

KTH ROYAL INSTITUTE OF TECHNOLOGY

Highly Available and Robust Network Services in  
Under-served Areas

JIANNAN GUO

Master's thesis

Supervisor: Prof. Björn Pehrson, KTH  
Examiner: Prof. Markus Hidell, KTH

Stockholm, 15<sup>th</sup> September, 2014

## Part I

# Robust Network Infrastructure in Rural Areas of Tanzania

Part II

Highly Available Web  
Services

# Chapter 1

## Introduction

### 1.1 Background

Affordable, yet stable web services are highly desired in rural area, as a mean to alleviate digital divide and improve life quality. When it comes to under-developed regions in Africa, requirements and conditions need to be carefully assessed and analyzed, for that challenges could be unique and dramatically different than metropolitan.

### 1.2 E-learning for Open University of Tanzania

Open University of Tanzania (OUT) [1] is the first university of East Africa Region to provide open and distance learning programmes. To distribute course content through the whole country, Moodle has been chosen as underlying digital resource management platform. Moodle[?] is an open-source industrial-level online learning platform and resource management system. As a typical data-driven web service, Moodle runs over an underlying database and assemble its webpages on-the-fly based on user requests. It is written in PHP and heavily tested against Apache, Nginx and MySQL. At present, the platform is running as a standalone web service in a central server and mainly serve static content such as PDF, Text and Slides. Although OUT has the vision to introduce multi-media materials to enhance education quality. OUT also establishes learning centers in major cities and towns all over the whole country and is ambitious to extend to a larger scale. An emerging obstacle is to provide services in remote areas with poor network connection and bandwidth.

### 1.3 Problem Identification

As part of the project, we investigated local conditions and needs within the scale of Serengeti Broadband Network, especially in areas with evident

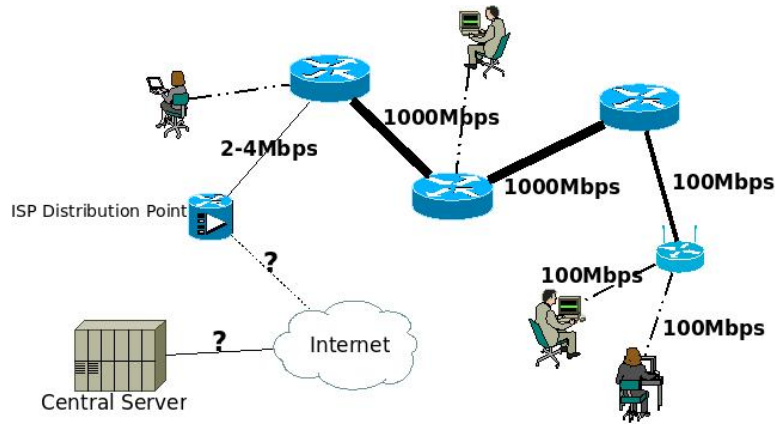


Figure 1.1: A Typical Setting of Rural Local Access Network (LAN)

demands of services and lack of infrastructures. And we were able to identify following challenges:

### 1.3.1 Power Outages

Power grid in rural areas of Tanzania is so unreliable that UPS for critical device is almost a must. While people are gradually adapting to mobile platforms, such as smart phones and tablets, backbone infrastructures are also required to be more persistent. Equipments powered up by solar and battery are highly desired due to cost-efficiency. Although power consumption need to be optimized in this circumstance in order to prolong battery life and improve reliability.

### 1.3.2 Poor network quality and frequent failure

Although local network is operated by ICT4RD project and can be fairly reliable, uplink is still depending on national-wide ISP and is somewhat unpredictable according to our observation. Network failure could occur any-time and can last for random period (several minutes to several days). Those web services that depend on a central server are apparently not accessible during the failure. On the other hand, the uplink can be very narrow due to poor infrastructure and limited budget. It could be difficult to squeeze multimedia services into such bandwidth.

To better illustrate this problem, suppose a typical setting in Figure 1.1. Major backbone components in this LAN are interconnected through fiber-optical lines, and network is distributed to users through WiFi or Ethernet. The LAN is linked to the Internet through ISP distribution link and central server resides on the otherside of the Internet.

Due to limited budget and ISP capacity, the upper link is equipped

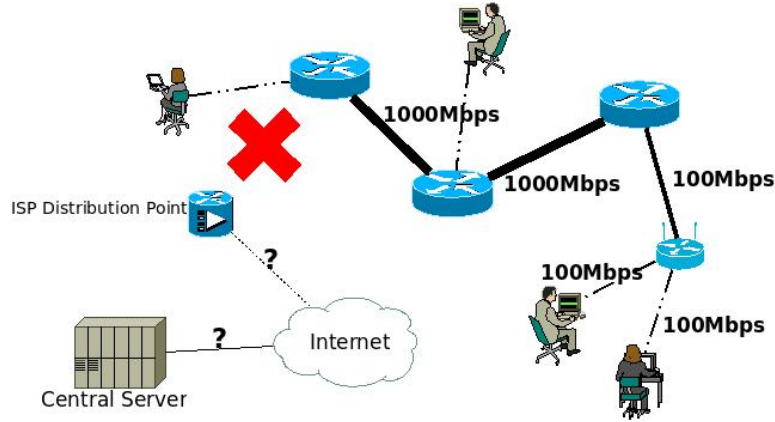


Figure 1.2: Uplink Failure leads to the isolation of LAN

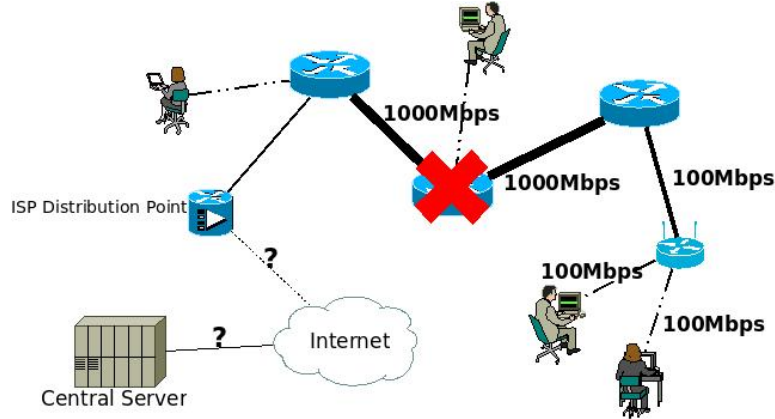


Figure 1.3: Network Separation due to Component Failure

with an average bandwidth of 2.4Mbps which is shared among all users in LAN. While a minimum bandwidth of 1.5Mbps is recommended for video streaming, it is difficult for users to get decent service from central server. To worsen the situation, the upper link is somewhat unpredictable, which leads to the isolation of LAN, as shown in 1.2.

On the other hand, components in the LAN can also break down which leads to network separation, see Figure 1.3. In the first case, multi-media content can hardly reach end users. And in other two cases, users cannot get service at all.

### 1.3.3 Limited budget

Cost is an essential factor during rural ICT development. Given relatively smaller user base and weaker demand, equipments need to be chosen wisely. Although future maintenance and development also need to be considered.

## Chapter 2

# Adapt to Frequent Network Failure and Limited Bandwidth

### 2.1 A closer look at the problem

As introduced in section 1.2, Moodle is deployed as underlying course management system for OUP E-learning platform. Moodle is an open source project written in PHP and well-documented[?][?]. Similar to other web applications, it can be deployed in a typical LAMP or LNMP stack. In this chapter, we mainly focus on possible solutions for two problems stated previously, and leave the choice of actual server to chapter 3

Moodle is a typical database-driven web application where all the pages are generated on-the-fly based on user request. The whole application is composed of three main components:

- PHP source code, typically in `/var/www/moodle/`
- A database to store data or metadata including site configuration, student information, course details, events, etc.
- A directory to store materials and resources, as well as cache and temporary files. Typically it is named as `moodledata/`

The problem addressed previously can be simplified and modelised as following, see Figure 2.1. Each node in the model denotes a local server/proxy and has a certain amount of users associated with it.

As simple as the model might be, components in it could be vastly heterogeneous while mapping to different techniques. Content stored in a node can either be web objects, SQL replies, codes or even entire databases. Communication in between can also be based on a variety of protocols.

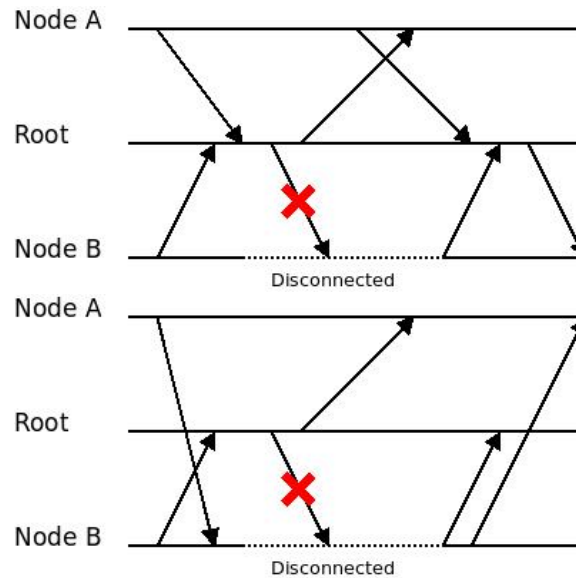


Figure 2.1: Web Delivery Model

As an online learning platform, users do not only passively accept information, but also interact with Moodle through forum, personal blogs and quiz. All the changes made by users must be stored and seen everywhere. Thus, the system should not be read-only under any circumstance.

Moodle has been in service and adding new services should affect existing structure as less as possible. Also, steps of adapting changes should be properly designed to avoid crushing the service.

To maintain consistency and serve up-to-date content, a reasonable amount of communication overhead is necessary and is normally positive proportional to the extent of consistency. Although, due to the presumption of poor network connection and narrow bandwidth, different nodes in the system are preferably decoupled and autonomous.

The autonomy is also closely correlated to the ability of performing offline operation. Many distributed systems have the ability to detect and recover from network partitioning, although it normally leads to a compromise of consistency and content freshness. When a user request a page, Moodle loads all privileges of the user, generate pages accordingly and log the session. This results in uncacheable content and interaction-must logins. It has been proven that consistency, high availability and partition tolerance are impossible to be achieved at same time[?][?], necessary assumptions must be made according to the condition and needs.

While the majority of web caching and content distribution techniques aim at better performance and delivery efficiency, we prioritize the ability of performing basic functionalities during network failure. We tolerate a



relatively loose consistency while ensuring eventual convergence.

Lastly, to realize affordability, we mainly focus on open source techniques and free ware. Thankfully, many successful projects and tools have been made open source and publicly available. In the following sections of this chapter, we evaluate a variety of techniques against the criteria stated above and propose our solution based upon the conclusion. Several of potential solution are also tried out.

## 2.2 Benchmark

### 2.2.1 Content Delivery Network

Content Delivery Network overlaps with Web Cache Proxy at the concept of pushing web content to users. A Content Delivery Network is a collaborative set of surrogate servers spanning the network, where web contents are mirrored[?]. Users will perceive a smaller latency while fetching content from a nearby CDN surrogate server rather than original web server. The essence of CDN is illustrated in figure X

Since more and more web services are evolving to provide dynamic content, CDN also takes advantages of cacheability hints when dealing with dynamic contents[?].

### 2.2.2 Simple Web Caching

An intuitive and common solution for the problem of limited bandwidth is to cache popular web content locally, as illustrated in Figure 2.2. A server-side web cache proxy typically sits in front of web server, attempting to serve user request with cached objects rather than triggering computational workload on web application. Web caching has been proven to be an effective approach to reduce bandwidth usage, user-perceived latency and loads on original server[?].

Web cache is greatly advantageous in our scenario that it does not require modification on web application, except that some TCP optimizations could be done between web cache proxy and web server frontend. Web cache proxy can continue serving requests if offline mode enabled, which also meets our requirements. The traffic between cache proxy and authentic server is standard HTTP request and response. The overhead and frequency of communication mainly depends on cache hit / cache miss ratio, expiration time and cache directory size.

Although, all write traffic traverses cache proxy and goes to original server, which leads to a read-only system immediately when network disconnects. This is magnified when Moodle is taken into consideration, where user logins require interaction with original PHP code and database.

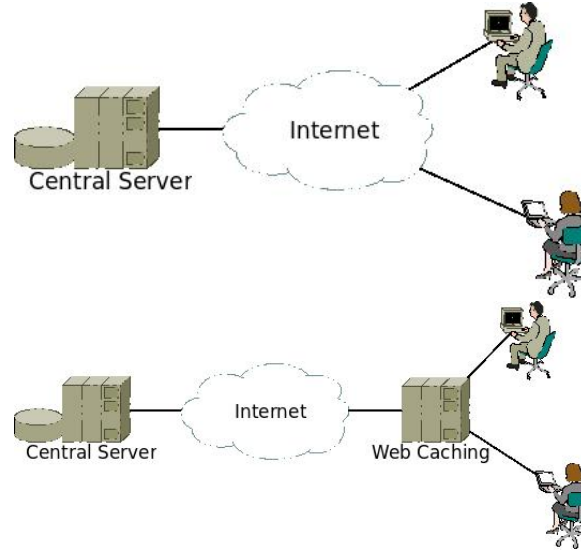


Figure 2.2: Serve users without and with a Gateway Cache

### 2.2.3 Web Caching with Edge Server

To address the issue of dynamic content generation and client-server interaction, an intuitive and brute-force solution is to generate user-specific page at the edge. A comprehensive study of edge servers can be found in ???. Four strategies are presented: (a) edge computing (b) content-aware caching (c) content-blind caching (d) data replication, see Figure X. In each of the strategy, the edge server attempts to reply user request on the behalf of original server with the information that is locally available. Edge computing still heavily relies on central database, hence out of our consideration. While CAC and CBC store partial database at the edge, data replication stores a complete copy of the database. When the size and complexity of database permit, data replication is desired since it outperforms other strategies in both response speed and offline operation. Although, it adds another layer of complexity to perform transaction processing and maintain the consistency through multiple distributed databases. An exhaustive survey is presented in section 2.3.

Application code rarely changes in our case and is always on-way synchronized from original server, thus consistency can be relatively easy to achieve by periodically utilizing tools like rsync??.

## 2.3 Multi-Master Database Synchronization

Database replication techniques has been intensively deployed over clusters and workstations for redundancy and load balancing (Figure X). A

LAN is always assumed by most of database replication techniques, whereas database replicas across the WAN are desired to realize our goals. We prioritize availability by compromising on consistency, as long as the system can reach eventual consistency. Offline operations on distributed databases implies simultaneous modification on the same entry, which result in conflict after reconnection. Techniques to detect and reconcile conflicts are studied and evaluated.

### 2.3.1 Database Cluster

Firstly, several databases with native support for replication are studied.

**CouchDB** is an open source distributed database system developed in Apache. The most attractive feature of CouchDB is that it natively supports bi-directional synchronization among multiple database replicas and offline operations. When one replica is disconnected from the network, it retains autonomy and continues as a fully functional database from user point of view. Although, it fails to be our candidate since it is NoSQL database, whereas Moodle heavily relies on SQL calls and it will a significant task to modify Moodle to use NoSQL database.

**MySQL Cluster** is an open source distributed database system based on MySQL. It supports database sharding and duplicating. A typical use case of MySQL Cluster is shown in Figure X. Although, redundant copy of database can only be accessed with the presence of management master, and cannot be updated during network failures. Furthermore, database nodes are closely coupled with the assumption of LAN (low latency and high bandwidth).

### 2.3.2 Middleware-based Repliation System

Several studies also proposed the approaches to solve database synchronization at middleware-level. **C-JDBC** is an Java implementation of **RAIDb**, aiming at a framework to manage heterogenous databases. With built-in functionality of scheduling transaction processes, C-JDBC is perceived by users as a single virtual database. Although, the system is still centrally managed and could not handle partitioned network. **Ganymed** middleware system, inspired by C-JDBC, achieves consistency by serializing update/write requests at master and propagating changes to replicas in a lazy fashion. Users see a consistent data state (snapshot isolation), even though stale might it be. The limitation of these two middleware is also clear, that no write can be served during network failures.

### 2.3.3 Multi-Master Synchronization System

Similar to MySQL Cluster Multi-Master setup, there exist several third-party tools for the same purpose.

- Galera
- Tungsten
- SymmetricDS

#### **2.3.4 Operational Transformation**

## Chapter 3

# Low Power, yet Powerful

### 3.1 Power Reduction on Servers

### 3.2 Benchmark of Server Ability

#### 3.2.1 Nginx or Apache

#### 3.2.2 Processor

#### 3.2.3 Find the bottleneck

### 3.3 Small-scale Cluster

## Chapter 4

# Conclusion and Future Work

### 4.1 Conclusion

### 4.2 Future Work

# Bibliography

- [1] <http://www.out.ac.tz/>. [Online]. Available: <http://www.out.ac.tz/>