

# Dark Nebula: Using the Cloud to build a RESTful Web Service

## John Fisher, Robert Fisher, and Peter Bui

### Department of Computer Science



## Introduction

With the emerging adoption of cloud computing systems, web sites are no longer just web pages, but also services that provide programmatic interaction. Cloud computing is becoming more integrated in industries and this demands web site evolution. To explore this topic, we developed Dark Nebula, a RESTful web service using Google App Engine, a cloud computing platform. Dark Nebula is a URL shortener which condenses lengthy URLs into shortened versions for ease of distribution. This project evaluates the performance of Dark Nebula as a web service and how it contributes to the growing cloud computing field.

## Implementation

We installed the Google App Engine SDK in a Linux development environment. Also, we programmed the web service in the Python programming language. Our system was implemented under four major design components.

1. Conversion Functions
2. Google NDB Datastore
3. Web Handlers
4. Program Modules

### Conversion Functions

Function Description	Input	Initial Identifier	Converted Identifier	Output
Long URL to Short ID	http://cs.uwec.edu	1000	3e8	http://go.yld.me/3e8
Short ID to Long URL	http://go.yld.me/3e8	3e8	1000	http://cs.uwec.edu

Web Page: <http://go.yld.me>

**GO YLD ME**

Short URL:  Shorten

Long URL: <https://developers.google.com/appengine/?csw=1>

Short URL	Long URL	Date/Time
<a href="http://go.yld.me/31">http://go.yld.me/31</a>	<a href="https://developers.google.com/appengine/?csw=1">https://developers.google.com/appengine/?csw=1</a>	2014-03-26 20:11:57.430630
<a href="http://go.yld.me/30">http://go.yld.me/30</a>	<a href="http://en.wikipedia.org/wiki/Python_(programming_language)">http://en.wikipedia.org/wiki/Python_(programming_language)</a>	2014-03-26 20:11:20.948950
<a href="http://go.yld.me/29">http://go.yld.me/29</a>	<a href="http://stackoverflow.com/questions/18621/python">http://stackoverflow.com/questions/18621/python</a>	2014-03-26 20:10:50.171190
<a href="http://go.yld.me/28">http://go.yld.me/28</a>	<a href="http://www.youtube.com/">http://www.youtube.com/</a>	2014-03-26 20:10:19.521420
<a href="http://go.yld.me/27">http://go.yld.me/27</a>	<a href="http://stackoverflow.com/questions">http://stackoverflow.com/questions</a>	2014-03-26 20:09:34.412740
<a href="http://go.yld.me/26">http://go.yld.me/26</a>	<a href="http://www.apple.com/">http://www.apple.com/</a>	2014-03-26 20:09:13.158440
<a href="http://go.yld.me/25">http://go.yld.me/25</a>	<a href="https://www.netflix.com/2locale=en-US">https://www.netflix.com/2locale=en-US</a>	2014-03-26 20:08:44.344680
<a href="http://go.yld.me/24">http://go.yld.me/24</a>	<a href="http://www.amazon.com/">http://www.amazon.com/</a>	2014-03-26 20:08:17.530260
<a href="http://go.yld.me/23">http://go.yld.me/23</a>	<a href="http://www.barnesandnoble.com/">http://www.barnesandnoble.com/</a>	2014-03-26 20:07:01.792900
<a href="http://go.yld.me/22">http://go.yld.me/22</a>	<a href="http://cs.uwec.edu/">http://cs.uwec.edu/</a>	2014-03-26 20:06:24.628140

Powered by Dark Nebula

### Python Module: URL shortener

```
def darknebula_shorten (long_url):
#1. Specify the domain
domains= 'http://go.yld.me'

#2. Define and encode the long_url and response_format
values = {'long_url': long_url, 'response_format': 'text'}
data = urllib.urlencode(values)

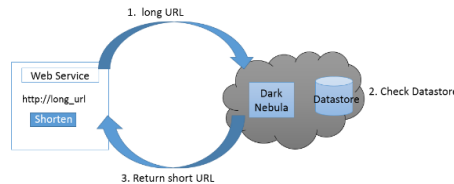
#3. POST data
req = urllib2.Request(domain, data)

#4. Open Dark Nebula (go.yld.me)
response = urllib2.urlopen(req)

#5. Retrieve the short_url
short_url = response.read()
return short_url
```

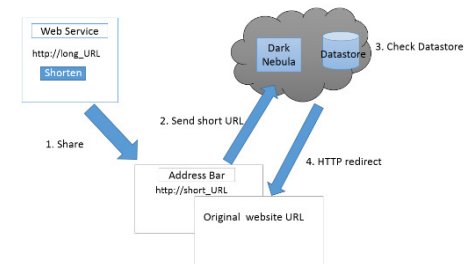
## Design: "Cloud Condensation"

### Part 1:



1. Send "long" URL: A URL is submitted into the Dark Nebula web service via its domain, either on a web page or by executing a POSTing script.
2. Check Datastore: Dark Nebula runs a query to validate URLs. If it finds a new long URL, the web service inserts the new entry into the Datastore.
3. Return "short" URL: The web service returns a short URL with a short URL identifier in hexadecimal following the domain's prefix.

### Part 2:



1. Share: The short URL is distributed with other users that need a condensed URL.
2. Send "short" URL: The short URL is sent to the web service as HTTP GET in a web browser's address bar.
3. Check Datastore: Dark Nebula translates the short URL by querying and retrieving the corresponding long URL.
4. HTTP redirect: After the lookup, Dark Nebula automatically redirects to the long (original) URL page via HTTP 302.

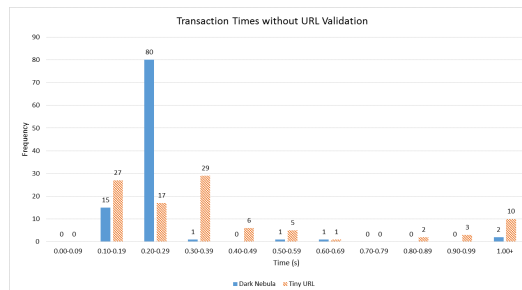
## Evaluation: "Partly Cloudy"

To evaluate Dark Nebula, we needed to analyze the underlying cloud's behaviors and web service's performance. To do so, we ran automated speed tests to compare with TinyURL.com, another URL shortening web service. Using the Dark Nebula module from the implementation section, we conducted various tests which detail the effects of URL transactions and input validation.

### URL Transaction Statistics

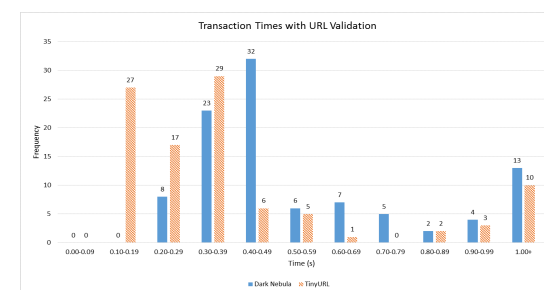
URLs	Validation	Dark Nebula Avg. Time (s)	TinyURL Avg. Time (s)	Avg. Ratio (DN/TU)	Dark Nebula Std. Dev (s)	TinyURL Std. Dev (s)
100	No	0.2371	0.4552	0.5209	0.1575	0.5531
100	Yes	0.5814	0.4552	1.2774	0.3625	0.5531

### Without Validation



Without validation, Dark Nebula demonstrated faster performance than TinyURL.com. This graph shows how cloud resources yield efficiency and consistency without client-side processing.

### With Validation



When validation is implemented, Dark Nebula's performance was significantly slowed down. This graph shows how client-side processing time can offset inherent cloud performance gains.

## Conclusion: "A Clouded Perspective"

The motivation of Dark Nebula is to understand how cloud platforms, like Google App Engine, are effected by web service integration. Based on our performance tests, our system demonstrates a balancing act of speed and correctness. Google App Engine makes it easy to take advantage of evolving cloud technologies. However, cloud resources are not a perfect solution for optimizing web service performance. Despite this, it can be easy for developers and users to take the cloud for granted. In other words, Dark Nebula proves how even the sky has its limits.