# A Tutorial of White-Box Cryptography Chapter 2 Notions and Definitions

Zheng Gong[1,2]
cis.gong@gmail.com

[1]School of Computer Science, South China Normal University
[2]Mobile Applications And Security Engineering Center of Guangdong Province

August 15, 2018

# The categories of white-box crypto: from crypto side

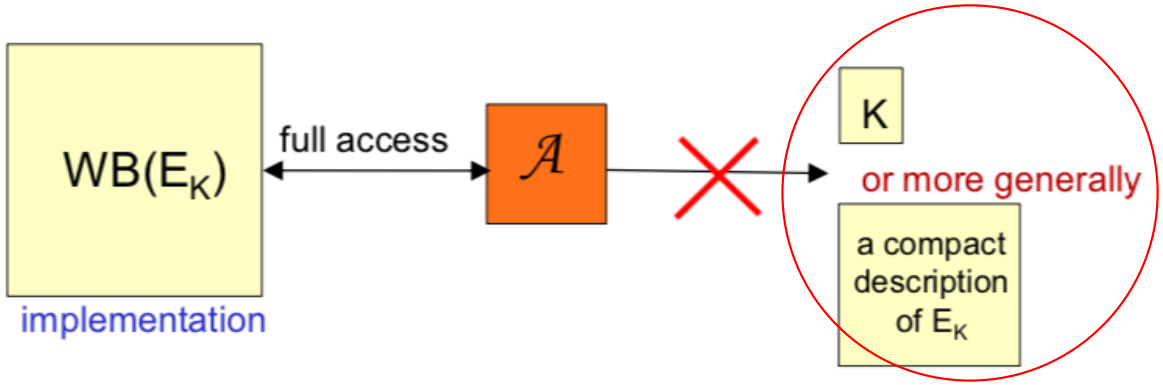
- symmetric-key
  - block cipher
  - stream cipher
  - message authentication code
  - pseudorandom generator
- asymmetric-key
  - public-key decryption
  - signature
  - multi-party computation

# The categories of white-box crypto: from white-box side

- key escrow resisntance:
  - Standalone protection: N-SPACE, SPNBox
  - Server-aided protection: Cloud signature
  - Complexity hardness: Chow et al.'s seminal work
- Code lifting resistance:
  - Obfuscation: Space/code/time hardness
  - Watermarking
  - Tamper-proofing

# Basic security definitions for white-box crypto (1)

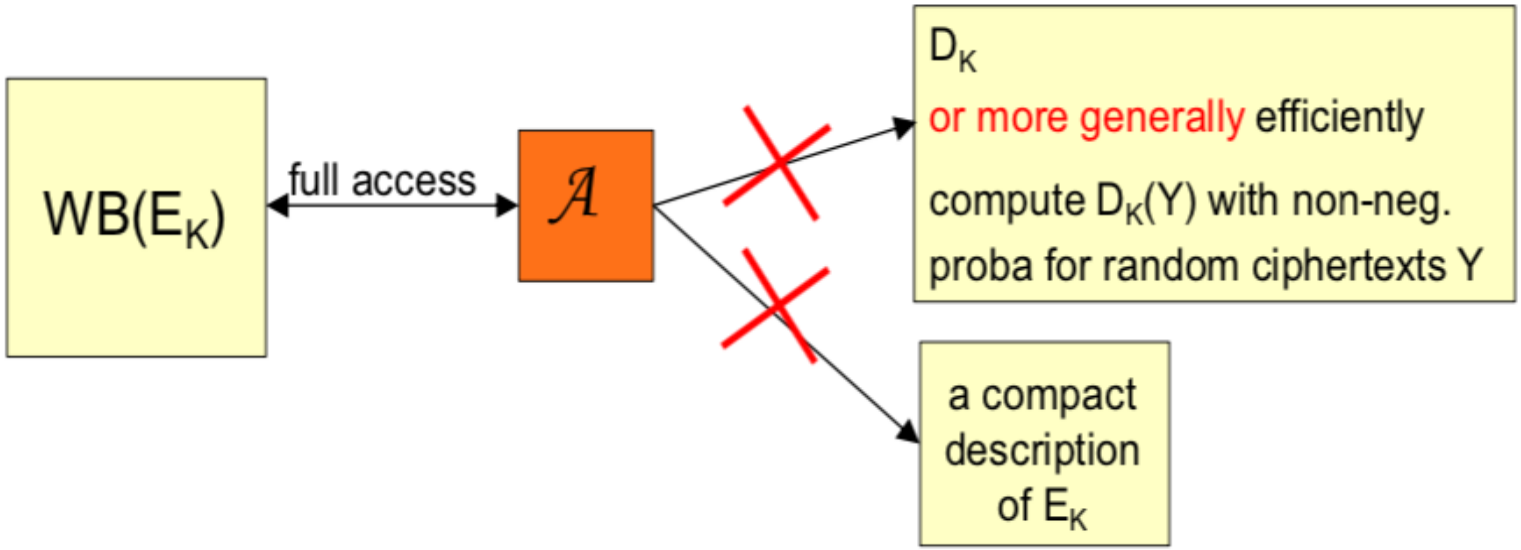
- In SAC 2002, the security notions have been informally described for white-box cryptography by Chow *et al.*. First the key recovery problem is informally defined by the weak white-box security




informal definition: (T, S)-incompressible implementation of $E_K$.
an adversary with full access to WB($E_K$) must be unable to derive
an equivalent* representation of $E_K$ of size lower than S in time T. **

# Basic security definitions for white-box crypto(2)

- For more general security, the strong white-box security has been defined by Chow et al.



# The shortcomings of traditional crypto model

- In the traditional crypto model, e.g., chosen-plaintext/ciphertext attacks, are defined for protecting Kerckhoff's principle. The security goal is just to protect the function (signing, decrypting, etc.)
- In the white-box security model, attackers already have the ability to use the keyed function, therefore the goal is moved to protect the secrete key or the integrity of the function (not the function itself)!
- How to define this security model and goal is a challenging work!

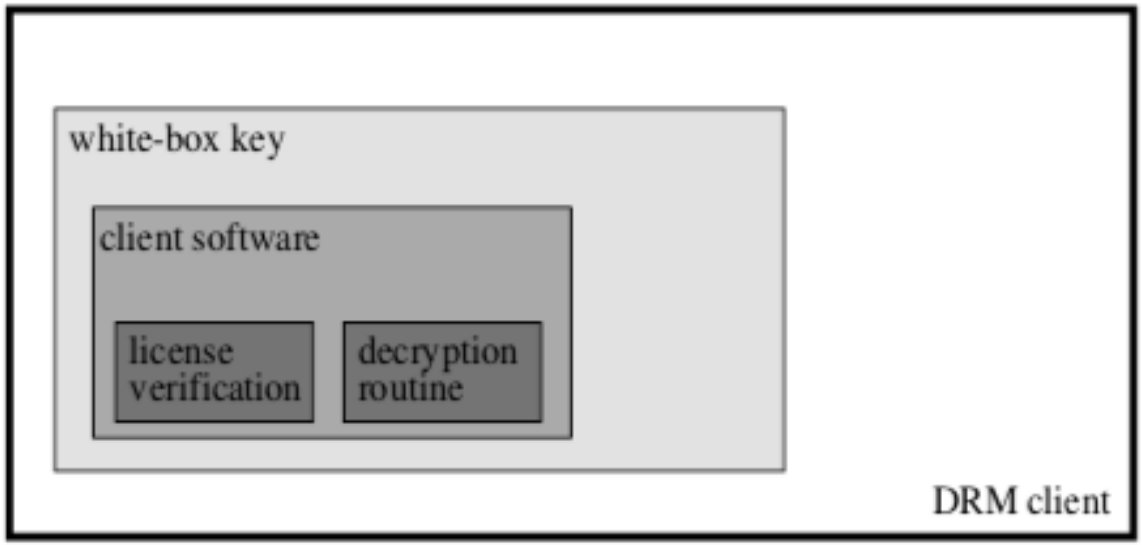
# Tamper resistance for DRM

- Michiels and Gorissen [DRM'07] proposed a method to protect the integrity of software which depends on the correct operation of the white-box implementation of a block cipher.
- If an attacker modifies the software, the white-box implementation stops decrypting/encrypting properly.

**security goal**

- The proposed method assume that it is the goal of an attacker to modify the protected software without losing the ability to decrypt/encrypt properly.

# Tamper resistance with white-box crypto for DRM



# The threat model of Michiels and Gorissen's proposal

1. A DRM client that is implemented in software and that has to validate conditions in a DRM license before it decrypts the corresponding content.

2. The content can be encrypted by AES as this block cipher allows a white-box implementation, which may only be decrypted during a specific time window.

3. An attacker may try to get around the license by tampering with the program code that verifies the license.

4. We need not only protect the license verification routine, but also (part of) the software that calls this routine.

# The steps of Michiels and Gorissen's proposal

1. Let B be the binary of the software that we want to protect.

2. Binary B can be linked and compiled code obtained from higher lever source code, such as C, but it can also be the binary representation of interpreted code, such as byte code in Java.

3. As binary B is just a string of bits, we can also interpret it as a collection of lookup tables. A code fragment of 1024 bytes can, for instance, be interpreted as a lookup table consisting of 256 rows of 4 bytes.

# The published standards

As a cryptographic product, white-box crypto also has to be regulated and governed by standards. There are two widely-accepted standards for the security of cryptographic modules:

- FIPS 140-2 (U.S. Standard, internationally accepted)
- GM/T 0028/0039-2014 (Chinese Standard, learn from FIPS 140-2)

# FIPS 140-2

In FIPS 140-2, 4 security levels are defined for cryptographic modules to be evaluated

- Level 1: Use a standard cryptographic algorithm and implement validately
- Level 2: Achieve Level 1 whilst enforce the role-base authentication on the cryptographic module
- Level 3: Achieve Level 2 whilst enforce the identity-base authentication on the cryptographic module with physical protection
- Level 4: Achieve Level 3, intrusion resistance with immediate zeroization, self-test

# FIPS 140-2: Level 1

Security Level 1 provides the lowest level of security. Basic security requirements are specified for a cryptographic module (e.g., at least one Approved algorithm or Approved security function shall be used). No specific physical security mechanisms are required in a Security Level 1 cryptographic module beyond the basic requirement for production-grade components. An example of a Security Level 1 cryptographic module is a personal computer (PC) encryption board.

# FIPS 140-2: Level 2

Security Level 2 requires, at a minimum, role-based authentication in which a cryptographic module authenticates the authorization of an operator to assume a specific role and perform a corresponding set of services.

Security Level 2 allows the software and firmware components of a cryptographic module to be executed on a general purpose computing system using an operating system that

- meets the functional requirements specified in the Common Criteria (CC) Protection Profiles (PPs) listed in Annex B and
- is evaluated at the CC evaluation assurance level EAL2 (or higher).

# FIPS 140-2: Level 3

Security Level 3 requires identity-based authentication mechanisms, enhancing the security provided by the role-based authentication mechanisms specified for Security Level 2. A cryptographic module authenticates the identity of an operator and verifies that the identified operator is authorized to assume a specific role and perform a corresponding set of services.

Security Level 3 requires the entry or output of plaintext CSPs (including the entry or output of plaintext CSPs using split knowledge procedures) be performed using ports that are physically separated from other ports, or interfaces that are logically separated using a trusted path from other interfaces. Plaintext CSPs may be entered into or output from the cryptographic module in encrypted form (in which case they may travel through enclosing or intervening systems).

# FIPS 140-2: Level 4

Security Level 4 provides the highest level of security defined in this standard. At this security level, the physical security mechanisms provide a complete envelope of protection around the cryptographic module with the intent of detecting and responding to all unauthorized attempts at physical access. Penetration of the cryptographic module enclosure from any direction has a very high probability of being detected, resulting in the immediate zeroization of all plaintext CSPs. Security Level 4 cryptographic modules are useful for operation in physically unprotected environments.

Security Level 4 also protects a cryptographic module against a security compromise due to environmental conditions or fluctuations outside of the module's normal operating ranges for voltage and temperature. Intentional excursions beyond the normal operating ranges may be used by an attacker to thwart a cryptographic module's defenses. A cryptographic module is required to either include special environmental protection features designed to detect fluctuations and zeroize CSPs, or to undergo rigorous environmental failure testing to provide a reasonable assurance that the module will not be affected by fluctuations outside of the normal operating range in a manner that can compromise the security of the module.

# Chinese standard GM/T 0028/0039-2014

- Also divided into 4 security levels and very similar to FIPS 140-2
- Add Security Level 2+ and 3+ to fix the gap between software and hardware security assurances.
- Only for Chinese commercial cryptographic schemes and protocols (SM1/2/3/4/9 series)

# Conclusion

- Security notions and definition for precisely analyze white-box crypto are at very begin
- Key protection definitions might not suitable in practice for non-key white-box applications, e.g., secure multi-party computation
- How can we define Strong/Weak?

# Thanks for your attentions!