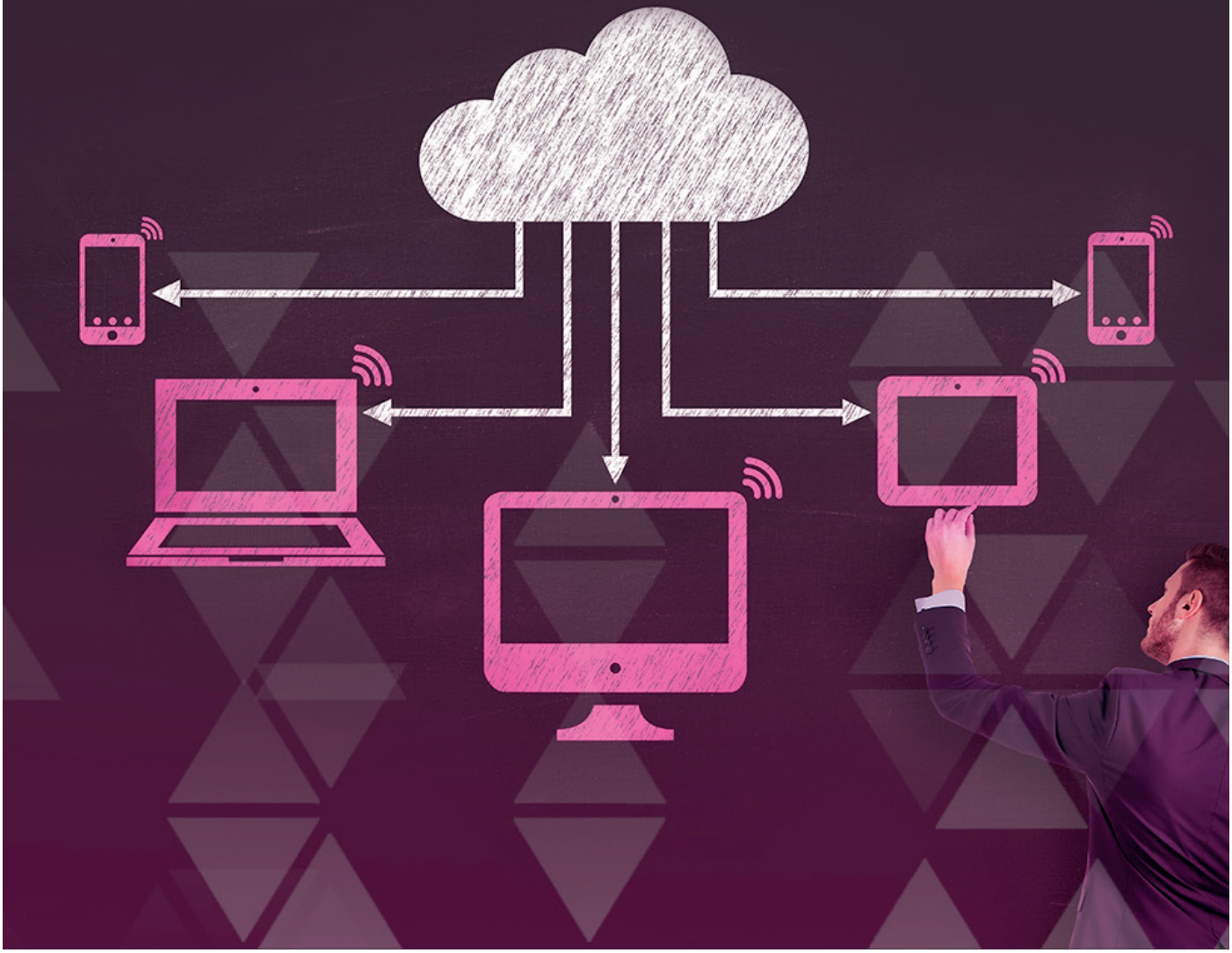
Building scalable and highly available websites in the cloud.

A short guide to making cloud-aware Internet applications for the 21st century.



LIFE IS FOR SHARING.

THE CLOUD ERA

There is no doubt about it. The age of the cloud is definitely here. The flexibility offered by virtual servers is impossible to resist: You can self-provision one in minutes, and scale its resources as needed. It is also safer to have a cloud provider host your applications, because it will most likely have more resources, knowledge, and experience in guaranteeing the security of the necessary infrastructures than most organizations could ever find in-house. The true beauty of the cloud, however, lies in the agility it brings. Businesses, regardless of size and segment, need to be able to provision IT services quickly and efficiently to keep up with the accelerated rate of change we now see all around us.

Internet applications, such as portals or e-shops, were among the first to make the shift to the cloud. They require a strong and stable Internet connection, which is the traditional domain of cloud service providers. Most online outlets, however, are still being built on a classic one-server architecture with limited scaling and availability.

This e-book discusses the high-level technical specificities of building and running scalable, highly available Internet applications.

It describes an architecture for horizontally scaled websites built on a cloud platform that can satisfy even the most demanding operating requirements. The constantly changing market conditions, the dynamics of Internet traffic, but also the growing number of attacks on online infrastructures make exactly this kind of agile platform indispensable.

> "The flexibility offered by virtual servers is impossible to resist."

TWO WEBSITE ARCHITECTURES

From an architectural point of view, all websites currently fall into two categories.

Monolithic applications

Most standard sites are run on this architecture, which is just a single virtual or dedicated server running both a database and an application engine that serves the web requests. The economic advantages of this type of platform are undeniable, and it is sufficient for many small and medium-sized e-shops or portals. The trouble starts when the server's size or performance limits are reached.

The single server can only be scaled vertically: In other words, memory, CPU, or hard drives must be added to it, which in the case of a physical machine means a considerable amount of downtime and possibly hitting up against scale limits. It is much simpler to add resources to a virtual server, but still requires a shutdown. Worse still, adding computing power may not always be enough to ensure the required additional web requests are actually processed.

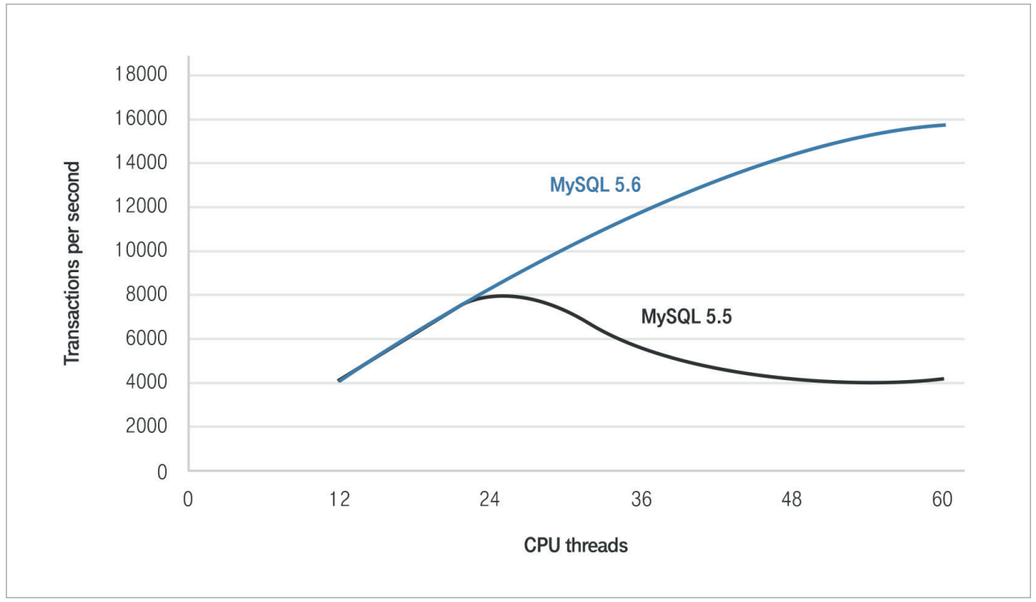
This is because a computer application's performance does not scale in linear fashion just by adding hardware resources. The graph below illustrates the fact that two versions of the Internet's most popular database, MySQL, do not scale performance proportionally to added CPU resources. In the case of the older version, performance even declines beyond a certain number of added CPU threads.

When it comes to continuous operations on a single server setup, it's simple: The website dies with the server. To achieve a high level of availability, complex, expensive, and performance-degrading cluster applications have to be added to replicate the data and functionality to secondary servers, which often just sit idle in standby mode.

Cloud-enabled applications

Cloud-enabled apps are designed from the ground up to be operated in the cloud. While the monolithic server stands alone and knows nothing about the underlying infrastructure, the true cloud server is a member of a group, and is aware of its other members, and their role. The number of participants in the group can grow or shrink as needed, for example to scale performance or jump in for a member experiencing problems.

The resources inside an individual server do not need to be altered to achieve scalability or continuous operations.



PETS VS. CATTLE

Pets are servers that are treated as unique and indispensable systems that require individual care. We give them “names” and they are manually built, managed, and “hand-fed”.

Cattle have no name or identity, they are just numbers in a herd. These are arrays of servers designed to sustain high loads or failures, where no single server is irreplaceable. They are usually built using automated tools, so if existing servers run into trouble or together are not enough to facilitate the requests, more are added automatically.

> ” Running an Internet application on an agile platform is a must. “

AGILE IT INFRASTRUCTURE

Running an online application requires a flexible and agile infrastructure that can react quickly to changes. The Internet is a wild and dynamic place, which keeps evolving and affects any applications opened to it. Running an Internet application on an agile platform is a must nowadays, and here are just a few of the reasons why:

Rising loads

A sudden surge in the number of visits to a website thanks to marketing activities is always a pleasant experience, as long as your monolithic server can handle the load associated with the increased traffic. If not, adding more computational capacity should not tie up the hands of administrators or have to be done manually.

Configuration changes, development, updates

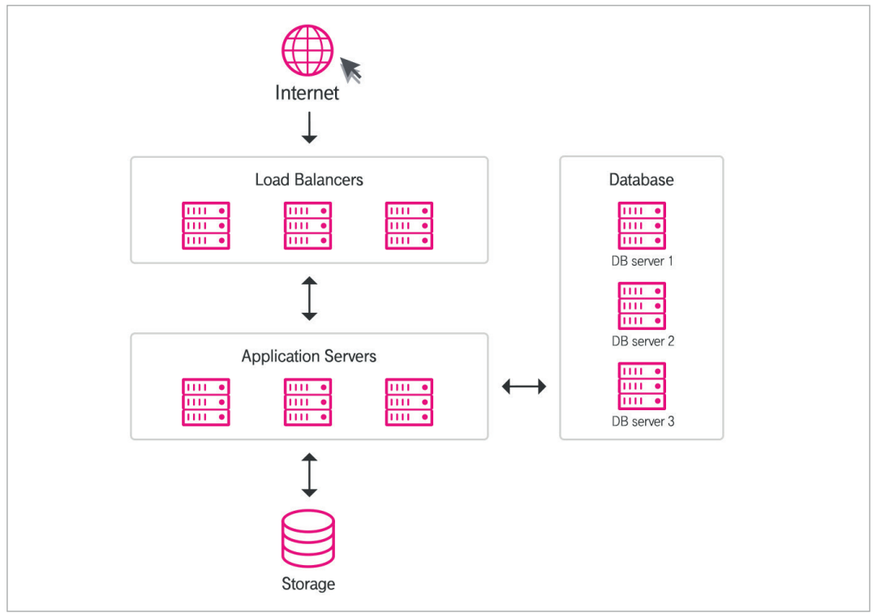
While updating a monolithic server is typically a daunting task, it should be a routine procedure that can be done whenever it’s necessary in an agile IT infrastructure. Only then can the system be fully trusted.

DDoS/botnet attacks

These are automated attacks that scan the Internet and attempt to find vulnerabilities or crack passwords. They can quickly use up all of the resources available to a monolithic server.

DESIGNING
HIGH AVAILABILITY

Here is what a cloud-based web infrastructure designed for high availability should look like:



Load balancers

They are a group of servers closest to the user that form the presentation layer. They serve data from application servers to user browsers. Load balancers handle HTTPS encryption, compression, cache, and serve static content. Most importantly, though, they distribute incoming traffic between application servers and monitor these to ensure that user requests are always directed to a healthy server.

User browsers typically round-robin through load balancers with different IP addresses based on DNS ANAME records. If a load balancer is unreachable, clients will simply retry the next address. It is also possible to design a hot-swappable standby, which assumes the IP address of a downed machine

Application servers

They process the request coming from the load balancers in the language of the web application, such as PHP, .NET or Ruby. Application files are typically accessible through a distributed file system, so that each application server can see the same data.

To achieve high availability, this layer again needs to be designed in such a way as to ensure that machine failures are handled properly. In most cases, the load balancers solve an issue with an unresponsive application server simply by detecting it and removing it from the rotation. However, if the application uses session persistence – which means that the whole user visit, such as a shopping basket in an online store, is tied to a particular application server – then special measures have to be taken to ensure seamless operation. In this case the session is typically stored in a shared database, so that another application server can pick it up if needed.

Databases

The backend layer stores files with data content. This may be a standard relational database such as MySQL, Oracle, or PostgreSQL, or just a storage server which stores whole files, such as media files. Object storage is becoming particularly popular as a simple and convenient way of storing data as objects in a scalable, high-availability scenario fully managed by a cloud provider.

Achieving continuous operations for the relational database layer requires automatically replicating a master database to a pool of slaves. Many cloud providers do this with a set of preconfigured templates or by providing highly available databases as a service. To scale reading from a relational database, it is recommended to launch multiple slaves and round-robin through them just like through stateless application servers.

Modern sites use non-relational NoSQL databases, which are designed to run in a distributed cluster, so scaling and high availability are handled by simply adding new servers to the deployment. Any new application developments requiring scalability and/or availability should consider using NoSQL databases.

AUTOMATION

The cloud is all about making IT resources available on short notice. Modern cloud providers make it possible to provision IT resources not just via intuitive web interfaces intended for human users, but also via public application programming interfaces (APIs). As a result, the task of installing a new server, which previously took people days to do in the physical world, can now be done by a line of code inside a program. Once a blank server is created, however, it is impossible to have an agile infrastructure without automated configuration management inside the server itself. That's why installing and configuring operating systems and other software infrastructure manually is rapidly becoming a thing of the past.

Automation tools such as Puppet, Chef, or Ansible are gaining broad popularity. They are used to install servers, implement changes, and keep the changes consistent. One obvious advantage of automated administration is that a task can be replicated and repeated as often as it is necessary.

CONCLUSION

All organizations and businesses, even the traditional brick-and-mortar industries, are suddenly facing new competition in the form of online companies. To remain profitable, they must change their current online services and introduce new ones in a truly agile, scalable, and "always-on" manner.

Luckily, the cloud era opens up revolutionary new possibilities for building and managing large or mission-critical websites. It should be clear by now that the monolithic architecture consisting of a single, irreplaceable server used to run these sites does not meet current requirements for a strong and dynamic online presence. That's why it is necessary to understand the existing options and consider using an automated, horizontally scalable, and highly available cloud platform to build and run your online sites. It's the only good way to build websites suitable for the 21st century.