Per-Session Security: Password-Based Cryptography Revisited

Grégory Demay[1], Peter Gaži[2], Ueli Maurer[1], and Björn Tackmann[3,⋆]

[1] Department of Computer Science, ETH Zürich, Switzerland
{gregory.demay,maurer}@inf.ethz.ch
[2] Institute of Science and Technology, Austria
peter.gazi@ist.ac.at
[3] Computer Science & Engineering, University of California San Diego
btackmann@eng.ucsd.edu

February 18, 2016

Abstract. Cryptographic security is usually defined as some form of guarantee that holds except when a bad event with negligible probability occurs, and nothing is guaranteed in that case. However, in settings where such 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 an unguessable 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. In particular, a user with a very strong password enjoys the full security guarantees of an analysis in which passwords are replaced by uniform cryptographic keys.

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 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 since we show that standard domain separation techniques employed in password-based cryptography, such as salting, can only provide *global* restrictions on the number of adversarial password-guessing queries. Quite surprisingly, we show that in this more realistic scenario the desired per-session confidentiality is unachievable. This impossibility result resolves an open problem stated by Bellare, Ristenpart and Tessaro (CRYPTO 2012).

Keywords: password-based encryption, simulation-based security, random oracle

⋆ Work partially done while author was at ETH Zürich.

Contents

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 [PTAI15]. 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 negligible in the analysis. 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 non-negligible terms from the analyses of the individual sessions accumulate.

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. [BRT12] on multi-instance security, which we discuss in greater detail below.

From a more general 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[4] that are used in protocols or applications and can be gradually weakened (we call this "downgrading"). Later, we apply the technique to password-based cryptography 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 [Mau12] 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 [Can00] would have to overcome certain technical hurdles that stem from the difference between these two frameworks, as we detail in Appendix A.

[4] These resources formalize the assumed and the expected guarantees in security definitions.

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. 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 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. This also solves an open problem posed in the full version of [BRT12].
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, a premise that is also present in [BRT12]. This assumption seems implausible in general, in fact we show that the salting technique often used in the context of password hashing does not satisfy it; 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. This result can be seen as a stronger (composable) analogue of the positive result from [BRT12].
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 [Kal00].

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

Multi-instance security. At a very high level, the notion of multi-instance security proposed in [BRT12] 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, even if the results of [BRT12] were to be generalized to deal with arbitrary password distributions (the results of [BRT12] are phrased for passwords which are independently and identically distributed), the resulting guarantees would be in the form of a global upper bound on the number of sessions that can be broken and would not give any guarantee for a specific session. In particular, they would not guarantee that a session whose password was not guessed is secure, independently of whether other sessions were compromised. Finally, [BRT12] only investigates PBE under the assumption that the number of local password guesses for each of the sessions is a priori known, which limits the applicability of the results in realistic scenarios where only a global bound on the attacker's computational power may be known or assumed.

Password-based security. There is an extensive body of work investigating password-based security mechanisms from various perspectives. On the empirical side, the weaknesses of passwords in practice were studied e.g. in [O'G03].

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) [Kal00], iteration [MT79, DGMT15], and hashing with moderately hard-to-compute functions [Per09, AS15, CGBS16]. 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 [BRT12] which studied key derivation functions proposed in PKCS #5 [Kal00] and did focus on security in a setting with multiple users.

A key-recovery security definition for password-based encryption was given in [AW05], but here also only single-session security was considered.

Finally, a separate line of work aims at realizing password-authenticated key exchange (PAKE) protocols [BPR00, GL03, CHK+05] that prevent the possibility of offline password-guessing attacks and result in keys that can then safely be used for encryption or authentication. The protocols are, however, intrinsically interactive and cannot be used in non-interactive password-based settings such as ours.

The commitment problem. A simulation problem similar to the one that we described in the context of PBE has already been observed in various other contexts such as public-key encryption with adaptive corruptions [Nie02], functional encryption [BSW11, BO13, MM15] and identity-based encryption [HMM15].

Multi-user security. An orthogonal notion called *multi-user security* was studied in [BBM00]. However, its aims are different: the adversary wins if she manages to break a single session, possibly exploiting the presence of other sessions. In contrast, we want to provide guarantees even after some sessions are broken.

2 Preliminaries

2.1 Basic Notation

We denote sets by calligraphic letters or capital Greek letters (e.g., $\mathcal{X}$, Σ). Throughout this paper, we consider only discrete random variables. A discrete random variable will be denoted by an upper-case letter X, its range by the corresponding calligraphic letter $\mathcal{X}$, and a realization of the random variable X will be 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.

2.2 Systems

Many cryptographic primitives (e.g. block ciphers, MAC schemes, random functions) can be described as $(\mathcal{X}, \mathcal{Y})$-*random systems* [Mau02] 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 $Y_1, \ldots, Y_{k-1}$. We consider three distinct types of random systems: resources, converters and distinguishers.

Resources and converters. Resources that can be accessed by multiple parties can be viewed as random systems with multiple interfaces and formalized by making an interface identifier an explicit part of the input (or output). The resources in this work have 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. As a notational convention, 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 strategy (or protocol machine) employed locally by a party is modeled by a *converter*, which can also be viewed as a random system with two interfaces: an *inside* interface and an *outside* interface, denoted by in and out, respectively. In this view, the inside interface is attached to the i-interface of a resource and models how the scheme makes use of this resource, where $i \in \{\mathsf{A}, \mathsf{B}, \mathsf{E}\}$, while the outside interface of the converter becomes the i-interface of the composite system and models how the scheme can be used in applications and higher-level protocols. 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 the inside interface of a converter α to the i-interface of a resource $\mathbf{R}$ is denoted by $\alpha^i\,\mathbf{R}$ and the resulting system is again a resource. We assume the existence of an identity converter id, which forwards all its inputs at one interface to the other one, such that $\mathsf{id}^i\,\mathbf{R} = \mathbf{R}$. Any two converters α and β can be composed sequentially, denoted $\beta \circ \alpha$, by connecting the inside interface of β to the outside interface of α. It can then be shown that the operations described are such that $(\beta \circ \alpha)^i\,\mathbf{R} = \beta^i\,\left(\alpha^i\,\mathbf{R}\right)$, and that the application of converters at different interfaces h and i commute in the sense that $\alpha^h\,\beta^i\mathbf{R} = \beta^i\,\alpha^h\,\mathbf{R}$.

For two resources $\mathbf{R}$ and $\mathbf{S}$, we denote by $[\mathbf{R}, \mathbf{S}]$ their parallel composition. For each interface $i \in \{\mathsf{A}, \mathsf{B}, \mathsf{E}\}$, the i-interface of $\mathbf{R}$ and $\mathbf{S}$ are merged and become the *sub-interfaces* of the i-interface of $[\mathbf{R}, \mathbf{S}]$, which we denote by i_1 and i_2, respectively. A converter α that connects to the i-interface of $[\mathbf{R}, \mathbf{S}]$ has two inside sub-interfaces, denoted by in_1 and in_2, where the first one connects to i_1 and the second one to i_2. Any two converters α and β can also be taken in parallel, denoted $\langle \alpha, \beta \rangle$, which can be defined such that $\langle \alpha, \beta \rangle^i\,[\mathbf{R}, \mathbf{S}] = \left[\alpha^i\,\mathbf{R}, \beta^i\,\mathbf{S}\right]$. These parallel composition operations and associated notations extend to the case of more than two systems in a straightforward manner.

Distinguishers and reductions. A natural notion of similarity for resources can be based on the concept of *distinguishing*. Intuitively, a distinguisher $\mathbf{D}$ can be viewed as a random system that connects to all the interfaces of a resource $\mathbf{R}$, interacts with this resource, and at the end of this random experiment outputs a single bit denoted B. The complete interaction of $\mathbf{D}$ and $\mathbf{R}$ defines a random experiment and the probability that the bit B is 1 in this experiment is written as $\mathsf{P}^{\mathbf{DR}}\,(B = 1)$. For two resources $\mathbf{R}$ and $\mathbf{S}$, the *distinguishing advantage* of a distinguisher $\mathbf{D}$ in telling apart $\mathbf{R}$ from $\mathbf{S}$ is then defined as

$$\Delta^{\mathbf{D}}\,(\mathbf{R}, \mathbf{S}) \quad := \quad \left|\mathsf{P}^{\mathbf{DR}}\,(B = 1) - \mathsf{P}^{\mathbf{DS}}\,(B = 1)\right| \quad .$$

The resources $\mathbf{R}$ and $\mathbf{S}$ are said to be *equivalent*, denoted $\mathbf{R} \equiv \mathbf{S}$, if $\Delta^{\mathbf{D}}\,(\mathbf{R}, \mathbf{S}) = 0$ for all distinguishers $\mathbf{D}$. A distinguisher $\mathbf{D}$ emulating a converter α at interface $i \in \{\mathsf{A}, \mathsf{B}, \mathsf{E}\}$ induces a new distinguisher, denoted $\mathbf{D}\alpha^i$, defined by $\Delta^{\mathbf{D}\alpha^i}\,(\mathbf{R}, \mathbf{S}) := \Delta^{\mathbf{D}}\,\left(\alpha^i\,\mathbf{R},\, \alpha^i\,\mathbf{S}\right)$.

We will often make reductions between two distinguishing problems. That is, a distinguisher $\mathbf{D}$ trying to tell apart two resources $\mathbf{U}$ and $\mathbf{V}$ is transformed into a new distinguisher whose task is instead to distinguish the resource $\mathbf{R}$ from $\mathbf{S}$. Such a reduction is done by exhibiting a *reduction system* $\mathbf{C}$ which translates one setting into the other. More formally, such a reduction system $\mathbf{C}$ is a converter (with one inside and one outside interface), where the inside interface of $\mathbf{C}$ connects to the merged interfaces of the resource $\mathbf{R}$, denoted $\mathbf{CR}$, and the outside interface of $\mathbf{C}$ can be accessed by a distinguisher. To reduce the task of distinguishing

$\mathbf{U}$ from $\mathbf{V}$ to that of $\mathbf{R}$ and $\mathbf{S}$, one exhibits a reduction system $\mathbf{C}$ such that $\mathbf{U} \equiv \mathbf{CR}$ and $\mathbf{V} \equiv \mathbf{CS}$. For all distinguishers $\mathbf{D}$ it then follows that $\Delta^{\mathbf{D}}(\mathbf{U}, \mathbf{V}) = \Delta^{\mathbf{D}}(\mathbf{CR}, \mathbf{CS}) = \Delta^{\mathbf{DC}}(\mathbf{R}, \mathbf{S})$, where the last equality comes from the fact that $\mathbf{C}$ can also be thought of as being part of the distinguisher.

2.3 The Construction Notion in the (Alice, Bob, Eve)-setting

We formalize the security of protocols by the following notion of construction, which was introduced in the work of Maurer [Mau12] and is a special case of the abstract cryptography framework developed by Maurer and Renner [MR11]. 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 system (i.e., the assumed resource together with the protocol) he can also accomplish in the ideal-world system (i.e., the desired resource with the simulator). 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[5].

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*

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

for all distinguishers $\mathbf{D}$.

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. Availability and security are preserved under sequential or parallel composition. More details can be found in [Mau12, Tac14].

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 $\mathbf{R} \overset{\pi, \sigma, \varepsilon}{\longmapsto} \mathbf{S}$ for $(\mathbf{R}, \mathbf{R}_\perp) \overset{(\pi, \sigma, (0, \varepsilon))}{\longmapsto} (\mathbf{S}, \mathbf{S}_\perp)$ and where $\mathbf{R}_\perp$ (respectively, $\mathbf{S}_\perp$) is implicitly understood from $\mathbf{R}$ (respectively, $\mathbf{S}$).

2.4 Symmetric Cryptographic Schemes

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 $m \in \mathcal{M}$.

A MAC scheme $\mathrm{MAC} := (tag, vrf)$ is considered *weakly unforgeable under chosen-message attack* (WUF-CMA) if it is computationally infeasible, even when given access to an oracle producing valid tags for chosen messages, to generate a valid tag for a fresh message that was not queried before to the oracle. The associated security game, denoted $\mathbf{G}^{\mathsf{CMA}}(\mathrm{MAC})$ is detailed in Alg. 1.

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

[5] Formally, $\mathbf{R}_\perp := \gamma^{\mathsf{E}}\mathbf{R}$ for some converter γ modelling the actions of a non-malicious adversary.

Alg. 1: WUF-CMA game $\mathbf{G}^{\mathsf{CMA}}$ (MAC)

$\mathsf{win} := 0$, $k \xleftarrow{\$} \mathcal{K}$, $\mathcal{B} := \emptyset$
on input (tag, m)
  $\mathcal{B} := \mathcal{B} \cup \{m\}$
  output $tag\,(k, m)$
on input (vrf, m, u)
  $b := vrf\,(k, m, u)$
  $\mathsf{win} := \mathsf{win} \vee (b \wedge (m \notin \mathcal{B}))$
  output b

Alg. 2: IND-CPA system $\mathbf{G}_b^{\mathsf{CPA}}$ (SE)

$k \xleftarrow{\$} \mathcal{K}$
on input $m_0 \in \mathcal{M}$
  $m_1 \xleftarrow{\$} \mathcal{M}_{|m_0|}$
  $c \leftarrow enc\,(k, m_b)$
  output c

$m' := dec\,(k, c)$. 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 $m \in \mathcal{M}$.

To define the security of an encryption scheme we use the "real or random" definition of *indistinguishability under chosen-plaintext attack* (IND-CPA) as it matches more closely constructive security definitions which involve the comparison of a "real" system with an "ideal" one. Relations to other notions of IND-CPA follow from the work of Bellare et al. [BDJR97]. Thus, an encryption scheme SE := (enc, dec) is said to be IND-CPA-secure if no efficient distinguisher can tell apart the system $\mathbf{G}_0^{\mathsf{CPA}}$ (SE), which encrypts input messages, from the system $\mathbf{G}_1^{\mathsf{CPA}}$ (SE), which encrypts random messages of the same length as the input messages. The system $\mathbf{G}_b^{\mathsf{CPA}}$ (SE) is described in Alg. 2, where $\mathcal{M}_\ell$ stands for messages of length ℓ 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*.

3.1 Core Systems and Triggers

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.

Hence, 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 thus 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, such as $\{0, 1\}$ in our example above. 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.*

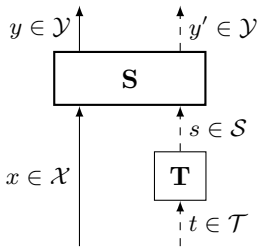

Fig. 1. A transformable system $\mathbf{S_T}$ formed by combining a core system $\mathbf{S}$ with a trigger system $\mathbf{T}$. "Normal" inputs $x \in \mathcal{X}$ are processed directly by $\mathbf{S}$, while trigger values $t \in \mathcal{T}$ go instead first through the system $\mathbf{T}$ whose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

Note that a transformable system is just a particular type of a random system, hence any security definition applying to random systems (e.g. the construction notion of Definition 1) also applies to transformable systems.

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.

3.2 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). Resources obtained by fixing the switch of such core resources to a particular value will no longer have this interface $\mathsf{E_S}$. 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.

We now introduce *downgradable key resources* and *downgradable secure channels*, examples of such resources that will be used throughout the paper. These resources are parametrized (among other) by a fixed number r of sessions. Intuitively speaking, 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[6] in Alg. 3.

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 Alg. 4.

[6] To match the formal definition of a random system, which provides an output for each received input, the core resources KEY^r and SEC^r should output a dummy message at the $\mathsf{E_N}$-interface every time a switch value is input at the $\mathsf{E_S}$-interface. This technicality is omitted here and below for the sake of clarity.

Alg. 3: Core resource KEY^r

$s_j := 0$ and $k_j \xleftarrow{\$} \{0,1\}^n$, for all $j \in \{1,\dots,r\}$
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 $\mathsf{E_S}$
  $(s_1, \dots, s_r) := s$
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. 4: Core resource SEC^r

$s_j := 0$ and $m_j := \diamond$, for all $j \in \{1,\dots,r\}$
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 $\mathsf{E_S}$
  $(s_1, \dots, s_r) := s$
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 $\mathsf{E_N}$

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 parametrized 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 guesses allowed, either locally for each of the sessions (a tuple $\boldsymbol{q} := (q_1, \dots, q_r)$) or globally (a parameter q). Both $\mathsf{LT}(\mathcal{P}, \boldsymbol{q})$ and $\mathsf{GT}(\mathcal{P}, q)$ initially sample r passwords $(w_1, \dots, w_r)$ according to $\mathcal{P}$. When a password guess (j, w) for the j^{th} session is received, $\mathsf{LT}(\mathcal{P}, \boldsymbol{q})$ changes the state of this session to "broken" if the password guess is correct and no more than q_j guessing queries were made to that session. In contrast, $\mathsf{GT}(\mathcal{P}, q)$ declares a session "broken" if the password was correctly guessed for this session and no more than q password-guessing queries were made in total to all sessions. Both triggers $\mathsf{LT}(\mathcal{P}, \boldsymbol{q})$ and $\mathsf{GT}(\mathcal{P}, q)$ are only accessible by Eve and are detailed in Alg. 5 and 6.

Alg. 5: Local trigger $\mathsf{LT}(\mathcal{P}, \boldsymbol{q})$

$(w_1, \dots, w_r) \leftarrow \mathcal{P}$
$s_j := 0$ and $\ell_j := 0$, for all $j \in \{1,\dots,r\}$
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 $(s_1, \dots, s_r)$ at $\mathsf{E_S}$

Alg. 6: Global trigger $\mathsf{GT}(\mathcal{P}, q)$

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

4 Password-Based Key Derivation

In this section we formalize the simple protocol for deriving a key from a password via hashing that was considered as an example in Section 3. We confirm that, as expected, depending on the assumed resources to start with, this protocol constructs one of the variants of the downgradable key resource. These constructions will turn out to be useful to us later.

Informally, the assumed resources our construction starts with consist of a shared password and a hash function (modelled as a random oracle) for each of the sessions. Note that these independent random oracles can be constructed from a single one via *salting* (i.e., domain separation), a point that we will discuss in greater detail later in Section 6.4.

More formally, we model the shared passwords as an explicit resource denoted PW. It is parametrized 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 excepted for the a priori known distribution $\mathcal{P}$.

Each hash function is modelled 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 [DGMT15].) 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.

Our key-derivation protocol is performed by a converter kd. Upon a key request (j, getkey) for the j^{th} session, it 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 shows that the protocol $\mathsf{KD} := (\mathsf{kd}, \mathsf{kd})$, where each party simply applies the converter kd, allows users to obtain downgradable keys in the sense of Section 3.2.

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*

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

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

Proof. The simulator σ_{kd} emulates r random oracles mostly by lazy sampling. More precisely, upon an adversarial query x made to the j^{th} random oracle, the simulator σ_{kd} forwards x as a password guess for the j^{th} session to the trigger $\mathsf{LT}(\mathcal{P}, \boldsymbol{q})$ or $\mathsf{GT}(\mathcal{P}, q)$ at its $\mathsf{in_S}$ interface and then tries to retrieve the key associated with that session by querying its $\mathsf{in_N}$-interface. If the password guess was correct, then the simulator σ_{kd} can output the retrieved key as a simulated output of the random oracle, and otherwise it just samples a uniform n-bit string. The simulator σ_{kd} is described in Alg. 7.

Note that the simulator σ_{kd} does not have to keep track of the number of queries since this is handled directly by the triggering system which receives every query received by σ_{kd}. The simulation is therefore perfect in both settings, independently of whether the random oracles are locally or globally restricted. □

Alg. 7: Simulator σ_{kd}

$g_j(x) := \diamond$, for all $x \in \{0,1\}^*$ and $j \in \{1, \ldots, r\}$
on input (j, x) at out_1
  output (j, x) at $\mathsf{in_S}$
  if $g_j(x) = \diamond$ **then**
    $(j, k) :=$ **result of querying** (j, getkey) at $\mathsf{in_N}$
    if $k \neq \blacklozenge$ **then** $g_j(x) := k$
    else $g_j(x) \xleftarrow{\$} \{0,1\}^n$
  output $(j, g_j(x))$ at out_1

5 Password-Based Message Authentication

In this section 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 refer to Fig. 2 for the depictions of the real and the ideal experiment involved in the construction statement.

We present these results partly to put them in contrast with our later findings on password-based encryption, where the situation turns out to be more intricate. As a consequence, in this section we deliberately remain slightly informal and postpone the full formal treatment to Appendix B.

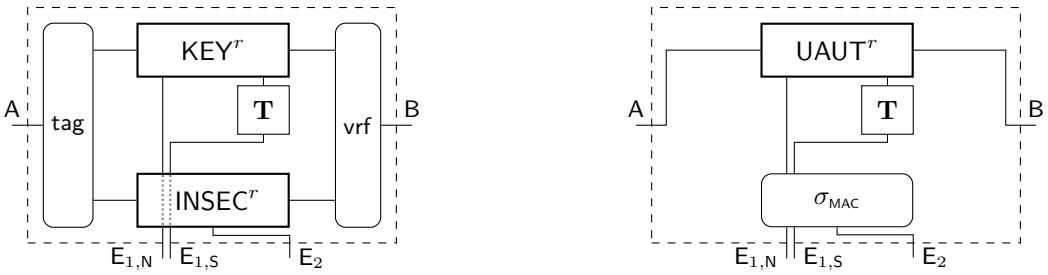

Fig. 2. Left: The assumed resource, a downgradable key $\mathsf{KEY}^r_{\mathbf{T}}$ and an insecure channel INSEC^r, with protocol converters tag and vrf attached to interfaces A and B, denoted $\mathsf{tag}^{\mathsf{A}}\mathsf{vrf}^{\mathsf{B}}[\mathsf{KEY}^r_{\mathbf{T}}, \mathsf{INSEC}^r]$. Right: The desired downgradable unordered authenticated channel $\mathsf{UAUT}^r_{\mathbf{T}}$ with simulator σ_{MAC} attached to interface E, denoted $\sigma^{\mathsf{E}}_{\mathsf{MAC}}\mathsf{UAUT}^r_{\mathbf{T}}$. The simulator σ_{MAC} 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 insecure channel at E_2.

Assumed resources. In the construction statement shown below, we assume 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 $\mathsf{LT}(\mathcal{P}, \boldsymbol{q})$ or $\mathsf{GT}(\mathcal{P}, q)$). The insecure channels are formalized as the resource INSEC^r which forwards any message sent by Alice directly to Eve, while any message injected by Eve is forwarded directly to Bob.

MAC schemes as protocols. Recall the definition of a MAC scheme stated in Section 2.4. Given a MAC scheme (tag, vrf), the protocol for Alice and Bob works 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 consisting of a pair (j, m), it retrieves the key k_j associated to this session from the downgradable key resource $\mathsf{KEY}^r_{\mathbf{T}}$, computes the tag $u := tag(k_j, m)$ and outputs to the insecure channel INSEC^r the pair $(j, m \,\|\, u)$. On the other end of the channel, whenever vrf receives a message and a tag for some session, consisting of a pair $(j', m' \,\|\, u')$, it first retrieves the key $k_{j'}$ associated to this session from $\mathsf{KEY}^r_{\mathbf{T}}$, computes $vrf(k_{j'}, m', u')$ and outputs (j', 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, thus 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, every message input by Alice is directly forwarded to Eve independently of the current switch value, while the switch value s_j of a particular session can also be retrieved by Eve. Note that the unordered authenticated channel UAUT^r, similarly to a MAC scheme, only aims to prevent Eve from being able to inject a *fresh* message, it does not a priori prevent the injection of a legitimate message multiple times, the reordering of legitimate messages, or the loss of some messages.

The next informal theorem states that 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 postponed to Theorem 4 in Appendix B.

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.

Proof (sketch). According to Definition 1, we need to find a simulator σ_{MAC} and show that the systems $\mathsf{tag}^{\mathsf{A}}\,\mathsf{vrf}^{\mathsf{B}}\,[\mathsf{KEY}^r_{\mathbf{T}}, \mathsf{INSEC}^r]$ and $\sigma^{\mathsf{E}}_{\mathsf{MAC}}\,\mathsf{UAUT}^r_{\mathbf{T}}$ are indistinguishable. The role of the simulator σ_{MAC} is to emulate what happens at Eve's interface E in the real system $\mathsf{tag}^{\mathsf{A}}\,\mathsf{vrf}^{\mathsf{B}}\,[\mathsf{KEY}^r_{\mathbf{T}}, \mathsf{INSEC}^r]$ by having only access to the idealized channel $\mathsf{UAUT}^r_{\mathbf{T}}$. The simulator σ_{MAC} has therefore two tasks: 1) to emulate key retrieval queries and password-guessing queries intended for the key resource $\mathsf{KEY}^r_{\mathbf{T}}$; and 2) to emulate messages injected by Eve to the insecure channel INSEC^r, as well as messages output by INSEC^r to Eve.

In the real system, a message (j, m) input by Alice for session j is seen by Eve as $(j, m \,\|\, u)$, where $u := tag\,(k_j, m)$ and k_j is a key which was initially selected uniformly at random. In the ideal system, such a message (j, m) is directly output by the channel $\mathsf{UAUT}^r_{\mathbf{T}}$ to the simulator which therefore needs to mimic the tagging process. The simulator σ_{MAC} can easily do so by initially selecting a key k_j uniformly at random and outputting $(j, m \,\|\, tag\,(k_j, m))$. The simulator σ_{MAC} forwards any password-guessing query to the trigger $\mathbf{T}$ of the channel $\mathsf{UAUT}^r_{\mathbf{T}}$, whereas σ_{MAC} deals with key retrieval queries for session j by first asking the channel $\mathsf{UAUT}^r_{\mathbf{T}}$ whether session j is broken, and then outputting accordingly the key k_j used to simulate the tagging process or the error symbol $\blacklozenge$.

Note that so far the simulation is perfect. The only difference between the real and ideal system lies in the way injection queries are handled. An injection message $(j', m' \,\|\, u')$ made by Eve into the real system is output to Bob as (j', m') only if the tag is valid, i.e., $vrf\,(k_{j'}, m', u') = 1$. Thus, when the simulator receives such a query it only outputs (j', m') to the channel $\mathsf{UAUT}^r_{\mathbf{T}}$ if the tag u' is valid. As a result, the channel $\mathsf{UAUT}^r_{\mathbf{T}}$ outputs the desired message (j', m') to Bob, unless the session j' is not broken and this message was never input by Alice. However, such an injection query $(j', m' \,\|\, u')$ constitutes a forgery and cannot be found efficiently if the underlying MAC scheme (tag, vrf) is weakly unforgeable, so overall the real and ideal systems are (computationally) indistinguishable. □

6 Password-Based Encryption

In this section we investigate the use of password-derived keys for symmetric encryption. In a multi-session setting, one would again intuitively 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. We now describe this desired construction in greater detail, referring to Fig. 3 for the depictions of the real and the ideal experiment.

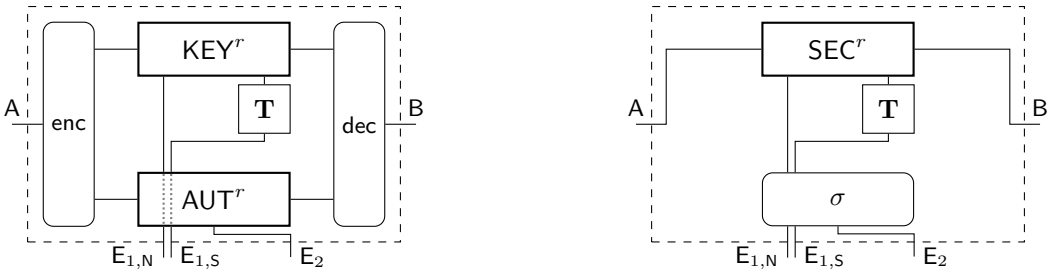

Fig. 3. Left: The assumed resource, a downgradable key $\mathsf{KEY}^r_{\mathbf{T}}$ and an authenticated channel AUT^r, with protocol converters enc and dec attached to interfaces A and B, denoted $\mathsf{enc}^{\mathsf{A}}\mathsf{dec}^{\mathsf{B}}\,[\mathsf{KEY}^r_{\mathbf{T}}, \mathsf{AUT}^r]$. Right: 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.

Assumed resources. These construction statements assume the availability of a password-derived key and an authenticated communication channel for each of the r sessions. For password-derived 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 $\mathsf{GT}\,(\mathcal{P}, q)$. The authenticated channel is formalized as a resource AUT^r which takes as input a pair (j, c) at Alice's interface A, corresponding

to a ciphertext c to be transmitted for the j^{th} session, and outputs (j, c) at both[7] Eve's interface E and Bob's interface B. For convenience, we will also assume that at most one message is transmitted per session, but our results could easily be extended to the case of multiple messages per session. We also simplify our statements by assuming that the channel AUT^r has no particular weaknesses, i.e., Eve can neither replay messages nor drop those coming from Alice.

Encryption schemes as protocols. Recall the definition of an encryption scheme stated in Section 2.4. Given an encryption scheme (enc, dec), the protocol for Alice and Bob again works in the natural way (their converters are denoted enc and dec, respectively). Both enc and dec expect two resources at their respective inside interface in: a downgradable key resource $\mathsf{KEY}^r_{\mathbf{T}}$ at their interface in_1 and an authenticated communication channel AUT^r at their interface in_2. The converter enc takes as input a pair (j, m) at its outside interface out, corresponding to a message m to be transmitted for the j^{th} session. It retrieves the shared secret key k_j for this session by inputting (j, getkey) at its in_1-interface, computes the ciphertext $c \leftarrow enc\,(k_j, m)$ and finally outputs (j, c) to the channel at its in_2-interface. Similarly, when a pair (j, c) is input to the in_2-interface of the converter dec, corresponding to an encrypted message sent for the j^{th} session, the converter dec retrieves the shared secret key k_j for this session (by inputting (j, getkey) at its in_1-interface) and outputs $(j, dec\,(k_j, c))$ at its out-interface. Throughout this section, we will assume the underlying encryption scheme (enc, dec) to be *correct*.

Desired 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 Section 3.2, which guarantees that any message sent by Alice for a particular session is transmitted confidentially to Bob, unless this session was "broken".

The goal of password-based encryption can then be loosely phrased as achieving the following construction

$$[\mathsf{KEY}^r_{\mathbf{T}}, \mathsf{AUT}^r] \quad \overset{(\mathsf{SE}, \sigma, \varepsilon)}{\longmapsto} \quad \mathsf{SEC}^r_{\mathbf{T}},$$

for some simulator σ and "reasonable" distinguishing advantage ε.

6.1 PBE for a Single Session

We start by focusing on PBE with a single session. This will serve as an introduction to the study of the multi-session setting, given in Sections 6.2 and 6.3.

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 actually identical for any distribution $\mathcal{P}$ of a single password and any number q of password-guessing queries. To be consistent with what lies ahead in Section 6.3, we will hence always mention $\mathsf{LT}\,(\mathcal{P}, q)$ in this section. The considered special case also allows us to drop the exponent r indicating the number of sessions and the index j of the session associated with any input or output, thus simplifying the notation.

We are interested in the possibility of constructing the downgradable secure channel $\mathsf{SEC}_{\mathsf{LT}(\mathcal{P},q)}$ 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 such that the systems $\mathsf{enc}^{\mathsf{A}}\,\mathsf{dec}^{\mathsf{B}}\,\left[\mathsf{KEY}_{\mathsf{LT}(\mathcal{P},q)}, \mathsf{AUT}\right]$ and $\sigma^{\mathsf{E}}\,\mathsf{SEC}_{\mathsf{LT}(\mathcal{P},q)}$ represented in Fig. 3 with $\mathbf{T} = \mathsf{LT}\,(\mathcal{P}, q)$, are 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. 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.

[7] The described authenticated channel AUT^r is formally not a random system since an input at A yields two outputs: one at B and another at E. This issue can be resolved by considering instead a channel in which transmitted messages are "pulled", i.e., on input (j, c) at A, this channel would return a dummy message at A, while the other parties Bob and Eve could retrieve the message input by Alice for session j only after having input (j, getmsg) at their respective interface. Such a technicality is omitted for the sake of clarity.

A naïve simulation strategy could start as follows. The simulator σ initially selects a key k uniformly at random and emulates 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 $\mathsf{SEC}_{\mathsf{LT}(\mathcal{P},q)}$.

However, this approach does not work. To see this, consider what happens when the password is guessed and the session is broken (which, depending on $\mathcal{P}$, may happen with a large probability). In the real world, the distinguisher can retrieve the key k used for encryption and check that the 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.

To prevent the simulation breakdown, we instead consider a different simulator σ_{LT} which, in contrast to σ, tries to break the session *before* having to output any ciphertext. So, 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 this initial brute-force step fails, σ_{LT} simply encrypts a random message of the correct length and declares any password guess as incorrect so that the actual key used to simulate the encryption is never output. However, if the initial 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 be appropriately "scaled up" to match the guessing probability of the real world. In this case, the key used to simulate encryption can obviously be output since the ciphertext produced is an actual encryption of the transmitted message.

In this setting with a single session, password-based encryption is therefore possible with respect to the simulation strategy σ_{LT} sketched above.

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

$$\left[\mathsf{KEY}_{\mathsf{LT}(\mathcal{P},q)}, \mathsf{AUT}\right] \quad \overset{(\mathsf{SE},\, \sigma_{\mathsf{LT}},\, \varepsilon)}{\vdash\!\!\Longrightarrow} \quad \mathsf{SEC}_{\mathsf{LT}(\mathcal{P},q)},$$

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

The generalization of the above statement for multiple sessions and its proof, together with the definition of the simulator σ_{LT}, are postponed to Theorem 2 in Section 6.3. This corollary then easily follows from Theorem 2 by taking $r = 1$.

6.2 General Impossibility of PBE

In this section we show that the above positive result concerning password-based encryption for a single session can in general *not* be lifted to the case of 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. 3, for *any* trigger system $\mathbf{T}$ with output space $\{0,1\}^r$ and *any* simulator σ. The lower bound obtained 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. We show later in Section 6.3 that in this specific case password-based encryption is actually possible.

The core of our impossibility result lies in exploiting the commitment problem explained in Section 6.1. Intuitively, the simulator σ can avoid 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 $\mathsf{LT}(\mathcal{P}, \boldsymbol{q})$ if the passwords are independently distributed), but in general following the optimal strategy for a particular session may prevent σ from following the optimal strategy for another session. This is for example 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.

We need to add some notation in order to quantify more precisely the distinguishing advantage achieved by $\mathbf{D}_\ell$. Let $\mathbf{T}$ denote an arbitrary trigger system with output space $\{0,1\}^r$, and consider a distinguisher $\mathbf{D}$ interacting with the trigger $\mathbf{T}$ alone. Note in particular that such a distinguisher sees the output $(s_1, \ldots, s_r) \in \{0,1\}^r$ of $\mathbf{T}$. The interaction of the distinguisher $\mathbf{D}$ with the trigger $\mathbf{T}$ defines a random experiment, and we denote by $\mathcal{G}_j$ the event that the j^{th} session is "broken", i.e., $\mathbf{T}$ output at least once an r-bit string $(s_1, \ldots, s_r)$ with $s_j = 1$, for all $j \in \{1, \ldots, r\}$. The probability that this event $\mathcal{G}_j$ happens in this random experiment is denoted $\mathsf{P}^{\mathbf{DT}}(\mathcal{G}_j = 1)$ and it will be convenient to see it as a function of both j and $\mathbf{D}$. To do so, let $\mathcal{D}$ denote the set of distinguishers for $\mathbf{T}$ and consider the function $\Gamma^{\mathbf{T}} : \{1, \ldots, r\} \times \mathcal{D} \to [0,1]$, where $\Gamma^{\mathbf{T}}(j, \mathbf{D}) := \mathsf{P}^{\mathbf{DT}}(\mathcal{G}_j = 1)$, for all $j \in \{1, \ldots, r\}$ and $\mathbf{D} \in \mathcal{D}$.

In the following, we denote by $\Gamma^{\mathbf{T}}_{\mathsf{opt}}$ the average of the maximum probability in breaking a particular session, while $\Gamma^{\mathbf{T}}_{\mathsf{avg}}$ denotes the maximum average probability of breaking all sessions,

$$\begin{aligned} \Gamma^{\mathbf{T}}_{\mathsf{opt}} &:= \frac{1}{r} \sum_{j \in \{1,\ldots,r\}} \max_{\mathbf{D} \in \mathcal{D}} \Gamma^{\mathbf{T}}(j, \mathbf{D}), \\ \Gamma^{\mathbf{T}}_{\mathsf{avg}} &:= \frac{1}{r} \max_{\mathbf{D} \in \mathcal{D}} \sum_{j \in \{1,\ldots,r\}} \Gamma^{\mathbf{T}}(j, \mathbf{D}) \ . \end{aligned} \tag{1}$$

Note that $\max_{\mathbf{D} \in \mathcal{D}} \Gamma^{\mathbf{T}}(j, \mathbf{D}) \geq \Gamma^{\mathbf{T}}(j, \mathbf{D})$, for every $j \in \{1, \ldots, r\}$, and thus $\Gamma^{\mathbf{T}}_{\mathsf{opt}} \geq \Gamma^{\mathbf{T}}_{\mathsf{avg}}$. In other words, $\Gamma^{\mathbf{T}}_{\mathsf{opt}}$ measures the success of an optimal strategy for breaking a *known* randomly chosen session j, while $\Gamma^{\mathbf{T}}_{\mathsf{avg}}$ measures the success of an optimal strategy for breaking an *unknown* randomly chosen session. Knowing in advance which session will be "attacked" is a clear advantage that we measure by the difference $\Gamma^{\mathbf{T}}_{\mathsf{opt}} - \Gamma^{\mathbf{T}}_{\mathsf{avg}} \geq 0$.

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* $\mathbf{D}_\ell$ *described in Alg. 8 such that for all simulators* σ *we have*

$$\Delta^{\mathbf{D}_\ell}\left(\mathsf{enc}^A \, \mathsf{dec}^B \, [\mathsf{KEY}^r_{\mathbf{T}}, \mathsf{AUT}^r], \ \sigma^E \, \mathsf{SEC}^r_{\mathbf{T}}\right) \quad \geq \quad \delta^{\mathbf{T}} - \frac{|\mathcal{K}|}{|\mathcal{M}_\ell|}, \tag{2}$$

where $\delta^{\mathbf{T}} := \Gamma^{\mathbf{T}}_{\mathsf{opt}} - \Gamma^{\mathbf{T}}_{\mathsf{avg}} \geq 0$*, with* $\Gamma^{\mathbf{T}}_{\mathsf{opt}}$ *and* $\Gamma^{\mathbf{T}}_{\mathsf{avg}}$ *defined in* (1)*.*

Remark 1. Before proving Theorem 1 let us comment on the lower bound obtained in (2). If there are almost as many keys as messages, such as in the one-time pad, then the bound in (2) becomes trivial. (Note that if the encryption scheme is the one-time pad, then password-based encryption is obviously possible since there is no "commitment" problem: doing the xor of a given ciphertext and of a given message leads to the appropriate key.) However, most encryption schemes used in practice have significantly less keys than messages and in this case the dominant term in (2) is $\delta^{\mathbf{T}}$, which quantifies for a trigger system $\mathbf{T}$ the advantage that can be obtained on average by knowing which session will be "attacked".

For example, $\delta^{\mathsf{GT}(\mathcal{P},q)}$ will be large for most distributions $\mathcal{P}$ of r *independent* passwords, since following the optimal strategy for session j, i.e., querying the q most likely passwords (for session j), prevents following the optimal guessing strategy for another session as the remaining number of passwords guessing queries will be less than q. Obviously, if the passwords are cryptographic keys, i.e., they are uniformly distributed over some large space, then the "passwords" are unguessable and $\delta^{\mathsf{GT}(\mathcal{P},q)}$ is negligible.

For the local password-guessing trigger $\mathsf{LT}(\mathcal{P}, \boldsymbol{q})$ we have $\delta^{\mathsf{LT}(\mathcal{P},\boldsymbol{q})} = 0$ for any number of password-guessing queries $q_1, \ldots, q_r$, since for this trigger each session is truly independent and the optimal strategy can be executed for each session.

Depending on the trigger system $\mathbf{T}$, the distinguisher $\mathbf{D}_\ell$ given in Alg. 8 may not necessarily be efficient, since it must be able to implement the optimal strategy for an arbitrary session of its choice. Nonetheless, it is worth mentioning that in the particular case where the trigger system is $\mathsf{GT}(\mathcal{P}, q)$, computing the optimal guessing strategy for a particular session is "easy" if the passwords are independently distributed. One simply queries the q most likely passwords for this session. Clearly, this assumes that the distinguisher $\mathbf{D}_\ell$ knows the password distribution $\mathcal{P}$, and is able to sort the passwords according to their respective probability to determine the q most likely for a particular session.

Proof (of Theorem 1). The main idea of the distinguisher $\mathbf{D}_\ell$ is to exploit the advantage quantified by $\delta^{\mathbf{T}}$ in knowing in advance which session $j^* \in \{1, \ldots, r\}$ is going to be "attacked". In order to have shorter notations within the proof, let $\mathbf{R}_\mathbf{T}$ denote the system $\mathsf{enc}^{\mathsf{A}}\,\mathsf{dec}^{\mathsf{B}}\,[\mathsf{KEY}^r_\mathbf{T}, \mathsf{AUT}^r]$, while $\mathbf{S}_\mathbf{T}$ denotes $\sigma^{\mathsf{E}}\,\mathsf{SEC}^r_\mathbf{T}$, for some simulator σ.

In a first step, the distinguisher $\mathbf{D}_\ell$ selects r messages of length ℓ uniformly at random, one for each session, and sends them at the A-interface to observe the corresponding ciphertexts $c_1, \ldots, c_r$ at the E_2-interface, which are real encryptions when interacting with $\mathbf{R}_\mathbf{T}$ or simulated encryptions when interacting with $\mathbf{S}_\mathbf{T}$ instead. The distinguisher $\mathbf{D}_\ell$ then selects uniformly at random a particular session $j^* \in \{1, \ldots, r\}$ and tries to break it by "running" the optimal strategy $\mathbf{D}_{\mathsf{opt}}(j^*)$ for that particular session, where

$$\mathbf{D}_{\mathsf{opt}}(j) \quad := \quad \arg\max_{\mathbf{D}\in\mathcal{D}} \; \Gamma^{\mathbf{T}}(j, \mathbf{D}),$$

for all $j \in \{1, \ldots, r\}$. "Running" $\mathbf{D}_{\mathsf{opt}}(j^*)$ implies the following three steps. First, the trigger value t output by $\mathbf{D}_{\mathsf{opt}}(j^*)$ is forwarded to the interface $\mathsf{E}_{1,\mathsf{S}}$, which when interacting with $\mathbf{R}_\mathbf{T}$ corresponds to querying the trigger $\mathbf{T}$ on t. Second, the distinguisher $\mathbf{D}_\ell$ outputs 1 if this query t resulted in breaking the session j^* and the key retrieved allows to decrypt the ciphertext c_{j^*} to the original message m_{j^*}. Otherwise, if the session j^* is still not broken, the distinguisher $\mathbf{D}_\ell$ queries the interface $\mathsf{E}_{1,\mathsf{N}}$ to retrieve a switch value $(s_1, \ldots, s_r)$, which when interacting with $\mathbf{R}_\mathbf{T}$ corresponds to the last switch value output by $\mathbf{T}$, and feed it to $\mathbf{D}_{\mathsf{opt}}(j^*)$ to obtain the next trigger value. The distinguisher $\mathbf{D}_\ell$ is given in more details in Alg. 8.

Alg. 8: Distinguisher $\mathbf{D}_\ell$

$m_j \xleftarrow{\$} \mathcal{M}_\ell$ and $c_j := \diamond$, for all $j\,\{1, \ldots, r\}$
$j^* := \diamond$
for $1 \le j \le r$ **do**
  $(j, c_j) :=$ **result at** E_2 **of querying** (j, m_j) at A
$j^* \xleftarrow{\$} \{1, \ldots, r\}$
return `BreakSession`(j^*)

Procedure `GetSwitch`
  $(s_1, \ldots, s_r) := 0^r$
  for $1 \le j \le r$ **do**
    $(j, k') :=$ **result of querying** (j, getkey) at $\mathsf{E}_{1,\mathsf{N}}$
    if $k' \neq \blacklozenge$ **then** $s_j := 1$
  return $(s_1, \ldots, s_r)$

Procedure `BreakSession`(j)
  $t :=$ 1st output of $\mathbf{D}_{\mathsf{opt}}(j)$
  while $t \neq \blacklozenge$ **do**
    output t at $\mathsf{E}_{1,\mathsf{S}}$
    $(s_1, \ldots, s_r) :=$ `GetSwitch`
    if $s_j = 1$ **then**
      $(j, k') :=$ **result of querying** (j, getkey) at $\mathsf{E}_{1,\mathsf{N}}$
      if $dec(k', c_j) = m_j$ **then return** 1
      else return 0
    $t :=$ **emulate result of querying** $\mathbf{D}_{\mathsf{opt}}(j)$ on $(s_1, \ldots, s_r)$
  return 0

We now lower bound the distinguishing advantage $\Delta^{\mathbf{D}_\ell}(\mathbf{R}_\mathbf{T}, \mathbf{S}_\mathbf{T})$. Note that when $\mathbf{D}_\ell$ interacts with $\mathbf{R}_\mathbf{T}$, the distinguisher breaks a given session j^* with probability exactly $\max_{\mathbf{D}\in\mathcal{D}} \Gamma^{\mathbf{T}}(j^*, \mathbf{D})$. If session j^* is

broken, then $\mathbf{D}_\ell$ always outputs 1 by correctness of the encryption scheme. Thus,

$$\mathsf{P}^{\mathbf{DR_T}}(1) \quad = \quad \frac{1}{r} \sum_{j^* \in \{1,\ldots,r\}} \max_{\mathbf{D} \in \mathcal{D}} \Gamma^{\mathbf{T}}(j^*, \mathbf{D}) \quad = \quad \Gamma^{\mathbf{T}}_{\mathsf{opt}} \ .$$

In contrast, when the distinguisher $\mathbf{D}_\ell$ interacts with $\mathbf{S_T}$, the event that matters is whether the simulator σ managed to break the session j^* before sending the ciphertext c_{j^*}, even though the session j^* to be broken will only be chosen afterwards by $\mathbf{D}_\ell$. In the random experiment $\mathbf{D}_\ell\mathbf{S_T}$, let $\mathcal{G}'_j$ be the event that the session j was broken at least once (the trigger $\mathbf{T}$ output at least once a switch value with $s_j = 1$) *before* the simulator σ outputs the simulated ciphertext c_j, for all $j \in \{1, \ldots, r\}$.

Note that if the simulator σ in $\mathbf{S}$ manages to break the chosen session j^* before having sent the ciphertext c_{j^*}, i.e., $\mathcal{G}'_{j^*} = 1$, then σ has the possibility of retrieving the corresponding message m_{j^*} and thus could potentially perfectly simulates this ciphertext, so that $\mathsf{P}^{\mathbf{D}_\ell\mathbf{S_T}}\left(1 \mid \mathcal{G}'_{j^*} = 1\right) = 1$ is possible. In contrast, when $\mathcal{G}'_{j^*} = 0$, the simulator σ is committed to the ciphertext c_{j^*} without knowing anything about the message m_{j^*} but its length ℓ. For the distinguisher $\mathbf{D}_\ell$ to output 1 in this case, the simulator must be able to provide a key k' such that $dec(k', c_{j^*}) = m_{j^*}$. Since the decryption algorithm dec is deterministic, it implies that for a fixed ciphertext c_{j^*} the output of $dec(\cdot, c_{j^*})$ can take at most $|\mathcal{K}|$ values. The message m_{j^*} is uniformly distributed over $\mathcal{M}_\ell$ and thus

$$\mathsf{P}^{\mathbf{D}_\ell\mathbf{S_T}}\left(1 \mid \mathcal{G}'_{j^*} = 0\right) \quad \leq \quad \frac{|\mathcal{K}|}{|\mathcal{M}_\ell|} \ .$$

Note that if the simulator σ can provoke the event $\mathcal{G}'_j = 1$ in the $\mathbf{D}_\ell\mathbf{S_T}$ random experiment, then σ can be used to win the event $\mathcal{G}_j = 1$ against the trigger $\mathbf{T}$ alone. Since σ does not know in advance which session is going to be selected by $\mathbf{D}_\ell$, its success is therefore at most the maximum of the average, i.e.,

$$\frac{1}{r} \sum_{j^* \in \{1,\ldots,r\}} \mathsf{P}^{\mathbf{D}_\ell\mathbf{S_T}}\left(\mathcal{G}'_{j^*} = 1\right) \quad \leq \quad \Gamma^{\mathbf{T}}_{\mathsf{avg}}.$$

Thus,

$$\begin{aligned}
\mathsf{P}^{\mathbf{D}_\ell\mathbf{S_T}}(1) \quad &= \quad \frac{1}{r} \sum_{j^* \in \{1,\ldots,r\}} \sum_{b \in \{0,1\}} \mathsf{P}^{\mathbf{D}_\ell\mathbf{S_T}}\left(\mathcal{G}'_{j^*} = b\right) \cdot \mathsf{P}^{\mathbf{D}_\ell\mathbf{S_T}}\left(1 \mid \mathcal{G}'_{j^*} = b\right) \\
&\leq \quad \frac{1}{r} \sum_{j^* \in \{1,\ldots,r\}} \mathsf{P}^{\mathbf{D}_\ell\mathbf{S_T}}\left(\mathcal{G}'_{j^*} = 1\right) + \frac{|\mathcal{K}|}{|\mathcal{M}_\ell|} \\
&\leq \quad \Gamma^{\mathbf{T}}_{\mathsf{avg}} + \frac{|\mathcal{K}|}{|\mathcal{M}_\ell|} \ .
\end{aligned}$$

Combining the previous equations gives the desired lower bound. □

6.3 PBE with Local Assumptions

As mentioned in Remark 1, 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 in the same way as in the case of a single session in Section 6.1.

We therefore assume in this section that the passwords are *independently*, but not necessarily identically, distributed. That is, there exist r password distributions $\mathcal{P}_1, \ldots, \mathcal{P}_r$ such that the distribution $\mathcal{P}$ of the r passwords can be written as $\mathcal{P}(w_1, \ldots, w_r) = \mathcal{P}_1(w_1) \times \cdots \times \mathcal{P}_r(w_r)$, for all $w_1, \ldots, w_r \in \mathcal{W}$. Such a distribution $\mathcal{P}$ will be called a *product distribution*.

Note that passwords in $\mathcal{W}$ can be sorted according to their likelihood of being chosen in the j^{th} session as given by $\mathcal{P}_j$. We denote by $\gamma_{j,q} \in [0, 1]$ the probability that the password selected in the j^{th} session belongs to the q most likely passwords (according to $\mathcal{P}_j$), for all $j \in \{1, \ldots, r\}$ and any integer q.

The following technical lemma shows that when constructing the downgradable secure channel $\mathsf{SEC}^r_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})}$ from the downgradable key $\mathsf{KEY}^r_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})}$ where the switch value can be changed through the trigger system $\mathsf{LT}(\mathcal{P},\boldsymbol{q})$, it is sufficient to instead look at systems $\mathsf{KEY}^r_{S^*}$ and $\mathsf{SEC}^r_{S^*}$ where the switch value is randomly chosen and fixed at the beginning according to some random variable S^*.

Lemma 2. *Let $\mathcal{P}$ be a product distribution of r passwords as described above and consider a tuple of r integers $\boldsymbol{q} := (q_1, \ldots, q_r)$. For each session $j \in \{1, \ldots, r\}$, let S^*_j be a binary random variable which is 1 with probability γ_{j,q_j}, and let $S^* := (S^*_1, \ldots, S^*_r)$. Then, there exist two converters σ_{KEY} and σ_{SEC} described in Alg. 9 and 10, respectively, such that*

$$\mathsf{KEY}^r_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})} \equiv \sigma^{\mathsf{E}}_{\mathsf{KEY}}\,\mathsf{KEY}^r_{S^*} \qquad \textit{and} \qquad \mathsf{SEC}^r_{S^*} \equiv \sigma^{\mathsf{E}}_{\mathsf{SEC}}\,\mathsf{SEC}^r_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})} \ .$$

Proof. The core systems KEY^r and SEC^r introduced in Section 3.2 are such that a change in their switch value modifies the behavior of the system only at the adversarial $\mathsf{E_N}$ interface and not at the others (interface A or B).

In the case of the key system, the converter σ_{KEY} has to be indistinguishable from $\mathsf{KEY}^r_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})}$, where the value of the switch can be changed progressively, by having only access to $\mathsf{KEY}^r_{S^*}$, where the value of the switch is (randomly) fixed at the beginning. To do so, the converter σ_{KEY} uses the fact it can easily infer the value of S^* by simply querying (j, getkey) at the $\mathsf{E_N}$-interface of $\mathsf{KEY}^r_{S^*}$. If the answer contains the error symbol $\blacklozenge$, then $S^*_j = 0$, and otherwise $S^*_j = 1$, for all $j \in \{1, \ldots, r\}$. If $S^*_j = 0$, then σ_{KEY} cannot retrieve the key associated to this session and therefore declares any password guess for that session as invalid. In contrast, if $S^*_j = 1$, which happens with probability γ_{j,q_j}, the converter σ_{KEY} must then "scale up" the password-guessing probabilities in order to match that of $\mathsf{KEY}^r_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})}$. A password guess w for a session where $S^*_j = 1$ is considered valid if a coin tossed with probability $\frac{\mathcal{P}_j(w)}{\gamma_{j,q_j}}$ is 1. Note that this is truly a probability distribution since γ_{j,q_j} represents the winning probability of the optimal guessing strategy for the j^{th} session with q_j queries and therefore $\mathcal{P}_j(w) \leq \gamma_{j,q_j}$, for all password guesses $w \in \mathcal{W}$. Overall, the probability of retrieving the key used in the j^{th} session is therefore $\mathcal{P}_j(w)$ and thus $\mathsf{KEY}^r_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})} \equiv \sigma^{\mathsf{E}}_{\mathsf{KEY}}\,\mathsf{KEY}^r_{S^*}$. A more formal description of σ_{KEY} is in Alg. 9.

Alg. 9: Converter σ_{KEY}

$s_j := 0$ and $k_j := \blacklozenge$, for all $j \in \{1, \ldots, r\}$
on input (j, w) at $\mathsf{out_S}$
  $(j, k) :=$ **result of querying** (j, getkey) at $\mathsf{in_N}$
  if $k \neq \blacklozenge$ **then**
    flip a bit b with probability $\frac{\mathcal{P}_j(w)}{\gamma_{j,q_j}}$
    $s_j := s_j \vee b$ and $k_j := k$
on input (j, getkey) at $\mathsf{out_N}$
  if $s_j = 1$ **then output** (j, k_j) at $\mathsf{out_N}$
  else output $(j, \blacklozenge)$ at $\mathsf{out_N}$

Alg. 10: Converter σ_{SEC}

$\mathsf{bforced}_j := 0$ for all $j \in \{1, \ldots, r\}$
on input $(j, |m|)$ at $\mathsf{in_N}$
  if $\mathsf{bforced}_j = 0$ **then** `BruteForce`(j, q_j)
  output $(j, |m|)$ at $\mathsf{in_N}$
on input (j, getmsg) at $\mathsf{out_N}$
  if $\mathsf{bforced}_j = 0$ **then** `BruteForce`(j, q_j)
  $(j, m') :=$ **result of querying** (j, getmsg) at $\mathsf{in_N}$
  output (j, m') at $\mathsf{out_N}$
Procedure `BruteForce`(j, q)
  for $1 \leq k \leq q$ **do**
    $w_k := k^{\text{th}}$ most likely password according to $\mathcal{P}_j$
    output (j, w_k) at $\mathsf{in_S}$
  $\mathsf{bforced}_j := 1$

In contrast to σ_{KEY}, the task of σ_{SEC} is to emulate $\mathsf{SEC}^r_{S^*}$, where the value of the switch is (randomly) fixed at the beginning, from $\mathsf{SEC}^r_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})}$, where its value can be changed progressively. To do so, σ_{SEC} simply does the optimal password-guessing strategy, i.e., σ_{SEC} queries the q_j most likely passwords for the j^{th} session, for all $j \in \{1, \ldots, r\}$, and hence $\mathsf{SEC}^r_{S^*} \equiv \sigma^{\mathsf{E}}_{\mathsf{SEC}}\,\mathsf{SEC}^r_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})}$. A more detailed description of σ_{SEC} is in Alg. 10. □

Remark 2. We point out that the strategies described by the converters σ_{KEY} or σ_{SEC} in Lemma 2 have demanding preconditions: They need to know the password distribution and, for each session, to also know

an upper bound on the number of password-guessing queries as well as to be able to sort through the set of passwords in order to determine the most likely ones.

The next lemma shows that the protocol $(\mathsf{enc}, \mathsf{dec})$ constructs a secure channel SEC_S^r from a key resource KEY_S^r and authenticated communication AUT^r if the underlying encryption scheme is IND-CPA secure, for every random switch value S over $\{0,1\}^r$.

Lemma 3. *Let* $\mathrm{SE} := (enc, dec)$ *be a correct encryption scheme and consider the associated protocol* $\mathsf{SE} := (\mathsf{enc}, \mathsf{dec})$*. Let* S *be a random variable arbitrarily distributed over* $\{0,1\}^r$*. Then, there exist a simulator* σ_{CPA} *described in Alg. 11 and an (efficient) reduction system* $\mathbf{C}^{\mathsf{CPA}}$ *described in the proof such that*

$$[\mathsf{KEY}_S^r, \mathsf{AUT}^r] \quad \overset{(\mathsf{SE}, \sigma_{\mathsf{CPA}}, \varepsilon)}{\Longrightarrow} \quad \mathsf{SEC}_S^r,$$

where $\varepsilon(\mathbf{D}) := r \cdot \Delta^{\mathbf{D}\mathbf{C}^{\mathsf{CPA}}}\left(\mathbf{G}_0^{\mathsf{CPA}}(\mathrm{SE}), \mathbf{G}_1^{\mathsf{CPA}}(\mathrm{SE})\right)$*, for all distinguishers* $\mathbf{D}$*.*

Proof. Let S be a random switch value arbitrarily distributed over $\{0,1\}^r$. Although the simulator σ_{CPA} does a priori not know the value of the random switch S, it can easily retrieve it by querying the channel SEC_S^r on (j, getmsg). If the answer contains the error symbol $\blacklozenge$, then $S_j = 0$, otherwise $S_j = 1$, for all $j \in \{1, \ldots, r\}$. If $S_j = 0$, then the simulator σ_{CPA} will never be able to retrieve the transmitted message of the j^{th} session and simulates therefore the transmission of a ciphertext by encrypting a random message of the correct length under a random key. For such a session, any key retrieval query (j, getkey) will be responded with the error symbol $\blacklozenge$. In the other case, if $S_j = 1$, the simulator σ_{CPA} can simply retrieve the transmitted message and encrypt it under a uniform random key to simulate perfectly the sent ciphertext. For such a session, the actual key used for simulating the encryption will be leaked if the simulator σ_{CPA} receives a key retrieval query. The simulator σ_{CPA} is described more precisely in Alg. 11.

Alg. 11: Simulator σ_{CPA}

$k_1, \ldots, k_r \xleftarrow{\$} \{0,1\}^n$
on input $(j, |m|)$ at $\mathsf{in_N}$
  $\left(j, m'_j\right) :=$ **result of querying** (j, getmsg) at $\mathsf{in_N}$
  if $m'_j = \blacklozenge$ **then**
    $m'_j \xleftarrow{\$} \{0,1\}^{|m|}$
  $c \leftarrow enc\left(k_j, m'_j\right)$
  output (j, c) at out_2
on input (j, getkey) at $\mathsf{out_{1,N}}$
  $\left(j, m'_j\right) :=$ **result of querying** (j, getmsg) at $\mathsf{in_N}$
  if $m'_j = \blacklozenge$ **then output** $(j, \blacklozenge)$ at $\mathsf{out_{1,N}}$
  else output (j, k_j) at $\mathsf{out_{1,N}}$

Alg. 12: Reduction $\mathbf{C}_{j^*}^{\mathsf{CPA}}$

$k_1, \ldots, k_r \xleftarrow{\$} \{0,1\}^n$
$(s_1, \ldots, s_r) \leftarrow \mathsf{P}_S$
on input (j, m) at A
  $c := \diamond$
  if $s_j = 0$ **then**
    if $j < j^*$ **then**
      $c \leftarrow enc(k_j, m_1)$, where $m_1 \xleftarrow{\$} \mathcal{M}_{|m|}$
    else if $j = j^*$ **then**
      $c :=$**result of querying** m at in
    else
      $c \leftarrow enc(k_j, m)$
  else $c \leftarrow enc(k_j, m)$
  output (j, m) at B
  output (j, c) at E
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 $\mathsf{E_N}$

In order to have shorter notations within the proof, let us denote by $\mathbf{R}_S$ and $\mathbf{S}_S$ the real system $\mathsf{enc}^{\mathsf{A}}\mathsf{dec}^{\mathsf{B}}[\mathsf{KEY}_S^r, \mathsf{AUT}^r]$ and the ideal system $\sigma^{\mathsf{E}}\mathsf{SEC}_S^r$, respectively. Note that for sessions which are "broken" $(S_j = 1)$ the simulation is perfect, while for the other sessions the only difference between the two systems $\mathbf{R}_S$ and $\mathbf{S}_S$ lies in the way ciphertexts are produced: $\mathbf{R}_S$ produces encryptions of messages which were input at interface A, whereas $\mathbf{S}_S$ produces encryptions of random messages of the same length as messages

input at A. Distinguishing $\mathbf{R}_S$ from $\mathbf{S}_S$ can therefore be reduced to the IND-CPA security of the underlying encryption scheme via a hybrid argument as follows.

We define a sequence of reduction systems $\mathbf{C}_1^{\mathsf{CPA}}, \ldots, \mathbf{C}_r^{\mathsf{CPA}}$, where each reduction $\mathbf{C}_{j^*}^{\mathsf{CPA}}$ has 4 outside sub-interfaces (labeled $\mathsf{A}, \mathsf{B}, \mathsf{E}_{1,\mathsf{N}}$, and E_2) and connects at its inside interface in to one of the two IND-CPA systems $\mathbf{G}_0^{\mathsf{CPA}}(\mathrm{SE})$ or $\mathbf{G}_1^{\mathsf{CPA}}(\mathrm{SE})$ described in Alg. 2, for all $j^* \in \{1, \ldots, r\}$. Initially, the reduction system $\mathbf{C}_{j^*}^{\mathsf{CPA}}$ selects r keys $k_1, \ldots, k_r$ uniformly at random, as well as a switch value $(s_1, \ldots, s_r)$ according to P_S, the distribution of the random switch S. Upon input (j, m) at its outside sub-interface A, the reduction system $\mathbf{C}_{j^*}^{\mathsf{CPA}}$ outputs (j, m) at its outside sub-interface B and (j, c) at E_2, where the ciphertext c is computed as follows. If session j is "broken" ($s_j = 1$), then c is a local encryption of the input message m under key k_j. Otherwise, if session j is not "broken" ($s_j = 0$), then c is a local encryption of a fresh random message of length $|m|$ under key k_j for session $j < j^*$, while for session $j > j^*$ the ciphertext c is an actual encryption of the input message under key k_j. Finally, only for an unbroken session $j = j^*$ is the ciphertext c the result of querying the system connected at its inside interface on the message m. Key retrieval queries (j, getkey) made to the outside sub-interface $\mathsf{E}_{1,\mathsf{N}}$ of $\mathbf{C}_{j^*}^{\mathsf{CPA}}$ are replied by $(j, \blacklozenge)$ or (j, k_j) according to the value of s_j. The reduction system $\mathbf{C}_{j^*}^{\mathsf{CPA}}$ is described more formally in Alg. 12.

Note that ciphertexts produce by $\mathbf{C}_1^{\mathsf{CPA}}\mathbf{G}_0^{\mathsf{CPA}}(\mathrm{SE})$ are always encryptions of messages which were input at interface A and thus $\mathbf{R}_S \equiv \mathbf{C}_1^{\mathsf{CPA}}\mathbf{G}_0^{\mathsf{CPA}}(\mathrm{SE})$. Similarly, ciphertexts produce by $\mathbf{C}_r^{\mathsf{CPA}}\mathbf{G}_1^{\mathsf{CPA}}(\mathrm{SE})$ are encryptions of random messages of the same length as messages input at A, unless they are for a "broken" session in which case the actual input message is encrypted, and thus $\mathbf{S}_S \equiv \mathbf{C}_r^{\mathsf{CPA}}\mathbf{G}_1^{\mathsf{CPA}}(\mathrm{SE})$. Furthermore, note that unless a session is broken, both $\mathbf{C}_{j+1}^{\mathsf{CPA}}\mathbf{G}_0^{\mathsf{CPA}}(\mathrm{SE})$ and $\mathbf{C}_j^{\mathsf{CPA}}\mathbf{G}_1^{\mathsf{CPA}}(\mathrm{SE})$ produce encryptions of random messages of the appropriate length until the j^{th} session, while ciphertexts for sessions $j+1, \ldots, r$ are actual encryptions of input messages. Thus, the sequence of reduction systems $\mathbf{C}_1^{\mathsf{CPA}}, \ldots, \mathbf{C}_r^{\mathsf{CPA}}$ is such that $\mathbf{C}_{j+1}^{\mathsf{CPA}}\mathbf{G}_0^{\mathsf{CPA}}(\mathrm{SE}) \equiv \mathbf{C}_j^{\mathsf{CPA}}\mathbf{G}_1^{\mathsf{CPA}}(\mathrm{SE})$, for all $j \in \{1, \ldots, r-1\}$. For a reduction system $\mathbf{C}^{\mathsf{CPA}}$ which selects a session $j^* \in \{1, \ldots, r\}$ uniformly at random and then implements the reduction $\mathbf{C}_{j^*}^{\mathsf{CPA}}$ we have

$$\Delta^{\mathbf{D}\mathbf{C}^{\mathsf{CPA}}}\left(\mathbf{G}_0^{\mathsf{CPA}}(\mathrm{SE}), \mathbf{G}_1^{\mathsf{CPA}}(\mathrm{SE})\right) \quad = \quad \frac{1}{r} \cdot \Delta^{\mathbf{D}}\left(\mathbf{R}_S, \mathbf{S}_S\right),$$

for every distinguisher $\mathbf{D}$. □

The previous lemmas allow us to easily prove the main result of this section, which is that password-based encryption with these additional local assumptions is possible if the encryption scheme used is IND-CPA secure and the r passwords are independently distributed.

Theorem 2. *Let* $\mathrm{SE} := (enc, dec)$ *be a correct encryption scheme and consider the associated protocol* $\mathsf{SE} := (\mathsf{enc}, \mathsf{dec})$. *For every product distribution* $\mathcal{P}$ *of* r *passwords and every tuple of* r *integers* $\boldsymbol{q} := (q_1, \ldots, q_r)$, *there exist a simulator* σ_{LT} *and an (efficient) reduction system* $\mathbf{C}^{\mathsf{LT}}$ *described in the proof such that*

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

where $\varepsilon(\mathbf{D}) := r \cdot \Delta^{\mathbf{D}\mathbf{C}^{\mathsf{LT}}}\left(\mathbf{G}_0^{\mathsf{CPA}}(\mathrm{SE}), \mathbf{G}_1^{\mathsf{CPA}}(\mathrm{SE})\right)$, *for every distinguisher* $\mathbf{D}$.

Proof. Consider a product distribution $\mathcal{P}$ of r passwords and a tuple of r integers $\boldsymbol{q} := (q_1, \ldots, q_r)$. In order to have shorter notations within the proof, let $\mathbf{R}_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})}$ denote the downgradable resource $\left[\mathsf{KEY}^r_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})}, \mathsf{AUT}^r\right]$ and let $\mathbf{S}_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})}$ denote $\mathsf{SEC}^r_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})}$. Given the encryption protocol $\mathsf{SE} := (\mathsf{enc}, \mathsf{dec})$ and some resource $\mathbf{U}$, it will convenient to write $\mathsf{SE}\,\mathbf{U} := \mathsf{enc}^{\mathsf{A}}\,\mathsf{dec}^{\mathsf{B}}\,\mathbf{U}$.

Then, Lemma 2 implies that there exist two converters σ_{KEY} and σ_{SEC}, as well as a random variable S^* over $\{0,1\}^r$, such that

$$\mathbf{R}_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})} \quad \equiv \quad \sigma'^{\mathsf{E}}_{\mathsf{KEY}}\,\mathbf{R}_{S^*} \qquad \text{and} \qquad \mathbf{S}_{S^*} \quad \equiv \quad \sigma^{\mathsf{E}}_{\mathsf{SEC}}\,\mathbf{S}_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})},$$

where $\sigma'_{\mathsf{KEY}} := \langle \sigma_{\mathsf{KEY}}, \mathsf{id} \rangle$ and accounts for the presence in parallel of the channel AUT^r in $\mathbf{R}$. Lemma 3 implies then that for such a random switch value S^*, there exist a simulator σ_{CPA} and a reduction $\mathbf{C}^{\mathsf{CPA}}$, such that for all distinguishers $\mathbf{D}$

$$\Delta^{\mathbf{D}}\left(\mathsf{SE}\,\mathbf{R}_{S^*}, \sigma^{\mathsf{E}}_{\mathsf{CPA}}\,\mathbf{S}_{S^*}\right) \quad \leq \quad r \cdot \Delta^{\mathbf{D}\mathbf{C}^{\mathsf{CPA}}}\left(\mathbf{G}_0^{\mathsf{CPA}}(\mathrm{SE}), \mathbf{G}_1^{\mathsf{CPA}}(\mathrm{SE})\right) \quad .$$

The proof then follows by composition. To see that, note that the previous system equation implies that

$$\mathsf{SE}\,\mathbf{R}_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})} \quad \equiv \quad \mathsf{SE}\,\sigma'^{\mathsf{E}}_{\mathsf{KEY}}\,\mathbf{R}_{S^*} \quad \equiv \quad \sigma'^{\mathsf{E}}_{\mathsf{KEY}}\,\mathsf{SE}\,\mathbf{R}_{S^*},$$

where we used the fact that converters connected at different interfaces commute. Let the simulator σ_{LT} be defined as the sequential composition of the aforementioned simulators, i.e., $\sigma_{\mathsf{LT}} := \sigma'_{\mathsf{KEY}} \circ \sigma_{\mathsf{CPA}} \circ \sigma_{\mathsf{SEC}}$. Then, we have $\sigma_{\mathsf{LT}}{}^{\mathsf{E}}\,\mathbf{S}_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})} \equiv \sigma'^{\mathsf{E}}_{\mathsf{KEY}}\,\sigma^{\mathsf{E}}_{\mathsf{CPA}}\,\mathbf{S}_{S^*}$, and thus

$$\Delta^{\mathbf{D}}\left(\mathsf{SE}\,\mathbf{R}_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})}, \sigma_{\mathsf{LT}}{}^{\mathsf{E}}\,\mathbf{S}_{\mathsf{LT}(\mathcal{P},\boldsymbol{q})}\right) \quad \leq \quad \Delta^{\mathbf{D}'}\left(\mathsf{SE}\,\mathbf{R}_{S^*}, \sigma^{\mathsf{E}}_{\mathsf{CPA}}\,\mathbf{S}_{S^*}\right),$$

where $\mathbf{D}' := \mathbf{D}\sigma'^{\mathsf{E}}_{\mathsf{KEY}}$. The proof is then finished by combining both distinguishing advantages and defining the overall reduction $\mathbf{C}^{\mathsf{LT}}$ as $\mathbf{C}^{\mathsf{LT}} := \sigma'^{\mathsf{E}}_{\mathsf{KEY}}\mathbf{C}^{\mathsf{CPA}}$. □

Remark 3. The simulation strategy employed by σ_{LT} in the construction of Theorem 2 is rather peculiar. Informally stated, whenever σ_{LT} has to simulate a ciphertext for a particular session, it first tries to brute force the password of this session by probing the most likely passwords (this step is taken care of by σ_{SEC} in Alg. 10). If this step succeeds, σ_{LT} can perfectly simulate the transmitted ciphertext, otherwise it simply encrypts a random message of the appropriate length (this step is taken care of by σ_{CPA} in Alg. 11). Finally, even though σ_{LT} exhausts its password-guessing queries at the beginning, it can emulate the correct answer to an outside password-guessing query by appropriately scaling the probabilities (this step is taken care of by σ_{KEY} in Alg. 9). Note that the assumptions discussed in Remark 2 are also required for σ_{LT}.

6.4 Salting and Per-Session Security

In this section we examine 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 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.

Assumed resources. The construction statement below assumes the availability of a single random oracle and the strings used for salting. The random oracle RO_q is restricted by allowing Eve to ask at most q queries. The source of salts is formalized as a resource SALT and parametrized by a distribution $\mathcal{R}$ of r *distinct* t-bit strings, for some appropriate integer t. The resource $\mathsf{SALT}\,(\mathcal{R})$ first samples $(v_1, \ldots, v_r)$ from $\mathcal{R}$ and then outputs (j, v_j) at interface $i \in \{\mathsf{A}, \mathsf{B}, \mathsf{E}\}$ whenever it receives a query $(j, \mathsf{getsalt})$ at the same interface, for any $j \in \{1, \ldots, r\}$. Note that the prefixes used in each session are public and can be retrieved by Eve.

The prefixing protocol. Both Alice and Bob use the bit strings provided by the salting resource to prefix their queries as described by the following converter pre: Upon input (j, x) at its interface out, corresponding to hashing a message $x \in \{0,1\}^*$ for the j^{th} session, the converter pre retrieves the prefix v_j associated with that session by querying $\mathsf{SALT}\,(\mathcal{R})$ on $(j, \mathsf{getsalt})$ and then outputs the value returned by the single random oracle when queried on $v_j \,\|\, x$.

Desired resource. The goal of the prefixing protocol $\mathsf{PRE} := (\mathsf{pre}, \mathsf{pre})$ is to construct r independent random oracles. Since the assumed random oracle RO_q can be queried by Eve at most q times, the constructed resource will naturally have a similar restriction. Namely, the following lemma shows that the prefixing protocol PRE constructs r *globally* restricted random oracles $[\mathsf{RO}, \ldots, \mathsf{RO}]_q$.

Lemma 4. *Consider the prefixing protocol* $\mathsf{PRE} := (\mathsf{pre}, \mathsf{pre})$ *described above. Then, for every distribution* $\mathcal{R}$ *of* r *distinct bit strings of equal length and every integer* q*, there exists a simulator* σ_{pre} *such that*

$$[\mathsf{RO}_q, \mathsf{SALT}\,(\mathcal{R})] \quad \overset{(\mathsf{PRE},\, \sigma_{\mathsf{pre}},\, 0)}{\Longrightarrow} \quad [\mathsf{RO}, \ldots, \mathsf{RO}]_q \quad .$$

Proof. The simulator σ_{pre} first selects r bit strings $(v_1, \ldots, v_r)$ of equal length according to the distribution $\mathcal{R}$. Then, whenever a salt retrieval query $(j, \mathsf{getsalt})$ for the j^{th} session is input at its interface out_2, the simulator σ_{pre} returns (j, v_j) at the same interface. When a message $x \in \{0,1\}^*$ to be hashed is input at its interface out_1, the simulator σ_{pre} first checks whether one of the strings v_j is a prefix of x, i.e., whether $x = v_j \,\|\, x'$ for some $j \in \{1, \ldots, r\}$ and $x' \in \{0,1\}^*$. If this is the case, then σ_{pre} returns the answer of the j^{th} random oracle when queried on x'. Otherwise, the simulator σ_{pre} simply returns a uniform n-bit string. The simulator σ_{pre} replies consistently to any repeating query and in addition keeps track of the number of queries to avoid too many "dummy" queries, i.e., queries which are not prefixed by one of the strings v_j. It can be readily verified that the simulation is perfect. □

Unfortunately, it is easy to show that the same salting protocol PRE 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 $j \in \{1, \ldots, r\}$ (the latter would render this construction uninteresting due to the blow-up in the number of adversarial queries). To see this, consider the systems $\mathbf{R} := \mathsf{pre}^{\mathsf{A}}\, \mathsf{pre}^{\mathsf{B}}\, [\mathsf{RO}_q, \mathsf{SALT}(\mathcal{R})]$ and $\mathbf{S} := \sigma^{\mathsf{E}}\, [\mathsf{RO}_{q_1}, \ldots, \mathsf{RO}_{q_r}]$ for some arbitrary simulator σ. In the real experiment, a distinguisher making a query (j, x) to the interface A and a query $(j, v_j \| x)$ to the interface E_1 of $\mathbf{R}$, where v_j is the prefix for the j^{th} session output by the salting resource, observes the same output. Hence, in the ideal experiment, for any query of the type $(j, v_j \| x)$ the simulator σ has to query the j^{th} random oracle RO_{q_j} on x in order to be able to mimic this behavior. Thus, a distinguisher can choose any session j and make all its q queries to the associated random oracle RO_{q_j} by appropriately prefixing each query, hence requiring $q_j \geq q$ for any successful simulator to exist.

Consequences for local restrictions. The above observation implies that relying on *local* query restrictions for multi-session security of password-based encryption (as in Theorem 2 and [BRT12]) appears to be in general rather unrealistic. The salting technique employed in PKCS #5 [Kal00] (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 $q_1, \ldots, q_r$.

6.5 On the Per-Session Security of PBE from PKCS #5

In this section we show how the arguments used to prove Theorem 1 also apply to the password-based encryption standard described in PKCS #5 [Kal00], which thus in general does not achieve per-session confidentiality.

Recall that the protocol described in [Kal00] consists of hashing a salt concatenated with a password and of using the result as a key for encryption.[8] Formalized as a pair of converters $(\mathsf{pbe}, \mathsf{pbd})$, this protocol corresponds to first doing the salting step via the protocol $(\mathsf{pre}, \mathsf{pre})$ described in Section 6.4, then the key derivation protocol $(\mathsf{kd}, \mathsf{kd})$ described in Section 4 and finally symmetric encryption $(\mathsf{enc}, \mathsf{dec})$ described in Section 6. As detailed in previous sections, such a protocol assumes the following resources to be available: a random oracle RO_q, a source of salts $\mathsf{SALT}(\mathcal{R})$, a source of shared passwords $\mathsf{PW}(\mathcal{P})$, and finally an authenticated communication channel AUT^r for each of the r sessions. We denote by $\mathsf{RKDF}(q, \mathcal{R}, \mathcal{P})$ the resources used for key derivation, i.e.,

$$\mathsf{RKDF}(q, \mathcal{R}, \mathcal{P}) \quad := \quad [\mathsf{RO}_q, \mathsf{SALT}(\mathcal{R}), \mathsf{PW}(\mathcal{P})] \ .$$

Theorem 3. *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 combined protocol* $(\mathsf{pbe}, \mathsf{pbd})$ *described above. Let* $\mathbf{T}$ *be a trigger system with input space* $\{1, \ldots, r\} \times \mathcal{W}$ *and 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, for every integer* q*, every distribution* $\mathcal{R}$ *of* r *distinct bit strings of the same length and every distribution* $\mathcal{P}$ *of* r *passwords, there exists a distinguisher* $\mathbf{D}'_\ell$ *described in Alg. 13 such that for all simulators* σ *we have*

$$\Delta^{\mathbf{D}'_\ell}\left(\mathsf{pbe}^{A}\, \mathsf{pbd}^{B}\, [\mathsf{RKDF}(q, \mathcal{R}, \mathcal{P}), \mathsf{AUT}^r],\ \sigma^{E}\, \mathsf{SEC}^r_{\mathbf{T}}\right) \quad \geq \quad \delta^{\mathsf{GT}(\mathcal{P},q)}_{\mathbf{T}} - \mu,$$

[8] Additionally, it also increases the cost of a brute-force attack by iterative hashing. This is however an orthogonal issue, see [DGMT15] for a detailed discussion.

where $\delta_{\mathbf{T}_1}^{\mathbf{T}_2} := \Gamma_{\mathsf{opt}}^{\mathbf{T}_2} - \Gamma_{\mathsf{avg}}^{\mathbf{T}_1}$, *with* $\Gamma_{\mathsf{opt}}^{\mathbf{T}_2}$ *and* $\Gamma_{\mathsf{avg}}^{\mathbf{T}_1}$ *defined in* (1), *and*

$$\mu \quad \leq \quad r \cdot \Delta^{\mathbf{D}_{\ell,q}^{\mathsf{CPA}}}\left(\mathbf{G}_0^{\mathsf{CPA}}(\mathrm{SE}), \mathbf{G}_1^{\mathsf{CPA}}(\mathrm{SE})\right) + (r+1) \cdot \frac{|\mathcal{K}|}{|\mathcal{M}_\ell|},$$

for a distinguisher $\mathbf{D}_{\ell,q}^{\mathsf{CPA}}$ *described in Alg. 14.*

The proof is analogous to that of Theorem 1 and consequently the same limitations mentioned in Remark 1 also apply to Theorem 3. Indeed, if the encryption scheme used is IND-CPA secure and uses significantly less keys than messages, then the dominant term in the lower bound above is $\delta_{\mathbf{T}}^{\mathsf{GT}(\mathcal{P},q)}$ which may become moot depending on the trigger $\mathbf{T}$ considered in the constructed resource. However, in the case where the passwords are independently distributed and $\mathbf{T} := \mathsf{GT}(\mathcal{P}, p)$, the quantity $\delta_{\mathbf{T}}^{\mathsf{GT}(\mathcal{P},q)}$ will be large for most password distributions until $p < rq$, since following the optimal guessing strategy for a particular session usually requires to exhaust all q guesses. Likewise, if instead $\mathbf{T} := \mathsf{LT}(\mathcal{P}, \boldsymbol{q})$, for some integers $\boldsymbol{q} := (q_1, \ldots, q_r)$, then $\delta_{\mathbf{T}}^{\mathsf{GT}(\mathcal{P},q)}$ will be large if $q_j < q$ for some $j \in \{1, \ldots, r\}$.

Proof (of Theorem 3). The main idea is to use the same strategy as the distinguisher $\mathbf{D}_\ell$ in Alg. 8 which, very roughly, consists of the following three steps: 1) input r messages $m_1, \ldots, m_r$ of length ℓ chosen uniformly at random; 2) observe the corresponding ciphertexts $c_1, \ldots, c_r$; and 3) "run" the optimal strategy to break a session j^* chosen uniformly at random. We need to adapt this last step since the resources considered, contrary to those in Theorem 1, do not have a switch value s_j indicating whether session j is broken.

In order to have shorter notations within the proof, the real system $\mathsf{pbe}^{\mathsf{A}}\,\mathsf{pbd}^{\mathsf{B}}\,[\mathsf{RKDF}(q, \mathcal{R}, \mathcal{P}), \mathsf{AUT}^r]$ will be denoted by $\mathbf{R}$ and the ideal system $\sigma^{\mathsf{E}}\,\mathsf{SEC}_{\mathbf{T}}^r$ will be denoted by $\mathbf{S}_{\mathbf{T}}$, for some simulator σ and some $(\{1, \ldots, r\} \times \mathcal{W}, \{0,1\}^r)$-trigger system $\mathbf{T}$. Eve's interface in $\mathbf{R}$ is split as follows: sub-interfaces $\mathsf{E}_{1,1}, \mathsf{E}_{1,2}$ and $\mathsf{E}_{1,3}$ denote the respective E-interface of the resources RO_q, $\mathsf{SALT}(\mathcal{R})$ and $\mathsf{PW}(\mathcal{P})$ in the combined resource $\mathsf{RKDF}(q, \mathcal{R}, \mathcal{P})$; while the sub-interface E_2 of $\mathbf{R}$ denotes the E-interface of the authenticated channel AUT^r.

Similarly to $\mathbf{D}_\ell$, the new distinguisher $\mathbf{D}'_\ell$ "runs" the optimal strategy $\mathbf{D}_{\mathsf{opt}}(j^*)$ for guessing the password used in a randomly chosen session j^*, where

$$\mathbf{D}_{\mathsf{opt}}(j) \quad := \quad \arg\max_{\mathbf{D} \in \mathcal{D}} \; \Gamma^{\mathsf{GT}(\mathcal{P},q)}(j, \mathbf{D}),$$

for all $j \in \{1, \ldots, r\}$. Note that $\mathbf{D}_{\mathsf{opt}}(j^*)$ expects as input an r-bit string indicating the "broken" state of each session. To do so, the distinguisher $\mathbf{D}'_\ell$ keeps a state $(s_1, \ldots, s_r)$ which is updated as follows. Given a password guess (j, w) for session j (which could a priori be different from j^*), the new distinguisher $\mathbf{D}'_\ell$ declares session j "broken" if the output k of the random oracle when queried on $v_j \,\|\, w$, where v_j is the prefix used for session j, is such that $dec(k, c_j) = m_j$. The distinguisher $\mathbf{D}'_\ell$ is given in more details in Alg. 13.

Alg. 13: Distinguisher $\mathbf{D}'_\ell$

$m_j \xleftarrow{\$} \mathcal{M}_\ell$, $c_j := \diamond$ and $s_j := 0$, for all $j \{1, \ldots, r\}$
$j^* := \diamond$
for $1 \leq j \leq r$ **do**
  $(j, c_j) :=$ **result at** E_2 **of querying** (j, m_j) at A
$j^* \xleftarrow{\$} \{1, \ldots, r\}$
return `BreakSession`(j^*)

Procedure `BreakSession`(j)
  $(j', w) :=$ 1st output of $\mathbf{D}_{\mathsf{opt}}(j)$
  while $(j', w) \neq \blacklozenge$ **do**
    $v_{j'} :=$**result of querying** $(j', \mathsf{getsalt})$ at $\mathsf{E}_{1,2}$
    $k_{j'} :=$**result of querying** $v_{j'} \,\|\, w$ at $\mathsf{E}_{1,1}$
    if $dec(k_{j'}, c_{j'}) = m_{j'}$ **then** $s_{j'} := 1$
    if $s_j = 1$ **then return** 1
    $t :=$ **emulate result of querying** $\mathbf{D}_{\mathsf{opt}}(j)$ on $(s_1, \ldots, s_r)$
  return 0

We now lower bound the distinguishing advantage $\Delta^{\mathbf{D}'_\ell}(\mathbf{R}, \mathbf{S}_{\mathbf{T}})$. Let $\mathcal{G}_j$ be the event that the password used in session j was correctly guessed in the $\mathbf{D}'_\ell\mathbf{R}$ random experiment, i.e., a query prefixed by v_j made to the random oracle at interface E matches a previous query made to the random oracle at interface A or B,

where v_j is is the prefix used for session j output by the salting resource $\mathsf{SALT}(\mathcal{R})$. If the distinguisher $\mathbf{D}'_\ell$ manages to guess the password of a given session j^*, it will necessarily output 1 due to the correctness of the encryption scheme, i.e.,

$$\mathsf{P}^{\mathbf{D}'_\ell\mathbf{R}}(1) \quad \geq \quad \frac{1}{r}\sum_{j^*\in\{1,\ldots,r\}} \mathsf{P}^{\mathbf{D}'_\ell\mathbf{R}}(\mathcal{G}_{j^*=1}) \ .$$

However, note that the switch values $(s_1,\ldots,s_r)$ emulated by the distinguisher $\mathbf{D}'_\ell$ and which are input to $\mathbf{D}_{\mathsf{opt}}(j^*)$ have a different distribution than when $\mathbf{D}_{\mathsf{opt}}(j^*)$ interacts with the trigger $\mathsf{GT}(\mathcal{P},q)$ alone. Indeed, in the $\mathbf{D}'_\ell\mathbf{R}$ random experiment, if a session j is declared to be "broken" ($s_j=1$), it does not necessarily mean that $\mathbf{D}_{\mathsf{opt}}(j^*)$ correctly guessed the password used in session j, it could also be due to either collisions in the random oracle or to the existence of a second key k such that $dec(k,c_j)=m_j$. This implies that $\mathsf{P}^{\mathbf{D}'_\ell\mathbf{R}}(\mathcal{G}_{j^*=1})$ may be smaller than $\max_{\mathbf{D}\in\mathcal{D}}\ \Gamma^{\mathsf{GT}(\mathcal{P},q)}(j^*,\mathbf{D})$ and we must thus bound this difference.

To do so, let $\mathcal{B}_j$ be the event in the $\mathbf{D}'_\ell\mathbf{R}$ random experiment that a query made to the random oracle at its E interface resulted in session j being declared "broken" ($s_j=1$) even though the password used in session j was not yet guessed ($\mathcal{G}_j=0$) and let $\mathcal{B}$ be the union of these events $\mathcal{B}_j$, for all $j\in\{1,\ldots,r\}$. Conditioned on the event $\mathcal{B}$ not happening, a session is declared "broken" if and only if its password is guessed and thus

$$\mathsf{P}^{\mathbf{D}'_\ell\mathbf{R}}(\mathcal{G}_{j^*=1}) \quad \geq \quad \max_{\mathbf{D}\in\mathcal{D}}\ \Gamma^{\mathsf{GT}(\mathcal{P},q)}(j^*,\mathbf{D}) - \mathsf{P}^{\mathbf{D}'_\ell\mathbf{R}}(\mathcal{B}=1)\,,$$

for every session $j^*\in\{1,\ldots,r\}$.

From the union bounds, it follows $\mathsf{P}^{\mathbf{D}'_\ell\mathbf{R}}(\mathcal{B}=1)\leq\sum_{j\in\{1,\ldots,r\}}\mathsf{P}^{\mathbf{D}'_\ell\mathbf{R}}(\mathcal{B}_j=1)$ and we now upper bound the probability of the event $\mathcal{B}_j$ happening in the $\mathbf{D}'_\ell\mathbf{R}$ random experiment, for some session j. Consider a query $v_j\,\|\,w$ made to the random oracle at its interface E, for some password guess $w\in\mathcal{W}$ and where v_j is the prefixed associated with session j, such that this query provoked the event $s_j=1$ even though $\mathcal{G}_j=0$. Note that such a query $v_j\,\|\,w$ was never asked before to the random oracle at interface E (since it provoked the event $s_j=1$), and was also never queried by the honest parties at interface A or B of the random oracle since the password for that session is not guessed ($\mathcal{G}_j=0$) and other sessions use different prefixes. Thus, the answer of the random oracle on such a fresh query $v_j\,\|\,w$ is uniformly and independently distributed and conditioned on a given message m_j and ciphertext c_j the probability that $\mathcal{B}_j=1$ is thus at most $q\cdot 2^{-n}\,|\{k'\in\mathcal{K}\mid dec(k',c_j)=m_j\}|$ since there are at most q queries made to the random oracle at interface E. As c_j is an encryption of m_j under a random key, it then follows that

$$\mathsf{P}^{\mathbf{D}'_\ell\mathbf{R}}(\mathcal{B}_j=1) \quad \leq \quad q\cdot\frac{\kappa_\ell}{2^n},$$

where κ_ℓ denotes the average number of keys decrypting a ciphertext to the original random message of length ℓ, i.e., $\kappa_\ell := \mathbb{E}_{m,c}\left[|\{k'\in\mathcal{K}\mid dec(k',c)=m\}|\right]$ where the expectation is taken over a message m chosen uniformly at random in $\mathcal{M}_\ell$ and $c\leftarrow enc(k,m)$ for a uniformly chosen key k.

Note that if the encryption scheme used is IND-CPA secure, then the ratio $q\frac{\kappa_\ell}{2^n}$ will be small. To see that, consider a distinguisher $\mathbf{D}^{\mathsf{CPA}}_{\ell,q}$ which interacts with the CPA systems $\mathbf{G}^{\mathsf{CPA}}_b(\mathrm{SE})$ described in Alg. 2. The distinguisher $\mathbf{D}^{\mathsf{CPA}}_{\ell,q}$ first selects a message m uniformly at random in $\mathcal{M}_\ell$, then outputs it to $\mathbf{G}^{\mathsf{CPA}}_b(\mathrm{SE})$ to retrieve the ciphertext c and finally tries to decrypt it by selecting q keys uniformly at random.

Alg. 14: Distinguisher $\mathbf{D}^{\mathsf{CPA}}_{\ell,q}$

$m \xleftarrow{\$} \mathcal{M}_\ell$
$c :=$ **result of querying** m to $\mathbf{G}^{\mathsf{CPA}}_b(\mathrm{SE})$
for $1\leq u\leq q$ **do**
  $k' \xleftarrow{\$} \mathcal{K}$
  if $dec(k',c)=m$ **then return** 0
return 1

The distinguisher $\mathbf{D}^{\mathsf{CPA}}_{\ell,q}$ outputs 0 when connected to $\mathbf{G}^{\mathsf{CPA}}_0(\mathrm{SE})$ with probability exactly $q\frac{\kappa_\ell}{2^n}$. In contrast, when interacting with $\mathbf{G}^{\mathsf{CPA}}_1(\mathrm{SE})$, the ciphertext c produced is independent of the message and thus $\mathbf{D}^{\mathsf{CPA}}_{\ell,q}$ outputs 0 with probability at most $\frac{|\mathcal{K}|}{|\mathcal{M}_\ell|}$. Thus,

$$q\cdot\frac{\kappa_\ell}{2^n} \quad \leq \quad \Delta^{\mathbf{D}^{\mathsf{CPA}}_{\ell,q}}\left(\mathbf{G}^{\mathsf{CPA}}_0(\mathrm{SE}),\mathbf{G}^{\mathsf{CPA}}_1(\mathrm{SE})\right)+\frac{|\mathcal{K}|}{|\mathcal{M}_\ell|} \ .$$

The previous equations imply that

$$\mathsf{P}^{\mathbf{D}'_\ell \mathbf{R}}(1) \quad \geq \quad \Gamma_{\mathsf{opt}}^{\mathsf{GT}(\mathcal{P},q)} - r \cdot \left(\Delta^{\mathbf{D}^{\mathsf{CPA}}_{\ell,q}} \left(\mathbf{G}_0^{\mathsf{CPA}}(\mathrm{SE}), \mathbf{G}_1^{\mathsf{CPA}}(\mathrm{SE}) \right) + \frac{|\mathcal{K}|}{|\mathcal{M}_\ell|} \right) \ .$$

Similarly to the proof of Theorem 1, the disitnguisher $\mathbf{D}'_\ell$ outputs 1 when interacting with the ideal system $\mathbf{S}_\mathbf{T}$ with probability at most $\Gamma_{\mathsf{avg}}^{\mathbf{T}} + \frac{|\mathcal{K}|}{|\mathcal{M}_\ell|}$. The desired bound is then obtained by combining the last two equations. □

7 Conclusion

The work of Bellare et al. [BRT12] initiated the provable-security analysis of the techniques used in the password-based cryptography standard [Kal00], leaving, however, several questions unanswered. First, the different security notions introduced do not formalize a desired per-session type of security guarantee and lack a composition theorem that would make them useful in applications and higher-level protocols. Second, the security statements made are with respect to a class of adversaries having a pre-determined number of password-guessing attempts for each session, which turns out to be a severe restriction even when considering only the simpler case of passwords that are identically and independently distributed. Our formalization overcomes these restrictions.

Even though Theorem 2 shows that the results of [BRT12] carry over to a composable model with the corresponding per-session assumptions, 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.

Finally, our general impossibility result in Theorem 1 shows that bounding the adversary's queries per session, although an unrealistic assumption (as discussed in Section 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 [Dam07] perspective on adaptively secure PKE, we view this question as being a fundamental research question still to be answered. 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.

Acknowledgments. Grégory Demay is supported by the Zurich Information Security and Privacy Center (ZISC). Peter Gaži is supported by the European Research Council under an ERC Starting Grant (259668-PSPC). Björn Tackmann is supported by the Swiss National Science Foundation (SNF).

References

[AS15] Joël Alwen and Vladimir Serbinenko. High parallel complexity graphs and memory-hard functions. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *47th ACM STOC*, pages 595–603. ACM Press, June 2015.

[AW05] Martín Abadi and Bogdan Warinschi. Password-based encryption analyzed. In Luís Caires, Giuseppe F. Italiano, Luís Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *ICALP 2005*, volume 3580 of *LNCS*, pages 664–676. Springer, Heidelberg, July 2005.

[BBM00] Mihir Bellare, Alexandra Boldyreva, and Silvio Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 259–274. Springer, Heidelberg, May 2000.

[BDJR97] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption. In *Foundations of Computer Science, 1997. Proceedings., 38th Annual Symposium on*, pages 394–403, Oct 1997.

[BO13] Mihir Bellare and Adam O'Neill. Semantically-secure functional encryption: Possibility results, impossibility results and the quest for a general definition. In Michel Abdalla, Cristina Nita-Rotaru, and Ricardo Dahab, editors, *CANS 13*, volume 8257 of *LNCS*, pages 218–234. Springer, Heidelberg, November 2013.

[BPR00] Mihir Bellare, David Pointcheval, and Phillip Rogaway. Authenticated key exchange secure against dictionary attacks. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 139–155. Springer, Heidelberg, May 2000.

[BR06] Mihir Bellare and Phillip Rogaway. The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426. Springer Berlin Heidelberg, 2006.

[BRT12] Mihir Bellare, Thomas Ristenpart, and Stefano Tessaro. Multi-instance security and its application to password-based cryptography. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 312–329. Springer, Heidelberg, August 2012. Full version at http://eprint.iacr.org/2012/196.

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 253–273. Springer, Heidelberg, March 2011.

[Can00] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. Cryptology ePrint Archive, Report 2000/067, 2000. http://eprint.iacr.org/2000/067.

[CGBS16] Henry Corrigan-Gibbs, Dan Boneh, and Stuart Schechter. Balloon hashing: Provably space-hard hash functions with data-independent access patterns. 2016.

[CHK+05] Ran Canetti, Shai Halevi, Jonathan Katz, Yehuda Lindell, and Philip D. MacKenzie. Universally composable password-based key exchange. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 404–421. Springer, Heidelberg, May 2005.

[Dam07] Ivan Damgård. *Automata, Languages and Programming: 34th International Colloquium, ICALP 2007, Wrocław, Poland, July 9-13, 2007. Proceedings*, chapter A "proof-reading" of Some Issues in Cryptography, pages 2–11. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[DGMT15] Grégory Demay, Peter Gaži, Ueli Maurer, and Björn Tackmann. Query-complexity amplification for random oracles. In Anja Lehmann and Stefan Wolf, editors, *ICITS 15*, volume 9063 of *LNCS*, pages 159–180. Springer, Heidelberg, May 2015.

[GL03] Rosario Gennaro and Yehuda Lindell. A framework for password-based authenticated key exchange. In Eli Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 524–543. Springer, Heidelberg, May 2003. http://eprint.iacr.org/2003/032.ps.gz.

[HMM15] Dennis Hofheinz, Christian Matt, and Ueli Maurer. Idealizing identity-based encryption. LNCS, pages 495–520. Springer, Heidelberg, December 2015.

[Kal00] B. Kaliski. PKCS #5: Password-based cryptography specification. RFC 2898, sept. 2000.

[Mau02] Ueli Maurer. Indistinguishability of Random Systems. In LarsR. Knudsen, editor, *Advances in Cryptology — EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 110–132. Springer Berlin Heidelberg, 2002.

[Mau12] Ueli Maurer. Constructive Cryptography – A New Paradigm for Security Definitions and Proofs. In Sebastian Mödersheim and Catuscia Palamidessi, editors, *Theory of Security and Applications*, volume 6993 of *LNCS*, pages 33–56. Springer Berlin Heidelberg, 2012.

[Mau13] U. Maurer. Conditional equivalence of random systems and indistinguishability proofs. In *2013 IEEE International Symposium on Information Theory Proceedings (ISIT)*, pages 3150–3154, July 2013.

[MM15] C. Matt and U. Maurer. A definitional framework for functional encryption. In *Computer Security Foundations Symposium (CSF), 2015 IEEE 28th*, pages 217–231, July 2015.

[MR11] Ueli Maurer and Renato Renner. Abstract Cryptography. In Bernard Chazelle, editor, *The Second Symposium in Innovations in Computer Science, ICS 2011*, pages 1–21. Tsinghua University Press, January 2011.

[MT79] Robert Morris and Ken Thompson. Password security: A case history. *Communications of the ACM*, 22(11):594–597, 1979.

[Nie02] Jesper Buus Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 111–126. Springer, Heidelberg, August 2002.

[O'G03] L. O'Gorman. Comparing passwords, tokens, and biometrics for user authentication. *Proceedings of the IEEE*, 91(12):2021–2040, Dec 2003.

[Per09] Colin Percival. Stronger key derivation via sequential memory-hard functions. *Self-published*, pages 1–16, 2009.

[PTAI15] Thanasis Petsas, Giorgos Tsirantonakis, Elias Athanasopoulos, and Sotiris Ioannidis. Two-factor authentication: is the world ready? Quantifying 2FA adoption. In *Proceedings of the Eighth European Workshop on System Security*, page 4. ACM, 2015.

[Tac14] Björn Tackmann. *A Theory of Secure Communication*. PhD thesis, ETH Zürich, August 2014.

A On Translating our Results into the UC Framework

While the statements proven in this work can in principle also be obtained in the UC framework, their formalization becomes more complex. The main relations and differences affecting the modeling and statements are as follows.

1. Resources in constructive cryptography (CC) correspond (roughly, see below) to functionalities in UC, and both the sender's and the receiver's converters of a protocol are encoded in a single protocol machine (the role is specified, for instance, in the first input). The insecure channel is not modeled explicitly, because the communication is relayed by the adversary in UC.
2. Since the definition of UC security, unlike our Definition 1, does not contain a statement about the correctness of a protocol, trivial protocols such as those never sending messages have to be excluded explicitly. (They are trivial to simulate by a simulator that never delivers the message.)
3. The separation of the transformable system into a core and a trigger system as described in Section 3 requires formal care, because the communication with these *different* systems has to be done using different Turing machine IDs, and the core system must accept the switch values only from the trigger system. This can be enforced using the machine IDs.
4. The technically most challenging difference is that resources in CC correspond to Turing machine *instances*; a UC statement about Turing machines allows for an *a priori* unbounded number of instances, whereas the number of instances is explicit in constructive statements. These restrictions can however be made explicit in the code of the Turing machines and enforced by referring to the machine ID. This means that our negative statement in Theorems 1 and 3 carries over to UC almost immediately, whereas the positive statements in Sections 6.1, 6.3 and 6.4 can also be proven but require functionalities that each allow for only a single instance.

B Password-Based Message Authentication

The goal of this section is to deepen and formalize the claims about password-based MACs sketched in Section 5. Since the security of a MAC scheme is stated as winning a game, and not as a distinguishing problem, we first recall known concepts about game winning.

B.1 Games

In the following, we denote a tuple of n random variables $(X_1, \ldots, X_n)$ by X^n. Similarly, x^n denotes a tuple of n values $(x_1, \ldots, x_n)$.

The behavior of an $(\mathcal{X}, \mathcal{Y})$-random system $\mathbf{S}$ can be captured as in [Mau02, Mau13] by a (possibly infinite) sequence of functions $\mathsf{p}^{\mathbf{S}}_{Y^k|X^k} : \mathcal{X}^k \times \mathcal{Y}^k \to [0,1]$, where $\mathsf{p}^{\mathbf{S}}_{Y^k|X^k}\left(y^k, x^k\right)$ is the probability of observing the outputs y^k given the inputs x^k.

A central tool in deriving an indistinguishability proof between two systems is to characterize both systems as being equivalent until a certain condition arises [Mau02, BR06]. Thus, being able to distinguish both systems requires to provoke this condition, and one is then interested in upper-bounding the probability of this event. Interacting with a random system in order to provoke a certain condition is naturally modeled by defining an additional *monotone binary output (MBO)* on the original system, where the binary output is monotone in the sense that it is initially set to 0 and that, once it has turned to 1, it can not turn back to 0. An $(\mathcal{X}, \mathcal{Y} \times \{0,1\})$-system where the second output component is monotone is often indicated by using a system symbol with a hat, such as $\widehat{\mathbf{R}}$.

For an $(\mathcal{X}, \mathcal{Y} \times \{0,1\})$-system $\widehat{\mathbf{R}}$ with an MBO, the $(\mathcal{X}, \mathcal{Y})$-system obtained from $\widehat{\mathbf{R}}$ by ignoring the MBO will usually be referred to as $\mathbf{R}$ (i.e., we simply omit the hat). Two $(\mathcal{X}, \mathcal{Y} \times \{0,1\})$-systems $\widehat{\mathbf{R}}$ and $\widehat{\mathbf{S}}$ with an MBO $A_1, A_2, \ldots$ are said to be *game equivalent*, denoted $\widehat{\mathbf{R}} \overset{\mathrm{g}}{\equiv} \widehat{\mathbf{S}}$, if they behave identically as long as they are not won, i.e,

$$\mathsf{p}^{\widehat{\mathbf{R}}}_{Y^k, A_k=0|X^k} = \mathsf{p}^{\widehat{\mathbf{S}}}_{Y^k, A_k=0|X^k},$$

for all k.

Given a distinguisher $\mathbf{D}$ and an $(\mathcal{X}, \mathcal{Y} \times \{0,1\})$-system $\widehat{\mathbf{R}}$ with an MBO, we denote by $\Gamma^{\mathbf{D}}\left(\widehat{\mathbf{R}}\right)$ the probability that $\mathbf{D}$ provokes the MBO in the random experiment formed by $\mathbf{D}$ interacting with $\widehat{\mathbf{R}}$. In this context, $\mathbf{D}$ will often be referred to as a *game winner*.

Lemma 5 ([Mau13, Lem. 1,2]). *Let $\widehat{\mathbf{R}}$ and $\widehat{\mathbf{S}}$ be two games such that $\widehat{\mathbf{R}} \overset{\mathrm{g}}{\equiv} \widehat{\mathbf{S}}$. Then, for any distinguisher $\mathbf{D}$*

$$\Delta^{\mathbf{D}}\left(\mathbf{R}, \mathbf{S}\right) \quad \leq \quad \Gamma^{\mathbf{D}}\left(\widehat{\mathbf{R}}\right) \quad = \quad \Gamma^{\mathbf{D}}\left(\widehat{\mathbf{S}}\right) \ .$$

B.2 Password-Based MACs

Given a MAC scheme $\mathsf{MAC} := (tag, vrf)$, recall the associated protocol $\mathsf{MAC} := (\mathsf{tag}, \mathsf{vrf})$ described in Section 5. The next theorem proves that if the MAC scheme employed is WUF-CMA secure, then the protocol MAC constructs an unordered authenticated channel $\mathsf{UAUT}^r_{\mathbf{T}}$, where UAUT^r is described in Alg. 15, from a downgradable key resource $\mathsf{KEY}^r_{\mathbf{T}}$ and an insecure channel INSEC^r, for any trigger system $\mathbf{T} \in \{\mathsf{LT}\left(\mathcal{P}, \boldsymbol{q}\right), \mathsf{GT}\left(\mathcal{P}, q\right)\}$. The real and the ideal experiment involved in the constructions statement are represented in Fig. 2 in Section 5. The arguments used in the proof of Theorem 4 could easily be extended to the case of any trigger $\mathbf{T}$ with output space $\{0,1\}^r$ and bit-wise non decreasing output.

Alg. 15: Channel UAUT^r

$s_j := 0$ and $\mathcal{Q}_j := \emptyset$, for all $j \in \{1, \ldots, r\}$
on input (j, m) at A
  $\mathcal{Q}_j := \mathcal{Q}_j \cup \{m\}$
  output (j, m) at $\mathsf{E_N}$
on input $s \in \{0,1\}^r$ at $\mathsf{E_S}$
  $(s_1, \ldots, s_r) := s$
on input (j, getsw) at $\mathsf{E_N}$
  output (j, s_j) at $\mathsf{E_N}$
on input (j', m') at $\mathsf{E_N}$
  if $(s_{j'} = 1) \ \vee \ (m' \in \mathcal{Q}_{j'})$ **then**
  output (j', m') at B

Theorem 4. *Let* $\mathrm{MAC} := (tag, vrf)$ *be a correct MAC scheme and consider the associated protocol* $\mathsf{MAC} := (\mathsf{tag}, \mathsf{vrf})$*. Then, there exists a simulator* σ_{MAC} *described in Alg. 16 such that for every distribution* $\mathcal{P}$ *of* r *passwords, every tuple of* r *integers* $\boldsymbol{q} := (q_1, \ldots, q_r)$ *and any integer* q*, and any trigger* $\mathbf{T} \in \{\mathsf{LT}(\mathcal{P}, \boldsymbol{q}), \mathsf{GT}(\mathcal{P}, q)\}$*,*

$$[\mathsf{KEY}^r_{\mathbf{T}}, \mathsf{INSEC}^r] \quad \stackrel{(\mathsf{MAC}, \sigma_{\mathsf{MAC}}, \varepsilon_{\mathbf{T}})}{\Longrightarrow} \quad \mathsf{UAUT}^r_{\mathbf{T}},$$

where $\varepsilon_{\mathbf{T}}(\mathbf{D}) := r \cdot \Gamma^{\mathbf{C}^{\mathsf{CMA}}_{\mathbf{T}}}(\mathbf{G}^{\mathsf{CMA}}(\mathrm{MAC}))$*, for every distinguisher* $\mathbf{D}$*, where the reduction system* $\mathbf{C}^{\mathsf{CMA}}_{\mathbf{T}}$ *depends on the trigger* $\mathbf{T}$ *and is described below in the proof.*

Proof. The simulator σ_{MAC} initially selects for each session j a key k_j uniformly at random and uses it to either compute tags, whenever a message is received at its in-interface, or to verify the tag of a message, whenever an injection query $(j, m' \,\|\, u')$ is input at its out_2-interface. The simulator forwards any password guessing query to the trigger of the resource and outputs the selected key k_j only if the password associated with this session was broken. The simulator σ_{MAC} is described in more details in Alg. 16.

Alg. 16: Simulator σ_{MAC}

$k_j \stackrel{\$}{\leftarrow} \{0,1\}^n$, for all $j \in \{1, \ldots, r\}$
on input (j, m) at in_{N}
  $u := tag(k_j, m)$
  output $(j, m \,\|\, u)$ at out_2
on input $(j', m' \,\|\, u')$ at out_2
  if $vrf(k_{j'}, m', u') = 1$ **then**
    output (j', m') at in_{N}
on input (j, w) at $\mathsf{out}_{1,\mathsf{S}}$
  output (j, w) at in_{S}
on input (j, getkey) at $\mathsf{out}_{1,\mathsf{N}}$
  $s :=$ **result of querying** (j, getsw) at in_{N}
  if $s = 1$ **then output** (j, k_j) at $\mathsf{out}_{1,\mathsf{N}}$
  else output $(j, \blacklozenge)$ at $\mathsf{out}_{1,\mathsf{N}}$

Alg. 17: Reduction $\mathbf{C}^{\mathsf{CMA}}_{\mathbf{T},j^*}$

$(s_1, \ldots, s_r) := 0^r$ and $k_1, \ldots, k_r \stackrel{\$}{\leftarrow} \{0,1\}^n$
on input (j, m) at A
  $u \leftarrow tag(k_j, m)$
  if $j = j^*$ **then** $u :=$**result of querying** (tag, m) at in
  output $(j, m \,\|\, u)$ at E_2
on input $(j', m' \,\|\, u')$ at E_2
  $b := vrf(k_{j'}, m', u')$
  if $j' = j^*$ **then** $b :=$ **result of querying** (vrf, m', u') at in
  if $b = 1$ **then output** (j', m') at B
on input (j, w) at $\mathsf{E}_{1,\mathsf{S}}$
  $(s_1, \ldots, s_r) :=$ **emulate result of querying** $\mathbf{T}$ on (j, w)
on input (j, getkey) at $\mathsf{E}_{1,\mathsf{N}}$
  if $(s_j = 0) \vee (j = j^*)$ **then output** $(j, \blacklozenge)$ at $\mathsf{E}_{1,\mathsf{N}}$
  else output (j, k_j) at $\mathsf{E}_{1,\mathsf{N}}$

In order to have shorter notations within the proof, the real system $\mathsf{tag}^{\mathsf{A}}\,\mathsf{vrf}^{\mathsf{B}}\,[\mathsf{KEY}^r_{\mathbf{T}}, \mathsf{INSEC}^r]$ will be denoted by $\mathbf{R}_{\mathbf{T}}$ and the ideal system $\sigma^{\mathsf{E}}_{\mathsf{MAC}}\,\mathsf{UAUT}^r_{\mathbf{T}}$ will be denoted by $\mathbf{S}_{\mathbf{T}}$, where $\mathbf{T} \in \{\mathsf{LT}(\mathcal{P}, \boldsymbol{q}), \mathsf{GT}(\mathcal{P}, q)\}$. Observe that the only difference between the systems $\mathbf{R}_{\mathbf{T}}$ and $\mathbf{S}_{\mathbf{T}}$ is that if a fresh message m' is injected by Eve together with a successfully forged tag, such a message m' is always output at Bob's interface B in the system $\mathbf{R}_{\mathbf{T}}$, whereas in $\mathbf{S}_{\mathbf{T}}$ such a message m' is only output if the password associated with that session was previously guessed. Let us add a monotone binary output to the systems $\mathbf{R}_{\mathbf{T}}$ and $\mathbf{S}_{\mathbf{T}}$, to obtain the corresponding games $\widehat{\mathbf{R}}_{\mathbf{T}}$ and $\widehat{\mathbf{S}}_{\mathbf{T}}$, defined as follows: the MBO $A_1, A_2, \ldots$ becomes 1 as soon as $(j, m' \,\|\, u')$ is input to the E_2-interface such that the message (j, m') was never input to the A-interface before, the tag is valid $vrf(k_j, m', u') = 1$, and the j^{th} password was not guessed ($s_j = 0$). Then, the previous observation implies that $\widehat{\mathbf{R}}_{\mathbf{T}} \stackrel{g}{\equiv} \widehat{\mathbf{S}}_{\mathbf{T}}$ and together with Lemma 5 we have

$$\Delta^{\mathbf{D}}(\mathbf{R}_{\mathbf{T}}, \mathbf{S}_{\mathbf{T}}) \quad \leq \quad \Gamma^{\mathbf{D}}\left(\widehat{\mathbf{R}}_{\mathbf{T}}\right),$$

for all distinguishers $\mathbf{D}$.

We now reduce the task of winning the game $\widehat{\mathbf{R}}_{\mathbf{T}}$ to the task of winning the WUF-CMA security game $\mathbf{G}^{\mathsf{CMA}}(\mathrm{MAC})$ described in Alg. 1. To do so, we consider a sequence of reduction systems $\mathbf{C}^{\mathsf{CMA}}_{\mathbf{T},1}, \ldots, \mathbf{C}^{\mathsf{CMA}}_{\mathbf{T},r}$, where each reduction $\mathbf{C}^{\mathsf{CMA}}_{\mathbf{T},j^*}$ has 5 outside sub-interfaces (labeled $\mathsf{A}, \mathsf{B}, \mathsf{E}_{1,\mathsf{N}}, \mathsf{E}_{1,\mathsf{S}}$ and E_2) and connects at its inside interface to the WUF-CMA game $\mathbf{G}^{\mathsf{CMA}}(\mathrm{MAC})$, for all $j^* \in \{1, \ldots, r\}$. The basic idea of the reduction $\mathbf{C}^{\mathsf{CMA}}_{\mathbf{T},j^*}$ is to forward any tag or verification query for session j^* to the system connected at its inside interface,

while other sessions are handled locally. The reduction $\mathbf{C}^{\mathsf{CMA}}_{\mathbf{T},j^*}$ also emulates internally the trigger $\mathbf{T}$, which in the case of $\mathsf{LT}(\mathcal{P}, \boldsymbol{q})$ or $\mathsf{GT}(\mathcal{P}, q)$ corresponds to first sample r passwords according to $\mathcal{P}$ and then to appropriately count the number of password-guessing queries, in order to keep track of which session should be considered "broken". Key retrieval queries are handled as usual, excepted for session j^* for which the error symbol $\blacklozenge$ is always returned. A more formal description of the reduction system $\mathbf{C}^{\mathsf{CMA}}_{\mathbf{T},j^*}$ is given in Alg. 17.

Consider a distinguisher $\mathbf{D}$ interacting with the game $\widehat{\mathbf{R}}_{\mathbf{T}}$ in order to provoke its MBO. Let $\mathcal{F}_j$ be the event that in this random experiment the distinguisher $\mathbf{D}$ won the game $\widehat{\mathbf{R}}_{\mathbf{T}}$ by finding a forgery for a fixed session j, i.e., $(j, m' \,\|\, u')$ was input at the E_2-interface such that (j, m') was never input to the A-interface before, the tag is valid $vrf(k_j, m', u') = 1$, and the j^{th} password was not guessed ($s_j = 0$), for some message $m' \in \mathcal{M}$ and tag $u' \in \mathcal{U}$. The probability that this event $\mathcal{F}_j$ happens in this random experiment is denoted $\mathsf{P}^{\mathbf{D}\widehat{\mathbf{R}}_{\mathbf{T}}}(\mathcal{F}_j = 1)$. Note that winning the game $\widehat{\mathbf{R}}_{\mathbf{T}}$ involves provoking one of the events $\mathcal{F}_j$, for some session $j \in \{1, \ldots, r\}$, and thus

$$\Gamma^{\mathbf{D}}\left(\widehat{\mathbf{R}}_{\mathbf{T}}\right) \quad \leq \quad \sum_{j \in \{1,\ldots,r\}} \mathsf{P}^{\mathbf{D}\widehat{\mathbf{R}}_{\mathbf{T}}}(\mathcal{F}_j = 1) \quad .$$

Let us fix the randomness used in the random experiment $\mathbf{D}\widehat{\mathbf{R}}_{\mathbf{T}}$ (consisting of the randomness of the distinguisher $\mathbf{D}$, the r keys used for tagging, the r passwords and the randomness of the algorithm *tag*). Conditioned on this randomness, any transcript of queries which provokes the event $\mathcal{F}_j$ in $\mathbf{D}\widehat{\mathbf{R}}_{\mathbf{T}}$ also wins the game $\mathbf{C}^{\mathsf{CMA}}_{\mathbf{T},j}\mathbf{G}^{\mathsf{CMA}}$(MAC). Moreover, in a such a transcript the password associated with the j^{th} session is not guessed ($s_j = 0$) and thus such a transcript happens with the same probability in the random experiment $\mathbf{D}\mathbf{C}^{\mathsf{CMA}}_{\mathbf{T},j}\mathbf{G}^{\mathsf{CMA}}$(MAC). Hence,

$$\mathsf{P}^{\mathbf{D}\widehat{\mathbf{R}}_{\mathbf{T}}}(\mathcal{F}_j = 1) \quad \leq \quad \Gamma^{\mathbf{C}^{\mathsf{CMA}}_{\mathbf{T},j}}\left(\mathbf{G}^{\mathsf{CMA}}(\mathrm{MAC})\right),$$

for all sessions $j \in \{1, \ldots, r\}$. The previous equations implies that the reduction $\mathbf{C}^{\mathsf{CMA}}_{\mathbf{T}}$, which consists in selecting a session j^* uniformly at random in $\{1, \ldots, r\}$ and then implementing the reduction $\mathbf{C}^{\mathsf{CMA}}_{\mathbf{T},j^*}$ is such that

$$\Gamma^{\mathbf{D}}\left(\widehat{\mathbf{R}}_{\mathbf{T}}\right) \quad \leq \quad r \cdot \Gamma^{\mathbf{C}^{\mathsf{CMA}}_{\mathbf{T}}}\left(\mathbf{G}^{\mathsf{CMA}}(\mathrm{MAC})\right),$$

for all distinguishers $\mathbf{D}$. □