Cloud4Home — Enhancing Data Services with @Home Clouds

Authors:
Sudarsun Kannan
Ada Gavrilovska
Karsten Schwan
(Georgia Institute of Technology)
Motivation

• Data center infrastructure is getting better, but, the rate at which end devices at internet’s periphery are transforming is even more.

• We can use the (excess) resources at end devices along with remote cloud to enhance the performance of applications.

• Purely local computation vs. remote computation.
Cloud4Home Paradigm

Cloud4Home = @home + @datacenter

QoS(Cloud4Home) > Max(QoS(@home), Qos(@dc))

Emphasis on services for storage, access and manipulation of data for home environment.
Cloud4Home - Design Principles

- **Fungibility (Interchangability):** of physical resources for dynamic flexibility.
- **Augmentation:** Services should not be constrained to execute within the boundaries of home cloud or remote cloud.
- **Guided Active Management:** Places where services actually run should be based on context, user needs and resource availability.
- **Automation and Independence:** Guided management should be automatic and free from end user participation, OS, application framework on the mobile devices.
Use Cases - Home Surveillance

• A typical solution would provide specialized hardware which takes pictures and videos and offload them to the public cloud, ..... But the **downside**?

• Cloud4Home solution will store and compute the data in local nodes in home environment, and hence can reduce the latency (certainly a very desirable quality in case of a theft).

• Guided management controls what is stored in the public vs. the home cloud, thereby limiting the service delays, costs and providing desired privacy guarantees.

• At the same time, large amounts of resources at cloud storage still remain available, storage resources for public databases of image training sets, or computational resources for parallel execution of face detection and recognition algorithms.
Other Use Cases

- Data services in hospital environment
- Camera-guided home gaming applications
A crucial entity: VStore++

- A virtualized object storage system that abstracts location of the object storage from the applications using them.
- Independent of OS or middleware by operating at the virtualization level of systems.
- Fungible, since data can be stored locally or remotely in a transparent manner, and independent of underlying OS etc.
- Interactions may take place through LAN or WAN.
- This results in a Cloud4Home comprised of heterogeneous devices that collectively provide seamless storage, access and data manipulation services, which interface with public cloud.
VStore++ Architecture

Figure 1. Cloud4home architecture.
Metadata Management

- Lookups data objects and services.
- Routes VStore++ requests to the appropriate location.
- Tracks information regarding available resources.

Mapping organized as Key-Value store (similar to a DHT) built on top of Chimera P2P system, where unique keys correspond to Either object names, service names, or node identifiers.

For Data Objects, Key = ObjName, Value = location
For Services, Key = SerName|SerID, Value = location
Resource Management

```
GetDecision( Objname, service_id, mechanism) {
    NodeList = getListOfNodesFromLogicalTree()
    getOptimalNode ( NodeList, service_id, mechanism)
/*mechanism denotes strategy to use for finding optimal node (greedy, fair)*/
}
Store Operation

• Application calls VStore++’s CreateObject() to map a file to an object. This creates metadata as well.
• Invoking StoreObject() transfers the object from the application’s guest domain to VStore++’s control domain, which decides where to store the object.
• By default, the object is stored in the node’s mandatory bin. When it is full, or when an explicit storage policy specifies that the data should be stored elsewhere, data is stored either in the voluntary resources available on other nodes in the home environment, or in a remote cloud.
Fetch operation

- FetchObject() operation is passed to VStore++ domain, where an IPC message is sent to the metadata module to determine the location of the object.

- The object is fetched, it is passed to the application’s guest VM.
Fetch and Process

• An object fetch operation may be explicitly associated with certain processing, specified via a service id.
• When ‘object owner’ receives the request, it uses the service id to check if the requestor is capable of executing the service itself. If yes the object is simply returned, and the service processing is performed at the requesting node’s guest domain. Otherwise, the object owner checks whether it is capable of performing the required service, and if so, returns the output of the operation.
• If none, the ‘value’ field for the service is used to determine other possible targets, including in the remote cloud for execution of service.
Interfacing with Public Clouds

• One or more nodes in the home cloud support a public cloud interface module, responsible for routing all remote cloud interactions. This is useful when some subset of nodes are more powerful (e.g. better Internet connectivity) than others.

• The location of data object and availability of services in remote cloud are maintained in the same key value store. For data object, object name is the key, URL location of object in S3 storage bucket is stored as value.
Implementation

VStore++

- Implemented in C++, uses boost library for multi-threading.
- Every method call is converted to a command, which is used for communication between VMs and remote nodes.
- Object transfers between remote machines happen via Linux zero copy mechanism, which provides kernel-to-kernel socket transfers and avoids user-space overheads.
- For data transfer between dom0 and host domains, XenSocket, a high throughput shared memory kernel module is used.
Implementation

Metadata Management

• Implemented in C by extending the DHT-based Chimera into a key-value store.
• VStore++ communicates with Chimera using IPC. It offers basic put and get interfaces.

Resource Management

• A resource monitoring utility is added to Chimera using the Linux glibtop library, which updates resource information in the key-value store after a configurable time period.
• A simple file system watcher component keeps track of mandatory and voluntary bin space.
Importance of Home Cloud Services

These measurements motivate the Cloud4Home approach in which services can also be run on ‘nearby’ resources (e.g., in the home) that are accessible with lower and less variable latencies.
Utility of joint usage of home and remote resources

Figure 6. Fetch Throughput
Summary

• Cloud4Home is an approach to leverage both home and datacenter resources to provide better services for the home.
• Current use cases target the home environment, but Cloud4Home could easily be generalized to operate in office or larger scale environments like hospitals.
• Performance gains are realized because of reduced latency of service provision, reduced data rates and bandwidth needs to/from end systems, while still retaining the potential benefits of using large datacenter storage and processing capabilities.
Issues

- No robust data privacy policies
- No means to exploit heterogeneity
- How to scale?
- Tune as per changing environment e.g. network
- Collaborating multiple Cloud4Home for neighborhood security
- Possibility of using better protocols for large object transfers
- No automation for selecting the locations at which certain services should be run or where objects are stored.
- Better infrastructure for easily formulating and running diverse policies.
My views

• No comparison against a similar framework.
• Virtual devices in the evaluation test setup, and hence extra virtualization overhead.
• Fault tolerance in case of node failures
• How to add and remove nodes from a live deployment?
• The notion of policy and mechanisms seemed a bit vague.
• How automatic guided active management is achieved is unclear.
• Interoperability with multiple cloud vendors.
• No mention on capacity of mandatory bin and voluntary bins
Questions,
Comments,
Suggestions
Thanks !
References

• http://wwwusers.cselabs.umn.edu/classes/Spring2017/csci8980/papers/EdgeArch/Cloud4Home_GT.pdf
• http://current.cs.ucsb.edu/projects/chimera/
• https://www.cs.cmu.edu/~dga/15744/S07/lectures/16-dht.pdf