PRESSIDIUM®
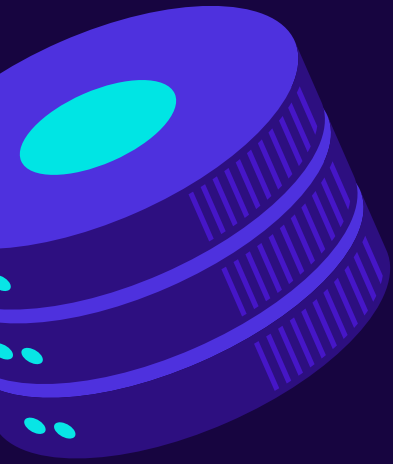
AN INTRODUCTION TO HIGH-AVAILABILITY DESIGN

For WordPress Developers!
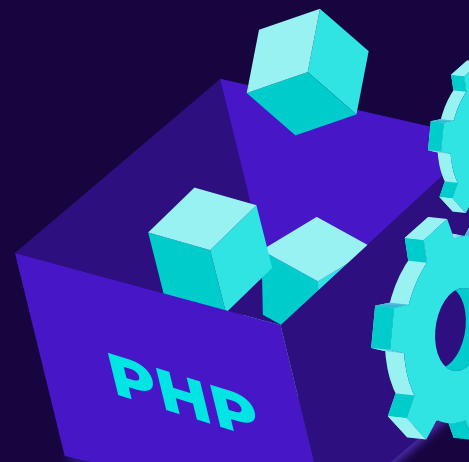
and not only!

PHP

Project Direction: Alexander Newnham
Design & Illustrations: Ioanna Aravani
Special Contributor: Linode

Introduction

Organizations can't risk having their website down for even a second. But, there is no hardware, no software, and no connection that is truly 100% reliable. WordPress relies on many interdependent components to deliver a truly powerful content management system.

Each component needs careful consideration so that in case of an unplanned failure or scheduled maintenance, a website can continue running seamlessly, and users will never notice. This type of careful planning is at the heart of High Availability: the ability to mitigate planned and unplanned failures with intentional system design principles.

An Introduction to High-Availability Design explains common design paradigms used to mitigate the effects of unplanned system failure. This ebook introduces three fundamental principles of high availability, common implementation architectures, and explains how WordPress behaves in highly-available deployments. It's an insightful overview for business owners who want a glimpse of what goes on behind the scenes of a WordPress provider like Pressidium. An Introduction to High-Availability Design also offers systems architects, engineers, and administrators suggestions so they can mirror the practices that Pressidium has perfected over the past two decades.

WordPress is no longer just a blog platform. It's now the largest content management system, which offers support to an almost infinite library of plugins and features that have given millions around the world the power to create highly-customizable websites for their audiences.

With the ability to add features and components, the need for a partner like Pressidium becomes more and more apparent. Adopting High Availability is a decision done at the business level, and it's a decision that everyone should have a say in. An Introduction to High-Availability Design will help your business make the right choice.

At Linode, we are no strangers to High Availability. We have worked tirelessly to provide our customers with a 99.9% uptime guarantee. Our promise to provide simple, affordable, and accessible cloud computing services for customers around the world wouldn't be possible if we didn't practice the principles presented in this ebook.

We've heard directly from our user base of over 800,000 developers who choose the Linode Cloud how important building highly available, fault-tolerant systems is to them. In addition to An Introduction to High-Availability Design, Linode has published our own guide, An Introduction to High Availability, which along with this ebook from Pressidium, will result in understanding how to build durable systems in the cloud, so they can operate continuously without failure.

Mike Quatrani, Solutions Engineer, Linode

We are going to talk about:

1. What is High Availability?

2. How does High Availability mitigate failures?

3. How is High Availabilty implemented?

4. Examples of how WordPress behaves in High Availability systems.

5. And in closing, how to decide whether your business needs High Availability or not!

What is
High Availability?

Let's begin with a somewhat
informal definition.

server, service

High availability is the ability of a system or component to be up and running, despite catastrophic failures.
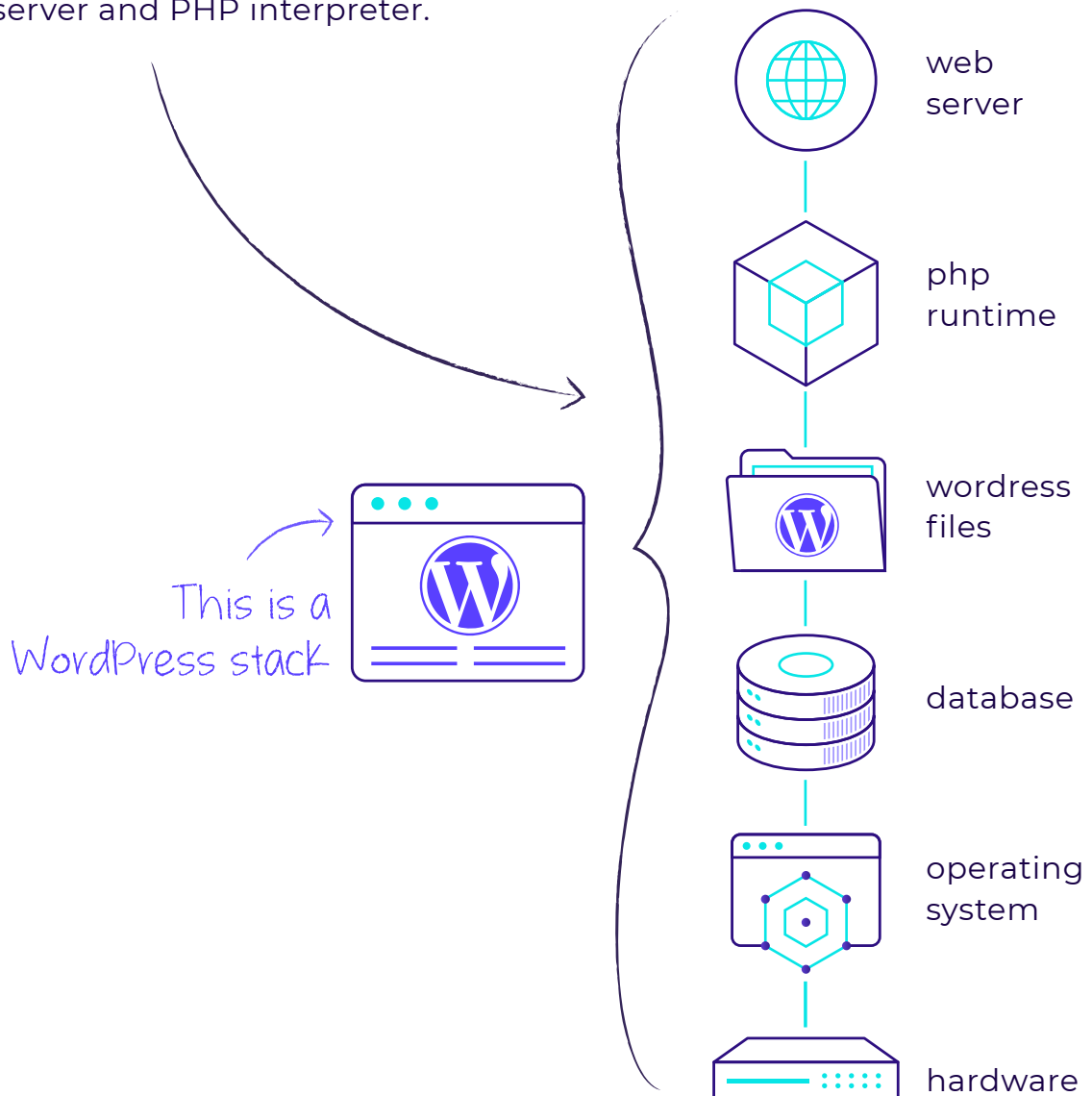
failed hardware

broken system upgrades

A stand-alone wordpress site

All WordPress sites can be thought of as a stack of components, starting from the hardware server and operating system at the base, all the way up to the web server and PHP interpreter.

This is a WordPress stack

web server

php runtime

wordress files

database

operating system

hardware

Planned and unplanned failures

Any of these components can fail at any time, rendering your wordpress site inoperable.

For example:

Power supply units get burnt, network ports go offline, hard disks fail.

- The server can experience hardware failures.

- Mistakes in configuration can cause outages.

- The wordpress database connection can be completely cut-off, and so on.

Services stop or become unresponsive, the OS gets mangled after a failed update, etc.

These types of unexpected failures are called unplanned. Failures caused by scheduled maintenance procedures on the other hand, such as stopping and starting services, replacing components, etc, are called planned.

High availability mitigates both of these types of failures.

How does High Availability Mitigate failures?

We take a look into the heart of High Availability.

Three Fundamental High Availability Principles

By using three fundamental system design principles:

✔ **Component redundancy.**
Every component has at least one backup.

✔ **No single points of failure.**
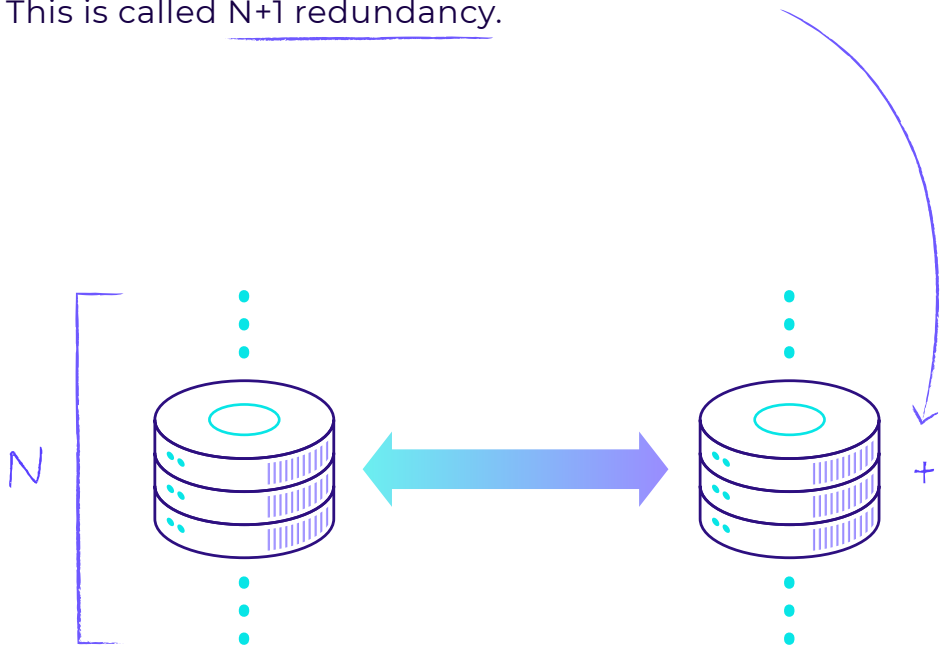There is no single component whose malfunction can cause complete system failure.

✔ **Failure detection and response.**
Component failures are detected on time, and dealt with accordingly.

Component redundancy

For every component, there is always one extra.
This is called N+1 redundancy.

Data and configuration is replicated to all, particularly the
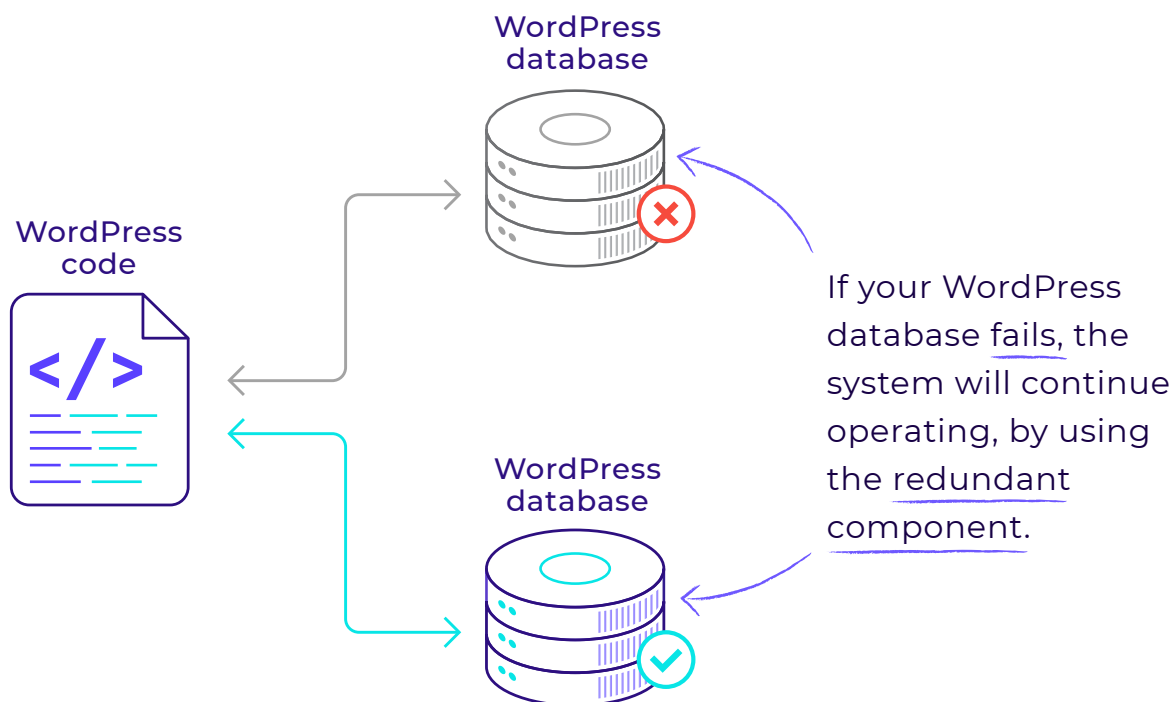WordPress database and filesystem.

This ensures consistency between all WordPress stacks.

Due to frequent updates.

No single points of failure

A single-point of failure is a component whose malfunction can cause total system failure. One such example in WordPress, is the database.

High Availability designs implement component redundancy, in order to eliminate single points of failure.

WordPress database

WordPress code

WordPress database

If your WordPress database fails, the system will continue operating, by using the redundant component.

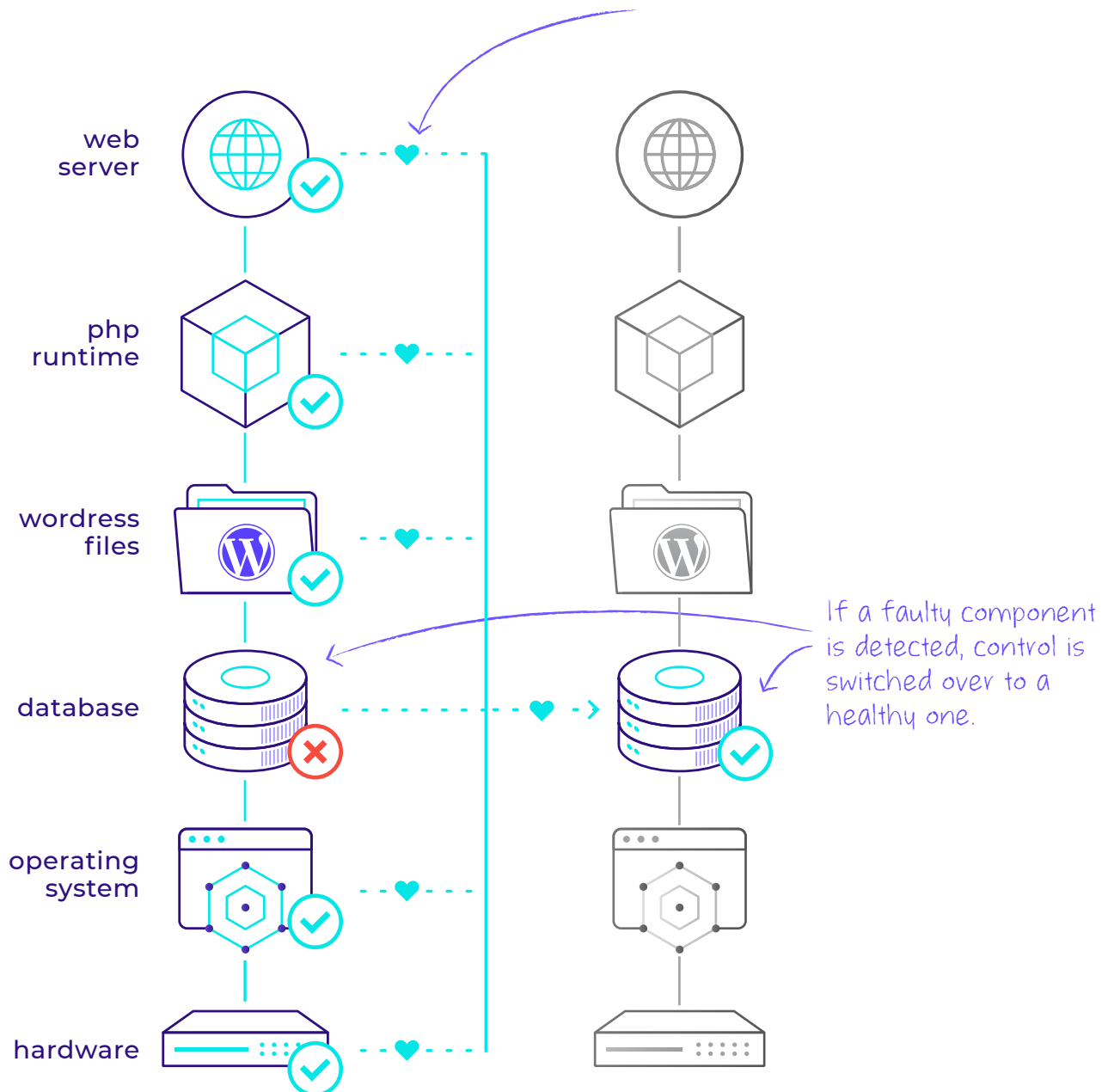For that to happen however, a third key ingredient is needed: Detecting and responding to failures.

Failure detection and response

High Availability designs implement a failure detection mechanism. This mechanism constantly checks for the good health of components.

web server

php runtime

wordress files

database

operating system

hardware

If a faulty component is detected, control is switched over to a healthy one.

How is
High Availability
Implemented?

There are lots of different ways,
we take a look at the most common two.

Two most common architectures

Earlier we talked about N+1 component redundancy, and how a redundant component can function as backup in case of failure. There are two most common ways of implementing this:

Active/Passive

- In Active/Passive architectures, there is at least one WordPress stack, that is called active, and one that is called passive. The active stack is serving live traffic requests, while the passive waits on standby.

- Each WordPress stack checks the health of the other, by sending "heartbeat" signals, via a private communication link.

- If the active WordPress stack fails, the passive one becomes active, and takes its IP address.
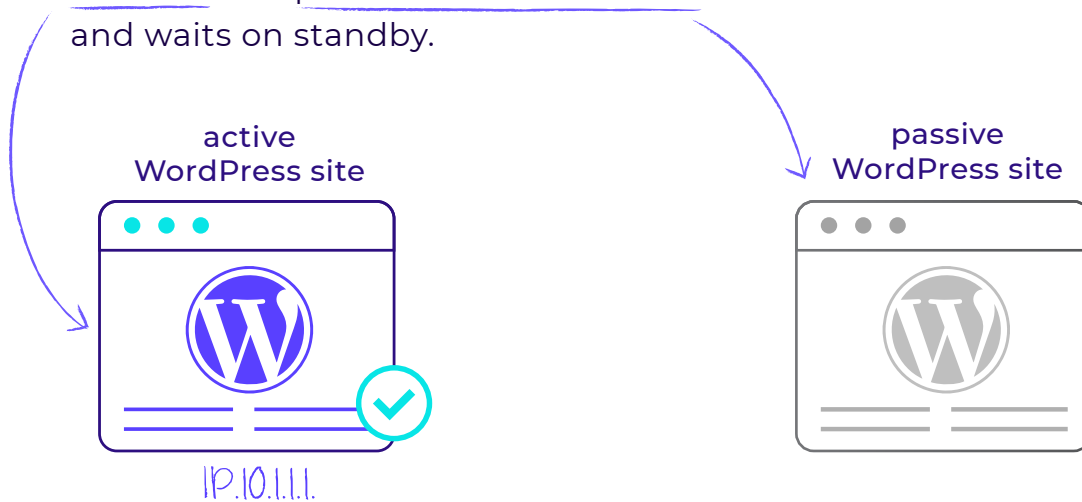
Active/Active

- In Active / Active architectures, all WordPress stacks are active and members of a pool.

- Health monitoring is done not by the WordPress stacks themselves, but by a device called a load balancer. *We'll talk about it later!*

- If a WordPress stack fails, it is removed from the pool. The rest of the active stacks continue their normal operation.

Active / passive

Let's see an example:

1 A WordPress site is served by the active WordPress stack with IP 10.1.1.1. The passive WordPress stack contains an identical copy, and waits on standby.

active
WordPress site

passive
WordPress site

IP.10.1.1.1.

2 When a failures happens, and the active stack becomes unresponsive, the passive stack wakes up, and becomes the active.

3 passive (disabled)
WordPress site

active
WordPress site

IP.10.1.1.1.
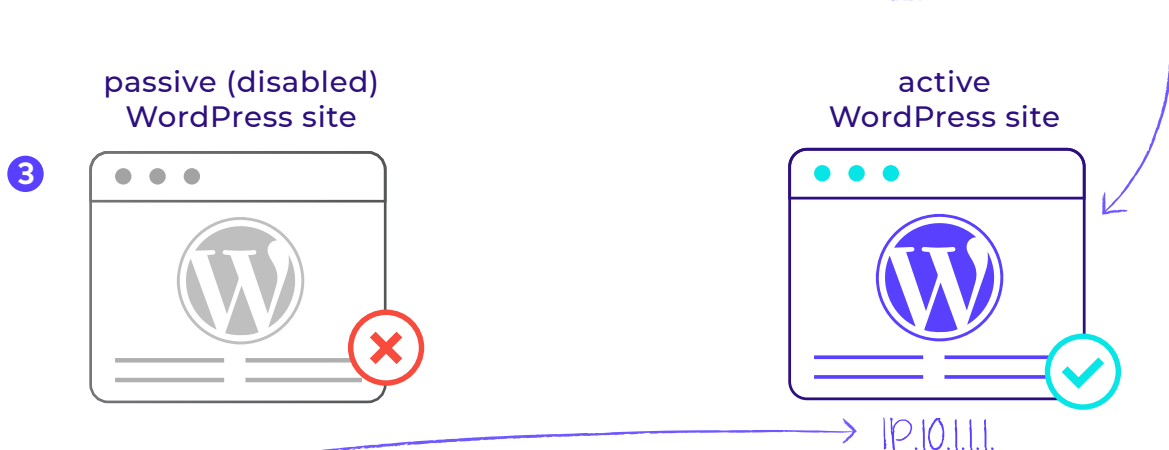
The IP address and the traffic of the malfunctioning stack is transferred over to the newly active one.
IPs capable of doing that are called floating IPs and play an important role in High Availability design.

Active / Active

In Active/Active architectures, there is at least one device called the load balancer, situated in front of the web server layer. Load balancers receive external traffic and attempt to distribute
it evenly across all healthy WordPress stacks.
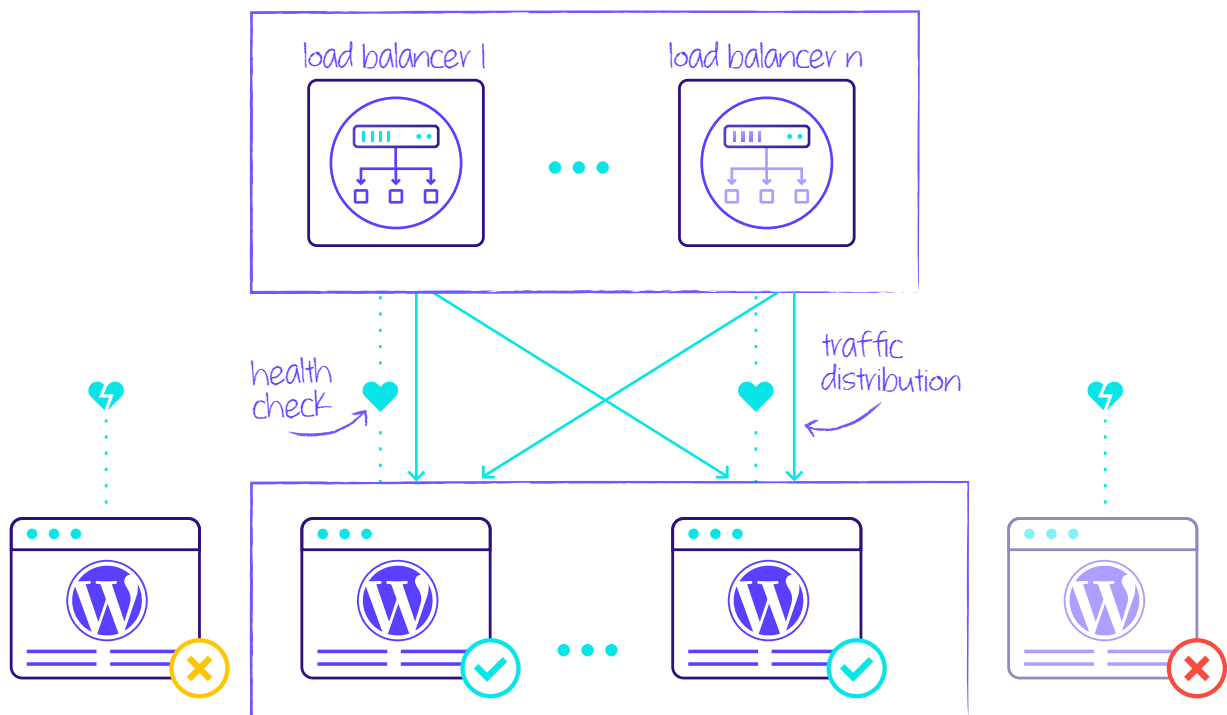
- ✓ If a WordPress stack is healthy, traffic is sent to it normally.

- ✗ If a WordPress stack seems slow but it's otherwise healthy, it is temporarily removed from the pool, until it recovers.

- ✗ If a WordPress stack is unresponsive, it is temporarily removed from the pool and no traffic is sent to it, until it recovers.

Advantages & Disadvantages

In Active/Active architectures, there is at least one device called the load balancer, situated in front of the web server layer. Load balancers receive external traffic and attempt to distribute
it evenly across all healthy WordPress stacks.

By using three fundamental system design principles:

ACTIVE / ACTIVE

✓ **Advantages**
Active/Passive architectures are easier to setup, and maintain.

✗ **Disadvantages**
They do not scale as easily as Active/Active architectures.

ACTIVE / PASSIVE

✓ **Advantages**
Active / Active architectures scale easier than Active/Passive ones.

✗ **Disadvantages**
They are significantly more complex to setup than Active/Passive ones, due to the extra load balancing layer.

How does WordPress behave in High Availability?

In this slide, we pull everything together to show you an example of Highly-Available WordPress design.
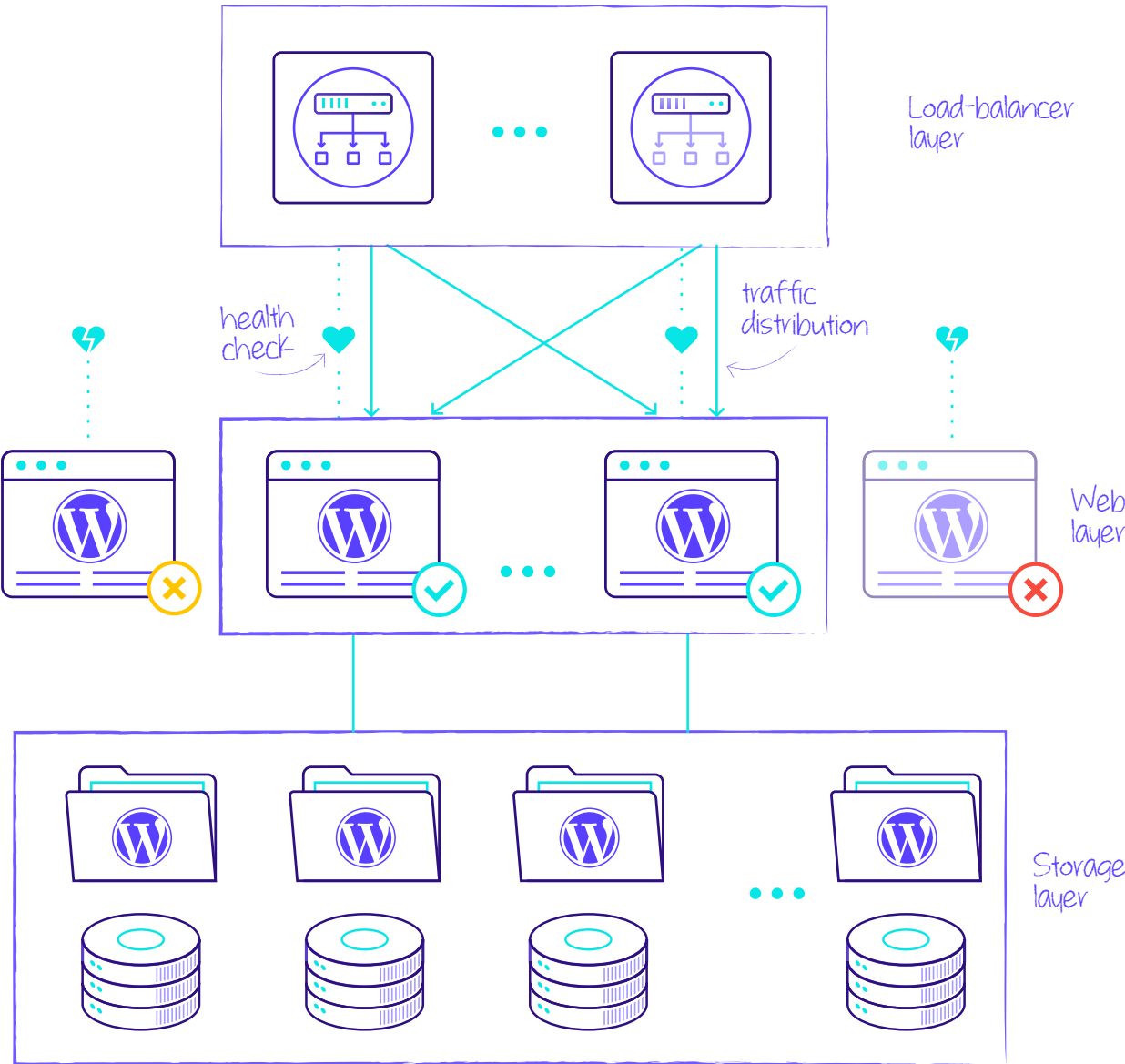
Highly-available WordPress

We divide the design into three separate layers:

1) Load balancers.
2) Web servers.
3) Storage.

- Each layer is highly-available, using Active/Active, Active/Passive, or a mix of both architectures.

- Load balancers are network devices that monitor the health of the web servers and storage, and dynamically distribute the traffic accordingly. The web servers are responsible for serving the WordPress site, while the storage layer can be the filesystem, an RDBMS database like MariaDB, or even an in-memory one, like Redis.

Data and configuration is replicated across every layer. As we mentioned at the start, this ensures consistency between all WordPress stacks, databases, filesystems, etc.
This is critical , as the user needs to see the same WordPress site, no matter which WordPress stack is serving it.

Load-balancer layer

health check

traffic distribution

Web layer

Storage layer

The WordPress storage layer

Until now, we've conceptually discussed two layers (the load balancing one and the WordPress stack), thinking of storage to be a component inside every WordPress stack.

By putting storage in a distinct layer and separating it from the rest of the stack we gain **flexibility in maintenance.** Operations in the storage layer do not affect the other two, and vice versa.

This provides **greater resilience** to the infrastructure and **better management tools** for support.

Two examples of Wordpress High Availability

With this design in mind, let's see two examples of how WordPress behaves in HA:

- ✅ **When installing a plugin or a theme.**

- ✅ **When a user session is created.**

WordPress plugins in High Availability

1. A user is logged in one of the available WordPress stacks and downloads a plugin from **WordPress.org.**

User

WordPress stack

WordPress.org

Web layer

/wp content

/wp content
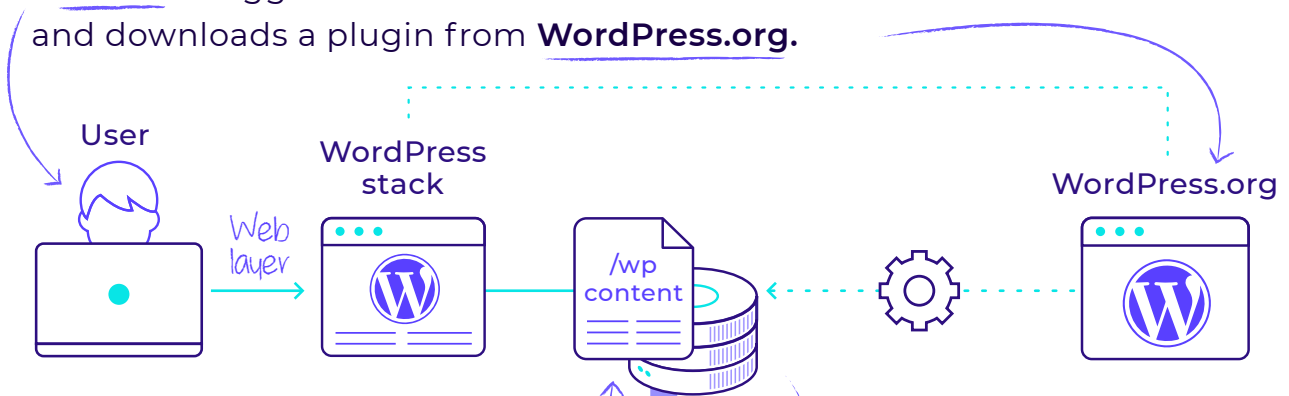
/wp content

2. The plugin is installed on the WordPress stack the user is currently logged in, inside **/wp-content/plugins.**

Data replication

4. In High Availability designs, data and configuration is replicated, thus the plugin is installed and operational in all WordPress stacks.

It is the same with WordPress themes as well.

3. On a conventional system if the user would log in to a **different stack,** the plugin would not be there.

Data replication

WordPress sessions in High Availability

Visitor

1. A visitor logs in to one WordPress stack that hosts the WooCommerce site, and creates a **WordPress session.**

The session data identifies the user, and is stored in a filesystem, or a database.

WordPress stack

WordPress session

This can be configured from PHP

2. However, next time the user logs-in, it might be to a **different WordPress stack** which does not contain the session data.

WordPress session

Data replication

3. Since all data and configuration is replicated, the session information is available to all WordPress stacks.

No matter in what WordPress stack the user logs in, the session information is available.

WordPress session

Software bugs and High Availability

Although High Availability ensures uninterrupted operation in the case of failures, **it does not protect you from bugs in the application level.**

If your code has bugs that can cause downtime, then these bugs will also be replicated across your WordPress stacks!

Does your business need High Availability?

Decide whether the cost and benefit of High Availability make sense to your business.

How much is too much?

Adopting High Availability is ultimately a **business-driven decision.**

It depends on two things:

- ✔ **How much revenue you potentially might lose in downtime.**

- ✔ **How much you are willing to pay, to mitigate that downtime.**

Let's examine a typical e-shop

Assume that you have an e-shop business that processes **10 orders per hour.** Each order brings **$20 of revenue**, for a total of **$200 per hour.**

A downtime of 2 hours a month, would then cost you **$400.**

Besides loss of revenue, downtime also brings loss of traffic, which is terrible if you are in a middle of a marketing campaign.
Downtime also negatively affects the opinions of your customers, and in extension, your **brand reputation and image.**

The Service-level Agreement

The **Service-level Agreement** is a document that commits the Service Provider to supply services to the client, that are of a specific quality and availability.

In hosting providers, the SLA also refers to the **guaranteed level of uptime.**

(!) However, every hosting provider calculates uptime in a different way! Find out if your hosting provider calculates uptime on a monthly or yearly basis, and whether maintenance downtime is included in the calculations.

☞ See if the uptime guaranteed in your hosting provider's SLA covers your needs as a business owner, and how much does that cost.

Some SLA numbers for comparison

The website **http://uptime.is** is a handy online uptime calculator. You can use it to calculate downtime for various SLA percentages.

For example, **99.9%** availability means the following potential downtime:

- Daily: 1m 26.4s
- Weekly: 10m 4.8s
- Monthly: 43m 49.7s
- Yearly: 8h 45m 57.0s

While a SLA of **99.5%:**

- Daily: 7m 12.0s
- Weekly: 50m 24.0s
- Monthly: 3h 39m 8.7s
- Yearly: 1d 19h 49m 44.8s

How to start thinking about It

You need to figure out two things:

☞ How much loss of revenue you are willing to suffer, due to downtime.

☞ How much uptime does your hosting provider guarantee, and with what cost.

High Availability becomes crucial, when you revenue depends on your website's uninterrupted operation.

Buyer beware!

SLAs in the hosting industry are quite different to those offered in the enterprise. For example, achieving a true uptime of 99.5% or more is difficult and costly. In the enterprise, the prices for such guarantees are of a higher magnitude. Similarly, SLA penalties are of equal weight, so organizations can mitigate their risk if the provider fails to deliver.

In contrast, hosting SLAs are usually not backed-up by significant penalties for the provider at all, thus the guarantee is of little actual value. There are also cases where downtime is not included in the total calculation: scheduled maintenance, and other similar activities, which are called exceptions. It's not uncommon then for 99.5% of daily uptime to mean 7 minutes plus several hours of downtime resulting from such exceptions.

Spend less time deciphering their SLA and more researching the **provider's infrastructure design, and uptime track record.**

Do not believe anyone who is selling you 100% uptime.

most hosting providers have a status page

Summary

- High Availability enables a system to be **up and running despite planned and unplanned failures.**

- Three fundamental design principles: **component redundancy, no single points of failure, failure detection and response.**

- Most common designs are Active / Active and Active / Passive. **Both have their advantages and disadvantages.**

- We've seen examples about how WordPress plugins, themes and sessions can become highly available. All data and configuration **are replicated across all WordPress stacks.**

- Adopting High Availability is **a business driven decision:** how much you lose in downtime vs how much you are willing to pay to reduce it.

- When choosing a provider, **inspect it's uptime track record and infrastructure design.**

- **Do not believe anyone that is selling you 100% uptime.**

About us

Pressidium is a technology company that builds powerful, secure and scalable WordPress infrastructure for businesses.
We are a tight knit of people with more than two decades of experience in high-pressure, mission-critical business IT environments.

By partnering with us you get access to:

- **Industry-leading, Highly Available WordPress** hosting regardless of your plan.

- **Multi-layered WordPress security:** locked-down PHP environments, security updates, malware, provisioning, network and Web application level firewalling.

- **24/7 Real WordPress DevOps support** by the same engineers who built the platform.

THANK YOU!

Discover the **award winning technology** behind
our Enterprise WordPress Architecture

Give us a shout at hello@pressidium.com
and take your WordPress site to the next level.

PRESSIDIUM®