

VAHANA

Hardware-Aware Noise Addition Against Gradient Inversion Attacks

Arsalan Ali Malik* · **Digvijay Anand*** · **Chau-Wai Wong** · **Aydin Aysu**

*Co-first authors | North Carolina State University, Raleigh, NC, USA

**37th IEEE International Conference on Application-specific
Systems, Architectures and Processors (ASAP 2026)**

NC STATE UNIVERSITY

Roadmap

- 1 Why edge federated learning leaks private data
- 2 Why today's defense doesn't fit edge hardware
- 3 VAHANA: approximate reparameterization
- 4 RISC-V hardware–software co-design
- 5 Results: privacy kept, cost removed

The headline

28.1×

lower reparameterization latency

34%

smaller ROM footprint

0%

gradient-inversion attack success

Federated learning is moving to the edge

- Clients train locally and share only gradient updates — raw data never leaves the device
- Deployed on phones, wearables, XR headsets, IoT nodes, in-vehicle and clinical systems
- Attractive exactly where data is most sensitive: health diagnostics, autonomy, fraud detection

The raw data stays local — but the gradients still travel.

Data locality

Images, speech, biometrics and scans stay on-device

Tight SWaP budgets

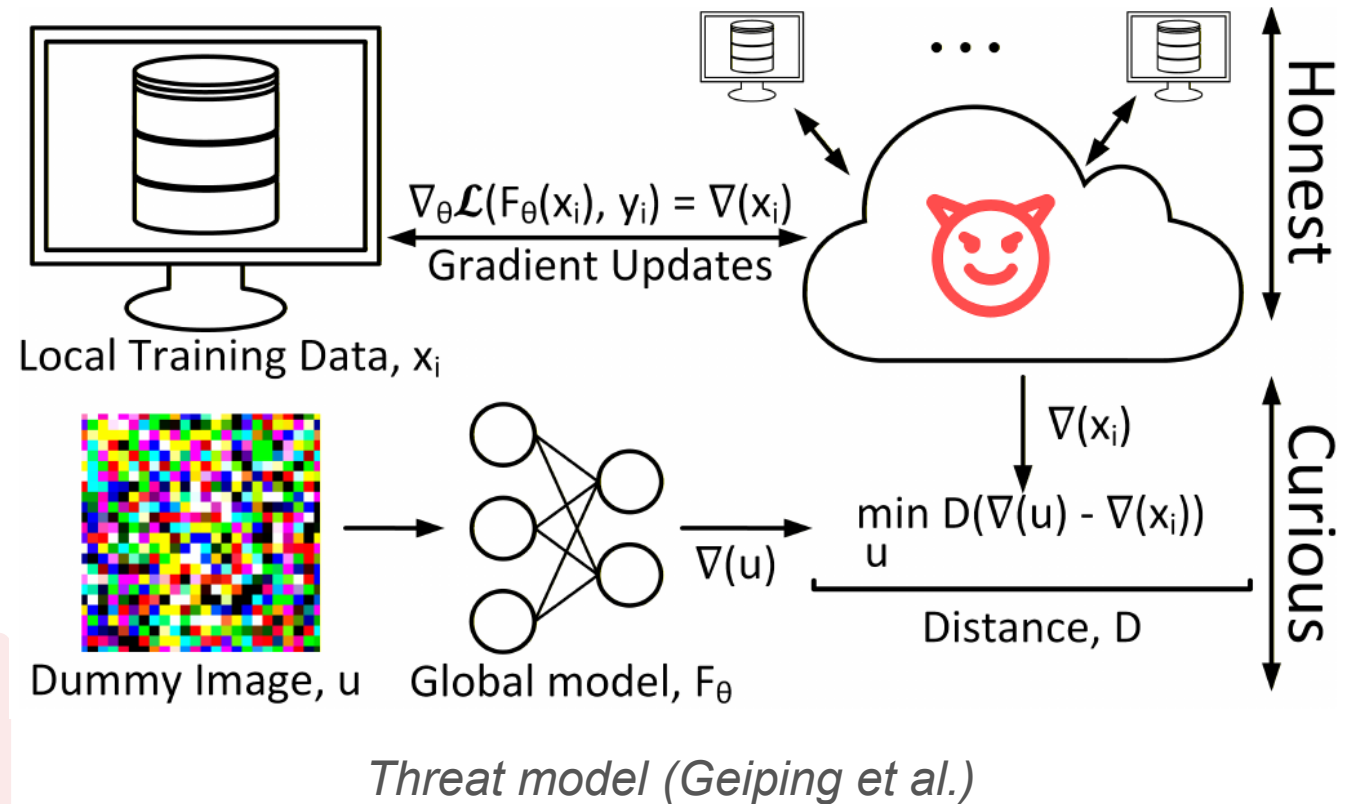
Delay, size, area and power are all hard constraints

Limited entropy

Deterministic compute graphs, little on-chip randomness

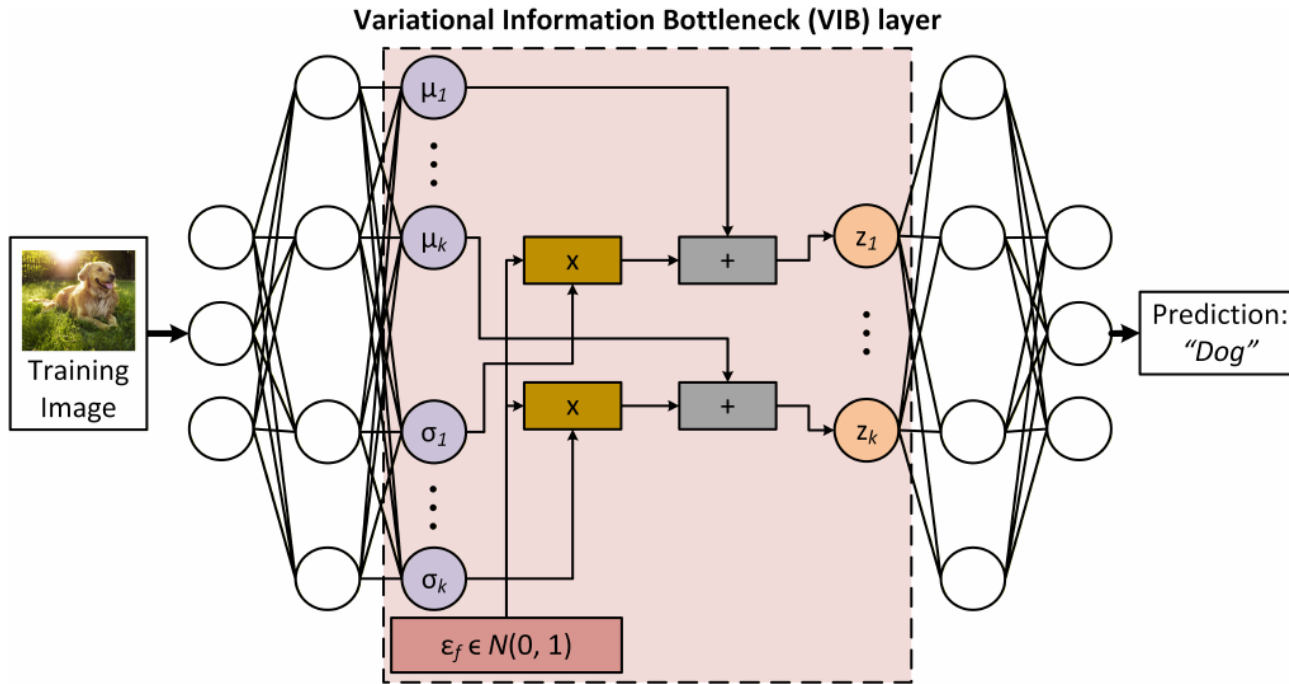
Gradient inversion breaks the privacy promise

- Honest-but-curious server: follows the protocol, but reconstructs client data
- Knows architecture, weights and loss — and receives the gradients
- Optimizes a dummy input u until $\nabla(u)$ matches $\nabla(x_i)$



Unprotected models leak almost perfectly:
SSIM ≈ 1.0 · attack success 100%

PRECODE: the variational bottleneck defense



- Insert a variational bottleneck (VIB) layer before the classifier
- Reparameterize: $z = \mu + \sigma \cdot \epsilon$, $\epsilon \sim N(0, 1)$
- Train with cross-entropy + $\beta \cdot \text{KL}(N(\mu, \sigma^2) \parallel N(0, 1))$
- Gradients decouple from individual samples — attack success drops to 0%

Effective — but it assumes a Gaussian RNG and a full-precision FP multiplier sitting on the training datapath.

Why that doesn't fit on edge hardware

Latency, area, memory

Reparameterization runs every training step and consumes scarce FP/FMA throughput

No Gaussian source

Edge SoCs expose uniform PRNGs; Box–Muller and Ziggurat need multipliers or large LUTs

'Precision by default'

Variational parameters are bounded and sparse — full FP buys memory traffic, not privacy

No ISA support

No native entropy or stochastic rounding; everything falls back to software emulation

1,249

cycles per Gaussian sample
(glibc)

3,873

cycles per sample (Trivium32)

98.8%

of baseline latency is RNG

Our key observation

Exact Gaussian samples are not required.

What the defense actually needs from the latent perturbation:

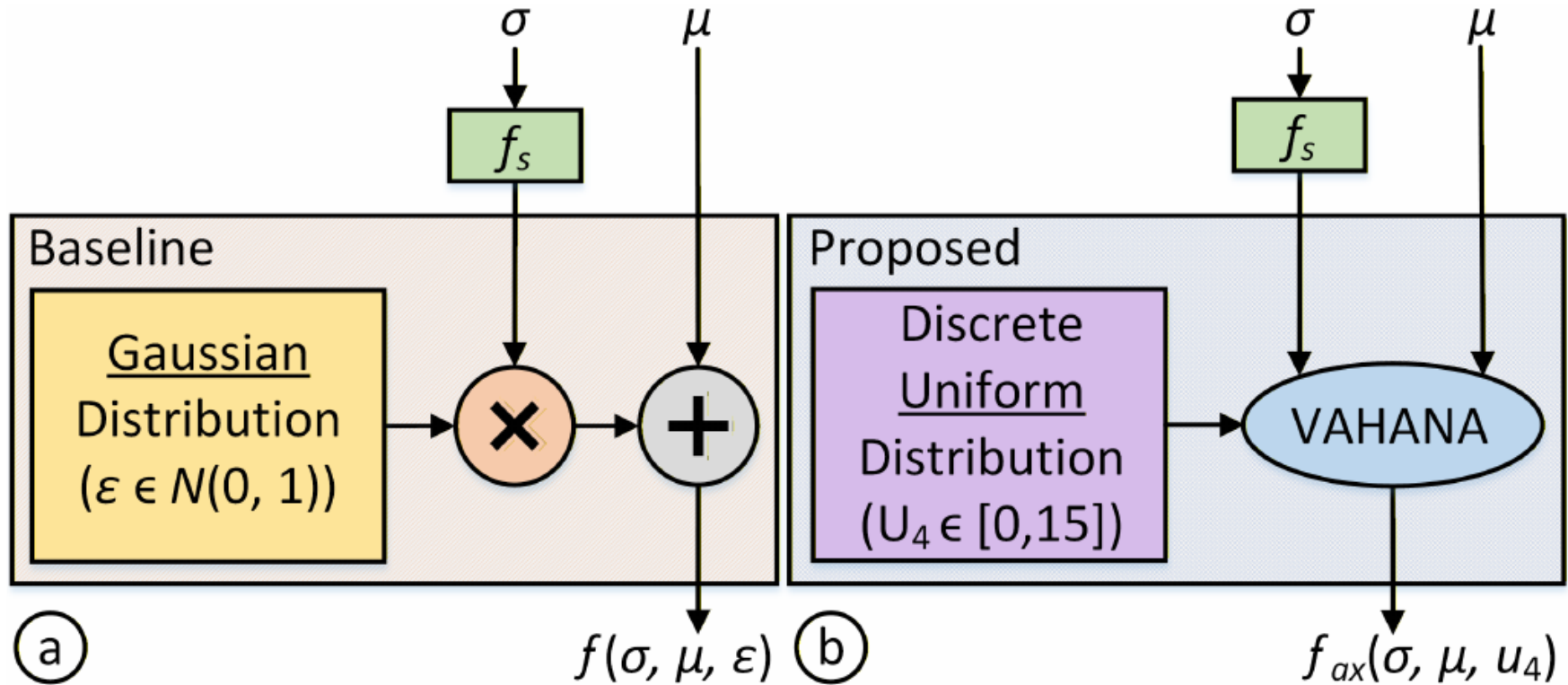
Bounded variance

**Sample-to-sample
variability**

**Decoupling from
individual examples**

We treat Gaussian replacement as an empirical hardware-approximation opportunity — not as a general privacy theorem.

VAHANA: approximate reparameterization



One Gaussian RNG and one FP multiply $\rightarrow$ four uniform bits and an approximate adder.

Inside the datapath: four lightweight blocks

1

Uniform randomness encoder

$q = 2(\sum_j U[j] - 2) \in \{-4, -2, 0, 2, 4\} \rightarrow$ controls ax_s, ax_m, ax_{en}

2

Sign-perturbation unit

$\Delta.\text{sign} = \text{sign}(\mu) \oplus ax_s$ — randomized additive / subtractive noise, no sampler

3

Exponent realignment

$\Delta.\text{exp} = \text{exp}(\sigma) + 1 + ax_m$ — an exponent adder replaces the FP mantissa multiplier

4

Mantissa sparsifier

Zero, or keep only the implied leading bit $\rightarrow \Delta \in \{0, \pm 2, \pm 4\}$

Δ enters the existing IEEE-754 FMA pipeline — align, normalize and round stages are untouched.

Worked example: four bits to one perturbation

Sample

$U = 8$
 $\rightarrow 1000_2$

Encode

$\sum_j U[j] = 1$
 $q = 2(1 - 2) = -2$

Control

$ax_s = 1$
 $ax_m = 0$
 $ax_{en} = 1$

Given the encoder outputs $(\mu, \sigma) = (1.25, 1.0)$

$\Delta \approx -2$

sign flipped, smaller magnitude,
injection on

$z = \mu + \Delta$

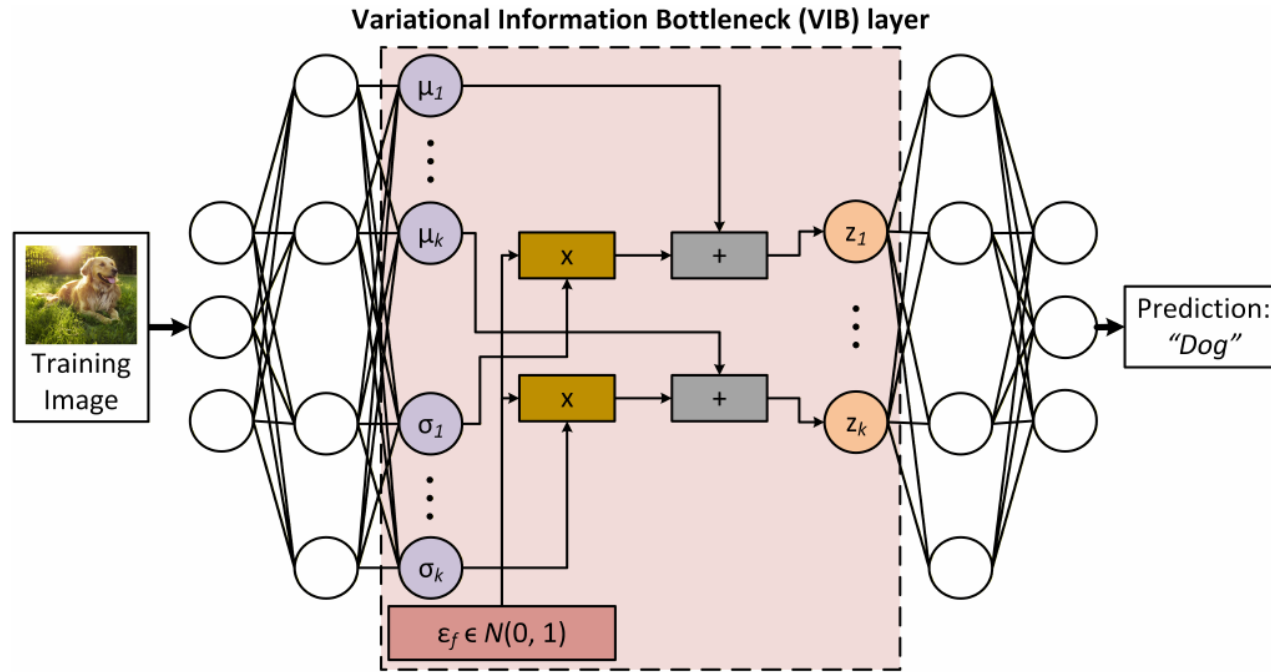
the approximate add reuses the
FMA pipeline

$z = -0.75$

stochastic latent, no Gaussian
hardware

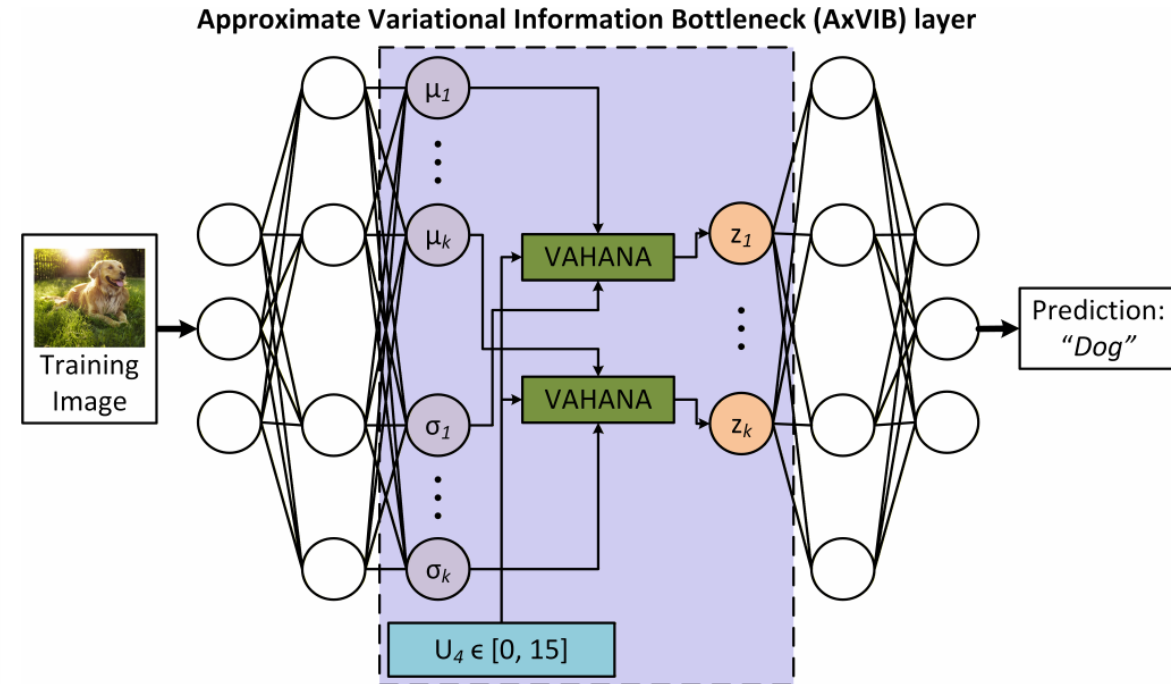
Same insertion point, cheaper internals

PRECODE — VIB layer



Gaussian sampler + FP multiplier + adder

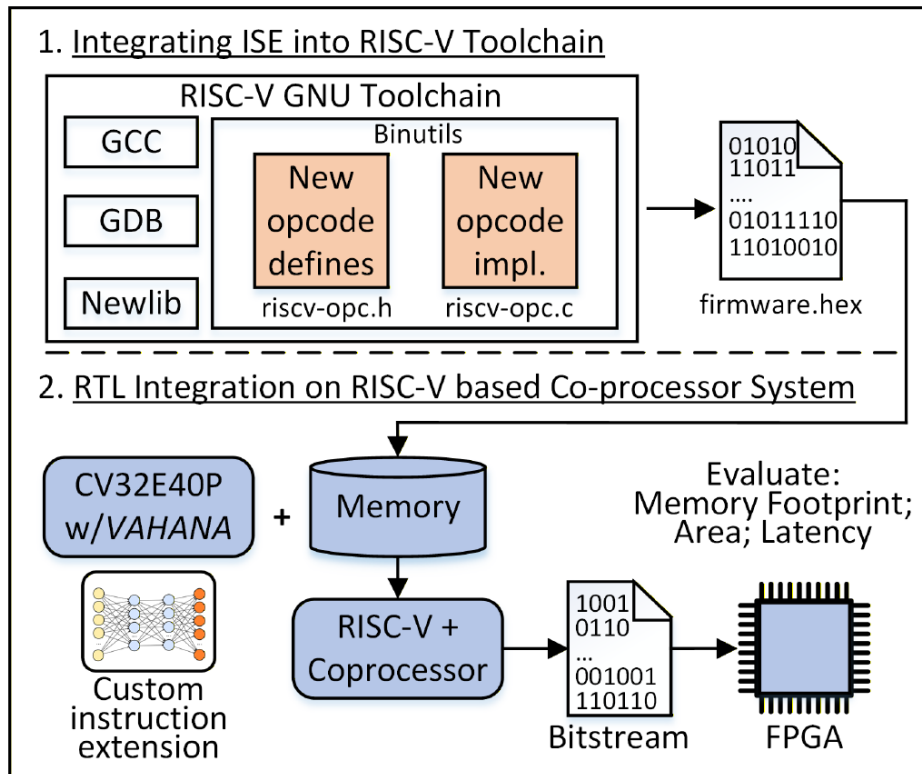
VAHANA — AxVIB layer



4-bit uniform source + approximate add

Applied independently per latent coordinate — changing the bottleneck width k scales latency, not the datapath.

Hardware–software co-design on RISC-V



```
vahana_appx_vib rd, rs1, rs2, rs3
```

R4-type, reserved opcode 5'b11010

rs1 = μ , rs2 = σ , rs3 = U, rd = z

- Toolchain: new opcode in binutils (riscv-opc.h / .c) plus a GCC inline-assembly template
- RTL: decode in CV32E40P feeds the approximate FMA front end; IEEE-754 backend untouched
- Prototype: Kintex-7 XC7K160T, Vivado 2022.2, Verilator for cycle-accurate profiling
- Portable to any RISC-V core exposing custom opcodes and an FP/FMA pipeline

Evaluation setup

Datasets

CIFAR-10
CIFAR-100
Medical MNIST

Backbones

SMLP (2×1024)
DMLP (4×1024)
LeNet (768-d latent)

Attack

Geiping et al.
Batch of 128 images
7,000 iterations

- Training: 300 epochs, Adam, PRECODE's objective, implemented in PyTorch
- Configurations compared: unprotected · PRECODE · VAHANA
- Entropy sources: glibc PRNG (cheap, weaker guarantees) and Trivium32 stream cipher (cryptographic)

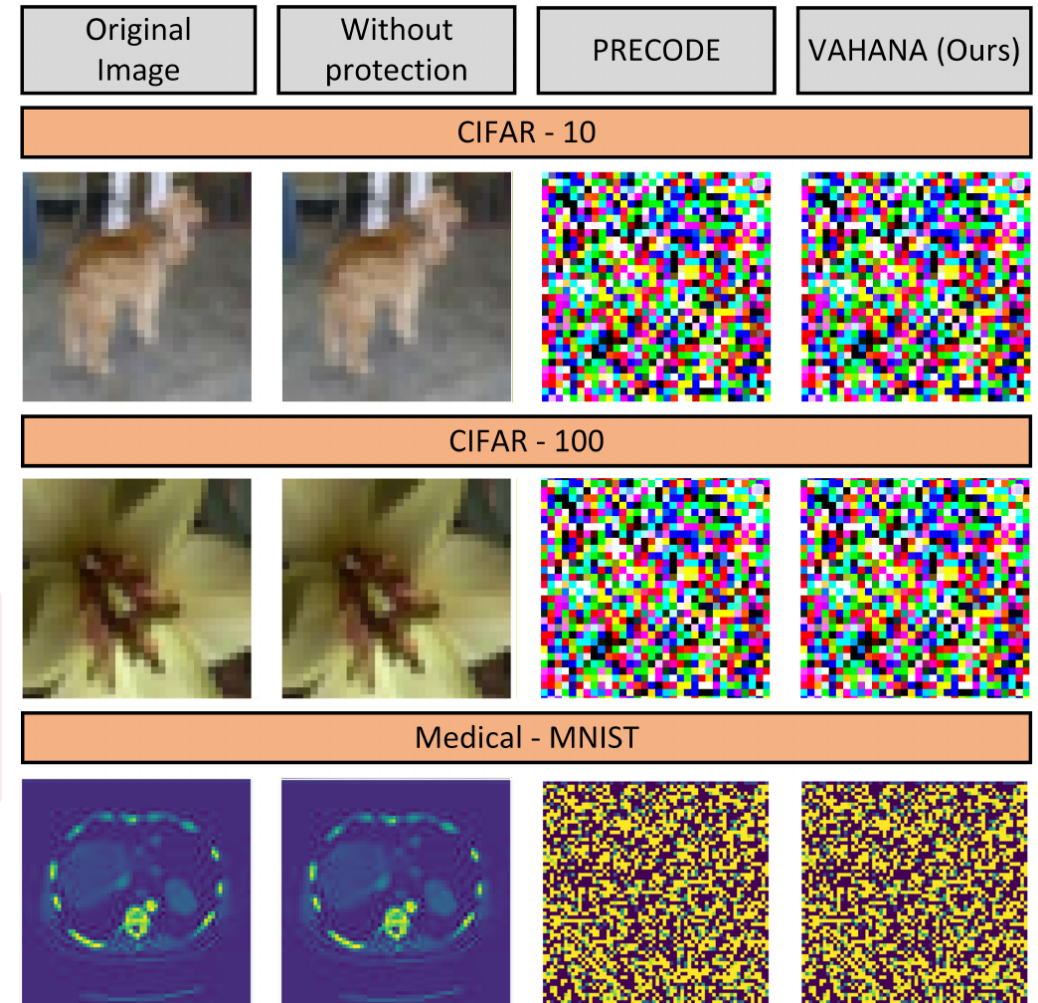
Privacy: reconstructions collapse to noise

- Unprotected: the attacker recovers a near-replica of the private sample
- PRECODE and VAHANA both destroy all semantic content
- Same behaviour on CIFAR-10, CIFAR-100 and Medical MNIST

SSIM 0.99 → 0.00

Attack success 100% → 0%

VAHANA's discrete perturbation is as disruptive as continuous Gaussian noise.



Privacy and utility preserved together

Metric	Unprotected	PRECODE	VAHANA
Test accuracy (%) ↑	53.30	51.70	52.50
MSE ↑	0.00	4.71	4.64
LPIPS ↑	0.00	0.39	0.40
PSNR ↓	87.24	5.53	5.51
SSIM ↓	0.99	0.00	0.00
Attack success (%) ↓	100.00	0.00	0.00

CIFAR-10, DMLP backbone · ↑ higher is better, ↓ lower is better

≤ 1 pt

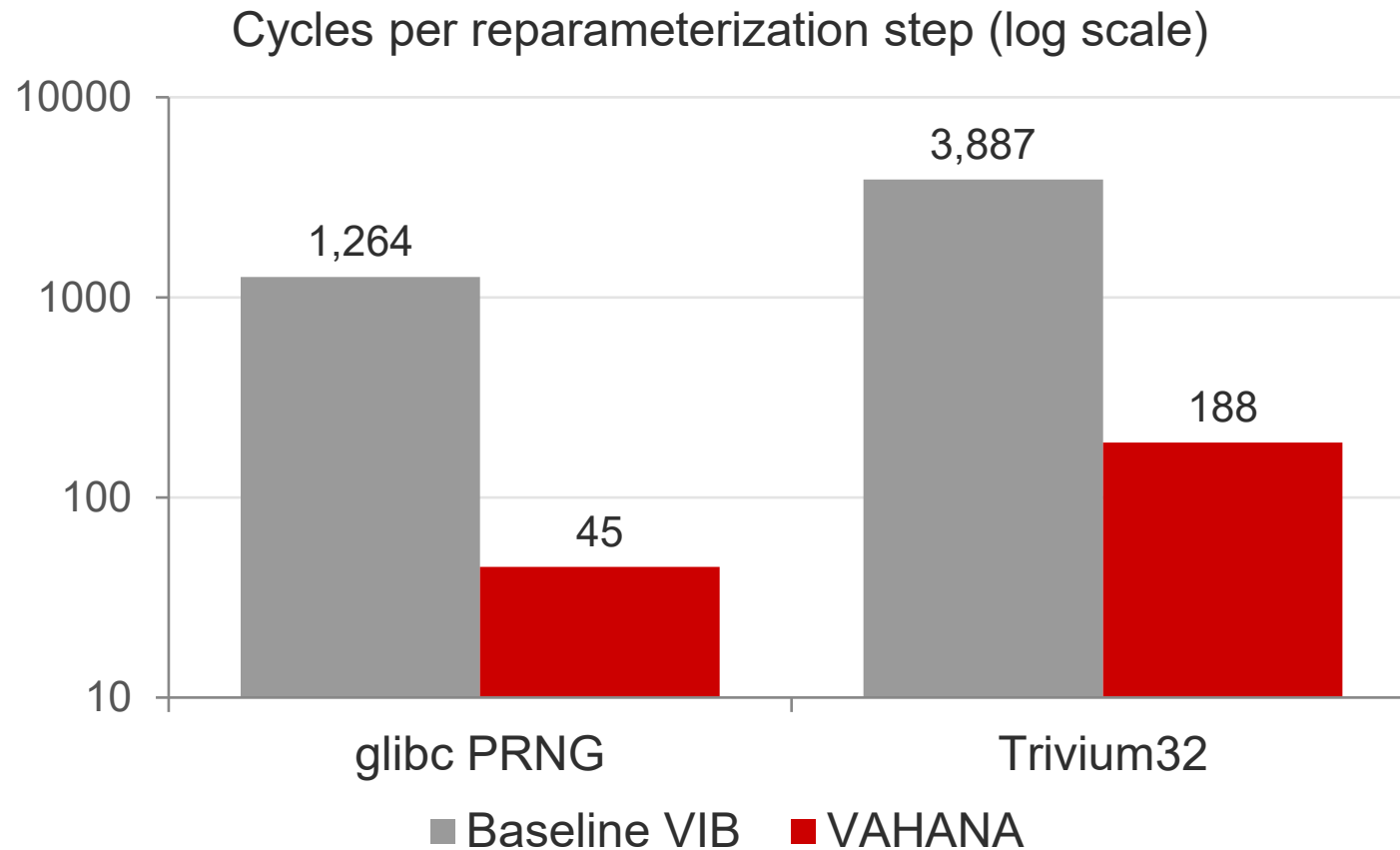
test-accuracy gap versus the unprotected baseline

0%

attack success on every dataset and backbone tested

Also holds on CIFAR-100 and Medical MNIST, where VAHANA matches or slightly improves MSE, PSNR and LPIPS.

Latency: reparameterization gets 28× faster (8%↓ training)



28.1×

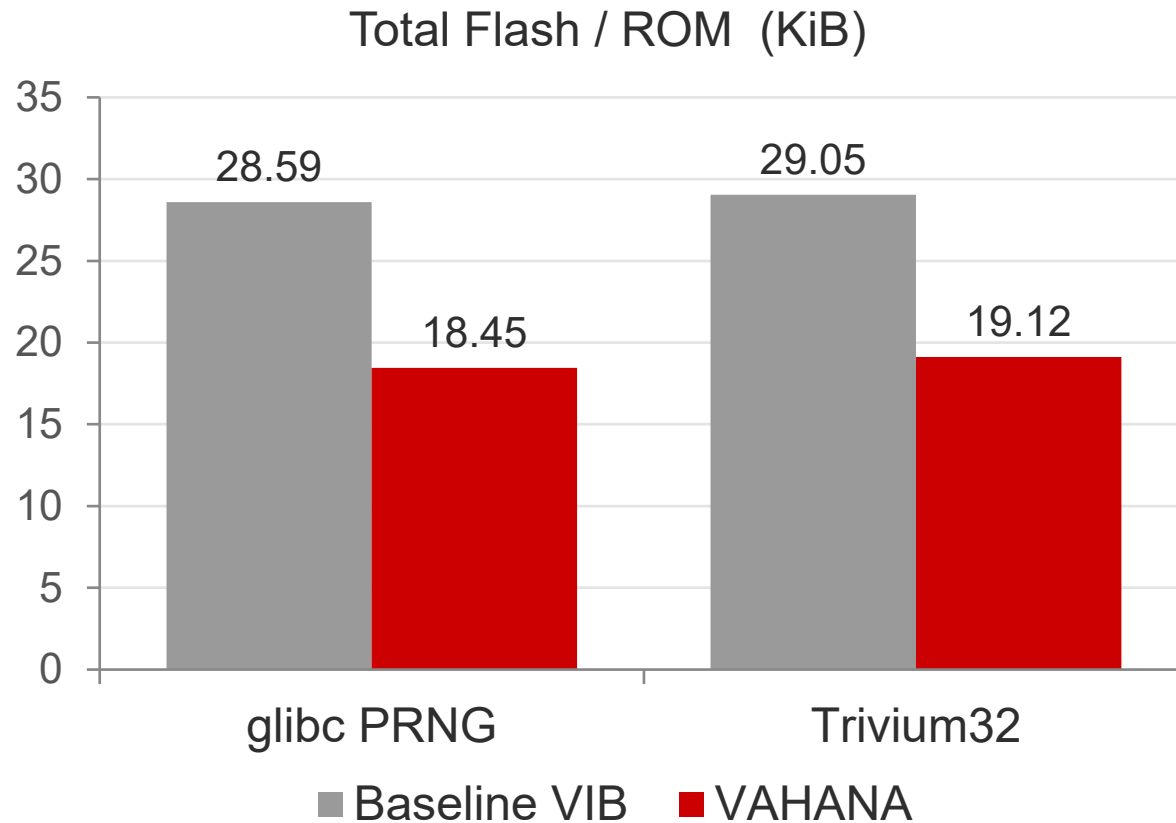
end-to-end speedup with glibc, at
latent width $k = 256$

20.7×

end-to-end speedup with Trivium32

RNG dominates the baseline:
1,249 of its 1,264 cycles. FMA
cost rises only $14 \rightarrow 19$ cycles for
the R4-type instruction.

Memory footprint and area



–35.5% / –34.2%

ROM saved (glibc / Trivium32); RAM essentially unchanged

Configuration	LUT	FF	DSP
VIB (no pipeline)	723	0	0
VAHANA (4-stage)	126	41	2
CV32E40P core	13,348	6,492	7

VAHANA reuses the core's floating-point pipeline: about 1% of the core's LUTs, with no standalone Gaussian RNG datapath and no full-precision mantissa multiplier.

Conclusion

- 1 Exact Gaussian sampling is not needed for empirical resistance to gradient inversion
- 2 Four uniform bits plus sign / exponent / mantissa logic reproduce PRECODE's protection and utility
- 3 As a CV32E40P custom instruction: up to $28.1\times$ lower latency and 34% smaller ROM

Future work: formal privacy guarantees · larger architectures · adaptive attacks on the discrete perturbation · power and energy measurements

Thank you

Questions?

Arsalan Ali Malik · aamalik3@ncsu.edu

Source code: anonymous.4open.science/r/VAHANA-3C29

Funded by ONR grant N00014-23-1-2103 and NSF (CNS-1943245, SaTC 2340856).
The views expressed are those of the authors and do not represent official policy of the DoD or the U.S. Government.

NC STATE UNIVERSITY