Cube attacks on Trivium

S. S. Bedi and N. Rajesh Pillai
Scientific Analysis Group,
Metcalfe House Complex, Delhi, INDIA
ssbedi53@hotmail.com, nrpillai@yahoo.com

Abstract

This paper discusses the Cube attacks proposed in [3] and [5] applied to Trivium variants. Independent verification of the equations given in [3] and [5] were carried out. Experimentation showed that the precomputed equations were not general. They are holding when applied to the class of IVs for which they were computed - where IV bits at locations other than those corresponding to the cube are fixed at 0. When these IV bits are fixed at some other values, the relations do not hold. The probable cause for this is given and an extra step to the method for equation generation is suggested to take care of such cases.

1 Introduction

Cube Attacks [3] were proposed by Dinur and Shamir as a generic class of attacks applicable to systems which can be described as polynomials. Here we consider equations for an output bit from the cipher for a fixed key and different Initialization Vectors (IVs). In cube attack one tries to obtain linear equations in the unknown key bits by combining the equations for an output bit of the cipher for a set of IVs. The IVs in a set are identical at all except k bit positions, where they take all possible combination of values. In other words, the IVs form coordinates of a k-dimensional hypercube.

By carefully choosing the cube, one can obtain linear equations involving only key bits and sum of output bits. One of the first attacks of this kind was the Algebraic IV Differential Attack (AIDA) described by Vielhaber in [5]. Vielhaber applied it on a Trivium variant with a reduced number of initialization rounds (576 instead of 1152). The paper [5] gave linear equations for the secret key bits in terms of sums of output bits which could be used for attacking the cipher. But no systematic method for obtaining such equations was described. A procedure for finding such linear equations was first given in the September 2008 on eprint [3] and later in Eurocrypt 2009 [4]. Linear equations for variants of Trivium with 672 and 767 rounds were given in [4].

Independent verification of the equations in [4] was carried out. It was found that the equations hold for the special cases considered - IV bits assigned to zero at all places except for those defining the coordinates of the cube. But the equations were not holding for the cubes when some random initial setting was used for the other bits in IV. The possible cause for this was investigated and an additional step to the method for generation of equations is suggested to obtain more general equations.

In the next section a brief introduction to cube attacks is given. In Section 3, verification (of the equations in [4]) performed and the results are given. In Section 4, the causes for equations not holding when remaining bits of IV are fixed to random values are investigated and modification to the equation search procedure is suggested. This is followed by Discussions and Conclusions.

2 Cube Attacks

Cube attacks are applicable to systems which can be modelled as a polynomial $F(K, IV) = Y$, where K is the secret key (of say n bits $x_1, ...x_n$) and IV is publicly known initialization vector (of say m bits $v_1, ..., v_m$). We use the expression $F_i(K, IV) = Y_i$ to denote the polynomial representation for the ith output bit of the cipher. The main idea of cube attacks is to combine equations for same K but various chosen IV in such a way that low degree equations in the variables in K are obtained. In particular the IVs can be chosen such that linear equations for the unknown bits in K can be obtained. Once sufficient number of linear equations are obtained, the key K can be recovered.

For obtaining such equations, the set C of chosen IVs is to be taken such that $\sum_{IV \in C} F_i(K, IV)$ is a linear combination of bits in K. Let $IV = (v_1, .., v_m)$ and $K = (x_1, .., x_n)$. Suppose that for a particular group of variables $U = \{v_{i_1}, .., v_{i_k}\}$ from the IV part, the expression for the ith output bit can be rewritten as

$$F_i(x_1, .., x_n, v_1, .., v_m) = v_{i_1} v_{i_2} .. v_{i_k} P(x_1, .., x_n, V) + Q$$

Where $P(.)$ is a polynomial over variables in $\{x_1, .., x_n\} \cup V$ with $V = \{v_1, .., v_m\} - U$ and $P(.)$ is linear over x_is. The polynomial Q is such that none of the terms in Q have the monomial $v_{i_1} v_{i_2} .. v_{i_k}$ as a factor. Let C be a set of $2^{|U|}$ points where the variables in U are taking all possible combinations of values and the variables in $\{x_1, .., x_n\} \cup V$ are fixed to some arbitrary value. Consider the sum

$$\begin{aligned} \sum_C F_i(x_1, \ldots, x_n, v_1, \ldots, v_m) &= \sum_C v_{i_1} \ldots v_{i_k} P(x_1, \ldots, x_n, V) + \sum_C Q \\ &= P(x_1, \ldots, x_n, V) \end{aligned}$$

The first summation reduces to $P(x_1, .., x_n, V)$ as the coefficient of $P(.)$ in the summation is nonzero for only one case in C. The second summation evaluates to zero as each monomial in Q gets added an even number of times (a monomial not divisible by j variables from the set U will be added 2^j times) and hence cancels out. The bit locations i for which $P(.)$ are polynomials of degree 1 are identified and stored. In the online phase, the ith output bit of the system with the same unknown key K and for all the IVs in C are xored and the result is equated to $P(.)$ to obtain a linear relation. Each such equation gives one bit of information about the key. Once n linearly independent relations over key bits are obtained, we can recover the key.

For precomputation of linear relations, the approach used in [4] was to randomly select $U = \{v_{i_1}, .., v_{i_k}\} \subset \{v_1, .., v_m\}$. The set of chosen IVs were formed by allowing variables in U to take all possible combinations of values while keeping variables in $V = \{v_1, .., v_m\} - U$ fixed to 0. A linearity check was performed for the polynomial $p(x_1, .., x_n) = P(x_1, .., x_n, 0)$. If the check was satisfied, the polynomial p was saved for use in the online phase of attack. If not, the set would be modified by adding or deleting an element from the set and the search continued.

From the construction of the equations, one can conclude that equation holds for the cube when bits at positions in V are zero. Nothing can be concluded about the $2^{|V|} - 1$ cubes where the bits at positions in V are not all simultaneously zero. It should be noted that in both [4] and [5], it was implicitly assumed that the equations obtained for the cube with variables in V fixed to 0 will be holding in general. That is they will also hold for the cubes where the variables in V are fixed to any of the $2^{|V|}$ possible combinations. Results of the verification exercises carried out for the equations given in [4] and [5] showed that the equations are not general (see Tables 1 and 2). As a result the equations obtained in [4, 5] are useful for attacking only for a restricted set of IVs

3 Verification of Equations for Trivium Variants

The precomputed equations for Trivium variants given in Tables 1 and 2 of [4] were considered. The variants considered differ from Trivium [2] only in the number of initialization rounds after which system generates outputs. The number of initialization rounds are 672 for Table 1 and 767 for Table 2 (instead of 1152 rounds as in Trivium).

In Table 1, the equations have been given as triples $(p(K), U, i)$ where $p(K)$ is the linear combination of bits from K; U is the set of IV positions which define the cube and i is the output bit position. In Table 2, there is an additional column which inidicates the elements of V which are fixed to 1. Let C denote the set of $2^{|U|}$ chosen IVs where bits at positions given in U are allowed to take all possible values and bits at remaining positions are kept fixed. Then equation denoted by the triple $(p(K), U, i)$ is

$$p(K) = \sum_{IV \in C} F_i(K, IV)$$

For computing the equations, the authors of [4] took C as the set of $2^{|U|}$ vectors where the bits in positions $V = \{1..80\} - U$ of IV (and the last column also for Table 2) are fixed to 0.

For verification, we performed the following steps. A random 80-bit vector was generated and used as K. We evaluated the coefficient polynomial P by summing output bits corresponding to the different IVs in the set C and the common key K. We then compared the result with simple polynomial evaluation of $p(x)$ on K.

When the bits at locations in V were 0, the relation ($p(x)$ =sum of output bits) was found to be holding for all the cases in Table 1 of [4]. When IV was such that bits at coordinates in V are fixed at randomly chosen values, the relation was found to be failing a large number of times. The last two columns of Table 1 show the number of times the relation was failing for the specific output bit over 1000 runs. The second last column gives count for the case when $V = 0$. The results show that the relation seems to hold with a high probability when $V = 0$. The last column gives the number of times the relation was failing when V was fixed to an arbitrary non zero value.

A similar exercise was carried out for the expressions given in [5]. In this case only 10 of the 47 polynomials were found to be holding even when bits at V were fixed to 0. When this constraint on IV was relaxed, none of the relations were holding. The last two columns of Table 2 show the number of times the relation was failing for the specific output bit for 10000 runs.

This shows that the relations in [4] and [5] may not be as general as implied.

4 Possible Causes and Modification Suggested

In this section the possible causes for the equations not holding for general case are discussed. The authors of [4] were aware of one – occurrence of a high degree terms in the coefficient polynomial for the monomial $v_{i_1}...v_{i_k}$. Since the linearity test is probabilistic, the chances of detecting presence of a high degree term is low. [1]

In this section we show that nonlinear polynomials are detected as a linear polynomial for many other cases also. To show this we first briefly describe the method suggested in [4] for detecting and finding linear coefficient polynomials p.

Let $U = \{v_{i_1}, .., v_{i_k}\}$ and V denote the complement set $\{v_1..v_m\} - U$. The expression for ith output bit can be written as

$$F_i(x_i, .., x_n, v_1, ..v_m) = v_{i_1} v_{i_2}..v_{i_k} P(x_1, ..x_n, V) + q(x_1, ..x_n, v_1, ..v_m)$$

where $v_{i_1} v_{i_2}..v_{i_k}$ does not divide any of the terms of q.

For the attack as described in [4] (and [5]) we are interested in finding U and i such that we get equations of the form

$$F_i(x_1, .., x_n, v_1, .., v_m) = v_{i_1} v_{i_2}..v_{i_k} p(x_1, ..x_n) + q(x_1, ..x_n, v_1, ..v_m)$$

where $p(.)$ is linear. To identify such U, the method suggested in [4] is to try for U of various sizes. For each U, set the remaining variables (those in V) to 0 and evaluate the cube. For the case when U is too large, the expression $\sum_{IV \in C} F_i(X, IV)$ evaluates to a constant irrespective of X. For smaller sizes, the expression will be a nonlinear polynomial. The idea is to keep trying till we hit the size in between where the sum evaluates to a polynomial of degree 1. To check if $\sum_{IV \in C} F_i(X, IV)$ is a polynomial of degree 1, the method in [4] checks if

$$\sum_{IV \in C} F_i(X + Y, IV) = \sum_{IV \in C} F_i(X, IV) + \sum_{IV \in C} F_i(Y, IV) + \sum_{IV \in C} F_i(\mathbf{0}, IV)$$

for sufficient number of randomly selected keys X and Y. If the equation holds for all the random cases tried, the polynomial P is assumed to be of the required form viz. linear in x_js and then the individual coefficients for the linear combination are calculated.

[1] Such high degree terms will contribute to the polynomial value with an equally low probability so the polynomial can be treated as linear for practical purposes and the attack will still work.

The set C of IVs used were such that bits at locations in V were fixed to 0 and bits at locations in U were allowed to run over all possible choices (Second para of section 4.2 of [4]). Because of this the polynomials of the kind $P(.) = x_1 + x_2 + v_k x_1 x_2$ for some $v_k \in V$ will also show up as linear. In fact both $x_1 + x_2 + v_{k_1} x_1 x_2$ and $x_1 + x_2 + v_{k_2} x_3$ will be detected as $x_1 + x_2$.

This shows that fixing the bits at positions in V to 0 will give us equations which need not hold for the general case. To detect such cases (with a high probability) verification of the equation for sufficient number of cases with IV bits at locations in V fixed to some random values should also be carried out. This can be done by choosing a random key X and two random settings V_1 and V_2 for the variables at positions in the set V. Check if the relation $P(X, V_1) = P(X, V_2)$ holds. If it holds for sufficient number(say 100) of Xs, then with a high probability $P(.)$ is independent of IV bits at positions in V. Once this is ensured, the other steps as given in [4] can be carried out for obtaining the equations.

4.1 Relaxation of the Maxterms

Observe that the maxterms computed in Tables 1 and 2 contain variables only from the secret key part. Where as in general case they can contain some linear terms from V, the fixed part of IV also. Using this observation, we tried to find polynomials $P(x_1, .., x_n, V)$ which were of degree 1 and having at least one variable from the secret key part.

Set of 45 equations found for Trivium with 576 initialization rounds are given in Table 3. The precomputed equations in this case will be applicable to a large set of IVs. These equations will help in reducing the effective keyspace of Trivium with 576 rounds to 2^{35}.

This approach of making equations was attempted on Trivium with 672 initial rounds. Till the time of writing this work, we were unable to obtain equations which hold for cubes of dimension up to 14. Work in this direction is still in progress.

5 Discussions

The method for finding equations given in [4] assumed that the bits in remaining positions of IV are set 0. The equations obtained though not general will be applicable for some other sets of IVs. The usefulness of a relation depends on the proportion of IVs for which it holds.

Finding cubes so that equations which hold in general may turn out to need more computation. In particular the cube dimensions may be larger than indicated in [4].

Larger cube dimension implies requirement of greater amount of crypts on same secret key setting. This may make the attack difficult to apply in practice.

Instead of precomputing equations holding in general, one can try to find equations for the particular class of IVs observed. One can look for cubes of lower dimension which give linear relations on key bits for the observed set of IVs.

Based on our experiments, we believe that using randomly chosen IVs with the additional constraint of a lower bound on Hamming weight (for IV) will reduce the chances of finding useful equations.

6 Applicability of Cube attacks

There have been discussions on the applicablity of cube attack. Some cryptographers believe that the cube attacks are applicable for systems of small degree only [1]. The reason being that it is highly unlikely that a random polynomial will have low degree maxterms.

Consider a random polynomial over n secret variables and m public variables where probability of each monomial occurring in the polynomial is 0.5. Let A denote a monomial formed by product of $d-1$ public variables. Let x denote a private variable. A monomial of the form Ax occuring in the polynomial $F(X, IV)$ will be maxterm (part of maxterm) when all the terms in F divisible by A are of degree d. This ensures that the coefficient polynomial is linear.

The number of d+1 and higher terms divisible by such a monomial other than itself are $2^{n+m-d}-1$. In a random polynomial,the probability of all these monomials simultaneously not occurring will be approximately $2^{-2^{n+m-d}+1}$ which is very small for small d. Thus for random polynomials with m public and n secret variables, it is unlikely that a maxterm of small degrees exists.

The attack is not dependent on the degree of the entire algebraic expression. Examples can be easily constructed where degree is high but cube attack is applicable. This is because the cube attack depends on the coefficients of low degree monomials over public (IV) variables turning out to be linear in secret key variables. For well designed systems, the polynomial describing the output bit in terms of the key and IV quickly becomes indistinguishable from random polynomials over the same set of variables. The probability of obtaining maxterms for random polynomials is extremely low.

7 Conclusions

Verification of the equations for cube attack in [4] for reduced round variants of Trivium was carried out. It was observed that the equations given in [4] are not general and are applicable for a very restricted class of IVs. The fact that equations identified may not hold in general was mentioned in [4] also. The reasons given were that it might be due to some high degree terms which will cause the linear equation to fail with a very low probability. We have given an example where terms which are not of high degree lead to a situation where an incorrect linear equation is detected.

Modifications to the equation generation step of cube attack is proposed to include a probabilistic check to rule out such cases. Equation generation was attempted with this modification. Equations were generated for Trivium with 576 initial rounds. The 45 linearly independent equations obtained by us are given in Table 3. With increase in the number of initialization rounds, one has to try cubes of higher dimensions to get linear relations.

Acknowledgements

The authors thank Dr P K Saxena, Director SAG, for his constant support and encouragement for the work.

References

[1] D. J. Bernstien. Why haven't cube attacks broken anything? http://cr.yp.to/cubeattacks.html.

[2] C. D. Cannière and B. Preneel. Trivium - a stream cipher construction inspired by block cipher design principle. eSTREAM, ECRYPT Stream Cipher Project, Report 2006/021, 2006. `http://www.ecrypt.eu.org/stream`.

[3] I. Dinur and A. Shamir. Cube attacks on tweakable black box polynomials. Cryptology ePrint Archive, Report 2008/385, 2008. `http://eprint.iacr.org/`.

[4] I. Dinur and A. Shamir. Cube attacks on tweakable black box polynomials. In A. Joux, editor, *EUROCRYPT*, volume 5479 of *Lecture Notes in Computer Science*, pages 278–299. Springer, 2009.

[5] M. Vielhaber. Breaking ONE.FIVIUM by AIDA an Algebraic IV Differential Attack. Cryptology ePrint Archive, Report 2007/413, 2007. `http://eprint.iacr.org/`.

Table 1: Verification of equations for Trivium with 672 initialization rounds

Verification of Equations from [4]. The count in last two columns denote the number of times equation failed in 1000 runs. A count value of 0 denotes that equation is holding for all the 1000 runs. The first count is when $V = \mathbf{0}$. The second count gives number of times equation failed when V was fixed to a random value during superpoly computation.

No.	Maxterm	Cube Indexes	Output bit index	Count (V=0)	Count Random V
1	1+x00+x09+x50	2 13 20 24 37 42 43 46 53 55 57 67	675	0	496
2	1+x00+x24	2 12 17 25 37 39 46 48 54 56 65 78	673	0	550
3	1+x01+x10+x51	3 14 21 25 38 43 44 47 54 56 58 68	674	0	527
4	1+x01+x25	3 13 18 26 38 40 47 49 55 57 66 79	672	0	503
5	1+x02+x34+x62	0 5 7 18 21 32 38 43 59 67 73 78	678	0	493
6	1+x03+x35+x63	1 6 8 19 22 33 39 44 60 68 74 79	677	0	519
7	x04	11 18 20 33 45 47 53 60 61 63 69 78	675	0	488
8	x05	5 14 16 18 27 31 37 43 48 55 63 78	677	0	503
9	x07	1 3 6 7 12 18 22 38 47 58 67 74	675	0	523
10	1+x08+x49+x68	1 12 19 23 36 41 42 45 52 54 56 66	676	0	511
11	x11	0 4 9 11 22 24 27 29 44 46 51 76	684	0	489
12	x12	0 5 8 11 13 21 22 26 36 38 53 79	673	0	470
13	x13	0 5 8 11 13 22 26 36 37 38 53 79	673	0	512
14	1+x14	2 5 7 10 14 24 27 39 49 56 57 61	672	0	500
15	x15	0 2 9 11 13 37 44 47 49 68 74 78	685	0	499
16	x16	1 6 7 12 18 21 29 33 34 45 49 70	675	0	495
17	x17	8 11 15 17 26 23 32 42 51 62 64 79	677	0	507
18	x18	0 10 16 19 28 31 43 50 53 66 69 79	676	0	498
19	x19	4 9 10 15 21 24 32 36 37 48 52 73	672	0	500
20	x20	7 10 18 20 23 25 31 45 53 63 71 78	675	0	515
21	1+x20+x50	11 16 20 22 35 43 46 51 55 58 62 63	675	0	511
22	1+x21+x66	10 13 15 17 30 37 39 42 47 57 73 79	673	0	512
23	x22	2 4 21 23 25 41 44 54 58 66 73 78	673	0	502
24	x23	3 6 14 21 23 27 32 40 54 57 70 71	672	0	502
25	1+x24	3 5 14 16 18 20 33 56 57 65 73 75	672	0	494
26	1+x28	6 11 14 19 33 39 44 52 58 60 74 79	676	0	506
27	x29	1 7 12 18 21 25 29 45 46 61 68 70	675	0	491
28	x30	2 8 13 19 22 26 30 46 47 62 69 71	674	0	512
29	x31	3 9 14 20 23 27 31 47 48 63 70 72	673	0	494
30	x32	4 10 15 21 24 28 32 48 49 64 71 73	672	0	521
31	x33	2 4 6 12 23 29 32 37 46 49 52 76	680	0	487
32	1+x34+x62	0 5 7 13 18 21 32 38 43 59 73 78	678	0	478
33	1+x35+x63	1 6 8 14 19 22 33 39 44 60 74 79	677	0	498
34	x36	2 4 5 8 15 19 27 32 35 57 71 78	677	0	503
35	x38+x56	0 3 4 9 20 28 33 41 54 58 72 79	678	0	504
36	1+x39+x57+x66	8 11 13 17 23 25 35 45 47 54 70 79	674	0	509
37	x40+x58+x64	0 6 10 16 19 31 43 50 66 69 77 79	676	0	457
38	1+x41	2 15 17 20 21 37 39 44 46 56 67 73	674	0	499
39	x42+x60	1 16 20 22 34 37 38 53 58 69 71 78	674	0	518
40	x43	2 7 14 22 41 45 48 58 68 70 72 76	673	0	526
41	x44+x62	3 14 16 18 20 23 32 46 56 57 65 73	672	0	480
42	1+x45+x64	0 6 10 16 18 28 31 43 53 69 77 79	676	0	499
43	x46+x55	2 8 11 13 28 31 35 37 49 51 68 78	684	0	495
44	x47	5 8 20 32 36 39 45 51 65 69 76 78	676	0	499
45	x48	2 4 10 14 16 22 25 44 49 51 57 78	678	0	529
46	x49+x62	1 12 19 23 36 41 42 45 52 56 69 75	676	0	496
47	x51+x62	1 7 8 13 21 23 28 30 47 68 71 75	674	0	477
48	x52	5 8 9 12 16 18 23 40 44 63 66 70	674	0	512
49	x53	2 11 21 24 32 55 57 60 63 66 70 77	675	0	500
50	1+x54+x60	4 7 10 18 20 25 50 53 61 63 71 78	675	0	496
51	x55+x64	5 12 16 19 22 36 47 55 63 71 77 79	674	0	519
52	1+x56	4 9 18 21 23 27 32 38 43 58 67 69	677	0	512
53	x57	1 7 9 14 18 21 33 40 45 49 59 68	675	0	497
54	1+x58	2 6 12 13 19 23 30 48 55 59 69 79	673	0	493
55	x60	5 7 10 13 15 17 28 40 47 73 76 79	681	0	477
56	x61	13 21 24 39 42 46 48 51 55 61 72 78	673	0	483
57	1+x62	2 4 10 11 19 34 47 55 56 58 69 77	674	0	539
58	x63	5 7 10 15 17 35 40 47 52 57 76 79	674	0	528
59	x64	8 11 13 17 23 25 35 47 62 64 68 79	673	0	471
60	x65	2 3 13 15 19 29 32 37 39 51 76 79	682	0	489
61	1+x66	5 7 10 13 15 17 35 40 52 70 76 79	678	0	497
62	1+x67	5 20 24 29 33 35 37 39 63 65 74 78	677	0	498
63	1+x68	1 12 19 23 36 41 52 54 56 66 69 75	676	0	514

Table 2: Verification of equations for Trivium with 576 initialization rounds

Verification of Equations from [5]. The count in last two columns denote the number of times equation failed in 10000 runs. A count value of 0 denotes that equation is holding for all the 10000 runs. The first count is when $V = \mathbf{0}$. The second count gives number of times equation failed when V was fixed to a random value during superpoly computation.

SNo.	Maxterm	Cube Indexes	Output bit	Count (V=0)	Count (Rand. V)
1	x01	4, 7,12,15, 2,56	597	0	4939
2	x02+x65	4, 7,12,15, 8,33	580	4965	4937
3	x03+x66+1	4, 7,12,15,14,32	580	4501	4889
4	x04	4, 7,12,15, 6,47	579	0	4869
5	x05	7,12,15, 1,79	577	4886	5014
6	x06	4, 7,12,15,41,51	611	3763	4975
7	x08	4, 7,12,15,23,54	589	4949	5011
8	x09	4, 7,12,15,36,63	589	4976	5036
9	x11	4, 7,12,15,24,41	595	0	5009
10	x14	4, 7,12,15,21,32	604	5012	4897
11	x16	4, 7,12,15,77,79	578	3750	5022
12	x17	4, 7,12,15,20,79	588	0	5069
13	x19	4, 7,12,15,23,40	587	0	4952
14	x25	4,12,15,23,49	580	4829	5022
15	x26	4,12,15,22,49	580	4925	5042
16	x27	4, 7,12,23,48	579	4548	4981
17	x36	4, 7,12,34,44	583	4849	5002
18	x38	7,12,15,49,55	580	4998	5009
19	x39	7,12,15,52,79	578	5051	5192
20	x55	4, 7,12,15,51,58	598	4461	4980
21	x56	4, 7,12,15,26,50	578	3693	5015
22	x57+x63	4, 7,12,14,24	588	5008	4990
23	x59+x65	4, 7,12,15,10,41	612	4349	5133
24	x60+x66	4,12,15,38,48	589	4967	4973
25	x61	4, 7,12,15,40,74	587	3739	5001
26	x62	4, 7,12,15,23,75	604	4931	4982
27	x63	4, 7,12,15,23,74	604	4908	4956
28	x64	4, 7,12,15, 3,30	597	0	5014
29	x65	4, 7,12,15, 2,33	580	4904	5068
30	x66	4, 7,12,15,16,34	580	4901	5096
31	x67+1	4, 7,12,15,40,65	596	4400	4978
32	x68+1	4, 7,12,15,40,64	596	4546	4986
33	x15	4,28,31,79, 3,47	581	4850	4964
34	x18	4,28,31,79, 1,69	600	4624	5012
35	x20	4,28,31,79, 3,50	598	2858	5144
36	x23	4,28,31,79, 8,12	625	4955	4999
37	x30	4,28,31,79,12,46	606	0	4992
38	x32	4,28,31,79, 1,17	606	4326	5083
39	x33	28,31,79, 2,37	591	4999	5068
40	x35	4,28,31,79,14,51	589	0	5007
41	x58+x64	4,28,31,79,35,38	588	0	4941
42	x21	2, 7, 8,12,19,45	583	4204	4969
43	x22	2, 7, 8,12,20,56	583	4429	5061
44	x10+x58	2, 8,80,19,43	583	4131	5002
45	x12+x60	2, 8,12,80,19,44	582	3780	4843
46	x58	2, 8,12,80,19,71	607	4654	5016
47	x69	2, 8,12,80,14,49	579	0	4925

Table 3: New Equations for Trivium with 576 initialization rounds

No.	$p(x_1, ..x_{80}, v_1, ..v_{80})$	Cube Indices	O/p bit index
1	x68	3,20,28,36,42,55,77,78	579
2	v77+v64+x67	18,26,36,45,61,73,78,79	579
3	v78+x66	11,18,34,37,45,51,70,79	588
4	x65	1, 3,28,34,51,61,67	581
5	x64	3,12,19,29,37,62,77	578
6	x63	8,13,21,39,53,73,74	577
7	x62	6, 7,12,13,15,16,36,73	576
8	x61	0,10,35,45,55,58,72,77	584
9	v72+v09+v08+x60	6, 7,10,27,35,36,67	581
10	x59	1,20,29,36,48,55,73	587
11	x58	8,16,19,28,52,62,69,72	586
12	x57	0,10,11,23,25,26,29,57,68,71	593
13	x56	5, 6,11,27,44,55,60,67	578
14	x55	0, 3, 7,20,21,31,66	578
15	x54	5, 6,11,44,60,65,67	577
16	v65+v64+v50+x53	17,25,27,35,54,62,63,79	581
17	v64+v63+v49+v07+x52	1, 2, 8,39,61,62,69,70	579
18	x51	15,23,32,47,49,58,76	584
19	x50	0, 5,14,23,38,48,67	584
20	x49	14,22,30,45,48,50,59,75	585
21	x48	4,29,38,43,46,47,57,66,73	586
22	x47	18,28,38,39,42,45,46,65,79	587
23	v25+x46	1,17,19,21,24,27,59,60,71	614
24	x45	9,18,25,28,43,45,55,69	590
25	1+v20+x44	2,21,29,40,57,66,73	577
26	v40+x43	1, 7, 8,32,39,42,67,74	591
27	v39+x42	7,15,29,38,41,42,50,75	592
28	1+v51+v38+x41	3, 9,12,22,30,49,52,53	589
29	x40	19,30,36,38,43,46,58,63,79	595
30	v36+x39	4, 5,21,22,37,38,39,72	595
31	1+v48+v35+x38	3, 7,11,23,44,49,50	580
32	v49+v48+v34+x37	1, 7, 9,15,46,47,59,68	582
33	v48+v47+v33+x36	7,21,23,45,46,58,74,76	584
34	v47+v46+v32+x35	22,25,41,44,45,51,55,58,67	581
35	1+v44+v31+x34	1,15,45,46,50,57,68,69	583
36	1+v45+v44+v30+v27+x33	5,22,28,31,42,43,51,75	582
37	v44+v43+v29+x32	0, 3,32,39,41,42,47,48,61	585
38	1+v41+v28+x31	4,20,37,42,43,54,64	587
39	v42+v41+v27+x30	10,11,25,26,39,40,47,56,70	588
40	1+v39+v26+x29	0, 2,11,30,40,41,53,54	589
41	v40+v39+v25+x28	18,28,37,38,42,45,46,65,79	588
42	x03	5, 9,10,11,12,42,68,77	579
43	v68+v29+x02	5, 8,12,28,31,67,74	576
44	v67+x01	9,10,19,33,41,68,77	590
45	v66+x00	3,12,37,63,65,71,74	578