ATTACK ON ORYX CIPHER – PART 1

TEAM MEMBERS
VIKASH RAJA SAMUEL SELVIN
SWATHI NAMBIAR KADALA MANIKOTH

Vikash Raja Samuel Selvin vikashrajasamuel.selvin@sjsu.edu
Swathi Nambiar Kadala Manikoth swathinambiar.kadalamanikoth@sjsu.edu

Table of Contents

Vikash Raja Samuel Selvin vikashrajasamuel.selvin@sjsu.edu
Swathi Nambiar Kadala Manikoth swathinambiar.kadalamanikoth@sjsu.edu

ORYX STREAM CIPHER PART 1

INTRODUCTION

The ORYX is a stream cipher that was designed for use with cell phones. ORYX is based on binary linear feedback shift registers (LFSRs) to protect cellular data transmissions (for wireless data services). It consists of three 32-bit LFSRs X, A, B labeled as LFSRA, LFSRB, LFSRK and a lookup table L which is a permutation of integer values between 0 and 255. The ORYX generates 1 byte of keystream per step. Due to mistakes in design the strength of ORYX is only 16 bits and any signal can be cracked after the first 25 bytes are observed.

PROBLEM STATEMENT

Given the permutation table L and the implementation of the ORYX cipher find the initial fill of the 3 registers X, A, B. The password for the challenge is of the form X||A||B in hexadecimal written as one word.

UNDERSTANDING OF THE CIPHER

The ORYX cipher key consists of the initial fills of the registers X, A and B. Given the values of the initial fills of registers X, A and B, the keystream bytes which are used to encrypt the plain text are generated. Each byte of the key stream is generated using separate iteration which uses high bits of X, A, B and a look up table.

Every iteration of the ORYX cipher works as follows

1) Register X right steps once.
2) If the value of 29th bit of register X is 0, then the register A right steps once using P_{A0} to generate the feedback bit (i.e the bit 0 of A), otherwise P_{A1} is used to fill the feedback bit.
3) If the 26th bit of register X is 0 then register B steps once or it steps twice.
4) The key stream is generated using the below equation

   keyStreamByte = (H(X) + L[H(A)] + L[H(B)]) mod 256

   where H(X) are the highest 8 bits of X
   L is the look up table, which contains the permutation of values from 0 to 255 and its known to the cryptanalyst. [1]

Vikash Raja Samuel Selvin vikashrajasamuel.selvin@sjsu.edu
Swathi Nambiar Kadala Manikoth swathinambiar.kadalamanikoth@sjsu.edu

DESCRIBING THE ATTACK

The ORYX attack needs some bytes of keystream to be known. Since ORYX cipher is a stream cipher known plain text together with the cipher text will yield the key stream. In general, we need only 25 bytes of key stream to be known before we start the attack. [1]

Attack goes as follows:

For calculating the keystream we are considering only high 8 bits of X, A and B. So 8 bits can hold value from 0 to 255. Trying all combinations of value for A and B we will get 256 * 256 = 65536 combinations of A and B. So for each combination of values we will use the first known key stream byte and calculate the X value using the formula.

$$\mathbf{H(X) = (k_i - L[H(A)] - L[H(B)]) \bmod 256}$$

Then we need to extend each combinations of A, B and extend X.
A and B can be extended in 12 ways as shown in the table:

Number	Right Shift A by	Right Shift B by	A fill value at 24th Bit	B fill value at 24th and 25th (if two shifts) or 24th (1 shift)
0	1	1	0	0
1	1	1	0	1
2	1	1	1	0
3	1	1	1	1
4	1	2	0	00
5	1	2	0	01
6	1	2	0	10
7	1	2	0	11
8	1	2	1	00
9	1	2	1	01
10	1	2	1	10
11	1	2	1	11

X can be extended in two ways

Number	Right Shift X by	X fill value at position 24
0	1	0
1	1	1

For each extension pair of A and B, we need to find out the X value using the above formula with k_i and i = 1 in this case.

Each X value calculated above needs to be matched with the two extended values of X. If anyone matches that combination of A and B are considered for the next iteration.
While right shifting the value of A, B and X we need to store the last bit which we lose every time and after 24 iterations, we will get the first 24 bits of A, B, and X. The last 8 bits can be calculated using the high of A, B and X at that moment.

RESULTS

The different combinations of A and B reduced drastically every time and it attained 1 at some position.

Iteration	Number of combinations of A and B
1	65536
2	6248
3	574
4	74
5	11
6	1
7	1
8	1
9	1

After the 24 iteration we found the initial values of X, A and b to be 0xbeefdead, 0xabcdaaaa and 0xbbbbabcd.

IMPLEMENTATION DETAILS IN C

1. Created two lists head and tmpHead.
2. Initially created 65536 combinations of A, B, computed X for all the values and inserted those into the list head.
3. For each element in the list, extended A, B with 12 possible combinations and validated each of the combinations with two extend values of X.
4. If any of the 65536 * 12 * 2 combinations is valid, its inserted into the new list tmpHead. After each iteration in 65536, that specific element is freed from list head.
5. Finally list head will be empty and now tmpHead will contain the extended values.
6. Each extended list will also contain the bit that is dropped while right shifting the variable.
7. Now the tmpHead is assigned to head list and iteration continues.

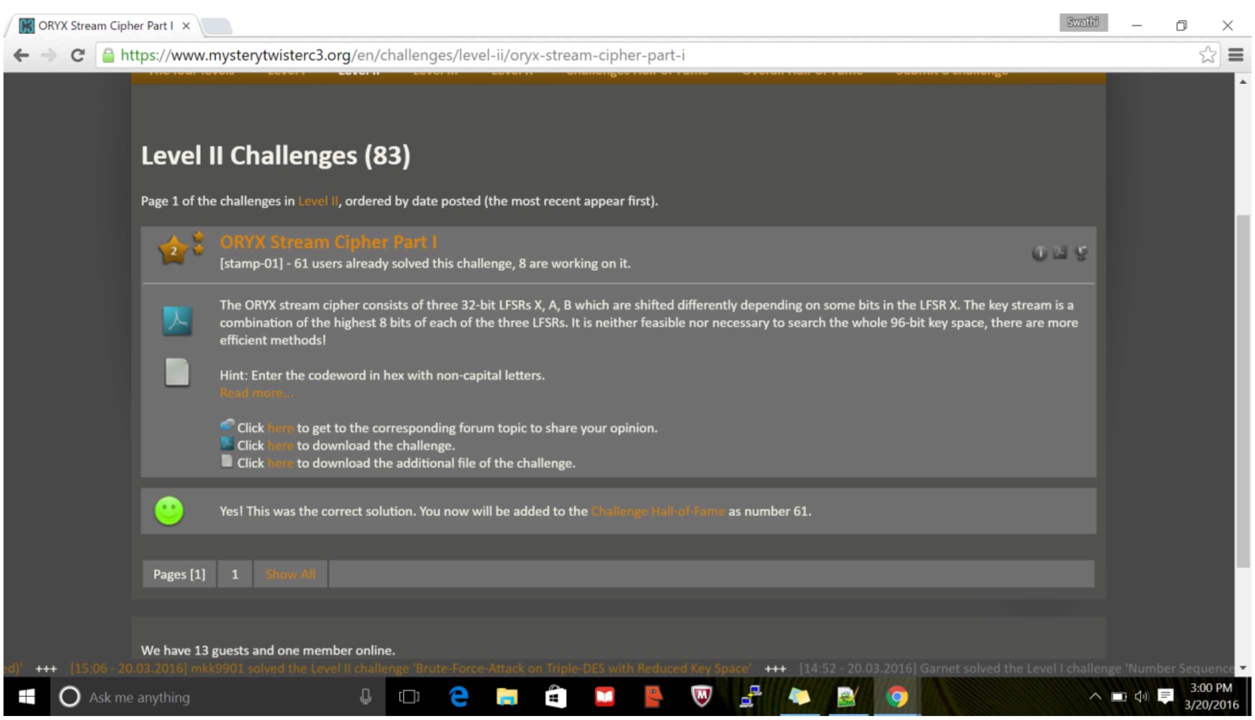
Vikash Raja Samuel Selvin vikashrajasamuel.selvin@sjsu.edu
Swathi Nambiar Kadala Manikoth swathinambiar.kadalamanikoth@sjsu.edu

8. When the total iteration reaches 24, the iteration is stopped and high values of X and A will contain the 8 MSB of X and A respectively. The variable which stored the dropped bits of variables while extending will contain the last 24 initial bits of X and A.
9. Since b would have crossed 32 shifts before 24 iterations, (B may shift twice), the variable which stores the dropped bits of B will contain the initial Fills of B. (We stopped storing bits when B reached 32 bit shift)
10. Finally X, A and B are left shifted once to get their initial fills. (B may shift twice).
    While doing left shift we need to consider again 24 ways

No	Left Shift X by	Left Shift A by	Left shift B by	Fill value for X	Fill value for A	Fill value for B	Key X||A||B
1	1	1	1	0	0	0	BEEFDEAC ABCDAAAA BBBBABCC
2	1	1	1	0	0	1	BEEFDEAC ABCDAAAA BBBBABCD
3	1	1	1	0	1	0	BEEFDEAC ABCDAAAB BBBBABCC
4	1	1	1	0	1	1	BEEFDEAC ABCDAAAB BBBBABCD
5	1	1	1	1	0	0	BEEFDEAD ABCDAAAA BBBBABCC
6	1	1	1	1	0	1	BEEFDEAD ABCDAAAA BBBBABCD
7	1	1	1	1	1	0	BEEFDEAD ABCDAAAB BBBBABCC
8	1	1	1	1	1	1	BEEFDEAD ABCDAAAB BBBBABCD
9	1	1	2	0	0	00	BEEFDEAC ABCDAAAA 77775798
10	1	1	2	0	0	01	BEEFDEAC ABCDAAAA 77775799
11	1	1	2	0	0	10	BEEFDEAC ABCDAAAA 7777579A
12	1	1	2	0	0	11	BEEFDEAC ABCDAAAA 7777579B
13	1	1	2	0	1	00	BEEFDEAC ABCDAAAB 77775798
14	1	1	2	0	1	01	BEEFDEAC ABCDAAAB 77775799
15	1	1	2	0	1	10	BEEFDEAC ABCDAAAB 7777579A

16	**1**	**1**	**2**	**0**	**1**	**11**	**BEEFDEAC ABCDAAAB 7777579B**
17	**1**	**1**	**2**	**1**	**0**	**00**	**BEEFDEAD ABCDAAAA 77775798**
18	**1**	**1**	**2**	**1**	**0**	**01**	**BEEFDEAD ABCDAAAA 77775799**
19	**1**	**1**	**2**	**1**	**0**	**10**	**BEEFDEAD ABCDAAAA 7777579A**
20	**1**	**1**	**2**	**1**	**0**	**11**	**BEEFDEAD ABCDAAAA 7777579B**
21	**1**	**1**	**2**	**1**	**1**	**00**	**BEEFDEAD ABCDAAAB 77775798**
22	**1**	**1**	**2**	**1**	**1**	**01**	**BEEFDEAD ABCDAAAB 77775799**
23	**1**	**1**	**2**	**1**	**1**	**10**	**BEEFDEAD ABCDAAAB 7777579A**
24	**1**	**1**	**2**	**1**	**1**	**11**	**BEEFDEAD ABCDAAAB 7777579B**

OUTPUT VERIFICATION



Vikash Raja Samuel Selvin vikashrajasamuel.selvin@sjsu.edu
Swathi Nambiar Kadala Manikoth swathinambiar.kadalamanikoth@sjsu.edu

Challenges Hall-of-Fame

All solvents in challenge ORYX Stream Cipher Part I

Level II
Start date: 2010-10-14 - 13:30

Rank (#62)	User	Achieved points (Global score)	Solved at y-m-d - h:m:s
#59	Vikash (vikash72)	1098 (1098)	2016-03-19 07:20:02
#60	Karl Schutt (günter)	1098 (34469)	2016-03-19 15:49:34
#61	Swathi Nambiar (Swathi18)	1098 (1098)	2016-03-20 22:59:25

```
<---------------- Different possible values of X, A, B -------------->

Different Combinations  X        A        B
X<<1|0  A<<1|0  B<<1|0  BEEFDEAC ABCDAAAA BBBBABCC
X<<1|0  A<<1|0  B<<1|1  BEEFDEAC ABCDAAAA BBBBABCD
X<<1|0  A<<1|1  B<<1|0  BEEFDEAC ABCDAAAB BBBBABCC
X<<1|0  A<<1|1  B<<1|1  BEEFDEAC ABCDAAAB BBBBABCD
X<<1|1  A<<1|0  B<<1|0  BEEFDEAD ABCDAAAA BBBBABCC
X<<1|1  A<<1|0  B<<1|1  BEEFDEAD ABCDAAAA BBBBABCD
X<<1|1  A<<1|1  B<<1|0  BEEFDEAD ABCDAAAB BBBBABCC
X<<1|1  A<<1|1  B<<1|1  BEEFDEAD ABCDAAAB BBBBABCD
X<<1|0  A<<1|0  B<<2|00 BEEFDEAC ABCDAAAA 77775798
X<<1|0  A<<1|0  B<<2|01 BEEFDEAC ABCDAAAA 77775799
X<<1|0  A<<1|0  B<<2|10 BEEFDEAC ABCDAAAA 7777579A
X<<1|0  A<<1|0  B<<2|11 BEEFDEAC ABCDAAAA 7777579B
X<<1|0  A<<1|1  B<<2|00 BEEFDEAC ABCDAAAB 77775798
X<<1|0  A<<1|1  B<<2|01 BEEFDEAC ABCDAAAB 77775799
X<<1|0  A<<1|1  B<<2|10 BEEFDEAC ABCDAAAB 7777579A
X<<1|0  A<<1|1  B<<2|11 BEEFDEAC ABCDAAAB 7777579B
X<<1|1  A<<1|0  B<<2|00 BEEFDEAD ABCDAAAA 77775798
X<<1|1  A<<1|0  B<<2|01 BEEFDEAD ABCDAAAA 77775799
X<<1|1  A<<1|0  B<<2|10 BEEFDEAD ABCDAAAA 7777579A
X<<1|1  A<<1|0  B<<2|11 BEEFDEAD ABCDAAAA 7777579B
X<<1|1  A<<1|1  B<<2|00 BEEFDEAD ABCDAAAB 77775798
X<<1|1  A<<1|1  B<<2|01 BEEFDEAD ABCDAAAB 77775799
X<<1|1  A<<1|1  B<<2|10 BEEFDEAD ABCDAAAB 7777579A
X<<1|1  A<<1|1  B<<2|11 BEEFDEAD ABCDAAAB 7777579B
```

FUTURE WORK

- The downside of Oryx is that it generates a byte stream which exposes a lot of information on the keystream. Instead of bytes, a bit stream could be generated and used.
- Instead of using the linear equation for finding the keystream, the cipher could be modified in a way that it uses non-linear equation which could thwart linear cryptanalysis.
- The number of shift registers can be increased, this would make the attacks costlier.
- After generating the 2^16 values for H(A), H(B) we could parallelize the process of finding the corresponding X,A,B.

REFERENCES

1. Stamp M, Low R.M, Applied Cryptanalysis.
2. ORYX (encryption algorithm). Retrieved from: https://en.wikipedia.org/wiki/ORYX_%28encryption_algorithm%29