Secret Sharing

Liam Hardiman

November 19, 2018

What is Secret Sharing?

- Eleven scientists are working on a secret project. They wish to lock up the documents in a cabinet so that the cabinet can be opened if and only if six or more of the scientists are present. What is the smallest number of locks needed? What is the smallest number of keys to the locks each scientist must carry?[1]

[1] Liu, C.L *Introduction to Combinatorial Mathematics*. McGraw-Hill, New York, 1968

What is Secret Sharing?

- Eleven scientists are working on a secret project. They wish to lock up the documents in a cabinet so that the cabinet can be opened if and only if six or more of the scientists are present. What is the smallest number of locks needed? What is the smallest number of keys to the locks each scientist must carry?[1]
- $\binom{11}{6} = 462$ locks and $\binom{10}{5} = 252$ keys.

[1] Liu, C.L *Introduction to Combinatorial Mathematics*. McGraw-Hill, New York, 1968

What is Secret Sharing?

The goal of secret sharing is dividing a secret S into n pieces, called *shares*, so that no fewer than $k \leq n$ shares are sufficient for reassembling S. This is called a (k, n)-**threshold scheme**.

Why is Secret Sharing?

"Threshold schemes are ideally suited to applications in which a group of mutually suspicious individuals with conflicting interests must cooperate." – Adi Shamir

Example Application

- Imagine a company stamps its checks with a digital signature

Example Application

- Imagine a company stamps its checks with a digital signature
- Risky to give the signing key to every low-level executive

Example Application

- Imagine a company stamps its checks with a digital signature
- Risky to give the signing key to every low-level executive

Example Application

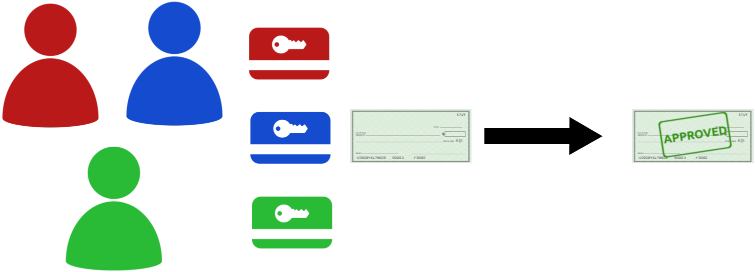
- Set up an $(n, 3)$-threshold scheme for the key

Example Application

- Set up an $(n, 3)$-threshold scheme for the key
- Give each executive a key fragment card and a check signing machine

Example Application

- Set up an $(n, 3)$-threshold scheme for the key
- Give each executive a key fragment card and a check signing machine

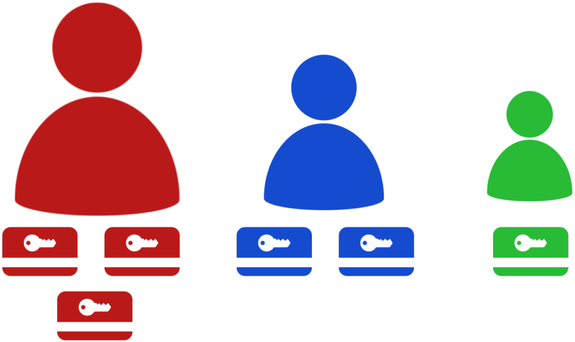


Example Application

- This solution is flexible

Example Application

- This solution is flexible



How is Secret Sharing? - Shamir's Method (1979)[2]

- Say we want to set up a (k, n)-threshold scheme

[2] A. Shamir, "How to share a secret". Commun. ACM 22(11), 612613 (1979)

How is Secret Sharing? - Shamir's Method (1979)[2]

- Say we want to set up a (k, n)-threshold scheme
- Choose a big prime p and let the secret, S, be an element in $\mathbb{Z}/p\mathbb{Z}$

[2] A. Shamir, "How to share a secret". Commun. ACM 22(11), 612613 (1979)

How is Secret Sharing? - Shamir's Method (1979)[2]

- Say we want to set up a (k, n)-threshold scheme
- Choose a big prime p and let the secret, S, be an element in $\mathbb{Z}/p\mathbb{Z}$
- Choose random elements $a_1, \ldots, a_{k-1} \in \mathbb{Z}/p\mathbb{Z}$. Set

$$p(x) = S + a_1 x + \cdots + a_{k-1} x^{k-1}.$$

[2] A. Shamir, "How to share a secret". Commun. ACM 22(11), 612613 (1979)

How is Secret Sharing? - Shamir's Method (1979)[2]

- Say we want to set up a (k, n)-threshold scheme
- Choose a big prime p and let the secret, S, be an element in $\mathbb{Z}/p\mathbb{Z}$
- Choose random elements $a_1, \ldots, a_{k-1} \in \mathbb{Z}/p\mathbb{Z}$. Set

$$p(x) = S + a_1 x + \cdots + a_{k-1} x^{k-1}.$$

- Issue to person i, $1 \leq i \leq n$, the share $D_i = (i, p(i))$.

[2] A. Shamir, "How to share a secret". Commun. ACM 22(11), 612613 (1979)

How is Secret Sharing? - Shamir's Method (1979)

- m shares, $D_1, \ldots, D_m$, give us m linear equations in k unknowns

How is Secret Sharing? - Shamir's Method (1979)

- m shares, $D_1, \ldots, D_m$, give us m linear equations in k unknowns

$$\begin{array}{ccccccccc}
S & + & a_1 \cdot 1 & + & \cdots & + & a_{k-1}(1)^{k-1} & = & p(1) \\
S & + & a_2 \cdot 2 & + & \cdots & + & a_{k-1}(2)^{k-1} & = & p(2) \\
 & & & & & & \vdots & & \\
S & + & a_1 \cdot m & + & \cdots & + & a_{k-1}(m)^{k-1} & = & p(m)
\end{array}$$

- Linear algebra tells us we get a solution if and only if we have at least k equations (shares)

Downside to Shamir's Scheme

Each share is an element of $\mathbb{Z}/p\mathbb{Z}$, just like the secret. n shares means n times the storage.

Information Dispersal - Rabin (1989)[3]

- Split a file F into n pieces so that any $k \leq n$ pieces can reconstruct F

[3] M. O. Rabin, "Efficient Dispersal of Information for Security, Load Balancing, and Fault Tolerance". In: Journal of the ACM, vol. 36, iss. 2, 1989, pp. 335-348

Information Dispersal - Rabin (1989)[3]

- Split a file F into n pieces so that any $k \leq n$ pieces can reconstruct F
- Each piece has size roughly $|F|/k$. That means roughly n/k blowup, which can be close to unity

[3] M. O. Rabin, "Efficient Dispersal of Information for Security, Load Balancing, and Fault Tolerance". In: Journal of the ACM, vol. 36, iss. 2, 1989, pp. 335-348

Information Dispersal - Rabin (1989)[3]

- Split a file F into n pieces so that any $k \leq n$ pieces can reconstruct F
- Each piece has size roughly $|F|/k$. That means roughly n/k blowup, which can be close to unity
- Not perfectly secret

[3] M. O. Rabin, "Efficient Dispersal of Information for Security, Load Balancing, and Fault Tolerance". In: Journal of the ACM, vol. 36, iss. 2, 1989, pp. 335-348

Information Dispersal - Rabin (1989)

- Split the 800 byte file, $F = b_1, \ldots, b_{800}$, among 15 people so that any 8 of them can reassemble it. Each b_i is an integer between 0 and 255

Information Dispersal - Rabin (1989)

- Split the 800 byte file, $F = b_1, \ldots, b_{800}$, among 15 people so that any 8 of them can reassemble it. Each b_i is an integer between 0 and 255
- Break the file into 8 byte blocks

$$\begin{aligned} F &= (b_1, \ldots, b_8), (b_9, \ldots, b_{16}), \ldots, (b_{793}, \ldots, b_{800}) \\ &= f_1, f_2, \ldots, f_{100} \end{aligned}$$

Information Dispersal - Rabin (1989)

- Split the 800 byte file, $F = b_1, \ldots, b_{800}$, among 15 people so that any 8 of them can reassemble it. Each b_i is an integer between 0 and 255
- Break the file into 8 byte blocks

$$
\begin{aligned}
F &= (b_1, \ldots, b_8), (b_9, \ldots, b_{16}), \ldots, (b_{793}, \ldots, b_{800}) \\
&= f_1, f_2, \ldots, f_{100}
\end{aligned}
$$

Information Dispersal - Rabin (1989)

- To create 15 shares, choose 15 vectors, a_j, $1 \leq j \leq 15$ in $(\mathbb{Z}/p\mathbb{Z})^8$ so that any 8 different vectors is linearly independent

Information Dispersal - Rabin (1989)

- To create 15 shares, choose 15 vectors, a_j, $1 \leq j \leq 15$ in $(\mathbb{Z}/p\mathbb{Z})^8$ so that any 8 different vectors is linearly independent
- To compute the j-th share, we compress each block of F by calculating the dot product $F_{ji} = a_j \cdot f_i \pmod{p}$ for each $1 \leq i \leq 100$.

Information Dispersal - Rabin (1989)

- To create 15 shares, choose 15 vectors, a_j, $1 \leq j \leq 15$ in $(\mathbb{Z}/p\mathbb{Z})^8$ so that any 8 different vectors is linearly independent
- To compute the j-th share, we compress each block of F by calculating the dot product $F_{ji} = a_j \cdot f_i \pmod{p}$ for each $1 \leq i \leq 100$.

$$\text{File block} \;\Big\|\; \underbrace{b_1 \mid b_2 \mid b_3 \mid b_4 \mid b_5 \mid b_6 \mid b_7 \mid b_8}_{f_1} \;\Big\|\; \underbrace{b_9 \mid \cdots}$$

$$j\text{-th Share Vector} \;\Big\|\; \underbrace{a_{j1} \mid a_{j2} \mid a_{j3} \mid a_{j4} \mid a_{j5} \mid a_{j6} \mid a_{j7} \mid a_{j8}}_{a_j} \;\Big\|\; \underbrace{a_{j1} \mid \cdots}$$

$$j\text{-th Share Compressed File} \;\Big\|\; F_{j1} = a_j \cdot f_1 \pmod{p} \;\Big\|\; \cdots$$

Information Dispersal - Rabin (1989)

- This gives a compressed file $F_j = F_{j1}, \ldots, F_{j(100)}$. Let the j-th share be $S_j = (a_j, F_j)$, $1 \leq j \leq 15$.

Information Dispersal - Rabin (1989)

- This gives a compressed file $F_j = F_{j1}, \ldots, F_{j(100)}$. Let the j-th share be $S_j = (a_j, F_j)$, $1 \leq j \leq 15$.
- If we're given 8 shares we get this matrix equation

$$\begin{bmatrix} - & a_1 & - \\ & \vdots & \\ - & a_8 & - \end{bmatrix} \begin{bmatrix} b_1 \\ \vdots \\ b_8 \end{bmatrix} = \begin{bmatrix} F_{1,1} \\ \vdots \\ F_{8,1} \end{bmatrix}.$$

Information Dispersal - Rabin (1989)

- This gives a compressed file $F_j = F_{j1}, \dots, F_{j(100)}$. Let the j-th share be $S_j = (a_j, F_j)$, $1 \le j \le 15$.
- If we're given 8 shares we get this matrix equation

$$\begin{bmatrix} - & a_1 & - \\ & \vdots & \\ - & a_8 & - \end{bmatrix} \begin{bmatrix} b_1 \\ \vdots \\ b_8 \end{bmatrix} = \begin{bmatrix} F_{1,1} \\ \vdots \\ F_{8,1} \end{bmatrix}.$$

- With 8 shares, this matrix is invertible and we can recover the first block. The same matrix recovers all blocks

Secret Sharing with Short Shares - Krawczyk (1994)[4]

- Let's set up a (k, n)-threshold scheme

[4] H. Krawczyk, "Secret Sharing Made Short". In: Stinson D.R. (eds) Advances in Cryptology CRYPTO 93. CRYPTO 1993. Lecture Notes in Computer Science, vol 773.

Secret Sharing with Short Shares - Krawczyk (1994)[4]

- Let's set up a (k, n)-threshold scheme
- Encrypt S with some secure cipher using key K, $E = Enc(S, K)$

[4] H. Krawczyk, "Secret Sharing Made Short". In: Stinson D.R. (eds) Advances in Cryptology CRYPTO 93. CRYPTO 1993. Lecture Notes in Computer Science, vol 773.

Secret Sharing with Short Shares - Krawczyk (1994)[4]

- Let's set up a (k, n)-threshold scheme
- Encrypt S with some secure cipher using key K, $E = Enc(S, K)$
- Use Rabin's information dispersal to split E into n pieces, each with size $\frac{n}{k} \cdot |E|$

[4] H. Krawczyk, "Secret Sharing Made Short". In: Stinson D.R. (eds) Advances in Cryptology CRYPTO 93. CRYPTO 1993. Lecture Notes in Computer Science, vol 773.

Secret Sharing with Short Shares - Krawczyk (1994)[4]

- Let's set up a (k, n)-threshold scheme
- Encrypt S with some secure cipher using key K, $E = Enc(S, K)$
- Use Rabin's information dispersal to split E into n pieces, each with size $\frac{n}{k} \cdot |E|$
- Use Shamir's secret sharing to split K into n pieces, each with size $|K|$

[4] H. Krawczyk, "Secret Sharing Made Short". In: Stinson D.R. (eds) Advances in Cryptology CRYPTO 93. CRYPTO 1993. Lecture Notes in Computer Science, vol 773.