Poster: EdgeKeeper – Resilient and Lightweight Coordination for Mobile Edge Computing Systems

S. Bhunia, R. Stoleru, A. Haroon, M. Sagor, A. Altaweel, M. Chao, M. Maurice†, R. Blalock†
Department of Computer Science and Engineering, Texas A&M University
†National Institute of Standards and Technology (NIST)
{sbhunia, stoleru, amran.haroon, msagor, altaweelala1983, chaomengyuan}@tamu.edu
†{maxwell.maurice, roger.blalock}@nist.gov

ABSTRACT

Mobile Edge Computing (MEC) is gaining significant interest from first responders and tactical teams. Typical cloud-based coordination (e.g., service discovery and coordination, device naming, authentication) does not work in MEC due to high user mobility. We design and implement EdgeKeeper to provide cloud-like service coordination to distributed edge computing applications for MEC systems. It maintains an edge cluster among devices and intelligently stores data on a group of replicas to guard against node failures/disconnections. We provide a full-system implementation of EdgeKeeper for Android and Linux platforms and evaluate it with MEC applications in a real-world wide-area search and rescue operation conducted by first responders.

1 Introduction

In disaster response mission the responders are assisted by handheld devices, on-body cameras, and other sensors. As the cellular wireless infrastructure is usually unavailable, these teams carry a deployable system, typically providing a diverse hardware configuration with 4G LTE, WiFi, or WiFi Direct. This enables mobile devices to provide distributed computing resources. Traditionally, popular mobile applications offload processing-intensive jobs to remote cloud servers. In the absence of cloud connectivity, forming an edge becomes an emerging necessity [1]. In our proposed design, as can be seen in Figure 1, multiple mobile nodes form an edge network where mobile devices can offload tasks to nearby mobile devices as well as the cloud server when it is available.

Similar to Apache Hadoop ecosystem, we developed the DistressNet-NG ecosystem, mainly targeting edge networks formed by handheld devices and deployable manpacks carried by first responders. EdgeKeeper runs on all the devices in the background and provides resilient coordination to client applications. It provides a comprehensive API for client applications that includes device naming, application coordination, metadata storage, authentication and edge status monitoring. It hides all the complexity of edge coordination from applications. It is implemented on Linux and Android platforms, and the source code is available on Github. Several tests during real-world deployment of first-responders prove its effectiveness.

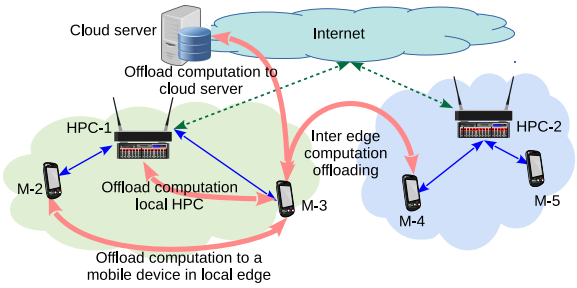


Figure 1: Mobile edge network architecture

2 EdgeKeeper Design

EdgeKeeper adopts the master-replica concept and replicates data over multiple devices to tackle link failure. The role of replica and client is chosen dynamically, depending on the network status.

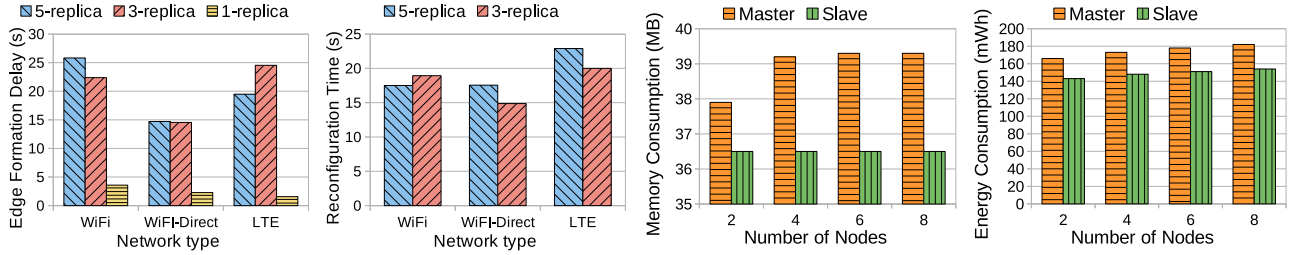
Identity, device naming and authentication: EdgeKeeper uses Global Naming Service (GNS) [2], which employs multiple name servers to deal with high name resolution rates across the globe. Each name record is associated with a primary key: a globally unique identifier (GUID). EdgeKeeper uses a local cache mechanism to store the name records at the edge and it also lazily updates the GNS server when the Internet is available. EdgeKeeper uses X509 certificates for node authentication and a certifying authority (CA) at each organization creates client certificates and signs them.

Service Discovery and Coordination: GUID records are used for service discovery. A device offering some service adds the service name and the role (e.g., server, client, etc.) in the GUID record to be discovered by other nodes. Any node who wants to find a list of nodes offering a particular service will query to retrieve a list of GUIDs.

Resilient MetaData Storage: EdgeKeeper provides resilient metadata storage to client applications. EdgeKeeper clusters dynamically chose the replica nodes (maintaining consensus) based on the availability of devices.

(a) Gypsum, CO open field (b) NIST deployable (c) Disaster City, TX (d) Manpack

Figure 2: Real-world deployments in Gypsum, CO (a-b) and Disaster City, TX (c-d)



(a) Edge formation (b) Edge reformation (c) Avg. memory consumption (d) Avg. energy consumption

Figure 3: EdgeKeeper edge formation latency and resource consumption on mobile devices

Monitor Edge Status: EdgeKeeper provides information about: 1) *the network topology*, and 2) *devices statuses* at the edge. It runs a topology discovery service where each device periodically pings other devices in the network to determine the device-to-device link quality. EdgeKeeper running on each device periodically measures the device status (functioning processors, available memory, remaining battery, available storage, etc.) and reports it to the local EdgeKeeper-master. This information is available to all devices.

3 Evaluation

We evaluated EdgeKeeper both in a controlled laboratory environment and in real-world outdoor deployments where we conducted multiple sets of experiments. The first set of experiments was carried out in an open mountainous region near Gypsum, CO, (Figure 2a) where the mobile phones were mostly connected to the LTE network (Figure 2b). The second set was carried out in Disaster City, TX (Figure 2c) by first responders with a deployable manpack (Figure 2d).

We conducted experiments on WiFi, WiFi-Direct, and LTE networks. EdgeKeeper on all devices, start serving clients as soon as the master is started in a 1-replica configuration. For multiple replica configurations (3 to 5), the EdgeKeeper cluster needs on an average 20s to 25s (Figures 3a-b) to start serving when the topology ping interval is set to 10s. We used the Android Studio profiler to measure resource consumption on mobile devices (Figures 3c-d). When running in master mode, EdgeKeeper consumes significantly higher memory and network traffic than running in replica mode. However, the processor usage and energy consumption are similar for master and slave modes. In slave mode, the memory consumption remains intact even when the number of nodes in the cluster increases. However, the network traffic increases with the number of devices in the cluster. In all cases, the memory consumption is below 40MB, which is negligible compared to the 4GB internal memory of a mobile phone. Processor usage is within 0.16%. It also consumes very low power (0.17W/h) compared to the battery capacity of 3050mAh. The results indicate that EdgeKeeper is lightweight and suitable for running on mobile devices.

We also evaluated the service discovery API provided to the client applications. We observed that the delays are within 8ms for LTE. We evaluated the effect of link quality on the replica consensus. We observed that the average response time increases significantly with the distance between the manpack and phones and beyond 100m distance, the phones could not maintain good communication with the manpack, which resulted in unsuccessful consensus.

4 Conclusions

We present EdgeKeeper, a resilient cloud-like application coordination service for mobile edge computing. The tests with real-world deployment showed that EdgeKeeper automatically discovers devices in the edge network, quickly forms an edge cluster, and reconfigures it after node departures.

5 References

[1] X Chen et al. Efficient multi-user computation offloading for mobile-edge cloud computing. *Transactions on Networking*, 5:2795–2808, 2016.

[2] A Sharma et al. A global name service for a highly mobile internetwork. *ACM SIGCOMM Computer Communication Review*, 44:247–258, 2014.