PB173 Domain specific development: side-channel analysis

Seminar 12: Presentation & Grading (Last Seminar)

Łukasz Chmielewski
chmiel@fi.muni.cz,

Consultation: A406 Monday 14:00-

CROCS
Centre for Research on Cryptography and Security

My recent work (published in 2023)

SoK: SCA-secure ECC in software – mission impossible?

Summary

“SoK: SCA-secure ECC in software – mission impossible?”

- Systematization of Knowledge (SoK) paper: a list of side-channel and fault-injection attacks that the implementation needs to withstand against
- The `sca25519` library consists of 3 implementations of X25519 for Cortex-M4 with countermeasures against all considered **side-channel** and **fault-injection** attacks.
  1. Unprotected implementation (constant-time);
  2. Implementation containing countermeasures required for ephemeral scalar multiplication;
  3. Implementation containing the most countermeasures for the static scalar multiplication.
- We also performed a side-channel evaluation including a single-trace template attacks feasibility assessment.

Unprotected Algorithm

Algorithm 1 The Montgomery ladder for x-coordinate-based X25519 scalar multiplication

Input: $k \in \{0, \ldots, 2^{255} - 1\}$ and the x-coord. x_P of a point P. **Output:** $x_{[k]P}$, the x-coordinate of $[k]P$.

$X_1 \leftarrow 1$; $Z_1 \leftarrow 0$; $X_2 \leftarrow x_P$; $Z_2 \leftarrow 1, p \leftarrow 0$

for $i \leftarrow 254$ downto 0 **do**

  $c \leftarrow [illegible] \oplus p$; $p \leftarrow k[i]$ ▷ $k[i]$ denotes bit i of k

  $(X_1, X_2) \leftarrow \mathsf{cswap}(X_1, X_2, c)$

  $(Z_1, Z_2) \leftarrow \mathsf{cswap}(Z_1, Z_2, c)$

  $(X_1, Z_1, X_2, Z_2) \leftarrow \mathsf{ladderstep}(x_P, X_1, Z_1, X_2, Z_2)$

return (X_1, Z_1)

Most protected Static Algorithm

Algorithm 3 Pseudocode of side-channel and fault-attack protected static X25519

Input: the x-coordinate x_P of point P. **Output:** $x_{[k]P}$.
Secure Input: a 64-bit blinding f, blinded scalar $k_{f-1} = k \cdot f^{-1} \mod l$, and blinding points $R, S = [-k]R$.

1: $ctr \leftarrow 0$; $x_P \xleftarrow{\$} \{0, \ldots, 2^{256} - 1\}$ ▷ Initialize iteration counter to 0 and output buffer to random bytes
2: Copy k_f to internal state while increasing ctr. ▷ Updating ctr in a loop protects copying against FI
3: $y_P \leftarrow$ ycompute(x_P)
4: Increase(ctr)
5: $(X_P, Y_P, Z_P) \leftarrow$ ecadd$((x_P, y_P), R)$ ▷ Point blinding, output of addition of R is projective
6: $(X_P, Y_P, Z_P) \leftarrow$ ecdouble(ecdouble(ecdouble$((X_P, Y_P, Z_P))))$ ▷ 3 doublings to multiply by co-factor 8
7: $r \xleftarrow{\$} \{1, \ldots, 2^{64} - 1\}$ ▷ Sample 64-bit non-zero random value for scalar blinding
8: $b \xleftarrow{\$} \{0, \ldots, \ell\}$ ▷ Sample blinding factor of non-constant-time inversion
9: $t \leftarrow r \cdot b \mod \ell$ ▷ Invert using extended binary gcd
10: $s \leftarrow t^{-1} \cdot b \mod \ell$ ▷ Unblind result of inversion
11: $k'_{f-1} \leftarrow k_{f-1} \cdot s \mod l$ ▷ Multiplicatively blind scalar k_{f-1}
12: $k' \leftarrow k'_{f-1} \cdot f \mod l$ ▷ Multiplicatively unblind scalar k'_{f-1} with f
13: Increase(ctr)
14: $x_P \leftarrow X_P \cdot Z_P^{-1}$; $y_P \leftarrow Y_P \cdot Z_P^{-1}$ ▷ Return to affine x and y coordinates

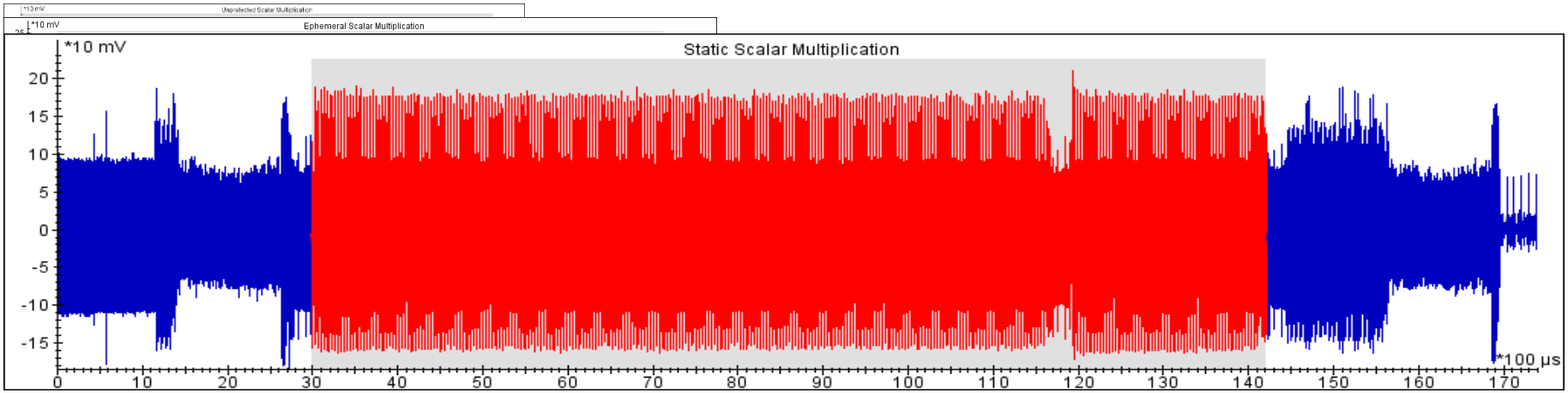
Not REALLY Readable

34: $r \leftarrow r \oplus a'$ ▷ Mask the random scalar r
35: Increase(ctr)
36: $(X_1, Z_1, X_2, Z_2) \leftarrow$ cswaprr $(X_1, Z_1, X_2, Z_2, a'[64])$ ▷ Projective re-rand.+cswap based on masking a'
37: **for** i from 64 downto 0 **do** ▷ Scalar multiplication by r
38: $(X_1, Z_1, X_2, Z_2) \leftarrow$ cswaprr $(X_1, Z_1, X_2, Z_2, r[i])$ ▷ Projective re-rand.+cswap based on masked r
39: **if** $i \geq 1$ **then**
40: $(X_1, Z_1, X_2, Z_2) \leftarrow$ ladderstep $(x_P, X_1, Z_1, X_2, Z_2)$
41: $(X_1, Z_1, X_2, Z_2) \leftarrow$ cswaprr $(X_1, Z_1, X_2, Z_2, a'[i-1])$ ▷ Projective re-rand.+cswap based on a'
42: Increase(ctr)
43: $Y_2 \leftarrow$ yrecover$(X_1, Z_1, X_2, Z_2, x_P, y_P)$
44: $(X_2, Y_2, Z_2) \leftarrow$ ecadd$((X_2, Y_2, Z_2), S)$ ▷ Remove point blinding, add in $S = [-k]R$
45: $x_P \leftarrow X_2 \cdot Z_2^{-1}$
46: Increase(ctr)
47: **if** ! Verify(ctr) **then** ▷ Detected wrong flow, including iteration count
48: $x_P \xleftarrow{\$} \{0, \ldots, 2^{256} - 1\}$ ▷ Set output buffer to random bytes
49: Update(R, S) ▷ 2 double-and-add scalar multiplications with the same 8-bit random scalar for R and S
50: Randomize(k_{f-1}, f) ▷ Generate new 64-bit random value f, securely compute f^{-1} and update k_{f-1}
51: Save(R, S, k_{f-1}, f)
52: **return** x_P

Static Algorithm (Extra) Countermeasures

1. Stored key blinding/randomization.
   - k is stored as $k \cdot f^{-1}$ together with f, which is a non-zero 64-bit random factor.
2. Point blinding
   - Static random points R and S for input blinding, where S = [−k]R.
     - Add R, perform scalar multiplication, and subtract S.
   - Re-randomizing $k \cdot f^{-1}$, f, R, and S; these secrets should be securely stored.
3. Inversion is blinded.
   - We use the extended Euclidean Algorithm (EEA) with multiplicative blinding.

Efficiency Analysis: price to pay for security (Plain ➔ Ephemeral ➔ Static)



- The above SCA traces are from a device running at 168MHz.

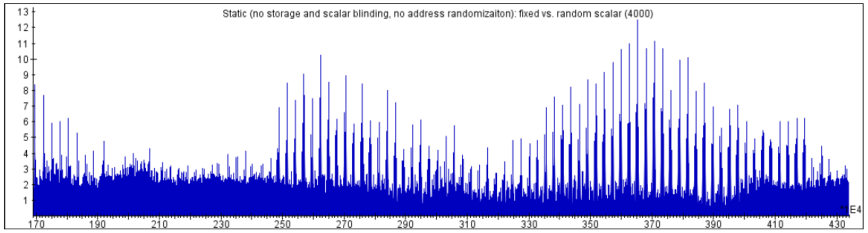
Efficiency Analysis: Related Crypto Libraries.

Implementation	Cycles:
Complete unprotected [Len20]:	548 873
Our unprotected baseline [HL19]:	680 097
Complete ephemeral:	932 204
Complete static:	2 338 681

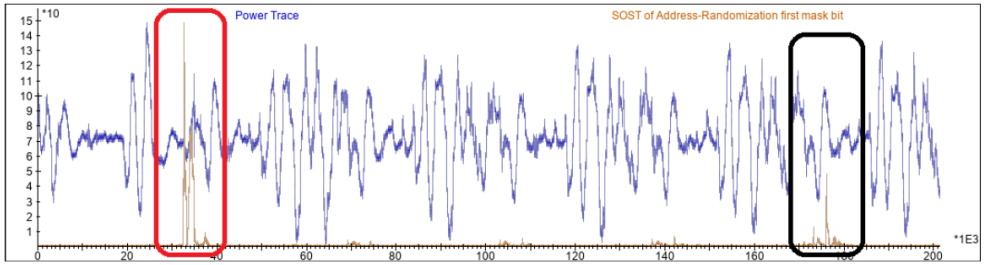
Library	Const. Time	Cycles:
wolfSSL [wol21] size:	yes	45 930 947
wolfSSL [wol21] speed:	yes	1 974 047
bearSSL i31 [Tho18]:	yes	2 576 639
BoringSSL [Goo21]: fixed base	yes	1 591 407
BoringSSL [Goo21]: var. base	yes	2 516 476
Arm MBed TLS [Arm21]	no	6 438 233

Side-Channel Evaluation

- TVLA: fixed vs. random input and scalar
- Preliminary results on Single-Trace Template Attacks
  - Profiled Device with turned-off countermeasures:



  - Find POIs for single-trace attacks:
  - Perform template attack
    - address mask: 63-64%,
    - masked scalar bit: 50.5%,
    - plain scalar bit: 52%



Links

- Resources:
  - https://tches.iacr.org/index.php/TCHES/article/view/9962 plus artifact
  - https://eprint.iacr.org/2021/1003
- Repository: https://github.com/sca-secure-library-sca25519/sca25519

ORGANIZATIONAL

Organization

- Group 1: Alignment
  - https://github.com/2lol555/pb173-side-channel/tree/main
- Group 2: Parallel computations with acquisition
  - https://github.com/makuga01/pb173-sidechannels

Remaining Seminars Plan

- 7: evaluation of progress on first steps: 1 point per person per work done till today also based on the commits in GIT
- **8:** evaluation of finished first steps : 3 points per group (personalized per person based on the Github) + giving the next tasks
  9: work in progress
- 10: 4 points per group (personalized per person based on the GitHub)
  This seminar: real SCA setup
- 11/12: national holiday / online consultation
- 13: final **2** points for work + 2 points for presentations + 2 points for activity, grading.

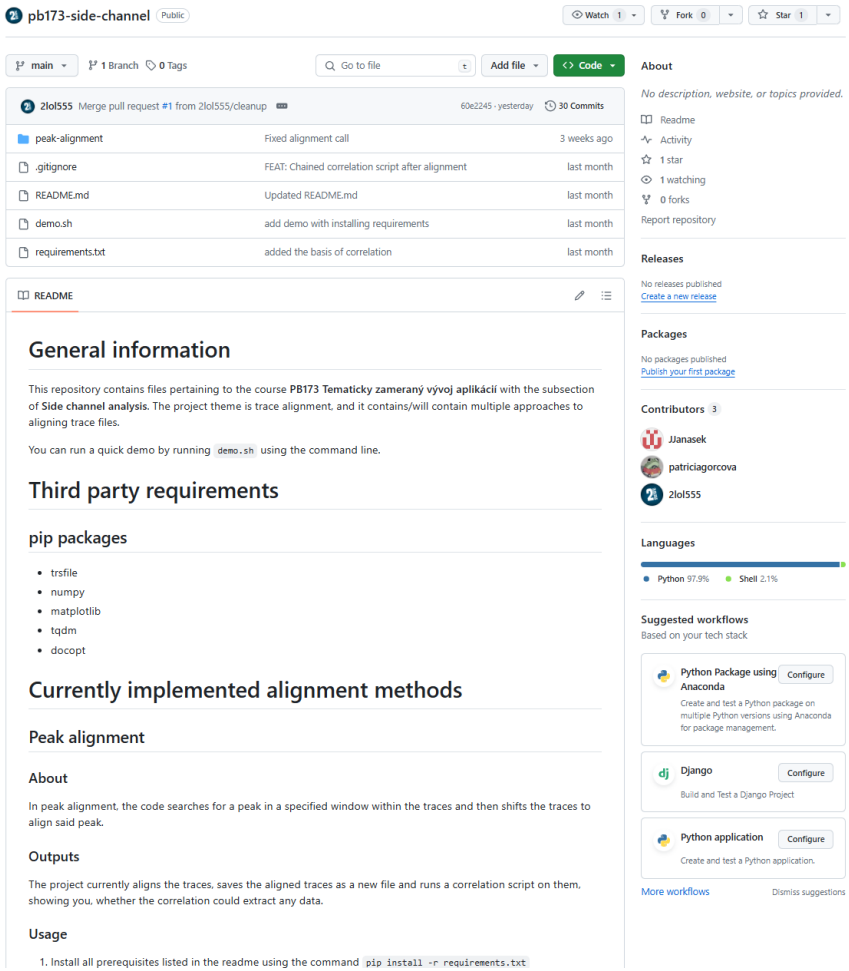
FINAL SEMINAR + GRADING

Group 1: Alignment

- Installation easy
- Nice!

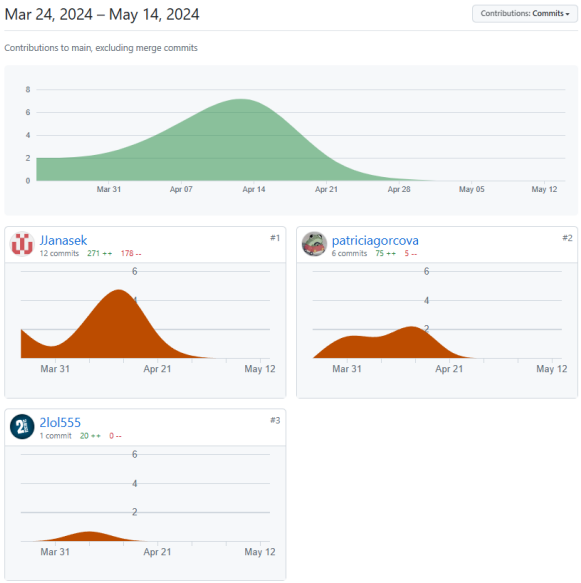
- Running hard
- No examples

- Did you push your recent changes to main?

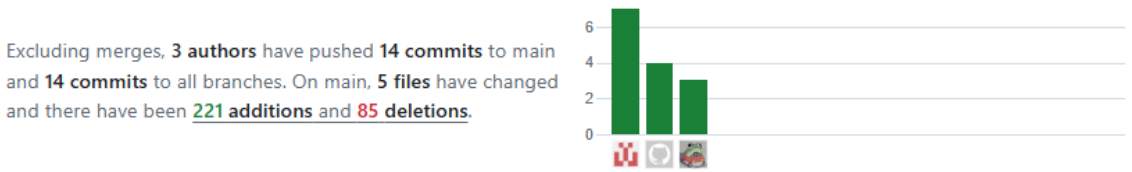


Group 1: Alignment

- Main:



- Commits:



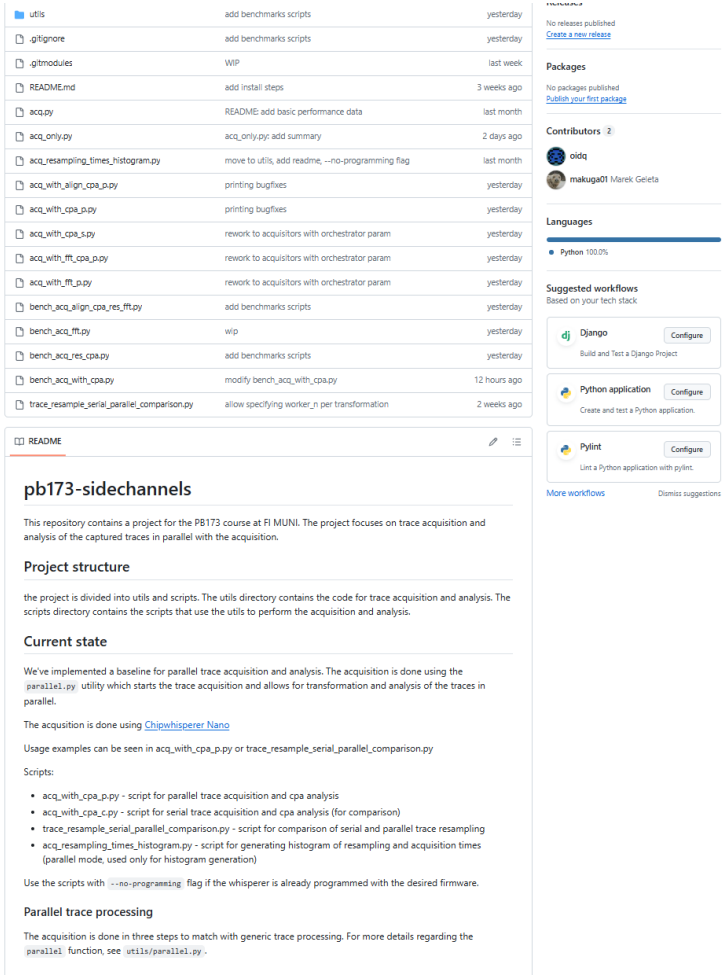
- Explain work division, etc., recent inactivity in main

Group 1 Final Tasks:

1. Finalize Correlation Alignment on the provided traces.
   - Potentially: investigate optimizations of calculating Normalized Cross Correlation (NCC) between the static reference and target traces. **Lukasz's idea:** find out how it is efficiently done at https://github.com/Riscure/Jlsca. There is an efficient implementation there!
2. Make Peak Correlation + Window Resampling work also for other trace sets:
   - Before 02/05/2024 I will upload two new tracesets to IS.
3. Help Group 2 to incorporate peak alignment into their acquisition pipeline.

Group 2: Parallel computations with acquisition

- Nice!
- Installation looks clear and nice
- I have not tried running, @Milan?
- It seems clear which script to run, but not completely clear with which target



utils	add benchmarks scripts	yesterday
.gitignore	add benchmarks scripts	yesterday
.gitmodules	WIP	last week
README.md	add install steps	3 weeks ago
acq.py	README: add basic performance data	last month
acq_only.py	acq_only.py: add summary	2 days ago
acq_resampling_times_histogram.py	move to utils, add readme, --no-programming flag	last month
acq_with_align_cpa_p.py	printing bugfixes	yesterday
acq_with_cpa_p.py	printing bugfixes	yesterday
acq_with_cpa_s.py	rework to acquisitors with orchestrator param	yesterday
acq_with_fft_cpa_p.py	rework to acquisitors with orchestrator param	yesterday
acq_with_fft_p.py	rework to acquisitors with orchestrator param	yesterday
bench_acq_align_cpa_res_fft.py	add benchmarks scripts	yesterday
bench_acq_fft.py	wip	yesterday
bench_acq_res_cpa.py	add benchmarks scripts	yesterday
bench_acq_with_cpa.py	modify bench_acq_with_cpa.py	12 hours ago
trace_resample_serial_parallel_comparison.py	allow specifying worker_n per transformation	2 weeks ago

README

pb173-sidechannels

This repository contains a project for the PB173 course at FI MUNI. The project focuses on trace acquisition and analysis of the captured traces in parallel with the acquisition.

Project structure

the project is divided into utils and scripts. The utils directory contains the code for trace acquisition and analysis. The scripts directory contains the scripts that use the utils to perform the acquisition and analysis.

Current state

We've implemented a baseline for parallel trace acquisition and analysis. The acquisition is done using the `parallel.py` utility which starts the trace acquisition and allows for transformation and analysis of the traces in parallel.

The acqusition is done using Chipwhisperer Nano

Usage examples can be seen in acq_with_cpa_p.py or trace_resample_serial_parallel_comparison.py

Scripts:

- acq_with_cpa_p.py - script for parallel trace acquisition and cpa analysis
- acq_with_cpa_c.py - script for serial trace acquisition and cpa analysis (for comparison)
- trace_resample_serial_parallel_comparison.py - script for comparison of serial and parallel trace resampling
- acq_resampling_times_histogram.py - script for generating histogram of resampling and acquisition times (parallel mode, used only for histogram generation)

Use the scripts with `--no-programming` flag if the whisperer is already programmed with the desired firmware.

Parallel trace processing

The acquisition is done in three steps to match with generic trace processing. For more details regarding the `parallel` function, see `utils/parallel.py`.

No releases published
Create a new release

Packages
No packages published
Publish your first package

Contributors 2
oidq
makuga01 Marek Gelela

Languages
Python 100.0%

Suggested workflows
Based on your tech stack

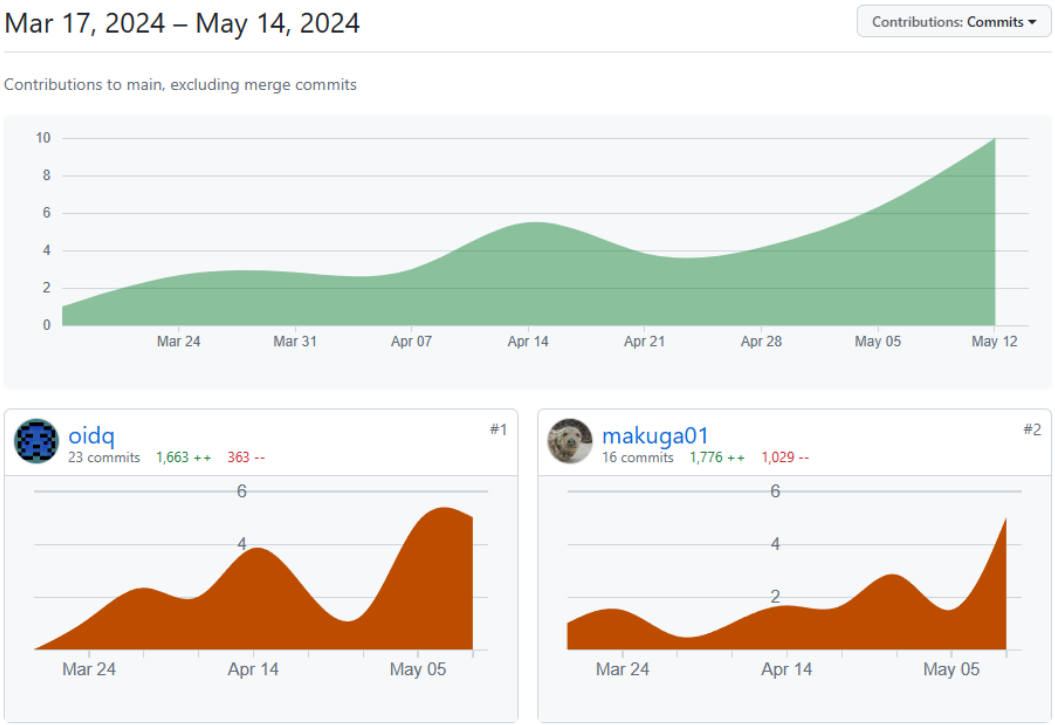
Django — Configure
Build and Test a Django Project

Python application — Configure
Create and test a Python application.

Pylint — Configure
Lint a Python application with pylint.

More workflows Dismiss suggestions

Group 2: Parallel computations acquisition



It looks evenly distributed, but please describe the division.

Group 2 Final Tasks:

1. Finish comparison of various settings with respect to the number of threads and the amount of traces acquired.
   - Clarify which approach is the best on your system.
   - Possibly use a profiler (e.g., `cProfile`) to identify the most important bottlenecks of your solution.
   - Experiment with various numbers of samples used (or acquired). Does it matter?
2. Add a peak alignment code from Group 1 to your pipeline and perform experiments.
3. **Optional:** add bandpass filtering to your pipeline: https://stackoverflow.com/questions/12093594/how-to-implement-band-pass-butterworth-filter-with-scipy-signal-butter

Reminder: Colloquium

- To get the colloquium
  - You must be present at seminars (2 absences OK)
  - You must be active at seminars (+2 points given by me at the end)
  - **You must submit and get:**
    - **50%: 7 points in total**

      (projects + presentation + activity = 14 points)

SUMMARY & PRESENTATIONS
THE FLOOR IS YOURS ☺

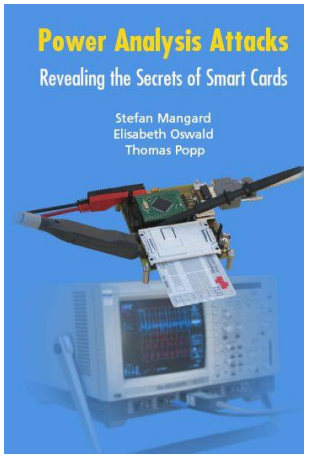
WRAPPING UP

Future Work (for me)

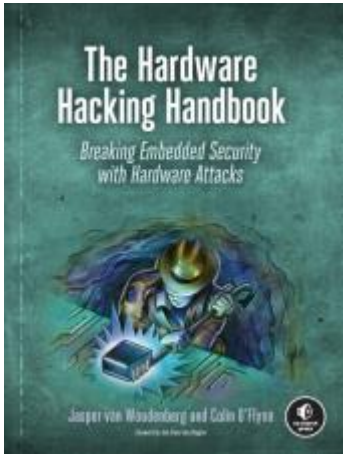
- I will try to run your code myself after the seminar and that will influence the grade a bit too.
- @ALL: Thank you for your hard work and participation!
- I potentiall would like to use your code in the future to help in the next year's seminars.
  - Would that be ok with you?

Still Future Reading

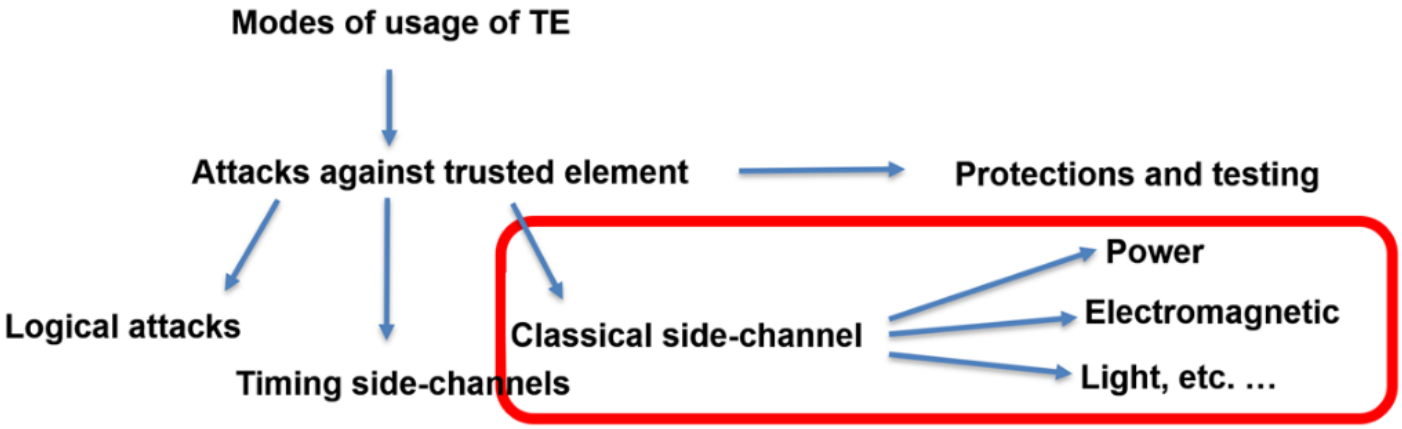
- For interested people
- Side-Channel Analysis – blue book:
  - http://dpabook.iaik.tugraz.at/
  - The books is available at the uni.
  - Look online



- The Hardware Hacking Handbook:
  - https://nostarch.com/hardwarehacking
  - I have an epub version.



Future Subjects



- PV080 (Information security and cryptography), PV079 (Applied Cryptography), PA018 (Advanced Topics in Information Technology Security)
- * PV181 (Laboratory of security and applied cryptography)
- * PV286/PA193 (Secure coding principles and practices)
- PV204 (Security Technologies)
- + Bachelor / Master (or even PhD) theses

Thank you very much for attending and for your work!!!

Questions