I

IBE

►Identity-Based Encryption

IC Integrated Circuit

►Trusted Boot

Iceman Attack

►Cold-Boot Attacks

IC-Integrated Circuit

►Root of Trust

ID-Based Encryption

►Identity-Based Encryption

IDEA

Alex Biryukov
FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg

Related Concepts

►Block Ciphers; ►Differential Cryptanalysis; ►IPES; ►Weak Keys

Definition

IDEA (previous name *IPES*) is a 64-bit, 8.5-round non-►Feistel block cipher with 128-bit keys, proposed by Lai and Massey in 1991 [14]. It is a modified version of a previous design called PES (Proposed Encryption Standard) by the same authors [13], with added strength against ►differential cryptanalysis.

Background

Since its publication, IDEA resisted intensive cryptanalytic efforts. In [12], p. 79], IDEA reduced to four rounds was claimed to be secure against differential attacks. Progress in cryptanalyzing round-reduced variants was very slow, starting with an attack on a two-round variant of IDEA in 1993 [15] by Meier, an improvement to 2.5 rounds by Daemen et al. [6], then an attack on 3.5 rounds published in 1997 [5] by Borst et al. ►Impossible differential attack significantly improved previous results for 3 and 3.5 rounds and could break up to 4.5 rounds [1] using the full codebook and 2^{112} steps. Further progress was an attack marginally breaking five rounds with a variant of ►meet-in-the-middle attack due to Demirci et al. [8] and using 2^{24} ►chosen plaintexts, 2^{58} memory and 2^{126} steps of analysis. More recently 6 rounds were cryptanalyzed with the full codebook, $2^{126.8}$ time and 2^{64} memory [2]. Also ►related key attacks on 8 rounds [3] were developed requiring relations between 256 keys, with 2^{72} chosen plaintexts 2^{99} time and $2^{66.6}$ memory.

In addition to these attacks, three relatively large and easily detectable classes of ►weak keys were found: in [7], 2^{51} weak keys out of the 2^{128} keys were found to be detectable with 16 chosen plaintexts and 2^{17} steps using differential *membership tests*, and in [9], 2^{63} weak keys were found to be detectable given 20 chosen plaintexts with a negligible complexity under ►differential–linear membership tests. In another work, a class of 2^{64} keys detectable via a ►boomerang membership of 2^{16} steps and queries was found [4].

Theory

The *key-schedule* of the cipher is completely linear. The main idea behind the design is the mix of non-commuting $\boxplus$: addition mod 2^{16} (denoted by $\boxplus$), XOR (denoted by $\oplus$), and multiplication mod $(2^{16} - 1)$(denoted by $\odot$, with $0 \equiv 2^{16}$). These operations work with 16-bit words.

Henk C.A. van Tilborg & Sushil Jajodia (eds.), *Encyclopedia of Cryptography and Security*, DOI 10.1007/978-1-4419-5906-5,

One round of IDEA is split into two different half-round operations: key mixing (denoted by T) and M-mixing denoted by $M = s^{\circ}MA$, where MA denotes a multiplication–addition structure and s denotes a swap of two middle words (as usual the composition of transformations is applied from right to left, i.e., MA is applied first, and the swap s is applied to the result). T divides the 64-bit block into four 16-bit words X_1,X_2,X_3,X_4 and mixes the key words Z_1,Z_2,Z_3,Z_4 with the data using $\odot$ and $\boxplus$:

$$(X_1, X_2, X_3, X_4) \xrightarrow{T} (X_1 \odot Z_1, X_2 \boxplus Z_2, X_3 \boxplus Z_3, X_4 \odot Z_4)$$

The transform MA provides *diffusion* between different words and mixes in two more key words Z_5,Z_6:

$$Y_1 = \left((X_1 \oplus X_3) \odot Z_5 \boxplus (X_2 \oplus X_4)\right) \odot Z_6,$$
$$Y_2 = Y_1 \left((X_1 \oplus X_3) \odot Z_5\right),$$
$$(X_1, X_2, X_3, X_4) \xrightarrow{MA} (X_1 \oplus C_2, X_2 \oplus C_1, X_3 \oplus C_2, X_4 \oplus C_1).$$

Both MA and s are involutions. The full 8.5-round IDEA can be written as

$$\text{IDEA} = T^{\circ}s^{\circ}(s^{\circ}MA^{\circ}T)^8 = T^{\circ}s^{\circ}(M^{\circ}T)^8.$$

The only changes between IDEA and its predecessor PES are in the order of operations in the key mixing subround T: PES uses the order $(\odot,\odot,\boxplus,\boxplus)$, while IDEA uses the order $(\odot,\boxplus,\boxplus,\odot)$, and in the swap of the words after the MA subround. In IDEA, the outer words X_1, X_4 are not swapped. These changes were motivated by a differential attack on PES given in [14].

Recommended Reading

1. Biham E, Biryukov A, Shamir A (1999) Miss in the middle attacks on IDEA and Khufu. In: Knudsen LR (ed) Fast software encryption, FSE'99. Lecture notes in computer science, vol 1636. Springer, Berlin, pp 124–138
2. Biham E, Dunkelman O, Keller N (2007) A new attack on 6-round IDEA. FSE 2007, Luxembourg, pp 211–224
3. Biham E, Dunkelman O, Keller N (2008) A unified approach to related-key attacks. FSE 2008, Lausanne, pp 73–96
4. Biryukov AJN Jr, Preneel B, Vandewalle J (2002) New weak-key classes of IDEA. In: Deng RH, Qing S, Bao F, Zhou J (eds) International conference on information and communications security, ICICS 2002. Lecture notes in computer science, vol 2513. Springer, Berlin, pp 315–326
5. Borst J, Knudsen LR, Rijmen V (1997) Two attacks on reduced IDEA (extended abstract). In: Fumy W (ed) Advances in cryptology—eurocrypt'97. Lecture notes in computer science, vol 1233. Springer, Berlin, pp 1–13
6. Daemen J, Govaerts R, Vandewalle J (1993) Cryptanalysis of 2.5 rounds of IDEA (extended abstract). Technical report 93/1, Department of electrical engineering, ESAT-COSIC
7. Daemen J, Govaerts R, Vandewalle J (1994) Weak keys for IDEA. In: Stinson DR (ed) Advances in cryptology—CRYPTO'93. Lecture notes in computer science, vol 773. Springe, Berlin, pp 224–231
8. Demirci, H, Selccuk A, Türe E (2004) A new meet-in-the-middle attack on the IDEA block cipher. In: Matsui M, Zuccherato R (ed) Selected areas in cryptography, SAC 2003. Lecture notes in computer science, vol 3006. Springer, Berlin
9. Hawkes P (1998) Differential–linear weak key classes of IDEA. In: Nyberg K (ed) Advances in cryptology—EUROCRYPT'98. Lecture notes in computer science, vol 1403. Springer, Berlin, pp 112–126
10. Hawkes P, O'Connor L (1996) On applying linear cryptanalysis to IDEA. In: Kim K, Matsumoto T (eds) Advances in cryptography—ASIACRYPT'96. Lecture notes in computer science, vol 1163. Springer, Berlin, pp 105–115
11. Kelsey J, Schneier B, Wagner D (1996) Key-schedule cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES. In: Koblitz N (ed) Advances in cryptology—CRYPTO'96. Lecture notes in computer science, vol 1109. Springer, Berlin, pp 237–251
12. Lai X (1992) On the design and security of block ciphers. Doctoral dissertation, Swiss Federal Institute of Technology, Zurich
13. Lai X, Massey JL (1990) A proposal for a new block encryption standard. In: Damgard IB (ed) Advances in cryptology—EUROCRYPT'90. Lecture notes in computer science, vol 473. Springer, Berlin, pp 389–404
14. Lai X, Massey JL, Murphy S (1991) Markov ciphers and differential cryptanalysis. In: Davies DW (ed) Advances in cryptology—EUROCRYPT'91. Lecture notes in computer science, vol 547. Springer, Berlin, pp 17–38
15. Meier W (1993) On the security of the IDEA block cipher. In: Helleseth T (ed) Advances in cryptology—EUROCRYPT'93. Lecture notes in computer science, vol 765. Springer, Berlin, pp 371–385

Identification

Carlisle Adams
School of Information Technology and Engineering (SITE), University of Ottawa, Ottawa, Ontario, Canada

Related Concepts

►Authentication; ►Certificate; ►Certification Authority; ►Entity Authentication; ►Identity Verification Protocol; ►Impersonation Attack; ►Public Key Infrastructure

Definition

Identification is the claim by an entity of some particular identity. The claimant interacts with a verifier, typically offering corroborating evidence for the claim, so that the verifier will come to rely upon the claim for some purpose.

Theory

A "name," or identity, is a set of information that distinguishes a specific entity from every other within a particular environment. In some environments, the "name" may just be a given name; in other environments, it will

be a given name and a family name; in still others, it may include additional data such as a street address, or may be some other form entirely (e.g., an employee number). In all cases, however, the identity depends upon the environment: the size and characteristics of the environment determine the amount of information required for uniqueness [1].

Identification is the claim of an identity. Each of two entities is involved in this process: the *claimant* claims an identity either explicitly or implicitly ("I am *x*"), and the *verifier* makes a corresponding claim with respect to the same identity ("The entity with whom I am dealing is *X*," where *X* is either *x* or a mapping from *x* to some other namespace that is meaningful to the verifier). In order for the verifier to believe its own claim enough to rely upon it for some purpose, there must be corroborating evidence of the claimant's claim. This evidence may come from the claimant directly or may come from some third party that is trusted by the verifier. In either case, the process of obtaining and verifying this evidence is known as ►authentication (also ►Entity Authentication) and may make use of a protocol exchange between the verifier and the claimant (►Identity Verification Protocol).

Depending upon the purpose for which the verifier needs the identity of the claimant (i.e., depending upon how much the verifier must rely upon this identity), the authentication process associated with an identification step may be relatively weak, relatively strong, or somewhere in between. The verifier needs to assess and manage the risk that this identity may have been stolen by another entity who is now trying to impersonate the true holder of this identity (►Impersonation Attack). Impersonation can potentially lead to unauthorized access to personal or corporate data, networks, applications, or functions; the verifier will typically use stronger authentication mechanisms if a successful impersonation attack leads to the release of sensitive data.

Identity Uniqueness

Identification is only possible within a domain when all identities in that domain are unique (i.e., no two entities in the domain have the same "name"). There are two general schemes for achieving identity uniqueness within a domain: *hierarchical namespaces* and *flat namespaces*. In a flat namespace, the Naming Authority (the authority that officially assigns identities to, or associates identities with, entities) is responsible for binding a unique identity to every entity in the domain and uses a fixed, nonextensible syntax to express this identity. Within a company, an *n*-digit employee number is an example of a flat namespace scheme.

In a hierarchical namespace, the Naming Authority is responsible for binding a unique identity to only a subset of the entities; these entities in turn are responsible for binding unique identities to other groups of entities, and so on, until every entity has a unique identity. The identity syntax is typically flexible and extensible. Examples of hierarchical namespace schemes include e-mail address, IP address, and X.500 distinguished name [2].

In general, there is a trade-off between uniqueness and usability (user-friendliness) in a flat namespace. Furthermore, for many large domains – in particular, the domain of the entire world – there is no single recognized naming authority. Consequently, hierarchical schemes are typically used in practice in large-scale environments.

Authorities for Naming and Authentication

Although they may in theory be the same, typically the Naming Authority and the Authentication Authority in a domain are distinct entities. The Naming Authority associates a "name" with an entity for the purposes of identification, while the Authentication Authority associates that same "name" with corroborating evidence for the purposes of authentication, and/or with an authentication mechanism for the purposes of identity verification. A ►Certification Authority in a ►Public Key Infrastructure(PKI) is an example of the latter type of Authentication Authority in that it binds a "name" (such as an e-mail address) to a public key pair in a ►certificate that will be used as an authentication mechanism by the entity associated with that "name" [1, 3]. A government department issuing driver's licenses is an example of the former type of Authentication Authority in that it writes a "name" (in this case, a given and family name, along with address and other information) into an official document (the driver's license) that may be used as corroborating evidence to validate a claim of identity.

Recommended Reading

1. Adams C, Lloyd S (2003) Understanding PKI: concepts, standards, and deployment considerations, 2nd edn. Addison-Wesley, Reading, MA
2. ITU-T Recommendation X.509 (2000) Information technology – open systems interconnection – the directory: public key and attribute certificate frameworks. (equivalent to ISO/IEC 9594–8:2001)
3. Housley R, Polk T (2001) Planning for PKI: best practices guide for deploying public key infrastructure. Wiley, New York

Identity Authentication

►Entity Authentication

Identity Management

Mike Just
Glasgow Caledonian University, Scotland, UK

Related Concepts

►Access Control from an OS Security Perspective; ►Authentication; ►Authorizations; ►Identification

Definition

An *identity* of an individual is the set of information known about that person. For example, a person's identity in the real world can be a set of a name, an address, a driver's license, a birth certificate, a field of employment, etc. This set of information includes items such as a name which is used as an identifier – it allows us to refer to the identity without enumerating all of the items; a driver's license or birth certificate which are used as an authenticator – they are issued by a relevant authority and allow us to determine the legitimacy of someone's claim to the identity; a driver's license which is used as a privilege – it establishes the permission to operate a motor vehicle.

A *digital identity* is the corresponding concept in the digital world. As people engage in more activities in the cyber world, the trend has been to link the real world attributes of identity with an individual's cyber world identity, giving rise to privacy and other concerns.

Identity management is the set of processes, tools, and social contracts surrounding the creation, maintenance, and termination of a digital identity for people or, more generally, for systems and services, to enable secure access to an expanding set of systems and applications.

Background

Traditionally, identity management has been a core component of system security environments where it has been used for the maintenance of account information for login access to a system or a limited set of applications. An administrator issues accounts so that resource access can be restricted and monitored. Control has been the primary focus for identity management. More recently, however, identity management has exploded out of the sole purview of information security professionals and has become a key enabler for electronic business.

Theory

Identity management solutions are modular and composed of multiple service and system components. Below, a number of key foundational, lifecycle, and consumable components are identified and described.

Foundational components of an identity management system are core components that form that basic building blocks on which an identity management system can be built. Such components are often not specific to identity management, but can be used in other security and system areas.

- Repository. At the core of the system is the logical data storage facility and identity data model that is often implemented as a directory or database. Policy information governing access to and use of information in the repository is generally stored here as well.
- Authentication Provider. The authentication provider, sometimes referred to as the identity provider, is responsible for performing primary authentication of an individual that will link them to a given identity. The authentication provider produces an authenticator – a token that allows other components to recognize that primary authentication has been performed. Each identity may be associated with more than one authentication provider. The mechanisms employed by each provider may be of different strengths and some application contexts may require a minimum strength to accept the claim to a given identity.
- Policy Control. Access to and use of identity information is governed by policy controls. Authorization policies determine how information is manipulated; privacy policies govern how identity information may be disclosed. Policy controls may cause events to be audited or even for the subject of an identity to be notified when information is accessed.
- Auditing. Secure auditing provides the mechanism to track how information in the repository is created, modified, and used. This is an essential enabler for forensic analysis – which is used to determine how and by whom policy controls were circumvented.

Lifecycle components of an identity management system address the temporal nature of managing an identity and relate to key milestones such as the start and end of an identity, as well as any changes during these periods.

- Provisioning. Provisioning is the automation of all the procedures and tools to manage the lifecycle of an identity: creation of the identifier for the identity, linkage to the authentication providers, setting and changing attributes and privileges, and decommissioning the identity. In large-scale systems, these tools generally allow some form of self-service for the creation and ongoing maintenance of an identity and frequently use a workflow or transactional system for verification of data from an appropriate authority and to propagate

data to affiliated systems that may not directly consume the repository.
- Longevity. Longevity tools create the historical record of an identity. These tools allow the examination of the evolution of an identity over time.

Consumable value components of an identity management system effectively allow for integration of the core and lifecycle components with applications that need to make use of identity management.

- Single Sign-on. Single sign-on allows a user to perform primary authentication once and then access the set of applications and systems that are part of the identity management environment.
- Personalization. Personalization and preference management tools allow application specific as well as generic information to be associated with an identity. These tools allow applications to tailor the user experience for a given individual leading to a streamlined interface for the user and the ability to target information dissemination for a business.
- Access Management. Similar to the policy controls within the identity management system foundation components, access control components allow applications to make authorization and other policy decisions based on privilege and policy information stored in the repository.

Applications

Identity management systems are primarily deployed in one of three models: as silos, as walled gardens, and as federations.

Silo. This is the predominant model on the Internet today. In this model the identity management environment is put in place and operated by a single entity for a fixed user community.

Walled Garden. Walled gardens represent a closed community of organizations. A single identity management system is deployed to serve the common user community of a collection of businesses. Most frequently this occurs in business-to-business exchanges and specific operating rules govern the entity operating the identity management system.

Federation. Federated identity management environments are now emerging. These include systems like OpenID, Windows Live ID (formerly Microsoft's .Net Passport), and the Liberty Alliance Project: Liberty Architecture. The central difference between federated identity systems and walled gardens is that there is no single entity that operates the identity management system. Federated systems support multiple identity providers and a distributed and partitioned store for identity information. Clear operating rules govern the various participants in a federation – both the operators of components and the operators of services who rely on the information provided by the identity management system. Most systems exhibit strong end-user controls over how identity information is disseminated amongst members of the federation.

Experimental Results

At time of writing this chapter, several organizations have implemented, and demonstrated interoperability, with the SAML standard (as part of the Liberty alliance) as part of their identity management products. In addition, several websites have adopted the OpenID standard.

Open Problems and Future Directions

Many of the challenges related to identity management relate to uptake and adoption by both service providers and users since the effectiveness for identity management (especially for components such as single sign-on) increase with adoption and use.

Recommended Reading

1. OpenID Foundation (2007) OpenID foundation. OpenID authentication 2.0 – final. http://openid.net/specs/openid-authentication-2_0.html. Accessed 1 Dec 2010
2. Security Services Technical Committee (SSTC) Security Assurance Markup Language (SAML) v 2.0. OASIS, (2005) http://saml.xml.org/saml-specifications. Accessed 1 Dec 2010

Identity Proof

►Authentication

Identity Verification Protocol

Robert Zuccherato
9 Carle Crescent, Ontario, Canada

Synonyms

Entity authentication protocol

Related Concepts

►Authentication; ►Authentication Token; ►Challenge-Response Protocol; ►Dictionary Attack; ►Dictionary

Attack (I); ►Entity Authentication; ►Password; ►Protocol; ►Schnorr Identification Protocol; ►Fiat–Shamir Identification Protocol and the Feige–Fiat–Shamir Signature Scheme; ►Trusted Third Party; ►Zero-Knowledge

Definition

An identity verification protocol is a protocol used to obtain entity authentication of one entity to another entity. The authentication provided by the protocol can be either *unilateral* (i.e., authenticates just one of the entities to the other entity) or *mutual* (i.e., authenticates both entities).

Theory

In addition to the two entities directly involved in the authentication (the *claimant* and *verifier*), some protocols require the participation of a trusted third party in order to achieve authentication. For example, an authentication server may provide an authentication token to a claimant, which would then be provided to the verifier as part of the identity verification protocol.

There are a number of different identity verification protocols that exist, but most of them fall into three main types. These are password-based schemes, challenge–response protocol schemes, and zero-knowledge identification techniques.

Applications

Password-based schemes typically involve the claimant providing a password or PIN (Personal Identification Number) that is either sent directly to the verifier or used to generate a token (►Authentication Token), or credentials, that can be validated by the verifier, who also knows the password. Most of these schemes are susceptible to replay attacks, password guessing attacks, dictionary attacks, and/or compromise of the password, at either the claimant or verifier. Thus, these schemes usually provide limited security, but have the advantage that they are easy to implement and deploy.

Challenge–response protocols require the claimant to verify its identity to the verifier by proving knowledge of a secret value that is known only to the claimant (and possibly the verifier) and will not be revealed as part of the protocol. Proving knowledge of this secret is accomplished by responding to a particular challenge provided by the verifier. Typically, the verifier produces a time-variant parameter (e.g., a nonce), and the claimant is required to apply some cryptographic transformation to that parameter using a key that only it (or possibly also the verifier) knows. Examples of challenge–response identity verification protocols can be found in ISO/IEC 9798-2 [1], ISO/IEC 9798–3 [2], FIPS 196 [3] and TLS (►Transport Layer Security) [4].

Zero-knowledge protocols use asymmetric techniques to prove the claimant's identity, but are based upon interactive proof systems and zero-knowledge techniques, and thus differ from asymmetric-based challenge–response protocols. These protocols are designed to reveal no information whatsoever beyond whether or not the claimant knows a secret (and thus has the claimed identity) and then only to the verifier. Examples of zero-knowledge identification protocols include the Fiat–Shamir identification protocol, the GQ identification protocol [5], and the Schnorr identification protocol.

It should be noted that identity verification protocols only provide entity authentication at the instant of protocol execution and do not necessarily prove that the authenticated entity was participating in an entire session. If entity authentication is required for the entire lifetime of a given session, then either the identity verification protocol can be repeatedly performed throughout the lifetime of the session, or it can be combined with a key establishment mechanism and an ongoing integrity service, as is done in TLS.

Recommended Reading

1. ISO/IEC 9798-2 (1999) Information technology-security techniques-entity authentication-Part 2: mechanisms using symmetric encipherment algorithms
2. ISO/IEC 9798-3 (1998) Information technology-security techniques-entity authentication-Part 3: entity authentication using digital signature techniques
3. FIPS 196 (1997) Entity authentication using public key cryptography. Federal Information Processing Standards Publication 196, U.S. Department of Commerce/NIST, National Technical Information Service, Springfield, Virginia
4. Dierks T, Rescorla E (2008) The Transport Layer Security (TLS) Protocol – Version 1.2. RFC 5246
5. Guillou LC, J.-J. Quisquater (1988) A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory. In: Günther CG (ed) Advances in Cryptology – EUROCRYPT'88, pp 123–128. Lect Notes in Computer Science, vol 330. Springer, Berlin

Identity-Based Cryptosystems

Benoît Libert, Jean-Jacques Quisquater
Microelectronics Laboratory, Université catholique de Louvain, Louvain-la-Neuve, Belgium

Synonyms

IBE: Identity-based encryption; IBS: Identity-based signature

Related Concepts

►Digital Signatures; ►Identification Schemes; ►Public Key Cryptography

Definition

Identity-based cryptography refers to a set of public key cryptographic primitives where public keys consist of users' identity information and nothing else. Its advantage is to alleviate the need for digital certificates linking public keys to the identity of the corresponding user.

Background

Identity-based public key cryptography is a paradigm introduced by Shamir in 1984 [36]. His motivation was to simplify key management and remove the need for public key certificates as much as possible by letting the user's public key be the binary sequence corresponding to an information identifying him in a nonambiguous way (e-mail address, IP address combined to a user name, telephone number, etc). The removal of certificates allows avoiding the trust problems encountered in current public key infrastructures (PKIs): it is no longer necessary to bind a public key to its owner's name since those are one single thing, and it also simplifies key management since public keys are human-memorizable. These systems involve trusted authorities called private key generators (PKGs) that have to deliver private keys to users after having derived them from their identity information (users do not generate their key pairs themselves) using a master secret key. End users do not have to enquire for a certificate for their public key. The only things that still must be certified are the public keys of trusted authorities. This does not completely eliminate the need for certificates but, since many users depend on the same authority, this need is drastically reduced. Several practical solutions for identity-based signatures (IBS) have been devised since 1984 [15, 23, 34], but finding a practical identity-based encryption scheme (IBE) remained an open challenge until 2001 when elegant solutions were provided by Boneh and Franklin [8] and Cocks [13]. Other identity-based signatures were proposed after 2001 (e.g., [10, 26]).

Basically, an identity-based cryptosystem consists of four algorithms. First, a *Setup* algorithm, which is run by a PKG, takes as input a security parameter to output a public/private master key pair $(\mathsf{mpk}, \mathsf{msk})$ for the PKG. A key generation algorithm *Keygen* is also run by the PKG: it takes as input the PKG's master secret key msk and a user's identity *ID* to return the user's private key d_{ID}. In the case of identity-based encryption, the third algorithm is an encryption algorithm *Encrypt* that can be publicly run by anyone and takes as input a plaintext M, the recipient's identity, and the PKG's master public key mpk to output a ciphertext C. The last algorithm is then the decryption algorithm *Decrypt* that takes as input the ciphertext C and the private decryption key d_{ID} to return a plaintext M. In the case of identity-based signatures, the last two algorithms are the signature generation algorithm *Sign* that, given a message M, the PKG's public key and a private key d_{ID} generates a signature on M that can be verified by anyone thanks to the signature verification algorithm *Verify*. The latter takes as input the PKG's key mpk and the alleged signer's identity *ID* to return 1 or 0 depending on whether the signature is acceptable or not.

This chapter surveys some simple identity-based schemes that have appeared in the literature since Shamir's call for proposals in 1984. Some of the most famous identity-based signature schemes are fist described, and the chapter then gives an example of identity-based encryption scheme based on modular arithmetic.

Theory

This section presents simple examples of identity-based signatures. The first one is a generic construction that can be based on any signature scheme. The second one is the Guillou–Quisquater [23] signature scheme that builds on the RSA assumption. The next example is a scheme, proposed by Bellare, Namprembre, and Neven [2], which relies on the difficulty of computing discrete logarithms. The chapter finally outlines a simple identity-based encryption scheme due to Cocks [13].

Identity-Based Signatures

Syntactically, an identity-based signature consists of the following four algorithms.

Setup: is a probabilistic algorithm run by a private key generator (PKG) that takes as input a security parameter to output a master public key mpk and a master secret key msk which is kept secret.

Keygen: is a private key generation algorithm run by the PKG on input of params and the master key msk to return a private key d_{ID} associated with the identity *ID*.

Sign: is a (possibly probabilistic) algorithm that takes as input public parameters params, a message *M* and the signer's private key d_{ID}, and outputs a signature $\sigma = \mathbf{Sign}(d_{ID}, M)$.

Verify: is a deterministic verification algorithm that takes as input a purported signature σ, the master public key mpk and the signer's identity *ID*. It outputs 0 or 1.

The security of IBS schemes is formalized via a game between a challenger and an adversary $\mathcal{F}$. More precisely, the definition used in [2, 14] extends the standard notion

[22] of existential unforgeability under chosen-message attacks by requiring any probabilistic polynomial-time algorithm $\mathcal{F}$ to have negligible advantage in the following game.

1. The challenger generates a master key pair $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathbf{Setup}(k)$ and hands mpk to the forger $\mathcal{F}$.
2. On polynomially-many occasions, the forger makes queries of the following two types:
   (a) Key generation queries: $\mathcal{F}$ chooses an arbitrary identity ID and the challenger replies by returning $d_{ID} \leftarrow \mathbf{Keygen}(\mathsf{msk}, ID)$.
   (b) Signing queries: $\mathcal{F}$ chooses a pair (ID, M). The challenger replies by computing $d_{ID} \leftarrow \mathbf{Keygen}(\mathsf{msk}, ID)$ and returning $\sigma = \mathbf{Sign}(d_{ID}, M)$.
   These queries can be made adaptively in that each one may depend on the answers to prior queries.
3. The forger outputs a triple $(ID^\star, M^\star, \sigma^\star)$ and wins if the three following conditions are satisfied:
   (a) $ID^\star$ was never queried for key generation.
   (b) The pair $(ID^\star, M^\star)$ was never queried for signature.
   (c) $\mathbf{Verify}(\mathsf{mpk}, ID^\star, \sigma^\star) = 1$.

Generic IBS from Any Signature

In [2], Bellare, Namprempre, and Neven pointed out that any digital signature can be made identity-based in a very simple manner using certification. This generic construction was previously implicitly described in [14] and goes as follows. Let $\Pi^{\mathsf{sig}} = (\mathsf{Keygen}, \mathsf{Sign}, \mathsf{Verify})$ be any ordinary (i.e., nonidentity-based) digital signature scheme providing existential unforgeability under chosen-message attacks [22]. Then, a secure IBS scheme $\Pi^{\mathsf{IBS}} = (\mathbf{Setup}, \mathbf{Keygen}, \mathbf{Sign}, \mathbf{Verify})$ can be obtained as follows.

Setup: Given a security parameter $k \in \mathbb{N}$, the algorithm generates a digital signature key pair $(pk, sk) \leftarrow \mathsf{Keygen}(k)$ and returns $(\mathsf{mpk}, \mathsf{msk}) = (pk, sk)$.

Keygen: To generate a private key for some user's identity ID, the PKG generates a fresh digital signature key pair $(pk_{ID}, sk_{ID}) \leftarrow \mathsf{Keygen}(k)$ and sets $d_{ID} = (sk_{ID}, pk_{ID}, \sigma_{ID})$, where $\sigma_{ID} = \mathsf{Sign}(\mathsf{msk}, ID \| pk_{ID})$ is a certificate binding the newly generated public key pk_{ID} to the identity ID.

Sign: To sign a message m using is private key $d_{ID} = (sk_{ID}, pk_{ID}, \sigma_{ID})$, the signer computes $\sigma_m = \mathsf{Sign}(sk_{ID}, m)$ and the identity-based signature is defined as the triple $\sigma = (\sigma_m, pk_{ID}, \sigma_{ID})$.

Verify: To verify an alleged signature $\sigma = (\sigma_m, pk_{ID}, \sigma_{ID})$ for message m under the identity ID, the verifier returns 1 if $\mathsf{Verify}(pk_{ID}, \sigma_m, m) = 1$ and $\mathsf{Verify}(\mathsf{mpk}, \sigma_{ID}, ID \| pk_{ID}) = 1$. Otherwise, it outputs 0.

The above construction was extended to provide hierarchical identity-based signatures [25] (i.e., IBS schemes involving a hierarchy of signers organized in a hierarchical setting) and identity-based signatures with special properties [18].

While simple and elegant, this construction leaves room for efficiency improvements, notably in terms of signature size since each signature comprises two ordinary digital signatures and a public key. Ideally, one would like to have an IBS scheme performing as well as ordinary digital signatures. The next subsections show two examples of such schemes that both derive from the Fiat–Shamir paradigm [15].

The Guillou–Quisquater IBS

This scheme is derived from a three round identification scheme. It was proposed in 1988 and consists of the following algorithms.

Setup: Given a security parameter k_0, the private key generator (PKG) picks two $k_0/2$-bit primes p and q and computes $n = pq$. It also picks a prime number $e \in \mathbb{Z}_{\varphi(n)}$ such that $\gcd(e, \varphi(n)) = 1$ and chooses cryptographic hash functions $H : \{0,1\}^* \to \mathbb{Z}_e$ and $G : \{0,1\}^* \to \mathbb{Z}_n$. The master public key is $\mathsf{mpk} = (n, e, G, H)$ while the master secret key is the pair $\mathsf{msk} = (p, q)$.

Keygen: Given a user's identity ID, the PKG computes $I = G(ID) \in \mathbb{Z}_n^*$ and $a \in \mathbb{Z}_n^*$ such that $Ia^e \equiv 1 \pmod{n}$. The obtained $d_{ID} = a$ is returned to the user as a private key.

Sign: Given a message m, the signer does the following:

1. Pick a random $k \xleftarrow{R} \mathbb{Z}_n^*$ and compute $r = k^e \bmod n$.
2. Compute $\ell = H(m \| r) \in \mathbb{Z}_e$.
3. Calculate $s = ka^\ell \bmod n$.

The signature on m is the pair (s, ℓ).

Verify: To verify a signature (s, ℓ) on m:

1. Compute $I = G(ID)$ from the signer's identity ID.
2. Compute $u = s^e I^\ell \bmod n$.
3. Accept the signature if $\ell = H(m \| u)$.

To verify the consistency of the scheme, note that

$$u \equiv s^e I^\ell \equiv (ka^\ell)^e I^\ell \equiv k^e (a^e I)^\ell \equiv k^e \equiv r \pmod{n}.$$

Hence $u = r$ and then $H(m \| u) = H(m \| r)$.

For security reasons, the parameter k_0 should be at least 1024 or 2048 to avoid attacks trying to factor the modulus.

This signature scheme is derived from the Guillou–Quisquater identification protocol (GQ) using the Fiat–Shamir heuristic [15] that turns any 3-move identification scheme into a digital signature. In the scheme, the signer's private key is an RSA signature generated by the PKG

on a message consisting of the user's identity: in other words, a GQ signature is actually a noninteractive proof of knowledge of an RSA signature.

The scheme can be proved existentially unforgeable (in the random oracle model [6]) provided it is hard to invert the RSA function. The first security proof can be traced back to the work of Pointcheval and Stern [32, 33] who showed how their "forking technique" yields security proofs for signature schemes derived from identification protocols. Their security proof was given in a model where the scheme was treated as an ordinary (i.e., nonidentity-based) signature. A security proof in the model of section "Identity-Based signatures" was provided by the general framework of Bellare et al. [2].

While the infeasibility of inverting the RSA function is sufficient to prove the security of the GQ signature scheme in the random oracle model, a stronger interactive assumption (introduced in [3]) is necessary to prove the security of the underlying interactive identification scheme. The security of the GQ identification scheme against active and concurrent attacks was established by Bellare and Palacio [5] in a traditional public key setting. Its security in the identity-based model was proved in [2].

The Bellare–Namprempre–Neven IBS

In 2004, Bellare et al. [2] described an identity-based signature based on the discrete logarithm problem. In the same way as the Guillou–Quisquater scheme can be seen as a proof of knowledge of an RSA signature, the Bellare et al. scheme can be viewed as a noninteractive proof of knowledge of a Schnorr [35] signature.

The Schnorr signature scheme makes use of a cyclic group $\mathbb{G}$ of prime order p. The signer holds a public key $X = g^x$, where $x \xleftarrow{R} \mathbb{Z}_p$ is the private key which is used to sign a message m as follows. The signer first computes $R = g^r$, for a randomly chosen $r \xleftarrow{R} \mathbb{Z}_p$, and sets $s = r + H(m||R)x \bmod p$, where $H : \{0,1\}^* \to \mathbb{Z}_p$ is a hash function modeled as a random oracle. The signature consists of (ℓ, s), where $\ell = H(m||R)$, and is verified by checking whether $\ell = H(m||g^s X^{-\ell})$. Alternatively, the signature can be (R, s) in such a way that the verifier can recompute $\ell = H(m||R)$ before checking whether $R = g^s X^{-\ell}$.

Setup: Given a security parameter k_0, the PKG chooses a cyclic group $\mathbb{G}$ of prime order $p > 2^{k_0}$ and a generator $g \in \mathbb{G}$. It picks $x \xleftarrow{R} \mathbb{Z}_p$ and sets $X = g^x$. It also chooses cryptographic hash functions $H : \{0,1\}^* \to \mathbb{Z}_p$ and $G : \{0,1\}^* \to \mathbb{Z}_p$. The master public key is $\mathsf{mpk} = (g, X, H, G)$ while the master secret key is $\mathsf{msk} = x$.

Keygen: Given a user's identity ID, the PKG chooses $r \xleftarrow{R} \mathbb{Z}_p$ at random and computes $R = g^r$ as well as $\ell = H(ID||R) \in \mathbb{Z}_p$. The private key is the pair $d_{ID} = (R, s)$ where $s = r + \ell x \bmod p$.

Sign: To sign a message m using $d_{ID} = (R, s)$, the signer proceeds as follows:

1. Pick a random $y \xleftarrow{R} \mathbb{Z}_p$ and compute $Y = g^y$ as well as $S = g^s$.
2. Compute $c = G(m\|S\|Y) \in \mathbb{Z}_p$.
3. Calculate $z = y + cs \bmod p$.

The signature on m is (R, S, c, z).

Verify: To verify a signature (R, S, c, z) on m:

1. Compute $Y = g^z S^{-c}$ and reject the signature if $c \neq G(m||S||Y)$.
2. Compute $\ell = H(ID||R)$.
3. Accept the signature if $S = RX^{\ell}$.

Bellare et al. [2] proved the security of their scheme assuming that computing discrete logarithms is hard and when the hash functions H and G are modeled as random oracles.

A BNN signature can be seen as a noninteractive proof of knowledge of a Schnorr signature in the same way as GQ signatures prove knowledge of an RSA signature. The above scheme can also be seen as an optimization of the generic construction described in section "Generic IBS from Any Signature" it indeed provides slightly shorter signatures using suitable parameters such as carefully chosen elliptic-curve subgroups. The BNN IBS was recently further optimized [19] to provide shorter signatures and also inspired an IBS scheme [24] supporting partial signature aggregation.

Other IBS Schemes Based on Specific Number Theoretic Assumptions

The GQ and BNN identity-based signatures are far from being the only IBS systems where signatures consist of a noninteractive proof of knowledge of an ordinary signature. Indeed, many other proposals based on the discrete logarithm problem [7, 19, 28], RSA [28], factoring [15, 17, 29], or groups with bilinear maps [1, 10, 24, 26, 30] can be found in the literature (see [2] for a comprehensive survey of these). Recent works [11, 12] also investigated how to efficiently base IBS schemes on assumptions stemming from coding theory.

It has been reported that identity-based signatures can be endowed with specific additional properties. In many cases, these can be generically obtained [18] by extending the generic construction of section "Generic IBS from Any

Signature". Other IBS schemes supporting signature aggregation [20, 24] or multiple signers [4] were designed under specific assumptions.

In addition, an observation made by Naor (and reported in [8]) implies that identity-based signatures can also be derived from hierarchical identity-based encryption (HIBE) [21, 27]. Following this observation, Paterson and Schuldt [31] proved the security (without using the random oracle model) of a scheme derived from a 2-level extension of the Waters IBE [37] under the Diffie–Hellman assumption in groups with a bilinear map.

Identity-Based Encryption from Quadratic Residuosity: The Cocks IBE

An identity-based encryption scheme (IBE) consists of a tuple of algorithms $(Setup, Keygen, Encrypt, Decrypt)$, the first two ones of which have the same functionalities as in identity-based signatures. Algorithm *Encrypt* takes as input a plaintext m, the master public key mpk, and a receiver's identity ID to output a ciphertext $C = Encrypt(m, \mathsf{mpk}, ID)$. On input of C and the private key d_{ID}, the corresponding decryption algorithm outputs either a plaintext m or a special symbol $\perp$ indicating that the ciphertext is invalid.

The appropriate definition of security for IBE schemes was given by Boneh and Franklin [8].

Definition 1 *An IBE scheme is said to be adaptively chosen-ciphertext secure (IND-ID-CCA) if no probabilistic polynomial time (PPT) adversary has a non-negligible advantage in the following game.*

1. *The challenger runs the* **Setup** *algorithm on input of a security parameter k and sends the domain-wide parameters* mpk *to the adversary* $\mathcal{A}$.
2. *In a find stage,* $\mathcal{A}$ *starts probing the following oracles:*
   - *Key extraction oracle: Given an identity ID, this oracle returns the extracted private key* $d_{ID} = \mathbf{Keygen}(\mathsf{msk}, ID)$.
   - *Decryption oracle: Given an identity* $ID \in \{0,1\}^*$ *and a ciphertext C, it generates the private key* d_{ID} *associated to ID and returns either a plaintext* $M \in \mathcal{M}$ *or a distinguished symbol* $\perp$ *indicating that the ciphertext was not correctly formed.*

   $\mathcal{A}$ can present her queries adaptively in the sense that each query may depend on the answers to previous ones. At some point, she produces two plaintexts $M_0, M_1 \in \mathcal{M}$, *and a target identity* ID^* *for which she has not requested the private key in stage 2. The challenger computes* $C = \mathbf{Encrypt}(M_b, \mathsf{mpk}, ID^*)$, *for a random hidden bit* $b \xleftarrow{R} \{0,1\}$, *which is sent to* $\mathcal{A}$.
3. *In the guess stage,* $\mathcal{A}$ *asks new queries as in the find stage but is restricted not to issue a key extraction request on the target identity* ID^* *and cannot submit C to the decryption oracle for the identity* ID^*. *Eventually,* $\mathcal{A}$ *outputs a bit* b' *and wins if* $b' = b$.

$\mathcal{A}$'s advantage is defined as $Adv(\mathcal{A}) := |2 \times Pr[b' = b] - 1|$.

The above definition captures the chosen-ciphertext scenario where the adversary is granted access to a decryption oracle throughout the game. There exists a weaker definition, called *chosen-plaintext* security (or IND-ID-CPA for short), where no such decryption oracle is given to the adversary.

The IBE scheme proposed by Cocks in 2001 [13] is based on quadratic *residues* and on the properties of the Legendre and Jacobi symbols for Blum integers (i.e., composite integers $n = pq$, where p and q are primes such that $p \equiv q \equiv 3 \pmod 4$). It is made of the four algorithms depicted below.

Setup: The PKG picks prime numbers p and q such that $p \equiv q \equiv 3 \pmod 4$, computes their product $n = pq$, and chooses a hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_n^*$. The PKG's master secret key is defined to be $\mathsf{msk} = (p, q)$, and the master public key consists of $\mathsf{mpk} = (n, H)$.

Keygen: Given an identity ID, the PKG computes a sequence of hash values starting from ID until obtaining $a = H(H(H \ldots (ID))) \in \mathbb{Z}_n^*$ such that $\left(\frac{a}{n}\right) = 1$. For such a $a \in \mathbb{Z}_n^*$, either a or $-a$ is a square in $\mathbb{Z}_n^*$. It is easy to verify that $r = a^{\frac{n+5-(p+q)}{8}} \bmod n$ satisfies $a = r^2 \bmod n$ or $a = -r^2 \bmod n$ depending on whether $\left(\frac{a}{p}\right) = \left(\frac{a}{q}\right) = 1$ or $\left(\frac{a}{p}\right) = \left(\frac{a}{q}\right) = -1$. The obtained r is returned to the user as a private key.

Encrypt: The sender does not know which of a or $-a$ is a square in $\mathbb{Z}_n^*$ and one first considers the case $a = r^2 \bmod n$. The sender generates a symmetric transport key K and encrypts the plaintext M with it. Each bit x of that symmetric key is then encrypted before being sent to the receiver B. To do this, A encodes x in $\{-1, 1\}$ rather than in $\{0, 1\}$ and does the following.

1. Pick a random $t \in \mathbb{Z}_n^*$ such that $\left(\frac{t}{n}\right) = x$.
2. Compute $s = \left(t + \frac{a}{t}\right) \bmod n$ (since $\left(\frac{t}{n}\right) \neq 0$, t is coprime with p and q and thus invertible in $\mathbb{Z}_n$) and send it to B.

Since A does not know which of a or $-a$ is the square of B's decryption key, A has to repeat the above process for a new t and, this time, send $s = (t - a/t) \bmod n$. Hence,

$2|n|$ bits, where $|x|$ denotes the bitlength of x, have to be transmitted for each bit of the symmetric key.

Decrypt: B recovers x as follows. Given that

$$t(1+r/t)^2 \equiv t+2r+\frac{r^2}{t} \equiv t+2r+\frac{a}{t} \equiv s+2r \pmod{n},$$

B can compute $\left(\frac{s+2r}{n}\right) = \left(\frac{t}{n}\right) = x$ and recover x using his/her private key r thanks to the multiplicative properties of the Jacobi symbol. Once the symmetric key K is obtained in clear, the ciphertext can be decrypted.

For 128-bit symmetric keys, the scheme is fairly cheap from a computational standpoint: the sender's cost is dominated by 2×128 Jacobi symbol evaluations and 2×128 modular inversions. The receiver just has to compute 128 Jacobi symbols since he/she knows which of a or $-a$ is the square of his/her private key. The drawback of the scheme is its bandwidth overhead: for a 1024-bit modulus n and a 128-bit symmetric transport key, at least 2×16 Kb need to be transmitted if all the integers s are sent together.

Cocks showed that his construction is secure (in the random oracle model) against chosen-plaintext attacks under the Quadratic Residuosity Assumption (i.e., the hardness of deciding whether or not a random integer a such that $\left(\frac{a}{n}\right) = 1$ is a square). Chosen-ciphertext security can be acquired via several generic transformations such as the one of Fujisaki and Okamoto [16].

While elegant, Cocks's construction is somewhat bandwidth-demanding. In addition, separately encrypting each bit of plaintext makes it difficult to turn the scheme into a chosen-ciphertext secure multi-bit encryption scheme. In 2007, Boneh et al. [9] showed how to construct a multi-bit quadratic-residuosity-based IBE scheme featuring much shorter ciphertexts. Other IBE systems avoiding the limitation of Cocks's proposal will be covered in the chapter dedicated to identity-based encryption.

Recommended Reading

1. Barreto PSLM, Libert B, McCullagh N, Quisquater JJ (2005) Efficient and provably-secure identity-based signatures and signcryption from bilinear maps. In: Advances in cryptology – Asiacrypt '05. Lecture notes in computer science, vol 3788. Springer, Heidelberg, pp 515–532
2. Bellare M, Namprempre C, Neven G (2004) Security proofs for identity-based identification and signature schemes. In: Advances in cryptology – Eurocrypt '04. Lecture notes in computer science, vol 3027. Springer, Heidelberg, pp 268–286
3. Bellare M, Namprempre C, Pointcheval D, Semanko M (2001) The power of RSA inversion oracles and the security of Chaum's RSA-based blind signature scheme. In: Financial cryptography 2001. Lecture notes in computer science, vol 2339. Springer, Heidelberg, pp 309–328
4. Bellare M, Neven G (2007) Identity-based multi-signatures from RSA. In: RSA conference cryptographers' track (CT-RSA '07). Lecture notes in computer science, vol 4377. Springer, Heidelberg, pp 145–162
5. Bellare M, Palacio A (2002) GQ and schnorr identification schemes: proofs of security against impersonation under active and concurrent attacks. In: Advances in cryptology – Crypto '02, Lecture notes in computer science, vol 2442. Springer, Heidelberg, pp 162–177
6. Bellare M, Rogaway P (1993) Random oracles are practical: a paradigm for designing efficient protocols. In: Proceedings of the 1st ACM conference on computer and communications security, Fairfax, pp 62–73
7. Beth T (1988) Efficient zero-knowledge identification scheme for smart cards. In: Advances in cryptology – Eurocrypt '88. Lecture notes in computer science, vol 330. Springer, Heidelberg, pp 77–84
8. Boneh D, Franklin M (2001) Identity based encryption from the Weil pairing, SIAM J of Comput 32(3): 586–615, 2003. Earlier version in advances in cryptology – Crypto '01. Lecture notes in computer science, vol 2139. Springer, Heidelberg, pp 213–229
9. Boneh D, Gentry C, Hamburg M (2007) Space-efficient identity-based encryption without pairings. In: Proceedings of the FOCS '07, Providence, pp 647–657
10. Cha JC, Cheon JH (2003) An identity-based signature from gap Diffie-Hellman groups. In: Public Key Cryptography 2003 (PKC '03). Lecture notes in computer science, vol 2567. Springer, Heidelberg, pp 18–30
11. Cayrel PL, Gaborit P, Girault M (2007) Identity-based identification and signature schemes using correcting codes. In: Workshop of cryptography and coding 2007, Versailles
12. Cayrel PL, Gaborit P, Galindo D, Girault M (2009) Improved identity-based identification using correcting codes. In: Computing Research Repository (CoRR) abs/0903.0069
13. Cocks C (2001) An identity based encryption scheme based on quadratic residues. In: Proceedings of cryptography and coding. Lecture notes in computer science, vol 2260. Springer, Heidelberg, pp 360–363
14. Dodis Y, Katz J, Xu S, Yung M (2003) Strong key-insulated signature schemes. In: Public key cryptography 2003 (PKC '03). Lecture notes in computer science, vol 2567. Springer, Heidelberg, pp 130–144
15. Fiat A, Shamir A (1986) How to prove yourself: practical solutions to identification and signature problems. In: Advances in cryptology – Crypto '86. Lecture notes in computer science, vol 263. Springer, Heidelberg, pp 186–194
16. Fujisaki E, Okamoto T (1999) Secure integration of asymmetric and symmetric encryption schemes. In: Advances in cryptology – Crypto '99. Lecture notes in computer science, vol 1666. Springer, Heidelberg, pp 537–554
17. Fischlin M, Fischlin R (2002) The representation problem based on factoring. In: RSA conference cryptographers' track (CT-RSA '02). Lecture notes in computer science, vol 2271. Springer, Heidelberg, pp 96–113
18. Galindo D, Herranz J, Kiltz E (2006) On the generic construction of identity-based signatures with additional properties, In: Avances in cryptology – Asiacrypt '06. Lecture notes in computer science, vol 4284. Springer, Heidelberg, pp 178–193

19. Galindo D, Garcia FD (2009) A schnorr-like lightweight identity-based signature scheme. In: Progress in cryptology – Africacrypt '09. Lecture notes in computer science, vol 5580. pp 135–148
20. Gentry C, Ramzan Z (2006) Identity-based aggregate signatures. In: Public key cryptography 2006 (PKC '06). Lecture notes in computer science, vol 3958. Springer, Heidelberg, pp 257–273
21. Gentry C, Silverberg A (2002) Hierarchical ID-based cryptography. In: Advances in cryptology – Asiacrypt '02. Lecture notes in computer science, vol 2501. Springer, Heidelberg, pp 548–566
22. Goldwasser S, Micali S, Rivest R (1998) A digital signature scheme secure against adaptive chosen-message attacks. SIAM J Comput 17(2):281–308
23. Guillou L, Quisquater JJ (1998) A "Paradoxical" identity-based signature scheme resulting from zero-knowledge. In: Advances in cryptology – Crypto '88. Lecture notes in computer science, vol 403. Springer, Heidelberg, pp 216–231
24. Herranz J (2006) Deterministic identity-based signatures for partial aggregation. Comput J 49(3):322–330
25. Kiltz E, Mityagin A, Panjwani S, Raghavan B (2005) Append-only signatures. In: International colloquium automata, languages and programming (ICALP '05). Lecture notes in computer science, vol 3580. Springer, Heidelberg, pp 434–445
26. Hess F (2003) Efficient identity based signature schemes based on pairings. In: Proceedings of SAC '02. Lecture notes in computer science, vol 2595. Springer, Heidelberg, pp 310–324
27. Horwitz J, Lynn B (2002) Toward hierarchical identity-based encryption. In: Advances in cryptology – Eurocrypt '02. Lecture notes in computer science, vol 2332. Springer, Heidelberg, pp 466–481
28. Okamoto T (1992) Provably secure and practical identification schemes and corresponding signature schemes. In: Advances in cryptology – Crypto '92. Lecture notes in computer science, vol 740. Springer, Heidelberg, pp 31–53
29. Ong H, Schnorr CP (1990) Fast signature generation with a fiat shamir-like scheme. In: Advances in cryptology – Eurocrypt '90. Lecture notes in computer science, vol 473. Springer, Heidelberg, pp 432–440
30. Paterson KG (2002) ID-based signatures from pairings on elliptic curves. Available at http://eprint.iacr.org/2002/004/
31. Paterson KG, Schuldt J (2006) Efficient Identity-based signatures secure in the standard model. In: 11th Australasian conference on information security and privacy (ACISP '06). Lecture notes in computer science, vol 4058. Springer, Heidelberg, pp 207–222, 387–398
32. Pointcheval D, Stern J (1996) Security proofs for signature schemes. In: Advances in cryptology – Eurocrypt '96. Lecture notes in computer science, vol 1070. Springer, Heidelberg, pp 387–398
33. Pointcheval D, Stern J (2000) Security arguments for digital signatures and blind signatures. J Cryptol 13(3):361–396
34. Sakai R, Ohgishi K, Kasahara M (2000) Cryptosystems based on pairing. In: The 2000 symposium on cryptography and information security, Okinawa, Japan
35. Schnorr CP (1989) Efficient identification and signatures for smart cards. In: Advances in cryptology – Crypto '89. Lecture notes in computer science, vol 435. Springer, Heidelberg, pp 239–252
36. Shamir A (1984) Identity based cryptosystems and signature schemes. In: Advances in cryptology – Crypto '84. Lecture notes in computer science, vol 196. Springer, Heidelberg
37. Waters B (2005) Efficient identity-based encryption without random oracles. In: Advances in cryptology – Eurocrypt 2005. Lecture notes in computer science, vol 2567. Springer, Heidelberg, pp 114–127

Identity-Based Encryption

Martin Gagné
Department of Computer Science, University of Calgary, Calgary, Canada

Synonyms

IBE; ID-based encryption

Related Concepts

►Public Key Cryptography

Definition

An identity-based encryption scheme is a public key encryption scheme in which any bit string can serve as a public key.

Background

The concept of identity-based encryption (IBE) was first introduced by Shamir in 1984 [8]. His original motivation was to eliminate the need for directories and certificates by using the identity of the receiver as the public key. While solutions for the related problem of identity-based signature were quickly found, identity-based encryption proved much more challenging.

The first usable and provably secure IBE scheme was proposed in 2001 by Boneh and Franklin using bilinear pairings in elliptic curve groups [2]. In the same year, Cocks built an IBE scheme based on the quadratic residuosity problem [4], but that scheme was very inefficient. A more efficient provably secure scheme based on the same problem was later presented by Boneh, Gentry, and Hamburg [3]. Recently, Gentry, Peikert, and Vaikuntanathan showed that IBE schemes could also be constructed using new hard problems in lattices [6].

Theory

In traditional public key encryption, public keys tend to look fairly random, and are not easily associated with a user. For example, when one wants to send his credit card number to make a purchase on a Web site, he wants to be sure that he is really encrypting the message using the public key of the Web site, and not that of a hacker posing as the Web site. For that reason, it is necessary to have a trusted

authority who can issue certificate that confirms which public key belongs to which user and prevents impersonation. With an identity-based encryption scheme, one could simply encrypt the message using the URL of the Web site as the public key.

An identity-based encryption scheme consists of four randomized algorithms: Setup, Extract, Encrypt, and Decrypt.

Setup: Takes as input a security parameter and outputs the system parameters and master-key. The system parameters must include the description of the message space $\mathcal{M}$ and the ciphertext space $\mathcal{C}$. The system parameters will be publicly known while the master-key is known only to the private key generator.

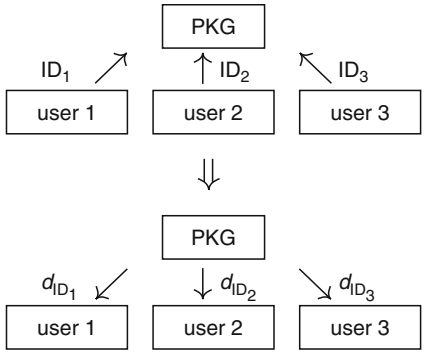
Extract: Takes as input the system parameters, the master-key, and an arbitrary string ID and outputs the private key d_{ID} corresponding to the public key ID.

Encrypt: Takes as input the system parameters, a public key ID, and a plaintext $M \in \mathcal{M}$ and outputs a corresponding ciphertext.

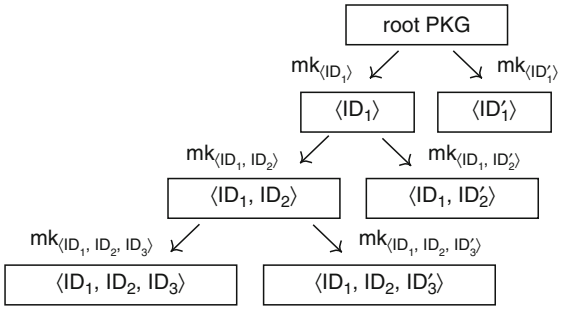
Decrypt: Takes as input the system parameters, a private key d_{ID}, and a ciphertext $C \in \mathcal{C}$ and outputs the corresponding plaintext.

The algorithms *Setup* is run by a trusted party, the *private key generator* (PKG). The PKG also runs the algorithm *Extract* at the request of a user who wishes to obtain the private key corresponding to some string (see Fig. 1). Note that the user needs to prove to the PKG that he is the legitimate "owner" of this string (for example, to obtain the private key corresponding to "bob@yahoo.com," the user must prove that bob@yahoo.com is truly his email address), and the private key must be returned to the user on a secure channel in order to keep the private key secret. The algorithms *Encrypt* and *Decrypt* are run by the users to encrypt and decrypt messages. They must satisfy the standard consistency constraints, namely, if all the algorithms are applied correctly, then any message in the plaintext space encrypted with the algorithm *Encrypt* should be correctly decrypted by the algorithm *Decrypt* when using the appropriate private key.

Some identity-based encryption schemes can support an hierarchy of public keys [7]. In those hierarchical identity-based encryption (HIBE) schemes, public keys are vectors of bit strings and the private corresponding to public key $\langle ID_1, \ldots, ID_n \rangle$ can be used as a master key to produce private keys corresponding to public keys of the form $\langle ID_1, \ldots, ID_n, ID_{n+1} \rangle$ (see Fig. 2). These hierarchical identity-based encryption schemes allow a root PKG to distribute the workload of producing private keys and verifying identities by delegating to lower-level PKGs. They



Identity-Based Encryption. Fig. 1 Private key request in an IBE scheme



Identity-Based Encryption. Fig. 2 Hierarchy of public keys in an HIBE

are also useful in environments in which a natural hierarchy is present (for example, a head of a company computes keys for managers, who in turn compute keys for employees).

Applications

As mentioned before, identity-based encryption was originally designed to simplify public key infrastructure. It is, however, a very versatile tool that can be used for a multitude of other applications. Here are a few examples.

Ephemeral public keys: An identity-based encryption scheme can be used to make public keys expire by using the public key "receiver-address ‖ current-date," where current-date can be the day, week, month, or year depending on the frequency at which the users are required to renew their key. This way, a user's decryption capability can be revoked simply by letting his key expire and not issuing a new one.

Managing user credentials: By encrypting the messages using the address "receiver-address ‖ current-date ‖

clearance-level," the receiver will be able to decrypt the message only if he has the required clearance.

Chosen-ciphertext security: Public key encryption schemes secure against adaptive chosen ciphertext attack can be devised from identity-based encryption schemes secure against chosen plaintext attack [1]. This method is in fact frequently used to construct depth d hierarchical identity-based encryption schemes secure against chosen ciphertext attack from depth $d + 1$ hierarchical identity-based encryption schemes secure against plaintext attack.

Open Problems

The most active area of research in identity-based encryption is the construction of efficient identity-based encryption schemes based on lattice problems. These schemes could offer a lower asymptotic computational complexity than the schemes based on pairings or quadratic residues, and could potentially resist attacks from quantum computers, but the most efficient schemes known today remain slower than the best pairing-based scheme. A lattice-based scheme whose efficiency rivals that of pairing-based schemes would be a significant breakthrough.

On a more theoretical point of view, all the current identity-based encryption schemes are either inefficiently reduced to a well-known complexity assumption [9], or based on slightly exotic problems [5]. It would be interesting to see if one could construct an efficient identity-based encryption scheme whose security can be tightly reduced to a well-known complexity assumption.

Recommended Reading

1. Boneh D, Canetti R, Halevi S, Katz J (2006) Chosen-ciphertext security from Identity-based encryption. SIAM J Comput 36(5):915–942
2. Boneh D, Franklin M (2001) Identity based encryption scheme from the weil pairing. In: Kilian J (ed) Advances in cryptology – CRYPTO 2001. Lecture notes in computer science, vol 2139. Springer-Verlag, Berlin, pp 213–229
3. Boneh D, Gentry C, Hamburg M (2007) Space-efficient identity based encryption without pairings. In: Proceedings of the 48th annual IEEE symposium on foundations of computer science – FOCS 2007, Providence, RI. IEEE Computer Society, Los Alamitos, CA, pp 647–657
4. Cocks C (2001) An identity based encryption scheme based on quadratic residues. In: Proceedings of the 8th IMA international conference on cryptography and coding, Cirencester, UK. Lecture notes in computer science, vol 2260. Springer-Verlag, Berlin, pp 360–363
5. Gentry C (2006) Practical identity-based encryption without random oracles. In: Vaudenay S (ed) Advances in cryptology – EUROCRYPT 2006. Lecture notes in computer science, vol 4004. Springer-Verlag, Berlin, pp 445–464
6. Gentry C, Peikert C, Vaikuntanathan V (2008) Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the 40th annual ACM symposium on theory of computing – STOC 2008, Victoria, BC. ACM, New York, pp 197–206
7. Gentry C, Silverberg A (2002) Hierarchical ID-based cryptography. In: Zheng Y (ed) Advances in cryptology – ASIACRYPT 2002. Lecture notes in computer science, vol 2501. Springer-Verlag, Berlin, pp 149–155
8. Shamir A (1985) Identity-based cryptosystems and signature schemes. In: Blakley GR, Chaum D (eds) Advances in cryptology – CRYPTO 84. Lecture notes in computer science, vol 196. Springer, Berlin, pp 47–53
9. Waters B (2005) Efficient identity-based encryption without random oracles. In: Cramer R (ed) Advances in cryptology – EUROCRYPT 2005. Lecture notes in computer science, vol 3494. Springer-Verlag, Berlin, pp 114–127

Impersonation Attack

Carlisle Adams
School of Information Technology and Engineering (SITE), University of Ottawa, Ottawa, Ontario, Canada

Related Concepts

►Entity Authentication; ►Identification

Definition

An *impersonation attack* is an attack in which an adversary successfully assumes the identity of one of the legitimate parties in a system or in a communications protocol. The goal of a strong ►identification or ►entity authentication protocol is to make negligible the probability that for a given party A, any party C distinct from A, carrying out the protocol and playing the role of A, can cause another party B to complete and accept A's identity [1].

Recommended Reading

1. Menezes A, van Oorschot P, Vanstone S (1997) Handbook of applied cryptography. CRC, Boca Raton, FL

Implicit Key Authentication

►Key Authentication

Impossible Differential Attack

Alex Biryukov
FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg

Related Concepts

►Block Ciphers; ►Differential Cryptanalysis

Definition

Impossible differential attack is a ►chosen plaintext attack and is an extension of ►differential cryptanalysis.

Background

Impossible differential attack [1] was defined in 1998 and has been shown to break 31 out of 32 rounds of the cipher ►Skipjack [5], designed by the NSA and declassified in 1998. Independently, an attack based on similar principles was used by Knudsen in 1998 to cryptanalyse six rounds of the cipher *DEAL* [3] which was one of his proposals for the AES (►Rijndael/AES). The attack, using impossible differentials, was shown to be a generic tool for ►cryptanalysis [2] and was applied to improve on the best known attacks for such strong and long standing ►block ciphers as ►IDEA and Khufu [4], breaking round-reduced versions of these ciphers. The two main ideas were the ►miss-in-the-middle technique for construction of impossible events inside ciphers and the *sieving technique* for filtering wrong key-guesses.

Theory

Once the existence of impossible events in a cipher is proven, it can be used directly as a *distinguisher* from a *random permutation* (►Substitutions and Permutations). Furthermore one can find the keys of a cipher by analyzing the rounds surrounding the impossible event, and guessing the subkeys of these rounds. All the keys that lead to a contradiction are obviously wrong. The impossible event in this case plays the role of a *sieve*, methodically rejecting the wrong key guesses and leaving the correct key. It is important to note that the miss-in-the-middle technique is only one of the ways to construct impossible events and that the sieving technique is only one of the possible ways to exploit them.

In order to get a feel of the attack, consider a cipher $E(\cdot)$ with n-bit blocks, a set of input differences P of cardinality 2^p and a corresponding set of output differences Q of cardinality 2^q. Suppose that no difference from P can cause an output difference from Q. One may ask how many chosen texts should be requested in order to distinguish $E(\cdot)$ from a random permutation? In general, about 2^{n-q} pairs with differences from P are required. This number can be reduced by using *structures* (a standard technique for saving chosen plaintexts in differential attacks). In the optimal case, one may use structures of 2^p texts which contain about 2^{2p-1} pairs with differences from P. In this case $2^{n-q}/2^{2p-1}$ structures are required, and the number of chosen texts used by this distinguishing attack is about $2^{n-p-q+1}$ (assuming that $2p < n - q + 1$). Thus the higher $p + q$ is, the better is the distinguisher based on the impossible event.

Impossible differential attack has become a popular tool of cryptanalysis and has been used recently for the analysis of many ciphers and hash functions (Camelia, Aria, Safer, Clefia, Misty, and various versions of AES).

Recommended Reading

1. Biham E, Biryukov A, Shamir A (1999) Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials. In: Stern J (ed) Advances in cryptology – eurocrypt'99. Lecture notes in computer science, vol 1592. Springer, Berlin, pp 12–23
2. Biham E, Biryukov A, Shamir A (1999) Miss in the middle attacks on IDEA and Khufu. In: Knudsen LR (ed) Fast software encryption, FSE'99. Lecture notes in computer science, vol 1636. Springer, Berlin, pp 124–138
3. Knudsen LR (1998) DEAL – a 128-bit block cipher. Technical report 151, Department of informatics, University of Bergen, Norway
4. Merkle RC (1991) Fast software encryption functions. In: Menezes A, Vanstone SA (eds) Advances in cryptology – crypto'90. Lecture notes in computer science, vol 537. Springer, Berlin, pp 476–501
5. NIST (1998) Skipjack and KEA algorithm specification. Technical report, http://csrc.nist.gov/CryptoToolkit/skipjack/skipjack-kea.htm. version 2.0

Index Calculus Method

Kim Nguyen
Bundesdruckerei GmbH, Berlin, Germany

Related Concepts

►Discrete Logarithm Problem; ►Factor Base; ►Function Field Sieve; ►Number Field Sieve for the DLP; ►Smoothness

Definition

Index calculus originally refersto a method for computing discrete logarithms in a ►finite field in ►subexponential time complexity (►Discrete Logarithm Problem). The basic ideas appeared first in the work of Western and

Miller [1]. The original algorithm was invented independently by Adleman [2], Merkle [3], and Pollard [4] according to Odlyzko [5]. The first partial analysis of the complexity of the algorithm is due to Adleman [2].

Theory

Consider a finite field k with multiplicative ►group G generated by g. The main and trivial observation on which index calculus relies is that once the discrete logarithms $\log_g(g_i)$ are known, then the discrete logarithm for an element defined by $\prod_i g_i^{n_i}$ is given by the sum

$$\sum_i n_i \log_g(g_i) \bmod |G|.$$

Given the element h of G, index calculus computes the discrete logarithm $\log_g(h)$ in two main steps:

1. In a precomputation step, $\log_g(g_i)$ for sufficiently many elements g_i of G is computed.
2. Find an element g^y such that hg^y has the form $hg^y = \prod_i g_i^{n_i}$. Then by the above remark the discrete logarithm $\log_g(h)$ can be computed easily.

The way the discrete logarithms in step 1 are obtained is as follows:
Assume we have a surjective map $\phi : R \to G$ from a ►ring R, in which we have unique factorization in prime elements, to G. In the case that the surjection ϕ can be efficiently inverted, we can lift elements from G to R. Assume we have a factorization

$$\phi^{-1}(g^x) = \prod_{p \in S} p^{n_{p,x}},$$

where S, the ►factor base, denotes a subset of the set P of prime elements of R. This implies, via application of ϕ, that the equality $g^x = \prod_{p \in S} \phi(p)^{n_{p,x}}$ in G holds. Hence taking discrete logarithms to the base g we obtain

$$x = \sum_{p \in S} n_{p,x} \log_g(\phi(p)) \bmod |G|.$$

In order to recover the values $\log_g(\phi(p))$ for $p \in S$ it is now sufficient to run through two steps:

- In the first step (often referred to as ►sieving step), we collect more than $|S|$ distinct relations; these then form an overdefined system of linear equations modulo the order of the group $|G|$.
- Solve the resulting system of linear equations mod$|G|$ using linear algebra and obtain the values $\log_g(\phi(p))$ for $p \in S$.

Given any element $h \in G$, we then search for an element g^y such that $\phi^{-1}(hg^y)$ has the form

$$\phi^{-1}(hg^y) = \prod_{p \in S} p^{n_{p,y}}.$$

$\log_g(h)$ can than be computed by

$$\log_g(h) \equiv -y + \sum_{p \in S} n_{p,y} \log_g(\phi(p)) \bmod |G|.$$

Applications

Implementation Choice and Complexity Estimate

The most obvious choices for the ring R are:

1. The natural numbers $\mathbb{N}$ in the case of prime fields $G = \mathbb{F}_p^*$.
2. The ring $\mathbb{F}_p[x]$ of polynomials over $\mathbb{F}_p$ in the case of extension fields $G = \mathbb{F}_{p^n}$ (►Extension field).

For the set S we choose smooth prime elements of the ring R (►smoothness):

1. The prime numbers less than or equal to a smoothness bound B:
$$S = \{p \in \mathbb{N} \text{ prime} \mid p \le B\}$$
in the case of prime fields.
2. The ►irreducible polynomials of norm less than or equal to a smoothness bound B:
$$S = \{f \in \mathbb{F}_p[x] \text{ irreducible} \mid p^{\deg(f)} \le B\}$$
in the case of extension fields.

Using results on the distribution of smooth elements one sees that using a smoothness bound $B = L_{p^n}(1/2, \sqrt{1/2})$, the first step of collecting relations by computing random powers g^x takes expected time $L_{p^n}(1/2, \sqrt{2})$ (►L-notation for a definition of the preceding notation). In the case of prime fields, we have $n = 1$ in this and all subsequent expressions. Here one has to be careful to also take into account the time needed for the smoothness test of the lifted elements $\phi^{-1}(g^x)$. However, using the ►Elliptic Curve Method for factoring in the prime field case one has an expected running time of $L_p(1/2, 1)$ for factorization (in the worst case), while in the case of extension fields, for a polynomial f over $\mathbb{F}_p$ of degree n factorization using the Berlekamp algorithm has complexity $O(n^3 + tpn^2)$, where t denotes the number of irreducible factors of the polynomial f.

For the linear algebra part, one notes that the resulting system of around $L_{p^n}(1/2, \sqrt{1/2})$ linear equations is

sparse, hence special methods can be applied for the solution of this system which have expected running time $L_{p^n}(1/2, \sqrt{2})$.

Finally, for the last step of finding a smooth lift of the form $\phi^{-1}(g^x)$ the running time estimation of $L_{p^n}(1/2, \sqrt{2})$ of the first step applies as well, resulting in an overall expected running time of $L_{p^n}(1/2, \sqrt{2})$ for the complete index calculus algorithm.

Variants of the Algorithm

By replacing $\mathbb{N}$ or $\mathbb{F}_p[x]$ with different rings also allowing unique factorization one obtains variants of the classical index calculus algorithm.

One of the most important variants in the case of prime fields uses the ring of Gaussian integers $\mathbb{Z}[i]$ of the imaginary quadratic field $\mathbb{Q}(i)$ and is therefore called the *Gaussian integer method* (refer to [6] for more details). It has expected running times $L_p(1/2, 1)$ for the precomputation part of the algorithm and $L_p(1/2, 1/2)$ for the computation of individual discrete logarithms.

In the case of extension fields, Coppersmith's method is of special importance. It is applicable only in the case of small characteristic and was the first method to compute discrete logarithms in subexponential complexity better than $L_{2^n}(1/2, c)$; its expected running time is $L_{2^n}(1/3, 1.588)$ (see [7]). It was realized later that this method in fact is a special case of the function field sieve (►Sieving in function fields).

Index Calculus Using the Number Field Sieve

A different idea to solve the discrete logarithm problem in a finite field $k = \mathbb{F}_p$ is to find integers s and t such that the equation $g^s \cdot h^t = w^q$ holds for some $w \in k$, where q is a divisor of the order of the multiplicative group k^*. If t is coprime to q we have then computed $\log_g(h) \bmod q$ since we have

$$log_g(h) \equiv -st^{-1} \bmod q.$$

Having computed $\log_g(h)$ modulo the different primes dividing the order of k^* we can then recover $\log_g(h)$ using the ►Chinese Remainder Theorem.

We are thus led to the problem of constructing q–th powers in k, i.e., relations of the form $g^s \cdot h^t = w^q$. Index calculus techniques can be applied to this problem. If we choose R to be the ring of integers of a more general, nontrivial number field, we are led to techniques related to the ►Number Field Sieve for factoring.

Contrary to the techniques described above, the approach using ►Number Field Sieve for DLP uses two factor bases: one consisting of small rational primes, the other consisting of algebraic primes of small norm.

This is due to the fact that the Number Field Sieve approach uses two different maps ϕ: one is the natural projection $\phi_1 : \mathbb{N} \to \mathbb{F}_p$, while the second one ϕ_2 is a certain projection from the ring of algebraic integers R to $\mathbb{F}_p$.

In the sieving stage, pairs of smooth elements s_1, s_2 are collected which have the additional property that the equality $\phi_1(s_1) = \phi_2(s_2)$ holds. Again we are led to a system of linear equations, and solving this system will lead to the construction of a q–th power in $\mathbb{F}_p$ and will thus yield the solution of the discrete logarithm problem. Schirokauer presented an algorithm based on this approach that has complexity $L_p(1/3, (64/9)^{1/3})$ (see [8]).

Details regarding the application of the Number Field Sieve in this setting can be found in [9] and [8]. See also the entry on the ►Discrete Logarithm Problem for more details on recent challenges and attacks.

Significant advances could be achieved recently in the special case of medium primes both for the function field and the number field sieve. Refer to [10], [11] and the entry on the ►Function Field Sieve.

Index Calculus and Elliptic Curves

The ►elliptic curve discrete logarithm problem (ECDLP) has attracted great interest since its introduction to cryptography in 1985. One of the most interesting features of this problem is that it has up till now resisted all attempts to apply index calculus techniques to it.

Consider the ECDLP on a curve over the finite field $\mathbb{F}_{p^n}$. The most naive idea would be to take a surjection $\phi : K \to \mathbb{F}_{p^n}$, choose an elliptic curve E' over K reducing to E via ϕ, and thus obtain a map between elliptic curves $\phi : E'(K) \to E(\mathbb{F}_{p^n})$, the most obvious one being $\phi : \mathbb{Q} \to \mathbb{F}_p$ in the case of prime fields. However, properties of elliptic curves over global fields imply that there is no chance to lift sufficiently many points from $E(\mathbb{F}_p)$ to $E(\mathbb{Q})$ with reasonably sized coefficients in order to be able to apply index calculus techniques. The main ingredient here is the existence of a quadratic form called canonical height on the Mordell–Weil group (modulo torsion) of the global elliptic curve.

A different approach (known as XEDNI calculus) suggested to first lift sufficiently many points from $E(\mathbb{F}_p)$ to $\mathbb{Q}$ and then fit a globally defined elliptic curve through these global points. However, this idea was proven to have expected running time $O(p)$, far worse than the square root complexity of the exhaustive search approach.

Recently significant advances with respect to index calculus methods on elliptic curves and curves of higher genus could be achieved. See especially [12] that contains a survey of the state of the art with respect to index calculus applied to the divisor class group of curves. Also,

I

Diem ([12]) and Gaudry ([13]) independently came up with an index calculus attack of subexponential complexity for elliptic curves defined over small degree extension fields. More precisely, one considers the extension field $\mathbb{F}_{q^n}$, then if one allows n to go to infinity, the complexity of their algorithm remains subexponential as long as n is upper bounded by $O(\sqrt{\log(q)})$. Although this result does not have any impact on cryptosystems being currently used in practice, it is a significant breakthrough as for the first time it was possible to establish an index calculus attack on elliptic curves that – at least asymptotically – has subexponential complexity.

Recommended Reading

1. Wester AE, Miller JCP (1968) Tables of indices and primitive roots. Royal society mathematical tables, vol 9. CUP
2. Adleman LM (1979) A subexponential algorithm for the discrete logarithm problem with applications to cryptography. In: Proc 20th IEEE found comp sci symp, pp 55–60
3. Merkle R (1993) Discrete logarithms in F_p using the number field sieve. SIAM J Discrete Math 6:124–138
4. Pollard J (1978) Monte Carlo methods for index computations (mod p). Math Comp 32:918–924
5. Odlyzko A (1985) Discrete logarithms in finite fields and their cryptographic significance. In: Advances in cryptology: proceedings of EUROCRYPT 84, lecture notes in computer science 209, Springer, pp 224–314
6. Lamacchia BA, Odlyzko AM (1991) Computation of discrete logarithms in prime fields. Designs, Codes and Cryptography 1:47–62
7. Coppersmith D (1984) Fast evaluation of discrete logarithms in fields of characteristic two. IEEE Trans Inform Theory 30:587–594
8. Schirokauer O (1993) Discrete logarithms and local units. Philos Trans Roy Soc London Ser A 345:409–423
9. Gordon DM (1979) Secrecy, authentication and public key systems. Ph.D. Dissertation, Dept. of Electrical Engineering, Stanford University
10. Joux A, Lercier R (2006) The function field sieve in the medium prime case. In: Proceedings EUROCRYPT 2006, pp 254–270
11. Joux A, Lercier R, Smart N, Vercauteren F (2006) The number field sieve in the medium prime case. In: Proceedings EUROCRYPT 2006, pp 326–344
12. Diem C (2009) On arithmetic and the discrete logarithm problem in class groups of curves. Habil Thesis, Leipzig
13. (2008) Index calculus for abeliam varieties of small dimension and the elliptic curve discrete logarithm problem. J Symbolic Comput

Indistinguishability of Encryptions

►Semantic Security

Inference Control

Joachim Biskup
Fakultät für Informatik, Technische Universität Dortmund, Dortmund, Germany

Related Concepts

►Access Control from an OS Security Perspective; ►Information Flow Control

Definition

Inference control is a mechanism to confine the information content of data or behavior made accessible to or being observable by a participant to whom some piece(s) of information should be kept confidential.

Background

Dorothy Denning gave a first authoritative introduction to inference control in her seminal book on cryptography and data security [12], mainly focusing on inference (and ►information flow) problems related to programming language constructs [1, 7, 11, 13, 23, 28] and ►statistical databases [6, 14, 15, 31]. Later on, inference control has been studied in a large variety of other fields, including general query processing in information systems [4, 5, 16, 27], labeling in ►mandatory access control systems [8–10, 22, 24, 30, 32], noninterference properties for general computing systems [18, 21, 26], and information hiding in various contexts [19, 20, 25]. The survey by Farkas/Jajodia [17] provides a critical summary, and both the textbook by Bishop [2] and the monograph by Biskup [3] treat important aspects of inference control.

Theory

Aiming to preserve the *confidentiality* of some piece(s) of *information*, the security *mechanism* of inference control restricts the *access* to *data* or *behavior* or manipulates data or behavior deliberately shown – or just being visible – to some participant, in order to confine that participant in the *usability* of his observations employing *rational reasoning*.

In principle, though in practice not always performed, considering the goals of inference control should be an integral part of both *granting privileges* for an *access control* system and of *distributing keys* for *encryption*, and thereby enabling access to data and thus to the information represented by that data. Moreover, as far as a cryptographic mechanism, using a secret or private key, converts data such that an unauthorized observer, without a knowledge of the key employed, cannot gain information about the original

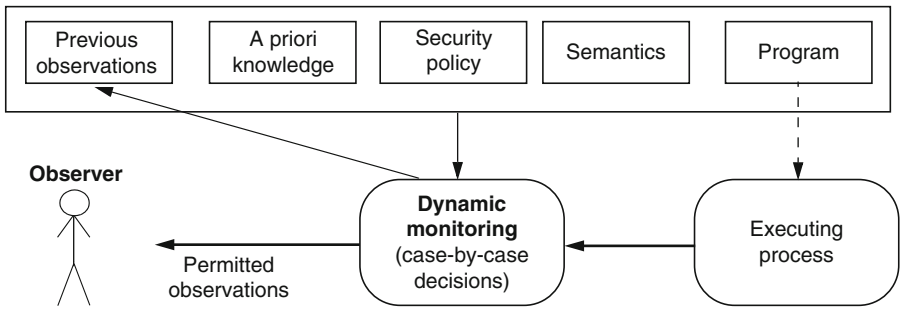
data, cryptography can be seen as dealing with inference control too. Like for cryptography, besides restricting the gain of information, for the sake of ►*availability*, inference control always has to ensure that the participants specified to acquire dedicated information are enabled to simply extract or at least to infer that information.

In general, a (potentially threatening) participant cannot be forced to ignore observations that actually happened or to refrain from inferring implications. Accordingly, inference control has to ensure that explicitly permitted, or inevitably accessible, observations are not harmful even if the participant tries their best to act as a "rational" and maybe even "omnipotent" observer. There are two basic approaches to achieving the goals of inference control, *dynamic monitoring* and *static verification*, which might also be suitably combined:

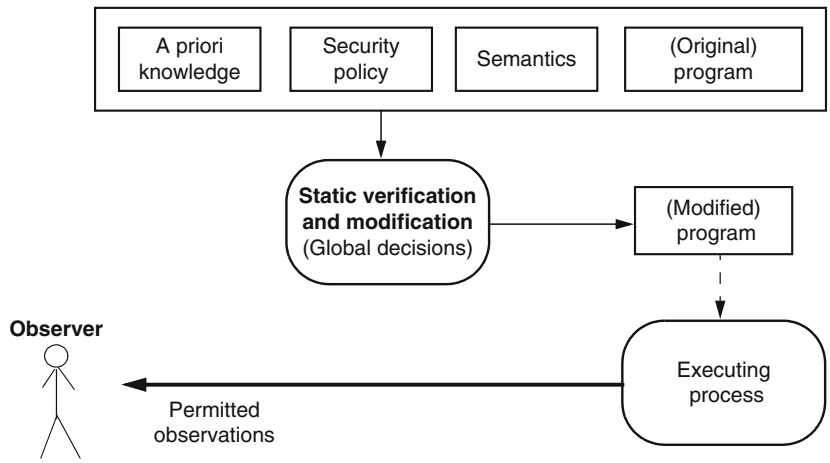
– While the computing system is running, *dynamic monitoring* inspects each relevant event as it occurs, case by case, concerning potential or actual harmful inferences, in each case *before* the event can actually be observed by the controlled participant, and if necessary *blocks* a critical observation. In doing so, dynamic monitoring keeps track of the actual history of *previous observations*, whether explicitly enabled or only assumed to have taken place.

– Before the computing system is started, *static verification* makes a *global analysis* of all possible runs of the computing system and the corresponding potential sequences of events observable by the controlled participant, and inspects them as a whole and *in advance* concerning potential or actual harmful inferences. If necessary, a planned computing system is modified in such a way that all critical observations are *blocked* right from the beginning, without further monitoring at runtime.

Rephrased briefly in terms of programming: inference control might supervise a running *process* and possibly *interferes* with the execution, as visualized in Fig. 1, or might examine a *program* as a text and possibly *modifies* the text, as visualized in Fig. 2. In addition to their particular components, both approaches have to use the applicable



Inference Control. Fig. 1 Inference control by dynamic monitoring of a process



Inference Control. Fig. 2 Inference control by static verification and modification of a program (text)

semantics and the actual or assumed a priori knowledge of the controlled participant. Furthermore, both approaches need a specification of the security requirements, i.e., a ►*security policy* that captures the pertinent notion of harmfulness.

Dynamic monitoring requires a high *runtime* effort, including keeping track of previous observations, but *only situations that actually occur* must be handled. Static verification puts a heavy burden on an administrator at *design time*, leaving nothing to do at run time, at the price that all *possible* situations must be analyzed. Both approaches demand an algorithmic treatment of implication problems of the applicable kind that might turn out to be, in general, computationally *unsolvable* and thus can only be *approximated* at best. Unfortunately, any correct approximation might result in more *blockings* than strictly needed. Dynamic monitoring is expected to enable more observations than static inspection, but case-by-case decisions might slow down the system in an unacceptable way. Dynamic monitoring then demands further approximations, resulting in additional blockings.

Regarding *blockings*, both basic approaches can employ the following kinds of technique, which allow many variations and combinations:

- An observation evaluated to be harmful is made *invisible*, or an underlying event may even be totally *suppressed*. In terms of programming, such a goal can be achieved by *refusing* to execute a crucial statement or to return a crucial output value, or by suitably modifying a program (considered as a text), respectively.
- An observation evaluated to be harmful is *substituted* by another one held to be harmless.

A further distinction is whether the observer is *notified* of a blocking, either explicitly or implicitly by some reasoning. Clearly, in some situations, recognizing a blocking might constitute harmful information too. Regarding an explicitly notified *refusal*, we have to take care about the observer's options to determine the reason for the refusal, and on the basis of that to find out about the hidden data or behavior.

Inference control is based on the crucial distinction between a participant just observing some *data* or *behavior* on the one hand, and gaining *information* on the other hand. The difference might depend on various circumstances, about which the controller has to make sufficiently appropriate postulates:

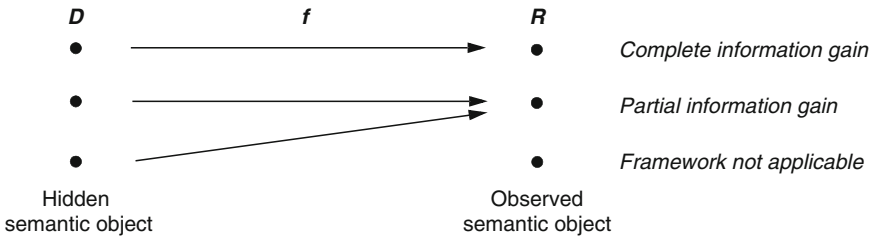
- The observer selects a *framework for reasoning* as the pertinent *universe of discourse*, and thereby assigns a *meaning* to the observation.
- The observer has some *a priori knowledge* related to that meaning.
- The observer employs an applicable notion of *implication*, and thus he can potentially reason about the fictitious *implicational closure* of his a priori knowledge and the added meaning of the observation. Under a suitable formalization, the observer might use the standard logical implication. However, an observer also might prefer to exploit other approaches as well, including those that deal with *probabilities, vagueness, uncertainty, preferences*, and related notions.
- The observer *computationally infers* selected or even all implications, *deploying* the *computational resources* available to him. However, the observer remains inevitably restricted by the fundamental limitations exhibited by computability theory and complexity theory.
- The observer treats the newly inferred implications as the *information gained* from his observation and the other features used, thereby getting *a posteriori knowledge*.

The underlying notion of *knowledge* allows the following complementary interpretations. An observer's knowledge determines those "worlds" that the observer sees as "possible," but all of them remain *indistinguishable* to him. Complementarily expressed, the observer's knowledge leaves the observer *uncertain* about which one of the indistinguishable *possible worlds* is the actual *real world* (which is thought of as basically hidden and as revealing some of its aspects by observable data or behavior).

The crucial distinction introduced above suggests a further distinction. While an observed message m is seen as a *syntactic object* (an uninterpreted *bit string*), the assigned meaning y is treated as a *semantic object* (some interpreted item, such as a number). Then, the semantic object and the current knowledge together might contain some information about possibly other *semantic objects* x, which are the real focus of a consideration.

Any instantiation of the general perspective requires us to select an appropriate and precise mathematical model, e.g., one of the following:

- *Algebra*-oriented models link the (actual) gain of information to (algorithmically) solving sets of equations.
- *Logic*-oriented models provide means to formally express sentences supposed to be true, or even to formally denote the modalities of an agent's *knowledge* or belief about such sentences. These models link the (actual) gain of information to (algorithmically) inferring logical implications and thus to mechanical theorem proving.



Inference Control. Fig. 3 A framework of reasoning, exemplifying three cases regarding information gain

- *Probability*-oriented models add the further aspect that alternatives might be weighted by their likelihood. Then, the (actual) gain of information is linked to (algorithmically) determining *conditional a posteriori probability* distributions.

To illustrate the use of such a model, a simple functional model can be mathematically described as follows and partially visualized by Fig. 3:

- The *framework for reasoning* is given by a function $f : D \to R$ and an abstract assignment $x \mapsto f(x)$ of function values to arguments, assumed to be *known* by the observer *a priori*.
- Seeing a *syntactic object* in the form of a bit string m, the observer interprets m as a *semantic object* $y \in R$, generated by a sender by applying the function f to some semantic object $x \in D$ that is the real focus of the consideration. Accordingly, the data m is seen as possibly containing information about some (hidden) *semantic object* $x \in D$ such that $f(x) = y$. The observer aims at gaining this information, i.e., the observer tries to invert the function f for the given range value y; equivalently expressed, he attempts to find the set of solutions of the equation $f(z) = y$ for the unknown variable z.

The potential for gaining information depends on the inversion properties of the range element $y \in R$ considered, case by case. More specifically, given $y \in R$, the observer might first aim at determining the *pre-image* $\{z | f(z) = y\}$. There are then four cases (three of which are shown in Fig. 3):

- *Complete* (potential) information gain: the pre-image contains exactly one element x.
- *Partial* (potential) information gain: the pre-image contains at least two (indistinguishable) elements but does not comprise the full domain D.
- *No* information gain: the pre-image is equal to the full domain D.
- *Framework not applicable*: the pre-image is empty (and thus the interpreted observation does not fully fit the framework).

The *actual* gain depends on the observer's possibilities of actually computing the relevant items. Basically, this means that effective techniques and efficient algorithms to *solve* the *equation* considered are crucial for an actual information gain.

An important example for inference control in *cryptography* is given by a *group* $(G, \bullet, e)$. The group properties ensure the *solvability of equations*: every equation of the form $k \bullet x = y$, where two of the items are given, has a unique solution for the third item. Observing the result $y \in G$ of an application of the group operation enables a *partial information* gain about the arguments, since the pre-image of y contains exactly as many elements as G. Fixing the first (or, similarly, the second) argument to some parameter $k \in G$, and now supposing that an observer sees the result $y \in G$, two cases can be distinguished: If the observer knows the pertinent parameter k, then he can gain *complete information* about the remaining argument; otherwise, the observer gains *no information* about the remaining argument (Fig. 3).

On the basis of such structures, more advanced cryptographic constructions aim at providing parameterized functions that are injective and additionally have a property called ►*one-way with trapdoor*. This property says roughly that the values of the function can be efficiently computed but the *pre-images* can not, except when the observer knows the pertinent parameter that serves as the "trapdoor."

Another example is taken from the field of *relational databases*. Supposing a *database schema* that is declared by two relation schemes $R(A, B)$ and $S(B, C)$, where R and S denote relation symbols, and A, B, and C denote attributes (column names), a *query* is syntactically given by a *relational expression* over the schema, and the semantics of a query is a function that maps *database instances* of the form (r, s), where r and s are relations fitting the schema, onto some output relation t.

I

For this example, the *natural join* of the two relations is not injective in general, owing to *dangling tuples* that do not find a matching tuple in the other relation. Accordingly, seeing the output relation enables only a *partial* information gain about the (stored but kept hidden) instance. However, if the mutual *inclusion dependencies* (*referential constraints*) for the join attribute B are declared in the schema and later on are enforced as invariants under updates, then dangling tuples cannot occur, the join becomes injective, and a *complete* information gain is enabled.

More elaborated models might consider, e.g.:

- An arbitrary *relation* over some domains (rather than just a function) and the dependency of hidden values of one projection on accessible values of another projection; e.g., such a relation might express the semantics of a *programming language* in terms of relationships between initial and final program states: then the task of information gain is basically to transform an (observed) *postcondition* into a *weakest precondition*.
- Only specific *properties* of the semantic items rather than on their *identity*.
- The impact of observation *sequences* (rather than a single observation).
- Either complete or incomplete formalizations using some fragment of classical propositional logic or first-order *logic*; or even modal variants of them to capture not only a (fictitious) "real world" but also an observer's *knowledge* or *belief* about "reality," in particular by employing *Kripke structures* to semantically describe the "possible worlds."
- Adding some kind of weights to possible values; e.g., seeing an item as a random variable, a *weight* taken from the interval $[0..1]$ can be interpreted as a *probability*; different interpretations could be given using, e.g., *fuzzy logic*.

Applications

Inference control is employed whenever a participant of a computing system should be prevented from gaining specific information about *hidden* parts of the system behavior from observing selected *visible* parts. There are numerous examples of constructs of computing systems for which their potential for (unwanted) information gain have been considered, together with mechanisms to *block* actual exploitation in order to meet some *confidentiality* requirements. The examples include, in particular, the following situations:

- Expressions, assignments, and procedure calls: causing direct or indirect information flows
- Sequential and parallel control structures: causing implicit information flows
- Real execution time: causing covert channels
- General database query answering: causing implicational information flows
- Statistical query answering: causing disclosure of individual data
- Survey publishing: violating anonymity
- Enforcing database integrity constraints: providing a priori knowledge
- Data mining: learning of private data
- Assigning (mandatory) security labels: enabling to infer higher classified data
- Composition of cryptographic primitives: compromising secret or private keys

Finally, the topic of inference control is closely related to studies on *noninterference* that capture the duality between preventing *information gain* for the sake of *confidentiality* and preventing *causality* for the sake of *integrity*.

Open Problems and Future Directions

First, there is an obvious challenge to find application-dependent and acceptable compromises for resolving the tradeoff between the highly ambitious goals of inference control on the one hand and the affordable computational costs to algorithmically detect and restrict an unwanted gain of information on the other hand. Usually, such a compromise has to exhibit appropriate approximations that potentially endanger the availability of information crucially needed.

Second, the effectiveness of a concrete instantiation of inference control essentially depends on suitable postulates about the a priori knowledge of the controlled participant and the computational resources that participant is able and willing to spend.

Third, and closely related to the preceding points, to be both efficient and effective, in each particular situation inference control should be based on a convincing assumption about how a controlled participant is forgetting information and when information is aging and thus can be considered as becoming harmless (downgraded).

Recommended Reading

1. Andrews GR, Reitman RP (1980) An axiomatic approach to information flow in programs. ACM Trans Program Lang Syst 2(1):56–76
2. Bishop M (2003) Computer security: art and science. Addison-Wesley, Boston
3. Biskup J (2004) Security in computing systems, challenges, approaches and solutions. Springer, Heidelberg

4. Biskup J, Bonatti PA (2004) Controlled query evaluation for enforcing confidentiality in complete information systems. Int J Inf Sec 3(1):14–27
5. Brodsky A, Farkas C, Jajodia S (2000) Secure databases: constraints, inference channels, and monitoring disclosures. IEEE Trans Knowl Data Eng 12(6):900–919
6. Chin FYL (1978) Security in statistical databases for queries with small counts. ACM Trans Database Syst 3(1):92–104
7. Cohen ES (1977) Information transmission in computational systems. In: Symposium on operating system principles, SOSP 7T. ACM Press, New York, pp 133–139
8. Cuppens F, Gabillon A (1999) Logical foundations of multilevel databases. Data Knowl Eng 29(3):259–291
9. Cuppens F, Gabillon A (2001) Cover story management. Data Knowl Eng 37(2):177–20I
10. Dawson S, De Capitani di Vimercati S, Samarati P (1999) Specification and enforcement of classification and inference constraints. In: IEEE symposium on security and privacy. IEEE, Los Alamitos, pp 181–195
11. Denning DE (1976) A lattice model of secure information flow. Commun ACM 19(5):236–243
12. Denning DE (1982) Cryptography and data security. Addison-Wesley, Reading
13. Denning DE, Denning PJ (1972) Certification of programs for secure information flow. Commun ACM 20(7):504–513
14. Denning DE, Denning PJ, Schwartz MD (1979) The tracker: a threat to statistical database security. ACM Trans Database Syst 4(1):76–96
15. Denning DE, Schlörer J (1983) Inference controls for statistical databases. IEEE Comput 16(7):69–82
16. Evfimievski AV, Fagin R, Woodruff DP (2008) Epistemic privacy. In: Principles of database systems, PODS 2008. ACM, New York, pp 171–180
17. Farkas C, Jajodia S (2002) The inference problem: a survey. SIGKDD Explorations 4(2):6–11
18. Goguen JA, Mesequer J (1984) Unwinding and inference control. In: IEEE symposium on security and privacy. IEEE, New York, pp 75–87
19. Halpern JY, O'Neill KR (2008) Secrecy in multiagent systems. ACM Trans Inf Syst Secur 12(1):5.1–5.47
20. Hughes D, Shmatikov V (2004) Information hiding, anonymity and privacy: a modular approach. J Comput Secur 12(1):3–36
21. Gray JW III (1991) Toward a mathematical foundation for information flow security. In: IEEE symposium on security and privacy. IEEE, New York, pp 21–35
22. Jajodia S, Sandhu RS (1991) Towards a multilevel secure relational data model. In: Clifford J, King R (eds) SIGMOD conference. ACM Press, New York, pp 50–59
23. Jones AK, Lipton RJ (1975) The enforcement of security policies for computation. In: Symposium on operating system principles, SOSP 75. ACM, New York, pp 197–206
24. Lunt TF, Denning DE, Schell RR, Heckman M, Shockley WR (1990) The SeaView security model. IEEE Trans Softw Eng 16(6):593–621
25. Machanavajjhala A, Kifer D, Gehrke J, Venkitasubramaniam M (2007) L-diversity: privacy beyond k-anonymity. TKDD, 1(1)
26. Mantel H (2002) On the composition of secure systems. In: IEEE symposium security and privacy. IEEE Computer Society Press, New York, pp 88–101
27. Miklau G, Suciu D (2007) A formal analysis of information disclosure in data exchange. J Comput Syst Sci 73(3):507–534
28. Myers AC, Liskov B (2000) Protecting privacy using the decentralized label model. ACM Trans Softw Eng Method 9(4): 410–442
29. Olivier MS, von Solms H (1994) A taxonomy for secure object-oriented databases. ACM Trans Database Syst 19(1):3–46
30. Sandhu RS, Jajodia S (1992) Polyinstantation for cover stories. In: Deswarte Y, Eizenberg G, Quisquater J-J (eds) ESORICS. Lecture notes in computer science, vol 648. Springer, Heidelberg, pp 307–328
31. Traub JF, Yemini Y, Wozniakowski H (1984) The statistical security of a statistical database. ACM Trans Database Syst 9(4):672–679
32. Winslett M, Smith K, Qian X (1994) Formal query languages for secure relational databases. ACM Trans Database Syst 19(4):626–662

Information Assurance

►Levels of Trust

Information Flow and Noninterference

Heiko Mantel
Department of Computer Science, TU Darmstadt, Darmstadt, Germany

Synonyms

Information flow security

Related Concepts

►Covert Channels; ►Information Theory; ►Mandatory Access Control; ►Multilevel Security Policies; ►Refinement Paradox; ►Security Certification; ►Security Policy; ►Side-Channel Analysis; ►Side-Channel Attacks

Definition

Noninterference is a property that restricts the information flow through a system. It can be used to express aspects of confidentiality and integrity.

Background

Goguen and Meseguer introduced noninterference in 1982 as a declarative definition of the property "no illegitimate information flow can occur" for deterministic state machines. Subsequently, numerous security properties were proposed in order to relax the restrictive original definition or to adapt the intuition underlying noninterference to other models of computation.

Theory

A process A is said to be noninterfering with another process B across a system M if A's input to M has no effect on M's output to B. This property implies that no information flows from A to B through M. Noninterference expresses a confidentiality guarantee because if the observations of B are completely independent of the actions of A, then M does not leak anything to B about A's input and A cannot reveal any secrets to B via M. Noninterference also expresses an ►integrity guarantee because if no information flows from A to B through M, then B cannot be corrupted by A via M.

Goguen and Meseguer defined noninterference formally assuming that the system's behavior is specified by a deterministic state machine M. Let M be a tuple $(S, s_0, P, I, O, do, out)$ where S is a set of system states with an initial state $s_0 \in S$, P is a set of processes, I is a set of inputs, O is a set of outputs, $do : (S \times P \times I) \to S$ is a transition function, and $out : (S \times P) \to O$ is an output function. That is, $do(s, p, i)$ is the state that results when M is in state s and process p provides input i to M, and $out(s, p)$ is M's output to p in state s. The reaction of M to a sequence of inputs corresponds to a function $do^* : (S \times (P \times I)^*) \to S$ that is defined inductively by $do^*(s, ()) = s$ and $do^*(s, ((p, i).in)) = do^*(do(s, p, i), in)$ for $s \in S$, $p \in P$, $i \in I$, and $in \in (P \times I)^*$. The definition of noninterference can now be made precise. Given an arbitrary input sequence $in \in (P \times I)^*$, process B must receive the same output no matter whether in occurs or the sequence $del(in, A)$ occurs, which results from in by deleting all pairs with A as first element. This leads to the following definition of "A noninterferes with B across M":

$$\forall in \in (P \times I)^* : out(do^*(s_0, in), B) = out(do^*(s_0, del(in, A)), B)$$

Noninterference can be adapted to other models of computation than deterministic state machines. However, this adaptation may result in more than one sensible property, in particular, if the model admits nondeterministic behavior. In the so-called determinism-based approach to defining noninterference for nondeterministic models, one requires that, for any given input by B, the output of M to B must be identical for all possible inputs by A in all runs that are possible for M given A's and B's inputs. In contrast, noninterference is defined as a closure property on the set of possible runs of M in the so-called possibilistic approach. For instance, noninference is a possibilistic variant of noninterference requiring that for each possible run of M there must be at least one possible run of M that is indistinguishable from B's perspective and in which A provides no input. Many further possibilistic variants of noninterference were proposed, and this variety motivated studies that revealed some surprisingly intriguing advantages and disadvantages of these properties. In the so-called probabilistic approach, one additionally requires that the probabilities of possible runs must match in some well-defined way. Instead of restricting the flow of information through a system M by a lack-of-dependency property like noninterference, one can view M as a communication channel from A to B and apply concepts from ►information theory to restrict the bandwidth of this channel.

►Security policies for noninterference and its variants may be more complex than a single noninterference requirement between two processes A and B, and, moreover, such policies may be formulated at other levels of abstraction than in terms of processes. For instance, a flow policy is a pair $(\mathcal{D}, \rightsquigarrow)$ where $\mathcal{D}$ is a set of so-called security domains and $\rightsquigarrow \subseteq \mathcal{D} \times \mathcal{D}$ is a so-called interference relation. Security domains are abstract entities like, for example, the security levels "top secret," "secret," and "unclassified," without any inherent connection to a particular system. The connection to the system under consideration is explicitly specified by a mapping, the so-called domain assignment, that assigns a security domain to each relevant system entity. Relevant system entities need not be processes, but could be finer grained like, for example, I/O actions, memory sections, or program variables. Naturally, this requires a definition of noninterference that is suitable for the chosen granularity. The interference relation $\rightsquigarrow$ specifies the permissible flow of information. Consequently, if $D \rightsquigarrow D'$ does not hold for two security domains D and D' then "D noninterferes with D'" must hold for the given system and domain assignment for a suitable definition of noninterference.

A ►multi-level security policy is the special case of a flow policy with an interference relation $\rightsquigarrow$ that is a partial order on $\mathcal{D}$. Strict multi-level security is too restrictive for systems that reveal data that is generated from sensitive information, like, for example, the result of a password-based authentication test or the cipher-text that results from encrypting a secret. Traditionally, security models enable such exceptions by allowing some processes to declassify information, even if this violates the multi-level security policy. These so-called trusted processes can be avoided in security models by using a weakened noninterference property that is capable of permitting exceptions to a multi-level security policy. This approach offers more fine-grained control over declassification than using trusted processes, and there are variants

of noninterference that control what may be declassified, where declassification may occur, and who may initiate a declassification.

Applications

Noninterference and its descendants can be employed for expressing confidentiality and integrity guarantees. Hence, these properties can be used in a ►security certification that requires a formal security model. Noninterference-like properties are not limited to a specific system layer and, in particular, can be used for application programs as well as for system-level software.

Unwinding is a verification technique that can be used to simplify the proof that a given system satisfies noninterference under a given policy. Compositional verification is possible, but only some variants of noninterference are preserved under composition, in general. When refining an abstract system specification, noninterference-like properties might not be preserved, even if other system properties are. This is known as the ►refinement paradox.

Noninterference-like properties can also be employed to declaratively capture what guarantees are enforced by a security analysis. In language-based security, for instance, such properties are often used as a soundness criterion for static program analysis techniques.

Open Problems

A key challenge is how to engineer software systems such that they have secure information flow by construction. Due to the refinement paradox, this requires a deeper understanding of the interplay between software engineering and information flow security. Several fundamental issues need further investigation. Firstly, a clarification is needed of how a given security mechanism contributes to the enforcement of a system-wide, declaratively specified security property such as noninterference. Currently, there is a substantial conceptual gap between guarantees provided by security mechanisms and system-wide information flow security. Secondly, usable criteria need to be identified that allow one to determine when a combination of multiple security mechanisms is sufficient to guarantee that the overall system has secure information flow. That is, the traditional mechanism-centered approach to security engineering needs to be complemented by a property-centered approach. Thirdly, notions of abstraction, composition, decomposition, and refinement need to be developed that are suitable thinking tools for software engineering as well as for achieving information flow security. Currently, software engineering does not respect information flow security, leading to undetected possibilities for illegitimate information flow, ►covert channels, and ►side channels. Fourthly, formally verified guarantees about information flow security need to be captured in certificates that provide reliable assurance, but that are also comprehensible. Otherwise, customers cannot assess the advantages of the security of one system over another.

Recommended Reading

1. Focardi R, Gorrieri R (1995) A taxonomy of security properties for process algebras. J Comput Secur 3(1):5–34
2. Goguen JA, Meseguer J (1982) Security policies and security models. In: Proceedings of the IEEE symposium on security and privacy, Oakland, pp 11–20
3. Gray III JW (1991) Toward a mathematical foundation for information flow security. In: Proceedings of the IEEE symposium on security and privacy, Oakland, pp 21–34
4. Mantel H (2001) Preserving information flow properties under refinement. In: Proceedings of the IEEE symposium on security and privacy, Oakland, pp 78–91
5. Mantel H (2002) On the composition of secure systems. In: Proceedings of the IEEE symposium on security and privacy, Oakland, pp 88–104
6. Mantel H (2005) The framework of selective interleaving functions and the modular assembly kit. In: Proceedings of the ACM workshop on formal methods for security engineering: from specifications to code, Alexandria, pp 53–62
7. McLean JD (1996) A general theory of composition for a class of "possibilistic" properties. IEEE Trans Softw Eng 22(1):53–67
8. Roscoe AW, Woodcock JCP, Wulf L (1994) Non-interference through determinism. In: Proceedings of the European symposium on research in computer security. Lecture notes in computer science, vol 875. Springer, Berlin, pp 33–53
9. Rushby JM (1992) Noninterference, transitivity, and channel-control security policies. Technical report CSL-92-02. SRI International, Menlo Park, Dec 1992

Information Flow Security

►Information Flow and Noninterference

Information Integrity

►Authentication, From an Information Theoretic Perspective

Information Security Management System

►ISMS: A Management Framework for Information Security

Information Theoretic Model

►Shannon's Model

Information Theory

Friedrich L. Bauer
Kottgeisering, Germany

Related Concepts

►Cryptology (Classical); ►Shannon's Model

The *entropy* function $H(X)$ is a measure of the uncertainty of X, in formula

$$H(X) = -\sum_{a:p_X(a)>0} p_X(a) \cdot \log_2 p_X(a),$$

where $p_X(a) = \Pr[x = A]$ denotes the probability that random variable X takes on value a. The interpretation is that with probability $p_X(a)$, X can be described by $\log_2 p_X(a)$ bits of information.

The *conditional entropy* or *equivocation* (Shannon, 1949) $H(X/Y)$ denotes the uncertainty of X provided Y is known:

$$H(X|Y) = -\sum_{a,b:\, p_{X|Y}(a|b)>0} p_{X,Y}(a,b) \cdot \log_2 p_{X|Y}(a|b)$$

where $p_{X,Y}(a,b) =_{\text{def}} \Pr[(X = a) \wedge (Y = b)]$ and $p_{X|Y}(a|b)$ obeys Bayes' rule for conditional probabilities:

$$p_{X,Y}(a,b) = p_Y(b) \cdot p_{X/Y}(a|b), \text{ thus}$$
$$-\log_2 p_{X,Y}(a,b) = -\log_2 p_Y(b) - \log_2 p_{X|Y}(a|b).$$

The basic relation on conditional entropy follows from this:

$$H(X,Y) = H(X|Y) + H(Y)$$

In particular, it is to be noted that the entropy is additive if and only if X and Y are independent:

$$H(X,Y) = H(X) + H(Y),$$

in analogy to the additive entropy of thermodynamical systems.

The *redundancy* of a text is that part (expressed in bits) that does not carry information. In common English, the redundancy is roughly 3.5 [bit/char], the information is roughly 1.2 [bit/char], redundancy and information sum up to 4.7 = $\log_2$ 26 [bit/char].

Three possible properties of a cryptosystem are now described using this terminology. A cryptosystem is *of Vernam type* if $H(K) = H(C)$, where $H(K)$ is the entropy of the key K and $H(C)$ is the entropy of the ciphertext C. A cryptosystem has *independent key* if the plaintext P and keytext K are mutually independent: $H(P) = H(P|K)$ and $H(K) = H(K|P)$ ("knowledge of the keytext does not change the uncertainty of the plaintext, and knowledge of the plaintext does not change the uncertainty of the keytext").

A cryptosystem is called *perfect* if plaintext and ciphertext are mutually independent: $H(P) = H(P|C)$ and $H(C) = H(C|P)$ ("knowledge of the ciphertext does not change the uncertainty of the plaintext, and knowledge of the plaintext does not change the uncertainty of the ciphertext"). This means that the security of the system depends entirely on the key; perfect cryptosystems correspond to holocryptic keytexts (►Key), which are extremely difficult to achieve in practice.

Shannon's Main Theorem

In a cryptosystem, where the key character is uniquely determined by the plaintext character and the ciphertext character ("ciphertext and plaintext together allow no uncertainty on the keytext"), any two of the following three properties of the cryptosystem imply the third one:

Vernamtype, independentkey, perfect.

The *unicity distance* for a given language, a given cryptosystem, and a given cryptanalytic procedure of attack is the minimal length of the plaintext such that decryption is unique. Example: let Z be the cardinality of keytext space, assume simple *substitution* (►Substitutions and Permutations) and an attack by letter frequency. Then for English with an alphabet of 26 letters, the unicity distance U is given by:

(1) $U \approx \frac{1}{0.53} \log_2 Z$ for decryption with single-letter frequencies
(2) $U \approx \frac{1}{1.2} \log_2 Z$ for decryption with bigram frequencies
(3) $U \approx \frac{1}{1.5} \log_2 Z$ for decryption with trigram frequencies
(w) $U \approx \frac{1}{2.1} \log_2 Z$ for decryption with word frequencies
(*) $U \approx \frac{1}{3.5} \log_2 Z$ for decryption using all grammatical and semantical rules

For simple substitution with $Z = 26!$, one has $\log_2 Z \approx 88.38$. This leads to the values 167, 74, 59, 42, and 25 for the unicity distance, which are confirmed by practical experience.

For bigram substitution with $Z = 676!$, there is $\log_2 Z \approx 5,385.76$ and $U \approx 1,530$ for decryption using all grammatical and semantical rules.

The situation is rather similar for German, French, Italian, Russian, and related Indo-European languages.

For holocryptic sequences of key elements, the unicity distance is infinite.

Recommended Reading

1. Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin
2. McEliece RJ (1977) The theory of information and coding. In: Encyclopedia of mathematics and its applications, vol 3. Addison-Wesley, Reading

Insider Threat Defense

Salvatore J. Stolfo, Brian M. Bowen, Malek Ben Salem
Department of Computer Science, Columbia University, New York, NY, USA

Related Concepts

►Behavior Profiling; ►Decoy Technology

Definition

Insider Threat can be defined by the vulnerability created from those within the traditional security perimeter. Furthermore, insiders can be divided amongst two non-mutually exclusive classes: *Masqueraders* (attackers who impersonate another system user) and *Traitors* (attackers using their own legitimate system credentials) who each have varying levels of knowledge. The masquerader is presumed to have less knowledge of a system than the victim user whose credentials were stolen. The distinction between these two classes is made primarily to construct threat models [3].

Background

The complexity and potential cost of the malicious insider problem makes it a significant challenge for organizations. Although the scope of the problem extends beyond computing, the vastness of the cyber domain provides a fertile environment for exploitation by insiders. The ideal defense strategy is to deploy systems that prevent insider attack. However, prevention-oriented methods such as policy-based mechanisms and access control systems have been the subject of study for quite some time but have not succeeded in eliminating the problem. Monitoring, detection, and mitigation technologies are realistic necessities that are summarized in this chapter.

Theory

A great deal of research has been conducted exploring the use of behavior monitoring to detect insider attacks. The general approach of behavior-monitoring techniques requires establishing a baseline of normal user behavior by profiling user actions. Subsequent monitoring for behaviors that deviate from this baseline can be used to signal a potential insider attack. User profiling techniques can be useful for detecting masqueraders. Because they may have stolen a user's credentials, they are unlikely to have the ability to assume a user's behavior. On the other hand, detecting traitors with user profiling techniques is far more challenging because traitors may exhibit normal behavior, yet perform malicious acts [7].

An example of a profiling technique that has been explored relies on modeling user *search* behavior on host systems. To demonstrate this, host-based sensors have been designed to monitor user-initiated events. The sensor functions by first building a baseline of normal search behavior for a user. It then monitors for abnormal file search behaviors that may indicate the presence of a masquerader.

While user profiling is most suitable for detecting masqueraders, the use of deception, or decoys, plays a valuable role in the protection of systems, networks, and information particularly from traitors . The first use of decoys in the cyber domain has been credited to Stoll [8]. Stoll's methods included the use of bogus networks, systems, and documents to gather intelligence on the attackers, who were apparently seeking state secrets. Among the many techniques described, he crafted "bait" files, bogus classified documents that contained nonsensitive government information, and attached "alarms" to them so that he would know if anyone accessed them. The decoy system described here builds on that notion increasing the scope, scale, and automation of decoy generation and monitoring.

Deception-based information resources that have no production value other than to attract and detect adversaries (like those used by Stoll) are commonly known as honeypots. Honeypots serve as effective tools to gather intelligence to understand how attackers operate. Honeypots are considered to have low false positive rates since they are designed to capture only malicious attackers, except for occasional mistakes by innocent users.

In order to create decoys to bait insiders with various levels of knowledge and maximize the deception they induce, one must understand the core properties of a decoy. These properties guide the design of systems that automate the generation and placement of trap-based decoys. The properties include *conspicuousness, enticement, non-interference, variability, differentiable, detectability,* and *believability* [2].

A good decoy should make it difficult for an adversary to discern whether they are looking at an authentic document from a legitimate source or if they are looking at a decoy. For concreteness, the definition of "perfect secrecy" proposed in the cryptographic community is built upon to define a "perfect decoy" to be a decoy that is completely indistinguishable from one that is not. One approach used in creating decoys relies on a document marking scheme in which all documents contain embedded markings such that decoys are tagged with Hash-based Message Authentication Codes (HMACs) and non-decoys are tagged with indistinguishable randomness. Here, the challenge of distinguishing decoys reduces to the problem of distinguishing between pseudorandom and random numbers, a task proven to be computationally infeasible under certain assumptions about the pseudorandom generation process. Hence, these are examples of perfect decoys and the only attacker capable of distinguishing them is one with the key, perhaps the highly privileged insider.

Experimental Results

Insider attack research is made difficult due to the lack of readily available insider attackers or a complete set of realistic data they generate. For this reason, researchers must resort to generating their own data that simulates insider attacks. The Schonlau dataset [5] is the most widely used for academic study. It consists of sequences of 15,000 UNIX commands generated by 50 users with different job roles, but the data does not include command arguments or timestamps. The data has been used for comparative evaluations of different supervised machine learning algorithms. The Schonlau data is not a "true Masquerade" dataset: the data gathered from different users were randomly mixed to simulate a masquerader attack, making the dataset perhaps more suitable for "author identification" studies. An alternative approach to acquire sufficient data for evaluating monitoring and detection techniques is to devise a process to acquire human user data under normal operation as well as simulated attack data where "red team" users are tasked to behave as inside attackers. This type of study is typically subject to Institutional Review Board approvals since human subjects are involved. The process is costly, in time and effort, but is sensible and appropriate to protect personally identifiable data of individual volunteer subjects. This was the approach taken by Maloof et al. for evaluating ELICIT [4]. In order to evaluate the search behavior–based user profiling technique, a user study was conducted, where data from 48 users was gathered by distributing host sensors that upload system event data during normal system use. The volunteers, all CS students at Columbia University, have therefore a common "role" in the organization, and hence variations in the user behavior and their data are not attributable to different job functions as is undoubtedly the case with the Schonlau dataset. Data was also gathered from 14 paid volunteers who emulated masquerade attacks on lab equipment. The dataset, which is called RUU (Are You You?) dataset, is over 10 GB and is available to legitimate researchers for download [1]. The data collected for each user averages about 7 days of normal system use, ranging in the extreme between 1 day and 19 days, and an average of more than 1 million records per user.

In the experiment, the 14 masqueraders had no prior access to the file system of a lab machine, which was designed to look very realistic and to include potentially interesting patent applications, personally identifiable information, as well as account credentials stored in various files. The students were provided a scenario where they were asked to perform a specific task, which consisted of finding any data on the file system that could be used for financial gain. The features used for modeling were in essence volumetric statistics characterizing search volume and velocity, and describing the overall computer session in terms of the number of Internet browsing-related actions, the number of of development- and programming-related actions, the number of desktop games-related events, etc. A one-class Support Vector Machine (ocSVM) model was then trained for each user using those features. The same features were extracted from test data after dividing them into 60-s epochs. The ocSVM models were tested against these features, and a threshold was used to determine whether the user activity during the 60-s epochs was normal or abnormal. If the user activity was performed by the normal user, but was classified as abnormal by the ocSVM model, a false positive is recorded. The results using the collected data and this modeling approach show that masquerade activity can be detected with 100% accuracy and with a false positive rate of 0.13% [6].

Experiments involving deception or trap-based detection methods involved the creation of "decoy documents" with embedded HMACs. The HMAC key is kept secret and managed by the decoy system, where it is also associated with a particular registered host. Since the system depends on all documents being tagged, another component inserts

random decoy markers in non-decoy documents, making them indistinguishable from decoys without knowledge of the secret key.

A host sensor that detects malicious activity by monitoring user actions directed at HMAC-embedded decoy documents was built to evaluate the baiting technique. When a decoy document is opened or copied by any application or process, an alert is issued.

The decoy host sensor was tested on a Windows XP machine. A total of 108 decoy PDF documents generated through the decoy system were embedded in the local file system. Markers containing randomness in place of HMACs were embedded in another 2,000 normal PDF files on the local system. Any attempt to load a decoy file in memory was recorded by the sensor including content or metadata modification, as well as any attempt to print, zip, or unzip the file.

The sensor detects the loading of decoy files in memory with 100% accuracy by validating the HMAC value in the PDF files. However, it was discovered during validation tests that detection is susceptible to false positive rates. The problem encountered during testing was created by antivirus scans of the file system! The file accesses of the scanning process that touched a large number of files, resulted in the generation of a number of decoy alerts. A solution was engineered for this particular problem by filtering alerts generated by automatic antivirus scans and backup processes, but the test demonstrates a fundamental design challenge to architect a security system with potentially interfering competing monitors.

Open Problems

Probably the most dangerous of all attackers is the privileged and highly sophisticated user. Such attackers will be fully aware that the system is baited and will employ sophisticated tools to try to analyze, disable, and avoid decoys entirely. As an example of how defeating this level of threat might be possible, consider the analogy with someone who knows encryption is used (and which encryption algorithm is used), but still cannot break the system because they do not have knowledge of an easy-to-change operational parameter (the key). Likewise, just because someone knows that decoys are used in the system does not mean they should be able to identify them all. Devising hard-to-defeat insider attack detection systems for the most sophisticated and knowledgeable insider remains an open challenge.

The detection of insider threats and insider malfeasance is only part of the story. Mitigation strategies may include a significant effort to evaluate the security policies of an organization that may be warranted once an insider attack has been detected (see [9]). Much effort in large organizations is devoted to mapping the IT network, and defending it primarily from outside attackers. Internal security is typically handled by state-of-the-art authentication technology. An important new area of research involves measurement of the security posture of the entire organization, primarily focused on the credentialed users within that organization. New research is needed to design metrics to evaluate whether users are ill-informed of critical policies, and whether they commit too many serious errors that threaten the security of the entire organization. The insider threat problem requires a far deeper analysis and significant new research before anyone can say with any level of assurance "my organization is secure."

Recommended Reading

1. Bowen BM, Ben Salem M, Hershkop S, Keromytis AD, Stolfo SJ (2009) Designing host and network sensors to mitigate the insider threat. IEEE Secur Priv Mag 7(6):22–29
2. Ben Salem M, Hershkop S, Stolfo SJ (2008) A survey of insider attack detection research. In: Stolfo S, Bellovin S, Hershkop S, Sinclair S, Smith S (eds) Insider attack and cyber security: beyond the hacker. Springer, New York, pp 69–90
3. Stoll C (1988) Stalking the wily hacker. Commun ACM 31(5):484
4. Bowen BM, Hershkop S, Keromytis AD, Stolfo SJ (2009) Baiting inside attackers using decoy documents. In: Proceedings of the 5th international ICST conference on security and privacy in communication networks (SecureComm 2009), Athens, 2009
5. Matthias S (2001) Masquerading user data. http://www.schonlau.net/intrusion.html
6. Maloof MA, Stephens GD (2007) Elicit: a system for detecting insiders who violate need-to-know'. In Kruegel C, Lippmann R, Clark A (eds) Recent advances in intrusion detection (RAID 2007). Springer, Heidelberg, pp 146–166
7. Ben Salem M (2009) RUU dataset: http://www1.cs.columbia.edu/ids/ruu/data/
8. Ben Salem M, Hershkop S, Stolfo SJ (2010) Modeling user search-behavior for masquerade detection. Technical report # cucs-014-10, Columbia University Department of Computer Science, 2010
9. Lawrence-Pfleege S, Predd JB, Hunker J, Bulford C (2010) Insiders behaving badly: addressing bad actors and their actions. IEEE Trans Info Forensics Security Arch 5(1)

Integer Factoring

Arjen K. Lenstra
Laboratory for Cryptologic Algorithms - LACAL, School of Computer and Communication Sciences, École Polytechnique Fédérale de Lausanne, Switzerland

Related Concepts

►Factorization Circuits; ►Number Field Sieve for Factoring; ►Quadratic Sieve

Definition

Integer factoring is the following problem: given a positive composite integer n, find positive integers v and w, both greater than 1, such that $n = v \cdot w$.

Background

Integer factoring is widely assumed to be a hard problem. Obviously, it is not hard for all composites, but composites for which it is believed to be difficult can easily be generated. This belief underlies the security of ►*RSA public-key encryption* and the ►*RSA digital signature scheme*. To the present day, no proof of the difficulty of factoring has been published. This is quite unlike the ►*discrete logarithm problem*, where the difficulty is provable for a generic group [19, 27].

However, this result does not have much practical relevance. In particular it does not say anything about the hardness of computing discrete logarithms in multiplicative groups of finite fields, a problem that is widely regarded as being as hard (or as easy) as integer factoring. On a quantum computer, both problems are easy in the sense that they allow polynomial-time solutions. Given the current state of the art in quantum computer manufacturing, this is not yet considered to be a threat affecting factoring or discrete logarithm–based cryptosystems. Quantum computer factoring is not discussed here.

Theory and Applications

Methods for Integer Factorization

RSA cryptosystems are faster when smaller composites are used, but believed to be more secure for larger ones. Finding the right middle-ground between efficiency and security requirements requires the study of theoretical and practical aspects of the integer factorization methods. Often, two types of integer factoring methods are distinguished: general purpose and special purpose methods. For general purpose methods the factoring effort depends solely on the size of the composite n to be factored. For special purpose methods properties of n (mostly but not always of one of the factors of n) come into play as well. RSA composites are generally chosen in such a way that special purpose methods would be less efficient than general purpose ones. Special purpose methods are therefore hardly relevant for RSA composites. For randomly selected composites, however, special purpose methods are on average very effective. For example, almost 92% of all positive integers have a factor $< 1,000$; if such a factor exists, it will be found very quickly using trial division, the simplest of the special purpose methods (see below). Here the following factoring methods are sketched:

- *Special purpose*: trial division, Pollard's rho method, Pollard's $p - 1$ method and generalizations thereof, ►*elliptic curve method for factoring*
- *General purpose*: Fermat's method and congruence of squares, Dixon's random squares method, continued fraction method (CFRAC), linear sieve, ►*quadratic sieve*, ►*number field sieve for factoring*

For a more complete survey refer to [6] and the references therein.

Establishing Compositeness

►*Fermat's little theorem* says that $a^{p-1} \equiv 1 \bmod p$ if p is a ►*prime number* and a is a positive integer $< p$ (►*modular arithmetic*). Thus, an $a \in \{1, 2, \ldots, n-1\}$ for which $a^{n-1} \not\equiv 1 \bmod n$ would establish the compositeness of n at the cost of a single exponentiation modulo n. The proof of compositeness does not provide any information that may be useful to find a nontrivial factor of n. Also, this type of compositeness proof does not work for all composites, because for some composites $a^{n-1} \equiv 1 \bmod n$ for all a that are coprime to n. There are infinitely many of such composites, the so-called Carmichael numbers [1].

Fermat's little theorem allows an alternative formulation for which the converse is always useful for compositeness testing. Let $n-1 = 2^t \cdot u$ for integers t and u with u odd. If $n > 2$ were prime, then any integer $a \in \{2, 3, \ldots, n-1\}$ satisfies the condition that either $a^u \equiv 1 \bmod n$ or $a^{2^i u} \equiv -1 \bmod n$ for some $i \in \{0, 1, \ldots, t-1\}$. An integer $a \in \{2, 3, \ldots, n-1\}$ for which this condition does not hold is called a "witness to the compositeness of n." For odd composite n at least 75% of the numbers in $\{2, 3, \ldots, n-1\}$ are witnesses to their compositeness [24]. Therefore, it can in general be expected that n's compositeness can be proved at the cost of at most a few exponentiations modulo n, simply by trying elements of at random until a witness has been found. This probabilistic compositeness test is often referred to as the ►*Miller-Rabin probabilistic primality test*. If n itself is randomly selected too (as may happen during the search for a prime number), it is usually faster to establish its compositeness using trial division (see below).

Distinct Factors

Let a be a witness to the compositeness of n, as above. This witness can be used to check that n is not a prime power at negligible additional cost. By squaring the number $a^{2^{t-1}u} \bmod n$ that was last calculated, one calculates $(a^n - a) \bmod n$. If it is zero, then n is not a prime power because the odd parts of the $t + 2$ factors

$$a \cdot (a^u - 1) \cdot \prod_{i=0}^{t-1} (a^{2^i u} + 1)$$

of $a^n - a$ are pairwise relatively prime; actually, in that case one of those $t+2$ factors has a nontrivial factor in common with n, which can easily be found. If $(a^n - a) \bmod n \neq 0$, one verifies that $\gcd(a^n - a, n) = 1$, which shows that n is not a prime power: if n were p^k for a prime p, then $a^p \equiv a \bmod p$ and thus also $a^n \equiv a^{p^k} \equiv a \bmod p$, so that p would divide $a^n - a$.

Repeated Factors

If n is an odd composite and not a prime power, it may still be a proper power of a composite (i.e., $n = m^\ell$ for $m, \ell \in \mathbf{Z}_{>1}$ with m composite) or it may properly contain a square (i.e., $n = m^\ell \cdot w$ for $m, \ell, w \in \mathbf{Z}_{>1}$ with $\gcd(m, w) = 1$). Proper powers can be recognized by approximating ℓth roots of n for $1 \leq \ell \leq \left\lceil \frac{\log n}{\log 3} \right\rceil$ using a standard numerical method such as Newton's method. At present there is in general no better way to find out if n properly contains a square than factoring n.

Trial Division

Trial division up to bound B is the process of checking for all primes $\leq B$ in succession if they divide n, until the smallest prime factor p of n is found or until it is shown that $p > B$. This takes time proportional to $\log n \cdot \min(p, B)$. For randomly selected n trial division can be expected to be very effective. It cannot be recommended to use B larger than, say, 10^6 (with the precise value depending on the relative speeds of implementations) because larger p can be found more efficiently using one of the methods described below.

Pollard's rho Method

Pollard's rho method [21] is based on the ►*birthday paradox*: if $x_0, x_1, x_2, \ldots$ is a random walk on $\mathbf{Z}/p\mathbf{Z}$, then for any p there is a fair probability that $x_i = x_j$ for some indices $i \neq j$ up to about $\sqrt{p}$. Similarly, if $x_0, x_1, x_2, \ldots$ is a random walk on $\mathbf{Z}/n\,\mathbf{Z}$, then for any $p < n$ there is a fair probability for a collision $x_i \equiv x_j \bmod p$ for $i \neq j$ up to about $\sqrt{p}$; if p is an unknown divisor of n, such a collision can be recognized because it implies that p divides $\gcd(n, x_i - x_j)$.

In Pollard's rho method a walk on $\mathbf{Z}/n\mathbf{Z}$ is defined by selecting $x_0 \in \mathbf{Z}/n\mathbf{Z}$ at random and by defining $x_{i+1} = \left(x_i^2 + 1\right) \bmod n$. There is no a priori reason why this would define a random walk on $\mathbf{Z}/n\mathbf{Z}$, but if it does it may reveal the smallest factor p of n after only about $\sqrt{p}$ iterations. At the ith iteration, this would require $i-1$ gcd-computations $\gcd(n, x_i - x_j)$ for $j < i$, making the method slower than trial division. This problem is overcome by means of Floyd's cycle-finding method: at the ith iteration compute just $\gcd(n, x_i - x_{2i})$ (thus requiring computation of not only x_i but x_{2i} as well). As a result, and under the assumption that the walk is random, the expected time to find p becomes proportional to $(\log n)^2 \cdot \sqrt{p}$; this closely matches practical observations. The name of the method is based on the shape of the Greek character rho ("ρ") depicting a sequence that bites in its own tails. The method is related to Pollard's rho method for solving the ►*discrete logarithm problem*.

In practice, the gcd-computation per iteration is replaced by a single gcd-computation of n and the product modulo n of, say, 100 consecutive $(x_i - x_{2i})$'s. In the unlikely event that the gcd turns out to be equal to n, one backs up and computes the gcds more frequently. See also [16].

Pollard's $p-1$ Method

Pollard's $p-1$ method [20] is based on the following observation. It follows from Fermat's little theorem that if a is coprime to a prime p and k is an integer multiple of $p-1$, then $a^k \equiv 1 \bmod p$. Thus, if p is a prime factor of n, then p divides either $\gcd(a, n)$ or $\gcd(a^k - 1, n)$, where a is randomly selected from $\{2, 3, \ldots, n-2\}$. This means that primes p dividing n for which $p-1$ is B-smooth (►*smoothness*) may be found by selecting an integer $a \in \{2, 3, \ldots n-2\}$ at random, checking that $\gcd(a, n) = 1$, and computing $\gcd(a^k - 1, n)$ where k is the product of the primes $\leq B$ and appropriately chosen small powers thereof. This takes time proportional to $(\log n)^2 \cdot B$. In a "second stage" one may successively try $k \cdot q$ as well for the primes q between B and B', thereby finding p for which $p-1$ is the product of a B-smooth part and a single larger prime factor up to B'; the additional cost is proportional to $B' - B$.

For n with unknown factorization, the best values B and B' are unknown too and, in general, too large to make the method practical. However, one may try values B, B' depending on the amount of computing time one finds reasonable and turn out to be lucky; if not, one gives up as far as Pollard's $p-1$ method is concerned. Despite its low probability of success, the method is quite popular, and has led to some surprising factorizations.

Generalizations of Pollard's $p-1$ Method

Pollard's $p-1$ method is the special case $d = 1$ of a more general method that finds a prime factor p of n for which the order $p^d - 1$ of the multiplicative group $\mathbf{F}_{p^d}^*$ of $\mathbf{F}_{p^d}$ is smooth.

Because

$$X^d - 1 = \prod_{t \text{ dividing } d} \phi_t(X),$$

where $\phi_t(X)$ is the tth cyclotomic polynomial (thus, $\phi_1(X) = X - 1$, $\phi_2(X) = \frac{X^2-1}{X-1} = X + 1$, $\phi_3(X) = \frac{X^3-1}{X-1} = X^2 + X + 1$, $\phi_4(X) = \frac{X^4-1}{(X-1)(X+1)} = X^2 + 1$, etc.), the order of $\mathbf{F}_{p^d}^*$ is smooth if and only if $\phi_t(p)$ is smooth for all integers t dividing d. For each t the smoothness test (possibly leading to a factorization of n) consists of an exponentiation in a ring modulo n that contains the order $\phi_t(p)$ subgroup of the multiplicative group of the subfield $\mathbf{F}_{p^t}$ of $\mathbf{F}_{p^d}$. For $t = d = 2$ the method is known as Williams' $p + 1$ method [31]; for general d, it is due to Bach and Shallit [3].

Usage of "Strong Primes" in RSA

It is not uncommon, and even prescribed in some standards, to use so-called ►*strong primes* as factors of RSA moduli. These are primes for which both $p - 1$ and $p + 1$ have a very large prime factor, rendering ineffective a $p - 1$ or $p+1$ attack against the modulus. This approach overlooks other $\phi_t(p)$ attacks (which, for random moduli, have an even smaller probability of success). More importantly it overlooks the fact that the resulting RSA modulus is just as likely to be vulnerable to a single elliptic curve when using the ►*elliptic curve method for factoring*. It follows that usage of strong primes does in general not make RSA moduli more resistant against factoring attacks. See also [25].

Cycling Attacks against RSA

These attacks, also called "superencryption attacks" work by repeatedly re-encrypting an RSA ciphertext, in the hope that after k re-encryptions (for some reasonable k) the original ciphertext appears. They are used as an additional reason why ►*strong primes* should be used in RSA. However, it is shown in [25] that a generalized and more efficient version of cycling attacks can be regarded as a special purpose factoring method that is successful only if all prime factors of $p - 1$ are contained in $e^k - 1$ for one of the primes p dividing n, where e is the RSA *encryption exponent*. The success probability of this attack is therefore small, even when compared to the success probability of Pollard's $p - 1$ method.

Elliptic Curve Method for Factoring

The success of Pollard's $p - 1$ method (or its generalizations) depends on the smoothness of the order of one of the groups $\mathbf{F}_p^*$ (or $\mathbf{F}_{p^d}^*$) with p ranging over the prime factors of n. Given n, the group orders are fixed, so the method works efficiently for some n but for most n it would require too much computing time. In the elliptic curve method [10] each fixed group $\mathbf{F}_p^*$ of fixed order (given n) is replaced by the group E_p of points modulo p of an elliptic curve modulo n. For randomly selected elliptic curves modulo n, the order $\#E_p$ of E_p behaves as a random number close to p. If $\#E_p$ is smooth, then p can efficiently be found using arithmetic in the group of points of the elliptic curve modulo n. It is conjectured that the smoothness behavior of $\#E_p$ is similar to that of ordinary integers of that size (►*smoothness probability*), which implies that the method works efficiently for all n. It also implies that the method can be expected to find smaller factors faster than larger ones. In the worst case where n is the product of two primes of about the same size the heuristic expected runtime is $L_n[1/2, 1]$, with L_n as in ►*L notation*; this is subexponential in $\log(n)$. See ►*Elliptic Curve Method for Factoring* for a more complete description and more detailed expected runtimes.

Fermat's Method and Congruence of Squares

Fermat's method attempts to factor n by writing it as the difference of two integer squares. Let $n = p \cdot q$ for odd p and q with $p < q$, so that $q - p = 2y$ for an integer y. With $x = p + y$ it follows that $n = (x - y)(x + y) = x^2 - y^2$. Thus, if one tries $x = \lceil\sqrt{n}\rceil + 1, \lceil\sqrt{n}\rceil + 2, \lceil\sqrt{n}\rceil + 3, \ldots$ in succession until $x^2 - n$ is a perfect square, one ultimately finds $x^2 - n = y^2$. This is efficient only if the resulting y, the difference between the factors, is small; if it is large, the method is inferior even to trial division.

Integers x and y that satisfy the similar but weaker condition

$$x^2 \equiv y^2 \bmod n$$

may also lead to a factorization of n: from the fact that n divides $x^2 - y^2 = (x - y)(x + y)$ it follows that

$$n = \gcd(n, x - y)\gcd(n, x + y).$$

If x and y are random solutions to $x^2 \equiv y^2 \bmod n$, then there is a probability of at least 50% that this yields a nontrivial factorization of n. All general purpose factoring methods described below work by finding "random" solutions to this equation.

The Morrison–Brillhart Approach

To construct solutions to $x^2 \equiv y^2 \bmod n$ that may be assumed to be sufficiently random, Kraïtchik in the 1920s proposed to piece together solutions to $x^2 \equiv a \bmod n$. In the Morrison–Brillhart approach this is achieved using the following two steps [18]:

Relation collection. Fix a set P of primes (often called the ►*factor base*), and collect a set V of more than $\#P$ integers v such that

$$v^2 \equiv \left(\prod_{p \in P} p^{e_{v,p}}\right) \bmod n, \text{ with } e_{v,p} \in \mathbf{Z}.$$

These identities are often called "relations" modulo n. If P is the set of primes $\le B$, then v's such that v^2 is B-smooth lead to relations. For each v the exponents $e_{v,p}$ are regarded as a $\#P$-dimensional vector, denoted $(e_{v,p})_{p\in P}$.

Relation combination. Because $\#V > \#P$, the $\#P$-dimensional vectors $(e_{v,p})_{p\in P}$ are linearly dependent and there exist at least $\#V - \#P$ linearly independent subsets S of V such that

$$\sum_{v\in S}(e_{v,p})_{p\in P} = 2(s_p)_{p\in P}, \text{ with } (s_p)_{p\in P} \in \mathbf{Z}^{\#P}.$$

These subsets S with corresponding vectors $(s_p)_{p\in P}$ give rise to at least $\#V - \#P$ independent solutions to $x^2 \equiv y^2 \bmod n$, namely

$$x = \left(\prod_{v\in S} v\right) \bmod n, y = \left(\prod_{p\in P} p^{s_p}\right) \bmod n,$$

and thereby at least $\#V - \#P$ independent chances to factor n.

All current general purpose factoring methods are based on the Morrison–Brillhart approach. They differ in the way the relations are collected, but they are all based on, more or less, the same relation combination step.

Matrix Step

Because S and $(s_p)_{p\in P}$, as above, can be found by looking for linear dependencies modulo 2 among the rows of the $(\#V \times \#P)$-matrix $(e_{v,p})_{v\in V, p\in P}$, the relation combination step is often referred to as the "matrix step." With Gaussian elimination the matrix step can be done in $(\#P)^3$ steps (since $\#V \approx \#P$). Faster methods, such as conjugate gradient, Lanczos, or Wiedemann's coordinate recurrence method, require $O(w \cdot \#P)$ steps (►*O notation*), where w is the number of nonzero entries of the matrix $(e_{v,p} \bmod 2)_{v\in V, p\in P}$. See [5, 13, 17, 23, 29, 30] for details.

In the various runtime analyses below, $\#P$ is measured using the ►*L notation* and w turns out to be $c \cdot \#P$ for a c that disappears in the $o(1)$ of the L notation, so that the runtime $O(w \cdot \#P)$ simplifies to $(\#P)^2$.

Dixon's Random Squares Method

The simplest relation collection method is to define P as the set of primes $\le B$ for some bound B and to select different v's at random from $\mathbf{Z}/n\mathbf{Z}$ until more than $\pi(B)$ ones have been found for which $v^2 \bmod n$ is B-smooth. This method is called *Dixon's random squares method* [8]. The choice of B, and the resulting expected runtime, depends on the way the values $v^2 \bmod n$ are tested for B-smoothness. If smoothness is tested using trial division, then $B = L_n[1/2, 1/2]$ (with L_n as in ►*L notation*). For each candidate v, the number $v^2 \bmod n$ is assumed to behave as a random number $\le n = L_n[1,1]$, and therefore, according to ►*smoothness probability*, B-smooth with probability $L_n[1/2, -1]$. Testing each candidate for B-smoothness using trial division takes time $\#P = \pi(B) = L_n[1/2, 1/2]$ (using the properties of L_n as set forth in ►*L notation*), so collecting somewhat more than $\#P$ relations can be expected to take time

$$\overbrace{L_n[1/2,1/2]}^{\text{number of relations to be collected}} \cdot \overbrace{L_n[1/2,1/2]}^{\text{trial division}} \cdot \overbrace{(L_n[1/2,-1])^{-1}}^{\text{inverse of smoothness probability}} = L_n[1/2,2].$$

Gaussian elimination on the $\#V \times \#P$ matrix takes time

$$L_n[1/2,1/2]^3 = L_n[1/2,3/2].$$

Because at most $\log_2(n)$ entries are non-zero for each vector $(e_{v,p})_{p\in P}$, the total number of non-zero entries of the matrix is $\#V \cdot \log_2(n) = L_n[1/2,1/2]$ and the matrix step can be done in

$$L_n[1/2,1/2]^2 = L_n[1/2,1]$$

steps using Lanczos or Wiedemann algorithms. In either case the runtime is dominated by relation collection and the total expected time required for Dixon's method with trial division is $L_n[1/2,2]$. Unlike most methods described below, the expected runtime of the trial division variant of Dixon's method can rigorously be proved, i.e., it does not depend on any heuristic arguments or conjectures.

If B-smoothness is tested using the elliptic curve method, the time to test each $v^2 \bmod n$ is reduced to $L_n[1/2,0]$: the entire cost of the smoothness tests disappears in the $o(1)$. As a result the two stages can be seen to require time $L_n[1/2,3/2]$ each when Gaussian elimination is used for the matrix step. In this case, i.e., when using the elliptic curve method for smoothness testing, the runtime can be further reduced by using Lanczos or Wiedemann methods and a different value for B. Redefine B as $L_n[1/2, \sqrt{1/2}]$ so that relation collection takes time

$$L_n[1/2,\sqrt{1/2}] \cdot L_n[1/2,0] \cdot (L_n[1/2,-\sqrt{1/2}])^{-1} = L_n[1/2,\sqrt{2}]$$

and the matrix step requires $L_n[1/2,\sqrt{1/2}]^2 = L_n[1/2,\sqrt{2}]$ steps. The overall runtime of Dixon's method becomes

$$L_n[1/2,\sqrt{2}] + L_n[1/2,\sqrt{2}] = L_n[1/2,\sqrt{2}];$$

asymptotically relation collection and combination are equally expensive. As described here, the expected runtime of this elliptic curve–based variant of Dixon's method

depends on the conjecture involved in the expected runtime of the elliptic curve method. It is shown in [22], however, that the expected runtime of a variant of the elliptic curve smoothness test can rigorously be proved. That leads to a rigorous $L_n[1/2, \sqrt{2}]$ expected runtime for Dixon's method.

Continued Fraction Method (CFRAC)

The ▶*quadratic residues* $v^2 \bmod n$ in Dixon's method provably behave with respect to smoothness probabilities as random non-negative integers less than n. That allows the rigorous proof of the expected runtime of Dixon's method. However, this theoretical advantage is not a practical concern. It would be preferable to generate smaller quadratic residues, thereby improving the smoothness chances and thus speeding up relation collection, even though it may no longer be possible to rigorously prove the expected runtime of the resulting method. The earliest relation collection method where quadratic residues were generated that are substantially smaller than n was due to Morrison and Brillhart [18] and is based on the use of continued fractions; actually, this method (dubbed "CFRAC") predates Dixon's method.

If a_i/b_i is the ith continued fraction convergent to $\sqrt{n}$, then $|a_i^2 - nb_i^2| < 2\sqrt{n}$. Thus, if v is chosen as a_i for $i = 1, 2, \ldots$ in succession, then $v^2 \bmod n = a_i^2 - nb_i^2$ is a quadratic residue modulo n that is $< 2\sqrt{n}$ and thus much smaller than n. In practice this leads to a substantially larger smoothness probability than in Dixon's method, despite the fact that if prime p divides $v^2 \bmod n$, then $(a_i/b_i)^2 \equiv n \bmod p$ so that n is a quadratic residue modulo p. With $B = L_n[1/2, 1/2]$, P the set of primes $p \le B$ with $\left(\frac{n}{p}\right) = 1$, and elliptic curve smoothness testing, the heuristic expected runtime becomes $L_n[1/2, 1]$. The heuristic is based on the assumption that the residues $v^2 \bmod n$ behave, with respect to smoothness properties, as ordinary random integers $\le n$ and that the set of primes $p \le B$ for which $\left(\frac{n}{p}\right) \ne 1$ does not behave unexpectedly. In that case, when the L *notation* is used to express smoothness probabilities, the difference with truly random integers disappears in the $o(1)$.

In [11] it is shown how this same expected runtime can be achieved rigorously (by a method that is based on the use of class groups). If elliptic curve smoothness testing is replaced by trial division, $B = L_n[1/2, \sqrt{1/8}]$ is optimal and the heuristic expected runtime becomes $L_n[1/2, \sqrt{2}]$.

Note on the Size of RSA Moduli

In the mid 1970s, CFRAC (with trial division based smoothness testing) was the factoring method of choice. Strangely, at that time, no one seemed to be aware of its (heuristic) subexponential expected runtime $L_n[1/2, \sqrt{2}]$. Had this been known by the time the RSA challenge [9] was posed, Ron Rivest may have based his runtime estimates on CFRAC instead of Pollard's rho (with its exponential expected runtime) [26], come up with more realistic estimates for the difficulty of factoring a 129-digit modulus, and could have decided that 129 digits were too close for comfort (as shown in [2]). As a result, 512-bit RSA moduli may have become less popular.

Linear Sieve

It was quickly realized that the practical performance of CFRAC was marred by the trial division based smoothness test. In the late 1970s Richard Schroeppel therefore developed a new way to generate relatively small residues modulo n that can be tested for smoothness very quickly: look for small integers i, j such that

$$f(i,j) = (i + [\sqrt{n}])(j + [\sqrt{n}]) - n \approx (i + j)\sqrt{n}$$

is smooth. Compared to CFRAC the residues are somewhat bigger, namely $(i + j)\sqrt{n}$ as opposed to $2\sqrt{n}$. But the advantage is that smoothness can be tested for many i, j simultaneously using a sieve (▶*sieving*): if p divides $f(i,j)$ then p divides $f(i + kp, j + \ell p)$ for any $k, \ell \in \mathbf{Z}$. This means that if $f(i,j)$ is tested for B-smoothness for $0 \le i < I$ and $0 \le j < J$, the smoothness tests no longer take time $I \cdot J \cdot \pi(B) \approx I \cdot J \cdot B/\log B$, but

$$\sum_{p \le B} \sum_{0 \le i < I} \sum_{0 \le j < J} \frac{1}{p} = O(I \cdot J \cdot \log\log(B)).$$

This leads to a heuristic expected runtime $L_n[1/2, 1]$. Inconveniently, $(i + [\sqrt{n}])(j + [\sqrt{n}])$ is not automatically a square, which means that for all values $i + [\sqrt{n}]$ and $j + [\sqrt{n}]$ that occur in smooth $f(i,j)$'s columns have to be included in the matrix. The effect this has on the expected runtime disappears in the $o(1)$ in L_n.

This method, dubbed "linear sieve," was the first factoring method that was heuristically shown (by Schroeppel) to have subexponential expected runtime. (That the earlier CFRAC also had subexponential expected runtime was realized only later; see also [12].) Its main historical significance is, however, that it led to the ▶*Quadratic Sieve*, for many years the world's most practical factoring method.

Quadratic Sieve

The first crude version of the quadratic sieve was due to Carl Pomerance who realized that it may be profitable to take $i = j$ in Schroeppel's linear sieve. Although smoothness could still be tested quickly using a sieve and the heuristic expected runtime (with sieving) turned out to

be a low $L_n[1/2,1]$, in practice the method suffered from deteriorating smoothness probabilities (due to the linear growth of the quadratic residue $f(i,j)$). This problem was, however, quickly overcome by Jim Davis and Diane Holdridge which led to the first factorization of a number of more than 70 decimal digits [7]. Since then the method has been embellished in various ways (most importantly by Peter Montgomery's multiple polynomial version, as described in [28]) to make it even more practical. See ►*Quadratic Sieve* for details. At this point the largest factorization obtained using quadratic sieve is the 135-digit factorization reported in [15].

Number Field Sieve

Until the late 1980s the best factoring methods, including the most practical one (quadratic sieve), shared the same expected runtime $L_n[1/2,1]$ despite the fact that the underlying mathematics varied considerably: heuristically for quadratic and linear sieve, CFRAC, and the worst case of the elliptic curve method, and rigorously for the class group method from [11]. This remarkable coincidence fostered the hope among users of the RSA cryptosystem that $L_n[1/2,1]$, halfway between linear time $\log n$ and exponential time n (►*L notation*), is the "true" complexity of factoring.

The situation changed, slowly, when in late 1988 John Pollard distributed a letter to a handful of colleagues. In it he described a novel method, still based on the Morrison–Brillhart approach, to factor integers close to a cube and expressed his hope that, one day, the method may be used to factor the ninth Fermat number $F_9 = 2^{2^9} + 1$, back then the world's "most wanted" composite. It was quickly established that for certain "nice" n Pollard's new method should work in heuristic expected runtime $L_n\left[1/3, \left(\frac{32}{9}\right)^{1/3}\right] \approx L_n[1/3, 1.526]$. This was the first indication that, conceivably, the complexity of factoring would not be stuck at $L_n[1/2,\ldots]$. The initial work was soon followed by the factorization of several large "nice" integers, culminating in 1990 in the factorization of F_9 [14]. Further theoretical work removed the "niceness" restriction and led to the method that is now referred to as the "number field sieve": a general purpose factoring method with heuristic expected runtime $L_n\left[1/3, \left(\frac{64}{9}\right)^{1/3}\right] \approx L_n[1/3, 1.923]$. The method as it applies to "nice" numbers is now called the "special number field sieve." See ►*Number Field Sieve For Factoring* for details. The first time a 512-bit RSA modulus was factored, using the number field sieve, was in 1999 [4].

With hindsight, the property that all $L_n[1/2,1]$ factoring methods have in common is their dependence, in one way or another, on smoothness of numbers of order $n^{O(1)}$. The number field sieve breaks through the $n^{O(1)}$ barrier and depends on smoothness of numbers of order $n^{o(1)}$.

Recommended Reading

1. Alford WR, Granville A, Pomerance C (1994) There are infinitely many Carmichael numbers. Ann Math 193(3): 703–722
2. Atkins D, Graff M, Lenstra AK, Leyland PC (1995) The magic words are squeamish ossifrage. In: Pieprzyk J, Safavi-Naini R (eds) In: Advances in cryptology: ASIACRYPT'94, proceedings of the 4th international conference on the theory and applications of cryptology, Wollongong, Australia, 28 November–1 December, 1994. Lecture notes in computer science, vol 917. Springer, Berlin, 1995, pp 263–277
3. Bach E, Shallit J (1989) Factoring with cyclotomic polynomials. Math Comput 52:201–219
4. Cavallar S, Dodson B, Lenstra AK, Lioen WM, Montgomery PL, Murphy B, te Riele HJJ, Aardal K, Gilchrist J, Guillerm G, Leyland PC, Marchand J, Morain F, Muffett A, Putnam C, Putnam C, Zimmermann P (2000) Factorization of a 512-bit RSA modulus. In: Preneel B (ed) Advances in cryptology: EUROCRYPT 2000, proceedings of the international conference on the theory and application of cryptographic techniques, Bruges, Belgium, 14–18 May 2000. Lecture notes in computer science vol 1807. Springer, Berlin, 2000, pp 1–18
5. Coppersmith D (1994) Solving homogeneous linear equations over GF(2) via block Wiedemann algorithm. Math Comput 62:333–350
6. Crandall RE, Pomerance C (2001) Prime numbers: a computational perspective. Springer, Berlin
7. Davis JA, Holdridge DB (1984) Factorization using the quadratic sieve algorithm. In: Chaum D (ed) Advances in cryptology: Crypto 83. Plenum, New York, pp 103–113
8. Dixon JD (1981) Asymptotically fast factorization of integers. Math Comput 36:255–260
9. Gardner M (1977) A new kind of cipher that would take millions of years to break. Sci Am 237:120–124
10. Lenstra HW Jr (1987) Factoring integers with elliptic curves. Ann Math 126:649–673. URL: http://links.jstor.org/sci?sici=0003-486X(198711)2:126:3<649:FIWC>2.0.CO;2-V
11. Lenstra HW Jr, Pomerance C (1992) A rigorous time bound for factoring integers. J Am Math Soc 5: 483–516. URL: http://links.jstor.org/sci?sici=0894-0347(199207)5:3<483:ARTBFF>2.0.CO;2-S
12. Knuth DE (1997) The art of computer programming: seminumerical algorithms, vol 2, 3rd edn. Addison-Wesley, Reading
13. LaMacchia BA, Odlyzko AM (1991) Solving large spare linear systems over finite fields. In: Menezes AJ, Vanstone SA (eds) Advances in cryptology: CRYPTO'90. Lecture notes in computer science, vol 537. Springer, Berlin, 1991, pp 109–133
14. Lenstra AK, Lensta HW Jr, Manasse MS, Pollard JM (1993) The factorization of the ninth Fermat number. Math Comput 61: 319–349
15. Leyland PC, Lenstra AK, Dodson B, Muffett A, Wagstaff SS Jr (2002) MPQS with three large primes. In: Fieker C, Kohel DR (eds) Algorithmic number theory. In: Proceedings of the 5th international symposium, ANTS-V, Sydney, Australia, 7–12 July 2002. Lecture notes in computer science, vol 2369. Springer, Berlin, 2002, pp 446–460
16. Montgomery PL (1987) Speeding the Pollard and elliptic curve methods of factorization. Math Comput 48:243–264. URL:

I

http://links.jstor.org/sici?sici=0025-5718(198701)48:177<243:STPAEC>2.0.CO;2–3
17. Montgomery PL (1995) A block Lanczos algorithm for finding dependencies over GF(2). In; Guillou LC, Quisquater J-J (eds) Advances in cryptology: EUROCRYPT'95, Saint-Malo, 1995. Lecture notes in computer science, vol 921. Springer, Berlin, 1995, pp 106–120
18. Morrison MA, Brillhart J (1975) A method of factoring and the factorization of F_7. Math Comput 29:183–205
19. Nechaev VI (1968) Complexity of a determinate algorithm for the discrete logarithm. Math Notes 55(2):155–172. Translated from Matematicheskie Zametki 55(2): 91–101, (1994). This result dates from 1968
20. Pollard JM (1974) Theorems on factorization and primality testing. Proc Camb Phil Soc 76:521–528
21. Pollard JM (1975) A Monte Carlo method for factorization. BIT 15:331–334
22. Pomerance C (1987) Fast, rigorous factorization and discrete logarithm algorithms. In: Johnson DS, Nishizeki T, Nozaki A, Wilf HS (eds) Discrete algorithms and complexity. Academic Press, Boston, pp 119–143
23. Pomerance C, Smith JW (1992) Reduction of huge, sparse matrices over finite fields via created catastrophes. Exp Math 1:89–94
24. Rabin MO (1980) Probabilistic algorithm for testing primality. J Number Theory 12(1):128–138
25. Rivest R, Silverman R (2001) Are 'strong' primes needed for RSA. Cryptology ePrint Archive, Report 2001/007. http://eprint.iacr.org/
26. Rivest RL (1977) Letter to M. Gardner containing an estimate of the difficulty of factoring a 129-digit modulus using Pollard's rho method
27. Shoup V (1997) Lower bounds for discrete logarithms and related problems. In: Proceedings of EUROCRYPT '97. Lecture notes in computer science, vol 1233, pp 256–266
28. Silverman RD (1987) The multiple polynomial quadratic sieve. Math Comput 48:329–339
29. Villard G (1997) Further analysis of Coppersmith's block Wiedemann algorithm for the solution of sparse linear systems (extended abstract). In: Proceedings of the 1997 international symposium on symbolic and algebraic computation, ISSAC'97, ACM, New York, pp 32–39
30. Wiedemann DH (1986) Solving sparse linear equations over finite fields. IEEE Trans Inf Theory 32:54–62
31. Williams HC (1982). A p + 1 method of factoring. Math Comput 39:225–234

Integrated Circuit

►Levels of Trust

Integrated Circuit Card

►Smart Card

Integrity Model

►Biba Model

Intellectual Property

►Levels of Trust

Interactive Argument

Berry Schoenmakers
Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

Synonyms

Computationally sound proof system

Related Concepts

►Interactive Proof

An *interactive argument* (or *computationally sound proof system*) is a relaxation of an ►interactive proof, introduced in [1]. The difference is that the prover is restricted to be a ►polynomial-time algorithm for an interactive argument, whereas no such restrictions on the prover apply for an interactive proof. The prover's advantage over the verifier is that the prover gets a private input, which allows the prover to perform his or her task in polynomial time (completeness).

The soundness condition for an interactive argument, referred to as *computational soundness*, states as before that executions of the protocol between the prover and the verifier should result in the verifier rejecting the proof, if $x \notin L$ holds; here, the prover is not required to follow the protocol, that is, the prover may behave arbitrarily, but the prover is limited to be a (*probabilistic*) *polynomial-time* algorithm.

Hence, cheating by the prover is not required to be impossible; rather, cheating is required to be *infeasible*. Therefore, interactive arguments are easier to achieve than interactive proofs; in particular, while perfect ►zero-knowledge arguments are known to exist for every language in NP, it is considered unlikely that perfect zero-knowledge proofs exist for every language in NP.

Recommended Reading

1. Brassard G, Chaum D, Crépeau C (1988) Minimum disclosure proofs of knowledge. J Comput Syst Sci 37(2):156–189
2. Goldreich O (2001) Foundations of cryptography – basic tools. Cambridge University Press, Cambridge
3. Naor M, Ostrovsky R, Venkatesan R, Yung M (1998) Zero-knowledge arguments for NP can be based on general assumptions. J Cryptol 11(2):87–108

Interactive Proof

Berry Schoenmakers
Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

Synonyms

Interactive proof systems

Related Concepts

►Interactive Argument

Definition

The notion of an *interactive proof* plays an important role in complexity theory. An interactive proof is a ►protocol between two parties, called the *prover* and the *verifier*. The crucial point is that the verifier is restricted to be a (probabilistic) ►polynomial-time algorithm, whereas no such restriction applies to the prover. By means of an interactive proof the prover convinces the verifier of the validity of a given statement. A statement is of the form $x \in L$, where x is a word and L is a formal language. The interesting languages are those for which no polynomial-time membership tests (are known to) exist. It follows that the verifier cannot determine on its own whether $x \in L$ holds.

Theory

An interactive proof is required to satisfy two conditions. The first condition is *completeness*, which means that executions of the protocol between the prover and the verifier should result in the verifier accepting the proof, if $x \in L$ holds. The second condition is *soundness*, which means that executions of the protocol between the prover and the verifier should result in the verifier rejecting the proof, if $x \notin L$ holds; here, the prover is not required to follow the protocol, that is, the prover may behave arbitrarily.

A simple example of an interactive proof runs as follows. Consider the language L_H consisting of graphs containing a Hamiltonian cycle. It is well known that the problem of determining membership for L_H is NP-complete. Hence, it is supposedly hard to determine whether a given graph contains a Hamiltonian cycle. However, given a purported Hamiltonian cycle for a graph, it is easy to check whether this is indeed the case. An interactive proof for L_H is obtained if the prover simply sends a Hamiltonian cycle for the graph under consideration to the verifier. The conditions of completeness and soundness are clearly satisfied.

In the context of cryptography, interactive proofs are usually required to satisfy some additional conditions. Many interactive proofs are in fact ►proofs of knowledge. Also, ►zero-knowledge proofs are a main example of interactive proofs used for cryptographic purposes, noting that zero-knowledge ►interactive arguments and ►witness hiding protocols are possible alternatives. The above interactive proof for L_H is neither zero-knowledge nor witness hiding, as the prover simply gives away a Hamiltonian cycle.

Recommended Reading

1. Goldwasser S, Micali S, Rackoff C (1989) The knowledge complexity of interactive proof systems. SIAM J Comput 18:186–208. Preliminary version in 17th ACM symposium on the theory of computing, 1982
2. Goldreich O (2001) Foundations of cryptography – basic tools. Cambridge University Press, Cambridge.
3. Shamir A (1992) IP = SPACE. J ACM 39(4):869–877

Interactive Proof Systems

►Interactive Proof

Interactive Theorem Proving and Security

►Theorem Proving and Security

Interception

►Eavesdropping

Interpolation Attack

Christophe De Cannière
Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

Related Concepts

►Block Ciphers

The *interpolation attack* is a technique for attacking ►block ciphers built from simple algebraic functions. It was introduced by Jakobsen and Knudsen [2, 3] in 1997 and applied to variants of SHARK, a predecessor of ►Rijndael/AES.

The attack is based on a well-known principle: given an unknown polynomial $y = f(x)$, if the degree of $f(x)$ does not exceed $n - 1$, then its coefficients can efficiently be recovered by taking n distinct samples (x_i, y_i) with $y_i = f(x_i)$. The *Lagrange interpolation formula* provides a general expression for the polynomial reconstructed this way:

$$f(x) = \sum_i y_i \prod_{j \neq i} \frac{x - x_j}{x_i - x_j}.$$

This mathematical property has interesting implications when considering a block cipher with a fixed but unknown secret key. If the ciphertext is written as a polynomial (with unknown coefficients) of the plaintext, and if the degree of this polynomial is sufficiently low, then a limited number of plaintext–ciphertext pairs suffice to completely determine the encryption function. This allows the attacker to encrypt and decrypt data blocks for the unknown key without doing any key-recovery.

An interesting property of the basic interpolation attack is that it is not affected by the internal structure of the cipher, apart from the degree of the polynomial representing the encryption function. In fact, a low degree is not strictly necessary for an efficient attack; it suffices that the number of unknown coefficients is sufficiently small, and this happens to be the case for a number of ciphers which were optimized against linear and differential attacks (for example, the $\mathcal{KN}$ cipher by Knudsen and Nyberg).

The attack outlined above can be extended and generalized in many ways. It can, for example, also be applied by only expressing a part of the ciphertext as a function of the plaintext or by constructing an implicit polynomial expression involving parts of the plaintext and the ciphertext. The latter could be derived from a rational expression or obtained by applying a ►meet-in-the-middle attack. Furthermore, in a subsequent paper [1], Jakobsen demonstrated that the interpolation ideas can still be applied when the polynomials are probabilistic. The method is based on Sudan's algorithm, designed to decode Reed–Solomon codes (►Cyclic Codes). Finally, note that all attacks described above can easily be turned into key-recovery attacks by guessing the last round key of the cipher and checking the correctness of the guess by applying the interpolation attack on the remaining rounds.

Recommended Reading

1. Jakobsen T (1998) Cryptanalysis of block ciphers with probabilistic non-linear relations of low degree. In: Krawczyk H (ed) Advances in cryptology – CRYPTO?98. Lecture notes in computer science, vol 1462. Springer, Berlin, pp 212–223
2. Jakobsen T, Knudsen LR (1997) The interpolation attack on block ciphers. In: Biham E (ed) Proceedings of fast Software encryption – FSE?97. Lecture notes in computer science, vol 1267. Springer, Berlin, pp 28–40
3. Jakobsen T, Knudsen LR (2001) Attacks on block ciphers of low algebraic degree. J Cryptol 14:197–210

Intrusion Detection in Ad Hoc Networks

Qijun Gu
Department of Computer Science, Texas State University-San Marcos, San Marcos, Texas, USA

Related Concepts

►Ad Hoc Network

Definition

Intrusion detection in ad hoc networks is a type of defense mechanism that monitors, collects, and analyzes activity information of mobile ad hoc nodes in order to detect intrusive actions.

Background

Intrusion detection is one of the mechanisms to protect a system. It captures the current user activities in a system and compares them with past user behavior profiles, known attack patterns, or predefined system specifications. An intrusion is detected in the presence of a deviation of normal user activities, a match of known attack patterns, or an operation not following system specifications. Upon detecting an intrusion, intrusion detection systems (IDSs) may further alert and trigger the response mechanisms of the system to counteract the intrusion and minimize the damage.

Many IDSs deployed in wired networks are built-in devices that are trustable and can capture and inspect traffic directly. However, ad hoc networks do not support the deployment of such devices as they are infrastructure-less

and distributed. IDSs have to be implemented in mobile ad hoc nodes, even though some of them may be malicious. No nodes can inspect all the packets as their communication range is limited and they cannot use all the resources for intrusion detection solely. The mobility of ad hoc networks further increases the difficulty of intrusion detection. As nodes are moving, it is harder to distinguish malicious and benign behaviors as mobility may result in abnormal activities due to lost or out-of-date information. Hence, intrusion detection in ad hoc networks is different from wired network in many aspects.

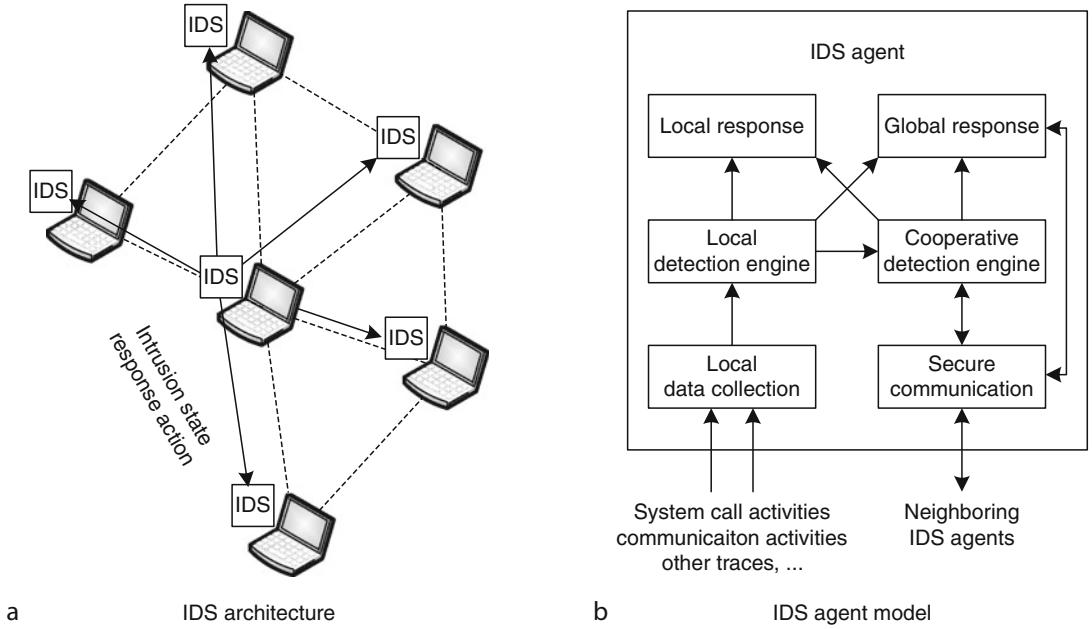
Theory

As mobile ad hoc networks (MANETs) are distributed and mobile in nature, intrusion detection requires cooperation among nodes. All the nodes need to participate in monitoring activities of other nodes and exchanging collected information to make joint decisions. Figure 1 shows a distributed and cooperative intrusion detection architecture [1] that adapts the characteristics of ad hoc networks.

In the network of Fig. 1a, every node has a running IDS agent. Each agent independently monitors local activities, collects traffic traces, detects possible intrusions, and initiates responses. When anomaly is detected but evidence is inconclusive, IDS agents participate in global intrusion detection cooperatively to broaden evidence collection and make joint decisions. The structure of an IDS agent is depicted in Fig. 1b. Although IDS agents could be constructed in various ways, the six components in the structure capture the core functions of an agent. The left side are three components for local intrusion detection and the right side are three components for cooperative intrusion detection.

The local data collection module collects communication activities of this node and its neighboring nodes. The data includes packets of cross multiple network layers. Then, the local detection engine analyzes the collected traces. If an intrusion is detected with strong evidence, the local detection engine can request the local response module independently to initiate a response. Otherwise, i.e., if an intrusion is detected with weak or inconclusive evidence, the local detection engine can request the cooperative detection engine to start global detection. The cooperative detection engine enables exchanging local detection information with neighboring nodes via the secure communication module and also coordinates their detection actions. With more evidences, neighboring nodes can make a joint decision and elect a remedy action. Thus, the global response module can carry the decision and the action. The response could be identifying and excluding compromised nodes, or rekeying and re-authenticating good nodes.

Because the all nodes have the same IDS agent, the distributed and cooperative IDS architecture is well suited to the flat network topology, where all nodes are peers. However, an ad hoc network may have the hierarchical



a IDS architecture b IDS agent model

Intrusion Detection in Ad Hoc Networks. Fig. 1 Distributed and cooperative IDS in ad hoc networks, proposed in [1]

Intrusion Detection in Ad Hoc Networks. Table 1 IDS quality, reported in [1]

Detection performance on DSR				
Trace	RIPPER		SVM	
	TPR	FPR	TPR	FPR
100k-rt	90.7 ± 3.24%	15.3 ± 4.08%	99.1 ± 0.16%	0.0667 ± 0.002%
100k-tf	85.2 ± 2.38%	13.7 ± 4.30%	99.1 ± 0.09%	0.0556 ± 0.022%
10k-rt	90.9 ± 3.07%	9.56 ± 4.27%	99.1 ± 0.37%	0.0360 ± 0.042%
10k-tf	89.8 ± 4.23%	10.12 ± 5.53%	99.0 ± 0.33%	0.0533 ± 0.065%
Detection performance on DSDV				
Trace	RIPPER		SVM	
	TPR	FPR	TPR	FPR
100k-rt	88.34 ± 5.03%	23.8 ± 7.41%	86.0 ± 0.96%	26.3 ± 5.49%
100k-tf	90.61 ± 2.99%	24.1 ± 6.70%	85.6 ± 0.83%	25.5 ± 2.05%
10k-rt	87.93 ± 4.31%	15.8 ± 4.32%	85.3 ± 4.82%	20.5 ± 10.0%
10k-tf	85.23 ± 3.28%	14.5 ± 4.87%	84.4 ± 0.60%	23.4 ± 5.78%
Detection performance on AODV				
Trace	RIPPER		SVM	
	TPR	FPR	TPR	FPR
100k-rt	91.71 ± 3.23%	20.2 ± 6.27%	95.3 ± 0.79%	1.27 ± 0.38%
100k-tf	88.48 ± 4.14%	17.8 ± 5.10%	93.9 ± 0.72%	2.06 ± 0.63%
10k-rt	92.36 ± 3.79%	14.4 ± 4.87%	94.7 ± 0.51%	3.28 ± 0.93%
10k-tf	89.91 ± 5.31%	15.7 ± 3.39%	97.1 ± 0.32%	3.57 ± 0.79%

topology, where nodes are organized in clusters with one cluster header for each cluster. Hence, the IDS architecture is extended to having hierarchy to match the roles of nodes in the hierarchical network topology. As the cluster headers usually have more computational capability than other members in the clusters, they are responsible for forwarding packets among clusters. Their IDS agents include an enhanced data collection module that collects and inspects the packets within their clusters as the routers in wired networks. Their IDS agents also have a controlling module for coordinating nodes in their clusters, making decisions, and initiating global responses. On the contrary, regular nodes in a cluster can have IDS agents with simplified components for cooperative intrusion detection, since they act under the control of their cluster headers. Such an IDS [2] was proposed, in which a network is partitioned into zones (equivalent to clusters). The header of each zone is also the gateway of the zone. A zone header aggregates and correlates reports from nodes in the zone and collaborates with other zone headers to perform intrusion detection.

Anomaly-based intrusion detection is the major approach to detect misbehavior nodes in ad hoc networks, as the malicious nodes may seemingly follow specifications or not use known attack approaches. The success of intrusion detection is indicated by a high true positive rate (TPR) and a low false positive rate (FPR). The success depends on a few assumptions. First, the majority of nodes should be good and they should monitor the network as designed. Second, the majority of the collected network traces should not be poisoned by attackers and they should be shared among nodes. Third, the majority nodes should behave normally even though benign nodes may present abnormal behavior sometimes. Using the behavior profile of each node can help make decisions and avoid malicious nodes. Watchdog and pathrater were used in such an IDS [3] to monitor the behaviors of neighboring nodes and give ratings to the nodes accordingly. By keeping the rating of every node in the network that it knows, the pathrater can compute the path metric by combining the node ratings together with the link quality to choose the path with the highest metric. As a result, paths containing misbehaving nodes will be avoided.

Applications

Intrusion detection in ad hoc network has been applied in mobile ad hoc networks, sensor networks, vehicular ad hoc networks, and so on. It is deployed to detect compromised, misbehaving, or selfish nodes. It protects these networks from various attacks including dropping, misrouting, forging, and injecting packets in networks.

Experimental Results

Whether or not an IDS can protect a system effectively against attacks is the main quality of the IDS. It is usually measured as the TPR and the FPR. The TPR measures the percentage of detected malicious events in all intrusive

events. The FPR measures the possibility of misclassifying a good event as malicious.

To measure the quality, the IDS proposed in [1] was studied with three ad hoc routing protocols: dynamic source routing (DSR) protocol, ad hoc on-demand distance vector (AODV) routing protocol, and destination-sequenced distance-vector (DSDV) routing protocol. The study built two anomaly detection models using the repeated incremental pruning to produce error reduction (RIPPER) algorithm and the support vector machine (SVM) algorithm. The two models were tested on four different traces with embedded intrusion sessions. 100k-rt and 10k-rt are traces with intrusions on routing logic and with running time as 100,000 and 10,000 seconds, and 100k-tf and 10k-tf are traces with distortion on traffic patterns.

The quality of the IDS is illustrated in Table 1. The result showed that the quality of an IDS is highly determined by the classification algorithms. In the experiments, SVM showed a better performance than RIPPER in both TPR and FPR. As SVM can better catch the dynamic characteristics of ad hoc nodes and traffic patterns, it can detect more malicious activities and misclassify fewer good ones. The result also showed that the quality of an IDS is related to the subjects of protection as well. The IDS worked better for DSR and AODV than DSDV. The further analysis showed that the redundancy in routing information in DSR and AODV helped the IDS to distinguish attacking traffic from normal traffic.

Open Problems

Security threats present in all layers of ad hoc networks. Current intrusion detection systems in ad hoc networks mainly focus on routing attacks and link layer attacks. Attacks in other layers, such as physical layer and transport layer, need the same attention. New threat models and features across multiple layers also need to be studied so that new intrusion detection mechanisms can be developed to better aggregate cross-layer threat information to detect attacks.

Another challenge in current intrusion detection mechanisms is that many IDSs in ad hoc networks can only achieve a TPR less than 95% and a FPR greater than 5%. With such quality, it is hard to use an IDS in network, because it will miss a significant portion of attacking nodes but detain a significant portion of good nodes. The problem is caused by the distributed and mobile nature of ad hoc networks. Better understanding of such nature is necessary to develop novel intrusion detection mechanisms with better quality.

Recommended Reading

1. Zhang Y, Lee W, Huang YA (2003) Intrusion detection techniques for mobile wireless networks. Wireless Networks 9(5):545–556
2. Sun B, Wu K, Pooch UW (2003) Alert aggregation in mobile ad hoc networks. Proceedings of ACM Wise, Samrego, pp 69–78
3. Marti S, Giuli TJ, Lai K, Baker M (2000) Mitigating routing misbehavior in mobile ad hoc networks. Proceedings of ACM MobiCom, Boston, pp 255–265

Invasive Attacks

Assia Tria[1], Hamid Choukri[2]
[1]CEA-LETI, France
[2]Gemalto Compagny

Definition

Invasive attacks refer to attacks of physical systems where the physical properties of the chip are irreversibly modified. Different kinds of attacks are possible using "standard" reverse engineering techniques with optical or scanning electron microscopes (SEM). The aim is to capture information stored in memory areas, or data flowing through the data bus, registers, etc. Such techniques are also used to disconnect circuits, to override sensors, or to defeat blown fuse links (using probe stations or Focused Ion Beam [FIB]). These attacks are not specific to smart cards. At first, tools used for invasive attacks were dedicated to failure analysis and debugging by semiconductor manufacturers.

Background

A smart card contains an embedded microchip with metallic contacts on the front. The smart card does not usually have its own power supply, yet it operates as a very small computer. The embedded operating system (OS) controls application execution, access condition, cryptographic routines, and communication protocols with the outside world (usually a terminal). Some smart cards have several applications embedded at a time; these are called "multi-application" smart cards.

The largest application is the GSM. The subscriber identity module (SIM) is found in all handsets that use the GSM wireless communication standard (Europe, Asia, Latin America, and increasingly North America). Smart cards are also widely used in banking, pay-TV, access control, health insurance, public transportation, government and online services. Broadly speaking, the smart card is used to control access to specific devices, networks, or services. When smart cards are combined with a ►Public

I

Key Infrastructure (PKI), they efficiently implement secure spaces within a broader IT environment, which are useful for applications such as online payment and identification.

Smart Card Security

The purpose of a smart card is to ensure the secure storage of sensitive data, and also the integrity and tamper-resistant execution of cryptographic applications. Highly sensitive data is never released outside the card; all operations are carried out by the operating system inside the card. The operating system also handles security and data access for each of these applications. This section describes the mechanisms that protect on-card data and applications.

Smart cards have an integrated CMOS circuit (Fig. 1) commonly referred to as a chip comprising:

- Logical functions.
- CPU from 8 bits up to 32 bits.
- Different kinds of memories like ROM, EEPROM, RAM and more recently Flash and Feram (Ferroelectrics RAM). Here the ROM is a permanent memory storing all or part of the operating system. The EEPROM is a nonvolatile erasable memory that stores application data, mainly keys, PIN codes, or personal data, and, in some cases, parts of the operating system and its applications. The RAM is a volatile memory that stores temporary data such as intermediate internal application data or session keys or access rights.
- I/O interface for communication with the external world.
- Peripherals such as crypto-coprocessors, Random-Number Generators.

Overview of Attacks on Smart Cards

Basically, smart card attacks can be classified into three main categories: social, logical, and physical attacks.

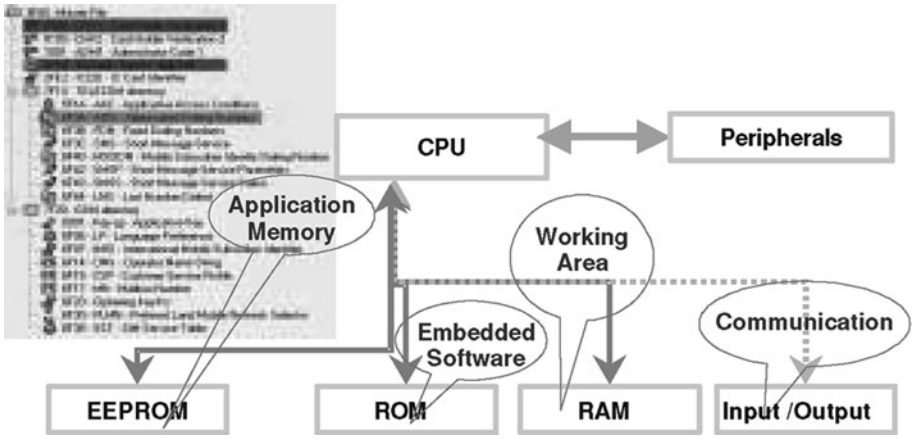
Social Attacks

These are the oldest. The idea behind these attacks is to obtain information directly from the manufacturer using classical social engineering techniques. Countermeasures in this case range from physical security such as access control to the sites, physically isolated labs, and the education of employees to the application of strict security procedures. This kind of attack falls outside the scope of this document.

Logical Attacks

These attacks are used to recover secret data from secure devices without actually damaging the device. By monitoring execution time, power consumption, or electromagnetic radiation of a chip, it is frequently possible to infer information about the processed data. Performing ►Side-channel analysis on a secure device requires advanced knowledge in electronics, cryptography, signal processing, and statistics. A well-known class of attacks in this group is based on the analysis of smart card *power consumption*. This class includes ►Differential Power Analysis (DPA), Simple Power Analysis (SPA), and timing analysis.

- **SPA** uses variations in the global power consumption of the chip and infers information that is normally held within the chip. For example, an increase in power consumption might indicate that a modular exponentiation (an important cryptographic function) is being performed. In general, SPA will give better results if



Invasive Attacks. Fig. 1 Microprocessor card architecture

the attacker has extensive knowledge of the hardware architecture.

- **DPA** is more sophisticated than the SPA. Statistical analysis of power consumption curves is carried out for several executions of the same algorithm. The input data is changed in such a way that sensitive information can be deduced.
- ►**Timing attacks** have posed serious problems in the past, because in older designs the execution time would vary according to the data and/or the cryptographic keys that were being processed. Current smart card chips have been designed with constant timing, or at least timings that do not depend on data or secret keys.
- **Electromagnetic analysis** is a newer type of attack. It is based on the same techniques as those used for DPA and SPA, but the physical quantities that are measured are not the same. In this case, the RF signals provide the essential information. Such attacks fall into the category of side-channel attacks, but they differ in a number of crucial points from power attacks.
- ►**Fault attacks** are conducted using a combination of several environmental conditions that cause the chip to produce computational errors that can leak protected information. Sensors that detect abnormal operating conditions are thus used to preclude the need for costly software and hardware countermeasures (it is always better to anticipate rather than to correct errors).
- **Software attacks** target software flaws using a normal communication channel to interface with the card. These flaws may weaken the security features of the card or allow them to be bypassed, leaving the system open to frauds. There is a wide range of software attacks, some of which are not specific to the smart card. Incorrect file access conditions, malicious code, flaws in cryptographic protocols, design and implementation errors are common flaws in computing systems.

Invasive Attacks

The goal of these attacks is to unearth information stored in memories, or data flowing through the data bus, registers, etc. These attacks are not specific to smart cards but are generic to CMOS components. Modifying an electronic component requires considerable time and resources, sophisticated and expensive tools, and extensive hardware expertise. This cost can be considered as a first barrier. The more you know about the internal architecture of the chip the easier these attacks will be.

Access to Silicon

Delayering

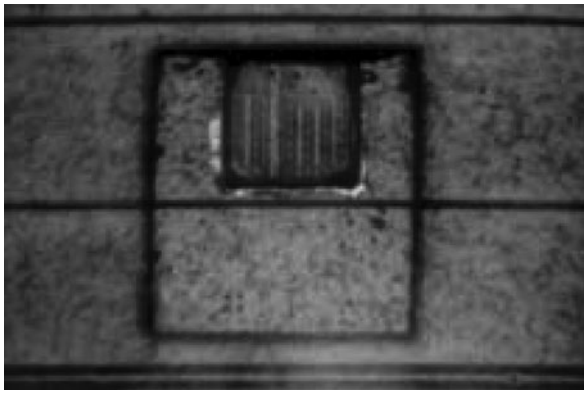
The first step consists in doing some reverse engineering. Reverse engineering allows block localization, like memories, buses, random-number generation, inputs, etc., and also an understanding of the chip architecture.

The chip is covered by a globe-top made of epoxy resin that can be removed by using hot fuming nitric acid. At this stage, the chip's surface is not yet accessible for probing or modification; only optical or electrical analysis could be feasible depending on the chip manufacturer's process.

CMOS chip structure is made of multiplayer stacking going from three to five metal layers. The upper layer is called the passivation layer (silicon oxide), which protects the chip against environmental hazards and ionic contaminations.

As a large amount of stress can be generated on the die during the assembly process, a thick film of polyimide over the passivation layer is deposited. These stresses may lead to cracking of the protective passivation layer. Before getting access to the silicon, the polyimide and passivation layers must be removed using ethylendiamine and fluoridric acid, respectively. For the passivation layer the most convenient depassivation technique is to use a laser cutter, but this technique can only be used to create small windows used for probing (Fig. 2).

There is no particular way to prevent optical reverse engineering except by increasing the design complexity and sharing specific parts of the circuit among different layers.



Invasive Attacks. Fig. 2 Example of window opened in the passivation layer using laser cutter

Block Localization

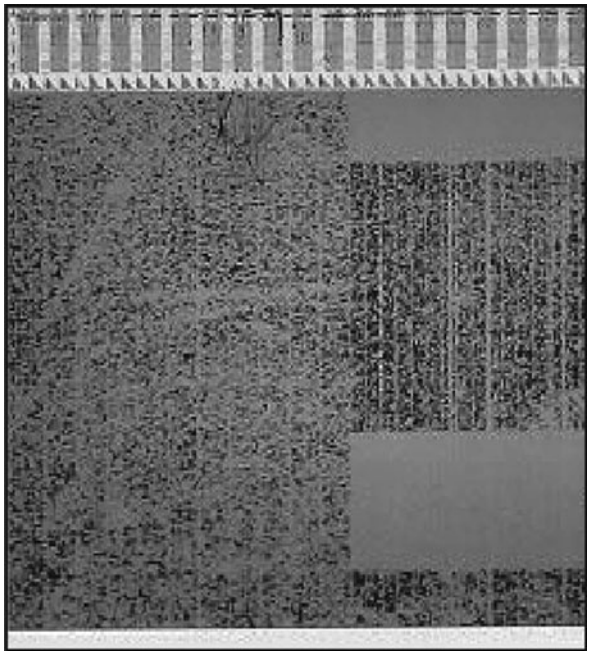
Once the passivation layer has been removed, the upper metal layer becomes accessible. Each metal layer can be selectively removed by chemical attacks. Depending on the layer structure, it can be dry-etched using plasma or wet etching. This step can be destructive: several chips are usually required for these attacks.

Memory blocks like ROM, EEPROM, RAM, and parts of analogue blocks such as charge pumps and capacitors are easily recognizable, an example is given in Fig. 7.

Memory Content Extraction

- **ROM**

The ROM is a critical part of the circuit because it is where the code (operating system, JAVA virtual machine, API) is stored. In new generations of products (130 nm), it is preferable to use a metal (Fig. 3) or ionic implantation ROM process. The difficulty of reverse-engineering the ROM content in these two cases is identical. In the case of a metal ROM a selective delayering of the chip metal layers is necessary in order to reach the appropriate metal layer (usually M1) and the metal connections. In the case of an ionic implantation ROM a complete delayering is required in order to reach the silicon level. A chemical attack is further performed to reveal doping (in order to create active region in the semiconductor, impurities like Phosphor or boron are introduced) differences and make the content readable by optical observations, a process that can be automated. And then ROM mapping of zero and one is extracted.



Invasive Attacks. Fig. 3 Metal ROM showing unused area

In order to protect the ROM contents from illegal extraction, all chip manufacturers encrypt the ROM. The ciphering algorithm depends on both the data value and the address. So a complete reverse engineering of the decoder's logic is necessary. In addition, ROM code will be systematically filled with random data when the code size is not equal to the complete ROM size.

- **RAM**

The RAM is a temporary working area. The RAM data is, for example, used for cryptographic calculations and session key transfers. Using a SEM specialized in voltage contrast, it is possible to observe RAM activity in operating mode. These microscopes have the ability of detecting variations in voltage. Applying power to the chip and observing the chip in image mode reveals the DC (direct current) conditions on the surface layers of the chip.

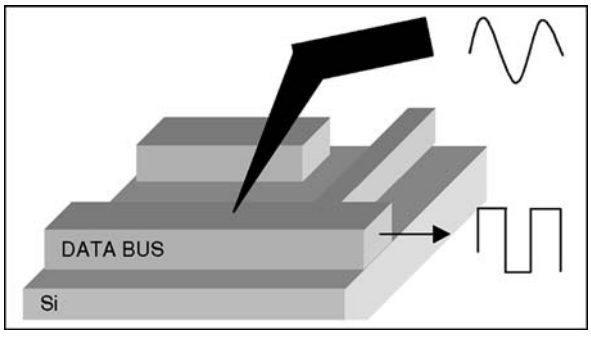
Like for the ROM, chip manufacturers encrypt or scramble the RAM. Scrambling depends only on the address (address transposition), while ciphering depends on both the address and the data: these mechanisms can be static or dynamic. Frequency sensors, used for restricting the operating range of the chip, could protect against the use of SEM in voltage contrast.

- **EEPROM or FLASH**

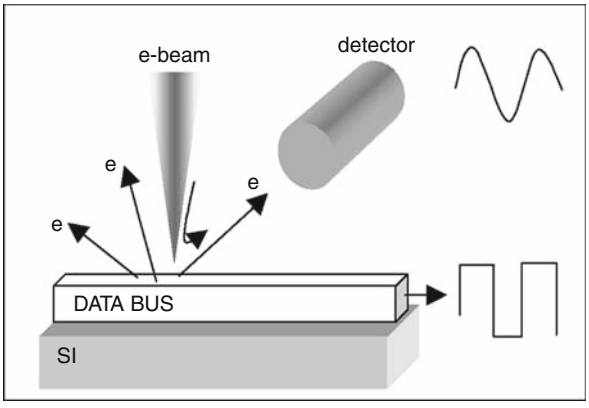
This write/erase nonvolatile memory is used to store application data, part of executable code, and sensitive data like PIN code or session keys. Reversing the EEPROM or Flash memory consists in finding the state of the floating gate. At this time, no method has been found to extract the complete memory content. As before, nonvolatile memory can be encrypted or scrambled to increase security level. There is an emergence of new nonvolatile memory point like Flash and Feram in the smart card chip market. These new memory maps have the particularity of being very compact, of taking less space, and are probably more difficult to attack. Soon there will be smart cards where the executable code will be loaded in flash. With a strong memory management and a shrinking technology, one would think that these products will be more secure. However, the resulting level of security depends on other factors such as the time available before the product needs to be finished.

Bus Localization

Using Voltage contrast techniques, bus activity can be observed. Besides, in a slow scan image mode, it is possible to visualize different clock rates across the bus lines.



Invasive Attacks. Fig. 4 Mechanical probing



Invasive Attacks. Fig. 5 e-Beam probing

Observing the changing contrast on the signal path can be correlated to the changing logic state. However, manufacturers forecast specific countermeasures comprising:

- Static ciphering
- Dynamic ciphering
- Complemented logic
- Buried Buses
- Dummy activity on buses

Chip Probing

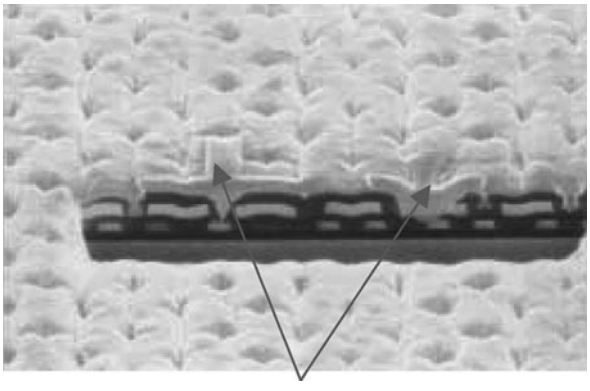
Once the first step of reverse engineering is completed, chip probing can start. As buses are connected to the CPU and also to the memories, there is a great interest in taping data passing through these buses. Hence, it could be possible to retrieve the full running program. In this case a small reverse engineering must have been done previously and passivation removal or circuit modification is sometimes required.

Probing could be mechanical using a microprobe on a micromanipulator with a probe station (Fig. 4). This technique is very difficult for sub-Micronics technology smaller than 0.35 µm.

The smaller the line is, the more difficult it will be to access them by mechanical probing. Instead, e-beam probing, which is a Contactless method based on secondary electron analysis giving voltage values, will be used (Fig. 5).

Chip probing could be prevented by adding countermeasures like:

- An active shield that is a metal mesh where data passes continuously on lines. If there is a disconnection or modification of this mesh then the chip does not operate anymore.
- A passive shield where a full metal mesh covers some sensitive parts of the circuit
- Bus static or dynamic scrambling
- Buried lines



Invasive Attacks. Fig. 6 Probe pads added by FIB to reach M1 through a shield

Chip Modification

Modifying or disconnecting part of a circuit in order can constitute an interesting attack method. Using this method it is possible to connect or disconnect hardware security mechanisms.

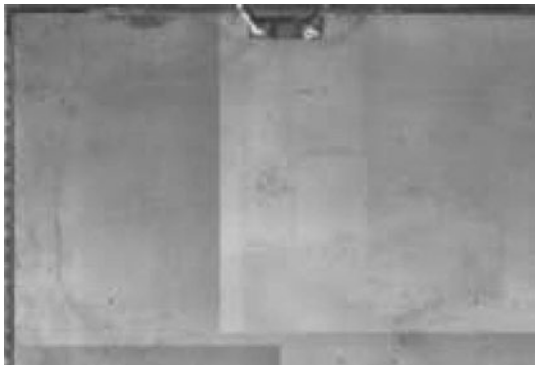
The Focused Ion Beam (FIB) is a convenient and powerful tool. A FIB allows material deposited for the creation of metal lines or metal cross-pads allowing access to the bus, as illustrated in the following figure:

In a similar manner, removing materials using a FIB allows the track to be cut, the disconnection of security sensors, or the opening of a window through the passivation layer to get access to buried levels (Fig. 6). As mentioned previously, countermeasures can be added in order to prevent such attacks:

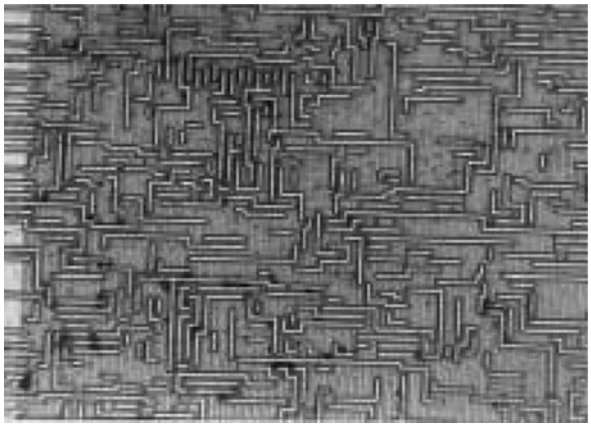
- Active shield
- Complexity of the design
- Glue logic

I

Invasive Attacks. Fig. 7 Old chip design



Invasive Attacks. Fig. 8 New chip design



Invasive Attacks. Fig. 9 Part of glue logic design

Protection Against Physical Attacks

Conducting physical attacks against smart cards at the semiconductor level requires expensive equipment and considerable technical expertise. Therefore, the threat is limited to few organizations and specialists. However, smart card manufacturers cannot afford to release cards without effective countermeasures against such attacks. Some of the principles used to physically secure cards are described below.

Modern smart cards use semiconductor technology for the chip, making reverse engineering by observation difficult. The size and density of the transistors on the chip surface has drastically shrunk (130 nm), and chips are now considered secure against visual analytical reverse engineering.

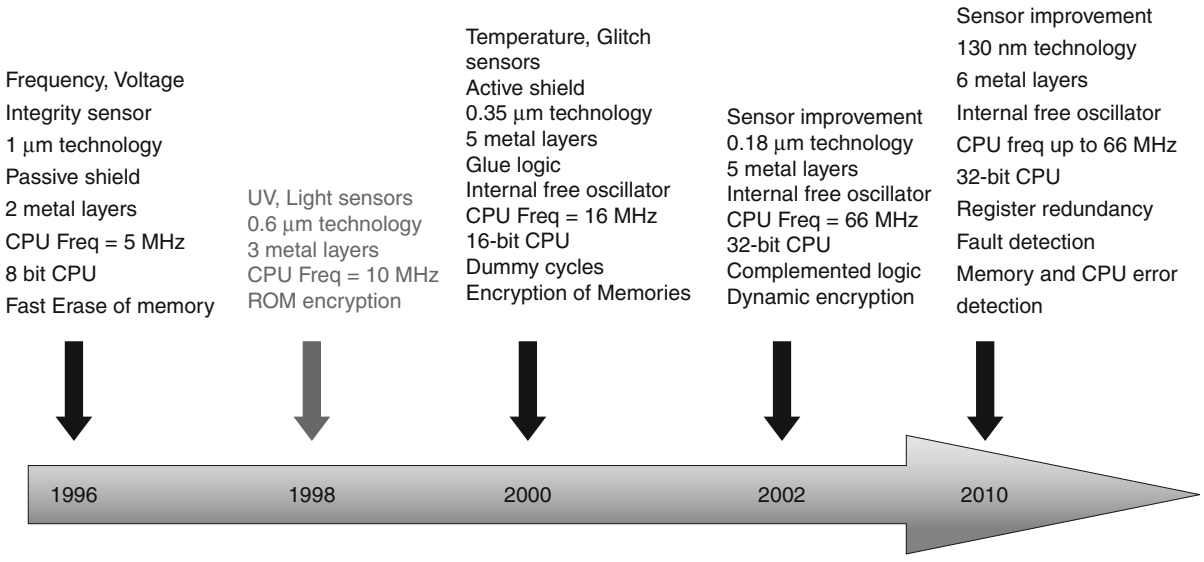
Functional blocks are mixed (Fig. 9), producing what is called a glue logic design (block logic randomization). This makes it much more difficult for an attacker to analyze the structure of the chip and to localize functional blocks (CPU, RAM, ROM, EEPROM, buses, registers, etc.).

Moreover, the buses are scrambled or ciphered and are thus inaccessible from outside the chip, so connections cannot be easily made to recover memory contents. Memories are also scrambled or ciphered in order to protect the chip from selective access/erasure of individual data bytes.

Chips are made of multiple layers, allowing manufacturers to hide sensitive components (e.g., data lines, connections) in between different layers that contain less sensitive components. For instance, the ROM is located in the lower (least accessible) layers of the chip.

A current-carrying protective layer or active shield is added at the top of the chip for power supply. If this layer is removed, the chip will no longer operate. This layer prevents analysis of electrical voltage on the chip to infer information.

Moreover, a set of sensors is activated to detect abnormal variations of environmental variables (refer also ►Smart Card Tamper Resistance). It guarantees that the chip will not be able to operate in abnormal conditions of use. These sensors measure values such as voltage, temperature, clock frequency, and light. Such sensors offer protections against fault attacks (among others).



Invasive Attacks. Fig. 10 Security feature evolutions

Conclusion

Smart cards turn out to be the strongest components in the system. In practice, it tends to be much easier to exploit weaknesses in protocols or in the implementation, than to physically penetrate the card and extract its secrets. In the last five years the level of hardware security found in smart cards has increased enormously (Fig. 8). In 1998 the first publications on potential smart card vulnerabilities were written. Afterwards chip manufacturers have made tremendous efforts to increase the security levels.

This is illustrated in the following figure (Fig. 10):

However, security by the use of countermeasures must be added at each smart card process level such as component, software layer, applicative layer, etc. Smart card device using appropriate and well design countermeasures is the most secure token and what about future trends?

- Chip design technologies will be smaller and smaller (0.07 µm).
- CPU frequencies will reach 100 MHz.
- New nonvolatile memories will be introduced (Feram, MRam, etc.).
- More complex design and countermeasures will be developed.

Hence, invasive attacks will be increasingly difficult and will require powerful tools and expert knowledge in chip architecture and electronics.

Inversion Attack

Anne Canteaut
Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

Related Concepts

►Filter Generator; ►Stream Cipher

Definition

The *inversion attack* is a ►known plaintext attack on some particular ►filter generators. It was proposed by Golić in 1996 [1]. A generalization to any filter generator, called *generalized inversion attack*, was presented by Golić, Clark, and Dawson in 2000 [2]. Both inversion attack and generalized inversion attack aim at recovering the initial state of the ►linear feedback shift register (LFSR) from a segment of the ►running-key when the LFSR feedback polynomial, the tapping sequence, and the filtering function are known.

Theory

Original Inversion Attack

The original inversion attack only applies when the filtering function f is linear in its first input variable (forward attack) or in its last input variable (backward attack), i.e., when

$$f(x_1, x_2, \ldots, x_n) = x_1 + g(x_2, \ldots, x_n)$$

or

$$f(x_1, x_2, \ldots, x_n) = g(x_1, \ldots, x_{n-1}) + x_n$$

where g is a ►Boolean function of $n-1$ variables. In the first case, the keystream $\mathbf{s}$ is defined by

$$\begin{aligned} s_t &= f(u_{t+\gamma_1}, u_{t+\gamma_2}, \ldots, u_{t+\gamma_n}) \\ &= u_{t+\gamma_1} + g(u_{t+\gamma_2}, \ldots, u_{t+\gamma_n}), \end{aligned}$$

where $(u_t)_{t\geq 0}$ is the sequence generated by the LFSR and $(\gamma_i)_{1\leq i\leq n}$ is a decreasing sequence of non-negative integers. The attack relies on the fact that the bit $u_{t+\gamma_1}$ can be deduced from the $(\gamma_1 - \gamma_n)$ previous terms, $(u_{t+\gamma_n}, \ldots, u_{t+\gamma_1-1})$, if the running-key bit s_t is known. The relevant parameter of the attack is then the *memory size* of the filter generator, defined by $M = \gamma_1 - \gamma_n$. Indeed, the complete initialization of the LFSR can be recovered by an exhaustive search on only M bits as described in Table 1.

The backward attack, which applies when the filter function is linear in its last variable, is similar. The complexity of both forward and backward attacks is $\mathcal{O}(L2^M)$. It follows that the memory size of a filter generator should be large and preferably close to its maximum possible value $L-1$.

Moreover, the complexity of the attack dramatically decreases when the greatest common divisor of all spacings between the taps, $d = \gcd(\gamma_i - \gamma_{i+1})$, is large. Indeed, the inversion attack can be applied to the d-decimation of the LFSR sequence, i.e., to the sequence obtained by sampling the LFSR sequence at intervals of d clock cycles (see [1]). Therefore, the effective memory size of the filter generator corresponds to

$$M' = \frac{\gamma_1 - \gamma_n}{\gcd(\gamma_i - \gamma_{i+1})}.$$

The related design criterion is then that the greatest common divisor of all spacings between the taps should be equal to 1.

Inversion Attack. Table 1 Inversion attack

Input. $s_0 s_1 \ldots s_{N-1}$, N keystream bits.
Output. $u_{\gamma_n} \ldots u_{L+\gamma_n-1}$, L consecutive bits of the LFSR sequence, where L is the LFSR length.
For each choice of the M-bit vector $u_{\gamma_n} \ldots u_{\gamma_1-1}$
Compute the next $(L-M)$ bits of the LFSR sequence by

$$u_{t+\gamma_1} \leftarrow s_t + g(u_{t+\gamma_2}, \ldots, u_{t+\gamma_n}), \quad 0 \leq t \leq L-M.$$

Compute $(N-L)$ additional bits of the LFSR sequence with the LFSR recurrence relation, and the corresponding running-key bits, $\hat{s}_t$, for $L-M \leq t < N-M$.
If the $N-L$ bits $\hat{s}_t$ are equal to the observed keystream sequence, then return $(u_{\gamma_n} \ldots u_{L+\gamma_n-1})$.

Generalized Inversion Attack

A similar attack can be mounted even if the filtering function is not linear in its first or last variable. In the general case, the keystream is given by

$$s_t = f(u_{t+\gamma_1}, u_{t+\gamma_2}, \ldots, u_{t+\gamma_n}).$$

Exactly as in the original inversion attack, the basic step of the attack consists in deducing the bit $u_{t+\gamma_1}$ from the knowledge of the keystream bit s_t and of the M previous terms of the LFSR sequence, $(u_{t+\gamma_n}, \ldots, u_{t+\gamma_1-1})$. For fixed values of s_t and of $(u_{t+\gamma_n}, \ldots, u_{t+\gamma_1-1})$, the unknown bit $u_{t+\gamma_1}$ may take 0, 1, or 2 possible values. Then, an exhaustive search for the M bits $u_{\gamma_n} \ldots u_{\gamma_1-1}$ of the LFSR sequence can still be performed. For a given value of the M-bit vector $u_{\gamma_n} \ldots u_{\gamma_1-1}$, a binary tree of depth $L-M$ representing all the solutions for the next $(L-M)$ bits of $\mathbf{u}$ is formed. Each node at level t corresponds to a guessed value of $(u_{t+\gamma_n} \ldots u_{t+\gamma_1-1})$. Then, the number of edges out of this node is 0, 1, or 2 according to the number of solutions x of the equation $s_t = f(x, u_{t+\gamma_2}, \ldots, u_{t+\gamma_n})$. If a tree of depth $L-M$ can be constructed from a given M-bit root, some additional bits of the LFSR sequence are computed and their consistency with the observed keystream is checked. It is shown that the typical number of surviving nodes at level $L-M$ is linear in L. Then, the typical complexity of the attack is $\mathcal{O}(L2^M)$. Exactly as in the inversion attack, the parameter involved in the attack is the effective memory size, i.e.,

$$M' = \frac{\gamma_1 - \gamma_n}{\gcd(\gamma_i - \gamma_{i+1})}.$$

Another technique based on a trellis representation and on the Viterbi algorithm is described in [3]. Its efficiency is comparable to the generalized inversion attack.

Recommended Reading

1. Golić JDj (1996) On the security of nonlinear filter generators. In: Fast software encryption 1996. Lecture notes in computer science, vol 1039. Springer, Berlin, pp 173–188
2. Golić JDj, Clark A, Dawson E (2000) Generalized inversion attack on nonlinear filter generators. IEEE Trans Comput 49(10): 1100–1108
3. Leveiller S, Boutros J, Guillot P, Zémor G (2001) Cryptanalysis of nonlinear filter generators with (0, 1)-metric Viterbi decoding. In: Cryptography and coding – 8th IMA international conference, UK, 17–19 December 2001. Lecture notes in computer science, vol 2260. Springer, Berlin, pp 402–414

Inversion in Finite Fields and Rings

Christof Paar
Lehrstuhl Embedded Security, Gebaeude IC 4/132,
Ruhr-Universitaet Bochum, Bochum, Germany

Synonyms

Inversion in Galois fields

Related Concepts

►Euclidean Algorithm; ►Finite Field

Definition

Inversion is the computation of the element A^{-1} such that

$$AA^{-1} = 1$$

where A is a nonzero element of a finite field or ring, and "1" is the neutral element of the algebraic structure. In finite fields, or Galois fields, the inverse exists for all nonzero elements. In finite rings, not all elements have an inverse.

Theory

One distinguishes between inversion in a finite ring and in a ►finite field (or Galois field). In the case of inversion in a finite integer ring or polynomial ring, the extended ►Euclidean algorithm can be used. Let u be the element whose inverse is to be computed and v the modulus. Note that u and v must be relatively prime in order for the inverse to exist. The extended Euclidean algorithm computes the coefficients s and t such that $us + vt = \gcd(u, v) = 1$. The parameter s is the inverse of u modulo v. In the case of finite integer rings, using the ►binary Euclidean algorithm leads often to faster executions on digital computers. The binary Euclidean algorithm does not require integer divisions but only simple operations such as shifts and additions.

In the case of finite field, there are several approaches to computing multiplicative inverses of nonzero elements:

Extended Euclidean Algorithm

This is the most general and in many cases most efficient method. The application is completely analogous to the case of finite rings as discussed above. In the case of prime fields, the standard extended Euclidean algorithm applies, and the binary Euclidean algorithm is often computationally advantageous. If the inverse in an extension field is to be computed, the Euclidean algorithm with polynomials has to be used.

Fermat's Little Theorem

This method has a higher computational complexity than the extended Euclidean algorithm but can nevertheless be relevant in certain situations, for example, if a fast exponentiation unit is available or if an algorithm with a simple control structure is desired. From ►Fermat's little theorem it follows immediately that for any element $A \in GF(q^m)$, $A \neq 0$, the inverse can be computed as $A^{-1} = A^{(q^m-2)}$. For fields of characteristic two, that is, fields $GF(2^m)$, the use of addition chains allows to dramatically reduce the number of multiplications (though not the number of squarings) required for computing the exponentiation to the $(2^m - 2)$th power. This method is referred to as ►Itoh–Tsujii Inversion.

Look-Up Tables

A conceptually simple method is based on look-up tables. In this case, the inverses of all field elements are precomputed once with one of the methods mentioned above, and stored in a table. Assuming the table entries can be accessed by an appropriate method, for example, by the field elements themselves in a binary representation, the inverses are available quickly. The drawback of this method are the storage requirements, since k memory locations are needed for fields $GF(k)$. Since the storage requirements are too large for the finite fields commonly needed in public-key cryptography, inversion based on look-up tables is mainly useful in cases of small finite fields. For instance, inversion in GF(256) is the main operation in the ►AES S-Box.

Reduction to Subfield Inversion

In the case of extension fields $GF(q^m)$, $m \geq 2$, inversion in the field $GF(q^m)$ can be reduced to inversion in the field $GF(q)$. This reduction comes at the cost of extra operations (multiplications and additions) in the field $GF(q^m)$. If the inversion in the subfield $GF(q)$ is sufficiently inexpensive computationally compared to extension field inversion, ►Itoh–Tsujii Inversion can have a low overall complexity. The method was introduced for fields in normal basis representation in [2] and generalized to fields in polynomial basis representation in [1]. The method can be applied iteratively in fields with multiple field extensions, sometimes referred to as tower fields. In the case of fields $GF(2^m)$, m a prime, the Itoh–Tsujii algorithm degenerates into inversion based on Fermat's little theorem. It should be stressed that this method is not a complete inversion algorithm since it is still necessary to eventually perform an

inversion in the subfield. However, inversion in a (small) subfield can often be done fast with one of the methods described above.

Direct Inversion

This method is applicable to extension fields $GF(q^m)$, and mainly relevant for fields where m is small, for example, $m = 2, 3, 4$. Similar to ►Itoh–Tsujii Inversion, direct inversion also reduces extension field inversion to subfield inversion. As an example, in the following the method for fields $GF(q^2)$, introduced in [3], is shown. Consider a nonzero element $A = a_0 + a_1 x$ from $GF(q^m)$, where $a_0, a_1 \in GF(q)$. An irreducible field polynomial with the form $P(x) = x^2 + x + p_0$ is assumed, where $p_0 \in GF(q)$. If the inverse is denoted as $B = A^{-1} = b_0 + b_1 x$, the equation

$$A \cdot B = [a_0 b_0 + p_0 a_1 b_1] + [a_0 b_1 + a_1 b_0 + a_1 b_1] x = 1$$

must be satisfied, which is equivalent to a set of two linear equations in b_0, b_1 over $GF(q)$ with the solution:

$$\left.\begin{aligned} b_0 &= (a_0 + a_1)/\Delta \\ b_1 &= a_1/\Delta \end{aligned}\right\}, \text{ where } \Delta = a_0(a_0 + a_1) + p_0 a_1^2. \quad (1)$$

The advantage of this algorithm is that all operations are performed in $GF(q)$. Note that there is one inversion of the parameter Δ in the subfield $GF(q)$ required. The algorithm can be applied recursively. The relationship between direction inversion and the Itoh–Tsujii method is sketched in [4].

Applications

The need to compute the multiplicative inverse of an element of a finite field (or Galois field) or of a finite ring occurs frequently in cryptography. The main application domain are asymmetric algorithms, for instance in the computation of the private–public key pair in ►RSA, in the group operation of ►elliptic curves, or in the signature generation and verification of the ►Digital Signature Standard. The finite structures in asymmetric algorithms are large, typically in the range of 160–3,072 bits long, and inversion is a computationally intensive operation. A second application domain in cryptography are inversions in small finite fields, which occur in the context of block ciphers, for example, within the S-box of the ►AES.

Recommended Reading

1. Guajardo J, Paar C (2002) Itoh-Tsujii inversion in standard basis and its application in cryptography and codes. Designs, Codes and Cryptography 25:207–216
2. Itoh T, Tsujii S (1988) A fast algorithm for computing multiplicative inverses in GF(2^m) using normal bases. Inform and Comput 78:171–177
3. Morii M, Kasahara M (1989) Efficient construction of gate circuit for computing multiplicative inverses over GF(2^m). Trans IEICE E 72:37–42
4. Paar C (1995) Some remarks on efficient inversion in finite fields. In Proceedings of 1995 IEEE International Symposium on Information Theory, Whistler, B.C. Canada, 1995 September 17–22, p 58

Inversion in Galois Fields

►Inversion in Finite Fields and Rings

IP Traceback

Steven M. Bellovin
Department of Computer Science, Columbia University, New York, NY, USA

Related Concepts

►Attribution

Definition

Traceback is a term used for two different technologies: learning which machine has sent a particular packet or sequence of packets, and learning which person has initiated a particular connection. The latter is sometimes called *attribution*.

Background

Often, especially during certain *Distributed Denial of Service (DDoS)* attacks, it is desirable to learn what machines are participating in the attack. The goal might be to filter out packets from the offending machines, to contact its ISP or organization and have them filter the packets, to notify the owner of the machine, or to provide evidence to law enforcement or other legal authorities. In many cases, the source IP address is sufficient; in other cases, particularly during *reflector attacks*, the source IP address does not point to the perpetrator, and other mechanisms must be used. In addition, many early DDoS attacks used forged source addresses, precisely to avoid detection.

Theory

There are three major approaches to packet tracing: return packets, in-net state, and packet marking. Each is considered in turn.

Return Packets

Return packets are the oldest tracing technology. The original motivation was to track down the source of packets with forged source addresses. With some low probability, upon receipt of any packet, each participating router sends a special ICMP packet towards the destination, identifying itself and the packet. In normal operation, these "tracer packets" are discarded. During an attack, however, they are saved; statistically, tracer packets for attack packets should be received in approximately the same proportion as the unwanted traffic. Some percentage of the tracers will be from the border routers nearest the attack sources; presumably, at this point, the originating organization can locate the offending machines.

ICMP packet tracing raises a number of issues, most notably that it inherently sends more traffic at a time when that is least affordable. If the probability of sending tracers is low enough, this is not a major problem. A more serious issue is deciding, in an automated fashion, which packets are attack packets, and hence for which the tracers should be examined and acted upon. On the other hand, it is a technology that lends itself to incremental deployment. It is beneficial even if deployed by only a few ISPs, and even then need only be deployed on border routers; conversely, deployment as far as interior organizational routers can help localize traffic sources even further.

The IETF worked on standardizing *itrace*, one particular form of return packet tracing. This effort was abandoned due to lack of interest.

In-Net State

A second major class of tracing mechanism relies on routers keeping track of packets flowing through them. The difficulty here is volume: routers, especially core routers, process a lot of traffic. Furthermore, the need for per-flow state does not exist apart from the need for measurements; unlike telephone switches, there is no other need for state on the router.

One specialized mechanism for tracing attack traffic is based on Bloom filters. A precis of each packet's content is entered into a Bloom filter on each participating router; later, an attack packet can be matched against the collected set of Bloom filters exported from various routers, to see which ones have seen which packets. Note, though, that the accuracy of a Bloom filter, and in particular its susceptibility to false positives, is dependent on both the size of the filter and the number of items entered into it. There is thus a need to export and reset filters periodically; the exact frequency depends on their size, traffic volume, and desired accuracy rate. Bandwidth from the control plane of the router to the outside world may be a limiting factor.

A more general mechanism uses flow accounting data ("netflow"). Many routers can record who is sending how much data to whom, primarily to support network traffic engineering and (in some cases) customer billing. By its nature, though, netflow data can be used for traceback since it does make note of traffic through the router.

To conserve memory, many routers accumulate sampled netflow data. As such, they will often miss small flows entirely. When tracing denial of service attacks, this matters less, since if the attack traffic through the router is consequential, there will generally be enough packets towards the victim to be tracked. The technique is less useful for spotting particular flows of interest.

Netflow data is the only traceback technology that is actually deployed today.

Packet Marking

The final tracing technology is based on packet marking. That is, some randomly selected fraction of the packets passing through certain routers is somehow marked – i.e., their contents are somehow altered – for later examination. Many such schemes have been proposed in the literature; none have been adopted. Differences include how the marking is encoded, what information must be known to understand the markings, what can be learned, etc.

All marking technologies face one serious limitation: there are no fields in the IP header that are completely suitable for this purpose. Frequently, the IP id field is overwritten since it often has no end-to-end significance. It is not always available – it is used, end-to-end, for fragmented packets, and if AH is in use it is a protected field – but if a sufficiently small proportion of the packets are marked, it won't matter; the effect of the marking will be to cause the packet to be dropped by the receiving system, which in turn will act as a bandwidth limiter.

Experimental Results

Many of the concepts discussed here have implemented or simulated in the lab. None except for use of netflow data has been deployed on live networks, since all require changes to production routers.

Open Problems

Apart from technical issues at some point, a party wishing to trace a particular incident will end up with a set of IP addresses. The next step is to map that to a particular individual. There are several steps involved: determining the responsible ISP or organization from the IP address, mapping the IP address to a machine, and finding the individual actually responsible. The latter two are quite difficult.

The responsible organization can be found by consulting public routing tables, either online via a "looking glass" server or via well-known archives of BGP announcements. This process yields the "autonomous system" number; the assorted "WHOIS" databases list the proper contacts for the organization. Alternatively, in many cases the WHOIS data will directly yield the organizational contacts based on the IP address, though subassignment data is often incomplete.

The next step — securing the organization's cooperation — is difficult procedurally. For privacy reasons (and often to comply with local or national ordinances), most organizations will not disclose any further information except in response to formal legal process. Different countries have different standards and standards; securing cooperation can require navigating the thickets of international law. Often, it is a fruitless endeavor; some countries are even reputed to tolerate the existence of ISPs that cater to criminal enterprises.

Once the ISP is willing to cooperate, there are some technical obstacles. Some ISPs reassign IP addresses frequently; finding the proper mapping depends on both parties having logs with accurate timestamps and per-connection port numbers. Even as mundane an issue as time zones has been known to interfere. Open Internet hotspots may not have any logs at all and few logs have the necessary port numbers. Finally, some IP addresses actually represent many hosts hidden behind a Network Address Translator (NAT). Some ISPs, many corporations, most hotels, and virtually all residences with multiple computers employ such devices. These devices very rarely keep logs adequate to determining which computer was contacting which other site at a given time. It is often impossible to trace a contact beyond this point.

The last obstacle, though, is often the most formidable. While the attribution techniques given above may point to a particular computer, they say nothing about who made the request. Most attacks are launched from previously hacked machines – "bots" – that are remotely controlled by attackers. Finding the command and control node of a botnet is a difficult process since it relies on assessing past communications patterns, with all of the traceback and attribution problems encountered when trying to find the bot. There have been occasional proposals to add requirements for strong authentication to all outbound Internet connections; apart from privacy concerns, their utility generally founders on this last point.

Recommended Reading

1. Bellovin SM, Leech M, Taylor T (2003) ICMP traceback messages. Obsolete Internet draft, Feb 2003
2. Clayton R (2005) Anonymity and traceability in cyberspace. PhD thesis, University of Cambridge, Darwin College. Also published as technical report UCAM-CL-TR-653
3. Savage S, Wetherall D, Karlin A, Anderson T (2000) Practical network support for IP traceback. ACM SIGCOMM '00, Stockholm, Sweden, pp 295–306
4. Snoeren AC, Partridge C, Sanchez LA, Strayer WT, Jones CE, Tchakountio F, Kent ST (2001) Hash-based IP traceback. In: SIGCOMM '01, San Diego, Aug 2001
5. Srisuresh P, Holdrege M (1999) IP Network address translator (NAT) terminology and considerations. RFC 2663, Internet Engineering Task Force, Aug 1999
6. Zhang Y, Paxson V (2000) Detecting stepping stones. In: Proceedings of the 9th USENIX security symposium, Denver, Aug 2000
7. Bellovin SM, Leech M, Taylor T (2003) ICMP traceback messages. Obsolete Internet draft, Feb 2003. http://www.cs.columbia.edu/~smb/papers/draft-ietf-itrace-04.txt
8. Savage S, Wetherall D, Karlin A, Anderson T (2000) Practical network support for IP traceback. SIGCOMM '00, Proceedings of the conference on Applications, Technologies, Architectures, and Protocols for Computer Communication. Technical Report UW-CSE-2000-02-01, Department of Computer Science and Engineering, University of Washington, Seattle. http://www.cs.washington.edu/homes/savage/traceback.html
9. Snoeren AC, Partridge C, Sanchez LA, Strayer WT, Jones CE, Tchakountio F, Kent ST (2001) Hash-based IP traceback. BBN Technical Memorandum No. 1284, http://www.ir.bbn.com/documents/techmemos/TM1284.ps, 7 Feb 2001

IPES

Alex Biryukov
FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg

Related Concepts

►Block Ciphers; ►Differential Cryptanalysis; ►IDEA

Definition

IPES is an alternative name for the ►IDEA cipher. IPES stands for "improved PES," where *PES* [1] is a cipher predecessor of IDEA which was cryptanalyzed by ►differential cryptanalysis in [2]. The only changes between IDEA and PES are in the order of operations in the key-mixing subround: PES uses the order $(\odot, \odot, \boxplus, \boxplus)$, while IDEA uses the order $(\odot, \boxplus, \boxplus, \odot)$, and in the swap of the words after the *MA* subround. In IDEA, the outer words X_1, X_4 are not swapped.

Recommended Reading

1. Lai X, Massey JL (1990) A proposal for a new block encryption standard. In: Damgard IB (ed) Advances in cryptology – proceedings of eurocrypt '90. Lecture notes in computer science, vol 473. Springer, Berlin, pp 389–404

2. Lai X, Massey JL, Murphy S (1991) Markov ciphers and differential cryptanalysis. In: Davies DW (ed) Advances in cryptology – proceedings of eurocrypt '91. Lecture notes in computer science, vol 547. Springer, Berlin, pp 17–38

IPsec

John Ioannidis
Google, Inc., New York, NY, USA

Definition

IPsec is the set of protocols, conventions, and mechanisms that provide security services at the IP layer.

Background

Development of IPsec started at the Internet Engineering Task Force (IETF) in the early 1990s; the latest round of standards documents came out in 2005, and some development is still going on. Originally, IPsec was intended to address all network security needs. Placing security at the network layer meant that the same protocol could be used by different types of links, but also that the same protocol could be used by most network applications to secure their traffic. It turned out that such a goal was too ambitious. In particular, most Web traffic ended up being secured with a transport-level security protocol, SSL (subsequently TLS). As a result, IPsec is now mostly used for Virtual Private Networks (VPNs) and VPN-like applications, such as remote access.

Theory

IPsec Components

IPsec consists of three distinct components: the *traffic protocols* (AH and ESP) to protect the network traffic; *key management* to negotiate the cryptographic keys and parameters to protect the traffic; and a *policy* component, which determines what traffic needs protection and with what parameters.

There are two distinct wire protocols: Authentication Header (AH) and Encapsulating Security Payload (ESP). The reason for having two such protocols, when arguably the features offered by AH can also be offered by ESP, is historical: when IPsec was being standardized in the mid-1990s, there were laws in the USA and other countries preventing the export of products that could perform encryption; thus a version of a product that only supported AH could be exportable. Here we describe ESP; the interested reader can refer to the standards documents for a description of AH.

A fundamental concept in IPsec is the *Security Association*, or *SA*. An SA is a one-way association (note how we avoid the use of the term "connection") that provides security services to the protected traffic. In practice, SAs are negotiated and created in pairs, one for each direction of traffic, but conceivably the "forward" traffic could be protected while the "return" traffic could be in the clear. In practice, an SA is represented by a data structure consisting of *Security Parameters Index* (SPI), a 32-bit number that uniquely describes an SA in the context of each host, the wire protocol (ESP or AH) to use, and the keys and additional configuration parameters to use with those transforms. The list of active SAs in each host is kept in the Security Association Database (SADB). A corresponding data structure, the Security Policy Database (SPD), describes what ought to happen to packets leaving or entering a host. If a packet has an SPD entry but no corresponding SADB entry (thus no SA has been established), the key exchange protocol is triggered to set up a pair of SAs. Note that the SA, the SADB, and the SPD are *concepts*; nothing is said about how they are to be implemented in actual code. In particular, some implementations combine the SPD and SADB functionality with that of the firewalling and forwarding code because packet classification and corresponding actions are involved in all these operations.

The term "cryptographic suite" appears frequently in the IPsec standards documents. Each component of IPsec uses different kinds of algorithms (e.g., encryption, integrity, D-H exchange, pseudo-random number generation), with each having several choices (e.g., Triple-DES or AES for encryption, MD-5 or SHA-1 for integrity, each of which can accommodate different key sizes). Not all combinations make sense, and choice of key lengths should be such that the work factor to "break" each of the algorithms is the same. For this reason, instead of allowing individual choices for each one, many user interfaces, as well as later versions of standard documents, give reasonable premade choices for each of the individual algorithms and present them as a "suite."

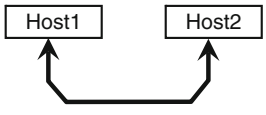
Packet Processing

The wire protocols, or *transforms* as they are referred to in the standards, can operate in two modes: *transport* mode and *tunnel* mode. Correspondingly, we can talk about *transport-mode SAs* and *tunnel-mode SAs*. In transport mode, IP traffic is protected on an end-to-end basis: an IP packet leaving the source host has its payload (everything following the IP header) protected by IPsec; when the packet reaches its destination, the IPsec code examines the now-secured payload, and if it has not been tampered with,

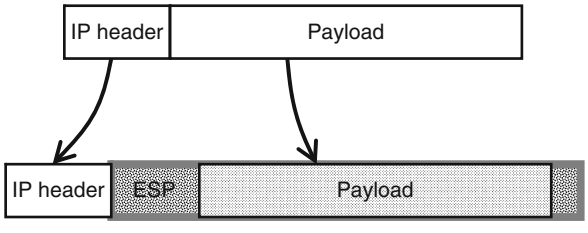
restores the original IP packet and passes it up to the next-level protocol. In tunnel mode, each outgoing IP packet is *encapsulated* in its entirety inside another IPsec packet. This "outer" packet may have different source and destination IP addresses from the "inner" packet. Conceptually, tunnel mode can be considered as a concatenation of an IP-in-IP encapsulation protocol and transport mode, where the new "transport" protocol is the aforementioned IP-in-IP encapsulation protocol. Conversely, transport mode can be considered an optimization when the "inner" and the "outer" IP addresses would be the same.

Let us now illustrate the three common use cases of IPsec and the corresponding modes to use. Figure 1 shows two random Internet hosts wishing to communicate securely. The mode they use is *transport* mode. After key and security-association negotiations, every IP packet leaving one host gets its payload protected with ESP (Fig. 2). The IP header stays largely the same: the only difference is that the IP Protocol field gets replaced with the value for ESP (protocol 50), and the original protocol field describing the payload (e.g., 6 for TCP or 17 for UDP) gets carried as part of the ESP encapsulation data.

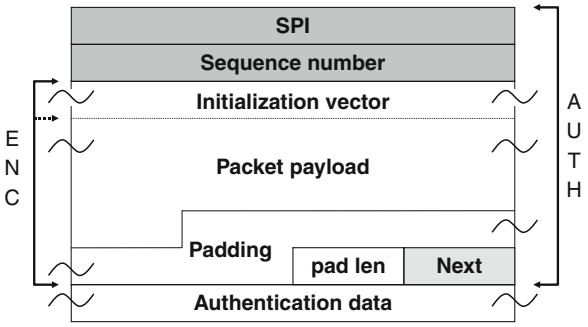
To understand exactly how the ESP transform works in the most common case, where ESP is used for both integrity protection and data confidentiality, observe Fig. 3. First, the SPI was determined by consulting the SADB and finding the corresponding SA. The Sequence Number field is a monotonically-increasing number, also kept in the SA, and its function is to protect against replay attacks. Most encryption algorithms require an initialization vector (IV); one is created at random. The original packet's payload is placed after the IV. Most encryption ciphers used in IPsec are block ciphers, and thus padding is added up to a multiple of the cipher block size minus two. Some transforms



IPsec. Fig. 1 Two hosts communicating in transport mode



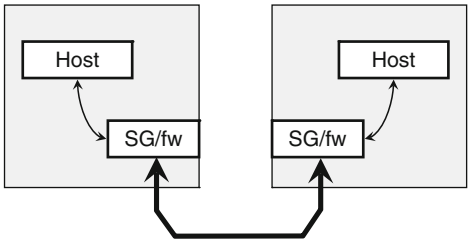
IPsec. Fig. 2 Packet processing in transport mode



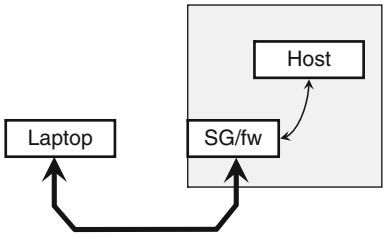
IPsec. Fig. 3 Encapsulating Security Payload packet construction

may allow for paddings that include additional multiples of the block size, for example, to frustrate traffic analysis that may depend on packet size. One more octet indicates the amount of padding used, and a final octet carries the original protocol number. This entire construct (payload plus padding plus length plus protocol) is encrypted with the block cipher. Obviously, the IV is not encrypted, but it stays in the packet so that decryption can happen at the receiving end. Finally, the message authentication code is applied on the encrypted payload and the SPI and Sequence Number fields, and the resulting packet is transmitted. When the packet is received at the far end, the same operations are carried out in reverse order. The SPI is consulted to find the corresponding SA. The MAC is applied to the SPI-plus-SN-plus-encrypted payload construct, and the result is compared to the transmitted authentication data. If they are equal, the packet has not been tampered with. The payload is the decrypted, the padding is removed, all the extra fields are discarded, the IP protocol field in the IP header is restored, and the packet is passed up the network stack for further processing.

While transport mode is sufficient to provide security services, practical considerations such as lack of IPsec software on end hosts, the need to centrally administer security policies (such as on a firewall or security gateway) render having a tunnel mode be an integral part of IPsec useful. Figures 4 and 5 show the two common cases: in the first, two or more distinct sites join to form a Virtual Private Network. A host in one site does not necessarily know that its peer is on another physical network; from a routing perspective, a tunnel is just a virtual link, and this one happens to be encrypted. Moreover, each site may be numbered out of private address space, which means that tunneling is the only way traffic from one site can reach the other. The security gateways will generally have only one IPsec tunnel between them: the traffic selectors in their



IPsec. Fig. 4 Two hosts communicating over a Virtual Private Network. The security gateways are running IPsec in tunnel mode; the hosts are unaware of any security processing taking place



IPsec. Fig. 5 The "road warrior" communicates with its security gateway over IPsec in tunnel mode. Hosts inside the protected domain are also unaware of any security processing

corresponding SPDs will specify the entire address range from each site.

Another common use of tunnel mode is the "Road Warrior"; a user in some remote location needs to connect to their main network securely. A tunnel-mode SA is set up between the remote machine and the security gateway. Traffic from the remote machine to its home network is protected by that tunnel and delivered to the security gateway, which in turn passes it on to its eventual destination. An interesting configuration question here is whether to route *all* traffic from the remote machine via the tunnel to the security gateway, or only traffic destined for the home network. To wit, if the road warrior needs to access internal resources and also to browse the web, should the web-browsing traffic first be tunneled through to security gateway and then routed to the Internet, or should it simply use its local interface? This actually is a security issue, not just a configuration issue, as allowing direct Internet-access to the remote machine while the VPN session is in effect may create a traffic path that bypasses the firewalling mechanisms of the site. A properly configured and managed installation should not have such problems, but they have been known to happen.

Key Exchange

The Internet Key Exchange, version 2 (IKEv2) protocol is responsible for negotiating Security Association parameters and establishing and maintaining cryptographic keys between IPsec endpoints. All IKEv2 communications occur with message pairs between the *Initiator* and the *Responder*. Which endpoint is which is merely a matter of timing; whichever happens to start the process becomes the Initiator, and the other becomes the Responder. Each such message pair is called an *exchange*; unsurprisingly, an exchange consists of a *request* and a *response*. The first pair of exchanges occur once and establish all the information necessary to populate SA data structures of both sides for the particular connection. A third exchange establishes the traffic keys that are used by the IPsec transforms. Because hosts can fail, reboot, etc., an Informational exchange is also defined for management during the lifetime of the SAs.

Any key management protocol, and IKEv2 (and its predecessor, IKE) is no exception, has to work in a somewhat peculiar environment: its own payloads must be cryptographically protected, but obviously IPsec cannot be used for that purpose. Because it runs over UDP, and worse, because it has to run in the presence of NAT, there are limits to the exchange packet sizes, which required very careful design of the protocol.

The first exchange, called IKE_SA_INIT, creates a pair of IKE Security Associations (not to be confused with IPsec SAs). The Initiator sends the responder the SPIs, the list of cryptographic suites it supports for negotiating the IKE SA, its Diffie–Hellman value, and a nonce. The responder sends its own SPIs, chooses one of the suites offered, presents its D-H value and nonce, and optionally requests certificates for the initiator. At this point, both sides can compute the shared D-H secret, called SKEYSEED, which is used by all subsequent operations to generate session keys and traffic keys.

The second exchange, called IKE_AUTH, has each side send, protected by keys derived from SKEYSEED, its own identity, corresponding certificates and certificate requests, and the traffic selectors to use in establishing the actual IPsec SAs.

The third exchange, called CREATE_CHILD_SA, actually establishes the IPsec SAs. The initiator sends the set of IPsec cryptographic suites it supports, and the responder selects one of them. CREATE_CHILD_SA can be repeated as often as desired to establish new traffic keys. The usual reason this is done is because the Sequence Number is about to wrap around (in cases of high traffic rates), or because policy dictates that traffic keys should not stay unchanged for more than a certain amount of time.

There are some additional options that may be specified during the CREATE_CHILD_SA exchange, as well as an INFORMATIONAL exchange which is used to pass control messages. There are also options to enable external authentication systems to be used in lieu of certificates for mutual authentication of initiator and responder. While all this may appear very complex, IKEv2 is actually a much simpler protocol than the one it replaced (IKE), which arguably had too many control "knobs."

IPsec and NAT

Network Address Translation (NAT) is a reality that IPsec has had to face ever since the first "road-warrior" clients appeared. A NAT device can interfere with IPsec in several ways: it may not even pass protocols 50 and 51 (ESP and AH); even if it passes them, there is no specification of how the equivalent of address and port translation should happen, i.e., what to use as a port number; the private IP address of the road-warrior may be used by an unsuspecting key management daemon to pass its IP address to its peer; the original version of IKE expected all traffic to use UDP port 500, but some devices change the source port of translated traffic.

These problems were addressed with IKEv2 and the 2005 revision of the IPsec protocols, which adds the use of UDP port 4500, defines an encapsulation mechanism over UDP. The encapsulation is trivially simple: an ESP header follows the UDP header. Because the same ports are used for IKE traffic, and also for NAT keepalives, the SPI in the ESP header cannot be 0; four consecutive 0 octets following the UDP header indicate that the payload is IKEv2 (or IKE), not ESP. Also, a single 0xff octet is used as a NAT keepalive. The interested reader is referred to the corresponding RFCs: 3947 and 3948.

Policy and Configuration

The architecture document (RFC4301) specifies the concept of the Security Policy Database, but this is done in an abstract way and mostly just to have a consistent way of talking about the problem in subsequent documents. Several attempts were made over the years to standardize the configuration/policy model for IPsec. There are actually two independent problems: configuring the filters, or conditions in general, whereby outgoing traffic should be protected and, correspondingly, incoming traffic would be expected to arrive protected; and also provide a way for applications to know whether they are communicating securely.

Most of the effort has been spent on the former problem. At least one early IPsec implementation under OpenBSD used a policy management language (technically, the KeyNote Trust-Management system) to configure policy in great detail. The IP Security Policy (IPSP) working group produced two documents (RFC 3585 and 3586) which were, however, only informational. A Security Policy Database MIB was eventually also produced (RFC 4807). Much of the configuration work, however, remains system-specific, with each implementation having its own ways of specifying all the various parameters.

Since IPsec operates at the network layer, higher-layer protocols and applications are not necessarily aware of its existence. There are cases, however, when it may be convenient for an application to know whether traffic it is exchanging with another remote application is secured (e.g., a mail user agent talking to a mail repository). While this kind of feature was envisioned early in the development stages of IPsec, it turned out that a transport-layer security protocol (SSL/TLS) was much better suited for this kind of interaction.

Recommended Reading

1. IPsec is still evolving. As of this writing, the core set of standards documents are RFCs 4301-4309. Many other RFCs address individual aspects. The interested reader is referred to the IETF web page (http://www.ietf.org for the authoritative RFC repository.

IPSec Policy Analysis

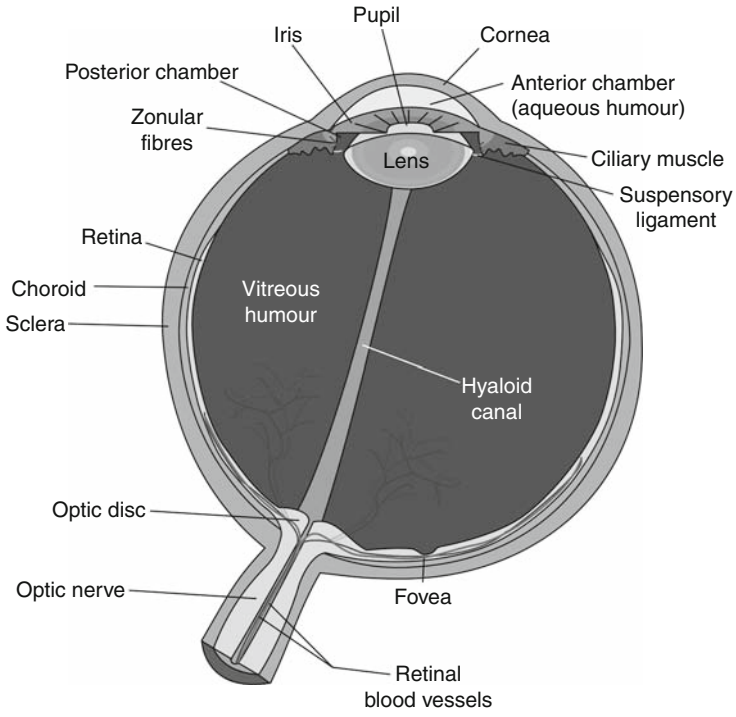
►Firewalls

Iris

Yu Chen, Malek Adjouadi
Department of Electrical and Computer Engineering, Florida International University

Definition

Iris is the colored organ inside the eye. In the human eye, as illustrated in Fig. 1, the iris is a thin diaphragm that lies behind the cornea and anterior chamber. The muscles of the iris expand and contract the aperture of the iris (also known as pupil) to adjust the amount of light which passes through the lens [14]. In the information technology field, automated pattern recognition of the human iris can be used to identify each person for security purposes and access control (Fig. 1).



Iris. Fig. 1 An anatomical view of the human eye

Background

Interestingly, the main structures of the iris begin to form in the third month of gestation and are completed by the eighth month [2]. The accretion of iris pigment continues within the first postnatal years [10, 14]. Consequently, the structures of the iris are said to be unique to an individual and do remain stable with age [6].

The overall structure of the iris consists of the following two layers: the anterior layer and the posterior layer. The anterior layer contains the stroma, iris sphincter muscle, iris dilator muscle, and anterior pigment myoepithelium [10]. The iris sphincter muscle and dilator muscle contract and expand to reduce or enlarge the pupil size, respectively. The posterior layer of the iris mainly contains the pigmented epithelial cells beneath the anterior layer [18].

Theory

The patterns of the iris are caused by many various features of the anterior layer: the arching ligaments, interlacing ridges, contraction furrows, corona, among others [12]. These features are based on the structure of the anterior layer surface and also the tissues beneath the surface such as blood vessels, and stroma support. The iris color which is usually referred to as the eye color is mainly determined by the anterior melanin pigment [18]. The pigment can absorb the short-wavelength light. Thus, if the iris is less pigmented (has less density of the iris pigment), the iris appears as light colored (grey, blue, and green), otherwise, the iris appears dark colored (brown and black).

Although the basic structures of the irises are generally common, the pattern of each human iris is distinct. The genetic differences and the uniqueness of the circumstances of each person, i.e., the condition in the embryo development give the fact that the detailed pattern of each human iris is unique. Through clinical observation, it has been claimed that even the patterns in the same person's two eyes are believed to be different.

Applications

The human iris serves as a perfect physical feature for biometric applications, because of its uniqueness and stability. Furthermore, the patterns of irises cannot be changed incidentally or intentionally, since the iris is a protected organ inside the human eye. Among various biometric measures which include face recognition, iris recognition, fingerprint recognition, and voice recognition, among others, iris recognition is the most reliable and accurate one. Currently, iris recognition remains as one of the most popular methods that are widely used in security-related applications.

Traditional Iris Recognition

Since the first automatic iris recognition system was proposed by J. Daugman in 1993, a variety of commercial systems are developed to deal with the eye images and conduct identification or verification processes [5, 6]. Most of the established iris recognition systems rely on four main steps during the iris recognition process. These steps are iris information (still image/video) acquisition, iris segmentation, iris pattern analysis, and iris pattern matching for eventual recognition.

Most of the existing commercial systems adopt near-infrared (NIR) illuminations (with the wavelength between 700 and 900 nm) as a source of lighting. It is believed that with NIR, the iris shows more detailed patterns as the iris tissues absorb the short-wavelength light and let the long-wavelength go through. During the iris acquisition step, the user (object) is usually still and looks at the camera, and the acquisition system access the focus of the iris and automatically adjust the camera to take the iris images or videos.

The iris segmentation step isolates the iris region from the eye images. Some of the segmentation strategies consider the boundaries of the pupil and the iris to be circular in shape. The circles of pupil and outer iris (limbic) boundaries are detected in order to localize the iris which is the part within these two circles. In this step, noise reduction methods are used to attenuate to some degree the inherent noise effects and preprocessing methods are used to deal with occlusions caused by eyelids, eyelashes, and reflections.

In the iris pattern analysis step, various filters are used by most iris recognition systems to encode the texture of the iris. The decision of iris verification and identification is generated in the iris matching step. In the matching step, the distances between codes generated for the irises are calculated, and a threshold of the distance is usually used to determine if any two codes belong to a same iris. For the application of verification, the system would provide a degree of similarity in subject's codes within a stored database. For the identification process, the system would identify the subject based on whether the degree of similarity has crossed a set threshold that guarantees the highest degree of certainty in comparison to all the stored iris codes in the database.

Among most of the established iris recognition systems, Daugman's approach and Wildes' approach remain the most important and well-known approaches.

Daugman's Approach

The first automatic iris pattern encoding and recognition method was proposed by Daugman. Since then, the basic idea of Daugman's original approach has led to several other research findings and to the development of commercial products on iris biometrics [2].

The monochrome cameras with NIR illumination lighting sources are used in the iris acquisition step of Daugman's system. The total high-frequency power in the 2D Fourier spectrum of each captured image/frame is calculated to access the focus of the iris. The camera is adjusted to focus on the iris and a feedback (voice or vision signal) is given to help the user position his/her head within the camera's field of view.

In the iris segmentation step, the pupil boundary and limbic boundary are treated as two nonconcentric circles. Each circle is detected separately and described by their center coordinates and radius. An integro-differential operator is used to search for the boundary circles. This particular integro-differential operator is defined as:

$$\max_{(r,x_0,y_0)} \left| G_\sigma(r) * \frac{\partial}{\partial r} \oint_{r,x_0,y_0} \frac{I(x,y)}{2\pi r} \mathrm{d}s \right| \tag{1}$$

where $I(x,y)$ is an intensity value of the image domain (x,y). This equation seeks the maximum value in the blurred partial derivative with respect to increasing radius r, along an arc denoted by (ds) of a circle with center coordinates (x_0,y_0) and radius r. The $G_\sigma(r)$ is a Gaussian smoothing function of a scale set by σ, and the symbol $(*)$ refers to the convolution operation.

The upper and lower eyelids are modeled as two arcs, and the arcs are detected with this integro-differential operator mentioned above. The difference is the path of contour integration in the operator that is changed from circular to arcuate.

Because the image sizes of the irises are different and the degrees of pupil dilations vary based on the illumination, a normalization method is used to transform the segmented iris image to a normalized polar space. In the normalized polar space, every pixel in the iris image is represented by an angle between 0° and 359° and a radial coordinate between 0 and 1.

A two-dimensional (2D) Gabor wavelet filter is then used to extract the textures from the normalized iris image. The encoded texture is represented as a binary code and compared with stored iris codes to generate a final recognition decision. During the comparison, the Hamming distance is calculated to measure the difference between two iris codes. A smaller distance indicates a better confidence of the similarity between the two iris codes. A threshold of the distance would be used to decide if the two iris codes belong to the same iris or not.

It is reported that Daugman's approach achieves very high accuracy results on the basis of a threshold Hamming

distance of 0.330 over 2.3 million comparisons between different irises.

Wildes' Approach

The Wildes' approach is also very prominent in the field of iris recognition. It uses different image acquisition and iris segmentation process which gives it some advantages over Daugman's system in some aspects.

In the acquisition system, a more complicated set-up, which includes "*a diffuse source and polarization in conjunction with a low light level camera*" is used to eliminate the reflection spots [21]. In the iris segmentation step, the circular Hough transform is used to detect the pupil and limbic boundaries. The Hough transform is known to be tolerant to gaps in edge descriptions and is relatively unaffected by image noise. The well-known circular Hough transform is often defined as:

$$H(x_c, y_c, r) = \sum_{j=1}^{n} h(x_j, x_j, x_c, y_c, r) \tag{2}$$

where $h(x_j, x_j, x_c, y_c, r) = 1$ if the edge point (x_j, y_j) is on the circle with center (x_c, y_c), and radius r; otherwise, $h(x_j, x_j, x_c, y_c, r) = 0$. With such a circular Hough transform, each edge point in the image space votes for each possible circle passing it in the parameter space. The maximum value of $H(x_c, y_c, r)$ indicates the target circle which has a center at (x_c, y_c) and a radius of r.

In the iris pattern analysis step, a Laplacian of Gaussian (LOG) filter with multiple scales is used to analyze the iris texture, and the filtering results are compared without binarizing to a compact representation as is in the case in the Daugman's approach. Thus, more feature details may be extracted and compared when using the Wildes' method.

It is claimed that no false positives or false negatives are experienced in the evaluation of the Wildes' approach; however, this evaluation was based on a smaller database which contains 600 iris images from 40 different persons.

Open Problems and Future Directions

Less-Constrained Iris Biometrics

In all of the four steps of iris recognition, the iris information acquisition step which captures the iris images or videos of the users (subjects) is the most restrictive. It is this initial and only step that needs users' cooperation. Thus, the most distinguishing difference between traditional iris recognition and unconstrained iris recognition is in the set-up of the iris acquisition system to free the user from any constraint.

Amongst most of the conventional iris recognition systems, iris acquisition is the most time-consuming step and is considerably inconvenient to users. To obtain the iris images with required ideal or demanding qualities for traditional iris recognition system, various rigid constraints are imposed on the subject's stands, head alignments, movements, light illuminations, etc. Most of the unconstrained iris recognition systems benefit from their innovative endeavors in the more convenient iris acquisition setups. Among those, the "Iris on the move" (IOM) system and visible wavelength systems are the two most well-known and promising prototypes.

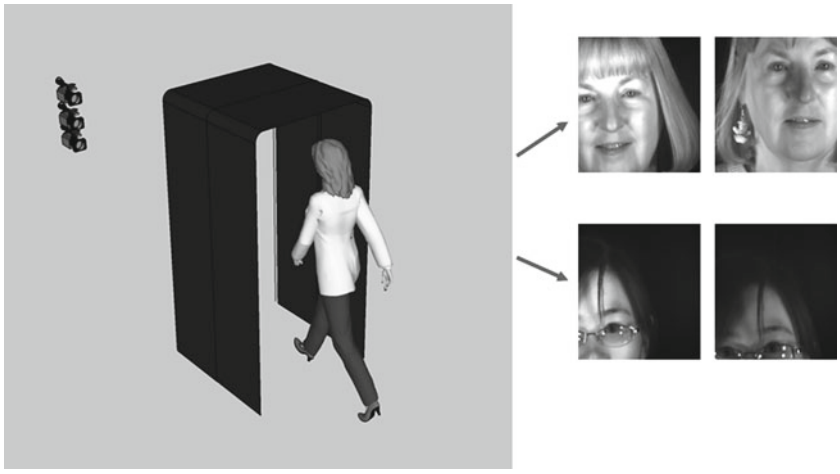
Because of the innovative iris acquisition setups, the properties of the iris images captured by less-constrained iris recognition systems are different and more challenging, compared with those obtained from the more controlled conventional iris recognition systems. Most challenging are the severe noise effects that are inherent to these unconstrained iris recognition systems. Typical sources of noise include motion blur, defocus, eyelashes or eyelids obstruction, specular reflection, among others. These noise effects could make it impossible for the system to normalize, encode and generate accurate matching results. Thus, new alternate strategies and algorithms are proposed by the biometrics research community to compensate for those newly introduced difficulties and obtain acceptable recognition performances under unconstrained environments [4]. The rest of this article gives an up-to-date overview of unconstrained iris recognition systems and related techniques.

The IOM Approach

Matey et al. introduced an iris recognition system called IOM, which is based on near infrared (NIR) video taken at a distance, with its targets being moving subjects. One of the significant merits of the IOM system is that it allows for iris identification or verification even when a subject is walking at a normal pace at a speed of up to 1 m/s [13].

The scenario of the iris recognition under the IOM system is illustrated in Fig. 2. The prototype uses three high resolution video cameras toward a portal at a distance of about 3 m. The NIR illumination sources are embedded within the portal. The NIR video is taken when the subject walks through the portal, and the subject could be wearing eyeglasses or contact lenses.

For iris verification or identification, the NIR videos of subjects' faces are taken with a resolution of $2,048 \times 2,048$. The iris images acquired from these videos are different from those acquired through traditional acquisition methods. For most of the traditional methods, the total number of pixels across the iris is usually more than 200, but eye images from the videos only offer about 120 pixels across the iris part.



Iris. Fig. 2 Image acquisition setup for the IOM system

The subject to be pictured should be at an ideal distance for the fixed focus video camera to take the desired frames of the subject's face. Through their study, Matey et al. claim that those frames which are taken with a depth of field at about 12 cm depth interval yield the perfect focus distance which can be accepted for iris verification. While conducting iris verification with the IOM system, the video is taken at 15 frames/s and the subject's walking pace is allowed to be approximately 1 m/s. Consequently, there are no more than two frames that can be captured within that 12 cm depth of field. For subjects that are on-the-move, the iris images extracted from limited frames may suffer from various kinds of noisy effects caused by off angle, motion blur, occultation caused by eyelid or eyelashes, and unexpected light reflections.

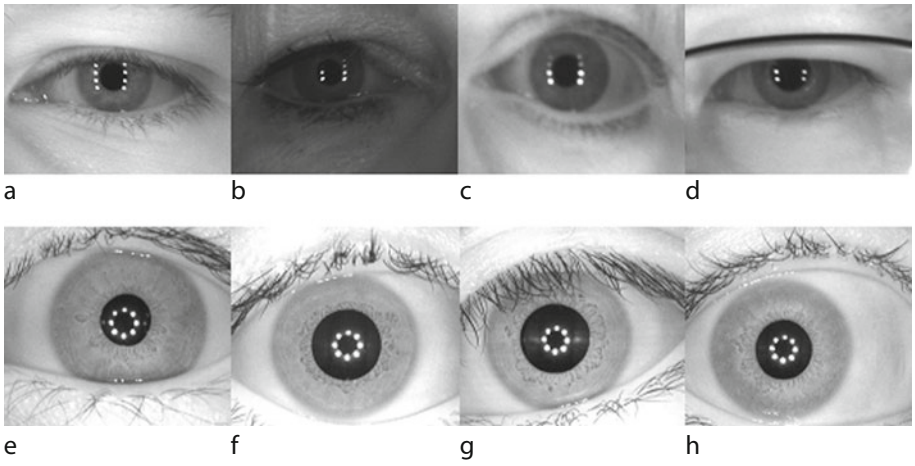
To appreciate the complexity of the problem, some examples of iris images obtained from the IOM system and from the traditional system are provided in Fig. 3 for comparative purposes.

Matey et al. adequately proved the feasibility of the IOM system [13]. Through their experiments, it is claimed that the overall recognition rate for 119 subjects is 78% and with some improvement on the acquisition process, the success rate could rise up to as high as 99%. They conclude that the IOM system "can capture iris images of recognition quality from subjects walking at a normal pace through a minimally confining portal."

The detailed segmentation approach for IOM system is proposed by Chen et al. [3]. Their proposed approach starts from the eye image extraction which is based on the reflections on the eyes generated by the illumination system. An adaptive histogram method is used to detect the pupil. The searching of limbic boundary and eyelids benefits from the obtained pupil information. The outer boundary of the iris was detected by a modified Hough circular transform. A new eyelids and eyelashes detection method was introduced in this approach to overcome the noise from such unconstrained images, which do affect the detection process. In their approach, the pixels which are on the edge between iris and eyelids or eyelashes will be detected first, then the Hough line transform would be performed. Their experimental results based on the MBGC database shows the approach yielding a 93% segmentation accuracy rate.

Lee et al. proposed an "object detector" algorithm based on the Viola–Johns method combined with a modification made by Lienhart–Maydt to detect the eye in the NIR video. Then, an automatic segmentation algorithm based on the Wildes et al. method is used to localize the iris. The feature extraction is conducted by a 1D Log-Gabor filter. The Hamming distance (HD) measure is used in the matching stage. Their experimental results shows that an eye detection accuracy of 97.69% in two-eye detection and of 81.50% in the iris region segmentation process. The claimed matching rate is reported to be greater than 54%.

An iris image evaluation method is proposed by Zhou et al. to select the extracted NIR iris images with acceptable quality for recognition in the IOM system [22]. Their image evaluation strategy includes a so-called quality filter unit to delete the low-quality images after image extraction, a segmentation evaluation unit and a segmentation scores unit to evaluate the segmentation accuracy, and a score fusion unit to combine the quality and segmentation measurements to generate a confidence score in the recognition stage. The two-dimensional Gabor wavelet method and the 1-D-Log-Gabor wavelet method are used for the



Iris. Fig. 3 Figure1 eye images acquired from the IOM system and the traditional iris acquisition method. Eye images (**a, b, c, d**) are extracted from the IOM system (Courtesy of MBGC Database), while eye images (**c, d, e, f**) are from the conventional acquisition system as provided by the CASIA database version 3. (Image provided courtesy of CASIA-IrisV3, http://www.cbsr.ia.ac.cn/IrisDatabase)

recognition stage. They claim that with the proposed image evaluation method, the genuine acceptance rate (GAR) increases by 407.1% and 216.8%, respectively, in the left and right eyes, compared with the GAR obtained with traditional methods at the same false accepted rate of 10^{-4}.

The Visible Wavelength Approach

One of most difficult and important issues for the design of a less-intrusive iris recognition system is the trade-off between the demanding irradiance and the illumination safety requirements for the eyes. Being able to capture iris images of the subjects with on-the-move and at-a-distance conditions, the acquisition system needs to achieve larger depth-of-field with considerable short exposure time. Thus, increased intensity of irradiance would be necessary for the optical system in the acquisition step. It is indicated that the use of visible wavelength illumination would be better and safer to achieve such a lighting requirement. Different with the NIR illuminations, the visible wavelength illumination makes people react instinctively, as a safety measure, with eye blinking, pupil contraction/dilation, and evasion, if the illumination happens to be intense. A visible wavelength (VW) prototype approach was proposed by Proença et al. in University of Beira Interior [15, 16]. And more research interests have currently been drawn to this idea.

With the VW system, the iris images can also be taken when the users are at a given distance. The system can also capture iris images when users are on the move and with different pose or head alignments. Furthermore, the system uses the visible light instead of the NIR illuminations in order to offer better safety for the users.

The experimental setup of the iris acquisition adopted in the visible wavelength approach is illustrated in Fig. 4. The commercially available camera (Canon EOS 5D in the proposed prototype) is used as the iris image capture device. The acquisition system is designed to function in an environment without rigid illumination control. In their prototype, the setup of the imaging system works under both natural light source and visible artificial light source.

The system is claimed to be able to capture multiple iris images when the subject is walking at a slightly slower pace within a range of distances between 4 and 8 m [16]. With such an imaging system, the proposed iris recognition system can tolerate occasional effects of movements such as blinking, turning the head, and looking askance.

Because images are taken with the less-constrained conditions on illumination, subject's movements, poses, and head alignments, the iris images captured with the Proença's acquisition system generate much more realistic noise conditions, such as specular reflections, off-angle situation, and the wearing of glasses. Some example images are shown in Fig. 4. Besides noise effects, the visible wavelength characteristic of iris images shows substantial differences between the iris images captured by visible wavelength system and images obtained with traditional iris recognition systems. Because of such differences, conventional iris segmentation approaches have not yet been able to process visible wavelength iris images; and hence a lot of research efforts have been made with the focus on

Iris. Fig. 4 The setup of the image acquisition prototype, signs of **A, B** in the image denote the two cameras; **C, D** are the artificial and natural light sources; **E** is the moving subject. (Image provided courtesy of Dr. Hugo Proença)

Iris. Fig. 5 Iris image examples captured by the Proença's approach (Courtesy of UBIRIS V.2 Database) [16]

the iris segmentation stage toward more effective visible wavelength-based systems (Fig. 5).

A segmentation approach starting with a clustering-based coarse iris localization method is proposed in [19]. The performance of their approach greatly benefits from this novel initial processing step. The specular reflection removal followed by a region growing strategy is conducted to cluster the iris image into coarse iris region and skin region. Then, with the considerations of semantic priors, the iris region is refined and non-iris regions including eyelashes, eyebrow, eye glasses, and hair are also detected. Also, an enhanced integro-differential operator is used to detect the pupil and limbic boundaries. The eyelid is detected by a 1-D horizontal rank filter, adopting an eyelid curvature model. In their final step, the eyelash occlusions are excluded, using parametric models based on the intensity statistics of different iris regions.

In the approach proposed by Sankowski et al., the specular reflections are detected and filled in a YIQ color space [17]. Then, again the conventional integro-differential operators are used to detect the pupil and limbic boundaries. Finally, parametric models based on the properties of the eye image are used to describe the upper and lower eyelids.

A knowledge-based segmentation approach was later proposed by Almeida, an idea that was inspired by the paradigm of expert systems [1]. Thus, a set of rules are

defined to drive the system as a "human expert." Those rules are set based on the perceived natural properties of the human eye such as "the pupil should be a very dark small circle," "the centers of the pupil and the iris should be close," "region around the pupil should have a reasonable color of iris," as semantic priors. The approach consists of the following steps: image pre-processing, pupil and iris localization, and the combination of pupil and iris eyelids detection.

The so-called AdaBoost algorithm was adopted in both of the segmentation approaches proposed by Li et al. and Jeong et al. to detect a coarse location of the eye [9, 12]. In Li et al. approach, a novel limbic boundary identification strategy is proposed. It uses K-means clustering based on the co-occurrence histogram to localize the limbic boundary. They use a new parabolic integro-differential operator combined with a RANSAC (random sample consensus)-like method to detect the upper eyelid boundary. On the other hand, in Jeong et al. approach, they adopt the color information of the image for detecting the obstructions caused by the image ghosting effects. And the identification of the corneal specular reflection is used to determine if there is an "open eye" appearing in the image.

With the same intent, Chen et al. proposed an approach to achieve both accurate segmentation and fast processing speed [4]. The proposed approach relies on an effective search for the sclera area of the image. A threshold of saturation value of the HSI color model is obtained by calculating the biggest group derivative of the original color image histogram. A binary map is then generated to indicate the sclera area. The method determines a more refined target area in order to accelerate the circle searching for the outer iris boundary. The outer boundary of the iris was detected using a very fast and accurate modified circular Hough transform. The linear Hough transform is then used recursively to extract the edges of eyelids. A novel new method of verification and correction for the noncircular outer iris boundary is also developed. It is claimed that their approach achieves an accuracy higher than 97% with an execution speed of 0.83s per image, which is quite significant.

Labati et al. applied an intro-differential technique to roughly localize the center and radius of the outer iris [7]. The search region for pupil is significantly reduced by this estimation method. Polar transformation is used to linearize the estimated region of iris and pupil boundaries. Thus, two obtained image strips containing iris boundaries are processed to define the accurate location of the pupil and the outer iris boundary of the iris.

The recognition process remains in the discussion stage among the biometrics community due to the extreme challenges posed by illumination, capturing distance, and subject movement among others.

However, Proenca et al. proved the feasibility of the recognition process by applying Daugman's traditional recognition strategy with manually segmented color iris images [15]. They selected 1,000 segmented good quality iris images obtained from visible wavelength system as templates. Then, they compared them with 10,000 non-iris or partial iris images and 10,000 natural and synthetic texture images, generating a false match rate at $P(s < 0.33) \approx 1.03 \times 10^{-12}$, which can be considered to be negligible. They point out that the visible wavelength iris recognition system has potential to produce an extremely low false match rate, which is viewed favorably in biometrics applications.

Other Less-Constrained Iris Recognition Approaches

Wheeler et al. present a minimal-user-intrusive iris recognition system [20]. Their work is focused on the development of an automatic iris acquisition system to the convenience of the users. The system uses a pair of fixed wide-field-of-view (WFOV) surveillance cameras to detect the head position with stereo vision. A pan-tilt head is used to direct the NIR illuminators and the iris camera to focus on the subject. Thus, the system allows the user to stand in front of the camera and face it for carrying out the iris verification process. A classic Daugman-style approach is used to perform the segmentation and recognition processes. They claim that, with the use of a normalized Hamming distance of 0.3, their system achieves the verification false recognition rate of 10^{-6}, and the whole iris identification process can be completed in 3.2 s on an average.

Furthermore, an iris image deblurring method is proposed by Huang et al. to improve the quality of the images with defocus or motion blur defects for less-cooperative iris recognition systems [8]. Since in the less-constrained iris recognition scenarios, the user is usually allowed to walk or move, the movement or defocus degrades the image quality significantly; thus, the intent in this study was to improve the recognition process by introducing a deblurring method. They apply a depth sensor in the iris acquisition system to obtain the 3D depth information. Their deblurring algorithm is based on the depth information and prior knowledge on the iris image. Although their segmentation and recognition algorithms are not fully detailed for a thorough assessment, they claim that the application of the deblurring method can reduce the mean of the authentic distribution by 12%.

I

Recommended Reading

1. Almeida P (2009) A knowledge-based approach to the iris segmentation problem. Image Vis Comput 28(2):238–245
2. Bowyer KW, Hollingsworth KP, Flynn PJ (2007) Image understanding for iris biometrics: a survey. Comp Vis Im Understanding 110(2):281–307
3. Chen Y, Wang J, Adjouadi M (2008) A robust segmentation approach to iris recognition based on video. Proc IEEE-AIPR pp 1–8, ISBN:978-1-4244-3125-0
4. Chen Y, Adjouadi M, Han C, Wang J, Barreto A, Rishe N, Andrian J (2010) A highly accurate and computationally efficient approach for unconstrained iris segmentation. Image Vis Comput 28(2):261–269
5. Daugman JG (2001) Statistical richness of visual phase information: update on recognizing persons by their iris patterns. Int J Comput Vis 45(1):25–38
6. Daugman JG (2004) How iris recognition works. IEEE T Circ Syst Vid Technol 14(1):21–30
7. Donida Labati R, Scotti F (2009) Noisy iris segmentation with boundary regularization and reflections removal. Image Vis Comput 28(2):270–277
8. Huang X, Ren L, Yang R (2009) Image deblurring for less intrusive iris capture, Proceedings of IEEE conference on computer vision and pattern recognition CVPR 2009, 20–25 June 2009, Miami, FL, pp 1558–1565
9. Jeonga D, Hwanga J, Kangb B, Parka K, Wonc C, Parkd D, Kim J (2009) A new iris segmentation method for non-ideal iris images. Image Vis Comput 28(2):254–260
10. Kanski JJ (2007) Clinical ophthalmology : a systematic approach. Butterworth-Heinemann/Elsevier, Edinburgh, New York
11. Lee Y, Phillips PJ, Micheals RJ (2009) An automated video-based system for iris recognition. Proceedings of the third international conference on advances in biometrics, 2–5 June, Alghero, Italy, Oct. 15–17, 2008, pp 1160–1169
12. Li P, Liu X, Xiao L, Song Q (2009) Robust and accurate iris segmentation in very noisy iris images. Image Vis Comput 28(2): 238–245
13. Matey JR, Naroditsky O, Hanna K, Kolczynski R, LoIacono D, Mangru S, Tinker M, Zappia T, Zhao WY (2006) Iris on the move: acquisition of images for iris recognition in less constrained environments. Proc IEEE 94(11):1936–1946
14. Oyster CW (1999) The human eye: structure and function. Sinauer Associates, Sunderland, USA
15. Proença H (2009) On the feasibility of the visible wavelength, At-A-Distance and On-The-Move iris recognition. IEEE symposium series on computational intelligence in biometrics: theory, algorithms, and applications, Nashville, Tennessee, USA, March 30–April 2, vol 1, p 9–15, ISBN:978-1-4244-2773-4. (invited paper)
16. Proença H, Filipe S, Santos R, Oliveira J, Alexandre LA (2009) The UBIRIS.v2: a database of visible wavelength iris images captured On-The-Move and At-A-Distance. IEEE T Pattern Anal Mach Intelligence, doi:10.1109/TPAMI.2009.66
17. Sankowski W, Grabowskia K, Napieralskaa M, Zuberta M, Napieralski A (2009) Reliable algorithm for iris segmentation in eye image. Image Vis Comput 28(2):231–237
18. Snell RS, Lemp MA (1998) Clinical anatomy of the eye. Blackwell Science, Malden, MA, USA
19. Tan T, Hea Z, Sun Z (2009) Efficient and robust segmentation of noisy iris images for non-cooperative iris recognition. Image Vis Comput 28(2):223–230
20. Wheeler FW, Perera AGA, Abramovich G, Yu B, Tu PH (2008) Stand-off Iris recognition system, Proceedings of 2nd IEEE international conference on biometrics: theory, applications and systems. BTAS 2008, Sept. 29-Oct.1, 2008, Washington DC, pp 1–7
21. Wildes R (1997) Iris recognition: an emerging biometric technology. Proc IEEE 85:1348–1363
22. Zhou Z, Du Y, Belcher C (2009) Transforming traditional iris recognition systems to work on non-ideal situations. IEEE T Industry Electronics 56(8):3203–3213

Irreducible Polynomial

Burt Kaliski
Office of the CTO, EMC Corporation, Hopkinton MA, USA

Related Concepts

►Extension Field; ►Field; ►Finite Field

Definition

A polynomial that is not divisible by any smaller polynomials other than trivial ones is an *irreducible polynomial*.

Theory

Let $f(x)$ be a polynomial

$$f(x) = f_d x^d + f_{d-1} x^{d-1} + \cdots + f_1 x + f_0,$$

where the coefficients $f_0, \ldots, f_d$ are elements of a ►field **F**. If there is another polynomial $g(x)$ over **F** with degree between 1 and $d - 1$ such that $g(x)$ divides $f(x)$, then $f(x)$ is *reducible*. Otherwise, $f(x)$is irreducible. (Nonzero polynomials of degree 0, i.e., nonzero elements of **F**, divide every polynomial so they are considered "trivial" factors.)

As an example, the polynomial $x^2 + 1$ over the finite field $\mathbf{F}_2$ is reducible since $x^2 + 1 = (x + 1)^2$, whereas the polynomial $x^2 + x + 1$ is irreducible.

A representation of the ►finite field $\mathbf{F}_{q^d}$ can be constructed from a representation of the finite field $\mathbf{F}_q$ together with an irreducible polynomial $f(x)$ of degree d, for any d, as the polynomial ring over $\mathbf{F}_q$ modulo the polynomial $f(x)$. The polynomial $f(x)$ defines the ►Extension Field. In the example just given, $x^2 + x + 1$ defines $\mathbf{F}_4$ over $\mathbf{F}_2$ as $\mathbf{F}_2[x]/(x^2 + x + 1)$.

ISMS: A Management Framework for Information Security

Angelika Plate
Director, AEXIS Security Consultants, Bonn, Germany

Synonyms

Information security management system; ISO/IEC 27001

Definition

An ISMS (information security management system) is that part of the overall management system, based on a business risk approach, to establish, implement, operate, monitor, review, maintain, and improve information security.

Background

The first publication of a standard for an ISMS appeared in 1998 as a British Standard (BS) 7799 Part 2. BS 7799 Part 2 was based on the idea to provide a management system for the application of the information security controls contained in BS 7799 Part 1. After UK-internal revisions, both standards were considered in ISO due to the large interest these standards generated all over the world.

After further improvements of these standards in the ISO revision process, in the year 2005, the revised version of BS 7799 Part 2 was published as ISO/IEC 27001, and the revised BS 7799 Part 1 as ISO/IEC 27002. Since then both standards are frequently applied by plenty of organizations all round the world.

Theory

The ISMS standard ISO/IEC 27001 provides a foundation for designing and deploying a management system for information security to prevent a variety of business-threatening risks such as the following:

- Financial losses and damages
- Loss of the organization's intellectual capital and intellectual property rights
- Loss of market share
- Poor productivity and performance ratings
- Ineffective operations
- Inability to comply with laws and regulations
- Loss of image and reputation

ISO/IEC 27001 specifies the requirements and processes for enabling a business to establish, implement, review, monitor, manage, and maintain effective information security. Like ISO 9001, it is built on the Plan-Do-Check-Act process cycle model, as well as on the requirement for continual improvement.

An ISMS relates to the broader roles and responsibilities of an organization such as corporate social responsibility, governance, and legal and regulatory obligations. All these aspects need to be addressed due to the increasing dependence of businesses on information systems and information and communication technologies.

The ISMS is a risk-based specification designed to take care of the information security aspects of corporate governance, protection of tangible and non-tangible assets information and legal and contractual obligations, as well as the wide range of threats to the organization's ICT systems and business processes.

Applying the ISMS risk-management philosophy as part of the business's overall risk approach provides an organization with the means to implement effective information security management in compliance with the organization's objectives and business requirements.

Several other standards have been developed to support the implementation of an ISMS:

- ISO/IEC 27002 contains a holistic set of information security controls, which can be used to reduce the risks identified in the risk assessment to an acceptable level. The controls address the different topics that need to be considered to achieve consistent information security, such as security aspects related to personnel, physical security, security in operations and communications, access control, incident handling, and business continuity.
- ISO/IEC 27005 provides explanation about how to carry out a risk assessment and how to successfully implement the resulting controls to achieve overall sound risk management. This International Standard will only provide guidelines for an organization; it will not specify a particular methodology for information security risk management. It is the organization's responsibility to define an approach to risk management that is most suitable to their business.
- ISO/IEC 27006 specifies requirements for bodies providing certification of information security management systems (see also Applications below). It is intended to ensure their creditability as well as their competence to perform audits.

Other standards supporting the ISMS processes are also currently under development, for example, sector-specific standards that address the implementation of an ISMS for

different business sectors. An overview of all these standards can be found in ISO/IEC 27000; this standard is publicly available and can be downloaded at http://standards.iso.org/ittf/PubliclyAvailableStandards/index.htm.

Applications

When implementing an ISMS, the organization first needs to have a clear understanding of why information security is important and what should be achieved with the ISMS. This means understanding how information security relates to its specific business objectives, taking into account the expectations of its customers, the financial objectives of the organization, and any relevant regulatory or legal requirements.

The organization's senior management needs to be actively involved in the decision-making processes concerning objectives, priorities and implementation timeframes. Senior management needs to actively support the leadership of ISMS activities and needs to provide necessary resources and to ensure that all ISMS personnel are adequately trained.

As the ISMS processes and the controls selected to achieve information security are all based on a risk assessment, the selection of a suitable risk-assessment approach and tools are critical to the ongoing effectiveness of an ISMS. The approach taken must be consistent with the culture of the organization concerning the management of other types of risk.

A successful ISMS implementation also requires follow through from planning to operation. The selected controls need to be implemented based on the priorities that the risk assessment has identified and it is necessary that staff is trained in the ISMS processes and controls.

It is inevitable that security incidents will occur and that, from time to time, management reviews or audits will detect nonconformities with ISMS standards, policies, and procedures. When such circumstances arise, don't just take a tactical approach to solve the problem on an ad hoc basis. Instead, use the ISMS. If procedures and processes are found wanting, then improve them. For example, if they do not support rapid response to a crisis, update them so that they will in future.

Another important aspect of the ISMS standard is certification, which can be achieved by organizations applying the ISMS standard. The certification model used for ISO/IEC 27001 is exactly the same as for a QMS (quality management system in accordance with ISO 9001) and all the other management system standards, i.e., accredited certification with initial, surveillance, and recertification audits. Therefore, it is also easy to combine several management systems, such as QMS and ISMS and also have combined audits.

Certification can help organizations

- To have an independent review of their information security arrangements
- To prove to business partners that they are working security
- To address requirements in RFPs or from current or future customers

It is important to understand that the ISMS standard ISO/IEC 27001 is the only standard that is relevant for certification, all other supporting standards, such as ISO/IEC 27002 and ISO/IEC 27005, are just guidelines that are intended to help implementing the ISMS but are in no form mandatory for certification.

There are currently more than 5,800 organizations that have been certified, and overview of these can be found under www.iso27001certificates.com. ISO has also published several ISO Focus papers and other magazines that highlight user experiences and further ideas in the area of ISMS (www.iso.org/iso/magazines.htm).

Recommended Reading

1. ISO/IEC 27001:2005, Information technology – Security techniques – Information security management systems – Requirements
2. ISO/IEC 27002:2005, Information technology – Security techniques – Information security management – Code of practice
3. ISO/IEC 27005:2005, Information technology – Security techniques – Information security risk management
4. ISO/IEC 27000:2008, Information technology – Security techniques – Information security management systems – Overview and vocabulary

ISO 15408 CC – Common Criteria

►Common Criteria

ISO 19790 2006 Security Requirements for Cryptographic Modules

►FIPS 140-2

ISO/IEC 15408

►Common Criteria, From a Security Policies Perspective

ISO/IEC 27001

►ISMS: A Management Framework for Information Security

ISO-9796 Signature Standards

Mehdi Tibouchi
Laboratoire d'informatique de l'ENS, École normale supérieure, Paris, France

Related Concepts

►RSA Cryptosystem; ►Signature Encoding Signature Scheme

Definition

The ISO 9796 Signature Standards are a series of standards published by ISO for digital signatures using the RSA cryptosystem. Each of these standards specifies an encoding function μ such that the signature of a given message m is computed as:

$$\sigma = \mu(m)^{1/e} \bmod N \qquad (*)$$

Moreover, these standards support message recovery, in the sense that the message m, or at least a part of it, is embedded in $\mu(m)$ with some redundancy and can, thus, be recovered as part of signature verification.

Background

The first realization of digital signatures was proposed by Rivest et al. [11] as part of their seminal 1977 paper: a signer with RSA public key (N, e) signs a message m as $\sigma = m^{1/e} \bmod N$.

This method is not really satisfactory, however, since an adversary seeing (m, σ) can produce valid signatures on many other messages: for example, $(c^e \cdot m, c \cdot \sigma)$ is a valid message-signature pair for any $c \bmod N$. This vulnerability was exploited to forge certificates as early as the mid-1980s [10]. In particular, this "textbook RSA" signature scheme does not satisfy what is now recognized as the proper security notion for signatures, namely, *existential unforgeability under chosen message attacks*, as introduced by Goldwasser et al. [5].

Some signature schemes provably secure in this strong sense were constructed in the 1980s, but they were extremely inefficient. To achieve this level of security in a practical manner, it was proposed to use "padded" RSA signatures of the form $(*)$ with encoding functions μ carefully chosen to thwart the aforementioned homomorphic attacks, even though a complete proof of security was out of reach.

Theory

ISO/IEC 9796-1 [7], initially published in 1991, was the first international standard for digital signatures. In this case, the encoded message $\mu(m)$ contains the bytes of m interspersed with redundancy from an error-correcting code. The encoding function μ is constructed to avoid homomorphic attacks [6]; for example, it ensures that $c \cdot \mu(m)$ is never a valid encoding for $c \neq 1 \bmod N$. The standard could, thus, withstand cryptanalytic efforts for most of the 1990s [10]. However, successful attacks were published in 1999 [2], which caused it to be withdrawn.

With the widespread availability of efficient cryptographic hash functions, a second standard, ISO/IEC 9796-2 [8], was published in 1997. For long messages, it features partial message recovery with an encoding function taking the following form:

$$\mu(m) = 6\mathrm{A}_{16} \| m[1] \| \textsc{hash}(m) \| \mathrm{BC}_{16}$$

where HASH is a cryptographic hash function of fixed bit length k_h, and $m[1]$ is a prefix of m long enough to ensure that $\mu(m)$ is of full size. The recommended digest size k_h was initially between 128 and 160 bits. This was shown to be insecure in 1999 [3]. The updated, currently valid standard, ISO/IEC 9796-2:2002 [9], mandates that $k_h \geq 160$, which was eventually shown to be insecure as well in 2009 [4].

This current version of ISO/IEC 9796 also standardizes a more robust encoding function, PSS, which contrary to all the previous encodings, is proved to be secure (unforgeable under chosen message attacks) in the random oracle model [1].

Applications

Despite its vulnerabilities, the ISO/IEC 9796-2:2002 ad hoc encoding remains in widespread use. It is the signature scheme used in such applications as the EMV specification for credit card payment authentication.

Recommended Reading

1. Bellare M, Rogaway P (1996) The exact security of digital signatures – how to sign with rsa and rabin. In EUROCRYPT, pp 399–416
2. Coppersmith D, Coron J-S, Grieu F, Halevi S, Jutla CS, Naccache D, Stern JP (2008) Cryptanalysis of iso/iec 9796-1. J Cryptol 21(1):27–51
3. Coron J-S, Naccache D, Stern JP (1999) On the security of rsa padding. In: Wiener MJ (ed) CRYPTO. Lecture notes in Computer Science, vol 1666. Springer, pp 1–18

I

4. Coron J-S, Naccache D, Tibouchi M, Weinmann R-P (2009) Practical cryptanalysis of iso/iec 9796-2 and emv signatures. In: Halevi S (ed) CRYPTO. Lecture notes in Computer Science, vol 5677. Springer, pp 428–444
5. Goldwasser S, Micali S, Rivest RL (1988) A digital signature scheme secure against adaptive chosen-message attacks. SIAM J Comput 17(2):281–308
6. Guillou L-C, Quisquater J-J, Walker M, Landrock P, Shafer C (1990) Precautions taken against various potential attacks in ISO/IEC DIS 9796. In EUROCRYPT, pp 465–473
7. ISO/IEC 9796-1 (1999) Information technology – Security techniques – Digital signature schemes giving message recovery – Part 1: Mechanisms using redundancy. ISO, Geneva, Switzerland
8. ISO/IEC 9796-2 (1997) Information technology – Security techniques – Digital signature schemes giving message recovery – Part 2: Mechanisms using a hash function. ISO, Geneva, Switzerland
9. ISO/IEC 9796-2:2002 (2002) Information technology – Security techniques – Digital signature schemes giving message recovery – Part 2: Integer factorization based mechanisms. ISO, Geneva, Switzerland
10. Misarsky J-F (1998) How (not) to design rsa signature schemes. In: Imai H, Zheng Y (eds) Public key cryptography. Lecture notes in computer science, vol 1431. Springer, pp 14–28
11. Rivest RL, Shamir A, Adleman LM (1978) A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21(2):120–126

Issuer

Marijke De Soete
Security4Biz, Oostkamp, Belgium

Definition

In card retail payment schemes and electronic commerce, there are normally two parties involved in a payment transaction: a customer and a merchant. The issuer is the bank of the customer. In other words, the financial institution that makes payment cards available to customers (cardholders), authorizes transactions at POS terminals or ATMs and guarantees payment to the acquirer for transactions that are in conformity with the rules of the relevant payment scheme.

In more general terms, an issuer is used in the context of service providers. It indicates the party that is responsible for the application related to the service which distributes it, possibly on a token, to the customer. Examples are mobile network operators issuing SIMs/UICCs, governments issuing e-ID cards and passports, etc.

Recommended Reading

1. www.ecb.europa.eu
2. www.emvco.com
3. www.europeanpaymentscouncil.eu
4. www.gsmworld.com
5. www.mobeyforum.org

Itoh–Tsujii Inversion Algorithm

Jorge Guajardo*
Bosch Research and Technology Center North America, Pittsburgh, USA

Definition

Originally introduced in [5], the Itoh and Tsujii algorithm (ITA) is an exponentiation-based algorithm for ▶inversion in finite fields which reduces the complexity of computing the inverse of a nonzero element in $\mathbb{F}_{2^n}$, when using a normal basis representation, from $n-2$ multiplications in $\mathbb{F}_{2^n}$ and $n-1$ cyclic shifts using the ▶binary exponentiation method to at most $2\lfloor \log_2(n-1) \rfloor$ multiplications in $\mathbb{F}_{2^n}$ and $n-1$ cyclic shifts. As shown in [4], the method is also applicable to finite fields with a polynomial basis representation.

Related Concepts

It is a well-known fact that there are several possibilities to represent elements of a ▶finite field. In particular, given an irreducible polynomial $P(x)$ of degree m over $\mathbb{F}_q$ and a root α of $P(x)$ (i.e., $P(\alpha) = 0$), one can represent an element $A \in \mathbb{F}_{q^m}$, $q = p^n$ and p prime, as a polynomial in α, i.e., as $A = a_{m-1}\alpha^{m-1} + a_{m-2}\alpha^{m-2} + \cdots + a_1\alpha + a_0$ with $a_i \in \mathbb{F}_q$. The set $\{1, \alpha, \alpha^2, \ldots, \alpha^{m-1}\}$ is then said to be a polynomial basis (or standard basis) for the finite field $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$ (▶extension field). Another type of basis is called a normal basis. Normal bases are of the form $\{\beta, \beta^q, \beta^{q^2}, \ldots, \beta^{q^{m-1}}\}$ for an appropriate element $\beta \in \mathbb{F}_{q^m}$. Then, an element $B \in \mathbb{F}_{q^m}$ can be represented as $B = b_{m-1}\beta^{q^{m-1}} + b_{m-2}\beta^{q^{m-2}} + \cdots + b_1\beta^q + b_0\beta$, where $b_i \in \mathbb{F}_q$. It can be shown that for any field $\mathbb{F}_q$ and any extension field $\mathbb{F}_{q^m}$, there exists always a normal basis of $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$(see [6, Theorem 2.35]). Notice that $(\beta^{q^i})^{q^k} = \beta^{q^{i+k}} = \beta^{q^{i+k \bmod m}}$ which follows from the fact that $\beta^{q^m} \equiv \beta$. Thus, raising an element $B \in \mathbb{F}_{q^m}$ to the qth power can be easily accomplished through a cyclic shift of its coordinates, i.e., $B^q = \left(b_{m-1}\beta^{q^{m-1}} + b_{m-2}\beta^{q^{m-2}} + \cdots + b_1\beta^q + b_0\beta\right)^q = b_{m-2}\beta^{q^{m-1}} + b_{m-3}\beta^{q^{m-2}} + \cdots + b_0\beta^q + b_{m-1}\beta$ in a normal basis representation. This follows by applying the binomial theorem repeatedly and the fact that if $\mathbb{F}_q$ is a field of prime

*Work done while the author was at Philips Research, The Netherlands

characteristic p, $(A+B)^p \equiv A^p + B^p$ for all $A, B \in \mathbb{F}_q$, where $q = p^n$.

Before describing the Itoh and Tsujii algorithm, it is beneficial to understand how the inverse would be computed via an exponentiation algorithm. In particular, it can be shown that for all $A \in \mathbb{F}_{2^n}$, $A \neq 0$, one can compute A^{-1} as

$$A^{-1} \equiv A^{2^n-2} = A^2 \cdot A^{2^2} \cdots A^{2^{n-1}}$$

which follows from a generalization of Fermat's little theorem. This requires $n-2$ multiplications and $n-1$ squarings using the binary method for exponentiation. Notice that if A is an element of a field of characteristic two (►finite field), squaring (A^2) for any $A \in \mathbb{F}_{2^n}$ can be computed with one cyclic shift.

Theory

Itoh and Tsujii proposed in [5] three algorithms. The first two algorithms describe addition chains for exponentiation-based inversion in fields $\mathbb{F}_{2^n}$, while the third one describes a method based on subfield inversion. The first algorithm is only applicable to values of n such that $n = 2^r + 1$, for some positive r, and it is based on the observation that the exponent $2^n - 2$ can be rewritten as $(2^{n-1} - 1) \cdot 2$. Thus if $n = 2^r + 1$, it follows that $A^{-1} \equiv \left(A^{2^{2^r}-1}\right)^2$. Furthermore, rewriting $2^{2^r} - 1$ as

$$2^{2^r} - 1 = \left(2^{2^{r-1}} - 1\right) 2^{2^{r-1}} + \left(2^{2^{r-1}} - 1\right) \tag{1}$$

one obtains Algorithm 1. Notice that Algorithm 1 performs $r = \log_2(n-1)$ iterations. In every iteration, one multiplication and i cyclic shifts, for $0 \leq i < r$, are performed which leads to an overall complexity of $\log_2(n-1)$ multiplications and $n-1$ cyclic shifts.

Algorithm 1 Multiplicative inverse computation in $\mathbb{F}_{2^n}$ with $n = 2^r + 1$ [5, Theorem 1]

Input: $A \in \mathbb{F}_{2^n}$, $A \neq 0$, $n = 2^r + 1$
Output: $C = A^{-1}$
  $C \leftarrow A$
  for $i = 0$ to $r-1$ **do**
    $D \leftarrow C^{2^{2^i}}$ {NOTE: 2^i cyclic shifts}
    $C \leftarrow C \cdot D$
  $C \leftarrow C^2$
  Return (C)

Example 1 Let $A \in \mathbb{F}_{2^{17}}$, $A \neq 0$. Then according to Algorithm Algorithm 1, the inverse A^{-1} can be computed with the following addition chain:

$$\begin{aligned} A^2 \cdot A &= A^3 \\ \left(A^3\right)^{2^{2^1}} \cdot A^3 &= A^{15} \\ \left(A^{15}\right)^{2^{2^2}} \cdot A^{15} &= A^{255} \\ \left(A^{255}\right)^{2^{2^3}} \cdot A^{255} &= A^{65535} \\ \left(A^{65535}\right)^2 &= A^{131070} \end{aligned}$$

A quick calculation verifies that $2^{17} - 2 = 131070$. Notice that in accordance with Algorithm 1, four multiplications in $\mathbb{F}_{2^{17}}$ have been performed and, if using a normal basis representation, $2^4 = 16$ cyclic shifts would also be required to complete the computation.

Algorithm 1 can be generalized to any value of n [5]. To see this, write $n-1$ as

$$n - 1 = \sum_{i=1}^{t} 2^{k_i}, \quad \text{where } k_1 > k_2 > \cdots > k_t \tag{2}$$

Then, using the fact that $A^{-1} \equiv \left(A^{2^{n-1}-1}\right)^2$ and (2), it can be shown that the inverse of A can be written as:

$$\left(A^{2^{n-1}-1}\right)^2 = \left[\left(A^{2^{2^{k_t}}-1}\right)\left(\left(A^{2^{2^{k_{t-1}}}-1}\right)\cdots\right.\right.$$
$$\left.\left.\left[\left(A^{2^{2^{k_2}}-1}\right)\left(A^{2^{2^{k_1}}-1}\right)^{2^{2^{k_2}}}\right]^{2^{2^{k_3}}} \cdots\right)^{2^{2^{k_t}}}\right]^2 \tag{3}$$

An important feature of (3) is that in computing $A^{2^{2^{k_1}}-1}$ all other quantities of the form $A^{2^{2^{k_i}}-1}$ for $k_i < k_1$ have been computed. Thus, the overall complexity of (3) is:

$$\begin{aligned} \#\text{MUL} &= \lfloor \log_2(n-1) \rfloor + HW(n-1) - 1 \\ \#\text{CSH} &= n - 1 \end{aligned} \tag{4}$$

where $HW(\cdot)$ denotes the Hamming weight of the operand, i.e., the number of ones in the binary representation of the operand (►cyclic codes), MUL refers to multiplications in $\mathbb{F}_{2^n}$, and CSH refers to cyclic shifts over $\mathbb{F}_2$ when using a normal basis.

Example 2 Let $A \in \mathbb{F}_{2^{23}}$, $A \neq 0$. Then according to (2), one can write $n-1$ as $22 = 2^4 + 2^2 + 2$, where $k_1 = 4$, $k_2 = 2$, and $k_3 = 1$ and $A^{-1} \equiv A^{2^{23}-2}$ can be computed with the

following addition chain:

$$
\begin{aligned}
A^{2^2-1} &= A^2 \cdot A \\
A^{2^4-1} &= \left(A^3\right)^{2^2} \cdot A^3 \\
A^{2^8-1} &= \left(A^{15}\right)^{2^4} \cdot A^{15} \\
A^{2^{16}-1} &= \left(A^{255}\right)^{2^8} \cdot A^{255} \\
A^{2^{23}-2} &= \left(A^{2^2-1} \cdot \left(A^{2^4-1} \cdot \left(A^{2^{16}-1}\right)^{2^4}\right)^{2^2}\right)^2
\end{aligned}
$$

The above addition chain requires 6 multiplications and 22 cyclic shifts which agrees with the complexity of (4).

In [5], the authors also notice that the previous ideas can be applied to extension fields $\mathbb{F}_{q^m}$, $q = 2^n$. Although this inversion method does not perform a complete inversion, it reduces inversion in $\mathbb{F}_{q^m}$ to inversion in $\mathbb{F}_q$. It is assumed that subfield inversion can be done relatively easily, e.g., through table look-up or the extended ►Euclidean algorithm. These ideas are summarized in Theorem 1. The presentation here follows [4], and it is slightly more general than [5] as a subfield of the form $\mathbb{F}_{2^n}$ is not required, rather general subfields $\mathbb{F}_q$ are used.

Theorem 1 [5, *Theorem* 3] *Let* $A \in \mathbb{F}_{q^m}$, $A \neq 0$, *and* $r = (q^m - 1)/(q - 1)$. *Then, the multiplicative inverse of an element A can be computed as*

$$A^{-1} = (A^r)^{-1} A^{r-1}. \tag{5}$$

Computing the inverse through Theorem 1 requires four steps:

Step 1 Exponentiation in $\mathbb{F}_{q^m}$, yielding A^{r-1}.
Step 2 Multiplication of A and A^{r-1}, yielding $A^r \in \mathbb{F}_q$.
Step 3 Inversion in $\mathbb{F}_q$, yielding $(A^r)^{-1}$.
Step 4 Multiplication of $(A^r)^{-1} A^{r-1}$.

Steps 2 and 4 are computational negligible since both A^r, in Step 2, and $(A^r)^{-1}$, in Step 4, are elements of $\mathbb{F}_q$ [6]. Both operations can, in most cases, be performed with a complexity that is well below that of one single extension field multiplication. The complexity of Step 3, subfield inversion, depends heavily on the subfield $\mathbb{F}_q$. What remains is Step 1, exponentiation to the $(r-1)$th power in the extension field $\mathbb{F}_{q^m}$. Observe that the exponent can be expressed in q-adic representation as

$$r - 1 = q^{m-1} + \cdots + q^2 + q = (1 \cdots 110)_q \tag{6}$$

Thus, this exponentiation can be computed through repeated raising of intermediate results to the qth power and multiplications. The number of multiplications in $\mathbb{F}_{q^m}$ can be minimized by using the addition chain in (3). Thus, computing A^{r-1} requires [5]:

$$
\begin{aligned}
\#\text{MUL} &= \lfloor \log_2(m-1) \rfloor + HW(m-1) - 1 \\
\#q\text{-EXP} &= m - 1
\end{aligned} \tag{7}
$$

where q-EXP refers to the number of exponentiations to the qth power in $\mathbb{F}_q$.

Example 3 Let $A \in \mathbb{F}_{q^{19}}$, $A \neq 0$, $q = p^n$ for some prime p. Then, using the q-adic representation of $r - 1$ from (6) and the addition chain from (3), one can find an addition chain to compute $A^{r-1} = A^{q^{18}+q^{17}+\cdots+q^2+q}$ as follows. First, one writes $m-1 = 18 = 2^4+2$ where $k_1 = 4$, and $k_2 = 1$. Then, $A^{r-1} = \left(A^{q^{16}+q^{15}+\cdots+q^2+q}\right)^{q^2} \cdot \left(A^{q^2+q}\right)$ and one can compute $A^{q^{16}+q^{15}+\cdots+q^2+q}$ as

$$
\begin{aligned}
A^{q^2} &= \left(A^q\right)^q \\
A^{q^2+q} &= A^q \cdot A^{q^2} \\
A^{\sum_{i=1}^{4} q^i} &= \left(A^{q^2+q}\right)^{q^2} \cdot A^{q^2+q} \\
A^{\sum_{i=1}^{8} q^i} &= \left(A^{\sum_{i=1}^{4} q^i}\right)^{q^4} \cdot A^{\sum_{i=1}^{4} q^i} \\
A^{\sum_{i=1}^{16} q^i} &= \left(A^{\sum_{i=1}^{8} q^i}\right)^{q^8} \cdot \left(A^{\sum_{i=1}^{8} q^i}\right)
\end{aligned}
$$

Notice that in computing $A^{q^{16}+q^{15}+\cdots+q^2+q}$, one has also computed A^{q^2+q}. The complexity to compute A^{r-1} (and, thus, the complexity to compute A^{-1} if the complexity of multiplication and inversion in $\mathbb{F}_q$ can be neglected) in $\mathbb{F}_{q^{19}}$ is found to be 5 multiplications in $\mathbb{F}_{q^{19}}$ and 18 exponentiations to the qth power in agreement with (7).

In their work, Itoh and Tsujii [5] assume a normal basis representation for the field elements of $\mathbb{F}_{q^m}$, $q = 2^n$. In this case, the exponentiations to the qth power are simply cyclic shifts of the m coefficients that represent an individual field element. In polynomial (or standard) basis, however, these exponentiations are, in general, considerably more expensive. Guajardo and Paar [4] take advantage of finite field properties and of the algorithm characteristics to improve on the overall complexity of the ITA in polynomial basis. In particular, the authors make use of two facts: (1) the algorithm performs alternating multiplications and several exponentiations to the qth power in a sequential manner and (2) raising an element $A \in \mathbb{F}_q$, $q = p^n$, to the q^eth power is a linear operation in $\mathbb{F}_{q^m}$, since q is a power of the field characteristic.

In general, computing A^{q^e} has a complexity of $m(m-1)$ multiplications and $m(m-2)+1 = (m-1)^2$ additions

in $\mathbb{F}_q$ [4]. This complexity is roughly the same as one $\mathbb{F}_{q^m}$ multiplication, which requires m^2 subfield multiplications if one does not assume fast convolution techniques (e.g., the ►Karatsuba algorithm for multiplication). However, in polynomial basis representation computing A^{q^e}, where $e > 1$, can be shown to be as costly as a single exponentiation to the qth power. Thus, [4] performs as many subsequent exponentiations to the qth power in one step between multiplications as possible, yielding the same multiplication complexity as in (7), but a reduced number of q^e-exponentiations. This is summarized in Theorem 2.

Theorem 2 [4 *Theorem* 2] *Let* $A \in \mathbb{F}_{q^m}$. *One can compute* A^{r-1}, *where* $r - 1 = q + q^2 + \cdots + q^{(m-1)}$ *with no more than*

$$\#MUL = \lfloor \log_2(m-1) \rfloor + HW(m-1) - 1$$
$$\#q^e\text{-}EXP = \lfloor \log_2(m-1) \rfloor + HW(m-1)$$

operations, where #MUL and $\#q^e$*-EXP refer to multiplications and exponentiations to the* q^e*th power in* $\mathbb{F}_{q^m}$, *respectively.*

It should be stressed that Theorem 2 is just an upper bound on the complexity of this exponentiation. Thus, it is possible to find addition chains which yield better complexity as shown in [2]. Finally, observe that from Theorem 2, it follows that Step 1 of the ITA algorithm requires about as many exponentiations to the q^eth power as multiplications in $\mathbb{F}_{q^m}$ if a polynomial basis representation is being used.

Optimizations and Applications

In the previous section, it has been shown that raising an element $A \in \mathbb{F}_{q^m}$ to the q^eth power can be computationally roughly as costly as performing one multiplication in $\mathbb{F}_{q^m}$. Hence, if it is possible to make exponentiations to the q^eth power more efficient, considerable speedups of the algorithm can be expected. Three classes of finite fields are introduced in [4] for which the complexity of these exponentiations is in fact substantially lower than that of a general multiplication in $\mathbb{F}_{q^m}$. These are:

- Fields $\mathbb{F}_{2^{nm}}$ with binary field polynomials.
- Fields $\mathbb{F}_{q^m}$, $q = p^n$ and p an odd prime, with binomials as field polynomials.
- Fields $\mathbb{F}_{q^m}$, $q = p^n$ and p an odd prime, with binary equally spaced field polynomials (ESP), where a binary ESP is a polynomial of the form $x^{sm} + x^{s(m-1)} + x^{s(m-2)} + \cdots + x^{2s} + x^s + 1$.

The ITA has found applications in cryptography. Early applications of the algorithm to the cryptographic setting are described in [3], where fields of binary characteristic are used. However, even characteristic composite fields have been found to be weak and are generally not longer recommended for cryptographic applications. On the other hand, the ITA remains an attractive choice for the computation of the inverse in odd characteristic fields and, in particular, in ►optimal extension fields [1]. Recently, Rodríguez-Henríquez et al. [7] have shown that the Itoh and Tsujii inversion algorithm is amenable to parallelization as well, by expressing the addition chain in terms of square root operations instead of squarings. For irreducible trinomials $P(x) = x^m + x^k + 1$, with m and k odd numbers, the square root operation is particularly efficient (simpler than a squaring operation) and therefore the overall inversion computation benefits.

Recommended Reading

1. Bailey DV, Paar C (2001) Efficient arithmetic in finite field extensions with application in elliptic curve cryptography. J Cryptol 14(3):153–176
2. Chung JW, Sim SG, Lee PJ (2000) Fast implementation of elliptic curve defined over $GF(p^m)$ on CalmRISC with MAC2424 coprocessor. In: Koç ÇK, Paar C (eds) Workshop on cryptographic hardware and embedded systems – CHES 2000. Lecture notes in computer science, vol 1965. Springer, Berlin, 17–18 Aug 2000, pp 57–70
3. Guajardo J, Paar C (1997) Efficient algorithms for elliptic curve cryptosystems. In: Kaliski Jr B (ed) Advances in Cryptology – CRYPTO '97. Lecture notes in computer science, vol 1294. Springer, Berlin, Aug 1997, pp 342–356
4. Guajardo J, Paar C (Feb 2002) Itoh-Tsujii inversion in standard basis and its application in cryptography and codes. Des Codes Cryptogr 25(2):207–216
5. Itoh T, Tsujii S (1988) A fast algorithm for computing multiplicative inverses in $GF(2^m)$ using normal bases. Inf Comput 78:171–177
6. Lidl R, Niederreiter H (1997) Finite fields. Encyclopedia of mathematics and its applications, 2nd edn. vol 20. Cambridge University Press, Cambridge
7. Rodríguez-Henríquez F, Morales-Luna G, Saqib NA, Cortés NC (2007) Parallel Itoh-Tsujii multiplicative inversion algorithm for a special class of trinomials. Des Codes Cryptogr 45(1):19–37