How to do Dilithium TVLA
(with Adams Bridge examples)

Markku-Juhani O. Saarinen
`<markku-juhani.saarinen@tuni.fi>`

Optimist Short Talk
March 27, 2025

Tampereen yliopisto
Tampere University

History: Test Vector Leakage Assessment

- TVLA identifies differences between two sets of side-channel measurements, such as power and traces. Does not recover secret keys etc.

- Typically used to for **positive assurance** – to demonstrate lack of leakage.

- TVLA was “Invented” at Cryptography Research Inc. (now Rambus)

  Proposed for FIPS 140: G. Goodwill, B. Jun, J. Jaffe, P. Rohatgi: *"A testing methodology for side-channel resistance validation."* CMVP & AIST Non-Invasive Attack Testing Workshop (NIAT 2011), September 2011

  https://csrc.nist.gov/csrc/media/events/non-invasive-attack-testing-workshop/documents/08_goodwill.pdf

- TVLA (and its standardized form ISO 17825) has been criticized for the non-detection of practical higher-order attacks, and also for statistical “experiment design” (this has improved – but standards have no PQC.)

Recall basic ISO 17825 “TVLA”

Outline of the General Statistical Test Procedure

0. Determine the required sample size $N = N_A + N_B$ and t-test threshold C from the experiment parameters.
1. Collect Subsets A and B and compute their pointwise averages (μ_A, μ_B) and standard deviations (σ_A, σ_B).
2. Compute the pointwise Welch t-test statistic vector

$$T = \frac{\mu_A - \mu_B}{\sqrt{\frac{\sigma_A^2}{N_A} + \frac{\sigma_B^2}{N_B}}}.$$

3. If at any point $|T| > C$, the test results in a FAIL. If the threshold was is not crossed, the test is a PASS.

I use two basic kinds of leakage assessments

Fixed vs Random (“FIX”) and A/B Classification (“ABC”)

1. **Fixed vs Random** (non-specific t-test) can be used in “live” testing:
   - Trace set A: Fixed CSP for every trace.
   - Trace set B: New random CSP secret for each trace.

2. **A/B Categorization** works with capture-then-analyze flow:
   - Records traces with detailed test vector metadata; CSPs are known in analysis.
   - Traces are categorized *after capture* to A and B sets based on CSP selection criteria,
     Examples: a specific internal CSP variable or secret key bit, “plaintext checking” bit.
   - The same trace data can be categorized to A and B in a number of different ways.

In both cases: Set A and Set B statistically differentiable with t-test = **FAIL**.

Tricky detail: Dilithium Secret Key TVLA

Not everything in the secret key is secret!

- The basic TVLA fix-vs-random is really only suitable for symmetric ciphers

- Dilithium secret key has six components, two of which are actually secret:

$$\mathbf{SK} = (\,\rho,\, K,\, tr,\, \mathbf{s_1},\, \mathbf{s_2},\, \mathbf{t_0}\,)$$

- The public parts, e.g. matrix A expansion from symmetric seed ρ do not need protection. So one can easily get false positives in fix-vs-random

- One creates the test vectors for TVLA so that the random set is not entirely random, but just bits of the secret key bits are varied between traces.

- Alternative: randomize fully and just fix some secret key bits.

Adams Bridge – One way to implement Dilithium

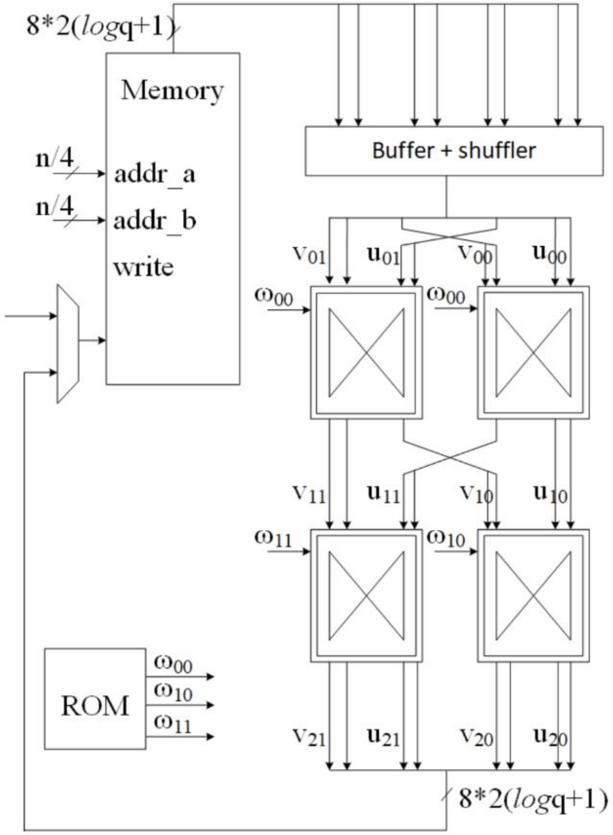
- **Status, Mar ‘25**: A standalone ML-DSA-87 accelerator, close to RTL freeze?
- Available, 100% SystemVerilog: https://github.com/chipsalliance/adams-bridge
- Only the “Category 5” parameters supported. Nothing related to Kyber visible.
- Self-contained module that does { KeyGen, Sign, Verify } from start to the finish. Includes a SHA3 module etc. Recently memory iface has been moved out.
- **Memory mapped (AHB)**: User writes keys, random, message (hash), sets trigger. Waits for status to become <ready> (perhaps intr), then read the signature out.
- **Very fast!** Verify: 20,000 cycles. / Sign: 160,000 cycles (40,000 per round).
- **Very big!** No shared components. Something like 400k GE + memories?

Protecting only things that *have been exploited*..

E. Karabulut, K. Upadhyayula, **"Side-Channel Countermeasures for the Adams Bridge Accelerator"**, 2024 OCP Global Summit

Masking in Adam's Bridge – NTT, PWM

- Some PWM and INTT operations in CRYSTAL-DILITHIUM must have strong countermeasures to protect secrets
- Shuffling is not strong enough
- Masked BFU with 2 shares per input
- 4x memory overhead
- >4x NTT area overhead
- 4x latency overhead
- Very strong countermeasure



ABR is not really masked (as we understand it)

- **Secret keys are not masked.**

  "Operations Protected with Masking: Point-wise multiplication and the first state of inverse NTT."

- **Key generation is not protected at all**.

  "The key generation operation does not have a non-profiled attack vector since its nature is inherently secure against CPA-style attacks. This is because non-profiled attacks require multiple traces captured while constant secret or private values are being processed."

  Dilithium may be used in a mode where secret keys are stored as short "seeds" and always expanded before use. Adams Bridge supports this..

Presilicon Testing of Current Version

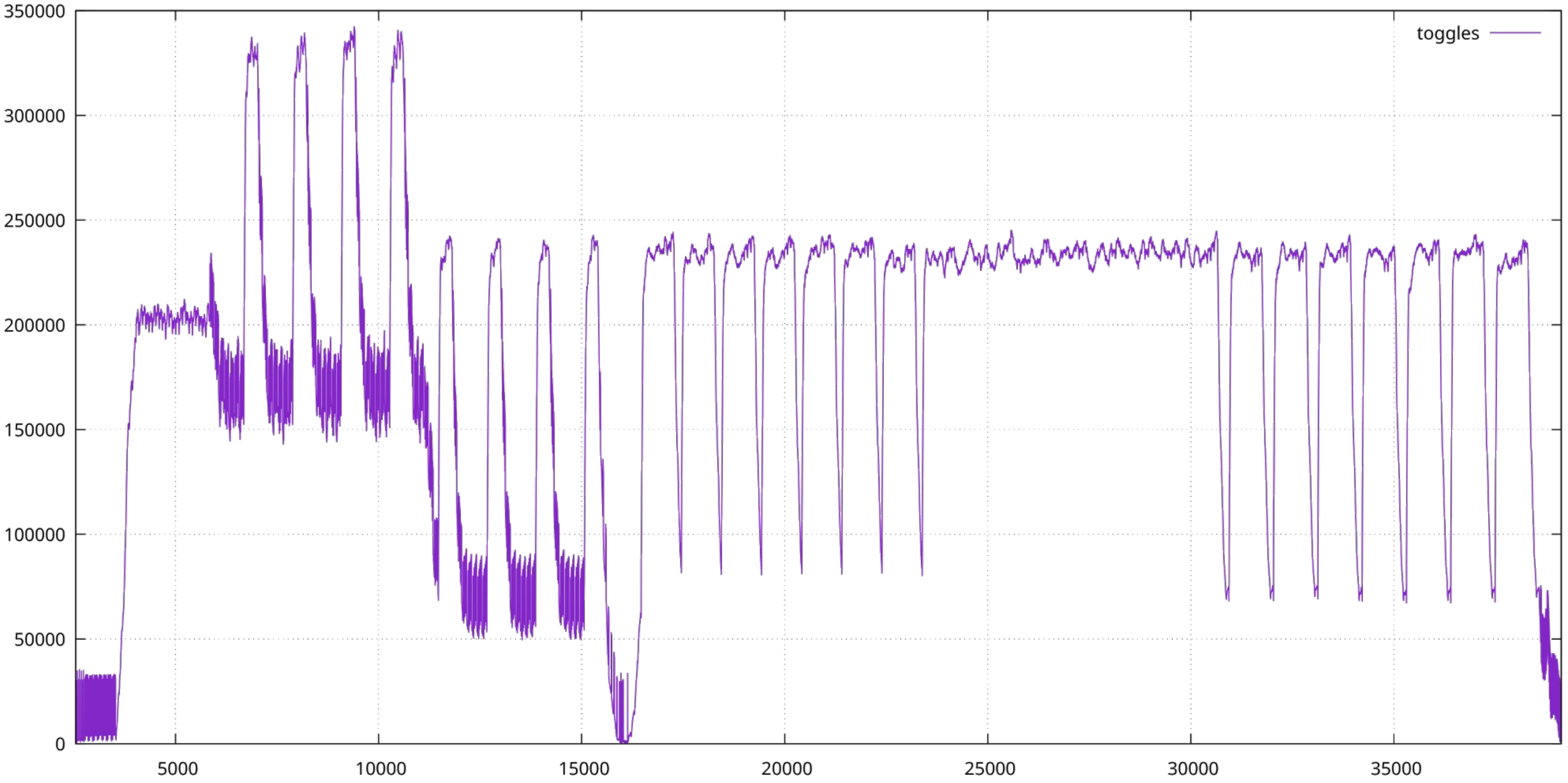
- Get VCD traces from verilator, DUT doing signing operations

- Presilicon VCD-to-Trace program reads VCD file, keeps track of all state bits and records Hamming distance for each clock cycle.

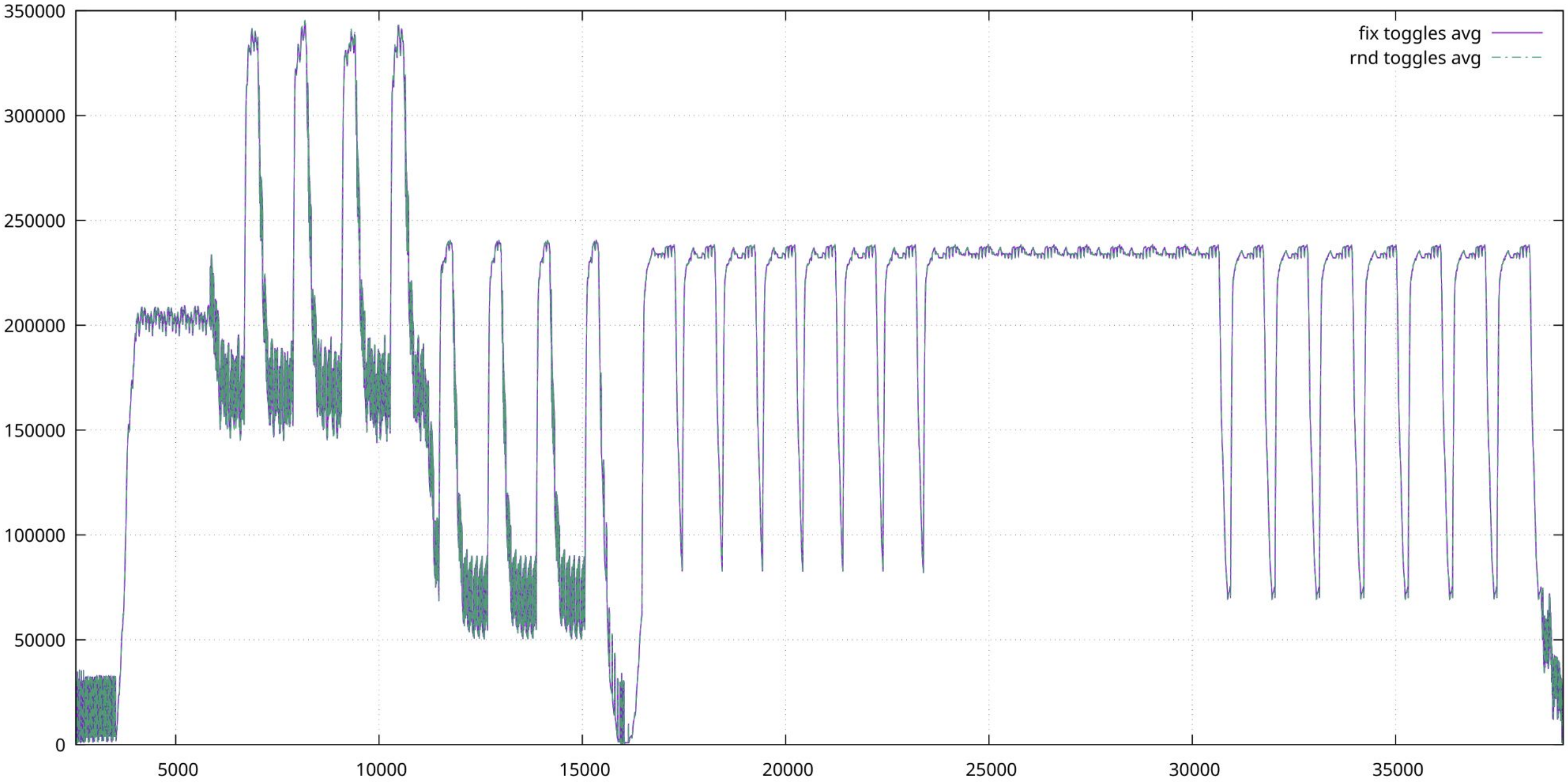
- Since the signal is very “clean”, not nearly as many traces are required than from FPGA-oscilloscope setup (rule-of-thumb, perhaps 10%).

- Very precise; we get exact cycle of leak points and can check (from VCD) the names of wires and signals that were active and causing it.

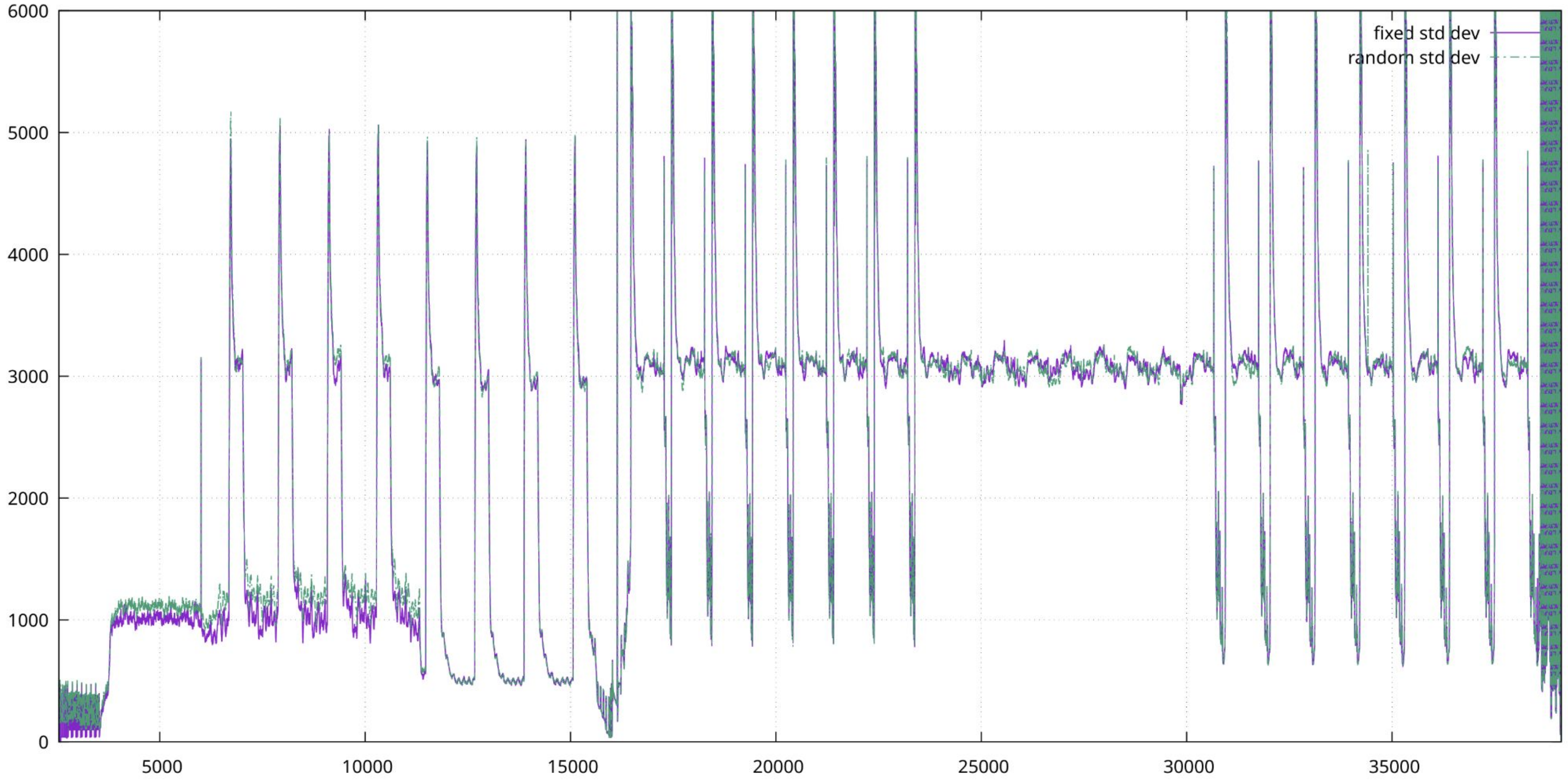
A Pre-Silicon "Toggle Trace" of ML-DSA Signing



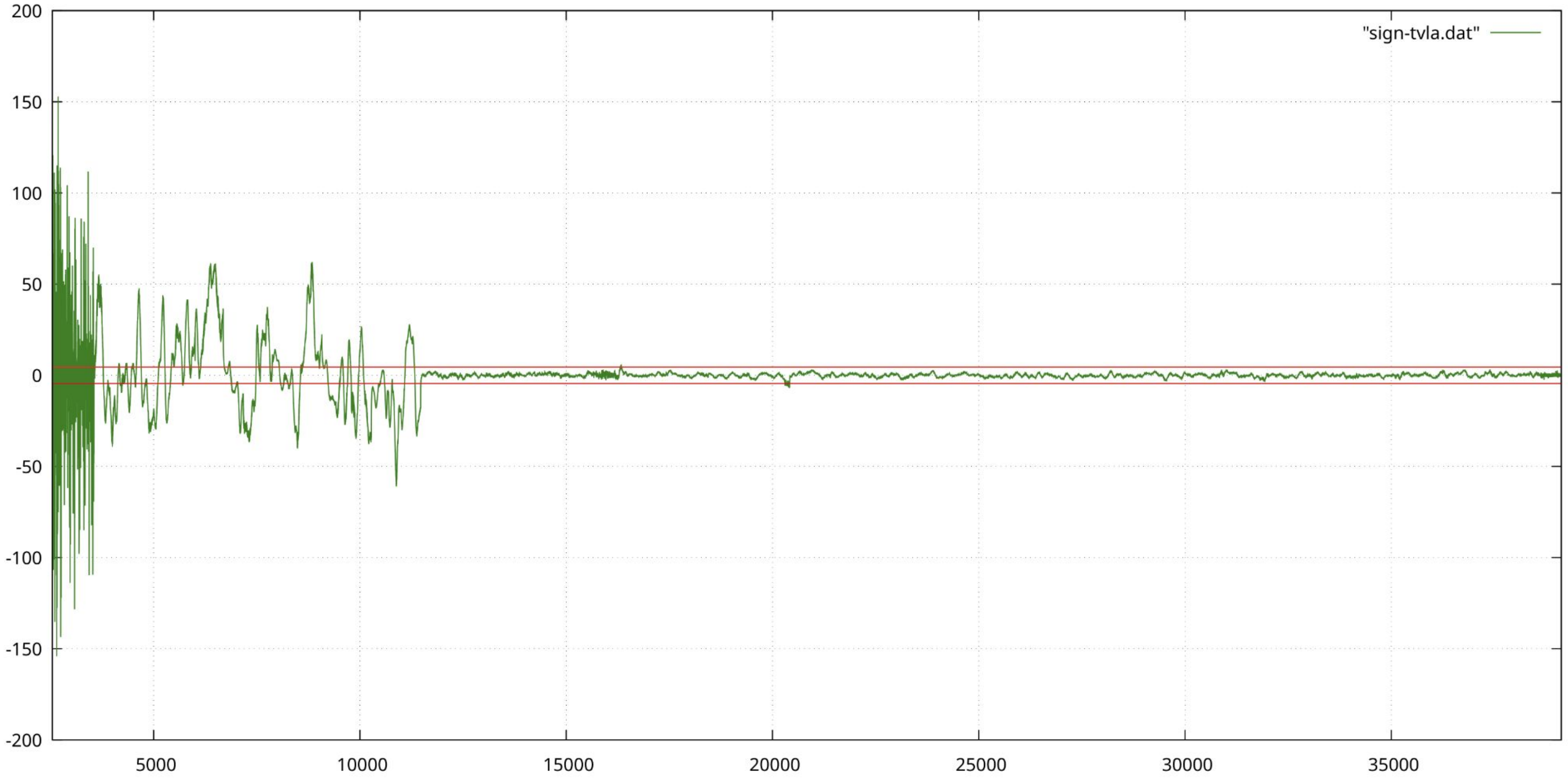
Fixed vs Random (3500 each) Averages overlap..



Standard Deviations lower for Fixed -> Leakage



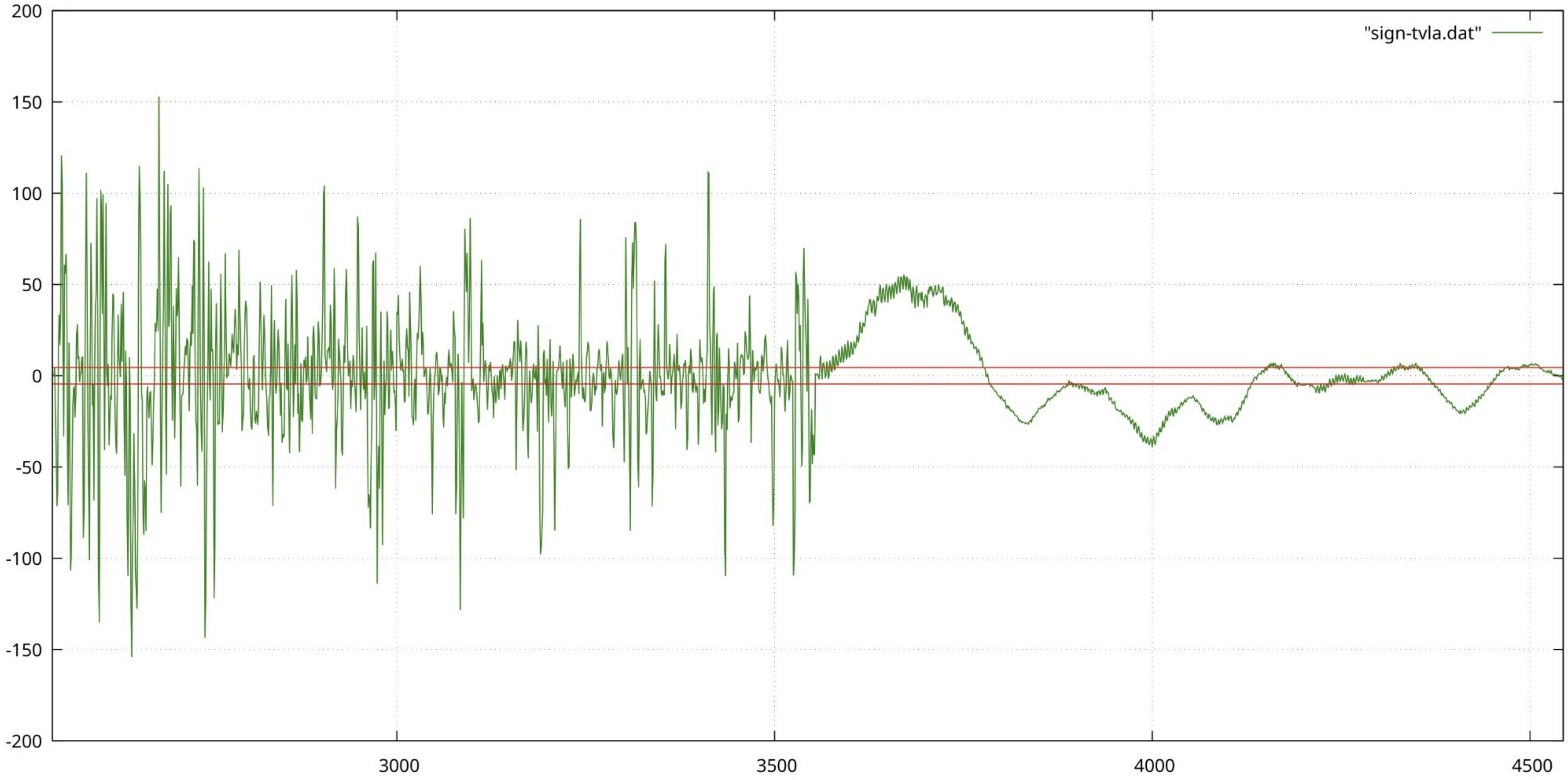
TVLA: 3500+3500 traces of ML-DSA Signing



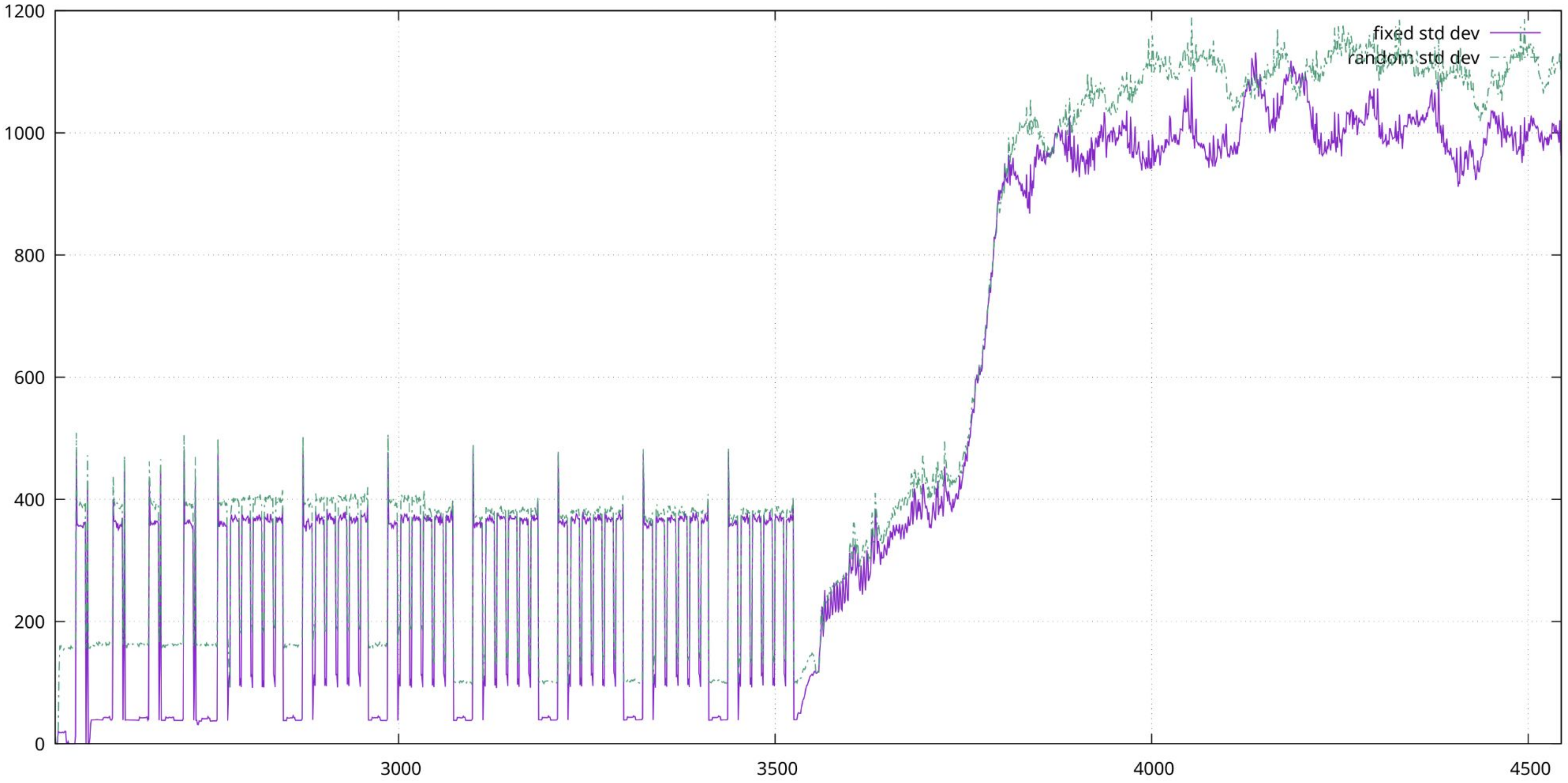
Checking Sequencer Program Counters..

Cycle	Sequencer Activities
2542	Start signature, compute hashes
2733	y=ExpandMask(ρ’ ,κ)
3553	y=ExpandMask(ρ’ ,κ), NTT(t), NTT(s1), NTT(s2)
5821	A ⟵ExpandA(ρ), Aˆ ◦NTT(y), NTT(t), NTT(s1), NTT(s2)
11097	Computing w ..
15397	Set y
16465	Validity checks..

Zoom into the leakage points



Zoom into the leakage points



Examining the leakage points

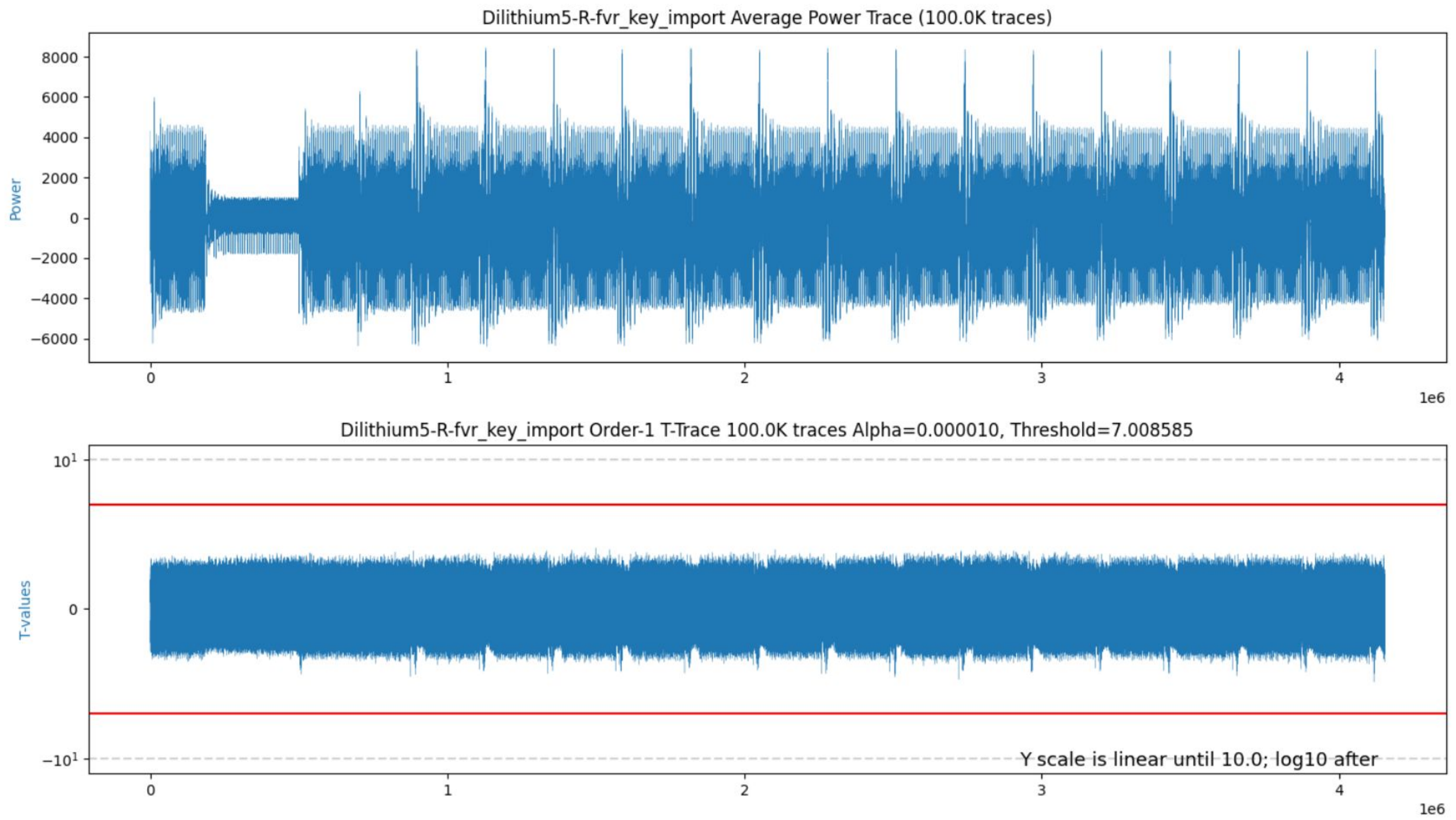
- **No surprise**: Leakage happens during early phases when the “plaintext” secret key is being moved about and transformed (NTT(s1), NTT(s2) ..)

- This would be automatically considered “broken” by the theory. However, leakage alone does not imply efficient key recovery or forgery attacks.

- Somewhat saved by wide data paths – large chunks are being moved in each cycle so one learns the total hamming weight or distance.

- Questions: **Where do the keys come from? How are they stored?**

Example 2: Dilithium5 WrapQ Key Import RvF CSP (#3)



On Dilithium Side-Channel Countermeasures

- Attack papers do not even claim describe all of the vulnerabilities, just what happened to be the low hanging fruit. One vulnerability is enough!

- Researchers know that many side-channel attacks work against Dilithium. Lattice countermeasure “theory” work has been going on for many years.

- I recommend taking a theoretically sound **masking approach** – must be complemented with other countermeasures, and adversarial analysis.

- Masking and other countermeasures **impact architecture**. Don’t try to somehow “patch” countermeasures into an unprotected implementation.