Evaluation and minimization of kleptography risks in cryptographic algorithms

Bohdan Kovalenko and Anton Kudin

NTUU "KPI", Kyiv, Ukraine

Zagreb, Croatia, 2020

Contents

01 Objectives

Kleptography researches are quite relevant now because of many factors:

1. dissemination of new kleptographic attack vectors on practical cryptosystems;
2. insufficient formalizing of kleptographic models and underdevelopment of kleptography theory;
3. the inefficiency of classical cryptanalysis for kleptography trapdoor detection;
4. a lot of practical cryptosystems requires components that are working outside the secured perimeter.

Goals of the research are:

1. formalize one class of kleptographic mechanisms;
2. suggest a metric of kleptographic risks for the considered class;
3. suggest methods of kleptographic risks reducing.

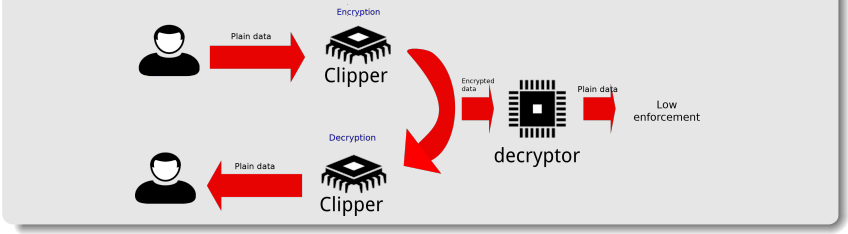
What is kleptography system?

1. kleptography system is a cryptographic system with an implemented kleptography mechanism (trapdoor).
2. A trapdoor is implemented and exploited by *Developer* (kind of Attacker which has the capability to manipulate design and implementation of cryptosystem).
3. The kleptography mechanism allows Developer to perform some practical cryptoanalysis (e.g., to recover of victim's secret key or to search for hash collisions in a practical time).
4. Other cryptosystem actors can't use these mechanisms to simplify cryptoanalysis.
   + Undetectability property (optional): nobody can detect the kleptography mechanism.
   + Strong undetectability property (optional): nobody can detect even *the fact* of the mechanism existence.

In a short, it's 'violation' of cryptographic properties for someone who knows the hidden secret in the cryptosystem design.

Examples of practical kleptography systems:

Escrowed Encryption Standard: Skipjack + Clipper



DES encryption:

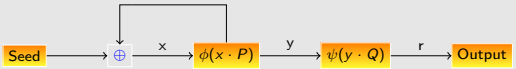
DES

Key: reduced from 64 to 56 bits, Message: 64 Bits

S-boxes vulnerable to The linear cryptoanalysis

The LC complexity: $\approx 2^{43}$ operations
Bruteforce: 2^{56} operations

DualEC DRGB (EC based random number generator):



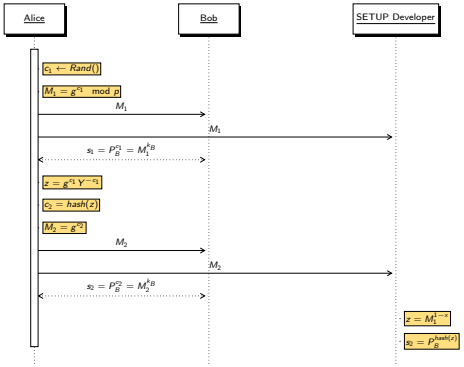
$$Q = Embed(k) \equiv k \cdot P,\ Leak(Output, k) \equiv k^{-1} \cdot \phi^{-1}(Output) = y$$

02 Kleptography mechanisms overview: SETUP

SETUP (Secretly Embedded Trapdoor with Universal Protection) – it's modifying of an existing cryptosystem (by Developer) to implement subliminal secret leakage channel (proposed by A. Young and M. Yung).
Assume that C is a black-box cryptosystem with a publicly known specification. A SETUP mechanosm is an algorithmic modification made to C to get C' such that:

1. The input of C' agrees with the public specifications of the input of C.
2. C' computes efficientlly using the attacker's public encryption function E, contained within C'.
3. The attacker's private decryption function D is not contained within C' and is known only by the attacker.
4. The output of C' agrees with the public specifications of the output of C. At the same time, it contains published bits (of the user's secret key) which are easily derivable by the attacker (the output can be generated during key-generation or during system operation like message sending.
5. Furthermore, the output of C and C' are polynomially indistinguishable to everyone except the attacker.
6. After the discovery of the specifics of the setup algorithm and after discovering its presence in the implementation, users cannot determine past keys.

Diffie-Hellman SETUP modification:
Alice: tries to establish secure connection with *Bob*
Bob: owns a key pair $(k_B, P_B), P_B = g^{k_B}$
Developer: owns a key pair $(x, Y), Y = g^x$, he modifies protocol implementation from *Alice* side using his public key



Thus, after 2 rounds of Diffie-Hellman agreement Developer recover 2-nd session key.

03 Taxonomy of kleptography mechanisms

Criteria	Type	Example
By design opennes	opensource or freeware implementations	software modules, hardware descriptions, references
	proprietary implementations	proprietary solutions with obfuscated code and logic flows
	hardware secured implementations	cryptocontrollers, smart cards
By destructiveness	non-destructive (without system bricking)	reverse engineering of program components, ROM (mask) memmory , studying of specs
	destructive (with system bricking)	hardware analysis of chips, EEPROM memmory etc.
By design level	modifying of existent cryptosystems	injection of leackage channel into impementation, e.g BEAST attack on TLS1.0 protocol
	development of specific cryptosystem with trapdoors	DES, DualEC DRGB
By distribution way	injection of trapdoors into user side impementation	e.g. via malwares, web weaknesses exploitation etc.
	open distribution of protocols with a backdoor	e.g., opensource modules or specs.
	distribution of proprietary hardware modules	
	the lobby in standardizing of cryptosystems with kleptography backdoors, enforcing their usage via legal mechanisms, corporative policies	

04 Kleptography potential

In our research, we consider a specific class of kleptography mechanisms: crypto primitives which are designed with kleptography trapdoor.

The investigation of this class is very important because it targets cryptographic standards candidates that take parts in open competitions.

If we define kleptography risks and methods to reduce them, it will enhance kleptography security in new cryptographic standards.

Let the process of crypto primitive development be a function $Prim : Par \to Out$, where Par – candidate's inputs space (plain text, keys, initial vectors, etc.), Out – outputs space. Let define a set of acceptable functions $\mathbb{F} = Par^{Out}$, $Prim \in \mathbb{F}$ and a set of forbidden function $\hat{\mathbb{F}} \in \mathbb{F} : Prim \notin \hat{\mathbb{F}}$ (algorithms that don't comply with cryptographic properties and additional design requirements).

Kleptography potential

Assume that requirements on candidate $Prim$ are defined as the function set $R_{Prim} = \mathbb{F}_{Prim} \backslash \hat{\mathbb{F}}_{Prim}$. The kleptography potential is the amount of information in its design, that is handled by Developer:

$$\phi(Prim) = H(R_{Prim}) - H(R_{Prim}|D) \leq log_2(|R_{Prim}|),$$

де $H(R_{Prim})$ – unconditional entropy of algorithm's structure before publication, $H(\cdot|D)$ – uncertainty of algorithm's structure before publication but after initializing by Developer.

The idea of such metric: we estimate the entropy in the design of the probable crypto algorithm and subtract the part of uncertainty which isn't under Developer's control, so the final value is a Developer's information amount that remains in the algorithm. Actually, in most cases, conditional entropy is zero because usually Developer totally controls crypto primitive design. However, there are use cases where final design is negotiated by many parts, we need conditional entropy namely for this situation.

The problem: it's difficult to estimate in practice kleptography potential directly by definition because restrictions on possible functions are nontrivial. Also, sometimes it's necessary to estimate kleptography risks for existent crypto primitives that complicate to define function restrictions. Further, we introduce the more practical term "kleptography redundancy".

To define kleptography redundancy we firstly need to define the term "alternative algorithm".

Alternative algorithm

Crypto primitive $Prim^*$ is called an alternative algorithm for *Prim* if it keeps cryptographic properties of algorithm *Prim*. Also, members of Community must (informally) recognize the structure of algorithms $Prim^*$ and *Prim* are the same.

The definition above describes a class of alternative algorithms in a blurred way and there may be different approaches to build this class. In fact, the main requirement is that Community members negotiated and accepted the approach. For example, we can consider alternative algorithms for primitive *Prim*:

1. primitive *Prim* with any changes in round constants (keeping additional requirements and cryptographic properties);
2. primitive *Prim* after replacement of s-boxes (keeping additional requirements and cryptographic properties);
3. primitive *Prim* after replacement of initial vectors (keeping additional requirements and cryptographic properties);
4. primitive *Prim* after replacement of other constants in design (keeping additional requirements and cryptographic properties);
5. primitive *Prim* after replacement of bitwise operations and nonlinear function (keeping cryptographic properties).

The set of rules may be extended according to the kind of algorithm and its design. Additional requirements may be: the impossibility of zero constants, impossibility nonrandom constants, use h maximal nonlinear functions in s-boxes, use only primes or primitive polynomials in some constants etc.

Kleptography redundancy

For crypto primitive *Prim* with alternative algorithm class which is set with relation $R^{\simeq}_{Prim}$ kleptography redundancy is the amount of information that may be inserted by Developer into the design of crypto primitive *during choosing of the instance* from the alternative algorithm class:

$$\rho_{\simeq}(Prim) = H(R^{\simeq}_{Prim}) - H(R^{\simeq}_{Prim}|D) \leq log_2(|R^{\simeq}_{Prim}|).$$

Theorem

Kleptography redundancy is a low estimate of kleptography potential

$$\phi(Prim) \geq \rho_{\simeq}(Prim)$$

04 Kleptography potential

Steps to evaluate kleptography redundancy:

1. define rules that determine the class of alternative primitives;
2. determine class of alternative primitives and estimate cardinality;
3. calculate kleptography redundancy. If the primitive is totally handled by Developer $H(R^{\simeq}_{Prim}|D) = 0$ та $\rho_{\simeq}(Prim) = log_2(|R^{\simeq}_{Prim}|)$.

Example for AES:

1. let assume, such algorithms are alternative:
   - AES with modified SubBytes. Originaly, it's inversion in finite field $GF(2)[X]/x^8 + x^4 + x^3 + x + 1$. We can replace module with any irreducible polynomial (there are 30 irreducible polynomials of degree 8).
   - AES with modified MixColumns. Originaly, it's multiplication by fixed polinomial $c(x) = 03x^4 + 01x^3 + 01x + 02$ from $GF(2^8)/x^4 + 1$. We can replace this polinomial with any other from the field (there are $2^{32} - 1$ alternatives).
   - We consider other parameters (number of rounds and shift rows values) to carry a negligible amount of entropy.
2. We determine a set of all modifications above as AES alternative. The cardinality is $30 * (2^{32} - 1)$.
3. kleptography redundancy is $\rho_{\simeq}(Prim) = log_2(|R^{\simeq}_{Prim}|) = log_2(30 * (2^32 - 1)) \approx 36.9$.

Kleptography redundancy estimation examples

Primitive	Base scheme	source of redundancy	Klepto redundancy
AES	SP-network	procedures SubBytes та MixColumns	36.9
SHA-256	unbalanced Feistel scheme	non linear functions in S-box	78
GOST 28147-89	Feistel scheme	S-box	512
GOST P 34.12-2015 "Kuznechik"	SP-network	S-box and linear transformation	2176
DSTU 7624:2014 "Kalyna"	SP-network	4 S-boxes	8192
GOST P 34.11-2012 "Strybog"	SP-network	S-box, bytewise permutation, linear transformation $V^8 \to V^8$, round consts	12582.19

Parametrized framework

The framework is crypto primitive: $Cand_1 : Consts \times Par \rightarrow Out$ with an additional argument from space *Consts*, which is set up only one time during algorithm initialization (standardization) and is the same for all crypto system's users.

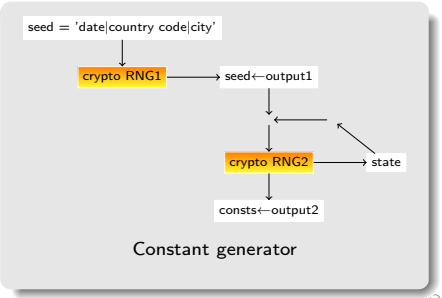
The framework is a trick that allows us to take a lot of entropy out of the candidate design. Further, cryptographic primitive must pass initializing procedure where Community negotiates constants and final initialized framework becomes a cryptographic primitive.

It's also possible to reduce kleptography redundancy in existent primitives. To do that, one needs preliminary to reduce crypto primitive to the framework and to perform framework reinitializing.

There are 2 ways to initialize the framework:

1. use Random Number Generator and unique seed with low entropy (e.g., authors names and first publication date);
2. negotiate constants between Community using group key agreement protocols.

Using this method we are able to reduce kleptography potential by $\Delta\phi = log_2|Consts|$.



Constant generator

06 Conclusion

During the research we have reached such goals:

1. kleptography risks for cryptographic primitives candidates have been formalized;
2. a metric of kleptography risks (“kleptography potential”) has been suggested together with its low estimation – “kleptography redundancy”;
3. evaluation of kleptography redundancy for several famous crypto primitives has been demonstrated;
4. a parametrized framework has been suggested as a trick that allows reducing kleptography redundancy in crypto primitives.

Thanks