

# SCALING YOUR INTERNET BUSINESS

## INTRODUCTION

Growing your business at "Internet speed" means scaling your web application according to business and market requirements. The unique value proposition of Cloud Computing creates new opportunities to align IT and business goals. In order to consume and use Cloud Computing resources to their best advantage, it is incumbent upon business owners, IT managers, startup founders, and senior developers to understand the options that Cloud Computing provides. Using the complete range of Cloud and traditional resources will assure that your web and IT infrastructure can expand to meet your business needs even in high growth situations.

### **SCOPE**

This document describes how to grow your web-based Internet business with a focus on using GoGrid cloud computing infrastructure to achieve maximum scale. It is not a detailed how-to document, but rather a high level description of what scalability is, how web applications scale, how cloud computing helps, and why GoGrid, in particular, is uniquely positioned to help your business succeed.

### **WHAT IS SCALABILITY?**

Scalability is a measure of the ability of an application to expand to meet your business needs. Generally, you can scale a given application by adding more (or bigger) resources when needed. Resources can be many things including servers, storage, and networking bandwidth.

Be careful! Not all applications are scalable by default.

Scalable applications are able to operate normally as they grow and can have more resources added at any time to service more customer demand. Applications that are **not** scalable encounter performance and service availability problems as demand increases. These kinds of applications may not be able to take advantage of more resources.

The design of your architecture is up to you, but this document will help you understand how your infrastructure choices can affect how you grow an application through its life-cycle.

### **HOW WEB APPLICATIONS SCALE**

All web applications scale just a little bit differently. The reason is that everyone's business is also a bit different. While two businesses may grow similarly, in general, one web application doesn't scale in the same manner as another. Differences in business models or architectures mean differences in scalability.

For example, two photo sharing websites like Flickr and SmugMug will have many similar kinds of technology challenges they need to overcome in order to meet customer demands, like resizing uploaded images in real time and storing large amounts of image data. In

comparison, a very different web application like Amazon.com has no image uploads, but does need to build customer purchasing data to provide relevant product recommendations. Gathering, collating, and processing this data is very different from resizing images and brings its own challenges.

While true that not all web applications meet customer demands in the same way, it's also true that not all web applications are *capable* of being scaled well. Twitter, and others, has had notorious issues with growing to meet customer demand. Applications that grow well have generally been designed for scalability, but occasionally architectural challenges make this difficult. The only way to know how your application will perform under increased customer load is to understand where your scaling points are, use capacity planning to measure load on these points, and to add more resources when they are needed.

### **SCALING POINTS**

Although web applications don't perform in the same way, there are very common areas where resources become constrained called 'scaling points.' For example, it's very unlikely you will have a web application that can grow to meet all of your growth on a single server. At some point you will need two, ten, or more. The same is generally true of storage, networking, and other scaling points.

A starting list of scaling points includes:

- Storage capacity (GB)
- Server processing (CPU cycles) & RAM capacity (GB)
- Network bandwidth (Gbps)
- Database transactions per second (TPS)
- Storage input/output operations per second (IOPS)
- Web server threads or application processes (concurrency)

This is far from a complete list. Understanding how your web application consumes resources, how it behaves under high load, and what happens for all of the potential scaling points of the entire system is critical to understanding how your web application performs.

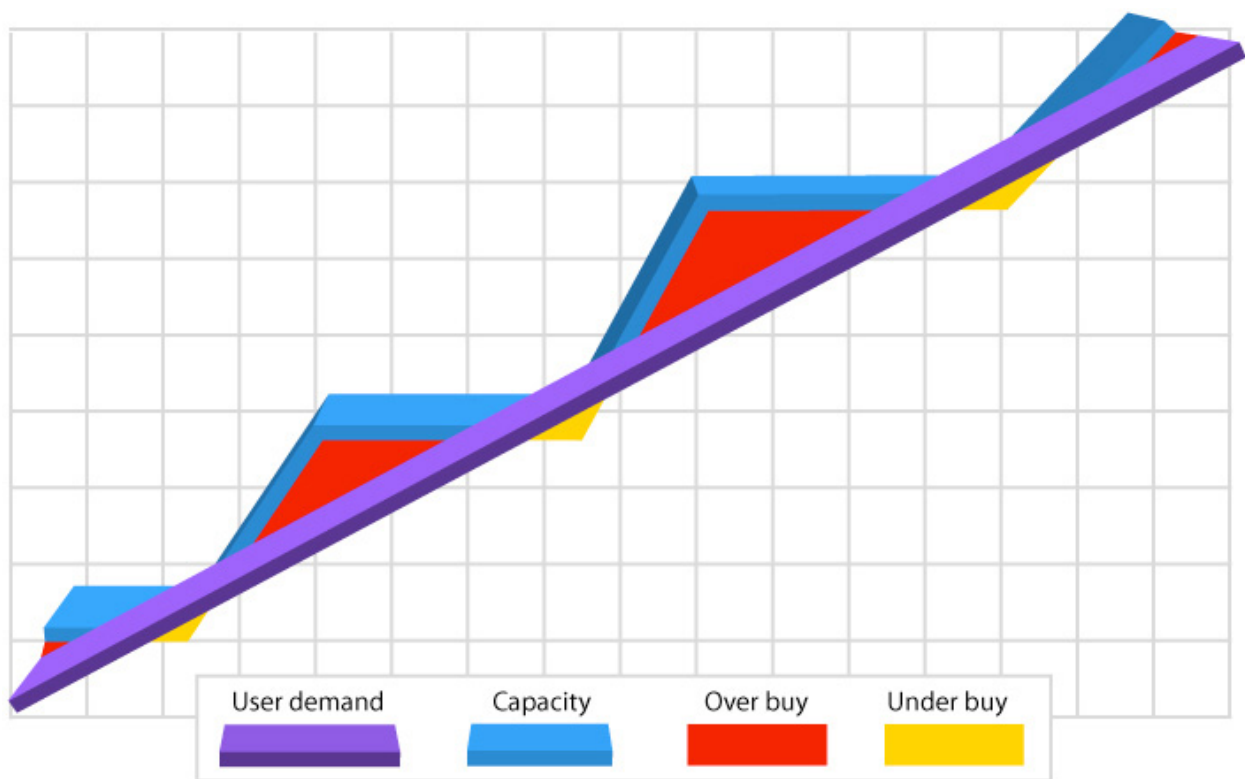
Well-designed web applications allow you to add more capacity to each scaling point as customer demand increases, growing your capacity to exactly meet demand.

**CLOUD COMPUTING & SCALABILITY**

Despite the hype, cloud computing is not a panacea. What cloud computing does provide are resources on-demand for many of the typical scaling points that a web application needs including servers, storage and networking. The on-demand nature of cloud computing combined with the pay-as-you-go model means that as your application demand grows, so can the resources you use to service that demand. In this situation, your capacity equals exactly your demand as long as your application was designed properly and its architecture is amenable to scaling well.

**SCALING BEFORE CLOUDS**

The diagram below highlights how web infrastructures were grown before cloud computing.

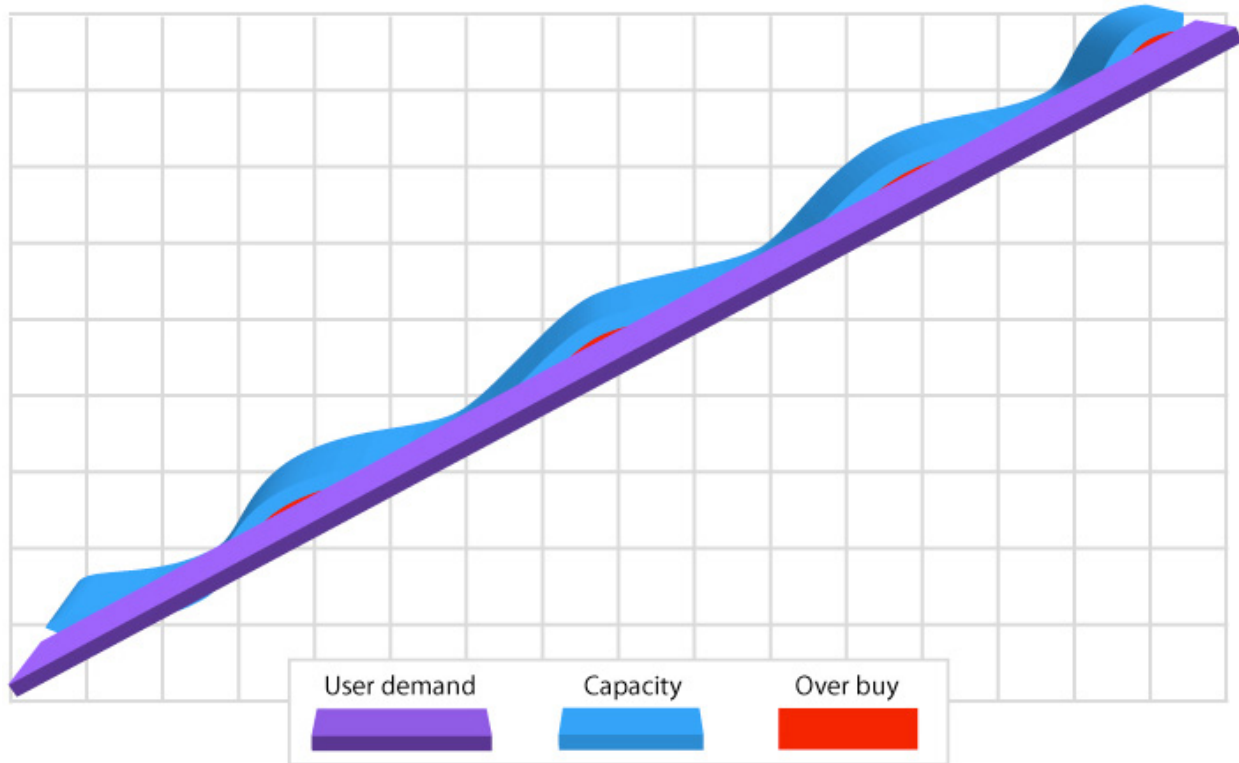


Essentially, you had to take a best guess at your ‘maximum’ capacity and build out to that capacity in advance because you had no way to add new resources on-demand. This meant you were in one of two modes: either under-buying or overbuying. When under-buying, your customers would see service impacts where your website was slow or even down. When overbuying, you frequently had unused capacity and hence wasted capital.

In best case scenarios, with great staff, very good capacity planning techniques, and some luck you could minimize the amount of overbuying, but the nature of hardware ordering and provisioning lead-times means that you were almost always purchasing 1-2 months in advance.

**CLOUD SCALING**

Using cloud computing it is possible to simply add capacity as needed, typically with lead times of only 10-20 minutes or less. Since you pay only for what you provision and use, your capacity can very closely match your demand curve like in the diagram below.



Clearly, cloud computing provides a new compelling mechanism for dealing with web infrastructure scalability, but you still need to make sure that your web application code and architecture are scalable as well.

Let’s dig deeper into how infrastructure scaling works.

## WEB INFRASTRUCTURE SCALABILITY

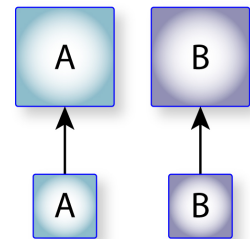
Fortunately, growing infrastructure to meet demand is a lot better understood for web applications today than it was in the past. Many successful web businesses like Amazon and Google have blazed trails so that we can understand how to overcome the challenges in growing web application architectures.

### TWO AXES OF SCALE

Historically there are only two dimensions in which we can scale a system: ‘up’ or ‘out’, also referred to as ‘vertical’ and ‘horizontal’.

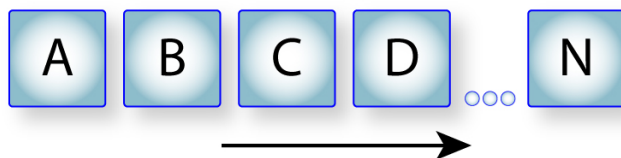
#### Vertical Scalability

Scaling up involves taking a single unit of resource (AKA ‘scaling point’) and making it larger. For example, you can buy a new larger server and replace an older slower server resulting in additional server capacity. The diagram to the right highlights this approach.



#### Horizontal Scalability

In contrast, scaling ‘out’ means adding more individual units of resource doing the same job. Instead of one server, you have two, ten, or more of the same server doing the same work, as illustrated below.



### Comparing Axes

It turns out that both techniques, up and out, have pros and cons that are important to understand. Typically vertical scaling is much easier as many applications are able to simply consume more resources when given them. It’s like having a bigger car engine.

Vertical scaling can be costly as bigger engines typically cost more. For example, a 400 horsepower car motor might be twice as powerful as a 200 horsepower car motor, but cost four times as much.

Horizontal scaling, on the other hand, allows each engine to remain the same cost while adding more individual engines. Imagine two 200 horsepower car engines instead of one single 400 horsepower engine. Unfortunately, we’ve discovered over time that many systems and applications are not efficient, by default, in using multiple resources in this manner. A car with two separate 200 horsepower motors? How do you combine two motors? You can assign one engine to each wheel. But what about when you have four? Or ten? Or a 100?

This table summarizes the differences:

Axis	How Scales	Scalability	Complexity	Capital Costs	Engineering Costs
Horizontal	More	Massive	High	Low	High
Vertical	Bigger	Moore’s Law-based <sup>1</sup>	Low	High	Low

**CHOOSING A SCALING STRATEGY**

It’s possible to design web applications to be efficient when scaling horizontally, but it turns out to be a non-trivial problem to solve, requiring good engineers spending a lot of time and resource making your application scale instead of working on core features<sup>2</sup>. This means that when you are considering how to scale your web applications you have to pick between whether you scale vertically, horizontally, or some combination of the two<sup>3</sup>.

Traditionally, most businesses have been best served by using vertical techniques as long as possible and then scaling individual parts of the web application horizontally when vertical is no longer sufficient.

By deferring the work necessary to scale horizontally you can spend valuable developer resources on product differentiation early in the life cycle of your business.

**Don’t Be Late**

It’s critical to understand that deferring horizontal scalability is not the same as ignoring it. Woe to the engineer, business leader, or architect that tries to bolt it on at the last minute. Some forethought should always be given to understanding your application’s scaling points in advance so that you are prepared when the time comes.

**REAL WORLD SCALABILITY**

Fortunately, there are some great examples in the real world of mid-size web application businesses that have successfully grown to meet their customer demands. These examples are far more accessible than trying to study an Amazon or Google, which while representing very large businesses, don’t represent the ‘norm’.

---

<sup>1</sup> Moore’s Law predicts that compute capacity doubles every 18 months: <http://tinyurl.com/6tt7p2>

<sup>2</sup> Sometimes referred to as ‘premature optimization’ where an overemphasis on performance, architecture, or scaling slows development times, impairs growth, and makes it difficult to capture early business value before competitors

<sup>3</sup> Occasionally referred to as ‘diagonal scaling’, but not by this author!

Let's take a look.

### **Vertical Scalability Example: 37Signals**

[37Signals](#) is a well-known Web 2.0 company and the creator of [Ruby on Rails](#). Their hosted small business solutions, such as [Basecamp](#) and [Campfire](#), offer robust software-as-a-service (SaaS) products to many. 37Signals has well publicized their use of vertically scaling. In this blog posting they highlight using large physical database servers and why they chose to defer 'sharding' (i.e. horizontal database scaling) as long as possible:

#### [37Signals vertical database scaling solution](#)

As you will see, 37Signals felt it was wiser to defer designing and managing extra complexity until much later. In fact, they believe that their growth will generally be outpaced by Moore's Law and hence may never need to scale their databases horizontally.

### **Horizontal Scalability Example: SmugMug**

Just like 37Signals, [SmugMug](#), a large and profitable photo-sharing site, scales their critical database servers vertically. However, SmugMug has found ways to leverage horizontal scaling by spending engineering resources learning how to use cloud computing systems to cost-effectively for their image processing system. Don MacAskill, SmugMug CEO and Chief Geek, provides details in the blog posting below:

#### [SmugMug's horizontal image scaling solution](#)

SmugMug's unique approach to processing images using elastic compute capacity has had a direct positive impact on their bottom line.

## GOGRID SCALABILITY OPTIONS

Today, GoGrid is unique amongst all cloud computing players in providing a full range of options for Internet businesses. We believe this rich product lineup allows customers to pick the solution that best suits their needs right now and in the future. It may not be sufficient to shoehorn your particular web application into a strict cloud computing model. Perhaps you need to 'go vertical' longer? Or perhaps you need a hybrid approach where some capacity is goes vertical and some horizontal.

Regardless, only with GoGrid do you have the option to pick what is right for your business needs. Let's look at all of your options for scalability with GoGrid:

Resource	Attributes	Use
Small Cloud Servers (.5-2GB RAM)	1 CPU core, medium I/O throughput	Web and application servers
Medium Cloud Servers (4 & 8GB RAM)	3 or 6 CPU cores, high I/O throughput	Application servers and small to medium databases
Dedicated server + Local RAID	8-16 CPU cores, extreme I/O throughput	High end databases and file servers
GoGrid Cloud Storage (NAS)	Infinitely scalable mountable storage with medium to high I/O throughput	Nearline storage, archives, backups, and low to mid-range database storage
GoGrid Load Balancers	High speed, high throughput network I/O	Load balancing
Co-located servers	Customizable	Customizable

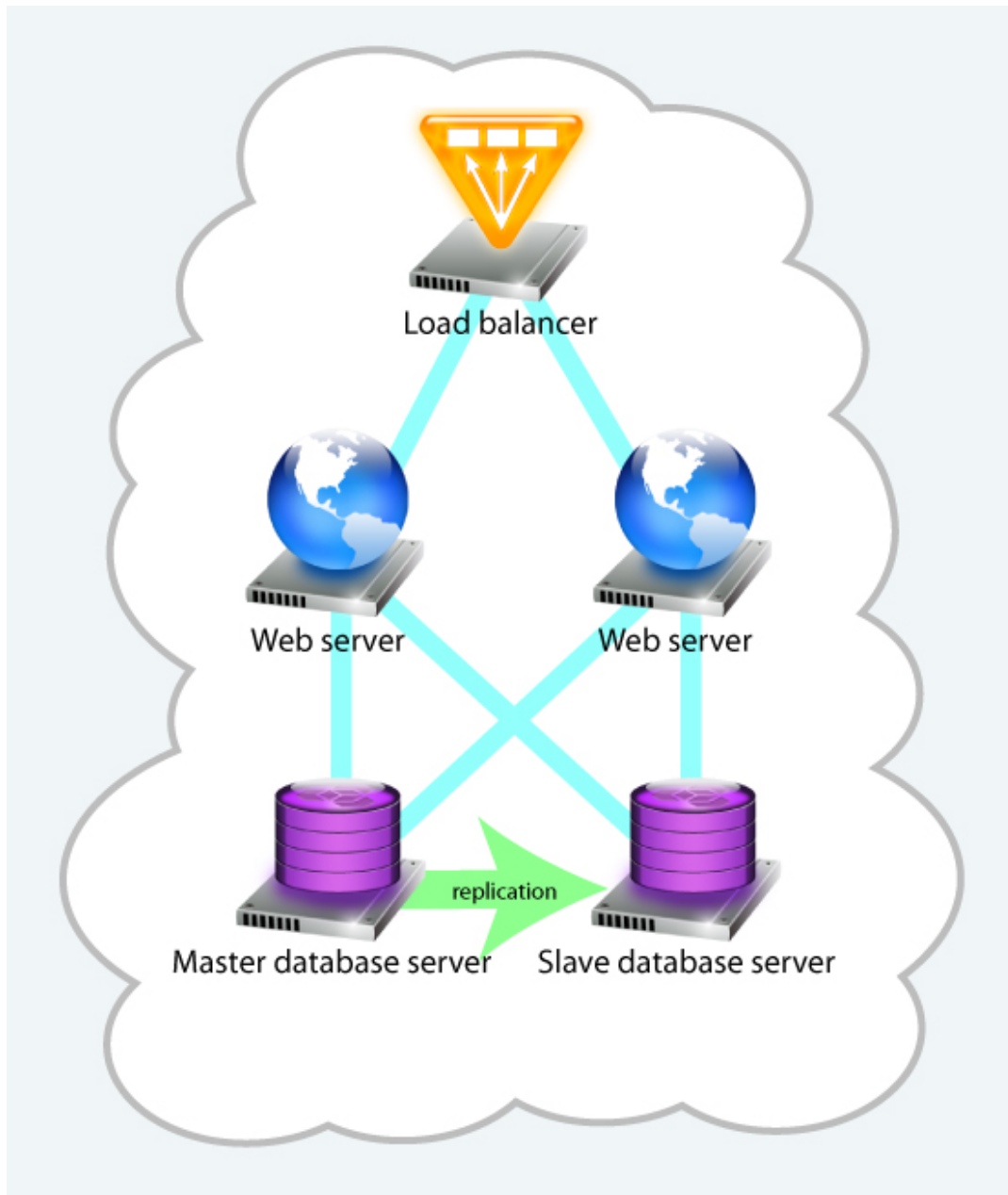
You can use any or all of the above resources to enable your web application infrastructure. Start slowly with a handful of GoGrid small cloud servers and as you grow replace them with larger ones. Once you reach the point where your database needs more I/O throughput you can move it to real dedicated hardware with high speed disks. Backup your virtual and dedicated servers to GoGrid Cloud Storage and use our free hardware load balancer to send traffic to all of your web servers. Finally, once you've got a clear need for custom hardware or specialized solutions, GoGrid can provide co-located servers on the same network as your virtual and dedicated servers!

Let's walk through some examples.

**SOLUTION #1 — A SMALL WEB APP USING VIRTUAL SERVERS**

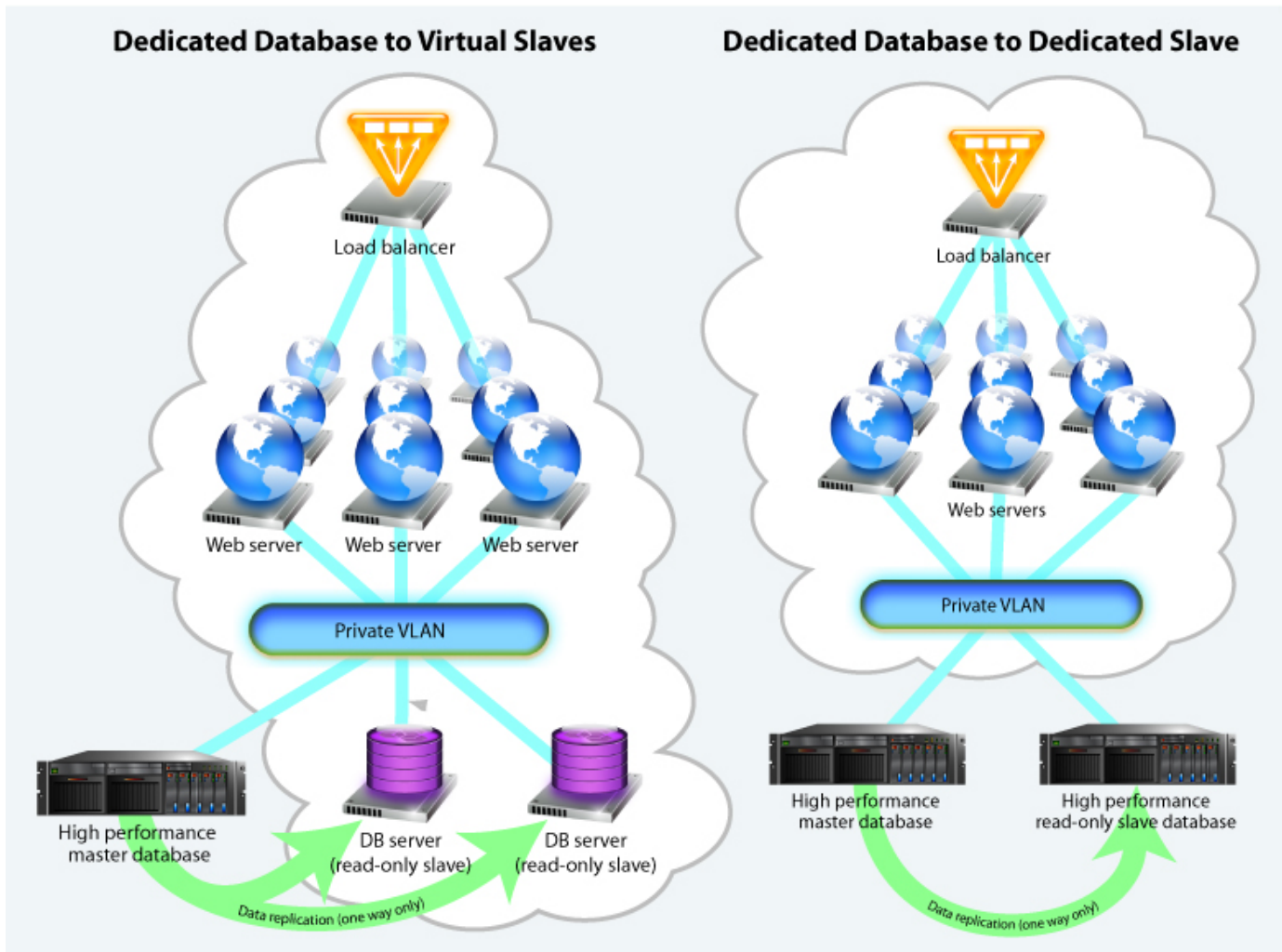
Smaller web applications can be easily served using GoGrid. The simplest case is two web and/or application servers, two database servers, and the free GoGrid load balancer service as shown in this diagram:

**Cloud Server Only Solution**



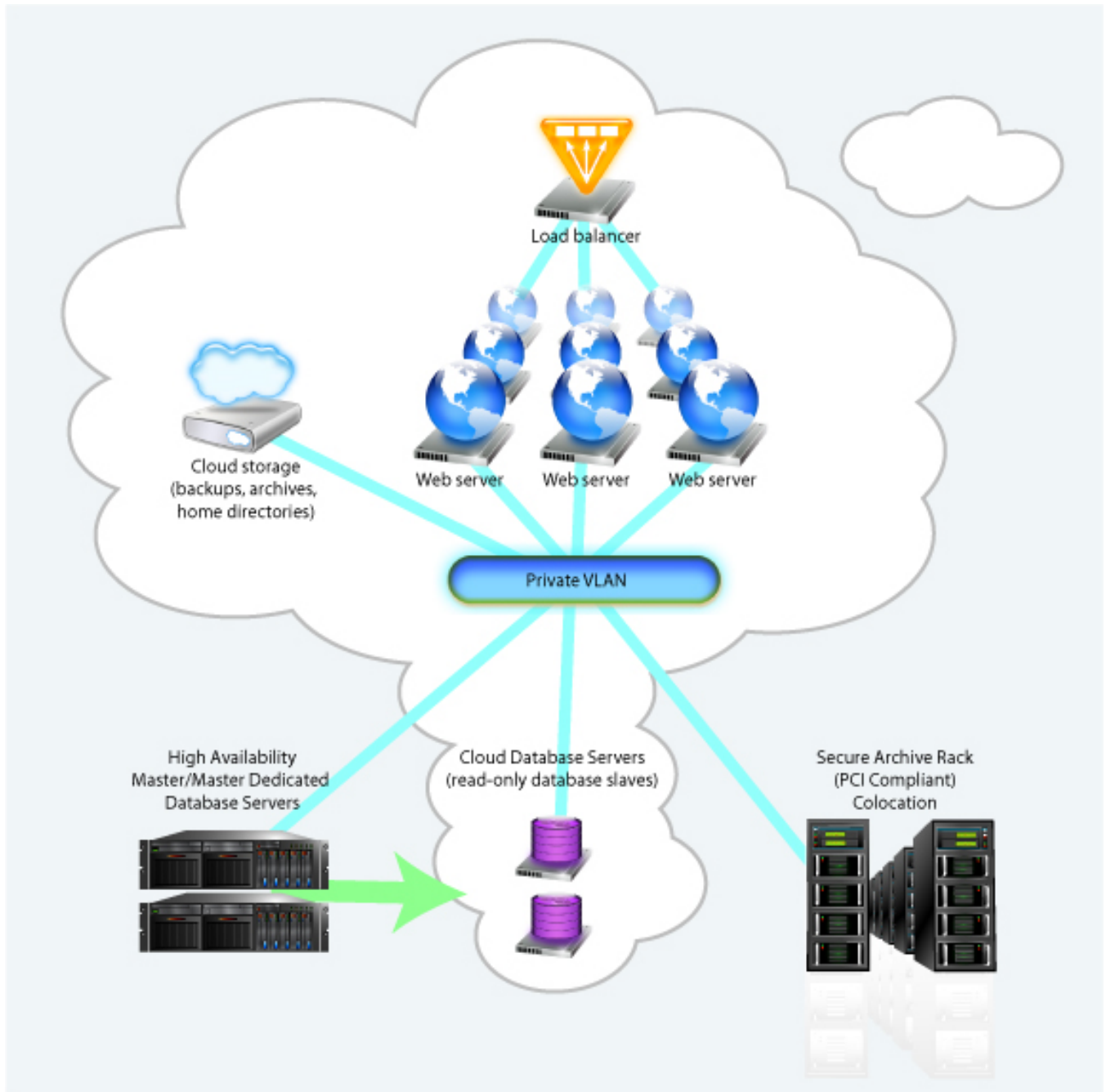
**SOLUTION #2 — VERTICALLY SCALING DATABASES**

More demanding applications can use our Cloud Connect service to add dedicated high-performance servers for high-end databases. You can start with a single dedicated database server that handles reads and writes. Once there is an even greater demand you can simply add cloud servers as read-only slave replicas. Of course, you can always add more dedicated servers when needed.



**SOLUTION #3 — COMPLEX SCALING REQUIREMENTS**

Sometimes you’re just going to have extreme requirements or need a high level of customizability. Using a combination of virtual, dedicated, and co-location services it’s possible to devise quite complex architectures. Like SmugMug, you can use virtual servers for batch processing, dedicated servers for high-performance databases, and co-location for when you need your own custom hardware.



## CONCLUSION

Your business is on the line and you must spend your time and resources wisely! GoGrid's expansive product offerings allow fast-growing Internet businesses a wide variety of tools for scaling web applications. By choosing the best solution and focusing on growing your business instead of re-architecting your web application it's possible to cost-effectively meet customer demand as you grow. Today, only GoGrid provides a complete, well-rounded product offering that enables your business objectives.

### ***ABOUT GOGRID ([HTTP://WWW.GOGRID.COM](http://www.gogrid.com))***

GoGrid is the leading Cloud Computing, hosted, Internet provider that delivers true "Control in the Cloud™" in the form of cloudcenters. GoGrid enables system administrators, developers, IT professionals and SaaS (Software as a Service) vendors to create, deploy, and control load balanced cloud servers and complex hosted virtual server networks with full root access and administrative server control. GoGrid server instances maintain the industry standard specifications with no requirement to learn and adapt to propriety standards. Bringing up servers and server networks takes minutes via a unique web control panel or GoGrid's award winning API. GoGrid delivers portal controlled servers for Windows Server 2003, Windows Server 2008, SQL Server, ASP.NET, multiple Linux operating systems (Red Hat Enterprise and CentOS) and supports application environments like Ruby on Rails. Free f5 hardware load balancing and other features are included to give users the control of a familiar datacenter environment with the flexibility and immediate scalability of the cloud, a "cloudcenter." GoGrid won the coveted 2008 LinuxWorld Expo's Best of Show award.

### ***ABOUT SERVEPATH ([HTTP://WWW.SERVEPATH.COM](http://www.servepath.com))***

ServePath, a Microsoft Gold Certified Partner, is the leading managed and dedicated hosted server provider, delivering custom solutions and managed services to businesses that require powerful Internet hosting platforms for their production environments. Thousands of companies worldwide look to ServePath for its reliability, customization, and speed. ServePath has a Keynote-rated A+ network and guarantees uptime with a 10,000% guaranteed™ Service Level Agreement. The employee-owned company has been in business for nine years and operates its own San Francisco data center and is SAS70 Type II certified.

## RESOURCES & REFERENCES

For further reading, we recommend the following references and resources:

### **BOOKS**

[The Art of Capacity Planning: Scaling Web Resources](#) by [John Allspaw](#)

[Scalable Internet Architectures](#) by [Theo Schlossnagle](#)

[Building Scalable Web Sites](#) by [Cal Henderson](#)

[High Performance Web Sites](#) by [Steve Souders](#)

[Release It!](#) by [Michael Nygard](#)

[High Performance MySQL](#) by [Jeremy Zawodny](#) and [Derek Balling](#)

[Cloud Application Architectures](#) by [George Reese](#)

### **BLOGS & WEBSITES**

[GoGrid Blog](#) by [Michael Sheehan](#) & [Randy Bias](#)

[neoTactics](#) by [Randy Bias](#), VP Technology Strategy, GoGrid

[Nati Shalom's Blog](#) by [Nati Shalom](#), CTO Gigaspaces

[Scalable Web Architectures](#) by [Royans Tharakat](#)

[High Scalability](#) by [Todd Hoff](#)

[The MySQL Performance blog](#) by various