On Tolerant Cryptographic Constructions

draft – comments appreciated

Amir Herzberg

Computer Science Department

Bar Ilan University

http://www.cs.biu.ac.il/~herzbea/

Abstract

We investigate how to construct secure cryptographic schemes, from candidate schemes, some of which may be insecure. Namely, tolerant constructions tolerate the insecurity of some of the component schemes used in the construction. Few tolerant cryptographic constructions are known, often widely used; we present the first general definition, prove security of basic, folklore constructions, and present new constructions. Our new constructions are obtained by composing the (known) cascade and parallel constructions; some use a general composition lemma we present. The new, composite constructions provide optimal tolerance for confidentiality and integrity schemes, including encryption, MAC, signatures, signcryption and authenticated encryption. Our constructions are efficient and practical; they may be used to combine several cryptographic schemes or to design new schemes. Our analysis of the concrete security of constructions, and definition of confidentiality and integrity schemes, may be of independent interest.

Keywords: cryptography; signcryption; authenticated encryption; encryption schemes; digital signatures; public key; commitment schemes; TLS.

1. Introduction

Most cryptographic functions do not have an unconditional proof of security. The classical method to establish security is by cryptanalysis i.e. accumulated evidence of failure of experts to find weaknesses in the function. However, cryptanalysis is an expensive, time-consuming and fallible process. In particular, since a seemingly-minor change in a cryptographic function may allow an attack which was previously impossible, cryptanalysis allows only validation of specific functions and development of engineering principles and attack methodologies and tools, but does not provide a solid theory for designing cryptographic functions. Indeed, it is impossible to predict the rate or impact of future cryptanalysis efforts; a mechanism which was attacked unsuccessfully for years may abruptly be broken by a new attack[1]. Hence, it is desirable to design systems to be *tolerant* of cryptanalysis and vulnerabilities (including known trapdoors). A tolerant cryptographic system remains secure following successful cryptanalysis of one or more cryptographic subsystems it contains. Tolerance does not imply unconditional-security; however, it would hopefully provide sufficient advanced-warning time to replace broken cryptographic components.

Many cryptographic systems and constructions use redundant components in the hope of achieving tolerance. The most familiar such construction is cascade. Cascading of cryptosystems is very natural; novices and experts alike believe that the cascade $E \circ E'$ of two cryptosystems E,E' is at least as secure as the more secure of the two, hopefully even more secure than both. Indeed, cascading of cryptosystems has been a common practice in cryptography for hundreds of years.

However, so far, there are few publications on tolerant cryptographic constructions. In [AB81], Asmuth and Blakely present a simple construction of a randomized cryptosystem from two

[1] In practice, we try to use conservative estimates of progress in cryptanalysis, based on past progress and other factors; see e.g. [LV01].

component ciphers, with the hope of achieving tolerance; proof of security was given only in [GM84]; see variant for block ciphers in [HP86]. The highly related problem of cascading of block ciphers received some attention. Even and Goldreich showed that keyed cascade ensures tolerance against message recover attacks on block ciphers [EG85, Theorem 5], and conjectured that the result holds for other specifications of ciphers. Damgard and Knudsen [DK94] proved that it holds for security against key-recovery under chosen-plaintext attacks. Maurer and Massey [MM93] claimed that the proof in [EG85] "holds only under the uninterestingly restrictive assumption that the enemy cannot exploit information about the plaintext statistics", but we disagree. We extend the proof of [EG85] and show that, as expected intuitively and in [EG85], keyed cascading provides tolerance to many confidentiality specifications, not only of block ciphers but also of other schemes such as public key and shared key cryptosystems. Our proof uses a strong notion of security under indistinguishability test – under plaintext only and non-adaptive chosen ciphertext attack (CCA1). One the other hand, we note that cascading does *not* provide tolerance for *adaptive* chosen ciphertext attack (CCA2), or if the length of the output is not a fixed function of the length of the input. This shows the importance of backing the intuition with analysis and proof.

Tolerance is applied in many practical cryptographic systems, not just for confidentiality. In particular, it is widely accepted that the parallel construction $g(x)||g'(x)$, using the same input x to both functions, ensures tolerance for several *integrity* properties, such as (several variants of) collision-resistant hashing as well as Message Authentication Codes (MAC) and digital signatures. We prove that the parallel construction indeed provide tolerance for such integrity specification. The parallel construction is used, for tolerance, in practical designs and standards, e.g. in the W3C XML-DSIG specifications and in the TLS protocol [RFC2246].

The TLS standard uses also *parallel-XOR construction*, to ensure security as long as at least one of two standard hash functions (MD5 and SHA-1) satisfies certain security properties. The *parallel-XOR construction* of f,g is denoted $f \oplus g$ and defined as $f \oplus g(x)=f(x) \oplus g(x)$, where $\oplus$ denotes bit-wise exclusive OR. The *parallel-XOR construction* of f_k , g_k is denoted $f_k \oplus g_{k'}$ (or simply by $f \oplus g$), and defined as $f_k \oplus g_{k'}(x)= f_k(x) \oplus g_{k'}(x)$. The parallel-XOR construction is used in the TLS standard, to provide cryptanalysis-tolerant construction of two pseudo-random functions (one based on MD5 and the other based on SHA-1). This improves on the (older) SSL protocol, which *also* combined MD5 and SHA-1, but failed to ensure tolerance; see details of both TLS and SSL in [R00]. Our techniques can be used to analyze the tolerance of this construction.

However, there are many important cryptographic mechanisms for which there is no known tolerant design (even without analysis). In particular, this holds for several important cryptographic primitives that combine confidentiality and integrity properties; these primitives are often critical for important applications and protocols.

We present a general definition of confidentiality/integrity scheme (CIS), and appropriate confidentiality and integrity specifications, s.t. several important cryptographic primitives become a special case. These include confidentiality-only schemes (public and shared key encryption), integrity-only schemes (MAC and signatures), as well as schemes ensuring both integrity and confidentiality such as authenticated encryption [BN00, J01] and signcryption schemes [Z97, BSZ02, ADR02, GH04]. It also seems possible to provide additional specifications for commitment schemes [DPP94, DPP98, HM96], and committing encryption schemes [GH04]. By using the general CIS definition, we can prove tolerance properties for multiple schemes (e.g. encryption and signature); this general definition may be of independent interest.

We analyze the tolerance of confidentiality/integrity schemes (CIS) under cascade and parallel constructions, and obtain concrete bounds on the security and tolerance of the constructions. Since our definition of CIS is general and includes public key and shared key encryption schemes as well as signatures and MAC schemes, our results are related, and generalize on, previous works which

focused on combining encryption and signatures/MAC, including informal design principles and specific attacks as in [AN95, AN96, B98] and analysis as in [BN00, K01, ADR02, GH04]. As expected, cascade ensures tolerance for confidentiality, while parallel construction ensures tolerance for integrity, but neither ensures tolerance for both confidentiality and integrity.

We then show new constructions, which are compositions of cascade and parallel constructions, and ensure tolerance for *both* integrity and confidentiality specifications. This provides tolerance for authenticated encryption and signcryption schemes. The new constructions are practical, efficient and simple; yet they have not been proposed before and certainly not analyzed (as we do).

Two of our constructions are obtained from a general composition lemma, allowing compositions of any two tolerant constructions; this may be useful for composing other basic constructions to provide tolerance for advanced specifications. The composition lemma allows us to compose a pair of constructions in many structures, defined by a simple combinatorial element which we call a *composition-structure*. We present two very simple composition structures, which we call *D* and *E* after their graphical structure, and use them to compose the cascade and parallel constructions, providing tolerance for both confidentiality and integrity specifications.

We then present another composite construction, which we call the *T* construction. The *T* construction uses specific observations from the analysis of the cascade and parallel constructions, instead of using the general composition lemma. As a result, it provides optimal tolerance for confidentiality and integrity schemes, including signcryption and authenticated encryption schemes, from candidate signcryption and authenticated encryption schemes, or directly from encryption and signature/MAC schemes. The *T* construction is very simple and natural, as illustrated in Figure 1, showing the combination of three encrypt-and-sign functions of signcryption schemes S_1, S_2 , S_3; the resulting scheme is secure as long as at least one of the three component schemes is secure (this generalizes to any number of components). Notice that the *T* construction could be used to create tolerant signcryption (and authenticated encryption) schemes by connecting in parallel several signature (respectively MAC) schemes, and encrypting the result by cascade of encryption schemes.

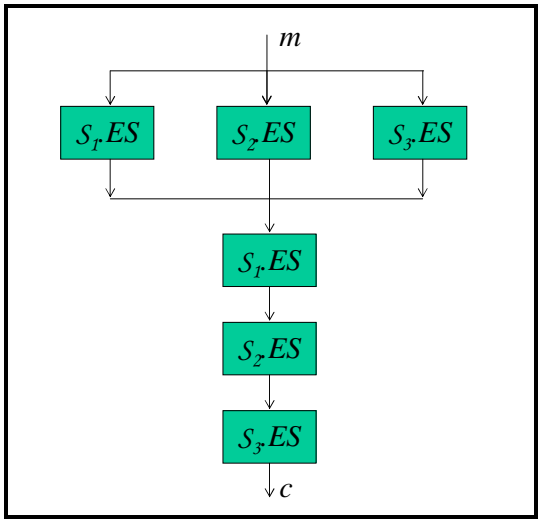

Figure 1: T-construction of three encrypt-and-sign functions

To our knowledge, the *D, E* and *T* constructions are the first provably-secure tolerant constructions of general cryptographic functions, beyond the simple, classical constructions mentioned above, and few additional cryptographic constructions proven secure based on validity of either of two (specific) `hardness` assumptions, e.g. [Sh00, O92]. We note that in theory, it may be possible to ensure tolerance by using provable constructions of cryptographic mechanisms from few `basic`

cryptographic mechanisms, which have simple tolerant designs. For example, many cryptographic mechanisms can be constructed from one-way functions; and (as we show) it is sufficient that one of $\{g , g' \}$ is a one-way function, to ensure their *parallel construction* $f(x,x')=g(x)||g'(x')$ is also a one-way function. Provably-secure constructions based on one-way functions exist for many cryptographic mechanisms, e.g. pseudo-random generators [Go01,HILL99] and signature schemes [NY89]. Therefore, by using a tolerant construction of one way function (from multiple candidate one-way functions) as the basis of some cryptographic scheme, the scheme retains the proven security properties even if one of the candidate one-way functions is not secure. However, such constructions are often inefficient, and involve unacceptable degradation in security parameters (e.g., require absurd key and/or block sizes).

In this work, we use concrete security measures, following [HL92, BKR94,BDJR97], to compute the exact loss in security due to construction. Our analysis is a bit more precise than prior works, in that we are explicit also about the overhead of using the component functions by the adversaries; this may be significant since some tolerant constructions may involve many applications, and possibly substantial increase in the length of the inputs (in previous works, these factors were often negligible). Using this precise analysis, we show that our constructions inherit almost the entire security of the `most secure` component. This precise concrete security analysis may be useful for other applications.

Organization. In Section 2 we present specifications for confidentiality and integrity cryptographic schemes. In section 3 we present the general concept of tolerant constructions and present the cascade and parallel constructions, and their tolerance properties for confidentiality and integrity schemes. In section 4 we present compositions of constructions, including the composition lemma, and present the *E* and *D* compositions and constructions, as well as the *T* construction. We conclude with several open questions**.**

2. Specifications of Confidentiality/Integrity Schemes

In this section we define and present specification for confidentiality/integrity schemes, a general abstraction of cryptographic schemes with confidentiality and/or integrity properties including (symmetric and asymmetric) encryption schemes, signatures schemes, message authentication codes (MAC), authenticated encryption, committed encryption, and signcryption. Our definitions are concrete, namely we explicitly identify all the resources including randomness.

In general, specifications are merely predicates over a set of functions *F*. Let $S(F)$ be the set of all specifications (predicates) over *F*. We say that $f \in F$ satisfies $s \in S(F)$ if $s(f)=True$. For convenience, we often refer to the conjunction of the predicate for a set of functions; $s(f_1,\ldots f_n)=True$ iff for all $i \in \{1,\ldots n\}$ holds $s(f_i)=True$.

Cryptographic specifications are often phrased in terms of *experiments*, representing attempts by an adversarial algorithm *A* to `break` the function. We now give an example: we use the following experiment *OWF*, with $parms=<\rho_A,\tau_A>$, to establish (concrete and poly-time) specifications for one-way functions.

Definition 2-1 Let *A* be an (adversarial) algorithm and let $f \in F(\{0,1\}^*)$, $k, \tau_A, \rho_A \in \mathcal{N}$. Let $OWF^{A,f}(k,\rho_A,\tau_A)$ be the following experiment:

(1) $m \in_R \{0,1\}^k$;

(2) $m'=A(f(m))$;

(3) Return *"win"* if *all* of the following conditions hold, otherwise return *"loss"*:

a. $f(m')=f(m)$, and

b. *A* uses at most ρ_A random bits and at most τ_A computational steps.

2.1. Confidentiality and Integrity Schemes

Specifications can be applied also to finite sets of functions, which we call (*cryptographic*) *schemes*[2]. This is easily achieved by representing schemes by a single function, with an additional input parameter π choosing one of the functions in the scheme. For example, consider an encryption scheme $\mathcal{E}$, defined by the three functions <*KG,E,D*> (for key generation, encryption and decryption, respectively). It is convenient to refer to a specific function in the scheme using dot notation, e.g. $\mathcal{E}.E$ is the encryption function. Therefore, we can view $\mathcal{E}.\pi(x)$ as one function, whose first parameter $\pi \in \{KG, E, D\}$ identifies the `member` function to be used.

Our results apply to several important schemes, including shared and public key cryptosystems (encryption schemes), signature schemes, commitment schemes, committed-encryption schemes, message authentication codes (MAC), authenticated-encryption schemes and signcryption schemes (which combine signature and encryption schemes). The security specifications of these different schemes consist of confidentiality and/or integrity specifications, each of which is quite similar for the different schemes; we show constructions that ensure tolerance for the confidentiality and/or integrity specifications for all of the above schemes.

Rather than restate the results for each type of scheme, we present them for the following general confidentiality/integrity cryptographic scheme, such that each of the schemes is a special case of it. Using a generalized scheme saves repetitions, but does introduce some added complexity; for simplicity, we exclude commitment schemes, which have slightly different syntax. The definitions and tolerant constructions of the general confidentiality/integrity scheme (CIS) are not much more complex than of signcryption and authenticated encryption schemes (whose tolerance is one of our major results).

There is a trivial mapping of every (symmetric or asymmetric) encryption scheme, message authentication code, signature scheme, authenticated encryption and signcryption scheme to a corresponding confidentiality/integrity scheme, defined as follows.

Definition 2-2 A **confidentiality/integrity scheme (CIS)** $\mathcal{S}$ is a collection of functions ***(KG,ES,D,V)*** such that for every m, r, r_{ed}, r_{sv}, $e= \mathcal{S}.KG.e(r_{ed})$, $d=\mathcal{S}.KG.d(r_{ed})$, $s=\mathcal{S}.KG.s(r_{sv})$ and $v=\mathcal{S}.KG.v(r_{sv})$ holds $\mathcal{S}.D.M_d(\mathcal{S}.ES_{e,s,r}(m))=m$ and $\mathcal{S}.V_{v,e}(m,\mathcal{S}.D_d(\mathcal{S}.ES_{e,s,r}(m)))=True$, where:

1. The key generation function $\mathcal{S}.KG$ input is a random string, and its output is a set of four keys: *e, d, s, v* for encryption, decryption, signature and verification, respectively. We refer to particular key using dot notation, e.g. $\mathcal{S}.KG.e$ returns the encryption key.
2. The encrypt and sign function $\mathcal{S}.ES$ inputs are the encryption key *e,* the signature key *s,* a message *m* and a random string *r*, and its output is ciphertext/signature *c.*
3. The decrypt (message recovery) function $\mathcal{S}.D$ inputs are the decryption key *d*, the verification key *v* and ciphertext *c,* and its outputs are a message *m* and (optionally) a `hint` *hint* for verification. To refer only to the message (hint) output we write $\mathcal{S}.D.M$ (respectively, $\mathcal{S}.D.H$). For simplicity we did not include random input (i.e. decryption is deterministic), since it is rarely (if ever) used.

[2] Some definitions of cryptographic schemes are not as a collection of functions, but as a collection of probabilistic algorithms or machines. Often we can view them as functions with additional inputs for randomness and/or state.

4. The (public) verification function $\mathcal{S}.V$ whose inputs are the verification key v, the encryption key e, a message m, ciphertext c and hint h, and whose output is *Ok* or *Not*.

To allow concrete security analysis of constructions, we need concrete bounds on the complexities of the scheme, as follows.

Definition 2-3 [Concrete complexity bounds of CIS] Let $k, k', l, \Delta, \rho, h \in \mathcal{N}$, with $l > \Delta$, and let $\tau:\{KG,ES,V,D\} \rightarrow \mathcal{N}$. Then for every confidentiality/integrity scheme $\mathcal{S}$ we define predicate $bounds[k, k', l, \Delta, \rho, h, \tau](\mathcal{S})$ as *True* if and only if:

1. For inputs of length up to k, $\mathcal{S}.KG$ is computable by a deterministic machine in time $\tau[KG]$, and its output are keys of length at most k' bits each.
2. For every $m,e,r,s \in \{0,1\}^*$ s.t. $|m| \leq l-\Delta$, the value of $\mathcal{S}.ES_{e,s,r}(m)$ is computable by a deterministic machine in time $\tau[ES]$. Also, the machine reads up to ρ bits from the (random input) r, and $|\mathcal{S}.ES_{e,s,r}(m)| \leq l$.
3. For every $m, m', e, s, r \in \{0,1\}^*$, if $|m|=|m'| \leq l-\Delta$ holds $|\mathcal{S}.ES_{e,s,r}(m)|=|\mathcal{S}.ES_{e,s,r'}(m')|$.
4. For every $c,d \in \{0,1\}^*$ s.t. $|c| \leq l$, the value of $\mathcal{S}.D_d(c)$ is computable by a deterministic machine in time $\tau[D]$. Also, $|\mathcal{S}.D.M_d(c)| \leq l-\Delta$ and $|\mathcal{S}.D.H_d(c)| \leq h$.
5. For every $m, c, hint, e, v \in \{0,1\}^*$ s.t. $|m| \leq l-\Delta$, the value of $\mathcal{S}.V_{v,e}(m,c,hint)$ is computable by a deterministic machine in time $\tau[V]$.

Notice that condition 3 above requires that for inputs of the same length, the outputs will also be of the same length. This is required to allow confidentiality definitions based on indistinguishability of ciphertext from any two inputs *of the same length*. Since this condition is kept by all practical CIS we are aware of, it seems best to require it as part of the complexity bounds of CIS.

2.2. Confidentiality Specifications

We first define an indistinguishability experiment **IndExp**, as a generalization of known two-phase indistinguishability experiments for shared and public key confidentiality primitives (e.g. cryptosystems). Specifically, for public encryption key e, use the experiment with input $ISPUB(e)=True$; similarly, for public verification key v, use $ISPUB(v)=True$. In the first phase of the experiment, the adversary chooses two plaintexts, and in the second phase the adversary tries to distinguish the encryption of one of two plaintexts. At each phase $\varphi \in \{1,2\}$, we allow the adversary up to $q[\varphi,f]$ queries to oracles for functions $f \in \{ES,D,V\}$, corresponding to the functions[3] of the scheme $\mathcal{S}$.

Definition 2-4 [Indistinguishability Experiment for CIS] Let $\mathcal{S}$ be a CIS and let $k, l, t, \rho, \Delta, \rho_A \in \mathcal{N}$, $ISPUB:\{e,v\} \rightarrow \{T/F\}$ and $q:\{select,find\} \times \{ES,D,V\} \rightarrow \mathcal{N}$. Let A^O be an (adversarial) algorithm with access to oracle O for the functions in $\mathcal{S}$. Let $\mathbf{IndExp}_{A,\mathcal{S},ISPUB}(k,q,l,\Delta,t,\rho,\rho_A)$ be the following experiment:

(1) $r_{ed}, r_{sv} \in_R \{0,1\}^k$; $e=\mathcal{S}.KG.e(r_{ed})$; $d=\mathcal{S}.KG.d(r_{ed})$; $s=\mathcal{S}.KG.s(r_{sv})$; $v=\mathcal{S}.KG.v(r_{sv})$;

---

[3] We did not include `feedback-only chosen ciphertext attacks` of the kind used in the attacks of [B98, K01]. It seems not difficult to extend the definition and results to cover this important type of attacks.

(2) Let O be an oracle to the functions: $\{\mathcal{S}.ES_{e,s,r}, \mathcal{S}.D_d, \mathcal{S}.V_{v,e}\}$

(3) If $ISPUB(e)=T$ then $e'=e$ else $e'=\lambda$; If $ISPUB(v)=T$ then $v'=v$ else $v'=\lambda$;

(4) $(p[0],p[1],state) \leftarrow A^O(\text{“select”},e',v',1^k)$; /* select phase */

(5) $b \in_R \{0,1\}$;

(6) $r \in_R \{0,1\}^\rho$; $c=\mathcal{S}.ES_{e,s,r}(p[b])$;

(7) $\beta = A^O(\text{“find”},c,state)$; /* find phase */

(8) Return “*win*” only if *all* of the following conditions hold, otherwise return “*loss*”:

a. $\beta=b$, and

b. $|p[1]|=|p[0]| \leq l-\Delta$, and

c. total running time of A^O is less than t, and

d. A^O makes at most $q[\varphi,f]$ calls to oracle $\mathcal{S}.f$ at phase $\varphi \in \{select, find\}$, and

e. A^O uses at most ρ_A random bits, and

f. A^O does <u>not</u> make oracle query $\mathcal{S}.D_d(c)$ during select phase, and

g. in its oracle queries, A^O uses m,c s.t. $|m| \leq l-\Delta$ and $|c| \leq l$.

We can now derive the confidentiality specifications for CIS. First, concrete specifications.

Definition 2-5 We say that $\mathcal{S}$ satisfies **specification** $\boldsymbol{IND_{ISPUB}(a,k,q,l,\Delta,t,\rho,\rho_A)}$ if for every adversary A holds $\Pr[\mathbf{IndExp}_{A,S,ISPUB}(k,q,l,\Delta,t,\rho,\rho_A)=\text{“win”}] < \frac{1}{2}+a$.

We now also present asymptotic, polynomial-time complexities. Allowing polynomial number of each type of queries, including queries to D (chosen ciphertext) during the `find' phase, corresponds to adaptive chosen ciphertext (CCA2) attacks; weaker notions (e.g. CCA1, CPA) must restrict the queries appropriately. We define only CCA2 and CCA1; other variants (e.g. CPA) are similar.

Definition 2-6 We say that $\mathcal{S}$ satisfies specification $CCA2\text{-}IND_{ISPUB}$ if $\mathcal{S} \in PPT$ and for any strictly positive polynomials $l, \Delta, t, \rho, \rho_A, a$ and positive polynomials $q[\varphi,f]$ for $\varphi \in \{select, find\}$ and $f \in \{ES,D,V\}$, exists some integer k_0 such that for every $k \geq k_0$, holds: $\Pr[\text{IndExp}_{A,S,ISPUB}(k,q(k),l(k),\Delta(k),t(k),\rho(k),\rho_A(k))=\text{“win”}] < \frac{1}{2}+a(t)$. We say that $\mathcal{S}$ satisfies specification $CCA1\text{-}IND_{ISPUB}$ if $\mathcal{S}$ satisfies $CCA2\text{-}IND_{ISPUB}$ restricted to $q[\text{“find”},D]=0$. We say that $\mathcal{S}$ satisfies specification $CPA\text{-}IND_{ISPUB}$ if $\mathcal{S}$ satisfies $CCA1\text{-}IND_{ISPUB}$ restricted to $q[\text{“select”},D]=0$.

2.3. Integrity Specifications

In this work we consider only the strongest and most common security notion for signature schemes, Existential Unforgeability. This is the “hardest” notion of security for signatures. Existential unforgeability means that any PPT adversary $\mathcal{A}$ should have a negligible probability of generating a valid signature of a “new" message. Notice, $\mathcal{A}$ is not required to “know” the message whose signature was obtained, so it may be random or nonsensical.

We first define a forgery experiment **ForExp**, as a generalization of Existential Unforgeability experiment introduced by Goldwasser, Micali, and Rivest [GMR95]. Like for confidentiality, we again use parameter *ISPUB* to define whether we use for encryption and/or authentication a shared

secret key or a public/private key pair. We allow the adversary up to $q[f]$ queries to oracles for functions $f \in \{ES,D,V\}$, corresponding to the functions of $\mathcal{S}$.

In the integrity specifications we include a parameter *SPOOF,* to specify whether the attacker can perform key-spoofing attacks as in [AN95], namely present a forgery by using a different encryption key e than the one used in the encrypt and sign (ES) function.

Definition 2-7 [Forgery Experiment for CIS] Let $\mathcal{S}$ be a CIS and let $k, t, \rho_A \in \mathcal{N}$, $ISPUB:\{e,v\} \rightarrow \{T/F\}$, $SPOOF \in \{T,F\}$ and $q:\{ES,D,V\} \rightarrow \mathcal{N}$. Let A^O be an (adversarial) algorithm with access to oracle O for the functions in $\mathcal{S}$. Let $\mathbf{ForExp}_{A,\mathcal{S},ISPUB,SPOOF}(k,q,l,\Delta,t,\rho_A)$ be the following experiment:

(1) r_{ed} , $r_{sv} \in_R \{0,1\}^k$; $e = \mathcal{S}.KG.e(r_{ed})$; $d = \mathcal{S}.KG.d(r_{ed})$; $s = \mathcal{S}.KG.s(r_{sv})$; $v = \mathcal{S}.KG.v(r_{sv})$;

(2) Let O be an oracle to the functions: $\{\mathcal{S}.ES_{e,s}$, $\mathcal{S}.D_d$, $\mathcal{S}.V_{v,e}\}$

(3) If $ISPUB(e)=T$ then $e_A=e$ else $e_A=\lambda$; If $ISPUB(v)=T$ then $v_A=v$ else $v_A=\lambda$;

(4) $(type,\sigma) \leftarrow A^O(e_A,v_A,1^k)$;

(5) If $type='m'$ then (m,c,h,e_s)=Parse(σ)
else { (c,e_s)=Parse(σ); $(m,h)=\mathcal{S}.D_d(c)$ } ;

(6) If $SPOOF=T$ and $e_s \neq \lambda$ then $e^*=e_s$ else $e^*=e$;

(7) Return "*win*" only if *all* of the following conditions hold, otherwise return "*loss*":

a. $\mathcal{S}.V_{e^*,v}(m,c,h)=Ok$, and

b. A^O did <u>not</u> make oracle query $\mathcal{S}.ES_{\gamma,s}(m)$ for any encryption key γ, and

c. total running time of A^O is less than t, and

d. A^O makes at most $q[f]$ calls to oracle $\mathcal{S}.f$, and

e. A^O uses at most ρ_A random bits.

f. in its oracle queries, A^O uses inputs of size at most $|m| \leq l-\Delta$ and $|c| \leq l$.

We can now derive the integrity specifications for CIS. Note that if $q[ES]>0$, this is a known message attack (KMA).

Definition 2-8 We say that $\mathcal{S}$ satisfies specification $\boldsymbol{INT_{ISPUB,SPOOF}(a,k,q,l,\Delta,t,\rho_A)}$ if for every adversary A holds $\Pr[\text{ForExp}_{A,\mathcal{S},ISPUB,SPOOF}(k,q,l,\Delta,t,\rho_A)="win"] < \frac{1}{2}+a$. We say that $\mathcal{S}$ satisfies specification $\boldsymbol{CMA\text{-}INT_{ISPUB,SPOOF}}$ if $\mathcal{S} \in PPT$ and for any strictly positive polynomials l, Δ , t, ρ_A, a and positive polynomials $q[f]$ for $f \in \{ES,D,V\}$, exists some integer k_0 such that for every $k \geq k_0$, holds: $\Pr[\text{ForExp}_{A,\mathcal{S},ISPUB,SPOOF}(k,q(k),l(k),\Delta(k), t(k), \rho_A(k))="win"] < a(t)$.

Comment: integrity does not ensure commitment. Notice that the integrity specification does not imply binding between the ciphertext and the plaintext, as required by commitment schemes [DPP94, DPP98, HM96] and committed-encryption schemes [GH04]. However, it seems that our techniques may yield similar results for commitment specifications.

3. Tolerant Constructions

In this section we introduce the concept of tolerant constructions, namely a mapping c of one or more `candidate functions` $f_1,\ldots f_p$ into a single `redundant` function $c(f_1,\ldots,f_p)$, such that $c(f_1,\ldots,f_p)$ satisfies some specification s' as long as a sufficient number among $f_1,\ldots f_p$ satisfy specifications s (where possibly $s=s'$). We then present the cascade and parallel constructions, two of the most

important and basic constructions. We show that cascade preserves integrity and provides tolerance for confidentiality (indistinguishability), while parallel construction preserves confidentiality and provides tolerances for integrity.

3.1. Constructions and Tolerant Constructions

We first define the general concept of a *construction*. Given F and an integer p, let c be a mapping of ordered set $<f_1, \ldots, f_p>$ of functions in F to a function in F, i.e. $c:F^p \rightarrow F$. We say that ***c* is a construction of plurality *p* over *F***.

We now define *tolerant constructions*. A tolerant construction c accepts as input several candidate functions $<f_1, \ldots, f_p>$, e.g. for specification $s \in S(F)$, and output a single function $c(f_1, \ldots, f_p)$ which satisfies specifications $s' \in S(F)$ as long as 'enough' of the candidates satisfy s (optionally $s=s'$). In addition, we often require that *all* of the candidate functions f_i satisfy some minimal specifications $b \in S(F)$, such as bounds on their complexities.

Definition 3-1 Consider some set of functions F and predicates $s,s',b \in S(F)$. Construction c of plurality p over F is ***t* -tolerant for *s*→*s'* with prerequisite *b***, where t is an integer between 0 and $p-1$, if for every set $\{f_1,\ldots f_p\} \in F^p$ s.t. $b(f_1,\ldots f_p)=T$ holds:

$$\left[(\exists i_1,\ldots,i_{p-t})(\forall j:1\le j\le p-t)\left(s(f_{i_j})=True\right)\right]\Rightarrow s'\left(c(f_1,\ldots,f_p)\right)=True$$

When (as often) $b(f)=True$ for all $f \in F$, we omit it, and say e.g. that c is t -tolerant for $s \rightarrow s'$. If construction c of plurality p over F is ***0*-tolerant for *s*→*s'*** then we say that ***c* preserves *s*→*s'***. If c is t- tolerant for $s \rightarrow s$, then we say simply that c is t-tolerant for s; if $t=0$ then we say that c preserves s.

3.2. Cascade Constructions

The most basic tolerant construction of cryptographic functions is the cascade construction $c_\circ$. By cascading we mean applying in sequence. We begin by discussing `simple cascading', which is cascading of functions with a single input and output, such as hash functions, namely $c_\circ(f,g)=f\circ g(x)=f(g(x))$; some readers may skip this subsection as it is mostly as an exercise, since we can only show that cascading provides tolerance for very specialized specifications of keyless functions. In the following subsections we discuss cascading of keyed schemes.

3.2.1. Cascading of Keyless Functions (may be skipped)

The cascade of two (keyless) functions f and g, denoted $f\circ g$ or $c_\circ(f,g)$, is defined as $c_\circ(f,g)=f\circ g(x)=f(g(x))$. Keyless cascading is a construction of plurality 2 for functions whose domain D contains their range R, i.e. $R \subseteq D$. This holds for some cryptographic primitives such as One Way Functions (OWF), and some specifications of keyless hash functions. In the next subsection we discuss cascading of cryptographic primitives, e.g. cryptosystems, whose domain consists of multiple inputs such as data, key and random bits.

We now show that cascade does not preserve either OWF or WCRHF. We also show that One-Way Permutations (OWF property restricted to permutations) is tolerant under cascade; this may not be very useful but at least shows that cascading could be useful for some keyless functions.

More precisely, we show that cascade ensures tolerance for the *polytime-OWF* specification, over the set $P(\{0,1\}^*) \subseteq F(\{0,1\}^*)$, which consists of polynomially-time computable permutations for any given input (and output) length, i.e. $f \in P(\{0,1\}^*)$ if and only if $f \in PPT$ and $(\forall k)(\forall x \in \{0,1\}^k)(|f(x)|=k) \wedge (\forall y \in \{0,1\}^k : y \neq x)(f(y) \neq f(x))$. Similarly, cascade ensures tolerance for

concrete-OWF^f*(a,k,*ρ_A*,* τ_A*)* $\wedge$ *[Time(f,k)*$\leq \tau_F$*]* $\rightarrow$ *concrete-OWF*^{f}*(a,k,*ρ_A*,* $\tau_A+\tau_F$*)* $\wedge$ *[Time(f,k)*$\leq 2\tau_F$*]* over the set P_k, which consists of permutations over $\{0,1\}^k$.

Lemma 3-1 Cascade of keyless functions is…

1. 1-tolerant over $P(\{0,1\}^*)$ for specifications *polytime*-O*WF*.
2. 1-tolerant over P_k for specifications: *concrete-OWF*^{f}*(a,k,*ρ_A*,* τ_A*)*$\wedge$*[Time(f,k)*$\leq \tau_F$*]* $\rightarrow$ *concrete-OWF*^{f}*(a,k,*ρ_A*,* $\tau_A+\tau_F$*)*$\wedge$*[Time(f,k)*$\leq 2\tau_F$*]*
3. Not (even) *0*-tolerant over $F(\{0,1\}^*)$ for specifications *polytime-OWF* and *polytime-WCRHF*.

Proof: To prove claim 3, let *h* be a OWF and/or WCRHF. Let $g(x)=h(x)||0^{|x|}$ and $f(x)= f(x) = \begin{cases} 0 & if\ x = y0^{|x|/2} \\ h(x) & else \end{cases}$. Trivially, both *f* and *g* are OWF and/or WCRHF, respectively, yet $f\circ g$ is neither OWF not WCRHF; in fact, $f\circ g(x)=0$ for every *x*.

It remains to prove claim 2 (from which claim 1 immediately follows). Trivially, if *Time(f,k)*$\leq \tau_F$ and *Time(g,k)*$\leq \tau_F$ then *Time(f*$\circ$*g,k)*$\leq 2\tau_F$. Let *s(f)= concrete-OWF*^{f}*(a,k,*ρ_A*,* τ_A*)* $\wedge$ *[Time(f,k)*$\leq \tau_F$*]*, *s'(f)= concrete-OWF*^{f}*(a,k,*ρ_A*,* $\tau_A+\tau_F$*).* It remains to prove that *s(f)* $\rightarrow$ *s'(f*$\circ$*g)* and that *s(g)* $\rightarrow$ *s'(f*$\circ$*g).*

Assume *f,* $g\in P_k$ and *Time(f,k)*$\leq \tau_F$, *Time(g,k)*$\leq \tau_F$. Trivially, $f\circ g\in P_k$. Assume that *s'(f*$\circ$*g)=False*; we prove that both *s(f)=False* and *s(g)=False.*

Since *s'(f*$\circ$*g)=False*, there is some (possibly probabilistic) algorithm *A* s.t. $(\forall x \in \{0,1\}^k)time(A(f \circ g(x))) \leq \tau_A + \tau_F$ and $\Pr_{m\in_R\{0,1\}^k}\left[f \circ g(A(f \circ g(m))) = f \circ g(m)\right] \geq a$.

Define algorithms A_f, A_g as follows:

$A_f(y)=g(A(y))$, $A_g(y)=A(f(y))$

We first show that the running time of A_f and A_g over inputs of length *k* is bounded by $t+\tau$. Suppose A_f is given input *f(m)* where $|f(m)|=|m|=k$ (remember that *f* is a permutation for inputs of any length *k*). Therefore, A_f gives input of the same length *k* to *A*. WLOG, we can assume that the output of *A* is also of length *k* (since otherwise clearly *A* loses). Therefore, the running time of A_f on input of length *k* is at most $t+\tau$; a similar argument holds for A_g.

It remains to show that $\Pr_{m\in_R\{0,1\}^k}\left[f(A_f(f(m))) = f(m)\right] \geq a$ and $\Pr_{m\in_R\{0,1\}^k}\left[g(A_g(g(m))) = g(m)\right] \geq a$.

Let *X* denote the *k*-bit strings *x* for whom *A* succeeds in inverting $f\circ g$, i.e. for every $x\in X$ holds $f\circ g(A(f\circ g(x)))= f\circ g(x)$. Since $\Pr_{x\in_R\{0,1\}^k}\left[f \circ g(A(f \circ g(x))) = f \circ g(x)\right] \geq a$, we know that $|X|\geq a\cdot 2^k$. Similarly let $X_f \equiv \left\{x \in \{0,1\}^k \middle| f\left(A_f\left(f(x)\right)\right)= f\left(x\right)\right\}$, $X_g \equiv \left\{x \in \{0,1\}^k \middle| g\left(A_g\left(g(x)\right)\right)= g\left(x\right)\right\}$. We show that $\left|X_f\right| \geq \left|X\right|$, $\left|X_g\right| \geq \left|X\right|$ and since $|X|\geq a\cdot 2^k$, the claim follows.

Let $x\in X$. Hence $f\circ g(A(f\circ g(x)))= f\circ g(x)$, namely $f(g(A_g(g(x)))=f(g(x))$. Since *f* is a permutation, it follows that $g(A_g(g(x)))=g(x)$, namely $x\in X_g$.

Similarly, let $x_f=g(x)$. Since $x\in X$, then $f\circ g(A(f\circ g(x)))= f\circ g(x)$, namely $f(A_f(f(x_f)))=f(x_f)$, i.e. $x_f\in X_f$. Since *g* is a permutation, it follows that $|X_f|\geq|X|$. ∎

We believe that the positive parts of the Lemma (claims 1 and 2) could be generalized for an appropriate family of regular functions. It would be interesting to find other cryptographic

specifications of keyless functions for which cascading provides tolerance, or any tolerant construction for some useful specifications for keyless hash functions.

3.2.2. Cascading of Confidentiality/Integrity Schemes (CIS)

We next consider cascading of keyed cryptographic schemes, specifically of confidentiality and integrity schemes (CIS).

The *keyed cascade* of two CISs $\mathcal{S}, \mathcal{S}'$, denoted $c_\circ(\mathcal{S},\mathcal{S}')$ or $\mathcal{S}\circ\mathcal{S}'$, is defined as follows (see simplified illustration in Figure 2). The definitions and proofs extend trivially to cascade of arbitrary number of CISs.

1. $\mathcal{S}\circ\mathcal{S}'.KG(r,r')=\mathcal{S}.KG(r)||\mathcal{S}'.KG(r')$.
2. $\mathcal{S}\circ\mathcal{S}'.ES_{e,e',s,s',r,r'}(m)=\mathcal{S}.ES_{e,s,r}(\mathcal{S}'.ES_{e',s',r'}(m))$
3. $\mathcal{S}\circ\mathcal{S}'.D.M_{d,d'}(c)=\mathcal{S}'.D.M_{d'}(\mathcal{S}.D.M.c_d(c))$
4. $\mathcal{S}\circ\mathcal{S}'.D.H_{d,d'}(c)=\mathcal{S}'.D.H_{d'}(\mathcal{S}.D.M_d(c)) \,||\, \mathcal{S}.D_d(c)$
5. $\mathcal{S}\circ\mathcal{S}'.V_{e,e',v,v'}(m,c,<h',h,c'>)= \mathcal{S}.V_{e,v}(c',c,h) \wedge \mathcal{S}'.V_{e',v'}(m,c',h')$

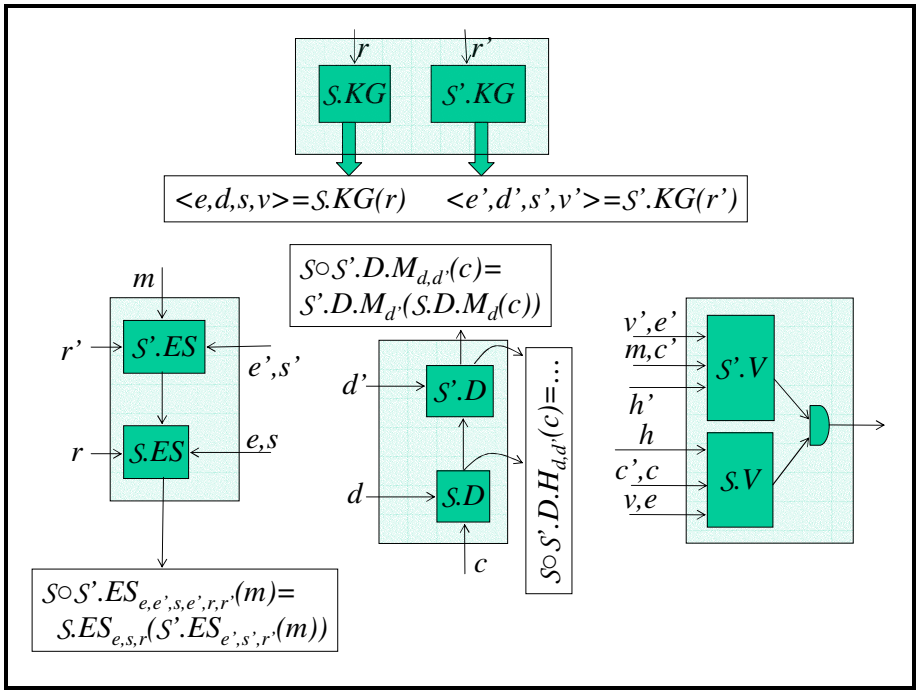

Figure 2: Keyed Cascade of CISs

Clearly, keyed cascade is a construction of plurality 2 of CISs (namely, if $\mathcal{S}, \mathcal{S}'$ are both CIS, then $\mathcal{S}\circ\mathcal{S}'$ is also a CIS). We can also easily cap the complexities of the construction, and the results extend to cascade of more than 2 schemes; we summarize these observations as follows:

Lemma 3-2 Let $\mathcal{S}, \mathcal{S}'$ be a pair of CISs such that $bounds[k, k', l, \Delta, \rho, h, \tau](\mathcal{S},\mathcal{S}')=True$ with $l>2\Delta$. Then $c_o(\mathcal{S},\mathcal{S}')=\mathcal{S}\circ\mathcal{S}'$ is also a CIS and $bounds[2k, 2k', l, 2\Delta, 2\rho, 2h+l, 2\tau](\mathcal{S}\circ\mathcal{S}')=True$. Let $\mathcal{S}_1, \ldots \mathcal{S}_n$ be n CISs such that $bounds[k, k', l, \Delta, \rho, h, \tau](\mathcal{S}_1, \ldots \mathcal{S}_n)=True$ with $l>n\Delta$. Then $c_o(\mathcal{S}_1, \ldots \mathcal{S}_n)=\mathcal{S}_1\circ \ldots\circ\mathcal{S}_n$ is also a CIS and $bounds[nk, nk', l, n\Delta, n\rho, nh+(n-1)l, n\tau](\mathcal{S}_1\circ \ldots\circ\mathcal{S}_n)=True$. ∎

3.2.3. Cascading provides Tolerance for Confidentiality Specifications

Trivially, cascading does *not* ensure tolerance under CCA2, adaptive chosen ciphertext attack; see other issues with CCA2, and an alternative definition (gCCA), in [ADR02]. We now show that cascade is a 1-tolerant construction for the indistinguishability confidentiality specification, when *not* allowing adaptive chosen ciphertext queries (i.e. with $q[find,D]=0$).

Lemma 3-3 Let $ISPUB:\{e,v\}\rightarrow\{T,F\}$. Then:

1. The cascade construction c_0 is 1-tolerant for $CCA1\text{-}IND_{ISPUB}$ and for $CPA\text{-}IND_{ISPUB}$.
2. Let $k, k', l, \Delta, \rho, h, t^o, \rho^o_A \in N$ s.t. $l>2\Delta$, $\tau:\{KG,ES,V,D\}\rightarrow N$, $q:\{select,find\}\times\{ES,D,V\}\rightarrow N$ s.t. $q[find,D]=0$. Then c_o is 1-tolerant for $s\rightarrow s^o$ with prerequisite $bounds[k, k', l, \Delta, \rho, h, \tau]$, where $s^o=IND_{ISPUB}(a,k,q,l,2\Delta,t^0, \rho,\rho^o_A)$, $s=IND_{ISPUB}(a,k,q,l,\Delta,t, \rho,\rho_A)$ with $t=t^o+\tau[KG]+2\tau[E]+\sum_{j\in\{find,select\}}\sum_{f\in\{ES,D,V\}} q[j,f]\tau[f]$, $\rho_A=\rho^o_A+2k+2\rho$.
3. For any $n>1$, the cascade of n schemes c_o is $(n-1)$-tolerant for $s\rightarrow s^o$ with prerequisite $bounds[k, k', l, \Delta, \rho, h, \tau]$, where $s^o=IND_{ISPUB}(a,k,q,l,n\Delta,t^0, \rho,\rho^o_A)$, $s=IND_{ISPUB}(a,k,q,l,\Delta,t, \rho,\rho_A)$ with $t=t^o+(n-1)\tau[KG]+n\tau[E]+(n-1)\sum_{j\in\{find,select\}}\sum_{f\in\{ES,D,V\}} q[j,f]\tau[f]$, $\rho_A=\rho^o_A+2k+2\rho$.

Proof: We prove only claim 2 (claim 3 follows similarly, and claim 1 follows from claim 2). The proof is by contradiction; namely assume that for some $\mathcal{S}, \mathcal{S}'$ holds $s^o(\mathcal{S}\circ\mathcal{S}')=False$, and we show that $s(\mathcal{S})=s(\mathcal{S}')=False$.

Since $s^o(\mathcal{S}\circ\mathcal{S}')=False$, then there is some adversary A^o such that $p^o\equiv\Pr[\mathbf{IndExp}_{A,\mathcal{S},ISPUB}(k,q,l,t^o,\rho,\rho^o_A)=\text{"win"}]\geq \frac{1}{2}+a$. We next show that given such adversary A^o as a black box, we can construct adversaries A, A' such that $p\equiv\Pr[\mathbf{IndExp}_{A,\mathcal{S},ISPUB}(k,q,l_0,t,\rho,\rho_A)=\text{"win"}]\geq p^o\geq \frac{1}{2}+a$ and $p'\equiv\Pr[\mathbf{IndExp}_{A',\mathcal{S}',ISPUB}(k,q,l_0,t,\rho,\rho_A)=\text{"win"}]\geq p^o\geq \frac{1}{2}+a$, where $l_0=l+\Delta$.

Namely, we prove (below) the following claims A, A':

Claim A (A'): given adversary A^o such that $p^o\geq \frac{1}{2}+a$ as a black box, we can construct adversary A (respectively A') such that $p\geq p^o\geq \frac{1}{2}+a$ (respectively $p'\geq p^o\geq \frac{1}{2}+a$).

This completes the proof, by showing that indeed $s(\mathcal{S})=s(\mathcal{S}')=False$. ∎

Proof of Claim A: We construct adversary A as follows. In the "select" phase, A selects randomly $r'_{ed}, r'_{sv}\in\{0,1\}^k$, and then uses $\mathcal{S}'.KG$ to compute the keys $e'=\mathcal{S}'.KG.e(r'_{ed})$, $d'=\mathcal{S}'.KG.d(r'_{ed})$, $s'=\mathcal{S}'.KG.s(r'_{sv})$, $v'=\mathcal{S}'.KG.v(r'_{sv})$.

Next, A invokes the "select" phase of A^o which returns plaintexts $p^o[0], p^o[1]$ and state s^o. In its operation, A^o may invoke the oracles for functions $\{ES,D,V\}$ of $\mathcal{S}^o$; trivially, A can answer these queries by using the corresponding oracle for $\mathcal{S}$, and computing the corresponding function of $\mathcal{S}'$ (using keys e', d', s' and v'). For example, to answer query of $\mathcal{S}^o.D_{d,d'}(c)$, we first invoke the oracle $\mathcal{S}.D_d$ on input c; denote the result as x. We now compute $\mathcal{S}^o.D_{d,d'}(c)=\mathcal{S}'.D_{d'}(\mathcal{S}.D_d(c))=\mathcal{S}'.D_{d'}(x)$, which is possible since A knows d'.

To complete the "select" phase, A computes $p[j]=\mathcal{S}'.ES_{e',s',r'[j]}(p^o[j])$ for $j\in\{0,1\}$ and $r'[j]\in_R\{0,1\}^*$ (for public key encryption, i.e. if $ISPUB[e]=F$, then concatenate e' to $p[0]$ and $p[1]$). It then returns $p[0], p[1]$ and $s=<s^o,e,e',d',s',v'>$. We later show that using $r'[j]\in_R\{0,1\}^\rho$ suffices.

In the "find" phase, A receives ciphertext c and state $s=<s^o,e,e',d',s',v'>$. It simply invokes the "find" phase of A^o on c and s^o, and returns the bit x returned by A^o.

We now show that $p^o \leq p$. The probabilities are taken over the coin tosses by A, by the `black-box` algorithm A^o, and by the experiment (for key-generation, encryption and b). Denote the coin tosses as follows:

- r^o_A : coins used by the `black-box` adversary A^o (provided by and known to A)
- Coins tossed by the experiment (unknown to A): for key generation (r_{ed}, r_{sv}), to select the challenge plaintext $p[b]$ (coin b), and for encrypting $p[b]$ (bits r).
- Coins tossed by A for its own use, namely: for key generation (r'_{ed}, r'_{sv}) and to compute the encryptions of the plaintexts $S'.ES_{e',s',r'[j]}(p^o[j])$ (bits $r'[0]$, $r'[1]$).

Let $r_A=r^o_A||r'_{ed}||r'_{sv}||r'[0]||r'[1]$ denote all the random bits tossed by algorithm A. Let $w(r_A, r_{ed}, r_{sv}, r, b)=true$ if and only if $\mathbf{IndExp}_{A,S,ISPUB}(k,q,l,t,\rho,\rho_A)=$"win" with the corresponding coin tosses.

Let $r^o_{ed}=r_{ed}||r'_{ed}$, $r^o_{sv}=r_{sv}||r'_{sv}$, $r^o=r||r'$. Let $w^o(r^o_A, r^o_{ed}, r^o_{sv}, r^o, b)=true$ if and only if $\mathbf{IndExp}_{A^\circ,S^\circ,ISPUB}(k,q,l,t^o,\rho,\rho^o_A)=$"win" with the corresponding coin tosses.

The claim follows by showing that $w^o(r^o_A, r^o_{ed}, r^o_{sv}, r^o, b) \rightarrow w(r_A, r_{ed}, r_{sv}, r, b)$. We show this holds, by showing that all conditions of step 8 of experiment $E=$ $\mathbf{IndExp}_{A,S,ISPUB}(k,q,l,t,\rho,\rho_A)[r_A, r_{ed}, r_{sv}, r, b]$ hold if they (conditions of step 8) hold in experiment $E^o=$ $\mathbf{IndExp}_{A^\circ,S^\circ,ISPUB}(k,q,l,t^o,\rho,\rho^o_A)[r^o_A, r^o_{ed}, r^o_{sv}, r^o, b]$.

We use the following notation: let $x@E$ ($x@E^o$) denote the value of variable x during experiment E (respectively E^o). We omit the @ notation when the value is clearly identical in the two experiments. Also, let $c8\varphi@E$ ($c8\varphi@E^o$), where $\varphi \in \{a,b,\ldots,f\}$, be *true* if claim φ of step 8 holds during experiment E (respectively E^o).

Algorithm A returns the same bit β as returned by A^o, namely $\beta@E=\beta@E^o$. Hence if $c8a@E^o$ is *true*, i.e. $\beta@E^o=b$, then also $\beta@E=b$ and $c8a@E=true$.

By design of A above, for $j=\{0,1\}$ holds $p[j]@E=S'.ES_{e',s',r'[j]}(p^o[j]@E)$. If $c8b@E^o=true$, then $|p^o[1]@E|=|p^o[0]@E| \leq l-2\Delta$. Since $bounds[k, k', l, \Delta, \rho, h, \tau](S')=True$, we have $|p[1]@E|=|p[0]@E'| \leq l-\Delta$. Hence, $c8b@E^o \rightarrow c8b@E$.

For $c8c$, we note that the running time of A consists of the running time of A^o in the corresponding experiment, plus the additional work by A. This extra work consists essentially of invoking the key generation algorithm once, doing two encryptions (to compute $p[0]$ and $p[1]$), and answering the oracle queries of A^o. Each oracle query $S^o.f$ requires A to compute $S'.f$; it follows that if the running time of A^o at E^o is bounded by t^o, then the running time of A at E is bounded by $t=t^o+\tau[KG]+2\tau[E]+\sum_{j\in\{find,select\}}\sum_{f\in\{ES,D,V\}} q[j,f]\tau[f]$. Hence, $c8c@E^o \rightarrow c8c@E$.

We note that A involves oracle $S.f$ only to answer oracle call $S^o.f$ of A^o. Hence, $c8d@E^o \rightarrow c8d@E$, $c8g@E^o \rightarrow c8g@E$ and $c8f$ holds since $q[find,D]=0$.

It remains to show that $c8e@E^o \rightarrow c8e@E$. Adversary A users random bits $r_A=r^o_A||r'_{ed}||r'_{sv}||r'[0]||r'[1]$ (including the random bits r^o_A for running A^o internally). Both r'_{ed} and r'_{sv} are k bit long. Bits $r'[0]$, $r'[1]$ are used by A (only) to compute $p[0]$,$p[1]$. Assume that $c8b$ holds at E^o, i.e. $|p[1]@E^o|=|p[0]@E^o| \leq l-2\Delta$. Since $bounds[k, k', l, \Delta, \rho, h, \tau](S')=True$, we use at most ρ bits from $r'[j]$, for $j=\{0,1\}$, in computing $p[j]@E=S'.ES_{e',s',r'[j]}(p^o[j]@E)$. It follows that the total number of random bits used by A is at most $|r^o_A|+2k+2\rho$. If $c8e@E^o$ holds, i.e. $|r^o_A| \leq \rho^o_A$, it follows that A uses at most $|r_A| \leq \rho^o_A+2k+2\rho=\rho_A$ random bits. Hence $c8e@E^o \wedge c8b@E^o \rightarrow c8e@E$. ∎

It remains to prove:

Claim A': given adversary A^o such that $p^o \geq ½+a$ as a black box, we can construct adversary A' such that $p' \geq p^o \geq ½+a$.

Proof of Claim A': We construct adversary A' as follows. In the "select" phase, A' selects randomly r_{ed}, $r_{sv} \in \{0,1\}^k$, and then uses $\mathcal{S}.KG$ to compute the keys $e=\mathcal{S}'.KG.e(r_{ed})$, $d=\mathcal{S}.KG.d(r_{ed})$, $s=\mathcal{S}.KG.s(r_{sv})$, $v=\mathcal{S}.KG.v(r_{sv})$.

Next, A' invokes the "select" phase of A^o which returns plaintexts $p[0]$, $p[1]$ and state s^o. In its operation, A^o may invoke the oracles for functions $\{ES,D,V\}$ of $\mathcal{S}^o$; trivially, A' can answer these queries by using the corresponding oracle for $\mathcal{S}'$, and computing the corresponding function of $\mathcal{S}$. Finally, A' returns $p[0]$, $p[1]$ and $s=<s^o,e',e,d,s,v>$. We later show that using $r[j] \in_R \{0,1\}^\rho$ suffices.

In the "find" phase, A' receives ciphertext c and state $s=<s^o,e',e,d,s,v>$. It computes $c'=\mathcal{S}.ES_{e,s,r}(c)$ and invokes the "find" phase of A^o on c' and s^o, and returns the bit x returned by A^o.

The rest of the proof follows exactly like in claim A. ■

3.2.4. Cascading Preserves Integrity Specifications

We now show that cascading is 0-tolerant (i.e. preserves) for the integrity specifications we presented for confidentiality/integrity schemes (CIS). In fact, it is sufficient that the *first* scheme applied to the message satisfies integrity, as we show in the next lemma. Integrity of the second scheme is *not* sufficient to ensure integrity of the cascade (however, it suffices e.g. if the first scheme satisfies the (weaker) integrity requirements of committed encryption; see in [GH04]).

Lemma 3-4 For any $ISPUB:\{e,v\} \rightarrow \{T,F\}$, $SPOOF \in \{T,F\}$ holds:

1. The cascade construction c_o is 0-tolerant for $CMA\text{-}INT_{ISPUB,SPOOF}$.
2. For any $\{\mathcal{S}_1,\ldots\mathcal{S}_n\}$ holds $CMA\text{-}INT_{ISPUB,SPOOF}(\mathcal{S}_n) \rightarrow CMA\text{-}INT_{ISPUB,SPOOF}(\mathcal{S}_1 \circ \ldots \circ \mathcal{S}_n)$
3. For any $\{\mathcal{S}_1,\ldots\mathcal{S}_n\}$, $a \in [0,1], k, q, n, \Delta, t^o, \rho^o_A \in \mathcal{N}$, $l > n\Delta$, and $q:\{ES,D,V\} \rightarrow \mathcal{N}$, let
   $t=t^o+\tau[KG]+n\tau[E]+(n-1)\sum_{f\in\{ES,D,V\}} q[f]\tau[f]$, $\rho_A=\rho^o_A+nk$,
   $s^o=INT_{ISPUB,SPOOF}(a,k,q,l,n\Delta,t^0,\rho^o_A)$, and $s=IND_{ISPUB,SPOOF}(a,k,q,l,\Delta,t,\rho_A)$.
   Then $s(\mathcal{S}_n) \rightarrow s^o(\mathcal{S}_1 \circ \ldots \circ \mathcal{S}_n)$.

Proof: Claims 1,2 follows immediately from claim 3. We prove claim 3 by contradiction, and for simplicity, only for $n=2$. Namely assume that $s^o(\mathcal{S} \circ \mathcal{S}')=False$ and we show that $s(\mathcal{S}')=False$. Namely, assume that there is some adversary A^o such that $p^o \equiv \Pr[\text{ForExp}_{A,S,ISPUB,SPOOF}(k,q,l,2\Delta,t^o,\rho^o_A)=\text{"win"}] \geq a$. Using A^o as a black box, we construct adversary A' such that $p' \equiv \Pr[\text{IndExp}_{A',S',ISPUB}(k,q,l,\Delta,t,\rho_A)=\text{"win"}] \geq p^o \geq a$.

We construct adversary A' as follows. In the "select" phase, A' selects randomly r_{ed}, $r_{sv} \in \{0,1\}^k$, and then uses $\mathcal{S}.KG$ to compute the keys e,d,s and v. Next, A' invokes A^o which returns $(type, \sigma^o)$. We answer oracle queries of A^o like in Lemma 3-3.

We now follow step (5) of the Forgery experiment (Definition 2-7) to compute m,c^o,h^o,e^o_s. If $type='m'$ then we simply let $(m,c^o,h^o,e^o_s)=\text{Parse}(\sigma^o)$. Let $c'=\mathcal{S}.D.M.c_d(c^o)$, $<h',h,c'>=\text{Parse}(h^o)$ and $<e_s, e'_s>=\text{Parse}(e^o_s)$; adversary A' returns $('m',(m,c',h',e'_s))$.

Otherwise, i.e. when $type \neq 'm'$, let $(c^o, e^o_s)=\text{Parse}(\sigma^o)$. Let $c'=\mathcal{S}.D.M(c^o)$, $<e_s, e'_s>=\text{Parse}(e^o_s)$; adversary A' returns $(type,(c',e'_s))$.

We now show that indeed, when A' is defined as above, then $p' \geq p^o \geq a$. The probabilities are taken over the coin tosses by A', A^o, and the experiment (for key-generation). Denote the coin tosses as follows:

- r^o_A : coins used by the `black-box` adversary A^o (provided by and known to A')
- r'_{ed}, r'_{sv}: coins tossed by the experiment (unknown to A'), for key generation.
- r_{ed}, r_{sv}: coins tossed by A' to generate keys for $\mathcal{S}$.

Let $r'_A = r^o_A || r_{ed} || r_{sv}$ denote all the random bits tossed by A'. Let $w'(r'_A, r'_{ed}, r'_{sv}) = true$ if and only if $\mathbf{ForExp}_{A',S,ISPUB,SPOOF}(k,q,l,\Delta,t,\rho_A) = "win"$ with the corresponding coin tosses.

Let $r^o_{ed} = r_{ed} || r'_{ed}$, $r^o_{sv} = r_{sv} || r'_{sv}$. Let $w^o(r^o_A, r^o_{ed}, r^o_{sv}) = true$ if and only if $\mathbf{ForExp}_{A^o,S^o,ISPUB,SPOOF}(k,q,t^o,\rho^o_A) = "win"$ with the corresponding coin tosses.

The claim follows since $w^o(r^o_A, r^o_{ed}, r^o_{sv}) \rightarrow w'(r'_A, r'_{ed}, r'_{sv})$. This holds, since (trivially) all conditions of step 7 of experiment $E' = \mathbf{ForExp}_{A',S,ISPUB,SPOOF}(k,q,t,\rho_A)[r'_A, r'_{ed}, r'_{sv}]$ hold if they (conditions of step 7) hold in experiment $E^o = \mathbf{ForExp}_{A^o,S^o,ISPUB,SPOOF}(k,q,t^o,\rho^o_A)[r^o_A, r^o_{ed}, r^o_{sv}]$. ∎

3.2.5. Relations to `Sign-then-Encrypt` and `Authenticate-then-Encrypt`

While our work is the first to analyze cascading of confidentiality and integrity schemes, including encryption, signature and MAC schemes, it is related to works which analyze how to achieve confidentiality and integrity by a cascade of an encryption scheme and a signature or MAC scheme (since all of these are special cases of confidentiality and integrity scheme). Indeed, from Lemma 3-3 and Lemma 3-4 it follows trivially that sign-then-encrypt as well as MAC-then-encrypt ensure both confidentiality (against CCA1) and integrity (i.e., a secure signcryption or authenticated encryption scheme). This provides a new proof for several corresponding theorems in [K01,BN00] (for MAC-then-encrypt) and [ADR02] (for sign-then-encrypt).

We note that sign-then-encrypt (and MAC-then-encrypt) does not provide IND-CCA2 security, as shown by [B98], [BN00] and [K01]. Notice these attacks require only `feedback only CCA2`, i.e. the adversary does not receive the plaintext but only the result of the verification. Also, proposition 4.6 of [BN00] shows that cascading does *not* ensure tolerance for non-malleable encryption. Cascading or other tolerant constructions under feedback-only CCA2 attacks, for non-malleable encryption, and for commitment schemes require additional investigation.

3.3. Parallel Construction

We now consider another basic construction: the parallel application of two cryptographic functions to the same input, where the output is the concatenation of the outputs of both functions. We call this the *Same-Input, Multiple-Outputs (SIMO) Parallel Construction*, or simply the *parallel construction.* The *parallel construction* of single-input (keyless) functions f,g is denoted as $f||g$ or $c_{||}(f,g)$, and defined as $c_{||}(f,g) = f||g(x) = f(x)||g(x)$. When the functions have inputs for random bits and/or keys, these are selected independently for the two functions, and the parallel construction is $f_{k,r}||g_{k',r'}(x) = f_{k,r}(x)||g_{k',r'}(x)$. We next define the parallel construction of confidentiality/integrity schemes.

The *parallel construction* of two CISs $\mathcal{S}$, $\mathcal{S}'$, denoted $c_{||}(\mathcal{S},\mathcal{S}') = \mathcal{S}||\mathcal{S}'$, is defined as follows. The definitions and proofs extend trivially to parallel construction of arbitrary number of CISs.

1. $\mathcal{S}||\mathcal{S}'.KG(r,r') = \mathcal{S}.KG(r)||\mathcal{S}'.KG(r')$.
2. $\mathcal{S}||\mathcal{S}'.ES_{e,e',s,s',r,r'}(m) = \mathcal{S}.ES_{e,s,r}(m) \; || \mathcal{S}'.ES_{e',s',r'}(m)$
3. $\mathcal{S}||\mathcal{S}'.D.M_{d,d'}(c) = \mathcal{S}.D.M_d(c)$

4. $\mathcal{S}||\mathcal{S}'.D.H_{d,d'}(c)=\mathcal{S}'.D.H_{d'}(c)\ ||\ \mathcal{S}.D.H_d(c)$
5. $\mathcal{S}||\mathcal{S}'.V_{e,e',v,v'}(m,c,<h',h>)=\mathcal{S}.V_{e,v}(m,c,h)\ \wedge\ \mathcal{S}'.V_{e',v'}(m,c,h')\ \wedge\ [\mathcal{S}'.D.M_{d'}(c)=\mathcal{S}.D.M_d(c)]$

Using two schemes in parallel provides tolerance for integrity but not for confidentiality. However, it does preserve confidentiality. We state these facts in the following lemma. The proof is a trivial adaptation of the lemmas for cascade, and omitted. Note also that this lemma is an extension of Theorem 4.3 of [BN00] (which considered only shared key encryption and MAC).

Lemma 3-5 For any $ISPUB:\{e,v\}\rightarrow\{T,F\}$, $SPOOF\in\{T,F\}$ holds:

1. The parallel construction $c_{||}$ of $n>1$ CIS schemes is $(n-1)$-tolerant for $CMA\text{-}INT_{ISPUB,SPOOF}$ and 0-tolerant for $CCA1\text{-}IND_{ISPUB}$ and $CPA\text{-}IND_{ISPUB}$
2. The parallel construction $c_{||}$ of $n>1$ CIS schemes is 0-tolerant for $s\rightarrow s^o$ with prerequisite $bounds[k, k', l, \Delta, \rho, h, \tau]$, where $s^o=IND_{ISPUB}(a,k,q,nl,n\Delta,t^0,\rho,\rho^o_A)$, $s=IND_{ISPUB}(a,k,q,l,\Delta,t,\rho,\rho_A)$ with $t=t^o+(n-1)\tau[KG]+n\tau[E]+(n-1)\sum_{j\in\{find,select\}}\sum_{f\in\{ES,D,V\}}q[j,f]\tau[f]$, $\rho_A=\rho^o_A+nk+n\rho$.
3. The parallel construction $c_{||}$ of $n>1$ CIS schemes is $(n-1)$-tolerant for $s\rightarrow s^o$ with prerequisite $bounds[k, k', l, \Delta, \rho, h, \tau]$, where $s^o=INT_{ISPUB,SPOOF}(a,k,q,nl,n\Delta,t^0,\rho,\rho^o_A)$, $s=INT_{ISPUB,SPOOF}(a,k,q,l,\Delta,t,\rho,\rho_A)$, $t=t^o+(n-1)\tau[KG]+n\tau[E]+(n-1)\sum_{j\in\{find,select\}}\sum_{f\in\{ES,D,V\}}q[j,f]\tau[f]$ and $\rho_A=\rho^o_A+nk+n\rho$.

4. Composite Tolerant Constructions

Often, neither cascade nor parallel construction provides tolerance for the desired specifications, e.g. for signcryption and authenticated encryption, which require both confidentiality and integrity. In such cases, we need a new construction. Often we can build the new construction by composing known constructions. In this section, we first define compositions of constructions, and prove a general composition lemma. We then present three composite constructions: *D, E* and *T,* which are all `serial-parallel` constructions, i.e. combination of cascade and parallel constructions. Constructions *D* and *E* are applications of the composition lemma, while *T* uses more specific details from the analysis of cascade and parallel constructions in the previous section.

4.1. Compositions of Arbitrary Constructions

While in general one could consider compositions of many constructions, we restrict our attention to compositions of two constructions. Such compositions accept as input two constructions c and c' and produce a composed construction denoted $c'\circ_I c$, where I is a mapping of the `candidate functions` to the constructions. Our definitions and results in this subsection refer to arbitrary specifications, including both concrete security specifications and asymptotic security specifications, as well as specifications which are not security related at all.

Let c be a construction of plurality p over $<D,R>$ which is t-tolerant for $s\rightarrow s'$, and let c' be a construction of plurality p' over $<D, R>$ which is t'-tolerant for $s'\rightarrow s''$. Let p° denote the plurality of the composition of c and c'; namely the input to the composite construction is a set f of p° functions, $f[i]\in F(D\rightarrow R)$. The composite construction first applies c to p' sets of p functions each, and then applies c' to the p' resulting functions. The composition is defined by the selection of the p functions input to each of the p' applications of the c construction, namely by a mapping $I:\{1,\ldots,p\}\times\{1,\ldots,p'\}\rightarrow\{1,\ldots,p^\circ\}$, where $I_i[j]$ identifies the j^{th} function input to the i^{th} c construction. Given I, the I-**composition** of c' and c, denoted $c'\circ_I c$, is

$$c'\circ_I c\left(f[1],\ldots,f[p^\circ]\right)=c'\left(c\left(f\left[I_1(1)\right],\ldots,f\left[I_1(p)\right]\right),\ldots,c\left(f\left[I_{p'}(1)\right],\ldots,f\left[I_{p'}(p)\right]\right)\right)$$

The following lemma shows that the security of the *I*-composition depends on a simple combinatorial property of mappings *I*. Consider mapping $I:\{1,\ldots,p\}\times\{1,\ldots,p'\}\rightarrow\{1,\ldots,p^{\circ}\}$ and some set $T\subseteq\{1,\ldots,p^{\circ}\}$ (of `weak inputs`). Let $G_i(I,T)=\{ I_i[j] \mid j=1,\ldots,p \} - T$, i.e. values $I_i[j]$, for some j, which are *not* in T; think of $G_i(I,T)$ as the `good selections` of I_i. We say that I is a (good) (t,t',t°)*-composition-structure* if for every $T\subseteq\{1,\ldots,p^{\circ}\}$ s.t. $|T|\leq t^{\circ}$ holds: $\left|\left\{1\leq i\leq p' \,\middle|\, |G_i(I,T)|\geq p-t\right\}\right|\geq p'-t'$.

Lemma 4-1 Let $I:\{1,\ldots,p\}\times\{1,\ldots,p'\}\rightarrow\{1,\ldots,p^{\circ}\}$ be a (good) (t,t',t°)-composition-structure. Let c be a construction of plurality p over $\langle D,R\rangle$ which is t-tolerant for $s\rightarrow s'$, and let c' be a construction of plurality p' over $\langle D, R\rangle$ which is t'-tolerant for $s'\rightarrow s''$. Then $c'\circ_I c$, is a construction of plurality p° over $\langle D, R\rangle$ which is t°-tolerant for $s\rightarrow s''$.

Proof: Consider any set f of p° functions, $f[i]\in F(D,R)$, and assume that $p^{\circ}-t$ of them satisfy specification s. Namely, for some set $\{i_j\}$ of $p^{\circ}-t$ indexes holds $s(f[i_j])=True$. We need to prove that for every choice $T\subseteq\{1,\ldots,p^{\circ}\}$ of up to t° functions in f which do not satisfy s, the function resulting from applying composed construction $c\circ_I c'$ to f satisfies s'' . Namely, we need to prove that $s''(c'\circ_I c(f[1],\ldots,f[p^{\circ}]))=True$. Let $f'[1],\ldots,f'[p']$ denote the p' intermediate functions, i.e. $f'[i]=c(f[I_i(1)],\ldots,f[I_i(p)])$; hence $c'\circ_I c(f[1],\ldots,f[p^{\circ}])=c'(f'[1],\ldots,f'[p'])$. Let $G(I,T)=\{i \text{ s.t. } |G_i(I,T)|\geq p-t\}$. By definition of I holds: $|G(I,T)|\geq p'-t'$.

If $i\in G(I,T)$, namely $|G_i(I,T)|\geq p-t$, then for at least $p-t$ of the functions $f[I_i(1)],\ldots,f[I_i(p)]$ holds $s(f[I_i(j)])=True$. Since c is t-*tolerant* for $s\rightarrow s'$ it follows that $s'(f'[i])=True$, for every $i\in G(I,T)$. Since c' is t'-*tolerant* for $s'\rightarrow s''$, it follows that: $s''(c'\circ_I c(f[1],\ldots,f[p^{\circ}]))=s''(c'(f'[1],\ldots,f'[p']))=True$. ∎

4.2. Serial-Parallel Compositions (E and D)

Serial-parallel constructions are composed of several cascading and parallel constructions, e.g. using composition structures. In particular, consider the following simple composition structures:

- Composition structure $D:\{0,1\}\times\{0,1\}\rightarrow\{0,1,2,3\}$ defined as $D_i[j]=2i+j$
- Composition structure $E:\{0,1\}\times\{0,1,2\}\rightarrow\{0,1,2\}$ defined as $E_i[j]=i+j \bmod 3$

These two structures allow us to compose construction c which is 1-tolerant for specifications $x\rightarrow x'$ and 0-tolerant for specifications $y\rightarrow y'$, with construction c' which is 0-tolerant for specifications $x'\rightarrow x''$ and 1-tolerant for specifications $y'\rightarrow y''$, resulting in constructions $c'\circ_I c$ and $c'\circ_I c$ which are both 1-tolerant for *both* $x\rightarrow x''$ and $y\rightarrow y''$. Namely,

Lemma 4-2 D and E are both (good) $(0,1,1)$ and $(1,0,1)$ composition-structures.

From the two Lemmas, we get:

Lemma 4-3 Let c, c_D', c_E' be constructions of plurality 2, 2 and 3 respectively. If c is t-tolerant for $s\rightarrow s'$ where $t\in\{0,1\}$, and c_D', c_E' are $(1-t)$-tolerant for $s'\rightarrow s''$, then $c_D'\circ_D c$ and $c_E'\circ_E c$ are both 1-tolerant for $s\rightarrow s''$.

Let $c_{||}$ denote the parallel construction of plurality 2, and $c_{|||}$ denote the parallel construction of plurality 3. Define constructions $c_D= c_D'\circ_D c_{||}$, $c_E=c_E'\circ_{\boldsymbol{E}} c_{|||}$. From Lemma 4-3 together with Lemma 3-3, 3-4 and 3-5, we get:

Lemma 4-4 For any $ISPUB:\{e,v\}\rightarrow\{T,F\}$, $SPOOF\in\{T,F\}$ holds:

1. The D and E constructions c_D, c_E are 1-tolerant for <u>both</u> $CMA\text{-}INT_{ISPUB,SPOOF}$ <u>and</u> $CCA1\text{-}IND_{ISPUB}$ and $CPA\text{-}IND_{ISPUB}$

2. The E construction c_E is *1*-tolerant with prerequisite *bounds[k, k', l, Δ, ρ, h, τ]* for both $IND \rightarrow IND^o$ and $INT \rightarrow INT^o$, where:
   a. $IND^o = IND_{ISPUB}(a,k,q,3l,6\Delta,t^0, \rho,\rho^o_A)$,
   b. $IND = IND_{ISPUB}(a,k,q,l,\Delta,t, \rho,\rho_A)$
   c. $INT^o = INT_{ISPUB,SPOOF}(a,k,q,3l,6\Delta,t^0, \rho,\rho^o_A)$
   d. $INT = INT_{ISPUB,SPOOF}(a,k,q,l,\Delta,t, \rho,\rho_A)$
   e. $t = t^o + 2\tau[KG] + 5\tau[E] + 5 \sum_{j\in\{find,select\}} \sum_{f\in\{ES,D,V\}} q[j,f]\tau[f]$,
   f. $\rho_A = \rho^o_A + 3k + 5\rho$.
3. The D construction c_D is *1*-tolerant with prerequisite *bounds[k, k', l, Δ, ρ, h, τ]* for both $IND \rightarrow IND^o$ and $INT \rightarrow INT^o$, where:
   a. $IND^o = IND_{ISPUB}(a,k,q,2l,4\Delta,t^0, \rho,\rho^o_A)$,
   b. $IND = IND_{ISPUB}(a,k,q,l,\Delta,t, \rho,\rho_A)$
   c. $INT^o = INT_{ISPUB,SPOOF}(a,k,q,2l,4\Delta,t^0, \rho,\rho^o_A)$
   d. $INT = INT_{ISPUB,SPOOF}(a,k,q,l,\Delta,t, \rho,\rho_A)$
   e. $t = t^o + 3\tau[KG] + 4\tau[E] + 3 \sum_{j\in\{find,select\}} \sum_{f\in\{ES,D,V\}} q[j,f]\tau[f]$,
   f. $\rho_A = \rho^o_A + 3k + 3\rho$.

4.3. The T construction

The D and E constructions are efficient and practical, and provide tolerant design for any pair of specifications when one if tolerant under cascade and the other is tolerant under parallel construction. However, in the special case of confidentiality and integrity schemes (CIS), there is an even better construction – the T construction. This construction takes advantage of the observation that if scheme $\mathcal{S}'$ satisfies integrity specifications, then $\mathcal{S}\circ\mathcal{S}'$ also preserves integrity, for any scheme $\mathcal{S}$.

The *T construction* of $\mathcal{S}_1,\ldots\mathcal{S}_n$, denoted $T(\mathcal{S}_1,\ldots\mathcal{S}_n)$, is defined as $T(\mathcal{S}_1,\ldots\mathcal{S}_n) = \mathcal{S}_1\circ\ldots\circ\mathcal{S}_n\circ(\mathcal{S}_1||\ldots||\mathcal{S}_n)$.

Lemma 4-5 For any *ISPUB:{e,v}→{T,F}, SPOOF∈{T,F}* holds:

1. The T construction of *n>1* CIS schemes is *(n-1)*-tolerant for both $CMA\text{-}INT_{ISPUB,SPOOF}$ and $CCA1\text{-}IND_{ISPUB}$ and $CPA\text{-}IND_{ISPUB}$
2. The T construction of *n>1* CIS schemes is *(n-1)*-tolerant with prerequisite *bounds[k, k', l, Δ, ρ, h, τ]* for both $IND \rightarrow IND^o$ and $INT \rightarrow INT^o$, where:
   a. $IND^o = IND_{ISPUB}(a,k,q,nl,2n\Delta,t^0, \rho,\rho^o_A)$,
   b. $IND = IND_{ISPUB}(a,k,q,l,\Delta,t, \rho,\rho_A)$
   c. $INT^o = INT_{ISPUB,SPOOF}(a,k,q,nl,2n\Delta,t^0, \rho,\rho^o_A)$
   d. $INT = INT_{ISPUB,SPOOF}(a,k,q,l,\Delta,t, \rho,\rho_A)$
   e. $t = t^o + (n-1)\tau[KG] + 2n\tau[E] + (n-1) \sum_{j\in\{find,select\}} \sum_{f\in\{ES,D,V\}} q[j,f]\tau[f]$,
   f. $\rho_A = \rho^o_A + (n-1)k + (2n-1)\rho$.

Proof: immediate from Lemma 3-3, 3-4 and 3-5. ■

5. Conclusions and Open Questions

In this work we presented tolerant constructions and compositions for some of the most important and practical cryptographic mechanisms, including cryptosystems, signature/MAC schemes, signcryption and authenticated encryption. The constructions are very efficient and practical.

An obvious challenge is to find tolerant constructions for additional cryptographic primitives. In particular, we believe it should be feasible to extend our results for commitment and committing-encryption schemes. A harder challenge is to find a tolerant construction for keyless hash functions, which are widely deployed in practice. Other primitives which are related to the current work include non-malleable commitment schemes and cryptosystems [DDN91], Universally Composable commitment schemes [CF01] and Perfectly one-way hash functions [C97,CMR98].

Acknowledgements

I wish to thank Mihir Bellare, Ran Canetti, Shai Halevi, Boaz Patt-Shamir, Avi Wigderson and anonymous referees for helpful comments and discussions. This work is supported by ISF grant.

References

[AB81] C. A. Asmuth and G. R. Blakley. *An efficient algorithm for constructing a cryptosystem which is harder to break than two other cryptosystems*. Comp. and Maths. with Appls., 7:447-450, 1981.

[ABCV98] B. Aiello, M. Bellare, G. Di Crescenzo, and R. Venkatesan, Security amplification by construction: the case of doubly-iterated, ideal ciphers, Proc. of CRYPTO 98.

[ADR02] Jee Hea An, Yevgeniy Dodis and Tal Rabin, On the Security of Joint Signature and Encryption, in Theory and Application of Cryptographic Techniques, pp. 83-107, 2002. Also in Advances in Cryptology - EUROCRYPT 2002, volume 2332 of Lecture Notes in Computer Science, pages 83-107. Springer-Verlag, 2002.

[AN95] Ross Anderson, Roger Needham. *Robustness Principles for Public Key Protocols*. In Proceedings of Int'l. Conference on Advances in Cryptology (CRYPTO 95), Vol. 963 of Lecture Notes in Computer Science, pp. 236-247, Springer-Verlag, 1995.

[AN96] Martin Abadi, Roger Needham. *Prudent Engineering Practice for Cryptographic Protocols*. IEEE Transactions on Software Engineering, 22, 1 (Jan.), 1996, pp. 6-15.

[B98] Daniel Bleichenbacher. *Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS*#1. In Advances in Cryptology - CRYPTO '98, LNCS 1462, pages 1-12. Springer, 1998.

[BDJR97] M.Bellare, A.Desai, E.Jokipii, P.Rogaway: A Concrete Security Treatment of Symmetric Encryption, Proceedings of the 38th IEEE Symposium on Foundations of Computer Science (FOCS), pp. 394-403, 1997. Revised version at http://www-cse.ucsd.edu/users/mihir/papers/sym-enc.html.

[BKR94] Mihir Bellare, Joe Kilian and Phil Rogaway, "The security of cipher block chaining", Journal of Computer and System Sciences, Vol. 61, No. 3, Dec 2000, pp. 362-399. Extended abstract in Advances in Cryptology - Crypto 94 Proceedings, Lecture Notes in Computer Science Vol. 839, Y. Desmedt ed, Springer-Verlag, 1994.

[BN00] Mihir Bellare and Chanathip Namprempre. *Authenticated encryption: Relations among notions and analysis of the generic construction paradigm*. In T. Okamoto, editor, Asiacrypt 2000, volume 1976 of LNCS, pages 531-545. Springer-Verlag, Berlin Germany, Dec. 2000.

[BR97] Mihir Bellare and Phillip Rogaway, Collision-Resistant Hashing: Towards Making UOWHFs Practical, Extended abstract was in Advances in Cryptology- Crypto 97 Proceedings, Lecture Notes in Computer Science Vol. 1294, B. Kaliski ed, Springer-Verlag, 1997. Full paper available at http://www.cs.ucsd.edu/users/mihir/papers/tcr-hash.html.

[BSZ02] Joonsang Baek, Ron Steinfeld, and Yuliang Zheng. *Formal proofs for the security of signcryption*. In David Naccache and Pascal Pailler, editors, 5th International Workshop on Practice and Theory in Public Key Cryptosystems - PKC 2002, pp. 80-98, LNCS Vol. 2274, 2002.

[DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography. In Proceedings of the 23rd Symposium on Theory of Computing, ACM STOC, 1991.

[DK94] Ivan B. Damgård, Lars Ramkilde Knudsen. Enhancing the Strength of Conventional Cryptosystems, BRICS report RS-94-38, November 1994.

[DPP94] Ivan B. Damgård, Torben P. Pedersen, Birgit Pfitzmann: On the Existence of Statistically Hiding Bit Commitment Schemes and Fail-Stop Signatures; Crypto '93, LNCS 773, Springer-Verlag, Berlin 1994, 250-265.

[DPP98] Ivan B. Damgård, Torben P. Pedersen, Birgit Pfitzmann: Statistical Secrecy and Multi-Bit Commitments; IEEE Transactions on Information Theory 44/3 (1998) 1143-1151.

[EG85] S. Even and O. Goldreich, On the Power of Cascade Ciphers, ACM Transactions on Computer Systems, Vol. 3, 1985, pp. 108-116.

[FIP180] National Institute of Standards and Technology, Federal Information Processing Standards Publication, FIPS Pub 180-1: Secure Hash Standard (SHA-1), April 17, (1995), 14 pages.

[Go01] Oded Goldreich, The Foundations of Cryptography, Volume 1 (Basic Tools), ISBN 0-521-79172-3, Cambridge University Press, June 2001.

[Go02] Oded Goldreich, Fragments of a Chapter on Encryptions Schemes, Extracts from working drafts of Volume 2, The Foundations of Cryptography.

[GGM84] Oded Goldreich and Shafi Goldwasser and Silvio Micali "How to Construct Random Functions" Journal of the ACM, 33(4), 1984, 792-807.

[GH04] Yitchak Gertner and Amir Herzberg, “Committed encryption and publicly verifiable signcryption”, submitted for publication, 2004.

[GIL*90] Oded Goldreich, R. Impagliazzo, L. Levin, R. Venkatesen, D. Zuckerman. "Security preserving amplification of randomness", 31st Annual Symposium on Foundations of Computer Science, IEEE Computer Society Press, (1990), 318-326.

[GM84] Shafi Goldwasser and Silvio Micali. "*Probabilistic Encryption*," JCSS (28), 1984, 270-299.

[HILL99] Johan Hastad, Rudich Impagliazzo, Leonid A. Levin, and Mike Luby, Construction of a Pseudorandom Generator from any One-Way Function. SIAM Journal on Computing, Vol. 28, No. 4, pp. 1364-1396, 1999.

[HL92] Amir Herzberg and Mike Luby, "Public Randomness in Cryptography", proceedings of CRYPTO 1992, ICSI technical report TR-92-068, October, 1992.

[HM96] Shai Halevi and Silvio Micali, "Practical and Provably-Secure Commitment Schemes from Collision Free Hashing", in Advances in Cryptology - CRYPTO96, Lecture Notes in Computer Science 1109, Springer-Verlag, 1996, pp. 201-215.

[HP86] Amir Herzberg and Shlomit Pinter, “Composite Ciphers”, EE Pub. no. 576, Dept of Electrical Engineering, Technion, Haifa, Israel, Feb. 1986.

[K01] Hugo Krawczyk, "*The Order of Encryption and Authentication for Protecting Communications (or: How Secure Is SSL*?)," In Crypto '01, pp. 310-331, LNCS Vol. 2139, J. Kilian ed., Springer-Verlag, 2001.

[LV01] Arjen K. Lenstra and Eric R. Verheul. *Selecting Cryptographic Key Sizes*. Journal of Cryptology: The Journal of the International Association for Cryptologic Research, 14(4):255--293, September 2001.

[MM93] U.M. Maurer and J.L. Massey, Cascade ciphers: the importance of being first, Journal of Cryptology, Vol. 6, No. 1, pp. 55-61, 1993.

[MOV96] Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone, Handbook of Applied Cryptography, Section 9.2.6, CRC Press, ISBN 0-8493-8523-7, October 1996. Available online at http://www.cacr.math.uwaterloo.ca/hac/.

[RFC2246] T. Dierks, C. Allen, The TLS Protocol: Version 1.0, Network Working Group, Internet Engineering Task Force (IETF). Available online at http://www.ietf.org/rfc/rfc2246.txt.

[R00] Eric Rescorla. SSL and TLS: Designing and Building Secure Systems. Addison-Wesley, 2000.

[Sc96] Bruce Schneier, Applied Cryptography, John Wiley & Sons, 1996.

[Sh00] Victor Shoup, *Using hash functions as a hedge against chosen ciphertext attacks*, Adv. in Cryptology -- Proc. of Eurocrypt '2000, LNCS 1807, pp. 275-288.

[Z97] Yuliang Zheng, *Digital signcryption or how to achieve cost(signature+encryption) << cost(signature)+cost(encryption*), in Advances in Cryptology - CRYPTO'97, Berlin, New York, Tokyo, 1997, vol. 1294 of Lecture Notes in Computer Science, pp. 165--179, Springer-Verlag.