R-Drive: Resilient Data Storage and Sharing for Mobile Edge Clouds

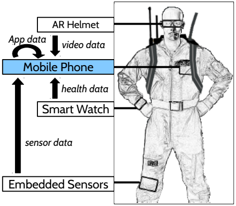
Mohammad Sagor Radu Stoleru Amran Haroon Suman Bhunia Mengyuan Chao Ala Altaweel Maxwell Maurice[†] Roger Blalock[†]

Texas A&M University, Computer Science and Engineering, College Station, TX
[†]National Institute of Science and Technology (NIST), Boulder, CO

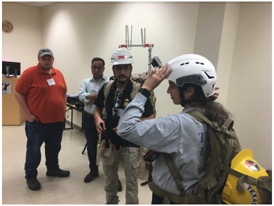
The 19th IEEE International Conference on Mobile AdHoc and Smart Systems (MASS), 2022

Outline

Mobile Edge Clouds for Next Generation Disaster Response



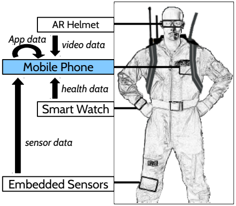
(a) Next-Gen First Responders



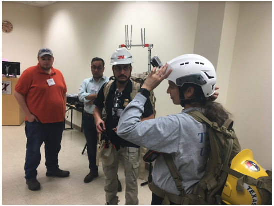
(b) Rescue Team A

(c) Rescue Team B

Mobile Edge Clouds for Next Generation Disaster Response



(a) Next-Gen First Responders



(b) Rescue Team A

(c) Rescue Team B

Mobile Edge Cloud Advantages:

- Existing applications work in the absence of network and cloud infrastructures
- Energy savings stemming from local processing when compared with cloud processing
- Lower application latencies when compared with the cloud

An Architecture for Mobile Edge Clouds (MEC)

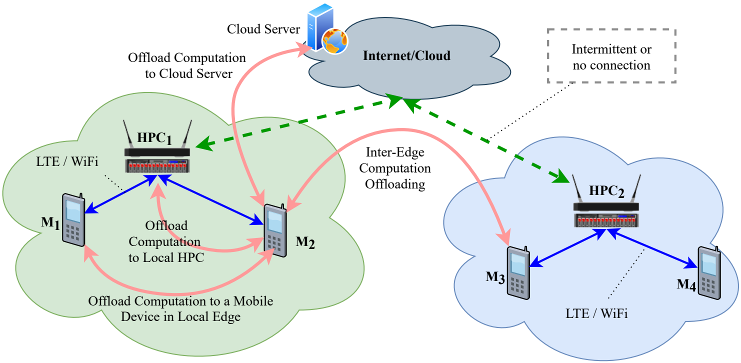


Figure: MEC (M_1, ..., M_4, HPC_1, HPC_2) offload computation intra-edge, inter-edge and to the cloud

The Needs, Research Challenges and Contributions

Disaster response/tactical applications generate gigabytes of mission-critical and personal data that needs to be readily available for seamless processing.

The Needs, Research Challenges and Contributions

Disaster response/tactical applications generate gigabytes of mission-critical and personal data that needs to be readily available for seamless processing.

Research Questions

- How to ensure reliability of data stored?
- How to efficiently use MEC storage space and communication?
- How to ensure privacy and integrity protection of data stored?
- How to leverage existing MEC infrastructures?

The Needs, Research Challenges and Contributions

Disaster response/tactical applications generate gigabytes of mission-critical and personal data that needs to be readily available for seamless processing.

Research Questions

- How to ensure reliability of data stored?
- How to efficiently use MEC storage space and communication?
- How to ensure privacy and integrity protection of data stored?
- How to leverage existing MEC infrastructures?

Proposed Solution

- R-Drive, a resilient data store, implemented and evaluated in a real system
- An Adaptive Erasure Coding mechanism, suitable for dynamic MEC
- A seamless data sharing solution for existing cloud-based applications

Disconnection Tolerant Storage

- CODA [TOCS'92]
- Not resilient; store and forward mechanism during disconnection

Disconnection Tolerant Storage

- CODA [TOCS'92]
- Not resilient; store and forward mechanism during disconnection

Distributed Storage for Cloud

- GFS [SOSP'03], HDFS [MSST'10]
- Heavyweight, designed for cloud

Disconnection Tolerant Storage

- CODA [TOCS'92]
- Not resilient; store and forward mechanism during disconnection

Distributed Storage for Cloud

- GFS [SOSP'03], HDFS [MSST'10]
- Heavyweight, designed for cloud

Distributed Storage for Android

- Hyrax [CMU'09]
- Still heavyweight, due to simple code porting, low performance

Disconnection Tolerant Storage

- CODA [TOCS’92]
- Not resilient; store and forward mechanism during disconnection

Distributed Storage for Cloud

- GFS [SOSP’03], HDFS [MSST’10]
- Heavyweight, designed for cloud

Distributed Storage for Android

- Hyrax [CMU’09]
- Still heavyweight, due to simple code porting, low performance

Mobile Edge Storage

- MEFS [WoWMoM’19], FogFS [CCNC’19]
- Not designed for dynamic networks, assumes infrastructure networks are present

Erasure Coding for Reliability with Reduced Storage Cost

- MDFS [TCC'15], HACFS [FAST'15], OctopusFS [SIGMOD'17]
- No procedure mentioned to select Erasure Coding parameters

Erasure Coding for Reliability with Reduced Storage Cost

- MDFS [TCC'15], HACFS [FAST'15], OctopusFS [SIGMOD'17]
- No procedure mentioned to select Erasure Coding parameters

Data Sharing at MEC

- Griffin [HotEdge'20], Li et al. [IEEE-IoT'18], Zhang et al. [INFOCOM'21]
- Cannot share data in the absence of infrastructure networks

Erasure Coding for Reliability with Reduced Storage Cost

- MDFS [TCC'15], HACFS [FAST'15], OctopusFS [SIGMOD'17]
- No procedure mentioned to select Erasure Coding parameters

Data Sharing at MEC

- Griffin [HotEdge'20], Li et al. [IEEE-IoT'18], Zhang et al. [INFOCOM'21]
- Cannot share data in the absence of infrastructure networks

Commercial Solutions

- Dropbox, OneDrive, Google Drive
- Require cloud for reliable storage and sharing

Erasure Coding for Reliability with Reduced Storage Cost

- MDFS [TCC'15], HACFS [FAST'15], OctopusFS [SIGMOD'17]
- No procedure mentioned to select Erasure Coding parameters

Data Sharing at MEC

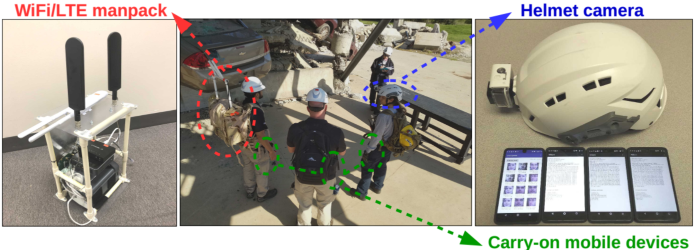
- Griffin [HotEdge'20], Li et al. [IEEE-IoT'18], Zhang et al. [INFOCOM'21]
- Cannot share data in the absence of infrastructure networks

Commercial Solutions

- Dropbox, OneDrive, Google Drive
- Require cloud for reliable storage and sharing

How to answer the aforementioned research questions?

Mobile Edge Clouds with DistressNet-NG



Mobile Edge Clouds with DistressNet-NG

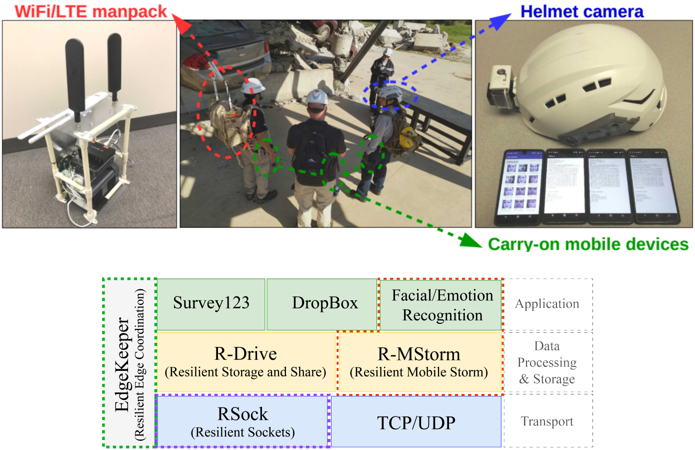


Figure: DistressNet-NG Hardware and Software Architecture for Mobile Edge Clouds

Mobile Edge Clouds with DistressNet-NG

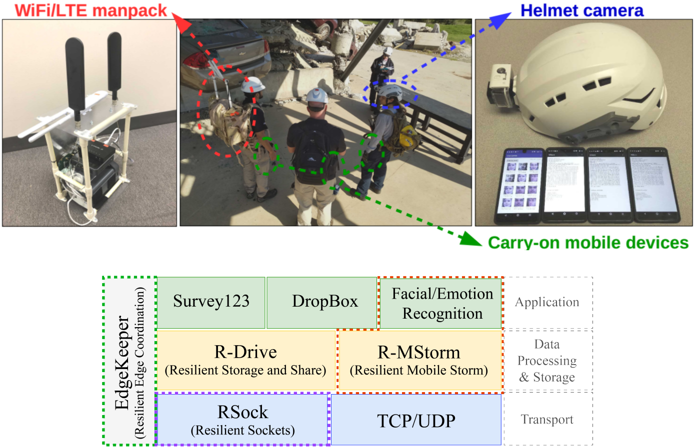


Figure: DistressNet-NG Hardware and Software Architecture for Mobile Edge Clouds

RSock: Elsevier’22, R-MStorm: SEC’20, EdgeKeeper: MASS’22

R-Drive Architecture

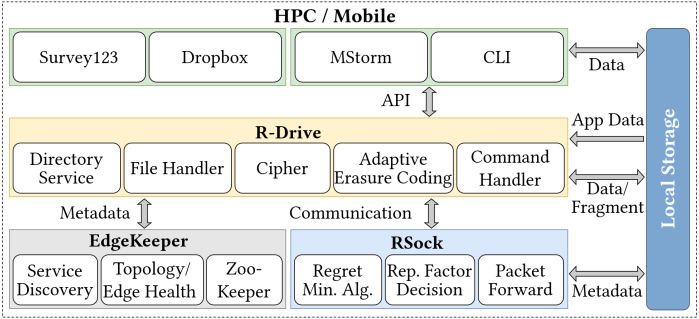


Figure: R-Drive architecture and its integration with DistressNet-NG

R-Drive Architecture

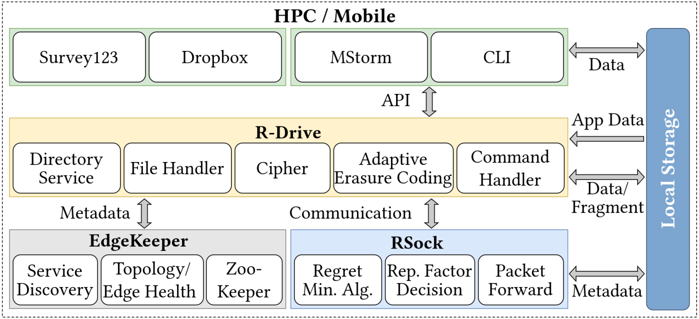
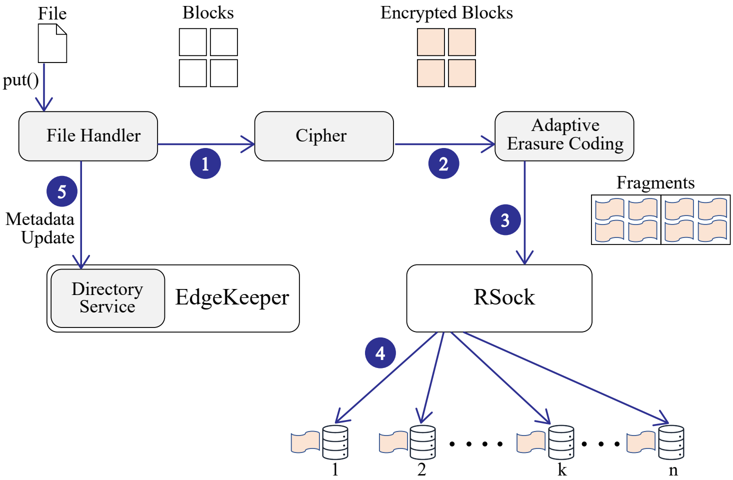


Figure: R-Drive architecture and its integration with DistressNet-NG

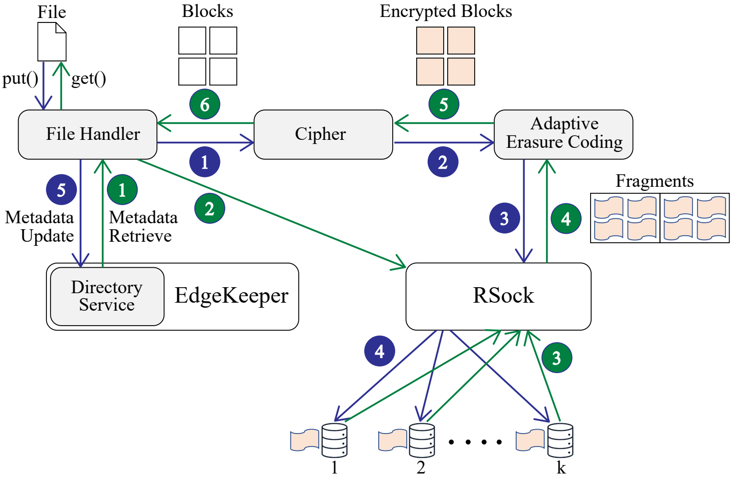
Components we focus here:

- File Handler
- Adaptive Erasure Coding

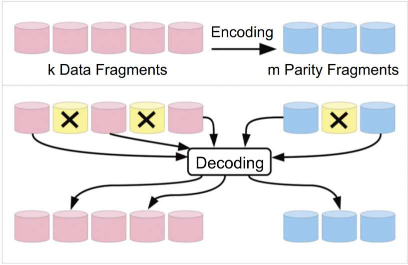
File Handler - File Creation



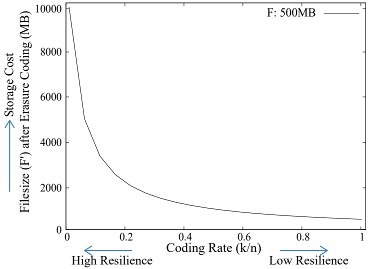
File Handler - File Creation & Retrieval



Adaptive Code Rate Selection



(a)



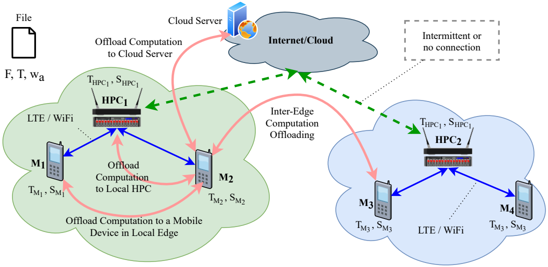
(b)

Erasure Coding - Reed Solomon *

- Need any k out of n fragments ($n = k + m$). The ratio k/n is called *Code Rate*.
- Reducing the Code Rate increases Resilience, at the price of storage.

* J. S. Plank, “Erasure codes for storage systems: A brief primer,” USENIX Mag., vol. 38, no. 6, pp. 44-50, 2013

Adaptive Code Rate Selection



Challenge

How to chose *k* and *n* for a particular system? Which *n* devices?

Solution

We need an online algorithm that takes edge parameters as inputs and decides best *k* and *n*, and the fittest *n* devices.

Adaptive Code Rate Selection

Availability

A device's battery remaining time impacts Availability.

Device availability:

$$p_i = \begin{cases} 1, & T_i \geq T \\ T_i/T, & 0 < T_i < T \end{cases}$$

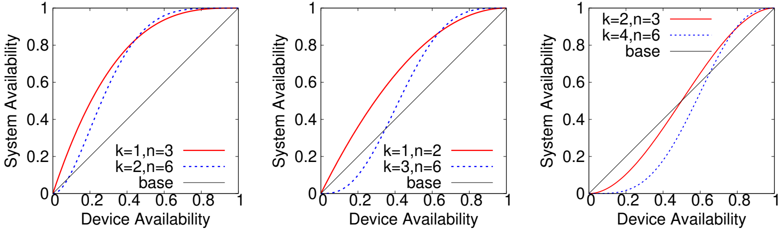
System availability:

$$A(k, n, p) = C_k^n p^k (1-p)^{(n-k)} + ... + C_n^n p^n$$

where:

- p_i = i[th] device Availability
- T_i = i[th] device battery remaining Time
- T = user's desired file availability Time
- A = system availability, when $p_i = p_j, \forall i, j, i \neq j$

Adaptive Code Rate Selection



(a) k/n=1/3 (b) k/n=1/2 (c) k/n=2/3

Figure: System availability as a function of device availability and Code Rate. “base”represents pure local storage.

Similar Coding Rates may not provide similar System Availability due to variable Device Availability

Adaptive Code Rate Selection

Objective: Minimize Storage Cost while ensuring Availability

$$\underset{(k,n)}{\text{minimize}} \quad C(k,n,w_a) = \boxed{w_a * k/n} + \boxed{(1-w_a) * n/k} \tag{1}$$

cost for high reliability

storage cost for high reliability

$$\text{subject to:} \quad F/k \leq S_n, \tag{2}$$

$$T \leq T_k, \tag{3}$$

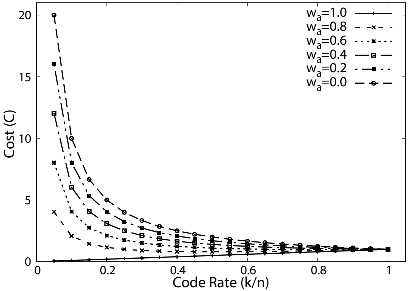
$$1/N \leq k \leq n \leq N, k, n \in Z^+ \tag{4}$$

$$0 \leq w_a \leq 1 \tag{5}$$

- C = Cost function for 3-tuple (k, n, w_a)
- w_a = User's importance for data reliability
- F = Initial file size

- S_n = n^{th} maximum available storage among N devices
- T = User expected File availability time
- T_k = k^{th} longest remaining time among N devices

Cost vs. Code Rate



$$\frac{\partial C}{\partial (k/n)} = 0 \Rightarrow CR = \sqrt{\frac{1-w_a}{w_a}}$$

w_a	**Code rate**	**Optimal Cost**
1.0	0.05	0.05
0.9	0.35	0.6007
0.8	0.50	0.8
0.7	0.65	0.9165
0.6	0.8	0.98
0.5	1.0	1.0

Table: Minimum Cost (C) for w_a and the corresponding Code Rates (k/n)

For every w_a, there is a Code Rate for which the cost is the lowest

Adaptive Code Rate Selection Algorithm

Algorithm 1: Choose (k, n) and n devices

Input : F, T, w_a, S_i, T_i
Output: (k,n) and n devices

$(k, n) \leftarrow (1, 1)$
$C_{min} \leftarrow 1$
for $n' \in 1 \ldots N$ **do**
  for $k' \in 1 \ldots n'$ **do**
    if *Satisfying Eq. (2)(3)* **then** → Satisfy Storage and Battery Requirement
      if $C(k', n') < C_{min}$ **then** → Compare Cost
        $(k, n) \leftarrow (k', n')$
        $C_{min} \leftarrow C(k', n')$
      if $k'/n' = k/n$ **then** → Compare System Availability
        if $A(k, n, \overline{p}) < A(k', n', \overline{p})$ **then**
          $(k, n) \leftarrow (k', n')$
$V \leftarrow$ pick up devices with $S_i > F/k$
sort V based on T_i in descending order
$V_n \leftarrow$ choose top n devices with the largest T_i
return (k, n) and V_n

System Implementation and Performance Evaluation

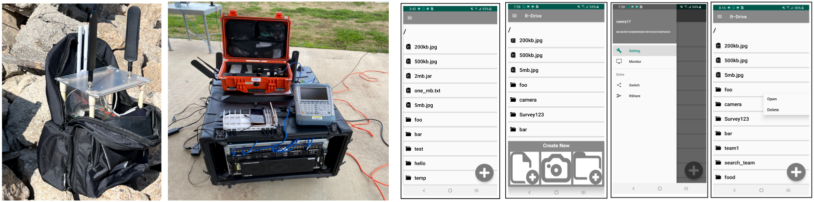


Figure: DistressNet-NG HPC nodes and the R-Drive app

System Implementation and Performance Evaluation

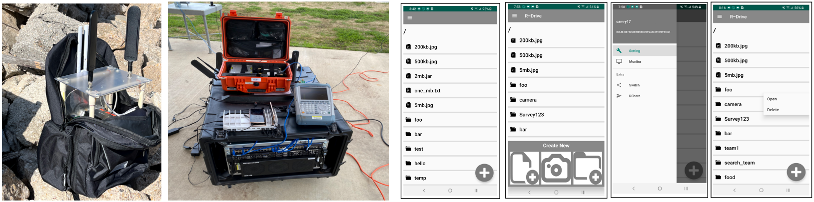


Figure: DistressNet-NG HPC nodes and the R-Drive app

Metrics for evaluation

- Storage Cost
- Read throughput (as a function of k/n and link availability)
- Write throughput (as a function of k/n and link availability)
- R-Drive Overhead (processing, energy, execution time)
- No work on code rate adaptation for comparison

Rate Selection : Achieved Cost

w_a	Lower Bound	Achieved Cost		
		NS=30	NS=20	NS=10
1.0	0.05	0.2402	0.3613	0.66
0.9	0.6	0.6	0.6048	0.6782
0.8	0.8	0.8	0.8121	0.8360
0.7	0.9165	0.9165	0.9166	0.9183
0.6	0.9797	0.9797	0.9799	0.9807
0.5	1.0	1.0	1.0	1.0

Table: Achieved cost for different w_a and Network Sizes (NS)

For larger network size, achieved cost is closer to the optimal cost

Rate Selection : CR Decision

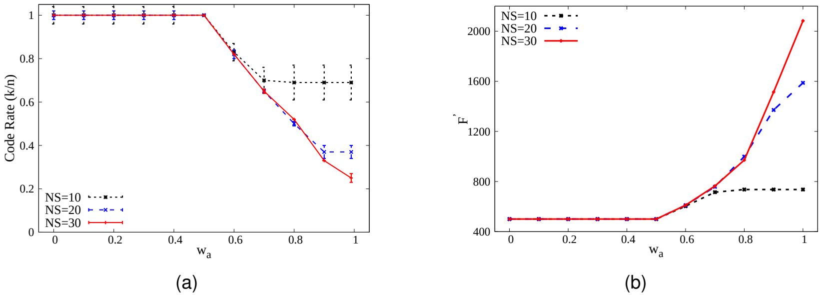


Figure: Impact of w_a on: a) code Rate (k/n); and b) file size F', for network sizes, NS=10, 20 and 30

Data Write Throughput

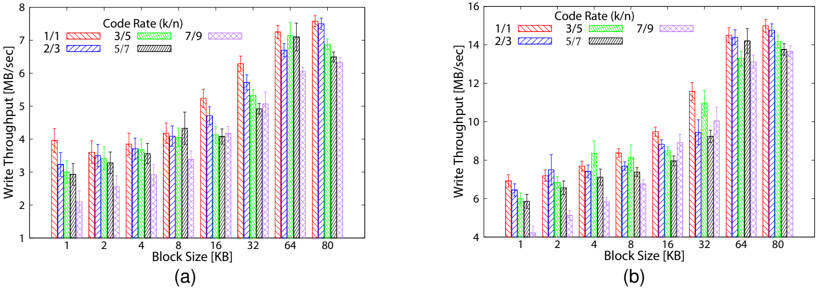


Figure: Data write throughput, for 0.5 link availability (a) and 1.0 link availability (b)

Data Read Throughput

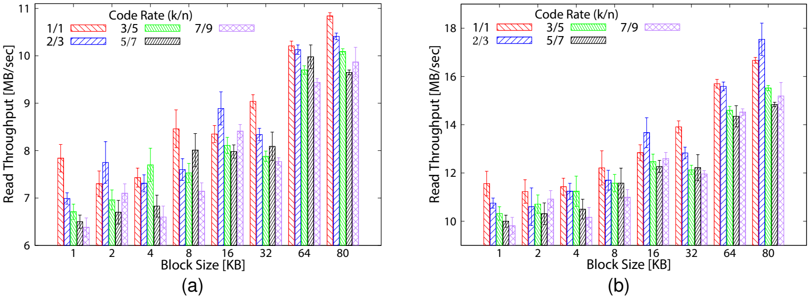


Figure: Data read throughput, for 0.5 link availability (a) and 1.0 link availability (b)

R-Drive Overhead

Energy consumption for different Android devices

Device	*Runtime h:min*	*Consumed %*	*Consumed mAh*	*Consumed Wh*	*Dist-NG Wh*
Samsung S8	3:30	12.5	377.4	1.5	3.5
Google Pixel	3:05	11.9	323.5	1.2	3.2
Essential PH1	3:15	12.6	381.8	1.5	3.8

R-Drive Overhead

Energy consumption for different Android devices

Device	*Runtime*	*Consumed*			*Dist-NG*
	h:min	*%*	*mAh*	*Wh*	*Wh*
Samsung S8	3:30	12.5	377.4	1.5	3.5
Google Pixel	3:05	11.9	323.5	1.2	3.2
Essential PH1	3:15	12.6	381.8	1.5	3.8

Processing overhead as percentage of total delay

	Shamir	*AES*	*Reed-Solomon*
Read	5%	87%	8%
Write	3%	84%	13%

R-Drive Overhead

Energy consumption for different Android devices

Device	*Runtime*	*Consumed*			*Dist-NG*
	h:min	*%*	*mAh*	*Wh*	*Wh*
Samsung S8	3:30	12.5	377.4	1.5	3.5
Google Pixel	3:05	11.9	323.5	1.2	3.2
Essential PH1	3:15	12.6	381.8	1.5	3.8

Processing overhead as percentage of total delay

	Shamir	*AES*	*Reed-Solomon*
Read	5%	87%	8%
Write	3%	84%	13%

Adaptive Rate Selection Algorithm Execution Time (in msec)

Device	*NS=30*	*NS=20*	*NS=10*
Samsung S8	101.6	15.3	0.541

Conclusions

- MEC require careful design of their architectural components, for seamless, optimized operation.
  - R-Drive integrated with DistressNet-NG
- MEC benefit from Adaptive Code Rate selection.
  - R-Drive employs Adaptive Code Rate selection.

Future Work

- Recovery of lost file fragments, to continue guarantee k/n
- Moving fragments from one device to another before device failure
- Extend R-Drive API to allow per-block operations

Acknowledgements and Code Releases

Cooperative Agreement #70NANB17H190: “DistressNet-NG: Resilient Mobile Broadband Communication and Edge Computing for FirstNet”

- R-Drive: `https://github.com/LENSS/R-Drive`
- EdgeKeeper: `https://github.com/LENSS/EdgeKeeper`
- RSock: `https://github.com/LENSS/RSock`
- MStorm: `https://github.com/LENSS/EdgeStorm`
- EmuEdge: `https://github.com/LENSS/EmuEdge`