The Truth on TPy

Eli Biham* Jennifer Seberry†

February 29, 2008

Abstract

In this note we refer to the security of the Py and TPy families, as discussed in several papers in recent years. We investigate the published attacks, and show that **no attacks on TPy, TPypy and TPy6 have ever been published**, whose complexities are within the bounds set by the security claims of these ciphers. Also, only a single attack on Py was published which may be considered within the scope of the security claims — as a response to this attack the Py family was tweaked to TPy, TPypy and TPy6. We conclude that the TPy family of ciphers is highly secure, and recommend using them in applications that require secure stream ciphers.

1 Introduction

In this note we investigate the claims of the papers that study the Py and TPy families [2, 3, 4], and discuss their accuracy.

As stated in the original paper on Py [2], the inventors of Py and TPy welcome any technique for the cryptanalysis or study of these families. The inventors of Py and TPy also believe that many of the techniques discussed in the various papers are interesting and important for the further study of these families, and point out some possible weaknesses that may be used in the future to attack these families. As such, we are especially pleased to thank the authors of [18] who found a weakness of the IV setup of the Py family, and thus allowed us to improve the ciphers, by a tweak, leading to the TPy family. We also thank all the authors that put their faith in the design of the Py and TPy families, and based their ciphers on these families (e.g., [15, 16]).

Though the papers studying the Py and TPy families are interesting on their own, in many cases the quotes and conclusions are inaccurate. In this note we

*Computer Science department, Technion, Haifa 32000, Israel. biham@cs.technion.ac.il, http://www.cs.technion.ac.il/∼biham/.

†Centre for Computer and Information Security Research, SCSSE, University of Wollongong, 2522, NSW, Australia. j.seberry@uow.edu.au, http://www.uow.edu.au/∼jennie/.

draw attention to the main facts and claims given in various papers regarding the security of the Py and TPy families, and the meaning of their results. We do not discuss here the details of the attacks, whether they are accurate or not in their details, or whether they are correct at all. Following our discussion on these papers, the reader will be able to assess for himself the validity of the claims of these attacking papers.

We start with some general cryptographic background that is used later in the discussion. We then discuss attacks that accept the rules, i.e., whose requirements and complexities are within the bounds set by the security claims on the ciphers, and are also within the bounds set by generic attacks that apply for all ciphers. We then discuss non-attacks (that do not accept the rules).

2 General Cryptographic Background

2.1 The Security Claims on the Py Family

The following security claims are quoted from [2]:

> Py is a stream cipher designed for very fast and secure encryption of extremely long streams. It is intended for use with keys of up to 256 bits (32 bytes), and initial values (IVs) of up to 128 bits (16 bytes) (but it also allows longer keys of up to 256 bytes, and IV sizes up to 64 bytes; keys and IVs should be in multiples of a byte, and at least one byte in length). The streams generated for a given pair of key and IV is restricted to lengths of up to 2^{64} bytes, which is long enough for any practical purpose, but does not require the extra precautions that would be required for still longer streams. In particular, protection against distinguishing attacks on variants with longer streams may require additional operations to verify that the distribution of the values of the 32-bit output words is extremely close to uniform, and that these words are extremely independent.
> ...
>
> The goals of this design is to be extremely fast, as well as very secure. It was designed to satisfy the following security properties:
>
> 1. For keys of up to 256 bits, and key streams of up to 2^{64} bytes, no attack can find the key, or expand a key stream, with a complexity less than of exhaustive search.
> 2. There are no distinguishing attacks that succeed given less than 2^{64} bytes of key stream, with a complexity less than of exhaustive search.

The same claims apply to Pypy, TPy and TPypy. Py6 (and thus also TPy6) is addressed as follows:

> A second variant of Py, called Py6, intended for shorter streams, is also presented. ... As the indices in this variant are shorter, we restrict the length of the generated streams to 2^{40}.

2.2 Relations Between Key-Recovery and Distinguishing Attacks

Kerckhoffs' principle states that the adversary knows the cipher used to encrypt the data, while only the key is unknown. An example for a distinguishing attack on stream ciphers that may recover useful information is given in [1, Section 2.3], but it is rarely applicable in practice. Therefore, we might arrive to the conclusion that distinguishing attacks are usually of low importance in the common cryptographic context.

However, most key-recovery attacks are based on some distinguishers (usually of fewer rounds, when the cipher is iterating a round function), while the rest of the rounds are analyzed by a technique that extracts the key. This behavior is common in block ciphers, hash function, and also in some stream ciphers. In this context, distinguishing attacks are very important for the design and study of key-recovery attacks, as they serve as the base for key-recovery attacks, and at the same time may be used to predict (or bound) the complexity of future key-recovery attacks.

As any key-recovery attack may also serve to distinguish whether the cipher is the claimed one or not, the complexity of any key recovery attack must be greater than or equal to the most efficient distinguishing attack. In other words, $T_{DIST} \leq T_{KR} \leq 2^k$ where k is the key size in bits, T_{KR} is the complexity of the best key-recovery attack, and T_{DIST} is the complexity of the best distinguishing attack.

2.2.1 Time/Memory/Data Tradeoffs

Time/memory/data tradeoffs is one of the kinds of generic key-recovery attacks that are applicable for any stream cipher. Attacks of this kind give upper bounds on the complexity of valid key-recovery attacks, and as such on the complexity of valid distinguishing attacks. Any attack whose complexity is larger than the ones predicted by a time/memory/data tradeoff attack is not valid in the sense that it is not better than previously known attacks.

Consider the generic time/memory/data tradeoff attack of [6, 9], whose tradeoff curve is $TM^2D^2 = N^2$, where T is the on-line time complexity of the attack, M is the memory size required by the attack, D is the keystream size, $N = 2^{k+v}$, and k and v are the key size and IV size, respectively. In the context of eSTREAM [7], where the suggested key size is 256 bits and the IV size is 128 bits, a (time/memory/data tradeoff) key-recovery attack of the type described in [6], with data, time and memory complexities of $T = M = D = 2^{154}$, exists for *any* stream cipher, with precomputation time $P = 2^{230}$. Another alternative

that exists for *any* stream cipher with these parameters has $T = D = P = 2^{192}$ and $M = 2^{96}$, in which case the cost of the precomputation is not larger than the complexity of the on-line phase of the attack.

3 Attacks that Accept the Rules

The attacks of [18, 10] show a weakness in the IV setup of the Py family, that allows the adversary to recover several bytes of a key using about 2^{23} streams that differ in the IV. Following these attacks, a tweak was made, which corrects the IV setup. The new ciphers form the TPy family of stream ciphers.

4 Non-Attacks (that do not Accept the Rules)

All the rest of the "attacks" on the Py and TPy families turn out to be distinguishing attacks. As such, their claims should be compared to the security claims on distinguishing attacks, i.e., keys up to 256 bits and total size of streams of 2^{64} bits, as can be easily seen in Section 2.1 (as quoted from [2]), in particular in item 2: "*There are no distinguishing attacks that succeed given less than* 2^{64} *bytes of key stream, with a complexity less than of exhaustive search*".

1. In [16] a related-key distinguishing attack on TPypy, TPy, Pypy and Py is developed, whose data complexity is $2^{192.3}$. Quoting the paper,

   > Under related keys, we show a distinguishing attack on TPypy with data complexity $2^{192.3}$ which is lower than the previous best known attack on the cipher by a factor of 2^{88}. It is shown that the above attack also works on the other members TPy, Pypy and Py. ...
   >
   > Such related key pairs are used to build a distinguisher for each of the aforementioned ciphers with $2^{193.7}$ output words and comparable time (note that, in total, there are 2^{2048} such pairs, while our distinguisher needs any $2^{193.7}$ randomly chosen pairs of keys). ... However, the attack complexities increase with shorter keys.

   The variants affected by the attack of [16] have 2048-bit keys, and the attacks require a total of $2^{193.7}$ bytes of streams (or have shorter keys with a higher number of required stream bytes). The paper continues,

   > This result constitutes the best attack on the strongest member of the Py-family of ciphers TPypy; they are also shown to be effective on the other members TPy, Pypy and Py.

We remind the reader that the security claims on the Py and TPy families ([2], also quoted in this note in Section 2.1) are explicitly limited to keys of up to 256 bits, and explicitly address the total size of the streams: "*There are no distinguishing attacks that succeed given less than* 2^{64} *bytes of key stream, with a complexity less than of exhaustive search*".

2. In [15] another distinguishing attack is reported, which requires $2^{224.6}$ data and comparable time against TPy6 and Py6. As these ciphers are the weaker variants in the family, the size of the allowed data size is only 2^{40} (rather than 2^{64}, see [2] or Section 2.1 of this note). We conclude that these attacks require a factor of about 2^{180} more data than the security claims cover.

3. In [12] a distinguishing attack on Py that requires the first 24 bytes of each of about $2^{89.2}$ randomly chosen keys/IV pairs is designed. The abstract of [12] acknowledges that

   > The Py specification allows a 256-bit key and a keystream of 2^{64} bytes per key/IV.

   but ignores the security claims regarding distinguishing attacks, that limit the total data size to 2^{64}. Furthermore, the introduction also states

   > Therefore, we believe that our results present a theoretical break of the cipher; see Sect. 9 for an elaborate discussion on this issue.

   and Section 9 states

   > Do Our Distinguishing Attacks on Py Violate the Designers' Claims? The stream cipher Py is claimed by the designers to have up to 256-bit security (see Appendix A of [3]). In the authors' words, "The security claims are for keys up to 256 bits (32 bytes) and IVs up to 128 bits (16 bytes)". 256-bit is also the category of security level under which Py is included in the ECRYPT project [6]. According to the discussion on the definition of *n-bit security* of a perfectly secure stream cipher, it is clear that this claim is compromised by our attacks.
   >
   > However, in Sect. 6.1 of [3], the authors claim, "There are no distinguishing attacks that succeed given less than 2^{64} bytes of key stream with a complexity less than of exhaustive search." It is understood from [2], that those 2^{64} bytes, as mentioned in the claim, may be generated by many keys rather than a single key. Under this interpretation, our attacks do not violate this claim, since our best attack requires $2^{87.7}$ bytes of output.
   >
   > As a result we conclude that two claims, mentioned above, contradict each other with respect to the attacks mentioned in this

> paper. At this point, we leave it to the reader to decide on the implications of our distinguishers.

In short, the authors agree that they break the rules, but claim that the designers of Py were too strict in their security claims. At the same time, they show that the designers of Py knew what they were doing when they were strict in their security claims.

Certainly, eSTREAM could complain if they would think that the security claims are too strict for the rules of the project, but they did not complain when they accepted Py as a candidate. Moreover, the Call for Stream Cipher Primitives of eSTREAM [8] does not indicate any major importance for distinguishing attacks. It addresses distinguishing attacks as follows:

> distinguishing attacks are likely to be of interest to the cryptographic community. However the relative importance of high complexity distinguishing attacks may become an issue for wider discussion.

We concur with this assessment concerning distinguishing attacks. Therefore, our designs protect against all practical types of distinguishing attacks, and our paper clearly addresses the security claims regarding distinguishing attacks. On top of that, the TPy family of stream ciphers suggests an excellent tradeoff between speed and security, that would not be possible if the goal would be to have a perfect solution against attacks that require almost unlimited sizes of streams, and huge keys of more than 256 bits.

4. In [5], the techniques of [12] are improved by a different author to require only 2^{72} bytes of streams. Yet, the author correctly refers to the security claims with

   > Surprisingly, this attack is disallowed by the security goals set out in [2], which limit the attacker to at most 2^{64} bytes of keystream total.

   It is actually not so surprising, as this kind of attacks were considered during the design process, when the designers made the decision to output two words (instead of a single word) in each round, and end up with a much faster cipher.

5. In [11], a distinguishing attack against Py6 that requires $2^{68.61}$ data is described. The authors say

   > we stress that when such a PRBG turns out to be extremely fast – such as Py, Py6, IA, ISAAC, NGG, GGHN – an alert message

> should better be issued for the designers to recheck that they are free from the weaknesses described here.

The designers of Py, not only discussed this issue in the original design paper [2], but also fully addressed this issue in their explicit security claims. Moreover, [11] claims that

> Although the cipher Py, a variant of Py6, was successfully attacked [15,5], Py6 has so far remained alive.

Their references to [15,5] are references [12, 5] in our note. We already discussed their claims above.

6. The paper [13] continues with unsupported quotes

> The main achievement of the paper is the detection of input-output correlations of TPypy that allow us to build a distinguisher with 2^{281} randomly chosen key/IVs and as many output words (each key generating one output word). *The cipher TPypy was claimed by the designers to be secure with keysize up to 256 bytes, i.e., 2048 bits.* Our results establish that the TPypy fails to provide adequate security if the keysize is longer than 35 bytes, i.e., 280 bits. *Note that the distinguisher is built within the design specifications of the cipher.*

We have typeset two of the quoted sentences in italics, where the authors give *unsupported* claims, which are not in line with the real security claims on TPypy (which are limited to 256-bit keys only, rather than to 2048 bits).

Then that paper says

> When TPypy is used with key of size longer than 35 bytes (or, 280 bits), our attack is better than the exhaustive key search and therefore, constitutes an academic break of the cipher. ... These weaknesses result in the first attacks on TPypy, TPy and TPy6.

and even

> Note that if the keysize is more than 35 bytes (or, 280 bits), this attack can be built within the design specifications.

No further words are needed on our part.

7. In [14] the same authors write again

> The designers claimed that TPy would be secure with a key size up to 256 bytes, i.e., 2048 bits. ... This paper shows how to build a distinguisher with 2^{275} key/IVs and one output word per each key (i.e., the distinguisher can be constructed within the design specifications).

They continue

> If the ciphers are used with key size longer than 275 bits then our attacks are better than exhaustive search. It is also worth noting that the distinguisher can be built within the design specifications of the ciphers.

Again, we refer the reader to the real security claims of the Py and TPy families.

8. In [17] another set of authors design a distinguishing attack on TPypy that requires 2^{199} words (i.e., 2^{201} bytes) of streams. The authors do not address any reference to whether this is a valid attack or not, and do not address the security claims of the attacked cipher.

Note that the attacks of [13, 14, 15, 11] are addressed by Daniel Bernstein [1] as non-attacks.

In addition, generic time/memory/data tradeoff attacks can recover keys of 256-bit stream ciphers with 128-bit IV's with complexity 2^{154} for any stream cipher (with precomputation time of 2^{230}), e.g., for all the eSTREAM candidates. Similar attacks with complexity 2^{192} exist with precomputation time that is bounded by the complexity of the on-line phase (i.e., the precomputation can be considered as part of the on-line phase). As any key-recovery attack is also a distinguishing attack, any stream cipher submitted to eSTREAM is vulnerable to distinguishing attacks with these complexities.

5 Summary

The Py family of stream ciphers was removed from consideration in phase 3 of the eSTREAM project based on various non-attacks and wrong claims, before the IV setup attack was published.

Following the extensive research on the security of the Py and TPy families of stream ciphers, and the lack of any attack on the TPy family, we strongly believe that this family is composed of highly secure ciphers, with an excellent tradeoff between speed and security. We recommend their use in applications in which such highly secure stream ciphers are needed.

References

[1] Daniel J. Bernstein, *Which eSTREAM ciphers have been broken?*, Ecrypt report 2008/010.

[2] Eli Biham, Jennifer Seberry, **Py** *(Roo, רו): A Fast and Secure Stream Cipher using Rolling Arrays*, submitted to the eSTREAM project, 2005.

[3] Eli Biham, Jennifer Seberry, **Pypy** *(Roopy, רופי): Another Version of TPy*, submitted to the eSTREAM project, 2006.

[4] Eli Biham, Jennifer Seberry, *Tweaking the IV Setup of the Py Family of Ciphers – The Ciphers Tpy, TPypy, and TPy6*, Published on the author's webpage at `http://www.cs.technion.ac.il/~biham/`, January 25, 2007.

[5] Paul Crowley, *Improved cryptanalysis of Py*, presented in SASC 2006, 2006.

[6] Orr Dunkelman, Nathan Keller, *Treatment of the Initial Value in Time-Memory-Data Tradeoff Attacks on Stream Ciphers*, SASC 2008.

[7] Ecrypt project, eSTREAM project web page, `http://www.ecrypt.eu.org/stream/`.

[8] Ecrypt project, *Call for Stream Cipher Primitives*, Version 1.3, 12th April 2005, http://www.ecrypt.eu.org/stream/call/.

[9] Jin Hong, Palash Sarkar, *New Applications of Time Memory Data Tradeoffs*, ASIACRYPT 2005, LNCS 3788, pp. 353–372, 2005.

[10] Takanori Isobe, Toshihiro Ohigashi, Hidenori Kuwakado and Masakatu Morii, *How to Break Py and Pypy by a Chosen-IV Attack*, SASC 2007, 2007.

[11] Souradyuti Paul, Bart Preneel, *On the (In)security of Stream Ciphers Based on Arrays and Modular Addition*, ASIACRYPT 2006, LNCS 4284, pp. 69–83, 2006.

[12] Souradyuti Paul, Bart Preneel and Gautham Sekar, *Distinguishing Attacks on the Stream Cipher Py*, Fast Software Encryption, FSE 2006.

[13] Gautham Sekar, Souradyuti Paul and Bart Preneel, *Weaknesses in the Pseudorandom Bit Generation Algorithms of the Stream Ciphers Tpypy and Tpy*, EUROCRYPT 2007.

[14] Gautham Sekar, Souradyuti Paul and Bart Preneel, *New Weaknesses in the Keystream Generation Algorithms of the Stream Ciphers Tpy and Py*, technical report.

[15] Gautham Sekar, Souradyuti Paul, Bart Preneel, *New Attacks on the Stream Cipher TPy6 and Design of New Ciphers the TPy6-A and the TPy6-B*, Eprint 2007/436.

[16] Gautham Sekar, Souradyuti Paul, Bart Preneel, *Related-Key Attacks on the Py-Family of Ciphers and an Approach to Repair the Weaknesses*, Indocrypt 2007, LNCS 4859, pp. 58–72, 2007.

[17] Yukiyasu Tsunoo, Teruo Saito, Takeshi Kawabata, Hiroki Nakashima, *Distinguishing Attack Against TPypy*, SAC 2007, LNCS 4876, pp. 396–407, 2007.

[18] Hongjun Wu, Bart Preneel, *Differential cryptanalysis of the stream ciphers Py, Py6 and Pypy*, EUROCRYPT 2007, LNCS 4515, pp. 276–290, 2007.