A Framework for Evaluating Privacy Protection of Authentication Systems

Nakamura, Toru
Graduate School/Faculty of Information Science and Electrical Engineering, Kyushu University

Inenaga, Shunsuke
Graduate School/Faculty of Information Science and Electrical Engineering, Kyushu University

Ikeda, Daisuke
Graduate School/Faculty of Information Science and Electrical Engineering, Kyushu University

Baba, Kensuke
Graduate School/Faculty of Information Science and Electrical Engineering, Kyushu University

他

出版情報：Abstracts of the 2008 Symposium on Cryptography and Information Security, pp.236-241, 2008-01-24
バージョン：
権利関係：

SCIS 2008 The 2008 Symposium on Cryptography and Information Security
Miyazaki, Japan, Jan. 22-25, 2008
The Institute of Electronics, Information and Communication Engineers

A Framework for Evaluating Privacy Protection of Authentication Systems

Toru Nakamura Shunsuke Inenaga Daisuke Ikeda Kensuke Baba Hiroto Yasuura
Graduate School/Faculty of Information Science and Electrical Engineering, Kyushu University
Moto'oka 744, Nishi-ku, Fukuoka 819-0395, Japan
{toru,inenaga,yasuura}@c.csce.kyushu-u.ac.jp {daisuke,baba}@i.kyushu-u.ac.jp

Abstract— Authentication plays a central role in a variety of social infrastructures such as access control or e-money. Not only should authentication systems be convenient, practical and secure, but also they are often required to protect users' privacy. In this paper, we present an authentication framework for which identifiability, anonymity, linkability and unlinkability are well defined. Using our framework, we are able to evaluate and compare privacy protection properties of various authentication systems based on different core technologies. One of our main results is that authentication systems based on group signatures and one-time ID are shown to be equivalent in the sense of protecting users' privacy (more precisely they both have anonymity and unlinkability).

Keywords: authentication, anonymity, unlinkability, privacy protection

1 Introduction

Recently varieties of social information infrastructures, such as access control or e-money, and so on, have been developed due to the rapid progress of information and security technologies. In such infrastructure systems, it is critical to check whether or not a person who tries to use the service is a qualified person (a user). In addition, protecting privacy of users is often demanded for the systems. By user privacy protection we mean to protect personally identifiable information of users from someone else. Two types of user privacy protection have extensively been studied so far: privacy protection from eavesdroppers (attackers monitoring the communication) [2, 7, 4], and privacy protection from service providers [3, 8]. This paper considers the latter type of privacy protection.

There can exist some services that do not require identification of specific users. For instance, in a door access control application where permission to access the door is given to a certain group of users, it suffices for each user to show to the service provider whether or not he/she belongs to the group. In this case each user is not identified by the service provider.

Although huge amount of researches on privacy protection have been done, unfortunately it seems that there have been few criteria for comparing different authentication systems based on different core technologies. Since selection of authentication method can be critical for designing social information infrastructure, a general framework, which enables us to evaluate various kinds of authentication systems from a privacy protection point of view, is thus needed. ISO/IEC 15408 [5] introduced privacy protection properties called *anonymity*, *unlinkability* and *pseudonymity*. Pfitzmann et al. [6] also introduced those properties in different manners. However, neither of them is general enough to evaluate or compare authentication systems based on different core technologies, from a viewpoint of privacy protection.

In this paper we present *a general framework* for evaluating authentication systems in terms of user privacy protection. The advantage of our framework is that it does *not* depend on the core technologies used in the authentication systems. We firstly show an general authentication protocol and give formal definitions of *identifiability*, *anonymity*, *linkability*, and *unlinkability*. In our definitions, every authentication system with identifiability has linkability, and every authentication system with anonymity has either linkability or unlinkability. It is noteworthy that the combination of our anonymity and linkability corresponds to the pseudonymity of the literature [6, 5]. We then describe several authentication systems using our general framework, and evaluate their privacy protection properties. A main result of the paper is that an authentication system based on group signatures [3, 1] and one based on one-time ID are shown to be equivalent in terms of privacy protection (more precisely they both have anonymity and unlinkability), which we believe is non-trivial.

2 Authentication Framework

In this paper we present a new authorization model under which we can compare different systems in a uni-

form manner from the viewpoint of privacy protection.

2.1 Notion and Authentication Protocol

In this section, we firstly introduce some preliminary notion, and then we present an authentication protocol which is independent of the implementation of authentication systems.

Consider the situation where a *user* uses some *service* under permission of the service provider. Logging in a computer system and opening a door with an electronic key are examples. When a user tries to use a service, then the service provider needs to verify his/her identity or attached group in order to check if this user is eligible to use the service. We call this process *authentication*. The only assumption is that (digital) information is used for authentication. Information used for authentication is called an *claim* and is in the form of a string over Σ which is a finite alphabet.

We now introduce an *issuer* I who prepares for authentication between a service provider and its users. For example, I receives personally identifiable information from users, and creates accounts for them. In some authentication systems, a trusted third party may be issuer I. On the other hand, in authentication systems without a trusted third party, each service provider is the issuer for the service.

Let E be a countable set of *entities* including users, service providers and issuers. An entity could be a human being or a group of human beings. However, we treat an entity as an computer (a universal Turing machine) with a finite memory, and knowledge as a program (a Turing machine). Hence we say that an entity has some knowledge if it has a program which correctly outputs for a given input. For any set S of inputs, any two programs are said to be *S-equivalent* if their outputs are the same for any elements of S. To the contrary, any two programs are said to be *S-distinct* if their outputs are different for any elements of S, except for a considerably small, ignorable number of elements of S.

One of the simplest implementation of knowledge is a look-up table (a finite automaton). An authentication protocol using a look-up table is described as follows:

1. user u sends claim $h \in \Sigma^*$ to service provider s,
2. s checks if $h \in \Sigma^*$ is included in the look-up table, and
3. s provides its service provider with u if h is included in the table.

The look-up table is constructed by issuer I when service provider s is set up by I.

Now consider the situation where h contains personally identifiable information of u, from privacy protection point of view u may want to keep h secret from third parties or even service providers. In another case, eavesdroppers or malicious service providers may spoof as u by h. In so doing, and u needs to translate h into another string c from which h is not easily inferable. Encryption is an example of such a translation. The claim h that u keeps secret from s is called a *secret claim*, and the claim c translated from secret claim h is called a *proof claim*.

For any user u and service provider s, $H(u, s)$ denotes the set of secret claims of u with respect to s and $C(u, s)$ denotes the set of proof claims of u with respect to s. We may omit their arguments and simply write as H and C, when clear from the context.

Definition 1 (Authentication Protocol)

1. *user u translates secret claim h stored in its memory into proof claim c,*
2. *u sends proof claim c to service provider s,*
3. *s verifies c,*
4. *s provides its service with u if c is verified.*

The above protocol is an abstract model, which does not depend on the implementation or encryption algorithm, etc.

In the protocol of Definition 1, two programs are essential; one is the *translator* $\mathcal{T}$ which translates a secret claim to a proof claim, and the other is the *verifier* $\mathcal{V}$ which verifies if the proof claim is valid. For any entity $e \in E$, let $\mathcal{V}_e$ denote the verifier that e has as knowledge.

In our setting, issuer I prepares for authentication between a service provider s and its users. We assume that I constructs verifiers $\mathcal{V}_I$, and provide s with $\mathcal{V}_s$ that is an equivalent program to $\mathcal{V}_I$.

2.2 Privacy Protection Properties

This subsection is devoted to introduction of some properties on privacy protection. We regard privacy issues as a mapping from proof claims to entities. Therefore we first define identification of entities, and then we introduce a program which maps proof claims to entities.

We assume that a unique identifier, called an *ID*, is assigned to each entity $e \in E$[1], and we denote the set of all IDs by D. For a fixed world, we have generally several assignments each of which can identify an entity. For example, a pair of the name and address is an assignment and a social security number is another one. In this paper, we just use one of such assignments and we do not care what they are. Therefore, we assume that each entity is assigned to exactly one ID.

The topic of this paper is privacy protection from a service provider. In order for the service provider to verify a given user, $C(u, s)$ is stored in the memory of the service provider for any user u of s. If each proof claim in $C(u, s)$ is informative enough, then the ID of u may be inferable from the proof claim by the service provider. For any service provider s, let $\mathcal{I}_s$ denote a

[1] For some service providers, we do not need personally identifiable information. However, we only consider the service providers which do require such information.

program, called an *infer* of s, which maps $\bigcup C(u,s)$ to D.

Another privacy issue is that proof claim $c_1 \in C(u,s)$ may be linked to another proof claim $c_2 \in C(u,s)$ by service provider s. For any service provider s, let $\mathcal{L}_s$ denote a program, called a *link checker* of s, which checks if two given proof claims are linked in the sense that they are sent to s by the same user.

Now we are ready to define privacy protection properties in our authentication framework. In the following definitions, for any user u and service provider s, let us abbreviate $C(u,s)$ to C.

Definition 2 (Identifiability) *user u is said to have* identifiability *to service provider s if $\mathcal{I}_s$ is C-equivalent to $\mathcal{I}_I$.*

Definition 3 (Anonymity) *user u is said to have* anonymity *to service provider s if $\mathcal{I}_s$ is C-distinct to $\mathcal{I}_I$.*

Definition 4 (Linkability) *user u is said to have* linkability *to service provider s if $\mathcal{L}_s$ is $C \times C$-equivalent to $\mathcal{L}_I$.*

Definition 5 (Unlinkability) *user u is said to have* unlinkability *to service provider s if $\mathcal{L}_s$ is $C \times C$-distinct to $\mathcal{L}_I$.*

3 Evaluation of Authentication Systems in Terms of Privacy Protection

In this section, we evaluate some authentication systems from the viewpoint of privacy protection, using the framework proposed in the previous section.

In what follows, for any two strings $x, y \in \Sigma^*$, let $x\|y$ denote the concatenation of x and y. Let γ denote the empty string. Let u denote any user, s the service provider, and I the issuer.

3.1 Authentication with account name and password

Here we consider an authentication system based on an account name and a password. The system is without a trusted third party, and thus the service provider s is also the issuer I.

Preprocessing for authentication of user u:

1. user u sends ID d to service provider s.
2. s creates an account name $name \in \Sigma^*$ and password $pass \in \Sigma^*$ bound to d. s stores $name$ and $pass$ to the look-up table $Table$, such that $Table[\ell+1,1] \Leftarrow name$ and $Table[\ell+1,2] \Leftarrow pass$, where ℓ is the number of already stored account names.
3. s sends secret claim $name\|pass$ to u.

Recall Definition 1 for the authentication protocol. The program verifier $\mathcal{V}_s$ is shown in Algorithm 1. The translator $\mathcal{T}_u$ is so trivial that it outputs the input string as it is, so is omitted.

Algorithm 1 verifier $\mathcal{V}_s$ in authentication with account name and password

Input: $name\|pass \in \Sigma^*$
Output: **true** or **false**

```
for i = 1 to ℓ do
  if name = Table[i,1] && pass = Table[i,2] then
    return true
  end if
end for
return false
```

The infer $\mathcal{I}_s$ is shown in Algorithm 2.

Algorithm 2 infer $\mathcal{I}_s$ in authentication with account name and password

Input: $name\|pass \in \Sigma^*$
Output: $d \in \Sigma^*$

```
for i = 1 to ℓ do
  if name = Table[i,1] then
    return Table[i,1]
  end if
end for
return γ
```

Lemma 1 *In the authentication system with an account name and a password, u has identifiability to s.*

proof: Since $s = I$, the proposition is clear from Definition 2 and Algorithm 2. □

The link checker $\mathcal{L}_s$ of service provider s is shown in Algorithm 3.

Algorithm 3 link checker $\mathcal{L}_s$ in authentication with account name and password

Input: $name_x\|pass_x \in \Sigma^*$, $name_y\|pass_y \in \Sigma^*$
Output: **true** or **false**

```
if name_x = name_y then
  return true
end if
return false
```

Lemma 2 *In the authentication system with an account name and a password, u has linkablity to s.*

proof: Since $s = I$, the proposition is clear from Definition 4 and Algorithm 3. □

3.2 Authentication with public key encryption

In this subsection we consider an authentication system based on public key encryption. In the system issuer I is a trusted third party.

We denote by $Encrypt(x,y) = z$ that plaintext $x \in \Sigma^*$ is encrypted to ciphertext $z \in \Sigma^*$ by encryption key $y \in \Sigma^*$. Similarly, we denote by $Decrypt(z,w) = x$ that ciphertext z is decrypted to plaintext x by decryption key w. Let $Generate\_random()$ be a function that generates random string $r \in \Sigma^*$.

Preprocessing for authentication of user u:

1. user u creates a pair of secret key $SK \in \Sigma^*$ and public key $PK \in \Sigma^*$.
2. u sends ID d and PK to I.
3. issuer I creates account name $name \in \Sigma^*$ bound to d and PK. I stores $name$, PK and d to the look-up table $Table$ such that $Table[\ell+1,1] \Leftarrow name$ and $Table[\ell+1,2] \Leftarrow PK$, and $Table[\ell+1,3] \Leftarrow d$, where ℓ be the number of already stored account names.
4. I sends $name$ to u and u memorizes $name\|SK$ as secret claim w.r.t. s.
5. I constructs *Verify_table* consisting of the columns of $name$ and PK of $Table$, and I sends *Verify_table* to s.

Recall Definition 1 for the authentication protocol. The programs translator $\mathcal{T}_u$ and verifier $\mathcal{V}_s$ are shown in Algorithms 4 and 5, respectively.

Algorithm 4 translator $\mathcal{T}_u$ in authentication with public key encryption

Input: $name\|SK \in \Sigma^*$
Output: $c \in \Sigma^*$

$r \Leftarrow Generate\_random()$
$sign \Leftarrow Encrypt(r, SK)$
return $c \Leftarrow name\|r\|sign$

Algorithm 5 verifier $\mathcal{V}_s$ in authentication with public key encryption

Input: $name\|r\|sign \in \Sigma^*$
Output: **true** or **false**

for $i = 1$ to ℓ **do**
  if $name = Verify\_table[i,1]$ && $r = Decrypt(sign, Verify\_table[i,2])$ **then**
    return true
  end if
end for
return false

The infer $\mathcal{I}_I$ of issuer I is shown in Algorithm 6.

Algorithm 6 infer $\mathcal{I}_I$ in authentication with public key encryption

Input: $name\|r\|sign \in \Sigma^*$
Output: $d \in \Sigma^*$

for $i = 1$ to ℓ **do**
  if $name = Table[i,1]$ **then**
    return $Table[i,3]$
  end if
end for
return γ

Lemma 3 *In the authentication system with public key encryption, u has anonymity to s.*

proof: In Algorithm 6 issuer I uses $Table$ that contains IDs. On the other hand, service provider s only has *Verify_table* in its memory, which does not contain IDs. Thus s can only have an inferrer that is $C(u,s)$-distinct to $\mathcal{I}_I$. By Definition 3, the proposition is now established. □

The program link checker $\mathcal{L}_s$ of service provider s is shown in Algorithm 7.

Algorithm 7 link checker $\mathcal{L}_s$ in authentication with public key encryption

Input: $name_x\|r_x\|sign_x \in \Sigma^*$, $name_y\|r_y\|sign_y \in \Sigma^*$
Output: **true** or **false**

if $name_x = name_y$ **then**
  return true
end if
return false

Lemma 4 *In the authentication system with a public key encryption, u has linkability to s.*

proof: By Definition 4 and Algorithm 7. □

3.3 Authentication with group signature

In this subsection, we consider an authentication system based on group signatures [3]. In a version of group signature scheme proposed in [1], only the members of the group can sign messages by their secret keys and certificates. The receiver can verify if it is a valid signature from a group member, but can not identify which group member made it. The special person has the group secret key opening the signature, and can identify who signed the message.

We denote by $Sign(x,y,z) = w$ to sign message x by certificate y and secret key z and to obtain group signature w. We denote by $Verity(x,w,v)$ to verify with group public key v if a group signature w is made by signing message x. We denoted by $Open(w,q) = y$ to open a group signature w by group secret key q and to obtain a certificate y.

Preparation for authentication of user u:

1. issuer I creates group public key $GPK \in \Sigma^*$ and group secret key $GSK \in \Sigma^*$.
2. user u sends ID d to I.
3. I creates u's secret key $SK \in \Sigma^*$ [2] and certificate $Cert$ bound to d. I stores $Cert$ and d to the look-up table $Table$, such that $Table[\ell+1,1] \Leftarrow Cert$ and $Table[\ell+1,2] \Leftarrow d$, where ℓ is the number of already stored certificates.
4. I sends GPK to service provider s.

Recall Definition 1 for the authentication protocol. The programs translator $\mathcal{T}_u$, verifier $\mathcal{V}_s$ are shown in Algorithms 8 and 9, respectively.

[2] For simplicity we assume that I creates secret keys.

Algorithm 8 translator $\mathcal{T}_u$ in authentication with group signature

Input: $cert \| SK \in \Sigma^*$
Output: $c \in \Sigma^*$

1: $r = Generate\_random()$
2: $sign \Leftarrow Sign(r, cert, SK)$
3: **return** $r \| sign$

Algorithm 9 verifier $\mathcal{V}_s$ in authentication with group signature

Input: $r \| sign \in \Sigma^*$
Output: **true** or **false**

1: **return** $Verity(r, sign, GPK)$

The infer $\mathcal{I}_I$ of issuer I is shown in Algorithm 10.

Algorithm 10 infer $\mathcal{I}_I$ in authentication with group signature

Input: $r \| sign \in \Sigma^*$
Output: $d \in \Sigma^*$

1: $temp \Leftarrow Open(sign, GSK)$
2: **for** $i = 1$ to ℓ **do**
3: **if** $temp = Table[i, 1]$ **then**
4: **return** $Table[i, 2]$
5: **end if**
6: **end for**
7: **return** γ

Lemma 5 *In the authentication system based on group signatures, u has anonymity to s.*

proof: In Algorithm 10 issuer I uses $Table$ that contains IDs. On the other hand, service provider s only has GPK in its memory, hence s can only have an inferrer that is $C(u, s)$-distinct to $\mathcal{I}_I$. By Definition 3, the proposition is now established. □

The link checker $\mathcal{L}_I$ of issuer I is shown in Algorithm 11.

Algorithm 11 link checker $\mathcal{L}_I$ in authentication with group signature

Input: $r_x \| sign_x \in \Sigma^*, = r_y \| sign_y \in \Sigma^*$
Output: **true** or **false**

1: **if** $Open(sign_x, GSK) = Open(sign_y, GSK)$ **then**
2: **return true**
3: **end if**
4: **return false**

Lemma 6 *In the authentication system based on group signatures, u has unlinkability to s.*

proof: Under the strong RSA and the decisional Diffie-Hellman assumptions, deriving the signers secret key or certificate from the group signature or deciding two group signatures were computed by the same signer is hard without group secret key GSK [1]. Since s does not have GSK in its memory, s cannot have an binder $C(u, s) \times C(u, s)$-equivalent to $\mathcal{I}_I$. By Definition 5 the proposition is now established. □

3.4 Authentication with one-time ID

Lastly, we consider an authentication system based on one-time ID [4]. The original concept of one-time ID is a key exchange protocol with unlinkability for eavesdroppers. In this section, we propose an authentication system based on one-time ID, where users have unlinkability to service providers.

We denoted by $CreateID(x, y) = t$ to create one-time ID t by seedID x and nonce y. We assume that $CreateID$ is one-way function (e.g. secure hash function), therefore it is hard to compute x or y from t.

Preparation for authentication:

1. user u sends ID d to issuer I.
2. I creates a seedID $seed$ bound to d. I stores $seed$ and d to the look-up table $Table$ as $Table[\ell + 1, 1] \Leftarrow seed$ and $Table[\ell + 1, 2] \Leftarrow d$, where ℓ is the number of already stored seeds.
3. I stores $seed$ to table *Verify_table randomly.*
4. I sends *Verify_table* to s.

Recall Definition 1 for the authentication protocol. The programs translator $\mathcal{T}_u$ and verifier $\mathcal{V}_s$ are shown in Algorithms 12 and 13, respectively.

Algorithm 12 translator $\mathcal{T}_u$ in authentication with one-time ID

Input: $seed \in \Sigma^*$
Output: $c \in \Sigma^*$

1: $r = Generate\_random()$
2: **return** $CreateID(seed, r)$

Algorithm 13 verifier $\mathcal{V}_s$ in authentication with one-time ID

Input: $oid \in \Sigma^*$
Output: **true** or **false**

1: **for** $i = 1$ to n **do**
2: **if** $oid = Verify\_table[i]$ **then**
3: **return true**
4: **end if**
5: **end for**
6: **return false**

The infer $\mathcal{I}_I$ of issuer I is shown in Algorithm 14.

Algorithm 14 infer $\mathcal{I}_I(s)$ in authentication with one-time ID

Input: $c = oid \in \Sigma^*$
Output: $d \in \Sigma^*$

for $i = 1$ to ℓ **do**
  for each string $p \in \Sigma^*$ **do**
    if $oid = CreateID(Table[i, 1], p)$ **then**
      return $Table[i, 2]$
    end if
  end for
end for
return γ

Lemma 7 *In the authentication system based on one-time ID, u has anonymity to s.*

proof: In Algorithm 14 issuer I uses $Table$ that contains IDs. On the other hand, service provider s only has *Verify_table* in its memory, hence s can only have an inferrer that is $C(u, s)$-distinct to $\mathcal{I}_I$. By Definition 3, the proposition is now established. □

The link checker $\mathcal{L}_I$ of issuer I is shown in Algorithm 15.

Algorithm 15 link checker $\mathcal{L}_I$ in authentication with one-time ID

Input: $oid_x \in \Sigma^*$, $oid_y \in \Sigma^*$
Output: **true** or **false**

for $i = 1$ to ℓ **do**
  for each string $p \in \Sigma^*$ **do**
    if $oid_x = CreateID(Table[i, 1], p)$ **then**
      for each string $q \in \Sigma^*$ **do**
        if $oid_y = CreateID(Table[i, 1], q)$ **then**
          return true
        end if
      end for
    end if
  end for
end for
return false

Lemma 8 *In the authentication system based on one-time ID, u has unlinkability to s.*

proof: Since the elements of *Verify_table* are in random order, s can not understand the relation between *Verify_table* and $Table$. Due to the difficulty of reverse conversion of $CreateID$, it is hard for s to compute *seed* from *oid*. Hence s can only have a binder $C(u, s) \times C(u, s)$-distinct to $\mathcal{L}_I$. By Definition 5, the proposition is now established. □

As a main theorem of the paper, the following theorem follows from Lemmas 5, 6, 7 and 8.

Theorem 1 *Authentication systems based on group signature and one-time ID have the same privacy protection properties.*

4 Conclusions and Future Work

This paper presented an authentication framework and gave formal definitions of privacy protection properties of users against service providers, identifiability, anonymity, linkability, and unlinkability. We showed that our framework is general in the sense that several authentication systems based on different core technologies can be evaluated and compared in terms of privacy protection. As a main result of the paper, we showed that authentication systems based on group signature and one-time ID have the same privacy protection properties.

Our future work includes extention of our framework to authentication with multiple times of challenge and response. In the current framework, we did not consider authentication where every user does not have identifiability even to the issuer. In reality, there exists authentication where proof claims are not bound to IDs at all. We will discuss such authentication systems which do not have traceability.

References

[1] G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *Proc. CRYPTO'00*, volume 1880 of *LNCS*, pages 255–270. Springer-Verlag, 2000.

[2] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24:84–88, 1981.

[3] D. Chaum and E. van Heyst. Group signatures. In *Proc. EUROCRYPT'91*, volume 547 of *LNCS*, pages 257–270. Springer-Verlag, 1991.

[4] K. Imamoto and K. Sakurai. Authenticated key transport system using one-time id with trusted third party. In *Proc. ISEE'03*, pages 138–141, 2003.

[5] ISO. ISO/IEC 15408. http://standards.iso.org/, 2005.

[6] A. Pfitzmann and M. Köhntopp. Anonymity, unobservability and pseudonymity - a proposal for terminology. In *Proc. Designing Privacy Enhancing Technologies: International Workshop on Design Issues in Anonymity and Unobservability*, volume 2009 of *LNCS*, pages 1–9. Springer-Verlag, 2000.

[7] M. G. Reed, P. F. Syverson, and D. M. Goldschlag. Anonymous connections and onion routing. *IEEE Journal on Selected Areas in Communications*, 16(4):482–494, 1998.

[8] R. L. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In *Proc. ASIACRYPT'01*, volume 2248 of *LNCS*, pages 552–563. Springer-Verlag, 2001.