Per-Session Security: Password-Based Cryptography Revisited

Grégory Demay[1(✉)], Peter Gaži[2], Ueli Maurer[3], and Björn Tackmann[4]

[1] Ergon Informatik AG, Zürich, Switzerland
gregory.demay@ergon.ch
[2] IOHK Research, Vienna, Austria
peter.gazi@iohk.io
[3] Department of Computer Science, ETH Zürich, Zürich, Switzerland
maurer@inf.ethz.ch
[4] IBM Research - Zurich, Rüschlikon, Switzerland
bta@zurich.ibm.com

Abstract. Cryptographic security is usually defined as a guarantee that holds except when a bad event with negligible probability occurs, and nothing is guaranteed in that case. However, in settings where a failure can happen with substantial probability, one needs to provide guarantees even for the bad case. A typical example is where a (possibly weak) password is used instead of a secure cryptographic key to protect a session, the bad event being that the adversary correctly guesses the password. In a situation with multiple such sessions, a *per-session* guarantee is desired: any session for which the password has not been guessed remains secure, independently of whether other sessions have been compromised.

Our contributions are two-fold. First, we provide a new, general technique for stating security guarantees that degrade gracefully and which could not be expressed with existing formalisms. Our method is simple, does not require new security definitions, and can be carried out in any simulation-based security framework (thus providing composability). Second, we apply our approach to revisit the analysis of password-based message authentication and of password-based (symmetric) encryption (PBE), investigating whether they provide strong per-session guarantees.

In the case of PBE, one would intuitively expect a weak form of confidentiality, where a transmitted message only leaks to the adversary once the underlying password is guessed. Indeed, we show that PBE does achieve this weak confidentiality if an upper-bound on the number of adversarial password-guessing queries is known in advance for each session. However, such *local* restrictions appear to be questionable in reality and, quite surprisingly, we show that in a more realistic scenario the desired per-session confidentiality is unachievable.

G. Demay—Full version available at https://eprint.iacr.org/2016/166.

P. Gaži—Work done while author was at ETH Zürich and supported by the Zurich Information Security and Privacy Center.

U. Maurer—Work done while author was at ETH Zürich and UC San Diego, in part supported by SNF fellowship P2EZP2-155566 and NSF grant CNS-1228890.

B. Tackmann—Work done while author was at ETH Zürich and IST Austria, in part supported by the ERC grants 259668-PSPC and 682815-TOCNeT.

S.N. Foley et al. (Eds.): ESORICS 2017, Part I, LNCS 10492, pp. 408–426, 2017.
DOI: 10.1007/978-3-319-66402-6_24

1 Introduction

1.1 Motivation of This Work

Human-memorable passwords represent one of the most widely deployed security mechanisms in practice. They are used to authenticate human users in order to grant them access to various resources such as their computer accounts, encrypted files, web services, and many more. Despite well-known problems associated with this mechanism, its practicality and simplicity from the users' perspective is the main cause of its persisting prevalence. As an example, more than 90% of Google users employ passwords as the only authentication mechanism for accessing their accounts [25]. Acknowledging this situation, it is extremely important that security engineers, including designers of cryptographic protocols, have a precise understanding of the security guarantees that passwords provide for multiple sessions (where one session corresponds to one password; this is often referred to as the *multi-user setting*).

There has been significant effort in formalizing the use of passwords, but the standard provable-security approach in cryptography, focusing on a single session, falls short of modeling the expected guarantees. The main reason for this is that passwords, in contrast to cryptographic keys, can be guessed by the attacker with a probability that can hardly be considered insignificant in the analysis (independently of whether a concrete or asymptotic security approach is being used). This is because they are chosen by the users, and therefore typically do not contain sufficient entropy. When inferring the security guarantees for multiple sessions via the standard hybrid argument, these substantial terms from the analyses of the individual sessions accumulate, and may render the overall statement trivial.

To obtain practically relevant statements about systems that allow for many sessions with passwords, we cannot resign on all security guarantees as soon as any password is guessed. Ideally, one would instead hope that as long as not all passwords were broken, the sessions with passwords that are still safe from the attacker enjoy a non-reduced degree of security. This simple yet important observation has been emphasized before, most notably in the work of Bellare et al. [5] on multi-instance security. At a very high level, their definition aims at ensuring that, in a setting where the security of each single session cannot be guaranteed, the amount of work needed for breaking many sessions cannot be amortized, i.e., it grows (linearly) with the number of sessions considered.

We believe that this approach, while bringing to light a problem of great practical relevance, suffers from certain shortcomings that we illustrate on the example of password-based cryptography. By focusing only on the number of sessions that can be broken, multi-instance security *cannot* capture the intuition that sessions protected by strong passwords should be less vulnerable than sessions protected by weak passwords. Indeed, as the resulting guarantees are in the form of a global upper bound on the number of sessions that can be broken, they do not give any specific guarantee for a session whose password was not guessed, independently of whether other sessions were compromised.

From a broader perspective, a setting with multiple sessions relying on passwords can be seen as an instance of a scenario where the considered resource (e.g., a webmail server) can be gradually weakened by the adversary (e.g., by guessing the passwords in some of the sessions), while it is still expected to provide some security guarantees (e.g., for the other sessions) after such weakening.

1.2 Our Contributions

We develop a technique for modeling resources that are available to parties and used in protocols or applications and can be gradually weakened (we call this "downgrading"). Later, we apply the technique to password-based cryptography in the random oracle model and analyze the security of schemes that use password-derived keys.

DOWNGRADABLE RESOURCES. As our first contribution, we provide a natural and intuitive formalization of settings where a considered resource can be potentially downgraded by the actions of an attacker, but still maintains some security guarantees afterwards. While there are many possible ways to analyze such settings, our formalization allows for the natural decoupling of the descriptions of (1) the resource's behavior at various "levels" of the downgrade; and (2) the mechanism that controls how the system is currently downgraded (as a response to the actions of the attacker). We believe that this modularity allows for simpler analyses of a wide range of resources that can be seen in this way, we discuss the concrete case of password-based cryptography below. The technique is, however, more general, and may also find applications in other scenarios where guarantees may degrade gradually, such as the failure of (some) computational assumptions.

The modeling as proposed is carried out in the constructive cryptography framework [19] and does not require any modifications of its security definitions. We believe that a similar approach would be possible in any simulation-based framework, although in particular an analogy in the universal composability framework [7] would have to overcome certain technical hurdles that stem from the difference between these two frameworks, as we detail in the full version [12].

APPLICATIONS TO PASSWORD-BASED CRYPTOGRAPHY. As our second contribution, we apply this modeling approach to several settings that involve multiple sessions using cryptographic keys derived from hashing passwords in the random oracle model. The potential downgrading that we consider here corresponds to guessing the passwords in some of the sessions.

Idealizing the hash function as a random oracle, a natural expectation for any such setting is that one obtains a *per-session guarantee*, i.e. that as long as the attacker does not guess a password in a particular session, the security guarantees provided in this session remain identical to the case where a perfect key is used (i.e., chosen uniformly at random from a large key space). In particular, the security guarantees of one session are not influenced by other sessions, such as by other users' poor choice of a password.

We show that this intuitive view is not generally correct. Below we explain the reason of this breakdown (which is a variant of the commitment problem

that occurs in adaptive attacks on public-key encryption), and by giving a series of results we draw a map of settings that do/do not succumb to this problem:

1. Password-based MACs. We show that if the password-derived keys are used by a MAC to authenticate insecure channels, a per-session message authentication is achieved.
2. Single-session PBE. For password-based (symmetric) encryption (PBE), obtaining a composable statement (i.e., in a simulation-based framework) is much more delicate even in a single-session case. The reason for this is that, roughly speaking, the simulator in the ideal world is expected to produce a simulated ciphertext upon every encryption and without any knowledge of the actual plaintext. However, if the distinguisher later guesses the underlying password (and hence can derive the encryption key), it can easily decrypt the simulated ciphertext and compare the result to the (known) plaintext. But the simulated ciphertext essentially *committed* the simulator to a message (or a small subset of the message space), so the check will fail with overwhelming probability. Nonetheless, we show that in the single-session setting designing a simulator, while non-trivial, is possible.
3. Multi-session PBE. In line with our motivation, the desired result would be to obtain per-session confidentiality, an analogue of the above single-session statement for the setting with multiple sessions. Surprisingly, as our next contribution, we show that lifting this positive result to the multi-session setting is unachievable. Roughly speaking, any construction of r secure channels from r authenticated channels and the corresponding r password-derived keys will suffer from a simulation problem analogous to the single-session case described above. However, this time we formally prove that it cannot be overcome.
4. Multi-session PBE with local assumptions. To side-step the above impossibility statement, our next result considers the setting of password-based encryption under an additional assumption that the number of adversarial password guesses in each of the sessions is a priori known.
   This assumption seems implausible in general, in fact we show that it cannot be achieved by the salting technique often used in the context of password hashing; instead, as we also show, salting (only) guarantees a global upper bound. (Yet, there may be specific settings in which the validity of the per-session bounds can be argued.) We show, however, that the assumption of local bounds *is* sufficient to overcome the commitment problem and prove that the intuitively expected guarantees described above are indeed achieved. We stress, however, that the simulator constructed in the proof depends on the password distribution.
5. PBE scheme from PKCS #5. Finally, we observe that the arguments underlying the above impossibility result in item 3 can also be applied to the password-based encryption as standardized in PKCS #5 [15].

Composability. Overall, our results yield a characterization of when password-derived keys can be used in a composable simulation-based security framework for the task of secure communication. Our aim for strong, composable security

guarantees is motivated by the particular relevance of password-based cryptography in the Internet, where various cryptographic schemes are used concurrently and as building blocks of larger protocols. To the best of our knowledge, this work represents the first composable treatment of (non-interactive) password-based encryption and message authentication.

1.3 Related Work

Beyond the work on multi-instance security by Bellare *et al.* [5] that was discussed in the introduction above, there are large amounts of literature on passwords. On the empirical side, the weaknesses of passwords in practice were studied e.g. in [23]. We attempt to focus on the literature most relevant to our work.

For password-derived keys, most provable-security works focused on the single-session setting, analyzing ways to augment the key-derivation process to slow down offline brute-force password-guessing attacks. Techniques to achieve this include salting (which was introduced in a scenario with multiple users but without a provable-security analysis) [15], iteration [11,21], and hashing with moderately hard-to-compute functions [2,9,24]. However, the security analyses of those works have a different aim from ours as none of them considers the multi-session scenario. A notable, already mentioned exception is [5] which studied key derivation functions proposed in PKCS #5 [15] and did focus on security in a setting with multiple users.

A key-recovery security definition for password-based encryption was given in [1], but here also only single-session security was considered.

Finally, a separate line of work aims at realizing password-authenticated key exchange (PAKE) protocols [4,8,13,16] that prevent the possibility of offline password-guessing attacks and result in keys that can then safely be used for encryption or authentication. While some of these results are obtained in a composable, simulation-based framework and hence extend naturally to the multi-session case, the protocols are intrinsically interactive and cannot be used in non-interactive password-based settings such as ours.

2 Preliminaries

We denote sets by calligraphic letters or capital Greek letters (e.g., $\mathcal{X}$, Σ). A discrete random variable is denoted by an upper-case letter X, its range by the corresponding calligraphic letter $\mathcal{X}$, and a realization of the random variable X is denoted by the corresponding lower-case letter x. Unless stated otherwise, $X \xleftarrow{\$} \mathcal{X}$ denotes a random variable X selected independently and uniformly at random from $\mathcal{X}$. A tuple of r integers $(q_1, \ldots, q_r)$ will be denoted by a bold letter $\boldsymbol{q}$. The set of bit strings of finite length is denoted $\{0,1\}^*$ and $x \| y$ denotes the concatenation of two bit strings x and y. The empty bit string is denoted $\diamond$, while $\blacklozenge$ is used as an error symbol.

Discrete systems. Many cryptographic primitives (e.g. block ciphers, MAC schemes, random functions) can be described as $(\mathcal{X}, \mathcal{Y})$-*random systems* [18]

taking inputs $X_1, X_2, \ldots \in \mathcal{X}$ and generating for each input X_k an output $Y_k \in \mathcal{Y}$. In full generality, such an output Y_k depends probabilistically on all the previous inputs $X_1, \ldots, X_k$ as well as all the previous outputs $Y_1, \ldots, Y_{k-1}$.

Resources and converters. The security definitions in this work are stated in terms of the resources available to parties. The resources in this work are discrete systems with three interfaces, which we naturally label by elements of the set $\{\mathsf{A}, \mathsf{B}, \mathsf{E}\}$, for Alice's, Bob's and Eve's interface, respectively. We generally use upper-case bold-face letters, such as $\mathbf{R}$ or $\mathbf{S}$ for generic resources, and upper-case sans-serif letters for more specific resources, such as KEY for a shared secret key resource or AUT for an authenticated channel resource.

A protocol machine employed locally by a party is modeled by a so-called *converter*. Attaching a converter α at the i-interface of a resource, where $i \in \{\mathsf{A}, \mathsf{B}, \mathsf{E}\}$, models that party i uses α to access this resource. A protocol then corresponds to a pair of converters, one for each honest party. Converters are denoted by lower-case Greek letters (e.g., α, σ) or by sans-serif fonts (e.g., $\mathsf{enc}, \mathsf{dec}$). The set of all converters is denoted by Σ. Attaching a converter α to the i-interface of a resource $\mathbf{R}$ is denoted by $\alpha^i \mathbf{R}$. Any two resources $\mathbf{R}$ and $\mathbf{S}$ can composed in parallel, denoted by $[\mathbf{R}, \mathbf{S}]$. For each $i \in \{\mathsf{A}, \mathsf{B}, \mathsf{E}\}$, the i-interface of $\mathbf{R}$ and $\mathbf{S}$ are merged and can be accessed through the i-interface of $[\mathbf{R}, \mathbf{S}]$.

The construction notion. We formalize the security of protocols by the following notion of construction, as introduced by Maurer and Renner [19,20]. To be considered secure, a protocol must satisfy two requirements. First, the protocol must construct the desired resource in a setting where no attacker is present. This condition is referred to as the *availability* or correctness condition and excludes trivial protocols. Second, the protocol must also construct the desired resource when the adversary is present, which we refer to as the *security* condition. This condition requires that everything the adversary can achieve in the real world he can also accomplish in the ideal world. To state these two conditions, we consider pairs of resources $(\mathbf{R}, \mathbf{R}_\perp)$, where $\mathbf{R}_\perp$ stands for the resource $\mathbf{R}$ when no adversary is present.

Definition 1. *Let ε_1 and ε_2 be two functions mapping each distinguisher $\mathbf{D}$ to a real number in $[0,1]$. A two-party protocol $\pi := (\alpha, \beta) \in \Sigma^2$ constructs a pair of resources $(\mathbf{S}, \mathbf{S}_\perp)$ from an assumed pair of resources $(\mathbf{R}, \mathbf{R}_\perp)$ relative to simulator $\sigma \in \Sigma$ and within $\varepsilon := (\varepsilon_1, \varepsilon_2)$, denoted $(\mathbf{R}, \mathbf{R}_\perp) \overset{(\pi, \sigma, \varepsilon)}{\longmapsto} (\mathbf{S}, \mathbf{S}_\perp)$, if*

$$\begin{cases} \Delta^{\mathbf{D}}\left(\alpha^{\mathsf{A}}\beta^{\mathsf{B}}\mathbf{R}_\perp, \quad \mathbf{S}_\perp\right) & \leq & \varepsilon_1(\mathbf{D}) & (\textit{availability}) \\ \Delta^{\mathbf{D}}\left(\alpha^{\mathsf{A}}\beta^{\mathsf{B}}\mathbf{R} \ , \ \sigma^{\mathsf{E}}\mathbf{S} \ \right) & \leq & \varepsilon_2(\mathbf{D}) & (\textit{security}), \end{cases}$$

for all distinguishers $\mathbf{D}$, where $\Delta^{\mathbf{D}}(\mathbf{U}, \mathbf{V}) := \left|\mathsf{P}^{\mathbf{DU}}(B=1) - \mathsf{P}^{\mathbf{DV}}(B=1)\right|$ denotes the advantage of $\mathbf{D}$ in distinguishing between $\mathbf{U}$ and $\mathbf{V}$.

An important property of Definition 1 is its composability. Intuitively, if a resource $\mathbf{S}$ is used in the construction of a larger system, then the composability implies that $\mathbf{S}$ can be replaced by $\alpha^{\mathsf{A}}\beta^{\mathsf{B}}\mathbf{R}$ without affecting the security of the composed system. More details can be found in [19,26]. All the constructions stated in this paper are such that the availability condition is trivially satisfied and we therefore omit it from now onwards. That is, we write $\mathbf{R}$ for $(\mathbf{R}, \mathbf{R}_{\perp})$.

Message authentication. A message authentication code (MAC) scheme with message space $\mathcal{M} \subseteq \{0,1\}^*$, key space $\mathcal{K} := \{0,1\}^n$, and tag space $\mathcal{U} \subseteq \{0,1\}^*$ is defined as a pair (tag, vrf), where tag is a (possibly probabilistic) function taking as input a key $k \in \mathcal{K}$ and a message $m \in \mathcal{M}$ to produce a tag $u \leftarrow tag\,(k, m)$, and vrf is a *deterministic* function taking as input a key $k \in \mathcal{K}$, a message $m \in \mathcal{M}$ and a tag $u \in \mathcal{U}$ to output a bit $b := vrf\,(k, m, u)$ asserting the validity of the input tag u. A MAC scheme is *correct* if $vrf\,(k, m, tag\,(k, m)) = 1$, for all keys $k \in \mathcal{K}$ and all messages $m \in \mathcal{M}$.

Symmetric encryption. A symmetric encryption scheme with message space $\mathcal{M} \subseteq \{0,1\}^*$, key space $\mathcal{K} := \{0,1\}^n$, and ciphertext space $\mathcal{C} \subseteq \{0,1\}^*$ is defined as a pair (enc, dec), where enc is a (possibly probabilistic) function taking as input a key $k \in \mathcal{K}$ and a message $m \in \mathcal{M}$ to produce a ciphertext $c \leftarrow enc\,(k, m)$, and dec is a *deterministic* function taking as input a key $k \in \mathcal{K}$ and a ciphertext $c \in \mathcal{C}$ to output a plaintext $m' := dec\,(k, c)$. The output of dec can also be the error symbol $\blacklozenge$ to indicate an invalid ciphertext. An encryption scheme is *correct* if $dec\,(k, enc\,(k, m)) = m$, for all keys $k \in \mathcal{K}$ and all messages $m \in \mathcal{M}$.

3 Transformable Systems

In this section, we present our approach to modeling systems that can be gradually transformed, in a way that clearly separates the *effects* of the transformation from *how it can be provoked*.

As a warm-up example, consider a key obtained by hashing a secret password shared between two users Alice and Bob. Idealizing the hash function as a random oracle, the resulting key is completely random from the perspective of any third party Eve unless she also queried the random oracle on the same input; in other words, unless she correctly guessed the password. If we model the key obtained by this process as a resource, we consider two separate parts of it. The first one specifies the behavior of the resource before and after the transformation (a "strong" version gives the key only to Alice and Bob, a "weak" version also gives it to Eve); the second part triggers one of these two versions based on Eve's actions (providing a password-guessing game for her, triggering the weaker version as soon as she wins).

In general, a transformable system is therefore the combination of two random systems: a *core* and a *trigger* system. The core system specifies how it behaves as an internal switch value changes, while the trigger system specifies how this switch value can be changed. More formally, a core system $\mathbf{S}$ is simply an $(\mathcal{X} \cup \mathcal{S}, \mathcal{Y})$-random system, where the set of inputs is partitioned into two

sets $\mathcal{X}$ and $\mathcal{S}$ with $\mathcal{X} \cap \mathcal{S} = \emptyset$. The set $\mathcal{X}$ is the set of "normal" inputs, while $\mathcal{S}$ is the set of possible switch values. A trigger system $\mathbf{T}$ is a $(\mathcal{T}, \mathcal{S})$-random system which outputs a switch value. Elements of $\mathcal{T}$ are called trigger values and correspond to password guesses in our example above.

Definition 2. *Let $\mathcal{X}, \mathcal{Y}, \mathcal{S}$ and $\mathcal{T}$ be four discrete sets such that $\mathcal{X} \cap \mathcal{S} = \emptyset$ and $\mathcal{X} \cap \mathcal{T} = \emptyset$. An $(\mathcal{X} \cup \mathcal{S}, \mathcal{Y})$-random system $\mathbf{S}$ and a $(\mathcal{T}, \mathcal{S})$-random system $\mathbf{T}$ form an $(\mathcal{X} \cup \mathcal{T}, \mathcal{Y})$-random system, denoted $\mathbf{S_T}$, defined as follows. On input $x \in \mathcal{X}$, the system $\mathbf{S_T}$ outputs $y \in \mathcal{Y}$, where y is the output of the system $\mathbf{S}$ when queried on the input x. On input $t \in \mathcal{T}$, the system $\mathbf{S_T}$ outputs $y' \in \mathcal{Y}$, where y' is the output of $\mathbf{S}$ when queried on the output $s \in \mathcal{S}$ of the system $\mathbf{T}$ which was queried on the original input t (see Fig. 1).*
The random system $\mathbf{S_T}$ will be referred to as a transformable *system, the random system $\mathbf{S}$ as a* core *system, and the random system $\mathbf{T}$ as a* trigger *system.*

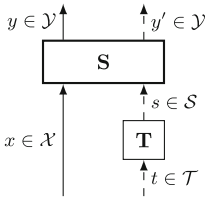

Fig. 1. A transformable system $\mathbf{S_T}$ formed by combining a core system $\mathbf{S}$ with a trigger system $\mathbf{T}$. "Normal" inputs $x \in \mathcal{X}$ are processed directly by $\mathbf{S}$, while trigger values $t \in \mathcal{T}$ go instead first through the system $\mathbf{T}$ whose output $s \in \mathcal{S}$ is then used as an input to the system $\mathbf{S}$.

Fixed Switches. Given an $(\mathcal{X} \cup \mathcal{S}, \mathcal{Y})$-core system $\mathbf{S}$, it will be sometimes convenient to argue about the behavior of $\mathbf{S}$ for a particular fixed switch value $s \in \mathcal{S}$. To do so, we denote by $\mathbf{S}_s$ the $(\mathcal{X}, \mathcal{Y})$-random system obtained by initializing $\mathbf{S}$ as follows: the switch value s is initially input to $\mathbf{S}$ and its resulting output is discarded. In other words, $\mathbf{S}_s$ corresponds to the system $\mathbf{S}$ where the value of its switch is fixed from the beginning to s and cannot be changed. In particular, the input space of $\mathbf{S}_s$ is only $\mathcal{X}$ and not $\mathcal{X} \cup \mathcal{S}$. Given a random variable S over $\mathcal{S}$, we denote by $\mathbf{S}_S$ the system selected at random in $\{\mathbf{S}_s \mid s \in \mathcal{S}\}$ according to S.

downgradable Keys and downgradable Secure Channels. The core systems that we will consider will actually be resources, i.e., random systems with 3 interfaces A, B and E for Alice, Bob, and Eve, respectively, where the switch values are controlled via the interface E. Formally, we model this interface as being split into two sub-interfaces: $\mathsf{E_N}$ (for "normal" inputs/outputs) and $\mathsf{E_S}$ (for switch values). Typically, Eve will not have a direct access to the interface $\mathsf{E_S}$ of the core resource, instead she will only be allowed to access a trigger system $\mathbf{T}$, which itself produces the switch values. Neither Alice nor Bob have access to $\mathbf{T}$. Such a core resource combined with a trigger system will be called a *downgradable* resource.

Alg. 1. Core resource KEY^r

$s_j := 0$ and $k_j \xleftarrow{\$} \{0,1\}^n$, for all $j \in \{1,\ldots,r\}$
on input (j, getkey) at $i \in \{\mathsf{A},\mathsf{B}\}$
  output (j, k_j) at i
on input $s \in \{0,1\}^r$ at $\mathsf{E_S}$
  $(s_1,\ldots,s_r) := s$
on input (j, getkey) at $\mathsf{E_N}$
  if $s_j = 0$ **then output** $(j, \blacklozenge)$ at $\mathsf{E_N}$
  else output (j, k_j) at $\mathsf{E_N}$

Alg. 2. Core resource SEC^r

$s_j := 0$ and $m_j := \diamond$, for all $j \in \{1,\ldots,r\}$
on first input (j, m) at A
  $m_j := m$
  output (j, m_j) at B
  output $(j, |m_j|)$ at $\mathsf{E_N}$
on input $s \in \{0,1\}^r$ at $\mathsf{E_S}$
  $(s_1,\ldots,s_r) := s$
on input (j, getmsg) at $\mathsf{E_N}$
  if $s_j = 0$ **then output** $(j, \blacklozenge)$ at $\mathsf{E_N}$
  else output (j, m_j) at $\mathsf{E_N}$

We now introduce *downgradable key resources* and *downgradable secure channels*, examples of such resources that will be used throughout the paper. These resources are parameterized (among other) by a fixed number r of sessions. Intuitively, these resources provide a graceful deterioration of security by associating each session with a password and guaranteeing that a session remains secure as long as its password is not guessed, irrespectively of the state of other sessions. We first describe the corresponding core resources and then the trigger systems.

Example 1 (Key). The *core* resource KEY^r for r sessions takes as switch at interface $\mathsf{E_S}$ an r-bit string $(s_1,\ldots,s_r)$ which specifies for each session whether it is "broken" ($s_j = 1$) or not ($s_j = 0$). Alice and Bob can retrieve a uniform and independent key for a given session, while Eve can only retrieve it if the session is marked as "broken". The resource KEY^r is formalized in Algorithm 1.[1]

Example 2 (Secure Channel). The *core* resource SEC^r for r sessions also takes as switch value at interface $\mathsf{E_S}$ an r-bit string which specifies for each session whether or not confidentiality is "broken". The resource SEC^r allows Alice to send one message per session to Bob. Eve learns nothing about the transmitted message but its length, unless this session was marked as "broken", in which case the message is leaked to her. The channel SEC^r does not allow Eve to inject any message, regardless of the value of the switch, and is formalized in Algorithm 2.

Example 3 (Local and Global Password-Guessing Triggers). Eve will not be allowed to influence the switch values of KEY^r or SEC^r directly, instead she will have to interact with a trigger system which captures the guessing of per-session passwords. We consider two different such trigger systems, in both of them the number of guesses allowed to Eve is restricted. These two systems differ in whether the restriction on the number of guesses is local to each session or global over all r sessions. We refer to them as *local and global (password-guessing) triggers* and denote them by LT and GT, respectively.

Formally, both triggers are parameterized by a password distribution $\mathcal{P}$ over $\mathcal{W}^r$ (where $\mathcal{W} \subseteq \{0,1\}^*$ is a set of passwords) and the number of password

[1] Each session corresponds to a single use of a password. The re-use of passwords is modeled by password distributions that output multiple copies of the same password.

guesses allowed, either locally for each of the sessions (a tuple $\boldsymbol{q} := (q_1, \ldots, q_r)$) or globally (a parameter q). Both $\mathsf{LT}(\mathcal{P}, \boldsymbol{q})$ and $\mathsf{GT}(\mathcal{P}, q)$ initially sample r passwords $(w_1, \ldots, w_r)$ according to $\mathcal{P}$. When a password guess (j, w) for the j^{th} session is received, both triggers change the state of this session to "broken" if the password guess is correct and their respective constraint on the number of password-guessing queries is satisfied. Both triggers $\mathsf{LT}(\mathcal{P}, \boldsymbol{q})$ and $\mathsf{GT}(\mathcal{P}, q)$ are only accessible by Eve and are detailed in Algorithms 3 and 4.

Alg. 3. Local trigger $\mathsf{LT}(\mathcal{P}, \boldsymbol{q})$

$(w_1, \ldots, w_r) \leftarrow \mathcal{P}$
$s_j := 0$ and $\ell_j := 0$, for all $j \in \{1, \ldots, r\}$
on input (j, w) at $\mathsf{E_S}$
  $\ell_j := \ell_j + 1$
  $s_j := s_j \vee ((w = w_j) \wedge (\ell_j \leq q_j))$
  output $(s_1, \ldots, s_r)$ at $\mathsf{E_S}$

Alg. 4. Global trigger $\mathsf{GT}(\mathcal{P}, q)$

$(w_1, \ldots, w_r) \leftarrow \mathcal{P}$
$s_j := 0$ for all $j \in \{1, \ldots, r\}$
$\ell := 0$
on input (j, w) at $\mathsf{E_S}$
  $\ell := \ell + 1$
  $s_j := s_j \vee ((w = w_j) \wedge (\ell \leq q))$
  output $(s_1, \ldots, s_r)$ at $\mathsf{E_S}$

Combining the core systems and triggers given above via Definition 2 leads to four downgradable resources: two with local restrictions, $\mathsf{KEY}^r_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})}$ and $\mathsf{SEC}^r_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})}$, where the number of password-guessing queries is restricted per session; and two with a global restriction, $\mathsf{KEY}^r_{\mathsf{GT}(\mathcal{P},q)}$ and $\mathsf{SEC}^r_{\mathsf{GT}(\mathcal{P},q)}$, where only the total number of password-guessing queries is limited. To simplify the notation, we will often drop the parameters $\mathcal{P}$, q, $\boldsymbol{q}$ when clear from the context. The results presented in the next sections hold for *any* distribution $\mathcal{P}$ of r passwords, including correlated distributions.

4 Password-Based Key Derivation

The simple protocol for deriving a key from a password via hashing as considered in Sect. 3 can be proven to construct, from a pre-distributed password and a random-oracle resources in each session, a downgradable key resource. Multiple independent random oracles can be constructed from a single one via *salting* (i.e., domain separation), a point that we will discuss in Sect. 6.4.

More formally, we model the shared passwords as an explicit resource denoted PW. It is parameterized by a joint distribution $\mathcal{P}$ of r passwords. The resource $\mathsf{PW}(\mathcal{P})$ first samples from the distribution $\mathcal{P}$ to obtain r passwords $(w_1, \ldots, w_r)$ and then outputs (j, w_j) at interface $i \in \{\mathsf{A}, \mathsf{B}\}$ whenever it receives as input (j, getpwd) at the same interface i. Note that Eve does not learn anything about the sampled passwords except for the a priori known distribution $\mathcal{P}$.

Each hash function is modeled as a random oracle available to all parties, denoted by RO. Notably, we model the restriction on Eve's computational power by a restriction on the number of invocations of the random oracles that she is allowed to do. (For a rationale behind this choice and how it allows to model complexity amplification via iteration, see [11].) We consider either a tuple of random oracles with local restrictions denoted $[\mathsf{RO}_{q_1}, \ldots, \mathsf{RO}_{q_r}]$, where each random

oracle has its own upper bound q_j on the number of adversarial queries it allows; or a tuple of random oracles with one global restriction denoted $[\mathsf{RO}, \ldots, \mathsf{RO}]_q$, where at most q adversarial queries are allowed in total.

The key-derivation protocol $\mathsf{KD} := (\mathsf{kd}, \mathsf{kd})$ consists of both parties applying a converter kd. Upon a key request (j, getkey) for the j^{th} session, kd queries $\mathsf{PW}(\mathcal{P})$ to retrieve the shared password w_j for this session, then queries the j^{th} random oracle on w_j and returns its output. The following simple lemma proved in the full version shows that the protocol KD constructs downgradable keys.

Lemma 1 *For the key derivation protocol* $\mathsf{KD} := (\mathsf{kd}, \mathsf{kd})$ *described above, there exists a simulator* σ_{kd} *such that for all distributions* $\mathcal{P}$ *of* r *passwords, for all integers* $\boldsymbol{q} := (q_1, \ldots, q_r)$ *and* q*, we have*

$$[[\mathsf{RO}_{q_1}, \ldots, \mathsf{RO}_{q_r}], \mathsf{PW}(\mathcal{P})] \xrightarrow{(\mathsf{KD}, \sigma_{\mathsf{kd}}, 0)} \mathsf{KEY}^r_{\mathsf{LT}(\mathcal{P}, \boldsymbol{q})} \quad \textit{and}$$

$$\left[[\mathsf{RO}, \ldots, \mathsf{RO}]_q, \mathsf{PW}(\mathcal{P})\right] \xrightarrow{(\mathsf{KD}, \sigma_{\mathsf{kd}}, 0)} \mathsf{KEY}^r_{\mathsf{GT}(\mathcal{P}, q)} \ .$$

This lemma is very similar to [5, Theorem 3.3], although the results are technically slightly different. While [5, Theorem 3.3] is stricter in terms of the information given to the distinguisher (which obtains the passwords in clear), our statement comes with an explicit composition guarantee.

5 Password-Based Message Authentication

We investigate the use of password-derived keys for message authentication using MACs. We prove that such a construction meets the intuitive expectation that in a multi-user setting, as long as a password for a particular session is not guessed, the security (in this case: authenticity) in that session is maintained at the same level as if a perfectly random key was used. We present these results partly to put them in contrast with those on password-based encryption, where the situation is more intricate. As a consequence, in this section we deliberately remain slightly informal and postpone the full formal treatment to the full version [12].

Assumed resources. The construction statement shown below assumes the availability of a password-derived key and an insecure communication channel for each of the r considered sessions. For password-derived keys, we simply use the downgradable resource $\mathsf{KEY}^r_{\mathbf{T}}$ which can be constructed e.g. via one of the statements in Lemma 1 (here $\mathbf{T}$ stands for either LT or GT). The insecure channels are formalized as the resource INSEC^r which forwards any message sent by Alice to Eve, while any message injected by Eve is forwarded to Bob.

MAC schemes as protocols. A MAC scheme is used by Alice and Bob in the natural way (we denote their converters tag and vrf, respectively). When tag receives as input a message m for the j-th session, it retrieves the key k_j associated to this session from the resource $\mathsf{KEY}^r_{\mathbf{T}}$, computes the tag u according to the MAC scheme and outputs to the insecure channel INSEC^r in the j-th

session the message $m\|u$. On the other end of the channel, whenever vrf receives a message and a tag $m'\|u'$ for the j'-th session, it first retrieves the key $k_{j'}$ from $\mathsf{KEY}^r_{\mathbf{T}}$, verifies the tag and outputs m' only if the verification succeeds.

Constructed resource. The channel that Alice and Bob obtain by using the protocol $(\mathsf{tag}, \mathsf{vrf})$ guarantees that any message that Bob receives for a particular session must have been sent before by Alice, unless this session was "broken." This *(core) unordered authenticated channel*, denoted UAUT^r takes an r-bit string $(s_1, \ldots, s_r)$ as a switch value, specifying for each session j whether it is broken ($s_j = 1$), in which case Eve can send any message to Bob for this particular session, or not ($s_j = 0$), in which case the messages that Eve can send to Bob for session j are limited to those that Alice already sent. The channel UAUT^r does not offer any secrecy: messages input by Alice are directly forwarded to Eve. The channel UAUT^r only prevents Eve from to injecting a *fresh* message, it does not prevent the injection of a legitimate message multiple times, the reordering of legitimate messages, or the loss of some messages.

If the MAC scheme used by the protocol $(\mathsf{tag}, \mathsf{vrf})$ is weakly unforgeable, then it constructs the downgradable unordered authenticated channel $\mathsf{UAUT}^r_{\mathbf{T}}$ by using the downgradable key $\mathsf{KEY}^r_{\mathbf{T}}$ and the insecure channel INSEC^r. The formal statement together with its proof are in the full version [12].

Theorem (Informal). *There exists a simulator σ_{MAC} such that for every distribution $\mathcal{P}$ of r passwords, every number of queries $\boldsymbol{q} := (q_1, \ldots, q_r)$ and q, and any trigger $\mathbf{T} \in \{\mathsf{LT}(\mathcal{P}, \boldsymbol{q}), \mathsf{GT}(\mathcal{P}, q)\}$,*

$$[\mathsf{KEY}^r_{\mathbf{T}}, \mathsf{INSEC}^r] \xrightarrow{((\mathsf{tag}, \mathsf{vrf}), \sigma_{\mathsf{MAC}}, \varepsilon)} \mathsf{UAUT}^r_{\mathbf{T}},$$

where the distinguishing advantage ε can be reduced to the weak unforgeability of the underlying MAC scheme.

6 Password-Based Encryption

We investigate the use of password-derived keys for symmetric encryption. In a multi-session setting, one may expect that as long as a password for a particular session is not guessed, the confidentiality in that session is maintained. This would, roughly speaking, correspond to a construction of (downgradable) secure channels from authenticated channels and password-derived keys.

Assumed resources. We assume the availability of a password-derived key and an authenticated communication channel for each of the r sessions. For the keys, we use the downgradable resource $\mathsf{KEY}^r_{\mathbf{T}}$, where $\mathbf{T}$ typically stands for either $\mathsf{LT}(\mathcal{P}, \boldsymbol{q})$ or $\mathsf{GT}(\mathcal{P}, q)$. We also assume an authenticated channel AUT^r described in Algorithm 5. The channel AUT^r takes in each session a message c at Alice's interface A, and outputs it at both Eve's interface E and Bob's interface B.

Alg. 5. Channel AUT^r

on first input (j, c) at A
  output (j, c) at B
  output (j, c) at E

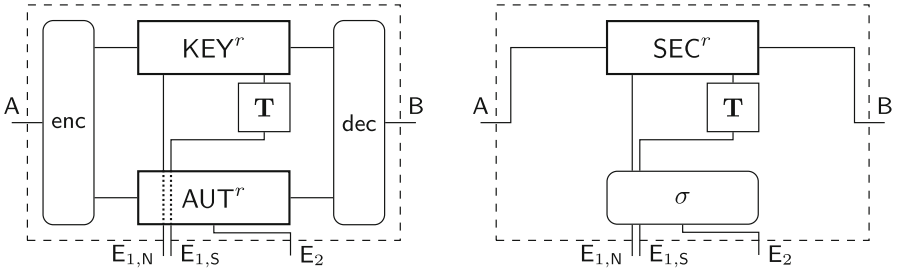

Fig. 2. Left: The assumed resource, a downgradable key $\mathsf{KEY}^r_{\mathbf{T}}$ and an authenticated channel AUT^r, with protocol converters enc and dec attached to interfaces A and B, denoted $\mathsf{enc}^{\mathsf{A}}\mathsf{dec}^{\mathsf{B}}[\mathsf{KEY}^r_{\mathbf{T}}, \mathsf{AUT}^r]$. Right: The desired downgradable secure channel $\mathsf{SEC}^r_{\mathbf{T}}$ with simulator σ attached to interface E, denoted $\sigma^{\mathsf{E}}\mathsf{SEC}^r_{\mathbf{T}}$. The simulator σ must emulate Eve's interface in the left picture, i.e., key retrieval queries at $\mathsf{E}_{1,\mathsf{N}}$, trigger queries at $\mathsf{E}_{1,\mathsf{S}}$ and the authenticated channel at E_2.

Using the authenticated channel $\mathsf{UAUT}^r_{\mathbf{T}}$ as constructed in Sect. 5 is also possible, but requires to encompass a mechanism to decide when a message is delivered to Bob based on Eve's actions (similarly to $\mathsf{UAUT}^r_{\mathbf{T}}$).

Encryption schemes as protocols. Given an encryption scheme (enc, dec), the encryption protocol (formalized by converters enc and dec, respectively) proceeds similarly to the message authentication protocol in Sect. 5. For each transmitted message, both enc and dec obtain the key from $\mathsf{KEY}^r_{\mathbf{T}}$, and the ciphertexts are transmitted over the channel AUT^r. Throughout this section, we will assume the encryption scheme (enc, dec) to be *correct*.

Constructed resource. The channel that Alice and Bob wish to obtain by using the protocol $\mathsf{SE} := (\mathsf{enc}, \mathsf{dec})$ is the downgradable resource $\mathsf{SEC}^r_{\mathbf{T}}$ described in Sect. 3, which guarantees that any message sent by Alice for a particular session is transmitted confidentially to Bob, unless this session was "broken".

6.1 PBE for a Single Session

We start by focusing on PBE with a single session, where we are interested in the possibility of constructing the downgradable secure channel[2] $\mathsf{SEC}_{\mathsf{LT}(\mathcal{P},q)}$ from a downgradable key $\mathsf{KEY}_{\mathsf{LT}(\mathcal{P},q)}$ and an authenticated channel AUT using the protocol $\mathsf{SE} = (\mathsf{enc}, \mathsf{dec})$. According to Definition 1 we must thus find a simulator σ that makes the systems according to Fig. 2 indistinguishable.

The commitment problem. In the real world, whenever a message m is input at Alice's interface A, the corresponding ciphertext is output at Eve's interface E_2. On the other hand, in the ideal world only the length $|m|$ of the transmitted message m is output by the channel $\mathsf{SEC}_{\mathsf{LT}(\mathcal{P},q)}$ to the simulator σ.

[2] In the particular case of a single session, the local password-guessing trigger $\mathsf{LT}(\mathcal{P}, q)$ and the global one $\mathsf{GT}(\mathcal{P}, q)$ are identical, for any $\mathcal{P}, q$.

The simulator must therefore emulate that a ciphertext was sent by only knowing the length $|m|$ of the transmitted message and not the message m itself.

A naïve simulation strategy could initially select a key k uniformly at random and emulate the transmission of a ciphertext by encrypting a fresh random message v of the correct length under key k, while password-guessing queries are simply forwarded to the trigger $\mathsf{LT}(\mathcal{P}, q)$ of the downgradable channel $\mathsf{SEC}_{\mathsf{LT}(\mathcal{P},q)}$.

This approach fails when the password is guessed and the session is broken. In the real world, the distinguisher can retrieve the key k used for encryption and check that a previously seen ciphertext c is indeed an encryption of the transmitted message m. In contrast, in the ideal world the simulator σ can retrieve the transmitted message m, but note that it cannot output the key k that it chose at the beginning to simulate encryption since $dec(k, c) = v$ is a random message which (with overwhelming probability) is different from the actual transmitted message m. The simulator σ must therefore "decommit" by finding a key k' such that the decryption of the simulated ciphertext c under that key k' yields the transmitted plaintext m, i.e., $dec(k', c) = m$. However, it is not hard to see that unless the key space of the encryption scheme contains as many keys as there are messages (which is only true for impractical schemes such as the one-time pad), it is highly unlikely that such a key even exists and the simulation therefore fails.

Brute-force to the rescue. The previous paragraph only shows that one particular simulation strategy fails. The source of the commitment problem is that the simulator σ only breaks the session *after* having output the simulated ciphertext. The key insight is that this does not have to be the case: consider a simulator σ_{LT} which attempts to break the session *before* having to output any ciphertext. Instead of faithfully forwarding the q password-guessing queries, the simulator σ_{LT} initially exhausts all of the allowed q queries to optimally brute-force the session by querying the q most likely passwords. If the brute-force step fails, σ_{LT} encrypts a random message of the correct length and declares any password guess as incorrect. If the brute-force step succeeds, σ_{LT} has access to the transmitted message and can therefore perfectly simulate the corresponding ciphertext, while password-guessing queries can easily be responded appropriately.

In this setting with a single session, password-based encryption is therefore possible with respect to the simulation strategy σ_{LT} sketched above. The generalization of the above statement for multiple r sessions is discussed in Sect. 6.3. The below corollary then follows by taking $r = 1$ in the result of Sect. 6.3.

Corollary (Informal). *For every distribution $\mathcal{P}$ of a single password and every integer q, there exists a simulator σ_{LT} such that*

$$\left[\mathsf{KEY}_{\mathsf{LT}(\mathcal{P},q)}, \mathsf{AUT}\right] \xrightarrow{(\mathsf{SE}, \sigma_{\mathsf{LT}}, \varepsilon)} \mathsf{SEC}_{\mathsf{LT}(\mathcal{P},q)},$$

where the distinguishing advantage ε can be reduced to the IND-CPA security of the underlying encryption scheme.

6.2 General Impossibility of PBE

The positive result for a single session can in general *not* be lifted to multiple sessions. Our impossibility result consists of providing a lower bound on the distinguishing advantage of a particular distinguisher $\mathbf{D}_\ell$ in distinguishing the systems $\mathsf{enc}^{\mathsf{A}}\,\mathsf{dec}^{\mathsf{B}}\,[\mathsf{KEY}^r_{\mathbf{T}}, \mathsf{AUT}^r]$ and $\sigma^{\mathsf{E}}\,\mathsf{SEC}^r_{\mathbf{T}}$ depicted in Fig. 2, for *any* trigger system $\mathbf{T}$ with output space $\{0,1\}^r$ and *any* simulator σ. The lower bound depends on the properties of the trigger system $\mathbf{T}$ and while giving a clear impossibility result for some triggers, for others it becomes moot. In particular, while it gives a strong bound for the case of the global password-guessing trigger $\mathsf{GT}\,(\mathcal{P}, q)$, the bound is inconclusive for the local trigger $\mathsf{LT}\,(\mathcal{P}, \boldsymbol{q})$ and independently distributed passwords, where in Sect. 6.3 we show that password-based encryption is actually possible.

The core of our impossibility result lies in exploiting the commitment problem explained in Sect. 6.1. The simulator $\sigma = \sigma_{\mathsf{LT}}$ there avoids this commitment problem by trying to break the session associated with the plaintext *before* having to output the corresponding ciphertext. This works out if σ follows the optimal strategy for breaking this particular session, since an arbitrary distinguisher would no be able to do better. However, since σ does not a priori know which session will have to be "decommitted", the simulator σ must be able to follow the optimal strategy for *each* session. This might be possible depending on the trigger system $\mathbf{T}$ (such as in the case of $\mathsf{LT}\,(\mathcal{P}, \boldsymbol{q})$ with independent passwords), but in general following the optimal strategy for a particular session may prevent σ from following the optimal strategy for another session. This is the case for the trigger $\mathsf{GT}\,(\mathcal{P}, q)$ where following the optimal strategy for a particular session consists of exhausting all the q allowed password-guessing queries on this session.

The high level idea of the distinguisher $\mathbf{D}_\ell$ is therefore to first force the simulator to be committed to a ciphertext in every session; and second, to pick a session j^* uniformly at random and to follow the optimal strategy to break it. To avoid the commitment problem, the simulator must in contrast try to break the maximum number of sessions before simulating the ciphertexts since it does not know which session j^* will be chosen by the distinguisher.

Theorem 1. *Let* $\mathrm{SE} := (enc, dec)$ *be a* correct *encryption scheme with key space* $\mathcal{K} := \{0,1\}^n$ *and message space* $\mathcal{M} \subseteq \{0,1\}^*$*, and consider the associated protocol* $\mathsf{SE} := (\mathsf{enc}, \mathsf{dec})$*. Let* $\mathbf{T}$ *be a trigger system with output space* $\{0,1\}^r$ *and let* $\mathcal{M}_\ell$ *denote a non-empty set of messages of fixed length* ℓ *in* $\mathcal{M}$*, for some integer* ℓ*. Then, there exists a distinguisher* $\mathbf{D}_\ell$ *such that, for all simulators* σ *and with* $\delta^{\mathbf{T}} := \Gamma^{\mathbf{T}}_{\mathsf{opt}} - \Gamma^{\mathbf{T}}_{\mathsf{avg}} \geq 0$*, we have*

$$\Delta^{\mathbf{D}_\ell}\left(\mathsf{enc}^{A}\,\mathsf{dec}^{B}\,[\mathsf{KEY}^r_{\mathbf{T}}, \mathsf{AUT}^r]\,,\ \sigma^{E}\,\mathsf{SEC}^r_{\mathbf{T}}\right) \quad \geq \quad \delta^{\mathbf{T}} - \frac{|\mathcal{K}|}{|\mathcal{M}_\ell|}\,. \tag{1}$$

The value $\Gamma^{\mathbf{T}}_{\mathsf{opt}}$ is the average advantage of optimal strategies per-session, whereas $\Gamma^{\mathbf{T}}_{\mathsf{avg}}$ is the optimal advantage of a global strategy. The formal definitions and a discussion on the bound obtained in (1) are in the full version.

6.3 PBE with Local Assumptions

Our impossibility result does not apply to the particular case of the local password-guessing trigger $\mathsf{LT}(\mathcal{P}, \boldsymbol{q})$ if the passwords are independently distributed, allowing for the existence of password-based encryption under these assumptions. Intuitively, since each session has its own restriction on the number of password-guessing queries, the simulation strategy can optimally brute-force each session independently to avoid the commitment problem, as in the simpler case of a single session discussed in Sect. 6.1.

The next informal theorem states that under these assumptions PBE achieves per-session confidentiality if the encryption scheme is IND-CPA secure. The formal statement and its proof are postponed to the full version [12].

Theorem (informal). *For every distribution $\mathcal{P}$ of r* independent *passwords and every tuple of r integers $\boldsymbol{q} := (q_1, \ldots, q_r)$, there exists a simulator σ_{LT} such that*

$$\left[\mathsf{KEY}^r_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})}, \mathsf{AUT}^r\right] \overset{(\mathsf{SE},\sigma_{\mathsf{LT}},\varepsilon)}{\Longrightarrow} \mathsf{SEC}^r_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})},$$

where ε can be reduced to the IND-CPA security of the encryption scheme.

6.4 Salting and PKCS #5

We examine in the full version [12] the well-known *salting technique*, a standard tool to achieve domain separation in password hashing. This technique consists of prefixing all queries made to a single random oracle RO_q, where q is an upper bound on the number of queries made by Eve, by a distinct bit string in each of the r sessions, making the queries from different sessions land in different subdomains of the random oracle. In practice, a randomly chosen bit string is used for every session, maintaining the same properties with high probability. Indeed, the salting technique constructs r *globally* restricted random oracles $[\mathsf{RO}, \ldots, \mathsf{RO}]_q$ but it *cannot* construct r *locally* restricted random oracles $[\mathsf{RO}_{q_1}, \ldots, \mathsf{RO}_{q_r}]$, at least not unless $q_j \geq q$ for all $j \in \{1, \ldots, r\}$ (which would render this construction uninteresting due to the blow-up in the number of adversarial queries). Intuitively, since the prefixes used are public, a distinguisher can use the same prefix for all its q queries, thereby forcing the simulator to query the same random oracle.

Consequences for local restrictions and PKCS #5. The above observation implies that relying on *local* query restrictions for multi-session security of password-based encryption appears to be in general rather unrealistic. The salting technique employed in PKCS #5 [15] (and more generally, any domain separation technique which is public) fails to construct locally restricted random oracles $[\mathsf{RO}_{q_1}, \ldots, \mathsf{RO}_{q_r}]$ from a single random oracle RO_q for any meaningful values of $q_1, \ldots, q_r$. As a consequence, we show in the full version that the same arguments used to prove Theorem 1 imply that PKCS #5 does provably *not* achieve per-session confidentiality.

7 Conclusion

The work of Bellare et al. [5] initiated the provable-security analysis of the techniques used in the password-based cryptography standard [15] and its application in password-based encryption. As discussed in Theorem 1, however, they do not prove the desired per-session security guarantee for PBE.

Even though we show that the results of [5] carry over to a composable model with per-session guarantees, this requires corresponding per-session assumptions on the distribution of adversary computation, and the simulation strategy we use is already quite peculiar: the simulator needs to know the password distribution and it must also make all password-guessing attempts before simulating the first ciphertext. This means that the constructed resource allows the attacker to aggregate its entire "computational power" and spend it in advance rather than distributed over the complete duration of the resource use, which results in a weaker guarantee than one might expect.

Our general impossibility result in Theorem 1 shows that bounding the adversary's queries per session, although an unrealistic assumption (as discussed in Sect. 6.4), is necessary for a simulation-based proof of security of PBE. Otherwise, a commitment problem akin to the one in adaptively secure public-key encryption (PKE) surfaces. Does that mean that we should stop using PBE in practice? In line with Damgård's [10] perspective on adaptively secure PKE, where a similar commitment-problem occurred [22], we view this question as being a fundamental research question still to be answered.[3] On the one hand, we lack an attack that would convincingly break PBE, but on the other hand we also lack provable-security support, to the extent that we can even show the impossibility in our model. Applications using these schemes should therefore be aware of the potential risk associated with their use. We believe that pointing out this commitment problem for PBE, analogously to adaptively secure PKE, is an important contribution of this paper.

References

1. Abadi, M., Warinschi, B.: Password-based encryption analyzed. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 664–676. Springer, Heidelberg (2005). doi:10.1007/11523468_54
2. Alwen, J., Serbinenko, V.: High parallel complexity graphs and memory-hard functions. In: Servedio, R.A., Rubinfeld, R. (eds.) 47th ACM STOC, pp. 595–603. ACM Press, June 2015
3. Bellare, M., O'Neill, A.: Semantically-secure functional encryption: possibility results, impossibility results and the quest for a general definition. In: Abdalla, M., Nita-Rotaru, C., Dahab, R. (eds.) CANS 2013. LNCS, vol. 8257, pp. 218–234. Springer, Cham (2013). doi:10.1007/978-3-319-02937-5_12
4. Bellare, M., Pointcheval, D., Rogaway, P.: Authenticated key exchange secure against dictionary attacks. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 139–155. Springer, Heidelberg (2000). doi:10.1007/3-540-45539-6_11

[3] Also affected are functional encryption [3,6,17] and identity-based encryption [14].

5. Bellare, M., Ristenpart, T., Tessaro, S.: Multi-instance security and its application to password-based cryptography. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 312–329. Springer, Heidelberg (2012). doi:10.1007/978-3-642-32009-5_19
6. Boneh, D., Sahai, A., Waters, B.: Functional encryption: definitions and challenges. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 253–273. Springer, Heidelberg (2011). doi:10.1007/978-3-642-19571-6_16
7. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. Cryptology ePrint Archive, Report 2000/067 (2000). http://eprint.iacr.org/2000/067
8. Canetti, R., Halevi, S., Katz, J., Lindell, Y., MacKenzie, P.: Universally composable password-based key exchange. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 404–421. Springer, Heidelberg (2005). doi:10.1007/11426639_24
9. Corrigan-Gibbs, H., Boneh, D., Schechter, S.: Balloon hashing: Provably space-hard hash functions with data-independent access patterns (2016)
10. Damgård, I.: A "proof-reading" of some issues in cryptography. In: Arge, L., Cachin, C., Jurdziński, T., Tarlecki, A. (eds.) ICALP 2007. LNCS, vol. 4596, pp. 2–11. Springer, Heidelberg (2007). doi:10.1007/978-3-540-73420-8_2
11. Demay, G., Gaži, P., Maurer, U., Tackmann, B.: Query-complexity amplification for random oracles. In: Lehmann, A., Wolf, S. (eds.) ICITS 2015. LNCS, vol. 9063, pp. 159–180. Springer, Cham (2015). doi:10.1007/978-3-319-17470-9_10
12. Demay, G., Gaži, P., Maurer, U., Tackmann, B.: Per-session security: Password-based cryptography revisited. Cryptology ePrint Archive, Report 2016/166, February 2016
13. Gennaro, R., Lindell, Y.: A framework for password-based authenticated key exchange. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 524–543. Springer, Heidelberg (2003). doi:10.1007/3-540-39200-9_33
14. Hofheinz, D., Matt, C., Maurer, U.: Idealizing identity-based encryption. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015. LNCS, vol. 9452, pp. 495–520. Springer, Heidelberg (2015). doi:10.1007/978-3-662-48797-6_21
15. Kaliski, B.: PKCS #5: Password-based cryptography specification. RFC 2898, September 2000
16. Katz, J., Ostrovsky, R., Yung, M.: Efficient password-authenticated key exchange using human-memorable passwords. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 475–494. Springer, Heidelberg (2001). doi:10.1007/3-540-44987-6_29
17. Matt, C., Maurer, U.: A definitional framework for functional encryption. In: IEEE 28th IEEE CSF, pp. 217–231, July 2015
18. Maurer, U.: Indistinguishability of random systems. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 110–132. Springer, Heidelberg (2002). doi:10.1007/3-540-46035-7_8
19. Maurer, U.: Constructive cryptography – a new paradigm for security definitions and proofs. In: Mödersheim, S., Palamidessi, C. (eds.) TOSCA 2011. LNCS, vol. 6993, pp. 33–56. Springer, Heidelberg (2012). doi:10.1007/978-3-642-27375-9_3
20. Maurer, U., Renner, R.: Abstract cryptography. In: Chazelle, B. (ed.) The Second Symposium in Innovations in Computer Science, ICS 2011, pp. 1–21. Tsinghua University Press, January 2011
21. Morris, R., Thompson, K.: Password security: A case history. Commun. ACM **22**(11), 594–597 (1979)

22. Nielsen, J.B.: Separating random oracle proofs from complexity theoretic proofs: the non-committing encryption case. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 111–126. Springer, Heidelberg (2002). doi:10.1007/3-540-45708-9_8
23. O'Gorman, L.: Comparing passwords, tokens, and biometrics for user authentication. Proc. IEEE **91**(12), 2021–2040 (2003)
24. Percival, C.: Stronger key derivation via sequential memory-hard functions. Self-published, pp. 1–16 (2009)
25. Petsas, T., Tsirantonakis, G., Athanasopoulos, E., Ioannidis, S.: Two-factor authentication: is the world ready? Quantifying 2FA adoption. In: Proceedings of the Eighth European Workshop on System Security, p. 4. ACM (2015)
26. Tackmann, B.: A Theory of Secure Communication. Ph.D. thesis, ETH Zürich, August 2014