I. J. Computer Network and Information Security, **2016, 10, 1-11**
Published Online October 2016 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2016.10.01

Computational Independence in the Design of Cryptographic Protocols

István Vajda
Technical University of Budapest, Department of Informatics, Budapest, Hungary
E-mail: vajda@hit.bme.hu

Abstract—Statistical independence of instances of primitives and protocols is a clear-cut approach for guaranteeing protection against harmful interactions in concurrent and multi-execution environment. Therefore it is surprising that computational indistinguishability of independence from dependence between two or several random variables received no attention since the introduction of classic binary pseudorandom sequences. In this work we propose the use of the notion of computational independence (CI) in the analysis and design of provably secure cryptographic protocols. We generalize the classic result on equivalence of unpredictability and CI to general non-binary random variables. An application of this result is the use of unpredictability-based standard secure primitives in supporting the achievement of CI. This work is inherently related to Canetti's universal composition framework [4], [5].

Index Terms—Cryptographic protocols, provable security, universal composability.

I. Introduction

When an adversary aims to break or fabricate a cryptographic object with some unknown parameter it tries to collect/learn associated data from past or concurrent events. Such a learning effort can be neutralized if statistically independent observations are only available for the adversary. Though such clear-cut approach cannot be implemented in general, it can serve as a starting step for feasible (simplified) design and analysis of security protocols. Cost efficient realizations unavoidably introduce statistical dependence within and between protocol instances. Fortunately security guarantees remain intact in a computational environment even in such circumstances if at least some form "computational independence" can be attained.

Four types of independence relations between random variables will be considered: unpredictability (UP), computational independence (CI), statistical independence (SI) and functional independence (FI). Keeping statistical independence between instances of primitives and instances of protocols is a clear-cut approach for guaranteeing protection against harmful interactions in concurrent and multi-execution environment. However protection provided by statistical level of independence costs a lot, therefore several times it is used only as a tool in the beginning step of the analysis for reduction of the complexity of the analysis. Changing from SI to cost efficient computational independence brings into the picture the notion of CI, UP and FI. Our definitions for notions of CI, UP and FI will be introduced in the paper. UP essentially prevents attacks against correctness and privacy carried out via fabrication of valid cryptographic blocks and exploration of private data, respectively. CI as its name indicates provides the same independence guaranties under complexity constraint as SI does in unconditional case.

Our main technical result claims equivalence between unpredictability and computational independence for general non-binary random variables. This result can also be considered as an extension of the analogous classic result on pseudorandom generators, the equivalence of pseudo-randomness and unpredictability for ensembles of binary random variables. In our case uniformity of the underlying distributions also plays crucial role as we show that for general distributions such equivalence does not stand.

We explore useful relationships between independence notions under study:

In general CI is stronger property than UP, as the former implies the latter. We show separation result for notions CI and SI. We define CI of protocol instances and show relationships between the CI of protocol messages and the CI of corresponding protocol instances. Notion of functional independence (FI) provides the bridge to security notion by universal composability on one side and CI of protocol instances on the other side.

We are not aware of publication which mentions the notion of CI in the field of protocol analysis.

We underline that UP is a constructive property which can be directly associated to standard security properties of primitives. Our main example is that UP provides the bridge to EU-CMA standard-security of primitives as the latter is an important example for the weak version of the UP property. In this sense, we give a unified approach to the role of independence in the analysis and design of provably secure protocols.

Several examples help deeper understanding and serve as illustration for our notions and arguments.

The structure of the paper is the following. Section 2 presents the related works. In Section 3 we analyze the relationship between notions CI and SI as well as between CI and UP. Here we show that CI is stronger

property than UP, in general, but under "plausible" assumption the two notions become equivalent. Section 4 presents several important applications for computational independence (CI), discussing both cryptographic primitives and protocols. Here we also show the relationship to the Universal Composition framework. In Section 5 we draw conclusions. In Appendix we give our notions for strong and weak unpredictability (UP), illustrated with several examples.

II. Related Works

A computational version of statistical independence appeared first in the classic paper of Yao [21] under notions of "effective conditional entropy" and "effective mutual information". The aim was to define a computational counterpart of Shannon's perfectly secure encryption. The notion of computational indistinguishability of two probability distributions (equivalently two random variables) is fundamental in the classic theory of standard secure cryptographic primitives. Interestingly similar indistinguishability notion for independence/dependence of random variables has not been introduced until recently. Indeed, the only reference we found to it is rather fresh [8] (compared to the time of publication of [21]). The author of [8] after coining the name CI returned to Yao's application for encryption security in [21]. In our paper we will use CI in different context both theoretically and practically. We embed this notion into the set of related notions mentioned above. Furthermore our application environment for CI is much wider than that of [8] as our aimed application environment is the design and analysis of provably secure protocols, in particular universally composable protocols [4], [5]. Such a setting where many protocol instances may run concurrently under adversarial coordination is obviously the richest one for potential dependence problems. Understandably, our examination is inherently related to Canetti's universal composition framework [4], [5]. This framework has strong theoretical foundation and the corresponding analysis and design techniques have had wide application in the last decade [1-3], [6-7], [9-20]. Notions UP, CI and SI are defined for random variables, while notion FI is inherently related to the notion of UC-security of protocols.

III. Computational Independence

"Natural" definition of computational independence of two random variables is their computational indistinguishability (CI) from a pair of statistically independent random variables. One of the main points in this work is to underline the crucial role of CI in the design and analysis of cryptographic protocols.

Definition 1 (computational independence):

Computational indistinguishability of independence (computational independence, CI) means computational indistinguishability of a pair $V_1=(m_1,m_2)$ of random variables from a pair of random variables $V_2=(M_1,M_2)$, where M_1 and M_2 are statistically independent.

In the game of CI a distinguishing algorithm Z gets a sample $v=(v_1,v_2)$ which is a realization of V_1 or V_2 chosen by coin flipping result σ. Algorithm Z successfully distinguishes V_1 and V_2 if $P(Z(v)=\sigma) \geq 1/2+\varepsilon/2$ for non-negligible ε. If none of PPT algorithms is successful in the above game we say that random variables m_1 and m_2 are independent by CI.

For simplicity of presentation we consider the case when random variables m_1 and M_1 similarly random variables m_2 and M_2 have the same probability distribution (p.d.). Let the former be denoted by $H(x)$ the latter by $G(y)$. In other words the marginal distributions of the two dimensional random vectors V_1 and V_2 are equal. We also introduce conditional probability distribution (c.p.d.) $F(y|x)=P(m_2=y|m_1=x)$. The main result of this chapter claims equivalence between CI and UP, where our main assumption will be that p.d-s $G(y)$ and $F(y|x)$ are computationally indistinguishable from uniform (for illustration see Fig. 1).

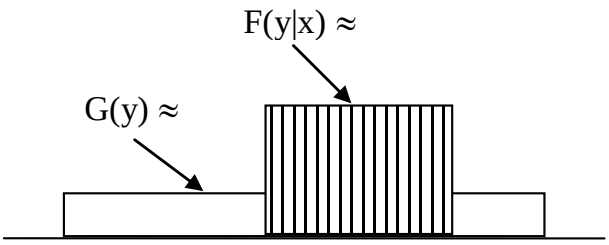

Fig.1. Assumption of Uniform Distributions

First we separate notions SI and CI.

Property 1: Random variables m_1 and m_2 are CI independent if $\lambda = \sup_x \mathrm{Var}(F(y|x), G(y))$ is negligible, where mapping $\mathrm{Var}(.,.)$ stands for the statistical distance.

Proof: Using the definitions of CI and of statistical distance we get

$$P\left(Z(v1,v2)=\sigma\right) = E_x[\int_y P\left(Z(x,y)=\sigma \mid x\right) dF(y \mid x)]$$

$$= E_x[\int_y P\left(Z(x,y)=\sigma \mid x\right) dG(y)] +$$

$$E_x[\int_y P\left(Z(x,y)=\sigma \mid x\right) d(F(y \mid x) - G(y))]$$

$$\leq \alpha + E_x[\int_y | dF(y \mid x) - dG(y) |] \leq 1/2 + 2\lambda$$

where α=P$\left(\mathrm{Z}\left(\mathrm{M}_1, \mathrm{M}_2\right)=\sigma\right)$=1/2.

In this chapter we examine the relationship between unpredictability (UP) and CI (our definition for UP is placed in the Appendix). First we separate these two notions by their strength.

Property 2: Set D of distinguishing algorithms in CI game is strictly larger than set P of predictors (F algorithms in Definition 5 in the Appendix).

Proof: $|P| \subseteq |D|$: predictor (F,δ) can be used to generate a distinguisher d: d(v)=1 if $F(v_1)= \delta(v_2)$, otherwise $\mathbf{d(v)=0}$. Note the number of $V_2^{k_1} \to V_2^{k_2}$ mappings (set P) is $2^{k_2 2^{k_1}}$ while the number of $V_2^{k_1+k_2} \to V_2^1$ mappings (set D) is $2^{2^{k_1+k_2}}$, where $|v_1|=k_1$, $|v_2|=k_2$ and δ is the identity mapping.

The next example illustrates the intuitive meaning of notion CI.

Example 1: Consider two instances I_1 and I_2 of a protocol where the only cryptographic primitive is EU-CMA secure digital signature, furthermore the instances use a common signature key. EU-CMA security prevents successful prediction of signatures in instance I_2 from signatures in instance I_1 and as digital signatures are the only cryptographic mappings, instance I_2 is unpredictable from instance I_1 and vice versa. In spite of this (weak) unpredictability, the two instances are not computationally independent, which were the case if I_1 and I_2 would use (statistically) independent signature keys. Indeed, a distinguishing algorithm being aware of the public verification algorithm could easily distinguish the case of a common key from the one with independent keys.

Consider now the case when instead of digital signature primitive message authentication (MAC) primitive is used. The question remains the same: can we distinguish the cases when a common key and when independent keys are used in the instances. Now there is no publicly available (efficient) verification algorithm. Assume PRF-CPA secure symmetric key encryption algorithm generates CBC-MAC with full block-length. Furthermore, we observe sets of MACs $S_1=\{(m_{11}, MAC_{k_1}(m_{11})),..., (m_{1n}, MAC_{k_1} m_{1n}))\}$ and $S_2=\{(m_{21}, MAC_{k_2}(m_{21})),..., (m_{2n}, MAC_{k_2} m_{2n}))\}$ from instance I_1 and I_2, respectively, where the messages are different with overwhelming probability. Clearly, if we used random function for encryption we were not able to decide about corresponding keys if $k_1=k_2$ or not. Consequently this remains the case also for pseudorandom encryption, which implies that we identified an example for CI which is separated from SI.

Property 2 suggests that computational independence is strictly stronger than unpredictability, in general. In this chapter we show a condition under which UP implies CI. A restriction will be imposed on the corresponding distributions $\left(\mathrm{F}(y|x), \mathrm{G}(y)\right)$. We also show an argument that in the general case of distributions CI is strictly stronger than UP.

For simplicity of notations first we will consider the case of identity mapping δ. The extension to the general case of δ is given subsequently.

Let the domain of random variable m_1 and m_2 be denoted by X and Y, respectively. Let $H_{F,x}=\{y : F(y|x)>0\}$, i.e. set $H_{F,x} \subseteq Y$ denotes the support of c.p.d. $F(y|x)$. Similarly H_G denotes the support of p.d. G. Obviously $H_{F,x} \subseteq H_G \subseteq Y$ for all x.

Assumption 1: Assume that

a) c.p.d. $F(y|x)=P(m_2 = y|m_1=x)$ and p.d. $G(y)= P(m_2=y)$ are computationally indistinguishable from uniform distribution, respectively,
b) ratio $|H_{F,x}|/|H_G|$ is non-negligible except for negligible set of x values.

Comments follow to Assumption 1:

to a): It is implicit in this assumption that p.d. G(y) can efficiently be sampled. Indeed, variable m_2 is a protocol message or part of it, output of a known efficient mapping of constants and efficiently samplable random variables (typically coin flipping sequences).

to b) This assumption is a technical consequence of the standard definition of algorithmic indistinguishability of p.d-s, where the success of a distinguishing algorithm Z is equivalent to its non-negligible gain. When we want to inherit this gain by a predictor Z' built on Z we need this assumption.

Theorem 1: Under Assumption 1 unpredictability (UP) implies computational independence (CI).

Note this claim resembles the classic theorem on the completeness of the next-bit test: "An ensemble of sequences of binary random variables is pseudorandom if and only if it is non-predictable." Here pseudo-randomness means computational indistinguishability of an efficiently generated series of binary random variables from a series of statistically independent uniformly distributed binary variables (in short from a coin flipping sequence). Prediction means prediction of the next bit within the series from the previous bits. In contrast we consider the algorithmic independence of non-binary

random variables which may have different distributions, in general.

Proof of Theorem 1: Assume the existence of a binary algorithm Z such that it is able to distinguish the pair of random variables from a pair with independent elements. (Z outputs 1 when it decides to V_1). We construct an algorithm Z' such that it is able to predict with non-negligible probability. Algorithms have access to *pub_inf* which will not be explicit in the formulation below for simplicity of notations. According to Assumption 1/a we are allowed to use (genuine) uniform p.d-s for F(y|x) and G(y) in the calculations below.

The predictor is successful if its output random variable has distribution computationally indistinguishable from $F(y|x)$ (which by our assumption is uniform over $H_{F,x}$). Consider the following prediction algorithm:

Algorithm 1:

1. Z' gets sample x as input from X.
2. Z' choses sample y^*: $y^* \leftarrow_r H_G$.
3. Z' runs Z with input (x, y^*):

   if $Z(x,y^*)=1$ then Z' outputs y^*,
   otherwise Z' outputs y': $y' \leftarrow_r H_G$.

Let P_s denote the probability of event E that the output of Z' falls into set $H_{F,x}$. Note that as y^* and y' are uniformly random over H_G, furthermore $H_{F,x} \subseteq H_G$, therefore distribution of the output of Z' conditioned on event E is uniform. It remains to prove that P_s is non-negligible. According to Algorithm 1

$$P_s = P\left(\{Z(x, y^*) = 1\} \cap \{y^* \in H_{F,x}\}\right) + P\left(\{Z(x, y^*) = 0\} \cap \{y' \in H_{F,x}\}\right) \quad (1)$$

For the first term on the RHS of (1) we get

$$P\left(\{Z(x, y^*) = 1\} \mid \{y^* \in H_{F,x}\}\right) P\left(y^* \in H_{F,x}\right) \geq p_1 q_x,$$

$q_x = P\left(y^* \in H_{F,x}\right) = \frac{|H_{F,x}|}{|H_G|}$. Similar lower bound can be obtained for the second term in (1)

$$P\left(Z(x, y^*) = 0\right) P\left(y' \in H_{F,x}\right) \geq p_2 q_x,$$

and this leads to lower bound

$$P_s \geq q_x(p_1 + p_2) \geq q_x(1 + \varepsilon) = q_x + q_x\varepsilon$$

If distinguisher Z is not available then $P_s = q_x$. Therefore if q_x is non-negligible then predictor Z' has non-negligible gain $q_x\varepsilon$.

Theorem 2: In the general case of distributions over pair of random variables $V_1=(m_1,m_2)$ unpredictability does not imply computational independence.

Proof: Consider the best algorithm Z* for distinguishing V_1 and V_2. Algorithm Z* decides on V_1 over subset of samples $S=\{z: P_{V_1}(z) > P_{V_2}(z), z \in XxY\}$. Note Z* is insensitive to the relative magnitude (ratio $P_{V_1}(z)/P_{V_2}(z)$ or difference $P_{V_1}(z) - P_{V_2}(z)$) of probabilities. In other words Z* carries no information about those relative values. Recall the task for predictor Z' is to output sample y with p.d. $F(y|x)$. When we would like to generate a successful prediction algorithm Z' by reducing the task of prediction to a successful distinguishing algorithm Z*, in the general case of distributions we would fail. Indeed, as Z* is insensitive in the above mentioned meaning we cannot expect that the output of Z' will have the wanted distribution. Note that the assumed uniformity properties of distributions in Theorem 1 are immune to this insensitivity property. Furthermore, though the best distinguisher Z* is not polynomial in general, the above argument remains valid for our purposes in cases of distributions with polynomial size of support.

A. Extension to general mapping δ and to more than two variables

Recall the examination in the previous chapter refers to the special case when δ is the identity mapping (i.e. we would like to predicate the whole "block" of m_2). However according to Definition 5 for unpredictability of m_2 from m_1 we require that no part of m_2 can be predicted and in general the same must be true for any efficient, deterministic binary mapping of m_2.

Property 3: Computational independence implies unpredictability.

Proof: By contradiction assume there exists an efficient, deterministic binary mapping δ of m_2 such that $\delta(m_2)$ is successfully predictable from m_1 with prediction algorithm Z_δ. Distinguisher algorithm Z' in the computational independence game gets input $v=(v_1,v_2)$ and runs algorithm Z_δ with input v_1. Algorithm Z' outputs 1 (i.e. decides that pair (m_1,m_2) is its input) if Z_δ predicates v_2 successfully.

The extension of Assumption 1 and Theorem 1 to general mapping δ is straightforward.

Lemma 1: Assumption 1 implies Assumption 1' for the general case of δ, where

Assumption 1': Collect polynomial many random samples from distribution G(y) into set W and generate set $W'=\{\delta(y) : y \in W\}$. We assume that probability $P(\delta(m2) \in W' \mid m_1=x)$ is non-negligible possibly except for a negligible set of x values.

Proof: Note, that event $\{m_2 \in W\}$ on condition $\{m_1=x\}$ implies event $\{\delta(m2) \in W'\}$ on the same condition for arbitrary x.

The proof of Theorem 1 can be repeated with changing distribution G(y) to the distribution of random variable $\delta(m_2)$ (in Algorithm 1).

Definition 1 of CI is given for pairs of random variables. The extension to joint independence of several random variables is straightforward: we consider the computational indistinguishability of a vector of the corresponding vector of random variables $V_1=(m_1,...,m_k)$ from vector $V_2=(M_1,...,M_k)$ where coordinate variables $M_1,...,M_k$ are jointly statistically independent.

Example 2: For pseudorandom primitives CI property is straightforward by their definition. Essentially these definitions directly provide CI property (which in turn implies unpredictability). Recall in those definitions uniformity of the underlying random variables is part of the definition (or implicit in them). Pseudorandom security for primitives is the strongest guarantee within computational framework. In contrast for less strong definitions like EU-CMA, semantic security or non-malleability the definition is unpredictability-oriented and as shown above we arrive to CI properties under restrictions on (or better to say by specifying) underlying p.d-s.

IV. Applications

In this section we present several applications for the notion of computational independence both in the field of secure cryptographic primitives and protocols. Our main goal is to protect a protocol instance from the adversary attacking in a concurrent environment. The point is that the target instance should be computationally independent from all other instances running in the same environment. Such independence can be established via proper unpredictability properties at the level of applied cryptographic primitives. CI level of independence can be achieved based on equivalence between UP and CI properties like the one stated in Theorem 1.

A. CI and the cryptographic primitives

The main observation in this sub-section is that by our definition of UP the EU-CMA standard-security property of primitives becomes an example for the weak version of the UP property. Subsequently, further important examples will also be shown for unpredictability.

EU-CMA security

Consider a standard secure cryptographic primitive given by mapping f(k,u) , where argument k and u corresponds to key and message, respectively. The adversary has oracle access to algorithm f(K,.) with unknown key K and is tasked with forging output f(K,u) for a new message u:

EU-CMA security guaranties that the adversary will not be successful if for any poly size set $\{f(K,u_1),...,f(K,u_N)\}$ requested for arbitrarily chosen message set $U=\{u_1,...,u_N\}$, variable $m_2=f(K,u)$ is unpredictable from $m_1=(f(K,u_1),...,f(K,u_N))$ for any u such that $u \neq u_i$, i=1,...,n .

Note EU-CMA security for a primitive f is an example for (weak) unpredictability: we require (weak) unpredictability for any pair of random variables (m_1, m_2) , where random variable m_1 is learned via accessing the corresponding oracle and m_2 is an output of f for a fresh input.

For example when the adversary wishes to fabricate a valid digital signature or authentication code without knowing the corresponding secret key, m_1 corresponds to signatures (MAC values) obtained from the oracle for messages sent in the requests, m_2 will be the signature (MAC value) for a new message m'.

Obviously, if protocol instances running concurrently use an EU-CMA secure primitive f as a common primitive (i.e. with the same secret key X) then prediction-based attack against a target instance can be neutralized if different instances are coordinated in choosing inputs to f from disjoint subsets. Recall a corresponding technique is used in implementation of the JUC approach [5], where concurrent instances of the same protocol have a common module implementing primitive f.

In the more general GUC approach [7], the target instance and any other instances have access to a common public (global) variable. In this case coordination over the input space cannot be established in general (think on the case when one of the protocols is controlled by an adversary). With respect to predictability in GUC scenario the two extreme cases are the following:

In the first case anybody can code with a public algorithm but only one party can decode (e.g. public key encryption). Predictability is obvious and cannot be eliminated, therefore in this case we have to change from globally predictable instances to locally statistically independent ones (e.g. independent encryption keys per instance).

In the other case of primitives an honest party can access the secret key but anybody can verify with a public algorithm. For such primitives (weak) UP can be maintained by requiring EU-CMA-security. The known example is the digital signature. When a party controlling a signature key is a party in different instances of even different protocols he is able to ensure coordination of signatures issued by himself in spite of global (i.e. multi-protocol and multi-instance) use of the same instance of

signature primitive. It seems plausible that digital signature is the only primitive which provides global (weak) UP capability.

Semantically secure encryption

As we already mentioned, formulation of Definition 5 has resemblance to the standard definition of semantic security of public key encryption with the following cast of roles: random variables m_1 and m_2 correspond to a ciphertext and corresponding cleartext, respectively, where mapping δ models the partial information on the clear-text.

Non-malleable encryption

It is an example for (weak) unpredictability. The cast of roles is the following: m_1=(c,M) where c is a ciphertext for message u, M is the data collected, and furthermore m_2 is a ciphertext for message u' such that u and u' are in relation R (R can be thought as included into *pub_inf*).

Pseudorandom primitives

Standard secure pseudorandom primitives are PRG-, PRF- or PRP-based. Such primitives, in principle, can be used for key stream generation, encryption, digital signature or MAC. These are clear examples for unpredictability. For instance, by definition a PRP-secure symmetric key block encryption is computationally indistinguishable from random permutation, which implies that different messages are encrypted, essentially, into statistically independent ciphertexts chosen from the total space of ciphertexts (here "essentially" means that until we use an instance of the encryptor for at most polynomial number of messages the restriction by invertibility requirement is negligible).

B. CI of protocol instances

An instance is the concatenation of its component protocol messages. Let this random variable be denoted by $I(v_1,...,v_m)$ where $v_1,...,v_m$ are the random state variables of the protocol. Shortly we will refer to random variable I as the transcript of the instance. UP, CI, SI type of independence of instance can be considered between protocol messages or parts of them or between transcripts of different instances.

First we recall the natural definition for the statistical independence of protocol instances.

Definition 2 (SI of protocol instances): Protocol instances are statistically independent (SI) if their transcripts are statistically independent.

Property 4: Protocol instances are statistically independent iff their random state variables are statistically independent.

Property 5: SI between pairs of messages of different protocol instances does not imply SI of the instances (in general).

Proof: Consider two instances I_1 and I_2. Let instance I_1 consist of two messages U and V and instance I_1 of a single message W, where U, V, W are binary random variables with equal length. Let U and W be independent, furthermore V=U+W (mod 2). In this example messages of I_1 are (pairwise) independent from the message of I_2 but I_1 and I_2 are dependent.

According to Property 4 protocol instances become statistically dependent if some of their random state variables are statistically dependent. In the set of random state variables we find local coin flipping sequences and variables dependent between different instances. The most characteristic and practically most interesting dependence is caused by subroutines shared by different instances (for example, a common signature module). In this case dependence means that some state variables are equal. The task is that in spite of statistical dependence CI should be maintained between instances.

Definition 3 (CI for protocol instances): Protocol instances are computationally independent if their transcripts are computationally independent.

Property 6: SI of protocol instances implies CI of the instances.
Proof: Straightforward.

Property 7: CI between pairs of protocol messages of different protocol instances does not imply CI of the instances (in general).

Proof: The proof of Property 5 can be repeated. Indeed U and W as well as V(=U+W) and W are (pairwise) CI (as they are (pairwise) SI), however U+V (=W) and W are clearly not CI.

When we want to change from costly SI instances to less costly but statistically dependent ones we have to ensure their CI property. The corresponding base composition rule is the following:

Composition of CI variables: Assume a series of random variables $X(z) = X_1(z_1),...,X_n(z_n)$ are computationally indistinguishable from series $Y=Y_1,...,Y_n$, where $X_i(z_i)$ is an efficient random mapping of random variable z_i, $i=1,...,n$. Assume furthermore that series of random variables $z=z_1,...,z_n$ are computationally indistinguishable from series $w=w_1,...,w_n$. It is not hard to see by standard reduction technique) that $X(w)=X_1(w_1),...,X_n(w_n)$ will also be computationally indistinguishable from Y. If Y is a series of SI instances then X(z) is a series CI instances. Similarly if z is a series of SI instances then w is a series CI instances of random variables. This way by the above rule we embed a series of CI variables into a series if CI variables.

This rule of composition fits naturally to the following scenario. Let X stand for a series of different instances

from a protocol, where $X_i(z_i)$ corresponds to the transcript of the i-th instance. Let z_i and w_i correspond to the output of a cryptographic primitive by the ideal functionality and by realization, respectively. Series Y is the ideal realization with statistically independent states per protocol instance. In such scenario X(w) is the realization of Y with computationally independent realized instances where those instances use computationally independent outputs of the cryptographic primitive.

Composition of CI protocol instances: Composability of instances of protocols means that if we know/prove that a single instance of a protocol (the stand alone case) securely implements the task then the same is true for arbitrary many concurrent instances of the protocol. The main technical tool for proving such statement is the standard hybrid argument which assumes statistically independent instances. This is the way also in the Canetti's UC framework where the composition theorem [4] is proved for the generalized scenario where concurrent composition is carried out at the level of subroutines of a main protocol (in the hybrid protocol model). This is the first step of design and analysis. In the second step we change to less costly realization with computationally only independent instances.

In a typical example while in the ideal case cryptographic keys are independent (fresh) in different instances running between the same set of parties, in real implementation it is typical that the same keys are used in different instances of the same protocol. This way (statistical) dependence arises among such protocol instances induced by the (statistical) dependence of instances of primitives. However by careful selection and use of those primitives the protocol instances may remain computationally independent and the security guaranties remain intact. "Carefulness" means the unpredictability property for the primitives. This way though statistical dependence arises between instances it will not be of help for an adversary looking for harmful interactions between the instances if the implementation guarantees computational independence.

Intuitively, computational independence of the target instance and all other instances is equivalent to the requirement that even with adaptive access to concurrent instances the task of breaking the target instance remains hard for the adversary. An adversary is successful against a protocol realization if it is able to break privacy or correctness guaranties. In case of computationally dependent concurrent instances the adversary tries to exploit this dependence. Our point here is that this intuition is grounded as by definition a set of protocol instances with (mutually) computational independence property is equivalent to an "associated" set with (mutually) statistically independent elements. Here "associated" means replacing statistically dependent state variables with independent ones. In other words the standard hybrid argument can directly be applied to concurrent protocol instances with CI property. This means that composition of statistically dependent but computationally independent instances is a one step action.

C. Dependence models in UC

Three dependence models are defined in UC between concurrent protocol instances which are base UC [4], JUC (Joint-state UC) [5] and GUC (Global-state UC) [7].

In base UC in the real system the target instance has no common state variable with any other instances. In base UC statistical independence of concurrent instances is the result of the so-called "subroutine respecting", the statistical independence of inputs to different instances as well as the statistical independence of local random variables at different instances. Property of "subroutine respecting" means that different instances do not communicate with each other (directly or via a common module). The result is that in this base model different instances cannot have any dependent state variable (at any step of their run). It follows that the target instance is statistically independent (SI) from all other instances in the environment. Computational independence (CI) provides the same independence guaranties under complexity constraint as SI in unconditional case. For example protocol instances can securely be composed parallel in case of CI as in case of SI.

In JUC the target instance and instances from the same protocol share a common variable which implies their statistical dependence [5]. Computational independence of such instances can be ensured under UP assumptions on the applied primitives. Functional independence (FI) is straightforward link to the UC analysis of protocols. Instead of random variables like in case of UP, CI and SI it is defined for functioning protocol instances, where a need for independence guarantee arises both for outputs an also for the series of protocol messages. Accordingly FI is related both to (UC-) secure emulation and also to harmful interactions between concurrently running instances.

In GUC the target instance and any other instances have access to a common globally accessible variable [7]. In this case computational independence is much harder to establish.

D. Functional independence of protocol instances

Protocols can be considered as complex primitives, with special feature that they are executed via the interaction of at least two parties and therefore details of the computation of protocol outputs can also be seen and can potentially be attacked by an adversary. Accordingly both the series of protocol messages and the input/output messages has to be included into a random variable which represents the run of an instance of the protocol. Recall such a random variable corresponds to the information seen by the distinguishing environment at the global interface in the UC system model with dummy adversary [4]. We defined the UP, CI and SI type of independence of instances as the independence of their transcripts where the aim is reduction of the complexity of the analysis to that of a single instance. The second step is

the realization with CI only instances. By the complex notion of functional independence (FI) we want to cover this two step approach by one notion.

The base scenario is the following: consider two sets of concurrent instances $S1=\{I_1,...,I_n\}$, $S2=\{J_1,...,J_n\}$, where in set S_1 the instances are in SI relationship, while in set S_2 they are statistically dependent. The interface between S_1 (S_2) and the distinguisher corresponds to the standard global interface of the UC framework, where the instances of the sets are bunched into one virtual instance $S^{(1)}$ ($S^{(2)}$) the same way as by the multi-session extension in Joint State UC approach (informally, it is a parallel-to-serial converter followed by its inverse, a serial-to-parallel one on the level of input/output packets to/from instances).

Definition 4 (FI for protocol instances): Sets S_1 and S_2 of protocol instances are functionally independent if multi-session extensions $S^{(2)}$ is computationally indistinguishable from extension $S^{(1)}$ by the distinguishing environment at the global interface.

Obviously functional independence is more comprehensive property than CI, in general. Indeed, a protocol containing a protocol message "A→B: Output secret_key" might have been implemented with concurrent CI instances it will not provide FI, when S_1 is a set of ideal instances.

Property 8: Definition 4 of functional independence for sets S_1 and S_2 of instances of a protocol reduces to

a) stand-alone analysis of secure realization if elements of S_1 and S_2 are SI instances of the corresponding ideal protocol and instances of a realization with statistically dependent state variables, respectively,
b) Computational independence if elements of S_1 and S_2 are instances of a realization with statistically independent and dependent state variables, respectively.

Example 3: Consider the special case when only one type of primitives is used by the protocol. In such cases CI between protocol instances of the given protocol can be reduced to the CI between instances of the primitives.

Assume a protocol which uses a single type of cryptographic primitive, a digital signature primitive. Furthermore consider the common state scenario, i.e. different instances of the protocol share a common signature key, therefore they become statistically dependent.

In the first step the elements in S_1 and S_2 are (statistically independent) instances of the corresponding ideal protocol and SI instances of the protocol (via statistically independent signature keys), respectively. This first step is equivalent to the stand alone analysis, which is the test for UC secure emulation of the ideal functionality. As it is well known EU-CMA secure signature primitive is sufficient for reaching base UC security [4] (of course only if the hybrid protocol which is hybrid in ideal signature primitive is secure, e.g. it does not contain "catastrophic" protocol message like "A→B: Output secret_key").

In the second step $S_1'=S_2$ and S_2' contains instances with common signature key. The distinguisher is successful if it can exploit the statistical dependence between digital signatures in different instances within set S_2' . Intuitively, the only way for such success is prediction of digital signature with non-negligible probability. Here we can use once again the EU-CMA property to foil such prediction attack.

Note EU-CMA is the general security requirement in case of instance dependence caused by shared subroutines. Indeed, two security measures have to be taken in case of shared subroutines. First disjoint (sub)spaces of inputs to the common module has to be maintained by different subroutines to avoid trivial adversarial interactions. Secondly successful forging has to be prevented. As we saw in Section 3 UP property induced by EU-CMA implies also CI according to Theorem 1.

V. Conclusions

Unpredictability (UP, Definition 5) guaranties prevent attacks against correctness and privacy carried out via fabrication of valid cryptographic blocks and exploration of private data, respectively. Unpredictability-based standard secure primitives support achievement of computational independence (CI, Definition 1) within and between protocol instances via equivalence result in Theorem 1. Obvious fact that statistical independence (SI) of protocol instances provides the strongest guarantee against interaction attacks. By the equivalence of CI to statistical independence (SI) within an environment of efficient algorithms we arrive to a chain of equivalence relations between independence properties (UP $\approx$ CI $\approx$ SI) which provides the unified fundament for equivalently secure but cost efficient realizations. Computational uniformity of the corresponding distributions is crucial assumption in UP $\approx$ CI claim.

References

[1] M. Backes, I. Cervesato, A. D. Jaggard, A. Scedrov and J. K. Tsay. Cryptographically Sound Security Proofs for Basic And Public-Key Kerberos. *Proc. 11th European Symp. on Research. in Comp. Sec, 2006.*

[2] M. Burmester et.al: Universally Composable RFID Identification and Authentication Protocols, *ACM Transactions on Information and System Security* (TISSEC) TISSEC Homepage archive, Vol 12, Issue 4, Article No. 21, April 2009.

[3] J. Camensisch, S. Krenn and V. Soup. A Framework for Practical Universally Composable Zero-Knowledge Protocols, In: Lee, D.H., Wang, X. (eds.) *Asiacrypt 2011*.

[4] R. Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. *Cryptology ePrint Archive: Report 2000/067.* (received 22 Dec 2000, revised

13 Dec 2005).
[5] R. Canetti and T. Rabin. Universal Composition with Joint State. *Crypto'03, 2003.*
[6] R. Canetti et.al: Universally Composable Password-Based Key Exchange, Advances in Cryptology – *EUROCRYPT 2005*, LNCS Vol. 3494, pp. 404-421, 2005.
[7] R. Canetti, Y. Dodis, R. Pass and S. Walfish. Universally Composable Security with Global Setup. *Cryptology ePrint Archive: Report 2006/432*. 20 Nov 2006
[8] B. Fay, Computational independence, *Cryptology ePrint Archive: Report 2014/1013*, 2014.
[9] S. Gajek, M. Manulis, O. Pereira, A-R. Sadeghi, J. Schwenk. Universally Composable Security Analysis of TLS. *ProvSec 2008*: 313-327.
[10] J. A. Garay, P. MacKenzie, and K. Yang. Strengthening zero-knowledge protocols using signatures. *Journal of Cryptology*, 19(2):169-209, 2006.
[11] J. A. Garay, D. Wichs, H-S. Zhouz. Somewhat Non-Committing Encryption and Efficient Adaptively Secure Oblivious Transfer. *CRYPTO 2009*: 505-523.
[12] F. B. Hamouda et. al: Efficient UC-Secure Authenticated Key-Exchange for Algebraic Languages, *Proceedings of the 16th International Conference on Practice and Theory in Public-Key Cryptography* (PKC '13), 26 February - 1 March 2013, Nara, Japan, Kaoru Kurosawa Ed., Springer-Verlag, 2013.
[13] D. Hofheinz, J. Muller-Quade and R. Steinwandt: Initiator-Resilient Universally Composable Key Exchange, Computer Security – *ESORICS 2003* Volume 2808 of the series Lecture Notes in Computer Science, pp. 61-84, 2003.
[14] H. Jayasree and A. Damodaram: A Novel Fair Anonymous Contract Signing Protocol for E-Commerce Applications. *International Journal of Network Security & Its Applications* (IJNSA), Vol.4, No.5, September 2012.
[15] Y. Lindell. Highly-Efficient Universally-Composable Commitments based on the DDH Assumption. *EUROCRYPT 2011*: 446-466.
[16] I. Vajda. A Universal Composability Framework for Anonymous Communications. *Journal of Computer and Communications Security*. 3, 3, 33–44, 2013.
[17] I. Vajda. Provably Secure On-demand Routing Protocols. *Pioneer Journal of Computer Science and Engineering Technology*, 6, 1–2, 19–39, 2013.
[18] I. Vajda. A Proof Technique for Security Assessment of On-demand Ad Hoc Routing Protocols. *International Journal of Security and Networks*, 9, 1, 12–19. DOI: 10.1504/IJSN.2014.059329, 2013.
[19] I. Vajda. Can Universally Composable Cryptographic Protocols Be Practical? *International Journal of Computer Network and Information Security*, 7, 10, 1-12. DOI: 10.5815/ijcnis.2015.10.03, 2014.
[20] I. Vajda. On the Analysis of Time Aware Protocols in Universally Composable Framework. *International Journal of Information Security*, (online) 14, 4, 1-10. DOI: 10.1007/s10207-015-0300-2, August 2015, (print) 15:403-412, 2016.
[21] A.C.Yao. Theory and applications of trapdoor functions. *Proceedings of the 23rd IEEE Symposium on Foundations of Computer Science*, 1982.

Authors' Profiles

István Vajda graduated from the Telecommunication Department at the Technical University of Budapest. He received the PhD and DSc degrees in 1985 and 1997, respectively. Since 1998, he has been a Professor at the Department of Informatics. He is the co-founder of the Laboratory of Cryptography and Systems Security (CrySyS). During 1990's his research interest was in algebraic code designs for secure multiple access channels. Recently, his research interests are in design and analysis of secure systems, with a special emphasis on provably secure cryptographic primitives and protocols. His application expertise covers secure wireless communication, secure routing and sensor networks.

APPENDIX

Assume an adversary has access to a set S of protocol instances (observations from past or concurrent events) and aims to find a harmful interaction with a target instance T. He is considered successful if he is able attack T in its privacy or correctness properties, which means finding out a priori secret information carried by T or modifying instance T to T' in order to distort the output at honest parties, respectively. The modification typically means changing blocks within protocol messages of instance T where those blocks are composed from outputs of cryptographic primitives, for instance changing a digital signature. In this context the aim of the designer is to guarantee unpredictability between instances in set S and the wished instance T'.

Recall, random variables m_1 and m_2 are *statistically independent* (SI) if $P(m_2=y|m_1=x)=P(m_2=y)$ for all x,y from the corresponding spaces (in shorthand notation $P(m_2|m_1)=P(m_2)$). The intuitive meaning of the above definition is that if in the knowledge of the outcome of random variable m_1 the probability distribution of random variable m_2 is unchanged to its a priori distribution the two random variables are (statistically) independent. We look for an analogous definition when we can rely only on probabilistic polynomial time (PPT) algorithmic resources, in particular the above probability distributions are not available and at most we can access only to polynomial number of samples from them (polynomial in security parameter n).

Definition 5 (unpredictability): Random variable m_2 is unpredictable from random variable m_1 if for any PPT algorithm F and any efficiently computable binary mapping δ there exists a PPT algorithm F' (simulator) such that difference

$$I(F, \delta, m1, m2, n) = |P[F(m_1, pub\_inf, n) = \delta(m_2)]-P[F'(pub\_inf, n) = \delta(m_2)]|$$

is negligible in parameter n, where n is a natural number. Set *pub_inf* contains all public information.

Definition 6 (strong predictability): Random variable m_2 is strongly predictable from random variable m_1 if it is predictable for identity mapping $\delta(x)=x$.

As $P(m_2)=P(m_2|r)$ for an independent random variable r, difference $|P(m_2|\, m_1)-P(m_2|r)|$ is nonzero iff m_1 and m_2 are dependent. Note the analogy to the formulation of Definition 5. In this sense unpredictability by Definition 5 is also a definition of algorithmic independence. In spite of that analogy we will keep the name predictability (and unpredictability) in connection with the above definition as under computational independence we will mean algorithmic indistinguishability from statistical independence (defined in Section 2).

An adversary by standard is a network adversary, however an adversary may also corrupt honest parties. A corruption adversary can access to private data of the party. The adversary may use such extra information for increasing the success probability of attacking the protocol messages of the remaining honest parties. Formally the extra information is included in variable m_1 when predicting variable m_2.

For further motivation for formula (1) we give a few simple examples below:

Example 4: Let $m_2=\text{Rev}(m_1)$, where Rev is the bit order reversion algorithm, which is an obvious instance for the strongest dependence and predictability. Indeed for $F=\text{Rev}$ we get $I(F,\delta, m_1, m_2,n) \sim 1$, for any δ. If m_1 and m_2 are constants, formally they are statistically independent, however they would be in the strongest predictability relationship if we would not include constants into *pub_inf*. Consider also a refined version of this example. Let independent random variables Z_1 and Z_2 be such that they take values from sets $\{x, m_1\}$ and $\{y, m_2\}$, respectively, with probability distribution $(\mu, 1-\mu)$ both, where random variables x and y are not in relation of bit reversion. For bit reversion algorithm F and mapping $\delta(x)=x$, we get $\text{Prob}(F(Z_1, pub\_inf) = \delta(Z_2))=1-\mu$, which for small μ takes high value. However there exists algorithm F' which simply outputs value m_2 (from the known "space" Z_2), which leads to $I(F, \delta, Z_1, Z_2, n) = 0$. So the wanted consistency will not be broken.

Example 5: Consider the example where $m_1=(u, r_1)$, $m_2=(u^c,r_2)$ such that random variable u^c is the binary complement of variable u with length m, furthermore r_1 and r_2 are independent variables with length n. Obviously, m_1 and m_2 are statistically dependent. If we would define unpredictability by requiring the prediction of all bits of m_2 (case of $\delta(x)=x$) the obvious and strong dependence in the prefix between m_1 and m_2 could not be detected by formula (1). Indeed for the performance of best predictor F which complements the first m bits of the input but unavoidably fails predicting the n-bit suffix is $\text{Prob}(F(m_1, pub\_inf) = m_2)=2^{-n}$ which equals to the performance of the corresponding simulator, leading finally to the wanted result I=0.

Note, formulation of Definition 5 has some resemblance also to the standard definition of semantic security of public key encryption. In that definition random variables m_1 and m_2 correspond to a ciphertext and corresponding cleartext, respectively, where mapping δ models the partial information on the clear-text, furthermore randomness is generated by the random selection of the encryption key, the clear-text (by arbitrary distribution over the domain of clear-texts), one-time randomness used in encryption as well as random bits generated by PPT algorithms F and F'. In that respect, Definition 5 can be viewed as extension to this approach, where our main goal is to extend the concept to protocols, where the pair of random variables (m_1,m_2) (m_1,m_2) can be associated to different cryptographic objects not just to encryption, e.g. to protocol messages from concurrent or past instances. Furthermore Definition 5 aims to grasp the concept of independence/dependence, e.g. computationally indistinguishable substitution of statistically independent objects.

Usefulness of strong predictability (Definition 6) is obvious in the following application scenario. Assume an adversary wants to forge a protocol message (m_2) of a protocol instance with the aim to maliciously divert the run of the instance. For this aim the adversary collects a binary string m_1 from concurrent and/or past instances and devices an algorithm F, which for input m_1 outputs a predicate to m_2 (with non-negligible success). Note here complete protocol messages have to be forged (i.e. not only a subset of their bits selected by appropriate mapping δ). For instance, if m_2 contains a substring which is an element from the output domain of a cryptographic primitive, it is obvious that such an element has to be produced fully.

For another example consider the interleaving attack where complete protocol messages (or cryptographic blocks of them) are copy-pasted into the target instance from concurent or past instances. Such straightforward possibility is the simplest way of prediction.

For soundness of Definition 5 consistency must be kept between statistical independence and unpredictability: the former should imply the latter.

Property 9: Definition 5 of unpredictability is consistent with statistical independence: the latter implies the former.

Proof: For short referencing let random variable $\delta(m_2)$ be denoted by m' and let H stand for its probability distribution. Assume that random variables m_1 and m' are (statistically) independent. It follows that no predictor F can perform better than a (potentially unbounded) algorithm which outputs maximum likelihood estimate m" on m', where m"= indmax H(x), i.e.

$\text{Prob}(F(m_1, pub\_inf) = m') \leq H(m'')$ for any F.

Obviously, similar limit stands for the performance of any simulator F'. Now, the proof is obvious if $H(m'')$ is negligibly small. Fortunately, the proof is straightforward even for the general case:

Because of the assumed statistical independence $\text{Prob}(F(m_1, pub\_inf) = m') = \text{Prob}(F(r', pub\_inf) = m')$ for any F where r' is an independent random element. Note, the best simulator is trivial as F' is PPT algorithm and it is able to generate random element r' and run F with input (r', pub_inf). This implies $I(F, \delta, m_1, m_2, n) = 0$.

How to cite this paper: István Vajda,"Computational Independence in the Design of Cryptographic Protocols", International Journal of Computer Network and Information Security(IJCNIS), Vol.8, No.10, pp.1-11, 2016.DOI: 10.5815/ijcnis.2016.10.01

