Improved Differential Characteristic Searching Methods

Jiageng Chen*, Atsuko Miyaji†, Chunhua Su‡, Jesen Teh§
*Computer School, Central China Normal University, Wuhan, China.
chinkako@gmail.com
† ‡School of Information Science, Japan Advanced Institute of Science and Technology, Nomi, Japan.
{miyaji, chsu}@jaist.ac.jp
†Crest, Japan Science and Technology Agency, Kawaguchi Center, Japan.
†Graduate School of Engineering, Osaka University, Japan.
§Information Security Lab, Universiti Sains Malaysia, Malaysia.
jesen_teh@hotmail.com

***Abstract*—The success probability of differential and linear cryptanalysis against block ciphers heavily depend on finding differential or linear paths with high statistical bias compared with uniform random distribution. For large number of rounds, it is not a trivial task to find such differential or linear paths. Matsui first investigated this problem and proposed a solution based on a branch and bound algorithm in 1994. Since then, the research on finding good concrete differential or linear path did not receive much attention. In this paper, we revisit the differential attack against several S-Box based block ciphers by carefully studying the differential characteristics. Inspired by Matsui's algorithm, we provide an improved solution with the aid of several searching strategies, which enable us to find by far the best differential characteristics for the two investigated ciphers (LBlock, TWINE) efficiently. Furthermore, we provide another way to evaluate the security of ciphers against differential attack by comparing the strength of the ciphers from differential characteristic's point of view, and we also investigate the accuracy when using the active S-Box to evaluate the security margin against differential attack, which is the common method adapted when new ciphers are designed.**

***Keywords*-Differential Path, Branch and Bound algorithm, Active S-box, LBlock, TWINE.**

I. Introduction

Block cipher, especially the lightweight block cipher are increasingly applied in as smart devices and wearable systems which are playing important and ubiquitous roles in our everyday life. Differential cryptanalysis is a strong tool to attack such block ciphers. The success probability of differential cryptanalysis heavily relies on employing good differential characteristics to efficiently recover the secret key. For large number of rounds, it is not a trivial task to find such differential paths. Besides ad-hoc searching, Matsui first proposed a branch and bound algorithm [1] for DES, and he successfully found the best iterative characteristic for DES. Since then, a lot of cryptanalysis results were derived for various block ciphers, but previous work such as [2] usually put more emphasis on the key recovery phase rather than identifying the corresponding characteristic. Many of the corresponding differential characteristics were just taken from previously identified ones. Recent research focused on searching related-key differential characteristics. Since the difference will exist in both the internal states and the key, some cipher-specific techniques should be involved to do efficient searching. Representative results include [3] and [4]. However, differential characteristics in a single key model has not been investigated thoroughly. For block cipher with n-bit block size, any differential characteristic $\alpha \rightarrow \beta$ should have a probability of around 2^{-n} for large number of rounds if the cipher is well designed. Well designed ciphers will spread the overall differential probability to many paths with tiny probabilities, thus even if a significant path is identified, its probability is still far away from the actual one, making it a challenging task for the attacker. [5] recently studied the differential searching for ciphers with ARX design, while in this paper, we put our attention on ciphers with S-Box design.

For block ciphers that implement S-Boxes, usually

978-1-4673-9300-3/15 $31.00 © 2015 IEEE
DOI 10.1109/CSCloud.2015.42

the smallest number of active S-Boxes AS_r, which denotes the number of S-Boxes that has been passed by some non-zero difference, is computed for a number of rounds. Suppose the largest probability in the differential table is 2^{-x}, then the designer will choose some rounds which will have $2^{-x \times AS_r} \geq 2^{-n}$, where n is the block size in bits to resist differential attacks. Following this idea, many previous research work focused on how to derive the minimum number of active S-Boxes for both Substitution-Permutation Network (SPN) structure and Feistel structure such as [6], [7], [8], [9] and [10]. While it is convenient to evaluate the security margin of differential or linear attack by counting the number of active S-Box, it is unknown how accurate this estimation is. Intuitively, it seems that the security margin is overestimated, since it is not likely that every difference going through S-Boxes will have the best probability 2^{-x}. In other words, if the cipher is well designed, the evaluation of the number of active S-Boxes can bound a single path probability to some degree. However, it never takes the clustering effect of the characteristic into consideration. Depending on the cipher's structure, different clustering effects are exhibited that can help reveal the potential security level of the cipher, leading to better understanding and evaluation of the cipher's strength in a more comprehensive way. From both attacker and designer's point of view, good concrete differential paths or clusters are of great interest to accurately determine the security margin of the cipher instead of utilizing rough estimations. Unfortunately, besides Matsui's searching algorithm, concrete characteristic (cluster) searching techniques have not been studied widely. Recently, we noticed that there is another similar work on attacking LBlock [11]. The differentials on 12 rounds have probability about $2^{-52.08}$ and those for 16 rounds have probability about $2^{-67.53}$. Our result from another angle showed a strong evidence that the reported differentials on 15 rounds are indeed the best to date.

Contributions. We investigate the searching strategies for concrete differential (cluster) characteristic by first revisiting Matsui's branch and bound algorithm. We show that while Matsui's algorithm performs well in searching single best characteristics for some ciphers, it is not efficient for other ciphers which do not have short iterative characteristics. Furthermore, searching methods to obtain good clusters have not been thoroughly investigated. In this paper, we propose several searching strategies by improving Matsui's algorithm to first efficiently find the best single characteristic in different cipher structures, and then show that finding clusters can be achieved efficiently by using a meet-in-the-middle approach. It is especially efficient when applied to the Feistel (GFS) structure. We selected several block ciphers with both SPN and Feistel(GFS) structures (LBlock[12] and TWINE [13]) to investigate the property of the corresponding differential characteristic (clusters). We are able to derive by far the best concrete differential characteristics for the two ciphers, and our results also provide new comparisons among the ciphers from differential characteristic's point of view, which further assists the security evaluation of those ciphers from a different perspective.

Outline of the paper. Section II starts by introducing the previous differential path searching strategy with improved algorithms described later. Then in Section III, two lightweight block ciphers LBlock and TWINE are investigated by using the improved searching strategy. Finally, we conclude our result in Section IV.

II. Differential Characteristic (Cluster) Searching Strategy

In [1], Matsui proposed a branch and bound algorithm to efficiently search the high probability linear and differential path for cipher DES. As a result, it took around 100 minutes to find the best characteristic at that time on HP9735 computer. The clever branch and bound recursion algorithm allows the quick termination of the searching given the appropriate selection of the threshold value. The algorithm outputs the best n-round probability B_{n-1} (starting from round 0) based on the knowledge of the previous $n-1$ rounds best probability B_i, $0 \leq i \leq n-2$. In other words, we compute the best probability in the sequence of $B_0, B_1, ..., B_{n-1}$. Another point is that in order to compute B_{n-1}, besides the knowlege of $B_0, ..., B_{n-2}$, we will also need an estimated value of B_{n-1} which is denoted as $\overline{B_{n-1}}$. Although the algorithm works correctly as long as $\overline{B_{n-1}} \leq B_{n-1}$, the searching speed will be faster when the estimated value is close to the real value.

The differential path of DES found out by this algorithm confirmed that it can indeed find the best path so far, which was also used by Eli Biham and Adi Shamir. One thing to notice is that the path found

out by the Matsui's algorithm is actually iterative paths. The arbitrary number of rounds path is composed by the two rounds characteristic $(19600000, 00000000) \rightarrow (00000000, 19600000)$ repeatedly. Matsui's algorithm is especially efficient to identify such iterative characteristic. This is because if there exists any short iterative path, although the probability for the short iteractive path itself is not the best one, it will soon exceed other paths when first few cycles are proceeded. The assumption here is that the iterative path should be short, so that the algorithm is fast in identifying the first few cycles. After that, the iterative characteristic will stand out among other paths so that the B values are relatively high for the first many rounds. The high probability of B values means we can throw away many trails at an early stage which will speed up the searching dramatically. When apply the Matsui's algorithm to block cipher PRESENT [14], we confirmed the 15-round characteristic $(07000000, 00000700) \rightarrow (00000000, 00000009)$ used in [15] with probability 2^{-62}. The search is fast and terminates in a few seconds. And again the characteristic is also based on the 4-round iterative path $(00000000, 00004004) \rightarrow (00000004, 00000004) \rightarrow (00000909, 00000000) \rightarrow (00000000, 00004004)$.

Besides the original Matsui's searching algorithm, several previous researches take one step further to search the cluster paths such as [16]. It means that after identifying some main differential path, we also look for other paths which have the same input and output as the main path but differ in the middle process. If many such paths exist, the result probability should be potentially increased. For example in paper [16], the authors also used the cluster search for SIMON and SPECK, which result in some paths with better probability then before. For S-Box based ciphers, we can search the cluster by Algorithm 1.

Here the most important thing to notice is the threshold parameter "*BOUND*". If we do not set any restrictions, then all the paths will be searched, which is usually impossible to finish for large number of rounds. And also the search will include many paths with tiny small probability, that can hardly contribute to the final probability. Since we have the best rounds probability value B_i. The value "*BOUND*" is an experimental value which depends on the structure of the cipher. For example, in [16] "*BOUND*" was set to be 2^{-15}. We perform Algorithm 1 on block cipher PRESENT

Algorithm 1 Differential cluster characteristic searching algorithm

Input: Input and output difference α and β with probability p_0.
Output: Updated probability p for $\alpha \rightarrow \beta$ such that $p \geq p_0$.
procedure CLUSTER_SEARCH($r = 0, \alpha, \beta, N, p_0, B[N]$)
  $p = 0; pn = 1.0;$
  while $r! = N - 1$ **do**
    $r \leftarrow r + 1$
    Call $(\beta_{tmp}, p_r) \leftarrow$ **Round**$(r, \alpha, pn, sBoxNum = 1, B[N])$
    $\alpha \leftarrow \beta_{tmp}$; $pn \leftarrow p_r$
  end while
  if $\beta_{tmp} == \beta$ **then**
    $p \leftarrow p + p_r$
  end if
end procedure
procedure ROUND$(r, \alpha, pn, sBoxNum, B[N])$
  if sBoxNum ! = largest Sbox number **then** //usually max sBoxNum = 8 or 16
    $sBoxInput \leftarrow$ corresponding S-Box input of α //usually 4-bit value
    for All the possible diff output values given $sBoxInput$ **do**
      // $difftable[sBoxInput][i]$: ith diff probability for input sBoxInput
      $p\_tmp = pn * difftable[sBoxInput][i]$
      if $p\_tmp \geq B[r]$*BOUND* **then**
        Call **Round**$(r, \alpha, N, p_{tmp}, sBoxNum + 1, B[N])$
      end if
    end for
  end if
  if sBoxNum == largest Sbox number **then**
    for all the possible diff output values given $sBoxInput$ **do**
      Proceed permutation layer
      $p\_tmp = pn * difftable[sBoxInput][i]$
      if $p\_tmp \geq B[r]$*BOUND* **then**
        Return (Output diff, p_tmp)
      end if
    end for
  end if
end procedure

after we have derived the B_i values for 15 rounds. After applying Matsui's algorithm, we found around 2700 differential paths with the best probability 2^{-62}. Then we set *BOUND*$= 2^{-10}$ to run Algorithm 1 for each of the best paths derived by the original Matsui's algorithm. As a result, we are able to find that the probability for most paths are increased dramatically. The probability for some 14 rounds best ones increase from 2^{-62} to larger than 2^{-55}, and also 16 rounds characteristics better than 2^{-64} can be identified. We address this in the next Section in detail.

However, we also observed the limit of Matsui's algorithm. When applying to some recently designed lightweight block ciphers, such as LBlock and TWINE directly, we are not able to find a concrete differential path with probability that is close to $1/2^n$ where n is the block size. For example, for LBlock we limit the input difference to have hamming weight less than or equal to 2, namely, with only 1 or 2 4-bit subblocks having non-zero difference, the program will not generate result when start processing 7 rounds within reasonable amount of time despite that the best probability found is relatively high which is around 2^{-24}. One explanation is that there exists no short iterative path, so that the algorithm cannot identify one significant path quickly enough. In other words, most of the path will have relatively low probability. For LBlock, when we start the searching for 7 rounds, the number of paths searched has already reached 2^{37}, which is far more than the case for PRESENT. As a result, the algorithm fails to throw away path with small probability. If we cannot identify the best single path, it becomes very difficult for us to find a good cluster. Next we will show some strategies to work around this problem.

A. Improved Searching Strategy

As we have mentioned, the direct application of Matsui's algorithm will not give us good result for recently well designed ciphers such as LBlock and TWINE. This may infer that due to the good design of the cipher, there exists no single path which has the dominant probability, for example, larger than 2^{-n}. We notice that for ciphers with Feistel (General Feistel) structure with nibble based design such as LBlock and TWINE, some subblock (4-bit for both cases) go through S-Box and get permutated with other subblocks. In other words, the subblocks are the smallest unit during the permutation. Thus we can first search for a truncated differential or linear path with each nibble presenting one bit. For 64-bit ciphers with 4-bit subblock design, the input and output difference is shrinked down to only 16 bits, thus the truncated version should be easy to handle by applying the idea of Matsui's algorithm. We summarize the algorithm for finding best truncated path in Algorithm 2. This algorithm can also be used to find best number of active S-Box very efficiently as well. Notice that for operation at line 21, we can also apply the recursive way by going through S-Box one by one. For 16-bit truncated version, it does not make any difference for speed. For ciphers with large block size, we may proceed in this way.

After we derive the truncated path using Algorithm 2, we have no reason to stop here but to try searching clusters for underlined truncated paths before proceeding to the concrete differential characteristic searching. The method is similar to Algorithm 1 so we omit the full description here. The only different thing from Algorithm 2 is that instead of B values, we have minimum number of active S-Box for each rounds. And also we need a bound which is the extra number of active S-Box allowed in each round more than the previous found best path. However, we will show later that for the ciphers we investigated, the second minimum number of active S-Box found is much larger than the smallest one, so it seems to have little affect in accumulating probabilities.

Now after we derive the best truncated paths, which can be done very efficiently, we can start searching concrete differential characteristics by using the truncated paths as a guide. We call each of the best truncated path a structure, which can lead to many concrete differential paths. Since every structure is different, no two paths can have the same input and output difference in two different structures, we only limit our searching structure by structure. For each of the structure, we now have the knowledge of the number of active S-box for each of the rounds. Since this is the optimal case in the truncated version, we would assume that the best concrete differential characteristics will also follow the truncated path. Thus for each of the round, we throw away the paths which does not have the corresponding truncated form. Now the minimum number of active S-Box we have for each round can be treated as B values in Matsui's algorithm, thus by

Algorithm 2 Truncated differential path searching algorithm

Input: First n-1 rounds minimum number of active S-Box $AS[0], AS[1], ..., AS[N-2]$, estimate number active S-Box for nth round $\overline{AS[N-1]}$.
Output: Best n rounds active S-Box $AS[N-1]$, and truncated differential path set $\Omega = \{\alpha_i^{trunc} \rightarrow \beta_i^{trunc} | ActiveSbox(\alpha_i^{trunc} \rightarrow \beta_i^{trunc}) == AS[N-1]\}$
procedure TRUNC_SEARCH($r, N, AS[N], \Omega, roundAS[N], truncDiff[N]$)
  if $r == 0$ **then**
    // For Feistel structure, we can set the side which goes through S-box with smaller hamming weight than the other side
    for all truncated input difference α_i^{trunc} with small hamming weight **do**
      $truncDiff[r] \leftarrow \alpha_i^{trunc}$
      $as_r \leftarrow AS[N-r-2]$
      Call **truncRound**($roundAS[N], as_r, diff[N], r, N, AS[N], \Omega$)
    end for
  else
    $as_r \leftarrow AS[N-r-2]$
    for $0 \leq i \leq r-1$ **do**
      $as_r \leftarrow as_r + roundAS[i]$
    end for
    Call **truncRound**($roundAS[N], as_r, diff[N], r, N, AS[N], \Omega$)
  end if
end procedure
procedure TRUNCROUND($roundAS[N], as_r, diff[N], r, N, AS[N], \Omega$)
  // truncated 1 or 0 goes through S-Box determinstically, follow the xor rules when perform XOR operation
  perform the round operation; $\delta \leftarrow$ output diff of current round;
  $\Gamma \leftarrow$ all the possible combinations at the corresponding locations where both inputs are 1 before XOR operation
  for all $\gamma \in \Gamma$ **do**
    $\lambda \leftarrow \gamma \oplus \delta$
    tmp_as = as_r+hw(X) // assuming state X goes through S-Box, its hamming weight is computed by hw(X)
    if $tmp\_as \leq AS[N-1]$ **then**
      $truncDiff[r] \leftarrow \gamma$
      $roundAS[r] \leftarrow hw(X)$
      if $r! = N-1$ **then**
        call **trunc_search**($r+1, N, AS[N], \Omega, roundAS[N], truncDiff[N]$)
      else
        update $AS[r] \leftarrow tmp\_as$
        $\Omega \leftarrow truncDiff$
      end if
    end if
  end for
end procedure

applying Matsui's Algorithm and then Algorithm 1, we should be able to find clusters for concrete differential characteristics.

However, we found that we can actually do better than this by applying a meet-in-the-middle approach. Suppose we have derived a bunch of best single path $\alpha_i \rightarrow \beta_i$ with the same best probability. Since the number of branches will grow exponentially as the number of rounds grows, we will be able to greatly speed up the algorithm if we can limit the number of rounds to search. Thus instead of finding variant paths starting from α_i and ending at β_i, we divide the cipher into forward f rounds, and backwards b rounds. Starting from α_i, we compute f rounds and store the output difference γ_j^f along with the probability p_j^f in a hash table with γ_j be the key. Then we try

backward computing b rounds starting from β_i to γ_j^b with probability p_j^b to check if we have a matching output difference in the hash table. Add the probability $p_j^f * p_j^b$ to the total probability if there is a match. It seems to be a simple memory trade off trick to speed up the computing at the cost of more storage. But for feistel structure with subblock based permutation, there is another big advantage of this approach.

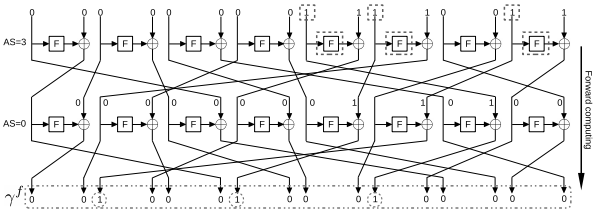

Figure 1: Memory bound (Example of block cipher TWINE)

Let's take general feistel structure (TWINE) for example. Assuming last two rounds of the forward path starting from α have active S-Box 3 and 0 as shown in Figure 1. Notice that only truncated path are marked in the figure, and it is actually the one we found for TWINE. Then the difference γ^f which need to be stored in memory has only hamming weight 3. That means no matter how large the number of round is, the memory consumption is bounded by the hamming weight of γ^f, which is $2^{3\times4} = 2^{12}$ 64-bit states in this example. Thus when we apply the middle-in-the-middle approach for feistel structures such as TWINE and LBlock, we want to find some round r in the middle, where the sum of active S-Box of round r and $r-1$ should be as small as possible. Algorithm 3 describe the procedure.

III. Differential Cryptanalysis Revisited on LBlock, TWINE

In this section, we will revisit the differential attack against several lightweight block ciphers LBlock and TWINE by applying the algorithms described in Section II.

LBlock is a 32 rounds, 64-bit block cipher with feistel structure proposed by Wenling Wu, et al. in [12]. In each round after the left 32-bit side goes through a non-linear function F, it is xored with the right side after it is left cyclic shifted for 8 bits. TWINE is also a 64-bit block cipher with GFS structure proposed by

Algorithm 3 Meet-in-the-middle approach for searching differential(cluster) characteristics

Input: Set of best single path candidates $\Omega = \{(\alpha_i, \beta_i, p_i) | p_i = Prob(\alpha_i \rightarrow \beta_i)$ is maximum$\}$; Active S-Box of each round of the best truncated path $AS[N]$.
Output: Updated Ω with $\Omega = \{(\alpha_i, \beta_i, p_i^{update}) | p_i^{update} \geq p_i\}$
procedure FORWARD_SEARCH(Γ, Ω)
    choose a meeting round r where $AS[r] + AS[r-1]$ is small.
    for $(\alpha_i, \beta_i) \in \Omega$ **do**
        Perform tree traversing starting from α_i for every branches
        Store every outputs $\Gamma \leftarrow (\alpha_i, \gamma_i^f, p_i^f)$
        call **backward_search**$(\beta_i, \Gamma, \Omega, r, p_i = 1.0)$
    end for
end procedure
procedure BACKWARD_SEARCH$(\beta_i, \Gamma, \Omega, r, p_i)$
    $(\gamma_j^b, p_j^b) \leftarrow$ Perform tree traversing starting from β_i for every branches
    if the output $\gamma_j^b \in \Gamma$ **then**
        Update $p_i = p_i + p_i^f \times p_j^b$ in Ω for charateristic $\alpha_i \rightarrow \beta_i$
    end if
end procedure

Tomoyasu Suzaki, et al. in [13]. Different from LBlock, it supports 80 and 128 bits key length which both have the same 36 rounds. F function of LBlock and round operation of TWINE are shown in Figure 2 and Figure 1.

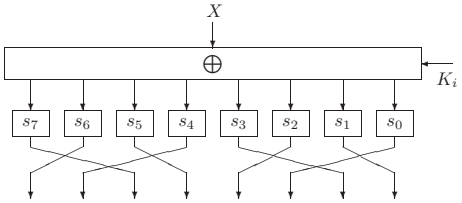

Figure 2: F function for LBlock

LBlock and TWINE are two very similar ciphers. They are all S-Box based feistel structure with nibble based permutation. The designers of both ciphers evaluate the security margin against differential or linear attack by using active S-Box. Surprisingly, both ciphers achieve best 32 active S-Box when reaching the 15 rounds. This again demonstrates the similarities of the

two ciphers. However, we will compare the two ciphers thoroughly regarding the security margin of differential attack by investigating the clusters carefully. Due to the nibble based permutation design, we proceed the searching in the following steps.

1) Apply Algorithm 2 to find the best truncated structure candidates along with the best number of active S-Box.
2) Apply Matsui's algorithm to find best single path within each truncated structure derived. Notice that using truncated structure pattern as a guide when perform the searching.
3) Apply Algorithm 3 to derive the cluster characteristic for each of the single path derived in the second step.

In the second step, in order to limit the number of single paths per structure, we set the probability bound for each of the S-Box to be $2^{-2.2}$ so that the we focus on some big probability single paths to search for the clusters. Notice that in step 3, we do not need to put any bound restriction any more, and it can finish very quickly. On the contrary. for non-meet-in-the-middle approach, we need to bound the round probability, otherwise, due to the large number paths, the speed will be very slow.

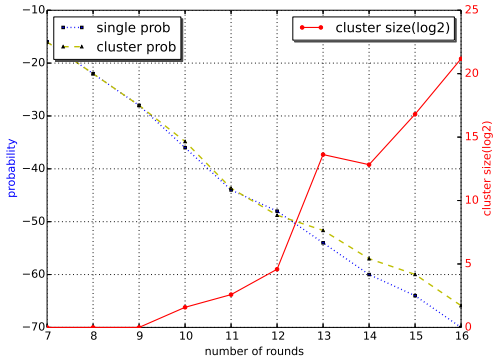

Figure 3: Differential characteristic for LBlock

From Figure 3 and 4, we can see that for number of rounds less than 10, we can hardly find clusters for path within the truncated structure. After that, clusters are starting to show up but do not contribute very much to the probability. In other words, LBlock and TWINE does a good job in spreading the differential probability to many small paths. Notice that the actual cluster size and probability should be bigger than we

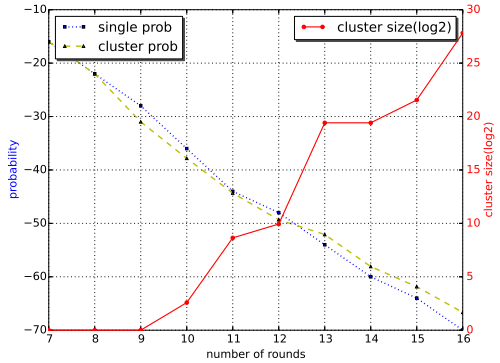

Figure 4: Differential characteristic for TWINE

have derived. This is because we only search the cluster within one specific truncated structure. As we mentioned before, there also exists clusters for each truncated structure. Take LBlock for example, for 15 rounds, the best truncated structures have 32 active S-Box, while the second smallest trucated ones have 11 more. So searching those truncated path will improve the final probabilities, but the searching is much more challenging, since we are facing many more paths with tiny probabilities.

We summarize some best concrete 15 rounds differential characteristics for both cipher shown in Table I and Table II. It confirms our early assumption that there are no single significant paths for both ciphers that is greater than 2^{-64}. However, there exists some interesting thing to compare between the two ciphers. The best probability for the cluster is around $2^{-59.9767}$ and $2^{-61.8099}$ for LBlock and TWINE. Although the difference is not so large, the number of forward paths, backward paths and cluster size differ a lot. We found out that TWINE has worse differential cluster probabilities despite that the cluster size is larger than LBlock. This shows that TWINE does a better job than LBlock in distributing the differential probabilities to many paths, and thus shows a better security margin against differential attack.

IV. Conclusion

In this paper, we revisited the differential attack by investigating the differential path carefully. For S-Box based block ciphers with bitwise or nibble-based permutations, we improved Matsui's searching algorithm and applied it to block ciphers LBlock and

Table I: Differential path for 15-round LBlock

r	α	β	Forward	Backward	Cluster	Single Prob	Cluster Prob
1	11030000	3222000200100	2696584	600248	13290	-64	-59.9767
2	11030000	3422000200900	2696584	600248	13290	-64	-59.9767
3	11030000	3522000200900	2696584	600248	13290	-64	-59.9767
4	11030000	3622000200100	2696584	600248	13290	-64	-59.9767
5	11030000	b222000200100	2696584	600248	13290	-64	-59.9767
6	11030000	b422000200900	2696584	600248	13290	-64	-59.9767
7	11030000	b522000200900	2696584	600248	13290	-64	-59.9767
8	11030000	b622000200100	2696584	600248	13290	-64	-59.9767

Table II: Differential path for 15-round TWINE

r	α	β	Forward	Backward	Cluster	Single Prob	Cluster Prob
1	60051000652	9000a70090	22763	472710	3066593	-64	-61.8536
2	60052000651	900000a790	22763	472710	3066593	-64	-61.8536
3	520006005106	909000a7000000	22763	472710	3066593	-64	-61.8536
4	510006005206	a790900000000000	22763	472710	3066593	-64	-61.8536
5	651000600520000	a700909000	22763	472710	3066593	-64	-61.8536
6	652000600510000	9090a7	22763	472710	3066593	-64	-61.8536
7	5206005100060000	9000a70090000000	22763	472710	3066593	-64	-61.8536
8	5106005200060000	90a7000090000000	22763	472710	3066593	-64	-61.8536
9	90089000983	60003f0060	9824	572237	2371915	-64	-61.9244
10	90083000989	6000003f60	9824	572237	2371915	-64	-61.9244

TWINE. As a result, we were able to find the best probability paths for all three ciphers to date. Our analysis shows that newly proposed lightweight ciphers such as LBlock and TWINE perform well in avoiding short iterative differential paths, therefore reducing the probability of the single significant path being found by attackers. By comparing the cluster sizes, our experiments show that TWINE outperforms LBlock in spreading out the differential probability to many paths. Also, we found that using active S-Boxes to evaluate the security margin against differential attack provides overly-optimistic security results as shown for the three ciphers. For LBlock and TWINE, the minimum number of active S-Boxes should be set to be one more than the current best approximations. For example, in the case of LBlock, the 16-round best cluster probability is approximately $2^{65.8}$, and it can surpass 2^{64} if we consider searching multiple truncated clusters using parallel super computing in the meet-in-the-middle phase. This will be one of our future works. Another interesting property is that for ciphers with similar structure to LBlock and TWINE, our algorithm can search the entire cluster efficiently even for longer rounds. Thus, instead of considering the differential distribution for the whole 64-bit cipher, we can consider the distribution which follows the corresponding truncated paths. Therefore, we are able to evaluate these ciphers in a highly accurate way and potentially lead to better attacks when considering truncated clusters.

ACKNOWLEDGMENT

The authors would like to thank anonymous reviewers for their valuable comments. And this work is partly supported by Grant-in-Aid for Scientific Research(C)(15K00183), (15K00189) and Grant-in-Aid for Young Scientific Research(B)(15K16005).

REFERENCES

[1] M. Matsui, "On correlation between the order of s-boxes and the strength of des," in *Advances in Cryptology — EUROCRYPT'94*, ser. Lecture Notes in

Computer Science, A. De Santis, Ed. Springer Berlin Heidelberg, 1995, vol. 950, pp. 366–375.

[2] O. Özen, K. Varıcı, C. Tezcan, and Ç. Kocair, "Lightweight block ciphers revisited: Cryptanalysis of reduced round present and hight," in *Information security and privacy*. Springer, 2009, pp. 90–107.

[3] A. Biryukov and I. Nikolifa, "Automatic search for related-key differential characteristics in byte-oriented block ciphers: Application to aes, camellia, khazad and others," in *Advances in Cryptology - EUROCRYPT 2010*, ser. Lecture Notes in Computer Science, H. Gilbert, Ed. Springer Berlin Heidelberg, 2010, vol. 6110, pp. 322–344.

[4] S. Sun, L. Hu, P. Wang, K. Qiao, X. Ma, and L. Song, "Automatic security evaluation and (related-key) differential characteristic search: Application to simon, present, lblock, des(l) and other bit-oriented block ciphers," in *Advances in Cryptology - ASIACRYPT 2014*, ser. Lecture Notes in Computer Science, P. Sarkar and T. Iwata, Eds. Springer Berlin Heidelberg, 2014, vol. 8873, pp. 158–178.

[5] A. Biryukov and V. Velichkov, "Automatic search for differential trails in arx ciphers," in *Topics in Cryptology – CT-RSA 2014*, ser. Lecture Notes in Computer Science, J. Benaloh, Ed., 2014, vol. 8366, pp. 227–250.

[6] T. Shirai, K. Shibutani, T. Akishita, S. Moriai, and T. Iwata, "The 128-bit blockcipher clefia (extended abstract)," in *Fast Software Encryption*, ser. Lecture Notes in Computer Science, A. Biryukov, Ed. Springer Berlin Heidelberg, 2007, vol. 4593, pp. 181–195.

[7] J. Daemen and V. Rijmen, "The wide trail design strategy," in *Cryptography and Coding*, ser. Lecture Notes in Computer Science, B. Honary, Ed. Springer Berlin Heidelberg, 2001, vol. 2260, pp. 222–238.

[8] M. Kanda, "Practical security evaluation against differential and linear cryptanalyses for feistel ciphers with spn round function," in *Selected Areas in Cryptography*, ser. Lecture Notes in Computer Science, D. Stinson and S. Tavares, Eds. Springer Berlin Heidelberg, 2001, vol. 2012, pp. 324–338.

[9] T. Shirai and K. Araki, "On generalized feistel structures using the diffusion switching mechanism," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 91, no. 8, pp. 2120–2129, 2008.

[10] K. Shibutani, "On the diffusion of generalized feistel structures regarding differential and linear cryptanalysis," in *Selected Areas in Cryptography*, ser. Lecture Notes in Computer Science, A. Biryukov, G. Gong, and D. Stinson, Eds. Springer Berlin Heidelberg, 2011, vol. 6544, pp. 211–228.

[11] A. Biryukov, P. Derbez, and L. Perrin, "Differential analysis and meet-in-the-middle attack against round-reduced twine," in *22nd International Workshop on Fast Software Encryption*, 2015.

[12] W. Wu and L. Zhang, "Lblock: A lightweight block cipher," in *Applied Cryptography and Network Security*, ser. Lecture Notes in Computer Science, J. Lopez and G. Tsudik, Eds., vol. 6715. Springer Berlin Heidelberg, 2011, pp. 327–344.

[13] T. Suzaki, K. Minematsu, S. Morioka, and E. Kobayashi, "TWINE: A lightweight block cipher for multiple platforms," in *Selected Areas in Cryptography*, ser. Lecture Notes in Computer Science, L. Knudsen and H. Wu, Eds. Springer Berlin Heidelberg, 2013, vol. 7707, pp. 339–354.

[14] A. Bogdanov, L. Knudsen, G. Leander, C. Paar, A. Poschmann, M. Robshaw, Y. Seurin, and C. Vikkelsoe, "Present: An ultra-lightweight block cipher," in *Cryptographic Hardware and Embedded Systems - CHES 2007*, ser. Lecture Notes in Computer Science, P. Paillier and I. Verbauwhede, Eds. Springer Berlin Heidelberg, 2007, vol. 4727, pp. 450–466.

[15] M. Wang, "Differential cryptanalysis of reduced-round present," in *Progress in Cryptology – AFRICACRYPT 2008*, ser. Lecture Notes in Computer Science, S. Vaudenay, Ed. Springer Berlin Heidelberg, 2008, vol. 5023, pp. 40–49.

[16] A. Biryukov, A. Roy, and V. Velichkov, "Differential analysis of block ciphers simon and speck," in *International Workshop on Fast Software Encryption-FSE*, 2014.