Table 1: Completion time, number of covered misuse cases, and correctness for each task, as well as for all tasks. Highlighted cells indicate analysis we replicated of the authors (dark gray) or of the aggregated analysis (light gray).

		Tasks						
Variable	Pattern?	B	C	D	E	F	G	All
Time [h]	Not present	0.78	0.89	0.88	0.81	0.91	0.53	2.42
	Present	0.92	1.06	1.09	0.87	1.02	0.69	2.81
U/t		0.418	0.302	0.661	0.036	1.481	0.245	1.242 (df = 30)
p		0.676	0.763	0.509	0.970	0.138	0.806	0.219
Misuse cases	Not present	43	13	31	48	38	39	212
	Present	65	20	34	50	35	53	257
U		1.557	1.150	0.925	0.188	0.589	0.944	1.866
p		0.119	0.250	0.355	0.851	0.556	0.345	0.062
Correctness	Not present	18	5	20	20	17	16	96
	Present	25	17	21	26	17	25	131
U		0.361	2.977	0.850	1.809	1.102	1.529	3.525
p		0.718	**0.003**	0.396	0.070	0.271	0.126	0.0004

Differing Results: Do Security Patterns Really Help Designers?

This study is by the same author team of the Study "Does Organizing Security Patterns Focus Architectural Choices?" ([**?**]) and has a similar structure [**?**]. The authors conduct a similar study, in which 64 graduate students in pairs of two should ensure the security of a banking system by completing six tasks. In a nutshell, the authors could not find an effect of security patterns, neither on time to solve task nor on the number of covered misuse cases. However, they found that some tasks were solved more often correctly when security patterns were provided.

- Independent variable: Security patterns (2 levels, operationalized as present or not present
- Tasks: 6 different tasks to improve security of a given software system (plus a warm-up task that was not analyzed
- Dependent variables:
  1. Task completion time [metric scale]
  2. Number of covered misuse cases (1 to 5) [ordinal scale]
  3. Correctness (wrong, some errors, correct) [ordinal scale]
- Null hypotheses:
  1. The usage of security patterns has no influence on the mean time needed to complete a task.
  2. The usage of security patterns has no influence on the mean number of covered misuse cases for a task.[1]
  3. The usage of security patterns has no influence on the correctness scores of a task (not explicitly stated in the paper, but still tested).
- Results:
  1. No significant effect of security patterns on the time to solve task
  2. No significant effect of security patterns on the number of covered misuse cases
  3. A significant effect for two of the six tasks for correctness

We summarize the number of covered misuse cases, correctness, and time to solve the tasks in Table 1. We cannot replicate the p values that the authors reported, possibly because we used Python (scipy) for the reanalysis, while the authors used R. This leads to the fact that we did not observe a significant difference in correctness for Task E, although the authors did. However, the authors did not correct for multiple comparisons. Using an FDR correction based on the p values of the authors, the difference for Task E also vanishes.

In the reanalysis, we aggregated the data over all tasks, such that we sum up the values for all tasks and then compare the means (task completion time) and ranks (number of misuse cases, correctness). In essence, we can confirm that for response time and number of misuse cases, the presence of security patterns has no effect. However, for correctness, we find a significant difference over all tasks in favor of security patterns.

[1] Technically, the mean is not correct here, as the authors used a Wilcoxon test, which does not compare the means, but the rank sums.

To conclude, for this case, the aggregation did not really change the overall picture. We confirmed the authors' result that correctness can be affected by the presence of security patterns. With the task-wise analysis, the authors looked deeper into the data. What would have been interesting now is to take a closer look at how Task E differs from the others. For example, this task was the only task for which the security pattern *Demilitarized Zone* was relevant. Maybe this pattern is especially helpful to understand how to implement security updates to an existing application (in this case, to prevent read access).