# DRESS Code For The Storage Cloud

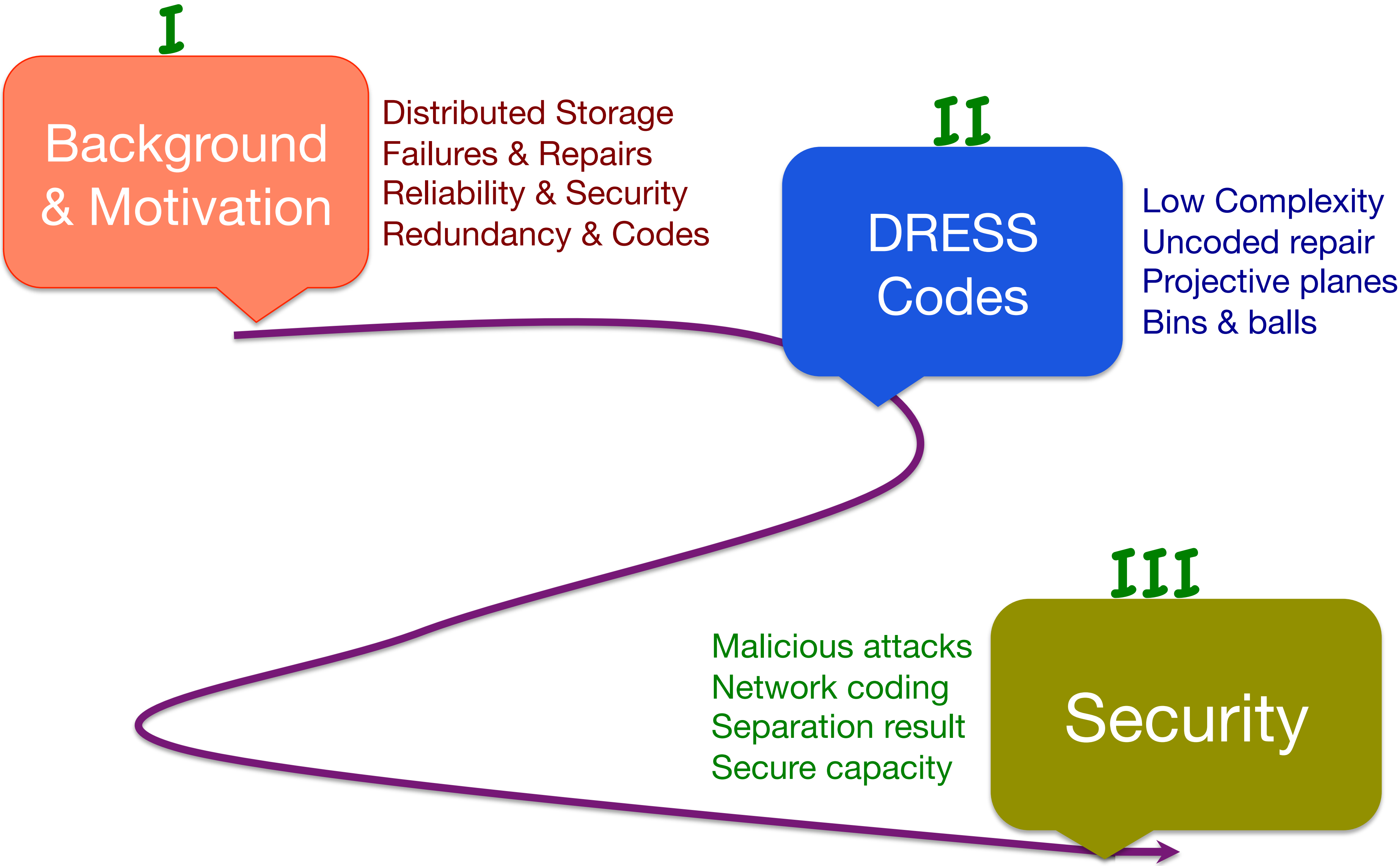
**D**istributed
**Re**plication-based
**E**xact **S**imple **S**torage

Salim El Rouayheb

EECS Department
University of California, Berkeley

Joint work with: Sameer Pawar
Nima Noorshams
Prof. Kannan Ramchandran

# Talk Roadmap



# Data Everywhere


- Increasing popularity of data-enabled devices and Hot spots, 4G, Wimax, Wi-fi etc, => “always online” infrastructure.
- Truly mobile users
- Want to be able to seamlessly access and share data anytime anywhere

# Off To The Cloud…


- Want our files to "follow" us wherever we go
  - Put them in the cloud
  - Get them from the cloud

Data Centers

amazon webservices™

Google

P2P Systems

oceanstore

wuala

Cloud Applications

NETFLIX

Dropbox

# Amazon Web Services

## Amazon Simple Storage Service (Amazon S3)

Amazon S3 is storage for the Internet. It is designed to make web-scale computing easier for developers.

Sign Up For Amazon S3

Dropbox

yelp

zmanda Open Source Backup

SmugMug

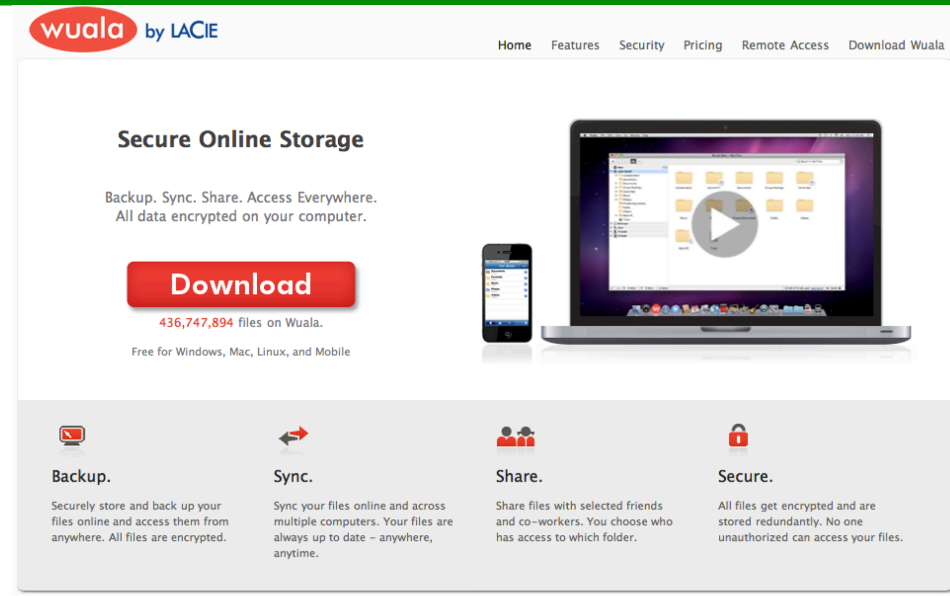
## Pricing

Region: US – Standard

| Storage (Designed for 99.999999999% Durability) | | Reduced Redundancy Storage (Designed for 99.99% Durability) | | Data Transfer** | | Requests | |
|---|---|---|---|---|---|---|---|
| **Tier** | **Pricing** | **Tier** | **Pricing** | **Tier** | **Pricing** | **Type** | **Pricing** |
| First 1 TB / month of Storage Used | $0.140 per GB | First 1 TB / month of Storage Used | $0.093 per GB | All data transfer in | $0.100 per GB | PUT, COPY, POST, or LIST Requests | $0.01 per 1,000 Requests |

# Wuala: P2P Cloud Storage

- Trade idle local disk space on your computer for online storage
- First 1GB is free

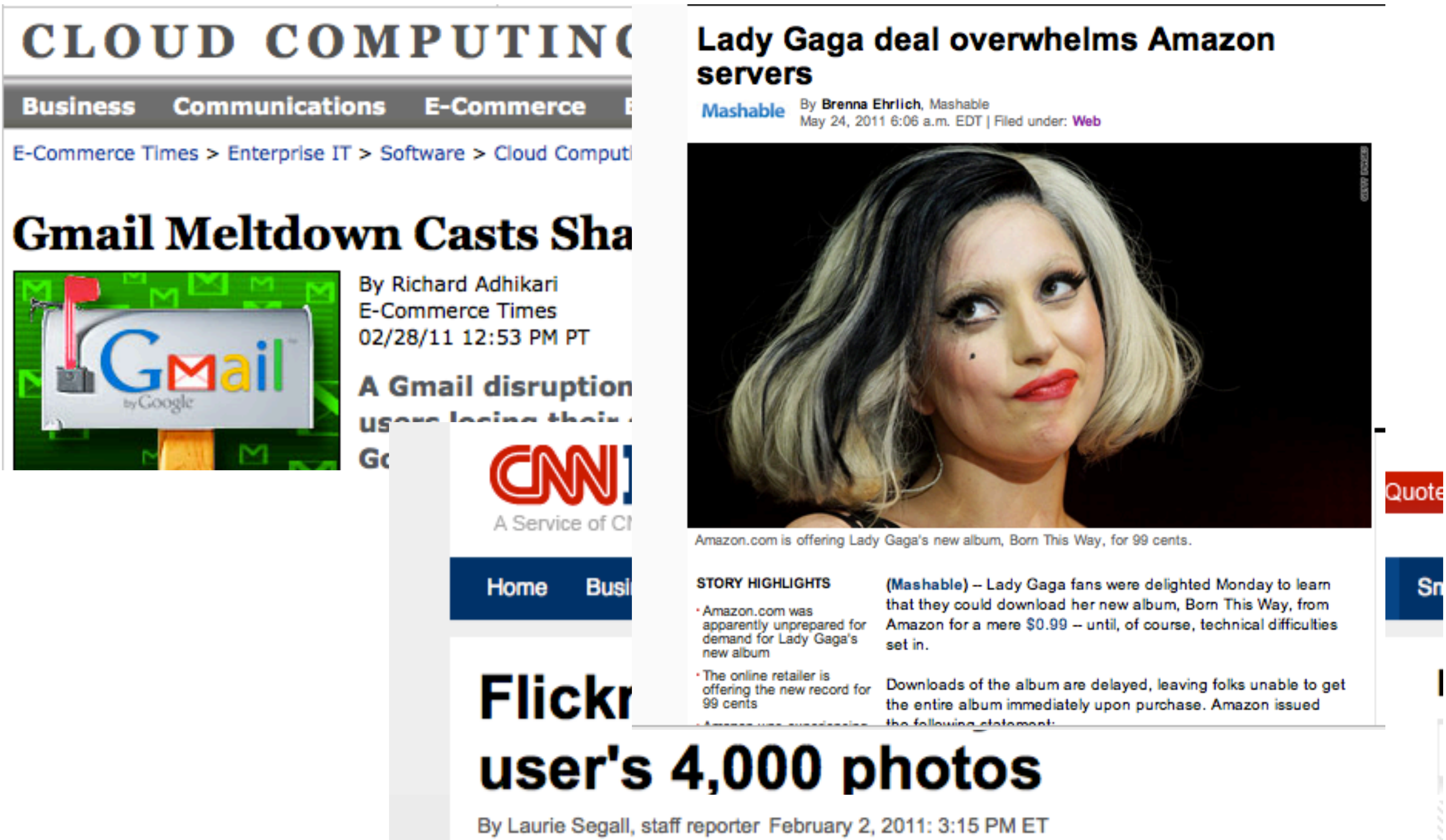



Cloud storage = local donated storage * percentage of online time

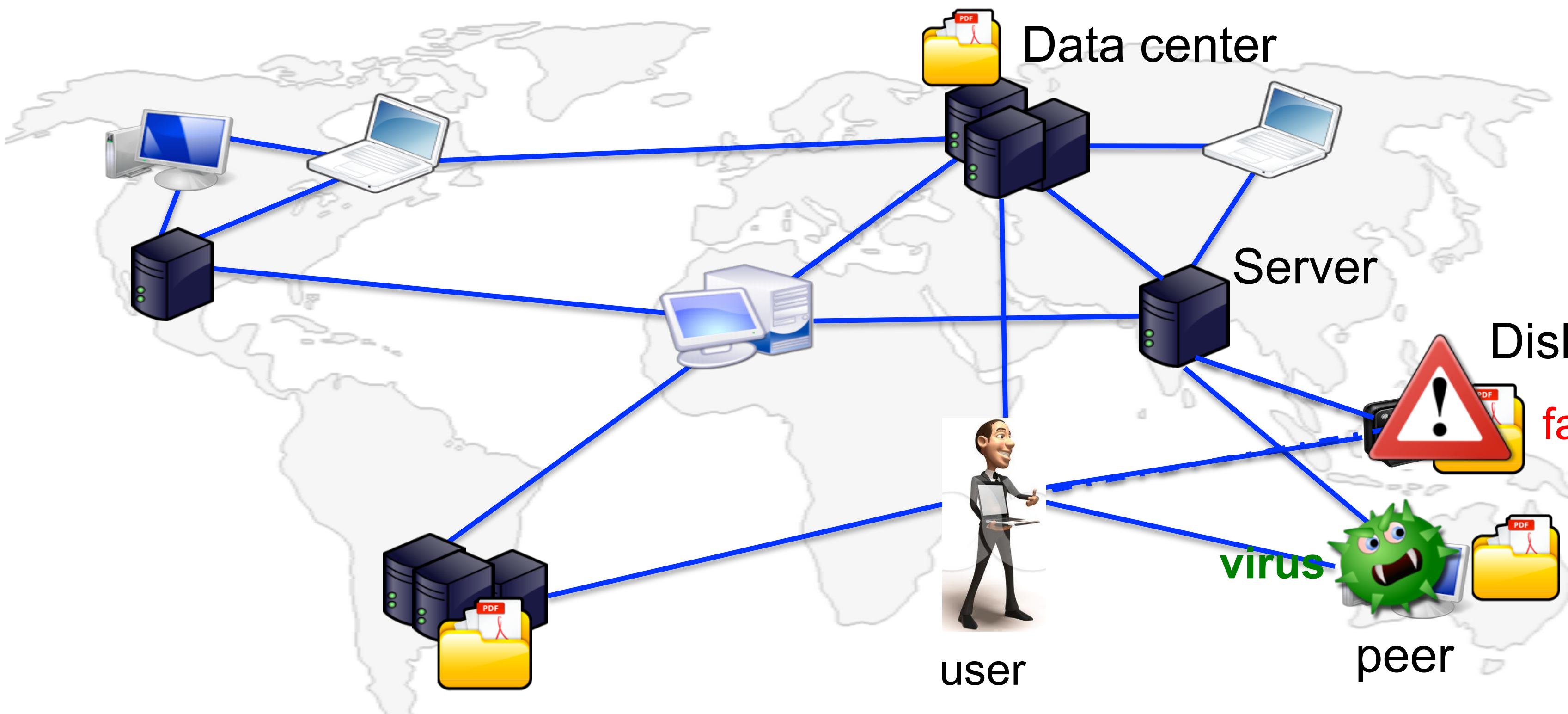
70 GB 100 GB 70%

# Can We Trust the Cloud?

**Main Challenge:** Cloud is formed of unreliable and untrusted components

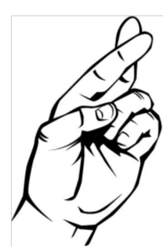


# Cloudy with a Chance of Failure



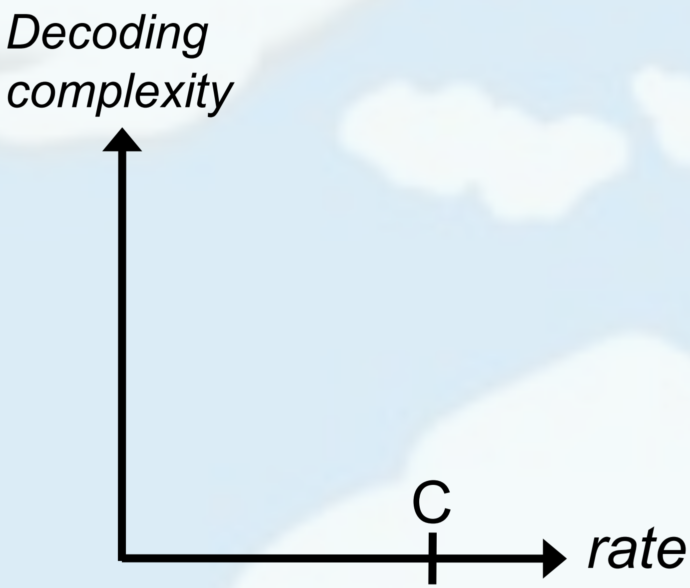

Component failures are the norm rather then exception.

- Nodes are unreliable by nature:
  - Hard disk failure
  - Network failure
  - Peer churning
  - Malicious nodes

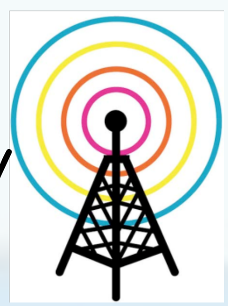


Solution: store data redundantly

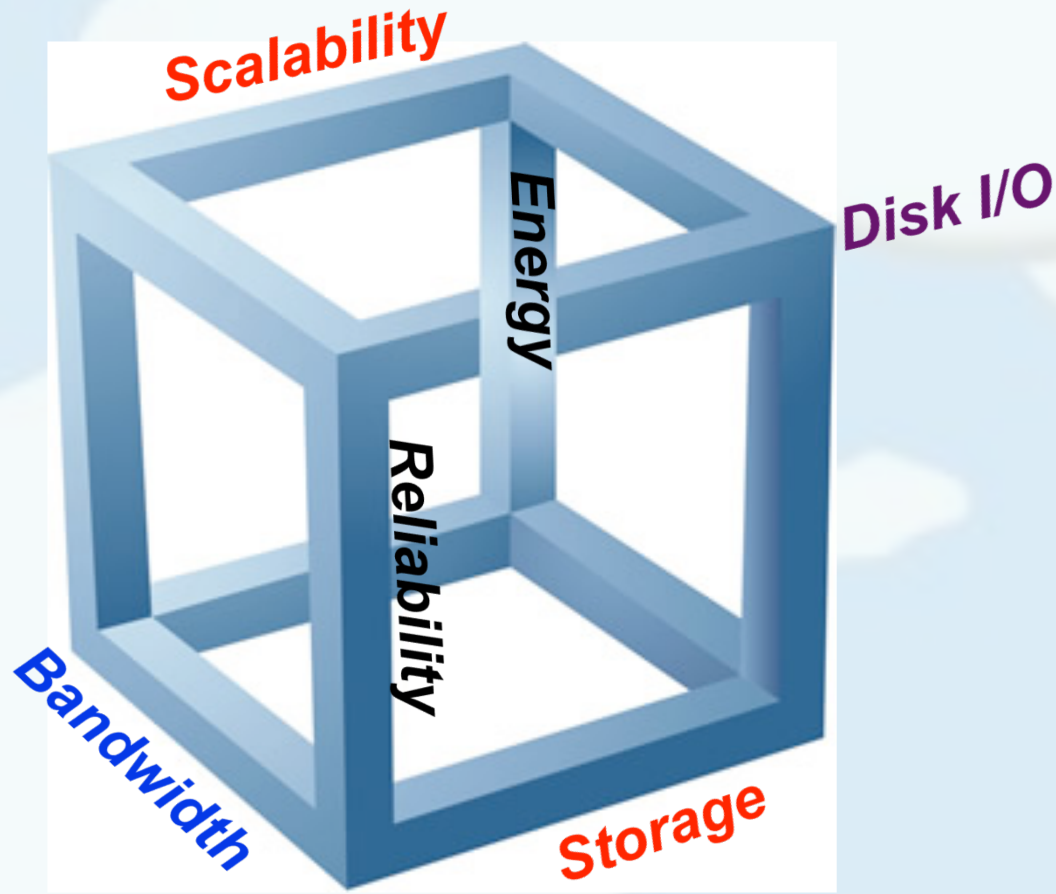
# Multiple System Considerations




"Traditional" Coding Theory

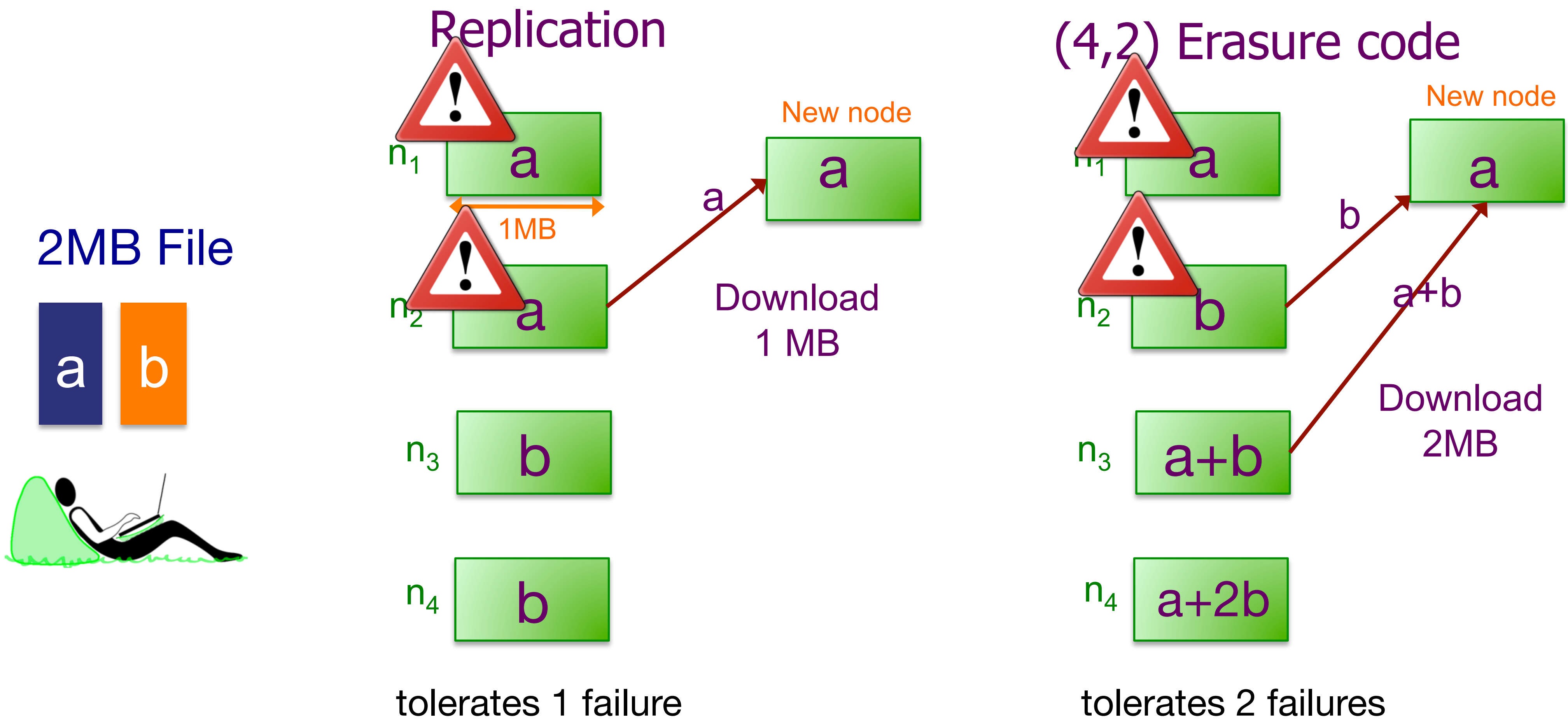


Coding for cloud storage




Escher "Impossible" Cube

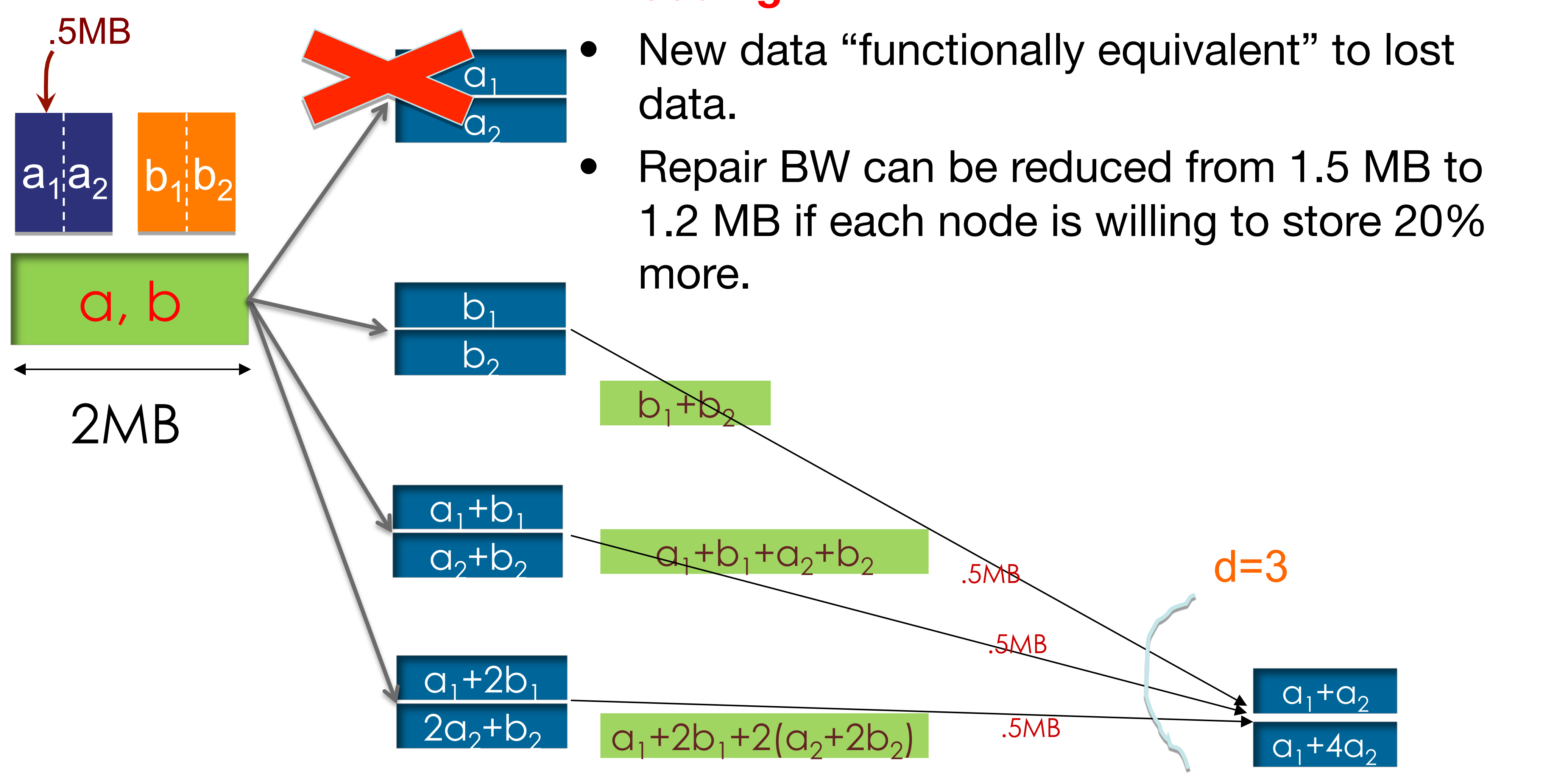
# Different Codes for Different Folks




| | | |
|---|---|---|
| **Reliability** | | ✓ |
| **Bandwidth** | ✓ | |
| **Complexity/ Disk I/O** | ✓ | |

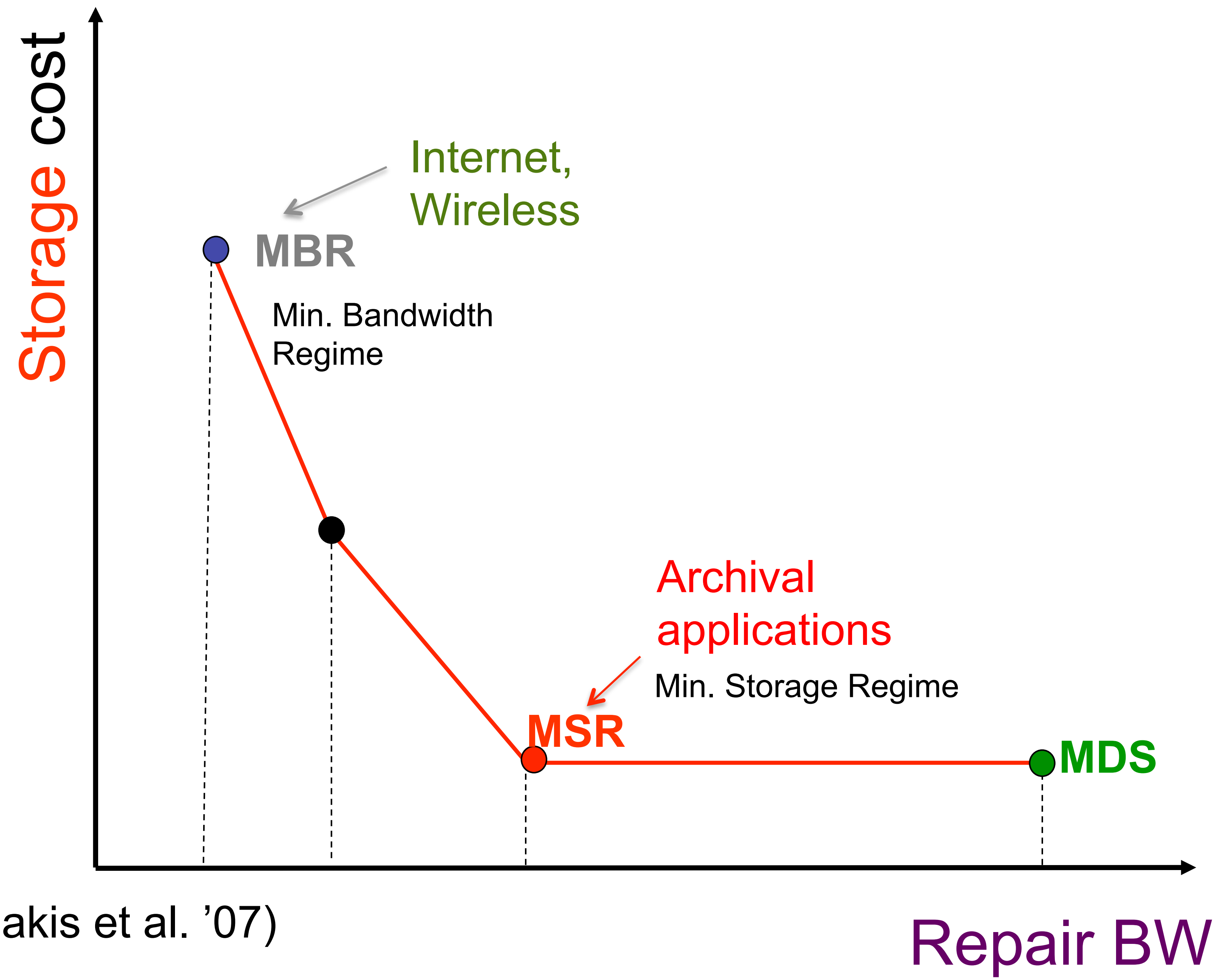
# Reducing Bandwidth Overhead

- Repair BW can be reduced from 2 to 1.5 MB
- Helper nodes send random linear combinations: **random linear network coding**
- New data “functionally equivalent” to lost data.
- Repair BW can be reduced from 1.5 MB to 1.2 MB if each node is willing to store 20% more.




Dimakis, Wu & Ramchandran ‘07

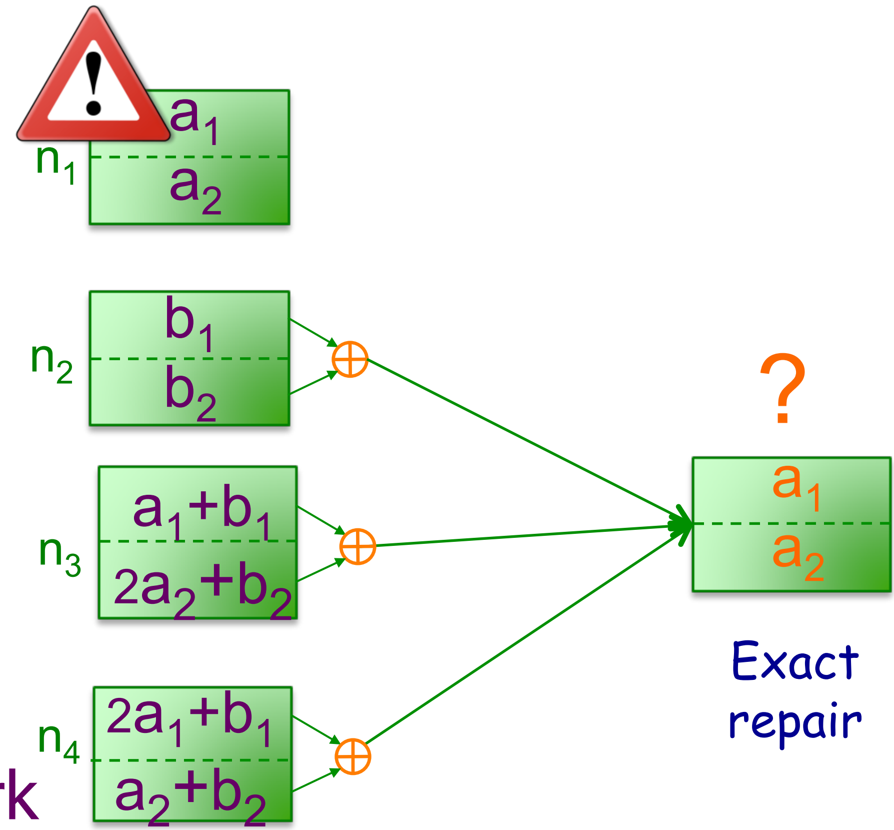
# Fundamental Storage/Bandwidth Tradeoff




(Dimakis et al. '07)

# We Want More: Exact Repair?

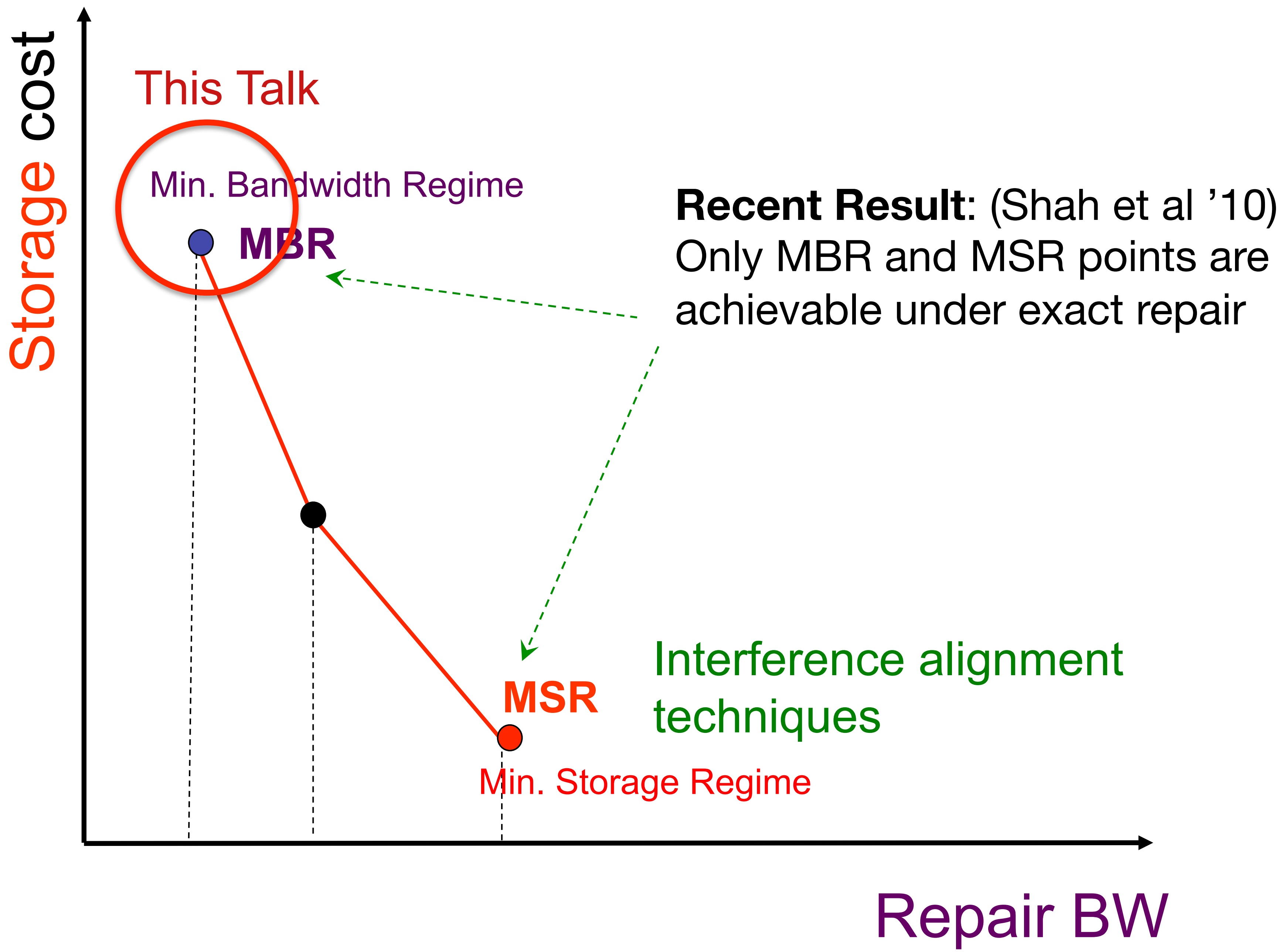
- We want exact repair:
  - Maintain a systematic copy of the data
  - Reduce system overhead
  - Security




- Exact repair makes the problem much harder
  - Relation to non-multicast network coding

- Recent important progress: Rashmi et al. (Product-Matrix Codes), Suh & Ramchandran '10, Cadambe et al. '10

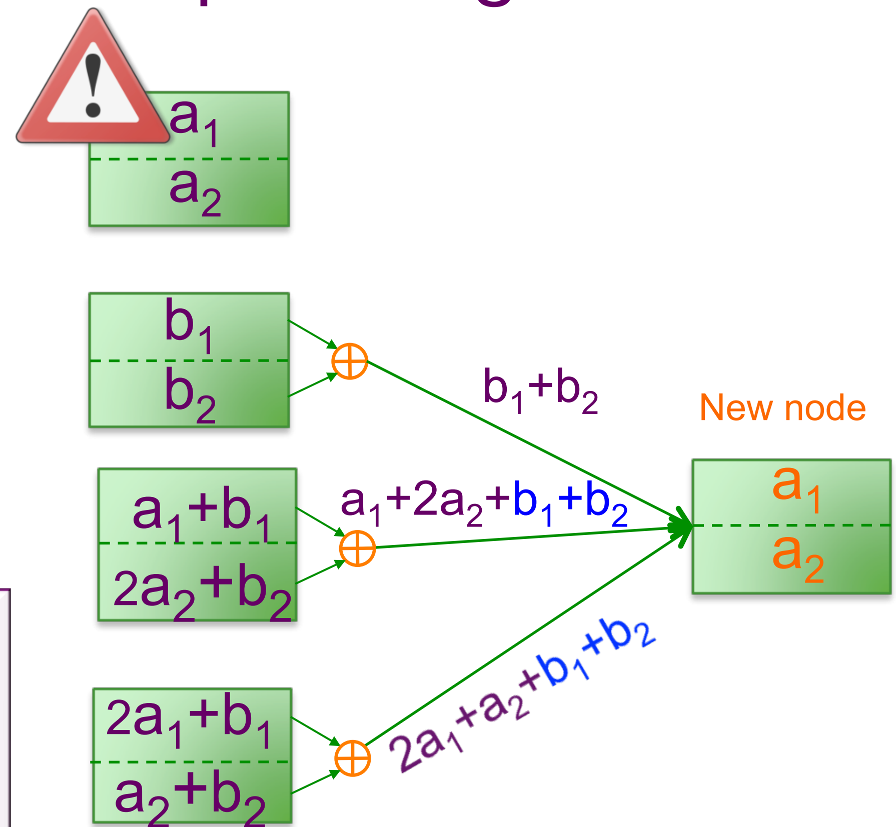
# Storage/Bandwidth Tradeoff



# Can we also reduce Disk I/O & Energy?

## Bandwidth-efficiency comes with a price-tag.

1) Excessive data reading & computations
2) Read/write bandwidth of a hard disk is the bottleneck (100 MB/s vs. network speed ~1000 MB/s)

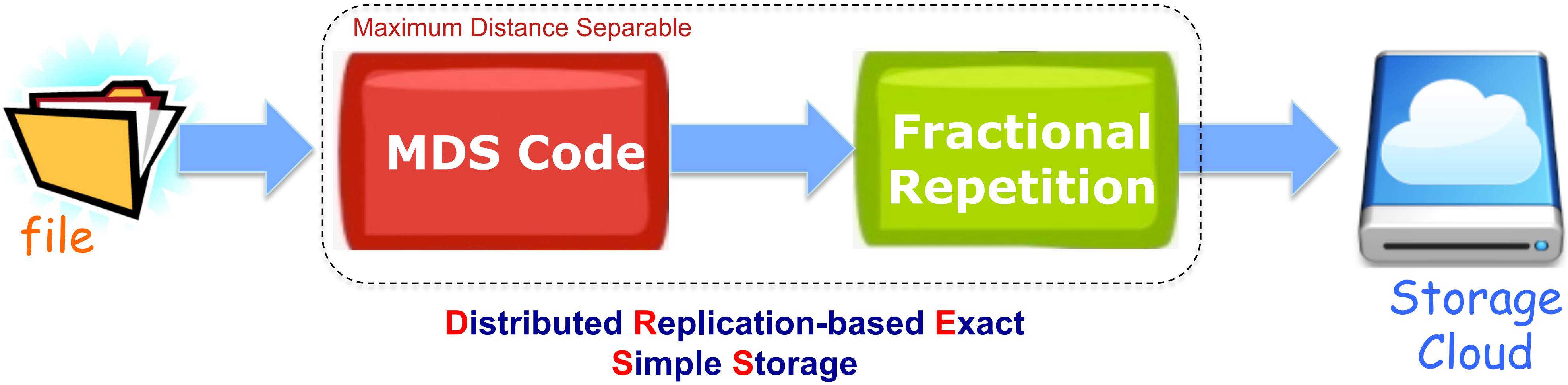



Can we have *exact* codes with minimum overheads of:

1) Repair bandwidth
2) Disk reads
3) Computations?

# Contributions: Yes We Can

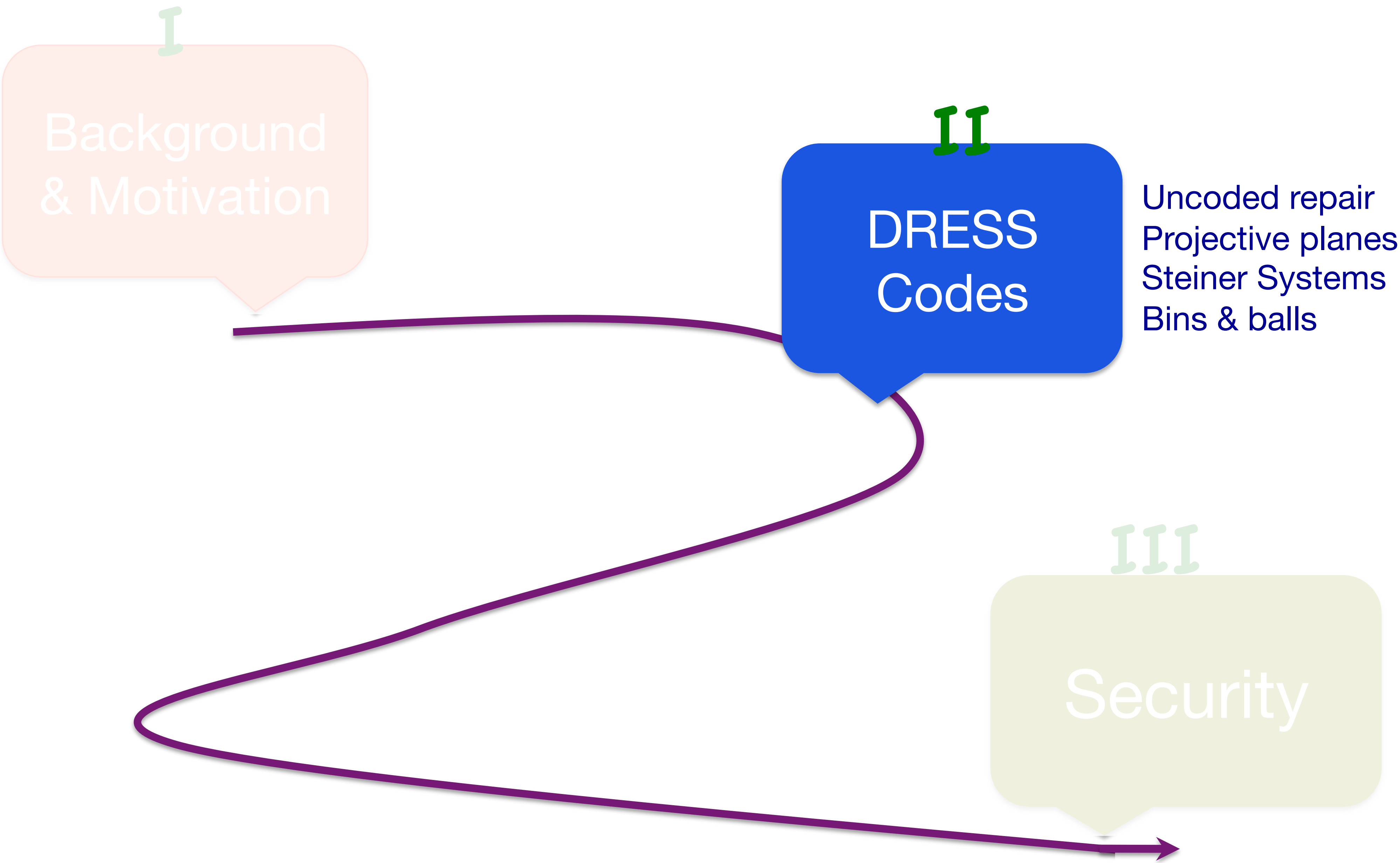
- DRESS codes:
  - ✓ Exact Repair
  - ✓ Minimum disk reads for repair
  - ✓ No data processing for repair: uncoded repair




**D**istributed **R**eplication-based **E**xact **S**imple **S**torage

- Surprise: no loss of bandwidth efficiency
- Bonus: well-suited for security applications
- System price: repair not as flexible w.r.t. *who* can help

# Talk Roadmap

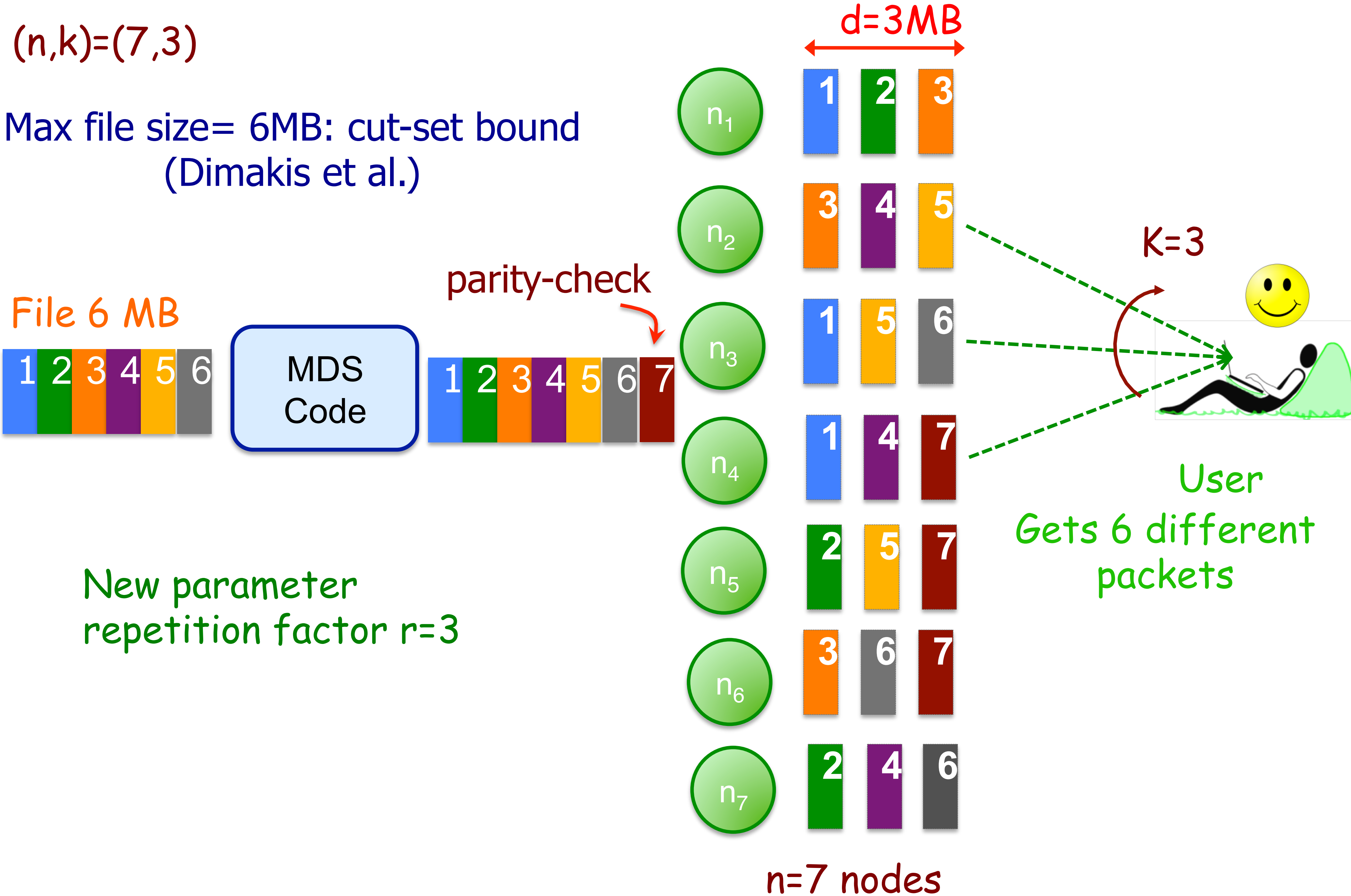



Uncoded repair
Projective planes
Steiner Systems
Bins & balls

# DRESS Code Example

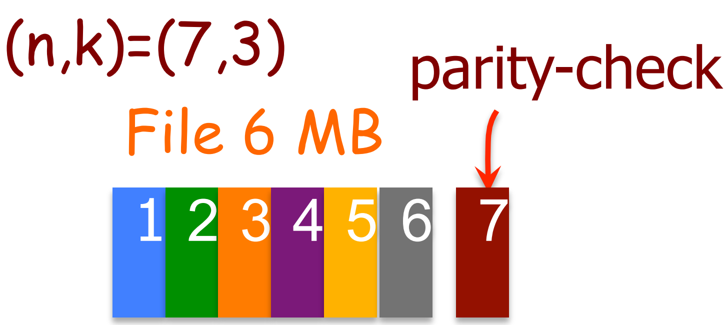
(n,k)=(7,3)

Max file size= 6MB: cut-set bound (Dimakis et al.)




New parameter repetition factor r=3

# Repairing DRESS Codes

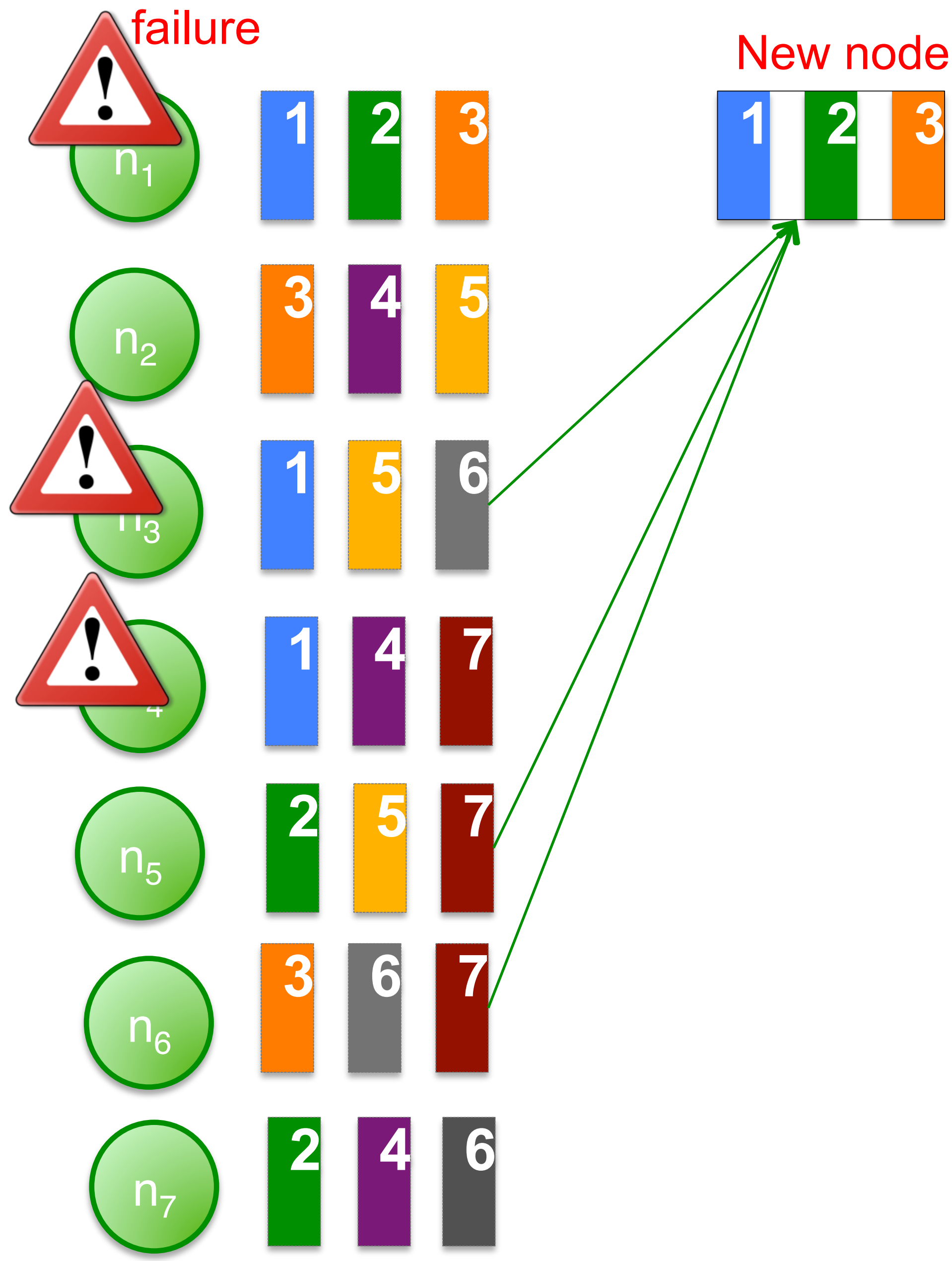



- ❖ Repetition r=3➔can tolerate up to 2 simultaneous failures
- ❖ Table based repair
- If 3 nodes fail?
  - Download complete file
  - Reconstruct the lost data
  - Rare bad even

✓ Min repair bandwidth

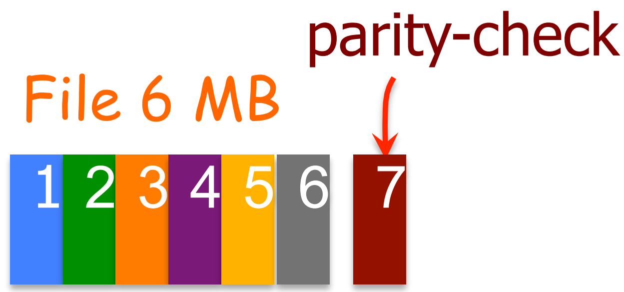
✓ Min disk I/O calls

✓ Uncoded repair

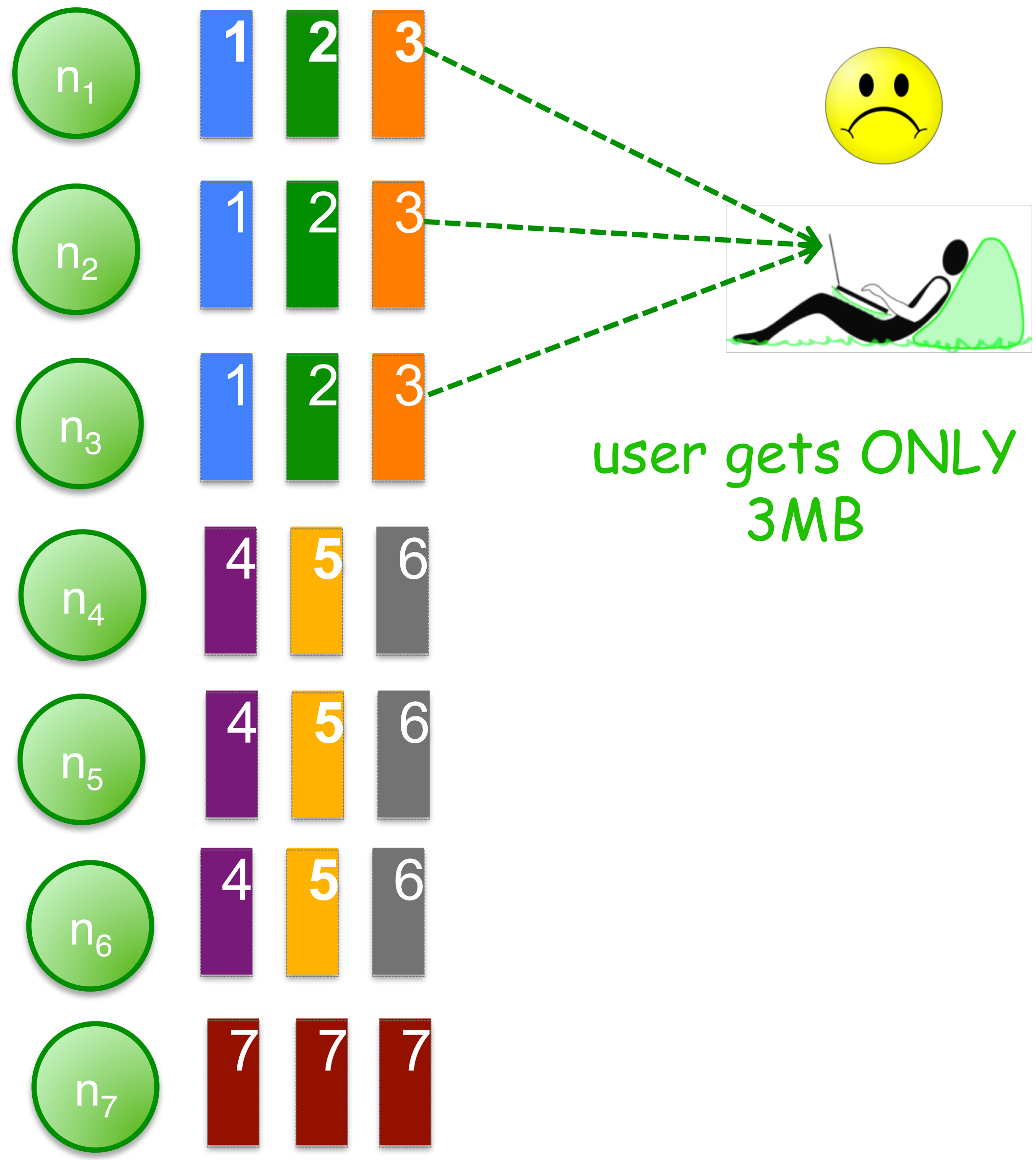


# Why “Traditional” Repetition Won’t Work?

Max file size= 6MB: cut-set bound (Dimakis et al.)




(n,k)=(7,3)
repetition factor r=3



# Smart Packet Placement

How should we place the packet replicas?
Goal: maximize file size

Max file size: $C = kd$

(Dimakis et al.)

# nodes contacted by user

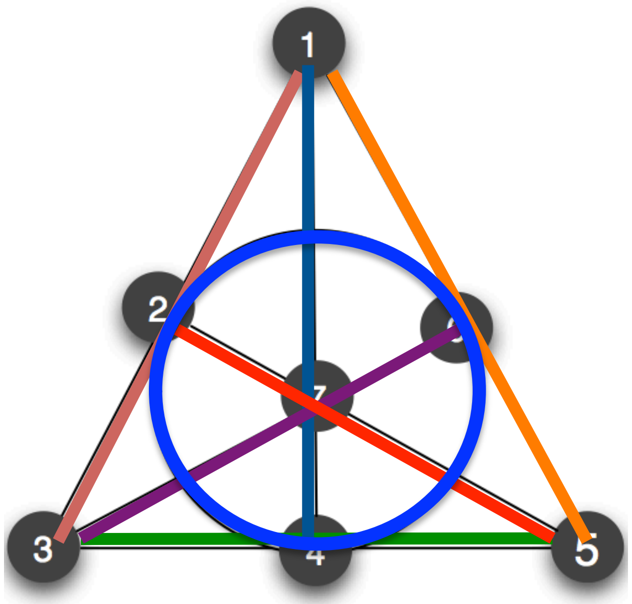
# packets/node

$$\frac{k(k-1)}{2} = \binom{k}{2}$$

**Rule:** Make sure that any two nodes have at most one packet in common

# Where Did That Solution Come From?

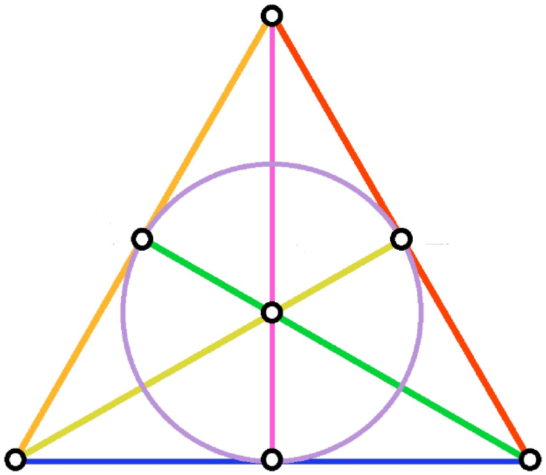
Lines in projective planes intersect in exactly one point



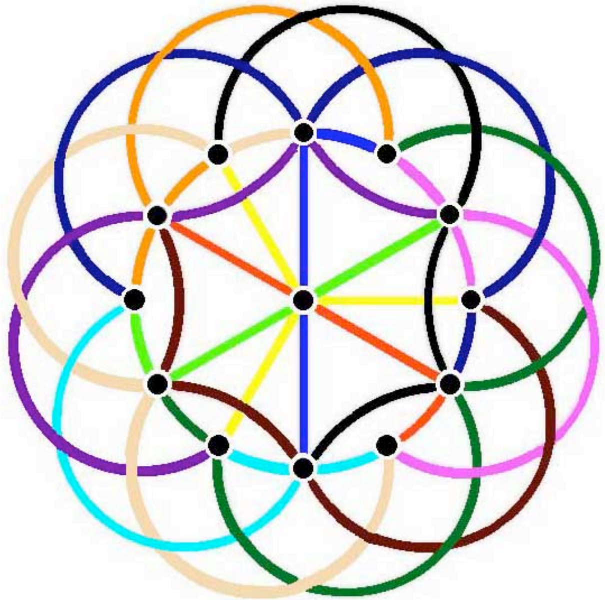

Fano Plane
Projective plane of order 2

Points → Packets
Lines → Storage nodes

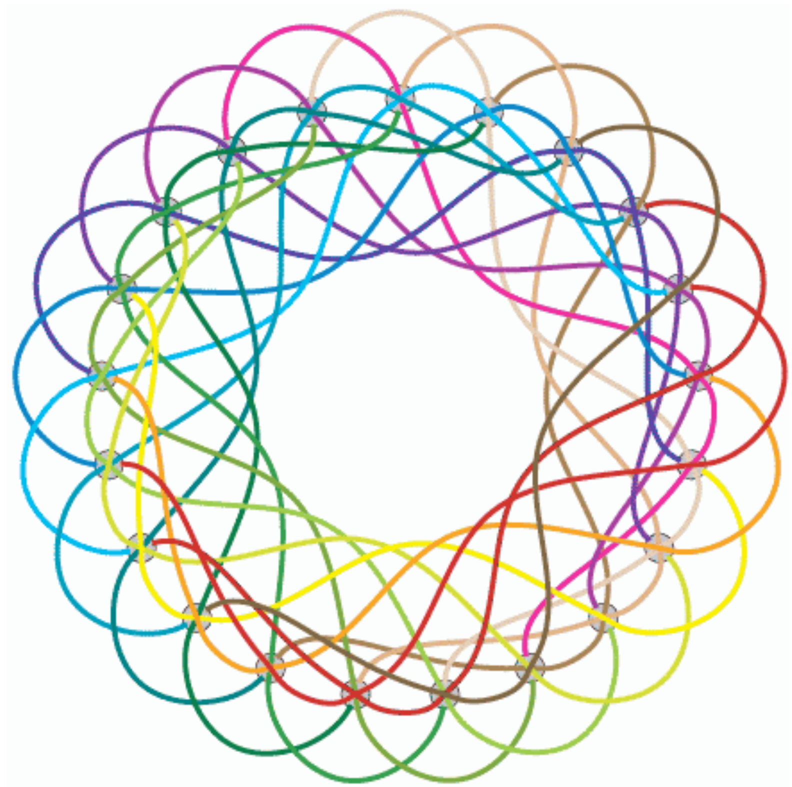
# Higher Order Projective Planes



order m=2



m=3

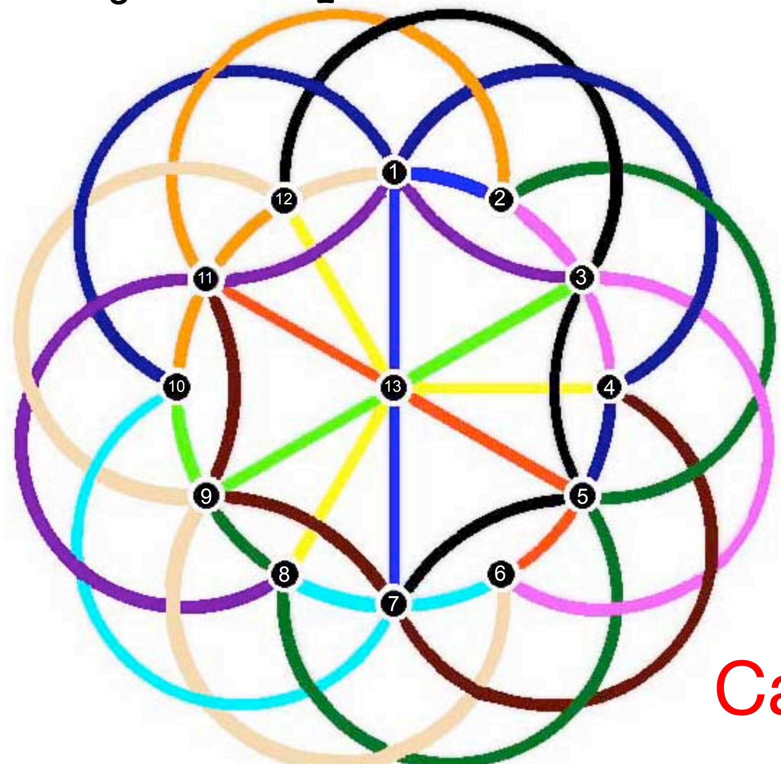


m=4

- Projective planes are guaranteed to exist for any prime power order m.

# Codes from Projective Planes

Projective plane of order m=3




n=13, r=d=4

| Projective Plane | Storage System |
|---|---|
| Lines | Storage nodes |
| Points | Packets |

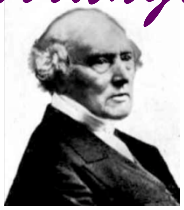
Can we have constructions spanning more system parameters?

**Theorem 1** (R. & Ramchandran Allerton'10)

A projective plane of *order m* gives DRESS codes with $n=m^2+m+1$ nodes and repetition factor r=repair degree d = m+1.

# Kirkman's Schoolgirls Problem

*"Fifteen young ladies in a school walk out three abreast for seven days in succession; it is required to arrange them daily so that no two shall walk twice abreast."*



Kirkman, 1847

- Fifteen schoolgirls: A, B,…, O
- They always take their daily walks in groups of threes

| | group 1 | group 2 | group 3 | group 4 | group 5 |
|---|---|---|---|---|---|
| **Sun** | ABC | DEF | GHI | JKL | MNO |
| **Mon** | ADH | BEK | MOI | FLN | GJC |
| ⋮ | ? | ? | ? | ? | ? |
| **Sat** | ? | ? | ? | ? | ? |

- Arrange them such that any two schoolgirls walk together exactly once a week

# Why Do We Care?

Storage node

A solution to Kirkman's problem

| | | | | | |
|---|---|---|---|---|---|
| **Sun** | ABC | DEF | GHI | JKL | MNO |
| **Mon** | ADH | BEK | CIO | FLN | GJM |
| **Tue** | AEM | BHN | CGK | DIL | FJO |
| **Wed** | AFI | BLO | CHJ | DKM | EGN |
| **Thu** | AGL | BDJ | CFM | EHO | IKN |
| **Fri** | AJN | BIM | CEL | DOG | FHK |
| **Sat** | AKO | BFG | CDN | EIJ | HLM |

7 days

5 groups

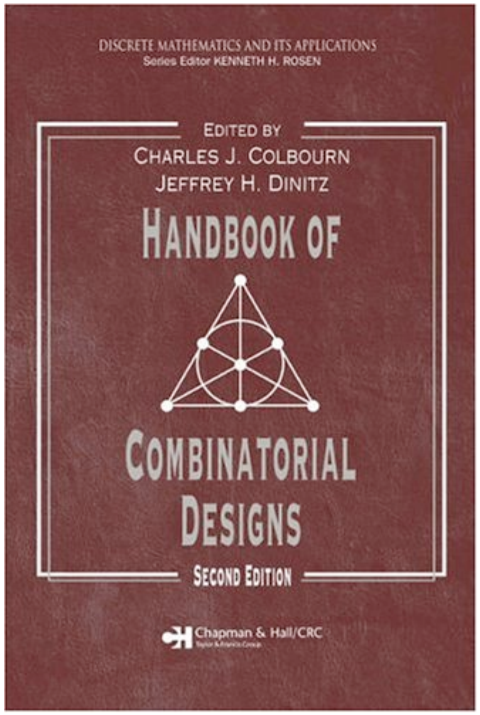
- Think of schoolgirls as packets and groups as storage nodes
- Kirkman's solution gives an optimal DRESS code for a
  - ✓ System with n=35 nodes
  - ✓ Each node stores 3 packets
- What is the repetition factor here? answer=7

# Not Your Average Puzzle…

- Kirkman's schoolgirls problem initiated a branch of combinatorics known now as *Design Theory*

- Deep connections to other branches in mathematics: Algebra, Geometry, matroid theory…

- Applications:
  - Coding Theory
  - Cryptography
  - Statistics

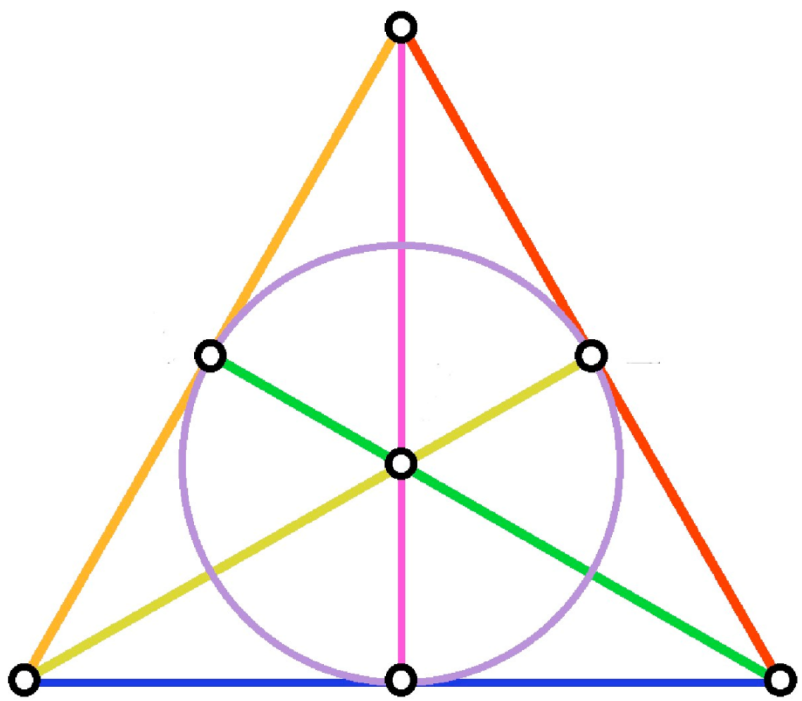


# Steiner Systems

A Steiner system $S(t, \alpha, v)$ is a collection of points & lines such that:

1. There are $v$ points in total
2. Each line contains exactly $\alpha$ points
3. There is exactly one line that contains any given $t$ points

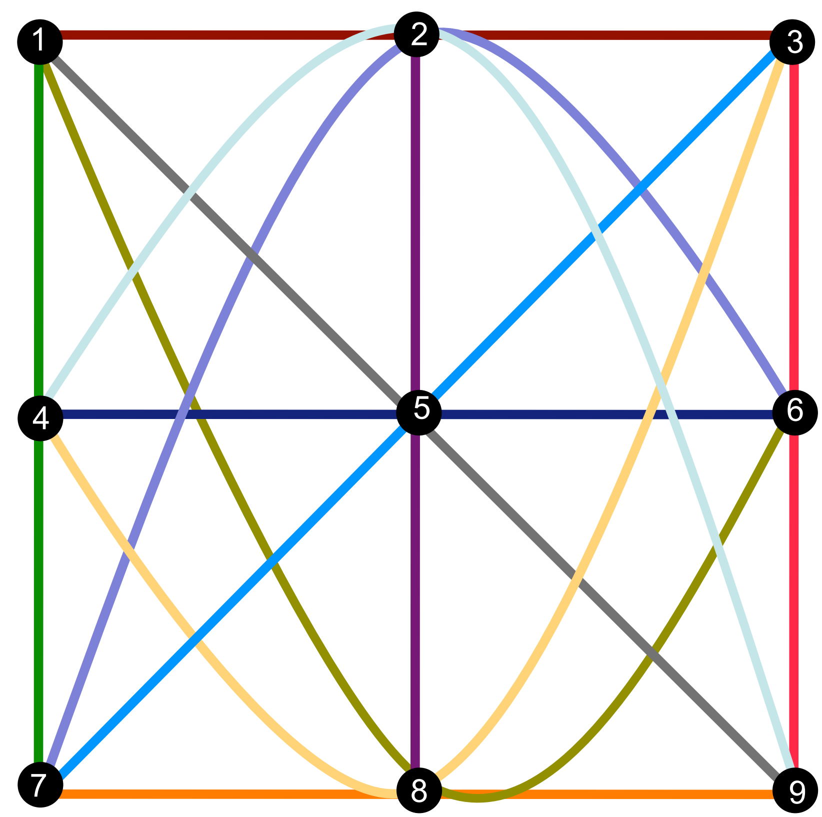
We want t=2



Fano plane =S(2, 3, 7)

# pts on a line

Total # points



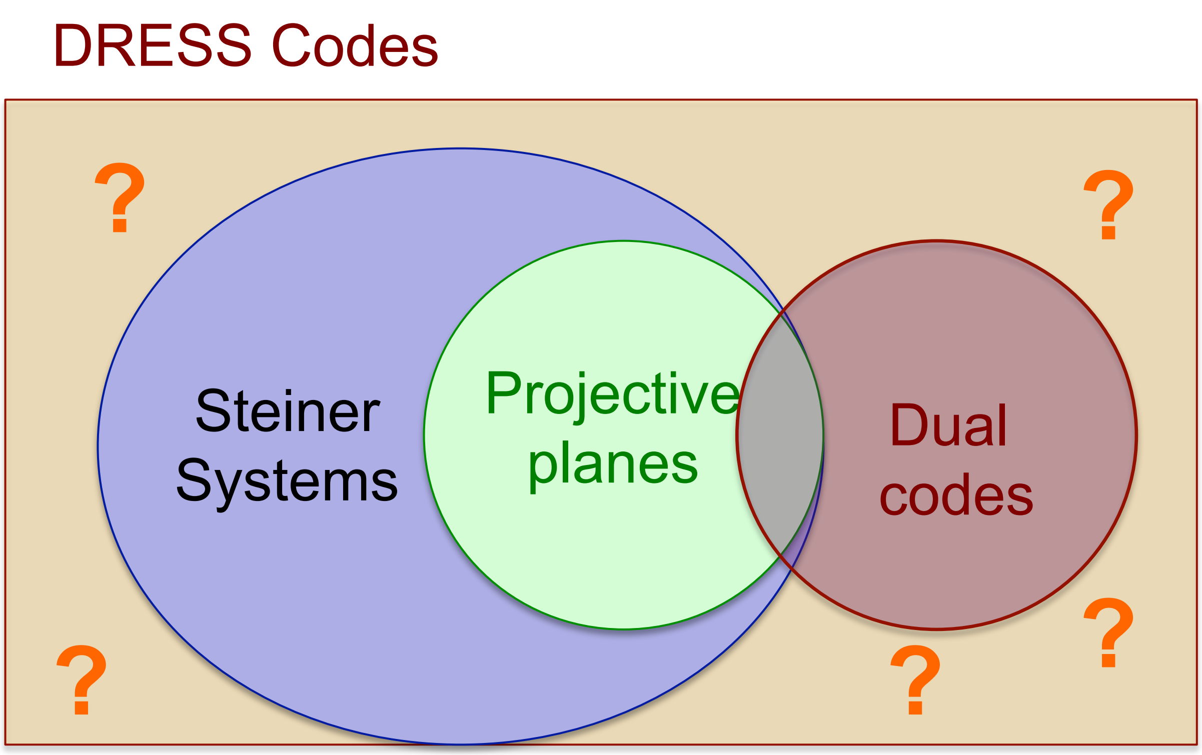

S(2,3,9)

# Codes from Steiner Systems

**Theorem 2:** (R. & Ramchandran Allerton'10)

A Steiner system $S(2, \alpha, v)$ gives a capacity-achieving DRESS code with parameters $r = \frac{v-1}{\alpha-1}$ $\quad n = \frac{rv}{\alpha}$

using correspondence lines: nodes, points: packets.




**Open problem I**

DRESS codes beyond Steiner systems?

# Min Reads Capacity

Min Bandwidth Capacity

Minimum reads

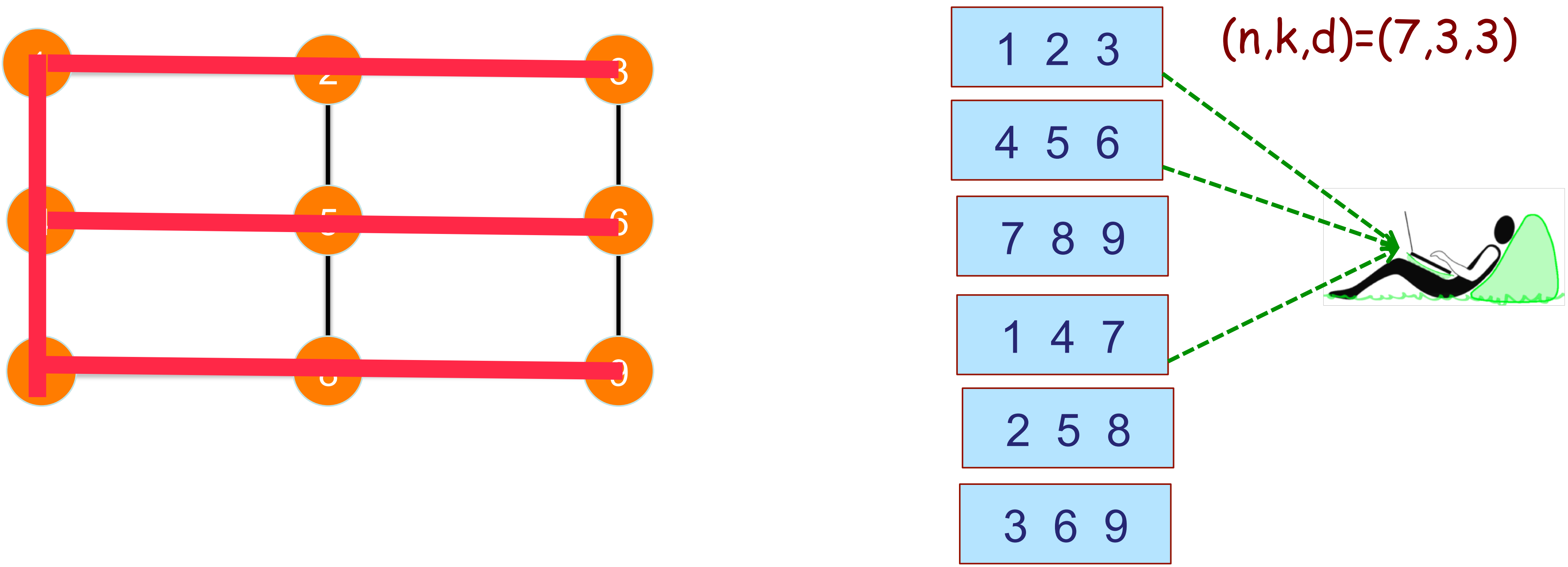
$$\boxed{C = k \times d - \binom{k}{2}}$$

Lookup Repair

**Open problem II**

Capacity for minimum reads (and lookup repair)?

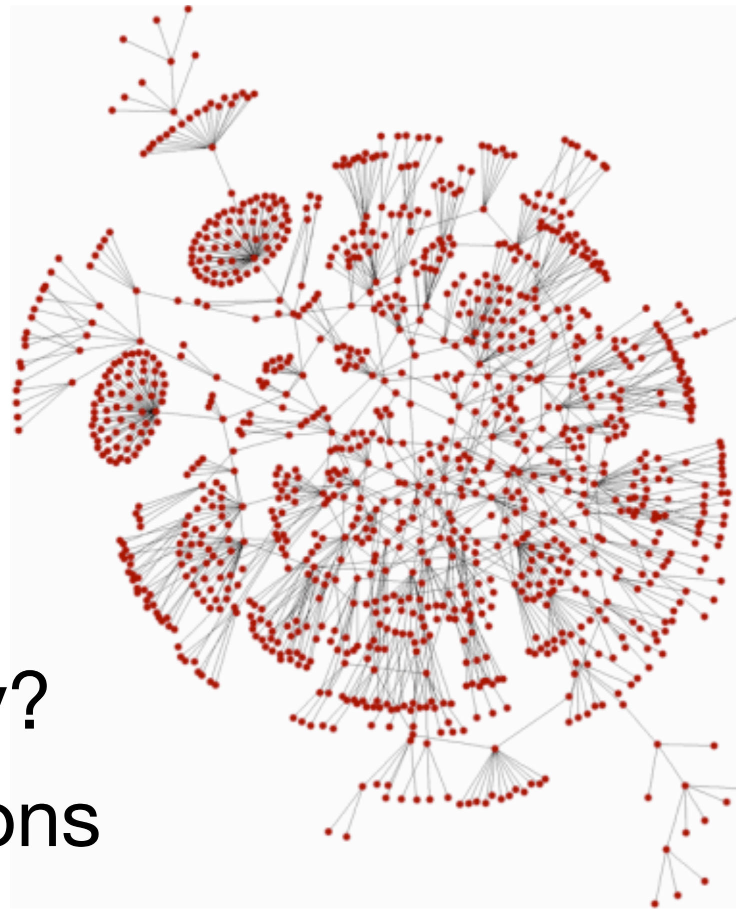
# Example




- Users contacting any 3 nodes will observe either 9 or 7 packets
- This code guarantees a “rate” = 7 to any user
- Capacity when repairing from any d nodes = 6

# We Want More

- Explicit constructions for Steiner systems exist for small values of r=3,4,5
- Existential guarantees for large number of nodes [Wilson 75]
- Want more flexibility
- Want to “grow”. Nested property?
- Solution: randomized constructions

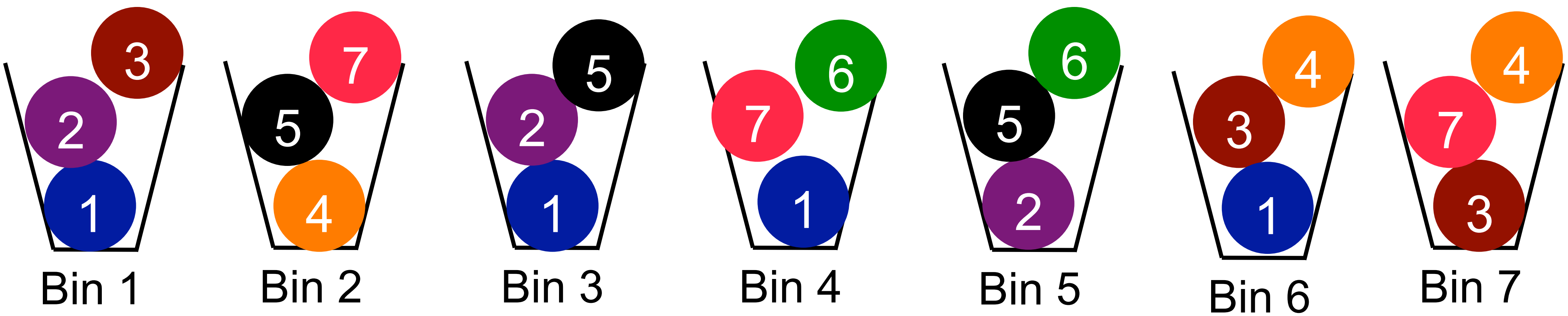


P2P network

# Lazy Man's Codes

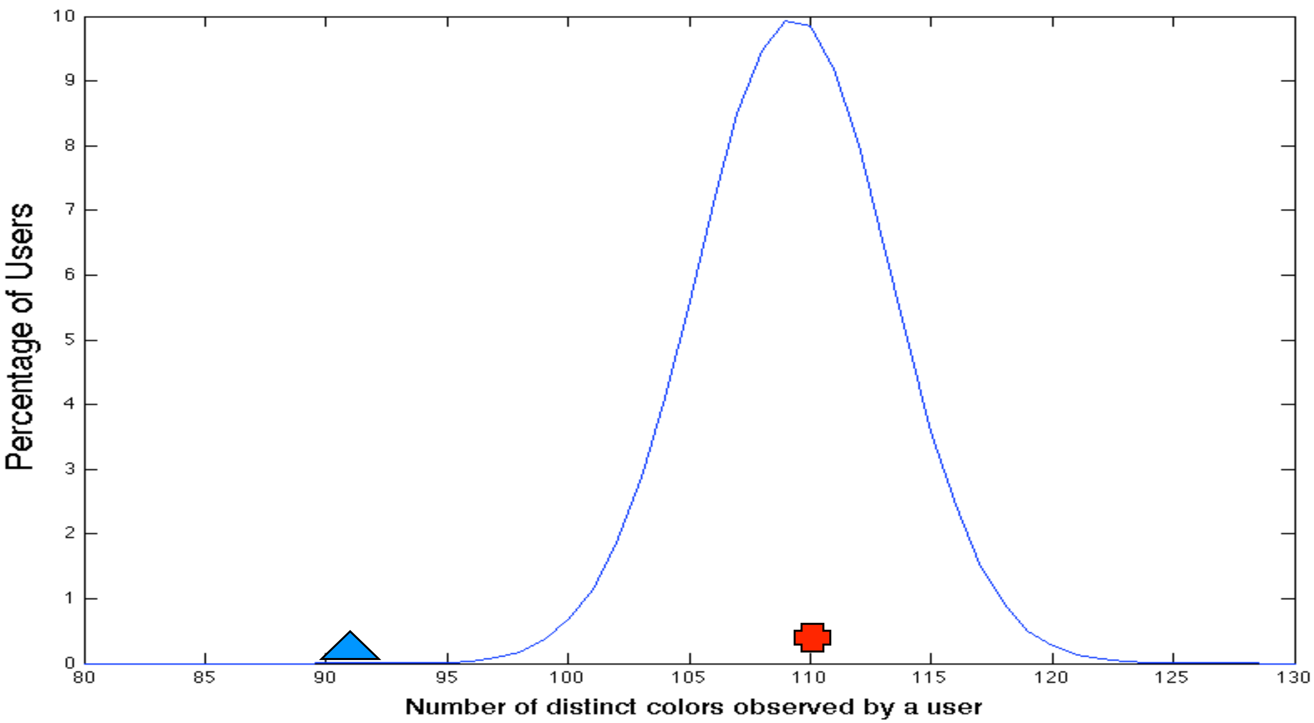

(n,k)=(7,3)
repetition factor r=3
File Size=6

*Growing*


- Fill the bins randomly by throwing 3 independent colors in each
- User takes a hit on the file size if he contacts bins that have two many colors in common
- Replication factor is random and can be less than 3
- Hope the bad events are rare

# It Pays To Be Lazy




(n,k)=(169,13); r=13

- Deterministic approach:
  cut-set bound = 91

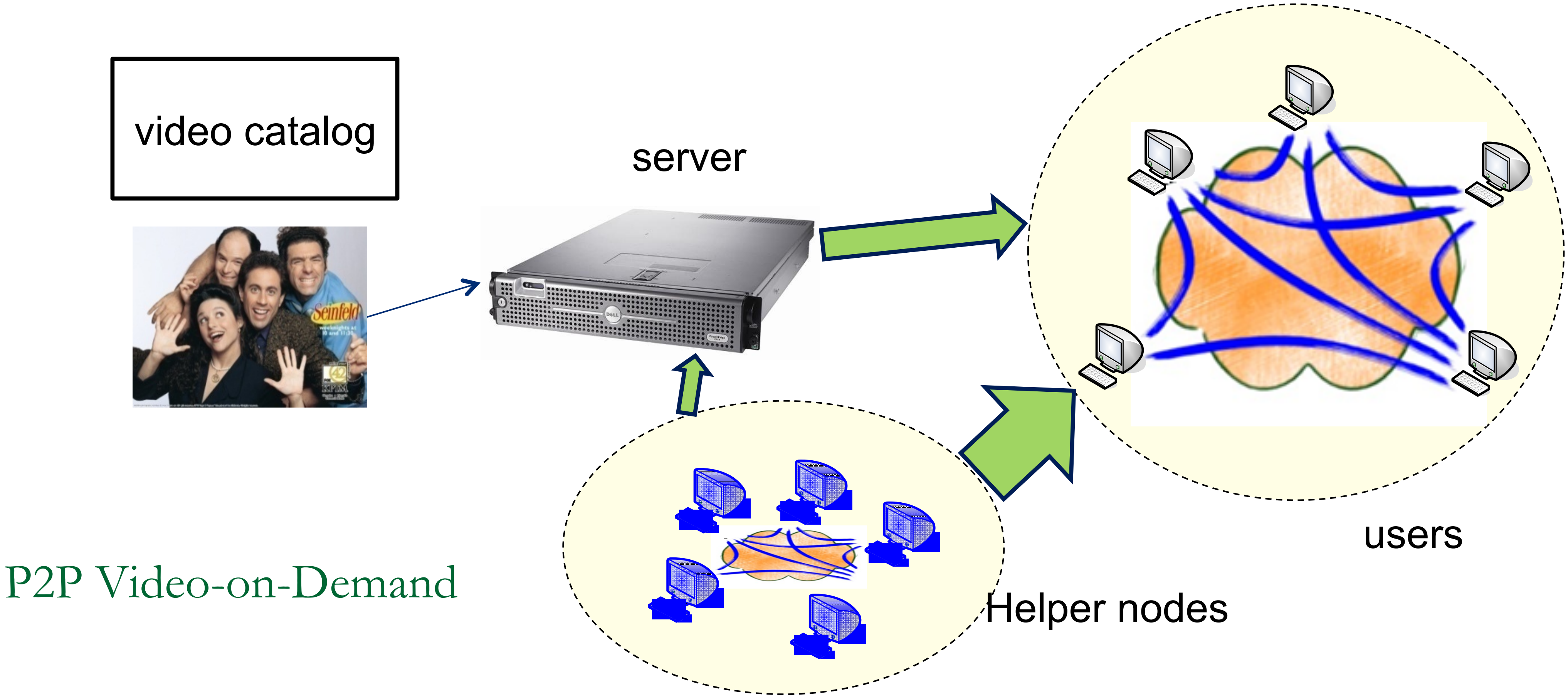
- Statistical approach:
  mean $\bar{F}$ =110

<u>Theorem 3:</u>(Concentration)
Let F be the number of colors observed by a user contacting k nodes then

$$P(|F-\bar{F}|\geq B)\leq 2\exp(-\frac{B^2}{2kd})$$

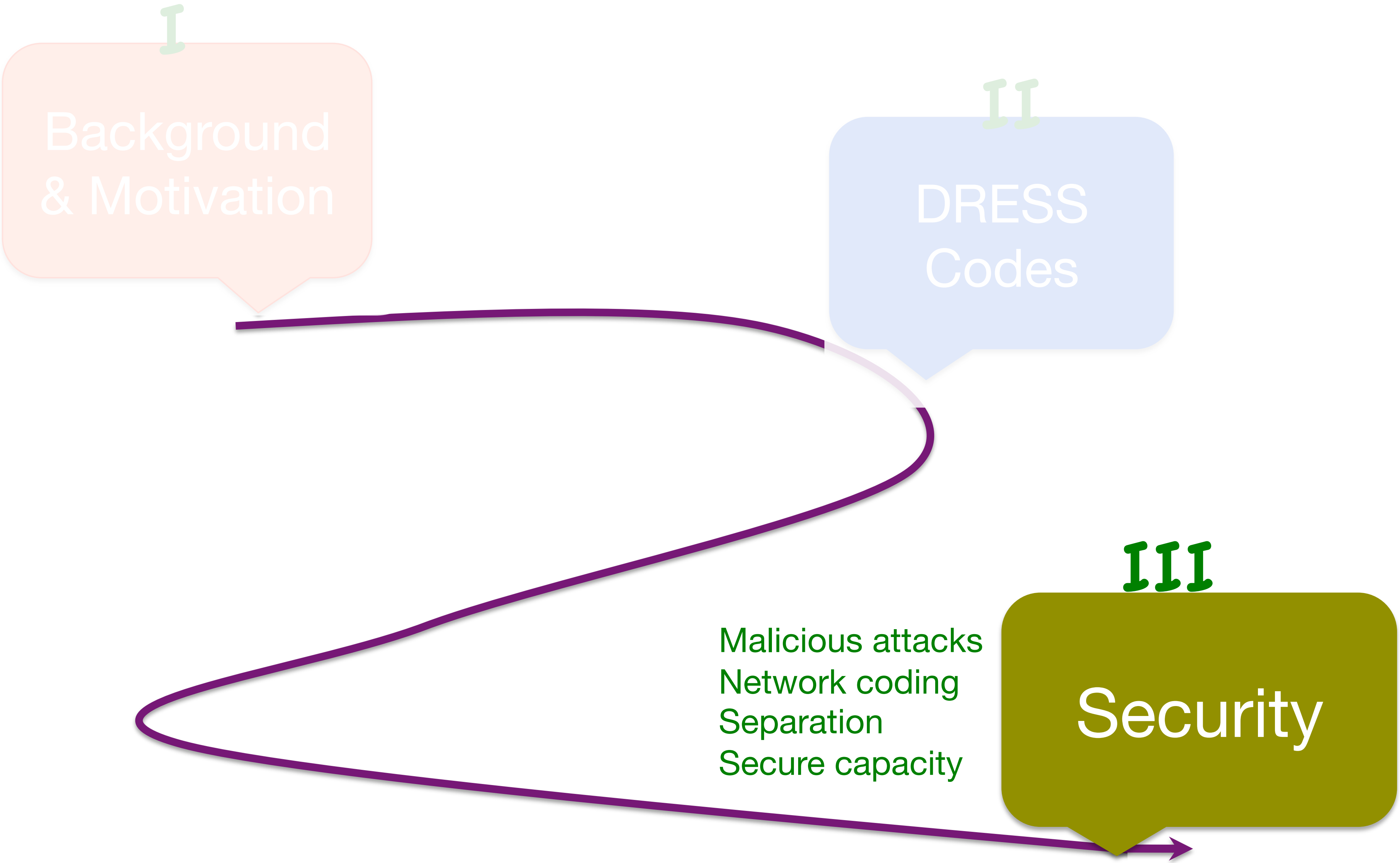
with $\bar{F}=\theta\left(1-(1-\frac{1}{\theta})^{kd}\right), \theta=nd/r.$

# Content Distribution with Helpers




- Examples of helper include:
  - ✓ idle Internet PC users: Thunder, PPStream, PPLive, Wuala etc…
  - ✓ set-top boxes, home gateways: Nano data centers
  - ✓ ISP P2P infrastructure nodes: Comcast
- Goal: minimize server load or maximize a utility function

# Talk Roadmap



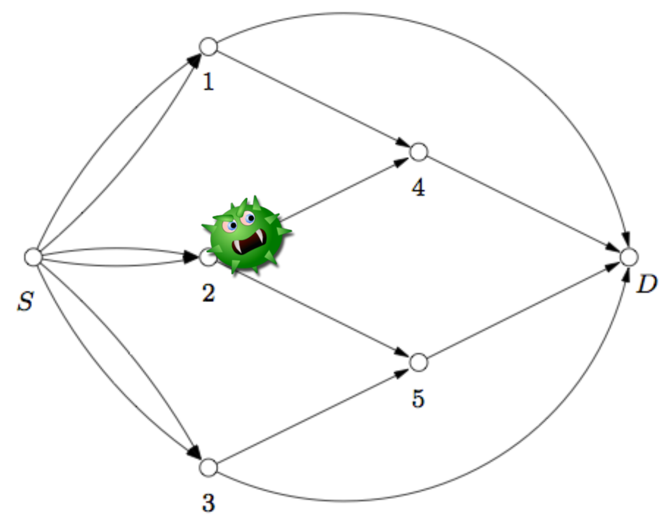
# Security, Security, Security

- How to guarantee data privacy and integrity?
- What are the fundamental limits on cloud security?

- Computationally unbounded intruder
- Two classes of intruders:
    - Passive: eavesdropper
    - Active: Malicious attacks

# Malicious Byzantine Nodes

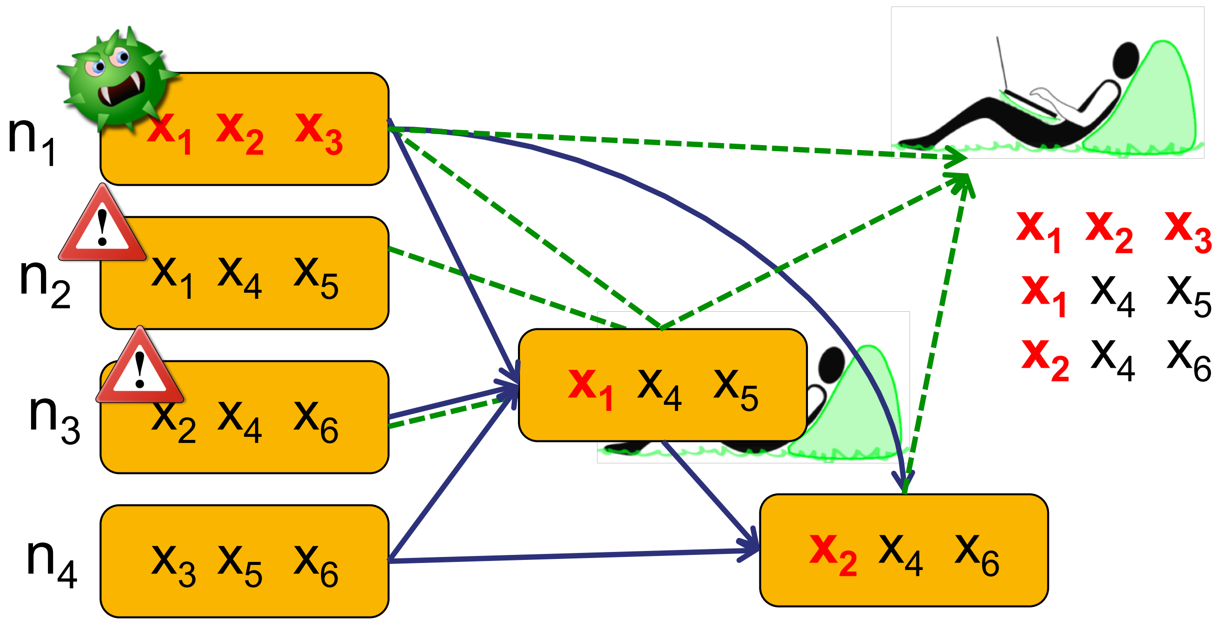
- Hard problem related to open problems in network coding. ( Kosut, Tong & Tse ’10)
- Non-linear coding, secure capacity still an open problem…
- “Node vs. Edge Curse” known in the network coding literature
- How about the cloud?




Cockroach network

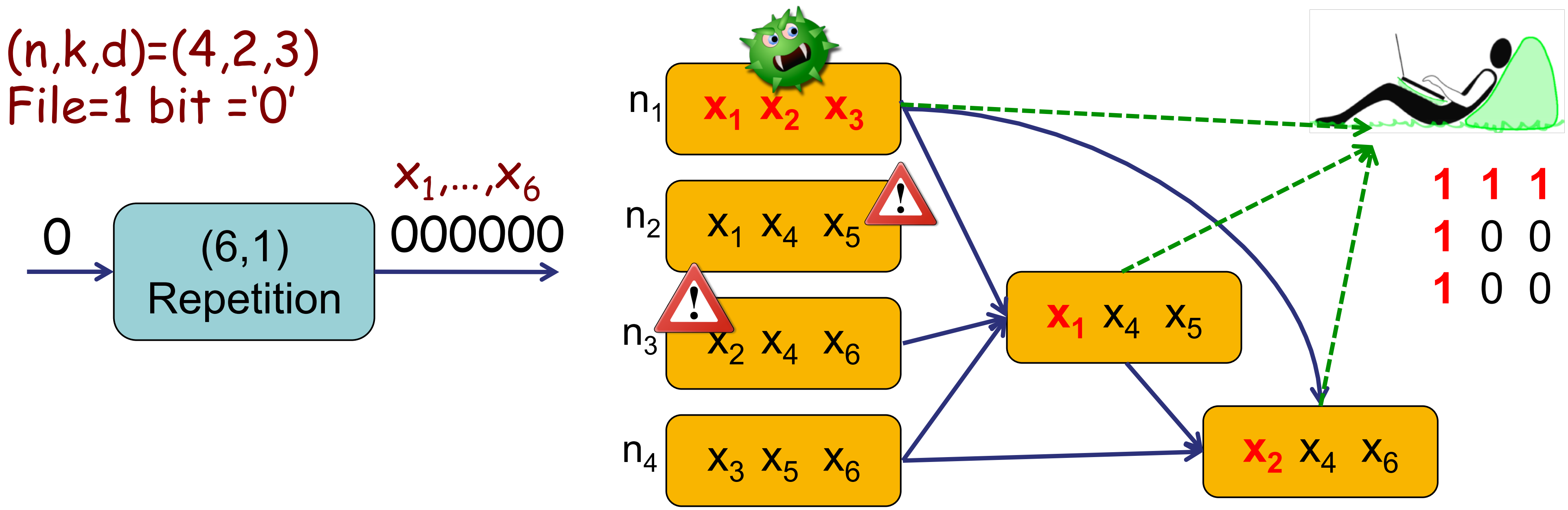
# Why Is It a Hard Problem?

(n,k)=(4,3)
File Size=6 bits
File: $x_1,\ldots,x_6$



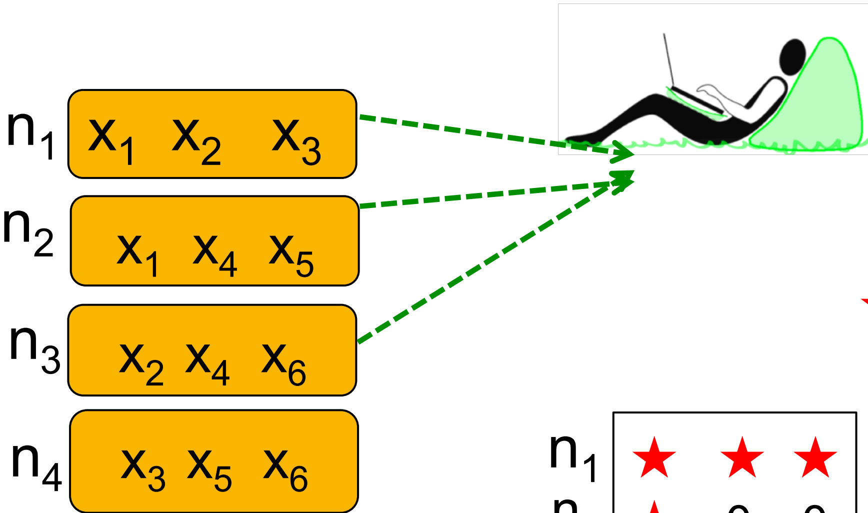

- Even a single malicious node can contaminate the whole system
- Although controlling 3 bits, the malicious node can introduce 5 errors
- Can we at least store 1 bit securely?

# No Bit Left Behind




- Repeat the 1 bit information on all the nodes
- 5 errors out of 9: simple majority decoding won't work.
- Idea: not any 5 errors can occur
- DRESS code forces errors to have certain patterns

# Tetris Decoder




★: possible error

$$\begin{array}{c}n_1\\n_2\\n_3\end{array}\begin{bmatrix}\star & \star & \star\\ \star & 0 & 0\\ \star & 0 & 0\end{bmatrix}\quad\begin{bmatrix}\star & 0 & 0\\ \star & \star & \star\\ 0 & \star & 0\end{bmatrix}\quad\begin{bmatrix}0 & \star & 0\\ 0 & \star & 0\\ \star & \star & \star\end{bmatrix}\quad\begin{bmatrix}0 & 0 & \star\\ 0 & 0 & \star\\ 0 & 0 & \star\end{bmatrix}$$

Bad node: $n_1$ $\quad$ $n_2$ $\quad$ $n_3$ $\quad$ $n_4$

- Decoder checks the error pattern and decode
- It will always decodes to the correct bit
- Bonus: can catch the malicious nodes

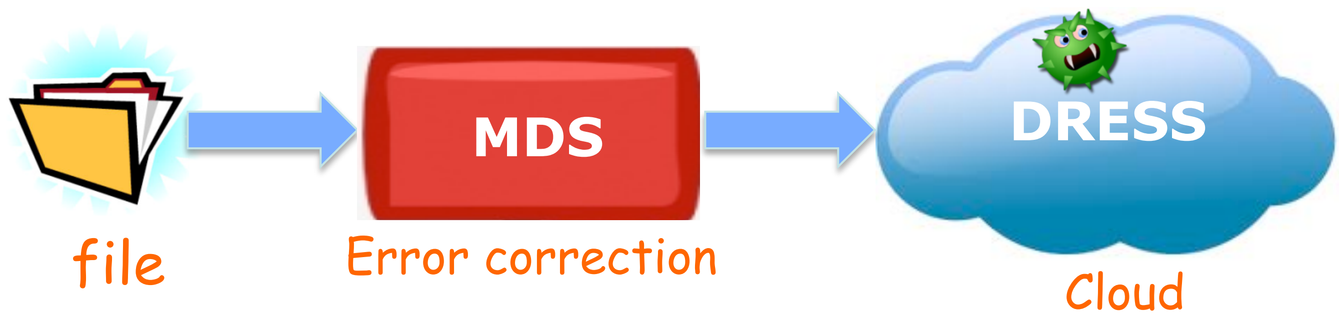
# Omniscient Adversary: Main Result

Theorem 4: Open Open

For the MBR regime and d=n-1, the secrecy capacity is given by (Network Singleton bound):

$$C_{sec} = \sum_{i=2b+1}^{k} (d - i - 1).$$

# malicious nodes




Separation scheme

- \+ Linear outer code. Internal nodes in the cloud operate as usual.
- \- Decoder: exponential time in number of malicious nodes

Open

# Conclusion

- Cloud storage: guarantee reliability using unreliable nodes!
- What codes should we use? storage, bandwidth, Disk I/O, computational complexity, latency, energy, security…

- DRESS codes: uncoded repair and efficient codes
- Deterministic constructions from Proj. Spaces & Steiner sys.
- Randomized constructions using balls & bins
- Separation result for achieving security

- Open problems: Code constructions for more system parameters, "Smart" random constructions for better concentration, general secure capacity…