Hybrid WBC: Secure and Efficient White-Box Encryption Schemes

Jihoon Cho[1], Kyu Young Choi[1], Orr Dunkelman[2], Nathan Keller[3], Dukjae Moon[1], and Aviya Vaidberg[3]

[1] Security Research Group, Samsung SDS, Inc., Republic of Korea
{jihoon1.cho,ky12.choi,dukjae.moon}@samsung.com
[2] Computer Science Department, University of Haifa, Israel
orrd@cs.haifa.ac.il
[3] Department of Mathematics, Bar-Ilan University, Israel
nkeller@math.biu.ac.il,aviya.v5@gmail.com

Abstract. White-box cryptography aims at providing security against an adversary that has access to the encryption process. Numerous white-box encryption schemes were proposed since the introduction of white-box cryptography by Chow et al. in 2002. However, most of them are slow, and thus, can be used in practice only to protect very small amounts of information, such as encryption keys.
In this extended abstract we present a new threat model for white-box cryptography which corresponds to the practical abilities of the adversary in a wide range of applications. Furthermore, we study design criteria for white-box primitives that are important from the industry point of view. Finally, we propose a class of new primitives that combine a white-box algorithm with a standard block cipher to obtain white-box protection for encrypting long messages, with high security and reasonable performance.

1 Introduction

The standard threat model considered in secret-key cryptography is the *black-box* model, in which the endpoints of communication channels are assumed to be secure, and thus, an adversary may only obtain (plaintext,ciphertext) pairs but no information on the encryption process itself. In 1996, Kocher introduced the *gray-box* model, in which the adversary may obtain *side-channel* information on the encryption process, such as execution time and power consumption. In 2002, Chow et al. [4] introduced the *white-box* model, in which the adversary is accessible to the entire information on the encryption process, and can even change parts of it at will.

The range of applications in which the white-box threat model corresponds to the practical abilities of the adversary is already extensive and continues to grow rapidly. One example is the Digital Rights Management (DRM) realm, where the legitimate user (who, of course, has full access to the encryption process), may be adversarial. Another example is resource-constrained Internet-of-Things (IoT) devices applied in an insecure environment (like RFID tags

on the products in a supermarket). Yet another example is smartphones and public cloud services. While certain security-critical services in such devices are provided with support of hardware security features, such as 'secure element' or TrustZone in mobile devices or 'hardware security modules' in the cloud, most services are implemented as software operating within Rich OS. The main reasons for that are low cost, development efficiency and complicated ecosystems. As a result, the cryptographic implementations are vulnerable to a wide variety of attacks in which the adversary has 'white-box' capabilities.

The ever-growing range of applications where the white-box threat model is relevant necessitates devising secure and efficient solutions for white-box cryptography. And indeed, numerous white-box primitives were proposed since the introduction of white-box cryptography in 2002. These primitives can be roughly divided into two classes.

The first class includes algorithms which take an existing block cipher (usually AES or DES), and use various methods to 'obfuscate' the encryption process, so that a white-box adversary will not be able to extract the secret key. Pioneered by Chow et al. [4], this approach was followed by quite a few designers. An obvious advantage of these designs is their relation to the original ciphers, which makes transition to the white-box primitive and compatibility with other systems much easier. Unfortunately, most of these designs were broken by practical attacks a short time after their presentation. Another disadvantage of the designs in this class is their performance – all of them are orders of magnitude slower than the 'black-box' primitives they are based upon.

The second class includes new block ciphers designed especially with white-box protection in mind. Recent designs of this class include the ASASA and SPACE families [1,2]. An important advantage of these designs is their better performance and higher security (though, some of them were also broken, see [5]). On the other hand, transition from existing designs to the entirely new ciphers is not an easy task, and so, quite often commercial users will be reluctant to make such a major change in the design.

In this extended abstract we propose a class of new primitives which, on the one hand, provide strong security with respect to a 'real-life' white-box adversary, and on the other hand, are convenient for practical use – meaning that the performance is reasonable and that transition from currently used primitives to the new primitives is relatively easy. To this end, in Section 2 we present a new threat model for white-box cryptography which corresponds to the practical abilities of the adversary in a wide range of applications. Once the security model is set, we study design criteria for white-box primitives that are important from the industry point of view. In Section 3 we propose a class of new primitives that combine a white-box algorithm with a standard block cipher to obtain white-box protection for encrypting long messages, with high security and reasonable performance. Preliminary security analysis of the new primitives, along with a comparison with previous works, can be found in the full version of this paper [3].

2 Practical Requirements and Design Strategy

2.1 Security requirements – a new threat model

Unlike the classical black-box model, in white-box cryptography the abilities of the adversary are not clearly defined, and different threat models are implicitly used by different authors. The basic intuition is that the adversary can 'do everything', but of course, this cannot be assumed as then no secret can be kept from the adversary whatsoever.

The works of Chow et al. [4] and their successors implicitly assume that there is a part of the encryption process, called *external encoding*, which is performed outside of the encryption device and cannot be accessed by the white-box adversary. Such an assumption is not realistic in scenarios where the entire encryption process is implemented in software.

Instead, we propose the following threat model, which is relevant in a wide variety in realistic scenarios. Assume that the same white-box encryption scheme is used in many devices, with at most a small difference between them (e.g., a unique identification number that is used in the encryption process). Further, assume that the adversary can mount an 'expensive' white-box attack on at most a few devices (e.g., by purchasing them and then analyzing in depth), and he is willing to break the encryption of *all other devices*. Formally, we assume that the adversary has a white-box access to several devices from the family and only black-box access to all devices in the family. Using the white-box access, the adversary can obtain full information on the devices he took control of. His goal is to break the encryption schemes of all other devices. Thus, the security goal in this model can be thought of as *minimizing the damage from one-time compromise*.

Our threat model is well suited for IoT environment. IoT devices are usually manufactured in a production line simply assembling flash memories with the same binary programmed including cryptographic keys, i.e. the same cryptographic keys are shared across multiple devices. This is because it would be quite expensive to embed separate keys into each device either in production lines or by consumers; additional key-embedding process and related key management, as well as adding UX layers to IoT devices, generally require considerable cost. In such an IoT environment, an adversary may implement the white-box attack for a single device, and try to compromise the whole system using the obtained key or any critical information, along with capabilities from the conventional black-box model.

We note that this threat model does not fit for *all* applications of white-box cryptography. However, it seems relevant in sufficiently many scenarios for being considered specifically.

2.2 Performance and cost requirements

While industry accepts the need in strong security of the algorithms, it is often the case that practical efficiency considerations are prioritized by commercial

users over security considerations. Hence, if we want to design a primitive that will be employed in practice, we should take into account the main practical requirements from the industry point of view.

The main two design criteria we concentrate on are the following:

Reasonable performance. Previously suggested white-box algorithms except the SPACE family are 12 to 55 times slower than AES. White-box primitives have thus been used to protect relatively small sizes of data. We aim at using the white-box primitive to protect large amounts of data, and so, the encryption speed must be reasonably fast – ideally, almost as fast as the AES.

Low transition cost. The new architecture should be designed so as to minimize the modification of the existing development or manufacturing process related to cryptographic implementations. Interestingly, this may be the most important factor for commercial adoption in reality.

2.3 Design strategies

The practical requirements listed above lead to the following design considerations.

First, if we use a white-box algorithm to encrypt each block of the message then the performance of the resulting encryption scheme is the same as that of the white-box algorithm. For most of the currently existing white-box algorithms, this means that the scheme is very slow. Moreover, even for the SPACE family whose members are not so slow, standard 'software obfuscation techniques' aimed at protecting the security of the running code, make the encryption process much slower, and thus too slow for our purposes. As a result, it is desirable to use the white-box algorithm to encrypt only part of the message blocks, and encrypt most blocks with a 'classical' algorithm.

Second, almost all existing solutions for data protection in data communication such as SSL, TLS and SSH are based on a shared secret (e.g. session key). Designers of some solutions for data communication want to apply this session key in white-box encryption with minimum modification of their cryptographic implementation. However, they cannot use this key directly in a white-box scheme since the initiation of a white-box algorithm is slow and in general is separate from running environment. In addition, in many cases users request a certificate algorithm to be used in their implementation. Hence, we aim at applying a session key directly in the components of our scheme, except the white-box algorithm.

Third, the most effective way to minimize the damage from one-time compromise is to encrypt each message by a one-time key which is protected by white-box algorithms. However, managing these one-time keys is a big burden and existing key exchange protocols do not provide a one-time session key. Thus, we will encrypt the nonce by a white-box algorithm and use it in the encryption process as a replacement for a one-time key.

3 The New Primitives

3.1 General structure and security goals

Our primitives use two separate keys – one for a white-box primitive and another for a 'classical' encryption algorithm (e.g., AES), where the white-box algorithm is only used for encryption of a nonce (e.g. initial vector (IV) or a counter) while the classical algorithm is used for encryption of plaintexts. The keys K_1 and K_2 are assumed to be permanent and may be shared by many devices, while the nonce in changed in every encryption session.

We restrict the use of our scheme to encrypting messages of length at most 2^{64} blocks in a single session (i.e. without rekeying). Obviously, this amount is sufficient for any practical purpose. Furthermore, as common in nonce-based algorithms, we do not allow re-use of the nonce.

The security level we aim at is data complexity of 2^{64} and memory and time complexities of 2^{80}. That is, any white-box attack that can recover the secret key K_1, or distinguish our scheme from random, or recover part of the plaintext in a non-compromised session, should require either more than 2^{64} messages, or more than 2^{80} time or more than 2^{80} memory.

Note that in each compromised session, the adversary can recover the full plaintext/ciphertext, as well as the key K_2 and the nonce (since only K_1 is white-box protected). However, as the random nonce acts as a one-time key for this structure, mere knowledge of the nonce does not help to attack other sessions.)[1]

3.2 The new Hybrid White-box schemes

In this subsection we present two new hybrid white-box schemes, which – according to our preliminary analysis – are secure in the white-box model.

The first scheme, called F-CTR-WBC and presented in Figure 1, is similar to the standard CTR mode of operation using the AES block cipher, but with three differences. First, a counter CTR is encrypted using a white-box primitive (e.g., white-box-AES or a member of the SPACE family). Second, the scheme contains a feed-forward operation (in order to thwart a trivial attack in the white-box model presented in [3]). Third, the block length is increased to 256 bits (e.g., by using Rijndael-256 instead of AES), in order to make a time-memory tradeoff attack presented in [3] infeasible. Our experiments show that this scheme is only 1.3 times slower than AES-CTR.

The second scheme we propose, presented in Figure 2, is a bit more complex, using AES with feed-forward also in the counter update function. If the full AES is used in both layers of the scheme, it is almost two times slower than F-CTR-WBC with Rijndael-256. However, as the upper layer is used mainly

[1] It may be possible to lift the whole binary of the white-box algorithm and then run it in a simulator, in which case the white-box primitive itself is acts as a key. We assume that the system has a counter-measure against such code-lifting attacks, e.g. an additional coding scheme, node-locking techniques, etc.

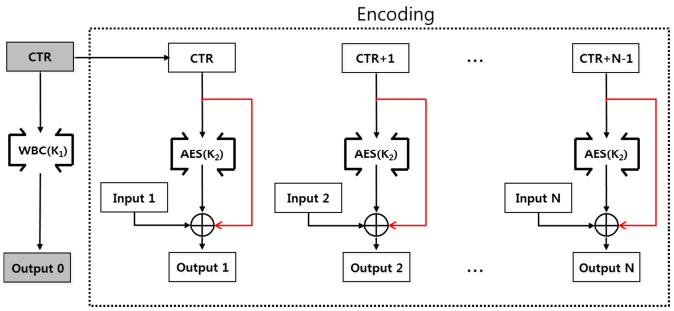

Fig. 1. F-CTR-WBC: A white-box variant of AES-CTR with a 256-bit block and a feed-forward operation

to reduce the relation between consecutive inputs to the second-layer AES and their relation to the initial CTR, it is actually sufficient to use 3-round AES-128 in the upper layer. As a result, this scheme has roughly the same performance like F-CTR-WBC presented above.
Initial security analysis of both schemes is presented in [3].

References

1. A. Biryukov, C. Bouillaguet, and D. Khovratovich, *Cryptographic Schemes Based on the ASASA Structure: Black-Box, White-Box, and Public-key*, proceedings of ASIACRYPT 2014, volume 8873 of Lecture Notes in Computer Science, pp. 63–84, Springer, 2014.
2. A. Bogdanov and T. Isobe, *White-box Cryptography Revisited: Space-Hard Ciphers*, proceedings of Computer and Communications Security (CCS) 2015, pp. 1058–1069, ACM, 2015.
3. J. Cho, K. Y. Choi, O. Dunkelman, N. Keller, D. Moon, and A. Vaidberg, *Hybrid WBC: Secure and Efficient White-Box Encryption Schemes*, IACR eprint report 2016:679.
4. S. Chow, P. A. Eisen, H. Johnson, and P. C. van Oorschot, *White-Box Cryptography and an AES Implementation*, proceedings of Selected Areas in Cryptography (SAC) 2002, volume 2595 of Lecture Notes in Computer Science, pp. 250-270. Springer, 2002.
5. H. Gilbert, J. Plût, and J. Treger, *Key-Recovery Attack on the ASASA Cryptosystem with Expanding S-Boxes*, proceedings of CRYPTO 2015 (1), volume 9215 of Lecture Notes in Computer Science, pp. 475–490, Springer, 2015.

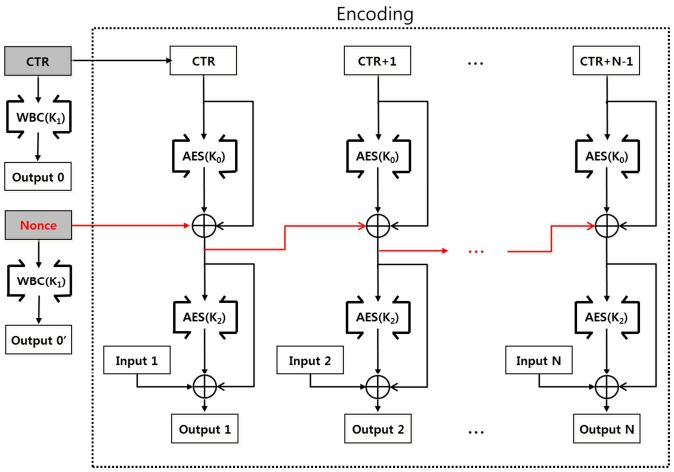

Fig. 2. UF-CTR-WBC: A Two-layered variant with feed-forwards