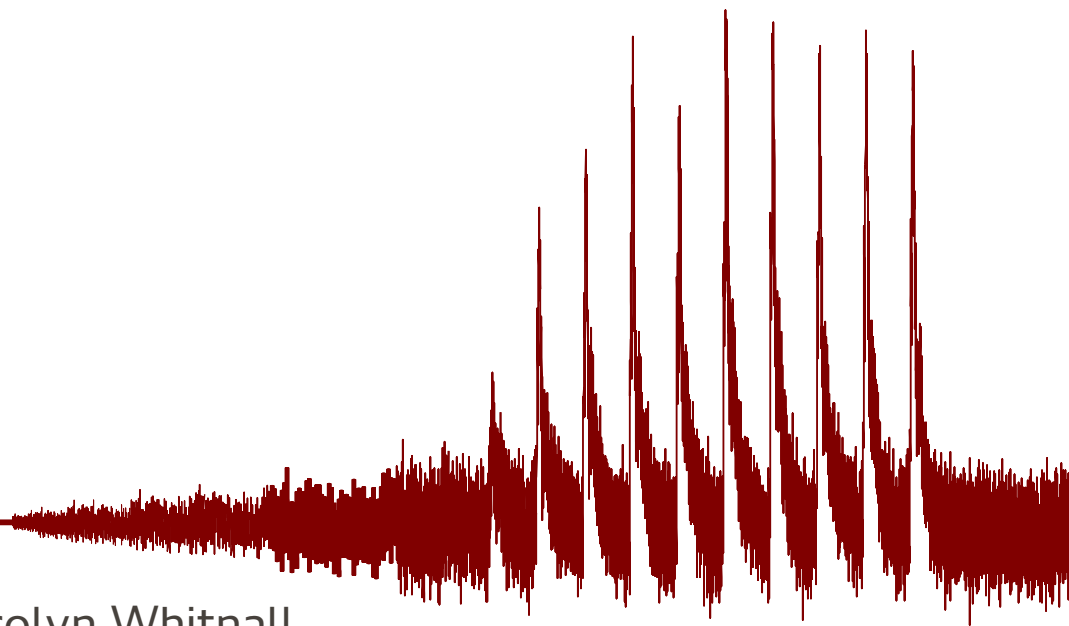
Multi-target DPA attacks: Pushing DPA beyond the limits of a desktop computer

December 2, 2014



Luke Mather, Elisabeth Oswald, Carolyn Whitnall

Cryptography Group, University of Bristol

MOTIVATION

Scenario:

MOTIVATION

Scenario:

- Side-channel techniques such as Differential Power Analysis (DPA) threaten security of devices

MOTIVATION

Scenario:

- Side-channel techniques such as Differential Power Analysis (DPA) threaten security of devices
- When evaluating devices, it is important to define an **optimal** adversary

MOTIVATION

Scenario:

- Side-channel techniques such as Differential Power Analysis (DPA) threaten security of devices
- When evaluating devices, it is important to define an **optimal** adversary
- Traditionally, an optimal adversary considered to use a **single attack** of minimal data complexity

MOTIVATION

Scenario:

- Side-channel techniques such as Differential Power Analysis (DPA) threaten security of devices
- When evaluating devices, it is important to define an **optimal** adversary
- Traditionally, an optimal adversary considered to use a **single attack** of minimal data complexity

However:

MOTIVATION

Scenario:

- Side-channel techniques such as Differential Power Analysis (DPA) threaten security of devices
- When evaluating devices, it is important to define an **optimal** adversary
- Traditionally, an optimal adversary considered to use a **single attack** of minimal data complexity

However:

- By taking results of a single attack only, we ignore information leakage in other parts of the data set

MOTIVATION

Scenario:

- Side-channel techniques such as Differential Power Analysis (DPA) threaten security of devices
- When evaluating devices, it is important to define an **optimal** adversary
- Traditionally, an optimal adversary considered to use a **single attack** of minimal data complexity

However:

- By taking results of a single attack only, we ignore information leakage in other parts of the data set
- High-performance computing (HPC) is alleviating computational restrictions placed on adversaries

MOTIVATION

Scenario:

- Side-channel techniques such as Differential Power Analysis (DPA) threaten security of devices
- When evaluating devices, it is important to define an **optimal** adversary
- Traditionally, an optimal adversary considered to use a **single attack** of minimal data complexity

However:

- By taking results of a single attack only, we ignore information leakage in other parts of the data set
- High-performance computing (HPC) is alleviating computational restrictions placed on adversaries

Question:

MOTIVATION

Scenario:

- Side-channel techniques such as Differential Power Analysis (DPA) threaten security of devices
- When evaluating devices, it is important to define an **optimal** adversary
- Traditionally, an optimal adversary considered to use a **single attack** of minimal data complexity

However:

- By taking results of a single attack only, we ignore information leakage in other parts of the data set
- High-performance computing (HPC) is alleviating computational restrictions placed on adversaries

Question:

- Can we find **practical** ways to exploit as much of the leakage as possible?

CLASSICAL DPA

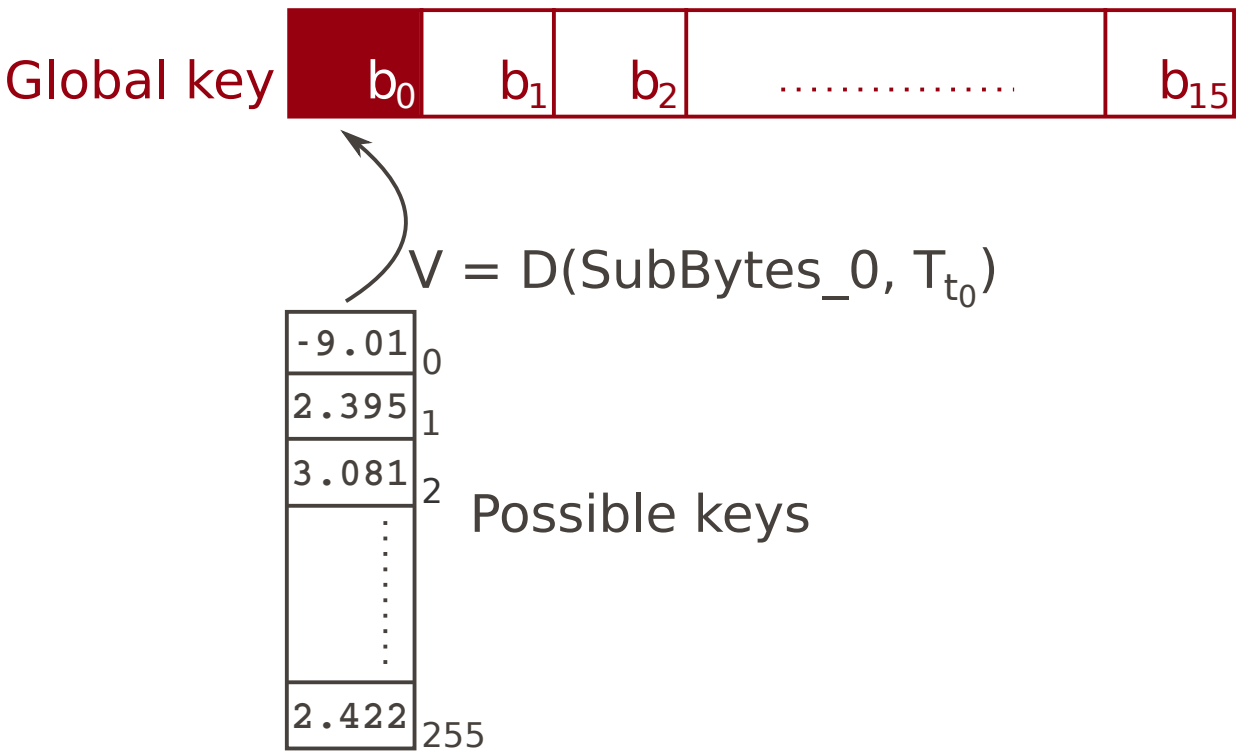
Adversary selects:

- T 1x set of trace acquistions captured over time
- F A selected target function (e.g for AES: SubBytes, AddRndKey, MixColumns). The choice of target allows the adversary to make predictions about the value of a **subkey** (e.g first SubBytes operation leaks on first byte of the key)
- t 1x time point within the set of traces
- D A "distinguisher"—statistical tool for guessing subkey values

The attack $D(F, T_t)$ produces a "distinguishing vector" V.

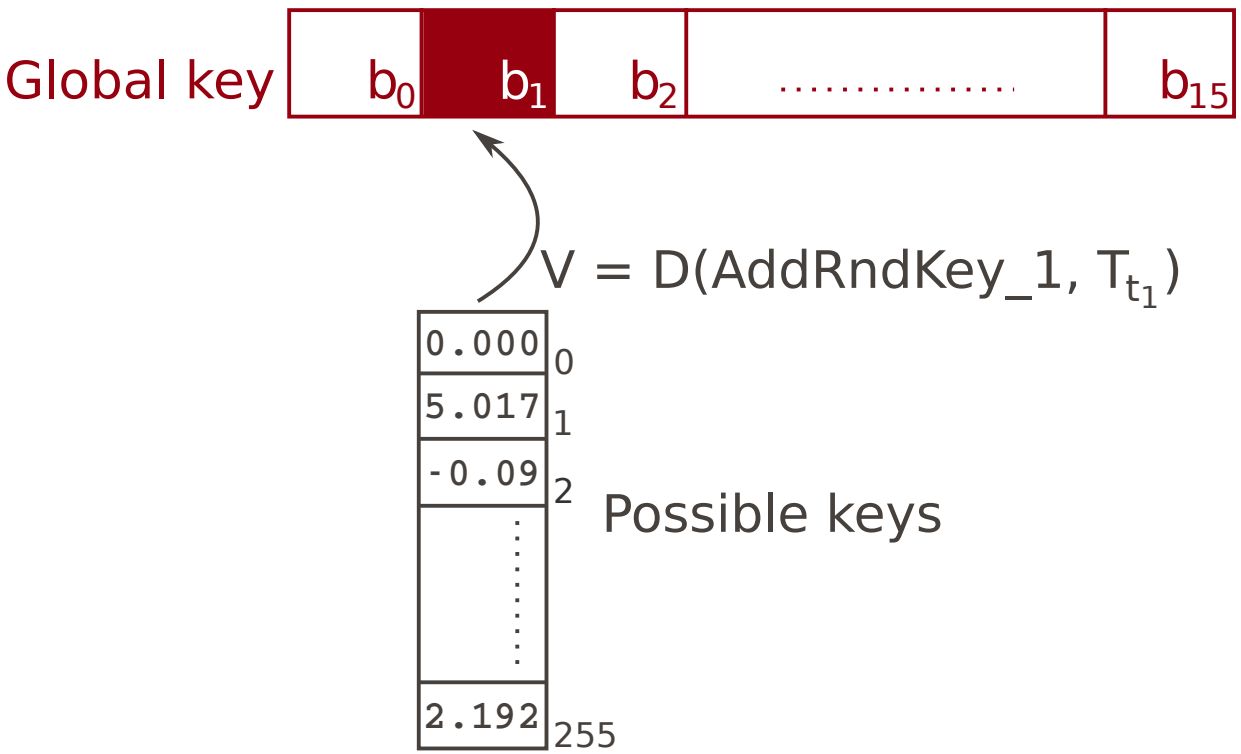
DISTINGUISHING VECTORS

- Contain ‘scores’ for each possible value of the subkey (portion of the global key) associated with the target function.



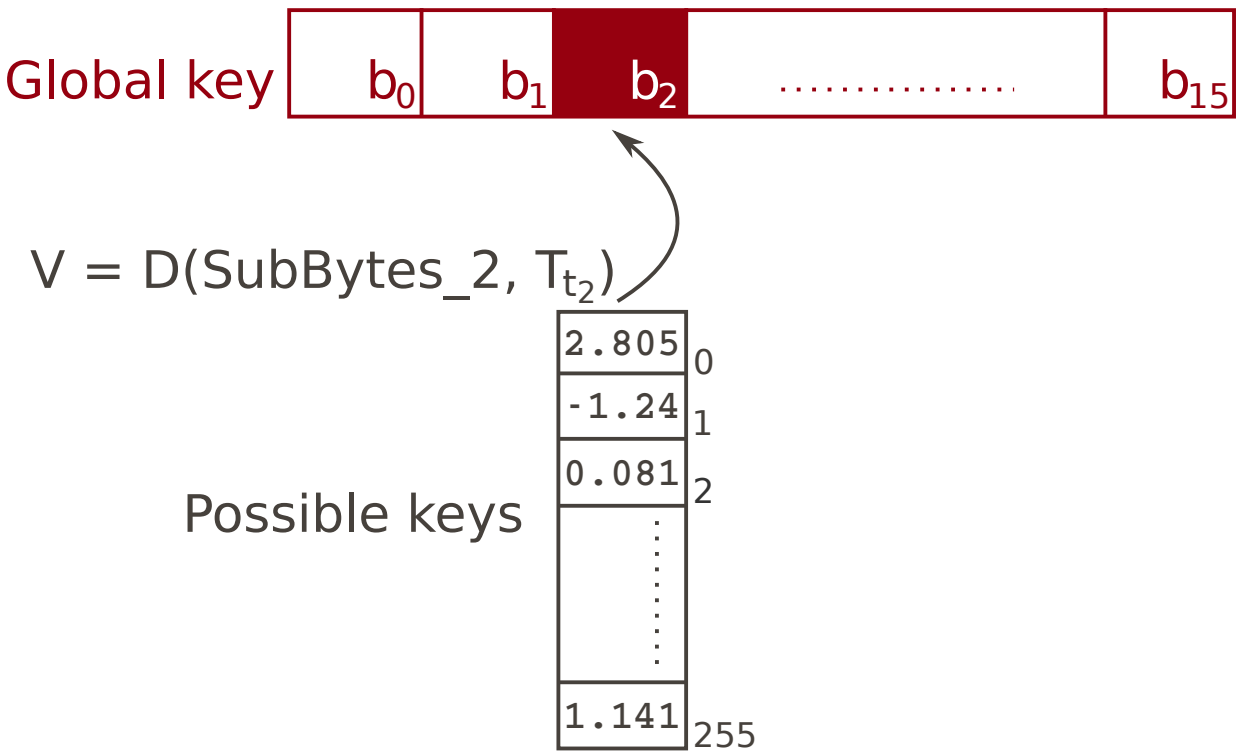
DISTINGUISHING VECTORS

- Contain ‘scores’ for each possible value of the subkey (portion of the global key) associated with the target function.



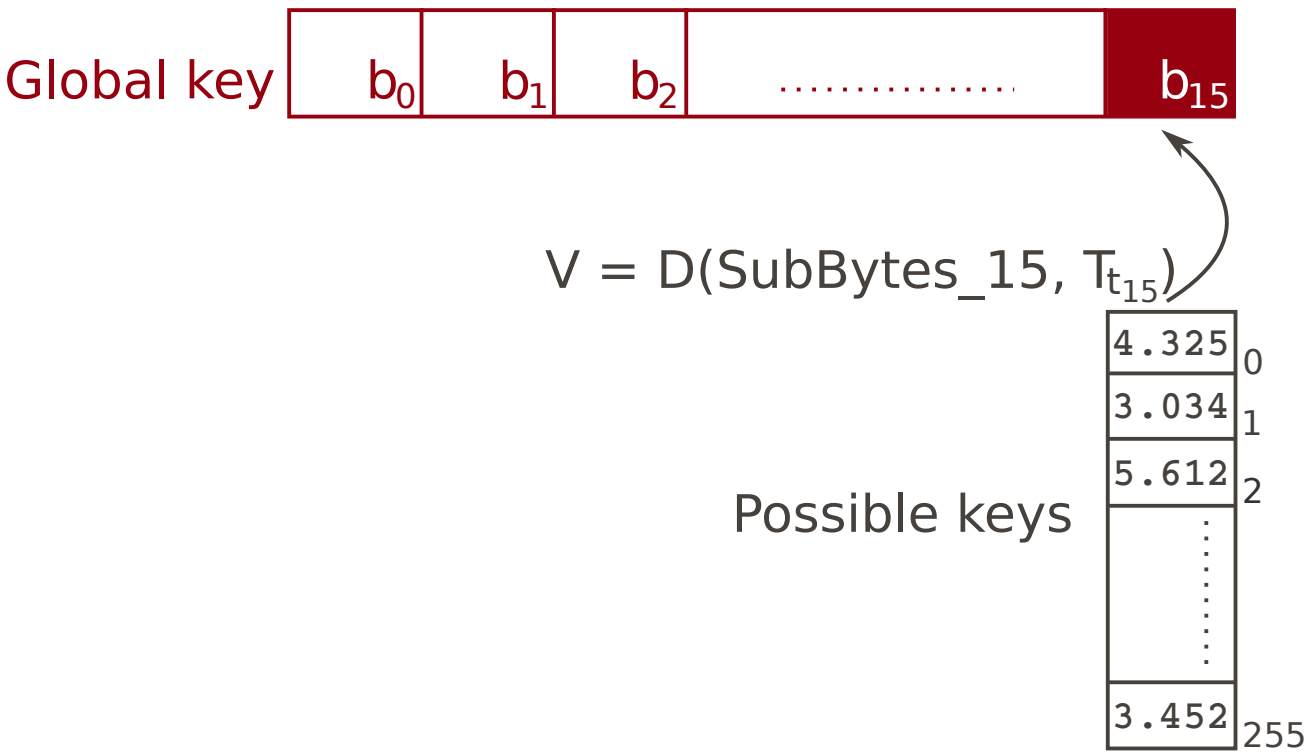
DISTINGUISHING VECTORS

- Contain ‘scores’ for each possible value of the subkey (portion of the global key) associated with the target function.



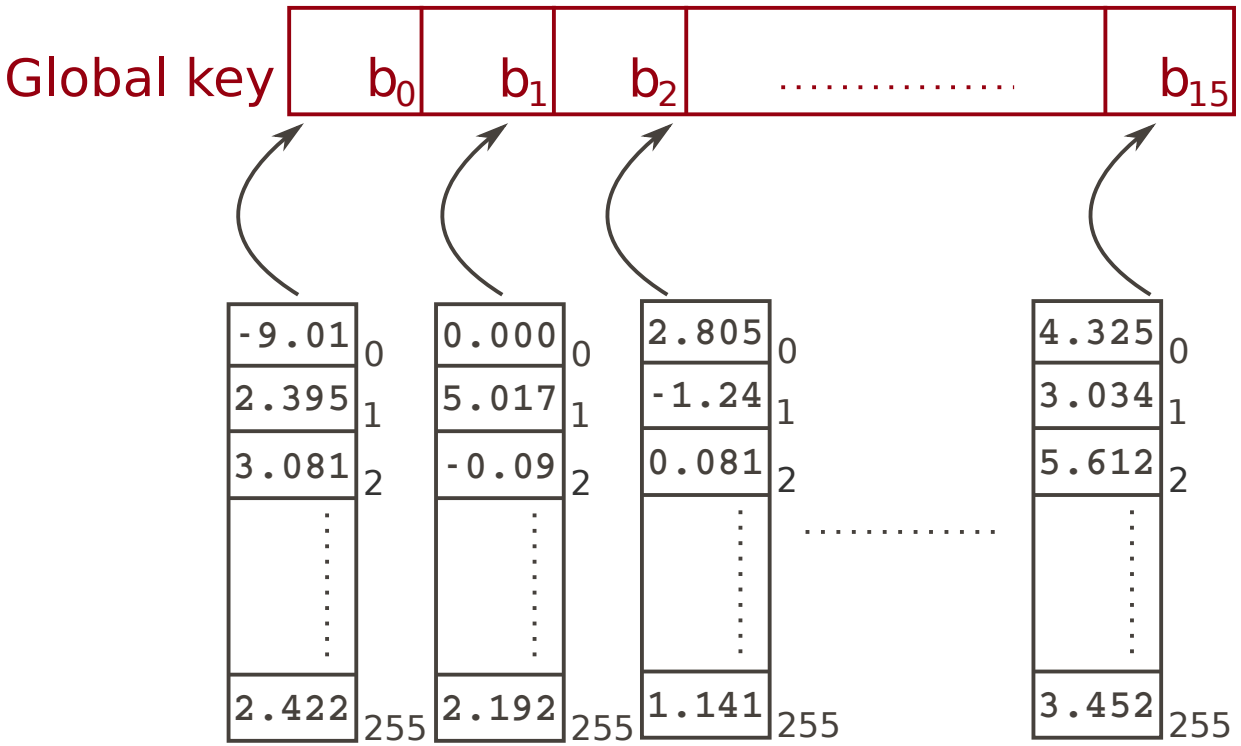
DISTINGUISHING VECTORS

- Contain ‘scores’ for each possible value of the subkey (portion of the global key) associated with the target function.



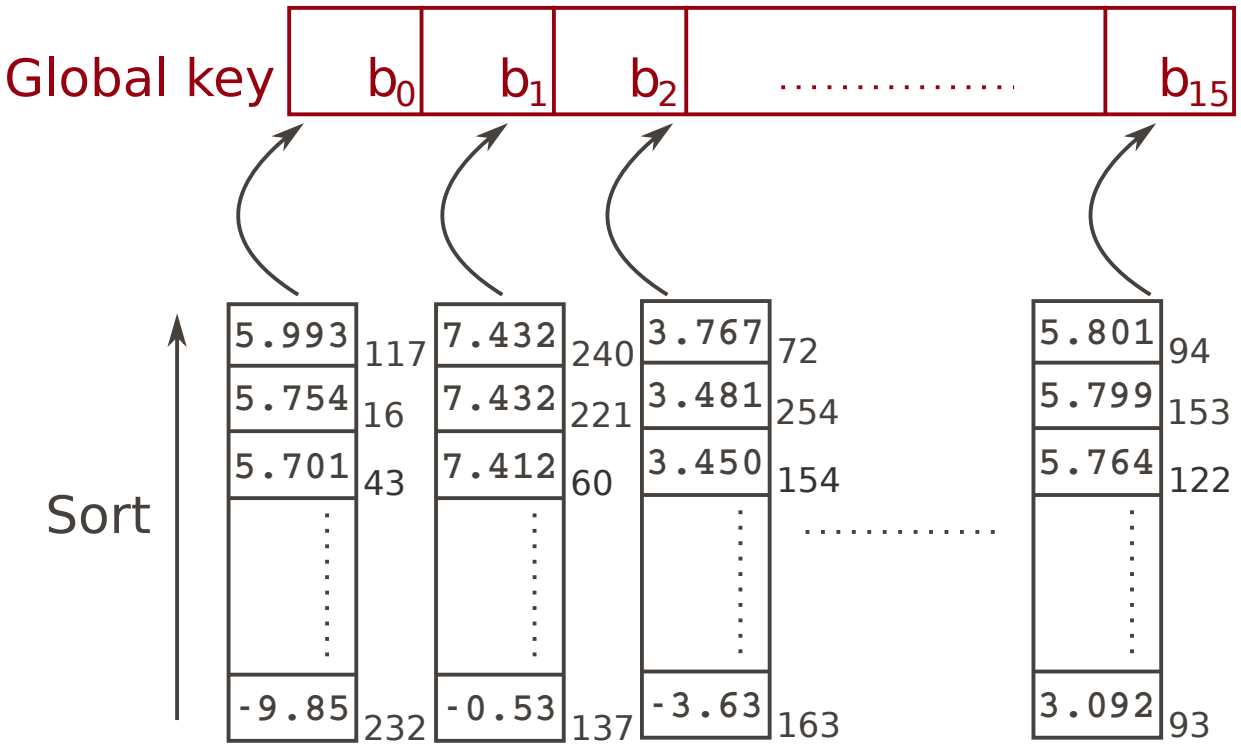
KEY ENUMERATION

- Correct sub-key value not necessarily ranked first in each distinguishing vector
- Use key enumeration (Veyrat-Charvillion SAC '12) to search the candidate space



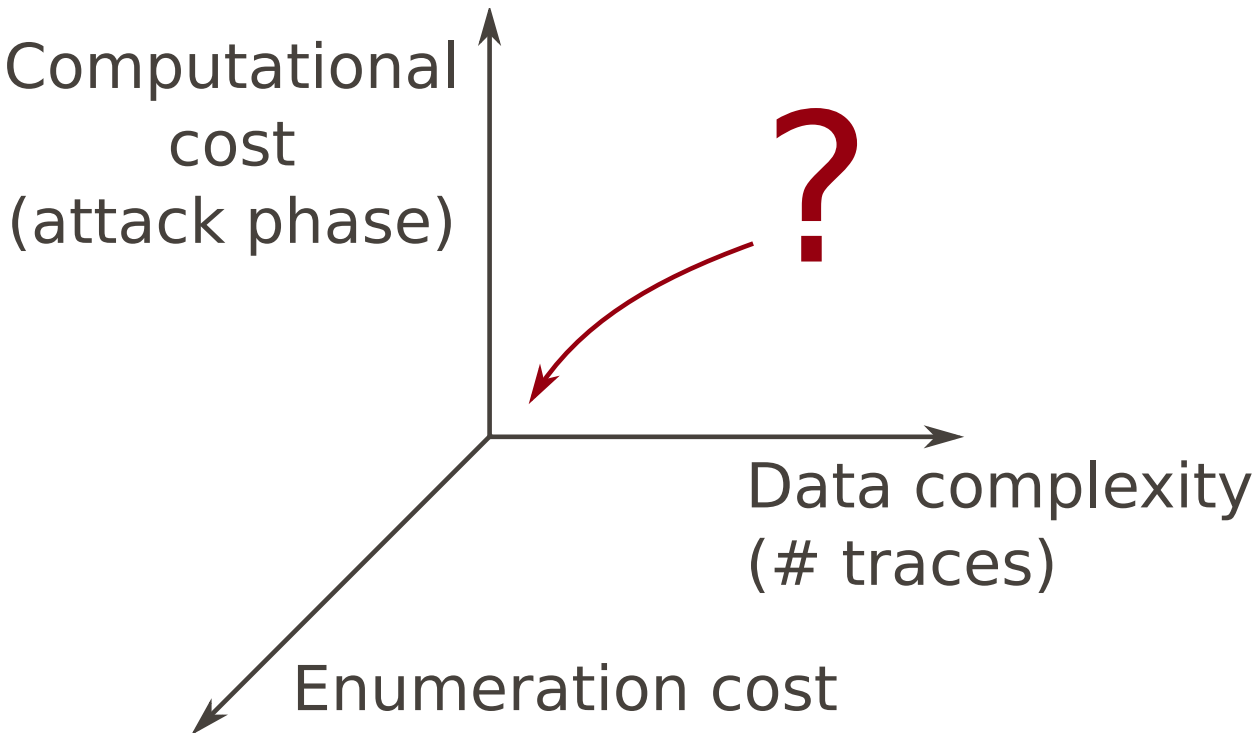
KEY ENUMERATION

- Correct sub-key value not necessarily ranked first in each distinguishing vector
- Use key enumeration (Veyrat-Charvillion SAC '12) to search the candidate space



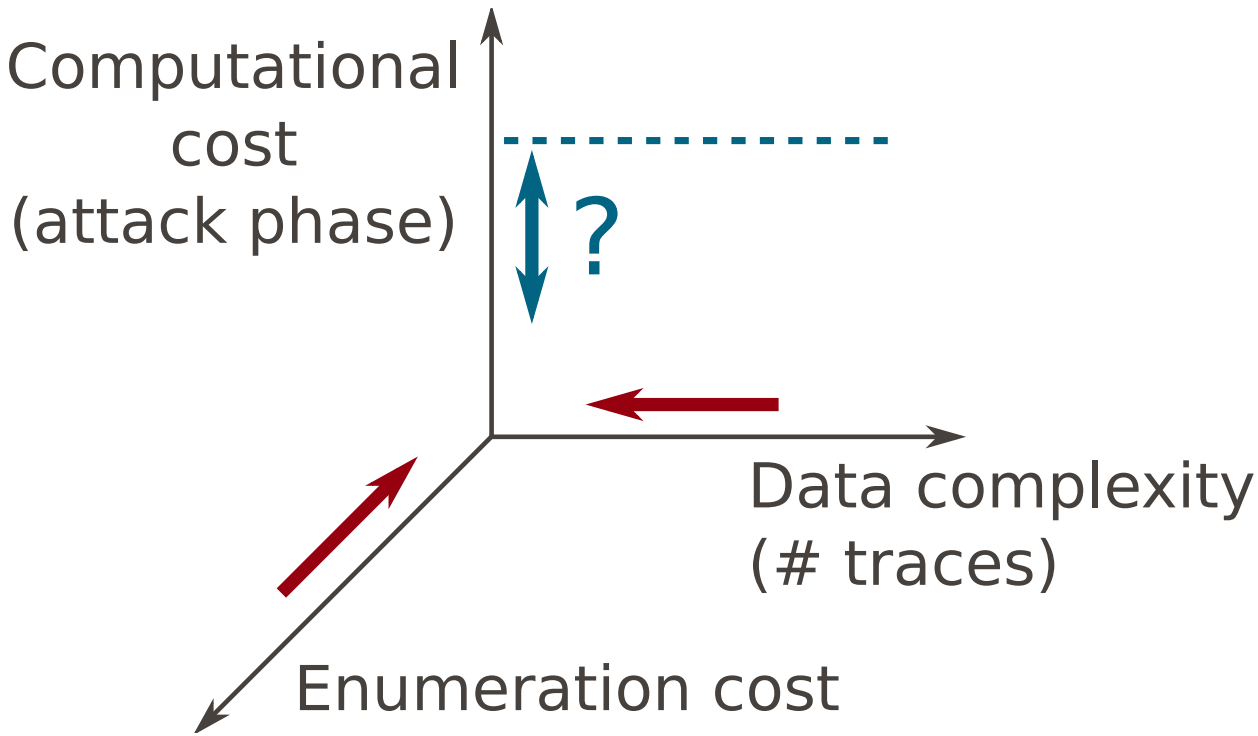
OPTIMALITY

Traditionally optimise for data complexity. But what is the best adversary?

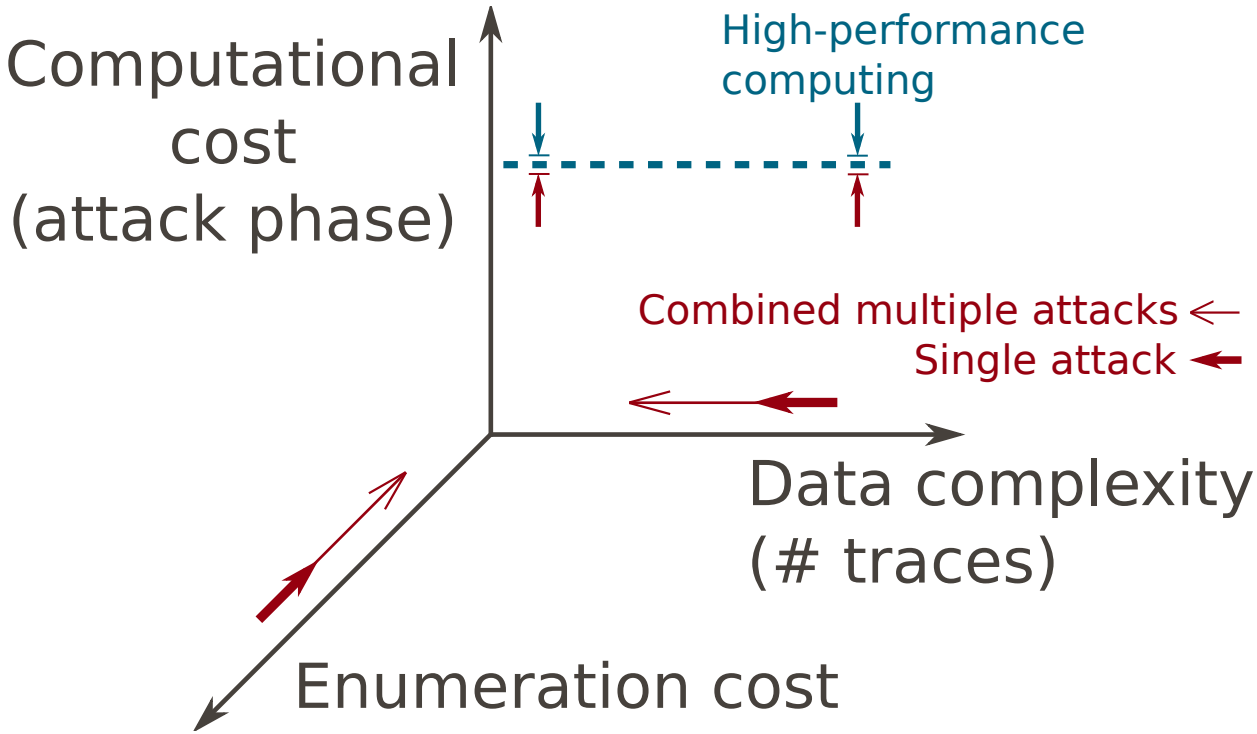


OPTIMALITY

Traditionally optimise for data complexity. But what is the best adversary?



THIS WORK

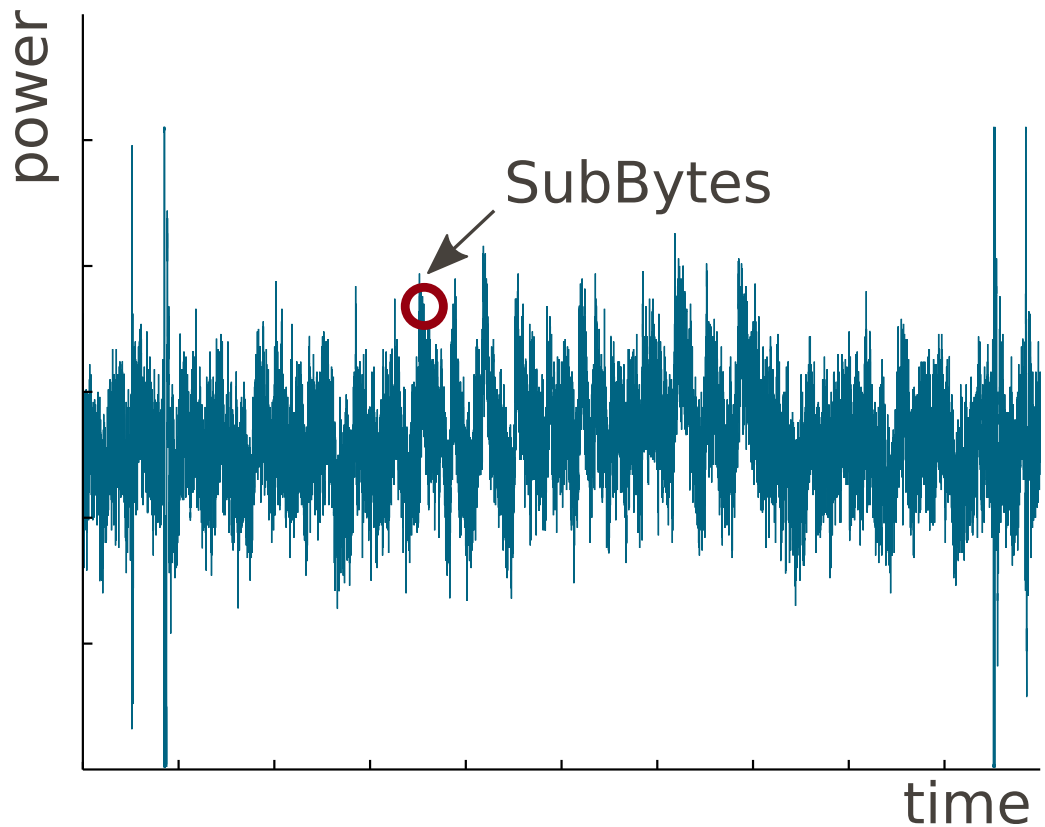


- Idea: improve by running **multiple** DPA attacks and **combining** the key information

MULTIPLE SOURCES OF INFORMATION LEAKAGE

Scenario: find the first key byte of an AES key.
Traditional approach: take results of **single best attack**

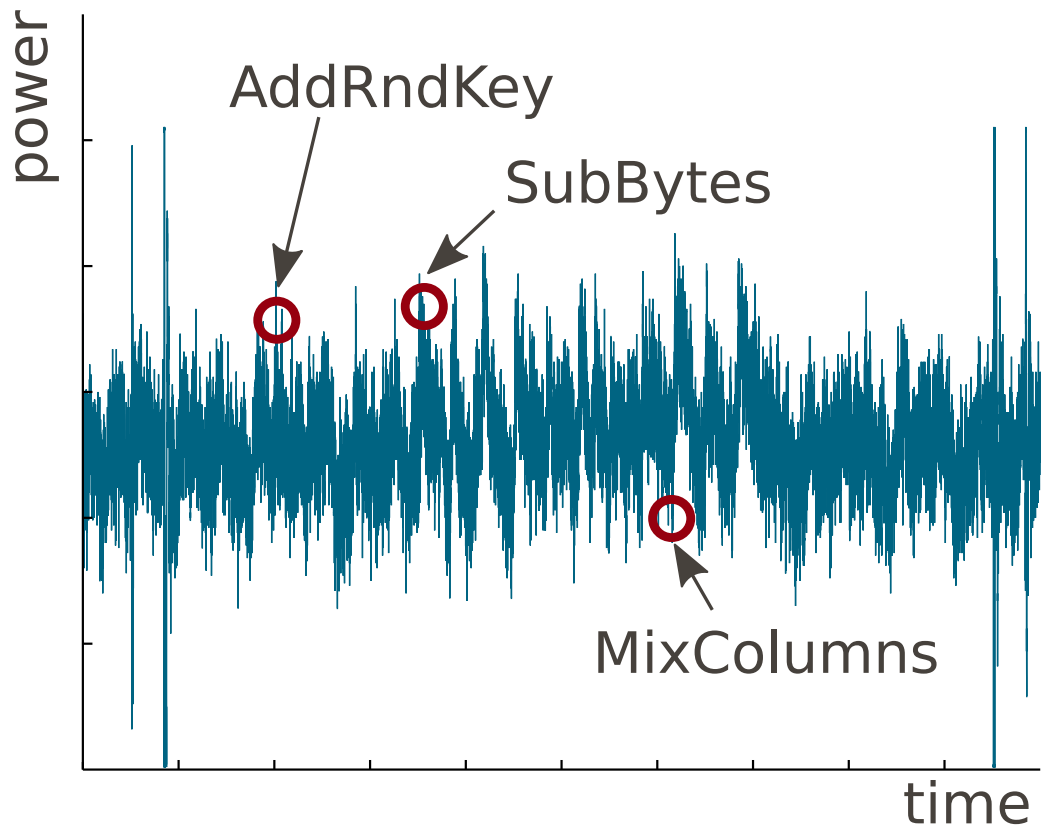
- $V = D(F, T_t)$



MULTIPLE SOURCES OF INFORMATION LEAKAGE

...or combine results calculated using different **target functions**, using best distinguishers and time points?

- $V = \text{Combine}(D(\boldsymbol{F_1}, T_{t_1}), D(\boldsymbol{F_2}, T_{t_2}), \ldots)$



METHODOLOGY

Need a method for combining results (distinguishing vectors) of multiple attacks:

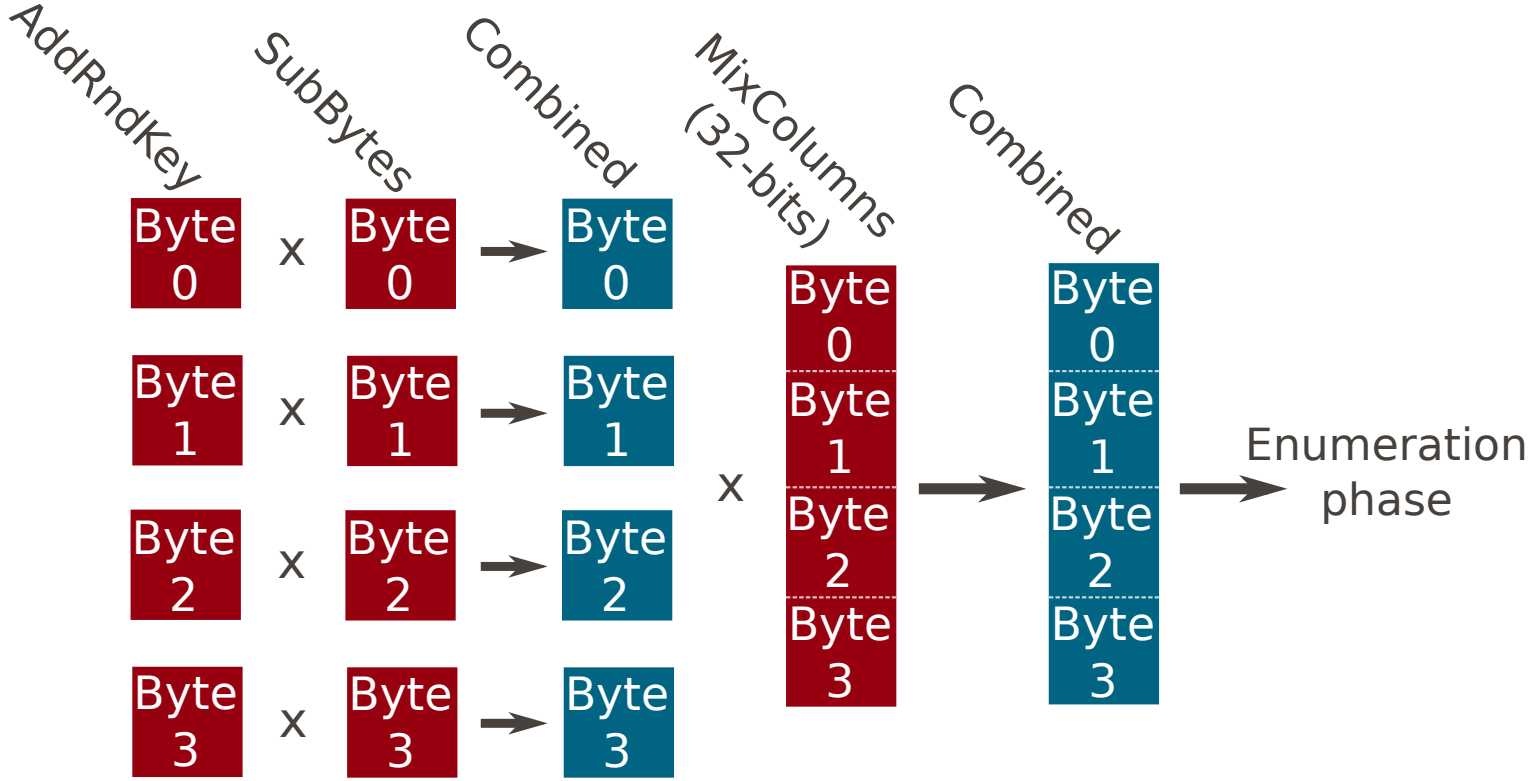
- Ideally: have "probabilities" for subkey candidates
- Distinguishers don't (usually) do this—need a conversion method
- Need to preserve the **ranks** and the **relative distance** between the subkey candidates

Solution:
Given two distinguishing vectors:

1. Transform to be positive-valued with a baseline of zero
2. Normalise the vector scores to sum to 1
3. Combine vectors by **pointwise multiplying**

AES CASE STUDY

Suppose exploitable information leakage on key bytes 0,..,3 occurs under the 8-bit AddRndKey and SubBytes operations and a 32-bit MixColumns target



Many other combinations possible!

ASSESSING EFFECTIVENESS OF THE METHOD

- **Aim**: compare the best single-target attack against various multiple-target attacks

ASSESSING EFFECTIVENESS OF THE METHOD

- **Aim**: compare the best single-target attack against various multiple-target attacks
- **How**: compare the sizes of the sets of remaining subkey candidates to test after the attacks

ASSESSING EFFECTIVENESS OF THE METHOD

- **Aim**: compare the best single-target attack against various multiple-target attacks
- **How**: compare the sizes of the sets of remaining subkey candidates to test after the attacks
- **Q**: 32-bit targets are time-consuming attacks—can we mitigate this?

ASSESSING EFFECTIVENESS OF THE METHOD

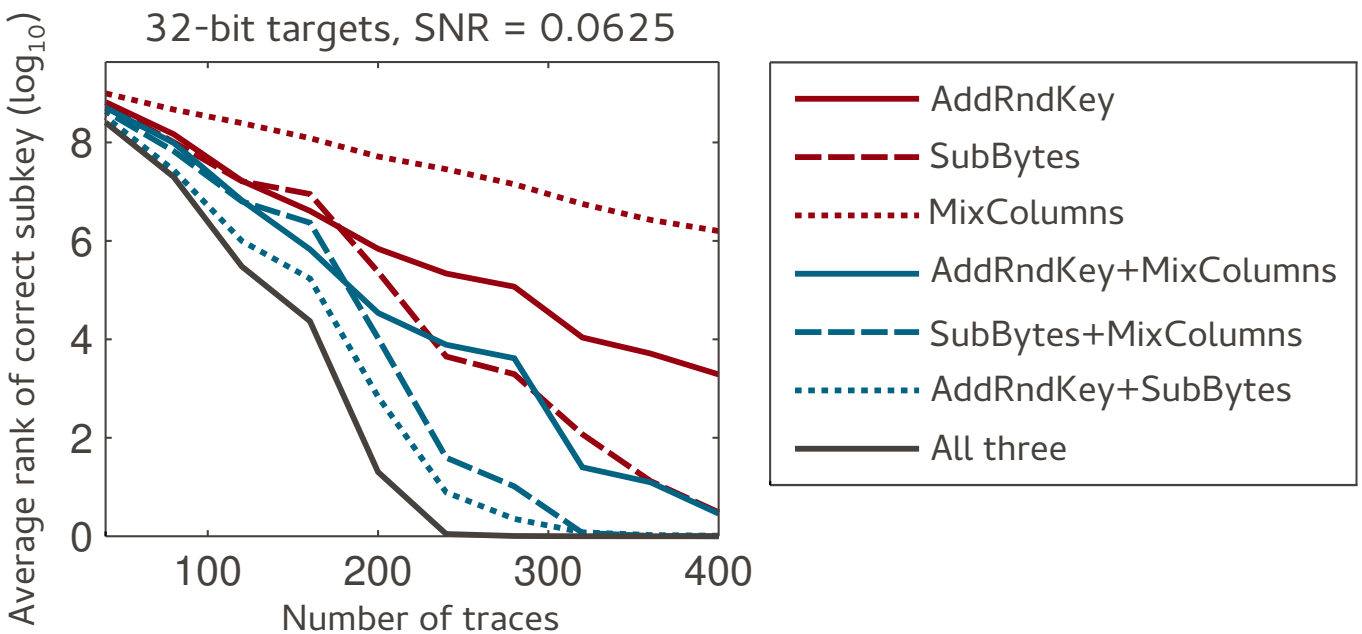
- **Aim**: compare the best single-target attack against various multiple-target attacks
- **How**: compare the sizes of the sets of remaining subkey candidates to test after the attacks
- **Q**: 32-bit targets are time-consuming attacks—can we mitigate this?
- **Q**: In reality we don't know where leakage occurs—does the combining strategy remain effective here?

SIMULATED EXPERIMENTS

Simulated leakage:

- Different signal-to-noise ratios
- Used correlation distinguisher with Hamming weight model

Example results:

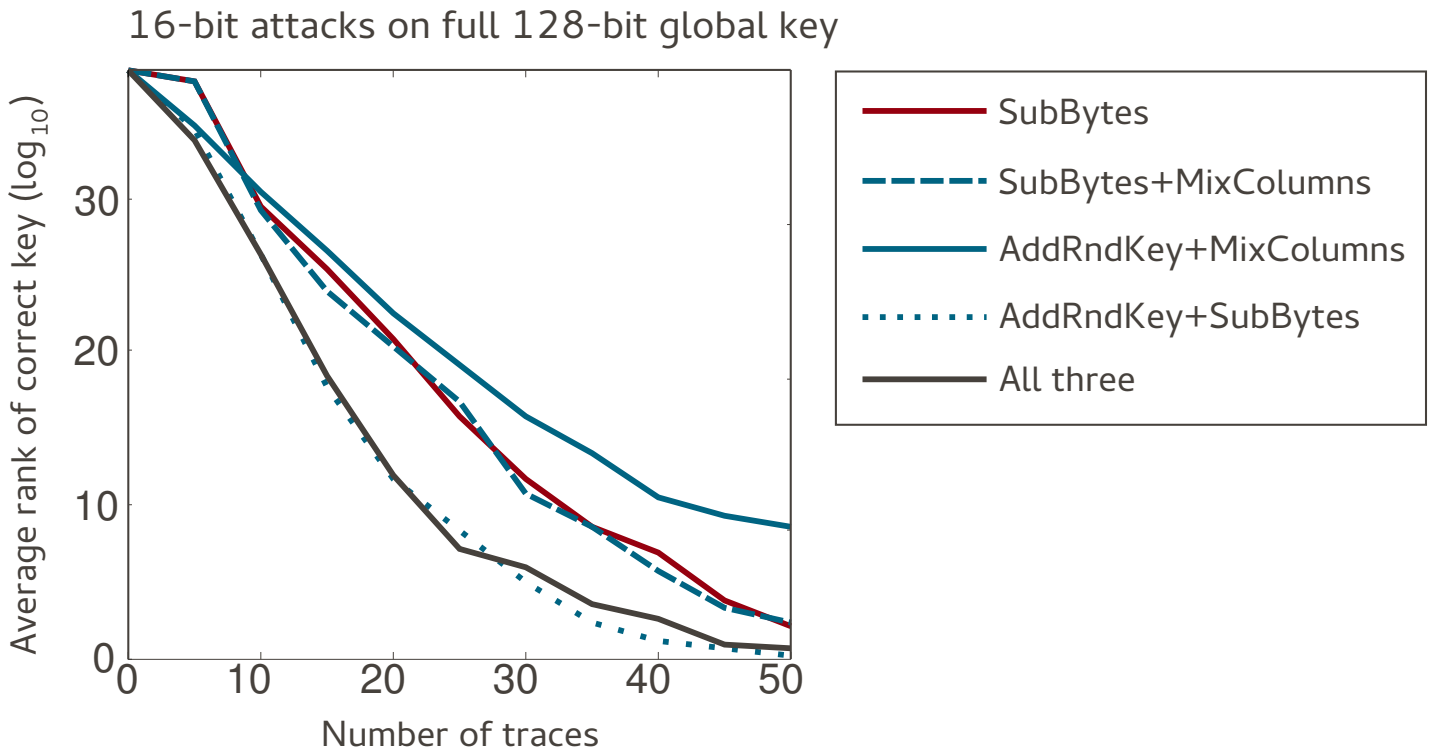


Generally: multi-target attacks did a better job

ARM7 EXPERIMENTS

- **Strongest adversary**: assume the points at which leakage occurs are known
- Unprotected AES: 10,000 traces, 200 repeat experiments

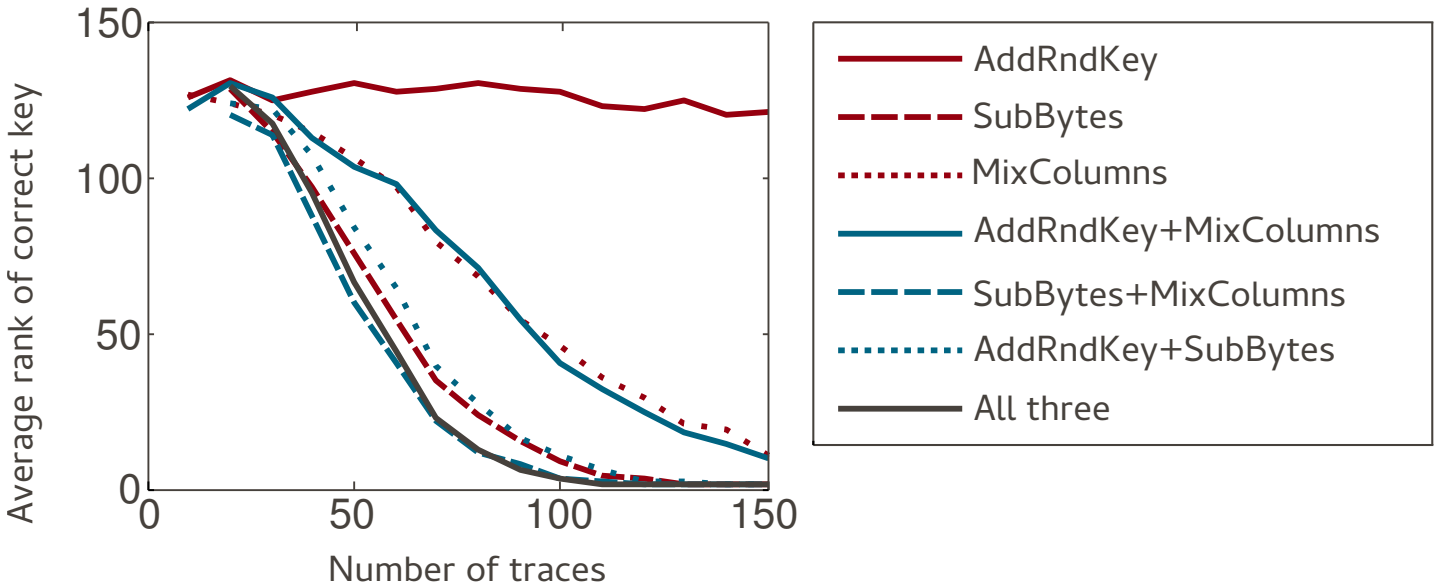
Example results:



ARM7 EXPERIMENTS

- **Weaker adversary**: assume windows of points in which leakage occurs are known
- Exhaustive search of all combinations of points in the windows

Example results:



COMPUTATIONAL COST

- 32-bit attacks on MixColumns are expensive—2^{32} subkey hypotheses at each time point

COMPUTATIONAL COST

- 32-bit attacks on MixColumns are expensive—2^{32} subkey hypotheses at each time point
- Moradi et al. in 2012 attack MixColumns and 60,000 traces using 4 Tesla C2070 GPUs in ~8.25 minutes

COMPUTATIONAL COST

- 32-bit attacks on MixColumns are expensive—2^{32} subkey hypotheses at each time point
- Moradi et al. in 2012 attack MixColumns and 60,000 traces using 4 Tesla C2070 GPUs in ~8.25 minutes
- In our setup: can attack 60,000 traces in ~15 seconds (33x faster) using 4 R9 290X GPUs

COMPUTATIONAL COST

- 32-bit attacks on MixColumns are expensive—2^{32} subkey hypotheses at each time point
- Moradi et al. in 2012 attack MixColumns and 60,000 traces using 4 Tesla C2070 GPUs in ~8.25 minutes
- In our setup: can attack 60,000 traces in ~15 seconds (33x faster) using 4 R9 290X GPUs
- Need to start accounting for this level of acceleration!

CONCLUSION

Summary

CONCLUSION

Summary

CONCLUSION

Summary

- Multiple-target attacks make good use of additional information leakage to create (often) stronger attacks

CONCLUSION

Summary

- Multiple-target attacks make good use of additional information leakage to create (often) stronger attacks
- Concept should extend naturally in presence of countermeasures, and is robust

CONCLUSION

Summary

- Multiple-target attacks make good use of additional information leakage to create (often) stronger attacks
- Concept should extend naturally in presence of countermeasures, and is robust
- Use of HPC allows for a significant enhancement of an adversary's capabilities

CONCLUSION

Summary

- Multiple-target attacks make good use of additional information leakage to create (often) stronger attacks
- Concept should extend naturally in presence of countermeasures, and is robust
- Use of HPC allows for a significant enhancement of an adversary's capabilities

Further work

CONCLUSION

Summary

- Multiple-target attacks make good use of additional information leakage to create (often) stronger attacks
- Concept should extend naturally in presence of countermeasures, and is robust
- Use of HPC allows for a significant enhancement of an adversary's capabilities

Further work

CONCLUSION

Summary

- Multiple-target attacks make good use of additional information leakage to create (often) stronger attacks
- Concept should extend naturally in presence of countermeasures, and is robust
- Use of HPC allows for a significant enhancement of an adversary's capabilities

Further work

- Do enhanced strategies for when we don't know where the leakage is exist?

CONCLUSION

Summary

- Multiple-target attacks make good use of additional information leakage to create (often) stronger attacks
- Concept should extend naturally in presence of countermeasures, and is robust
- Use of HPC allows for a significant enhancement of an adversary's capabilities

Further work

- Do enhanced strategies for when we don't know where the leakage is exist?
- Further exploration into attacks utilising combination as a building block

CONCLUSION

Thanks for listening!