Whitepaper

Migrating to AWS at Scale

Index

Introduction

Enterprises are in the midst of “the great migration” to the cloud. They see the rewards that come from scalability, cost-efficiency, and flexibility to meet the fluctuating demands of the market. However, as with any journey to a new destination, it is fraught with challenges. Many risks associated with moving to the cloud can have a big impact on an organization’s ability to execute its cloud migration strategy and to reap the expected benefits of it.

This whitepaper aims to address how an enterprise can execute a cloud migration **at scale**. There is no lack of recipes for cloud migrations[1], so these series perform a deep dive on the challenges, considerations and solutions for migrating **at scale.** The target audience is anyone looking to be inspired in their thought and design process to accelerate their migration journey.

1 See:

https://aws.amazon.com/migration-acceleration-program/

https://aws.amazon.com/professional-services/CAF/

https://oblcc.com/services/enablement-migration/

Chapter 1: High Level Overview of the Migration Journey

Let us first define the term “at scale” in the context of a cloud migration. We define it as “a required size to solve a problem at a certain velocity”. The size and velocity are relative to the project characteristics (timeline, application inventory, resources, skills). For these chapters, the objective is to gain maximum velocity within the given timeline, application inventory, resources, and skills.

Based on experience we describe the typical cloud journey containing the following milestones:

The journey does not have one starting or ending point, and each milestone offers a certain service to the clients. Let’s go through some of the important milestones for migration at scale:

1.1 Cloud Business Case

When making a cloud business case the goal is to clearly describe the value-add for the client by migrating to the cloud. The business drivers are identified and unfolded, and the key stakeholders commit to a common (high level) understanding of the cloud journey. A clear picture of the business drivers and commitment from key stakeholders is critical for the cloud migration to execute at scale as it simplifies and accelerates the decision-making process.

An example from a previous client is a hard deadline exit from existing data centers. As the deadline could not be postponed, it was identified as the primary business driver. All decisions taken during the migration process were taken to achieve the fastest pace possible and recognizing that certain subjects may have to be revisited.

In other cases, the expected benefits that should be incorporated into the business case come from increased resiliency, faster time to market for new functionality and a lower Total Cost of Ownership (TCO). With the exception of TCO, many companies struggle with identifying the financial value of a higher resiliency or fewer outage, higher agility, etc. There are ways to estimate (in cooperation with the business) the financial value of these benefits. Even if the effort to assign a financial value to these may seem futile, it is important to list these benefits (and the conditions under which these are expected to materialize) into the business case.

1.2 Cloud Operating Model

The Cloud Operating Model[2] (COM) describes a target for how the enterprise will operate in the Cloud. It answers the basic question of **who** is responsible for **what**. As the Cloud Operating Model often differs from the On-premises Operating Model, it is key for all involved parties to understand what their new role entails in terms of responsibilities. This promotes commitment to the cloud migration process, as everyone knows how to contribute, and who to approach for problem solving.

Establishing the right operating model during and after migration is critical, as sticking with the previous "data center-centric" model, will at best diminish the expected benefits of the cloud strategy, at worst, lead to a lack of control over the new cloud environment.

While designing the new cloud operating model, it is essential to match this against the expected benefits as expressed in the business case and to verify if the new operating model enables the benefits being realized.

Identifying and implementing the target COM can be a time-consuming process, and time is rarely on the client's side with regards to migration. Rarely will existing operation, platform and application teams accept a double responsibility of both on-premises and cloud, and this poses a challenge for the migration velocity.

Enterprises usually find themselves in creating plans for hiring more employees and supplementing with consultants. In this case one (or more) transitional COMs can be identified which cover the core migration team and key individuals in the organization needed for migration. To avoid overwhelming the participants, it is important to identify the success criteria early on for the transitional COM, so the enterprise knows when to move forward towards the final COM and leave the previous transitional COM.

Another important consideration for a COM and migrating at scale, is the degree of managed services. Identifying the transitional and target COMs is an optimal time to consider if some cloud services should be handled by more skilled staff than the Enterprise itself. AWS will provide support[3] and basic managed services, for example AWS RDS (managed relational database service). Clients can find great value in re-platforming on-premises services to AWS managed services, e.g. on-premises databases to AWS RDS, and thereby delegate the responsibility of (some) database operations & support to AWS and achieving the business agility by the elasticity of AWS RDS service.

But clients should look further than that. Xebia, and other AWS partners, offer strong managed services in the field of Cloud Financial Management (FinOps), Development & Operations (DevOps) and Security Operations (SecOps). This can significantly accelerate the cloud migration and the cloud adoption, as large chunks of work and responsibilities are lifted from the existing organization. This can minimize/eliminate the need for hiring more employees as existing staff can hand over rudimentary responsibilities and focus on Business development and innovation through Cloud.

An example from a previous client who has 6000 employees, decided to have two transitional COMs and a final COM. Each COM introduced a larger radius of cloud roles and responsibilities into the organization and extended the reach of the CCoE team. The first COM was to accelerate the launch of the migration. The second COM was to ensure stable operations of the migration. The final COM was to accelerate cloud adoption and modernization. Managed services were utilized from the beginning, both from AWS and AWS partners to accelerate the COM outcome with the existing organization.

2 See
https://docs.aws.amazon.com/whitepapers/latest/building-cloud-operating-model/building-cloud-operating-model.html
https://docs.aws.amazon.com/wellarchitected/latest/operational-excellence-pillar/operating-model-2-by-2-representations.html

3 https://aws.amazon.com/premiumsupport/plans/

1.3 Cloud Center of Excellence

The Cloud Center of Excellence (CCoE) is a core team, which is vital for cloud migration and adoption. This team is in the forefront leading and enabling the organization to use the cloud. Its staffing is fluid as business and IT is represented. Its role is dynamic with a focus starting with cloud migration shifting to cloud adoption. Establishing a CCoE, giving them decision making rights, and staffing them with migration skills is key for migrating at scale.

This raises an interesting question: Is the CCoE the same as the migration team? It can be, but it must shift its focus towards cloud adoption and modernization during the later stages of migration, which may require restaffing or onboarding new skills & roles.

1.4 Cloud Foundation – Designing the Landing Zone

Designing the first landing zone for the migration is an overwhelming experience for a new cloud customer. There are a gazillion things to learn, consider and decide. Areas such as, but not limited to, mapping the organization to cloud,environments, applications, geography, security, access management, connectivity, compliance, user experience and much more.

Clients who do not have the luxury of time for a steep learning curve must ally themselves with a strong AWS partner, who can hold their hand and advise on all the options at each decision point. Xebia offers a strong guided process where the client takes the key decisions based on options presented in context to the client's business domain and organization. This contextual-specific communication is key to enabling the client to take fast decisions and prepare for migration at scale.

Wrong decisions have a penalty in the form of time needed for revisiting subjects and reconsidering previously made decisions. Even though the design process of the first landing zone is iterative in nature, the wish is to minimize the length of each iteration and the number of total iterations to produce the first approved landing zone design needed for migration.

An example from a previous client was half day workshops over the span of two weeks, where each subject was covered in a pre/post-decision style manner. The pre-decision workshops identified the requirements, presented the options with recommendations, and decisions from the client. The in-between workshops digestion allowed the client to verify the decisions internally and the post-decision workshops were used to fine tune the decisions and document the design. As the design was built in an incremental manner with the client, the client was always up to date and aligned on AWS knowledge, decision impact, and the final landing zone design.

This style of a guided process executed in an accelerated fashion requires an upfront allocation of knowledge-holders and decision-makers within the organization, and these individuals must be part of the core migration team, allocated completely to the complete project lifecycle. Any additional ad hoc allocation must be fast and precise to avoid slowing down the process.

An example from another previous client who considered the migration project to be a part of operations only, had staffed the landing zone team with operational staff. With guidance from Xebia, they quickly realized that platform, application, economy, security, and business team members also had to be present to cover the width and depth of designing the first landing zone. By allocating key members from the various teams, they enabled themselves to allow decision verification within each team, during the workshop digestion period. The result was a landing zone precise and flexible enough to cover not only the migration but the future adoption of the client's cloud journey. This essentially saved the client a considerable amount of time during the migration and modernization phase.

Chapter 2: Preparing for Migration

Continuing from Chapter 1: Migration Overview, we reach a stage where we have a good understanding of the people (roles and responsibilities) involved in the migration, and they are fully allocated. And we have a good understanding of the destination landing zone in AWS, and the degree of managed services to utilize. In this chapter we explain how to prepare for a migration.

2.1 Application inventory

In order to understand **what** is to be migrated, an Application Inventory must be established. This consist of a tree mapping the:

- Applications
- The servers used by each application and their configuration:
  - CPU, memory, disks, network, etc
  - Operating system
- The dependencies for each application:
  - Databases
  - Network connections
  - Other shared services
- The application owner(s) who either have knowhow themselves on how the application operates or can allocate those resources ad hoc
- The platform owner(s) who either have knowhow themselves on how the platform operates or can allocate those resources ad hoc

Virtualization platforms, such as VMware and Hyper-V, will provide an export tool to kickstart the compilation of an inventory. Or even better, if the enterprise uses ITIL CMDB (Configuration Management Database), an export of this would be richer on data. AWS Application Discovery Service[4] is an AWS tool that can be used to discover the IT landscape, and provide a starting point, which can be enriched manually afterwards.

The application inventory can be put into a spreadsheet or preferably into a suitable tool such as AWS MPA[5]. AWS MPA provides strong import and enrichment features to allow the compilation and population of the inventory. An overlap may be present between the team who designs the landing zone and the team who compiles the application inventory.

If migration velocity is a key driver, then the application inventory should be built prior to designing the landing zone. This will create a similar setup, decreasing the mapping effort between on-premises and cloud as they are similar and promote migration velocity. Once the client is in the cloud, the modernization can be started.

If business agility is a key driver, then the landing zone should be built prior to the application inventory, as it will promote a landing zone design based on the current and future vision of the enterprise. This increases the mapping effort, and the need to revisit landing zone design as the on-premises and landing zone differ from each other. The mapping effort must be balanced against cost and time of migration.

4 https://aws.amazon.com/application-discovery/
5 https://docs.aws.amazon.com/prescriptive-guidance/latest/migration-tools/aws-services.html#mpa

2.2 Migration Planning

The migration process essentially consists of 3 phases executed in sequence:

01. Pre-Migration
02. Migration
03. Post-Migration

These 3 phases are collectively identified as a Migration Wave, and migration planning is the art of populating all migration waves with a scope and a team. The scope contains servers segregated by application boundaries, that is, each wave contains one to more applications in a complete form (optimally). This includes front end servers, backend servers, all dependencies, etc. This allows each wave to migrate independently from other waves.

In the real world, this can be difficult to do. On-premises often utilize shared services, such as a large monolithic database servicing multiple applications or shared load balancers. In this case, applications must be split with great care, taking increased network latency and network bandwidth into consideration as applications are migrated across multiple waves, and thereby temporarily live in a hybrid setup, where some components of the application are in Cloud and some on-premises.

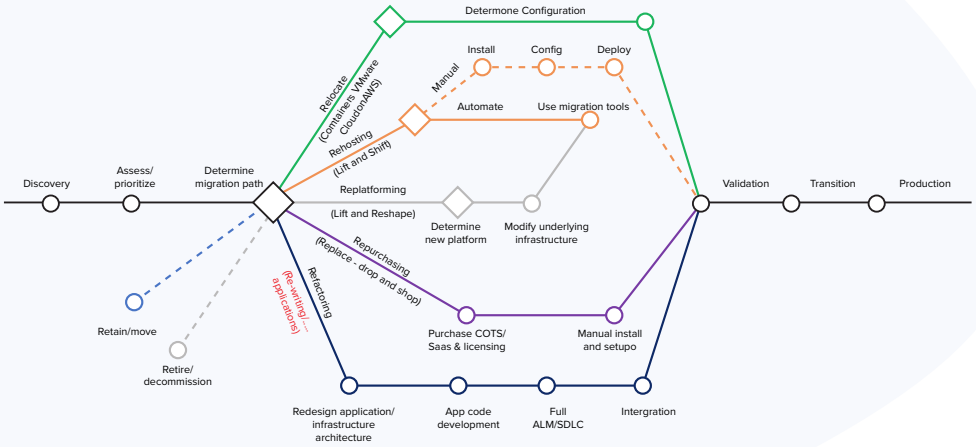
Each wave team will consist of members from the migration team and platform/application teams to ensure sufficient knowhow on how the application operates, so they can perform verification and troubleshoot issues if they arise. Quality wave planning is essential for migrating at scale. It enables the enterprise to identify and execute waves in parallel according to available resources and against an existing timeline.

The sequence of waves can be planned based on complexity, cost-savings, agility and more. This must match the business drivers identified in the business case. In our experience complexity combined with risk is usually chosen. This usually provides the shortest migration time and translates to migrating non-production simple applications in early waves and production and complex applications in final waves. This approach also matures the existing organization with cloud skills in an incremental fashion.

As an example: for a previous client who had a hard exit date from existing data centers, the key driver was velocity to respect the deadline. As the enterprise was new to cloud, there was a lack of skills and experience, and no time for prior training. Wave planning was performed with staffing each wave from Xebia to ensure skills and experience, and plan each wave with increasing application complexity, so the migration team could learn in parallel with migration delivery. This ensured quick wins, momentum and the ability to fan out into more parallel waves as their own people built the experience.

2.3 Migration Strategies

Based on the drivers defined in the business case, each application must be assessed against the following decision tree:



Retain

In some situations, an application is retained on-premises. Generally, we see this scenario where the application is critical for the business with impeding factors that prevent its running in the cloud. For example, the application is located in an area where AWS doesn't have a Region, and the application requires low latency. Customers can also revisit such applications that fall in this category later when circumstances change.

Retire

During a migration discovery phase, customers might find an application is no longer necessary, and it might be decommissioned.

Relocate

Applications running on VMware and containerized applications can be quickly relocated to AWS using the host applications familiar to enterprises.

Rehost

Also known as "lift-and-shift." Applications are brought to the AWS cloud, without modification, using tools like AWS Application Migration Service[6]. Applications are easier to optimize or re-architect after they are running in the cloud.

Repurchase

The application is moved as a software as a service (SaaS) platform that replaces all components of an application and assumes management tasks for the application infrastructure. AWS Marketplace enables customers to find, test, buy, and deploy software that runs on AWS from independent software vendors.

Replatform

Sometimes referred to as lift, tinker, and shift. In a replatform strategy, a client moves to the cloud and takes advantage of some cloud optimizations without changing the core of the application. The services used by clients target business drivers discovered in the business case, for example, to save time and increase business agility. Examples of replatform can be databases into AWS RDS[7] or applications into AWS Elastic Beanstalk[8].

Refactor

Redesign application architecture or rewrite an application during migration to make it a cloud-native application.

6 https://aws.amazon.com/application-migration-service/

7 https://aws.amazon.com/rds/

8 https://aws.amazon.com/elasticbeanstalk/

2.4 Identifying the Migration Strategy

To help navigate the decision tree, this flowchart[9] can provide guidance on assessing applications against migration drivers in the business case.

Each migration strategy has its benefits and cost in terms of complexity, effort, cost, and agility.

Strategy	Complexity	Effort	Cost to Migrate	Cost to Operate	Agility
Retire	-	-	-	-	○
Retain	●	○	-	●	○
Relocate	◔	◔	◔	●	○
Rehost	◔	◔	◔	◔	◔
Repurchase	◔	◔	◔	◔	◔
Replatform	◔	◔	◔	◔	◔
Refactor	●	●	●	◔	●

Use the business case drivers to identify the migration strategy for each application. Most larger enterprises will choose to exit existing data centers or reduce them to a minimum. Migrating hundreds or thousands of servers in a Repurchase/Replatform/Refactor can be very costly in time, regardless of the cost savings or the increased business agility achieved. Time is usually a rare commodity in migration projects and enterprises will therefore put most applications and servers into either Relocate or Rehost strategy.

2.5 Next Chapter

The upcoming chapters will focus on how to execute a migration wave. They will deep dive into the Rehost strategy to provide examples on how this strategy can be executed at scale.

9 https://docs.aws.amazon.com/prescriptive-guidance/latest/application-portfolio-assessment-guide/prioritization-and-migration-strategy.html#migration-r-type

Chapter 3: Wave - Pre-Migration

Continuing from Chapter 2: Preparing for Migration, we reach a stage where we have a good understanding of what we are migrating and the migration strategy to be applied to each application. In this chapter we dive deeper into the actual migration of an application.

Each migration wave consists of 3 phases:

01. Pre-Migration

02. Migration

03. Post-Migration

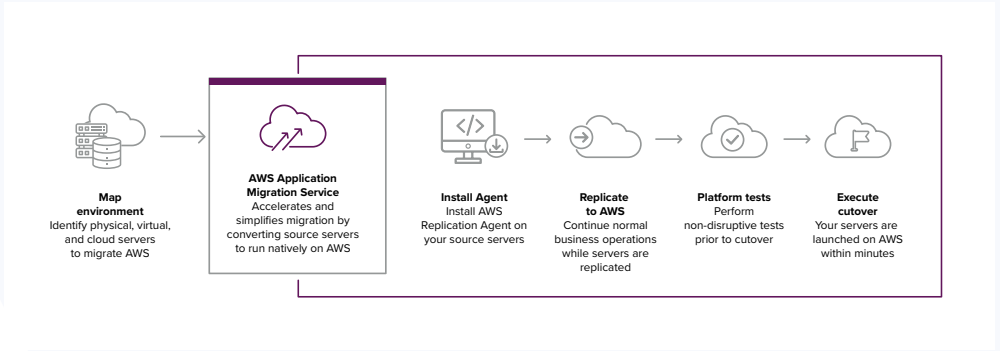
The tasks in each stage differ depending on what migration strategy has been chosen for the application. Most larger enterprises will include most applications into the “rehost” strategy, so they can quickly enter the cloud, and from there, start their modernization journey. This chapter will focus on “rehost” migration strategy pre-migration.

3.1 Migration Wave Process

The 3 stages can be broken down into the following tasks:

3.2 Technology

AWS Application Migration Service is the AWS service to use for migrating servers at scale in a Rehost migration strategy. It offers a non-intrusive agent which replicates running servers into AWS region and account of choice.



Agentless approach is also possible, but it has some limitations. The AWS recommendation is with an agent, and the focus of this chapter will be with the agent.

3.3 Roll-out of the Agent

The first main technical challenge is to design how the roll out of the agent should be implemented. The AWS Replication Agent is offered for Linux and Windows. For Windows 2003 and 2008 Base (not R2) legacy agent is provided.

Manually installing the agent on hundreds/thousands of servers is obviously not migration at scale. To devise the best plan for roll out, we must investigate the prerequisites of the agent:

- Different agent must be installed based on OS and OS version
- Prerequisites listed here must be fulfilled: https://docs.aws.amazon.com/mgn/latest/ug/Supported-Operating-Systems.html

This raises a challenge; how can automation be applied to the roll out of the agent at scale?

Installing new .Net versions or drivers/libraries is obviously not non-intrusive. And which tool to use? For example, for Windows, PowerShell was introduced as default in Windows 10 and Server 2016, prior to those versions it was an add-on. So, for any automation scripting we find ourselves using the lowest common denominator, e.g. old style DOS batch script or we write multiple scripts, each catering for different versions of the OS.

Most enterprises will have some flavor of IBM/HCL/Tivoli/<insert your choice here> of endpoint management software, which can roll out patches, manage inventory, ensure compliance, etc. to servers and workstations. A tool of this nature should be used to roll out the agent install and ensure prerequisites are met. Typically, such a tool will provide a more intelligent and flexible API/SDK/script language to ensure agent prerequisites are met and agent is installed. To use such a tool, a member from the endpoint management team must be allocated to the migration team.

The roll out should conform to the following design:

01. Roll out should be manageable, e.g., it should be possible for the migration team to define the wave to be migrated in the endpoint management tool, so roll out of the agent happens only to those servers & workstations when the migration wave is launched.

02. The roll out should ensure sufficient backup prior to modifying the system

03. The roll out should ensure prerequisites of the agent are met, and if needed, install any dependent software.

04. The roll out should install the agent using parameters, that is without user interaction, in the background, and initiate replication of the disks to the target AWS region and account.

05. It should be configurable which AWS region and account should be targeted for each agent installation, as a migration wave may contain servers to different AWS target accounts.

06. In case of any error, the automated process should switch over to a manual process where the migration team can manually resolve the issue.

In our experience, the percentage of automation increases throughout the early waves, as the migration team learns from their experience. Clients should aim for velocity in progress in the early waves and avoid targeting an unrealistically high degree of automation, until the initial experience has been achieved. Each migration wave will introduce its set of new complexity and challenges, which either should be included in the automation or handled manually, based on expected occurrence frequency. There will always be servers that fall outside of automation in a migration project. The aim is to minimize the volume of these servers.Once the agent is installed it will automatically initiate the connection to the AWS Replication server and start replicating the selected disks. The replication is a 2-stage process:

01. Replicate a snapshot of the complete disk image
02. Replicate the incremental changes since the snapshot

3.4 Agent Replication Transfer Rate

The early migration waves will define the transfer rate. Use throttling via AWS Application Migration Service (MGN) to avoid overloading the network connection or impact user experience. Unfortunately, at the time of writing this chapter, the throttling in MGN is not intelligent enough to be able to dynamically turn up/down depending on the time of the day.

Experience tells us that parallel replication is to be aimed for, with the first additional replications gaining 5-15% throughput, but as more are added, the returns are diminishing. The main parameters that impact transfer rate are (in no particular order):

01. CPU and RAM on source
02. Disk speed on source
03. Replication Server in AWS
04. Replication Disk speed in AWS
05. Network connection bandwidth

Of these, the first 2 may be inserted into the roll out to be a prerequisite of a minimum size or upgraded prior to migration. It's difficult to recommend a minimum size, as our experience shows that clients will usually overload their VMWare/Hyper-V/etc from anywhere between 1:1 to 1:5, so a minimum recommendation of X vCPU can be relative. However, for maximum transfer rate, ensure sufficient hardware configuration at both source and target end.

Put the transfer rate and time to completion metrics in a spreadsheet, together with the source server and AWS configuration. This will allow for easy comparison of the most optimal settings.

Once the initial transfer rate has been defined in the most optimal setting (around wave 2/3), a simple calculation can be made:

Total GB to replicate / transfer rate per day = total days to replicate

On top of that, the launching, testing and cut over time can be added, and the migration project timeline can be verified.

If the bottleneck is in the network connection bandwidth, check with your ISP if the bandwidth can be increased during the migration period.

If the calculated time exceeds the predefined timeline, then engage your AWS partner and AWS to find alternative solutions. AWS currently does not offer AWS Snow Family services with AWS Application Migration Services, so the alternative may be of more custom nature.

In the next chapter we further describe the activities in the migration stage

Chapter 4: Wave - Migration

Continuing from Chapter 3: Pre-Migration, we reach a stage where the servers in the current migration wave have been replicated as a snapshot, and are replicating incremental changes. These servers are now ready for launch and testing. In this chapter we dive deeper into these activities.

4.1 Launch target server

Finally, we reach the exciting stage, where the AWS Application Migration Service (MGN) Console displays the original snapshot to be replicated, and the server is ready for launch in AWS. However, before you press the launch button, there is a major consideration to make.

Most enterprises will have Active Directory or an equivalent to store users, groups, servers, and access management. In this chapter Active Directory is used as an example. Critical design thoughts must be put into how the AD is migrated, for example:

01. The AD is extended into AWS within the same domain. AWS Servers are provisioned within this directory and domain.

02. A new AWS managed Active Directory (AWS Directory Service) is provisioned with a new domain name. AWS Servers are provisioned into this directory.

AWS Route 53[10] is usually used in conjunction with AD and fronts AD. AWS Services that can join AD do so, or otherwise will have native support to join R53 domain.

In our experience, this is a very error-prone area, and strategic thoughts must be put into how AD is migrated and how it impacts the migration and what risks it entails. The migration of the directory is out of scope of this series of chapters, as the subject is complex enough for it to require its own series of chapters, however the two mostly used models are described in the next two sections.

10 https://aws.amazon.com/route53/

4.2 Extended AD domain

If this model is chosen, then the AD domain controllers are migrated in one of the final waves. Either with a rehost strategy, or with a replatform strategy to AWS Directory Service. This usually ends up with a big-bang directory switch approach in the end, with the risk profile of a big-bang.

Target servers launched in this domain should be subsequent to turning the source servers off, to avoid duplicate names conflict on the domain.

4.3 Separate AD domain

In this model, a new AD Domain is used for the new AWS servers which are launched. Target servers can be launched simultaneously with the source servers running, as they are on separate domains, and duplicate names conflict will not occur. However, great care must be taken to ensure that cross-server access is configured identically in the new domain or existing configuration is extended in the case of hybrid setup, where target servers are dependent on servers on-premises in another domain and vice versa. For example, in a scenario where SQL server database access is configured in AD for the application server(s). This requires usage of fully qualified domain names and AD 2-way trust, both changes introduce more challenges, which are out of scope for this whitepaper.

4.4 Launch of target server

So, you've got the directory sorted, and think the provisioning of an EC2 server (instance) from the replicated snapshot will be a cold breeze on a hot summer day. Think again, it is more likely to be like a hailstorm during a tsunami. In other words, it has its own set of complex challenges. Most common issues are well known by an AWS partner due to past experience, and some will be resolved using a trial-and-error approach. In our experience the first 5-10 waves will help filter out the most common issues, which can be made into prerequisites checked during the roll out of the agent.

The target server configuration will be pre-identified from the business case, where AWS Migration Portfolio Assessment (MPA) or some other tool was used to map source server configurations to target server configurations in a cost-efficient manner. And this dataset can be used to configure the target server configuration.

Once launched, the EC2 instance will undergo 4 main milestones:

01. Converting the source image into an EC2 instance

02. Booting the EC2 instance and loading the OS

03. Getting the EC2 instance onto the new network, getting an IP and joining the domain

04. Making the EC2 available across a Remote Desktop/SSH connection

If any of the above four milestones fail, the server will not be reachable/accessible and remote troubleshooting is very limited.

4.5 Converting the source image into an EC2 instance

The replicated source image is converted into an EC2 instance and prepared with an install of new drivers and configuration. This process can take anywhere from five minutes to two hours. In our experience, very small EC2 instance sizes should be launched with a larger size and then downgraded later to its original size, to allow for faster conversion. During the conversion, the OS is rebooted multiple times, and a premature login attempt may leave the user confused, as the OS disconnects the user's session and restarts. AWS, at the time of writing this chapter, does not provide a clear-cut message on when the EC2 instance is ready for use, the migration team should plan the launch of EC2 instances in a timely fashion.

4.6 Booting the server and loading the OS

The EC2 instance may fail to start up. This could be due to dual boot managers or incompatible hardware. Use the EC2 Instance screenshot feature to see where/how it fails during startup. For complete failure to startup OS, investigate if the OS is too old for the hardware used. E.g. Windows 2003 will not be compatible with the AWS Nitro EC2 family. If OS loading initiates, but fails to complete, shut down the EC2 instance, detach the EBS volume(s), attach them to a working EC2 instance as secondary disks and use EC2Rescue tool to troubleshoot. EC2Rescue[11] will collect vital logs and perform basic checks to provide a guideline on the troubleshooting direction.

4.7 Getting the EC2 instance onto the new network, getting an IP and joining the domain

Clients may run their source servers with static or dynamic IP address(es). AWS Application Migration Service (MGN) supports both and will prepare the EC2 instance based on the launch configuration. The target OS will be configured during launch to make the EC2 instance available on the network and the AD Domain. If this step fails, the EC2 instance will not be reachable, and therefore not accessible. There are no troubleshooting steps on the target EC2 instance since it is not reachable. All troubleshooting must happen on the source server, as something failed during the conversion step. The conversion of network drivers and configuration is simple and will not support complex network configurations on the source servers. Ensure additional secondary network configurations and custom configurations are removed prior to launching the EC2 instance. Attach the disks to another EC2 instance (as described above), to go through the logs for clues.

If the EC2 instance IP is unreachable via ping, then check your Network ACLs and Security Groups. Optionally use the AWS Reachability Analyzer to help troubleshoot. One must quickly clarify if the issue resides in the AWS VPC configuration, or within the OS. If the EC2 instance is reachable, but not pingable, then it is an OS issue.

If the EC2 instance can be accessed via IP, but not via fully qualified domain name, then check reachability to the AD domain controller(s) and ensure timeouts have not occurred due to increased latency between AWS region and on-premises.

4.8 Making the EC2 instance available across a Remote Desktop/SSH connection

If the EC2 instance is reachable from client workstation, but the administrator/user cannot log into the EC2 instance using Remote Desktop or SSH, ensure those ports are open in AWS (check NACLs and Security Groups), that the EC2 instance has joined the domain (if you are using a domain user to log in with) and that there is no firewall/software within the EC2 instance blocking those ports and/or services.

4.9 Testing

Once the migration team has verified that the target EC2 Instance has been launched successfully and is reachable, the application owners must verify the workload. A typical challenge lies here which should be addressed on a process level: What happens if the server is migrated successfully, but the application verification fails?

What typically happens is that the application is tested more thoroughly than usual, and it discovers existing application errors, or errors due to unresolvable dependencies if the domain name/security configuration has changed.

11 https://aws.amazon.com/premiumsupport/knowledge-center/ec2rescue-windows-troubleshoot/

In an optimal scenario, application errors would be for application teams and the migration team would continue with the wave and finalize it. In real-life it is usually a balanced approach, for example: production applications must be verified, and non-production applications must be handed over to the application teams for troubleshooting and resolution.

Regardless of how the process is defined, it is key for executing migration at scale, that the root cause is identified, and if it is not related to migration, a hand over to the application team or a support team is performed, so the migration team can continue their migration activities at the same pace. In the next chapter we describe the activities in the post-migration stage.

Chapter 5: Wave - Post-Migration

Continuing from Chapter 4: Migration we reach a stage where the servers are launched in AWS and tested. In this chapter we dive deeper into the post-migration activities and how to close a migration wave.

5.1 Cut over

AWS Application Migration Service (MGN) has a process in which an EC2 instance is launched, tested, terminated, and then cut over by relaunching it from the newest replicated data. This relaunch may make sense if the testing activities for cut over are different, the cut over test team is different, or if the newest replicated data must be included.

If neither of these cases apply, we suggest terminating the MGN process after testing, so the additional step of relaunching for cut over is saved. This will accelerate the migration process significantly, as a relaunch reintroduces the risk of issues described in the previous chapter.

5.2 Decommission source server

When a source server is migrated successfully, the source server will be in the powered on or off state. If one of the key drivers identified in the business case is cost savings, and by deleting the server cost can be saved, then each migration wave should delete the servers migrated from on-premises. If cost savings are not a driver, or cannot be achieved due to self-owned hardware, then they can remain powered down and be deleted in one of the final migration waves.

5.3 Rollback strategy

Rollback should be the last step taken in any migration wave and not be allowed after the migration wave is closed. The effort required to do a rollback in a wave ranges from easy to difficult, depending on the model chosen for the Directory (as described in an earlier chapter). If the hostnames and domain does not change, the effort should be minimal to rollback to on-premises servers. If the hostnames and/or domain changes, the effort is significant as CI/CD pipelines, application properties, security, etc. must be reconfigured for the rollback to be operational.

This effort will neutralize any acceleration within the migration and put pressure on the migration plan timeline.

If a rollback occurs, the migration team must absorb the lessons learned into their processes, prerequisite checks for the agent roll out, and knowledge database, so it never occurs again due to the same reasons. As waves are executed in parallel at scale, sufficient knowledge sharing practice should also be in place.
This is simplified by using central documentation across the migration wave teams.

Each unique application, database, load balancer, etc. will introduce a new set of findings, from which the migration team must absorb the learnings from, and proactively react upon future waves.

Data loss is one of the biggest challenges with rollback, hence the rollback cannot occur after the migration wave is closed. How to ensure that data is not lost for production applications during rollback? The answer to this is more complexity. Technology such as AWS DMS[12] and AWS DataSync[13] may help with data synchronization in a continuous fashion or a single run, but the increased complexity comes with increased effort and risk, both counterproductive to executing migration at scale.

12 https://aws.amazon.com/dms/
13 https://aws.amazon.com/datasync/

Closing Words

This wraps up our series on how an enterprise organization can execute a cloud migration at scale. We touched the main considerations, challenges, and solutions for migrating at scale, and skipped subjects well-documented by AWS and the internet in general for cloud migration.

Migration projects are complex (we have barely scratched the surface) and just aimed to inspire the reader's thought and design process for structuring their migration project. The cloud journey doesn't end here, however. Usually organizations will at this stage still have distance to travel in cloud adoption within the organization and the modernization journey for applications is still ahead.

Xebia has built their knowledge and skills from numerous migration projects, ranging from small to large scale, and are a strong AWS partner to create an alliance with. Our mission is to guide our customers to become self-sufficient and innovative on a successful and effective cloud journey. We do this through both consultancy and training, with sharing knowledge as our main goal. Start your cloud journey well-prepared and reach out to us!

xebia.com

About Xebia

Xebia is an IT Consultancy and Software Development Company that has been creating digital leaders across the globe since 2001. With offices on every continent, we help the top 250 companies worldwide embrace innovation, adopt the latest technologies, and implement the most successful business models.
To meet every digital demand, Xebia is organized into chapters. These are teams with tremendous knowledge and experience in Agile, DevOps, Data & AI, Cloud, Software Development, Security, Quality Assurance, Low Code, and Microsoft Solutions. In addition to high-quality consulting and state-of-the-art software, Xebia Academy offers the training that modern companies need to work better, smarter, and faster. Today, Xebia continues to expand through a buy and build strategy. We partner with leading IT companies to gain a greater foothold in the digital space.

Find more information on how Xebia is driving innovation at www.xebia.com.

Xebia