Stash in a Flash

Aviad Zuck, Yue li, Shuki Bruck,

Donald E. Porter, Dan Tsafrir

TECHNION Israel Institute of Technology

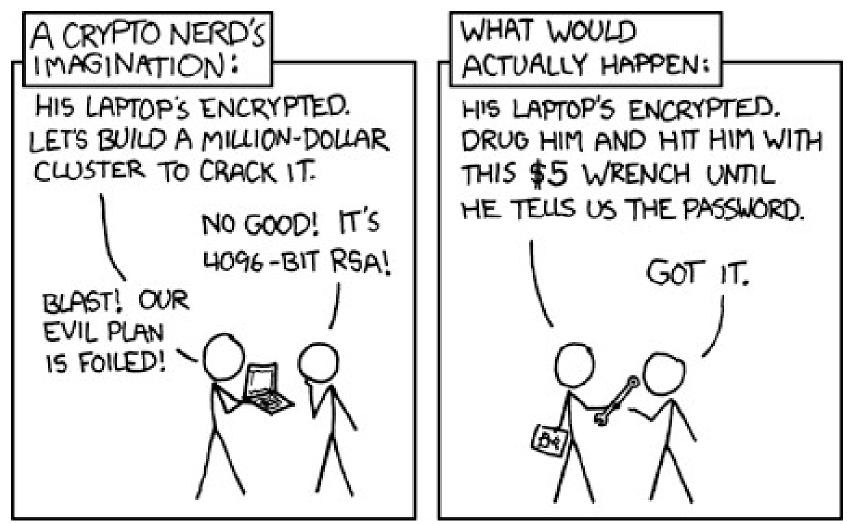
Caltech

Outline

- Motivation
- Background
- How to hide
- Detectability
- Performance
- Conclusion

Context

- This paper is about hiding data with **plausible deniability** in flash memories
- Encryption denies access to private data
- <u>Our goal</u>: adversary cant tell if system is even hiding data



Motivation

- Human rights activist crossing a border in a country ruled by a dictatorship
- User device carries sensitive data
- Intelligence officer at border checkpoint inspects device
- Can confiscate device, and demand encryption key!
  - May be resolved with plausible deniability

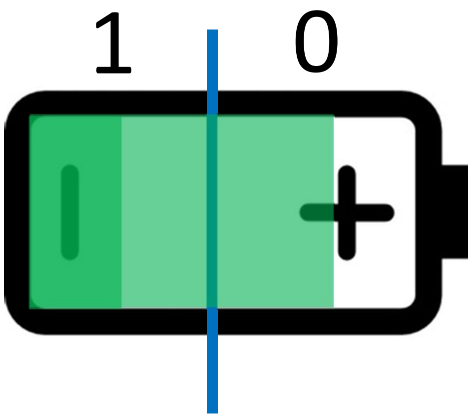
Our contribution (in context)

- New data hiding technique in flash
- Going against a potent adversary (e.g., government) is extremely challenging
- This paper: a building block towards complete solution
  - Some pieces solved by others
  - Some pieces open problems

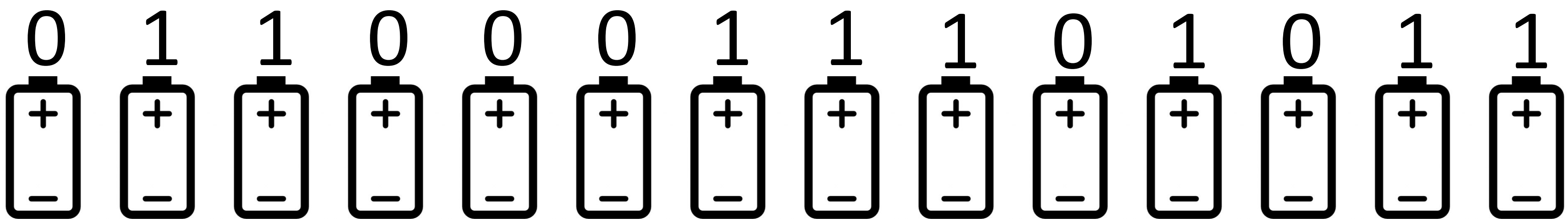
Outline

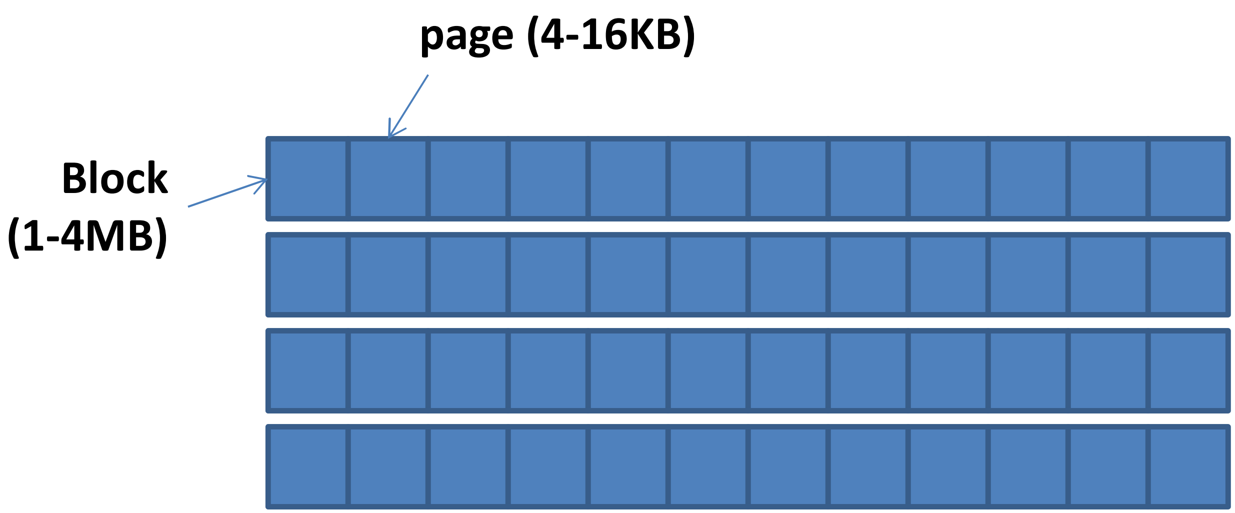
- Motivation
- Background
- How to hide
- Detectability
- Performance
- Conclusion

Storing a single bit in flash



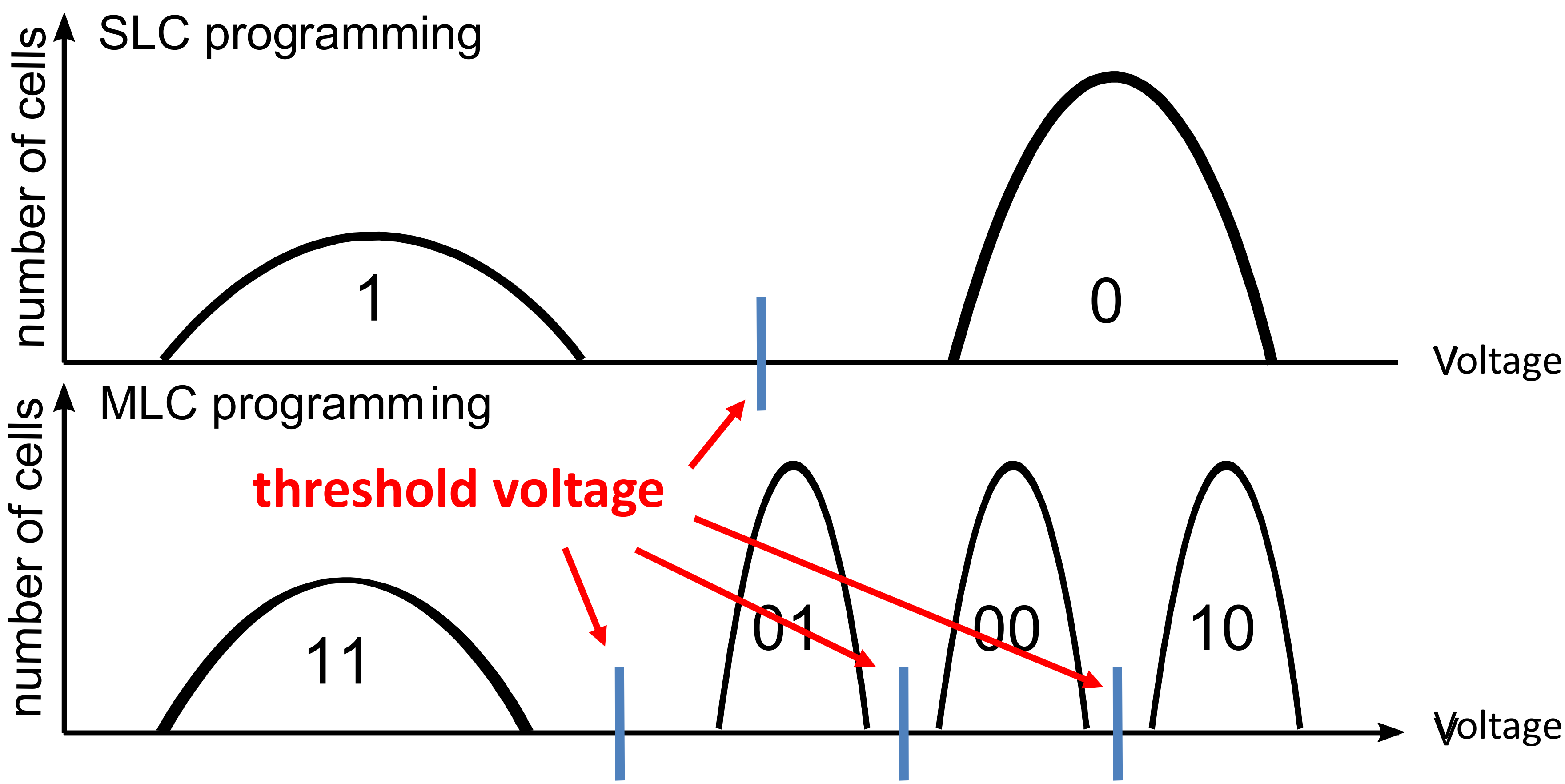
Storing multiple bits in flash



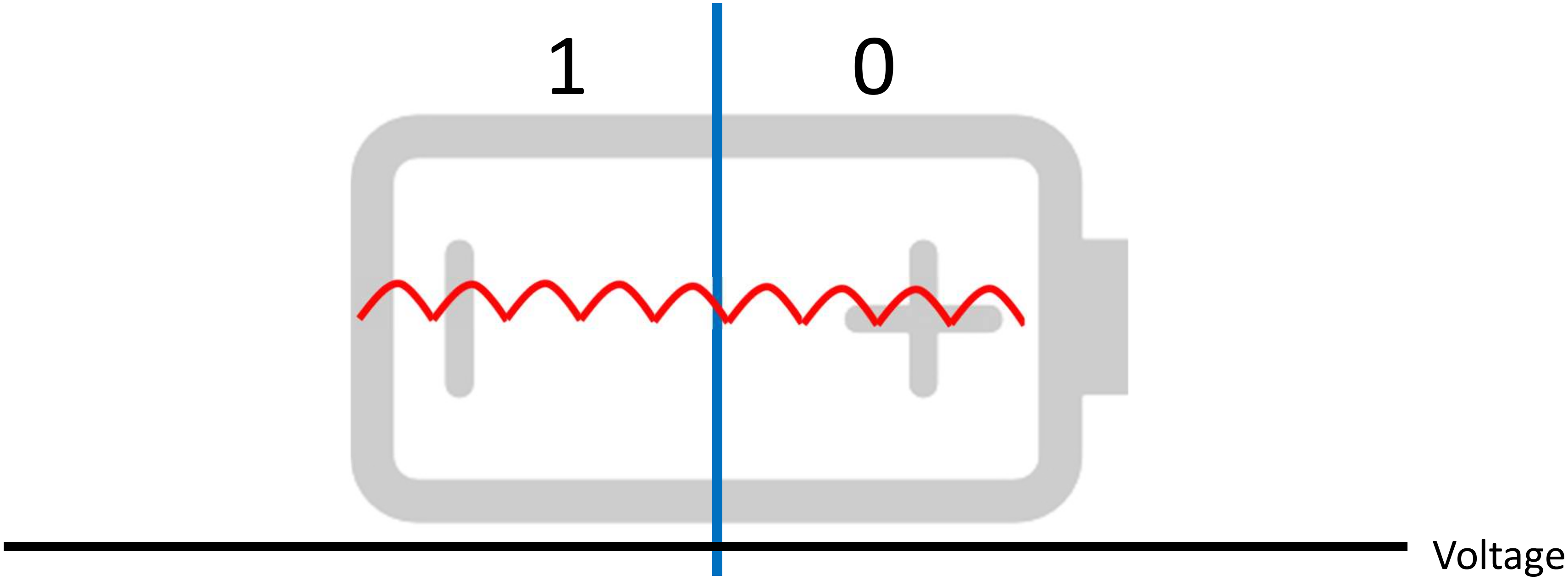


- Page is the read/write unit
- Block is the erase unit

Histogram of bits in a flash chip

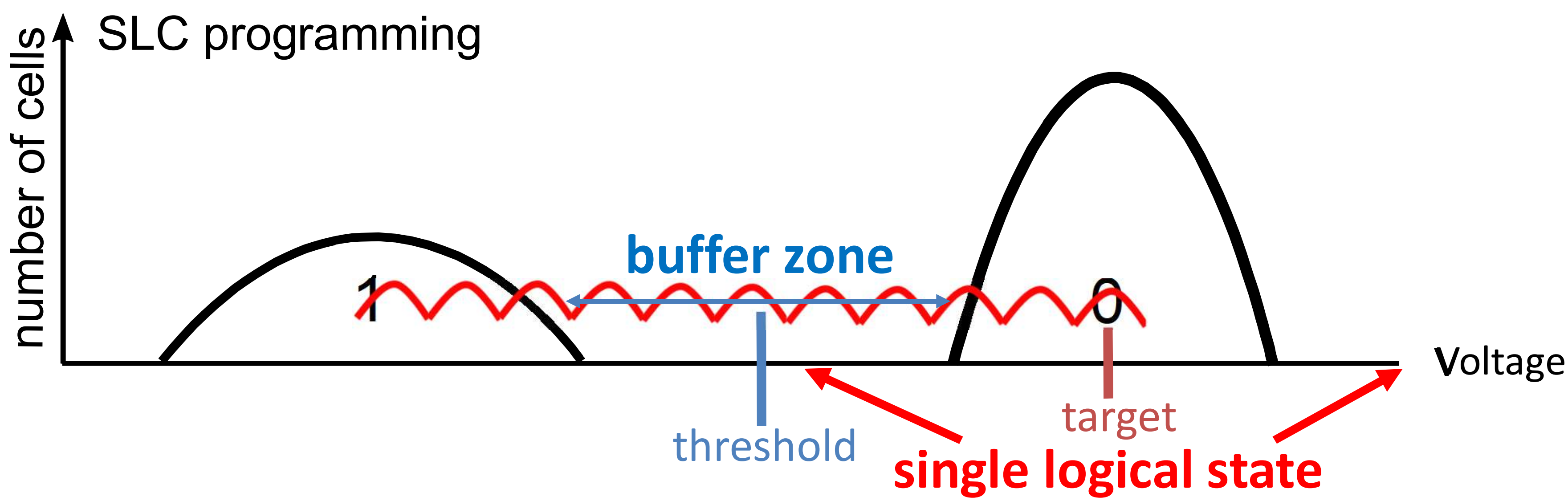


Programming a cell



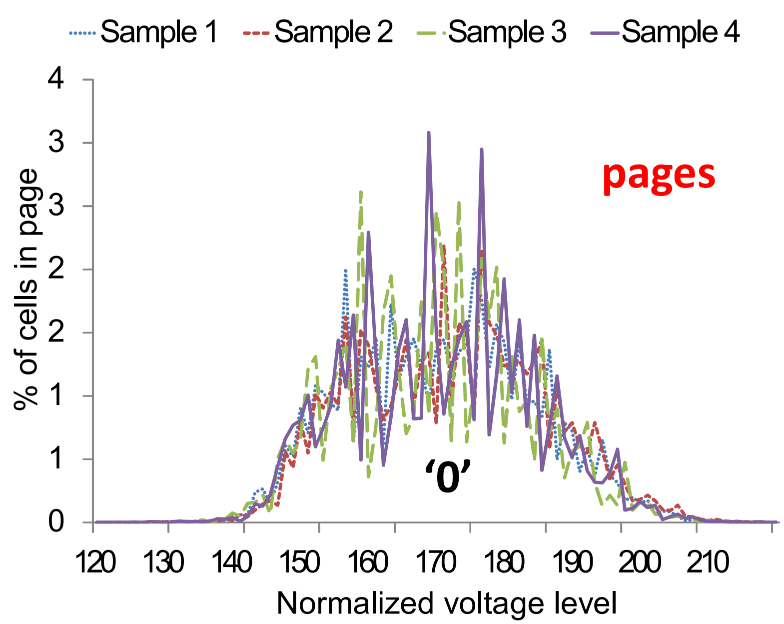
- Flash hardware logic internally applies multiple charging pulses

Programming is imprecise (1)



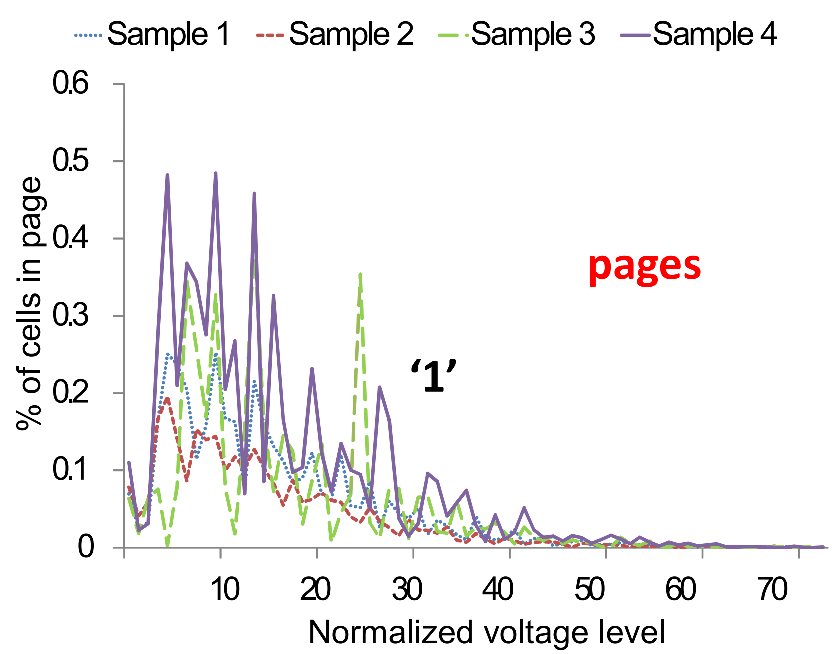
Programming is imprecise (2)

- Variations exist at all levels:
  - Flash chips of same vendor and model
  - Different areas in chip
  - Different blocks/pages in same area



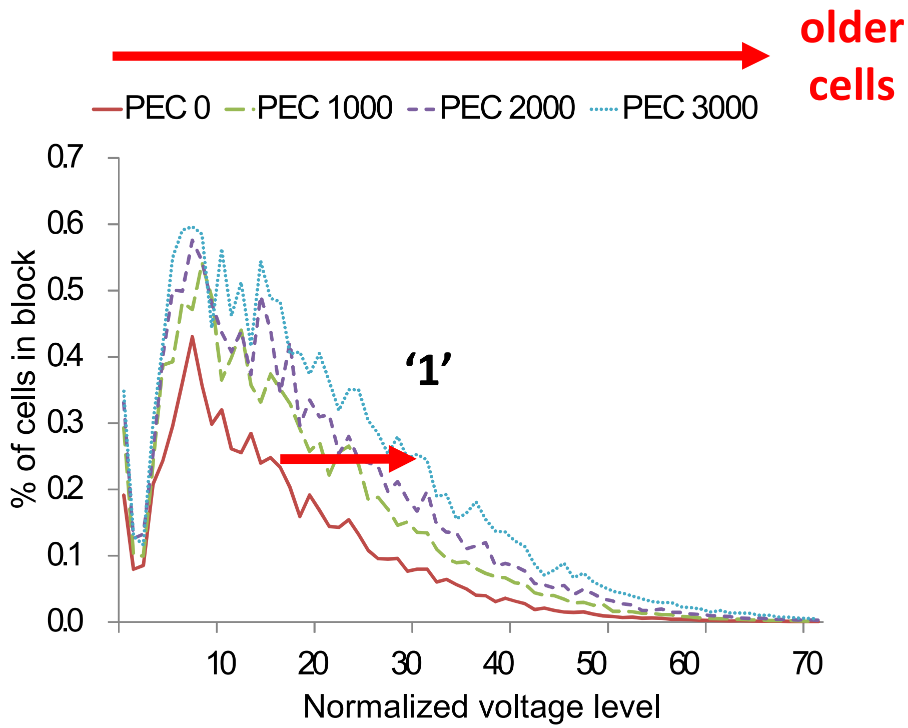
Interference increases variations

- Programming a cell partially charges neighboring cells
  - 20% of non-programmed cells positively charged



Wear-out adds more variations

- Cell degradation right-shifts distributions as more Program/Erase Cycles (PEC) applied



Outline

- Motivation
- Background
- How to hide
- Detectability
- Performance
- Conclusion

Threat model

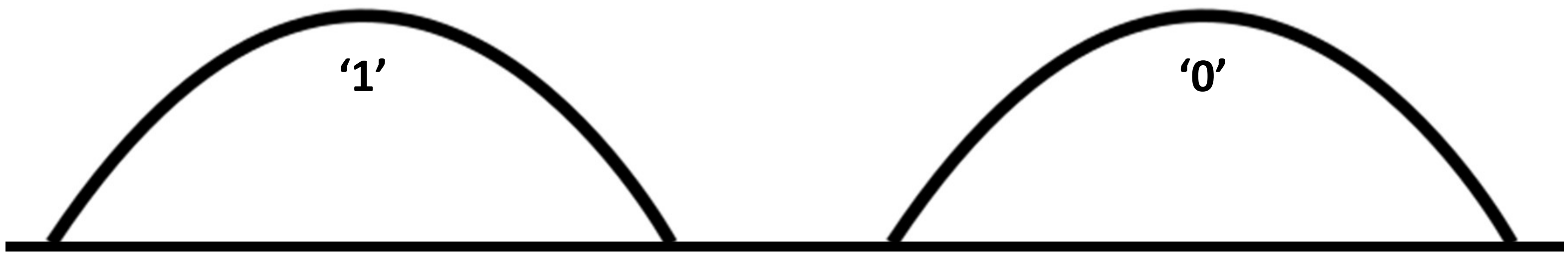
- User has "public" + secret key
  - Encrypts public data using "public" key
  - Secret key for hidden data w/plausible deniability!
- Adversary (e.g., NSA):
  - Confiscate device for inspection
  - Can probe visible data and voltage levels*

* Requires NDA with vendors

Storing a hidden bit in flash

Voltage-hide method

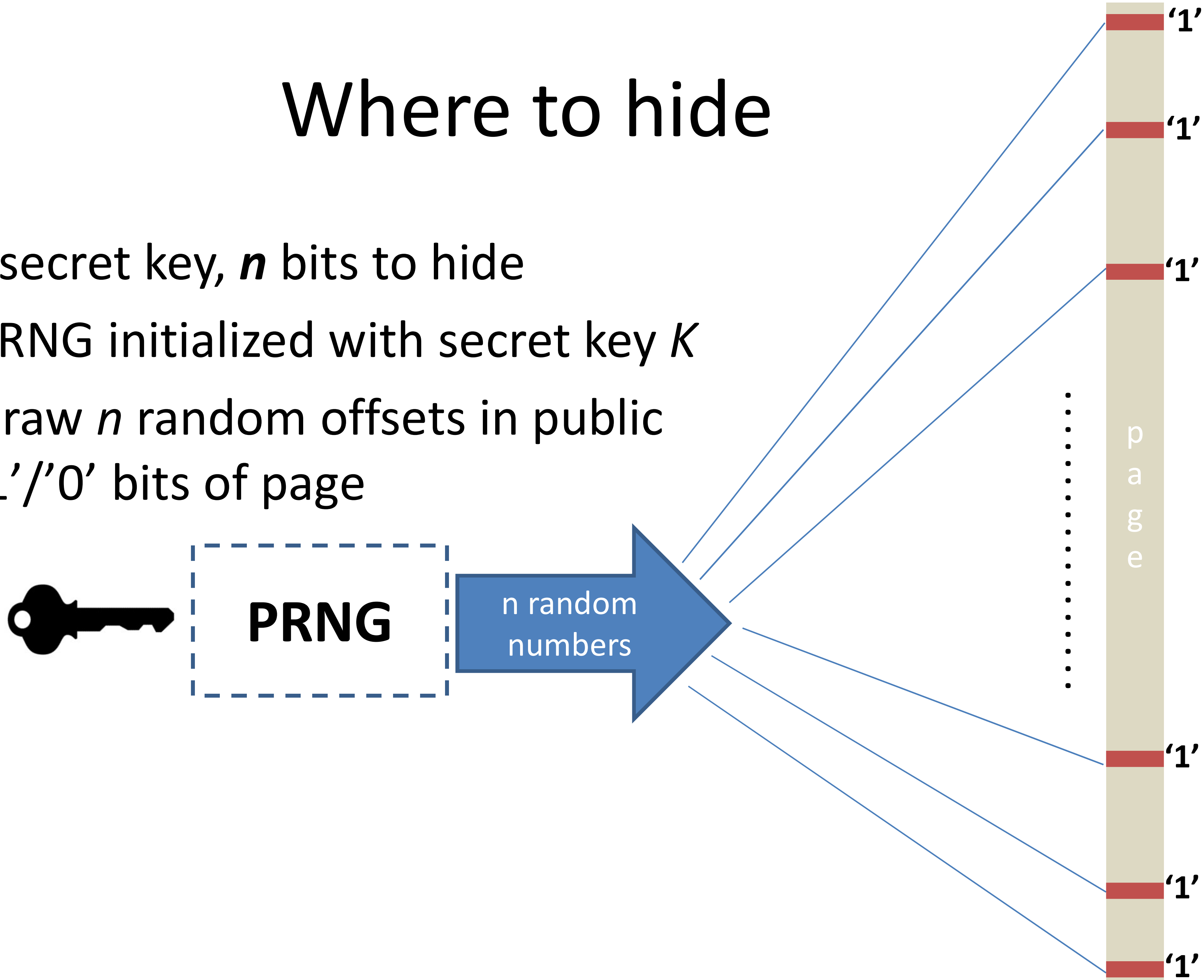
I. Store public data using coarse-grain programming



Where to hide

K = secret key, ***n*** bits to hide

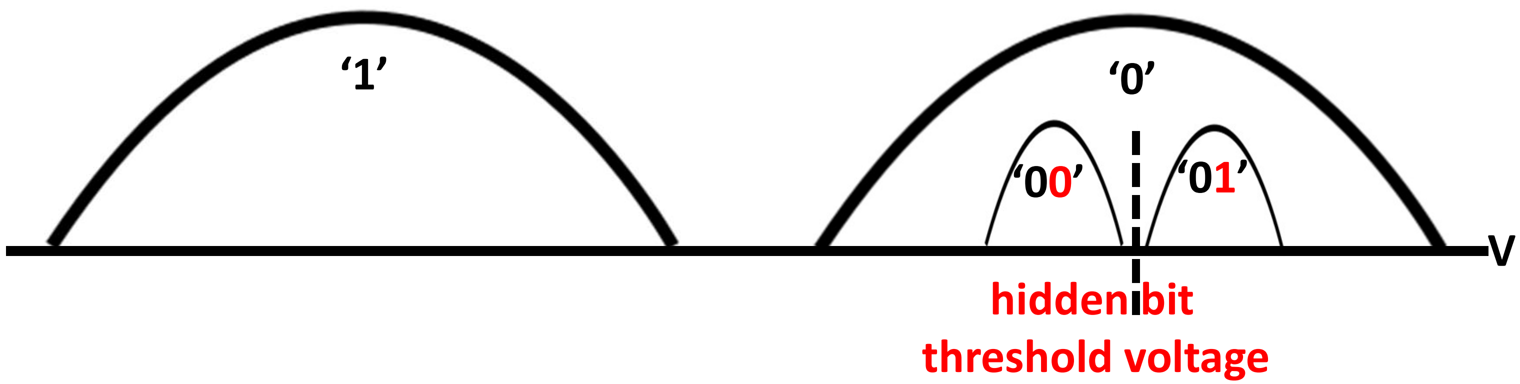
- PRNG initialized with secret key *K*
- Draw *n* random offsets in public '1'/'0' bits of page



How to hide

Voltage-hide method

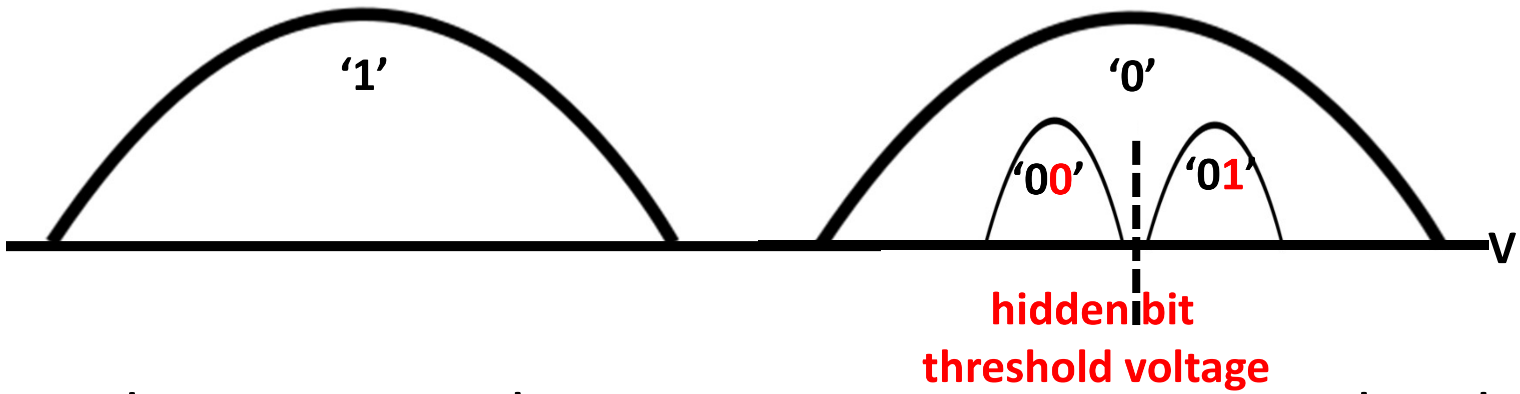
I. Store public data using coarse-grain programming
II. Select cells to store extra hidden bits (PRNG + secret key)
III. Store hidden data using fine-grain programming



How to hide

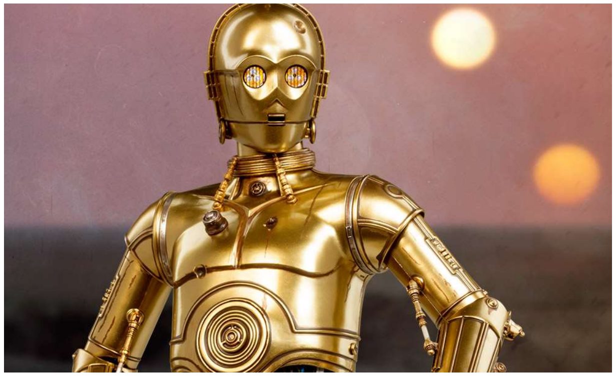
Voltage-hide method

I. Store public data using coarse-grain programming
II. Select cells to store extra hidden bits (PRNG + secret key)
III. Store hidden data using fine-grain programming



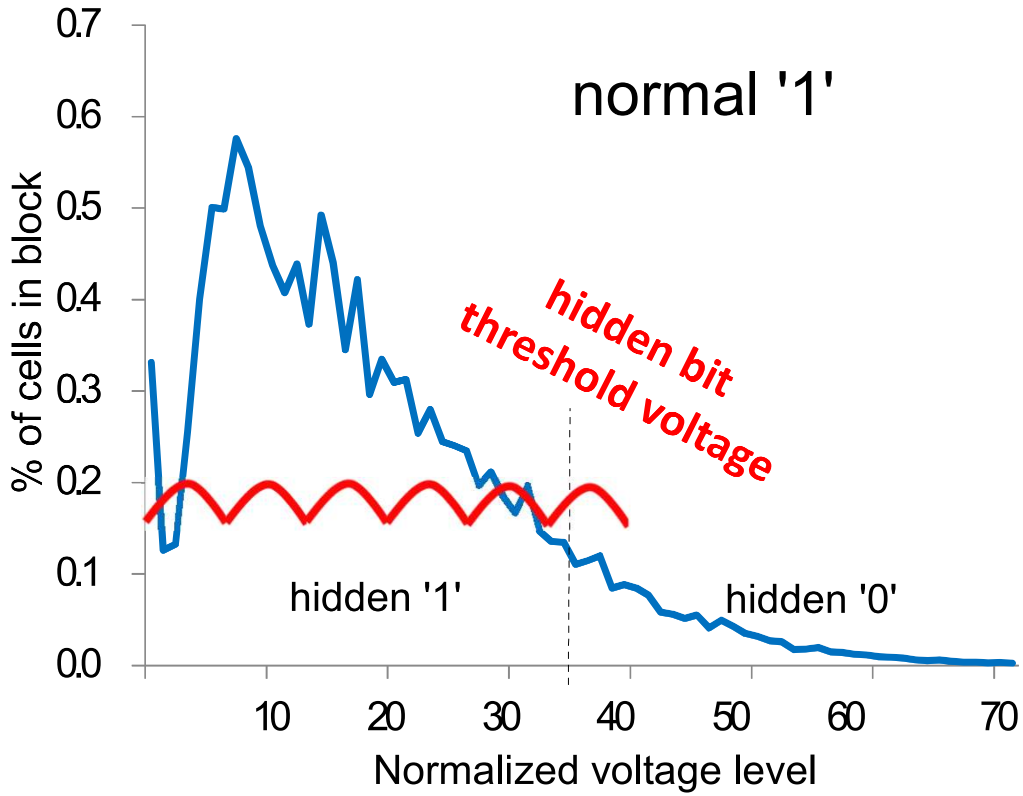
- Vendors can tweak programming accuracy on the chip!
  - Voltage-level distribution width
  - Target voltage
  - Threshold voltage

- Flash vendors: Control over low level features



- Us: Improvise by (very) crudely mimicking fine-grain programming

How to hide (cont.)

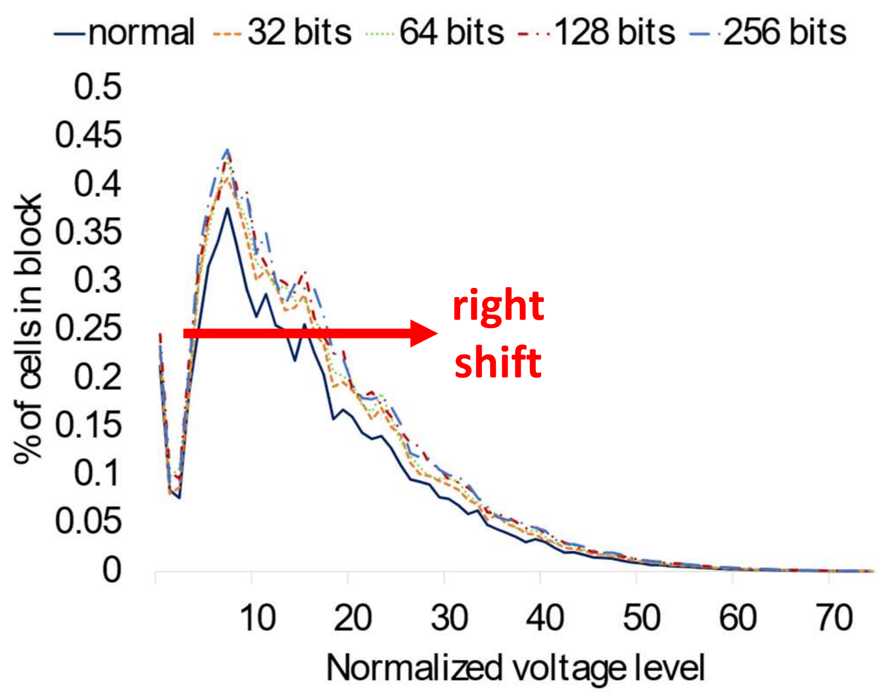


- Sequence of Partial-Programming (PP) steps (PROGRAM+ABORT)
- Hiding in programmed cells too slow & inaccurate
  → focus on non-programmed cells

- Vendors can implement our scheme in firmware
- We are not flash vendors
- We present an implementation on real hardware
  - Required vendor-specific voltage probing
  - Some limitations from inability to change firmware

Determining capacity

- Small number of non-programmed cells to manipulate (<1K)
  → **hide only 256 bits per page**
- Inherent limitation of not having vendor support



Outline

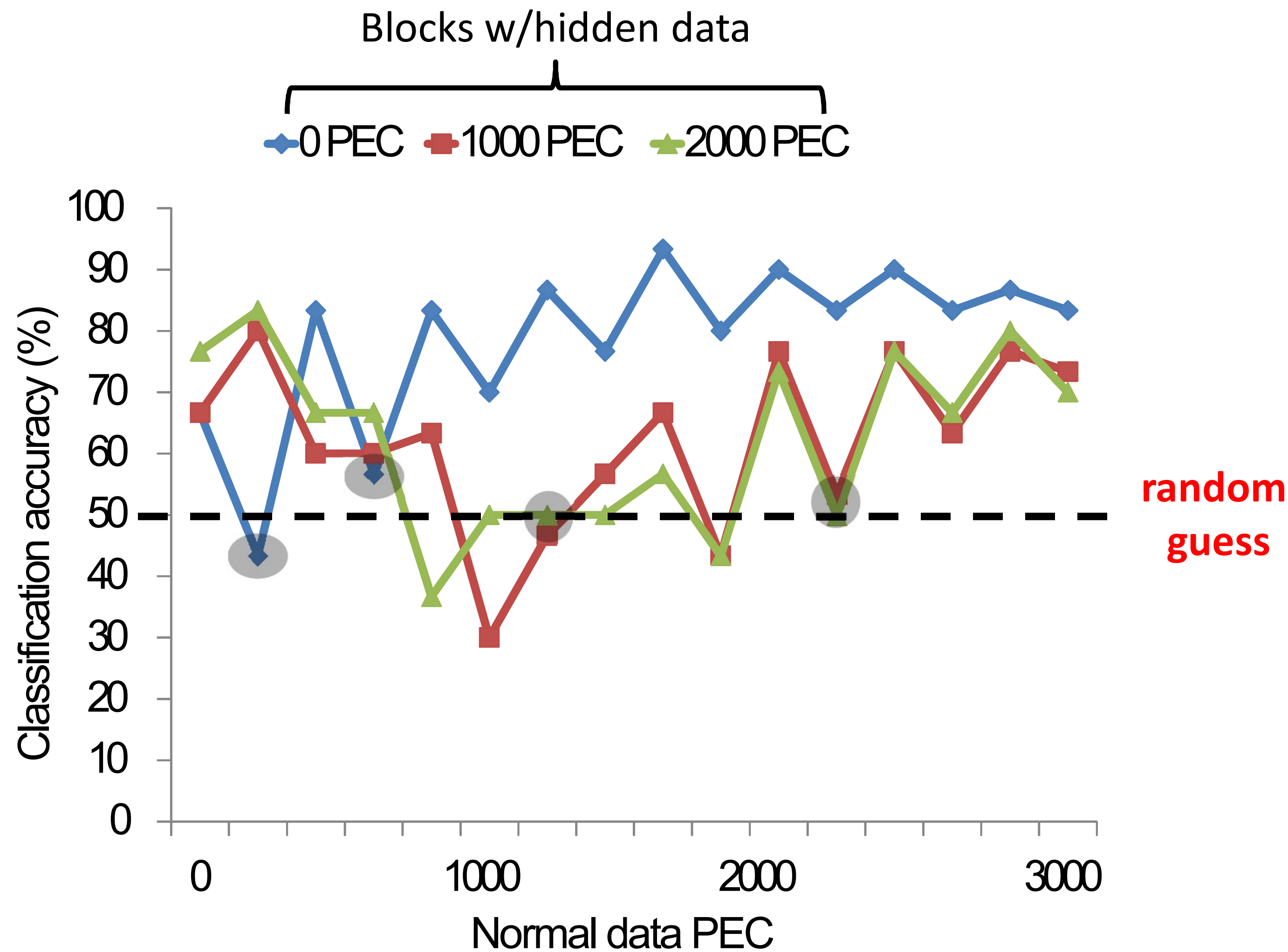
- Motivation
- Background
- How to hide
- Detectability
- Performance
- Conclusion

Basic idea

- Flash blocks with hidden data can be mistaken for “normal” blocks with similar age
- Voltage variations mistaken for naturally occurring ones (e.g., age, process variation)

Simulating adversary

- Apply Support Vector Machine (SVM) to voltage data
- Three chips with mix of public & hidden data
  - Train: two chips, know which pages have hidden data
  - Classify: 3rd chip: given voltages, has hidden data?
- Hidden and public data PEC vary
  - e.g., normal PEC 1000, hidden PEC 2000
  - Optimal for adversary!



- Works when hidden and normal data PEC are close enough

Outline

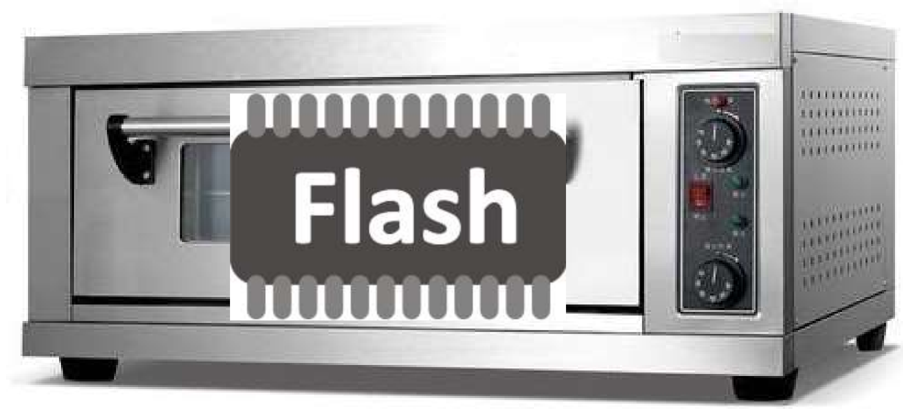
- Motivation
- Background
- How to hide
- Detectability
- Performance
- Conclusion

Metric	Our method	State of the art*	Why?
Encoding thr.	35 Kb/s	1.4 Kb/s	Fewer programming steps (10 vs. hundreds)
Latency (single bit)	6.9 ms	798 ms	
Energy	1,183 uJ	43,624 uJ	
Decoding thr.	2.7 Mb/s	54 Kb/s	Single read vs. dozens of programming steps → Reduced wear out!

* "Hiding information in flash memory", IEEE Symposium on Security and Privacy (SP) 2013

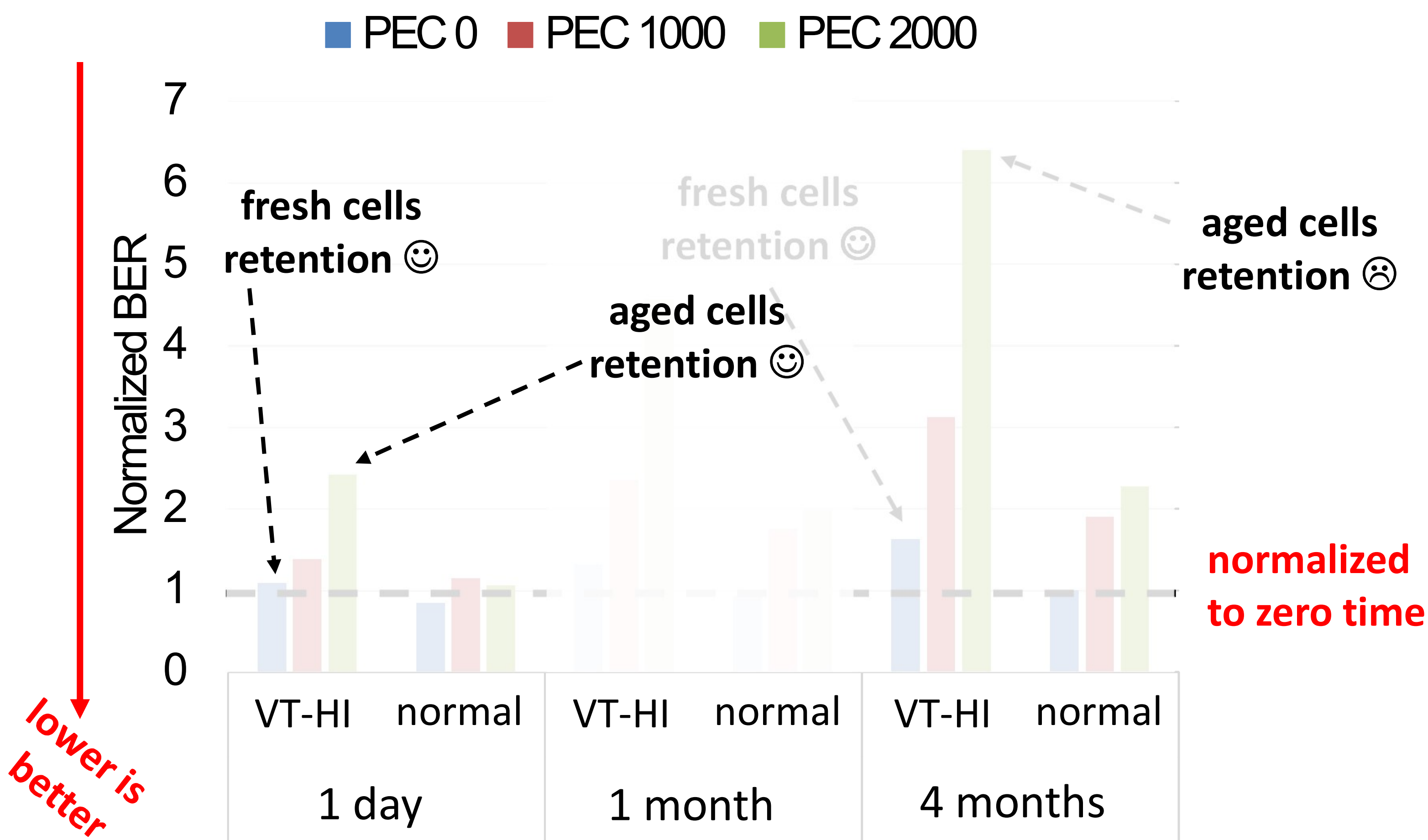
Reliability and retention

- Emulate different retention periods using standard techniques*
  - Bake flash chip in special oven



* Extended arrhenius law of time-to-breakdown of ultrathin gate oxides, APL'03

Reliability and retention (cont.)



- Over time need stronger ECC/refresh

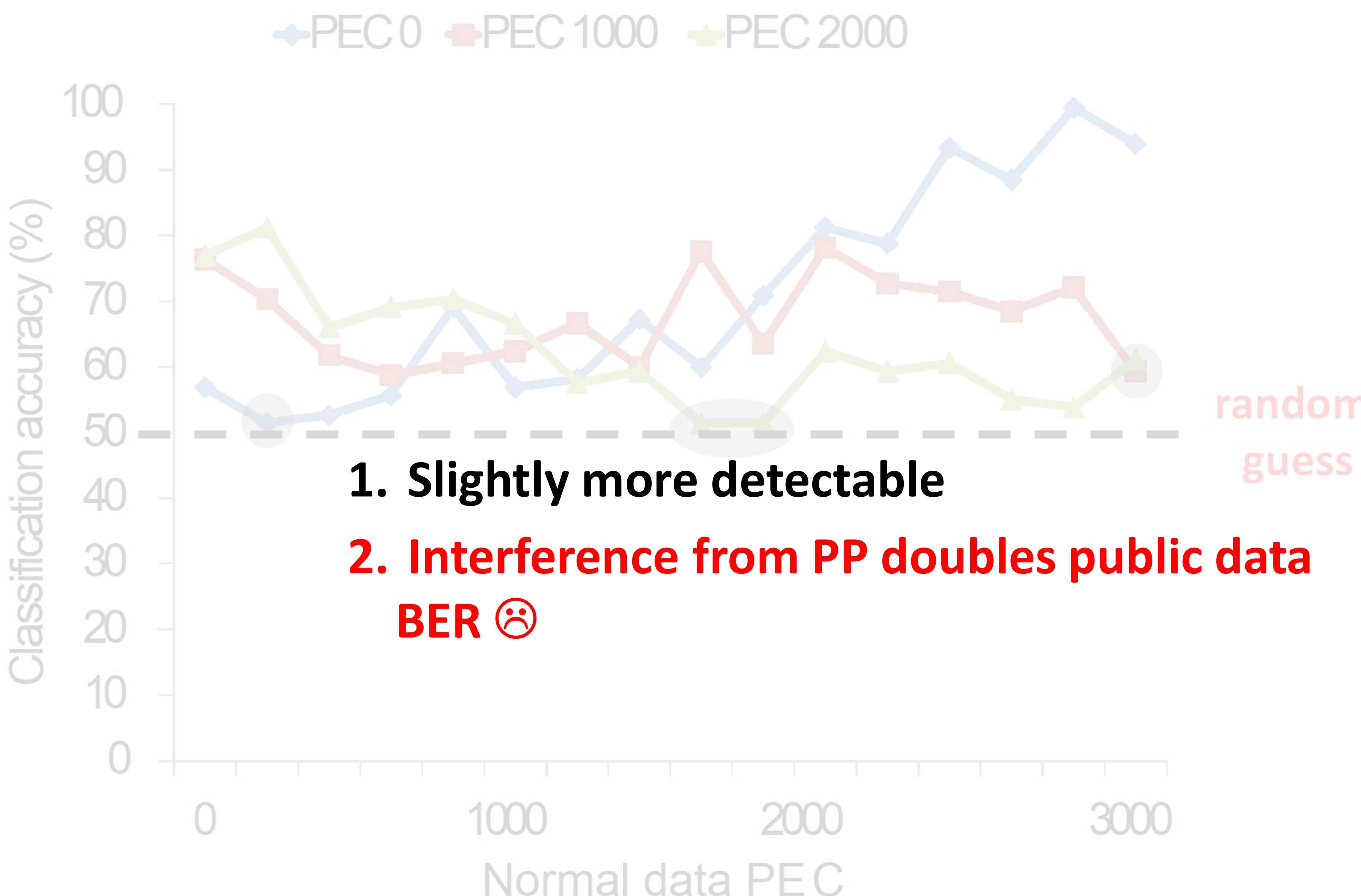
Reliability and retention (cont.)

- State of the art:
  - Similar BER for fresh cells
    (0.3% vs. 0.5% in VT-HI)
  - Unacceptable BER even for slightly aged cells
    (e.g., 12% BER for PEC 100)

Capacity

- So far mimicked fine-grain programming
  - Incremental PP
  - Bits per page: 256 vs. 1024 for state of the art ☹
- Lets simulate "what if" we had vendor support?
  - 10 PP → 1 PP
  - 256 bits x 10→ 2560 bits

How does hiding 10x more bits affect detectability?



1. **Slightly more detectable**
2. **Interference from PP doubles public data BER ☹**

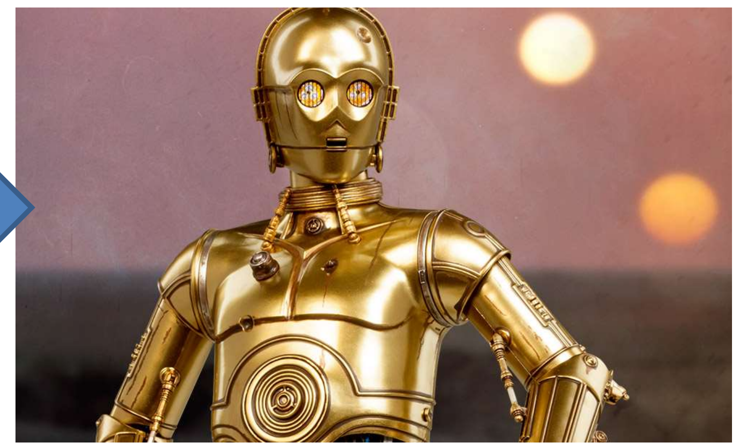
Vendor support (cont.)

- **Problems should be resolved with vendor support:**

  - Less interference, more accuracy

  - Can hide in <u>programmed</u> cells!

Low-capacity & suboptimal



High-capacity, Efficient & accurate

Conclusions

- We can hide data within natural voltage variations
  - Already common to increase flash densities
- Vs. State of art:
  - 24x and 50x faster encoding/decoding,
  - 37x more power efficient, and
  - less wear
- Capacity should improve with vendor support

Questions?