

Privacy Enhanced Intrusion Detection

Roland Büschkes

Aachen University of Technology – Department of Computer Science

Informatik 4 (Communication Systems)

D-52056 Aachen, Germany

roland@i4.informatik.rwth-aachen.de

Dogan Kesdogan

o.tel.o communications GmbH & Co

Dept. Enterprise Security

D-51063 Köln

Dogan.Kesdogan@o-tel-o.de

1.1 Abstract

This paper discusses solutions to protect the privacy of users during the application of intrusion detection systems (IDS) and introduces the notion of multilateral secure IDS. The surveillance of users by an IDS threatens their privacy. However, an IDS depends on data gathered by monitoring and must be able to unambiguously identify an intruder in case of an emergency. To mediate between the contrary interests of an IDS and the monitored users an IDS must obey the principles of data avoidance and reduction. Corresponding techniques concerning authentication and anomaly detection are discussed in this paper.

1.2 Introduction

The protection of the increasingly complex tele and data communication networks is a critical task, but the detection, repulse and prevention of abuse by in- and outsiders becomes more and more difficult.

Intrusion Detection Systems (IDS) provide a promising technique to enhance the security of complex information infrastructures. But their application mustn't weaken the security and privacy of the monitored users or any other cooperating components. An IDS fulfilling this requirement is called a multilateral secure IDS.

This paper deals with potential approaches towards the problem of privacy in the context of intrusion detection. It describes the approach of the *Aachener Network Intrusion Detection Architecture (ANIDA)* to the design of multilateral secure intrusion detection systems. ANIDA focuses on the surveillance of client-server network applications and the underlying protocol stacks, with a strong emphasize on privacy related issues.

The paper is organized as follows. In section 2 we give a short survey of intrusion detection techniques. Section 3 discusses the privacy issues related to the surveillance of users. Subsequently we describe our general architecture (section 4). In section 5 we discuss related works and finally draw some conclusions in section 6.

1.3 Intrusion Detection

Organizations and network providers have a rising demand concerning security. But classical security mechanisms, i.e. authentication and encryption, and infrastructure components like firewalls cannot provide sufficient security. Therefore, intrusion detection systems (IDS) have been introduced as a third line of defense (see e.g. [14] for a general overview).

The techniques classically applied within an IDS can be subdivided into the two main categories [17] of

- Misuse Detection, and
- Anomaly Detection.

Misuse detection (see e.g. [8, 12, 13]) tries to detect patterns of known attacks within the audit stream of a system, i.e. it identifies attacks directly.

Explicitly describing the sequence of actions an attacker takes, misuse detection is based on the specification of the undesirable or *negative behavior* of users and processes. The opposite approach would be the specification of the desired or *positive behavior* of users and processes. Based on this normative specification of positive behavior attacks are identified by observing derivations from the norm. Therefore, this technique is called *Anomaly Detection*.

The main problem with anomaly detection techniques is to determine the positive behavior. Two general approaches exist:

1. learning of user and process behavior
2. specification of user and process behavior

The former approach is often based on statistical methods (e.g. [9]). Other methods use learning algorithms like e.g. neural networks or Bayesian classifiers [2]. This approach is particularly popular for the profiling of users.

The latter approach, specification-based anomaly detection, was first proposed in [11]. It is based on the formal description of positive behavior, e.g. in form of a grammar.

Assuming that intrusions and intrusion attempts are an exception and not the rule within a network, the use of any of these techniques involves a certain danger, namely the breach of the users privacy.

1.4 IDS and Privacy

The application of an IDS explicitly introduces a surveillance (Figure 1) facility, which weakens the security and privacy of the monitored users. Obviously, there is a conflict between the organizations need for security on the one side, and individuals need for privacy on the other.

This conflict can be avoided by the application of a multilateral secure IDS, i.e. an IDS which allows all involved parties to protect their own interests.

A multilateral secure IDS must obey two major design principles:

1. data avoidance
2. data reduction

Following the principle of data avoidance a user should only be forced to disclose the minimum of information necessary to the IDS. Data avoidance is especially relevant in the context of identification

and authentication. An IDS does not need to know the identity of a monitored user, until it provably detects an abuse.

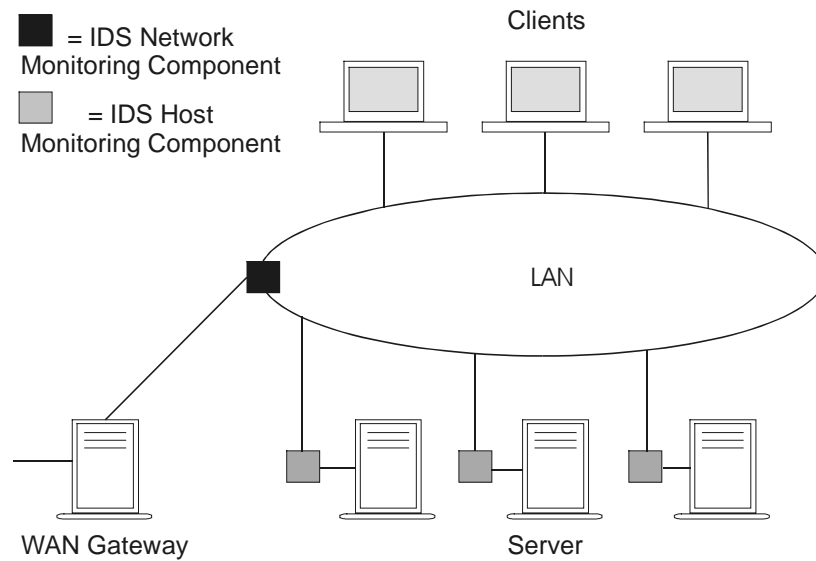


Figure 1 : LAN monitored by an IDS

Data reduction stresses the fact, that an audit stream contains a lot of unsuspecting events. There is no need – at least from a security point of view – to store these events or to make them accessible to a human operator. A multilateral secure IDS should therefore filter the audit stream and only store records of critical events and detected intrusions.

1.5 The Architecture

Our approach to the design of a multilateral secure IDS is driven by the principles of data avoidance and reduction. The adherence to these principles results in the concept of *transactions under pseudonyms* and the control flow depicted in Figure 2.

A user accesses a service over a network under a pseudonym (data avoidance). This pseudonym is generated in cooperation with a *trusted third party* (TTP). The TTP is responsible for checking the user's identity and issuing the corresponding credentials. As the network and the service are monitored by the IDS, the events contained within the audit stream are related to the pseudonym. The IDS continues its normal work. In our architecture we take a hybrid approach by combining anomaly and misuse detection components. As a matter of fact, our anomaly detection component uses a new technique of anomaly detection, namely transaction-based anomaly detection.

Transaction-based anomaly detection is especially suited to meet the data reduction requirement, as it filters out all non-critical and unsuspecting events (denoted by ϵ). All suspicious events remain in the audit stream and are passed on to the next level of processing. On this level a human operator can, in combination with a standard misuse detection component, examine the suspicious events more closely and classify them. This examination results in the deletion of additional uncritical events (denoted by ϵ). The remaining events indicate - with a certain error probability - an intrusion or intrusion attempt and provide evidence that an attack has been launched under a certain pseudonym. This evidence can be presented to the TTP and results in the revelation of the user's real identity.

As a consequence our architecture realizes the concept of two domains (see Figure 2). The first domain knows the true identity of the user, while the second or any other domain does not. In the second domain the IDS completely controls the surveillance process.

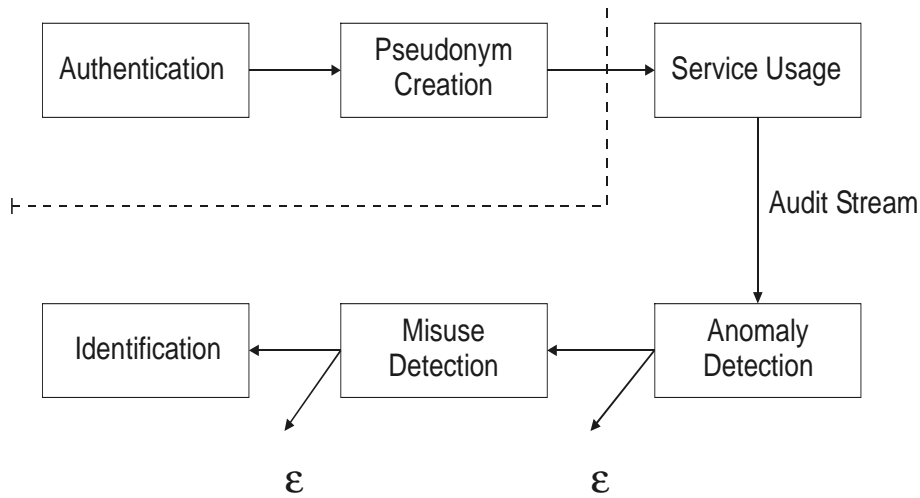


Figure 2: Control flow

For the further discussion we focus on the network scenario depicted in Figure 1 and assume the following scenario:

The IDS focuses on client-server network applications and the related communication processes. Our model is therefore based on the ISO/OSI reference model (see e.g. [6]), which is made up of seven protocol layers. The IDS runs monitors on the server and the network, i.e. it has access to the network services and network traffic related data. For ease of presentation we assume that a single network monitor collects all network relevant information (layer 1 – 4) and the host monitors collect only information related to the services (layer 5 – 7).

1.5.1 Data Avoidance

The principle of data avoidance is especially relevant in the context of identification and authentication. Until an IDS does not have a well-founded suspicion, it does not need to know anything about the identity of a user. And even in the case of a suspicion, the suspicion can be refuted through the presentation of additional credentials, but without the revelation of the user’s identity. A straightforward idea in order to build a multilateral secure IDS is therefore to apply anonymity or pseudonym techniques. It is obvious that anonymity techniques are not suited for the use in systems, which are monitored by an IDS, because in case of an incident it must be possible to reveal the true identity of a user. Therefore pseudonyms are the preferable choice.

A pseudonym is an identifier for a user to a transaction, which is not, in the normal course of events, sufficient to associate the transaction with an individual user. The user gives only as much information about himself as is strictly necessary to access a service.

Several questions are related to the usage of pseudonyms in the context of intrusion detection:

1. When to introduce the pseudonym for a user (at login time, before the pass on of log files, etc.)?
2. Where to introduce the pseudonym (at the login server, at the monitoring process, etc.)?
3. How to generate the pseudonym (general technique, number of participating parties, etc.)?

4. How to reveal the pseudonym in case of an intrusion (general technique, number of participating parties, etc.)?
5. What additional data must be treated in a special way in order to prevent unwanted revelation of a pseudonym (access to user specific resources like home directories, etc.)?

As we follow the concept of two domains, the pseudonym has to be introduced before any request of the client enters the domain monitored by the IDS. However, it is not possible to let the users or the IDS introduce the pseudonyms. If the users themselves introduce the pseudonyms, the IDS will depend on their cooperation. If the IDS inserts the pseudonyms into the audit stream, it will have complete control concerning the identification of the users.

A viable solution is to introduce a Trusted Third Party. The TTP approach introduces an intermediate between the IDS and the users. The TTP identifies a user and issues a certified pseudonym. In every subsequent service request the user is identified by the certified pseudonym. Therefore, the host or network based monitoring components of an IDS capture only the pseudonyms related to single events, but never the true identity of a user.

Nonetheless, the sole use of user pseudonyms does not prevent the IDS from identifying a user. If identification is not separated from the connection, the connection is sufficient to identify a user. In that case the underlying protocol stack has also to be taken into account.

This results in two possible approaches to the problem of data avoidance with regard to identification and authentication:

1. hiding identifying information related to a connection (addresses etc.)
2. hiding the connection

Before we consider these two possibilities in the following subsections in detail, we discuss the consequences of the early introduction of pseudonyms for the operating system and network environment and introduce the term of *group reference pseudonyms*.

Group Reference Pseudonyms

The early introduction of pseudonyms brings up some problems for other network components. If a user requests a network service and identifies himself through a pseudonym it is not possible for the service to verify the users access rights. In addition, an statistical anomaly detection component cannot profile users if they change their pseudonyms on a regular basis.

To solve these problems we enhance the concept of reference pseudonyms to the concept of group reference pseudonyms (GRPs). Group reference pseudonyms are issued by the TTP and differ from normal reference pseudonyms in the sense that they contain additional information about a user, namely the group(s) to which a user belongs. This group information fulfills the following functions:

- The group information defines the anonymity set of the user.
- The group information is the base for the verification of a user's access rights.
- The group information is the base for the profiling of users through a statistical anomaly detection component. Instead of profiling single users a group of users is profiled. This has the additional advantage that a single user cannot fool the anomaly detection component anymore by slowly modifying his standard behavior. This kind of fooling is only possible if the majority of the group to which he belongs cooperates.

- The group information provides the necessary information to detect related events belonging to an intrusion attempt. At first instance the group is identified as the origin of an intrusion.

The introduction of GRPs does not influence the work of a misuse detection component. Instead of dealing with real user identities, it makes its analysis on the base of pseudonyms and group affiliations.

Based on the GRPs we now discuss two solutions to the identification and authentication problem.

Ticket Based Solutions

Making the initial assumption that the connection itself does not reveal the identity of a user, we can use extended Kerberos tickets (see e.g. [4]). The TTP generates a modified Kerberos ticket for the user, which contains a certified GRP. The ticket has a limited period of validity and must be renewed periodically. For each renewal a new GRP is generated.

The base elements of the standard Kerberos protocol are tickets and authenticators. A ticket contains the identities of the client and the server, a validity period, and the key used for the communication between the two parties: $\{C, S, t_1, t_2, K_{C,S}\} \equiv \{ticket(C, S)\}_{K_S}$. We omit the network address from the ticket.

The authenticator contains the identity of the client, a timestamp and optionally an additional session key: $\{C, t\}_{K_{C,T}} \equiv \{auth(C)\}_{K_{C,T}}$.

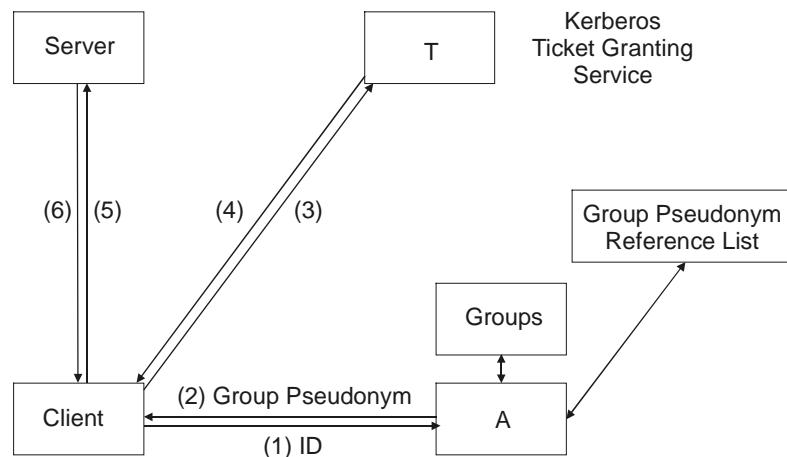


Figure 3 : Ticket based solution

We extend the standard Kerberos protocol in a way, that the ticket for the ticket granting service (T) does no longer contain the identity of the client C. Instead the authentication service (A) includes a GRP in the ticket, which is returned to C (2). C identifies himself towards T (3) and all other servers S (5) with the help of the group pseudonym. The total protocol is therefore:

$$C \rightarrow A \quad \{C, T, n_1\}_{PK_A} \quad (1)$$

$$A \rightarrow C \quad \{GRP, K_{GRP,T}, n_1\}_{K_C}, \{ticket(GRP, T)\}_{K_T} \quad (2)$$

$$C \rightarrow T \quad \{auth(GRP)\}_{K_{GRP,T}}, \{ticket(GRP, T)\}_{K_T}, S, n_2 \quad (3)$$

$$T \rightarrow C \quad \{K_{GRP,S}, n_2\}_{K_{GRP,T}}, \{ticket(GRP, S)\}_{K_S} \quad (4)$$

$$C \rightarrow S \quad \{auth(GRP)\}_{K_{GRP,S}}, \{ticket(GRP, S)\}_{K_S}, request, n_3 \quad (5)$$

$$S \rightarrow C \quad \{n_3\}_{K_{GRP,S}} \quad (6)$$

The steps different from the original Kerberos protocol are steps 1 and 2:

(1): We additionally introduce an asymmetric key pair for the TTP. The identity of the client is encrypted with the public key of the TTP, i.e. the Kerberos authentication service.

(2): A group reference pseudonym is added to the message returned to the client and the ticket for the ticket-granting server is issued with reference to the pseudonym.

In order to evaluate the security of the protocol, i.e. to analyze under what circumstance it is possible for the IDS to identify a user without the need to interact with the TTP, we refer to the network scenario depicted in Figure 1. Generally the IDS can extract and combine information from any combination of monitoring components, but additionally we make the necessary assumption that the TTP is not monitored by the IDS. This results in three general cases to be considered:

- Host Monitor only: If the IDS only uses the information available from the host monitor of the server S it cannot identify the user. S only sees the GRP based authenticator, the GRP based ticket and the service request.
- Network Monitor only: A standalone network monitor can try to detect the relation between step 1 and 2 of the protocol. The encryption of the first message and therefore the encryption of the identity of the user prevents an easy relation between the identity of the user and the pseudonym returned by C. In all subsequent protocol steps the GRP is encrypted and the network monitor can only observe the general existence of a communication relationship between C and S, i.e. between the corresponding network addresses.
- Host and Network Monitor: If the IDS uses information from the host and the network monitor it can relate the GRP to the network address of the requesting client. Recalling our assumption that the connection itself does not reveal the identity of a user it is not possible for the IDS to make a relation between the identity of the user, the network addresses he is using and his current pseudonym.

However, our assumption is only true if information related to the protocol stack (e.g. network addresses) cannot be associated directly with a certain identity. Dropping this assumption propagates the problem of pseudonym generation from the application layer down the protocol stack, i.e. the identifying information on every protocol layer has to be substituted by pseudonyms. For a TCP/IP protocol stack running on an Ethernet for instance, the *Dynamic Host Configuration Protocol* (DHCP) can be used on the IP level. The MAC layer must also use dynamic addresses. Solutions to this problem will be discussed elsewhere.

If the association of an identity and an address is unavoidable, other techniques have to be used.

MIX Based Solution

The second approach is based on the classic MIX technique [3, 5, 10, 15]. A MIX node acts as the intermediary between the clients and the servers. The MIX node relays the service requests of the clients and the answers of the servers while substituting the users real identities with group reference pseudonyms.

The general MIX functionality ensures that the service requests issued by the clients cannot be related to the service request issued by the MIX on behalf of the clients, i.e. the connection is hidden from the IDS. A MIX collects a number of packets from distinct users and changes the appearance (i.e. the bit pattern) and the order of these incoming packets. Normally the incoming packets are encrypted with the public key of the MIX; i.e. public key cryptography is used. In our case we presuppose an unconditional trustworthy MIX and do not deploy a MIX cascade. Therefore we can restrict ourselves to the use of symmetric cryptography.

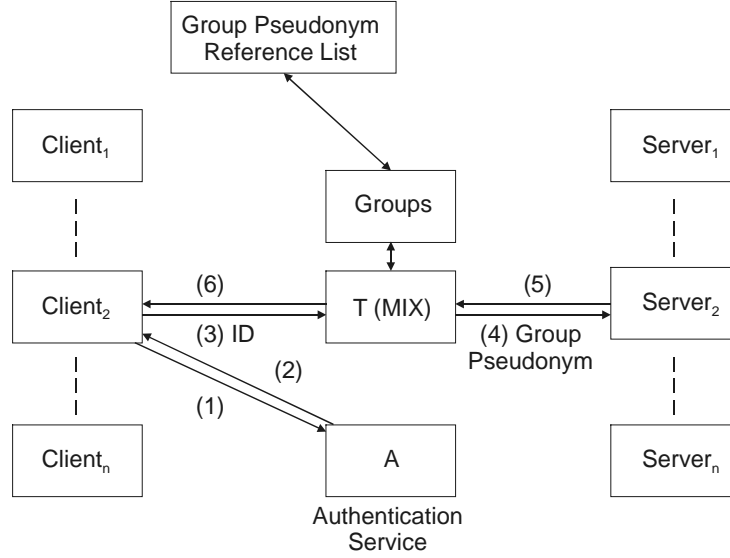


Figure 4: MIX based solution

The general protocol looks as follows:

$$C \rightarrow A \quad C, T, n_1 \quad (1)$$

$$A \rightarrow C \quad \{K_{C,T}, n_1\}_{K_C}, \{ticket(C, T)\}_{K_T} \quad (2)$$

$$C \rightarrow T \quad \{auth(C)\}_{K_{C,T}}, \{ticket(C, T)\}_{K_T}, \{S, request, n_2\}_{K_{C,T}} \quad (3)$$

$$T \rightarrow S \quad \{GRP, S, request, n_3\}_{K_{T,S}} \quad (4)$$

$$S \rightarrow T \quad \{S, GRP, answer, n_3\}_{K_{T,S}} \quad (5)$$

$$T \rightarrow C \quad \{S, answer, n_2\}_{K_{C,T}} \quad (6)$$

The first two steps of the protocol are the standard steps of the Kerberos protocol. The client C identifies himself towards the authentication service A and receives a ticket for the MIX T, which is responsible for the contact to the other servers. The following steps differ from the original Kerberos protocol:

(3): C presents T his ticket and authenticates himself. In addition he hands his service request, consisting of the target server S and the related data, over to T. A nonce protects the client from replay attacks.

(4): T relays C's service request to S. Before doing this T substitutes C's identity by a group reference pseudonym. A nonce protects T from replay attacks.

(5): S checks the access rights for the service requests on the base of the group pseudonym and returns the answer to T.

(6): T re-identifies the user and sends him the result of his service request.

For the security evaluation of the protocol we refer again to the network scenario depicted in Figure 1, i.e. the IDS monitoring the network consists out of host and network monitoring components. Again the TTP is not monitored. Note that this time the TTP corresponds to the MIX node. The authentication server A can be subject of monitoring. Again this results in three general cases to be considered:

- Host Monitor only: If the IDS only uses the information available from the host monitor of the server S it cannot identify the user. S only sees the GRP based authenticator and the service request.
- Network Monitor only: A standalone network monitor can neither detect the identity of a user nor the existence of a communication relationship. The former is guaranteed by the encryption of the service requests and the GRP. The network monitor can observe the identity of a user in step 1 and therefore relate it to a later service request (3), but this does neither reveal the relation between an identity and a pseudonym nor does it give the network monitor information about the requested service. The identification of the requested service is not only prohibited by the encryption, but also by the MIX functionality. The MIX functionality makes it impossible to relate incoming service requests (3) issued under the real identity of a user to outgoing service requests (4) issued under the GRP. The same property holds for the service replies.
- Host and Network Monitor: The combination of the information of the host and network monitors does not reveal the relation between an identity and a pseudonym. Because the network monitor cannot identify the communication relationship, i.e. the relation $C \leftrightarrow GPR$ and a single server only has the GRP under which a services was requested, it is not possible to re-identify a user.

The MIX based solution, under the assumption that the MIX is trustworthy, is therefore practical secure.

1.5.2 Data Reduction

The preceding protocols enable a user to hide his identity on all layers of the protocol stack and to disclose only the minimum of information necessary to access the network services. Due to the surveillance of these services and the related communication processes the actions of the users are logged under their pseudonyms.

Following the design principle of data reduction, the resulting audit stream should be reduced as far as possible. Misuse detection techniques are not suited for this task, as they only search the audit stream for attack patterns. Anomaly detection techniques are more suited, as they are based on the explicit definition of normal behavior. As a consequence, the observed normal events can be deleted from the audit stream. Assuming that intrusions and intrusion attempts are the exception, not the rule, this will significantly reduce the audit stream and the amount of audit data stored for further processing.

The anomaly detection technique used in the context of ANIDA is based on the following general observations:

1. Communication protocols can generally be specified as state transition systems.
2. The protocols can be considered as defining valid transactions.

Therefore, our approach utilizes the transaction model, which is typical for many services and communication processes. Transactions describe atomic operations. The provision of an atomic operation [4] means that the effect of performing any operation on behalf of one client is free from interference with operations being performed on behalf of other concurrent clients; and either an operation must be completed successfully or it must have no effect at all. Atomic transactions are often characterized by the ACID principle [7]:

1. *Atomicity*: All operations of a transaction must be completed, i.e. a transaction is treated as a single, indivisible unit.
2. *Consistency*: A transaction takes the system from one consistent state to another.
3. *Isolation*: Each transaction must be performed without interference with other transactions.
4. *Durability*: After a transaction has successfully been completed all its results are saved in permanent storage.

Obviously, the simple serial execution of transactions preserves the ACID properties, but is not efficient. Therefore, the task of a *scheduler* is to maximize concurrency and to execute transactions interlocked. The scheduler ensures that the transactions are executed in a *serially equivalent* way. Depending on whether or not the scheduler is able to execute the transaction without conflicts it is committed or aborted.

Transaction-based Anomaly Detection

One common argument concerning the difficulty of detecting intrusive behavior is that attacks normally consist of single steps, each of which performs a legal operation. These legal steps are generally used to interfere with another process also performing legal operations.

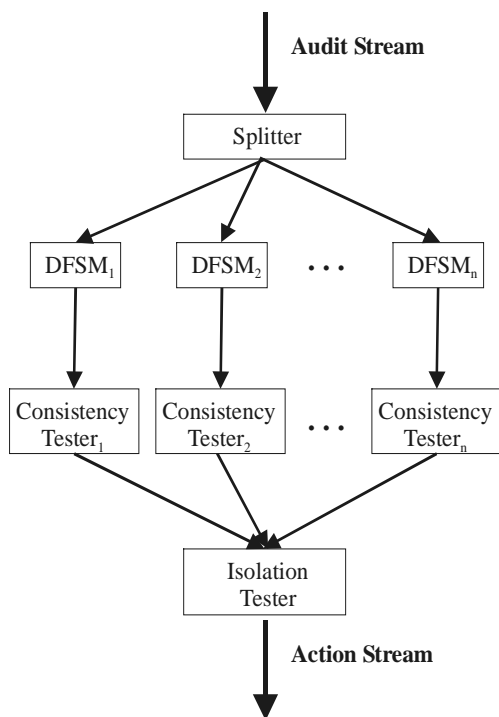


Figure 5: General architecture

Considering the victim's and the attacker's sequence of operations as transactions, an attack will obviously not be successful if the ACID properties are guaranteed for the victim's transaction. The transaction concept itself can therefore be used to detect intrusions. Like an optimistic scheduler, an IDS can check each transaction at its end concerning its serially equivalent execution and its adherence to the ACID properties.

If the check fails, the transaction is trapped. This is the basic idea of transaction-based intrusion detection. In general the transaction-based approach is similar to the specification-based approach as it formally describes positive behavior. In contrast to the specification-based approach it specifies the desired actions and sequence of actions by the definition of transactions. In addition it naturally supports the aims of a multilateral secure IDS, because committed transactions can immediately be deleted from the audit stream. In combination with the application of pseudonym techniques this results in *transactions under pseudonyms* as the basic unit of surveillance.

As already mentioned our model is based on the ISO/OSI reference model and its seven protocol layers. The proto-

cols of each of these layers can be described by *deterministic finite state machines* (DFSM).

A DFSM A is a 5-tuple $A = (Q, \Sigma, q_0, \delta, F)$ with Q being the set of states which can occur during a transaction, Σ the union of all inputs, q_0 the initial state, $\delta : Q \times \Sigma \rightarrow Q$ the transition function, and F the set of final states. The syntax and semantics of a DFSM are unambiguously determined by the transition function. The initial state of the DFSM is considered to be consistent. The DFSM passes through the operations (input events) that form the transaction. The transaction is successful if the final state is also consistent. To ensure this, atomicity, consistency and isolation properties have to be checked. The atomicity is checked on the base of the DFSM. The DFSM defines a language $L(A)$. By checking whether $w \in L(A)$ holds for a trace w extracted from the audit stream the transaction is tested for the atomicity property.

Subsequently the assertions (consistency property) are checked, which are expressed with the help of first order logic (FOL). Each DFSM can include assertions. Assertions can be valid only for a specific transaction (local assertions) or, on a more global level, for a certain layer (layer assertions) of the protocol stack.

Finally the serially equivalent execution (isolation) of the transaction is checked (only if necessary).

The overall architecture is depicted in Fig. 5 and described in more detail in the following subsections.

The Splitter

The audit stream contains all events received from the different sources. In case of the network monitoring component e.g. the audit stream consists of packets received from the network. The task of the splitter is to distribute these single events to the corresponding state machines. The assignment of events to a state machine is straightforward, as it follows the rules of the corresponding communication and service protocols. Taking the TCP/IP protocol stack as a practical example, a TCP session is uniquely identified by the 4-tuple (*source address, source port, destination address, address port*), whereas a UDP packet is identified by its socket address.

The splitter is also responsible for the scheduling of the isolation tester. The scheduler monitors the event stream for any potential conflicts and feeds the isolation tester with the necessary information.

Template State Machine

As mentioned above, the theory of DFSMs is in general sufficient to represent the protocol state machines. Nonetheless, a major disadvantage of the DFSM theory is the nonexistence of variables. Therefore, a separate transition has to exist for each possible value of a variable during a protocol run. This blows up the state machines substantially.

To keep the state machines small we use *Template State Machines* (TSM) [1]. A TSM represents a protocol in a value independent way. The actual instance of the protocol template, i.e. the concrete DFSM, is derived during run time from the audit data.

A TSM A is a 6-tuple $(Q, \Sigma, V, q_0, \delta, F)$ with Q being the set of states which can occur during a transaction, Σ the union of all inputs, V the set of variables, q_0 the initial state, δ the transition function, and F the set of final states.

The transition function δ is defined as $\delta : Q \times (AExp(\Sigma, V))^+ \rightarrow Q$ with $AExp(\Sigma, V)$ being defined by $e ::= z \mid x \mid e_1 + e_2 \mid e_1 - e_2 \in AExp(\Sigma, V)$ with $z \in \Sigma$ and $x \in V$.

In contrast to the DFSM, the semantics of a TSM is not determined solely by the transition function, but also by the instantiation of the variables.

The TSM mechanism serves several purposes:

- It enables the formal and compact specification of a protocol, which can dynamically be fed into a monitor.
- It prevents state explosions.
- It can be used for a consistency and correctness check of the protocol specification.
- It can be used for the graphical specification of protocols.

In this paper we concentrate on the two former points.

Consistency Tester

After the atomicity of a transaction has been checked with a TSM, the next step is to ensure that the transaction leaves the system in a consistent state. The consistent state itself is defined by so-called assertions. To express the assertions we use first order logic (*FOL*) with the restriction that the negation is only allowed for atomic formulas (*FOL*⁺).

In general, we distinguish between two kinds of assertions, local assertions and layer assertions, respectively:

1. Local assertions are related to a specific TSM, and are checked during the execution of transitions. Therefore, we have to add Boolean expressions to our TSM concept. The transition function is adapted to $\delta : Qx(AExp(\Sigma, V))^+ \times BExp \rightarrow Q$ with *BExp* being defined by $b ::= true \mid false \mid e \mid e_1 = e_2 \mid e_1 < e_2 \mid e_1 \neq e_2 \mid b_1 \wedge b_2 \mid b_1 \vee b_2$. Obviously, we do not need the whole expressiveness of *FOL*⁺ for local assertions, as quantifiers do not necessarily make sense within the context of a TSM. Each transition is triggered by a single packet, which is checked for obeying the local assertions. No quantifiers are necessary in this context.
2. Layer assertions are valid for all TSMs belonging to a single layer. In contrast to local assertions layer assertions can make use of the complete *FOL*⁺, including universal and existential quantification. They can be checked either at the initial state or at a final state, i.e. before or after checking for atomicity. In general, a violation of an assertion should be detected as early as possible.

Isolation Tester

The isolation tester is responsible for checking the isolation property, i.e. for checking whether a transaction performed without interference from other transactions. For general processes running on a single node the application of the isolation tester is obvious: it can detect attacks based on race conditions. To do so, the splitter has to monitor the audit stream for accesses to critical objects (e.g. directories and files) and start the isolation tester, which checks for a violation of the isolation property. With regard to communication processes the use of an isolation tester is not immediately obvious. Nonetheless, certain attacks on the protocol level can be interpreted as a violation of the isolation property. This is especially true if we take the distributed nature of a communication process into account, i.e. if we do not only consider the effects of a protocol run on a single system, but broaden our transaction concept to include all nodes involved in the communication process. This results in the so-called compound transaction concept. For ease of brevity, we omit the further details related to transaction-based anomaly detection and compound transaction in this paper and refer the interested reader to [1], which also contains concrete examples of TSMs and attacks.

A transaction which has passed the described tests for atomicity, consistency and isolation obviously obeys the defined security policy. From the security point of view, there is no reason to keep the events describing the details about the communication in the audit stream. They can completely be deleted. Alternatively, they can be substituted by the high-level description of the transaction. In both cases the amount of stored data is significantly reduced. This also eases the job of any succeeding misuse detection component.

1.6 Related Works

Privacy questions related to the application of an IDS have – to our knowledge – not been considered extensively yet. Especially there are no works dealing with the design of misuse or anomaly detection techniques, which address the privacy interests of the users directly.

[16] proposes a *pseudonymous audit* in the context of the *Adaptive Intrusion Detection (AID)* system. In order to provide a privacy enhanced audit based monitoring any user identifying information within the audit data is substituted by pseudonyms before it is stored in the local audit data files. This substitution is done within the kernel of the operating system. The pseudonyms themselves are created by secret key encryption and the key is switched from time to time.

The main difference to our approach is that it leaves the responsibility for the generation of pseudonyms with the monitored host itself. To provide a secure and fair monitoring of the network and the users it is necessary to transfer the power of re-identification to an independent third party or even to divide it between several parties.

1.7 Conclusions

The principles of data avoidance and reduction are important cornerstones in the design of multilateral secure intrusion detection systems.

The data avoidance solutions proposed in this paper stress that future intrusion detection systems must be embedded in the general network and operating system environments. Only the cooperation with the general authentication and access control mechanisms makes it possible to design and build a multilateral secure IDS. While the described approaches differ in their security and complexity, they provide a viable way to mediate between the contrary interests of the users and the network providers.

The proposed data reduction solution provides both, a promising anomaly detection approach and a useful privacy enhancing technique. Together with one of the data avoidance techniques, transactions under pseudonyms become the basic unit of surveillance.

As a concluding remark it is worth noting, that transaction-based anomaly detection can also be used to protect the IDS itself from attacks, as each IDS itself represents a valuable target for attacks. The same holds for the TTP.

Future work will focus on the extension and refinement of ANIDA. With regard to the privacy issues discussed in this paper this will include the validation of the technical feasibility from the viewpoint of an operating system (access control) and from the viewpoint of an IDS.

1.8 References

[1] R. Büschkes, M. Borning, and D. Kesdogan, “Transaction-based Intrusion Detection”, *Proceedings of the 1st USENIX Workshop on Intrusion Detection and Network Monitoring*, USENIX, April 1999.

- [2] R. Büschkes, D. Kesdogan, and P. Reichl, "How to increase security in mobile networks by anomaly detection", *Proceedings of the 14th Annual Computer Security Applications Conference (ACSAC'98)*, ACM, December 1998.
- [3] D.L. Chaum, "Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms", *Communications of the ACM*, 24(2): 84-88, February 1981.
- [4] G. Coulouris, J. Dollimore, and T. Kindberg, *Distributed Systems - Concepts and Design*. Addison-Wesley, 2nd edition, 1994.
- [5] A. Fasbender, D. Kesdogan, and O. Kubitz, "Variable and Scalable Security: Protection of Location Information in Mobile IP", *Proceedings of the VTC'96*, Atlanta, 1996.
- [6] Fred Halsall, *Data communications, computer networks, and open systems*, Addison-Wesley, 3rd edition, 1992.
- [7] T. Härder and A. Reuter, "Principles of transaction-oriented database recovery", *Computing Surveys*, 15(4), 1983.
- [8] K. Ilgun, Richard A. Kemmerer, and Phillip A. Porras, "State transition analysis: A rule-based intrusion detection approach", *IEEE Transactions on Software Engineering*, 21(3):181-199, March 1995.
- [9] H. Javitz and A. Valdes, "The SRI IDES statistical anomaly detector", *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 316-326, May 1991.
- [10] D. Kesdogan, J. Egner, and R. Büschkes, "Stop-And-Go-MIXes Providing Probabilistic Anonymity in an Open Systems", *Proceedings of the second workshop on information hiding (IHW98)*, Lecture notes in computer science (Springer-Verlag).
- [11] C. Ko, *Execution Monitoring of Security-Critical Programs in a Distributed System: A Specification-Based Approach*, PhD thesis, University of California, Davis, 1996.
- [12] S. Kumar, *Classification and Detection of Computer Intrusions*, PhD thesis, Department of Computer Science, Purdue University, August 1995.
- [13] A. Mounji, *Languages and Tools for Rule-Based Distributed Intrusion Detection*, PhD thesis, Facultés Universitaires Notre-Dame de la Paix Namur, Belgium, September 1997.
- [14] B. Mukherjee, L.T. Heberlein, and K.N. Levitt, "Network Intrusion Detection", *IEEE Network*, 8:26-41, May/June 1994.
- [15] A. Pfitzmann, B. Pfitzmann, and M. Waidner, "ISDN-MIXes: Untraceable Communication with Very Small Bandwidth Overhead", *Proceedings of IFIP/SEC 91*, Brighton, UK, 15-17 May 1991, D.T. Lindsay, W.L. Price (eds.), North-Holland, Amsterdam 1991, pp. 245-258.
- [16] M. Sobirey, S. Fischer-Hübner, and K. Rannenberg, "Pseudonymous Audit for Privacy Enhanced Intrusion Detection", *Proceedings of IFIP/SEC 97*, 1997.
- [17] A. Tucker Jr., editor, *CRC Computer Science and Engineering Handbook*, CRC Press, December 1996.