Four Rounds are Not Enough

Orr Dunkelman

Department of Computer Science,
University of Haifa

Joint work with Itai Dinur and Adi Shamir

Outline

Differential Characteristics for KECCAK

- Obviously, KECCAK was design to withstand differential cryptanalysis.
- The submission considers the *Column Parity Kernel* (CP-Kernel) characteristics, which maintains the low hamming weight.
- The idea is to use even hamming weight differences, so that the θ will maintain the low hamming weight.
- As Maria just presented, this can lead to a collision attack on 2-round KECCAK using a characteristic $\Delta_{IN} \rightarrow \Delta_{OUT}$.

We want more rounds!

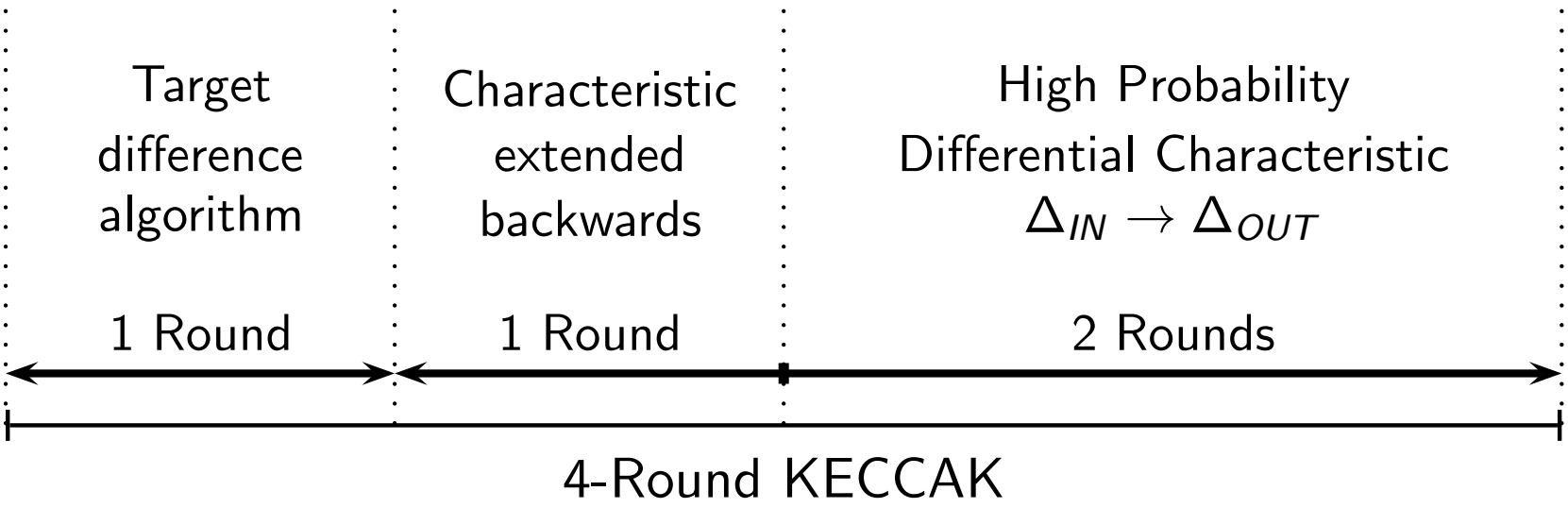
Differential Characteristics for KECCAK (cont.)

- While the characteristic can be extended one round forward, it no longer leads to a collision (but to a near collision).
- Extending backwards leads to a high hamming weight difference (i.e., $\underbrace{\theta^{-1}\rho^{-1}\pi^{-1}}_{L^{-1}}\chi^{-1}\iota^{-1}(\Delta_{IN})$).
- In addition, this input difference does not correspond to the bits we can control via the input.

Some Useful Observations

1. When extending the characteristic backwards, the first round has high probability.
2. There are actually several differences that lead to a high-probability characteristic.
3. The algebraic degree of KECCAK's S-box (χ) is just 2.
4. Which means that from a differential point of view, it's degree is 1...

The General Idea Behind the Attack



The Settings

Consider the 3-round differential characteristic:

```
|26978AF134CB835E|AF224C4D78366789|C4DAE35E2656F26B|357C4789AF3-6AF1|78D3526BC6A74C4D|
|26978AF134CB835E|AF224C4D78366789|C4DAE35E2656F26B|357C4789AF3-6AF1|78D3526BC6A74C4D|
|26978AF134CB835E|AF224C4D78366789|C4DAE35E2676F26B|357C4789AF3-6AF1|78D3526BC4A74C4D|  ΔS1
|26978AF134CB835E|AF224C4D78366789|C4DAE35E265EF26B|357C4789AF3-4AF1|78D3526BC6A74C4D|
|26978AF134CB835E|AF226C4D78366789|C4DAE35E2656F26B|35FC4789AF3-6AF1|78D3526BC6A74C4D|

|---------------|---------------|---------------1|-------4-------|---------------|
|---------------|---------------|---------------|---------------|---------------|
|---------------|---------------|---------------|---------------|---------------|  ΔS2
|---------------|---------------|---------------|-------4-------|----8----------|
|---------------|---------------|---------------1|---------------|----8----------|

|---------------|---------------|---------------|--8------------|2--------------|
|4--------------|---------------|---------------|---------------|2--------------|
|---------------|---------------|---------------|--8------------|---------------|  ΔS3
|---------------|---------------|---------------|---------------|---------------|
|4--------------|---------------|---------------|---------------|---------------|

|---------------|---------------|---------------|---------------|---------------|
|----------8----|----------2----|---------------|---------------|---------------|
|---------------|---------------|----------1----|---------------|----------1----|  ΔS4
|--------1------|-------4-------|---------------|---------------|---------------|
|---------------|---------------|---------------|---------------|---------------|
```

The Settings (cont.)

- We wish to find a way to link the input to ΔS_1.
- The input to 4-round KECCAK is of the form $\Delta S_0 =< \text{Something we control} > ||0$.
- We call this difference the "target difference".

$$\Delta S_0 = |\star|0|$$

$$\downarrow$$

```
|26978AF134CB835E|AF224C4D78366789|C4DAE35E2656F26B|357C4789AF3-6AF1|78D3526BC6A74C4D|
|26978AF134CB835E|AF224C4D78366789|C4DAE35E2656F26B|357C4789AF3-6AF1|78D3526BC6A74C4D|
|26978AF134CB835E|AF224C4D78366789|C4DAE35E2676F26B|357C4789AF3-6AF1|78D3526BC4A74C4D|  ΔS1
|26978AF134CB835E|AF224C4D78366789|C4DAE35E265EF26B|357C4789AF3-4AF1|78D3526BC6A74C4D|
|26978AF134CB835E|AF226C4D78366789|C4DAE35E2656F26B|35FC4789AF3-6AF1|78D3526BC6A74C4D|
```

The Target Difference Algorithm

- The **Target Difference Algorithm** is a tool that finds many pairs with difference ΔS_1.
- To do so, it performs two steps:
  1. Picks ΔS_0,
  2. Finds pairs of inputs which satisfy $\Delta S_0 \rightarrow \Delta S_1$.

Challenges

The Target Difference Algorithm faces two challenges:

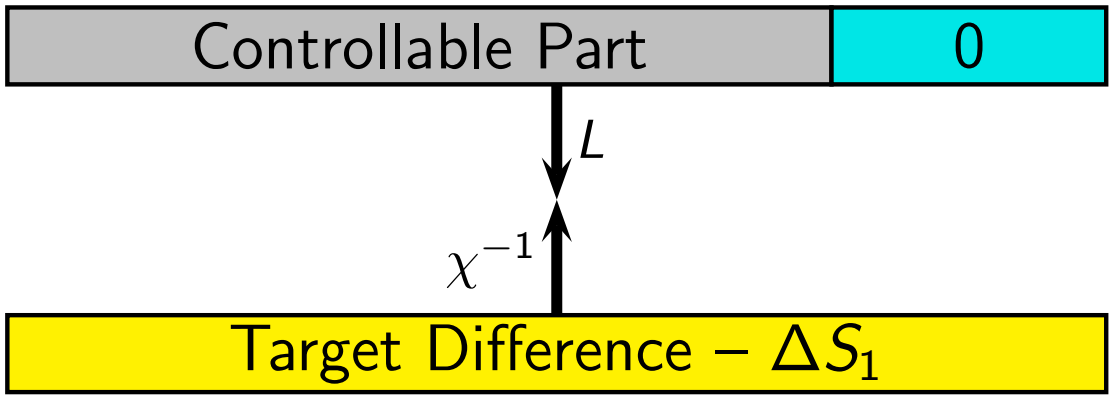
- The transition $\Delta S_0 \rightarrow \Delta S_1$ happens with a very low probability (many active S-boxes).
- The initial state has fixed bits (due to the KECCAK structure)

At the same time:

- We have 704 degrees of freedom for KECCAK-224, and 576 degrees for KECCAK-256.
- For an arbitrary target difference — we expect many solutions (albeit they are sparse).

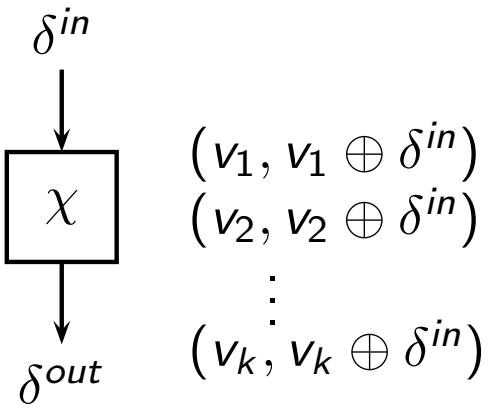
Formal Definition of the Problem

- KECCAK's round function has an algebraic degree of 2.
- Hence, it is possible to represent the problem of finding pairs as a quadratic equation, and solve it.
- The standard linearization uses too many degrees of freedom.



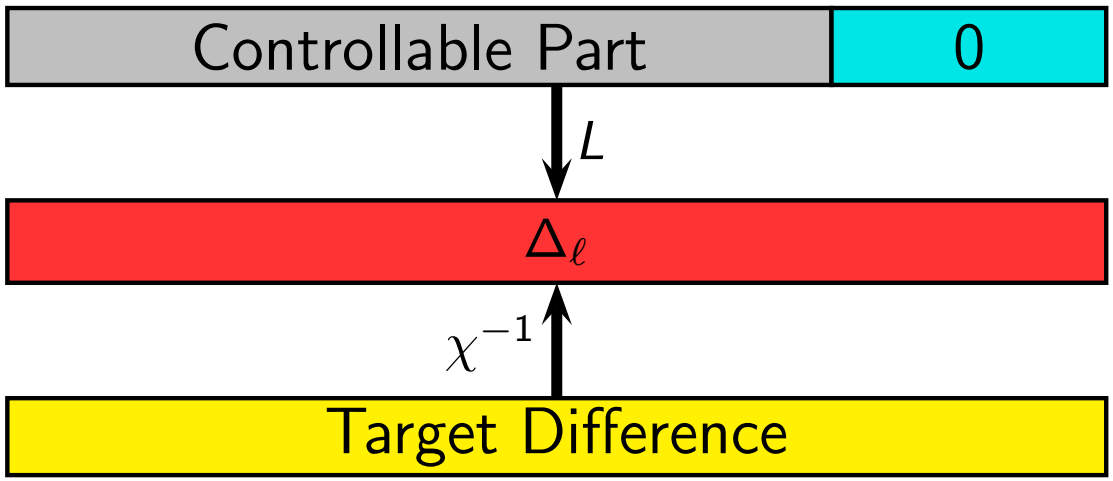
Using Differences

- Recall that the S-box degree is 2.
- Recall that its derivative is 1.
- In other words, once the input/output differences of an S-box is fixed, the values following it, form an **affine subspace**.



The Two Phases

- The algorithm works in two phases:
  1. **Difference Phase:** Finding a 1600-bit input difference to χ (denoted by Δ_ℓ),
  2. **Value Phase:** Finding the actual message pairs that lead to Δ_ℓ.



The Value Phase

- Given Δ_ℓ, finding message pairs which satisfy it is trivial (linear algebra).
- However, given the constraints, the equation set may be inconsistent.
- When this happens, we just pick a different Δ_ℓ.
- The output of the this phase is an **affine subspace** of message pairs (with difference Δ_ℓ).

The Difference Phase

- Recall that we are given some δ^{out} from an S-box.
- We need to pick the correct δ^{in}:
  1. Such that $DDT[\delta^{in}, \delta^{out}] > 0$.
  2. Such that when collecting all δ^{in}'s into Δ_ℓ, the value phase succeeds.
- The possible values for δ^{in} do not form a simple affine space.
- Trying to solve this problem using common "guess and determine" fails:
  1. The large search space requires "committing" to values that restrict the solution space,
  2. Very inefficient.

The Difference Phase (cont.)

- For a non-zero output difference δ^{out}, the set of possible input differences $\delta_1^{in}, \delta_2^{in}, \ldots$ contains at least five (and at most 17) 2-dimensional affine subspace.
- For each active S-box, we choose an *affine subspace* of input differences.
- This in turn allows **combining the linear constraints!**
- In other words, we do not commit to Δ_ℓ, but pick a subspace of it.
- Then, we pick the ones (out of the subspace) for which $L^{-1}(\Delta_\ell) = \star|0$.

The Differential Characteristic

- We went over all the "double-kernel CP-characteristics" of two rounds (571 such ones).
- 128 produce collisions for KECCAK-224, and 64 for KECCAK-256.
- We then extend the characteristic backwards, obtaining a high probability characteristic $\Delta S_1 \rightarrow \Delta S_4$.
- However, ΔS_1 has a large hamming weight, with most of the S-boxes (in round 0) being active.
- Which is actually something we desire for the target difference algorithm.
- (note that if the following fails for some ΔS_1, we can pick a different ΔS_1)

The Target Difference Algorithm Output

- Given ΔS_1, the target difference algorithm outputs an affine set of message pairs that satisfy the required input difference.
- Its size has to be large enough to contain a right pair.
- All the characteristics we considered have probability higher than 2^{-30}.
- Luckily, most returned subspaces had sufficiently large dimension (about 100 for KECCAK-224, and 35–50 for KECCAK-256).
- Unfortunately, these subspaces contain closely related message pairs, which means that some of the affine subspaces did not contain right pairs.
- In which case, we pick a different set of differences.

The Target Difference Algorithm

- Let the target difference be Δ_T with t active S-boxes and $320 - t$ non-active S-boxes.
- For each of the active S-boxes we have a list of precomputed subspace (depending on δ^{out}), stored in *input difference subset list (IDSL)*.
- All these lists are stored in the *input difference data structured (IDSD)*.
- The order in which values are stored both in IDSL and IDSD has impact on the success (and running time) of the target difference algorithm.

Basic Procedure for the Difference Phase

1. Initialize an empty linear equation system E_Δ with 1600 unknown bits of $L(\Delta S_0)$.
2. Add the $c + 8$ padding's equations to E_Δ.
3. For all $320 - t$ non-active S-boxes, add the 5 constraints to E_Δ.
4. Check for the consistency of E_Δ.
5. Iterate over the t active S-boxes (according to the IDSD ordering):
   a. Try the next 2-dimensional subset from the S-box IDSL, obtain the $5 - 2 = 3$ affine equations, and add them to E_Δ.
   b. If E_Δ is consistent, advance to the next S-box. Otherwise, change the subset from the IDSL. If all of them are exhausted, output "No Solution".
6. Output E_Δ.

The Main Procedure for the Difference Phase

1. Initialize a counter to 0. Initialize the IDSD by resetting all the IDSL pointers to the beginning of the lists, and randomizing the IDSD S-box order.
2. Execute the basic procedure. If the procedure succeeds, output E_Δ. If the procedure outputs Fail, abort. Otherwise go to Step 3.
3. Increment the counter. If the counters value is equal to $T1$, go to Step 1.
4. Reset the pointer of the failed S-box IDSL to its value before the last basic procedure.
5. Change the IDSD order by bringing the failed S-box to the front (and pushing the rest of the S-boxes one position backwards).
6. Go to step 2.

The Input Difference Subset List (IDSL)

- All the maximal input difference subsets are of dimension 2.
- There is no way to know a-priori whether the IDSL will lead to a success.
- However, we did find a (simple) heuristic concerning the order of the IDSL.
- We first note that $DDT(\delta^{in}, \delta^{out}) = 2, 4$, or 8.
- When we wish to decide between the two sets $\{\delta_1^{in}, \delta_2^{in}, \delta_3^{in}, \delta_4^{in}\}$ and $\{\delta_5^{in}, \delta_6^{in}, \delta_7^{in}, \delta_8^{in}\}$, we just compare the DDT values.
- This also increases the number of actual message pairs in the subset.

The Input Difference Data Structure (IDSD)

- The order of the S-boxes in the IDSD determines the order of adding equations to E_Δ.
- The actual order determines the success (and most importantly the time complexity).
- However, it seems that just picking the order at random works.
- Sometimes we need to restart and re-randomize the order.

Basic Procedure for the Value Phase

1. Initialize an empty linear equation system E_M with 1600.
2. Add the $c + 8$ bits of padding as constraints to the system.
3. For all t active S-boxes (according to the IDSD):
   a. Sort all possible input differences δ^{in} leading to δ^{out} in descending probability order.
   b. Add the current δ^{in} equations to E_M. If the new set is not consistent, try next possible δ^{in}.
      1. Add to E_Δ the remaining constraints on the difference following the chosen δ^{in}.
   c. Output the fully determined ΔS_0, and the equation system E_M.

Main Procedure for the Value Phase

1. Initialize a counter to 0.
2. Set E_Δ to the equation system returned by the difference phase.
3. Execute the basic procedure of the value phase. If the procedure succeeds, output ΔS_0 and E_M. Otherwise continue:
4. Increment the counter. If the counters value is equal to $T2$, output "No Solution".
5. Change the IDSD order by bringing the failed S-box to the front.
6. Go to Step 2.

Collisions in 4-Round KECCAK-224

```
M1=
FAC7AC69 2710BE04 8462C382 7ABF1BF9 D065CD30 DB64DEB8 1410CD30 C837D79B 22E446B7 31E9BD55
A6B2074C C86E32CC DE50A10A F7BAAA58 D96CBC88 9FBD75F6 5E0D735A D22AA663 16A574AA 7DB08692
558AB029 109B4D30 86CE5DCA 13A295C7 E7C9D94B 648794D2 62EE3CF8 69439337 8CAB9F15 AC7C3267
90F41CBE A20E6893 B4781F24 0BA37647 F29A67A0 81F628D0

M2=
CE5FBC81 47710FCC 462C92E0 48F5D2CF F92F6EC3 053E64E1 570780B9 F838EC54 8F74809F 66B4AC6F
70DD1843 BF34F0C5 5010C89A D8791148 D5CC073E 3239AEBC 7DF48D79 0EC7767B FB081604 AFA975B9
F8EFAE0F ED793473 479E931C F2F80A74 7192D08F 5EB5AB27 F1CAC04E F394232D 48656B2A A3471644
DB74E60A 05FB3B18 41DC27C3 8384BF53 32534C3E 811C00B5

Output=
826B10DC 0670E4E1 5B510CDA AB876AA8 B50557ED 267932FB AA4D38E8
```

Takes about 2–3 minutes on a PC.

Collisions in 4-Round KECCAK-256

```
M1=
C4F31C32 4C59AE6D 5D19F0F4 25C4E44B D8853032 8D5E12F2 BB6E6EE2 27C33B1E 6C091058 EB9002D5
3BA4A06F 4A0CC7F1 CCB55E51 8D0DD983 2B0A0843 9B21D3B0 53679075 526DDED2 48294844 6FF4ED2C
1ACE2C15 471C1DC7 D4098568 F1EBF639 EAF7B257 09FDAE87 688878E6 4875EB30 C9C32D80 3C9E6FCB
3C2ABCFA E6A4807B 2AB281B8 812332B3

M2=
A4D30EF7 80BB8F69 90C048DF EB7213B9 A6650424 3A65F63E 8C268881 B651B81F AADAFA3C EE2CA5C3
DB78EAC2 C8EAE779 442F9C35 3221E287 B3017A5A 90790712 1B1C8BDC E08B10A8 9A9D25CA 1BE7AAAC
4E2F3E9C 73717DAD 5566015A A198CFB9 5A1CA8C2 A0E3348A AE6C0BB1 3980F9E4 A4FA8B91 6E81A989
89A9BCAA E12BF1F1 30EF9595 812E8B45

Output=
61FB1891 F326B6D5 24DD94DF 73274984 05DA9A1D 3FD359B9 78B8393B F2E7990B
```

Takes about 15–30 minutes on a PC.

Some Concluding Remarks

- The 4-round collisions can be extended into 5-round near collisions (slightly more effort),
- We note that the difference phase and the value phase are independent.
- Hence, one can run the difference phase one, and "randomize" the sponge input using a few prefix blocks.
- Attacking larger variants — requires more degrees of freedom.

Some Concluding Remarks

4 Rounds of KECCAK are not Enough for Collision Resistance

Questions?

Thank you for your Attention!