Intrusion-Resilient Key Exchange in the Bounded Retrieval Model

David Cash[1], Yan Zong Ding[1], Yevgeniy Dodis[2], Wenke Lee[1], Richard Lipton[1], and Shabsi Walfish[2]

[1] College of Computing, Georgia Institute of Technology
{cdc,ding,wenke,rjl}@cc.gatech.edu
[2] Department of Computer Science, New York University
{dodis,walfish}@cs.nyu.edu

Abstract. We construct an intrusion-resilient symmetric-key authenticated key exchange (AKE) protocol in the bounded retrieval model. The model employs a long shared private key to cope with an active adversary who can repeatedly compromise the user's machine and perform any efficient computation on the entire shared key. However, we assume that the attacker is communication bounded and unable to retrieve too much information during each successive break-in. In contrast, the users read only a small portion of the shared key, making the model quite realistic in situations where storage is much cheaper than bandwidth.

The problem was first studied by Dziembowski [Dzi06a], who constructed a secure AKE protocol using random oracles. We present a general paradigm for constructing intrusion-resilient AKE protocols in this model, and show how to instantiate it *without* random oracles. The main ingredients of our construction are UC-secure password authenticated key exchange and tools from the bounded storage model.

1 Introduction

Robust systems for network security must guarantee resilience to compromises and intrusions. Company laptops, for example, often fall prey to Trojan horse viruses that users inadvertently "install" when they travel (without the protection of their company's firewalls). These viruses can persist until a sysadmin removes them, and then all credentials stored on the laptop must be replaced. A malicious virus could even steal a user's credentials with a key logger and erase itself, compromising all future use of the credentials until they are replaced.

A natural technique to overcome this problem is to use fresh session keys (or other credentials) *dynamically generated* using some secure key exchange protocol (e.g., Diffie-Hellman) between the parties involved. However, the problem with this latter approach is that such keys are not authenticated. So it seems like one must either sacrifice authentication, or lose one's resilience to compromises. The *bounded retrieval model*, introduced in various related contexts by [DLL05,CLW06,Dzi06a,Dzi06b],[3] provides an elegant and often very realistic

[3] These works used slightly different terminology and formalizations. Here we use the model of [Dzi06a], but borrow the nomenclature of [CLW06], since we feel that it reflects the general model most accurately.

way to resolve the above conflicting requirements. It assumes that storage is cheap and thus users can afford to store large quantities of data; in our context, this means that users can share very long symmetric keys (which can then be used to provide authentication). On the other hand, the bandwidth and retrieval capacities of both the users and the attacker are limited. For the users, this means that they only need to access very small portions of their long-term keys for authentication (and key exchange). On the other hand, we will be more generous to the attacker. We allow it to repeatedly compromise the user's machine for limited periods of time, and, during each such break-in, to perform any efficient computation on the *entire* shared key. However, we assume that the attacker's bandwidth capacity from the user's machine is bounded, so that the attacker is unable to retrieve too much information about the long symmetric key. Thus, in this model it might be possible to construct intrusion-resilient and, yet, authenticated key exchange protocols.

Indeed, this was recently done by Dziembowski [Dzi06a] who constructed such a protocol in the *random oracle model*. However, it is well known by now that a protocol proven secure in the random oracle model *might not* be secure when the random oracle is replaced by *any* secure cryptographic hash function [CGH04]. Therefore, an efficient construction without a random oracle is desired.

Our Main Result. We resolve the above open problem and construct an efficient intrusion-resilient authenticated key exchange (AKE) protocol in the bounded retrieval model. In fact, our contribution consists of two parts. First, we present a general paradigm for constructing such intrusion-resilient protocols, and, second, we then show how to instantiate our paradigm without random oracles. At a high level, our general paradigm first performs what we call a *Weak Key Exchange* (WKE), which only guarantees that the session keys output by the participants Alice and Bob are *individually unpredictable* (and equal in case the attacker is passive). After this stage, Alice and Bob use their "weak" keys as passwords for a *Password Authenticated Key Exchange* (PAK) protocol [BM93,BPR00,BMP00,GL01]. Such protocols are typically used in settings where the shared secrets are not uniformly distributed and have low entropy, and the goal is to construct AKE protocols resistant to offline dictionary attacks. Interestingly, in our setting the secrets will actually have high-entropy, and, with little effort, can even be made *individually* random. However, the utility of PAK protocols for our purposes comes from their implicit authentication guarantee: a party A with a password pw will only arrive at a shared session key only when interacting with a party B holding the *same* password pw. Thus, in our setting PAK is used to guarantee that Alice and Bob will agree on a session key after the WKE phase only if their individually unpredictable weak keys match.

Somewhat surprisingly, the security of our construction does not seem to follow from PAK protocols secure under most previously used definitions of PAK (*e.g.*, [BPR00,GL01,BDK$^+$05]). Instead, we need to employ Universally Composable (UC) PAK, as defined in [CHK$^+$05]. This is a very strong definition which guarantees the security of PAK even when used in arbitrary "environments" (see Section 2). Very informally (see the discussion at the end of Section 4 for more details), the strong guarantees of a UC PAK are required because of the extreme weakness of the security provided by the WKE definition described above. WKE may allow the adversary to *adaptively* correlate the session keys of Alice and Bob in an *arbitrary* manner, provided that they remain individually unpredictable. Indeed, the adversary is even provided with some *a priori* partial information about the secret keys of Alice and Bob, obtained via a previous intrusion. To further complicate matters, we require that the security of all future AKE sessions is preserved even after *multiple intrusions*, yet WKE provides no forward security guarantee at all. It is this latter complication which prevents most simulation based definitions of PAK from sufficing for our purposes. Namely, it is difficult to simulate the PAK protocol in a manner consistent with both past and *future* information obtained by the adversary about the secret key X. On the other hand, UC secure PAK protocols already support such simulation, since the environment in the UC experiment can also provide such partial information about party's secrets (which are, indeed, generated by the UC environment) directly to the adversary.

We can instantiate the above "WKE+PAK" paradigm in several ways to get concrete intrusion-resilient AKE protocols in our model. First, we observe that the original protocol of Dziembowski [Dzi06a] could be viewed as a special case our our method. Specifically, the 2-round WKE implicit in [Dzi06a] is built using purely information-theoretic tools used in the bounded storage model [Mau92] (most crucially, averaging samplers [BR94,Vad04]). In our construction, we will use a similar (and slightly simpler) WKE. As for the (high-entropy) PAK protocol, the protocol implicit in [Dzi06a] first applied the random oracle to the password, which simultaneously solved two main difficulties of the PAK setting: non-uniform passwords became uniform, and correlated passwords became independent (unless equal to begin with). Not surprisingly, after this application of the random oracle, a standard symmetric-key AKE protocol (more or less from [BR93]) was sufficient for the implicit PAK protocol of [Dzi06a]. In contrast, PAK protocols are much more complicated in the standard model, especially in the UC-model. Luckily, Canetti et al. [CHK$^+$05] built such an efficient UC PAK protocol in the common reference string (CRS) model. Since a CRS is trivially implementable in our model, — the parties can generate and store it as part of their long shared secret, — we immediately get the first intrusion-resilient AKE protocol without random oracles.

To summarize, we get our main result by first properly abstracting and modularizing the construction of [Dzi06a] (as consisting of "hidden" WKE and UC PAK), and then proving a general composition theorem allowing us to build intrusion-resilient AKE protocols (by composing WKE and UC PAK).

Proactive Security. One obvious issue in the bounded retrieval model is that even a long key will become useless after too many break-ins, as the adversary will eventually steal too much of the key. Of course, to solve this problem we can let Alice and Bob occasionally refresh their long keys as follows. They first run the AKE protocol to obtain a fresh key short key r, then expand r into a long key R using a pseudorandom generator (this operation will take some time, but is feasible overall), then XOR this long R to their currently shared long key (once again, this will take some time), and, finally, erase the short r. Needless to say, the parties should perform these last few steps "offline" and only when being absolutely sure that the AKE phase succeeded. Also, since most pseudorandom generators operate in the stream cipher mode, these periodic updates can be done "in place". Additionally, the final long key will be secure as long as either a large enough portion of the original key is not yet leaked, or if the AKE phase was not compromised.

Correcting Errors. We can also extend our model to allow the adversary to adaptively cause errors in some fraction of the symmetric keys. This extension allows us to tolerate either accidental hard-drive failures of some small part of the secret storage, or even malicious attacks when a virus might be able to rewrite or damage a small portion of the disk. In fact, our extension for this case is only nominally less efficient than our main construction. As our main tool, we employ *secure sketches* [DORS06] in this construction. The only caveat here is that we need to assume that the attacker cannot inject errors into the CRS. Thus, the CRS must be stored in a protected read-only area or made public in some other way. Details appear in Appendix A.

1.1 Related Works

The *Bounded Storage Model (BSM)* [Mau92] is closely related to the bounded retrieval model. In the BSM, a large random string (analogous to our long secret key) is broadcast publicly at a rate that overwhelms the adversary's ability to compress and store it. This is similar to the bounded retrieval restriction in our model. However, in the BSM the parties are assumed to share a short additional key which can only be compromised after the attacker lost access to a long string. In contrast, in our model the only secret is the long string, and the attacker can adaptively break-in and learn large parts of this long string. This makes the bounded retrieval model considerably more complicated than the BSM. Indeed, it is easy to see that, unlike the BSM model, it is impossible to have information-theoretically secure AKE protocols in our model. However, the techniques from the BSM model are useful in our model as well: indeed, the WKE protocol we use crucially utilizes tools from the BSM (such as averaging samplers).

The utilization of PAK protocols in our solution was influenced by the study AKE protocols using biometric data [BDK$^+$05]. In particular, this work also introduced the idea of using low entropy intermediate keys as inputs to a PAK protocol, which is fundamental to our construction. The usual PAK model was insufficient in that application as well, and was augmented to allow the adversary to specify some correlation between parties' (unequal) passwords. However, this extension was much weaker and much simpler than the one required in our setting. In particular, UC PAK protocols were not needed for that application.

Another related line of work is that of privacy amplification and *authenticated* key agreement using a shared weak secret key [MW03,Wol98,RW03,DKRS06]. As in our model, the secret key is only guaranteed to have some entropy. However, all these information-theoretic protocols require accessing and performing computation on the *entire* key. In contrast, a key feature of our model is that the keys are huge, and participants can only access a tiny portion of the key (we call this property *locality*). Thus, the above methods are inapplicable to our setting. In fact, we already mentioned that information-theoretic solutions (such as the above) are impossible in our setting.

Another related problem is protection against partial key exposure (so-called *exposure-resilient cryptography*; see [CDH$^+$00]), where an adversary can directly access most of the original bits of the secret key without effecting the security of the system. However, the solutions in this model are again non-local, and the adversary is not allowed to compute arbitrary functions of the key, like in our model.

A different direction for dealing with key exposure is the study of *key-evolving schemes*. Such schemes allow the attacker to obtain the *entire* (short) key, but assume the existence of global time, and update their secret key at each time period (thus, unlike our schemes, these schemes are stateful). In *forward secure* schemes [And97,BM99,CHK03,BY03], an adversary is allowed to compromise the system at some point, but is still unable to break the system for previous time periods all of which have not been compromised. Unfortunately, all "future" security is necessarily gone in this model.

The model of *key-insulated cryptography* [DKXY02,DKXY03] fixed this problem, where all past and future periods remain secure after a fixed number of compromises. However, this is achieved by introducing non-corruptible server which holds a master key and helps the user update its secret key from period to period. So called *intrusion-resilient* cryptosystems [IR02,DFK$^+$03] extend the above modeling and allow the attacker to corrupt both the user and the server, but not in the same time period. From our perspective, such schemes can be viewed as partitioning a key into two parts, allowing the attacker to obtain either part at each period, and updating both parts in between the periods.

Finally, we already mentioned several recent works that introduced the bounded retrieval model in various contexts. Dagon et al.[DLL05] used it for database protection, Di Crescenzo et al. [CLW06] — for password authentication resisting offline dictionary attacks,[4] and Dziembowski — for public-key encryption [Dzi06b] and intrusion-resilient AKE [Dzi06a] (the latter being the subject of this paper).

1.2 Structure of the Paper

In Section 2 we present some technical lemmas and tools. In Section 3 we formally define the model and security for intrusion resilient AKE. In Section 4 we present a secure construction of intrusion resilient AKE and discuss the use of Universally Composable PAK. Finally we discuss extending the model to allow for errors in Appendix A.

[4] Here the model is similar to ours, except the adversary may only steal original bits of the large key and not any function like in our model.

2 Preliminaries

In this section we briefly review some of the facts and tools used in this paper. Throughout, U_n denotes the uniform distribution on bit strings of length n, and $\|$ denotes string concatenation. We assume that the definitions of negligible function (negl(k)) and computational indistinguishability ($X \overset{c}{\equiv} Y$) are familiar.

Definitions and Facts from Probability. The *statistical distance* between two random variables taking values in S is defined to be

$$\Delta(X, Y) = \max_{T \subset S} |\Pr[X \in T] - \Pr[Y \in T]|.$$

The *min-entropy* $\mathrm{H}_\infty(X)$ of a discrete random variable X is defined as

$$\mathrm{H}_\infty(X) = \min_{x \in \mathrm{supp}(X)} \{-\log \Pr[\mathrm{X} = \mathrm{x}]\},$$

where $\mathrm{supp}(X)$ is the support of X. Let X be a random variable taking values in $\{0,1\}^n$, and let $k \leq n$. We say that X is a *k-source* if $\mathrm{H}_\infty(X) \geq k$, that is, for every $x \in \mathrm{supp}(X)$, $\Pr[X = x] \leq 2^{-k}$. Therefore, informally speaking, that X has high min-entropy means that the value X takes on is *hard to guess* (for an unbounded adversary). For a random variable X taking values in $\{0,1\}^n$ and $\alpha \in [0,1]$, we say that X has *entropy rate* α if X is an αn-source.

The following well known lemma quantifies the intuition that short compressions of long entropy-rich strings must leave out a lot of information. This allows us to tell how random our long secret keys look after some partial compromises.

Lemma 1 (c.f. [NZ96]). *Let X and Y be any two (correlated) random variables. Suppose that X is an n-source, and Y takes values in $\{0,1\}^r$. Then for every $\varepsilon > 0$, with probability at least $1-\varepsilon$ over $y \leftarrow Y$, $X|_{Y=y}$ is a $(n-r-\log(1/\varepsilon))$-source.*

Averaging Samplers. We also make use of a combinatorial tool called an *Averaging Sampler*. Averaging samplers, first introduced by [BR94], are procedures that approximate the mean of any function from bit strings to $[0,1]$ by taking a limited number of random samples in the domain of the function. We stress that an averaging sampler must work without any information about the function it is trying to approximate.

Definition 1 (Averaging Sampler). *A function* $\mathsf{Samp} : \{0,1\}^d \rightarrow [N]^m$ *is a (μ, θ, γ) averaging sampler if for every function $f : [N] \rightarrow [0,1]$ with average value $\frac{1}{N}\sum_i f(i) \geq \mu$, it holds that*

$$\Pr_{\{i_1,\ldots,i_m\} \leftarrow \mathsf{Samp}(U_d)} \left[\frac{1}{m} \sum_{j=1}^{m} f(i_j) < \mu - \theta \right] \leq \gamma$$

UC PAK. Our construction makes use of a Universally Composable Password Authenticated Key exchange (UC PAK) protocol, as defined in [CHK$^+$05]. Informally, the UC security notion of [Can01] requires that a protocol implementing an "ideal functionality" cannot be distinguished from the ideal functionality [5] it implements, even by an "environment" that chooses inputs to (and observes outputs from) the parties running the protocol while it is under attack by the adversary. One result of this very strong notion of security is that protocol designers may use an ideal functionality as a subroutine, yet rest assured that later replacing that ideal functionality with a UC secure protocol will not harm the security analysis of the newly designed protocol.

In this spirit, we will be making use of the UC PAK ideal functionality (Figure 2) as a subroutine in our protocol.

The UC PAK functionality captures the following intuitive guarantees: (1) If the parties are both honest *and* the adversary does not attempt to "compromise" their session, then they both receive an identical uniformly distributed random key, (2) if the adversary makes a failed attempt to compromise their session, honest parties each receive *independently generated* uniformly distributed random keys, and (3) the adversary is only able to successfully compromise a session by either correctly guessing the password of one of the parties (in a *single attempt*), or by corrupting one of the parties – in either case the adversary is allowed to choose the key(s) received by the honest parties. Unless the adversary successfully compromises a session, the only information provided by the functionality to the adversary is a notification when parties initiate the PAK protocol. (Of course, the adversary is also notified of the success or failure of an attempt to compromise a session.) Note the functionality *does not* guarantee that honest parties receive a shared key, even if the session was not compromised; they are merely assured that they each either have a good shared key, or a completely random one.

Unfortunately, like most non-trivial two-party UC functionalities, UC PAK protocols cannot be implemented in the plain model. Therefore, we rely on the construction of [CHK$^+$05], which assumes that a short Common Reference String (CRS) is publicly available. Since the parties in our AKE protocol already share large secrets, it is a simple matter for them to share a short (public) CRS value as well (alternatively, a small portion of the shared secret can be "sacrificed" to generate the CRS). Thus, the introduction of a CRS does not require any significant alterations to our model. Note that, in the special case where our security model is augmented with adversarial errors, the adversary should not be allowed to introduce errors into the CRS itself. Since a CRS is usually very short, protecting it within (public) "read-only memory" should not be costly.

[5] Technically, the behavior of the protocol under any given attack cannot be distinguished from the behavior of some "simulated" attack on the ideal functionality.

When interacting with an adversary S and a set of parties, with security parameter k, the functionality $\mathcal{F}_{pwKE}$ responds to the following queries:

- *NewSession*$(sid, P_A, P_B, pw, role)$*: Upon receiving a query of this form from party* P_A*, the message* $(NewSession, sid, P_A, P_B, role)$ *is sent to the adversary* S*. If this is the first NewSession query, or if this is the second NewSession query and there is a record* (P_B, P_A, pw')*, then record* (P_A, P_B, pw) *and mark this record* fresh*.*
- *TestPwd*(sid, P_A, pw')*: Upon receiving this query from the adversary* S*, if there is a record of the form* (P_A, P_B, pw) *which is* fresh*, then: If* $pw = pw'$*, mark the record* compromised *and notify* S *of a "correct guess". If* $pw \neq pw'$*, mark the record as* interrupted *and notify* S *of an "incorrect guess".*
- *NewKey*(sid, P_A, sk)*: Upon receiving this query from the adversary* S *(where* $|sk| = k$*), if there exists a record of the form* (P_A, P_B, pw) *and this is the first NewKey query for* P_A*, then one of the following actions occurs, after which the record* (P_A, P_B, pw) *is marked* completed*:*
  - *If this record is* compromised*, or either* P_A *or* P_B *is a corrupt party, then output* (sid, sk) *to* P_A*.*
  - *If this record is* fresh*, and there is a record* (P_B, P_A, pw') *with* $pw' = pw$*, and a key* sk' *was sent to* P_B*, and the record* (P_B, P_A, pw) *was marked* fresh *at the time, then output* (sid, sk') *to* P_A*.*
  - *In any other case, pick a new random key* sk' *of length* k *and send* (sid, sk') *to* P_A*.*

Fig. 1. Ideal Functionality $\mathcal{F}_{pwKE}$, from [CHK+05]

3 The Model and Definitions of Security

In this section we describe symmetric-key AKE protocols and their security in the BSM. Throughout this section, k is a security parameter. The length of the large symmetric-key is denoted by N, where $N = N(k)$ is a fixed sufficiently large polynomial. During a setup phase, a symmetric-key $X \in_R \{0,1\}^N$ is chosen uniformly and shared between two parties Alice and Bob.

An authenticated key exchange protocol Π is a pair of algorithms (A, B) describing the honest behavior of two parties. As described in the introduction, the adversary's limited access to X is vital to the security of our schemes, and we assume that the honest parties are similarly restricted (in fact, the honest parties will use a relatively small amount of their keys compared to the captured portion). We say that Π is *m-local* if A and B each access at most $m = m(k)$ bits in any execution of Π.

Adversaries will be able to steal a total of βN bits over the course of several executions, where $0 \leq \beta < 1$. We call β the *retrieval rate* of the adversary. Note that the adversary can adaptively decide on the size of the data to capture during a particular session, as long as it does not violate the total accumulated retrieval bound.

For the moment, we assume that a key X will be used for at most T sessions, where $T(k)$ is a polynomial determined by the retrieval rate of the adversary and the size of X. The assumption of an upper bound on the number of uses of X will be relaxed; below we show that security for a fixed $T(k)$ implies security for an arbitrary polynomial number of sessions under the assumption that the parties get some predetermined uncompromised sessions.

As with other models for authenticated key exchange, adversaries in our model (denoted C) completely control the channel between A and B. In particular, C can inject, drop, modify and delay messages. We also give C the power to temporarily intrude into the machines of A and B and retrieve some information about their internal state, including X. In this paper, our protocols are only proven secure for sequential executions of sessions. We comment on this limitation later.

At the beginning of each session, the adversary decides whether or not to intrude. From now on, we call a session during which the adversary intrudes a *compromised session*. After declaring a session (say session i) compromised, the adversary C outputs a circuit V (a virus) that will get to see the private information of A and B. This includes r^A and r^B, the private coins of A and B to be used in this session, and the secret key X. The virus V computes $S_i = V(X, r^A, r^B)$,[6] and sends S_i to the the adversary C.

We stress that V can be any polynomial-size circuit adaptively computed by C at the start of session i (so C may incorporate information gained from previous intrusions into V). The only other restriction on V is that the its output S_i is bounded. In particular, we require that $\sum_{i=1}^{T} |S_i| < \beta N$.

Clearly in a compromised session neither security nor correctness can be achieved. The best we can hope for is to construct a protocol that guarantees security and correctness for each uncompromised session. This is reflected in how success is determined in our definition below.

Our definition follows the style of [BPR00] and [Dzi06a].

The Adversarial Model. The power of an active adversary C is modeled by giving C oracle access to the protocol instances run by Alice and Bob. Denote by A and B the prescribed programs of Alice and Bob respectively. Denote by Π_i^A (resp. Π_i^B) the instance of protocol Π that Alice (resp. Bob) runs in the i-th session. For each $P \in \{A, B\}$, the instances Π_i^P are executed *sequentially*, that is, for each i, instance Π_{i+1}^P starts after instance Π_i^P completes. At the end of the i-th session, Π_i^A (resp. Π_i^B) outputs a bit acc_i^A (resp. acc_i^B) indicating whether A (resp. B) accepts or aborts in Session i. As in previous work, we assume that this bit is always known to the adversary C. Denote by sk_i^A (resp. sk_i^B) the session key output by Π_i^A (resp. Π_i^B) in the i-th session. If $\mathsf{acc}_i^A = 0$ (resp. $\mathsf{acc}_i^B = 0$), then $\mathsf{sk}_i^A = \perp$ (resp. $\mathsf{sk}_i^B = \perp$).

We define the following types of oracles that the adversary C is allowed to invoke, all of which except for the Intrude oracle are as defined in [BPR00]. We

[6] WLOG we assume that the adversary intrudes Alice and Bob simultaneously in a compromised session.

note that the adversary's retrieval bounded is expressed below in the definition of the $\mathsf{Intrude}$ oracle.

- $\mathsf{Execute}(i)$: Upon this call, the complete execution between protocol instances Π_i^A and Π_i^B takes place. The output of this call is the transcript, that is, the sequence of all messages exchanged between Π_i^A and Π_i^B. This oracle models passive eavesdropping in Session i.
- $\mathsf{Send}(P, i, M)$: This call sends the message M to the instance Π_i^P (where $P \in \{A, B\}$). The output of this call is the message the instance Π_i^P would send after receiving the message M, given its current state. This oracle models active man-in-the-middle attacks in Session i.
- $\mathsf{Intrude}(i, V)$: This oracle models intrusion into the machines of both A and B. The second input V to this call is a circuit (the virus) constructed based on the adversary C's current state. Upon this call, the oracle computes and outputs $S_i = V(X, r_i^A, r_i^B)$, where r_i^A and r_i^B are the private coins of A and B in session i. We require that $\sum_{j=1}^{T} |S_j| < \beta N$.
- $\mathsf{Test}(P, i)$: The output of this call is either the session key sk_i^P output by Π_i^P, or an independently chosen random string, each case happening with probability 1/2. The adversary's goal is to distinguish between the two cases. The call $\mathsf{Test}(P, i)$ can be invoked any time after Party P concludes Session i. The adversary may not invoke $\mathsf{Test}(P, i)$ when $\mathsf{acc}_i^P = 0$ (*i.e.* when $\mathsf{sk}_i^P = \perp$), or if the adversary has previously invoked $\mathsf{Intrude}(i, V)$.

The adversary C's advantage in Session i is defined as

$$\mathsf{Adv}_i(C) = |2 \cdot \Pr[C \text{ Succeeds in } \mathsf{Test}(P, i)] \; - \; 1|.$$

Remark: The query $\mathsf{Intrude}(i, V)$ is allowed only before the start of Session i and is not allowed during Session i.

Definition 2. *A session key protocol Π is intrusion-resilient for $T = T(k)$ sessions if for every PPT C with retrieval rate β, the following conditions are satisfied:*

- (Correctness) *With probability* $1 - \mathsf{negl}(k)$ *the following holds: For each* $i \in [T]$*, if session* i *is uncompromised and* $\mathsf{acc}_i^A = \mathsf{acc}_i^B = 1$*, then* $\mathsf{sk}_i^A = \mathsf{sk}_i^B$*.*
- (Privacy) *For each* $i \in [T]$ *s.t. the Test oracle is invoked for session* i*,* $\mathsf{Adv}_i(C) = \mathsf{negl}(n)$*.*

We now return to the issue of defining security for a fixed polynomial $T(k)$ number of sessions. We observe that an authenticated key exchange protocol Π can be used to refresh its own long secret keys during uncompromised sessions. The idea is that the parties can run Π to obtain a short key r, and then use that key as the seed for a pseudorandom generator. The output of the generator is then XORed with the previous long key, and the value r is erased. This ensures that the final long key will be "as good as new" if the attacker did not break-in right at the end of Π (i.e., r is uncompromised), and still "as good as before" even if the attacker compromised the value of r. Thus, as long as at least one uncompromised key update happened before the attacker obtained too much information about the long key, the long key remains secure. We defer the details to the full version on the paper.

We conclude this section by presenting a lemma that simplifies the analysis of our constructions. We show below that it suffices to construct a protocol that is intrusion-resilient for *three sessions.*

Lemma 2. *Suppose that a session key protocol Π is intrusion-resilient for* three sessions *against every PPT C that is β-retrieval bounded within the three sessions and attacks Π as follows: The adversary compromises the first session as usual; the second session is uncompromised; in the third session the adversary not only compromises the session but also gets the* entire key *X of Alice and Bob. Then for every polynomial $T = T(k)$, the protocol Π is intrusion-resilient for T sessions.*

The proof of Lemma 2 is a straightforward simulation of several sessions in only 3 sessions. Again, the proof will appear in the full version of this paper.

4 Authenticated Key Exchange from Weak Key Exchange

In this section we describe an approach for constructing AKE protocols in our model. We define the notion of Weak Key Exchange (WKE) and give an efficient construction, and then we show how to compose any WKE protocol with any UC PAK to realize a secure intrusion resilient AKE protocol.

4.1 Weak Key Exchange: Definition

Briefly, WKE provides only the guarantee that the output keys will have a high min-entropy from the viewpoint of the adversary. In particular, the adversary may possibly arrange for the keys to be unequal and correlated in an arbitrary fashion. Of course, we still require that the keys match when the protocol runs with no active interference from the adversary. WKE also provides no security guarantees on past keys once a subsequent WKE session is initiated (*i.e.* there is no forward security, and indeed, no long term security requirement at all). Our definition for WKE is a modification of the definition for authenticated key exchange, and thus we focus on the differences between the definition of WKE and that of AKE (as described above).

We use the same adversarial model as in AKE, but with a few critical weakenings. In particular, we will only allow for 2 sessions (in a similar spirit to Lemma 2), where the first session is compromised, and the second is not. Furthermore, we modify the Test oracle, and the corresponding experiment defining the adversarial advantage, replacing it with the following:

– $\mathsf{Test}(P, i, sk)$: The output of this call is 1 if $i = 2$ and $sk = sk_i^P \neq \perp$.

This oracle may only be called once by the adversary (for one party, using Session 2), in addition to the previous restrictions.

The adversary's advantage in the privacy requirement is redefined as

$$\mathsf{Adv}_2(C) = \Pr[C \text{ causes } \mathsf{Test}(i, P, sk) = 1].$$

There is no forward security guarantee for weak key exchange, as the adversary may not corrupt (or even invoke) any session that occurs after the second session (wherein it must query the Test oracle, attempting to break privacy). Furthermore, the adversary can only succeed by guessing the entire key sk_i^P (which is more difficult than merely distinguishing it from random), and thus the privacy guarantee is considerably weakened.

Definition 3 (Weak Key Exchange(WKE)). *A protocol is a weak key exchange if for every PPT C with retrieval rate β, the following conditions are satisfied:*

- (Weak Correctness) *With probability* $1-\mathsf{negl}(k)$ *the following holds: for each* $i \in \{1, 2\}$*, if Session* i runs honestly*, (*i.e. *the adversary does not tamper with any protocol messages) then* $\mathsf{sk}_i^A = \mathsf{sk}_i^B$*.*
- (Weak Privacy) *The advantage of the adversary is negligible, i.e.* $\mathsf{Adv}_2(C) = \mathsf{negl}(n)$*, where* $\mathsf{Adv}_2(\mathsf{C})$ *is as defined above, and in particular depends on the modified* Test *oracle.*

We note that in the event that the adversary chooses to alter the content of protocol flows, A and B may agree to accept a session where they receive differing keys.[7]

Below we present and analyze a construction meeting this definition, with information theoretic security.

4.2 Weak Key Exchange: Construction

Our protocol makes use of an *averaging sampler*, as described in [BR94], which samples a small number of bits from a much larger source (in this case, the shared secret X), while nearly preserving the min-entropy rate of the larger source. It was shown in [Vad04] how to explicitly construct samplers [8] for a δN-source using only $d = \log(N/m) + O(\log(1/\gamma))$ random bits and $m = O(\log(1/\gamma))$ bit samples from the input source to produce output that is γ-close to a $(2\delta/3)m$-source for any $\gamma > \exp(-N/2^{O(\log^* N)})$. Note that, given the practical importance of efficiency, here we obey the requirement that the number of bits of X which are read during the execution of a WKE is small. (This partly motivates the choice of parameters for the averaging sampler, and in particular, $m = O(\log(1/\gamma))$ is essentially the best one can hope for in terms of efficiency.)

Making use of the existence of such samplers, our protocol proceeds as follows (where we use $X_{\mathsf{Samp}(\cdot)}$ to denote the string formed by concatenating the bits of X located at the indices selected by Samp):

[7] Indeed, the adversary may attempt to guess either sk_2^A or sk_2^B in the attack scenario for the privacy requirement, and they may not be equal.

[8] See Lemma 8.4 and its usage in Theorem 8.5 in [Vad04] for details.

Setup:

- $X \in_R \{0,1\}^N$: A (large) secret key shared between Alice and Bob.
- $\mathsf{Samp} : \{0,1\}^d \to \{1,\dots,N\}^m$: An averaging sampler [Vad04].

Protocol:

1. Alice and Bob choose random values r_A and r_B, respectively.
2. Alice sends r_A to Bob. Bob receives a value r'_A, and then computes $K_B = X_{\mathsf{Samp}(r'_A)} \parallel X_{\mathsf{Samp}(r_B)}$. Bob outputs K_B as his weak session key.
3. Bob sends r_B to Alice. Alice receives a value r'_B, and then computes $K_A = X_{\mathsf{Samp}(r_A)} \parallel X_{\mathsf{Samp}(r'_B)}$. Alice outputs K_A as her weak session key.

Fig. 2. A WKE Protocol

Lemma 3. *The Weak Key Exchange protocol described in Figure 2 is secure for appropriate choices of the sampler parameters.*

Proof. The proof of security is direct, and the security guarantee is in fact information theoretic. The weak correctness property is obvious. The view of the adversary at the start of the second session is the output of some circuit $V(X)$, obtained during the compromise of the first session. As we are considering only β-retrieval bounded adversaries, the output of the circuit V must be no larger than βN-bits, and thus the adversary knows at most βN-bits of information about X. In particular, we observe that, from the adversary's point of view, X is nearly [9] a $(1-\beta)N$-source (since X is uniformly random over N bits, but the adversary's view is conditioned over the βN-bit output of $V(X)$ obtained during the first session).

Thus, by Lemma 6.2 of [Vad04] (and using the same sampler parameters as those of the explicit local extractor construction in Theorem 8.5 therein), the pairs $(r_A, X_{\mathsf{Samp}(r_A)})$ and $(r_B, X_{\mathsf{Samp}(r_B)})$ are (very nearly) individually γ-close to the distribution (U_d, W), where for every $r \in \{0,1\}^d$, the distribution $W|_{U_d=r}$ is at least a $(2(1-\beta)/3)m$-source. Thus, $K_A = X_{\mathsf{Samp}(r_A)} \parallel X_{\mathsf{Samp}(r'_B)} \approx W \parallel X_{\mathsf{Samp}(r'_B)}$, where W has (nearly) min-entropy $(2(1-\beta)/3)m$ even conditioned on r_A. Therefore, even conditioned on the adversary's view *after* the WKE protocol completes, K_A has min-entropy nearly $(2(1-\beta)/3)m$, irrespective of the adversary's choice of r'_B. An analogous argument applies to K_B, and thus, for sufficiently large choices of m, the weak privacy security requirement follows. □

4.3 Intrusion-Resilient AKE from WKE and UC PAK

On a high level, the protocol uses a WKE to select "passwords" with a high min-entropy for use by Alice and Bob in a standard PAK protocol. Since PAK

[9] Technically, with probability $(1-2^{-\lambda})$ taken over the distribution of $S \leftarrow V(X)$, the random variable $X|_{V(x)=S}$ (which is X conditioned on the adversary's view) has min-entropy $(1-\beta)N - \lambda$ for any choice of λ. This follows directly from Lemma 1.

protocols securely realizing the ideal functionality $\mathcal{F}_{pwKE}$ do not necessarily verify the output keys (*i.e.* check that the key exchange was successful, satisfying correctness), we use MACs to verify them, completing the protocol.

Setup:

- WKE: A Weak Key Exchange protocol between Alice and Bob.
- PAK: A Universally Composable Password Authenticated Key exchange protocol.
- (MAC,Verify): A MAC secure against chosen message attacks.

Protocol:

1. Alice and Bob run the WKE protocol, obtaining the weak keys K_A and K_B, respectively.
2. Alice and Bob exchange nonces, which are concatenated to form a session ID *sid*
3. Alice and Bob run the PAK protocol with session ID *sid*, using the passwords K_A and K_B as their inputs, respectively. Alice and Bob obtain their outputs, secret keys $sk_A = sk_A^1 \| sk_A^2$ and $sk_B = sk_B^1 \| sk_B^2$, respectively.
4. Alice sends Bob $tag_A = \mathsf{MAC}_{sk_A^1}(\text{Bob})$, Bob sends Alice $tag_B = \mathsf{MAC}_{sk_B^1}(\text{Alice})$.
5. Alice receives tag'_B and checks that $\mathsf{Verify}_{sk_A^1}(\text{Bob}, tag'_B) = 1$. If not, Alice rejects and aborts. Otherwise, Alice outputs sk_A^2 as her session key.
6. Bob receives tag'_A and checks that $\mathsf{Verify}_{sk_B^1}(\text{Alice}, tag'_A) = 1$. If not, Bob rejects and aborts. Otherwise, Bob outputs sk_B^2 as his session key.

Fig. 3. An AKE protocol based on WKE and UC PAK

Remark: PAK can be implemented using the protocol of [CHK$^+$05], which is a 6-round protocol that is efficiently implementable under standard number theoretic assumptions. Our WKE construction is a 2-round protocol, and we add four extra rounds in the above construction (two to exchange nonces, and two to verify the keys at the end), for a total of 12 rounds. However, in practice we can move the exchange of nonces in parallel with the WKE messages, reducing the number of rounds by 2, and we can move one of the verification messages in parallel with the last flow of the PAK protocol to save an additional round, bringing the total down to 9 rounds.

Theorem 1. *The* AKE *protocol described in Figure 3 is intrusion-resilient for three sessions defined in Lemma 2. Therefore, under the assumption that secure update sessions can be periodically scheduled, there is an intrusion-resilient* AKE *protocol for an unbounded number of sessions.*

Proof sketch: Since we are using a universally composable PAK protocol, by the UC Theorem of [Can01], we may substitute the execution of the PAK protocol with calls to the ideal functionality $\mathcal{F}_{pwKE}$ described above [10] (using K_A and K_B

[10] The substitution is legitimate since it is possible for an *environment* Z to internally simulate the rest of the protocol, including the setup phase with the shared secret

as the passwords for the calls by P_A and P_B, respectively). Clearly, the session keys output by the ideal functionality are indistinguishable from random to the adversary (by definition), unless the adversary makes a successful TestPwd query (which must be issued prior to the completion of the PAK protocol), allowing him to choose the output keys. However, as can be seen from the definition of the ideal functionality, the adversary is allowed only a single query to TestPwd. Following Definition 3, we observe that the adversary cannot guess the output from the WKE with non-negligible probability (even when conditioned on the entire view of the adversary up to and including the point at which the functionality $\mathcal{F}_{pwKE}$ is invoked, which is identical to the view when attacking just the WKE, plus the addition of a random *sid* and some other innocuous messages that may be sent from $\mathcal{F}_{pwKE}$ to the adversary), the probability that the adversary succeeds in a TestPwd query is at most negligible. Thus, we are guaranteed that sk_A and sk_B are indistinguishable from random, and either equal (in the event that no TestPwd query was issued) or independently generated (in the event that a failed TestPwd query was issued). Furthermore, it is easy to show that the last two flows of the protocol (Steps 4-6) ensure that correctness holds, provided that sk_A and sk_B are either identical, or independently chosen random values. Finally, we observe that the adversary can never distinguish sk_A or sk_B from random once the session has completed, even after being given all [11] of X, since the keys are chosen at random (independently of X) by the ideal functionality $\mathcal{F}_{pwKE}$ and are never revealed to the adversary. □

Remark: The UC PAK protocol of [CHK$^+$05] is only secure against "static" adversaries in the UC framework. At first glance, it might seem that we require security against "adaptive" adversaries here, since the random coins used by the parties are revealed during an intrusion. However, since we are only dealing with sequential sessions, and since there is no security guarantee provided for compromised sessions, we need not concern ourselves with the ability to simulate attacks on the compromised sessions (which would have necessitated the use of the adaptive security notion). The random coins used by uncompromised sessions are indeed erased, and thus static adversary UC security is sufficient.

The Need for UC-Secure PAK. We begin by remarking that the use of computationally secure tools is unavoidable in our setting (despite the information theoretic security of our WKE). This is due to a combination of the forward security requirement and efficiency requirements for the scheme: if it is efficient to compute a key-exchange, then the output must be information-theoretically determined by a small number of bits, all of which can be obtained by the adversary during a subsequent intrusion (without violating the retrieval bound).

X. Since UC security holds against any environment, in particular, this environment should be unable to distinguish calls to the ideal functionality $\mathcal{F}_{pwKE}$ from calls to the realized protocol.

[11] Indeed, if the adversary were given all of X prior to the completion of the WKE phase, the security property on the WKE would be broken, allowing the adversary to potentially issue a successful TestPwd query. Thus, it is in fact still critical that the adversary be β-retrieval bounded.

Given that reliance on a computationally secure tool is unavoidable, if we are going to use a WKE-based approach, one might initially suggest composing it with a standard PAK protocol relying on a traditional "stand-alone" security definition (particularly since we are restricting parties to sequential AKE sessions). Unfortunately, there are serious obstacles to this seemingly natural approach.

First, we observe that the passwords (the weak keys) output by our WKE phase may be correlated in an *arbitrary* manner. Many traditional definitions of PAK (as well as various other cryptographic tools) do not provide security in the scenario where the parties are using unequal but correlated passwords (or, resp., keys). Indeed, the protocol of [Dzi06a] employs random oracles specifically to transform the parties' correlated random secrets into independently random ones. Since we wish to avoid the use of random oracles, we are left to deal with definitions of PAK that allow the adversary to correlate passwords. For instance, the PAK definition of [BDK$^+$05] allows the adversary to specify (*a priori*) the joint distribution from which honest parties choose their passwords, so that they can be forced to obey an arbitrary correlation function.

Surprisingly, it seems difficult (perhaps even impossible) to prove the security of our construction even when composed with the (very strong) notion of PAK from [BDK$^+$05]. In particular, it is hard to construct a reduction from the AKE protocol to the security of the PAK protocol. To see this, consider the issue of simulating the large AKE secret X. If X is chosen directly by the reduction, then either (1) the passwords used by the parties being attacked by the reduction are not be properly derived from X, or (2) the reduction itself is also able to derive the passwords from X. Of course, in case (2) the reduction is not able to break the security of parties with *unknown* passwords, rendering it meaningless. On the other hand, in case (1), the simulation performed by the reduction does not faithfully reproduce the setting of the AKE adversary. In fact, it seems hopeless to prove that such simulations go undetected by the adversary in our setting, since the adversary can eventually obtain all the relevant portions of X (via an intrusion subsequent to the completion of the PAK protocol), and check for consistency. As an alternative approach, the reduction could specify a correlation function for the PAK security game that chooses the value of X, and then generates correlated passwords accordingly. Unfortunately, this too fails, since the reduction itself will not learn the value of X, and thus will be unable to properly simulate the input to the adversary during an intrusion.

Ultimately, we turn to UC PAK to overcome this difficulty. With a UC PAK, the reduction plays the role of the environment in the UC security definition. Here, the environment *is* allowed to choose the passwords used by the parties, and thus the reduction may choose X and generate passwords accordingly, as in case (1) above. This time, the reduction remains meaningful, since the adversary will *not* be privy to the passwords used by honest parties (due to the separation between the UC distinguishing environment, and the UC adversary). In fact, if we make use of the UC composition theorem, the entire security proof seems to follow the natural intuition for combining WKE and PAK.

To the best of the author's knowledge, this setting represents the first instance of protocol which, even when executed sequentially and in isolation, seems to require the use of a UC secure tool. Our setting thus provides a powerful and naturally occuring example of the benefits of UC security in modular protocol design. Here we are able to consider a protocol which, when designed intuitively using standard tools, (seemingly) cannot be proven secure even in a simple "stand-alone" protocol execution setting. Yet, when the same intuitive design is implemented using a UC secure tool, a proof of security far more readily presents itself.

References

[And97] Ross Anderson. Two remarks on public key cryptology. Invited Lecture. In *4th ACM Conference on Computer and Communications Security*, 1997.

[BM99] Mihir Bellare and Sara K. Miner. A forward-secure digital signature scheme. In *CRYPTO*, pages 431–448, 1999.

[BPR00] Mihir Bellare, David Pointcheval, and Phillip Rogaway. Authenticated key exchange secure against dictionary attacks. In *EUROCRYPT*, pages 139–155, 2000.

[BR93] Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In *CRYPTO*, pages 232–249, 1993.

[BR94] Mihir Bellare and John Rompel. Randomness-efficient oblivious sampling. In *FOCS*, pages 276–287, 1994.

[BY03] Mihir Bellare and Bennet S. Yee. Forward-security in private-key cryptography. In *CT-RSA*, pages 1–18, 2003.

[BM93] Steven M. Bellovin and Michael Merritt. Augmented encrypted key exchange: A password-based protocol secure against dictionary attacks and password file compromise. In *ACM Conference on Computer and Communications Security*, pages 244–250, 1993.

[BDK$^+$05] Xavier Boyen, Yevgeniy Dodis, Jonathan Katz, Rafail Ostrovsky, and Adam Smith. Secure remote authentication using biometric data. In *EUROCRYPT*, pages 147–163, 2005.

[BMP00] Victor Boyko, Philip D. MacKenzie, and Sarvar Patel. Provably secure password-authenticated key exchange using diffie-hellman. In *EUROCRYPT*, pages 156–171, 2000.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, pages 136–145, 2001.

[CDH$^+$00] Ran Canetti, Yevgeniy Dodis, Shai Halevi, Eyal Kushilevitz, and Amit Sahai. Exposure-resilient functions and all-or-nothing transforms. In *EUROCRYPT*, pages 453–469, 2000.

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. On the random-oracle methodology as applied to length-restricted signature schemes. In *TCC*, pages 40–57, 2004.

[CHK03] Ran Canetti, Shai Halevi, and Jonathan Katz. A forward-secure public-key encryption scheme. In *EUROCRYPT*, pages 255–271, 2003.

[CHK$^+$05] Ran Canetti, Shai Halevi, Jonathan Katz, Yehuda Lindell, and Philip D. MacKenzie. Universally composable password-based key exchange. In *EUROCRYPT*, pages 404–421, 2005.

[CLW06] Giovanni Di Crescenzo, Richard J. Lipton, and Shabsi Walfish. Perfectly secure password protocols in the bounded retrieval model. In *TCC*, pages 225–244, 2006.

[DLL05] David Dagon, Wenke Lee, and Richard J. Lipton. Protecting secret data from insider attacks. In *Financial Cryptography*, pages 16–30, 2005.

[DFK$^+$03] Yevgeniy Dodis, Matthew K. Franklin, Jonathan Katz, Atsuko Miyaji, and Moti Yung. Intrusion-resilient public-key encryption. In *CT-RSA*, pages 19–32, 2003.

[DKRS06] Yevgeniy Dodis, Jonathan Katz, Leonid Reyzin, and Adam Smith. Robust fuzzy extractors and authenticated key agreement from close secrets. In *CRYPTO*, pages 232–250, 2006.

[DKXY02] Yevgeniy Dodis, Jonathan Katz, Shouhuai Xu, and Moti Yung. Key-insulated public key cryptosystems. In *EUROCRYPT*, pages 65–82, 2002.

[DKXY03] Yevgeniy Dodis, Jonathan Katz, Shouhuai Xu, and Moti Yung. Strong key-insulated signature schemes. In *Public Key Cryptography*, pages 130–144, 2003.

[DORS06] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. Cryptology ePrint Archive, Report 2003/235, 2006. http://eprint.iacr.org/.

[Dzi06a] Stefan Dziembowski. Intrusion-resilience via the bounded-storage model. In *TCC*, pages 207–224, 2006.

[Dzi06b] Stefan Dziembowski. On forward-secure storage. In *CRYPTO*, pages 251–270, 2006.

[GL01] Oded Goldreich and Yehuda Lindell. Session-key generation using human passwords only. In *CRYPTO*, pages 408–432, 2001.

[IR02] Gene Itkis and Leonid Reyzin. Sibir: Signer-base intrusion-resilient signatures. In Moti Yung, editor, *CRYPTO*, volume 2442 of *Lecture Notes in Computer Science*, pages 499–514. Springer, 2002.

[Mau92] Ueli Maurer. Conditionally-perfect secrecy and a provably-secure randomized cipher. *Journal of Cryptology*, 5(1):53–66, 1992.

[MW03] Ueli Maurer and Stefan Wolf. Secret-key agreement over unauthenticated public channels iii: Privacy amplification. *IEEE Transactions on Information Theory*, 49(4):839–851, 2003.

[NZ96] Noam Nisan and David Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996.

[RW03] Renato Renner and Stefan Wolf. Unconditional authenticity and privacy from an arbitrarily weak secret. In *CRYPTO*, pages 78–95, 2003.

[Vad04] Salil P. Vadhan. Constructing locally computable extractors and cryptosystems in the bounded storage model. *Journal of Cryptology*, 17(1):43–77, 2004.

[Wol98] Stefan Wolf. Strong security against active attacks in information-theoretic secret-key agreement. In *ASIACRYPT*, pages 405–419, 1998.

A Augmenting the Model with Adversarial Errors

As mentioned earlier, we deal with the case where an adversary is allowed to change the party's keys adaptively so that they disagree at some limited number of indices. We first give a formal definition of the model with errors, and then we give the definition of *secure sketches* and show how they can be used to resist errors.

Definition 4. *Again we modify Definition 2. In this case, during a compromised session, the virus V may have two outputs: the first is the information sent back to the adversary, and second is the* error vector *which describes which bits of X to flip. We say that an adversary is γ-*error bounded *if it flips at most a γN bits of X between key updates.*

Our construction meeting Definition 4 uses *secure sketches*. Intuitively, a secure sketch is a primitive that lets us generate a *sketch* of a string w that will help a user with a noisy version w' of w correct errors, but will not help an outside observer significantly. Originally, secure sketches were defined for a general metric, but we only need the case for Hamming distance.

Definition 5 (Secure Sketch [DORS06]). *An (k, k', t)-*secure sketch *is a pair of algorithms* $(\mathsf{SS}, \mathsf{Rec})$ *such that*

1. *(*Security*) If W is a k-source, then W can be guessed by an adversary who sees $\mathsf{SS}(W)$ with probability at most $2^{k'}$.*[12]
2. *(*Correctness*) If w and w' differ at less than t indices, then $\mathsf{Rec}(w', \mathsf{SS}(w)) = w$.*

We now show how to construct a WKE protocol that functions correctly in the presence of errors. We use secure sketches to ensure that the strings $X_{\mathsf{Samp}(r_A)}$ and $X_{\mathsf{Samp}(r_B)}$ are corrected in the passive case. The details of the updated protocol appear below in Figure 4

We choose $\beta_1, \beta_2, \varepsilon$ and γ in Figure 4 so that when we have a β_1-retrieval bounded adversary who can flip a total of γN bits during the interaction, the adversary's chance at guessing a password handed to the PAK is at most $2^{-\beta_2 N}$.

Lemma 4. *For an appropriate setting of parameters, the protocol in Figure 4 satisfies the security definition of* WKE *in the augmented model where the adversary can inject a γ fraction of errors, and transmit $\beta_1 N$ bits during during the first round. (Recall that we only need to prove security in the case of two rounds for* WKE*).*

Proof sketch: Correctness in the passive case is obvious from the definitions of the primitives used. Privacy follows from the original analysis of our WKE construction, together with the observation that after seeing s_A (resp. s_B), it is still hard to guess $X^A_{\mathsf{Samp}(r_A)}$ (resp. $X^B_{\mathsf{Samp}(r_B)}$). We defer a detailed analysis, including parameter settings, to an expanded version of this work. $\square$

[12] We would like to briefly say that $\mathrm{H}_\infty(W|\mathsf{SS}(W)) > k'$, but what we actually need is that the *average min-entropy* $\widetilde{\mathrm{H}_\infty}(W|\mathsf{SS}(W))$ is greater than k', where $\widetilde{\mathrm{H}_\infty}(A|B) = -\log(\mathbf{E}_{b \leftarrow B}[\max_a \Pr[A = a | B = b]])$, which corresponds to the prose description given.

Setup:

- X^A and X^B where $X \in_R \{0,1\}^N$ and the Hamming distance between X_A and X is at most $\gamma/2N$ (similarly, X_B and X are within $\gamma/2N$ Hamming distance).
- $\mathsf{Samp} : \{0,1\}^d \to \{1,\ldots,N\}^m$: An averaging sampler
- $(\mathsf{SS}, \mathsf{Rec})$: A $(\beta_1 N - \log(1/\varepsilon), \beta_2 N, \gamma N)$-secure sketch.

Protocol:

1. Alice and Bob choose random values r_A and r_B, respectively.
2. Alice and Bob each compute the sketches s_A and s_B of $X^A_{\mathsf{Samp}(r_A)}$ and $X^B_{\mathsf{Samp}(r_B)}$, respectively.
3. Alice sends (r_A, s_A) to Bob, who receives (r'_A, s'_A), and first recovers ${X^{A}}'_{\mathsf{Samp}(r'_A)} = Rec(X^B_{\mathsf{Samp}(r'_A)}, s'_A)$, and outputs $K_B = {X^{A}}'_{\mathsf{Samp}(r'_A)} \parallel X^B_{\mathsf{Samp}(r_B)}$.
4. Bob sends (r_B, s_B) to Alice, who receives (r'_B, s'_B), and first recovers ${X^{B}}'_{\mathsf{Samp}(r'_B)} = Rec(X^A_{\mathsf{Samp}(r'_B)}, s'_B)$, and outputs $K_A = X^A_{\mathsf{Samp}(r_A)} \parallel {X^{B}}'_{\mathsf{Samp}(r'_B)}$.

Fig. 4. A WKE Protocol For Noisy Keys

Once we have an error-resistant WKE protocol, composing with a UC-secure PAK protocol results in an error-resistant and intrusion resilient AKE protocol. The reason this is true is because the PAK protocol does not access X on its own, and once the WKE property for its input passwords is established, the UC theorem allows us to replace the PAK protocol with an ideal functionality and the original proof goes through unchanged. Again, we defer the full details of the analysis.