Anonymous Authentication From Public-Key Encryption Revisited (Extended Abstract)

Daniel Slamanig

Carinthia University of Applied Sciences, Primoschgasse 10, 9020 Klagenfurt, Austria.
d.slamanig@cuas.at

Anonymous authentication apparently seems to be an oxymoron, since authentication is the task of proving one's identity to another party and anonymity is concerned with hiding one's identity. However, there are quite different constructions like ring [5] and group signatures [1] to solve this task. We are focusing on anonymous authentication protocols using *public-key encryption schemes* as their underlying building block, which, in contrast to the aforementioned, do receive only little attention. However, such anonymous authentication protocols are much simpler than other constructions and they can provide significant advantages over the aforementioned approaches. Firstly, they are fully compatible with deployed public-key infrastructures (PKIs) and thus can be adopted very easily. Secondly, such schemes enjoy an "ad-hoc" character and thus do not require involved registration or setup procedures. This is especially advantageous in dynamic environments, e.g. when users dynamically join and leave the group of authorized users. In this context existing primitives like group signatures to date lack of an efficient and practical solution. Furthermore, the "ad-hoc" character of these schemes allows users to flexibly choose their level of anonymity, i.e. the size of the group (anonymity set), for the sake of improved efficiency and additionally do not suffer from linear complexity such as ring signatures.

Such constructions were for the first time discussed in [6], although quite limited, since only applicable with deterministic public-key encryption (PKE) schemes. This idea was later on improved in [8] and [4] to the efficient use with probabilistic public-key encryption schemes. We review existing approaches to anonymous authentication from public-key encryption and present constructions which lead to the most efficient protocols.

Let us denote by U the set of authorized users and every user $u_i \in U$ (the prover) who wishes to authenticate to a verifier V should be able to pass the authentication (*correctness*). This means that u_i is able to prove to V that he belongs to $U' \subseteq U$, whereas even a dishonest V should not be able to tell which $u_j \in U'$ has actually conducted the proof (*anonymity*). Hence, from the point of view of the verifier every user is equally likely to be the one who is actually authenticating. Furthermore, the verifier should be able to restrict the ability of passing the authentication to members of U (*unforgeability*). The basic idea behind the protocols of interest is that a verifier runs n parallel instances of a challenge-response protocol based on PKE using n distinct public-keys (representing U') but a single challenge r. Note that removing a user from U means revocation, which is an easy task. The most crucial issue is that the verifier *may cheat* and try to identify the anonymous authenticator by encrypting distinct

challenges. When using deterministic PKE the authenticator can easily verify whether the verifier behaves honest by re-encrypting the challenge. However, in case of probabilistic PKE, one needs an additional round to obtain the respective random coins and only achieves *verifiable anonymity* (a cheating verifier is able to identify the anonymous authenticator, who, however, can detect this and terminate the protocol). Now we elaborate on which indicators characterize an optimal anonymous authentication protocol from PKE:

Communication: They require at least two messages (one round).
Bandwidth: The minimum bandwidth required by an anonymous authentication protocol are $|U'|$ ciphertexts plus the challenge (from u_i to V).
Computation: At least $|U'|$ encryptions (verifier) and at least one decryption (prover). In addition one has to check whether the verifier cheats. The most efficient solutions known so far require $|U'| - 1$ encryption operations.

A general approach to solve the *cheating problem* is to require that the verifier provides a proof of plaintext equality, e.g. using a general non-interactive zero knowledge proof (NIZK) of language membership. By "generalizing" the Naor-Yung paradigm one obtains such a proof system for the subsequent language L, but the size of the proof makes it impractical for real world applications:

$$L = \{(c_1, \ldots, c_n, pk_1, \ldots, pk_n) | \; \exists m \; s.t. \; c_1 = E_{pk_1}(m; \omega_1) \wedge \cdots \wedge c_n = E_{pk_n}(m; \omega_n)\}$$

Although one is able to construct more efficient proofs when putting restrictions on the used PKE schemes, this is far from being optimal.
Encrypt-EwH: Using the EwH transformation of [3], we can reduce the round complexity to a single round for probabilistic schemes, i.e. *any* IND-CPA secure public-key encryption scheme. The approach is as follows: The verifier chooses $r \in_R \{0,1\}^l$, chooses a suitable hash function H, computes the ciphertext sequence $\mathbf{c} = (E_{pk_1}(r; H(r)), \ldots, E_{pk_n}(r; H(r)))$ and sends $\mathbf{c}$ to the user (u_i). The user computes $r' = D_{sk_i}(\mathbf{c}[i])$ and checks whether $\mathbf{c}[j] = E_{pk_j}(r'; H(r'))$ for all $j \neq i$. If this holds he accepts, otherwise he terminates the authentication. This construction gives round optimal schemes and was already used to construct traceable anonymous identification schemes in [8]. Furthermore, we note that combining it with randomness re-use [2] and selection of suitable PKEs, we get round and bandwidth optimal schemes. One can also generalize this approach by using a PRNG (called Encrypt-PRNG) and iteratively apply it to the challenge r to derive the random coins.
Encrypt and Commit: Another approach is to use an unconditionally binding commitment scheme for bit strings in addition to public-key encryption. Basically, the idea is as follows: The verifier chooses a random challenge r and computes a commitment c for r. Then, he encrypts the open information using every public-key of U' and sends the ciphertext sequence $\mathbf{c}$ along with c to the anonymous user. The user decrypts the respective element of the sequence, obtainins the open information for the commitment, opens the commitment and thus receives the challenge. Then, he returns the challenge to the verifier, who in turn checks whether the sent and the received challenge match. Due to the unconditional binding property of the commitment scheme, it is infeasible for the

verifier to provide distinct open informations that will open the commitment to distinct values. Hence, the computational effort for the prover reduces to a single decryption operation and the opening of the commitment scheme. Note that a verifier's best cheating strategy is to partition the set U' into two equal sized sets whereas one receives the open information and the other half receives rubbish. Thus, the probability of detecting a cheating verifier is 0.5, which makes cheating not a good strategy for the verifier. However, to further reduce the cheating probability the approach below can be additionally applied.

Reducing Computations by Less Trial Encryptions: The most efficient approach so far requires the prover to perform only a single decryption operation and the opening of a commitment. All other approaches require the user to perform one decryption operation and $n-1$ trial encryptions. But we can obtain more efficient protocols by relaxing the anonymity and letting the prover solely check whether $k < n$ randomly chosen elements of the ciphertext sequence were encrypted properly. The verifier's best cheating strategy is to choose partitions of equal size, since all users are equally probable. Hence, the probability that a verifier will succeed equals l^{-k}. Thus, the chances to cheat unnoticeable decrease with the number of partitions exponentially in k. Note that in addition to a reduced computational effort, users also do not need to retrieve all the public-keys of other users in U', which saves bandwidth.

Postactive Anonymity: Another idea from [7] is that users do not compute trial encryptions anymore, but post their received ciphertext sequence along with a signature for the ciphertext sequence (and r - without message recovery) from the verifier and the decrypted random challenge to a public bulletin-board. This delegates verification to others while at the same time passing back the full risk to the dishonest verifier and makes honesty verifier's best behavior strategy.

References

1. Ateniese, G., Camenisch, J., Joye, M., Tsudik, G.: A Practical and Provably Secure Coalition-Resistant Group Signature Scheme. In: CRYPTO '00. LNCS, vol. 1880, pp. 255–270. Springer (2000)
2. Bellare, M., Boldyreva, A., Kurosawa, K., Staddon, J.: Multirecipient Encryption Schemes: How to Save on Bandwidth and Computation Without Sacrificing Security. IEEE Transactions on Information Theory 53(11), 3927–3943 (2007)
3. Bellare, M., Boldyreva, A., O'Neill, A.: Deterministic and Efficiently Searchable Encryption. In: CRYPTO 2007. LNCS, vol. 4622, pp. 535–552. Springer (2007)
4. Lindell, Y.: Anonymous Authentication. JPC 2(2) (2011)
5. Rivest, R.L., Shamir, A., Tauman, Y.: How to Leak a Secret. In: ASIACRYPT '01. LNCS, vol. 2248, pp. 552–565. Springer (2001)
6. Schechter, S., Parnell, T., Hartemink, A.: Anonymous Authentication of Membership in Dynamic Groups. In: FC '99. LNCS, vol. 1648, pp. 184–195. Springer (1999)
7. Slamanig, D., Rass, S.: Anonymous But Authorized Transactions Supporting Selective Traceability. In: SECRYPT '10. pp. 132–141. SciTePress (2010)
8. Slamanig, D., Schartner, P., Stingl, C.: Practical Traceable Anonymous Identification. In: SECRYPT '09. pp. 225–232. INSTICC Press (2009)