ZZFS: a hybrid device and cloud file system for spontaneous users

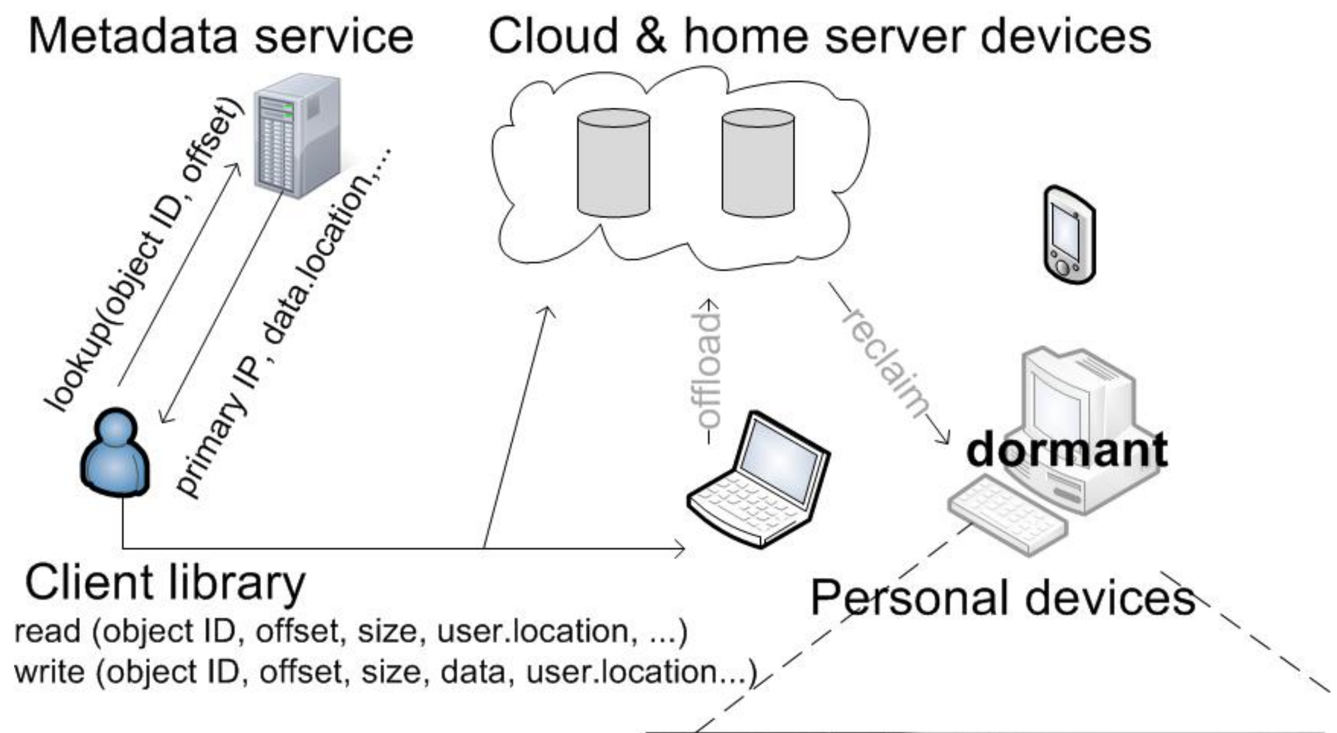
Michelle L. Mazurek, *Carnegie Mellon University*
Eno Thereska, Dinan Gunawardena, Richard Harper, and James Scott, *Microsoft Research, Cambridge, UK*

Motivation

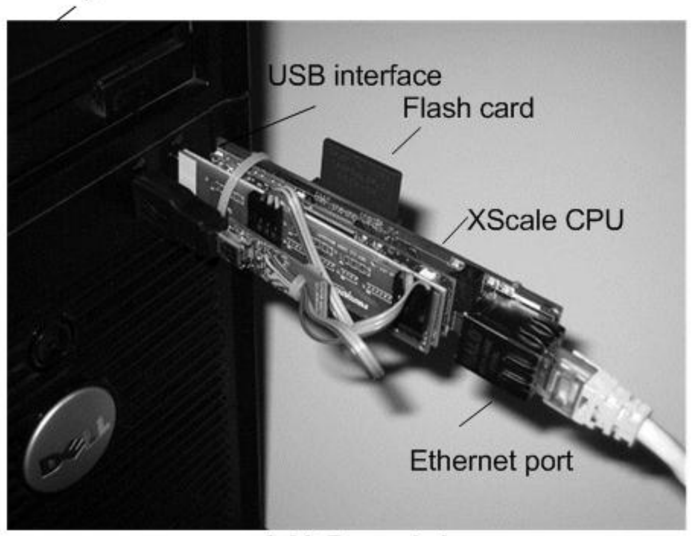
- Driving question: what is the right way to build a file system to store and retrieve my stuff, across devices and cloud/data center space?
- New approach (after many repeated failures): think from the users point of view. Life is messy and busy and data management is not their first priority. This is joint work between traditional “systems” folks and “HCI” folks. It involves user studies and system building.

Approach

- Understanding users: 2 small user studies (in paper) and 1 bigger one in ACM SIGCHI’2012 (on why “keep it all on the cloud” is not the answer).
- Main technical contribution: we build a low-power, always-on communication channel that is available even when a device is off. The hypothesis is that this channel reduces the likelihood that a device is unreachable and thus helps the execution of data placement and consistency policies. We build this channel using new hardware and storage system protocols.


Metadata service
Cloud & home server devices
lookup(object ID, offset)
primary IP, data.location,...
offload
reclaim
dormant
Client library
Personal devices

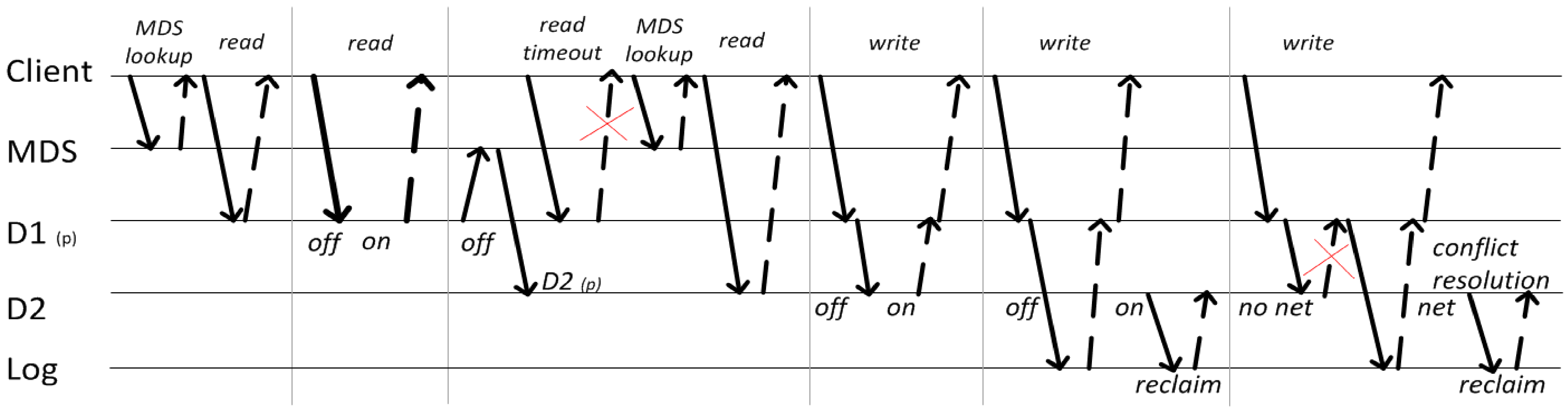
read (object ID, offset, size, user.location, ...)
write (object ID, offset, size, data, user.location...)



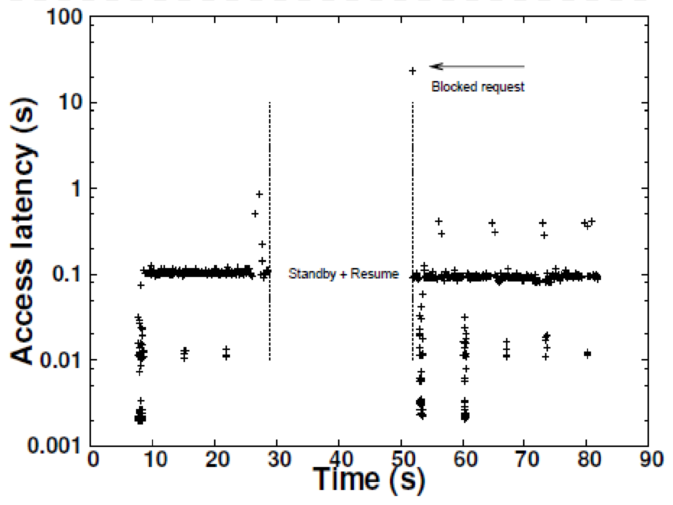
NIC add-on

Technical building blocks

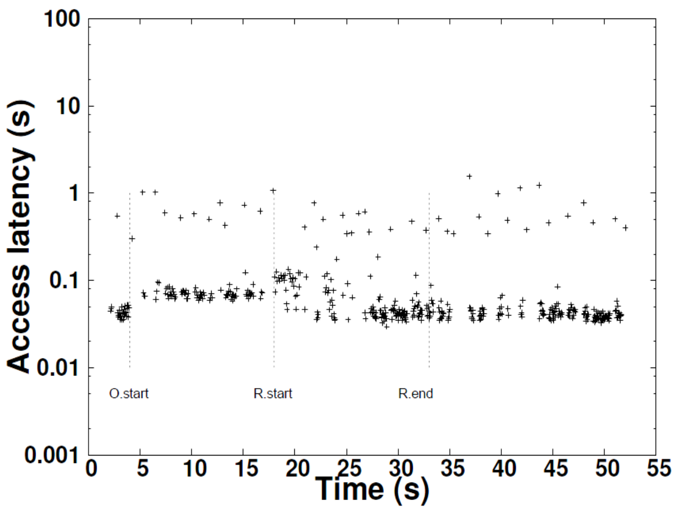
- Hardware: low-power NIC (Somniloquy) running embedded OS on 400 MHz Xscale processor. Exports >1 GB of flash space. Maintains network-awareness, i.e., illusion that device is always on, even when device is off.
- Storage system: Data placement protocols are based on versioned histories for consistency. I/O offloading diverts writes to NIC’s flash card or cloud space when device is off. Storage system exports a device-transparent view of the namespace, with the metadata residing on the cloud by default. ZZFS is a file system that runs in user space. Here are some typical scenarios it handles:



(a)Basic read (b)Turn on device and read (c) Read with device off- read from replica (d)Turn on device and write (e)Write with device off – offload (f)Write with no network connectivity



Worst-case latency when maintaining network awareness



Offloading to and reclaiming from the cloud

Microsoft Research, Cambridge, UK

SYSTEMS &NETWORKING