The Snapshot Trap: Why Your Backup Might Betray You

Bernard Sfez - 2026-01-01 17:04

If you have some experience, you know that making backups isn't the same as being secure. You have probably already heard this reassuring line: "No worries, we run an automatic backup of all files every day." With a bit of bad luck, you may have already experienced the limits of that approach when it's applied indiscriminately, without oversight or control.

At OpenSource Solutions, our years of experience have taught us a hard reality: having a backup doesn't mean being able to restore. Day-to-day server and data operations require heightened vigilance, particularly when it comes to cyber resilience. One of the most insidious pitfalls is blind trust in snapshots or global (bulk) backups. In this article, we will provide a quick diagnosis of these practices and share our "recipe" to avoid losing anything—but above all, to ensure optimal operability in record time in the event of a disaster.

The Illusion of Safety

Unusable Snapshots

Many administrators rely on a scheduled daily full snapshot (an "instant snapshot"), assuming the job is done. Sometimes they even use incremental snapshots—where only changes are recorded—to save time. In our view, that is a fundamental mistake.

Why? Because a snapshot captures a system's state at a given point in time, but it does not guarantee application-level consistency. If your database is in the middle of writing critical information at the exact moment the snapshot is taken, the resulting data can be unreadable or corrupted. In technical terms, you end up with a crash-consistent backup (as if after a power outage) rather than a clean, application-consistent backup.

Without frequent restore tests, your backup strategy is like an insurance policy whose reimbursement terms you never bothered to read: it exists on paper, but it may fail you at the worst possible moment. How many administrators have attempted a critical restore only to discover—horrified—that their only recovery point was invalid, incomplete, or corrupted?

Remember this: If you haven’t tested your restore this month, you don’t have a backup—you have hope.

The “Bloated Snapshot” Trap—and Its Costs

The "global" approach backs up everything: the valuable, but also the useless. Backing up an entire system without discernment is like storing gigabytes of temporary caches, expired session files, or obsolete logs. The result is mathematical: a 500 GB snapshot instead of the 50 GB of data actually needed drastically extends your RTO (Recovery Time Objective). While it is very tempting to believe that pressing a button or running a single command will put everything back in place, real-world experience shows that this simplicity is deceptive.

Each unnecessary gigabyte becomes painfully expensive during a crisis:

- Extended restore time : Searching through massive archives and decompressing them consumes precious hours, during which your business is down.
- Insurance and compliance costs : Today, cyber insurers are increasingly strict. A backup strategy deemed "negligent" or untested can lead to doubled premiums, or even denial of compensation in the event of a claim.
- Restart and training expenses : After a complicated restore, it is often necessary to hire expert help to clean up data or to urgently train your teams on newly implemented security procedures.
- Operational losses and brand reputation : The cost of idle employees and the erosion of customer trust are the hardest losses to quantify, but often the most severe.

In the end, the real cost is not limited to the storage bill: it is measured in the mobilization of your technical teams and direct financial impact. An ineffective backup is a financial sinkhole that doesn’t look like one.

Multiple Scenarios, Multiple Responses: The Art of Anticipation

A good strategy does not merely store data “just in case”; it anticipates the nature of the crisis and responds with precision. For example, we have developed specific protocols for each type of unexpected event:

- Human Error (Surgical precision) : A user accidentally deletes a critical page or a configuration folder. Rather than restoring the entire server—an operation that is long, risky, and would overwrite healthy data created in the meantime—we perform a granular restore. We extract only the missing data from our targeted "dumps". It’s fast, clean, and has no impact on the rest of your operations.
- Cyberattack (Isolation as a shield) : In the event of ransomware, the malware encrypts everything it can "see", including your local snapshots. Our response? The 3-2-1 strategy with external copies, encrypted and, above all, disconnected (air-gapped). We restart from a clean baseline, out of reach of attackers, to rebuild your environment without paying a ransom.
- Major incident (Portability as survival) : The OVH data-center fire in 2021 was a reminder that a building can disappear. Our method does not lock you into a hosting provider. Thanks to our code management via Git and the externalization of media, we can bring up a fresh

server and your data with any other provider in under 3 hours.
- The unpredictable (Agility by default) : For all scenarios we have not yet imagined, we rely on standard, interoperable tools. No proprietary "black box" format: your data remains accessible and recoverable, whatever happens.

The conclusion is simple: your backup solutions must adapt to real-world conditions. The goal is always the same: restore your services as quickly as possible, with maximum effectiveness and minimal incident cost.

Integrity Uncertainty: Is Your Backup Actually Readable?

Beyond the storage method and bloat, the crucial question is data fidelity. A fatal mistake is to confuse “backup success” with “data integrity.” Your backup system may confirm that the file was created successfully, while the contents are unusable.

The risk of silent corruption is very real:

- Database consistency : A database is a living organism. If it is captured “cold” by a snapshot without the appropriate locking mechanisms, you risk restoring a database with broken indexes or inconsistent tables.
- File corruption : Between a minor hardware failure and the transfer to storage, a file can be altered. Without a verification system (such as checksums), you won’t know the file is corrupted until the final extraction.
- Application drift : In complex systems, if your code files and your database are not perfectly synchronized at backup time, the restore can result in an unstable or “broken” system.

The outcome of bad practice is brutal: you end up with unusable backups at the very moment you need them most. The only way to remove this uncertainty is to build in granular validation. It’s not about checking that “the file is there,” but checking that “the data is usable.”

The Danger of Rolling Back: Critical Data Loss

It is every IT manager’s nightmare scenario: after a major outage or a cyberattack, you realize that the last few days of backups are corrupted. You eventually find a “valid” backup—but it is several weeks old.

That “jump into the past” is an operational disaster:

- Data volatility : In just a few weeks, hundreds of customer orders, incident tickets, new documentation pages, or code changes can vanish. Recovering that data manually is often impossible—or prohibitively expensive.
- Loss of trust : For your customers and partners, discovering that you have “lost memory” of their most recent interactions is a fatal blow to your reputation. Trust is often far slower to rebuild than the server itself.
- The RPO imperative : This highlights the need to define an RPO (Recovery Point Objective): how much data can you afford to lose? If the answer is “a few hours,” then a strategy of

occasional snapshots is no longer sufficient.

Defining clear objectives for different scenarios makes it possible to precisely assess the cost of downtime. By targeting an RTO (Recovery Time Objective)—the time needed to return to normal—of only a few hours, you turn a potential disaster into a minor operational incident.

The Reality on the Ground: Incidents That Show No Mercy

If these risks seem theoretical, the reality of recent years is a harsh reminder that no one is safe, regardless of the organization's size or the provider's reputation. Examples of "blackouts" are multiplying:

- Physical risk (Fires) : The major fire at a data center in Strasbourg in 2021 was a reminder that a building can disappear. While the hosting provider survived—despite financial provisions amounting to millions of euros to cover disputes (still €12.6M provisioned in the 2025 budget)—hundreds of SMEs went out of business because they had not externalized their backups outside the affected site.
- Cyber risk (Ransomware) : The attack on the MDPH of Hauts-de-Seine (92), or the one affecting the Département de l'Essonne, showed that even public institutions can be paralyzed for months. In such cases, the problem is not only "recovering" the data, but ensuring that the backups were not encrypted at the same time as the rest of the network.
- Critical technical failure (Update bugs) : In 2024, the CrowdStrike incident brought millions of Windows systems to a halt worldwide (airports, banks, hospitals)—not because of a virus, but because of a simple defective update. This proves that even security tools can become the vector of a total outage if you cannot instantly roll back to a stable state.
- "Cloud" data loss : Many also remember the GitLab incident a few years ago, where human error during a database maintenance operation led to the permanent loss of production data. Only multi-media backups make it possible to recover from that.

What do these crises have in common? The organizations that made it through were those that did not rely on a single solution or a "bulk" snapshot. They had a multi-site strategy, segmented backups, and above all, a recovery plan tested before disaster struck. Security is a shared responsibility: the hosting provider supplies the tool, but it is up to the user to ensure the isolation and freshness of their own data.

Priority to Key Elements: A Sound Organization for Agile Backups

To avoid the "all-in-one" snapshot trap, your installation method must enable segmented backups. At OpenSource Solutions, we treat each component of your server according to its nature.

Think of your server like a house: if a fire breaks out, you save your paperwork and family photos first—not the concrete blocks you can buy again at the hardware store. Here is our hierarchy, in order of priority:

1. The Database (The Heart): It contains all the intelligence of your site: texts, users, forums, calendars, and preferences. It is the most volatile element, changing every second. Absolute priority.
2. “Content” Files (Memory): Your PDF documents, images, videos, and specific files. These are unique data you will not find anywhere else.
3. Custom Code (Your Signature): Your visual theme, UI templates, your specific PHP or JavaScript scripts. Our philosophy is to keep the application source code “clean” (as clean as possible) and to isolate your customizations to avoid modifying the core.
4. The Application Code: Tiki Wiki, WordPress, Dolibarr... These files are important, but they are reproducible identically from official repositories.
5. The Technical Environment: Your database tools (PHP, MySQL), your publishing software (Apache, NGINX), your security tools (Firewall, Fail2ban), and finally the server kernel/OS (Debian, Ubuntu, Red Hat).

Why is this distinction critical? By separating these elements, you gain complete freedom during restoration. If only your visual theme is “broken,” you can restore only the “Custom Code” section in seconds. If the physical server disappears, you reinstall a clean OS and inject only your vital data.

This method delivers maximum operability with a minimal volume of data to transfer.

Back Up What Truly Matters

At OpenSource Solutions, we enforce an intelligent strategy that consists of distinguishing and prioritizing what needs to be backed up. By isolating each component, we apply the most appropriate backup method to get as close as possible to the ideal solution.

You must distinguish between what can be retrieved elsewhere and what exists nowhere else but on your server. Here is how we organize these elements into groups:

1. The Critical Group : Unique data (Must be protected at all costs)
These elements are dynamic and represent the real value of your work. If they disappear, they are gone forever.

- Your database: This is the most critical element. It changes every second (new articles, sign-ups, transactions). It requires regular “dumps” and constant monitoring of its integrity.
- Your "content" files (Data): Your documents, images, and media. They are also dynamic and specific to your business. We handle them separately to manage their volume without slowing down the rest of the system.
- Your customized "code" files: Your themes, templates, and specific developments. They do not change daily, but they are the product of your investment. We recommend backing them up with every change—ideally via a Git repository for a complete history.

2. The Resources Group : Reproducible elements
These components are essential for operations, but they are publicly available. Rather than storing them heavily, we back up their configuration.

- Your application code: (Tiki Wiki, WordPress, Drupal, etc.) Available from official repositories.

- The technical environment: (PHP, MySQL, Apache/NGINX) Standard software available on the Internet.
- The operating system: (Debian, Ubuntu, Red Hat) Reinstallable at any time.
- Security and monitoring tools: Firewall, Fail2ban. Except for logs, these tools are standards. For monitoring and traffic analysis, we also favor external tools to avoid weighing down the server.

The conclusion is simple: this segmentation allows us to focus our efforts (and your budget) on backing up what is irreplaceable, while ensuring we can always rebuild the surrounding infrastructure in record time.

A Practical Example: The OpenSource Solutions Methodology for Tiki Wiki CMS

Although we detail our approach here for Tiki Wiki, this methodology is agile and can be adapted to any application or platform. By following our priority plan, we isolate the database, media, and custom code for maximum efficiency.

The Database (the heart of the content)

We have developed our own scripts that generate a timestamped database dump at least once per day.

- Security: This dump is encrypted and exported to remote storage (off-server) to ensure availability in the event of a total loss of the primary machine.
- Size: Once compressed, this file ranges from 200 MB to 5 GB depending on the scope of the project, which makes it extremely fast to transfer in an emergency.

The "Media" files (documents and media)

Images, PDFs, videos... these user-uploaded files can reach massive volumes.

- Externalization: In the preferences of the Tiki instances we install and maintain, we take care to externalize the storage of these files.
- Management: This allows us to handle and back up these volumes (often tens or hundreds of gigabytes) independently from the database, ensuring a smooth restart without burdening critical processes.

Custom Code (Your Identity, interfaces, and refinements)

To restore a site in a matter of minutes, we isolate everything that makes your site unique (CSS theme, templates, fonts, specific JavaScript or PHP scripts, logos) in a dedicated directory.

- Clean integration: These elements are included within the source code without ever modifying the application "core".
- Versioning: Everything is managed via a Git repository with a dedicated branch. During a restore, we deploy the application code and instantly merge your custom changes.

The Application Code

As mentioned in the previous paragraph, we favor using Git and mirroring.

- Mirror and updates: We create one project per site with "mirror" branches of the upstream application. This contains the standard code distributed by the publisher, complemented by your custom code.

This method enables regular updates, safe rollbacks, and the creation of test environments before production. Restoring a branch is nearly instantaneous.

Operating System and Libraries

For the rest (Linux OS, Apache, PHP, MySQL), we start from a clean installation. Why waste time backing up gigabytes of standard software available everywhere on the Internet? Thanks to this lightweight approach, we can restore a complete site—including the server—in 1.5 to 3 hours.

Validation: Moving from Theory to Certainty

Having backups is one thing; being certain they are valid is another. How can you ensure a database is not corrupted at the moment it is captured? The answer comes down to one word: experimentation.

To avoid remaining in uncertainty, every organization should establish a verification routine. This means pulling the files out of storage, decompressing them, and attempting to mount them on an isolated machine. If you wait until the day disaster strikes to test your procedure, it will be too late. Of course, this requires method: consistent test protocols per project, comparing file counts and sizes, and verification by user role.

Our validation methodology rests on three pillars:

- Content and integrity validation : We do not merely check the file size. We run consistency tests (checksums) to ensure no silent corruption has slipped into your tables during archiving.
- Software reinstallation validation : We test the ability to rebuild the application environment. The source code, your custom themes, and the server libraries must come together flawlessly, without dependency errors.
- The crash test (Disaster scenario) : We recreate a server from A to Z on a clean infrastructure using only your external backups. Only if we restore your services in under 3 hours under these conditions is the backup declared valid.

This rigor takes time. It is an essential security task, not an option to be handled “when we have time.”

Conclusion: Why Entrust Your Resilience to Specialists?

Cybersecurity and the longevity of your data must not be left to chance—or to unmonitored automation. Working with independent specialists and Open Source solution experts like OpenSource Solutions provides concrete guarantees:

- Real domain expertise : Unlike a general-purpose hosting provider, we understand your applications from the inside (Tiki Wiki, WordPress, etc.). We know exactly what is critical and how to rebuild it.
- A strict verification policy : We run these disaster simulations at least once per month. Our maintenance plans natively include these processes to give you complete peace of mind.
- Transparency and evidence : The results of our tests are recorded in a detailed monthly report, providing factual proof that your business is truly resilient.
- Independence and freedom : By working with partners like OpenSource Solutions Pro, you remain in control of your data. Our segmented approach allows you to migrate or restore your infrastructure anywhere, without being locked into a single provider.

To learn more about our "Enterprise Grade" maintenance plans and secure your future, visit our dedicated page: Enterprise-grade server maintenance