More Fun with Secret Sharing

Overview

⋆ Threshold cryptosystems

- [DF89] Y. Desmedt and Y. Frankel. ‘‘Threshold cryptosystems’’. Advances in Cryptology --- Crypto ’89.

⋆ Proactive secret sharing

- [HJKY95] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung. ‘‘Proactive secret sharing, or: How to cope with perpetual leakage.’’ Advances in Cryptology --- Crypto ’95.

⋆ Verifiable secret sharing

- [Fel87] P. Feldman. ‘‘A practical scheme for non-interactive verifiable secret sharing.’’ Proceedings of the 28th Annual Symposium on the Foundations of Computer Science:427--437. IEEE, October 12--14, 1987.

Secret sharing (review)

[Sha79] How to share a secret D:

- $\star$ Create polynomial of degree $k-1$:

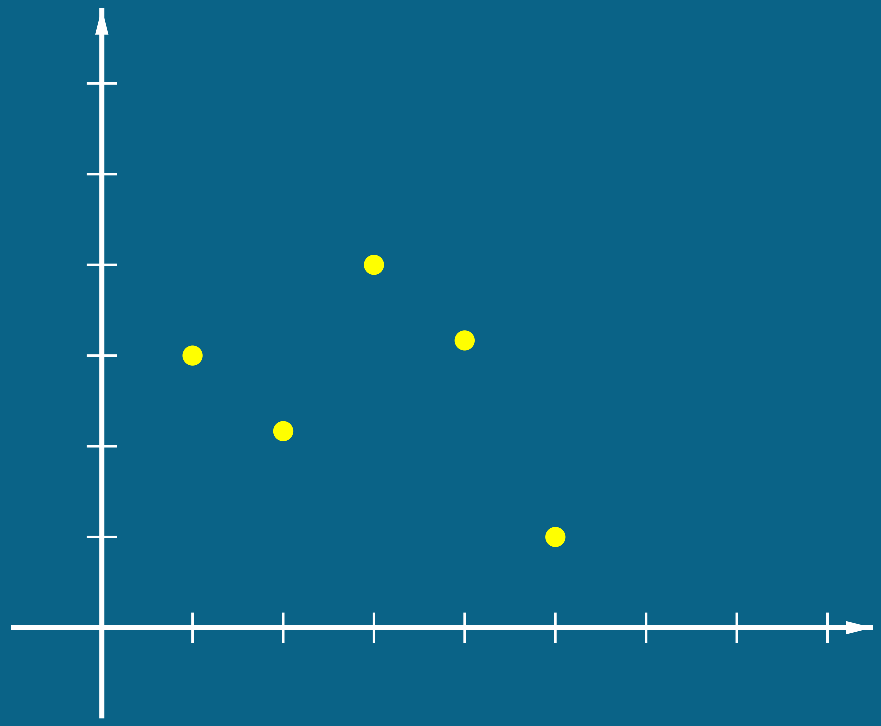
$$f(x) = c_0 + c_1 x + \cdots + c_{k-1} x^{k-1}$$

Assign $c_0 = D$ and choose the other c_i's randomly.

- $\star$ Calculate $f(1), f(2), \ldots, f(n)$
- $\star$ Distribute these $f(x)$ "shares" to participants (along with the

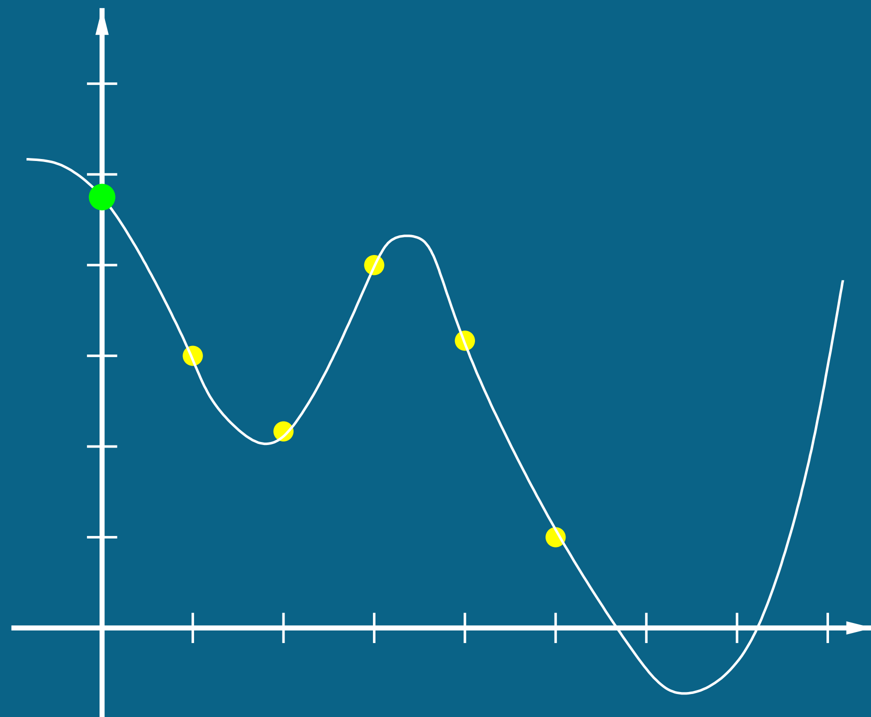
corresponding x values)

- ⋆ To reconstruct secret, gather shares from k participants and interpolate polynomial



corresponding x values)

- ⋆ To reconstruct secret, gather shares from k participants and interpolate polynomial



This is called a (k, n)-*threshold scheme.*

Secret sharing: practical usage?

Public key cryptography

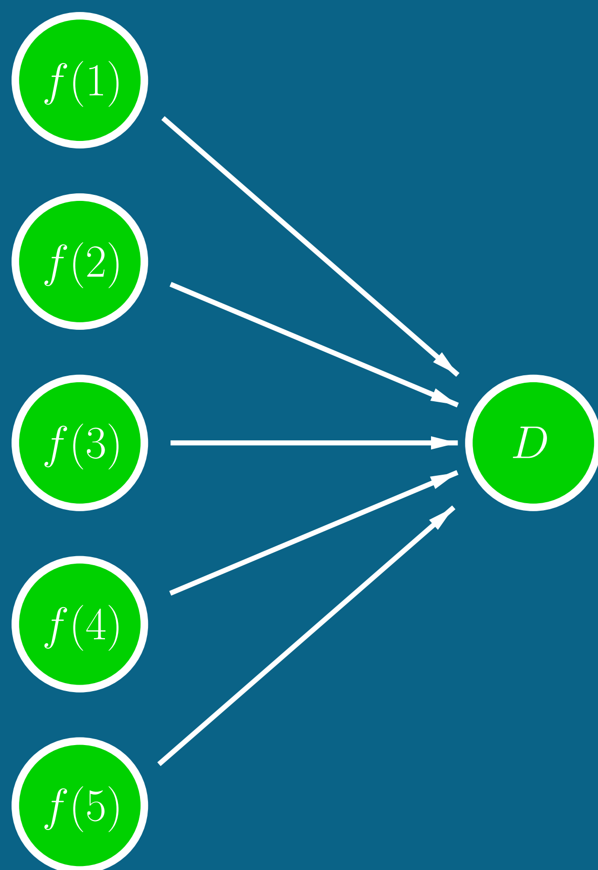
- ⋆ Want to be able to decipher incoming messages, sign outgoing messages for an entire organization
- ⋆ Don't want to distribute single private key to everybody—bad for security
- ⋆ Having private key compromised is more costly than having symmetric key compromised
- ⋆ Shamir's secret sharing sounds tempting

Secret sharing: security?

A (k, n)-threshold scheme is "$(k-1)$-fault tolerant", right?

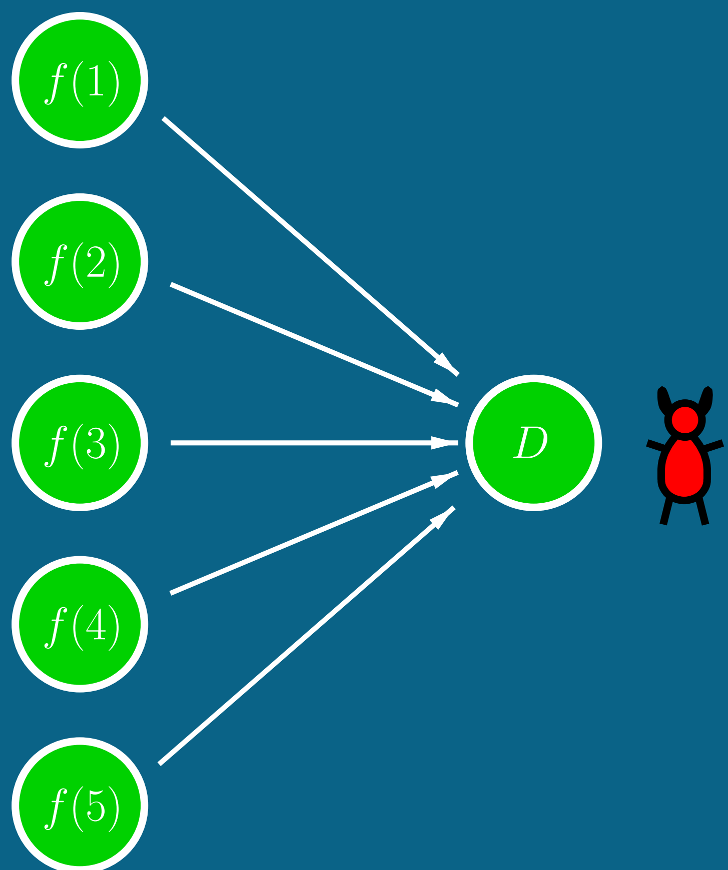
Secret sharing: security?

A (k, n)-threshold scheme is "$(k-1)$-fault tolerant", right?



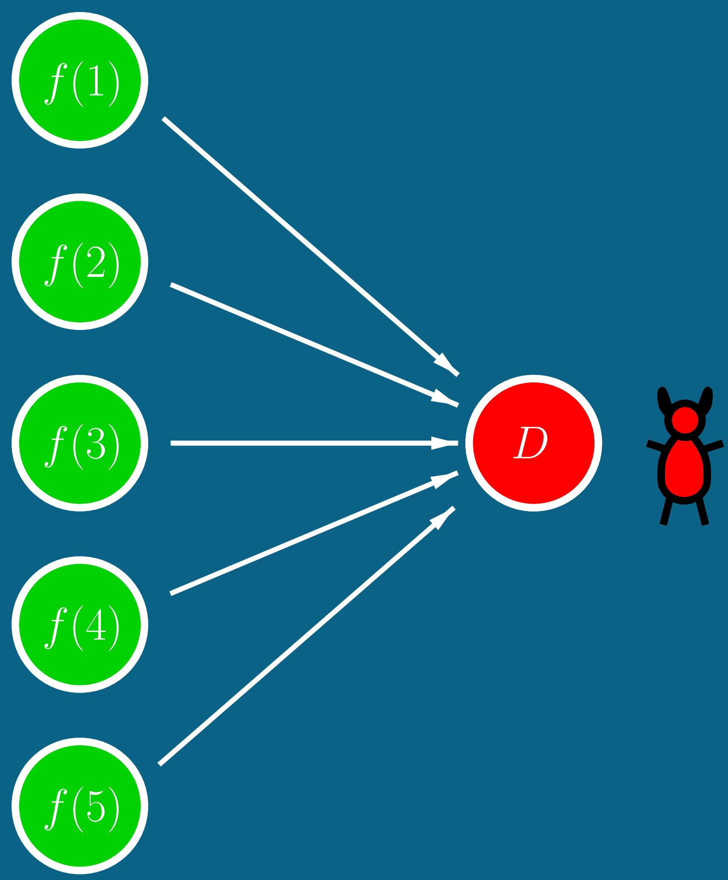
Secret sharing: security?

A (k, n)-threshold scheme is "$(k-1)$-fault tolerant", right?



Secret sharing: security?

A (k, n)-threshold scheme is "$(k-1)$-fault tolerant", right?



Secret sharing: security?

Straightforward application of Shamir's scheme does not provide much secrecy for distributed systems

- ⋆ Without secret sharing, secret is always extant—attacker may compromise designated node at any time
- ⋆ With secret sharing, secret is available intermittently—attacker has to compromise designated node at just the right time
- ⋆ Attacker can possibly trick shareholders into initiating a reconstruction round

Threshold cryptosystems

[DF89] Y. Desmedt and Y. Frankel. "Threshold cryptosystems". *Advances in Cryptology — Crypto '89*.

Threshold cryptosystems

"Threshold cryptosystem" (also called society-oriented cryptosystem)

- ⋆ Performs cryptographic operations without reconstructing private key
- ⋆ Not a generalized scheme like secret sharing—depends on the details of the underlying cryptosystem

ElGamal public key cryptography

[ElG85] T. ElGamal. "A public key cryptosystem and a signature scheme based on discrete logarithms." *IEEE Trans. Info. Theory* IT 31, 1985.

Components:

- ⋆ a large prime p
- ⋆ a generator g for the field $\mathbb{Z}_p$
  - a generator is a number such that $(0, 1, g, g^2, \ldots, g^{p-2})$ is a permutation of the elements in $\mathbb{Z}_p$.
  - For a given prime field $\mathbb{Z}_p$, it turns out there are $\phi(p-1)$ generators and they are not too hard to find.

ElGamal public key cryptography

Components (cont'd):

- ⋆ a secret key a, $0 < a < p - 1$

All calculations performed in $\mathbb{Z}_p$

Publish (p, g, g^a)

Keep a private

ElGamal public key cryptography

To encrypt a message M:

- ⋆ Sender chooses random integer b
- ⋆ Raises g and g^a to the b^{th} power (use successive squaring)
- ⋆ Sends the tuple (g^b, Mg^{ab})

To decrypt:

- ⋆ Receiver uses g^b and a to calculate $(g^{ab})^{-1}$
- ⋆ Multiplies by second entry to yield M

ElGamal public key cryptography

How to crack ElGamal

- If an eavesdropper could determine b from g and g^b, or determine a from g and g^a, the message could be decrypted.
- This is known as the *discrete logarithm* problem.
- No polynomial-time solution is known.
- ElGamal is believed to be secure in general.

ElGamal threshold decryption

To extend ElGamal with secret sharing techniques:

- ⋆ Generate polynomial for secret key a
- ⋆ Distribute (x_i, y_i) shares as normal and destroy polynomial
- ⋆ When an encrypted message arrives, select k participants
- ⋆ Each participant generates a *modified shadow* and computes a partial result on the message with this shadow
- ⋆ Designated node collects all partial results and uses them to decrypt message
- ⋆ Partial results reveal no more about the key than does g^a

Lagrange Interpolating Polynomials

Given k shares $(x_1, y_1), \ldots, (x_k, y_k)$, let

$$\pi_i(x) = \prod_{j=1, j \neq i}^{k} \frac{x - x_j}{x_i - x_j}$$

$$f(x) = \sum_{i=1}^{k} y_i \pi_i(x)$$

Lagrange Interpolating Polynomials

Given k shares $(x_1, y_1), \ldots, (x_k, y_k)$, let

$$\pi_i(x) = \prod_{j=1, j\neq i}^{k} \frac{x - x_j}{x_i - x_j}$$

$$f(x) = \sum_{i=1}^{k} y_i \pi_i(x)$$

Claim: $f(x)$ is our original polynomial

Lagrange Interpolating Polynomials

A simple case ($k = 3$):

$$f(x) = y_1 \frac{(x - x_2)(x - x_3)}{(x_1 - x_2)(x_1 - x_3)} + y_2 \frac{(x - x_1)(x - x_3)}{(x_2 - x_1)(x_2 - x_3)}$$

$$+ y_3 \frac{(x - x_1)(x - x_2)}{(x_3 - x_1)(x_3 - x_2)}$$

Note for all i, $f(x_i) = y_i$.

Lagrange Interpolating Polynomials

Modified shadow for shareholder i is

$$a_i = y_i \pi_i(0)$$

- Only requires knowledge of one's own share and the other x_i involved in this sharing round
- Observe

$$a_1 + a_2 + \cdots + a_k = f(0) = a$$

ElGamal threshold decryption

Each shareholder computes partial result $(g^{ba_i})^{-1}$ and sends to designated node

Designated node multiplies all partial results to decrypt message:

$$\begin{aligned} Mg^{ba}(g^{ba_1})^{-1}\cdots(g^{ba_k})^{-1} &= Mg^{ba}(g^{b(a_1+\cdots+a_k)})^{-1} \\ &= Mg^{ba}(g^{ba})^{-1} \\ &= M \end{aligned}$$

ElGamal threshold decryption

Enhancement to make this less interactive:

- ⋆ Each node computes partial result g^{by_i}
- ⋆ Designated node exponentiates each partial result by $\pi_i(0)$ before multiplying

Also provide solution using geometry-based secret sharing

Drawbacks:

- ⋆ Cannot prevent k shareholders from colluding with each other to reconstruct the secret key a

Later developments

⋆ Threshold signature scheme for RSA

- [FD92] Y. Frankel and Y. Desmedt. ‘‘Parallel reliable threshold multisignature.’’ TR-92-04-02, Dept. of EE and CS, Univ. of Wisconsin. April 1992.

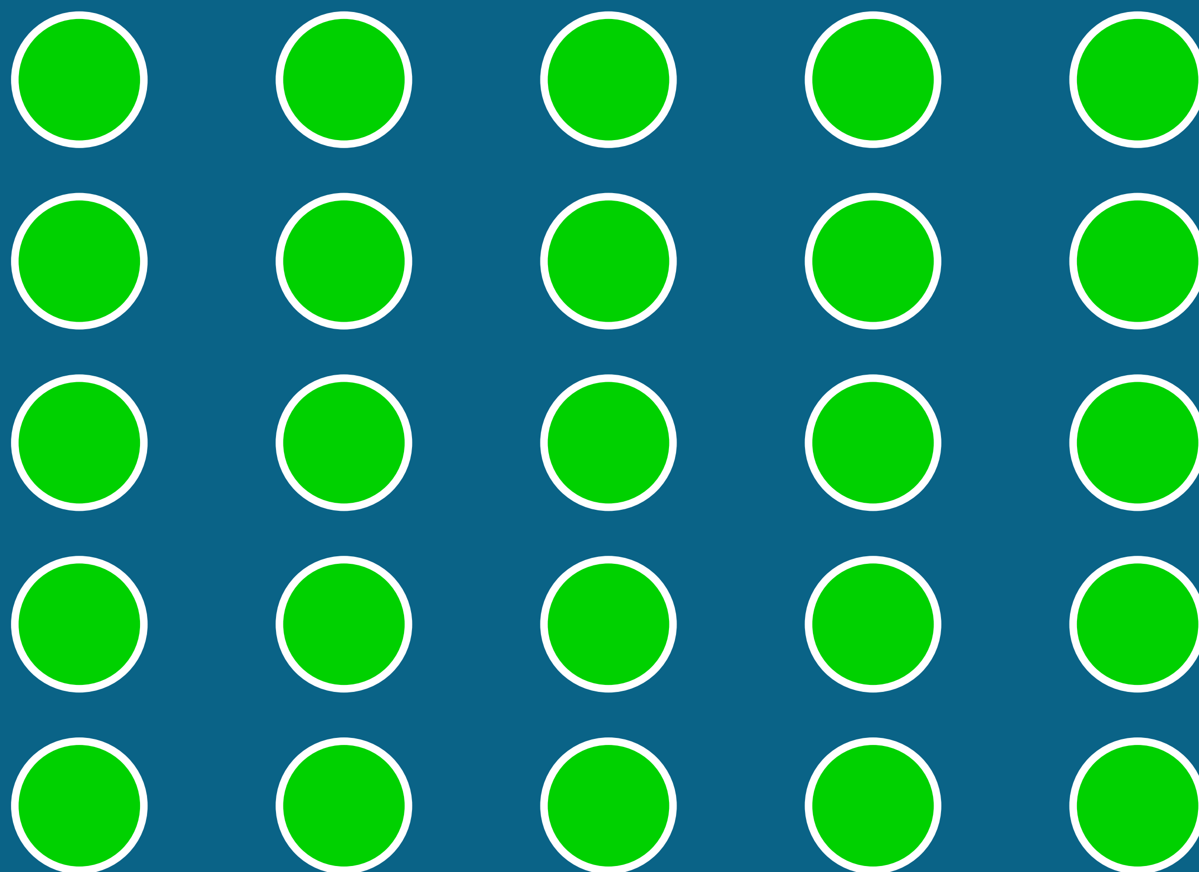
⋆ Threshold signature scheme for DSS

- [GJKR96] R. Gennaro, S. Jarecki, H. Krawczyk and T. Rabin. ‘‘Robust Threshold DSS Signatures.’’ Advances in Cryptology --- Eurocrypt ’96.

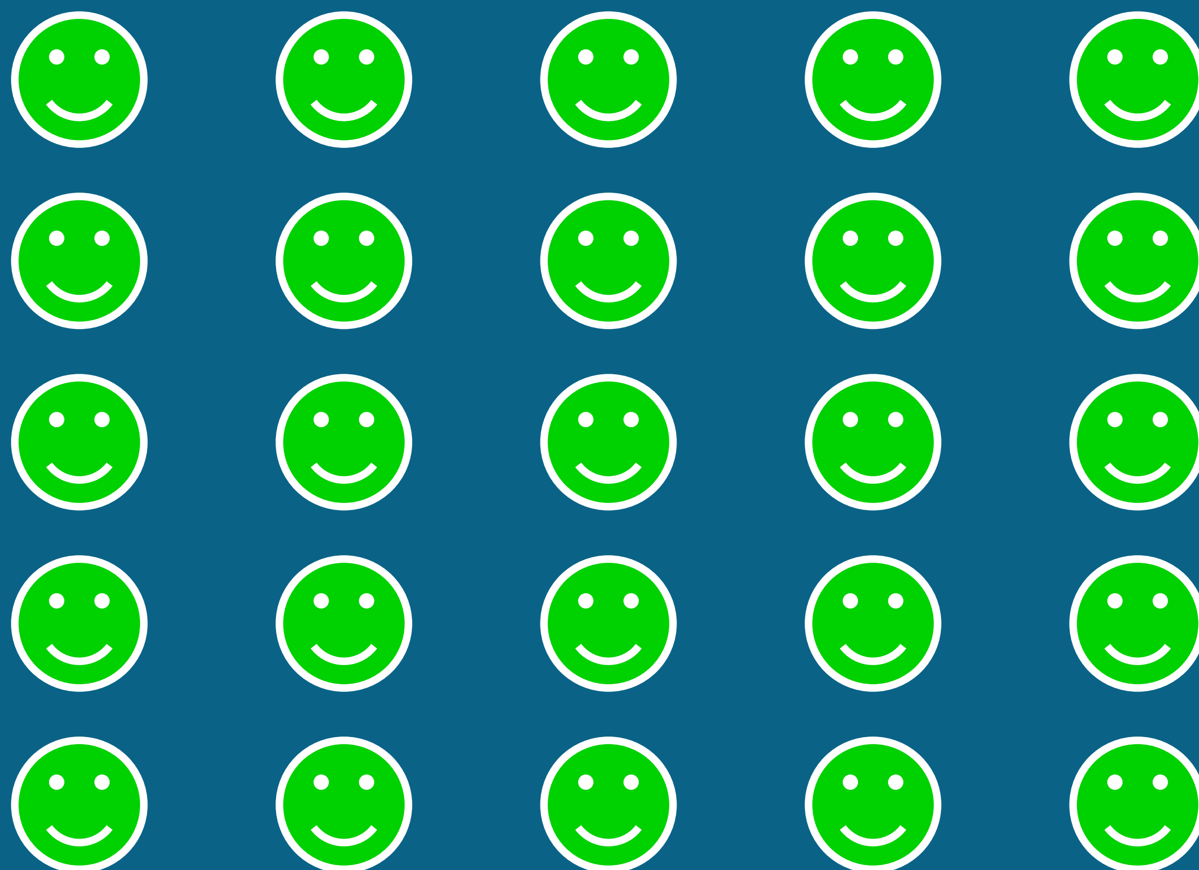
⋆ Improvement on RSA methods

- [Rab98] T. Rabin. ‘‘A Simplified Approach to Threshold and Proactive RSA.’’ Crypto ’98.

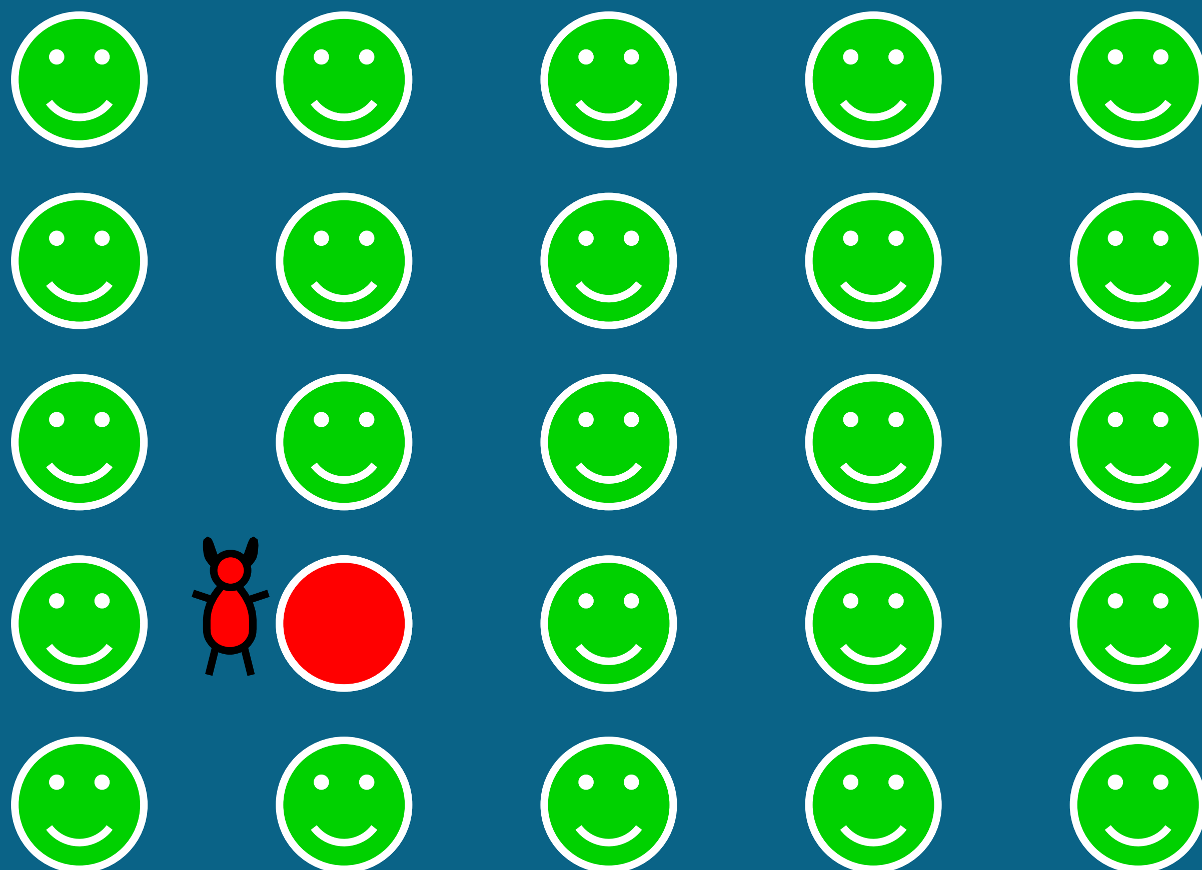
Secret sharing: Part II



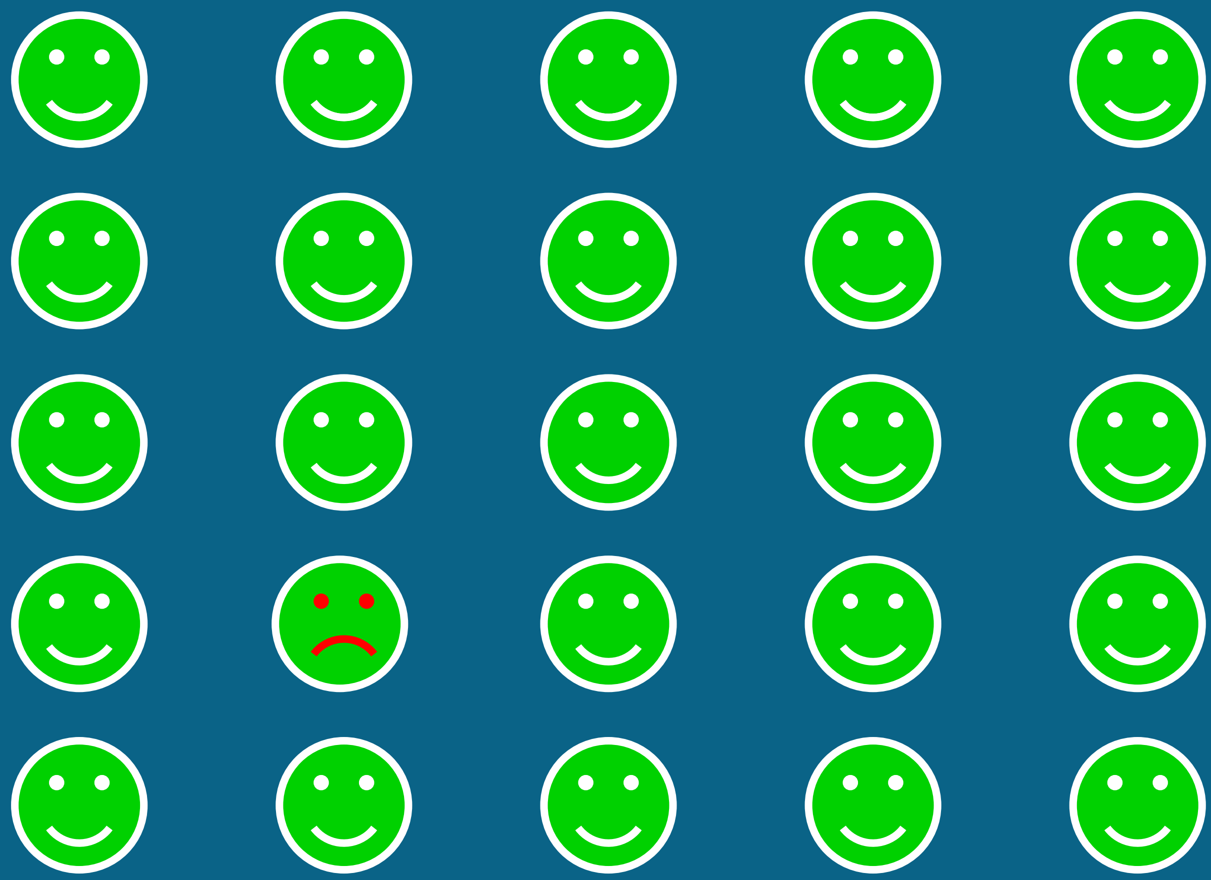
Secret sharing: Part II



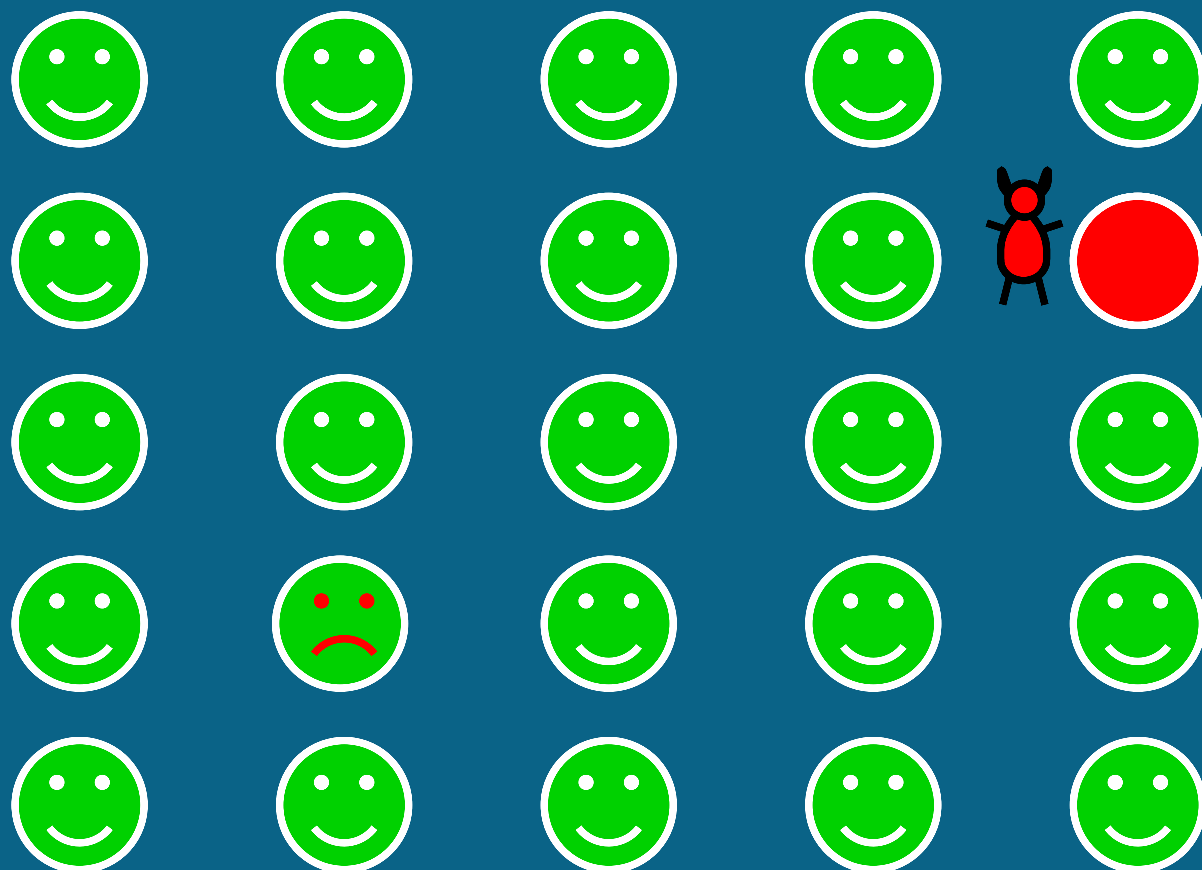
Secret sharing: Part II



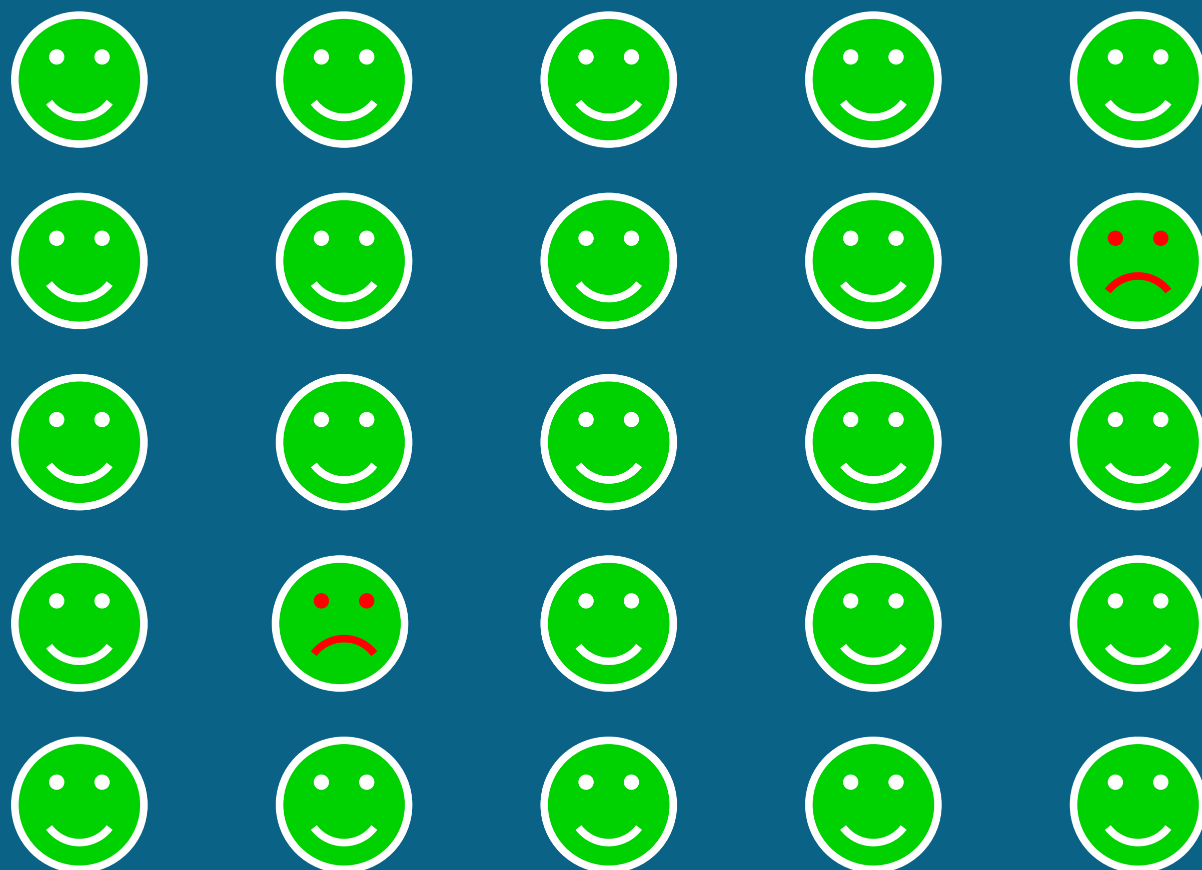
Secret sharing: Part II



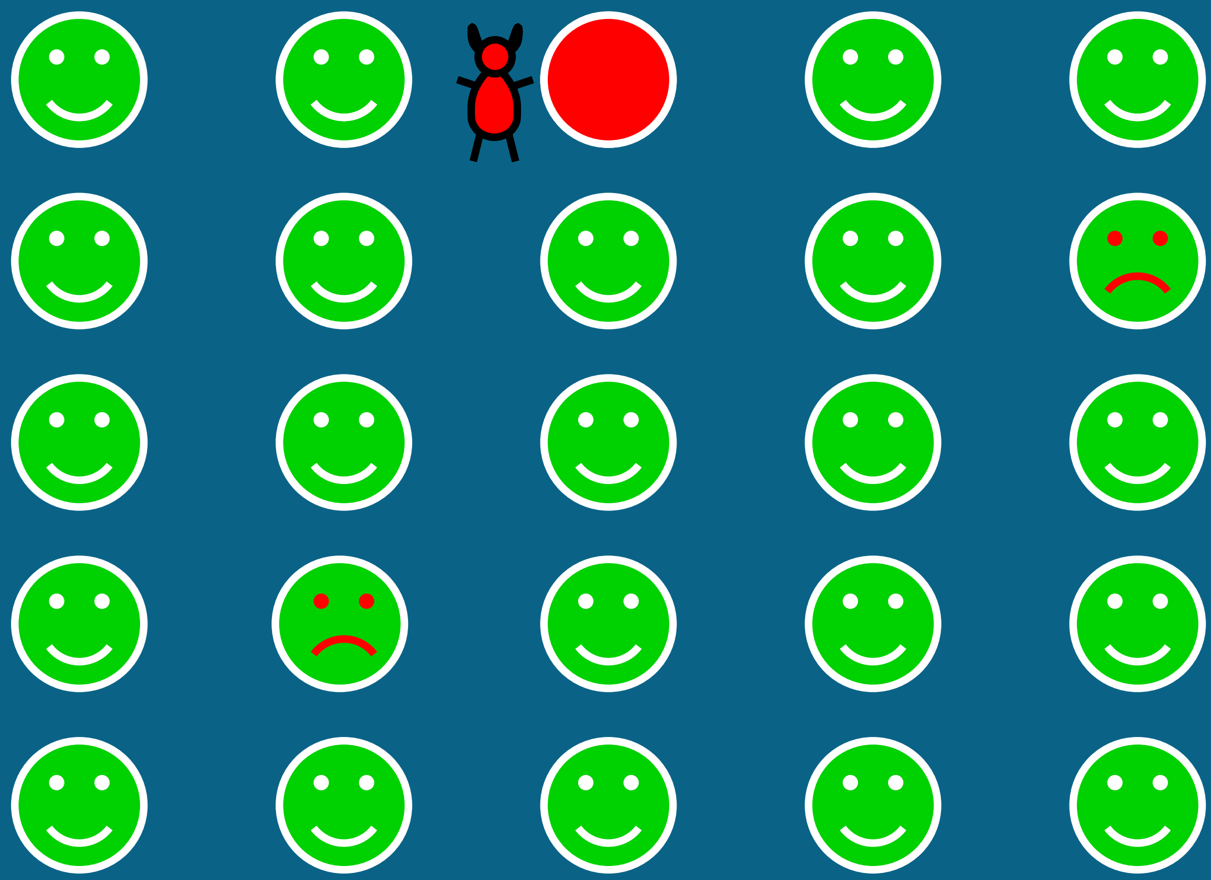
Secret sharing: Part II



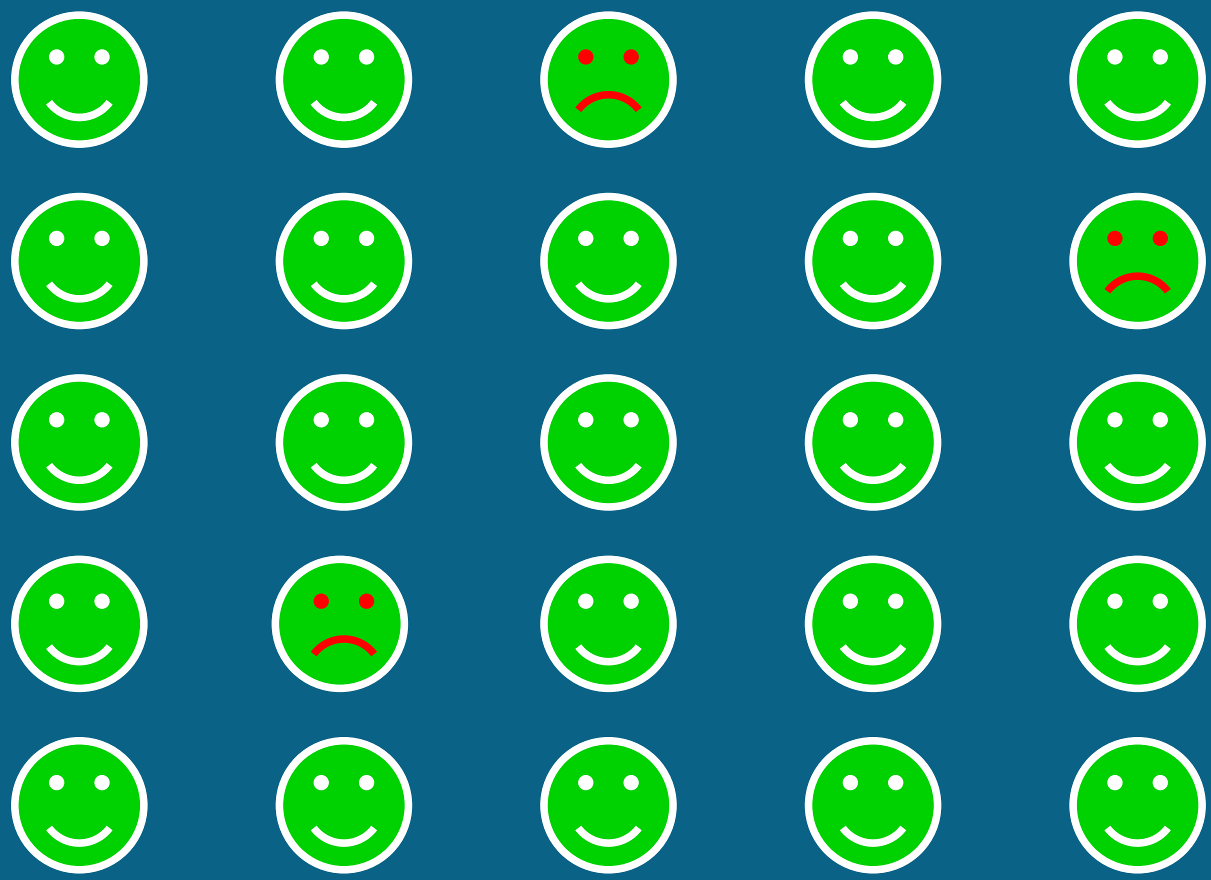
Secret sharing: Part II



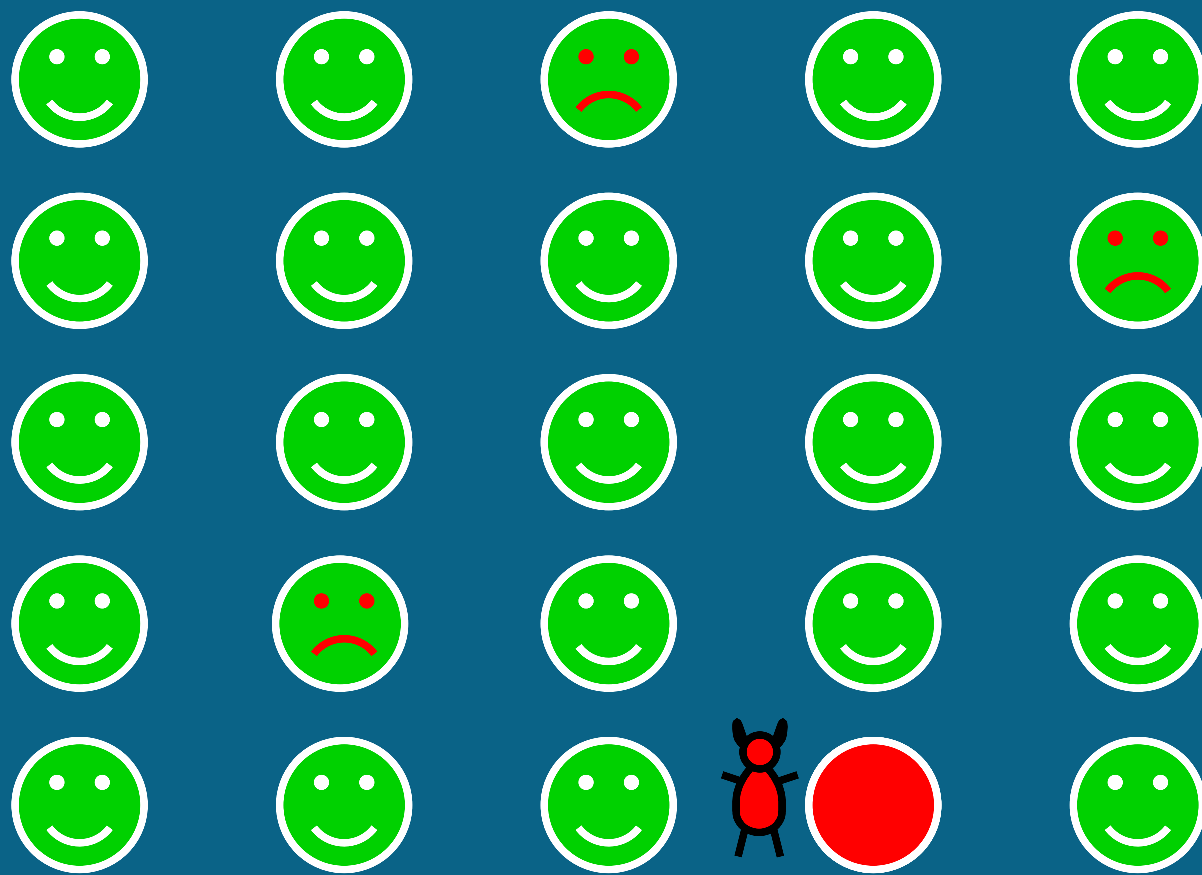
Secret sharing: Part II



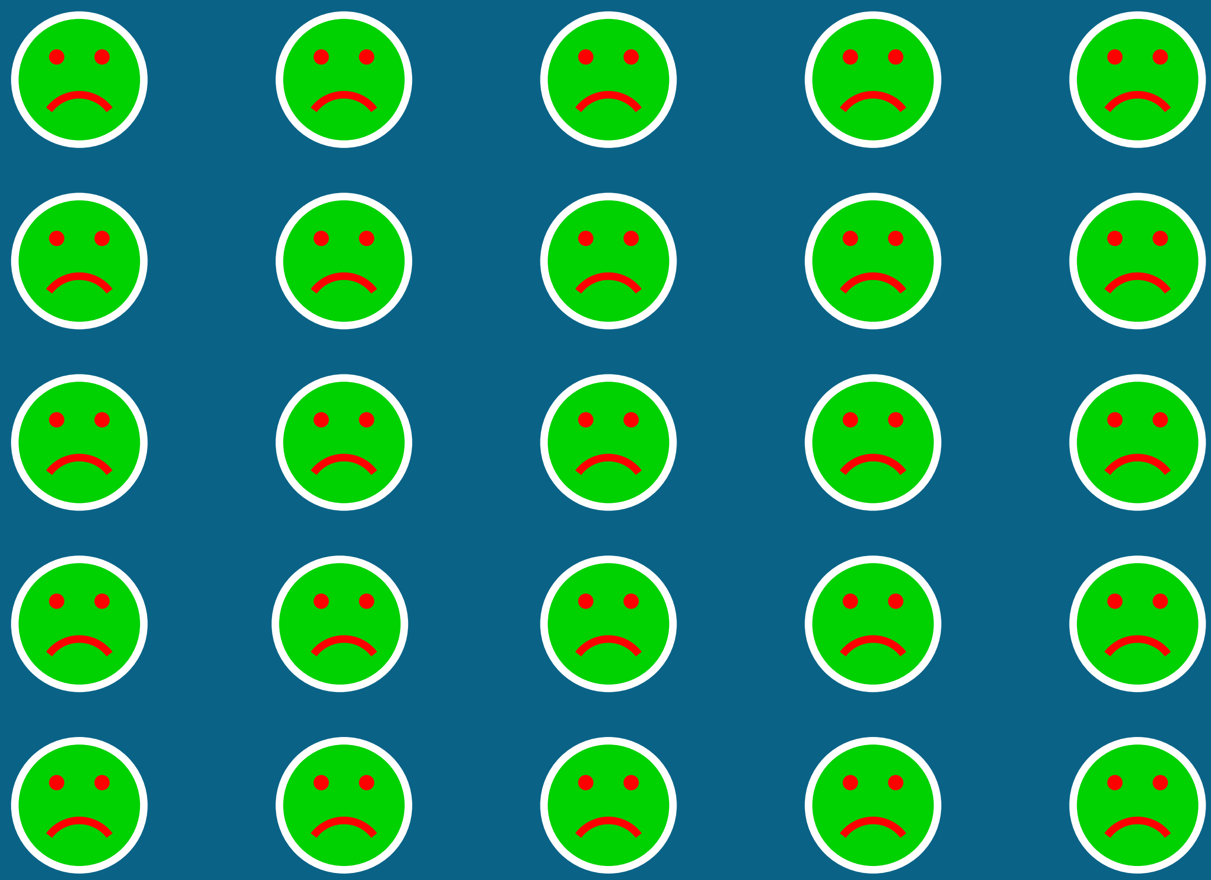
Secret sharing: Part II



Secret sharing: Part II



Secret sharing: Part II



Secret sharing: Part II

What to do?

- ⋆ Could throw away secret and start over with new one
  - Unacceptable for many applications
- ⋆ Could reconstruct secret and distribute new shares
  - This is a security hazard

Secret sharing: Part II

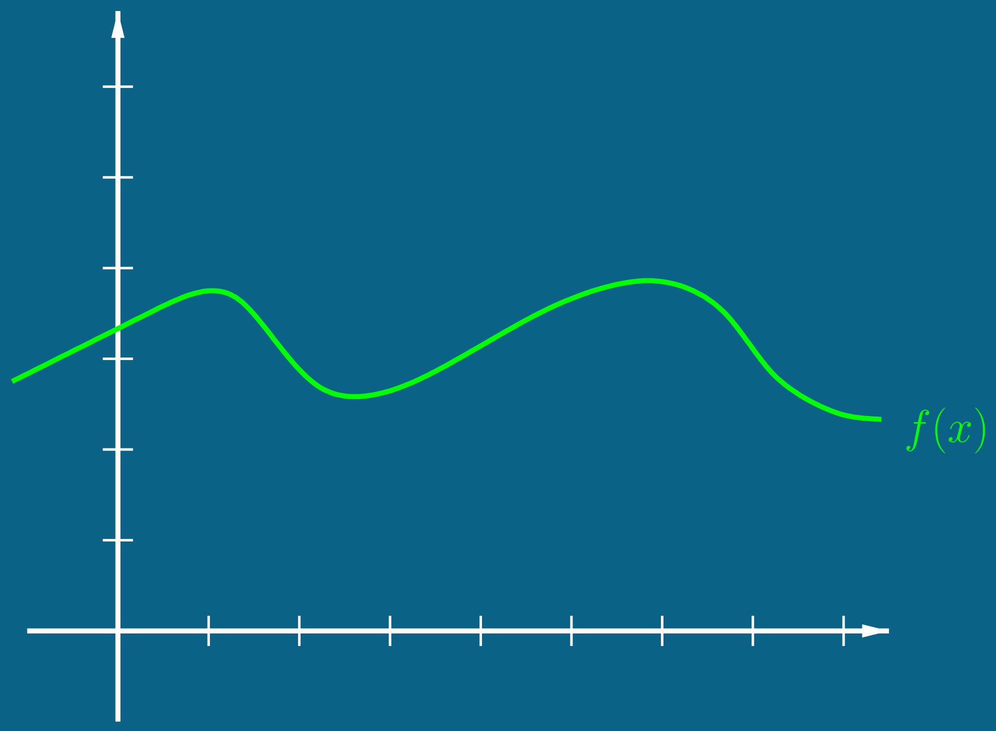
What to do?

- ⋆ Could throw away secret and start over with new one
  - Unacceptable for many applications
- ⋆ Could reconstruct secret and distribute new shares
  - This is a security hazard

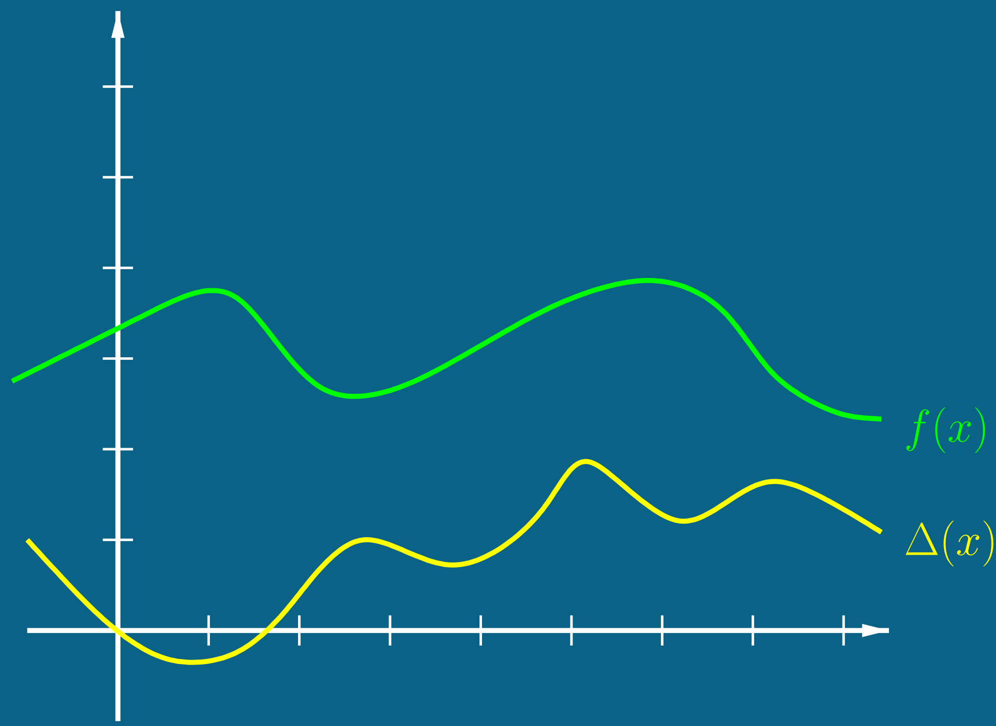
Answer: Proactive Secret Sharing

- ⋆ Get new shares for same secret *without* reconstructing secret

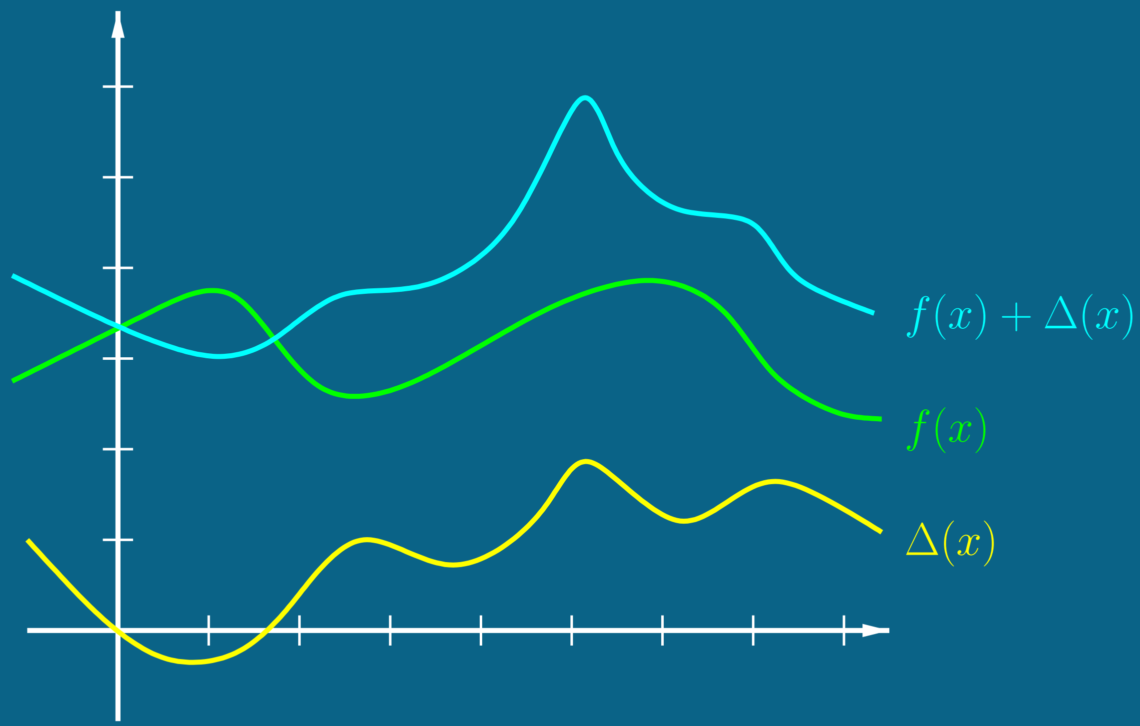
$(+,+)$-homomorphism property



$(+,+)$-homomorphism property



$(+,+)$-homomorphism property



$$f'(0) = f(0) + \Delta(0) = f(0)$$

Proactive secret sharing -- Take 1

Steps to refresh shares:

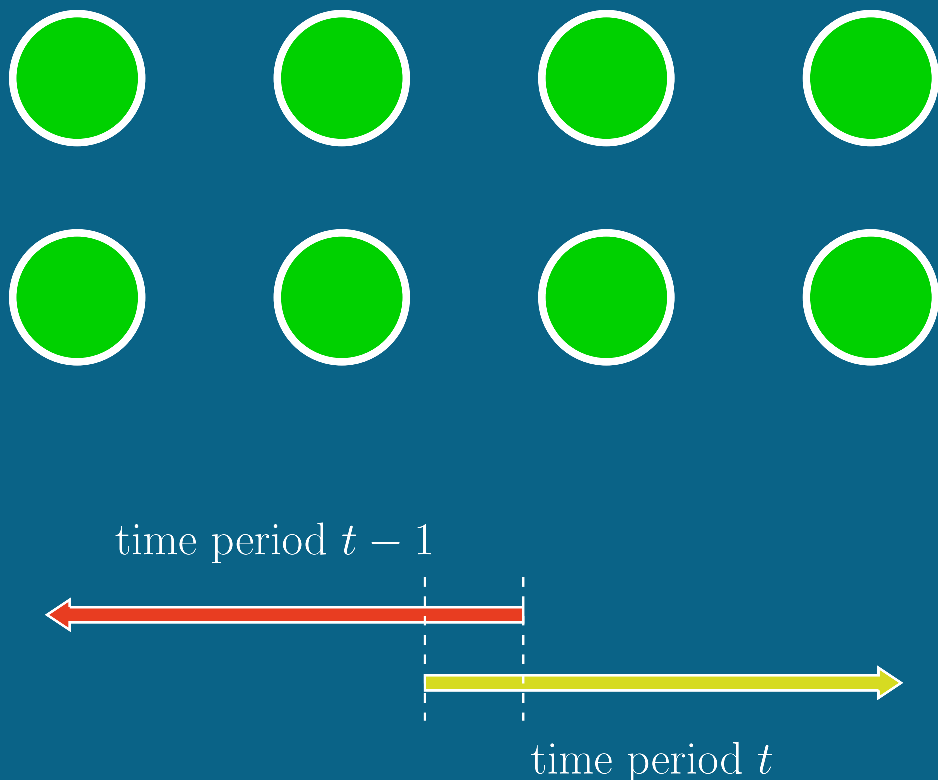
- ⋆ Designated node creates random polynomial

$$\Delta(x) = \delta_1 x + \cdots + \delta_{k-1} x^{k-1}$$

- ⋆ Distributes shares $\Delta(1), \ldots, \Delta(n)$
- ⋆ Each node makes new share $f'(i) = f(i) + \Delta(i)$
- ⋆ Destroy $f(i)$, $\Delta(i)$

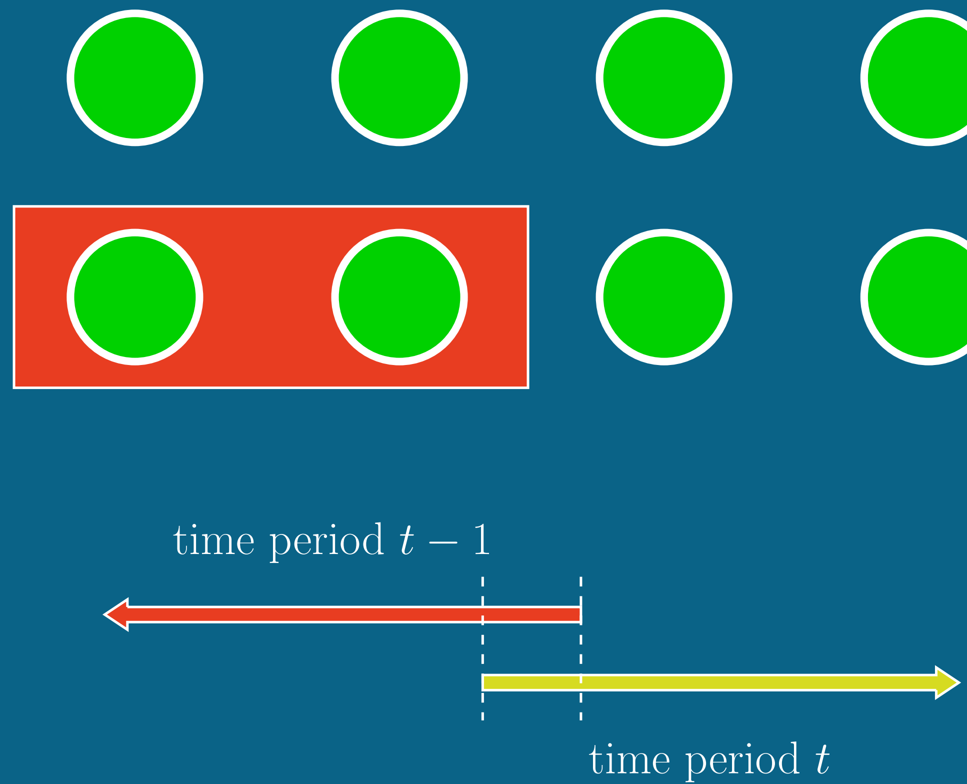
Proactive secret sharing -- Take 1

A scenario involving a $(5, 8)$-threshold scheme:



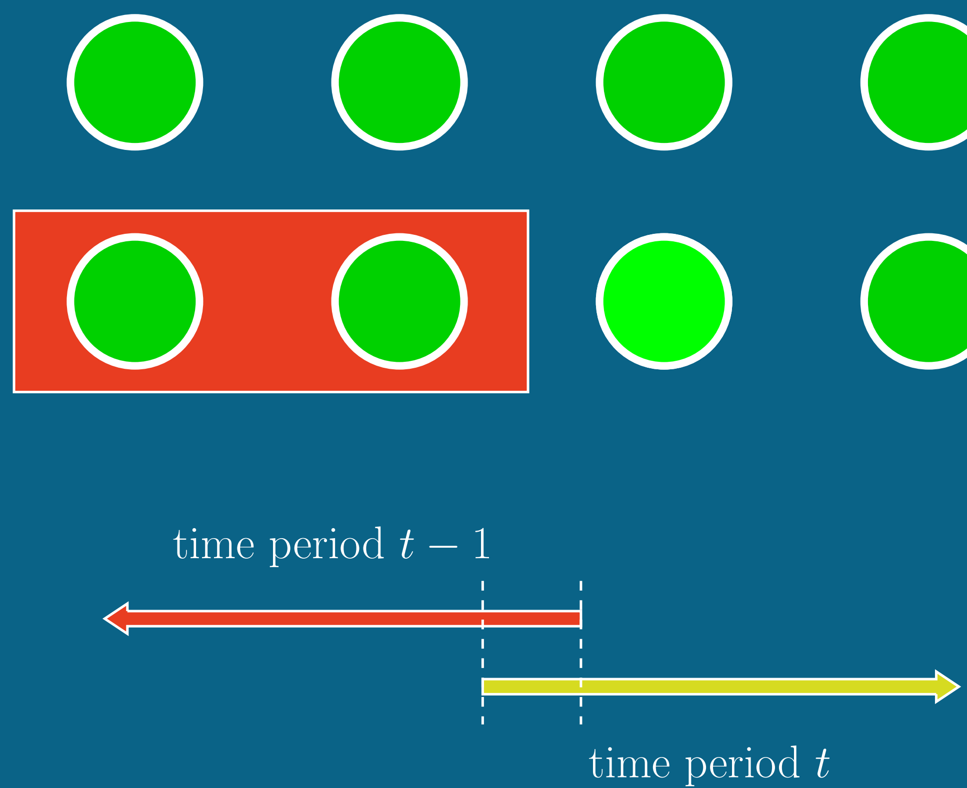
Proactive secret sharing -- Take 1

A scenario involving a $(5, 8)$-threshold scheme:



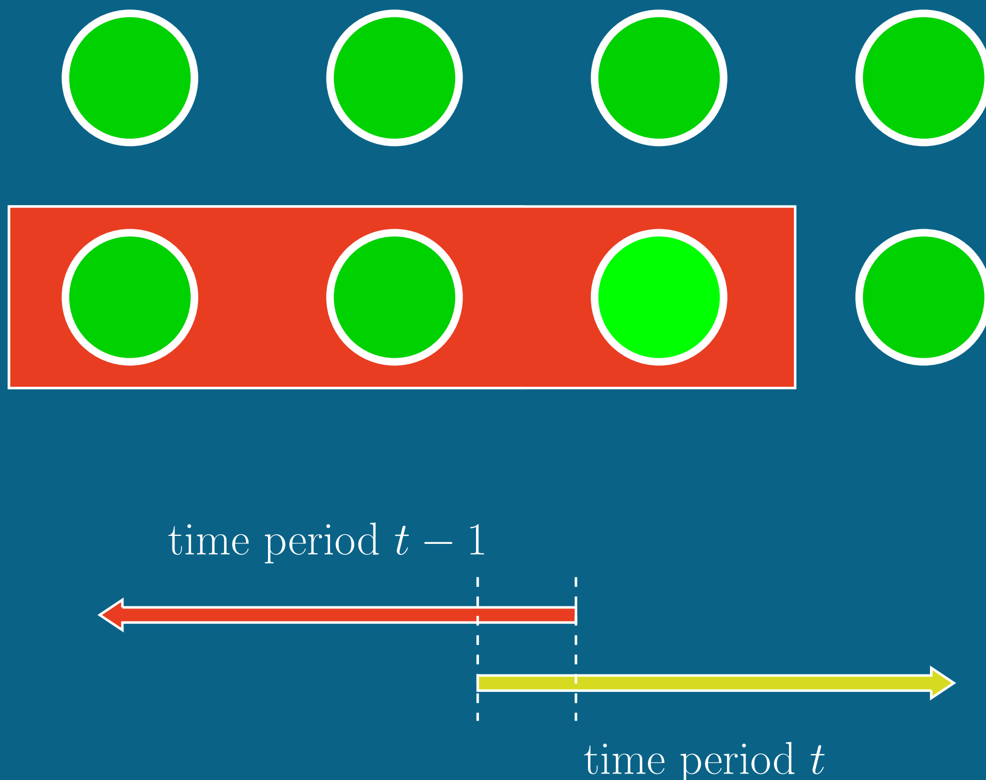
Proactive secret sharing -- Take 1

A scenario involving a $(5, 8)$-threshold scheme:



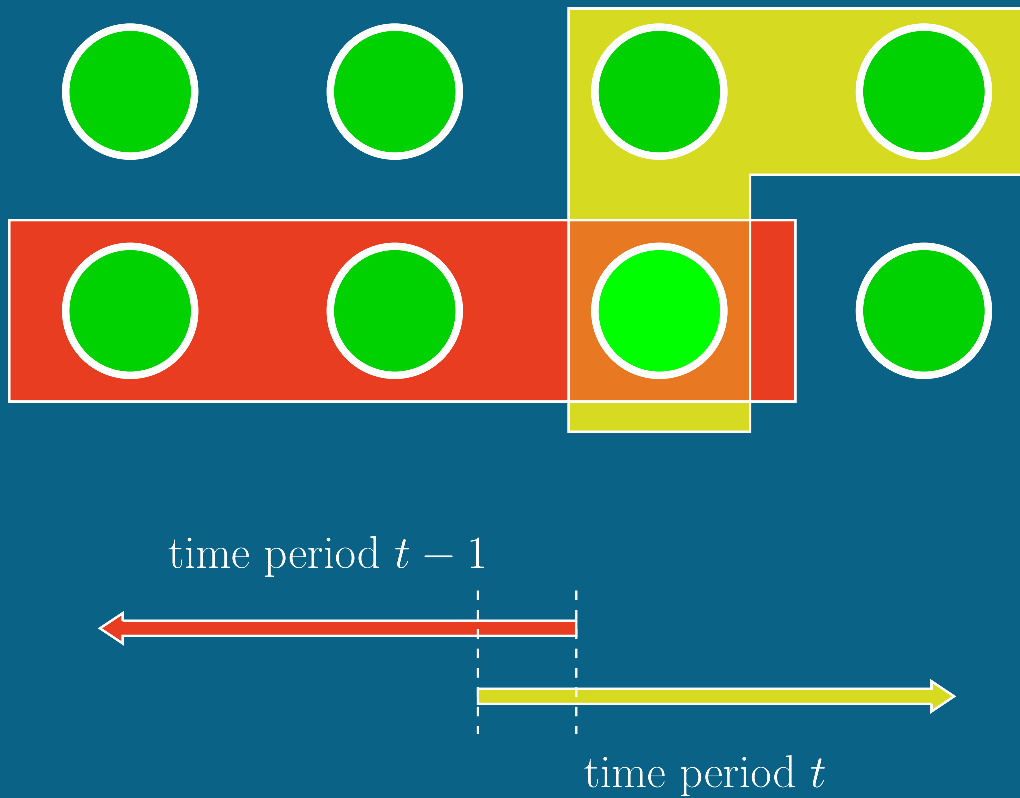
Proactive secret sharing -- Take 1

A scenario involving a $(5, 8)$-threshold scheme:



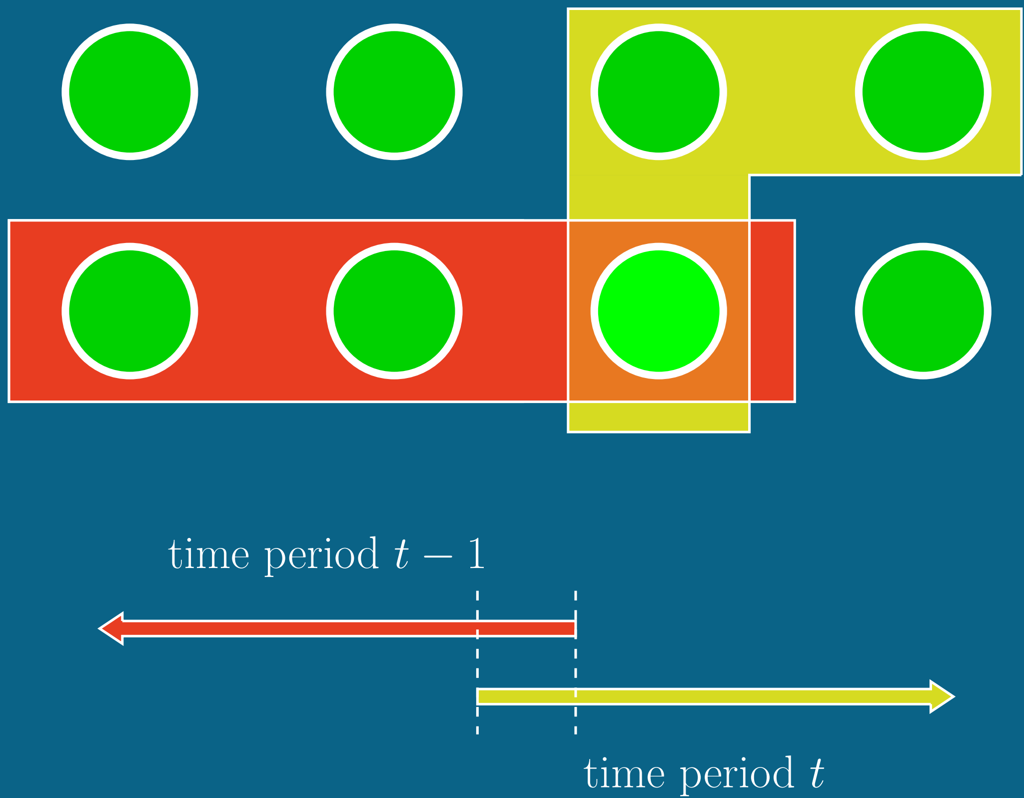
Proactive secret sharing -- Take 1

A scenario involving a $(5, 8)$-threshold scheme:



Proactive secret sharing -- Take 1

A scenario involving a $(5, 8)$-threshold scheme:



Less than k nodes compromised per time period, but secret revealed.

Proactive secret sharing -- Take 2

Solution: replicate!

- ⋆ Have $k-1$ nodes generate random polynomials $\Delta_1, \ldots, \Delta_{k-1}$ of degree $k-1$ passing through the origin

- ⋆ Each distributes delta shares $\Delta_j(1), \ldots, \Delta_j(n)$ privately to all nodes

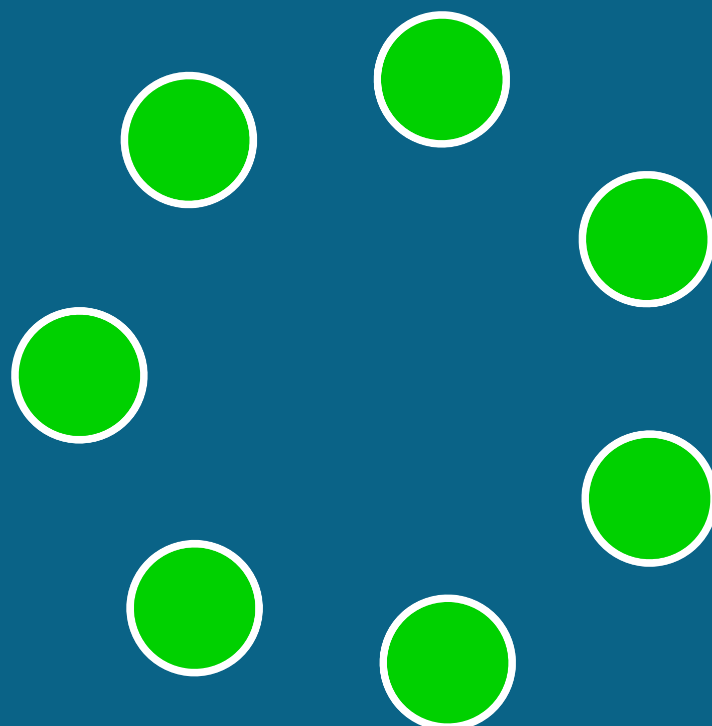
- ⋆ Each recipient i creates new share

$$f'(i) = f(i) + \sum_{j=1}^{k-1} \Delta_j(i)$$

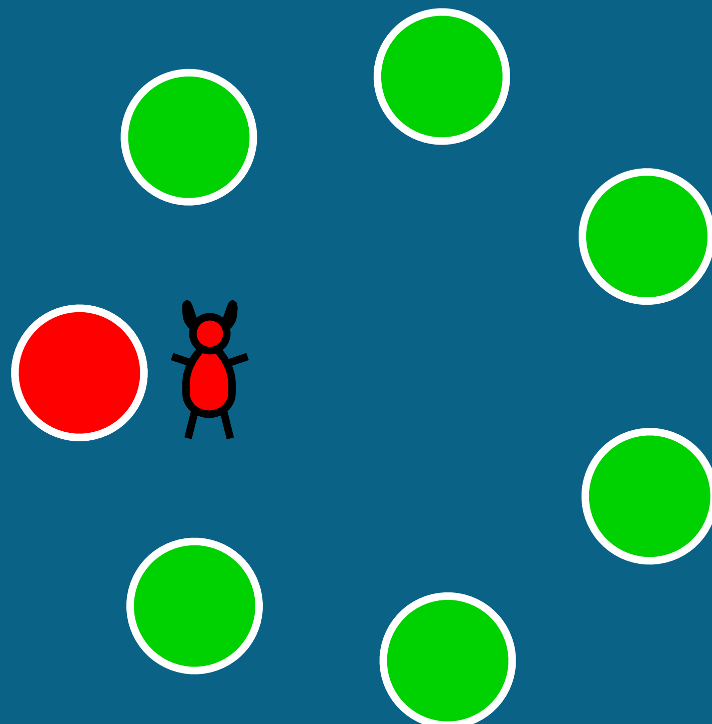
Proactive secret sharing -- Take 2

Result: attacker must now compromise k nodes per time period in order to learn secret.

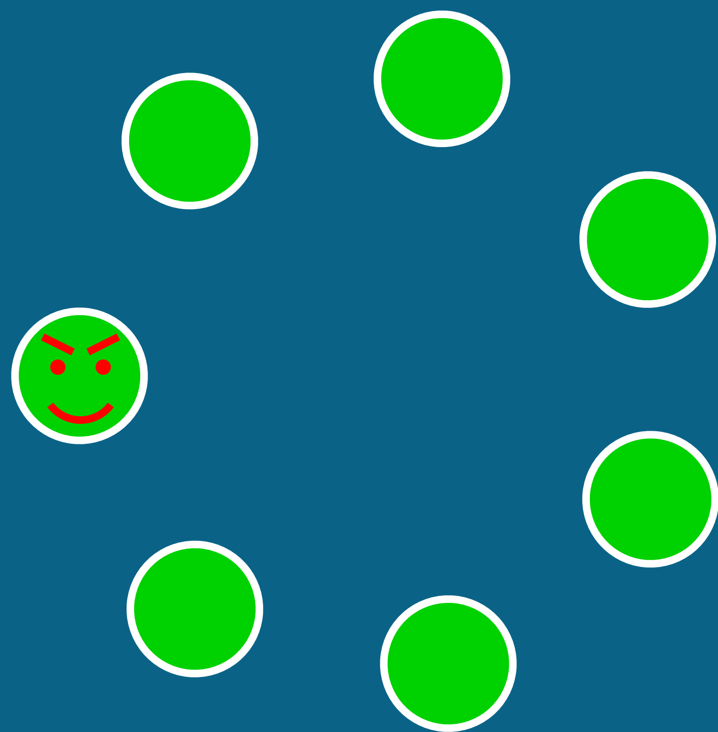
Secret sharing: Part III



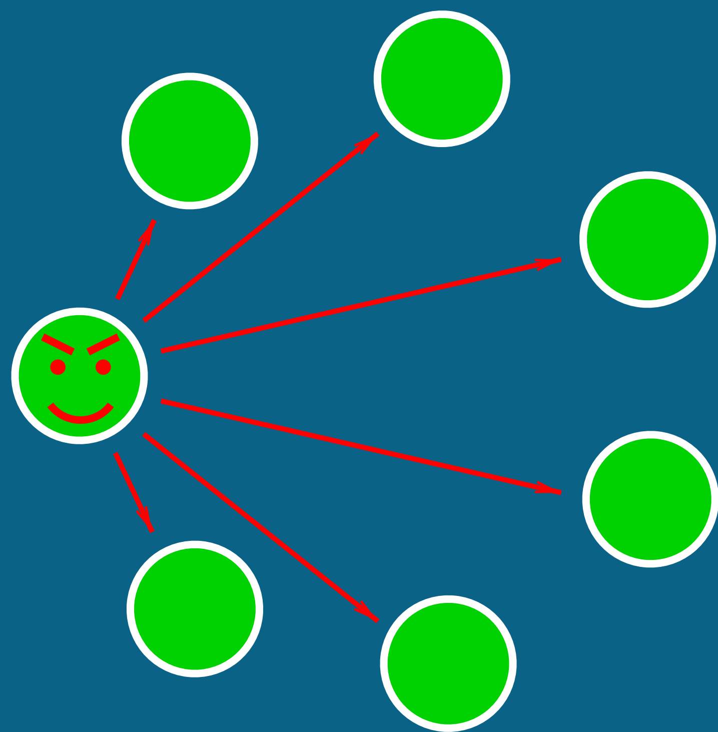
Secret sharing: Part III



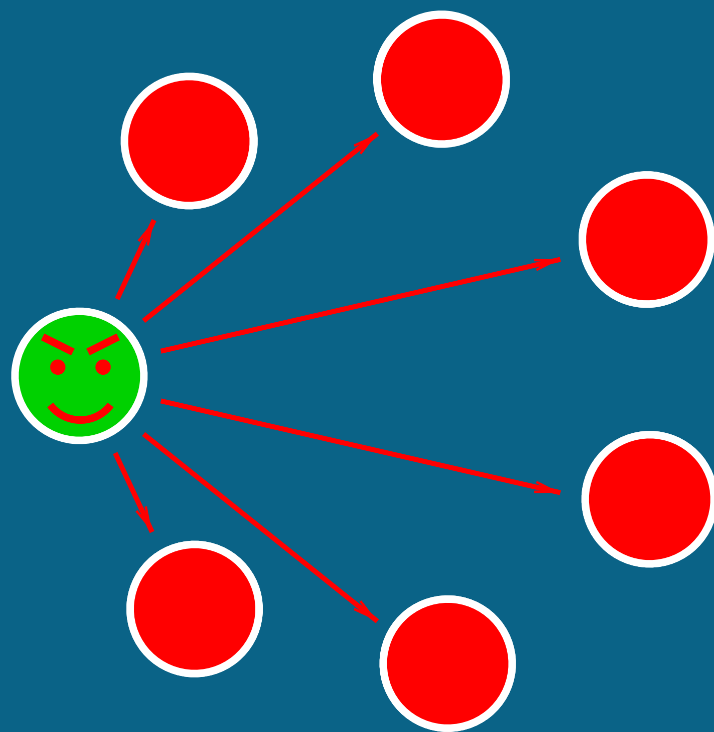
Secret sharing: Part III



Secret sharing: Part III



Secret sharing: Part III



Verifiable secret sharing

[Fel87] P. Feldman. "A practical scheme for non-interactive verifiable secret sharing." *Proceedings of the 28th Annual Symposium on the Foundations of Computer Science*:427–437. IEEE, October 12–14, 1987.

- ⋆ Provides a way to check shares for validity without reconstructing secret and without disclosing (too much) information

Steps for Feldman-VSS protocol

Either by consensus or predetermination:

- ⋆ Choose large primes p and q, $p = mq + 1$
- ⋆ Choose element g of order q in $\mathbb{Z}_p$ (i.e. $g^q \equiv 1 \bmod p$)
    - $\mathbb{Z}_p$ is used for verification
    - $\mathbb{Z}_q$ is actual secret sharing domain

Steps for Feldman-VSS protocol

Dealer:

- ⋆ Creates polynomial $f(x) = c_0 + c_1 x + \cdots + c_{k-1} x^{k-1}$ in $\mathbb{Z}_q$
  - Secret is c_0
- ⋆ Distributes shares $(x_1, y_1), \ldots, (x_n, y_n)$ privately
  - Note in $\mathbb{Z}$, $f(x_i) = rq + y_i$ for some r
- ⋆ Broadcasts $g^{c_0}, g^{c_1}, \ldots, g^{c_{k-1}}$

Steps for Feldman-VSS protocol

Each shareholder i:

- ⋆ Calculates $g^{y_i} \pmod p$ and verifies

$$
\begin{aligned}
(g^{c_0})(g^{c_1})^{x_i}(g^{c_2})^{{x_i}^2}\cdots(g^{c_{k-1}})^{{x_i}^{k-1}} &\equiv g^{c_0+c_1x_i+\cdots+c_{k-1}{x_i}^{k-1}} \\
&\equiv g^{f(x_i)} \\
&\equiv g^{rq+y_i} \\
&\equiv (g^q)^r g^{y_i} \\
&\equiv g^{y_i} \pmod p
\end{aligned}
$$

This holds iff the shares are valid and consistent with the g^c's.

Verifiable secret sharing

Possible drawback of $Feldman\text{-}VSS$ scheme:

⋆ Makes $g^{f(0)}$ public

- While entire $f(0)$ is hard to determine, the lowest-order bits are easily accessible—partial information disclosure

Remedies:

⋆ Encode actual secret into higher-order bits of an envelope

⋆ Use $Pedersen\text{-}VSS$ scheme (information-theoretically secure)

Proactive secret sharing

[HJKY95] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung. "Proactive secret sharing, or: How to cope with perpetual leakage." *Advances in Cryptology — Crypto '95*.

Also add robustness:

- ⋆ Use $n > 2(k-1)$ nodes
- ⋆ Have all n nodes distribute Δ-shares (instead of just $k-1$)
- ⋆ Accusation protocol
- ⋆ Share recovery scheme (to deal lost or corrupted nodes back in)