Information Theoretic Security of Big Data In Distributed Storage Systems

Rawad Bitar, Salim El Rouayheb

ILLINOIS INSTITUTE OF TECHNOLOGY

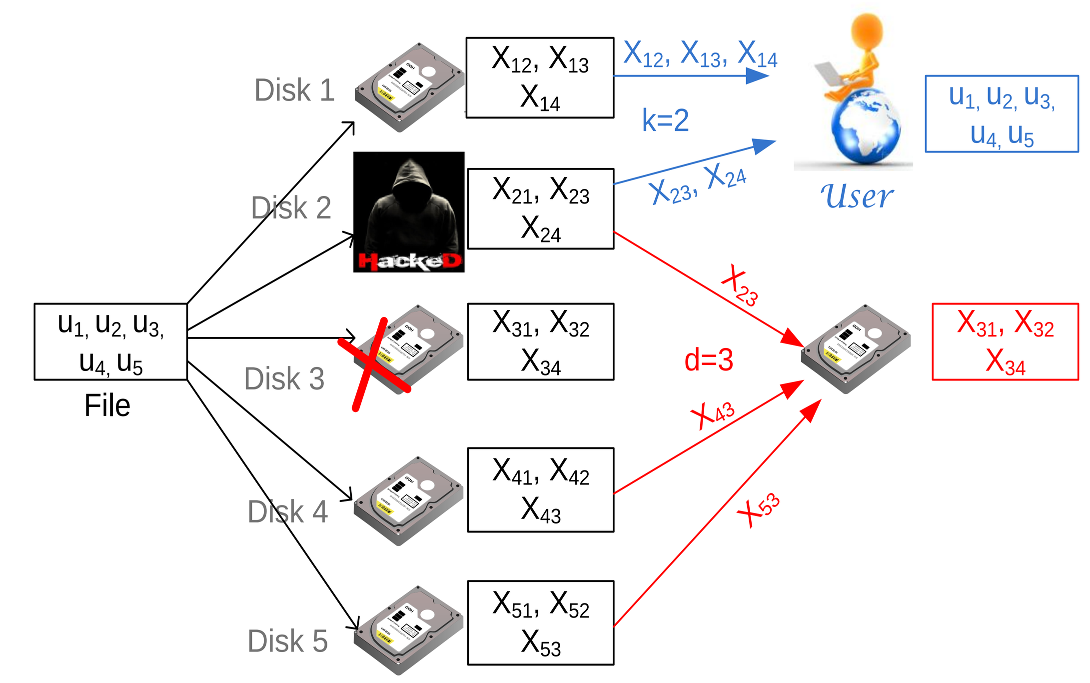
ece@iit

Distributed Storage Systems

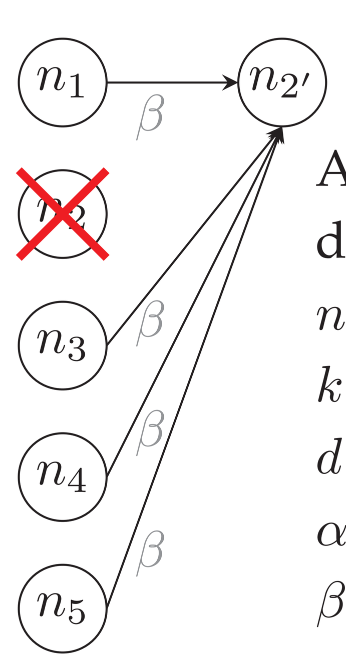
Distributed storage systems (DSS) consist of storing data on n individually unreliable disks out of which any k are needed to reconstruct the stored data. In order to repair a failed disk, d disks are contacted.

Applications of distributed storage systems include large data centers and peer-to-peer storage systems, that use a large number of disks spread across the internet.

To minimize the storage cost, coding was introduced to DSS, which brought new tradeoffs and problems. For example by minimizing per disk storage, the bandwidth used to repair a failed disk increases. Trying to secure the system against adversarial attacks, the maximum file size is reduced.



Regenerating Codes



An (n, k, d)-DSS storing data file $\mathcal{F}$ of M symbols in $\mathbb{F}_{q^v}$

$n = 5$: total # of nodes

$k = 3$: min # of nodes to reconstruct

$d = 4$: # of helper nodes to repair

α: storage per node

β: repair bandwidth

Regenerating codes is a family of codes introduced in [2]. They have following appealing properties:

Reconstruction property: *any* k out of n disks can recover $\mathcal{F}$.

Exact repair property: *any* d out of n disks can repair a failed disk by sending data less than M.
exact repair $\implies$ Recover an exact copy of the lost data.

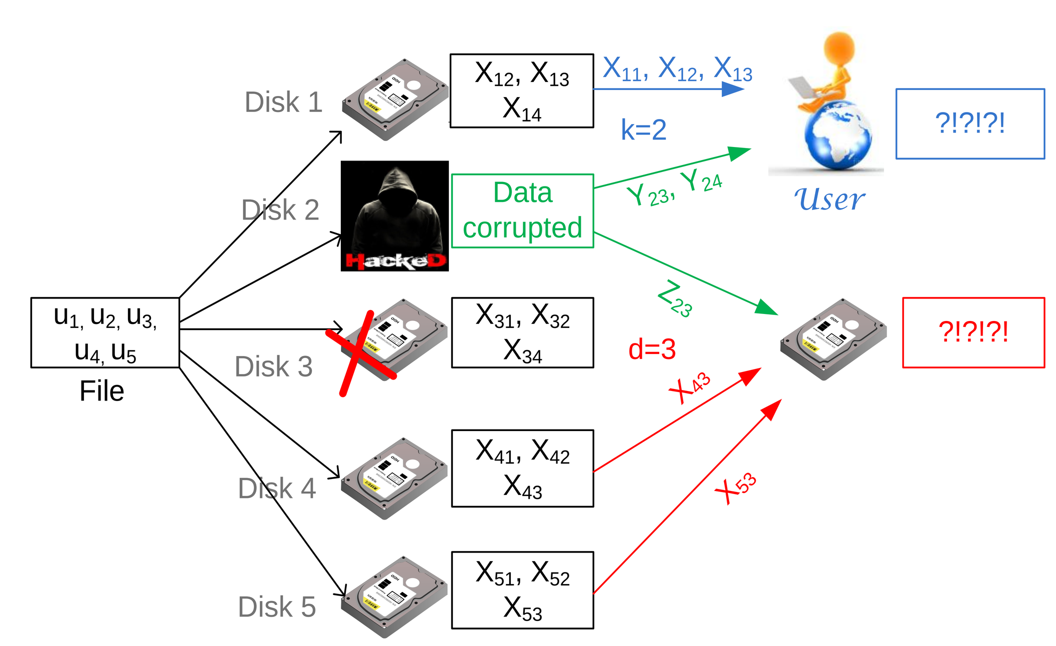
Optimal repair bandwidth [2]:

$$\beta = \frac{\alpha}{d} = \frac{\alpha}{4}.$$

In contrast to erasure codes such as Reed-Solomon, regenerating codes allow the system to repair a disk failure by downloading $d\beta < M$ symbols.

Security Problem

The adversary can observe and possibly corrupt the data on b nodes. If observing a replacement node, say n_2', he can observe the repair data used to replace n_2.



Resiliency capacity:

Given an (n, k, d)-DSS with b compromised nodes, its *resiliency capacity* $C_r(\beta)$ is defined to be the maximum file size that can be stored in the DSS, such that the reconstruction and the repair property will simultaneously hold.

Upper bound [3]

$$C_r \leq \sum_{i=b+1}^{k} \min\left(\alpha, (d-i+1)\beta\right).$$

Perfect secrecy: In some applications we also want to hide the data from the adversary.

Results

Theorem 1 *[1] An (n, k, d) regenerating code, operating in the bandwidth-limited regime, can be made resilient (with a small probability of error upper-bounded by $\frac{1}{q}$) against an active limited-knowledge adversary controlling $b < \frac{k}{2}$ nodes; and achieves with equality the resiliency capacity*

$$C_r \leq \sum_{i=b+1}^{k} min\{\alpha, (d-i+1)\beta\}.$$

Theorem 2 *The resilient (n, k, d) regenerating code can be made secure against Eavesdroppers controlling a subset l of the b nodes, achieving a maximum file size*

$$M_{rs} \leq \sum_{i=l+b+1}^{k} \min\left\{\alpha, (d-i+1)\beta\right\}.$$

Capacity Achieving Code Construction

To secure an $(n, k, d) = (5, 3, 4)$ DSS we build an $(n, k-b, d-b) = (5, 2, 3)$ DSS, transform it into uncoded-repair. By contacting b more nodes and detecting the corrupted data using a hashing scheme we ensure a safe reconstruction and repair.

Transforming the PM construction [4] into uncoded repair

disk 1	X_{12}	X_{13}	X_{14}	$\{X_{15}\}$
disk 2	X_{21}	X_{23}	X_{24}	$\{X_{25}\}$
disk 3	X_{31}	X_{32}	X_{34}	$\{X_{35}\}$
disk 4	X_{41}	X_{42}	X_{43}	$\{X_{45}\}$
disk 5	X_{51}	X_{52}	X_{53}	$\{X_{54}\}$

Hash function as introduced in [3]: By abuse of notation we look at $X_{ij} \in \mathbb{F}_{q^v}$ as a vector $X_{ij} \in \mathbb{F}_q^v$. We compute the dot product $X_{ij}.X_{lk} = \sum_{s=1}^{v} X_{ij}^s X_{lk}^s$ for $ij \neq lk$ where $X_{ij}^s \in \mathbb{F}_q$ and store them on a trusted server that can not be controlled by the adversary.

Adversary detection during disk repair:

	X_{12}	X_{32}	X_{42}	X_{52}
X_{12}		×	✓	✓
X_{32}	×		×	×
X_{42}	✓	×		✓
X_{52}	✓	×	✓	

Adversary detection during file reconstruction:

		disk 3			disk 2		
		X_{31}	X_{32}	X_{34}	X_{21}	X_{23}	X_{24}
disk 1	X_{12}	×	×	×	✓	✓	✓
	X_{13}	✓	✓	✓	✓	✓	✓
	X_{14}	×	×	×	✓	✓	✓
disk 2	X_{21}	×	×	×			
	X_{23}	✓	✓	✓			
	X_{24}	×	×	×			
disk 3	X_{21}				×	✓	×
	X_{23}				×	✓	×
	X_{24}				×	✓	×

References

[1] R. Bitar, S. El Rouayheb, "Securing data against Limited-Knowledge Adversaries in Distributed Storage Systems," accepted in *IEEE International Symposium on Information Theory*, 2015.

[2] A. Dimakis, P. Godfrey, Y. Wu, M. Wainright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Transactions on Information Theory*, vol. 56, no. 9, pp. 4539–4551, 2010.

[3] S. Pawar, S. El Rouayheb and K. Ramchandran, "Securing Dynamic Distributed Storage Systems Against Eavesdropping and Adversarial Attacks," *IEEE Transactions on Information Theory*, Volume:57 , Issue: 10, Oct 2011.

[4] K. V. Rashmi, N. B. Shah, and P. V. Kumar, "Optimal Exact-Regenerating Codes for Distributed Storage at the MSR and MBR Points via a Product-Matrix Construction," *IEEE Transactions on Information Theory*, Volume:57 , Issue: 8 , Aug 2011.

[5] K. V. Rashmi, N. B. Shah, K. Ramchandran and P. V. Kumar, "Regenerating Codes for Errors and Erasures in Distributed Storage," Proceedings of *IEEE International Symposium on Information Theory*, Jul 2012.