Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information
Wikidata item
Cite this page

Print/export
Create a book
Download as PDF
Printable version

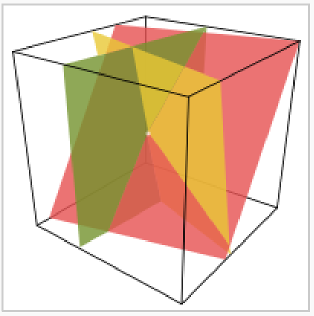
Languages
Dansk
Deutsch
Français
עברית
Nederlands
日本語
Polski
Русский
Simple English
Edit links

Secret sharing

From Wikipedia, the free encyclopedia

For cases when the whole secret is known by all participants, see shared secret.

Secret sharing (also called **secret splitting**) refers to methods for distributing a *secret* amongst a group of participants, each of whom is allocated a *share* of the secret. The secret can be reconstructed only when a sufficient number, of possibly different types, of shares are combined together; individual shares are of no use on their own.


Each secret share is a plane, and the secret is the point at which three shares intersect. Two shares yield only a line intersection.

In one type of secret sharing scheme there is one *dealer* and *n players*. The dealer gives a share of the secret to the players, but only when specific conditions are fulfilled will the players be able to reconstruct the secret from their shares. The dealer accomplishes this by giving each player a share in such a way that any group of *t* (for *threshold*) or more players can together reconstruct the secret but no group of fewer than *t* players can. Such a system is called a *(t, n)*-threshold scheme (sometimes it is written as an *(n, t)*-threshold scheme).

Secret sharing was invented independently by Adi Shamir[1] and George Blakley[2] in 1979.

Contents [hide]

Importance [edit]

Secret sharing schemes are ideal for storing information that is highly sensitive and highly important. Examples include: encryption keys, missile launch codes, and numbered bank accounts. Each of these pieces of information must be kept highly confidential, as their exposure could be disastrous, however, it is also critical that they should not be lost. Traditional methods for encryption are ill-suited for simultaneously achieving high levels of confidentiality and reliability. This is because when storing the encryption key, one must choose between keeping a single copy of the key in one location for maximum secrecy, or keeping multiple copies of the key in different locations for greater reliability. Increasing reliability of the key by storing multiple copies lowers confidentiality by creating additional attack vectors; there are more opportunities for a copy to fall into the wrong hands. Secret sharing schemes address this problem, and allow arbitrarily high levels of confidentiality and reliability to be achieved.

Secret sharing schemes are important in cloud computing environments. Thus a key can be distributed over many servers by a threshold secret sharing mechanism. The key is then reconstructed when needed. Secret sharing has also been suggested for sensor networks where the links are liable to be tapped by sending the data in shares which makes the task of the eavesdropper harder. The security in such environments can be made greater by continuous changing of the way the shares are constructed.

"Secure" versus "insecure" secret sharing [edit]

A secure secret sharing scheme distributes shares so that anyone with fewer than *t* shares has no extra information about the secret than someone with 0 shares.

Consider for example the secret sharing scheme in which the secret phrase "password" is divided into the shares "pa------," "--ss----," "----wo--," and "------rd,". A person with 0 shares knows only that the password consists of eight letters. He would have to guess the password from 26^8 = 208 billion possible combinations. A person with one share, however, would have to guess only the six letters, from 26^6 = 308 million combinations, and so on as more persons collude. Consequently this system is not a "secure" secret sharing scheme, because a player with fewer than *t* secret-shares is able to reduce the problem of obtaining the inner secret without first needing to obtain all of the necessary shares.

In contrast, consider the secret sharing scheme where X is the secret to be shared, P_i are public asymmetric encryption keys and Q_i their corresponding private keys. Each player J is provided with $\{P_1(P_2(...(P_N(X)))), Q_j\}$. In this scheme, any player with private key 1 can remove the outer layer of encryption, a player with keys 1 and 2 can remove the first and second layer, and so on. A player with fewer than N keys can never fully reach the secret X without first needing to decrypt a public-key-encrypted blob for which he does not have the corresponding private key - a problem that is currently believed to be computationally infeasible. Additionally we can see that any user with all N private keys is able to decrypt all of the outer layers to obtain X, the secret, and consequently this system is a secure secret distribution system.

Limitations [edit]

Several secret sharing schemes are said to be information theoretically secure and can be proven to be so, while others give up this *unconditional security* for improved efficiency while maintaining enough security to be considered as secure as other common cryptographic primitives. For example, they might allow secrets to be protected by shares with 128-bits of entropy each, since each share would be considered enough to stymie any conceivable present-day adversary, requiring a brute force attack of average size 2^{127}.

Common to all unconditionally secure secret sharing schemes, there are limitations:

- Each share of the secret must be at least as large as the secret itself. This result is based in information theory, but can be understood intuitively. Given *t-1* shares, no information whatsoever can be determined about the secret. Thus, the final share must contain as much information as the secret itself. There is sometimes a workaround for this limitation by first compressing the secret before sharing it, but this is often not possible because many secrets (keys for example) look like high-quality random data and thus are hard to compress.
- All secret sharing schemes use random bits. To distribute a one-bit secret among threshold *t* people, *t-1* random bits are necessary. To distribute a secret of arbitrary length entropy of *(t-1)*length* is necessary.

Trivial secret sharing [edit]

t = 1 [edit]

t = 1 secret sharing is very trivial. The secret can simply be distributed to all *n* participants.

t = *n* [edit]

There are several *(t, n)* secret sharing schemes for *t* = *n*, when all shares are necessary to recover the secret:

1. Encode the secret as an arbitrary length binary number *s*. Give to each player *i* (except one) a random number p_i with the same length as *s*. Give to the last player the result of $(s \text{ XOR } p_1 \text{ XOR } p_2 \text{ XOR } ... \text{ XOR } p_{n-1})$ where *XOR* is bitwise exclusive or. The secret is the bitwise XOR of all the players' numbers (*p*).
2. Additionally, (1) can be performed using any linear operator in any field. For example, here's an alternative that is functionally equivalent to (1). Let's select 32-bit integers with well-defined overflow semantics (i.e. the correct answer is preserved, modulo 2^32). First, *s* can be divided into a vector of M 32-bit integers called v_{secret}. Then (n-1) players are each given a vector of M random integers, player *i* receiving v_i. The remaining player is given $v_n=(v_{secret} - v_1 - v_2 - ... - v_{n-1})$. The secret vector can then be recovered by summing across all the player's vectors.

1 < *t* < *n*, and, more general, any desired subset of *n* [edit]

The difficulty lies in creating schemes that are still secure, but do not require all *n* shares. For example, imagine that the Board of Directors of a company would like to protect their secret formula. The president of the company should be able to access the formula when needed, but in an emergency any 3 of the 12 board members would be able to unlock the secret formula together. This can be accomplished by a secret sharing scheme with *t* = 3 and *n* = 15, where 3 shares are given to the president, and 1 is given to each board member.

When space efficiency is not a concern, trivial *t* = *n* schemes can be used to reveal a secret to any desired subsets of the players simply by applying the scheme for each subset. For example, to reveal a secret *s* to any two of the three players Alice, Bob and Carol, create three different (2,2) secret shares for *s*, giving the three sets of two shares to Alice and Bob, Alice and Carol, and Bob and Carol.

Efficient secret sharing [edit]

The trivial approach quickly becomes impractical as the number of subsets increases, for example when revealing a secret to any 50 of 100 players. In the worst case, the increase is exponential. This has led to the search for schemes that allow secrets to be shared efficiently with a threshold of players.

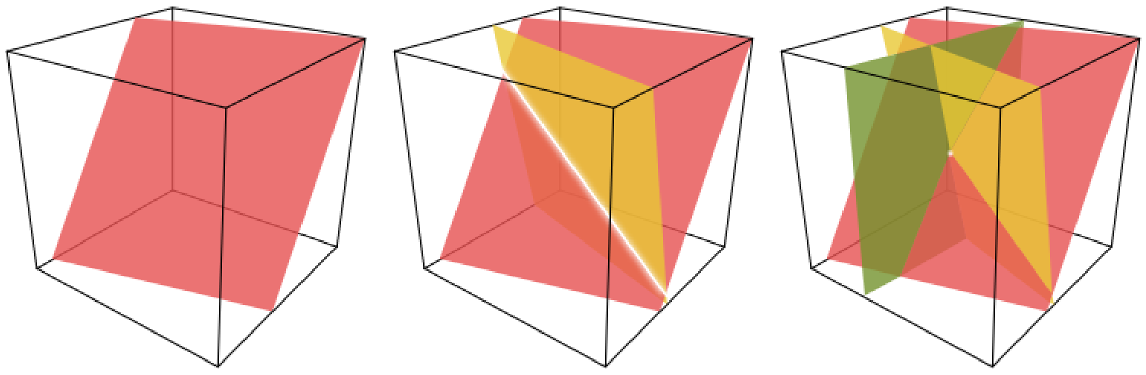
Shamir's scheme [edit]

Main article: Shamir's Secret Sharing

In this scheme, any *t* out of *n* shares may be used to recover the secret. The system relies on the idea that you can fit a unique polynomial of degree *(t-1)* to any set of *t* points that lie on the polynomial. It takes two points to define a straight line, three points to fully define a quadratic, four points to define a cubic curve, and so on. That is, it takes *t* points to define a polynomial of degree *t-1*. The method is to create a polynomial of degree *t-1* with the secret as the first coefficient and the remaining coefficients picked at random. Next find *n* points on the curve and give one to each of the players. When at least *t* out of the *n* players reveal their points, there is sufficient information to fit a *(t-1)*th degree polynomial to them, the first coefficient being the secret.

Blakley's scheme [edit]

Two nonparallel lines in the same plane intersect at exactly one point. Three nonparallel planes in space intersect at exactly one point. More generally, any *n* nonparallel *(n-1)-dimensional hyperplanes* intersect at a specific point. The secret may be encoded as any single coordinate of the point of intersection. If the secret is encoded using all the coordinates, even if they are random, then an insider (someone in possession of one or more of the (n-1)-dimensional hyperplanes) gains information about the secret since he knows it must lie on his plane. If an insider can gain any more knowledge about the secret than an outsider can, then the system no longer has information theoretic security. If only one of the n coordinates is used, then the insider knows no more than an outsider (i.e., that the secret must lie on the x-axis for a 2-dimensional system). Each player is given enough information to define a hyperplane; the secret is recovered by calculating the planes' point of intersection and then taking a specified coordinate of that intersection.


Blakley's scheme in three dimensions: each share is a plane, and the secret is the point at which three shares intersect. Two shares are insufficient to determine the secret, although they do provide enough information to narrow it down to the line where both planes intersect.

Blakley's scheme is less space-efficient than Shamir's; while Shamir's shares are each only as large as the original secret, Blakley's shares are *t* times larger, where *t* is the threshold number of players. Blakley's scheme can be tightened by adding restrictions on which planes are usable as shares. The resulting scheme is equivalent to Shamir's polynomial system.

Using the Chinese remainder theorem [edit]

Main article: Secret sharing using the Chinese remainder theorem

The Chinese remainder theorem can also be used in secret sharing, for it provides us with a method to uniquely determine a number *S* modulo *k* many relatively prime integers $m_1, m_2, ..., m_k$, given that $S < \prod_{i=1}^{k} m_i$. There are two secret sharing schemes that make use of the Chinese Remainder Theorem, Mignotte's and Asmuth-Bloom's Schemes. They are threshold secret sharing schemes, in which the shares are generated by reduction modulo the integers m_i, and the secret is recovered by essentially solving the system of congruences using the Chinese Remainder Theorem.

Proactive secret sharing [edit]

Main article: Proactive secret sharing

If the players store their shares on insecure computer servers, an attacker could crack in and steal the shares. If it is not practical to change the secret, the uncompromised (Shamir-style) shares can be renewed. The dealer generates a new random polynomial with constant term zero and calculates for each remaining player a new ordered pair, where the x-coordinates of the old and new pairs are the same. Each player then adds the old and new y-coordinates to each other and keeps the result as the new y-coordinate of the secret.

All of the non-updated shares the attacker accumulated become useless. An attacker can only recover the secret if he can find enough other non-updated shares to reach the threshold. This situation should not happen because the players deleted their old shares. Additionally, an attacker cannot recover any information about the original secret from the update files because they contain only random information.

The dealer can change the threshold number while distributing updates, but must always remain vigilant of players keeping expired shares.

Verifiable secret sharing [edit]

Main article: Verifiable secret sharing

A player might lie about his own share to gain access to other shares. A *verifiable secret sharing* (VSS) scheme allows players to be certain that no other players are lying about the contents of their shares, up to a reasonable probability of error. Such schemes cannot be computed conventionally; the players must collectively add and multiply numbers without any individual's knowing what exactly is being added and multiplied. Tal Rabin and Michael Ben-Or devised a *multiparty computing (MPC)* system that allows players to detect dishonesty on the part of the dealer or on part of up to one third of the threshold number of players, even if those players are coordinated by an "adaptive" attacker who can change strategies in realtime depending on what information has been revealed.

Computationally secure secret sharing [edit]

The disadvantage of unconditionally secure secret sharing schemes is that the storage and transmission of the shares requires an amount of storage and bandwidth resources equivalent to the size of the secret times the number of shares. If the size of the secret were significant, say 1 GB, and the number of shares were 10, then 10 GB of data must be stored by the shareholders. Alternate techniques have been proposed for greatly increasing the efficiency of secret sharing schemes, by giving up the requirement of unconditional security.

One of these techniques, known as *secret sharing made short*,[3] combines Rabin's information dispersal algorithm[4] (IDA) with Shamir's secret sharing. Data is first encrypted with a randomly generated key, using a symmetric encryption algorithm. Next this data is split into N pieces using Rabin's IDA. This IDA is configured with a threshold, in a manner similar to secret sharing schemes, but unlike secret sharing schemes the size of the resulting data grows by a factor of (number of fragments / threshold). For example, if the threshold were 10, and the number of IDA-produced fragments were 15, the total size of all the fragments would be (15/10) or 1.5 times the size of the original input. In this case, this scheme is 10 times more efficient than if Shamir's scheme had been applied directly on the data. The final step in secret sharing made short is to use Shamir secret sharing to produce shares of the randomly generated symmetric key (which is typically on the order of 16–32 bytes) and then give one share and one fragment to each shareholder.

A related approach, known as AONT-RS,[5] applies an All-or-nothing transform to the data as a pre-processing step to an IDA. The All-or-nothing transform guarantees that any number of shares less than the threshold is insufficient to decrypt the data.

Space efficient secret sharing [edit]

Information theoretically secure secret sharing schemes are space inefficient because a k-out-of-n secret sharing technique generates n shares each of size at least that of the secret itself, leading to a n-fold increase in required storage. In space efficient secret sharing, devised by Abhishek Parakh and Subhash Kak, each share is roughly the fraction (k-1) of the size of the secret.[6] This scheme makes use of repeated polynomial interpolation and has potential applications in secure information dispersal on the Web and in sensor networks. This method is based on data partitioning involving the roots of a polynomial in finite field.[7]

Other uses and applications [edit]

A secret sharing scheme can secure a secret over multiple servers and remain recoverable despite multiple server failures. The dealer may act as several distinct participants, distributing the shares among the participants. Each share may be stored on a different server, but the dealer can recover the secret even if several servers break down as long as they can recover at least *t* shares; however, crackers that break into one server would still not know the secret as long as fewer than *t* shares are stored on each server.

This is one of the major concepts behind the Vanish computer project at the University of Washington, where a random key is used to encrypt data, and the key is distributed as a secret across several nodes in a P2P network. In order to decrypt the message, at least *t* nodes on the network must be accessible; the principle for this particular project being that the number of secret-sharing nodes on the network will decrease naturally over time, therefore causing the secret to eventually *vanish*. However, the network is vulnerable to a Sybil attack, thus making Vanish insecure.[8]

Note also that any shareholder who ever has enough information to decrypt the content at any point is able to take and store a copy of X. Consequently although tools and techniques such as Vanish can make data irrecoverable within their own system after a time, it is not possible to force the deletion of data once a malicious user has seen it. This is one of the leading conundrums of Digital Rights Management.

A dealer could send *t* shares, all of which are necessary to recover the original secret, to a single recipient. An attacker would have to intercept all *t* shares to recover the secret, a task which is more difficult than intercepting a single file, especially if the shares are sent using different media (e.g. some over the Internet, some mailed on CDs).

For large secrets, it may be more efficient to encrypt the secret and then distribute the key using secret sharing.

Secret sharing is an important primitive in several protocols for secure multiparty computation.

See also [edit]

- Shamir's Secret Sharing
- Homomorphic secret sharing - A simplistic decentralized voting protocol.
- Byzantine fault tolerance
- Access structure
- Secure multiparty computation
- Visual cryptography
- Tontine
- Secret sharing using the Chinese remainder theorem
- Orthogonal array - Used to construct some threshold schemes.

Notes [edit]

1. ^ Shamir, Adi (1979). "How to share a secret". Communications of the ACM 22 (11): 612–613.
2. ^ Blakley, G. R. (1979). "Safeguarding cryptographic keys". Proceedings of the National Computer Conference 48: 313–317.
3. ^ Krawczyk, Hugo (1993). *Secret Sharing Made Short* (PDF). CRYPTO '93.
4. ^ Rabin, Michael O. (1989). "Efficient dispersal of information for security, load balancing, and fault tolerance". *Journal of the ACM* **36** (2).
5. ^ Resch, Jason; Plank, James (February 15, 2011). *AONT-RS: Blending Security and Performance in Dispersed Storage Systems* (PDF). Usenix FAST'11.
6. ^ Parakh, A. and Kak, S. Space efficient secret sharing for implicit data security. Information Sciences, vol. 181, pp. 335-341, 2011.
7. ^ Parakh, A. and Kak, S. Online data storage using implicit security. Information Sciences, vol. 179, pp. 3323-3331, 2009.
8. ^ "'Unvanish: Reconstructing Self-Destructing Data'".

External links [edit]

- ssss: A free (GPL) implementation of Shamir's Scheme with online demo.
- libgfshare is a free implementation (C library and commandline tools) of Shamir's scheme in GF(256).
- Description of Shamir's and Blakley's schemes
- Patent for use of secret sharing for recovering PGP (and other?) pass phrases U.S. Patent 6,662,299
- A bibliography on secret-sharing schemes
- Code signing systems using Shared Secret at the Wayback Machine (archived February 14, 2008)
- Christophe David's web based implementation of Shamir's scheme 'How to share a Secret'
- Software products from IBM, Sun, Netscape and ZenithSecure and hardware products from Safenet use secret sharing. There are libraries for secret sharing in several programming languages.
- SecretSharing A QT implementation of Shamir's scheme using GF(2^8) field arithmetic.
- Java library implementation of multiple secret sharing methods, opensource(LGPLv2)
- Beimel, Amos (2011). "Secret-Sharing Schemes: A Survey" (PDF).