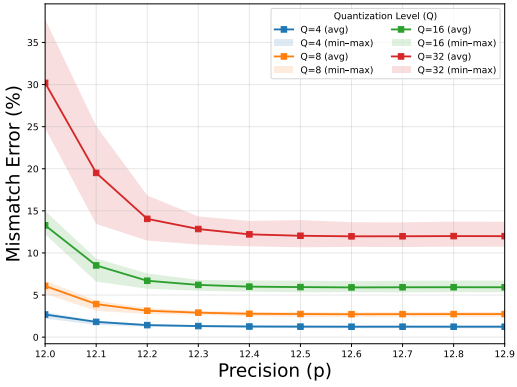
Impact of **iterations** on mismatch error ($p = 12.4$, $T = 32°\text{C}$, $w = 2^{20}$)



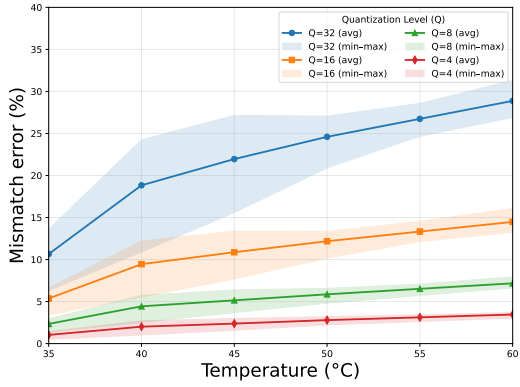
Impact of **window size** on mismatch error ($p = 12.4$, $T = 29°\text{C}$, $i = 16$)

Mismatch Error Analysis

Precision & Temperature

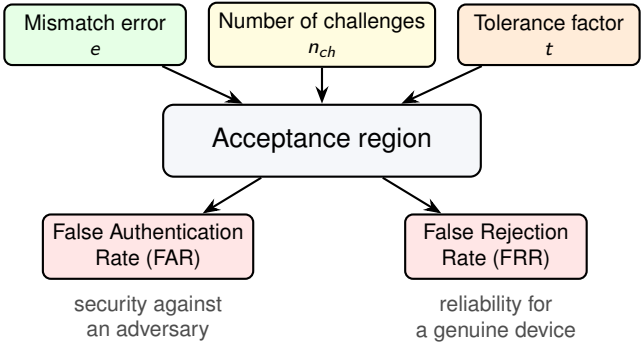


Impact of **precision** on mismatch error ($i = 16$, $T = 32°\text{C}$, $w = 2^{20}$)



Impact of **temperature** on mismatch error ($p = 12.4$, $i = 16$, $w = 2^{20}$)

Impact of Mismatch Error on Authentication

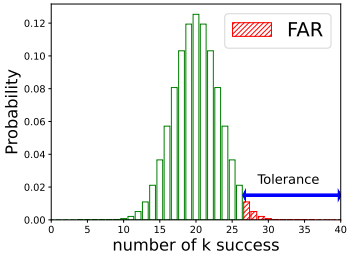


Main effect of mismatch

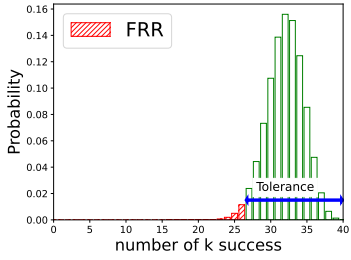
- As **mismatch error** e increases, the **FRR worsens**
- This can be compensated by:
  - increasing $\mathbf{n_{ch}}$
  - relaxing tolerance $\mathbf{t}$
- But too much tolerance increases **FAR**

PUF Authentication

Binomial distributions for FAR and FRR



(a) Adversary distribution

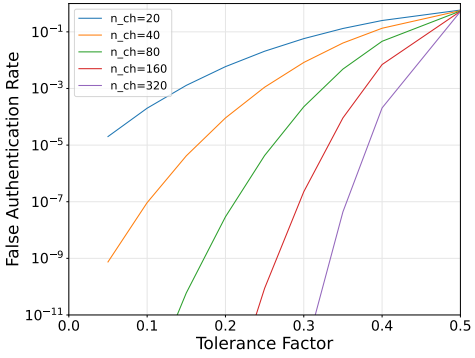


(b) Legitimate distribution

Key points

Mismatch error mainly increases **FRR**; tolerance must be tuned carefully to avoid raising **FAR**.

FAR vs. Tolerance for different numbers of challenges



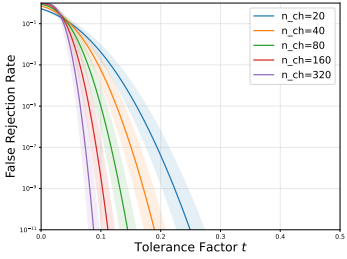
Key Points

- Larger tolerance t increases **FAR**
- More challenges n_{ch} strongly reduce FAR
- For high security, increasing n_{ch} is better than relaxing tolerance

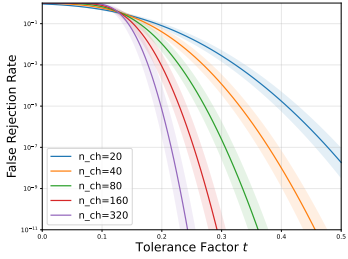
FRR vs. Tolerance for Different Numbers of Challenges

Key Points

- $e \uparrow \Rightarrow$ FRR curves shift right
- $t \uparrow \Rightarrow$ FRR $\downarrow$ but security $\downarrow$
- $n_{ch} \uparrow \Rightarrow$ FRR $\downarrow$



$Q = 8,\ e = 2.8\%$



$Q = 32,\ e = 12.2\%$

Physical Implementation Characteristics

Complexity

Artix-7 XC7A35T Resource utilization ($n = 64$, $p = 12.4$, $i = 16$, $w = 2^{20}$)

Module	LUTs	Registers	RAM (bits)
H-ch-gen	29	63	0
LPUF	137	156	0
Model builder	206	172	1280
Virtual LPUF	58	54	0
Delay models	0	0	1024
NMQ	74	62	0
TOTAL	**504**	**507**	**2304**

(H-ch-gen = Hadamard Challenge Generator)

Usage: 2.4% of LUTs, 1.2% of FFs,and <0.2% of BRAM, confirms lightweight.

Latency

Operational phase: $t_{\text{operation}}$ is very low for 64 delays and a 100 MHz clock, about 640 ns plus NMQ quantization.

Model building phase: t_{model} depends on window size w and iterations i.

Window size	t_{model}		$t_{\text{operation}}$
	$i = 8$	$i = 16$	
16	0.67 s	1.34 s	1.3 µs
17	1.34 s	2.68 s	1.3 µs
18	2.68 s	5.37 s	1.3 µs
19	5.36 s	10.73 s	1.3 µs
20	10.74 s	21.47 s	1.3 µs

Outline

Conclusion

Security	Practicality	Lightweight
High-Q NMQ harder ML modeling	**Virtual PUF** reliable and secure	**Low Hardware cost** suitable for IoT

Virtual PUF allows us to use high-Q NMQ for strong ML resistance while achieving perfect reliability by replacing the noisy physical PUF with a built in digital model.

Main challenge: mismatch error between server and device
Future work: lower mismatch and temperature analysis/compensation