

On Pseudonymization of Audit Data for Intrusion Detection*

Joachim Biskup

University of Dortmund, D-44221 Dortmund, Germany

biskup@ls6.cs.uni-dortmund.de

Ulrich Flegel

University of Dortmund, D-44221 Dortmund, Germany

flegel@ls6.cs.uni-dortmund.de

Abstract

In multilaterally secure intrusion detection systems (IDS) anonymity and accountability are potentially conflicting requirements. Since IDS rely on audit data to detect violations of security policy, we can balance above requirements by pseudonymization of audit data, as a form of reversible anonymization. We discuss previous work in this area and underlying trust models. Instead of relying on mechanisms external to the system, or under the control of potential adversaries, in our proposal we technically bind reidentification to a threshold, representing the legal purpose of accountability in the presence of policy violations. Also, we contrast our notion of threshold-based identity recovery with previous approaches and point out open problems.

1 Introduction

Stimulated by the increasing reliance on information and communication technology, particularly the broadening offer of services on the Internet, potentially conflicting demands emerge, such as accountability and anonymity. To establish accountability, suboptimal preventive security mechanisms and intrusion detection facilities are meant to complement one another. Since intrusion detection is based on audit data, which in part can be associated with real persons, it deals with personal data. Conventional approaches to intrusion detection prioritize the interests of IT-system owners, emphasizing accountability, event reconstruction, deterrence, as well as problem and damage assessment.

Unobservability, anonymity and unlinkability as instruments derived from the concept of informational self-determination, confine the collection and processing of personal data. Basic preconditions of informational self-determination are either the data subject's informed consent on or the legal or contractual necessity of processing the personal data, accompanied by strict purpose binding for collecting, processing and communicating personal data.

*The work described here is currently partially funded by Deutsche Forschungsgemeinschaft under contract number Bi 311/10-1.

Since data in processible form is perfectly reproducible, and since formally verifying, that a conventional IT system as a whole preserves its users' rights wrt. information self-determination, is impractical, it is insufficient to legally interdict misuse of personal data [1]. Consequently there is an increasing demand for the technical enforcement of privacy principles in multilaterally secure systems, taking into account all, and acceptably balancing contrary security requirements of all involved parties [2]. A multilaterally secure Intrusion Detection System (IDS) that respects the users' desire for privacy, may allow them to act pseudonymously, hiding the association between the data and the real persons, as far as the detection goals can still be achieved. To hold people responsible for violations of the security policy, a multilaterally secure IDS also (re)identifies users upon good cause shown.

In this paper we survey and contrast approaches to employ pseudonyms in intrusion detection, and we propose an approach to pseudonyms with the ability for threshold-based identity recovery, that is, the circumstances accompanying the necessity of reidentification can be described as the transgression of a threshold. By technically binding reidentification to a threshold, we can enforce a strong binding to the purpose of establishing accountability for violations of policy.

The rest of this paper is organized as follows. First we selectively review pertinent European legislation that inspires technical purpose binding (section 2), identify important aspects of pseudonymization (section 3) and survey related approaches (section 4). We then present our approach, specifying the underlying trust and threat model (Sect. 5), exemplifying the targeted audit environment (section 6), and proposing to apply secret sharing to pseudonymization (Sect. 7). A more technically inclined elaboration can be found in [3]. We conclude pointing out open issues for investigation (Sect. 8).

2 European Legislation

According to consideration (2) in the EC directive [4], IT-systems must respect the right to privacy. Since it is one objective of the European Community (EC) to ensure "economic and social progress by common action to eliminate the barriers which divide Europe" (ibid., consideration (1)), within the EC a harmonization of national laws has been initiated to achieve similar levels wrt. protection of personal data (ibid. considerations (3),(5),(7)-(9)). Based on articles 25, 26 and 4, for third countries the directive, as well as resulting national laws, are authoritative regarding to export and processing of personal data from and on EC territory, respectively. The directive [4] amalgamates different traditions in dealing with personal data. We found exemplary statements about fundamental protection principles in German law, specifically a census sentence [5] of the German Constitutional Court, which postulates the informational self-determination (ibid. C II 1a). This sentence clarifies that there exists no insignificant data, since its sensitivity depends not merely on its nature, but rather on its linkability and processibility, as well as the pursued objective (ibid. C II2). To fix data sensitivity, the objectives are fixed beforehand, and consequently the nature and scope of data being collected, as well as the nature of processing results have to be fixed. This fundamental privacy principle is henceforth referred to as *purpose binding*. Further principles require that personal data being collected is *adequate*, *relevant* and *not excessive* with respect to a legal purpose of processing, which is proportional wrt. to the data

sensitivity (ibid. C II 1b).

3 Assessing Pseudonymization

Consider multilaterally secure intrusion detection balancing accountability requirements of system owners and anonymity demands of users. In this scenario, both parties are represented by system security officers (SSO) and data protection officials (PPO), respectively. It is part of an individual's right of informational self-determination to separate different spheres of life on demand, for example by acting more or less anonymously. An entity is called anonymous if an attacker cannot link its identity with its role or activities within an anonymity group. Though there are several methods to achieve anonymity, we focus on pseudonymization here. By means of pseudonymization we unlink events from identities.

We consider some criteria determining the strength of anonymity. The strength should be proportionate wrt. the risk posed by data collection and processing, as well as the effort required by an attacker. The *threat model*, defines which potentially colluding parties may collect and correlate information. This model influences *when* and *where* data needs to be pseudonymized, since this should happen, before it enters domains monitored by adversaries.

Another role plays an instance with the ability for *reidentification*. The participant(s) required for identity recovery depend on the *method* of pseudonymization and on the pseudonym *introducer*. Generally, for the purpose of accountability, identity recovery should be independent from the pseudonymous entities. On the other hand, the IDS and SSOs are considered adversaries to the entities' privacy, requiring the IDS for identity recovery either to depend on the cooperation of a PPO, or to restrict analysis results exclusively to detected security policy violations. Consequently domains controlled by an IDS and by SSOs should by organizational and technical means be strictly separated from domains where personal data is processed in the clear.

The strength of anonymity depends on the cardinality of the *anonymity group*. The losses of an attacker should be higher than the gains when damage by reidentification is done to all members of the anonymity group instead of just one entity. For scarce anonymity group populations it seems advisable to choose a new pseudonym per transaction.

To what degree inferences can be extracted from audit data depends on the *linkability* of features being pseudonymized. Feature selection for pseudonymization and the class of pseudonyms being employed substantially influence linkability. For a discussion about user identifying features and feature selection refer to [6]. We need to trade the strength of anonymity off against the quality of analysis, which depends on the linkability of certain features.

Assuming an entity has only one identity, it may wish to act under one or more pseudonyms. We consider various classes of pseudonyms achieving increasing degrees of anonymity [1]:

Entity-based pseudonyms are substitutes for the identity of an entity being used for many transactions in differing contexts and relations. Actions of the same entity are linkable via equality of pseudonyms. Frequent use of the pseudonym implies the risk of information cumulation and eventually reidentification. Entity-based pseudonyms can be extended to represent groups of entities, severely limiting the capability to account events to individual entities.

public: Identity assignment is publicly available such as phone numbers, reidentification is feasible for everyone.

non-public: Identity assignment is available to a few selected sites only, such as unpublished phone numbers, which may reidentify the entity.

anonymous: Only the entity itself is acquainted with the assignment of its identity to the pseudonym.

Role-based pseudonyms are assigned to the roles an entity assumes in the context of different relations. Pseudonyms for roles mitigate the risk of reidentification by means of information cumulation.

relation-based: A pseudonym is used for many transactions in the context of a given relation, such as a club member number. Busy relations allow information cumulation and eventually reidentification.

transaction-based: A pseudonym is used in the context of a relation for one transaction only, similar to one-time passwords.

An IDS shouldn't collect and permanently store data not strictly necessary for the purpose of analysis (data avoidance). As soon as data becomes unnecessary for that purpose, it should be discarded (data reduction). An example is found in the IDS ANIDA (see section 4.4), where the anomaly detection component reduces the audit stream by discarding non-anomalous data, before it is handed to the misuse detection component. This approach is based on the following two assumptions: Known attack behavior as defined in misuse detection approaches is always detected as being an anomalous deviation from normal behavior as defined by anomaly detection approaches. Not all detectable anomalies posing a security threat are covered by the attack behavior encoded in misuse detection systems.

Finally, since false positives are affecting a users' privacy due to undesirable reidentifications, detection approaches with low false positive rate are required [7], [8].

4 Related Approaches

First of all we roughly describe the relevant details in work related to pseudonymous intrusion detection. Subsequently we review some more general approaches to anonymous service access.

4.1 IDA

The *Intrusion Detection and Avoidance* system (IDA) and its fundamental privacy concepts are described in [9], [10] and [6]. IDA audits certain system calls concerned with user sessions and clearance levels as well as object creation, deletion, access and respective permission modifications. Apart from protecting personal data by conventional mechanisms, IDA developed the notion of pseudonymous audit analysis. The kernel reference monitor introduces anonymous entity-based pseudonyms before audit data is passed to the audit trail and the kernel integrated analysis component, including anomaly detection as well as misuse detection analysis. The IDA

concept incorporates the introduction of pseudonyms wherever subject identifiers are used, particularly for access control. To prevent information cumulation with respect to a pseudonym, relation-based pseudonyms based on combinations of subject and object identifiers are proposed. These role-based pseudonyms are proposed for use in access control only, but not for audit analysis which in IDA requires linkability across subjects. Analysis proceeds on pseudonymized audit data. The close coupling of the analysis and decision components to the reference monitor allows enforcement of decisions without the need for automated reidentification. Obviously, in the IDA approach pseudonymization is deeply entangled with the innards of the operating system, limiting ease of application to today's off-the-shelf platforms.

The IDA concept proposes symmetric encryption of subject identifiers in audit data. To limit the risk of information cumulation and hence reidentification with respect to the entity-based pseudonyms, pseudonym changes by rekeying after certain time intervals are proposed. Since IDA can react without requiring automatic reidentification, enforcement of purpose binding for identity recovery is handled external to the system by sharing the symmetric key between SSO and PPO. Prerequisite to this proceeding is the symmetric key used in monitored systems being safe against attackers, meaning SSOs must not be able to control the responsible subsystems.

To prevent reidentification by correlation of information other than subject identifiers in audit trails or in profiles, it is proposed to pseudonymize all features that uniquely identify subjects. In addition it is proposed to pseudonymize features an attacker may have external knowledge about, such as actions and their time stamps, parameters and required privileges.

4.2 AID

Privacy related aspects concerning the *Adaptive Intrusion Detection* system (AID) are described in [11], [12], [6], [13] and [14]. In case of conflicting descriptions in those publications we base our summary on the latest information available. The AID implementation comprises a central RTworks based misuse detection¹ analysis station and distributed event monitoring agents. The Solaris based analysis station controls and adaptively polls the agents for Solaris BSM and Windows NT audit data respectively. Available publications do not describe the Windows NT matters.

AID uses Solaris BSM to audit system calls and selected applications concerned with user sessions, file creation, deletion, accesses and respective permission modifications as well as processes and administrative activities, some being non-attributable to any user. The fundamental concepts of pseudonymous audit analysis of IDA and AID are fairly similar, but the realizations differ in detail. AID introduces pseudonyms on monitored hosts after audit records have been emitted by the Solaris user level `auditd`. Owing to the implementation of AID being based on an off-the-shelf platform, there is no holistic concept as in IDA, pseudonymizing occurrences of identifying features outside the BSM audit stream. Monitoring agents pseudonymize audit records, add supplementary information needed for misuse analysis and convert the records into a reduced format. Analysis proceeds on pseudonymous records in the central station.

¹Further work on a neural network based anomaly detection analysis component has been announced in several publications, but results to date have not been published.

Reidentifications occur automatically if the expert system generates an alarm, or when alarm reports are generated, and before audit trails are archived in encrypted form. All AID monitoring agents reporting to the same central analysis station use the same key to symmetrically encrypt identifying features in order to pseudonymize them. It follows, that if an adversary monitors the key on any AID host, the adversary is able to reidentify any audit records he monitors. Since AID uses a confidential channel (SecureRPC) for audit data transport, an adversary is limited to AID host access, the central station being the most desirable target. Consequently SSOs must not be able to monitor either the pseudonymization keys, or the audit data, as well as the activities of users on any AID host. In the AID Solaris/Unix environment an SSO must therefore neither be able to control the AID machines, nor to monitor user activities, which are severe limitations, considering that particularly SSOs would want to control and monitor the security of the tools and machines they are responsible for. Reviewing archived audit records requires cooperation of an SSO and a PPO because the archive encryption key is shared between SSOs and PPOs. AID logs automatic reidentifications to an audit, but there are no provisions forcing SSOs to make this audit available PPOs.

AID symmetrically encrypts identifying features such as real and effective user and group IDs of subjects and object owners, audit, session, terminal and process IDs as well as host names and home directories in paths if they denote the user name. Additionally the AID concept proposes pseudonymization of host platform types, names of originating hosts and home directory names in command execution parameters and the respective environment. To limit the risk of reidentification by information cumulation with respect to the non-public entity-based pseudonyms, as in IDA, pseudonym changes by rekeying after irregular time periods are proposed.

4.3 Chalmers firewall audit pseudonymizer

Both [7] and [8] describe a yet to be named anomaly detection tool that uses statistical methods to analyze pseudonymous audit data. The tool analyzes login event audit records produced by a proxy-based firewall. Pseudonyms seem to be introduced in large batches², of audit records before exporting the batches and pseudonym mappings for anomaly analysis.

Some identifying features are pseudonymized: user names, server host names, client host names as well as free text fields containing user and host names. Other potentially identifying features not being used as profile discriminators, but statistically analyzed, such as activities and time stamps, were not pseudonymized. Audit records are pseudonymized by replacing them bijectively with sequentially numbered place holders denoting the respective feature type (eg. `user0`, `user1`, ...). Host names and each domain name part are mapped separately. This kind of entity-based non-public pseudonyms is memory inefficient, and after reidentifications the respective mappings are easily remembered by SSOs. IP addresses are mapped to an address in the same address class, allowing to determine an IP address's class from its pseudonym. The mapping of pseudonyms to the substituted data is stored in a separate database. Analysis proceeds on pseudonymized audit records. In [8] it is proposed to use group-based pseudonyms and thereby losing the ability to account events to individual users.

²The pseudonymizer though technically is prepared to pseudonymize audit data on the fly.

There are no documented precautions taken protecting the confidentiality of the pseudonym mapping neither while in transit nor during analysis. Reidentification is not supported in an automatic way and seems to be established manually by the SSO. The question remains, what technically keeps an SSO from reidentifying all audit records independently from analysis results. Certainly the concept could be extended to avoid such problems.

Both [7] and [8] propose and discuss several improvements. Based on [6], information cumulation regarding pseudonyms could be limited by changing the mapping of pseudonyms. This would also limit the problems generated by easily remembered pseudonyms. To restrict leakage of personal or business related information it is proposed to insert dummy or chaff audit records blurring the genuine patterns and statistics of resource usage. Similar to ANIDA's GRPs (see Sect. 4.4) it is proposed to associate users with groups to support the SSO's intuitive manual anomaly detection capabilities. These associations are not to be mixed up with aforementioned lossy group-based pseudonyms.

4.4 ANIDA

Privacy aspects concerning the Windows NT based *Aachener Network Intrusion Detection Architecture* (ANIDA) are described in [15]. In contrast to the other reviewed systems ANIDA's architecture is focused towards analysis of network-based audit data from client-server applications and underlying protocol stacks. The concept comprises components monitoring all traffic on the network and traffic exchanged with selected networked services. ANIDA fundamentally requires cooperation with a generic distributed authentication and access control mechanism. As a result, the monitored services need to support this mechanism. For the purpose of concealing identifying information related to a connection, Kerberos was chosen and modified concerning the tickets, initial authentication and some protocol steps. A stronger approach is concealing even the fact of a connection taking place, realized by the integration of MIXes [16], [17] with Kerberos.

ANIDA uses a kind of transaction-based pseudonyms introduced by the authentication service after user authentication or by one (ore more) trusted MIX(es). The pseudonyms are supplemented with group identifiers confirming group memberships for the purpose of access control. The anonymity group of a user therefore comprises all members of those groups she itself is member of [15]. Kerberos tickets were modified to use these augmented pseudonyms, referred to in ANIDA parlance as group reference pseudonyms³ (GRP), instead of a client principal name. Furthermore the client's host network address has been omitted from tickets. GRPs are valid and can be used for several transactions until the bearing ticket expires. Thus, GRPs can be classified somewhere in between relation-based and transaction-based pseudonyms. Choice of appropriate timeouts allows balancing strength of anonymity against the overhead involved in making out tickets.

Reidentification requires cooperation of the GRP introducer (the Kerberos authentication server or the MIX), which must be independent of SSOs and could be administered by PPOs.

³Again, this kind of transaction-based pseudonyms including group associations is not to be mixed up with lossy group-based pseudonyms.

As there are no monitoring components internal to client hosts and the GRP introducer, SSOs should not be able to monitor activities on client hosts and the GRP introducer.

The ANIDA approach implies group-based and thus coarse-grained access control and entity profiling for anomaly detection. While group-based access control is widely in use for authorization of network service accesses, it would be too limited for system internal authorizations such as file accesses. Group-based profiling for anomaly detection offers several advantages, such as requiring cooperation of the group majority to achieve unnoticed profile drift defining malicious behavior as acceptable. This concept emerged with NIDES [18].

In order to support the principle of data reduction, the anomaly analysis component supplies the misuse analysis component only with audit data known to be free of anomalies. In ANIDA's case the misuse analysis will not miss any anomalous data because the transaction-based anomaly detection technique used in ANIDA claims to isolate all anomalies with respect to defined valid transactions. In this scenario misuse detection based analysis should not be solely dependent on the linkability of the group identifiers. Otherwise it could become confused and mistake different entities for a single one, possibly leading to false alarms.

4.4.1 More General Approaches.

Some related work is described in [19], [20] and [21]. They propose modifications to authentication, based on the observation, that traditional access control requires the collection of identifying data, which is required for the resolution of extraordinary circumstances, before access is granted. Instead, authentication components can collect information strictly necessary for the purpose of access control and require a guarantee, that identifying data can be recovered with the help of an independent third party.

The approaches in [19] and [20] generate transaction-based pseudonyms during user authentication, comprising a zero-knowledge proof, that identity recovery is possible later on. [21] presents an approach based on [22], basically interpreting electronic cash as certificates of pseudonyms, leading to entity-based pseudonyms, but not transaction-based pseudonyms as in [19] and [20]. Entity-based pseudonyms provide linkability as required for some intrusion detection methods, respectively, but linkable pseudonyms allow cumulation of information related to the user of the pseudonym, which can eventually enable reidentification. After an initial session login entity-based pseudonyms could be used in traditional access control scenarios. Since in aforementioned approaches pseudonyms are introduced before resources are accessed, the identity of actors is protected against the entire accessed system. In the case of transaction-based pseudonyms this comes at the cost of major efforts required to modify the concerned access control systems traditionally used in off-the-shelf operating systems. On the other hand we were mainly interested in pseudonymous intrusion detection. If we can afford to place trust in a component to correctly introduce pseudonyms on behalf of the actors, and if we can technically separate observability of identifying accesses from pseudonymous event processing, we can easily integrate pseudonymous audit with major operating systems.

Different from related work, our approach allows for transaction-based pseudonyms that guarantee automatic reidentification independently of a third party, if certain conditions bound to a specified purpose are met.

5 Trust and Threat Model of our Approach

For an IDS observing user activities within a host, seamless integration with an off-the-shelf operating system is important, for it relies on event data, i.e. audit records, collected within the host. For Unix operating systems this always implies potential integrity loss of event data in case an attacker achieves `root` privileges. If we can afford to rely on event data under these conditions, we can also afford to rely on pseudonyms introduced within the same machine, after user authentication was performed and users are identified in the clear. Stronger approaches would require pseudonym introduction prior to or during user authentication at the console, which comes at the expense of major modifications of the operating system (see Sect. 4.4.1).

Starting from here, we have to take into account conflicting security requirements of basically two parties. Firstly, users are interested in anonymous use wrt. to IT system owners. Secondly, the IT system owners, specifically their representatives responsible for the system security (SSO), when faced with policy violations, are interested in establishing accountability. Neither party can be allowed to control the event generator as well as the pseudonymizer. Otherwise the respective mutual security requirement were at risk. We thus introduce a third party, the personal data protection official (PPO). The PPOs are trusted by the users to protect their anonymity interests, and they are trusted by the SSOs to protect their accountability requirements. In accordance with their position, PPOs control both, event generators and pseudonymizers (see Fig. 1). Users may verify both components for proper functionality, SSOs may do so during official user inactivity times. As outlined above, corruption of the event generator threatens accountability. Corruption of the pseudonymizers additionally threatens anonymity.

The event generators, either by themselves or by appropriate extensions, forward sensitive audit data to remote analyzers only if that data is required to detect harmful behavior of the involved entities. Pseudonymizers are implanted just behind the standard audit services, or their extensions, respectively. The pseudonymizers inspect audit data, and they substitute identifiers by carefully tailored pseudonyms. Exploiting Shamir's cryptographic approach to secret sharing, these pseudonyms are actually shares, which can be used for reidentification later on, if justified by sufficient suspicion. Sufficient suspicion is defined by a threshold on weighted occurrences of potentially harmful behavior, and accordingly, by a threshold on shares belonging together. Exceeding such a threshold results in the ability to reveal the secret, i.e. to recover the identity behind the pseudonyms. Thus the analyzers, holding the pseudonymized audit data including the shares, can reidentify the data if and only if the purpose of accountability requests to do so.

Since SSOs are considered potential adversaries threatening the anonymity of users, they must be confined to pseudonymous event data for user observation. Consequently, attacks on user anonymity are expected and countered as soon as pseudonymized event data leaves the pseudonymizer (see Fig. 1).

6 Envisioned Audit Architecture

Most modern operating systems offer similar audit components, and in the majority of cases is the identity of acting subjects part of the event information being recorded. A comprehensive solution requires pseudonymizing all event data observed by SSOs, but for our first prototype we chose to support the *syslog* event format only. Other possible event generators to be considered

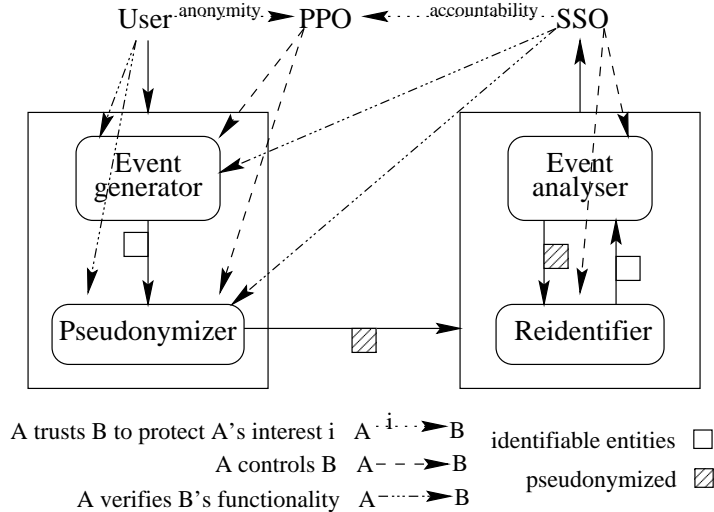


Figure 1: Model of trust relationships in our approach

include process accounting, network packet event data, TCSEC C2 audit data [23], [24], such as produced by Sun's BSM.

`syslogd` collects event data from user and kernel level sources, and it can additionally be fed remotely. `syslogd` can distinguish event sources, henceforth referred to as *facilities*, and event severity levels. Via *syslog* we can collect host-based data from applications and the kernel. Using additional kernel modules we can also collect event data derived from sources associated with the network. Out-of-band data derived from other sources is covered by corresponding applications using *syslog*. Owing to its uniformity and availability in all significant Unixes, *syslog* is widely utilized by the kernel and by user level applications for auditing. Generally, merely serious user actions appear in *syslog* events, but many important network services audit via *syslog* with diverse detail levels.

Pseudonymization can take place somewhere between the event generator and the border of the domain controlled by attackers. Placing the pseudonymizer closer to the event generator reduces opportunities to observe exposed plaintext identities, but scalability and ease of implementation suffer. We decided to pseudonymize the output event stream of `syslogd` by having the daemon write into pipes connected to the pseudonymizer. This way we can also pseudonymize event data normally written by applications to separate files.

6.1 Required Parsing

A pseudonymizer receives audit records from `syslogd` containing directly or indirectly identifying features [6]. Since each feature may contribute with a specific priority to an attack scenario A_g , the pseudonymizer shall be able to distinguish different event types and associate each feature type with an attack scenario and the respective priority or weight. The PPO specifies a priori knowledge about the syntax of events and the features to be pseudonymized. In cooperation with the SSO, the PPO models attack scenarios by specifying associations between features and

attack scenarios, as well as the weight of each feature’s contribution to its associated attack scenario.

To locate and pseudonymize identifying features in *syslog* audit records, we have to parse them. We quote some typical *syslog* records generated by applications on host *pony*:

```
Oct 20 20:48:29 pony identd[22509]: token TWpldDm02sq65FfQ82zX == uid 1000 (deedee)
Dec 19 16:02:45 pony su: BAD SU deedee to root on /dev/tty1
Dec 19 16:03:27 pony su: BAD SU deedee to root on /dev/tty1
Dec 19 16:03:53 pony su: BAD SU deedee to root on /dev/tty1
Dec 19 16:04:11 pony su: BAD SU deedee to root on /dev/tty1
Dec 19 16:04:39 pony su: deedee to root on /dev/tty1
Dec 20 09:58:46 pony postfix/smtpd[22190]: connect from pony.puf[192.168.1.1]
Dec 20 09:58:46 pony postfix/smtpd[22190]: BFF41AD8D: client=pony.puf[192.168.1.1]
Dec 20 09:58:52 pony postfix/cleanup[29752]: BFF41AD8D: message-id=<385D1476.4FDB096B@manda.rk>
Dec 20 09:58:53 pony postfix/qmgr[1519]: BFF41AD8D: from=<dexter@secret.lab>, size=7910 (queue active)
Dec 20 09:58:53 pony postfix/smtpd[22190]: disconnect from pony.puf[192.168.1.1]
Dec 20 09:58:55 pony postfix/local[4864]: BFF41AD8D: to=<deedee@puf>, relay=local, delay=7, ...
```

For brevity we will henceforth omit the *time and date* and *host* fields from audit records. Since the syntax of the records is mainly determined by the originating facility, recognition of different event and feature types is based on the evaluation of syntactical contexts, eg. by means of regular expressions for pattern matching as defined in POSIX 1003.2 Sect. 2.8.

The *facility* (framed fields 1 and 8 below) of a record indicates an application or a kernel component, optionally followed by its *process ID* (frame 2).

```
identd1 [225092]: token3 TWpldDm02sq65FfQ82zX == uid4 10005 (6deedee7)
su8: BAD SU9 deedee10 to root11 on12 /dev/tty113
```

An audit record represents an event, whose type is uniquely specified by the *event type context* (frames 3 with 4, and 9 with 12) and the facility. Each event type may contain a number of identifying *features* (frames 5, 7, 10, 11 and 13) of a type specified by a *feature type context* and the event type. A feature (f.i. frame 5) thus is an instance of a feature type determined by context (f.i. frames 4 and 6).

To determine a feature’s contribution to an attack scenario A_g , we assign its type f to a group I_g and a weight function $w_{fg}()$, representing A_g and the priority of f ’s contribution to A_g , respectively. The triplets $\langle f, I_g, w_{fg}() \rangle$ expand to quintuples:

$$\langle \text{facility, event type context, feature type context, } I_g, w_{fg}() \rangle$$

Above tuples, understood as a decision tree, such as in Fig. 2, can be used to parse audit records. By sequentially matching facility, event type context and feature type contexts, we isolate the event’s features, to be pseudonymized. For each feature, the corresponding entry in I_g is located, or instantiated in case the feature has not yet been observed. Then $w_{fg}()$ pseudonyms are generated, replacing the respective feature. The more pseudonyms are generated for a feature, the earlier can it be recovered (see Sec. 7).

For an illustrating example we chose some audit records including a simple password guessing attack to demonstrate the record handling (see above). We assume a knowledge specification that does not model realistic circumstances of attack scenarios, we rather arbitrarily chose the groups, thresholds and weights. According to that knowledge specification each of the example audit records has been augmented with the respective groups and weights for their matching, hence underlined features. Sequence numbers have been assigned to each record such that their original chronological order is preserved.

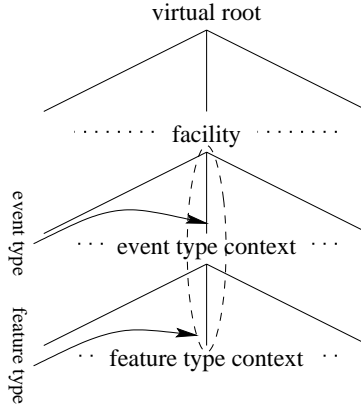


Figure 2: Representation of the parser knowledge as a tree

```

1: identd[22509]: token TWpldDm02sq65Ffq82zX == uid 1000I2,w=1 (deedeeI1,w=1)
2: su: BAD SU deedeeI1,w=3 to rootI1,w=1 on /dev/ttyI3,w=1
3: su: BAD SU deedeeI1,w=3 to rootI1,w=1 on /dev/ttyI3,w=1
4: su: BAD SU deedeeI1,w=3 to rootI1,w=1 on /dev/ttyI3,w=1
5: su: BAD SU deedeeI1,w=3 to rootI1,w=1 on /dev/ttyI3,w=1
6: su: deedeeI1,w=2 to rootI1,w=1 on /dev/ttyI3,w=1
7: postfix/smtpd[22190]: connect from pony.pufI4,w=1 [192.168.1.1I4,w=1]
8: postfix/smtpd[22190]: BFF41AD8D: client=pony.pufI4,w=1 [192.168.1.1I4,w=1]
9: postfix/cleanup[29752]: BFF41AD8D: message-id=<385D1476.4FDB096BI6,w=1@manda.rkI5,w=1>
10: postfix/qmgr[1519]: BFF41AD8D: from=<dexterI6,w=1@secret.labI5,w=1>, size=7910 (queue active)
11: postfix/smtpd[22190]: disconnect from pony.pufI4,w=1 [192.168.1.1I4,w=1]
12: postfix/local[4864]: BFF41AD8D: to=<deedeeI6,w=1@pufI5,w=1>, relay=local, delay=7, ...

```

Each of the twelve records has been pseudonymized as described above in accordance with the specification. Pseudonyms are denoted by place-holders indicating the group I_g , the identity's (id_{i_g}) index i_g into I_g , and the running number of the entity's pseudonym m_{i_g} . The six tables below present the states of the feature type groups I_1, \dots, I_6 after processing all twelve audit records. The column '#' marks the total number of audit records having been processed when the respective id_{i_g} is reidentifiable from its pseudonyms.

i_1	id_{i_1}	m_{i_1}	#
1	deedee	15	4
2	root	5	

$I_1, t_1 = 10:$

i_2	id_{i_2}	m_{i_2}	#
1	1000	1	

$I_2, t_2 = 50:$

i_3	id_{i_3}	m_{i_3}	#
1	/dev/tty1	5	5

$I_3, t_3 = 4:$

i_4	id_{i_4}	m_{i_4}	#
1	pony.puf	3	
2	192.168.1.1	3	

$I_4, t_4 = 100:$

i_5	id_{i_5}	m_{i_5}	#
1	manda.rk	1	
2	secret.lab	1	
3	puf	1	

$I_5, t_5 = 100:$

i_6	id_{i_6}	m_{i_6}	#
1	385D1476.4FDB096B	1	
2	dexter	1	
3	deedee	1	

$I_6, t_6 = 100:$

```

1: identd[22509]: token TWp1dDm02sq65FfQ82zX == uid I2:1.1 (I1:1.1)
2: su: BAD SU I1:1.2-4 to I1:2.1 on I3:1.1
3: su: BAD SU I1:1.5-7 to I1:2.2 on I3:1.2
4: su: BAD SU I1:1.8-10 to I1:2.3 on I3:1.3
5: su: BAD SU I1:1.11-13 to I1:2.4 on I3:1.4
6: su: I1:1.14-15 to I1:2.5 on I3:1.5
7: postfix/smtpd[22190]: connect from I4:1.1[I4:2.1]
8: postfix/smtpd[22190]: BFF41AD8D: client=I4:1.2[I4:2.2]
9: postfix/cleanup[29752]: BFF41AD8D: message-id=<I6:1.1@I5:1.1>
10: postfix/qmgr[1519]: BFF41AD8D: from=<I6:2.1@I5:2.1>, size=7910 (queue active)
11: postfix/smtpd[22190]: disconnect from I4:1.3[I4:2.3]
12: postfix/local[4864]: BFF41AD8D: to=<I6:3.1@I5:3.1>, relay=local, delay=7, ...

```

Note that the attacker’s identity `deedee`, according to the respective weights in records 2–6, generates more pseudonyms than other identities. With respect to the pseudonyms $I1 : 1.1 - 10$ of the first four records, `deedee` can be reidentified because the threshold t_1 is reached. Equally the identifying feature `/dev/ttyp1` reaches threshold t_3 with respect to the pseudonyms $I3 : 1.1 - 5$ in records 2–6. The identity `deedee` in record 12 is not associated with $I1$ as the other occurrences of `deedee`, but with $I6$. The respective pseudonym $I6 : 3.1$ therefore is *not* compatible with the pseudonyms $I1 : 1.1 - 15$.

7 An Approach Based on Secret Sharing

Basically our approach to pseudonymization should live up to the following requirements. Different pseudonyms can be issued for each transaction that involves a given identity, that is we have transaction-based pseudonyms. Pseudonyms issued for transactions of a given identity within a given attack scenario can be used to reveal that identity only if t or more of these pseudonyms are known. Knowledge of less than t pseudonyms does not enable identity recovery.

We can use secret sharing threshold schemes to split an identity into shares, which then are the pseudonyms, fulfilling above requirements. For this purpose we chose Shamir’s threshold scheme [25], which is perfect, ideal, and allows generation of new shares independently from preceding shares. Prioritization of transactions is possible by issuing a different number of shares for identities in the context of some kinds of transactions.

For our approach we apply Shamir’s threshold scheme somewhat different than in other applications of secret sharing, such as key splitting. Since the reidentifiers in our approach receive all shares generated by the pseudonymizer, and since each reidentifier is a potential adversary, it is not required to protect the confidentiality of shares during distribution. We also generate shares on demand, i.e. whenever specific events occur. Due to this stepwise generation of shares we pair the value of ordinate and abscissa, and we can’t take advantage of an optimization specific to Shamir’s scheme. To avoid inferences wrt. abscissa values, we have to use unique x-coordinates for all shares related to a given attack scenario.

To be independent of the size and characteristics of identities to be shared, we assign secrets to identities, which are unique within an attack scenario. We then use Shamir’s scheme to share these secrets. We have to fulfill some security requirements for the mapping of secrets to identities. First, reidentifiers need to know the mapping, but are controlled by SSOs. Then Unix access control cannot protect the confidentiality of the mapping. We thus use cryptograms of identities, considering the respective decryption keys as sharable secrets. Secondly, a user

reidentified once in a given attack scenario cannot act anonymously in further transactions that occur in that attack scenario, but don't complete it. We therefore consider expiring valid shares after an epoch, trading anonymity off against accountability.

Both, pseudonymizer and reidentifier know which shares belong to which attack scenario, but the shares, merely being ordinate-abscissa-pairs, provide no information, which of them belong to the same identity. If they did, they weren't transaction-based pseudonyms. A reidentifier thus can only guess, which shares stem from the same identity, i.e. are *compatible*. When guessing combinations of shares, a reidentifier sometimes will interpolate a value, which matches nowhere in above mentioned mapping. In other cases the reidentifier hits a secret. Unfortunately this match is *valid* only, if all involved shares are compatible. In case not all shares stem from the same identity, the match is considered a *mismatch*, which would result in holding an identity accountable for an event it did not cause.

We propose some straightforward approaches to handle mismatches. One can mark compatible shares with identical labels, but these labels effectively are relation-based pseudonyms for identities within a given attack scenario. To retain the advantages of transaction-based pseudonyms, one could defer the issuing of the labels of compatible shares, until t of them have been distributed.

We can also avoid mismatches without marking shares. To confirm a match, we have to provide a verifier for each identity cryptogram, which is the value of a secure one-way hash function, applied to the respective decryption key, i.e. the shared secret. The pseudonymizer can test for matches using all combinations of t incompatible shares, including the newly generated share to be issued. Since a match in this case is a mismatch, the new share must not be issued and a new one is to be chosen and tested. Although mismatches then will not occur at reidentifiers, they too have to search a match in combinations of t shares, including the new incoming share.

Note, that depending on t and the number of shares issued in the context of an attack scenario, the number of combinations to test grows very large. While reidentifiers could reside on machines dedicated to this task, pseudonymizers are located on the machine in use, to be able to protect the user's anonymity. If we need to avoid high computational cost at pseudonymizers, we let them issue unmarked shares without filtering mismatches. Reidentifiers then have to verify the validity of matches with the issuing pseudonymizer. In an online-protocol the reidentifier provides the pseudonymizer with information enabling validation of the compatibility property of the shares involved in the match, such as a reference to I_g , the shares and the match. Note, that we sacrifice the ability to perform offline identity recovery to reduce the pseudonymizer's performance costs.

Considering the unlinkability of shares, the ability for offline validation, and computational costs, the approach using deferred marking of shares seems most desirable.

To be able to take full advantage of transaction-based pseudonyms, it is necessary to hide or at least coarsen the number of actually involved users. This obviously cannot be achieved when shares are marked, but labels are not deferred. An approach to coarsen the number of actors is the introduction of dummy entities in the secret-to-identity mapping. The issues to be considered concerning dummy entities include how many dummies are needed for a certain degree of security, the upper bound for a useful number of dummies, the choice of the dummy identity string, and concealment of replacements of dummies with real identities.

The useful number of dummies, which from the viewpoint of attackers determines the number

of potential actors, is clearly bounded by the known total number of identities of the kind under consideration. Since this number may be large, it may be more storage-efficient to restrict the number of dummies to the product of the estimated number of actors and the estimated number of shares generated per actor. The identity string for dummies should be chosen to be indistinguishable from real identity strings, but if mismatches are not handled, dummies should be distinguishable from real identity strings. When a yet unknown real actor has to be added to the secret-to-identity mapping, we have to conceal which dummy is replaced with the new real identity. A simple way to achieve this, is to completely change the appearance of all entries, all exhibiting the same size. Since the updated mapping needs to be distributed to the reidentifiers, we also have to conceal the number of these transmission, e.g. by means of dummy traffic.

8 Open Issues and Further Research

There are a number of open issues to be investigated, such as the probability of mismatch occurrence, the transition from an epoch of share validity to the next epoch, as well as concerns mentioned above regarding dummy handling. Areas of interest wrt. to dummies are further methods for reducing the granularity of reidentifications, and potential benefits and required properties of dummy audit records. If we can do without the decoupling of identities and secrets, we don't need to solve the problems associated with the secret-to-identity mapping. Another important point is the correlation and reidentification of pseudonyms issued by distributed pseudonymizers for the same user.

Some desired properties wrt. reidentification are automation, technical purpose binding and infrequent involvement of a TTP. Also adversaries should not be able to link pseudonyms to the identities. This obviously is not the case, if pseudonyms are introduced within the system while the SSOs can observe the pseudonyms and the identities. As a result, we have to preclude SSOs from observing user identities in the monitored system. This is hard to accomplish if the IDS is local to the system under surveillance, or if IDS personnel has to perform tasks on that system (see Sect. 4.1, 4.3). Approaches such as ANIDA (Sect. 4.4), [19], [20] and [21] solve this problem by introducing pseudonyms external to the monitored system by means of a TTP, or by using a protocol that does not disclose the real identities. Since in these approaches the monitored system uses the pseudonyms for access control, damage can be avoided, even if pseudonyms are invalid, i.e. the identity will not be recoverable. Thus, users may generate the pseudonyms without involving a TTP.

We assume, that reasonable use of a complex system requires fine-grained access control. There are basically two different implementations of access control. Capabilities bestow the right to execute certain known operations on certain known objects upon subjects, which hold the capability. Since for the use of capabilities the identifier of the subject is irrelevant⁴, we can use all kinds of pseudonyms. Unfortunately the Unix systems we use do not support capabilities. Alternatively access control lists (ACL) bound to objects can be used to assign to certain known subjects the right to execute certain known operations. If the system trusts in the authenticity of a subject identifier, it just compares the identifier with entries in the ACL to determine the subject's rights. The subject then can use entity-based pseudonyms only, though

⁴Here we ignore the intricacies involved in implementing secure capabilities.

the pseudonyms stored in the ACL can be relation-based⁵. This basically is, what Unix allows for, if you consider user names being entity-based pseudonyms. Alternatively, if the system authenticates the subject in advance to each access using proofs as described in [19] and [20], there is no need for entity-based pseudonyms to be used by subjects. Then, even if transaction-based pseudonyms are used, the system needs to know against which property to authenticate. It seems advisable to authenticate group memberships, while the cardinality of the groups should be sufficiently large. Unfortunately, the Unix systems we use don't support this kind of access control. Pragmatism led us to the trust model we propose in Sect. 5. We seek to support transaction-based pseudonyms for IDS, but since off-the-shelf Unix only supports entity-based pseudonyms, we introduce pseudonyms independently from access control. Decoupling pseudonyms from access control is dangerous, because we can not avoid illicit accesses in the presence of invalid pseudonyms. We propose a TTP, known to generate valid pseudonyms, allowing identity recovery. If users were to generate pseudonyms on their own, the pseudonyms would have to comprise a proof vouching for pseudonym validity.

Assuming we have valid pseudonyms, i.e. reidentification is possible, then we have to decide, whether SSOs are trusted to respect the privacy of users. In case they are, adherence to purpose binding is expected when identity recovery information is used, thus recovery information may always be available to SSOs. Since primary tasks of SSOs in some environments may inherently conflict with user privacy, we may not be able to afford this kind of trust. We have to make sure, identity recovery information is usually not available to SSOs. In the approaches described in Sect. 4.1, 4.2 and 4.3, isolation of that information from SSOs would require extra tamper-proof hardware.

It is easy to achieve manual purpose binding for reidentification by placing the information needed for identity recovery at a TTP, which most of the time lays dormant. Only if accountability wrt. policy violations requires identity recovery, and upon good cause shown, the TTP makes use of the recovery information (see the approaches in Sect. 4.4.1). Manual purpose binding impedes automatic reidentification.

To achieve automatic identity recovery, which is a desired property of IDS, technical purpose binding is required. Note, that it is not sufficient to merely trust the IDS analysis component to adhere to purpose binding, for it is controlled by SSOs (see Sect. 4.2). Our approach provides for technical purpose binding independently of the IDS components controlled by SSOs, but it does so at the cost of always involving a TTP (here the PPO) to ensure generation of valid pseudonyms.

References

- [1] Birgit Pfitzmann, Michael Waidner, and Andreas Pfitzmann. Rechtssicherheit trotz Anonymität in offenen digitalen Systemen (in German). *Datenschutz und Datensicherheit*, 14(5-6):243–253, 305–315, 1990.
- [2] Kai Rannenberg, Andreas Pfitzmann, and Günther Müller. IT security and multilateral security. In Müller and Rannenberg [26], pages 21–29.

⁵The ACL-pseudonyms could be implemented as a one-way function of the concatenation of subject, object and operation.

- [3] Joachim Biskup and Ulrich Flegel. Transaction-based pseudonyms in audit data for privacy respecting intrusion detection. In *Proceedings of the Third International Workshop on Recent Advances in Intrusion Detection (RAID 2000)*, Toulouse, France, October 2000. Springer.
- [4] Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data. Official Journal L 281, October 1995. http://europa.eu.int/eur-lex/en/lif/dat/1995/en_395L0046.html.
- [5] Erster Senat des Bundesverfassungsgerichts. Urteil vom 15. Dezember 1983 zum Volkszählungsgesetz - 1 BvR 209/83 u.a. (in German). *Datenschutz und Datensicherung*, 84(4):258–281, April 1984. <http://www.datenschutz-berlin.de/gesetze/sonstige/volksz.htm>.
- [6] Michael Sobirey, Simone Fischer-Hübner, and Kai Rannenberg. Pseudonymous audit for privacy enhanced intrusion detection. In L. Yngström and J. Carlsen, editors, *Proceedings of the IFIP TC11 13th International Conference on Information Security (SEC'97)*, pages 151–163, Copenhagen, Denmark, May 1997. IFIP, Chapman & Hall, London.
- [7] Emilie Lundin and Erland Jonsson. Privacy vs intrusion detection analysis. In *Proceedings of the Second International Workshop on the Recent Advances in Intrusion Detection (RAID'99)*, West Lafayette, Indiana, September 1999. Purdue University, CERIAS.
- [8] Emilie Lundin and Erland Jonsson. Some practical and fundamental problems with anomaly detection. In *Proceedings of NORDSEC'99*, Kista Science Park, Sweden, November 1999.
- [9] Simone Fischer-Hübner and Klaus Brunnstein. Opportunities and risks of intrusion detection expert systems. In *Proceedings of the International IFIP-GI-Conference Opportunities and Risks of Artificial Intelligence Systems ORAIS'89*, Hamburg, Germany, July 1989. IFIP.
- [10] Simone Fischer-Hübner. *IDA (Intrusion Detection and Avoidance System): Ein einbruchsentdeckendes und einbruchsvermeidendes System (in German)*. Informatik. Shaker, 1993.
- [11] Michael Sobirey. Aktuelle Anforderungen an Intrusion Detection-Systeme und deren Berücksichtigung bei der Systemgestaltung von AID² (in German). In Hans H. Brüggemann and Waltraud Gerhardt-Häckl, editors, *Proceedings of Verlässliche IT-Systeme*, DuD-Fachbeiträge, pages 351–370, Rostock, Germany, April 1995. GI, Vieweg.
- [12] M. Sobirey, B. Richter, and H. König. The intrusion detection system AID – Architecture and experiences in automated audit trail analysis. In P. Horster, editor, *Proceedings of the IFIP TC6/TC11 International Conference on Communications and Multimedia Security*, pages 278–290, Essen, Germany, September 1996. IFIP, Chapman & Hall, London.
- [13] Michael Sobirey. *Datenschutzorientiertes Intrusion Detection (in German)*. DuD-Fachbeiträge. Vieweg, 1999.

- [14] Michael Meier and Thomas Holz. Sicheres Schlüsselmanagement für verteilte Intrusion-Detection-Systeme (in German). In Patrick Horster, editor, *Systemsicherheit*, DuD-Fachbeiträge, pages 275–286, Bremen, Germany, March 2000. GI-2.5.3, ITG-6.2, ÖCG/ACS, TeleTrust, Vieweg.
- [15] Roland Büschkes and Dogan Kesdogan. Privacy enhanced intrusion detection. In Müller and Rannenbergs [26], pages 187–204.
- [16] David Chaum. Untraceable electronic mail, return addresses, and digital signatures. *Communications of the ACM*, 24(2):84–88, February 1981.
- [17] D. Kesdogan, R. Büschkes, and J. Egner. Stop-and-go-mixes providing probabilistic anonymity in an open system. In *Proceedings of the 2nd Workshop on Information Hiding (IHW'98)*, number 1525 in LNCS, pages 83–98. Springer, 1998.
- [18] Teresa F. Lunt, R. Jagannathan, Rosanna Lee, Sherry Listgarten, David L. Edwards, Peter G. Neumann, Harold S. Javitz, and Al Valdes. IDES: The enhanced prototype, a real-time intrusion-detection expert system. Technical Report SRI-CSL-88-12, SRI Project 4185-010, Computer Science Laboratory SRI International, 1988.
- [19] Joe Kilian and Erez Petrank. Identity escrow. In *Proceedings of Advances in Cryptology (CRYPTO'98)*, pages 196–185, 1998.
- [20] Dan Boneh and Matt Franklin. Anonymous authentication with subset queries. In *Proceedings of the 6th ACM Conference on Computer and Communications Security*, pages 113–119, Kent Ridge Digital Labs, Singapore, November 1999. ACM SIGSAC.
- [21] Yuen-Yan Chan. On privacy issues of internet access services via proxy servers. In Rainer Baumgart, editor, *Secure Networking – CQRE[Secure]'99*, number 1740 in LNCS, pages 183–191, Düsseldorf, Germany, November 1999. secunet, Springer.
- [22] David Chaum, Amos Fiat, and Moni Naor. Untraceable electronic cash. In *Proceedings of Advances in Cryptology (CRYPTO'88)*, pages 319–327, 1988.
- [23] National Computer Security Center. US DoD Standard: Department of Defense Trusted Computer System Evaluation Criteria. DOD 5200.28-STD, Supersedes CSC-STD-001-83, dtd 15 Aug 83, Library No. S225,711, December 1985. <http://csrc.ncsl.nist.gov/secpubs/rainbow/std001.txt>.
- [24] National Computer Security Center. Audit in trusted systems. NCSC-TG-001, Library No. S-228,470, July 1987. <http://csrc.ncsl.nist.gov/secpubs/rainbow/tg001.txt>.
- [25] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. Discrete Mathematics and its Applications. CRC Press, Inc., Boca Raton, Florida, 1997.
- [26] Günter Müller and Kai Rannenbergs, editors. *Multilateral Security in Communications*. Information Security. Addison Wesley, first edition, 1999.