Efficient Mutual Authentication for Multi-Domain RFID Systems Using Distributed Signatures

Michael Braun[1], Ulrike Meyer[2], and Susanne Wetzel[3]

[1] University of Applied Sciences Darmstadt, Germany
[2] RWTH Aachen University, Germany
[3] Stevens Institute of Technology, Hoboken, NJ, US

Abstract. The use of RFID technology in complex and distributed environments often leads to a multi-domain RFID system in which security issues such as authentication of tags and readers, granting access to data, and revocation of readers turn into an administrative challenge. In this paper, we propose a new public-key-based mutual authentication protocol that addresses the reader revocation problem while maintaining efficiency and identity privacy. In addition, our new protocol integrates fine-grained access control and key establishment with mutual authentication. The core of our solution is the use of the concepts of key-splitting and distributed signatures to solve the validation and revocation problem. We show that our protocols can be implemented on RFID tags using lightweight implementations of elliptic curve cryptography.

Keywords: RFID Security, Elliptic Curve Cryptography, Authentication, Data on Tag, Secret Sharing, Distributed Signatures, Key Splitting, Identity Privacy

1 Introduction

Radio Frequency Identification (RFID) technology has become a ubiquitous part of our daily lives. Prominent applications include logistics and electronic travel documents. While in early applications only a unique identifier was stored on an RFID tag, a more recent trend leans towards storing more information on the tag. For example, the new electronic passports include sensitive data such as fingerprints as well as the facial image of the passport holder. As a consequence, security challenges have evolved from protecting the identifier and preventing tag cloning to enabling fine-grained access control to the data stored on the tag and enabling secure data exchange between the reader and the tag.

Common approaches to meet these challenges are either based on symmetric-key techniques (e.g., [1, 2, 7, 17]) or the use of public-key technology (e.g., [6, 9, 13]). While symmetric-key-based approaches are generally more efficient than public-key mechanisms, key management issues make them difficult to use in

practice. Typically, today's public-key-based mechanisms require the use of certificates and as a consequence the verification and validation of chains of certificates. The former requires signature verification on the tag which was shown to be feasible in practice [6]. However, the validation (checking the expiration date and the revocation status) of certificates is often not addressed [6] or still proves challenging. For example, the specification of the electronic travel documents [9] does not include any revocation mechanisms. In addition, checking the expiration date of a certificate is based on the tag approximating the current time with the most recent effective date of all valid certificates received. This possibly results in stolen or lost readers being able to read sensitive data of the tag as long as the tag's approximate time is before the expiration date of the certificate of the respective reader. Also, tags that are read infrequently and thus have an old approximate time can be read by readers with expired certificates as long as the tag's time is before the expiration date of the reader's expired certificate. While [9] does not specify any access control mechanisms for data on electronic travel documents, the specification for the German personal identification card [9] defines a reader's access control rights as part of its certificate. State-of-the-art mechanisms (regardless of whether symmetric or public-key-based) define key establishment procedures on top of the authentication to subsequently allow for a secure data exchange between the reader and the tag.

It is in this context that this paper proposes a new public-key-based mutual authentication protocol that addresses the validation and revocation problem while maintaining efficiency and identity privacy integrating fine-grained access control and key establishment. The core of our solution is the use of the concepts of key-splitting and distributed signatures to address the validation and revocation problem during authentication.

The rest of this paper is organized as follows: In Section 2 we provide a high-level overview of our system model, the building blocks we use, and discuss some additional related work. In Section 3 we describe our protocol in case a tag is to be read while it is in its home domain. In Section 4 the protocol is extended to support authentication in visited domains. Section 5 evaluates the feasibility of our protocol on RFID tags.

2 Our Approach

2.1 System Model

Our RFID system model consists of one or more domains which in turn include three types of entities: tags, readers, and authentication centers (see Figure 1). In particular, a tag T and a reader R belong to an administrative domain D which is controlled by an authentication center A—which in the following is referred to as home domain. While a tag is typically attached to an object that may roam to other administrative domains, also referred to as visited domains, a reader will always remain in its home domain only. Furthermore, we assume that a reader is always connected to its home authentication center via a secure channel and authentication centers of different domains are interconnected securely.

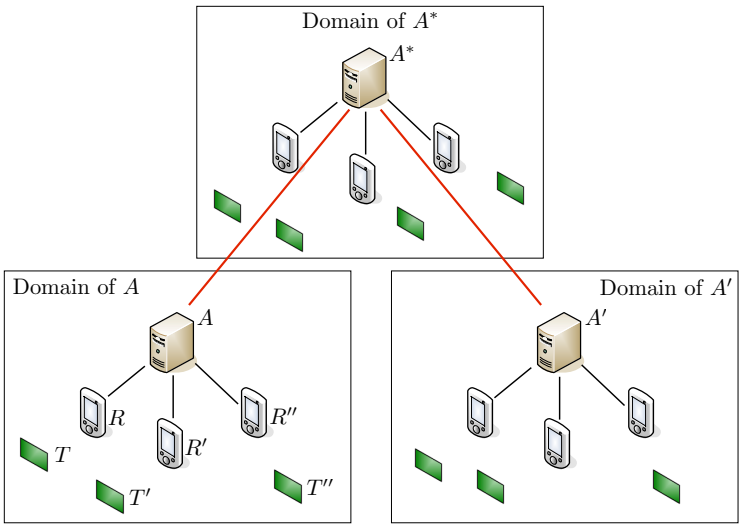


Fig. 1. Our RFID System Model

Each tag stores data in blocks grouped according to certain access controls. These access controls are read and/or write conditions which are set and controlled by the tag's authentication center. Different tags from the same domain may have different access controls. Similarly, different readers from the same domain may have different access rights for different tags (of its own or other domains). In our model, each authentication center has a single public key and each tag stores the public key of its home authentication center.

2.2 Building Blocks for our Protocols

Reader-to-Tag Authentication and Revocation: The key concepts used to achieve reader-to-tag authentication and revocation are key splitting and distributed signatures. Generally, the concept of key splitting refers to a private key being split into several key parts which are distributed to multiple parties in such a way that only specific combinations of these parties can together reconstruct the full private key. Based on key splitting of a private signature key it is possible to construct distributed signature schemes such that a valid signature can only be generated if the specific combination of parties contribute to the process.

In the context of this paper, the private key to be split corresponds to the public key of an authentication center. Assuming an authentication center A^* controls n^* readers and is connected to ℓ^* other authentication centers in the system, then the authentication center computes $n^* + \ell^*$ unique splits of its private key into two parts. For each such pair it keeps both parts to itself and distributes one part to either a reader or some other authentication center. The

splitting is done such that no combination of readers and other authentication centers can reconstruct the full private key without A^* contributing its corresponding key part. In addition, each authentication center A splits the key part it received from A^* into n pairs of key parts and distributes one part of each pair to one of its n readers.

For our new protocols we use the concept of distributed signatures in a novel way which enables reader authentication to the tag and solves the revocation and validation problem of reader certificates. Specifically, when a tag is in its home domain, reader authentication is based on a two-party signature generated by the tag's home authentication center and the reader itself. A similar approach was previously used in the context of WLANs [16]. When a tag is in a visited domain, reader authentication is based on a novel multi-party signature scheme in which the tag's home authentication center, the visited authentication center, and the reader each contribute their key part to the signature generation process.

As detailed in Sections 3 and 4, using the key-splitting and multi-party signature approach allows the construction of challenge-response reader-to-tag authentication protocols that have several advantages over the current state-of-the-art in which a certificate is issued for each reader. The new protocols simultaneously allow a tag to authenticate the reader, determine the reader's access rights, and check the revocation status of not only the reader but also that of the visited domain. In particular, a tag needs to handle only one key (i.e., the public key of its home authentication center) and needs to verify only one signature regardless of whether the tag is in its home or in a visited domain. While revocation in current certificate-based solutions require a tag to check the validity of a certificate (or even a chain of certificates in case the tag is in a visited domain), the new protocols implicitly allow for revocation checking as the visited authentication centers simply will not participate in the authentication (which is based on distributed signatures) of revoked readers. In addition, the home authentication center can revoke an entire domain by not participating in authentication sessions of a particular visited domain.

Key Agreement, Identity Privacy, and Tag-to-Reader Authentication: In order to achieve tag-to-reader authentication, allow for identity privacy, and establish a symmetric (short-lived) session key, our new protocols are based on a construction originally introduced in [6]. The construction in [6] is based on a combined challenge-response exchange and a Diffie-Hellman key agreement. The protocol roughly works as follows: Upon receiving the public Diffie-Hellman component from the tag, the reader signs both its public Diffie-Hellman component and that of the tag using its public key. Receiving the reader's signature and public Diffie-Hellman component, the tag can then check the authenticity of the exchange of the public Diffie-Hellman components and is thus assured of the reader's authenticity, i.e., that no man-in-the-middle is present. This, however, assumes that each tag has an authentic copy of every reader's public key which is a major drawback in the context of multi-reader and even more so multi-domain systems. Then, the tag will use the established Diffie-Hellman key to encrypt

a message which includes the response to a reader-issued challenge (computed using the tag's private key) and a certificate composed of the tag's identity and public key. Upon receiving the message from the tag, the reader first decrypts the message. A correct decryption guarantees the authenticity of the shared session key. Furthermore, the reader learns the identity of the tag from the certificate. The authenticity of the certificate combined with the verification of the response (using the tag's public key) ensures the authenticity of the tag.

Our construction, which is further detailed in Sections 3 and 4, is designed to address the major shortcoming of [6]. That is, while we make use of the Diffie-Hellman key exchange and the tag-to-reader authentication, we eliminate the need to store each reader's public key on each tag using the ideas of key splitting and distributed signatures described above. It is important to note that [6] does not address revocation of pre-stored public keys of the readers. In addition, our solution further improves on [6] by solving an issue that was not even addressed there, namely the introduction of access right management. Our scheme provides identity privacy that is equivalent to the granularity at which a tag's access conditions are specified. In particular, if all tags have the same access conditions, then our scheme provides full tag identity privacy as in [6].

It is important to note that the key splitting as used in our protocols does not require an encrypted channel for key transfer between the authentication center and the reader. Instead, the key is generated directly by the entities that use the key, namely the reader and the tag.

2.3 Other Related Work

The problem of certificate validation checking in the context of public-key-based reader-to-tag authentication has been addressed in some prior work. This includes the use of short-lived certificates, the use of hash-chain-based mechanisms, and the use of server-based authentication [13]. The first two approaches require time-synchronization which is questionable both with respect to security and practicality. In addition, for the second approach it is unclear how it can easily be extended to multiple domains. Finally, in the last approach the tag either needs to verify more than two signatures when roaming to visited domains or needs to pre-store more than one certificate. Our protocol addresses and overcomes all of these shortcomings.

$(2, \ell)$ threshold DSS signature schemes for $\ell \leq 3$ have first been suggested in [12] and make use of one party acting as a trusted third party. This approach was extended in [10] to (t, n) threshold signatures with $n \leq 2t + 1$. Since our construction requires an (n, n) signature scheme, it is not possible to directly use the schemes in [10, 12]. However, it is possible to modify the blinding in [12] to suit our purposes. In particular, while [12] blinds the shares of each of the two signers with random numbers and uses the third party to blindly construct

a full signature, our construction makes use of the fact that an entity has full knowledge of all parts of all splits it computes.[1]

A $(2, 2)$ DSS signature scheme was presented in [15]. This scheme requires a semantically secure homomorphic cryptosystems which is used to exchange data between the two signers. The necessity of a semantically secure homomorphic cryptosystem excludes the use of standard protocols such as TLS and IPsec for securing the communication between the signing entities and as such limits the practicality of the scheme. It furthermore is not obvious how this protocol can be extended to an (n, n) signature scheme.

3 Authentication in the Home Domain

In this section, we present our new mutual authentication protocol for the case that a tag is to be read by a reader in the tag's home domain. The protocol involves the tag T, the reader R and the tag's home authentication center A. The tag stores the public key of its home authentication center.

Reader-to-tag authentication is based on a two-party signature scheme in which the reader and the home authentication center jointly generate a signature. In the following, we first describe the key splitting between A and R and the novel two-party signature scheme based on ECDSA. We then describe how the two-party signatures are used in our new protocol.

3.1 Two-Party Signatures

We first briefly recall key and signature generation for the ECDSA scheme before we proceed to the actual key splitting and two-party signature generation. Note that our two-party signature generation also works with other DSA-based signature schemes. Exemplary, in the appendix we describe the two-party signature generation for the German version of the ECDSA signature scheme, the so-called ECGDSA.

ECDSA Signatures: Let P be the base point of an elliptic curve with prime order q. Let d be the private signature key. The corresponding public signature key is $Q = d \cdot P$. An ECDSA signature of a hashed message $h(m)$ is defined to be a pair (r, s) where the first component r is the affine x-coordinate of the point $k^{-1} \cdot P$ modulo q for a randomly selected ephemeral key k. Furthermore, the second component s satisfies the equation $s = k \cdot (h(m) + d \cdot r) \mod q$.

Key Splitting and Initial Setup: Let Q denote the public signature key of the authentication center A. The corresponding private signature key is denoted by d. A splits the private signature key d between itself and all of its readers. For

[1] Note that our signatures are not multi-party signatures in the traditional sense. In traditional multi-party signature schemes each party is assumed to know only its own key split.

this purpose, the authentication center A picks a reader's key part d_R uniformly at random and mutually different for different readers. A then computes the corresponding part d_A such that $d_A = d_R^{-1} \cdot d \mod q$. A provides the private key part d_R to the reader R and keeps the pair (d_A, d_R) for itself.

Signature Generation: To jointly generate a signature on the hash value $h(m)$ of a message m in a distributed manner, both parties, i.e., the authentication center A and the reader R, contribute to the ephemeral key k. The reader R chooses its part k_R uniformly at random and transmits it to A over the secure channel.[2] The authentication center A also picks a random value k_A and completes the ephemeral key generation by multiplying both parts $k = k_R \cdot k_A \mod q$.

The authentication center A now knows all parts of the private signature key d_A, d_R and of the ephemeral key k_A, k_R, whereas the reader knows the parts d_R and k_R only. The authentication center A starts the signature generation by calculating r as the affine x-coordinate of $k^{-1} \cdot P = (k_A \cdot k_R)^{-1} \cdot P$ modulo q and

$$s_A = (k_A - 1) \cdot k_R \cdot h(m) + (k_A \cdot d_A - 1) \cdot k_R \cdot d_R \cdot r \mod q$$

which satisfies $s = s_A + s_R$ where $s = k \cdot (h(m) + d \cdot r) \mod q$ is the second signature component and

$$s_R = k_R \cdot (h(m) + d_R \cdot r) \mod q.$$

Afterwards, the authentication center A sends s_A to the reader R, which completes the signature (r, s) by calculating s_R and adding it to s_A.

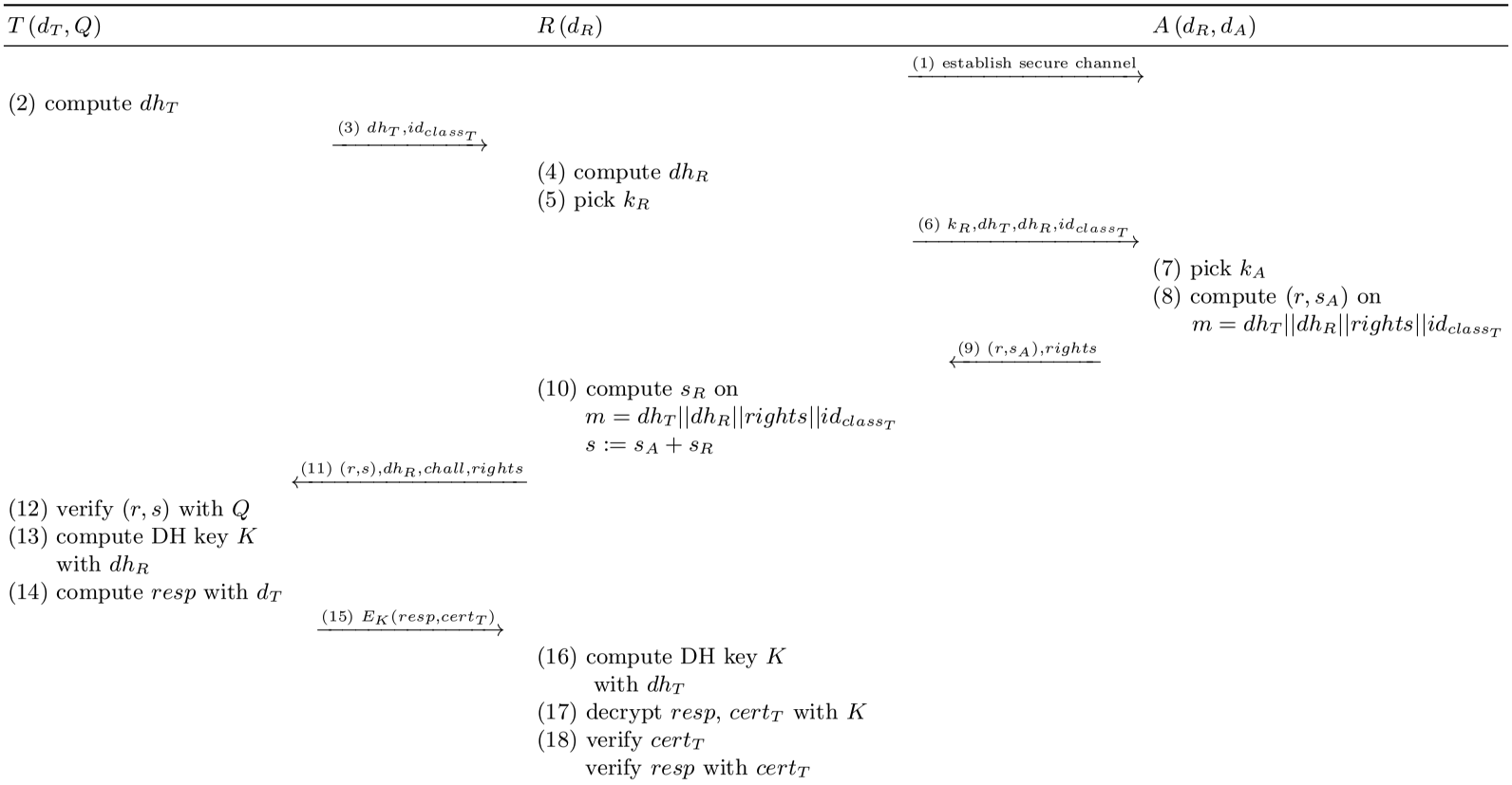
Security Analysis: An attacker cannot generate s_R without knowledge of d_R and k_R. In fact, (r, s_R) is an ECDSA signature with private key d_R, ephemeral key k_R and base point $P' = k_A^{-1} \cdot P$. The corresponding public key is $Q' = d_R \cdot (k_A^{-1} \cdot P) = d_R \cdot P'$. Two (or more) collaborating readers R and R' are not able to generate a valid signature (r, s) since they cannot reconstruct the secret key d from their parts d_R and d'_R.

3.2 The Protocol

Figure 2 illustrates the message flows and message contents for our new protocol. In the following, we explain how tag-to-reader authentication, reader-to-tag authentication, and key agreement are provided by the protocol. Finally, we discuss the relationship between the granularity of access rights to tags and the identity privacy our protocol provides.

[2] As mentioned in the introduction, we assume a secure channel between A and R, which is established prior to the authentication of the tag.

Fig. 2. Authentication Protocol



Reader Authentication: Reader authentication and checking its revocation status is based on a signature generation distributed between reader R and authentication center A. The value to be signed is dh_T, which is randomly generated by the tag (2) and sent to the reader as part of (3). The reader forwards dh_T to the authentication center as part of (6). Before generating its part of the signature, the authentication center first checks if the reader has been revoked. If this is the case, the authentication center sends an authentication failure to the tag. Otherwise, the authentication center computes its part (r, s_A) of the digital signature (r, s) on the message m including the random value dh_T (7), (8) and sends it to the reader as part of (9). The reader R completes the digital signature by computing a value s_R such that $s = s_A + s_R \mod q$ (10). After receiving (11), the tag T verifies the signature (r, s) with the help of the public signature key Q. If the signature is valid, the reader is authenticated to the tag and the tag knows that the reader is not revoked, since R was able to compute a valid signature on the challenge dh_T.

Tag Authentication: To authenticate the tag T to the reader R, a challenge-response-protocol is applied as it is described in [6]. The reader chooses a random challenge *chall* and transmits this value to the tag as part of (11). The tag computes the corresponding response *resp* with its private key d_T (14) and sends the response and its certificate back to the reader. The reader can then verify the response based on the tag's certificate $cert_T$.

Key Agreement: One major goal of the new protocol is to allow for the agreement on a common key K between the tag and the reader and subsequently use this key to establish a secure communication channel between the two. For the generation of this key K, the Diffie-Hellman protocol is used. The tag computes its Diffie-Hellman part dh_T uniformly at random (2) and sends this value to the reader (3). The corresponding part chosen by the reader in (4) is denoted by dh_R. Later, both parties T and R compute K from dh_T and dh_R (10), (13). Note that the tag's value dh_T is also used as challenge for the reader authentication as described before. The Diffie-Hellman key exchange is authenticated by including both values in the message that is signed in a distributed way by the authentication center and the reader. Upon receiving (11), the tag learns whether the reader has obtained its correct public Diffie-Hellman key part and whether itself has obtained the correct public Diffie-Hellman key part of the reader.

Rights: The rights of a reader are selected by the authentication center. To select the proper rights, the authentication center uses the identity of the reader as well as the class identity of the class the tag belongs to (as illustrated in the protocol in Figure 2). Note that the reader cannot claim another reader's identity to the authentication center as the reader authentication will then fail: the authentication center will not use the key split d_A corresponding to the reader's key split. Including the rights in the message m guarantees to the tag

that the rights were indeed set by the authentication center and were not changed by the reader. The identity of the tag needs to be included in the signed message to ensure that the reader does not obtain rights for another tag and presents them to the victim tag.

Identity Privacy: The protocol as illustrated in Figure 2 contains two messages that are crucial in the context of protecting the identity of a tag. These messages are message (3) which contains the identity of the class a tag belongs to (id_{class_T}) and message (15) which contains the encrypted certificate of the tag, and thereby its identity. In (15), the tag and the reader already share an authenticated symmetric encryption key, i.e., the tag's identity is not revealed to anyone but authenticated readers. The class of a tag indicates the tag type with respect to the access rights different readers have to the data stored on the tag, and thereby allows the authentication server to determine the rights of a reader. The identity protection our protocol provides against unauthenticated readers and passive eavesdroppers is therefore proportional to the number of tags in a specific class.

4 Authentication in a Visited Domain

In this section, we detail our new authentication protocol for the case that a tag is in a visited domain. In this case, the tag T, its home authentication center A^*, the authentication center of the visited domain A, and a reader R in the visited domain are involved in the protocol. Reader-to-tag authentication is based on a signature which is distributively generated by A^*, A, and R. In the following, we first explain how the key splitting and the novel signature scheme introduced in the last section can be extended to a multi-party signature scheme and then describe how we make use of it in our new protocol.

4.1 Multi-Party Signatures

The private key corresponding to A^*'s public key is denoted by d^* and split by A^* in two parts (d, d_{A^*}) satisfying $d^* = d \cdot d_{A^*} \mod q$. A^* provides d to A and keeps d and d_{A^*} for itself. In addition, A splits d into two parts (d_A, d_R) and provides d_R to reader R while keeping both parts for itself.

For the signature generation, the authentication center A^* chooses the ephemeral key part k_{A^*} and obtains the key part k from the authentication center A in the visited domain. In turn, k is determined as $k = k_R \cdot k_A \mod q$ by the authentication center A such that A knows k_R and k_A while the reader only knows its part k_R.

In a first step, the authentication center A^* computes the first signature component r^* as the affine x-coordinate of the point $(k^*)^{-1} \cdot P$ modulo q. The goal is now to determine the second signature component

$$s^* = k^* \cdot (h(m) + d^* \cdot r^*)$$

through distributed signature generation. The authentication center A^* starts by computing

$$s_{A^*} = (k_{A^*} - 1) \cdot k \cdot h(m) + (k_{A^*} \cdot d_{A^*} - 1) \cdot k \cdot d \cdot r^* \mod q.$$

The remaining part $s = s^* - s_{A^*}$ satisfies the equation

$$s = k \cdot (h(m) + d \cdot r^*) \mod q,$$

which means that s is a valid second signature component on the base point $k_{A^*}^{-1} \cdot P$. Hence, the authentication center A and the reader R can compute s in a distributed way like in the protocol described in the last Section 3, i.e., the authentication center A computes

$$s_A = (k_A - 1) \cdot k_R \cdot h(m) + (k_A \cdot d_A - 1) \cdot k_R \cdot d_R \cdot r^* \mod q$$

while the reader R completes $s = s_A + s_R$ by calculating

$$s_R = k_R \cdot (h(m) + d_R \cdot r^*) \mod q.$$

Finally, the reader R completes the overall signature s^* by computing

$$s^* = s_{A^*} + s = s_{A^*} + s_A + s_R.$$

4.2 The Protocol in the Visited Domain

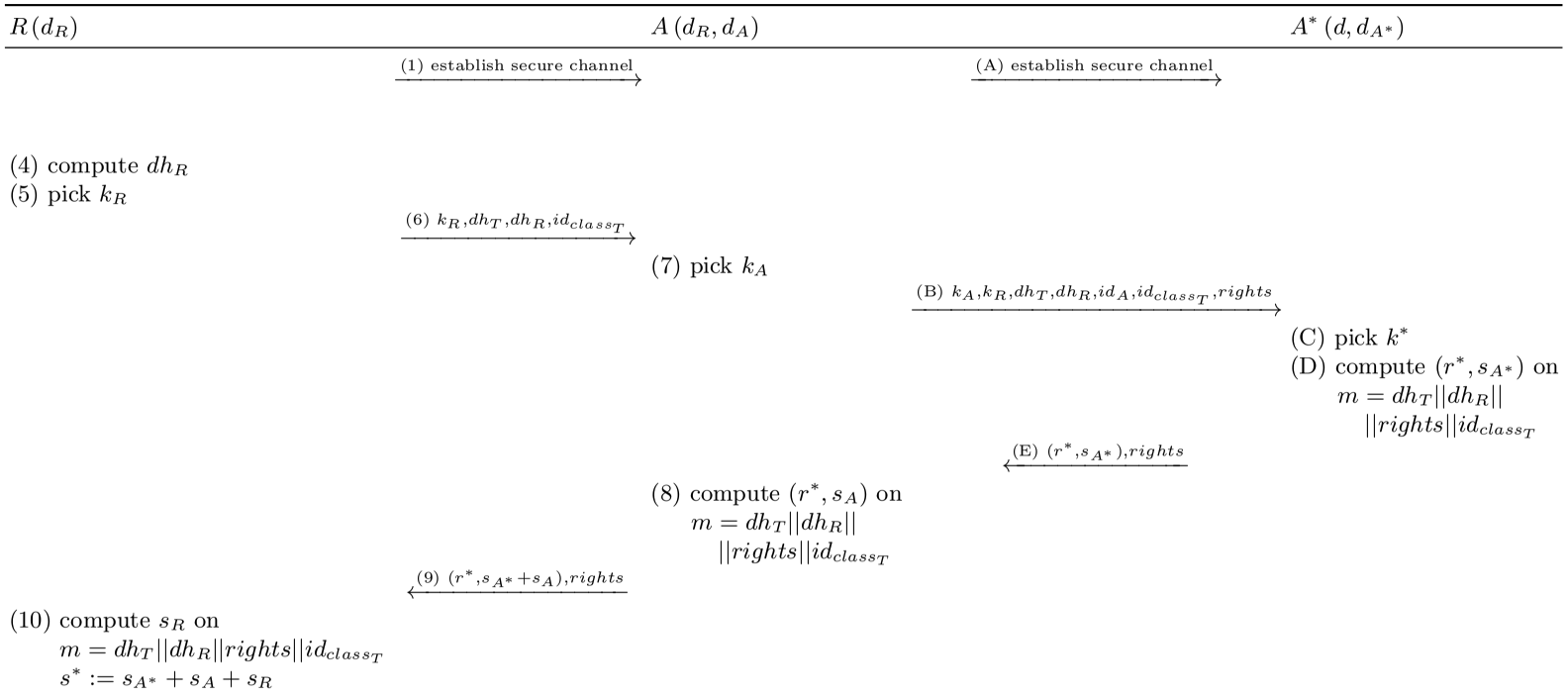
In order to embed the two-party signature generation into our protocol, we only have to replace protocol steps (4) - (10) in Figure 2 by the extended steps (4) - (10) and (A) - (E) as illustrated in Figure 3. The rest of the protocol description remains unchanged. Especially, the RFID tag has to execute exactly the same operations as if it was read in its home domain with the only difference that the signature to be verified by the tag is now denoted by (r^*, s^*).

5 Implementation Aspects

In this section we show that our protocols can in fact be implemented on RFID tags, i.e., they can meet the low power consumption and limited chip area constraints of an RFID tag.

The main operations a tag has to carry out in our protocols are: symmetric encryption, elliptic curve Diffie-Hellman key exchange, ECDSA signature verification, and response generation for tag authentication. In the following, we provide upper bounds on the gate equivalents (GEs) required for implementing these operations on RFID tags. It is reasonable to expect that an actual implementation of our protocols can leverage on synergies between the single operations with the result that the overall size of the footprint of our protocols will be smaller than the sum of the footprints of the individual operations.

Fig. 3. Multi-Party Signature Generation



5.1 Symmetric Encryption

A lot of research has been done in the area of block ciphers for resource-constrained devices (see [4, 8, 11, 14]). There are efficient implementations that require less than 3000 GE. The approaches for block cipher implementations on RFID tags can be divided into two classes. Either new algorithms are designed based on very lightweight operations such as xor, shifts, and vector products, or existing algorithms are optimized such that they become suitable for resource-constrained devices. Revising standard algorithms (such as the AES) for efficient hardware realization often leads to implementations which are less compact than implementations of proprietary algorithms. However, relying on proprietary algorithms bears a higher risk of flaws in their security design.

5.2 Scalar Multiplication

Scalar multiplication is a main operation in the elliptic curve Diffie-Hellman key exchange, the ECDSA signature verification, and in the response generation. The authors of [3] introduce an efficient implementation of scalar multiplication on curves with binary characteristic and provide a proof-of-concept by implementing the tag-to-reader authentication on a prototype RFID tag. The implementation is based on Montgomery's method for scalar multiplication, which we denote by $(X_2, Z_2) \leftarrow \texttt{Mul}(k, x_1)$ in the following. For the affine x-coordinate of a point $P_1 = (x_1, y_1)$ and a scalar k, Montgomery's method computes (X_2, Z_2) such that $x_2 = X_2/Z_2$ is the affine x-coordinate of the point $P_2 = k \cdot P_1$. The advantage of this algorithm is that it does not require any expensive inversions in a finite field. The multiplication of two finite field elements can be implemented in hardware by simple linear feedback shift register operations. The overall size of the elliptic curve engine for scalar multiplication in [3] is about 13000 GE.

5.3 Elliptic Curve Diffie-Hellman Key Exchange

The tag first selects a random λ and computes $dh_T = (X_\lambda, Z_\lambda) \leftarrow \texttt{Mul}(\lambda, x_P)$ where x_P denotes the x-coordinate of the base point P. The reader also chooses a random value μ and computes its own Diffie-Hellman part $dh_R = x_\mu$ where x_μ is the affine x-coordinate of $(x_\mu, y_\mu) = \mu \cdot P$. After receiving $dh_R = x_\mu$, the RFID tag computes $(X_{\mu\lambda}, Z_{\mu\lambda}) \leftarrow \texttt{Mul}(\lambda, x_\mu)$ and uses $K = X_{\mu\lambda}$ as symmetric encryption key. After encrypting the response $resp$ and its certificate $cert_T$ with K, the tag transmits $E_K(resp, cert_T)$ together with the value $Z_{\mu\lambda}$ to the reader. Then using $dh_T = (X_\lambda, Z_\lambda)$ the reader first calculates the affine x-coordinate $x_\lambda = X_\lambda/Z_\lambda$, computes $(X'_{\mu\lambda}, Z'_{\mu\lambda}) \leftarrow \texttt{Mul}(\mu, x_\lambda)$, and finally reconstructs the shared key $K = X'_\lambda/Z'_\lambda \cdot Z_{\mu\lambda}$.

5.4 ECDSA Signature Verification

The major task for the RFID tag in our new protocol is the signature verification. In [6], a hardware efficient method for verifying ECDSA signatures is proposed.

The signature verification of a hashed message $h(m)$ can be reduced to three executions of Montgomery's method `Mul` and evaluating a short polynomial of degree 2. Following this approach, we avoid expensive inversions in the binary field and the computation of long integers modulo the order of the base point P.

During signature generation and verification, a hash function h is applied to the message $m = dh_T||dh_R||rights||id_{class_T}$. Since the digital signature will be used for reader-to-tag authentication, the security requires no collision resistance, which means that the output length of at least 80 bits offers a sufficient security level for our purpose. In [5], an overview of hardware-efficient implementations of hash functions is given. The authors show that a construction of hash functions based on block ciphers is a reasonable approach and can be implemented with approximately 4000 GE. Using the same block cipher as for symmetric encryption is expected to result in a reduction of gate equivalents.

5.5 Response Generation for Tag Authentication

The challenge-response procedure in our new protocol is based on a Diffie-Hellman key exchange. The response generation requires one scalar multiplication. The reader generates a random point $P_1 = (x_1, y_1)$ by multiplying the base point P with a random value ξ and sends the x-coordinate $chall = x_1$ as challenge to the tag. The RFID tag runs $(X_2, Z_2) \leftarrow \texttt{Mul}(d_T, x_1)$ where d_T indicates the tag's private key. The corresponding response is $resp = (X_2, Z_2)$. A detailed evaluation of the tag-to-reader authentication is provided in [3].

6 Conclusion and Future Work

In this paper, we introduced new mutual authentication and key agreement protocols targeted at multi-domain RFID systems. The protocols are based on public-key primitives and allow for a simple reader revocation that neither relies on time synchronization between tag and reader nor on certificate revocation status checks. The new protocols are efficient in the sense that they require only one signature verification to be performed by the tag where the signature verification key is the same regardless of whether the domain in which the tag is read is its home domain or any other visited domain. In addition, our protocols allow the tag's identity to be hidden from unauthorized readers. Future work includes (1) implementing the protocols and evaluating their practical performance particularly in the multi-domain case and (2) specifying how rights can efficiently be encoded for different application scenarios.

References

1. G. Avoine, L. Buttyan, T. Holczer, and I. Vajda. Group-Based Private Authentication. In *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*. IEEE, 2007.

2. G. Avoine, E. Dysli, and P. Oechslin. Reducing Time Complexity in RFID Systems. In *Selected Areas in Cryptography*. Springer, 2005.
3. H. Bock, M. Braun, M. Dichtl, J. Heyszl, E. Hess, W. Kargl, H. Koroschetz, B. Meyer, and H. Seuschek. A Milestone Towards RFID Products Offering Asymmetric Authentication Based on Elliptic Curve Cryptography. In *RFIDSec2008 — Proceedings of the 4th Workshop on RFID Security, July 9-11, 2008, Budapest, Hungary*, July 2008.
4. A. Bogdanov, L.R. Knudsen, G. Leander, C. Paar, A. Poschmann, M.J.B. Robshaw, Y. Seurin, and C. Vikkelsoe. Present: An Ultra-Lightweight Block Cipher. In *CHES 2007*, pages 450–366. Springer-Verlag, 2007.
5. A. Bogdanov, G. Leander, C. Paar, A. Poschmann, M.J.B. Robshaw, and Y. Seurin. Hash Functions and RFID Tags: Mind the Gap. In *CHES 2008*, pages 283–299. Springer-Verlag, 2008.
6. M. Braun, E. Hess, and B. Meyer. Using Elliptic Curves on RFID Tags. *International Journal of Computer Science and Network Security*, 2:1–9, February 2008.
7. L. Buttyan, T. Holczer, and I. Vajda. Optimal Key-Trees for Tree-Based Private Authentication. In *Privacy Enhancing Technologies*. Springer, 2006.
8. M. Feldhofer, S. Dominikus, and J. Wolkerstorfer. Strong Authentication for RFID Systems Using the AES Algorithm. In *CHES 2004*, pages 357–370. Springer-Verlag, 2004.
9. Bundesamt für Sicherheit in der Informationstechnik. Advanced Security Mechanisms for Machine Readable Travel Documents: Extended Access Control (EAC), Password Authenticated Connection Establishment (PACE), and Restricted Identification (RI), Version 2.02, 2009.
10. R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Robust Threshold DSS Signatures. In *Advances in Cryptology — EUROCRYPT 1996*, London, UK, 1996. Springer-Verlag.
11. D. Hong, J. Sung, S. Hong, J. Lim, S. Lee, B.-S. Koo, C. Lee, D. Chang, J. Lee, K. Jeong, H. Kim, J. Kim, and S. Chee. HIGHT: A New Block Cipher Suitable for Low-Resource Device. In *CHES 2006*, pages 46–59. Springer-Verlag, 2006.
12. S.K. Langford. Threshold DSS Signatures without a Trusted Party. In *Advances in Cryptology — CRYPTO 1995*, pages 397–409, London, UK, 1995. Springer-Verlag.
13. M. Li, R. Poovendran, R. Falk, A. Köpf, K. Sampigethaya, R. Robinson, S. Lindelman, M. Braun, and H. Seuschek. Multi-Domain RFID Access Control Using Asymmetric Key Based Tag-Reader Mutual Authentication. In *ICAS2008 — Proceedings of the 26th international Congress of the Aeronautical Sciences, September 14-19, 2008, Anchorage, USA*, September 2008.
14. C. Lim and T. Korkishko. mCryption — A Lightweight Block Cipher for Security of Low-cost RFID Tags and Sensors. In *WISA 2005*, pages 243–258. Springer-Verlag, 2005.
15. P. MacKenzie and M. K. Reiter. Two-Party Generation of DSA Signatures. In *Advances in Cryptology — CRYPTO 2001*, London, UK, 2001. Springer-Verlag.
16. U. Meyer, J. Cordasco, and S. Wetzel. An Approach to Enhance Inter-Provider Roaming through Secret Sharing and its Application to WLANs. In *WMASH'05 — Proceedings of the 3rd ACM International Workshop on Wireless Mobile Applications and Services on WLAN Hotspots*, pages 1–13, New York, NY, USA, 2005. ACM.
17. D. Molnar and D. Wagner. Privacy and Security in Library RFID: Issues, Practices, and Architectures. In *Proceedings of the ACM Conference on Computer and Communications Security*. ACM, 2004.

A Two-Party ECGDSA Signature

The German version of the elliptic curve digital signature algorithm ECGDSA can also be used for distributed signature generation. The private signature key d and the public signature Q satisfy the equation $Q = d^{-1} \cdot P$. The inversion of the private signature key d leads to a simplified signature generation algorithm. In particular, the ephemeral key k does not need to be inverted anymore. The signature generation works as follows. The signer randomly chooses an ephemeral key k and computes the first signature component r as the affine x-coordinate of the point $k \cdot P$ modulo q. The second second component is obtained by $s = d \cdot (k \cdot r - h(m))$.

The private key and ephemeral key will be split and distributed between the authentication center A and the reader R the same way it has been described in Section 3 and 4 for ECDSA. Hence, the authentication center knows all parts of the private signature key d_A, d_R and of the ephemeral key k_A, k_R, where the reader knows the parts d_R and k_R only.

For the distributed signature generation, the authentication center A computes r, the affine x-coordinate of $k \cdot P = (k_A \cdot k_R) \cdot P$ modulo q, and its part

$$s_A = d_R \cdot k_R \cdot r \cdot (d_A \cdot k_A - 1) + d_R \cdot h(m) \cdot (1 - d_A).$$

The reader R finishes the signature $s = d \cdot (k \cdot r - h(m)) = s_A + s_R$ by computing

$$s_R = d_R \cdot (k_R \cdot r - h(m)).$$