Webellian
DIGITAL AND TECHNOLOGY

WHITE PAPER

10 best practices to succeed at cloud

Author: Maciej Bulwan, Senior Cloud Architect at Webellian

Webellian
DIGITAL AND TECHNOLOGY

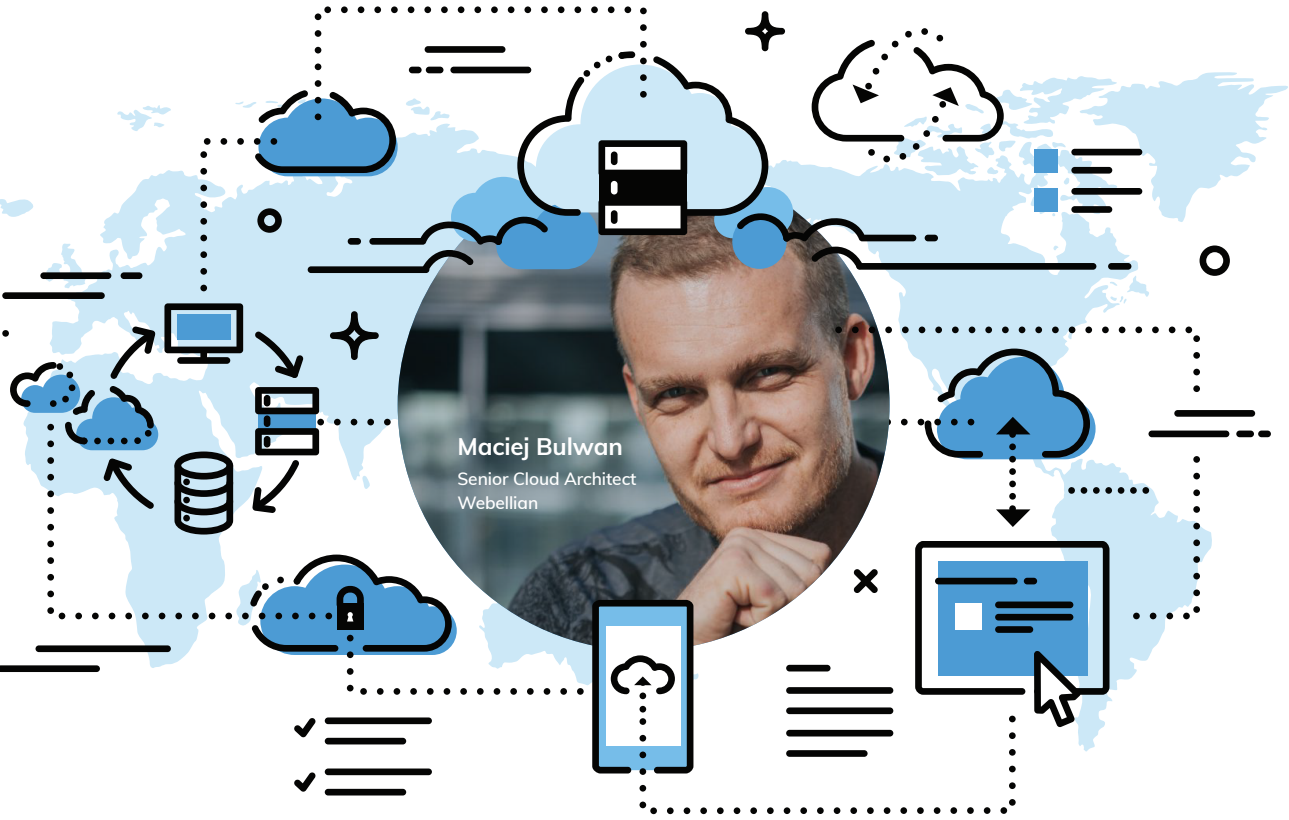
We anticipate, lead, and support your Digital Transformation.

Webellian is a privately owned French-Polish company founded in 2012 with headquarters in Warsaw, Poland.

Our international culture supports the Digital Transformation of big brands and global corporations, trusting us with their complex IT projects. Webellian is a fully integrated Digital Transformation and IT consulting firm helping its clients to face today's challenges and tomorrow's uncertainties by building long term partnerships and sustainable solutions. Our expertise ranges from bespoke software development to building Cloud solutions, from creating data solutions to building complex models.

With our passion for innovation, technology, and excellence, we empower you to move confidently forward on your Digital Transformation journey.

For more information go to webellian.com



Intro

Cloud is the foundation for the agile business world. It's the platform for enabling agile application development. Cloud-based infrastructure is key to delivering flexible, on-demand access to the resources underpinning new digital business offerings. It allows organisations to scale infrastructure as needed to support changing business priorities while reducing the risks of wasted IT resources that inhibit investments in new digital services.

When it comes to the cloud transition, for many enterprises, success or failure ties directly to the effectiveness of their IT service delivery environment.

Our goal here is to show you how you can successfully undertake your digital transformation and what to watch out for when planning, designing and implementing your transition to the Cloud.

Multi-cloud and vendor lock-in avoidance

There has been a lot of talk about multi-cloud. We can see two distinctive approaches for using multiple cloud providers:

- Cloud-native: taking advantage of proprietary offerings from different cloud providers and using the best-of-breed combinations for various business applications.
- Cloud-agnostic: delivering portable workloads that do not depend on proprietary features and can be deployed anywhere "easily".

The most often cited reason for using the "cloud-agnostic" strategy is to avoid being locked in with a single vendor. We often distrust cloud vendors regarding their future pricing, potential product discontinuation, a roadmap for new services, and data privacy implementations. While each of these arguments is valid, we may sometimes miscalculate the risk-benefit ratio. With a notable exception of GCP, other major vendors have so far demonstrated great discipline for not discontinuing offerings and keeping costs stable (if not lowering them over time).

When If you keep multiple clouds compatible, you will most likely:

- **Pay more.** When you run 33% of your workload on Azure, the other 33% on AWS, and the rest on Google Cloud Compute, you are effectively fragmenting you spend and automatically diminishing your bargaining position to secure discounts from Enterprise Agreements. All vendors encourage customers to use more by offering either tiered pricing or volume pricing that is characterized by unit price going down when the quantity of resources used increases. Savings are achieved by spend consolidation, not fragmentation.
- **Miss out on cutting-edge features.** Some vendors have outstanding technology, but it's proprietary to them and them only. Google might have the best big data warehouse (BigQuery), Microsoft might have the best services for Office tools, and AWS - the best ML services, but by being vendor-neutral, you miss out on those. Staying cloud-agnostic forces you to discard plenty of revolutionary and proprietary solutions that each vendor has to offer.
- **Reduce business agility.** By trying to be everywhere, you waste time and effort to make your workload portable in between public clouds. Except for dockerized applications, configurations of all other services (load balancing, network routing, monitoring, firewalls, identity services, H/A behaviour) are dominantly incompatible. It takes significant engineering time and effort to develop CI/CD pipelines, ensure the same performance, configure equivalent security, and effective disaster recovery routines to guarantee unified workload deployment and runtime characteristics across multiple vendors.
- **Experience skills shortage.** Finding and keeping the right people is a non-trivial challenge. You may be unlikely to find experts deeply knowledgeable in solutions from all major cloud providers. Talent is often in short supply, and attracting/retaining such a broad expertise pool will cost you.

Best Practice 1. Do not avoid vendor lock-in at all costs. In some cases betting on one single Cloud implementation can significantly benefit you. We encourage you to dive deep into one only cloud provider. You can get a much better deal and a vast improvement in agility if you pick the best features of one cloud vendor that are most suitable your business case. If you're an innovative you'd better focus on delivering what your business need.

There are more migration strategies than the “Lift and Shift”

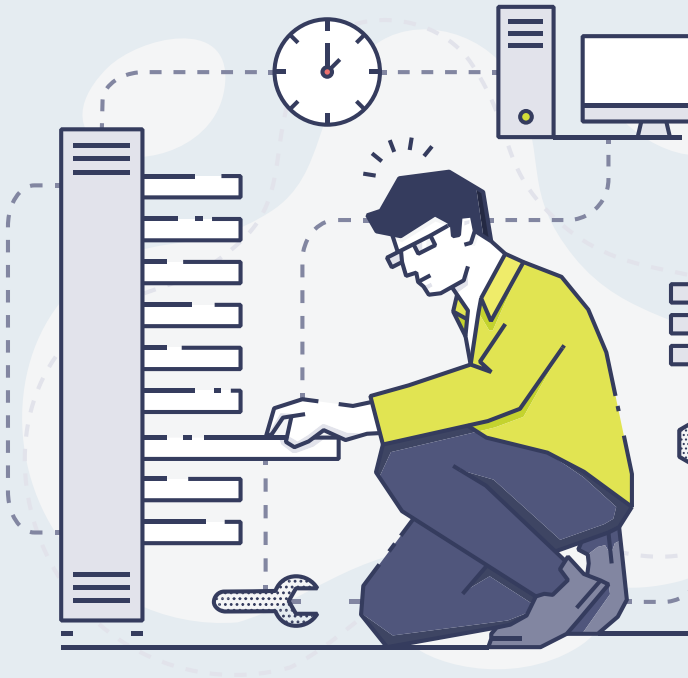
There are at least six different strategies when moving to the Cloud: Re-host, Re-purchase, Re-platform, Re-architect, Retain, Retire. The most popular one, Re-host, is also known as the “Lift and Shift.”

When you decide to migrate to the cloud, it may seem reasonable to vacate your on-premise data centres and just rehost all IT inventory in the cloud. You can see a vast selection of c ompute, storage, network, security management building blocks in all public clouds, so you start planning to “re-host” everything. This approach, however, doesn’t always work that well:

- After applications are relocated to different environments, they start demonstrating issues you haven’t observed before. Changed IP address ranges, virtual machine sizing, disk controllers, network shared file system incompatibilities make “Lift and Shift” turns silently into “move-and-improve,” with the “improve” part becoming significantly time-consuming.
- You rarely save on the amount of resourced consumed only by moving to the cloud. Savings come from elasticity, which is the ability of the cloud environment to allocate and dispose of resources dynamically depending on the current workload. Most “on-premise” applications do not handle elasticity.
- By simply rehosting, you gain very little agility from migration because you aren’t changing IT operations. The way you maintain servers, deploy applications, monitor them, and respond to incidents doesn’t improve.

It’s better to identify what matters most to you and adjust migration strategy accordingly:

- Infrastructure cost savings,
- Improvement in staff productivity,
- Operational resilience,
- Business agility.



Best Practice 2. Decide on what matters to you and assess other migration strategies that better support your goal. “Lift and Shift” seems easy, but it often isn’t. It can create many other problems that you wouldn’t expect and it doesn't necessarily solved your exsiting problems.

Let's talk about pricing

One of the ways cloud providers attract customers is the elasticity of resource provisioning, which is the ability to discard resources when they are not needed and instantiate them when required to serve increased workload.

The most obvious pricing model that aligns nicely with elasticity is "pay-per-hour", also known as "Pay-As-You-Go". This pricing model is typically the first option we see and consider. But the most straightforward pricing model is rarely the most cost-efficient. You might be overpaying by 20-40% compared to pricing based on usage commitment.

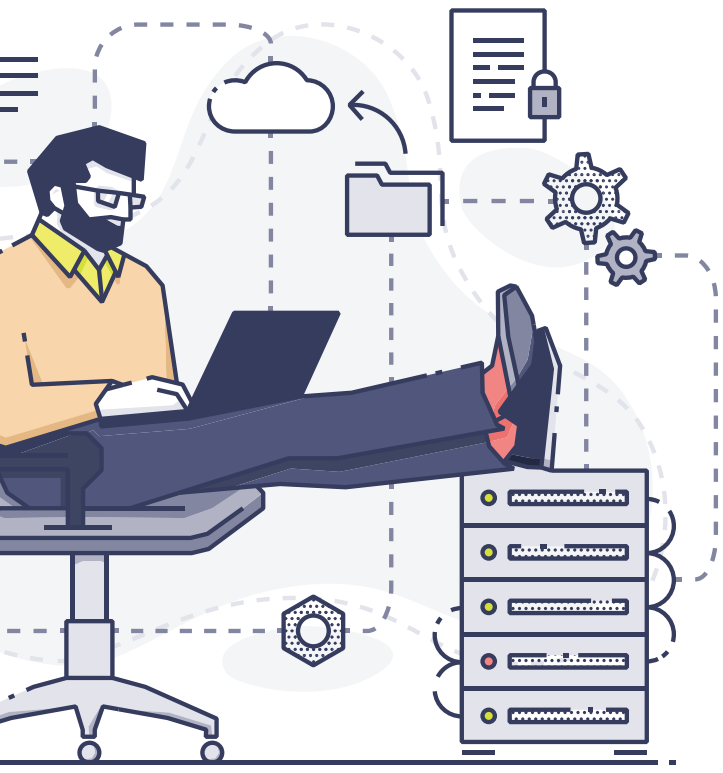
To understand why we need to think about how cloud vendors plan for investment into infrastructure powering their offering. They need to have slightly more capacity than all of us may need, but not that much for it to stay idle. They benefit significantly from the predictability of demand, and they are willing to reward us with discounts when we help them anticipate our future needs. They also reward us with pricing discounts for our dedication to utilising the requested infrastructure for a prolonged period. Duration of our commitment is the most critical factor that impacts discounts available to us (1yr, 3yr).

DISCOUNT COMPARISON STRATEGY (1 VM MATRIX)

		Pay as you go	1y No upfront	1y Partial upfront	1y All upfront
AWS C5.xlarge (4vCPU/ 8GB)	Upfront	--	$0	$510	$999
	Effective hourly	$0.194	$0.122	$0.166	$0.114
	Annual Cost	$1,700	$1,070	$1,450	$1,000
	Discount		**37%**	**40%**	**41%**
		Pay as you go	**1y No upfront**	**1y Partial upfront**	**1y All upfront**
Azure D3 (4vCPU / 14G RAM)	Upfront	--	--	--	--
	Effective hourly	$0.3080	--	--	$0.2098
	Annual Cost	$2,698	--	--	$1,838
	Discount				**32%**
		Pay as you go	**Sustained Use (50% time)**	**Sustained Use (100%)**	**1Y Committed usage**
Google Cloud n1-standard4 (4vCPU / 15G RAM)	Upfront	--	--	--	--
	Effective hourly	$0.2448	$0.2200	$0.1710	$0.1541
	Annual Cost	$2,145	$2,145	$1,498	$1,350
	Discount		**10%**	**30%**	**37%**

Reserving instances:

- With AWS, you have three options: no-up-front, full-upfront, and partial-upfront payment. The more of your commitment you're willing to pay up-front, the better effective discount on the hourly charge.
- With Azure, there is no option for partial up-front. It's full up-front in one go or split into monthly instalments, but no clear savings between monthly and annual up-front. Interestingly, Microsoft will allow you to renege on your commitment at a fee (12%) and under some conditions.
- As an alternative to the 1y / 3y "Committed Usage" contract option, Google Cloud Compute offers discounts for "Sustained Use". They measure the percentage of time your machine is up and running every month. The higher the measurement metric, the better the 'per-hour' rate for which you are charged. It discourages erratic provisioning of compute resources and incentivises planned and sustained consumption.



Hourly prices go down when you commit to using it (AWS, Azure, GCP) or actually use it (GCP).

Remember the fundamentals: When you declare your future utilisation, you're helping cloud providers planning their investment into data centres. In return, you get rewarded with better effective hourly-rates.

When you dive into pricing options deeper, you will find even more exciting pricing offerings:

- **"Spot instances"** - you're buying a reservation at a discounted price but agree it may be decommissioned suddenly.
- **"Convertible"** reservations: you can upgrade the specifications of the virtual machine.
- **"Savings plans"** - you can flexibly change VM specs (region, operating system, family, size) as long as you commit to spending a predefined amount of dollars per hour.

Best Practice 3. The simplest pricing model will most likely cost you more: you may overpay by 30% over the price you would pay otherwise. The more commitment you can make, the higher discounts and better pricing you can get. Take your time, analyse different pricing models, and pick the one that suits your needs better. Do not use 'pay-as-you-go' when you already have a predictably consistent workload.

Following new trends

New platforms or a new generation of software tools are exciting when announced. They are often advertised as remedies to many challenges we are facing. It makes us think they are likely to make our problems go away, and we adopt them as soon as possible. Our experience tells us that, by following new trends blindly, you can often put yourself in a situation where you risk:

- **Wasting valuable time.** Can you recall how complicated and expensive it was to set up a Hadoop ecosystem 12 years ago for the first few years? We can remember administrators studying for months to obtain certifications in Hadoop installations. Eventually, its distributions became consistent and stable, use-cases became more evident, anti-patterns emerged over time, and more business intelligence and ETL tools added support. Still, pioneers had plenty of frustrating experiences at first.
- **Gain little.** When containers started being popular, they quickly became synonymous with microservice architectures. We have seen many monolithic applications repackaged as multiple equally large containers and called microservices. True, it was easier to ship them between environments, but many underlying problems (dependencies, reusability, session state management) haven't disappeared.
- **Observing problems you haven't seen before.** Early releases of serverless functions, known as lambda, brought plenty of surprising and painful problems. Function executions weren't always as fast as functions required warm-up and suffered from cold start problems. It was more difficult to collect meaningful application logs and trace problems until logging and tracing requests across distributed executions became easier. Sudden and large spikes in traffic used to cause parallel lambda executions to spawn uncontrolled and consume all available IP addresses. As a result, subnets in which serverless functions executed could malfunction from starvation of the IP address pool. While lambdas scaled nicely, your servers couldn't because there were no spare IP addresses to allocate that were needed to scale out! Most of the problems got eventually solved (throttling via concurrency provisioning), but it took time for technology to mature, and early adopters had their share of surprises.
- **Cost.** Quite often, early releases of new frameworks, services, platforms miss out features that help optimize their capacity (metric collection, throttling, tooling for quick provisioning/deprovisioning, observability in the context of distributed system architecture) which makes early deployments of them not cost-optimal. It may be difficult to scale them up/down accordingly, and - to stay on the safe side - companies are forced to overprovision (=overpay).

Let's have a look at an example:

We need to serve web application report pages to eight concurrent users in parallel. Every request takes one second and requires 1 GB of RAM to produce a resulting report page. At eight requests per second, serverless execution will need to make 21 million calls a month (28k per hour, 700k a day), and your monthly bills for lambda usage will stand at $352.

If - instead - you decide to instantiate a virtual machine with eight virtual cores and 16GB RAM (c5.2xlarge) - roughly the capacity sufficient to serve the same content - you will pay $280 a month. Running a server is 20% cheaper than serverless in this case!

Furthermore, you can pre-pay this server up-front for one-year - that will further reduce the cost down to $176, which in result offers you a 50% reduction off the original 'serverless' price.

MONTHLY COST

Serverless $352

Serverfull $280

Best Practice 4. New cloud technologies are often exciting. Just be a little bit patient and wait for them to mature. Experiment with features and build POC before making critical architectural decisions. Usually, proper re-design of software architecture is required to fit existing applications into new paradigms, frameworks or platforms; simplistic 'refactoring' does not yield sufficient advantage from using new and 'revolutionary' technology. The good thing with the cloud is that you can relatively quickly build prototypes and run tests. Take advantage of that and try before you buy! Last but not least, calculate the cost to benefit ratio.

Region selection

It's essential to think more carefully and plan the exact location for your data and servers. Why?

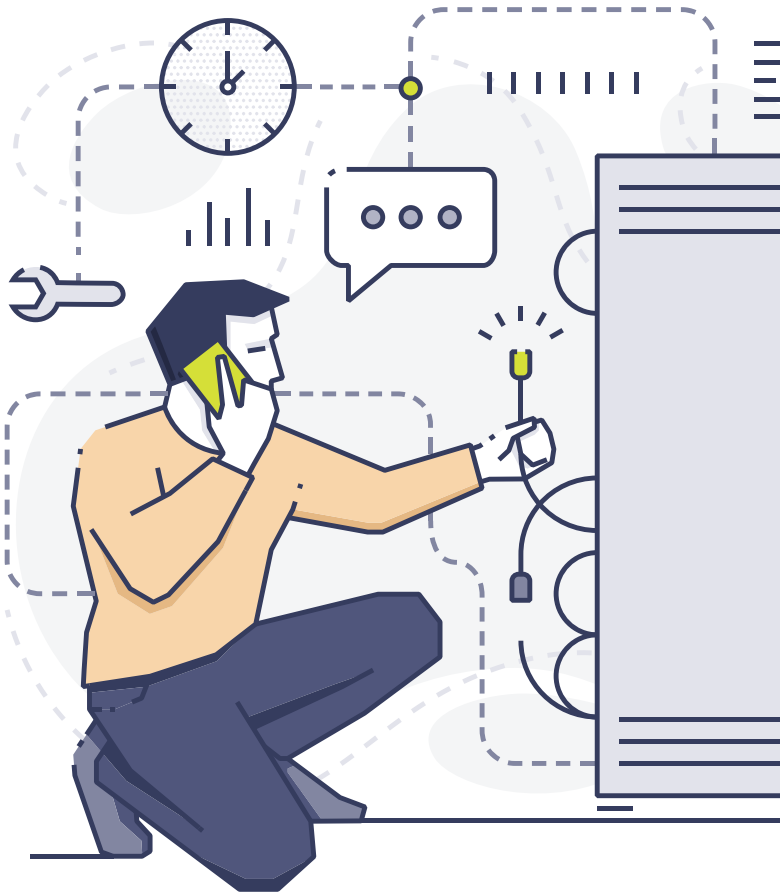
- **Cost.** Charges for using services, infrastructure, and data storage/transfers vary per region. If you place data in the wrong location, you pay more. Moving data in and out of a cloud geographical location costs much more than moving data within the same region's boundaries. Infrastructure costs also vary geographically.
- **Latencies.** You may experience latencies. Data needs to travel from one country to another, and the geographic distance results in additional journey time. These extra latencies always translate to lower application performance because applications need to wait for data to arrive.
- **Compliance.** Some regulations require your data to be stored in one jurisdiction, e.g., data should not leave the European Union. If you inadvertently place data in an incorrect region, you might be in breach of local regulations.
- **Cutting-edge features.** The availability of features is different across different regions. Some regions are lagging behind a lot when it comes to the availability of new features. You may want to store application data in the cloud location where the largest selection of most recent new features is available.
- **Disaster-recovery.** You need to be careful where data is placed so that when a region becomes unavailable, it doesn't cascade down to applications that would otherwise be fine.

Best Practice 5. Consider where to place your data carefully, taking into account all requirements and data related regulations in your industry.

Take a good look at your network infrastructure

You are about to move to the Cloud. You have already selected a geographical region to host your infrastructure. Now, you need to connect to the Cloud either from your on-premise infrastructure or from a private data centre or your laptops.

- You can use the Internet connection. However, it's not natively secure, and it needs at least VPN tunnelling to ensure data privacy and confidentiality. There are several issues with VPNs. Corporate VPNs are often seen as potential backdoors because they open access to large segments of networks, and the philosophy that "once you're in, you're trusted" is seen as too lax. Creating and maintaining multiple one-to-one VPN connections is complicated; the performance penalty is not negligible either.
- An alternative is to create new or extend existing physical links. That requires contacting an ISP provider and ordering dedicated connections directly from your office to cloud edge locations or cloud provider partners' data centres. It becomes an expensive alternative to the Internet. Moreover, you still have a dependency on a single ISP, and it may be unavailable to you unless your on-premise data centre is located conveniently in a large city.



Best Practice 6. Consider using a new generation of networking solutions. Explore possibilities software-defined networking offers to you. AppWANs create encrypted and secure communication tunnels between service endpoints and clients running in the cloud and on-premise. Implementation of Zero Trust Network Access and Micro Segmentations architectural principles address strict security requirements for hybrid- and multi-cloud deployments. You can also run AppWANs on top of the Internet as they improve its performance (better route selection, protocol enhancements, use of cloud operators' backbone networks) and replace the need for flawed VPN.

Your Cloud provider will only partially secure your need for backups

Cloud providers take backups of services and infrastructure in the Cloud. It is only partially true, and you need to be aware of several caveats:

- They only create backups of selected types of services (database, cache, elasticsearch, block devices).
- Backups aren't always enabled by default. You may need to explicitly specify the option for creating automatic backups, either during the initial resource creation or subsequent configuration. Often automated backup configuration will be as easy as specifying retention, encryption options, and time and duration of backup windows, but you need to do it yourself nonetheless.
- Frequency and retention of automated backups aren't always consistent between different services. For example, AWS Elasticsearch automated backups are taken hourly and kept for 14 days. RDS database automatic snapshots are taken once a day and stored for one or seven days, depending on how you created the database instance. You may need to adjust auto-backup parameters to suit your data recovery policies. However, to guarantee data consistency across many services, you will most likely require a well-thought DR plan with custom backups implemented.
- Backups are often taken to allow recovery of an entire cluster or instance. You may not be able to restore a selected index, selected file, or individual table.
- Automic backups or snapshots may not be available to you in a different cloud region unless you implement cross-region shipping/replication by yourself. When cloud outages happen, and they do, your DR strategy may require that you re-launch the system in another location from recent backups (RPO = Recovery Point Objective) within a specified time (RTO = Recovery Time Objective). With automated and unreplicated snapshots you may never be able to achieve your objectives for recovery.

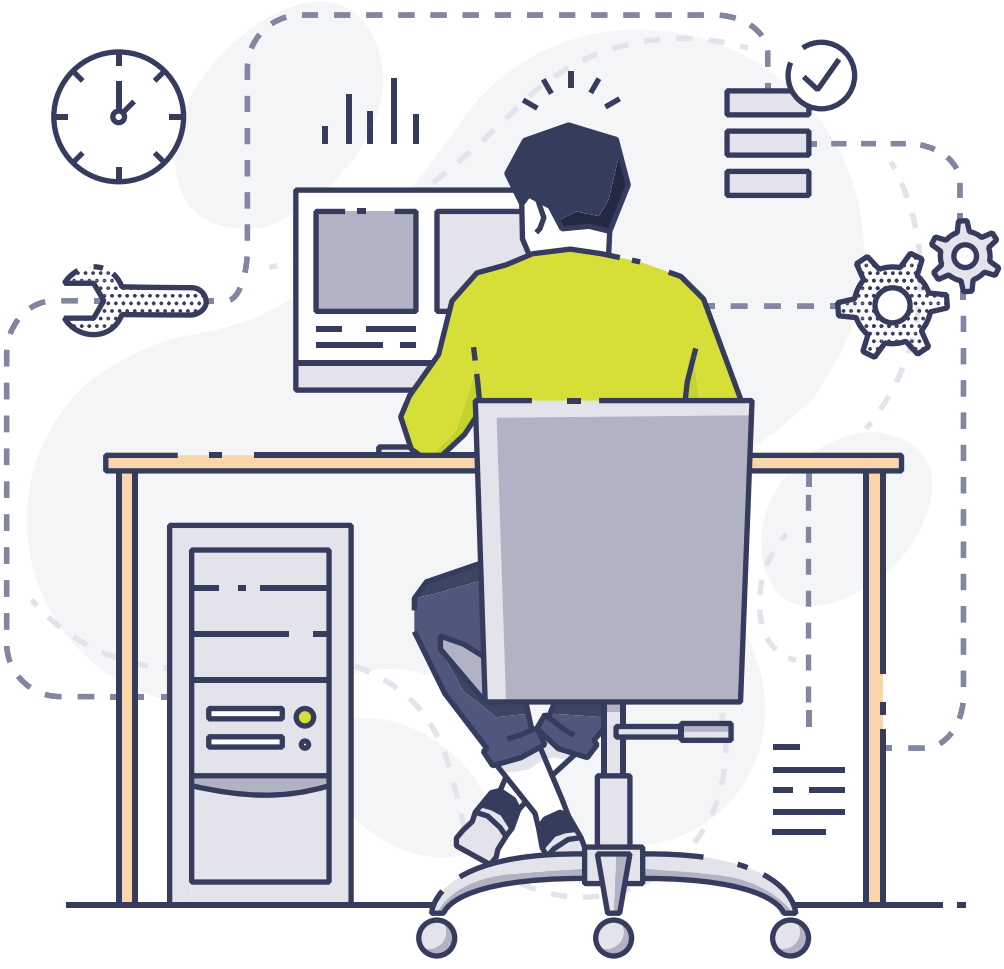
Best Practice 7. Do not rely only on cloud providers to secure your backups. Data losses and outages happen. Cloud providers cannot protect you from it completely. You should have a disaster recovery plan, implement backups according to that plan, and - most importantly - have a working and realistic disaster recovery strategy. You should also test your recovery procedures regularly. We can see that recovery plans become stale very soon, and they are not updated and tested often enough to stay up-to-date with production systems. When designing and testing a recovery plan, you do not have to obsess about the 'asteroid' hitting us-east coast nightmare excessively. Instead, a more realistic scenario, like database corruption, a malicious virus activity, or human errors are disastrous enough to trigger disaster recovery processes that you want to have tested.

Who's responsible for security?

It's true that Cloud Providers are responsible for the security of the cloud. That includes the physical security of data centres and infrastructure, but if you leave it like this, you will be left with:

- Unencrypted network traffic,
- Unencrypted data,
- Too weak password policy,
- Too permissive access levels,
- Operating systems vulnerable to certain types of attacks

Best Practice 8. Security in the cloud is your responsibility! You decide how you implement your system's security: what encryption protocols are used, what access policies look like, how complex and long-lived passwords are, which network ports and IP addresses are allowed to traverse firewalls, how often you rotate access or encryption keys, what security patches you install on servers you manage, and so forth.



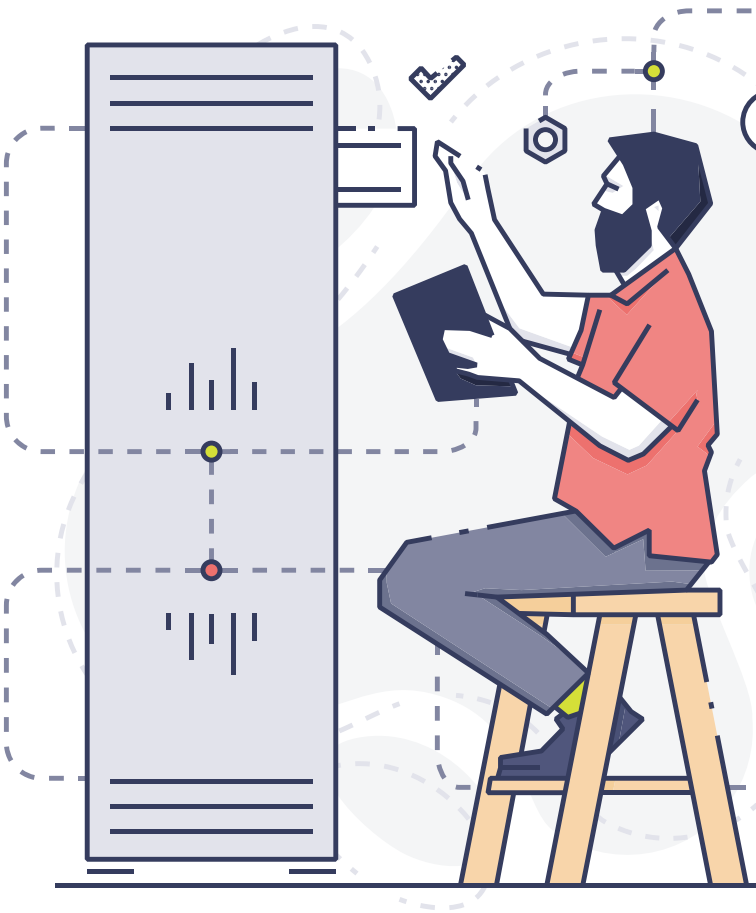
Account structure

We all start small, with a single cloud account, and start growing from there. We add users. Over time, these users are assigned more and more permissions. We add more networks and subnets to separate the growing number of workloads. We deploy both customer-facing applications and back-office applications. Software Engineers responsible primarily for the maintenance of back-office applications gain access to parts of the system where customer-facing systems reside. We build different categories of environments: development, testing, integration, production. When we set up the environment this way we eventually end up with a messy environments exposed to potential:

- Data leaks,
- Data integrity loss,
- Exceeded account limits,
- Running out of the capacity of shared resources,
- Blast radius propagation

The complexity of your cloud environment will grow over time, hand in hand with your business' growth. Hosting all services in a single account will eventually lead to an operational mess where the management of cost, security, and performance will become difficult.

Best Practice 9. Explore the idea of separating workloads into several accounts. You may want to keep testing environments separate from production environments. You can further separate them by a distinctive business function; you may also maintain different accounts to keep Identity Management and access policies. Source code, build, and delivery pipelines could all also reside in an account separate from other applications. It's often easier to design a multi-account strategy early in the process of moving to the cloud.



Automation of IT Processes

When migrating to the cloud, we often try to re-use as many existing, non-cloud processes and tools as we can. They have worked so far on-premise and are well battle-tested. We cannot afford the time to make too many changes at once. After all, we have the right people: trained, smart, knowledgeable, and dedicated. They know and fluently use all the tools at their disposal and have executed processes many times enough to be efficient. Our IT operations teams use the very same monitoring, alerting, system intrusion detection applications they used before. They run the same scripts to archive logs; the same daily job to check for the number of database connection failures; they use the same application to check free disk space on the database server disks. Is it good enough?

One of the most significant advantages of cloud computing is the vast selection of tools for automation, resource provisioning, anomaly detection, and error handling. Automation can enhance processes; IT personnel will spend less time carrying out repetitive tasks. Continuous Test and Deployment pipelines will allow developers to focus on building features that matter to your customers. With automation, you can reduce operational cost originating from investigating and fixing of manual errors.

Best Practice 10. Aim to automate IT operation processes from the start. Do not let automation be merely an afterthought and keep adding to your technical debt.

Configure rules preventing changes to resources that would lead to suboptimal configurations of security and compliance standards (e.g., creating publicly readable files storing unencrypted backups, web services with unsafe endpoint URLs).

Implement self-repair whenever possible; when free disk space on a virtual machine drops below a specified threshold, a new disk can be added automatically.

When looking for unusual spikes in traffic, enable machine-learning-based, adaptive, anomaly detection algorithms that can adjust observations by seasonality.

Keep build automation around software deployments; pipelines can easily use blue/green or canary types of delivery that ensure the much-improved quality of software deployments and eliminate configuration drift.

Take advantage of automated scaling; system administrators no longer need to adjust capacity periodically.

One page summary

Best Practice 1. Do not avoid vendor lock-in at all costs. In some cases betting on one single Cloud implementation can significantly benefit you. We encourage you to dive deep into one only cloud provider. You can get a much better deal and a vast improvement in agility if you pick the best features of one cloud vendor that are most suitable your business case. If you're an innovative you'd better focus on delivering what your business need.

Best Practice 2. Decide on what matters to you and assess other migration strategies that better support your goal. "Lift and Shift" seems easy, but it often isn't. It can create many other problems that you wouldn't expect and it doesn't necessarily solved your exsiting problems.

Best Practice 3. The simplest pricing model will most likely cost you more: you may overpay by 30% over the price you would pay otherwise. The more commitment you can make, the higher discounts and better pricing you can get. Take your time, analyse different pricing models, and pick the one that suits your needs better. Do not use 'pay-as-you-go' when you already have a predictably consistent workload.

Best Practice 4. New cloud technologies are often exciting. Just be a little bit patient and wait for them to mature. Experiment with features and build POC before making critical architectural decisions. Usually, proper re-design of software architecture is required to fit existing applications into new paradigms, frameworks or platforms; simplistic 'refactoring' does not yield sufficient advantage from using new and 'revolutionary' technology. The good thing with the cloud is that you can relatively quickly build prototypes and run tests. Take advantage of that and try before you buy! Last but not least, calculate the cost to benefit ratio.

Best Practice 5. Consider where to place your data carefully, taking into account all requirements and data related regulations in your industry.

Best Practice 6. Consider using a new generation of networking solutions. Explore possibilities software-defined networking offers to you. AppWANs create encrypted and secure communication tunnels between service endpoints and clients running in the cloud and on-premise. Implementation of Zero Trust Network Access and Micro Segmentations architectural principles address strict security requirements for hybrid- and multi-cloud deployments. You can also run AppWANs on top of the Internet as they improve its performance (better route selection, protocol enhancements, use of cloud operators' backbone networks) and replace the need for flawed VPN.

Best Practice 7. Do not rely only on cloud providers to secure your backups. Data losses and outages happen. Cloud providers cannot protect you from it completely. You should have a disaster recovery plan, implement backups according to that plan, and - most importantly - have a working and realistic disaster recovery strategy. You should also test your recovery procedures regularly. We can see that recovery plans become stale very soon, and they are not updated and tested often enough to stay up-to-date with production systems. When designing and testing a recovery plan, you do not have to obsess about the 'asteroid' hitting us-east coast nightmare excessively. Instead, a more realistic scenario, like database corruption, a malicious virus activity, or human errors are disastrous enough to trigger disaster recovery processes that you want to have tested.

Best Practice 8. Security in the cloud is your responsibility! You decide how you implement your system's security: what encryption protocols are used, what access policies look like, how complex and long-lived passwords are, which network ports and IP addresses are allowed to traverse firewalls, how often you rotate access or encryption keys, what security patches you install on servers you manage, and so forth.

Best Practice 9. Explore the idea of separating workloads into several accounts. You may want to keep testing environments separate from production environments. You can further separate them by a distinctive business function; you may also maintain different accounts to keep Identity Management and access policies. Source code, build, and delivery pipelines could all also reside in an account separate from other applications. It's often easier to design a multi-account strategy early in the process of moving to the cloud.

Best Practice 10. Aim to automate IT operation processes from the start. Do not let automation be merely an afterthought and keep adding to your technical debt. Configure rules preventing changes to resources that would lead to suboptimal configurations of security and compliance standards (e.g., creating publicly readable files storing unencrypted backups, web services with unsafe endpoint URLs). Implement self-repair whenever possible; when free disk space on a virtual machine drops below a specified threshold, a new disk can be added automatically. When looking for unusual spikes in traffic, enable machine-learning-based, adaptive, anomaly detection algorithms that can adjust observations by seasonality. Keep build automation around software deployments; pipelines can easily use blue/green or canary types of delivery that ensure the much-improved quality of software deployments and eliminate configuration drift. Take advantage of automated scaling; system administrators no longer need to adjust capacity periodically.

Webellian
DIGITAL AND TECHNOLOGY

Webellian Sp. z o.o.
ul. Domaniewska 45
02-672 Warsaw, Poland
+48 22 213 12 16

webellian.com