U

UMTS IC Card

►SIM/UICC

Uncoordinated Direct Sequence Spread Spectrum

Srdjan Capkun
System Security Group, Department of Computer Science, ETH Zurich, Zürich, Switzerland

Definition

Uncoordinated direct sequence spread spectrum (UDSSS) is a spread spectrum communication scheme where the sender and the receiver communicate using secret spreading codes that they choose randomly and independently from a public set of channels. The receiver is unaware of the codes used by the sender to transmit the messages prior to their communication. It was introduced by Popper, Strasser and Capkun.

Background

Spread spectrum (SS) techniques represent a common way to achieve anti-jamming communication. Spread spectrum techniques use data-independent, random sequences to spread a narrowband information signal over a wide (radio) band of frequencies. Under the premise that it is hard or infeasible for an attacker to jam the entire frequency band, the receiver can correlate the received signal with a replicate of the random sequence to retrieve the original information signal. Anti-jamming communication is used in commercial and military applications, both between paired devices and from one sender to multiple receiving devices (in multicast or broadcast settings). Important instances of spread spectrum techniques are frequency hopping spread spectrum (FHSS) and direct-sequence spread spectrum (DSSS).

Theory

Essential for both FHSS and DSSS communication is that the sender and receiver share a secret prior to their communication. This enables the receiver to generate the random sequence and to detect and decode the sender's spread signal. Opposed to (coordinated) DSSS, with uncoordinated direct sequence spread spectrum (UDSSS), the sender and the receivers do not agree on a secret sequence beforehand, but choose the sequence that they use for spreading and despreading from a predefined set of sequences.

UDSSS follows the principle of DSSS in terms of spreading the data using spreading sequences. However, in contrast to anti-jamming DSSS where the spreading sequence is secret and shared exclusively by the communication partners, in UDSSS, a public set of spreading sequences is used by the sender and the receivers. To transmit a message, the sender repeatedly selects a fresh, randomly selected spreading sequence from the set and spreads the message with this sequence. The code sequences are used to spread the entire message of a given length (each code thus contains enough chips to spread the message). Hence, UDSSS does neither require message fragmentation at the sender nor message reassembly at the receivers. The receiver records the signal on the channel and despreads the message by applying sequences from the public sequence set, using a trial-and-error approach. More precisely, each receiver samples the radio channel and stores the samples in a buffer. Note that the receiver is not synchronized to the beginning of the sender's message and thus record for (at least) twice the message transmission time, the factor of two is optimal. After the sampling, the receiver tries to decode the data in the buffer by using code sequences from the set and by applying a sliding-window protocol in which the current window is shifted to account for the lack of synchronization.

Jamming resilience of UDSSS is based on the fact that the attacker does not know which spreading sequences will be used by the sender prior to communication, and that during the message transmission it will not be able to find this sequence.

Applications

Applications of UDSSS include jamming-resistant key establishment, jamming-resistant broadcast (e.g., of navigation signals or emergency alerts).

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

Recommended Reading

1. Pöpper C, Strasser M, Capkun S (2009) Jamming-resistant broadcast communication without shared keys. In: Proceedings of the 18th USENIX security symposium. USENIX Association, Berkeley, pp 231–248. http://www.usenix.org/events/sec09/tech/full_papers/popper.pdf

Uncoordinated Frequency Hopping Spread Spectrum

Srdjan Capkun
System Security Group, Department of Computer Science, ETH Zurich, Zürich, Switzerland

Related Concepts

►Direct Sequence Spread Spectrum; ►Uncoordinated Direct Sequence Spread Spectrum

Definition

Uncoordinated Frequency Hopping Spread Spectrum (UFHSS) is a spread spectrum communication technique where the sender and the receiver hop between communication channels that they choose randomly and independently from a public set of channels. Neither the receiver nor the attacker knows the channel sequence used by the sender prior to communication. It was introduced by Strasser, Popper, Capkun and Cagalj.

Background

Spread Spectrum (SS) techniques represent a common way to achieve jamming-resistant communication. Spread spectrum techniques use data-independent, random sequences to spread a narrowband information signal over a wide (radio) band of frequencies. Under the premise that it is hard or infeasible for an attacker to jam the entire frequency band, the receiver can correlate the received signal with a replicate of the random sequence to retrieve the original information signal. Anti-jamming communication is used in commercial and military applications, both between paired devices and from one sender to multiple receiving devices (in multicast or broadcast settings). Important instances of spread spectrum techniques are Frequency Hopping Spread Spectrum (FHSS) and Direct-Sequence Spread Spectrum (DSSS).

Theory

Essential for both FHSS and DSSS communication is that the sender and receiver share a secret prior to their communication. This enables the receiver to generate the random sequence and to detect and decode the sender's spread signal. Opposed to (coordinated) FHSS, with Uncoordinated Frequency Hopping Spread Spectrum, the sender and the receivers do not agree on a secret channel sequence beforehand, but choose the channels on which they send and listen randomly from a predefined set of frequency channels. Communication is possible because, at recurring points in time, the sender and the receivers will be sending and listening on the same frequency channel.

Given that the sender and receiver hopping sequences are unknown to the attacker, UFHSS achieves jamming resilience that is equivalent to the one of FHSS. In order for (coordinated or uncoordinated) frequency hopping to resist reactive jamming attacks, the sender can dwell on the same frequency channel only for a short period of time. Each message is thus split into a set of fragments with a typical size of a few hundred bits only. After the fragmentation, the sender encapsulates each fragment into a packet, encodes the packets with error correcting codes, and repetitively transmits the encoded packets one after another on randomly chosen frequency channels. Note that although splitting the message into fragments is a straight-forward operation, the reassembly of the received fragments at the receiver is nontrivial if an attacker inserts additional fragments or modifies transmitted ones (that may be hard to distinguish from legitimate fragments); the attacker may easily achieve this for UFHSS because the receivers do not know the sender's channel selection. UFHSS thus needs to provide means for the receiver to efficiently assemble legitimate packets into the sender's message even if the attacker inserts the messages. One technique to achieve this is by linking all fragments of the same message to form a hash chain where each fragment is linked to its successor with a hash. Other techniques include using cryptographic accumulators and short signatures.

Applications

Applications of UFHSS include jamming-resistant key establishment, jamming-resistant broadcast (e.g., of navigation signals or emergency alerts).

Recommended Reading

1. Strasser M, Pöpper C, Capkun S, Cagalj M (2008) Jamming-resistant key establishment using uncoordinated frequency hopping. In: Proceedings of the 2008 IEEE symposium on security and privacy, Oakland, 18–21 May 2008. SPIEEE Computer Society, Washington, DC, pp 64–78. DOI:http://dx.doi.org/10.1109/SP.2008.9

Undeniable Signatures

Gerrit Bleumer
Research and Development, Francotyp Group,
Birkenwerder bei Berlin, Germany

Related Concepts

►Digital Signature; ►Forgery

Definition

Undeniable signatures are digital signatures that can be verified only by some help from the signer. Unlike an ordinary digital signature that can be verified by anyone who has accessed the public verifying key of the signer (*universal verifiability*), an undeniable signature can only be verified by engaging in a – usually interactive – protocol with the signer. The outcome of the protocol is an affirming or {rejecting} response telling the verifier whether the undeniable signature has originated from the alleged signer or not. The verifier cannot enforce a clarification about a signature's validity because a signer can always refuse to cooperate, but nonrepudiation is still guaranteed since a signer cannot convince a verifier that a correct signature is invalid or that an incorrect signature is valid.

Background

Undeniable signatures were introduced by Chaum and van Antwerpen [2].

Theory

An undeniable signature scheme has three operations: one for generating pairs of a private signing key and a public verifying key (public key cryptography), one for signing messages, and a *confirming operation* for proving signatures valid (confirmation) or invalid (disavowal). The confirming operation must have two defined outputs to signal confirmation or disavowal in order to distinguish three possible cases: (a) the signature in question is valid (operation returns "confirm"), (b) the signature in question is invalid (operation returns "disavow"), and (c) the alleged signer is not willing or not available to cooperate and lets the verifier find out whether (a) or (b) holds (operation fails). This latter problem is addressed by *designated confirmer signatures*.

The characteristic security requirements of an undeniable signature scheme are:

- *Unforgeability*: Resistance against existential forgery under adaptive chosen message attacks by a computationally restricted attacker.
- *Invisibility*: A cheating verifier, given a signer's public verifying key, a message, and an undeniable signature, cannot decide with non-negligible probability better than a pure guess whether the signature is valid for the message with respect to the signer's verifying key or not.
- *Soundness*: A cheating signer cannot misuse the confirming operation in order to prove a valid signature invalid (nonrepudiation), or an invalid signature valid (false claim of origin).
- *Nontransferability*: A cheating verifier obtains no information from the confirming operation that allows him to convince a third party that the alleged signature is valid or invalid, regardless of if the signature is in fact valid or not.

The property of nontransferability was intended by the original work of Chaum and van Antwerpen [2], but Jakobsson [7] showed that their particular undeniable signature construction cannot achieve nontransferability against mutually distrusting but interacting verifiers. Jakobsson et al. [8] proposed undeniable signature constructions that satisfy nontransferability as well.

Constructions of undeniable signatures have been based on groups, in which the discrete logarithm problem is hard [2, 4, 9] and on the problem of factoring integers [5].

Undeniable signature schemes can be equipped with additional features: the confirming operation can be noninteractive according to [8]. Pedersen [10] suggested distribution of the power of confirming signatures over a set of delegates in order to increase the availability of individual signers. Harn and Yang [6] proposed the concept of undeniable threshold signatures, where certain subsets (coalitions) of signers are authorized to produce signatures on behalf of a whole set of signers. Efficient and secure constructions were proposed by Michels and Stadler [9]. Chaum et al. proposed convertible undeniable fail-stop signatures [3], where signers can convert their undeniable signatures into fail-stop signatures. Sakurai and Yamane [11] have proposed undeniable blind signatures.

Convertible Undeniable Signatures

Convertible undeniable signatures [1] are an interesting extension to undeniable signatures. In a convertible undeniable signature scheme, a signer can convert each individual undeniable signature into an ordinary digital signature that is universally verifiable. Upon request by a verifier, the signer provides an individual receipt for a requested undeniable signature to the verifier. Henceforth, the verifier can unlock the respective undeniable signature and forward it together with the receipt to any third party, who

can now verify the signature against the signer's public verifying key. Moreover, the signer can provide a universal receipt that instantly allows a recipient to universally verify all signatures of the respective signer. In effect, convertible undeniable signature schemes support signers in gradually increasing the verifiability of their signatures in a controlled fashion.

Let us reconsider the software company mentioned in the introduction and imagine it is going bankrupt. It may still have contractual liabilities to support its customers in verifying their software for a number of years. However, this service may be too costly and there may even be no need any more to further control who is verifying which software packages. In this case, the company could release a universal receipt on their Web page, which would henceforth allow anyone to verify the signatures of its software packages at any time.

A convertible undeniable signature scheme has the same three operations as an undeniable signature scheme and the following three additional operations:

- *An individual conversion operation*, which takes as input a message, an undeniable signature, a signer's private signing key, and returns an ordinary, i.e., universally verifiable signature.
- *A universal conversion operation*, which takes as input a signer's private signing key and returns a universal receipt that allows to convert all undeniable signatures valid with respect to the signer's public verifying key into ordinary, i.e., universally verifiable signatures.
- *A universal verifying operation*, which takes as input a message, a converted undeniable signature, and a signer's public verifying key and returns whether the signature is valid or not with respect to the alleged signer's public verifying key.

Applications

Undeniable signatures are useful for signers of nonpublic sensitive information who seek to keep control over who can verify their signatures. For example, a company producing software for safety critical systems could deliver its executables with undeniable signatures. This would allow registered customers to verify the origin of the software, while software pirates could not do so. In case a significant bug is discovered later in the software, a registered customer could hold the software company liable for the bug and perhaps for its consequences.

Open Problems and Future Directions

The characteristic security requirements of a convertible undeniable signature scheme include those of an undeniable signature scheme and additional security requirements guaranteeing that valid (invalid) signatures can only be converted into valid (invalid) signatures, etc. These additional security requirements have not yet been formalized in the open literature.

Recommended Reading

1. Boyar J, Chaum D, Damgård I, Pedersen T (1991) Convertible undeniable signatures. In: Menezes AJ, Vanstone SA (eds) Advances in cryptology – CRYPTO'90. Lecture notes in computer science, vol 537. Springer, Berlin, pp 189–205
2. Chaum D, van Antwerpen H (1990) Undeniable signatures. In: Brassand G (ed) Advances in cryptology – CRYPTO'89. Lecture notes in computer science, vol 435. Springer, Berlin, pp 212–216
3. Chaum D, van Heijst E, Pfitzmann B (1990) Cryptographically strong undeniable signatures, unconditionally secure for the signer. In: Feigenbaum J (ed) Advances in cryptology – CRYPTO'91. Lecture notes in computer science, vol 576. Springer, Berlin, pp 470–484
4. Damgård IB, Pedersen TP (1996) New convertible undeniable signature schemes. In: Maurer U (ed) Advances in cryptology – EUROCRYPT'96. Lecture notes in computer science, vol 1070. Springer, Berlin, pp 372–386
5. Gennaro R, Krawczyk H, Rabin T (1997) RSA-based undeniable signatures. In: Kaliski BS (ed) Advances in cryptology – CRYPTO'97. Lecture notes in computer science, vol 1294. Springer, Berlin, pp 132–149
6. Harn L, Yang S (1993) Group-oriented undeniable signature schemes without the assistance of a mutually trusted party. In: Sebermy J, Zheng Y (eds) Advances in cryptology – ASIACRYPT'92. Lecture notes in computer science, vol 718. Springer, Berlin, pp 133–142
7. Jakobsson M (1995) Blackmailing using undeniable signatures. In: De Santis A (ed) Advances in cryptology – EUROCRYPT'94. Lecture notes in computer science, vol 950. Springer, Berlin, pp 425–427
8. Jakobsson M, Sako K, Russell I (1996) Designated verifier proofs and their applications. In: Maurer U (ed) Advances in cryptology – EUROCRYPT'96. Lecture notes in computer science, vol 1070. Springer, Berlin, pp 143–154
9. Michels M, Stadler M (1997) Efficient convertible undeniable signature schemes. In: International workshop on selected areas in cryptography (SAC'97). Springer, Berlin, pp 231–244
10. Pedersen PT (1991) Distributed provers with applications to undeniable signatures (Extended abstract). In: Davies DW (ed) Advances in cryptology – EUROCRYPT'91. Lecture notes in computer science, vol 547. Springer, Berlin, pp 221–242
11. Sakurai K, Yamane Y (1996) Blind decoding, blind undeniable sigatures, and their applications to privacy protection. In: Anderson R (ed) Information hiding (IHW'96). Lecture notes in computer science, vol 1174. Springer, Berlin, pp 257–264

Universal Integrated Circuit Card

►SIM/UICC

Universal One-Way Hash Functions (UOWHF)

Bart Preneel
Department of Electrical Engineering-ESAT/COSIC,
Katholieke Universiteit Leuven and IBBT,
Leuven-Heverlee, Belgium

Synonyms

Target collision resistant hash function; Everywhere second preimage resistant hash function (esec)

Related Concepts

►Hash Functions

Definition

A Universal One-Way Hash Function (UOWHF) is a class of ►hash functions indexed by a public parameter (called a ►key), for which finding a second preimage is hard. The main idea is that first the challenge input is selected, and subsequently the function instance (or parameter) is chosen. Only then should the opponent try to find a second input with the same output as the challenge.

Background

The concept of UOWHF has been introduced by Naor and Yung [6]. Bellare and Rogaway [1] propose the alternative name Target Collision Resistant (TCR) hash function. Alternative definitions and generalizations were introduced by Zheng et al. [11] and by Mironov [5]. Rogaway and Shrimpton [7] consider a broader class of properties of hash functions and call the security property of a UOWHF esec for everywhere second preimage resistance.

Theory

A UOWHF is a weaker notion than a collision resistant hash function (CRHF), hence they are easier to construct. In a CRHF , the opponent is first given the key and then he or she has to produce two colliding inputs. Finding collisions for a fixed parameter of a UOWHF may be rather easy, but this will not help the opponent to violate the security requirement, as the instance is chosen *after* the challenge. This also implies that the ►birthday paradox does not apply to a UOWHF and a hash result of 85 bits may offer adequate security (in 2011). Simon [10] has shown that there exists an oracle relative to which a UOWHF exist, but no CRHF.

UOWHFs can replace CRHFs in the construction of efficient ►digital signature schemes: In this case, the signer needs to pick first the message m and then the key K and sign the pair $(K, h_K(m))$. Note that this has the disadvantage that a cheating signer could reverse the order: First choose K, then find a collision (m, m') (this may be easy for a UOWHF) and later on claim that he or she has signed m' rather than m. The security model would make a signer responsible for both signatures, even if the scheme could also have been cryptanalyzed by an outsider with a (presumably harder) second preimage attack. It depends on the context whether or not this is a problem. However, this situation can be avoided by employing a CRHF. The ►Cramer–Shoup cryptosystem uses a UOWHF in the verification process of a ciphertext.

Naor and Yung construct a UOWHF based on a strongly universal hash function and a one-way permutation [6]. Rompel [8] describes an interesting but very inefficient construction to turn any ►one-way function in a UOWHF, which shows that one-way functions imply ►digital signature schemes (for a detailed proof, see [3]). Impagliazzo and Naor present a construction based on the subset sum problem [2] (►Knapsack Cryptographic Schemes).

Naor and Yung [6] describe a composition construction. Several papers have studied the problem of constructing an efficient UOWHF based on a fixed size UOWHF which compresses n bits to m bits (with $n > m$). Bellare and Rogaway show that the Merkle–Damgård construction, which is used for CRHFs (►Hash Functions) does *not* work [1]. They present a linear construction, that is sequential, and a tree construction, that allows for a parallel implementation. The linear construction has been optimized by Shoup [9]: His scheme requires 2^t invocations of the fixed size UOWHF and a key of t m-bit strings to hash a message of length $2^t(m - n) + m$ bits to m bits; this has been shown to be optimal [5]. The tree construction has been optimized by Lee et al. [4]; the best scheme allows for a t-fold parallelism in exchange for a slightly larger key (see [4] for a comparison of several alternatives).

Recommended Reading

1. Bellare M, Rogaway P (1997) Collision-resistant hashing: towards making UOWHFs practical. In: Kaliski B (ed) Advances in cryptology, proceedings Crypto'97. LNCS, vol 1294. Springer, Berlin, pp 470–484
2. Impagliazzo R, Naor M (1996) Efficient cryptographic schemes provably as secure as subset sum. J Cryptol 9(4):199–216
3. Katz J, Koo C-Y (2005) On constructing universal one-way hash functions from arbitrary one-way functions. Preprint, http://www.cs.umd.edu/~jkatz/papers/rompel.pdf
4. Lee W, Chang D, Lee S, Sung S, Nandi N (2003) New parallel domain extenders for UOWHFs. In: Lai CS (ed) Advances in cryptology, proceedings Asiacrypt'03. LNCS, vol 2894. Springer, Berlin, pp 208–227

5. Mironov I (2001) Hash functions: from Merkle-Damgård to Shoup. In: Pfitzmann B (ed) Advances in cryptology, proceedings Eurocrypt'01. LNCS, vol 2045. Springer, Berlin, pp 166–181
6. Naor M, Yung M (1990) Universal one-wayhash functions and their cryptographic applications. In: Proceedings 21st ACM symposium on the theory of computing, Baltimore, pp 387–394
7. Rogaway P, Shrimpton T (2004) Cryptographic hash function basics: definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In: Roy BK, Meier W (eds) Fast software encryption. LNCS, vol 3017. Springer, Heidelberg, pp 371–388
8. Rompel J (1990) One-way functions are necessary and sufficient for secure signatures. In: Proceedings 22nd ACM symposium on the theory of computing. ACM, New York, pp 387–394
9. Shoup V (2000) A composition theorem for universal one-way hash functions. In: Preneel B (ed) Advances in cryptology, proceedings Eurocrypt'00. LNCS, vol 1807. Springer, Berlin, pp 445–452
10. Simon D (1998) Finding collisions on a one-way street: can secure hash functions be based on general assumptions? In: Nyberg K (ed) Advances in cryptology, proceedings Eurocrypt'98. LNCS, vol 1403. Springer, Berlin, pp 334–345
11. Zheng Y, Matsumoto T, Imai H (Jul 1990) Connections between several versions of one-way hash functions. Trans IEICE E E73(7):1092–1099

Unlinkability

Gerrit Bleumer
Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

Related Concepts

►Anonymity; ►Untraceability

Definition

Unlinkability of two events occurring during a process under observation of an attacker is the property that the two events appear to the attacker *after the process* exactly as much related – or unrelated – as they did *before the process* started (see [1]).

Theory

In order to apply the notion of unlinkability to a particular cryptographic scheme, the attacker model needs to be specified, for example, whether it is a *passive attacker*, such as an eavesdropper, or an *active attacker* (►cryptanalysis for this terminology). If passive, which communication lines he can observe and when. If active, how he can interact with the honest system participants (e.g., *oracle access*) and thereby stimulate certain behavior of the honest participants, or how many honest participants he can control entirely (*resilience* in ►threshold signature), and whether the attacker is computationally restricted or computationally unrestricted (computational security). Based on a precise attacker model, certain events occurring in a given cryptographic scheme can then be defined as unconditionally or computationally unlinkable.

Applications

An individual who interacts with other individuals or authorities may keep its interactions unlinkable by using different ►pseudonyms in different transactions. As Rao and Rohatgi [3] showed, this may not be a sufficient measure to achieve unlinkability, but it is usually a necessary one. Anonymity, untraceability, and ►privacy are all closely related to the notion of unlinkability. In fact, many privacy-oriented ►payment *schemes*, ►credential *schemes*, ►electronic voting schemes, and secure auction schemes are built around the notion of unlinkability and employ transaction pseudonyms (see [2]).

Recommended Reading

1. Chaum D (1981) Untraceable electronic mail, return addresses, and digital pseudonyms. Commun ACM 24(2):84–88
2. Chaum D (1986) Showing credentials without identification – signatures transferred between unconditionally unlinkable pseudonyms. In: Pichler F (ed) Advances in cryptology – EUROCRYPT'85. Lecture notes in computer science, vol 219. Springer, Berlin, pp 241–244
3. Rao JR, Rohatgi P (2000) Can pseudonyms really guarantee privacy? In: Proceedings of the 9th USENIX security symposium, Denver, 14–17 Aug 2000

Unpacking Malware

Brent Byung Hoon Kang[1], Greg Sinclair[2]
[1]Department of Applied Information Technology and Centre for Secure Information Systems, The Volgenau School of Engineering, George Mason University, Fairfax, VA, USA
[2]The Volgenau School of Engineering and IT, Centre for Secure Information Systems, George Mason University, Fairfax, VA, USA

Synonyms

Deobfuscating malware

Definition

Unpacking is the process of recovering original binary code from the obfuscated and packed binaries.

Background

Packing is the process of obfuscating the original binary with high number of redirections and complex mesh of

stub codes into a new packed binary that are completely different from the original binary. The packing is often used in malware binary in order to make the reversing of the binary far more difficult and time-consuming.

Theory and Application

There have been numerous efforts in developing automated unpacking systems. An early implementation of an automatic unpacking process relies on the use of a debugger to step through each instruction of the packed binary in order to determine when the binary begins to execute code not originally found in the packed binary. For example, the principle behind the PolyUnpack [1] system relies on the fact that packed binaries do not contain the unpacked binary's code at start-up. In other words, the only code typically available during the initial execution of a packed binary is that of the packer. Therefore, as the packer's unpacking subroutines begin the process of recreating the original binary in memory (be in this memory newly allocated or in pre-allocated regions as defined by the image's header), by comparing the executing code against the original code found at the start of the executable's life cycle, it is impossible to determine when a packer has begun executing the original binary's executable code section.

It is possible to use multiple layers of packing when obfuscated a binary. A malware author can simply apply multiple, distinct packing systems to the same binary in an iterative manner to complicate the task of recovering the original binary. By taking a snapshot of the executable's memory space at the moment when the system detects an unpacked memory section being executed, the automatic unpacking system presented [1] can find the true original binary by repeating the process of monitoring the execution of "unpacked" code until such time as the unpacked code no longer generates new memory sections which it transfers executional control into.

This method of determining the beginning of the original binary, while effective, does have certain failings. The method does determine the location of the original binary in memory, but it may not necessarily find the right Original Entry Point (OEP), the point where the original program in its unpacked state would normally begin execution. Gou et al. explored this problem in the paper [2]. They proposed a new solution for making a more accurate OEP determination. By observing the state of the stack and location of the current instruction being executed at any given point during the execution of the unpacking subroutines, it is possible to more accurately determine the location of the OEP. The heuristic proposed consists of three key attributes: (1) Is the memory under which the current code executing different than the original executable's memory image or possibly new? (2) Is the stack in a similar state as would be found when an unpacked binary initially executes? (3) Are the command line arguments (argv, argc) located on the stack correctly?

These three conditions, if met, provide a reasonable method for accurately determining the location of the OEP. As an artifact of the unpacking process, a packer must return the stack allocated for the running binary image to the point where the unpacked binary will recognize the stack as if the binary were executed initially in its unpacked state. This requires establishing key stack locations such as the location of the command-line arguments passed when the program originally executes. If the stack does not exist in the correct format, then the unpacked binary, when given the initial stack, will fail to properly execute due to the inappropriate input.

Furthermore, the Guo heuristic extends the concept of detecting new executable code presented in PolyUnpack by putting additional emphasis on the transfer of execution control to newly allocated memory locations. When an unpacker executes, the original binary may indeed be compressed requiring the unpacker to allocate additional memory to accommodate the original binary's size. It is typically abnormal for an unpacked binary to allocate memory and transfer control to such a memory location. It is more common, however, for a binary to allocate memory and use the memory for storage of data, not code. Therefore, by noting this transfer of control to a newly allocated memory location, the heuristic can easily determine when control is being transferred to unpacked code.

Recommended Reading

1. Royal P, Halpin M, Dagon D, Edmonds R, Lee W (2006) Polyunpack: automating the hidden-code extraction of unpack-executing malware. In: Proceedings of the 22nd annual computer security applications conference (ACSAC), Miami Beach, December 2006
2. Guo F, Ferrie P, Chiueh T (2008) A study of the packer problem and its solutions. In: 11th international symposium on recent advances in intrusion detection Raid 2008, Cambridge, 15–17 Sep 2008

Untraceability

Gerrit Bleumer
Research and Development, Francotyp Group,
Birkenwerder bei Berlin, Germany

Related Concepts

►Anonymity; ►Unlinkability

Definition
Untraceability of an object during a process under observation of an attacker is the property that the attacker cannot follow the trace of the object as it moves from one participant or location to another.

Theory
Untraceability is achieved during a process under observation of an attacker is achieved if the attacker finds the distinct snapshots obtained from the transactions of any one object to be pairwise unlinkable. Untraceability is called unconditional or computational if the pairwise unlinkability is so.

Applications
The security property of untraceability is relevant to e-mail systems, where in certain anonymizing networks e-mails cannot be observed by attackers to flow from a sender to a recipient through a sequence of network nodes (see [1]). Another prominent example is an electronic coin, which in certain privacy-oriented ►electronic cash schemes cannot be traced being spent from one participant to another.

Recommended Reading
1. Chaum D (1981) Untraceable electronic mail, return addresses, and digital pseudonyms. Commun ACM 24(2):84–88

User Authentication

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

Related Concepts
►Authentication; ►Biometric Authentication; ►Biometrics: Terms and Definitions; ►Entity Authentication; ►Password; ►User Verification Method

Definition
User authentication is identical to ►entity authentication except that the term "entity" is restricted to denoting a human user (as opposed to a server, a process, a device, a computer terminal, or any other system entity). In practice, this restriction typically limits the possible ►authentication techniques to ►password-based schemes and ►biometric schemes.

User Verification Method

Marijke De Soete
Security4Biz, Oostkamp, Belgium

Synonyms
User authentication

Related Concepts and Keywords
►Password; ►PIN

Definition
A method for checking that a user (person) is the one claimed.

Theory
Verification that an entity is the one claimed is generally done through the usage of authentication factors. An authentication factor is a piece of information and process used to authenticate or verify a person's identity for security purposes. ►Two-factor authentication is a system wherein two different methods are used to authenticate. Using two factors as opposed to one delivers a higher level of authentication assurance.

There are three universally recognized factors or a combination thereof for authenticating individuals:

- Something the person knows
- Something the person possesses
- Something the person is (using a personal characteristic)

Using something only the person knows is the "classical way" to corroborate a user's identity based on a secret piece of information such as a password, a ►PIN code, a passphrase, etc.

Using or providing something the person possesses, is usually based on a physical token like an identity badge, a ►smart card, a magnetic strip card, an authentication token, etc.

Something the person is deals with making use of biometric attributes of the person in order to corroborate its identity such as a finger print, hand minutiae, a retina scan, etc.

Applications
User verification methods are used for access control and authentication purposes.

Recommended Reading
1. http://ec.europa.eu/internal_market/payments/docs/fraud/study-security/study_WP1_25112007_en.pdf