Threshold-Based Distance Bounding

Roel Peeters Dave Singelée Bart Preneel
K.U.LEUVEN - ESAT/COSIC & IBBT
Kasteelpark Arenberg 10 - 3001 LEUVEN, Belgium
firstname.lastname@esat.kuleuven.be

Abstract

Conventional access control mechanisms usually rely on the use of a single token for user authentication, and are generally vulnerable to relay attacks. In addition, these systems often suffer from usability issues. Since they are centrally managed, key management tends to be a rather slow and cumbersome process in this setting.

In this paper, we propose a threshold-based location-aware access control mechanism. It combines the concepts of secret sharing and distance bounding protocols to tackle various security vulnerabilities. Our proposed solution offers protection against any set of $(t-1)$ compromised user's devices, with t being an adjustable threshold number. It supports user-centered management, since users can alter the set of personal devices, and can tune the security parameters of the access control scheme towards the required level of security and resilience.

1. Introduction

Nowadays, access to secure facilities and buildings is often enforced using contactless smartcards. Each legitimate person carries his own personal token containing identifying information and a secret key. When a user wants to enter the building, he puts his contactless smartcard close to a reader installed in the proximity of the door. Both devices will then carry out a challenge-response protocol, in which the user's smartcard identifies itself to the reader (in some scenarios, mutual authentication is required). If the protocol finishes successfully, the user is granted access to the building. A similar mechanism is sometimes used to enter a car [18], to use public transport (e.g., [19–21]), and even for payments with contactless credit cards [15, 34].

Although widely used, this solution has some important drawbacks. It has several important security vulnerabilities: it creates a single point of failure in the system and is vulnerable to relay attacks. In addition this solution is also not user-friendly.

The use of single security token introduces a single point of failure. If the security token gets lost or stolen, an unauthorized attacker could get access to a secure building or resource. Security tokens and smartcards could also be compromised or cloned (e.g., the MIFARE attack discovered by Koning Gans, Hoepman and Garcia [10]). When an attack on a particular token or smartcard is detected, it must be revoked in order to patch the security breach.

Another important security vulnerability in various access control systems are *relay attacks*, also known as *mafia fraud attacks*. This is a man-in-the-middle attack where a verifier (e.g., the reader next to the door of a building) is tricked in believing that a prover (e.g., the smartcard) is in its close vicinity by surreptitiously forwarding the signal between the verifier and an out-of-range prover [14].

For each system, the user has to carry around a separate smartcard or security token. This is not always very convenient. A legitimate user that does not carry around the security token, automatically has no access. Furthermore, revocation of a particular token is often a cumbersome and relatively slow process. This is illustrated by the following plausible scenario. When initiating the revocation process, the user first has to inform the facility manager. Second, revocation lists have to be updated and distributed. Third, the user has to get a new token or smartcard. Since such a revocation process is slow, it also poses a security risk, since there is a grace period in which the attacker can still use the token before the revocation lists are updated.

Fortunately, both security vulnerabilities can be tackled by introducing several countermeasures. The single point of failure can be removed by introducing several devices that share a secret. The vulnerability against relay attacks can be solved by using distance bounding protocols. Secret sharing also provides resilience to the user and allows, through the mechanism of resharing, user-centered access control.

1.1 Secret sharing

Secret sharing was first introduced by Shamir [29]. Instead of storing a secret on one device, the secret is divided into k pieces, each stored on a different device, in such a way that the key can easily be reconstructed from any t pieces, with t a threshold number chosen by the user, but even complete knowledge of $(t-1)$ pieces reveals absolutely no information about the secret. If a device gets stolen, there is hence no need for revocation as the attacker obtains no information about the secret. Secret sharing also allows for user-centered access control, since the user can decide which devices get a piece of the key, how large the threshold value t should be, and when the secret shares should be updated (i.e. when the resharing process should take place).

1.2 Organization of the paper

This paper is organized as follows. In the introduction, we showed that conventional access control mechanisms suffer from several security vulnerabilities. We put forward the idea of employing distance bounding protocols in combination with secret sharing to solve several of these security issues. The general principles of distance bounding protocols are discussed more in detail in Section 2. Section 3 describes our threshold based access control scheme, which uses distance bounding protocols to tackle relay attacks, more in detail. In this section we also describe how a user can manage the access control configuration. In Section 4, we discuss the security properties of our solution more in detail. Section 5 concludes the paper.

2. Distance bounding protocols

Distance bounding protocols, which have been introduced by Brands and Chaum [3], are employed to enhance entity authentication protocols by incorporating location information. The concept of *proximity based authentication* is graphically depicted in

Second International Workshop on Security and Privacy in Spontaneous Interaction and Mobile Phone Use (IWSSI/SPMU)
May 17, 2010, Helsinki, Finland

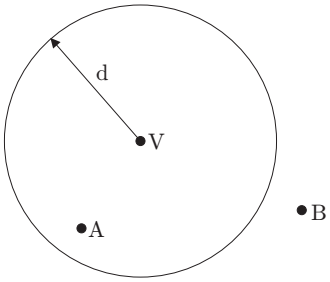

Figure 1. The concept of proximity based authentication.

Figure 1. Authentication requests originating from devices that are located within the range *d* of the verifier *V* are accepted, all other requests are rejected.

Distance bounding protocols combine physical and cryptographic properties to enable a verifying party to determine an upper bound on the distance between itself and a prover, who claims to be within a certain range. The basic idea is to measure the time of flight during a challenge-response protocol, which is the main building block of the distance bounding protocol. During *n* fast bit exchanges, the time between sending a challenge and receiving the corresponding response is measured. Multiplying the time of flight with the propagation speed of the communication medium (i.e. the speed of light when RF communication is used) gives an estimation on the distance between prover and verifier. If one can prevent an attacker being physically located between the prover and the verifier, one can use slower communication media (such as ultra-sound).

By employing the principle of distance bounding attacks in a clever way, one can preclude one or more of the following attacks:

Distance fraud attacks: One wants to prevent a dishonest prover claiming to be closer than he really is. This attack is called distance fraud attack and is conceptually shown in Figure 2. It is relatively easy to design a protocol which prevents this type of attack.

Figure 2. Distance fraud attack

Mafia fraud attacks: These were first described by Desmedt [5]. In this attack scenario, both prover and verifier are honest, but a malicious intruder is performing the fraud. This is a man-in-the-middle attack where the intruder *I* is modeled as a malicious prover $\bar{P}$ and verifier $\bar{V}$ that cooperate, as shown in Figure 3. The malicious verifier $\bar{V}$ interacts with the honest prover *P* and the malicious prover $\bar{P}$ interacts with the honest verifier *V*. The physical distance between the intruder and the verifier is small. This attack enables the intruder to identify himself to *V* as *P* being close to *V*, without any of *P* and *V* noticing the attack. Drimer and Murdoch [8] have presented a practical mafia fraud attack on the United Kingdom's EMV payment system Chip & Pin.

Terrorist fraud attacks: Terrorist fraud attacks [5] are an extension of mafia fraud attacks. The intruder (being close to the verifier) and the prover will collaborate in this attack. This implies that a protocol which is resistant to terrorist fraud attacks, also prevents mafia fraud attacks. The terrorist fraud attack is shown in Figure 4. The intruder must not know the private key of the prover, since the latter does not fully trust the former. If the intruder would know this private key, then it is impossible to make a distinction between the intruder and the prover, and as a result, terrorist fraud attacks can no longer be prevented. They would be the same party from a cryptographic point of view, because distance bounding protocols only check if an entity that knows the private key is close to the verifier.

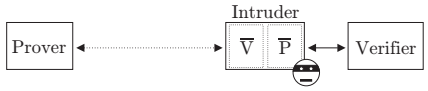

Figure 3. Mafia fraud attack

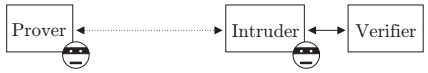

Figure 4. Terrorist fraud attack

In this paper, we will not focus on terrorist fraud attacks, as this would cause major overhead in our threshold based solution (which will be discussed later in section 3), and only concentrate on secure distance bounding protocols which prevent distance and mafia fraud attacks. Although the idea has been introduced more than fifteen years ago by Brands and Chaum [3], it is only quite recently that distance bounding protocols attracted the attention of the research community. Hancke and Kuhn [11] pointed out that distance bounding protocols should be designed to cope well with substantial bit error rates during the rapid single bit exchanges, as these are conducted over noisy wireless ad hoc channels. They incorporated this important requirement in the design of their RFID distance bounding protocol. Singelée and Preneel [32] have proposed a noise resilient extension of the Brands-Chaum protocol that provides mutual entity authentication. Later, various other distance bounding protocols have been proposed [4, 16, 25, 27, 30, 33]. A short overview can be found in [1, 14].

3. Location-aware access control

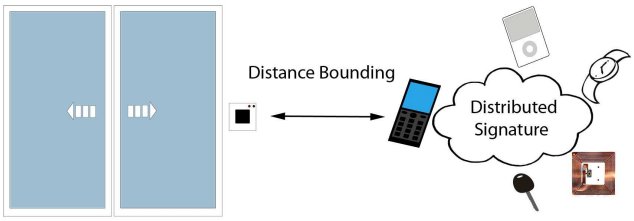

Figure 5. Distributed access control setting

We focus on a location-aware access control mechanism where access is granted based on the combination of identity and proximity information. The setting is illustrated in Figure 5, where access control is enforced to enter a particular building. The user has a group of devices that will carry out a proximity-based authentication protocol with a verifying entity, e.g., a reader placed next to the door of the building. The former will be denoted by the *prover*, the latter by the *verifier*. If the protocol finished successfully, the user can enter the building. Only one of the user's devices will communicate directly with the verifying entity. This device is denoted

by the *gateway device* in the rest of the paper. The gateway device is also responsible for initiating the access control mechanism. The other devices of the user are called *end-devices*. They will also contribute in the access control scheme, but will not communicate directly with the verifier. We envision the user's mobile phone to act as gateway device, but the user can also choose to use one of his other devices. The only requirements for the gateway device is that it should have sufficient computational resources (to carry out cryptographic algorithms such as hash functions and digital signatures) and that it has an adequate user-interface. Note that it is not required to use the same gateway device in different instances of the protocol. However, in each run of the protocol, there is always exactly one gateway device, all other devices act as end-device. Since only the gateway device will communicate directly with the verifying entity, the end-devices do not need to be able to carry out a distance bounding protocol. The gateway device can use onboard or separate dedicated hardware (connected to the device) to perform the fast bit exchanges of the protocol.

As discussed in Section 2, by carrying out a distance bounding protocol, a verifying party can determine an upper bound on the distance between itself and a prover, who claims to be within a certain range. Cryptographic distance bounding protocols often require the prover and the verifier to compute a non-probabilistic function on a known input and a shared secret, known to both the prover and the verifier. Typically a pseudorandom function [17] such as *HMAC* [2] or *CBC-MAC* [12] is used. Such functions cannot be used in our access control mechanism, as we have chosen for a distributed solution where a gateway device and several end-devices need to collaborate to successfully complete the protocol. Instead, we will distribute the secret among the user's devices, and replace the pseudo-random function by a cryptographic function which can be partially evaluated by each of the user's devices[1]. Afterwards, the gateway device can then combine several of these partial evaluations to obtain the evaluation of the cryptographic function. An interesting function that has all these properties is the computation of an RSA signature [26].

Through the mechanism of resharing, we support user-centered management. Users can alter the set of personal devices. They can add or remove devices from this set, and can tune the security parameters of the access control scheme towards the required level of security and resilience.

3.1 Adversarial model and assumptions

Each device has a public-private key pair. Public keys of potential gateway devices are known to all devices. All devices' public keys are known to devices that can contribute in resharing. How to register the public key with other devices is out of the scope of this paper. E.g., one can use pairing protocols to authenticate the public key. These protocols are standardized in the ISO/IEC 9798-6 standard [13].

The goal of the attacker is to get unauthorized access (e.g., enter a building). To achieve this objective, the attacker can perform passive and/or active attacks. The attacker, which is computationally bounded, can largely extend his communication range, and send/receive messages from a large distance. However, we assume that the attacker cannot increase the propagation speed of the communication medium (i.e. we implicitly assume that RF communication is used).

There are two scenarios that we need to investigate: the prover being located at a large distance from the verifier, and the prover being in the proximity of the verifier. The attacker (i.e. the person in control of the attacks) is assumed to be physically close to the verifier in both scenarios (otherwise, an attack would not make much sense from a practical point of view). Let us first focus on the first scenario. To get unauthorized access, the attacker needs to carry out a mafia fraud attack and forward all messages to a proxy device that is hidden in the neighborhood of the prover. An attacker can also compromise a set of the user devices and use these devices to perform the mafia fraud attack (i.e. sign particular data). The set of compromised devices is however assumed to be strictly smaller than t. As we will show later in the paper, our solution is resistant to such mafia fraud attacks. In the second scenario, where the user's devices are close to the verifier, we assume that the user can physically verify the presence of the attacker. We hence mainly need to focus on the first scenario.

3.2 Our threshold-based solution

We will now present our threshold-based access control solution that uses distance bounding protocols. When carrying out our scheme, several steps needs to be performed. Initially, each new user needs to be registered (the initialization process will be further discussed in Section 3.2.1). After this phase, which only needs to be done once for each new entity, the user is ready to actively use the access control mechanism. Access control is carried out by conducting a distance bounding protocol based on digital signatures (see Section 3.2.2). One of the main components of this protocol is a distributed RSA signature generation (this will be discussed in Section 3.2.3). For security and usability reasons, the private key used to compute the distributed RSA signature needs to be reshared at regular intervals (see Section 3.2.4).

3.2.1 Enrollment phase

Before a new user can actively use the access control mechanism, he first has to enroll. This enrollment phase is rather similar to the initialization phase of a conventional access control scheme, but there are some important differences. Instead of generating a shared secret key and storing it on the user's smartcard, the verifier will generate a shared private RSA key that will be distributed among the personal devices of the user. There are two options. The verifier can send the private key to the gateway device, which will then share the key among the user's devices; each device will receive a share d_i. Next the gateway device should delete the private key. Another solution is that the verifier itself acts as the trusted dealer, since it knows the private key.

The initial secret sharing phase goes as follows. The trusted dealer first broadcasts the public RSA parameters e, N. Next it computes and distributes the random number v and the initial verification keys $v_1 \dots v_k$ (with $v_i = v^{d_i}$). Each device i then gets its initial share d_i of the private key over a private channel. Techniques on how to construct a private channel are out of the scope of this paper. The verification keys are stored by all the user's devices. These are used to verify the gateway device's knowledge of its share, needed for the distributed signature generation. These are also used during resharing.

3.2.2 Distance bounding using digital signatures

As already discussed, we have opted to use a threshold-based distance bounding protocol to enable a verifying party to check that a particular device is within a certain range. Instead of only one prover, we have a group of personal devices that will collaborate during the proximity-based authentication process. As a starting point, we used the Hancke-Kuhn protocol [11], as it is noise resilient and does not require the computation of a signature at the end of the protocol. We slightly modified the protocol such that instead of a pseudo-random function, prover and verifier need to compute an RSA signature S on a message M, which depends on the nonces exchanged between prover and verifier. We will use RSA signatures

[1] One only needs a threshold number of cooperating devices to reconstruct the secret and hence evaluate the cryptographic function.

as defined in PKCS #1 version 2.1 [23], but for simplicity reasons denote the encoded message as M, The resulting distance bounding protocol is depicted in Figure 6.

The protocol works as follows. Initially the user has to confirm that he wants to start the access control mechanism, by performing a particular action (e.g., pressing a button) on the gateway device. After this approval by the user, the distance bounding protocol can start.

Next, the prover and verifier exchange a random nonce, N_P and N_V respectively. Both parties then compute an RSA signature *S*, using the private key *d*, which is known to both parties, on the message M, which is the result of applying a cryptographic hash function on the concatenation of N_V and N_P.

$$S = M^d \bmod N \qquad M = h(N_v||N_p) \ .$$

After applying a deterministic reduction[2] function RED on the signature, the result is split in two n-bit sequences $r^{(0)}$ and $r^{(1)}$.

$$RED(S) = r^{(0)}||r^{(1)} \ .$$

The verifier can directly compute the RSA signature on the message. However, at the prover's side the private key *d* is shared among the user's personal devices. The RSA signature on the message is generated in a distributed way. More details on the distributed RSA signature generation can be found in the next section.

Last, a series of *n* fast bit exchanges is performed. In each round, the verifier sends a random single bit challenge C_i to the prover. If this challenge equals 0, then the prover responds with the i-th bit of $r^{(0)}$. If the challenge equals 1, then the prover sends the i-th bit of $r^{(1)}$. In each round, the verifier measures the time between sending C_i and receiving the corresponding response. The maximum round trip time is selected and this measurement determines an upper bound on the estimation of the distance between prover and verifier. If at least $(n - x)$ of the responses sent by the prover are correct (the security parameter *x* denotes the number of allowed bit errors during the rapid bit exchange), the protocol succeeds.

3.2.3 Distributed RSA signature generation

Desmedt and Frankel [6] proposed the first (non-robust) threshold RSA signature scheme. Later robust, but less practical threshold RSA signatures schemes were proposed by Frankel *et al.* [9] and Rabin [24]. The first practical, robust threshold RSA signature scheme was presented by Shoup [31]. We will use this technique to generate the signature S on the message M.

The gateway device will initiate the distributed RSA signature generation. This device broadcasts the message M together with a zero knowledge proof (denoted by *ZKP*) of its share. The ZKP consists of a Schnorr signature [28] (c, z) on the message and the public RSA key e. The gateway device chooses r at random in the interval $\{0 \ldots 2^{L(N)+2L_1-1}\}$, with $L(N)$ the bitlength of N and L_1 a secondary security parameter.

$$v' = v^r \qquad c = H(v', M, e, N) \qquad z = d_i c + r \ .$$

The end-devices will first verify the ZKP, by computing v', using the public verification key v_i, and checking c:

$$v' = v^z v_i^{-c} \qquad\qquad c = H(v', M, e, N) \ .$$

Note that by carrying out this ZKP, one has the guarantee that the distributed signature generation is initiated by a (trusted) gateway device, and not by a (hidden) proxy device controlled by the attacker. Next, the end-devices broadcast their partial signatures S_i encrypted with the public key of the gateway device.

$$S_i = M^{2\Delta d_i} \ .$$

The result is that only the gateway device can combine the partial signatures (using a subset of t devices, with t being the secret sharing threshold) into the signature on the message M. Let λ_i be the Lagrange multipliers multiplied with $\Delta = \prod_{i=1}^{k} i$. This is necessary to get integer values, since the order of the subgroup $\varphi(N)$ is unknown.

$$w = \prod_i S_i^{\lambda_i} \qquad \lambda_i = \Delta \prod_{j \neq i} \frac{j-i}{j} \ .$$

We now have $w^e = M^{4\Delta^2}$. Since e and $4\Delta^2$ are coprime, we can find values α and β, for which $\alpha 4\Delta^2 + \beta e = 1$, using Euclid's algorithm.

$$S = w^\alpha M^\beta \ .$$

The gateway device can verify the correctness of the signature S using the public RSA parameters e, N.

$$S^e \bmod N = M \ .$$

If the verification of the signature does not hold, the subset contains one or more cheating devices. The gateway device then selects a different subset of t devices and recomputes the signature. This process is repeated until the verification of S succeeds.

3.2.4 Resharing

By carrying out secret resharing, new shares of the secret are generated and the old shares are rendered useless. This means that an adversary is forced to break the scheme within the time frame between two consecutive resharings. Resharing also allows to go from a (t, k)-secret sharing to a (t', k')-secret sharing, with t' the new threshold number and k' the new number of participants. The threshold number *t* determines the level of security, i.e., the number of devices an adversary needs to compromise within the available timeframe. The number of devices k together with the threshold number t determines the level of resilience, since the legitimate user will still be able to use the scheme when the combined number of devices that are not present and compromised by an adversary is less than $(k - t)$.

Because of the resharing mechanism, our access control solution supports user-centered management. The set of personal devices, of which a subset of at least t devices is needed during the authentication phase, can be changed, hence signature rights can be revoked and/or granted. Resharing is typically required when a device gets stolen, or when a user purchases a new device. From the moment a device is identified as compromised, it should be excluded from the set of devices that share the private RSA key. It will hence not receive a new share and its old share will be rendered useless.

Secret resharing, without reconstruction of the secret, was first described by Desmedt *et al.* [7] and Frankel *et al.* [9]. We will use the techniques proposed by Wong *et al.* [35], where new participants can verify the validity of their shares.

The resharing mechanism works as follows. Basically, every contributing device constructs a polynomial of degree (t' - 1), with t' the new threshold number. Commitments to the coefficients c_{ij} of the polynomial are broadcast.

$$f_i(x) = d_i + c_{i1}x + \ldots + c_{i(k'-1)}x^{t'-1}$$

$$\forall j \in \{1, \ldots, t'\} : C_{ij} = v^{c_{ij}} \ .$$

Subshares (evaluations of the polynomial $f_i(x)$) are handed out to the set of participating devices:

$$\forall j \in \{1, \ldots, k'\} : d_{ij} = f_i(j) \ .$$

[2] The reduction function RED reduces the bitlength of the input to a fixed bitlength $2n$.

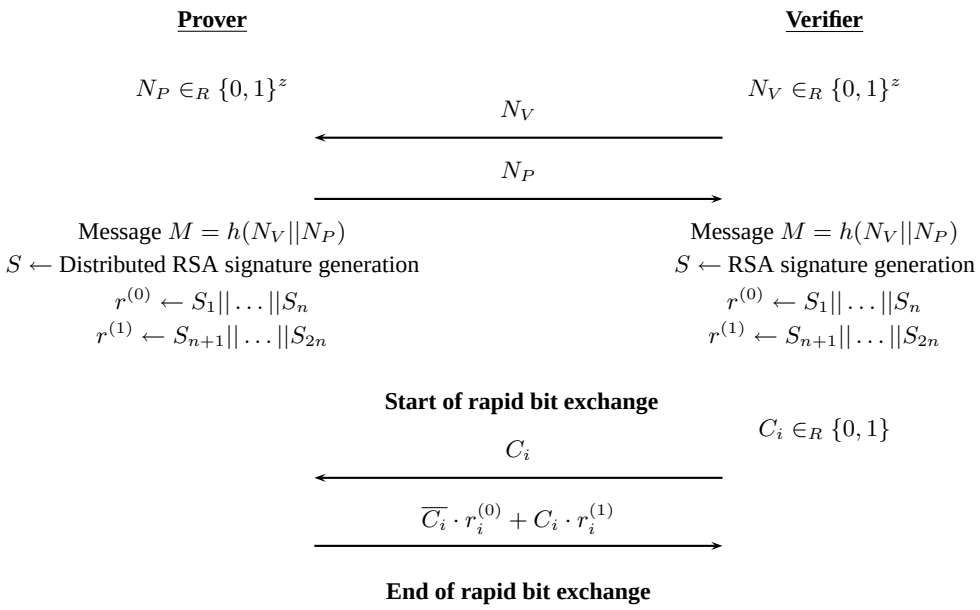

Figure 6. Threshold-based distance bounding protocol

The commitments to the coefficients of the polynomial and verifications keys allow to validate these subshares.

$$v^{d_{ij}} = v_i \prod_{l=1}^{t'-1} C_{il}^{j^l} \quad .$$

The subshares are combined into the new shares. Next, new verification keys are broadcasted.

$$d'_j = \frac{\sum d_{ij}\lambda_i}{\Delta} \qquad v'_j = v^{d'_j} \quad .$$

The validity of these new verification keys, hence the new shares, can be tested by combining them:

$$v^{\Delta} = \left(\prod v_i'^{\lambda_i}\right)^e \quad .$$

4. Security discussion

Due to its specific design, our threshold-based location-aware access control scheme improves the resistance to various important security vulnerabilities. Some of its security properties will now be briefly discussed (without formal proof).

Please note that in the case an adversary succeeds in compromising the threshold number of devices at one point in time, he can reconstruct the private RSA key, and hence break the access control scheme.

4.1 Relay attacks

The (slightly modified) distance bounding protocol of Hancke and Kuhn, which is the main building block of our threshold-based location-aware access control scheme, prevents relay attacks. The attacker cannot conduct a distance bounding protocol with the prover and the verifier simultaneously (i.e. forwarding the messages from the verifier to the prover and vice versa), as this would increase the time of flight (and hence cause the protocol to fail). The best attack strategy for an adversary, who wants to authenticate himself successfully to the verifier, consists of two phases.

In the first phase, the adversary first exchanges nonces with the verifier. Next he guesses all the challenges C_i in advance and carries out a distance bounding protocol with the honest prover. Since the user will not approve the execution of this distance bounding protocol, the adversary needs to compromise at least one device with a user interface (i.e. a potential gateway device). At the end of this phase, the adversary will have received n responses r_i (e.g., the compromised device can forward these responses to the adversary).

In the second phase of the attack, the adversary, who is located close to the verifier, will conduct a distance bounding protocol with the verifier. In each round, there are two scenarios. If the adversary has guessed the challenge C_i correctly, he replies with the response r_i (received from the prover). If he has guessed the challenge wrongly, he sends a random response to the verifier.

By following this strategy, an adversary has in each of the n rounds a probability of $3/4$ to send a correct response [11]. If the attack succeeds, the adversary is able to wrongfully convince the verifier that an entity in possession of the private RSA key is in the vicinity. The attack probability slightly changes when one incorporates the effect of bit errors due to noise [32].

4.2 User interface

As explained above, an attacker has to compromise at least one device with a user interface, since end-devices will only respond to signature request from a gateway device.

A compromised gateway device could carry out mafia fraud attacks, where it requests the other devices in the network to compute a distributed RSA signature. To mitigate this security risk, one could demand that the user approves this request on a predefined number of end-devices with a user interface. Analogue to the gateway device, these devices need to proof knowledge of their shares. This means an attacker would have to compromise at least this predefined number of devices with a user interface to carry out a successful mafia fraud attacks.

By varying this predefined number, one can change the attacker's success probability (and hence the security level of the protocol). Peeters et al. [22] showed that having a user verifying his request at a number of devices with a user-interface reduces the adversary's probability of success drastically. In case of an unsuccessful run, devices will keep on displaying the request, allowing

the user to identify compromised devices and hence detecting that an attack is taking place.

5. Conclusion

Contactless smartcards are often used in conventional access control mechanisms, during a challenge-response protocol. This entails several security vulnerabilities. Relying on the use of a single security token introduces a single point of failure in the system. Users cannot authenticate themselves without their token. Moreover, an attacker that steals or compromises a token will get the access privileges of the corresponding user, until revocation has taken place. Challenge-response protocols conducted in wireless networks are also vulnerable to relay attacks.

In addition to these security vulnerabilities, conventional access control mechanisms often suffer from usability issues. Since these systems are centrally managed, carrying out changes (such as revoking keys) tends to be a rather slow and cumbersome process. Users can also not tune the security properties of the access control scheme, as this is enforced by the system itself.

In this paper, we proposed a threshold-based location-aware access control mechanism. It combines the concepts of secret sharing and distance bounding protocols. Its main component is a distributed RSA signature generated by t user's devices: one gateway device and $(t-1)$ end-devices. The gateway device interacts directly with the verifier and combines the partial signatures. We particularly envision the user's mobile phone to act as gateway device in our access control scheme.

We demonstrated that our solution solves the dependency on a single token, and is resistant against relay attacks. It offers protection against any set of $(t-1)$ compromised user's devices. Compared to conventional access control mechanisms, our solution improves the user-friendliness as it supports user-centered management. Users can vary the set of personal devices, of which t need to be present during authentication. This threshold number determines the security level of our access control scheme and is tunable by the user.

Acknowledgments

This work is funded by the Katholieke Universiteit Leuven, and supported in part by the Concerted Research Action (GOA) Ambiorics 2005/11 of the Flemish Government, by the IAP Programme P6/26 BCRYPT of the Belgian State (Belgian Science Policy), and by the Flemish IBBT projects. Roel Peeters is funded by a research grant of the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen).

References

[1] G. Avoine and A. Tchamkerten. An Efficient Distance Bounding RFID Authentication Protocol: Balancing False-Acceptance Rate and Memory Requirement. In *Proceedings of the 12th International Conference on Information Security (ISC '09)*, volume 5735 of *LNCS*, pages 250–261. Springer Verlag, 2009.

[2] M. Bellare, R. Canetti, and H. Krawczyk. Keying Hash Functions for Message Authentication. In *Advances in Cryptology - CRYPTO '96*, volume 1109 of *LNCS*, pages 1–15. Springer Verlag, 1996.

[3] S. Brands and D. Chaum. Distance-Bounding Protocols. In *Advances in Cryptology - EUROCRYPT '93*, volume 765 of *LNCS*, pages 344–359. Springer Verlag, 1994.

[4] J.T.. Chiang, J.J. Haas, and Y.C. Hu. Secure and Precise Location Verification Using Distance Bounding and Simultaneous Multilateration. In *Proceedings of the Second ACM Conference on Wireless Network Security (WISEC '09)*, pages 181–192. ACM, 2009.

[5] Y. Desmedt. Major Security Problems with the "Unforgeable" (Feige)-Fiat-Shamir Proofs of Identity and how to overcome them. In *Proceedings of SecuriCom '88*, pages 15–17, 1988.

[6] Y. Desmedt and Y. Frankel. Shared Generation of Authenticators and Signatures. In *Advances in Cryptology - CRYPTO '91*, volume 576 of *LNCS*, pages 457–469. Springer Verlag, 1991.

[7] Y. Desmedt and S. Jajodia. Redistributing Secret Shares to New Access Structures and its Applications. Technical Report ISSE-TR-97-01, George Mason University, 1997.

[8] S. Drimer and S. Murdoch. Keep Your Enemies Close: Distance Bounding Against Smartcard Relay Attacks. In *Proceedings of the 16th USENIX Security Symposium*, pages 87–102. USENIX, 2007.

[9] Y. Frankel, P. Gemmell, P. D. MacKenzie, and M. Yung. Optimal Resilience Proactive Public-Key Cryptosystems. In *IEEE Symposium on Foundations of Computer Science*, volume 1294 of *LNCS*, pages 384–393. Springer Verlag, 1997.

[10] G. K. Gans, J.-H. Hoepman, and F. D. Garcia. A Practical Attack on the MIFARE Classic. In *CARDIS '08: Proceedings of the 8th IFIP WG 8.8/11.2 international conference on Smart Card Research and Advanced Applications*, volume 5189 of *LNCS*, pages 267–282. Springer Verlag, 2008.

[11] G. Hancke and M. Kuhn. An RFID Distance Bounding Protocol. In *Proceedings of the 1st International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM '05)*, pages 67–73. IEEE Computer Society, 2005.

[12] ISO/IEC 9797: Information Technology – Security Techniques – Data Integrity Mechanisms Using a Cryptographic Check Function Employing a Block Cipher Algorithm, 1994.

[13] ISO/IEC 9798-6. Information Technology – Security Techniques – Entity Authentication – Part 6: Mechanisms Using Manual Data Transfer, 2005.

[14] C. Kim, G. Avoine, F. Koeune, F. Standaert, and O. Pereira. The Swiss-Knife RFID Distance Bounding Protocol. *Information Security and Cryptology — ICISC 2008: 11th International Conference, Seoul, Korea, December 3-5, 2008, Revised Selected Paper*, pages 98–115, 2009.

[15] Mastercard PayPass. `http://www.paypass.com/`.

[16] C. Meadows, P. Syverson, and L. Chang. Towards More Efficient Distance Bounding Protocols. In *Proceedings of the 2nd International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM '06)*, 5 pages. IEEE Computer Society, 2005.

[17] A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[18] Microchip Technology Inc. KeeLoq Authentication Products. `http://www.microchip.com/keeloq/`.

[19] Octopus Cards Limited. `http://www.octopus.com.hk/`.

[20] Oyster Online - Transport for London. `https://oyster.tfl.gov.uk/`.

[21] OV-chipkaart. `http://www.ov-chipkaart.nl/`.

[22] R. Peeters, M. Kohlweiss, and B. Preneel. Threshold Things That Think: Authorisation for Resharing. In *Proceedings of iNetSec 2009 - Open Research Problems in Network Security*, volume 309 of *IFIP Advances in Information and Communication Technology*, pages 111–124. Springer Verlag, 2009.

[23] PKCS #1 v2.1: RSA Cryptography Standard, 2002.

[24] T. Rabin. A Simplified Approach to Threshold and Proactive RSA. In *Advances in Cryptology - CRYPTO '98*, volume 1462 of *LNCS*, pages 349–369. Springer Verlag, 1998.

[25] K. Rasmussen and S. Capkun. Location Privacy of Distance Bounding Protocols. In *Proceedings of the 2008 ACM Conference on Computer and Communications Security (CCS '08)*, pages 149–160. ACM, 2008.

[26] R. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[27] N. Sastry, U. Shankar, and D. Wagner. Secure Verification of Location Claims. In *Proceedings of the 2003 ACM Workshop on Wireless Security (WISE '03)*, pages 1–10. ACM, 2003.

[28] C.-P. Schnorr. Efficient Identification and Signatures for Smart Cards. In *Advances in Cryptology - CRYPTO '89*, volume 435 of *LNCS*, pages 239–252. Springer Verlag, 1989.

[29] A. Shamir. How to Share a Secret. *Communications of the ACM*, 22 (11):612–613, 1979.

[30] V. Shmatikov and M.H. Wang. Secure Verification of Location Claims with Simultaneous Distance Modification. In *Advances in Computer Science - ASIAN '07*, volume 4846 of *LNCS*, pages 181–195. Springer Verlag, 2007.

[31] V. Shoup. Practical Threshold Signatures. In *Advances in Cryptology - EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 207–220. Springer Verlag, 2000.

[32] D. Singelée and B. Preneel. Distance Bounding in Noisy Environments. In *Proceedings of the 4th European Workshop on Security and Privacy in Ad Hoc and Sensor Networks (ESAS '07)*, volume 4572 of *LNCS*, pages 101–115. Springer Verlag, 2007.

[33] N.O. Tippenhauer and S. Capkun. ID-Based Secure Distance Bounding and Localization. In *Proceedings of the 14th European Symposium on Research in Computer Security (ESORICS '09)*, volume 5789 of *LNCS*, pages 621–636. Springer Verlag, 2009.

[34] Visa Paywave. `http://www.visapaywave.co.uk/`.

[35] T. M. Wong, C. Wang, and J. M. Wing. Verifiable Secret Redistribution for Threshold Sharing Schemes. Technical Report CMU-CS-02-114, Carnegie Mellon University, 2002.