# RFID Privacy Without Killing

Ravi Pappu
Co-Founder
ThingMagic Inc.

Joint work with Ari Juels (RSA, CUSP) and Bryan Parno (CMU)

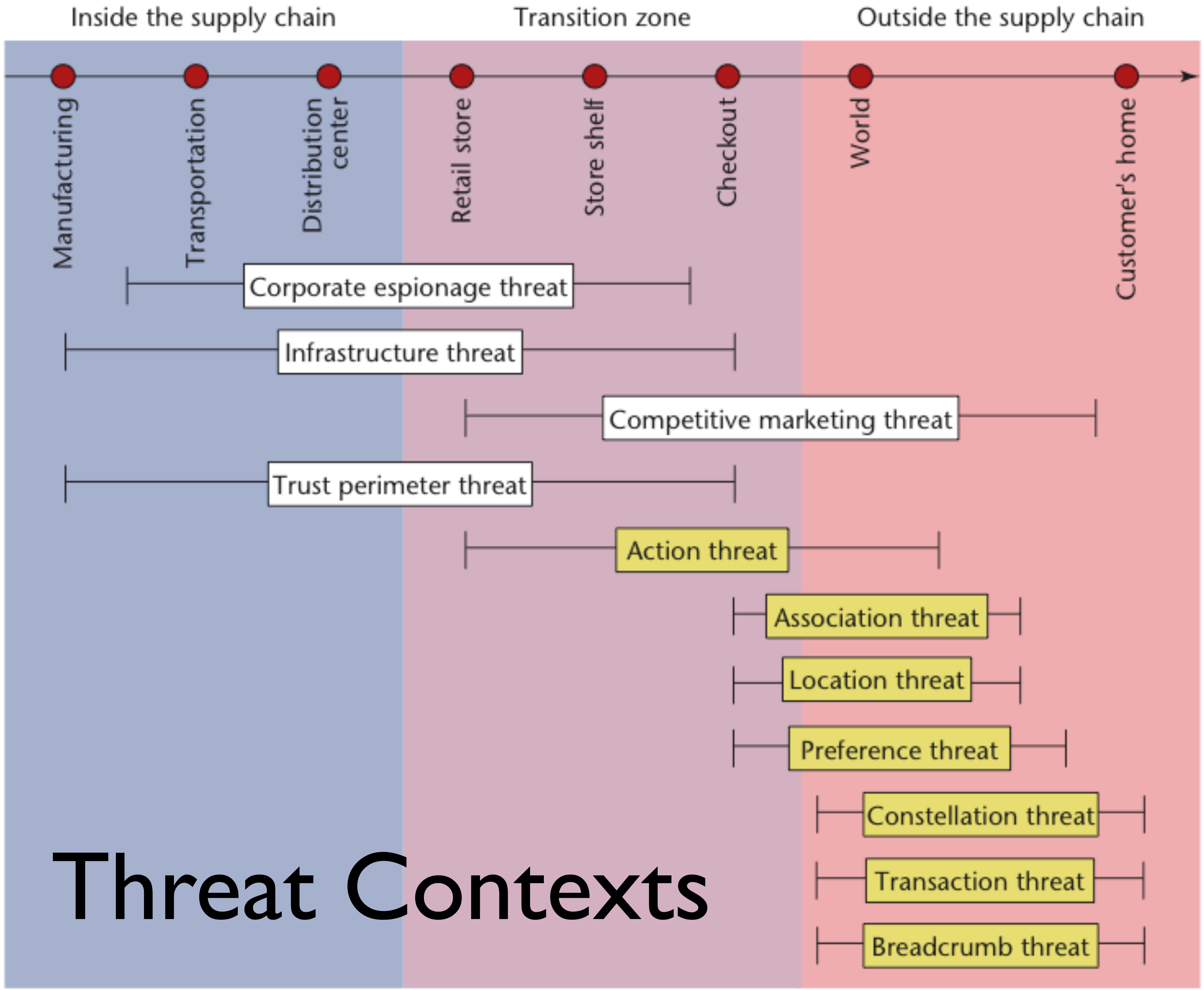
# Key messages

- Tiny Secret Sharing (TSS) enables consumer privacy *now*
    - No heroic measures required
    - No dependence on any particular standard ⇒ fully standards compliant
- Consumer privacy is achieved by exploiting the natural movement of tagged products through the supply chain
    - Privacy through dispersion and history erasure

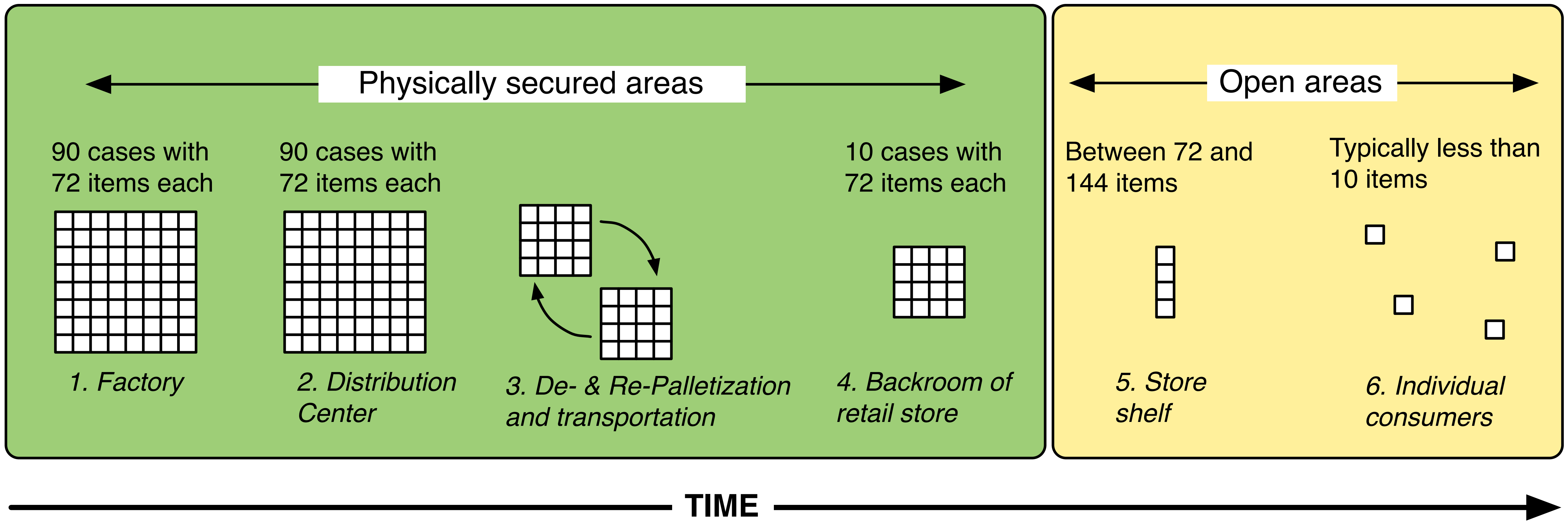
# Summary

- Tiny Secret Sharing (TSS) enables RFID privacy without killing
- Encryption key length is a free-variable - security can be tailored.
- TSS is protocol-independent, and completely local - no network required.
- It scales to item-level tagging
- The only resources needed are tag memory and some computing power at reader
- TSS allows use of standard cryptographic mechanisms for encryption, hashing
- TSS fits naturally in many supply-chain scenarios where we have less than 100% reads and where stray tags or counterfeits are present.
- TSS solves key-management problem - enables privacy and write/lock PIN distribution.

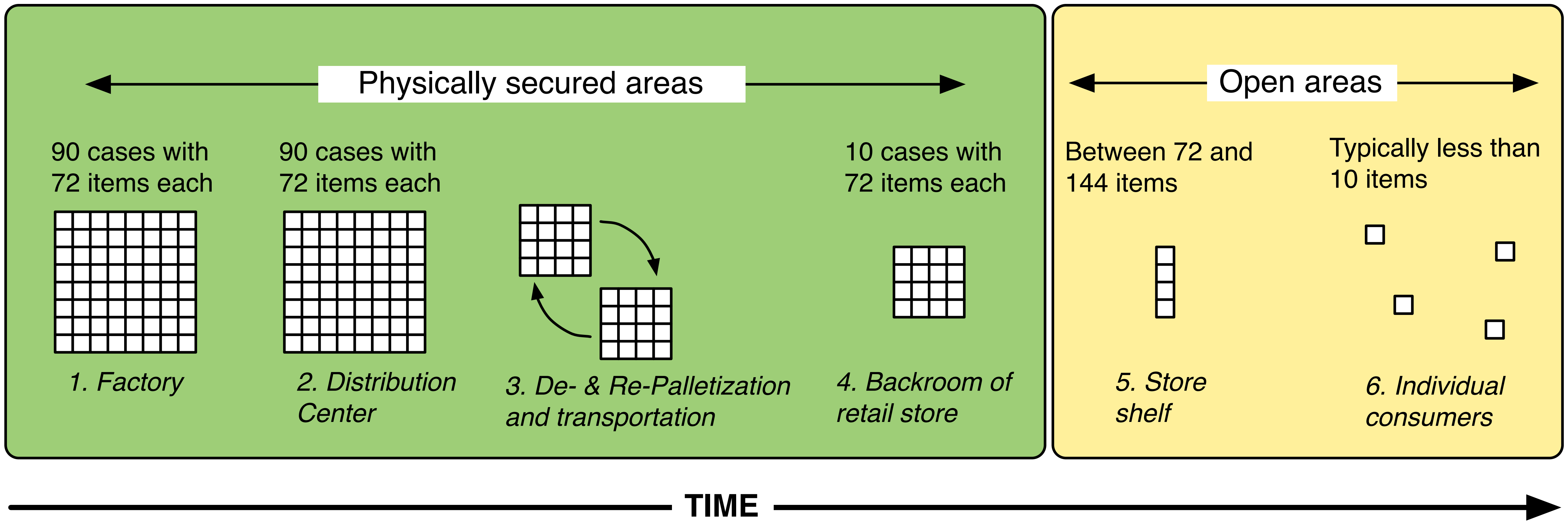
# Threat Contexts



# Object Hierarchies



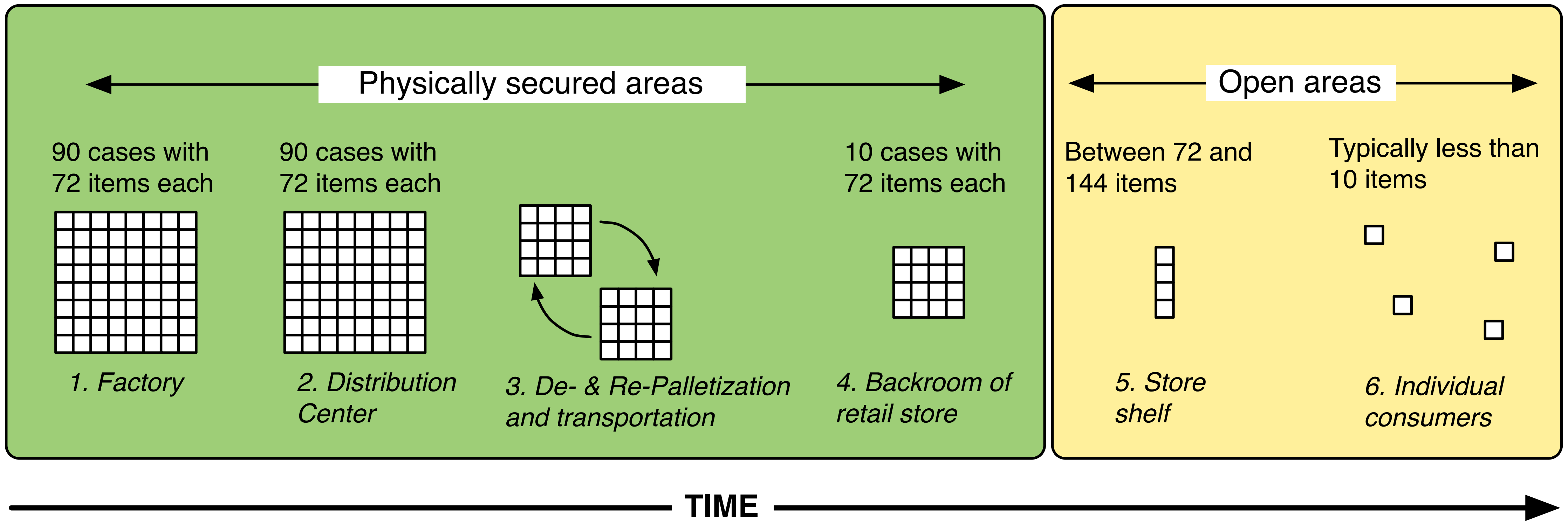
# Object Hierarchies



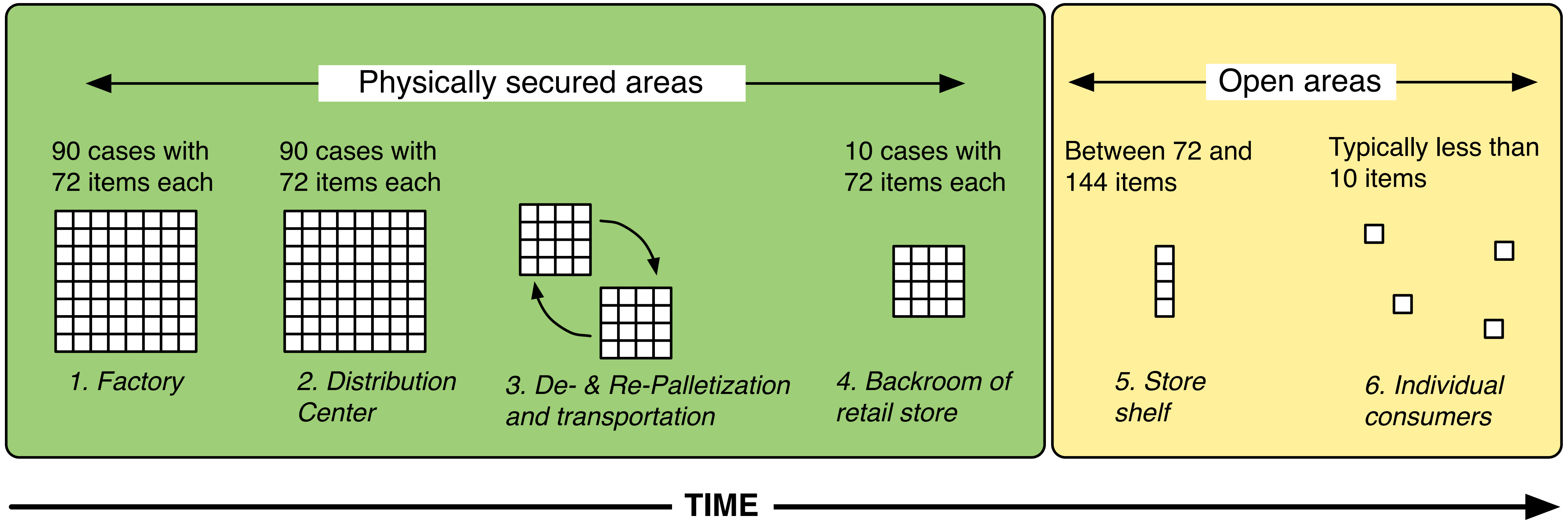

| | | | | |
|---|---|---|---|---|
| Razor blades | 6571 | 730 | 144 | 5 |
| DVDs | 5040 | 2520 | 400 | 24 |
| Pharma. | 7200 | 1920 | 150 | 6 |

Source: EPCGlobal Item Level JRG

# Object Hierarchies

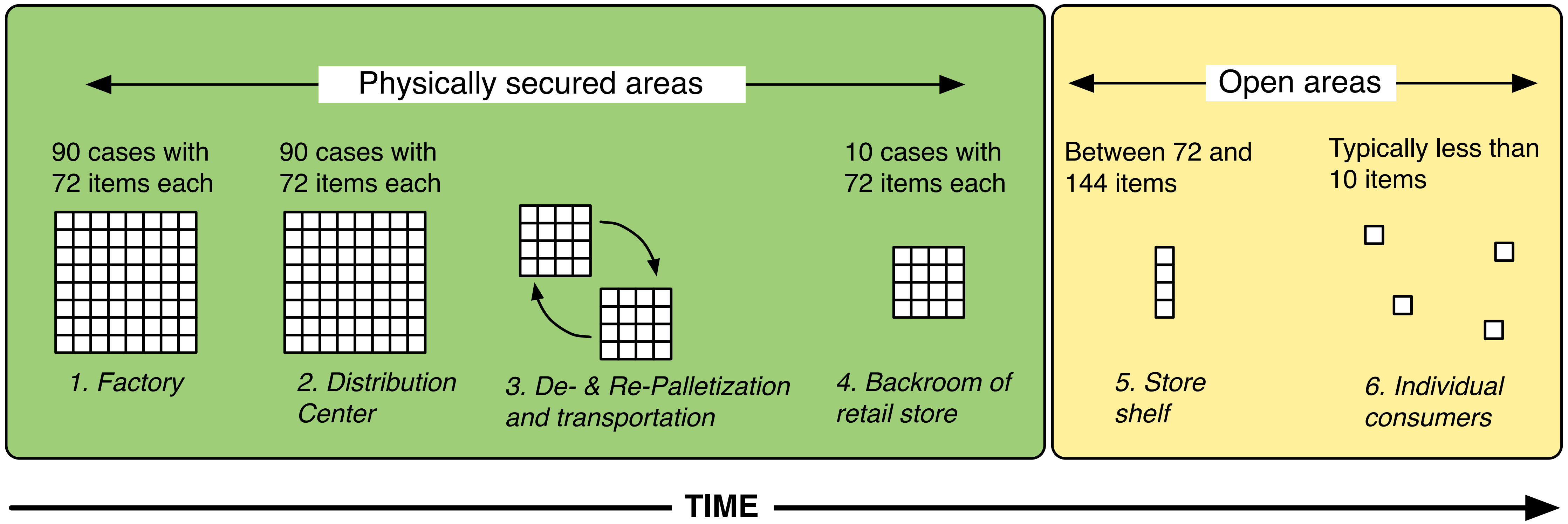


# Object Hierarchies




Tags start out in large collections which become smaller over time.

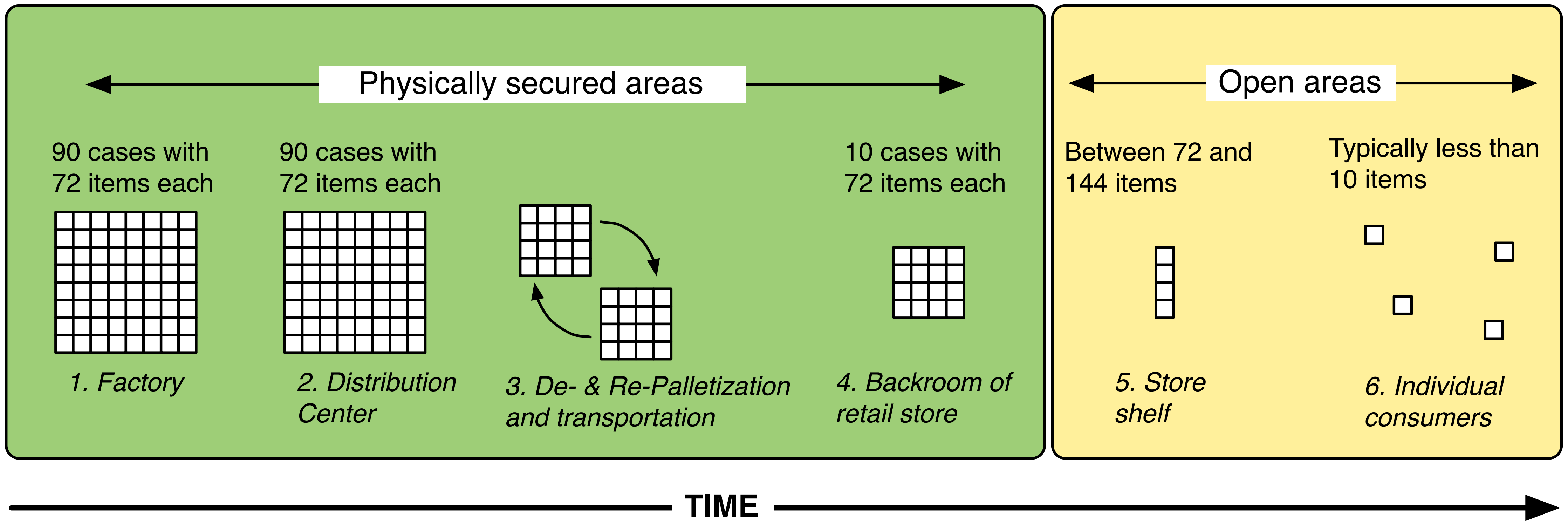
# Object Hierarchies




Tags start out in large collections which become smaller over time.

Larger collections of tags are typically located in secure areas.

# Object Hierarchies




Tags start out in large collections which become smaller over time.

Larger collections of tags are typically located in secure areas.

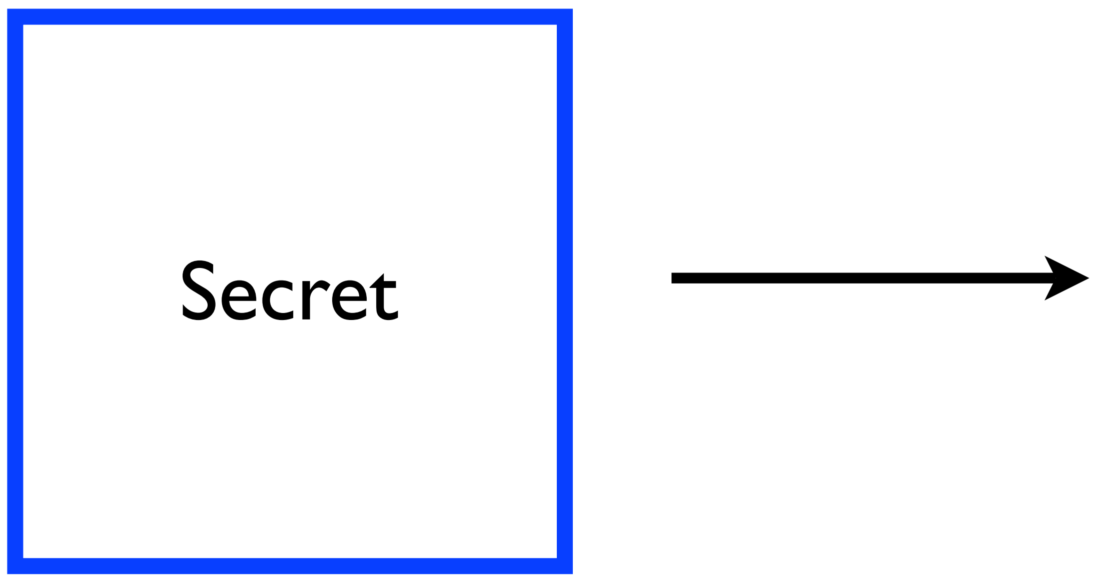
Shared context from earlier times is not available at later times to adversary.

# Key question

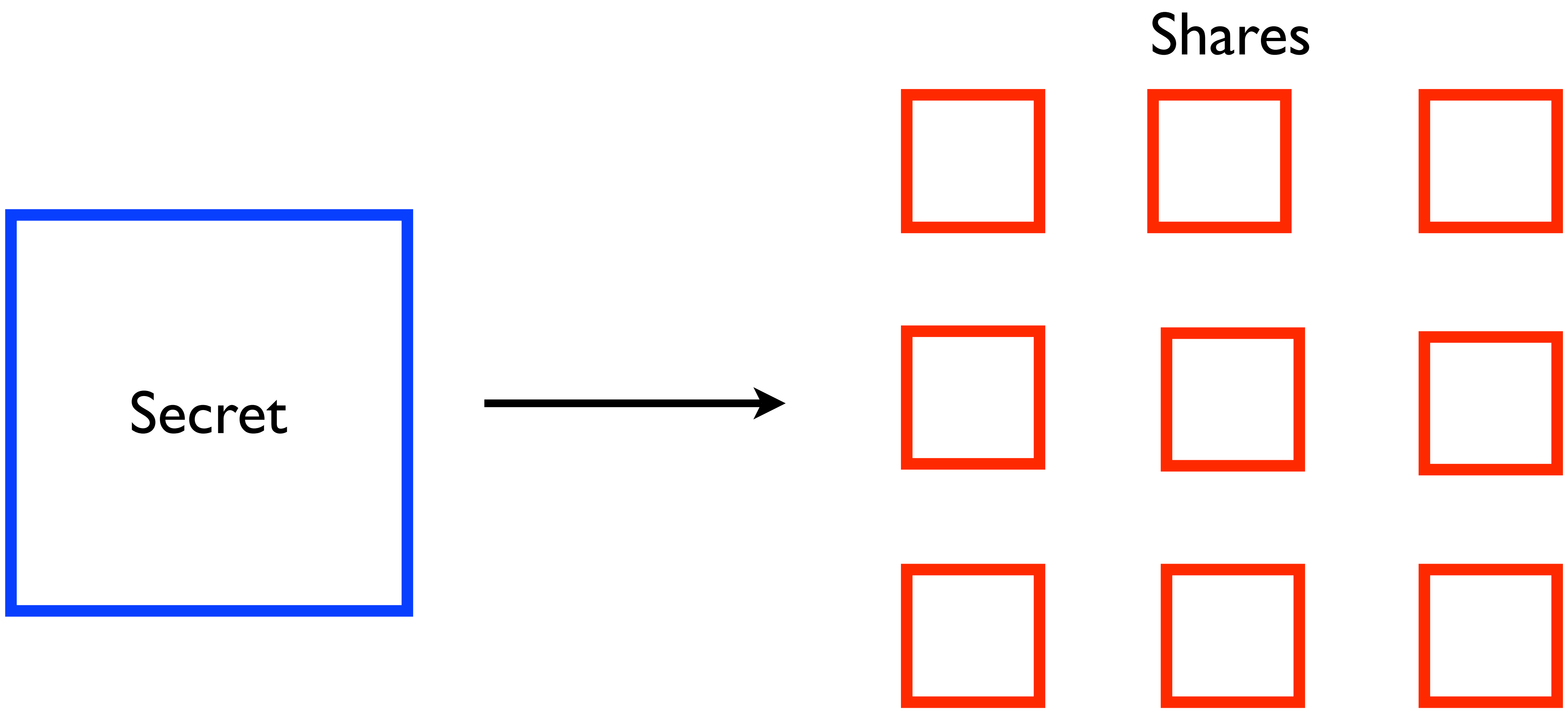
- Can we exploit the three observations to provide strong privacy in RFID-enables supply chains?
- Yes! In order to do so, we turn to a cryptographic method called secret sharing.

# Secret Sharing

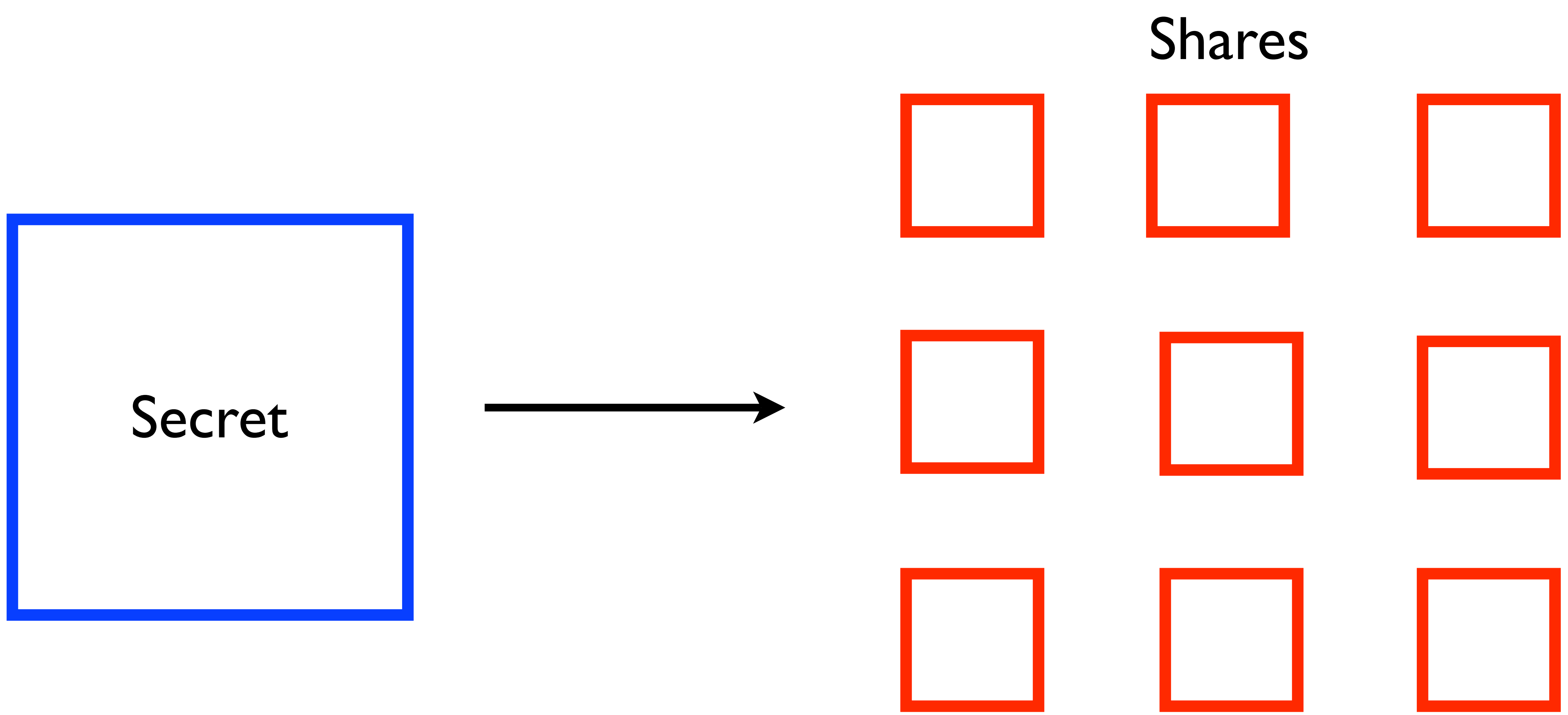
# Secret Sharing



# Secret Sharing

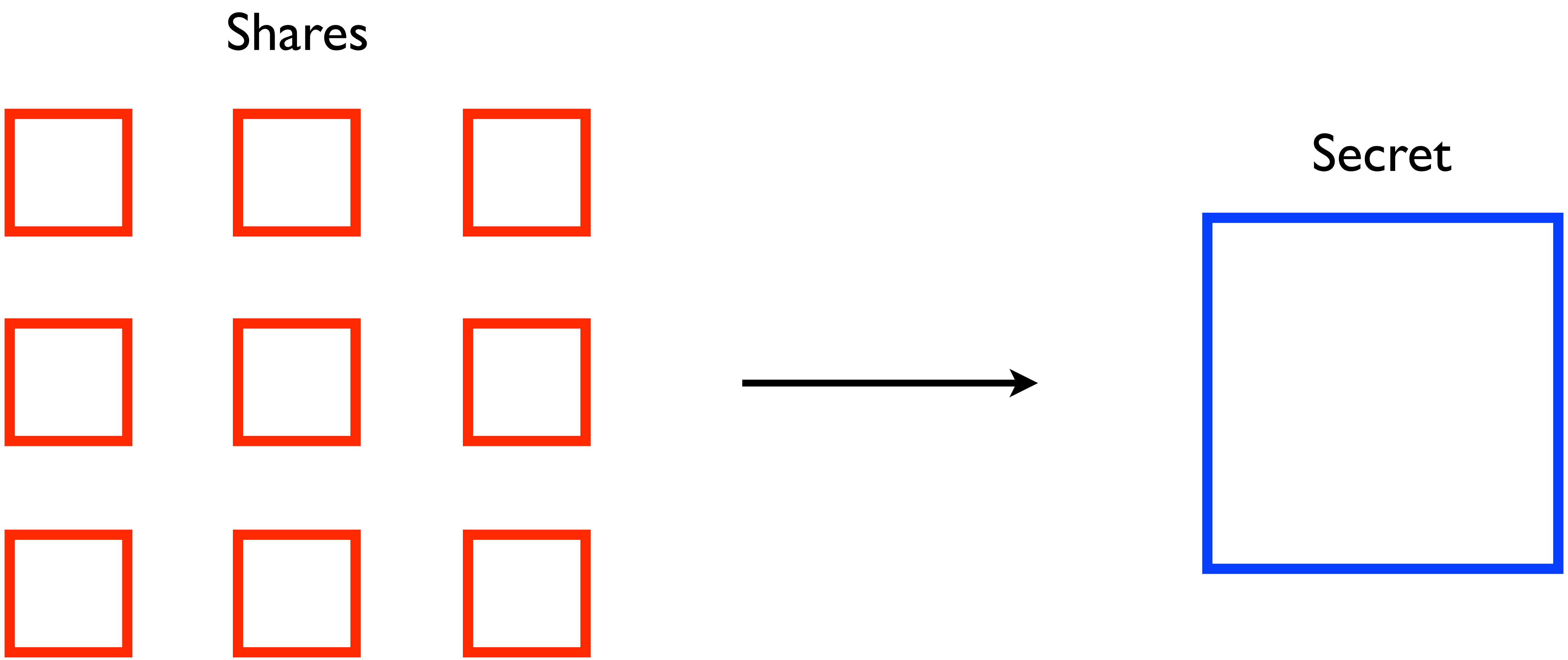


# Secret Sharing



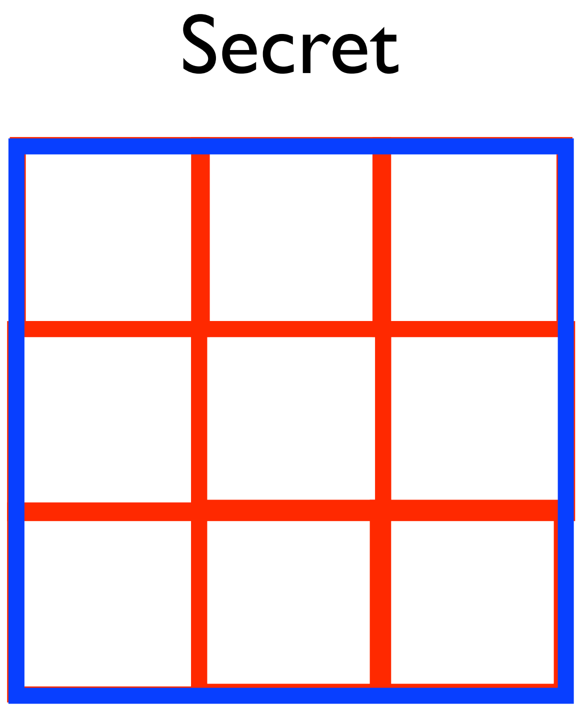

In practice, secrets are shared by evaluating polynomials over finite fields.

# (n, n) secret sharing



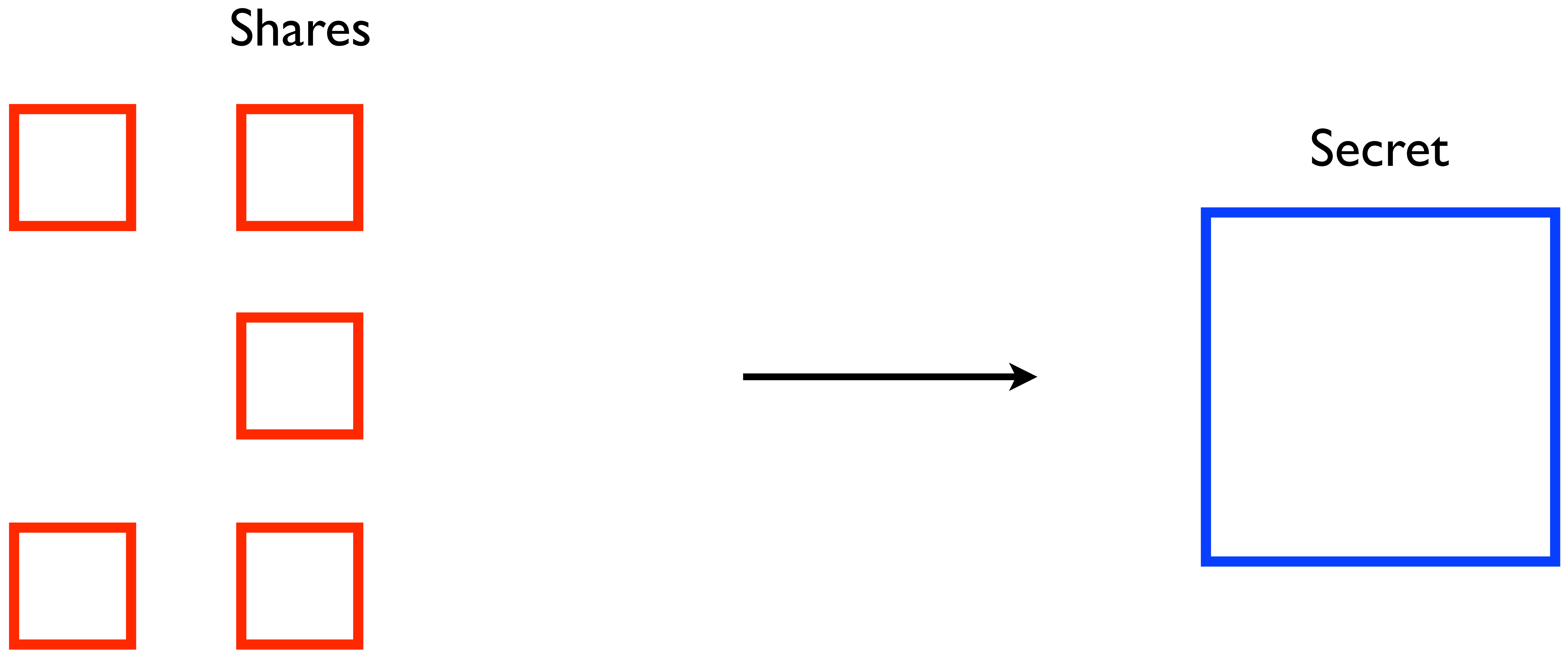

All n shares required to recover secret
No information revealed if fewer than n shares available
Developed independently by shamir and Blakely in 1974.

# (n, n) secret sharing




All n shares required to recover secret
No information revealed if fewer than n shares available
Developed independently by shamir and Blakely in 1974.

# (k, n) secret sharing

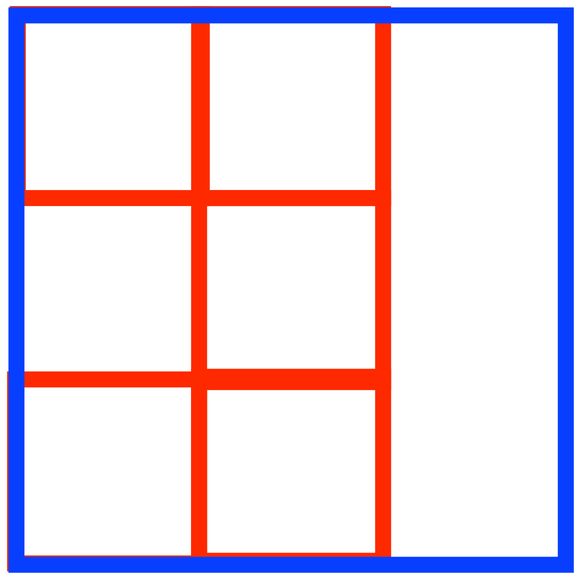



Any k out of n shares are sufficient to recover secret.

No information revealed if fewer than k shares are available.

# (k, n) secret sharing

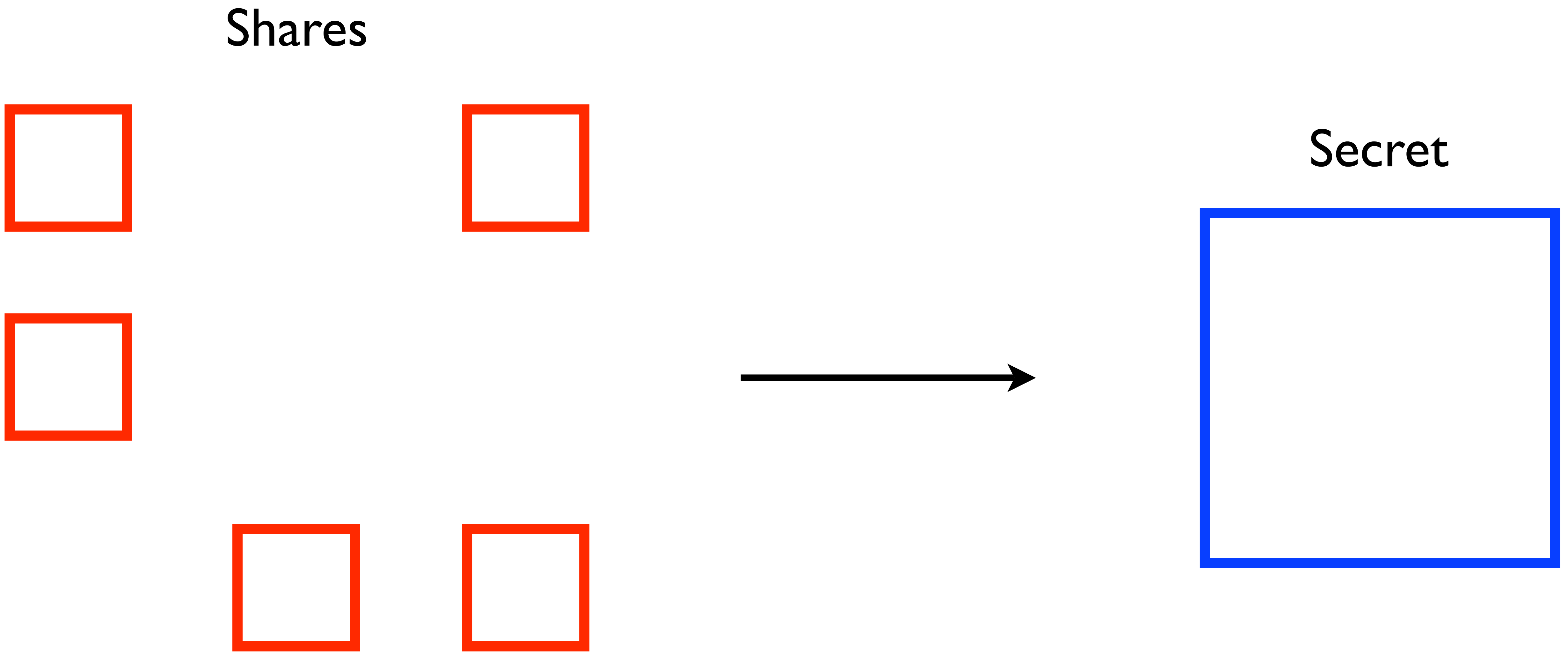
Secret



Any k out of n shares are sufficient to recover secret.

No information revealed if fewer than k shares are available.

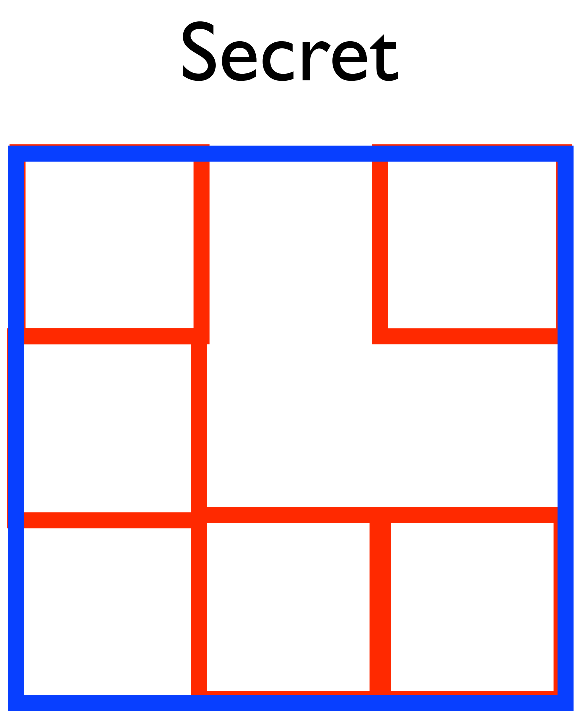
# (k, n) secret sharing




Any k out of n shares are sufficient to recover secret.

No information revealed if fewer than k shares are available.

# (k, n) secret sharing




Any k out of n shares are sufficient to recover secret.

No information revealed if fewer than k shares are available.

# Our approach

# Our approach


Pallet of n

$t_1$

$t_2$

$t_3$

$t_4$

.

.

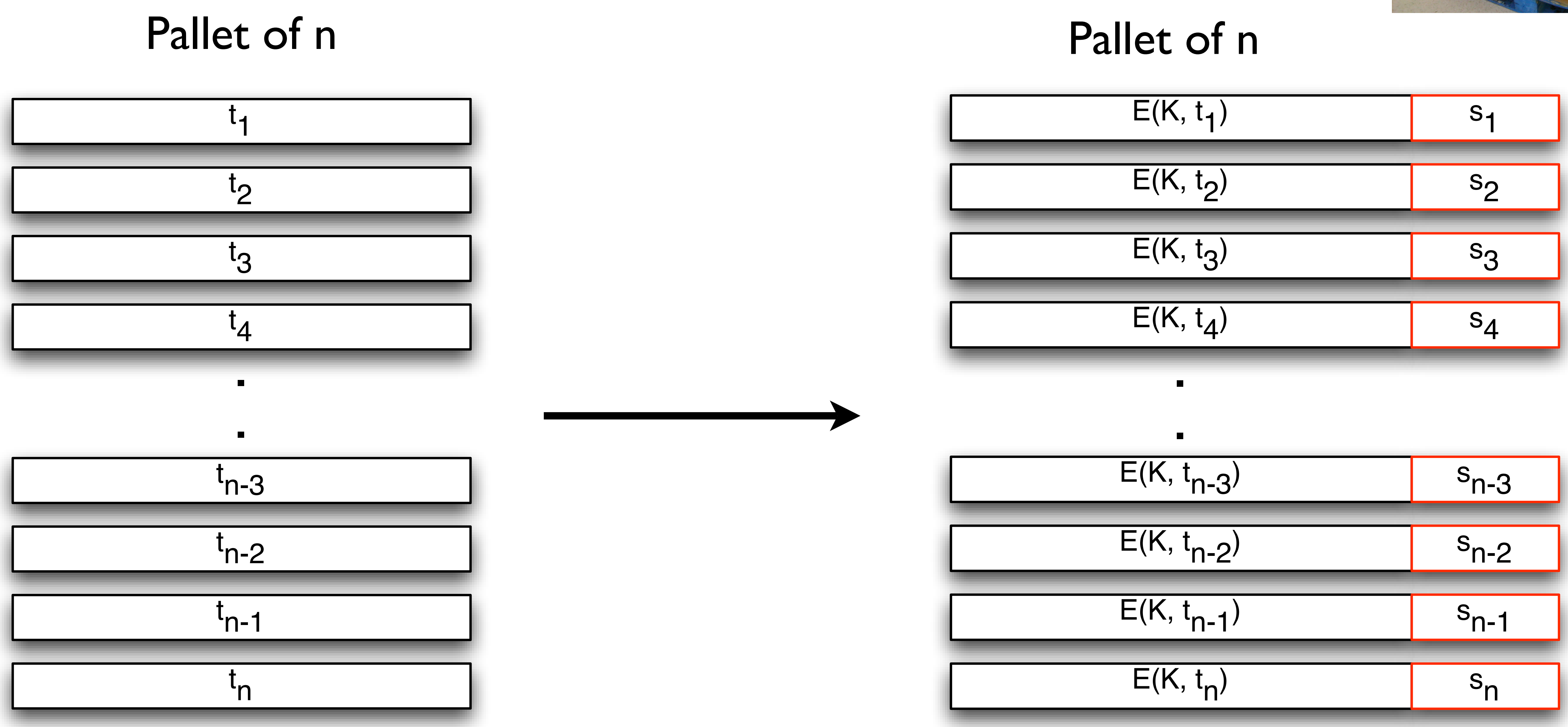
$t_{n-3}$

$t_{n-2}$

$t_{n-1}$

$t_n$

# Our approach




Encrypt all tags using a pallet-specific secret key K.

Append a share of the secret to each tag.

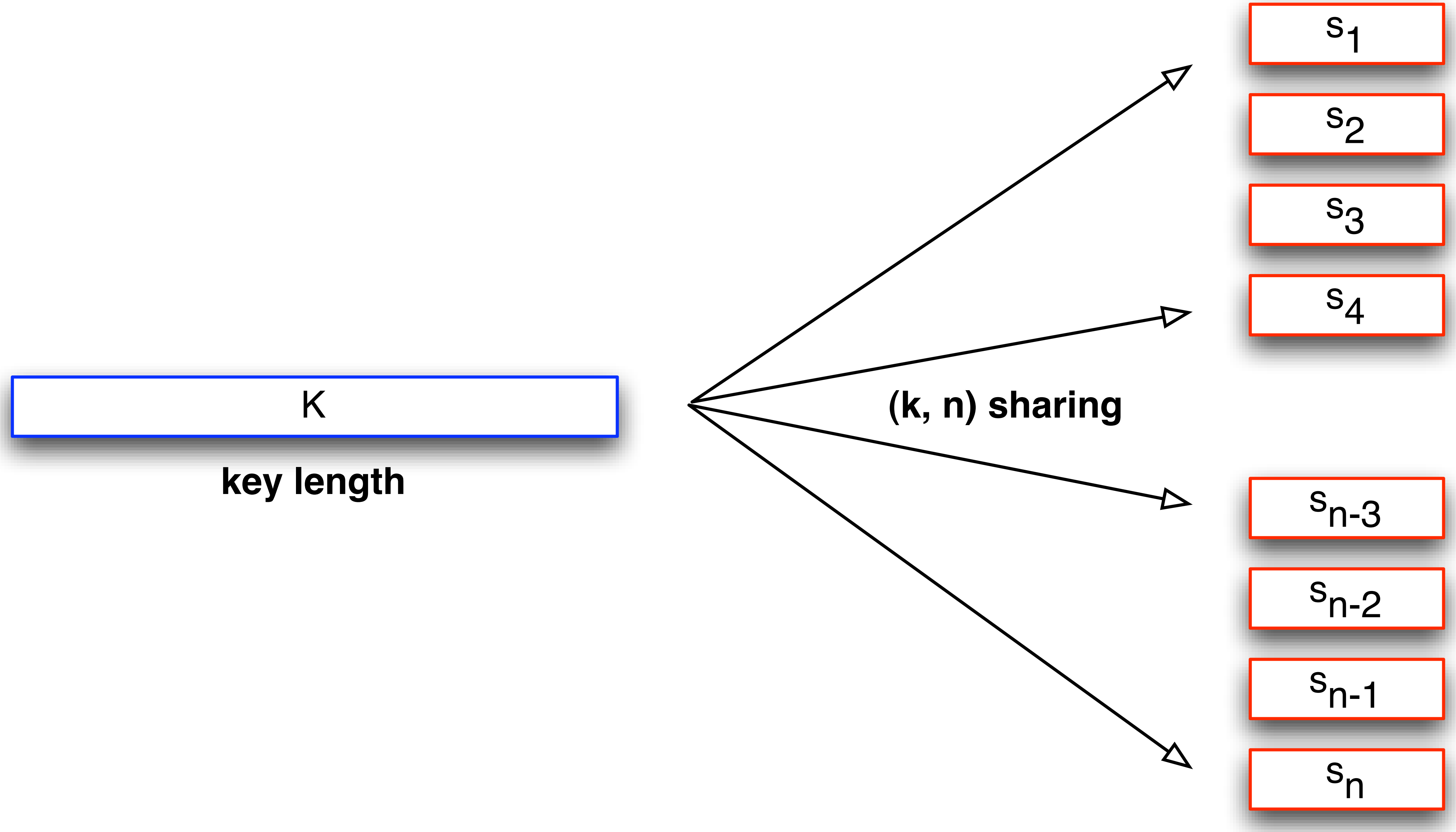
# System level challenges

Need to decide encryption method, key size, sharing/recovery algorithm, share size



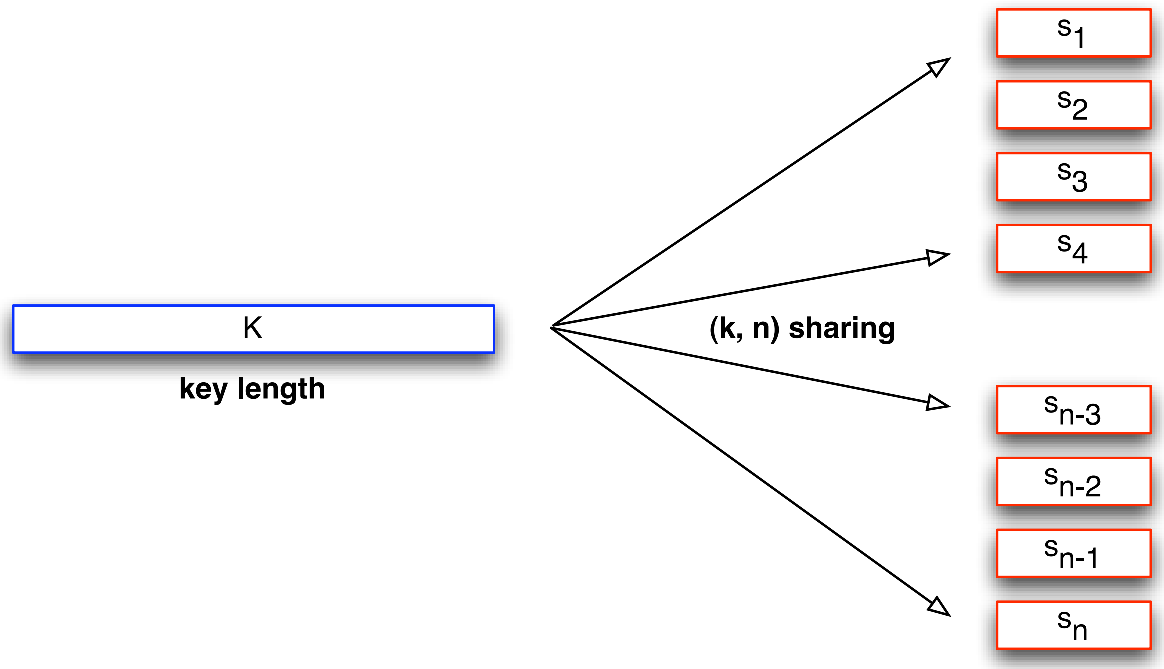

13

# System level challenges



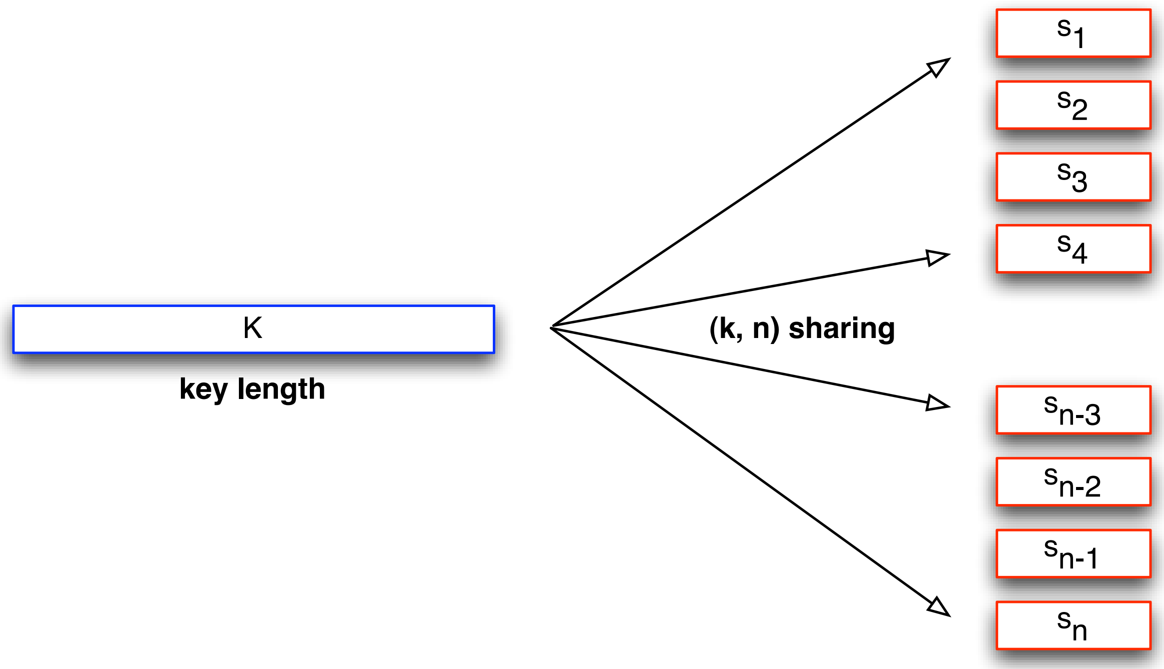

Need to decide encryption method, key size, sharing/recovery algorithm, share size

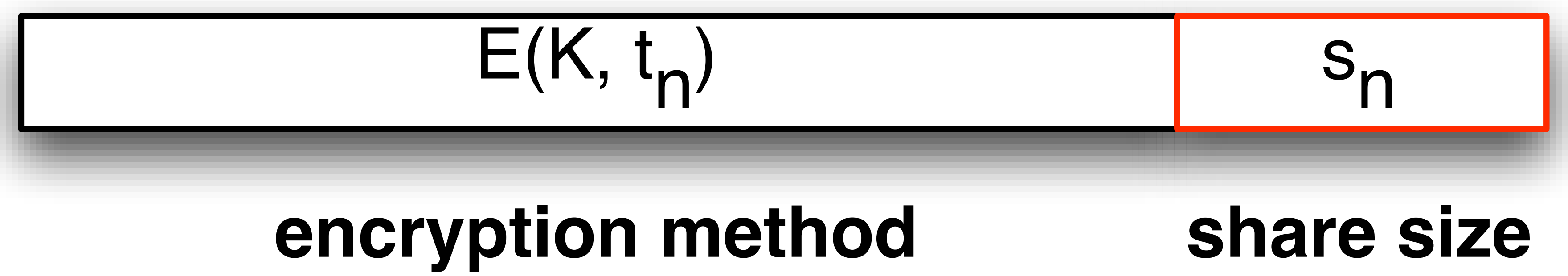
# System level challenges




Need to decide encryption method, key size, sharing/recovery algorithm, share size

# System level challenges



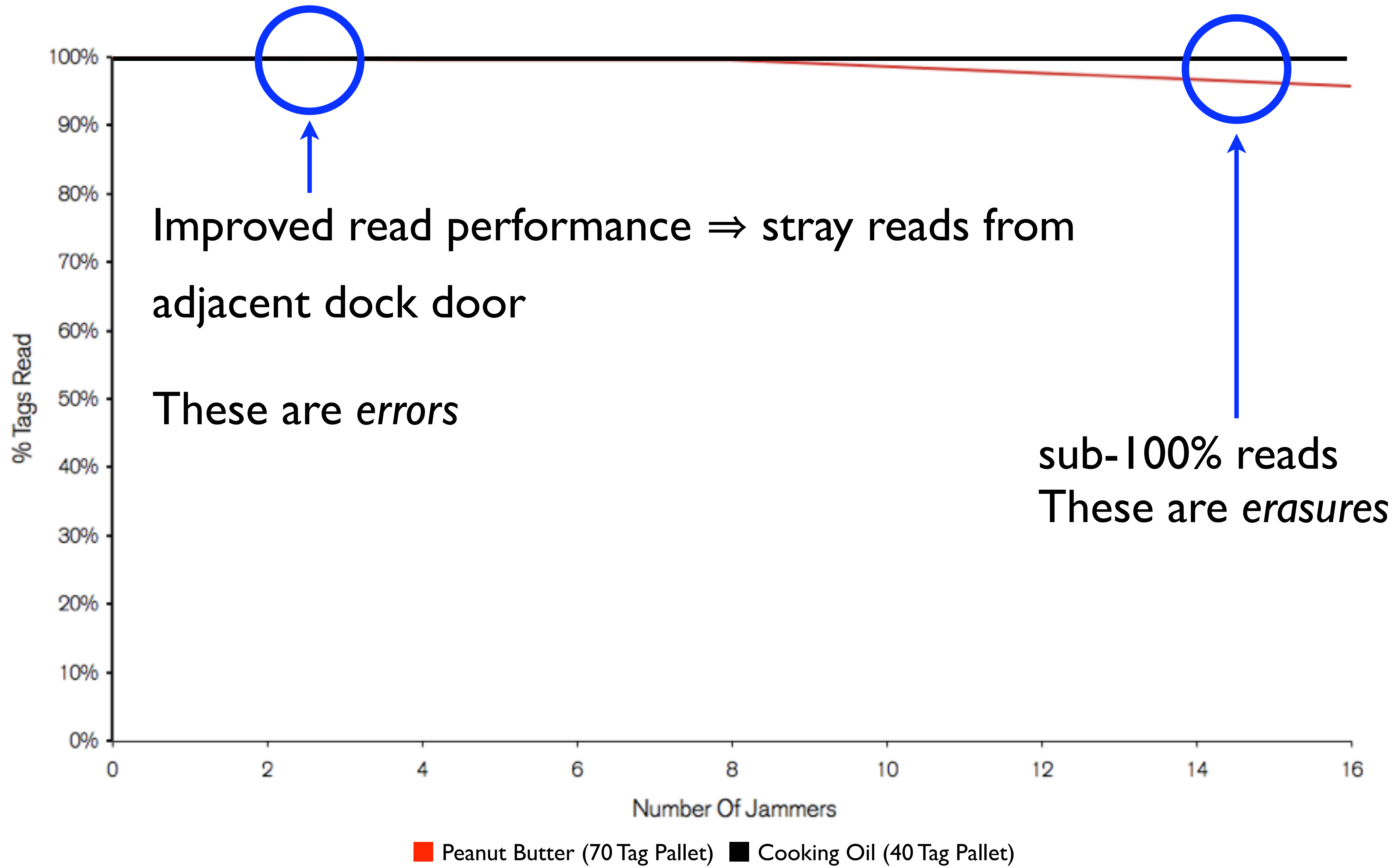




Need to decide encryption method, key size, sharing/recovery algorithm, share size

# Practical requirements

- Share size has to be *tiny*, because tag memory is at a premium
    - Krawczyk (1994) focused on *short* shares, but even these are 128 bits long.
    - Current memory capacity on EPC Gen2 tags is 96 bits ⇒Tiny Secret Sharing
- Sharing and recovery algorithms have to be computationally efficient.
- System should be robust against
    - changes in the order of tag reading i.e. permutation invariance
    - sub-100% read rates
    - stray reads (or counterfeits)

# Mercury5 read performance



# Reed-Solomon Codes

- $(n, k, d)$ codes over $GF(2^m)$
- Total number of symbols = number of tags = $n$
- Symbols required to recover message = threshold number of tags = $k$
- Code can detect and correct upto $s = (d/2)$ *errors* = stray tags or counterfeits
- Code can detect and correct upto $r = (d-1)$ *erasures* = missed tags
- With $s$ stray tags and $r$ missed tags, code can correct as long as $2s + r < d$
- This formulation is identical to $(k, n)$ secret sharing
- Encoding and decoding are efficient on low-powered machines

# Permutation invariance

- RS codes originally invented to solve digital communication problems
- Order of code symbols is usually preserved

$$(s_1, s_2, s_3.....s_{n-2}, s_{n-1}, s_n) \rightarrow (s_1, s_2, s_3.....s_{n-2}, s_{n-1}, s_n)$$

- In RFID, code symbols are always permuted, because order of tag reading is based on randomization at the MAC layer.

$$(s_1, s_2, s_3.....s_{n-2}, s_{n-1}, s_n) \rightarrow (s_6, s_1, s_{n-4}.....s_n, s_7, s_3)$$

- In order to use RS codes, we also need to make the symbol index available at destination.

$$(\{s_6, 6\}, \{s_1, 1\}, \{s_{n-4}, (n-4)\}.....\{s_n, n\}, \{s_7, 7\}, \{s_3, 3\})$$

# Permutation invariance 2




18

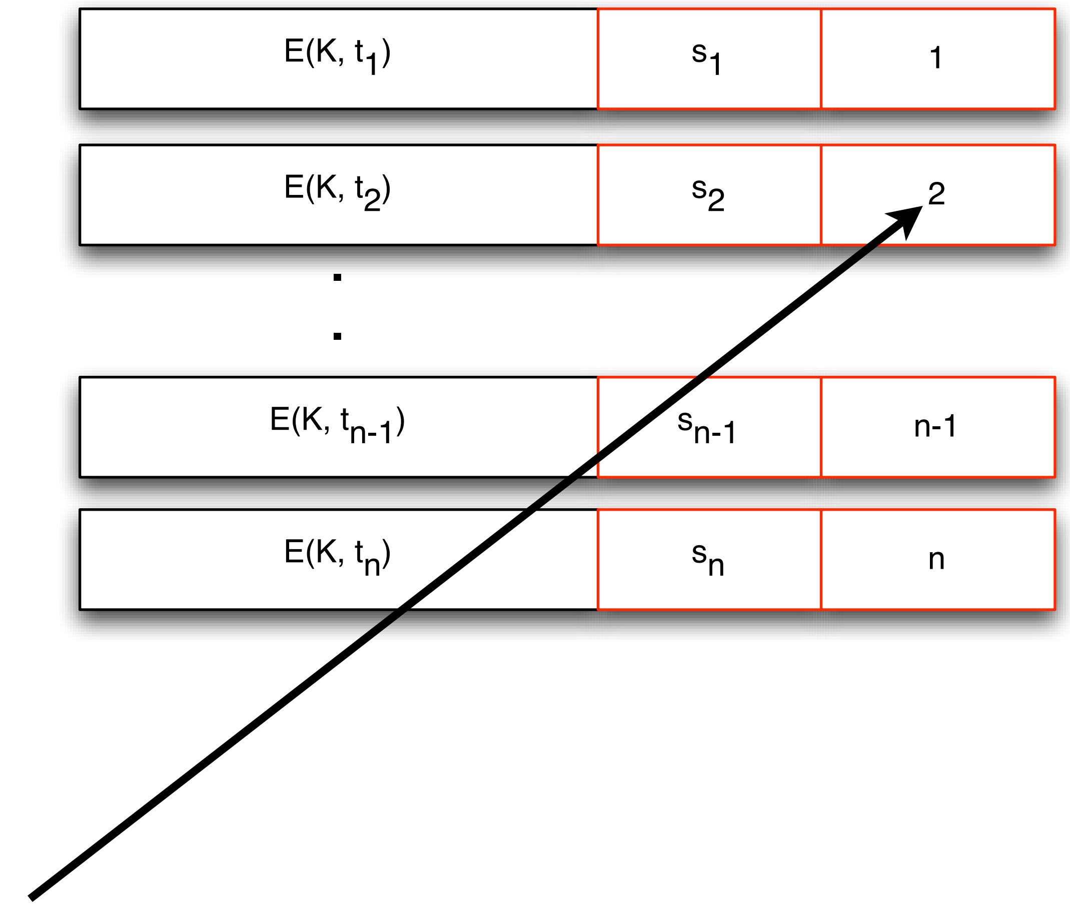
# Permutation invariance 2

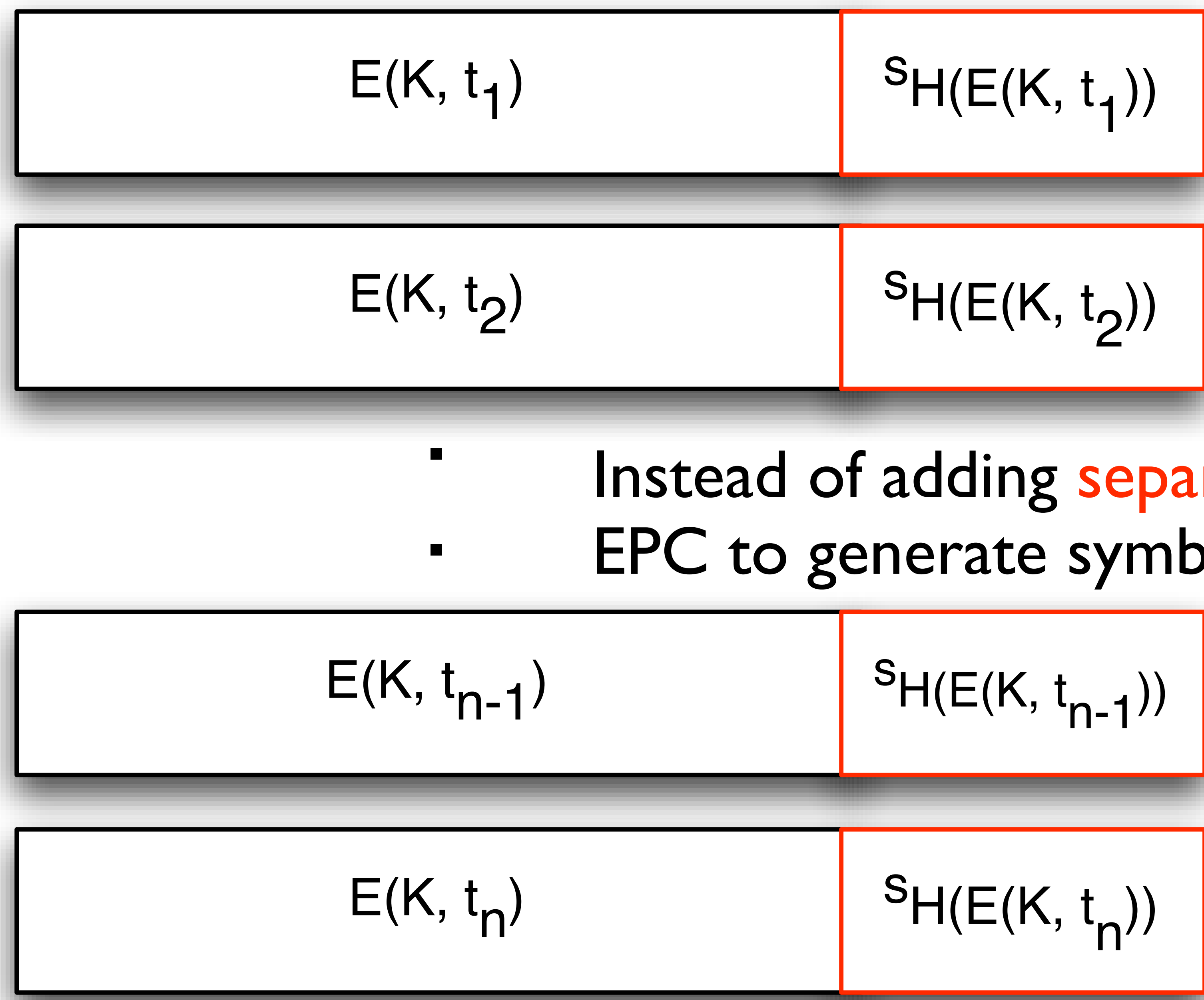
| | | |
|---|---|---|
| E(K, $t_1$) | $s_1$ | 1 |
| E(K, $t_2$) | $s_2$ | 2 |
| . . | | |
| E(K, $t_{n-1}$) | $s_{n-1}$ | n-1 |
| E(K, $t_n$) | $s_n$ | n |

# Permutation invariance 2

| | | |
|---|---|---|
| $E(K, t_1)$ | $s_1$ | 1 |
| $E(K, t_2)$ | $s_2$ | 2 |
| . | | |
| . | | |
| $E(K, t_{n-1})$ | $s_{n-1}$ | n-1 |
| $E(K, t_n)$ | $s_n$ | n |

# Permutation invariance 2



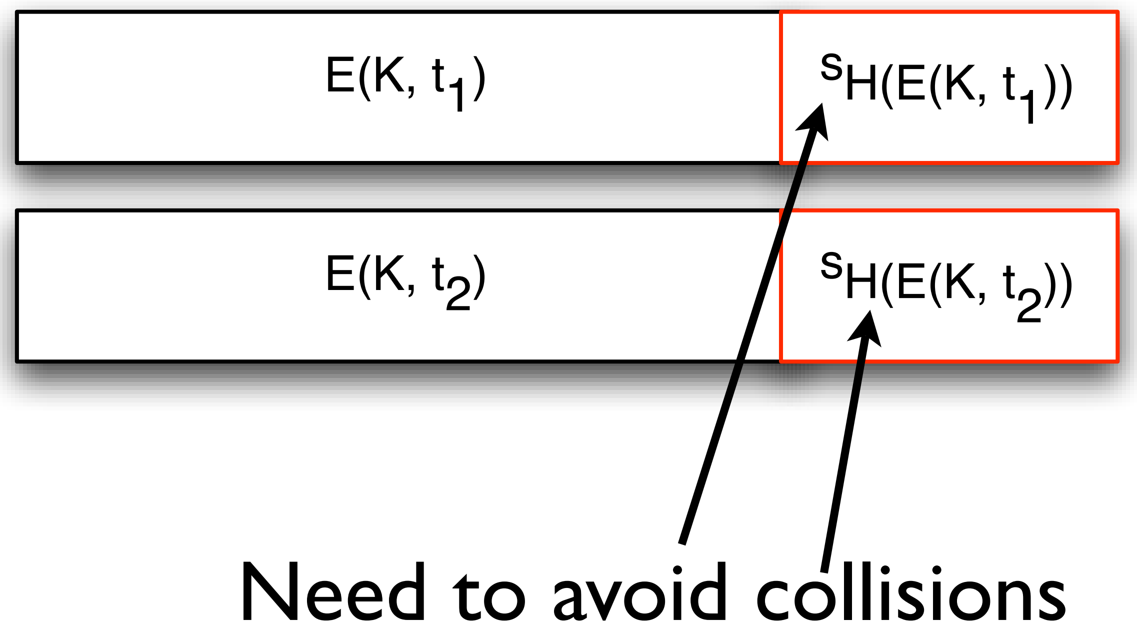




Instead of adding separate index, use uniqueness of the EPC to generate symbol index on the fly via hashing

# Field size

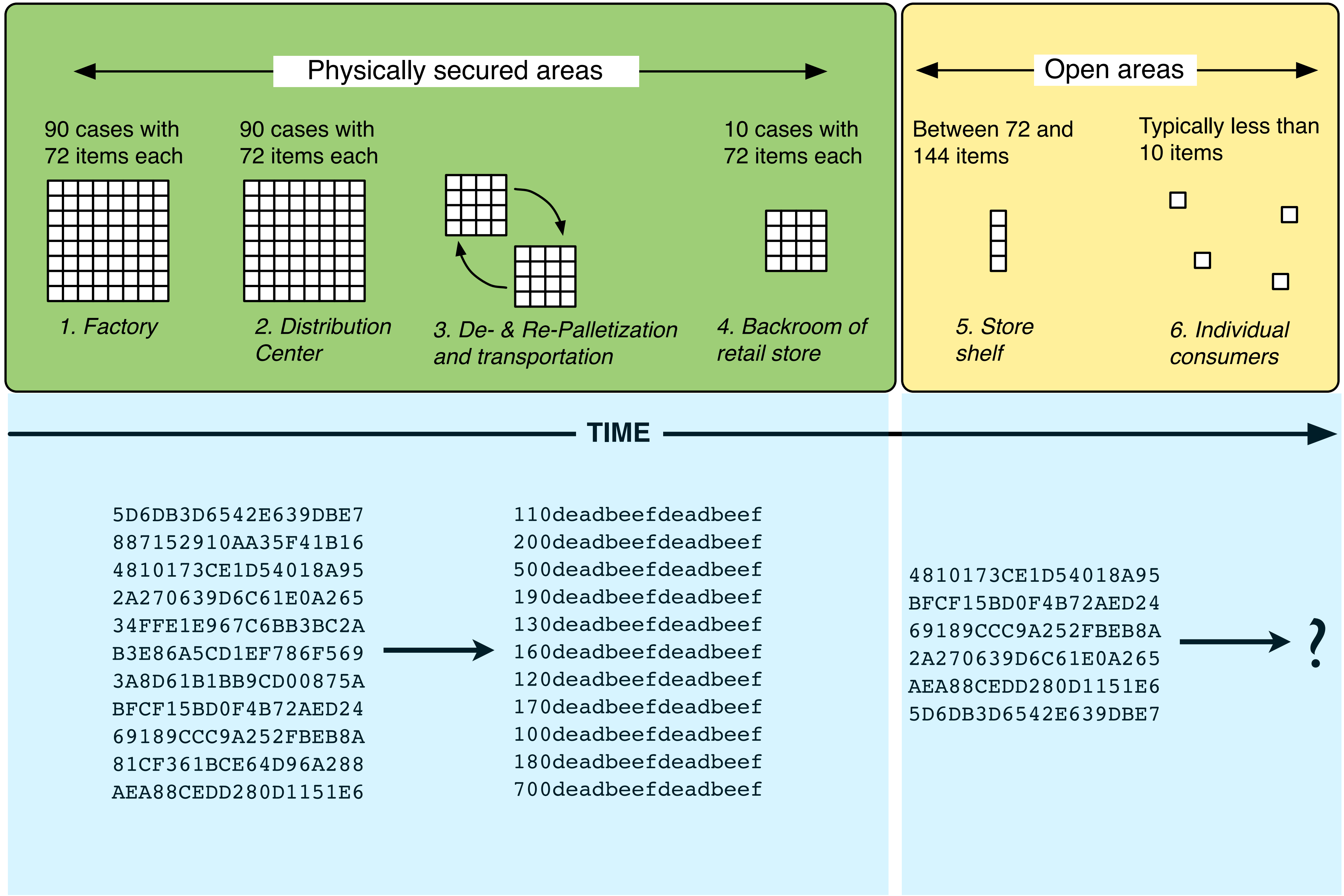
- Field size $GF(2^m)$ needs to be chosen to avoid index collisions.
- If we have $n$ tags, then $2^m \geq n^2$ or $m \geq log_2(n^2)$

| # of tags *n* | # of bits in shares *m* |
|---|---|
| 50 | 12 |
| 256 | 16 |
| 1000 | 20 |
| 10000 | 27 |



# (15, 20) TSS scheme over GF($2^{16}$)

# RFID privacy without killing



# Summary

- Tiny Secret Sharing (TSS) enables RFID privacy without killing
- Encryption key length is a free-variable - security can be tailored.
- TSS is protocol-independent, and completely local - no network required.
- It scales to item-level tagging
- The only resources needed are tag memory and some computing power at reader
- TSS allows use of standard cryptographic mechanisms for encryption, hashing
- TSS fits naturally in many supply-chain scenarios where we have less than 100% reads and where stray tags or counterfeits are present.
- TSS solves key-management problem - enables privacy and write/lock PIN distribution.

# Key messages

- Tiny Secret Sharing (TSS) enables consumer privacy *now*
  - No heroic measures required
  - No dependence on any particular standard ⇒ fully standards compliant
- Consumer privacy is achieved by exploiting the natural movement of tagged products through the supply chain
  - Privacy through dispersion and history erasure

# What's next?

- Real implementation on ThingMagic Mercury5 reader in progress
- Discussion about implementation in real-world needs to happen
  - Pharmaceutical supply chain appears to be ideal
- Secret sharing across time or Sliding Window Information Secret Sharing (SWISS) also detailed in paper below
- Preprint available at http://eprint.iacr.org/2008/044
- Questions: ravi.pappu@thingmagic.com