

Evaluating the Design of an Audit Data Pseudonymizer Using Basic Building Blocks for Anonymity*

Ulrich Flegel

University of Dortmund, D-44221 Dortmund, Germany

ulrich.flegel@udo.edu

Abstract:

Using an audit data pseudonymization system as an example, we show how the APES approach for basic anonymity building blocks can be used to informally evaluate the design of a given anonymity system. As a by-product we obtain indications of the usefulness and (in)completeness of the APES building blocks approach.

1 Introduction

Anonymity systems are often designed with a specific application in mind. Various systems, however, have similar functionality that can be reused for other applications. As a part of the APES project, De Win et al. [DWND⁺01] define reusable anonymity building blocks with minimal, yet useful functionality. This approach has several advantages: Firstly, similar building blocks can be compared more easily than the more complex systems they originate from. Secondly, given a list of building blocks with their properties, deficiencies in existing systems can be identified systematically. Thirdly, anonymity systems can be designed by systematically composing appropriate building blocks.

We present the APES anonymity building blocks approach in Sect. 2 in more detail. In Sect. 4 we use it to evaluate the design of a given system [BF00, Fle02] (see Sect. 3) by decomposing it into building blocks. In doing so we pursue the following goals in accordance to the above advantages: Firstly, the building blocks used in our design are compared to different building blocks with similar functionality with the two possible results: The system design is already composed of building blocks that are optimal for the given application, or we obtain specific indications how we can improve the design by replacing some building block with some other building block yielding stronger properties. Secondly, given the attacker and trust model of the system in Sect. 3 we may identify deficiencies in the design in an informal way by considering all building blocks in the supposedly exhaustive list given by De Win et al. [DWND⁺01]. Thirdly, as a by-product we obtain indications of the completeness of the list and classifications of APES building blocks with respect to the given design.

*This work is currently funded by the German Research Council (DFG) under grant number Bi 311/10-3.

2 Project APES: Anonymity and Privacy in Electronic Services

In the APES project (Anonymity and Privacy in Electronic Services) the state-of-the-art of anonymity systems has been surveyed and studied [SDDW⁺01]. For several of the applications there exists more than one anonymity system based on different anonymity techniques. The anonymity techniques themselves are often composed of several sub-components that are each responsible for a particular aspect of anonymity. In the APES project the anonymity systems have been decomposed into basic building blocks that can be reused for different systems, with a focus on *unconditional anonymity*, i.e. the anonymity cannot be revoked. The basic building blocks are identified, their properties and requirements are described, and their security and correctness are evaluated in an informal way [DWND⁺01]. An overview of the project is given by Diaz, Claessens and Preneel [DCP03] and in the project deliverables [SDDW⁺01, DWND⁺01, DNC⁺02].

APES basic building blocks are classified as being specific to the connection-level or the application-level: *Connection-level* basic building blocks are used to provide anonymous communication connections, whereas *application-level* basic building blocks are supposedly application-specific. To obtain a completely anonymous system, application-level anonymity often needs to be complemented by connection-level anonymity.

Basic building blocks at the connection-level hide or remove identifying information that is available at that level. Identifying information can occur *explicitly* like IP addresses in IP packet headers. Connections can also be traced along the communication path using *implicit* features of the appearance or of the flow of the communication. Network packets can be linked by *appearance* using e.g. content, format or size. Also the *flow* of network packets can be traced using the knowledge about the packet processing regarding e.g. order and timing. Accordingly, APES basic building blocks at the connection-level either change the appearance or the flow (see the second column of Tab. 1). To provide anonymous connections, explicitly as well as implicitly identifying information must be hidden. Therefore basic building blocks need to be composed to change the appearance as well as the flow of the messages. The following compositions of basic building blocks to so-called *local setups* are proposed:

serial: Building blocks are executed after each other, where the input of the latter block is the output of the former block.

parallel: Functionally unrelated building blocks can be executed in parallel, given that at most one building block changes the appearance of the message.

nested: The execution of the outer block is suspended for the execution of the inner block. This may be required for advanced message transformations or block dependencies [DWND⁺01].

Basic building blocks at the application-level hide or remove identifying information that is available at this level. They implement techniques that have been developed to add anonymity to a particular type of application (see the third column in Tab. 1).

3 Revisiting Audit Data Pseudonymization

Modern services and operating systems hosting them support the recording of audit data for various purposes, e.g. security and billing. Since audit data can normally be used without much effort to identify individual users of a system or of a service, recording such data may conflict with the users' expectancy for privacy and even with pertinent legislation concerning personal data of users. Audit data can be pseudonymized after its generation to avoid the conflict between the desire for security of services and the desire for privacy of users, in particular accountability and anonymity, respectively. We proposed concepts for the pseudonymization of audit data while balancing the conflicting requirements for anonymity and accountability [BF00]. Accordingly, we provided an implementation of pseudonymization, named *Pseudo/CoRe*, such that users appear under pseudonyms in Unix audit data, while maintaining the degree of linkability required for audit data analysis [Fle02]. During normal operation only the pseudonymized audit data is analyzed wrt. misuse suspicions. Only upon good cause shown, i.e. a sufficient misuse suspicion, the identifying data behind the pseudonyms can be revealed immediately, i.e., accountability can be established. When related to various other work, our approach exhibits several advantages, i.a. technical enforcement of purpose binding, the possibility of immediate reidentification independently of third parties, practicability due to independence of the user system and of expensive infrastructures such as PKI [Fle03].

Using two architectural models for anonymous authorization and surveillance, Fig. 1 illustrates the flow of explicitly identifying (solid arrows) and pseudonymized (dashed arrows) information as well as the control conditions (fat grey frames) when a user accesses a service where identifying data in audit records is replaced with pseudonyms after generation [Fle03]. Firstly, a user-side management software selects proper credentials that afford authorization for the desired access. Before access is granted, the credentials are verified wrt. trust, validity, authenticity and the service access policy. The credentials may contain explicitly identifying information about the user (e.g. account name and password or X.509 certificates). Also the communication system may reveal identifying information about the user. In our approach exclusively the identifying information being materialized in the audit data that is generated during the service access is hidden using pseudonymization (see E1 in Fig. 1). Identifying information that is available on the network, during credential verification or during service access, is assumed not to be available to attackers that might want to compromise user-identifying data [BF00, Fle02].

Since the responsibilities of the site security officer (SSO) include establishing accountability for certain user activities, his interest may conflict with the users' interest in anonymity. Therefore the SSO is explicitly precluded from controlling the user's management component as well as the service, including credential verification, audit data generation and pseudonymization (fat dark grey frames). Naturally, in the interest of the service's security the user is precluded from controlling the service, including credential verification, audit data generation and pseudonymization as well as the audit data analysis, alarm response and reidentification (fat light grey frames). In other words, the fat grey frames enclose those system components that enforce a specific interest of a given party, i.e., the dark grey frames stand for the user's interest in anonymity, whereas the light grey frames stand for the SSO's interest in accountability. Obviously, in this model the service,

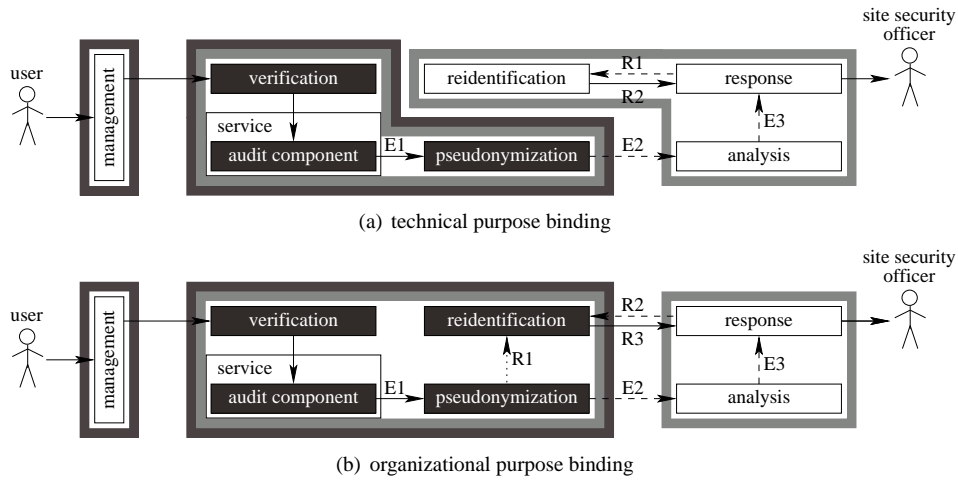


Figure 1: Trust and control in the architectural model of our approach

including credential verification, audit data generation and pseudonymization must not be controlled neither by the user nor by the SSO, but by a third party trusted by the user and the SSO for the enforcement and a fair balance of their conflicting interests. This third party may be the organization’s privacy protection official (PPO).

When the audit data has been pseudonymized, it is made available to the SSO (see E2 in Fig. 1), who analyzes the anonymous audit data for suspicions of misuse. If such an anonymous and sufficient misuse suspicion is found, a response (see E3 in Fig. 1) is generated, possibly implying to reveal the identifying data behind the pseudonyms (*conditional anonymity*) associated with the sufficient misuse suspicion (see R1 and R2 in Fig. 1a). It is a specific feature of our approach that pseudonyms can be disclosed immediately by the SSO without further assistance of any other party, if and only if the pseudonyms are associated with a sufficient misuse suspicion formerly defined by the PPO and recognized during pseudonymization. Since the reidentification is technically bound to (purposes requiring accountability of) sufficient misuse suspicions, we call this concept *technical purpose binding*. Technical purpose binding can always be achieved in a domain specific way only. We denote the domain independent counterpart of this concept as *organizational purpose binding*, where the mapping of pseudonyms to identifying information that is generated during pseudonymization is provided to several persons such that they can only together perform reidentification (see R1 in Fig. 1b). Each of these persons is responsible for protecting the interest of only one participating party when reidentification takes place (see R2 and R3 in Fig. 1b), i.e., the conflict is carried out in the *physical world* between these persons. As an example the PPO could look after the interests of the user whereas the SSO advocates his own interests. It is a disadvantage of this traditional approach that this process may take too much time for the SSO to be able to react appropriately to a misuse suspicion. On the other hand this concept is more flexible than the technical purpose binding, which is useful if a misuse suspicion has not been anticipated and defined

before it occurs. We therefore combine both concepts such that the SSO can perform re-identification timely if a predefined suspicion occurs (technical purpose binding), but if a suspicion occurs that has not been anticipated before, the SSO depends on the (other) trusted person(s), to perform reidentification (organizational purpose binding).

Possible attacks on the interests of both, the user and the SSO, result in specific requirements for the performance and the throughput of audit data pseudonymization. Firstly, the path taken by newly generated audit records should be as short as possible to be easily protectable in order to avoid the disclosure of identifying information of the user. Therefore audit records should ideally be pseudonymized on the same device where they are generated, i.e., the device that delivers the service. The response time of the service should not be unreasonably degraded by the pseudonymization. Secondly, if audit data is generated on the same device that an attacker is trying to gain control of, to avoid loss of accountability the audit data should be transported as quickly as possible to a remote location that securely stores audit data from possibly various audit components. Therefore, audit data should ideally be pseudonymized on the fly before transportation. Pseudonymization should not introduce a significant delay between the generation of an audit record and its transportation. Specifically, pseudonymization should be able to keep up with the audit data volume generated on the device. In summary, pseudonymization should not use building blocks that require a high computational complexity and/or introduce a significant delay [Fle03]. The performance of *Pseudo/CoRe* has been evaluated and acknowledged to live up to these requirements [Fle02].

4 Mapping our Approach to Building Blocks

In our approach we assume that the SSO can observe the behavior of service users merely by inspecting pseudonymized audit data. We also assume that the SSO cannot monitor the user's service accesses over the network, i.e., the SSO can only see where the pseudonymized audit data originates. Since this information needs not to be protected, there are no connection-level anonymity requirements in our approach. If we relax the latter assumption such that the SSO is able to monitor the user's service accesses over the network, the SSO can correlate monitored accesses with pseudonymous audit records. This situation raises the need for connection-level anonymity which can be implemented independently from our approach using existing solutions for anonymous connections (see Seys et al. [SDDW⁺01]). Fig. 2 shows how our approach can be decomposed into APES building blocks: Fig. 2a shows the conceptual system [BF00] and Fig. 2b shows the current *Pseudo/CoRe* implementation [Fle02]. In the following the description of the pseudonymization process focuses on the basic building blocks used.

The audit component(s) of the service are configured to generate only audit data required for sustained service provision, such as audit data indicating misuse (see the building block *filtering* as well as the input *audit data* in Fig. 2). The audit data is delivered to the *pseudonymizer* which inspects each incoming audit record for identifying features that, according to the PPOs definitions, shall be pseudonymized. Each of those features is replaced by an optionally padded random string that conforms to the requirements of the given audit data analysis tools wrt. format and content linkability of the feature (see the building blocks *padding 1* and *substitution* as well as the input *random string* in Fig. 2). These

strings are the pseudonyms embedded into the original (application-layer) audit data.

For each feature type the PPO defines a priori the possibilities to disclose the corresponding pseudonyms. If the possibility for disclosure is supported for a given pseudonym, then the corresponding identifying feature is encrypted, i.e., pseudonym disclosure is the process of recovering the correct decryption key (see the building block *encryption 1* as well as the input *random key* in Fig. 2). The pairing of the cryptograms of a given identifying feature together with the message digest of the corresponding decryption key, i.e. a verifier, forms a pseudonym mapping entry (see the building blocks *encryption 1* and *one-way function 1* as well as the output *pseudonym mapping* in Fig. 2a). Only a positively verified decryption key can be used to correctly decrypt cryptograms of the respective identifying feature [BF00].

To coarsen the number of actors in the system, dummy entries are added to the pseudonym mapping. To avoid the linkability of the pseudonym mapping entries over time, they are padded and reordered [BF00] (see the building blocks *dummy generation, padding 2* and *reordering* in Fig. 2a). To avoid the inference of the number of real pseudonym mapping entries based on the number of pseudonym mapping updates, dummy updates are used, even if no new entries have been inserted into the pseudonym mapping [BF00] (see the building block *dummy updates* in Fig. 2a). Note that the description of the pseudonym mapping and the respective dummy activity applies to the conceptual system depicted in Fig. 2a only. The implemented system *Pseudo/CoRe* illustrated in Fig. 2b differs substantially with respect to the pseudonym mapping and dummy activity.

The pseudonymity-layer data contains (among others) information enabling pseudonym disclosure. It is embedded in special audit records in the format of application-layer audit data (see the output *pseudonymity-layer data (including the pseudonym mapping)* in Fig. 2). To be able to use the pseudonym mapping for pseudonym disclosure, a valid decryption key is required. The recovery of such a key from the pseudonymity-layer data is subject to technical and/or organizational purpose binding. In the conceptual system the organizational purpose binding is enforced using a threshold cryptosystem to encrypt the decryption key such that only eligible sets of persons can recover it in cooperation (see the building block *threshold cryptosystem* in Fig. 2a). In *Pseudo/CoRe* the organizational purpose binding is implemented in a simpler way using a symmetric cryptosystem to enforce that decryption keys can be recovered with the cooperation of the PPO only (see the building block *encryption 2* in Fig. 2b).

The technical purpose binding is enforced by using threshold schemes for secret sharing to securely split the decryption key in shares that are provided in the pseudonymity-layer (see the building block *secret sharing schemes* in Fig. 2). Misuse scenarios are mainly modeled using appropriate thresholds for Shamir secret sharing schemes and associating appropriate misuse contexts and weights with the pseudonyms embedded in observations of potentially misuse-related activity [BF00]. A valid decryption key can be recovered from the shares belonging to pseudonyms associated with a given misuse context if and only if the misuse suspicion is sufficient, i.e., the suspicion level of the misuse context exceeds the threshold of a secret sharing scheme. Thus, pseudonym disclosure is technically bound to (purposes requiring accountability of) sufficient misuse suspicions. Since the above described shares per se are unlinkable, a tentative key recovered from an arbitrary

combination of these shares could just be invalid. In that case the tentative key does not match any verifier in the pseudonym mapping (see above and the building block *one-way function 1* in Fig. 2a). Even if the tentative key is valid, it could still represent a mismatch, i.e., the shares belong to pseudonyms of different identifying features [BF00]. To detect mismatches, each share is provided with a verifier (see the building block *one-way function 2* in Fig. 2a).

The conceptual system supports unlinkable pseudonymity-layer data. Since the computational complexity of selecting unlinkable shares for the recovery of a valid key discourages practical use, for performance reasons *Pseudo/CoRe* uses linkability labels in the pseudonymity-layer [Fle02]. This basic difference has substantial consequences for the resulting system design. Firstly, since the pseudonymity-layer data is linkable in *Pseudo/CoRe*, it would be futile trying to hide the number of actors using the pseudonym mapping (updates) (note that the corresponding building blocks are missing in Fig. 2b: *dummy generation, padding 2, reordering* and *dummy updates*). Secondly, in *Pseudo/CoRe* the linkability labels are used to avoid mismatches during the recovery of decryption keys. Therefore it is unnecessary to provide a verifier with each share to detect mismatches (note that the building block *one-way function 2* is missing in Fig. 2b). Thirdly, in *Pseudo/CoRe* the pseudonym mapping is embedded in the pseudonymity-layer, using the cryptograms of identifying features as linkability labels (see the output *pseudonymity-layer data including the pseudonym mapping* in Fig. 2b). As a result, the cryptograms are directly associated with the shares of the corresponding decryption key. Hence, accessing the cryptograms requires no search in the pseudonym mapping, such that there is no need for the verifiers in the pseudonym mapping (note that the corresponding *one-way function 1* building block is missing in Fig. 2b).

The pseudonymization component is complemented by a reidentification component (see Fig. 1). Our proposed solution and implementation for the reidentification can be decomposed into building blocks with analogous results, but does not give any additional and surprising insights. We therefore do not provide the details here.

5 Considering Building Blocks for Improved Design

The fourth column of Tab. 1 summarizes the building blocks used in our approach, as identified in Sect. 4. Interestingly, though our approach does not aim at providing connection-level anonymity, it uses many connection-level building blocks that have not been considered for application-level anonymity in APES (see also the third column of Tab. 1).

APES primarily aimed at providing building blocks for unconditional anonymity. Exceptions are the following building blocks that can be used to provide conditional anonymity: fair blind signature, group signature, threshold cryptosystem, secret sharing schemes, pseudonyms, trusted third party. Our approach aims at conditional anonymity and any given building block on this list is either used by our approach or it could be investigated for further improvement. Note that our approach does not use any building blocks for supporting conditional anonymity that were not defined for APES. But, our approach uses a more elementary building block named *one-way function* enabling to compare features for equality, while the features are hidden. A collision-resistant one-way function, as in our implementation a cryptographic hash function, is used to hide the features such that for two

Table 1: Basic anonymity building blocks used in our approach, ‘?’ = classification missing, ‘—’ = building block missing

building block	connection-level		application-level	our approach
	appearance	flow		
encryption	✓		✓	✓
padding	✓		?	✓
substitution	✓		?	✓
compression	✓			
reordering		✓	?	✓
latency		✓		
dummy activity		✓	?	✓
no replay		✓		
filtering		✓	?	✓
caching		✓		
broadcast		✓	✓	
multiplexing		✓		
bulletin board		✓	✓	
one-way function	—	—	—	✓
(fair) blind signature			✓	
group signature			✓	
threshold cryptosystem			✓	✓
multi-party computation			✓	
homomorphic encryption			✓	
deniable encryption			✓	
secret sharing schemes			✓	✓
zero-knowledge			✓	
pseudonyms			✓	✓
trusted third party			✓	✓

given features f_i and f_j if $h(f_i) = h(f_j)$ then with high probability $f_i = f_j$ [MvOV97].

For each building block used in our approach we give the anonymity goal to which it contributes, the aspect (appearance or flow) considered by the building block (for connection-level building blocks only), as well as the effect of the building block that contributes to the anonymity goal. The basic building blocks are considered in the order of appearance in Tab. 1; for details on each building block refer to De Win et al. [DWND⁺01]:

encryption: hide identifying features (appearance) (see *encryption 1* in Fig. 2): avoid linkability using the content of identifying features; organizational purpose binding of pseudonym disclosure (appearance) (see *encryption 2* in Fig. 2b): allow decryption of the decryption key only in cooperation with the PPO

padding: hide identifying features (appearance) (see *padding 1* in Fig. 2): avoid linkability using the size of identifying features; hide the number of actors (appearance) (see *padding 2* in Fig. 2a): avoid linkability using the size of the pseudonym mapping entries

substitution: hide identifying features (appearance) (see *substitution* in Fig. 2): replace identifying features by persistent, i.e. linkable, or one-time, i.e. unlinkable, pseudonyms¹

reordering: hide the number of actors (flow) (see *reordering* in Fig. 2a): avoid linkability using the order of mapping entries

¹Note that blanking of identifying features is in general not useful because it can break legacy analysis tools that rely on the existence of specific features.

dummy activity: hide the number of actors (flow) (see *dummy generation* and *dummy updates* in Fig. 2a): avoid linkability using the number of mapping entries and mapping updates

filtering: hide identifying features (flow) (see *filtering* Fig. 2): discard audit records that are not needed for analysis

one-way function: hide identifying features (appearance) (see *one-way function 1* and *one-way function 2* in Fig. 2a): enable the comparison of decryption keys with tentative recovered keys while hiding the content of the decryption keys

threshold cryptosystem: organizational purpose binding of pseudonym disclosure (see *threshold cryptosystem* in Fig. 2a): allow eligible sets of individuals to decrypt the decryption keys only cooperatively

secret sharing schemes: technical purpose binding of pseudonym disclosure (see *secret sharing schemes* in Fig. 2): allow anyone to recover only decryption keys that are involved in the transgression of the scheme's threshold

pseudonyms: hide identifying features (see *random string* in Fig. 2): the pseudonyms replacing the identifying features have the following properties [DWND⁺01]: unauthenticated, sharing is possible, linkable within the context of an attack, forgeable

trusted third party: (un)conditional anonymity (see double fat grey frames in Fig. 1, not shown in Fig. 2): the PPO can be trusted to properly configure the system components in the PPO domain

Several building blocks are identified that could not be applied to the current system together with the respective reasoning:

compression: items to be hidden either are too small or have too high an entropy to be compressed

no replay: since the channels between the audit components and the pseudonymizer are trusted, there is no need for the detection of audit record replay

caching: since the channels between the audit components and the pseudonymizer are uni-directional there is no responder that would need to be hidden

broadcast: no need to hide the identity of the recipient of the audit records (analyzer)

multiplexing: no need to hide the communication path of the audit records

bulletin board: see the broadcast building block

untraceable broadcast: there is no need to hide the identity of the senders of the audit records (audit-components)

blind signature: does not allow for conditional anonymity with technical purpose binding

deniable encryption: for the current system there seems to be no application

In the following a number of building blocks is identified that could be explored for possible improvements of the current system. Note, that except for the *latency* building block these building blocks are computationally expensive and may not meet the performance requirements of audit data pseudonymization (see Sect. 3).

latency: could be used instead of dummy pseudonym mapping updates if many new pseudonym mapping entries are created

fair blind signature: could be explored to provide technical purpose binding while reducing the power of the trusted third party

group signature: could be explored to be used for conditional anonymity with organiza-

tional purpose binding; though, its computational complexity is much higher than that of threshold cryptosystems [DWND⁺01]

multi-party computation, homomorphic encryption: for certain applications not only the application-layer audit data may be analyzed, but also certain properties of the pseudonymity-layer data may be exploited during analysis, independently of pseudonym disclosure; cryptographic primitives for multi-party computation and for homomorphic encryption could be explored for the generation of pseudonymity-layer data that has the desired properties

zero-knowledge: could be explored to provide a proof of validity for pseudonyms to reduce the power of the trusted third party

pseudonyms: other pseudonym systems could be explored to reduce the power of the trusted third party while gaining the following additional pseudonym properties: authenticated, sharing is impossible, unforgeable

6 Conclusion

We revisited the design of a pseudonymizer using the APES approach of basic anonymity building blocks with two goals: informally identifying room for improvement and informally identifying deficiencies in the design. Regarding the first goal, we identified the *latency* connection-level building block that could be used as an alternative to *dummy activity* under specific circumstances. We identified six further application-level building blocks that could be explored to further reduce the power of the trusted third party in our approach in order to obtain more useful properties of the pseudonyms or to replace the *threshold cryptosystem* or to support the exploitation of the pseudonymity-layer for the application-specific analysis. Though, probably none of the candidate building blocks will satisfy the requirements regarding computational complexity or delay. As a result, given the APES building blocks list, the current design may only be improved if some of the current requirements are relaxed, trading off stronger mechanisms against time or computational complexity. With respect to the second goal, and given the APES building blocks list, we could not identify any deficiencies in our design.

Our experience in the exercise of decomposing a given system into basic building blocks provides some indications about the APES approach: It is possible to analyze a given system design using APES basic building blocks. The analysis may stimulate improvement of the design and may point out weaknesses in the design. The statements regarding these two goals are the stronger, the more complete the list of building blocks considered is. Though De Win et al. claim to present an exhaustive list of building blocks for unconditional anonymity [DWND⁺01], it seems rather unlikely that future developments will not contribute new building blocks for unconditional anonymity. In fact, in our approach and for various purposes we found the necessity to compare features that need to be hidden. This can be achieved using one-way functions. Surely, one-way functions are already part of several APES building blocks. Anyway, they are also very useful as such for various applications without the functionality of the building blocks in which they are contained. We thus postulate the application-level building block named *one-way function*. This is an indication, that the list of APES building blocks is probably not exhaustive. As a result, we cannot even informally exclude that there exist better designs for our pseudonymizer

or that the design may have deficiencies. The above statements about the quality of our design merely express strong indications based on the current state of knowledge.

In addition to the incompleteness of the list of building blocks we found that the classification of building blocks regarding the level of their use is also incomplete, i.e., five connection-level building blocks were used at the application-level, in spite of the APES classification. Further investigation might show that (nearly) all connection-level building blocks can also be useful at the application-level. As a last note, we found that the given building blocks for conditional anonymity were sufficient to build our solution, though APES so far was focused on building blocks for unconditional anonymity, i.e., the given building blocks may be perfectly sufficient to build systems for conditional anonymity.

References

- [BF00] Joachim Biskup and Ulrich Flegel. Threshold-based Identity Recovery for Privacy Enhanced Applications. In Sushil Jajodia and Pierangela Samarati, editors, *Proceedings of the 7th ACM Conference on Computer and Communications Security*, pages 71–79, Athens, Greece, November 2000. ACM SIGSAC, ACM Press.
- [DCP03] Claudia Díaz, Joris Claessens, and Bart Preneel. APES — Anonymity and Privacy in Electronic Services. *Datenschutz und Datensicherheit (DuD)*, 27(3):143–145, March 2003.
- [DNC⁺02] C. Díaz, V. Naessens, J. Claessens, B. De Win, S. Seys, B. De Decker, and B. Preneel. Anonymity and Privacy in Electronic Services (APES) Deliverable 5 – Tools for Technologies and Applications. Technical report, K. U. Leuven, November 2002.
- [DWND⁺01] B. De Win, V. Naessens, C. Díaz, S. Seys, C. Goemans, J. Claessens, B. De Decker, J. Dumortier, and B. Preneel. Anonymity and Privacy in Electronic Services (APES) Deliverable 3 – Technologies Overview. Technical report, K. U. Leuven, November 2001.
- [Fle02] Ulrich Flegel. Pseudonymizing Unix Log Files. In George Davida, Yair Frankel, and Owen Rees, editors, *Proceedings of the Infrastructure Security Conference (InfraSec2002)*, number 2437 in Lecture Notes in Computer Science, pages 162–179, Bristol, United Kingdom, October 2002. Springer.
- [Fle03] Ulrich Flegel. Ein Architektur-Modell für anonyme Autorisierungen und Überwachungsdaten (in German). In R. Grimm, H. B. Keller, and K. Rannenber, editors, *Mit Sicherheit Informatik, Proceedings of the GI Conference on Sicherheit – Schutz und Zuverlässigkeit (in German)*, number P-36 in Lecture Notes in Informatics, pages 293–304, Frankfurt, Germany, September 2003. Gesellschaft für Informatik e.V.(GI), Köllen Verlag.
- [MvOV97] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*, chapter 9 Hash Functions and Data Integrity. Discrete Mathematics and its Applications. CRC Press, Inc., Boca Raton, Florida, 1997.
- [SDDW⁺01] S. Seys, C. Díaz, Bart De Win, V. Naessens, C. Goemans, J. Claessens, W. Moreau, B. De Decker, J. Dumortier, and B. Preneel. Anonymity and Privacy in Electronic Services (APES) Deliverable 2 – Requirement Study of Different Applications. Technical report, K. U. Leuven, May 2001.