WHITEPAPER

Cloud Services

EASY SYSTEMS

PART OF 4CEE

Revision history

Version	Date	Author(s)	Description
1.12	27-06-2023	Hans van der Kolk Erwin Rebergen Frits Frederiks	Changed product name “Next” to “Easy1” Better distinction between the cloud platform and its services. Section 2.2. Currentgen removed. Chapter 4 removed (Secure software development). Chapter 10 removed (Cloud Restrictions). All superseded by information in other chapters.
1.11	09-01-2023	Erwin Rebergen Sjoerd van den Top	Added links and spelling corrections Added chapter 2.1.5 Quality of our code
1.10	01-11-2021	Erwin Rebergen	Corrected paragraphs 8.3.1 and 8.3.2 Update own software vs third party. Removed Keel from paragraph 5.2 Software deployment tools.
1.9	17-03-2021	Bas Bremer	The following changes: - Chapter 7: Added the AWS shared responsibility model - Paragraph 10.3.3: Improved process description - Paragraph 10.6: Several additions - Paragraph 10.7: Added ISO 27017 - Paragraph 10.8.1: Removed typo
1.8	25-01-2021	Bart Timmermans	Audits initiated by customers
1.7	08-01-2021	Bas Bremer	Several changes
1.6	02-12-2020	Bas Bremer	Added CurrentGen software
1.5	26-02-2020	Bart Timmermans	Data classification and edit on 10.4 DATA PROTECTION
1.4	14-02-2020	Bart Timmermans	Adding time sync services and publish document
1.3	12-02-2020	Bart Timmermans	Publish
1.2	10-02-2020	Bart Timmermans	Lay-out and some comments
1.1	29-01-2020	Bart Timmermans	Lay-out and releasing document
1.0	28-01-2020	Bas Bremer	Cloud and security aspects

1.0	28-01-2020	Bart Timmermans	INTRODUCTION
1.0	28-01-2020	Leon Krol	Testing
0.9	24-01-2020	Jacco Dieleman	SOFTWARE ARCHITECTURE and TECHNOLOGY STACK
0.8	14-01-2020	Bart Timmermans	BASIC PRINCIPLES EASY SYSTEMS NEXT CLOUD
0.7	13-01-2020	Bart Timmermans	FUNCTIONALITIES AND USE SCENARIOS OF THE NEXT APPLICATIONS
0.6	10-01-2020	Leon Krol	SECURE SOFTWARE DEVELOPMENT
0.5	09-01-2020	Leon Krol	SECURE SOFTWARE DEPLOYMENT
0.4	08-01-2020	Bart Timmermans	Functional description of the product lines
0.3	06-01-2020	Bart Timmermans	Table of contents
0.2	16-12-2019	Bart Timmermans	Determining the content
0.1	03-12-2019	Bart Timmermans	Setting up framework

Table of contents

1 Introduction

This document describes the technical and operational details of the Easy Systems Cloud Platform, a SaaS (Software as a Service) platform for which Easy Systems uses Amazon Web Services (AWS) as its hosting provider. The purpose of this document is to give insight in the infrastructure, principles, security, development and deployment process of the Easy Systems Cloud Platform and the SaaS services that are provided with it.

2 Basic principles of Easy Systems Cloud

2.1 Software as a Service

The Easy Systems Cloud platform is designed for Continuous Delivery, Scalability, High Availability and Security.

2.1.1 Continuous Delivery

Easy Systems software like Easy1 is based on the idea to produce software in short cycles, ensuring that the software can be reliably released at any time. Building, testing, and releasing software with greater speed and frequency. The approach helps reduce the cost, time, and risk of delivering changes by allowing for more incremental updates to applications in production.

2.1.2 Scalability

Our SaaS solution is highly scalable thanks to the use of containerized applications which are hosted using tools such as Kubernetes. The software is hosted on the AWS Public Cloud Platform which makes it possible to scale in seconds with unlimited resources.

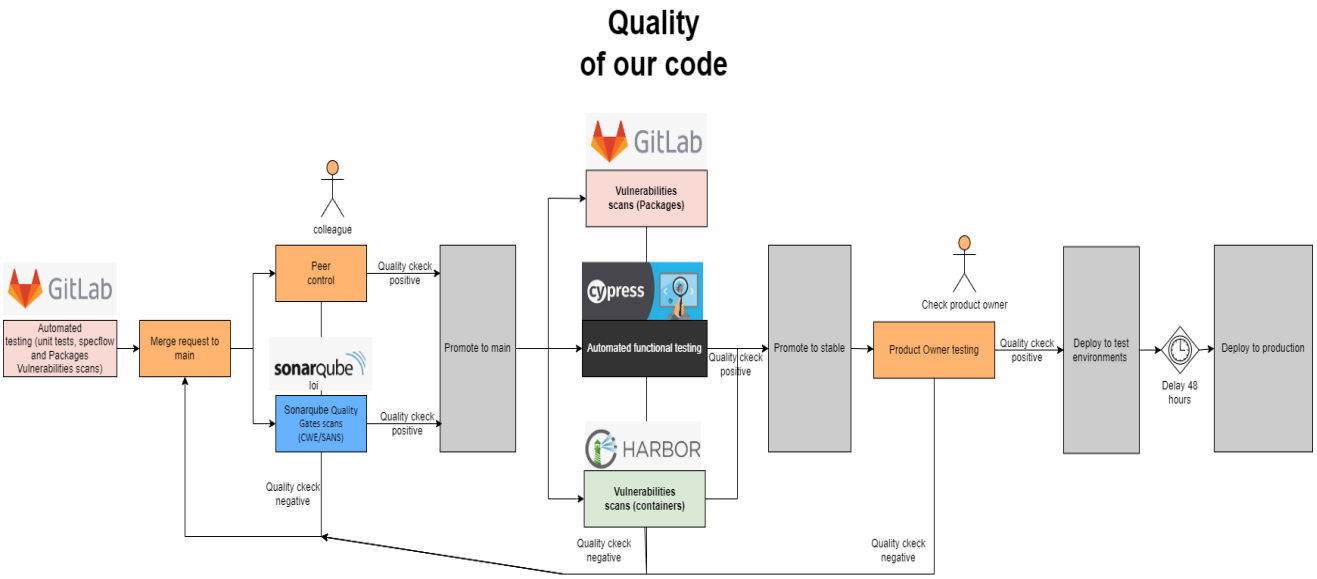
2.1.3 High availability

The SaaS solution is highly available thanks to the use of containerized applications which are hosted using tools such as Kubernetes. Kubernetes makes it possible to host multiple servers per service/application. The use the AWS Public Cloud Platform makes it possible to host applications in multiple regions or multiple zones within a region.

2.1.4 Security

The Easy Systems Cloud team is primarily responsible for the security management of virtual networks. The physical networks are maintained by our cloud supplier, Amazon Web Services. We work together to keep the software platform stable and secure. The security measures are described at chapter 8 - Cloud Services.

2.1.5 Quality of our code



Within the development process there are several moments when the code is reviewed. We do this through scanning, automated testing, through automated functional testing, through peer control/reviews and through scans of the developed code.

1. Quality Gates (quality policy)
2. Quality Gates Conditions on New Code
3. Quality Gates scans of the developed code OWASP top 10
4. Quality Gates scan on use of Hard-coded Credentials (CWE-798)
5. Quality Gates scan on use of Hard-coded Password (CWE-259)
6. Quality Gates scan on CWE/SANS TOP 25 Most Dangerous Software Errors
7. Automated testing (unit tests)
8. Through peer control/reviews
9. Automated functional testing

1. **Quality Gates** (quality policy)

Quality Gates enforce a quality policy in the organization by answering one question: is my project ready for release? To answer this question, you define a set of conditions against which projects are measured. We want to ensure stronger requirements on our applications. This is why you use a high quality gate. The "Sonar way" Quality Gate is provided by SonarSource, , and considered as built-in and read-only. This Quality Gate represents the best way to implement the Clean as You Code concept by focusing on new code. With each SonarQube release, we automatically adapt the default quality gate according to SonarQube's capabilities.

With the Quality Gate, We enforce ratings (reliability, security, security review, and maintainability) based on metrics on overall code and new code. These metrics are part of the default quality gate.

2. Quality Gates Conditions on New Code

Conditions on New Code apply to all branches and to Pull Requests.

Metric	Operator	Value
Coverage	is less than	80.0%
Duplicated Lines (%)	is greater than	3.0%
Maintainability Rating	is worse than	A
Reliability Rating	is worse than	A
Security Hotspots Reviewed	is less than	100%
Security Rating	is worse than	A

3. Quality Gates scans of the developed code (Automatic scans to preserve the quality of the delivered code)

In addition to a mandatory review from a colleague, we also have automatic scans to preserve the quality of the delivered code and ultimately our software. We do this based on the OWASP top 25 check, automatically with every software release using SonarQube. We use the standard settings. No exclusions are applied.
The check is done on multiple subjects;
Complexity: Complexity refers to cyclomatic complexity, a quantitative metric used to calculate the number of paths through the code. Whenever the control flow of a function splits, the complexity counter gets incremented by one. Each function has a minimum complexity of 1. This calculation varies slightly by language because keywords and functionalities do.
Cognitive complexity: How hard it is to understand the code's control flow.
Duplications : The number of duplicated blocks of lines.
Issues: New issues, False positive issues (The total count of issues marked false positive), Open issues (The total count of issues in the Open state), Confirmed issues (The total count of issues in the Confirmed state.), Reopened issues (The total count of issues in the Reopened state.)

Maintainability: Code smells (The total count of code smell issues.), New code smells (The total count of Code Smell issues raised for the first time on New Code.), Maintainability rating, Technical debt.
Quality gates: Quality gate status (The state of the quality gate associated with your project. Possible values are ERROR and OK.), Quality gate details (For all the conditions of your quality gate, you know which condition is failing and which is not.)
Reliability: Bugs (The total number of bug issues.), New Bugs (The number of new bug issues.), Reliability remediation effort, Reliability remediation effort on new code.
Security: Vulnerabilities (The number of vulnerability issues.), Vulnerabilities on new code (The number of new vulnerability issues.), Security Rating, Security hotspots (security_hotspots): The number of Security Hotspots, Security hotspots on new code (The number of new Security Hotspots on New Code.), Security review rating (The security review rating is a letter grade based on the percentage of Reviewed Security Hotspots. Note that security hotspots are considered reviewed if they are marked as Acknowledged, Fixed or Safe.), Security review rating on new code (The security review rating for new code.), Security hotspots reviewed (The percentage of reviewed security hotspots.), New Security Hotspots Reviewed (The percentage of reviewed security hotspots on new code.)
Size: Classes (The number of classes), Comment lines (The number of lines containing either comment or commented-out code.) Comments (The comment lines density), Directories (The number of directories.), Files (The number of files.), Lines (The number of physical lines), Lines of code (The number of physical lines that contain at least one character which is neither a whitespace nor a tabulation nor part of a comment.), Lines of code per language (The non-commented lines of code distributed by language.), Functions (The number of functions), Projects (The number of projects in a Portfolio.), Statements (The number of statements.)
Tests: Condition coverage (On each line of code containing some boolean expressions, the condition coverage answers the following question: 'Has each boolean expression been evaluated both to true and to false?'. This is the density of possible conditions in flow control structures that have been followed during unit tests execution.), Condition coverage on new code (This definition is identical to Condition coverage but is restricted to new/updated source code.), Condition coverage hits (A list of covered conditions.), Conditions by line (The number of conditions by line.), Covered conditions by line (The number of covered conditions by line.), Coverage (A mix of line coverage and condition coverage.), Line coverage on new code (This definition is identical to Line coverage but restricted to new/updated source code.), Line coverage hits (A list of covered lines.), Lines to cover (The number of lines of code that could be covered by unit tests (for example, blank lines or full comments lines are not considered as lines to cover.), Lines to cover on new code (This definition is Identical to Lines to cover but restricted to new/updated source code.), Skipped unit tests (The number of skipped unit

tests.), Uncovered conditions (The number of conditions that are not covered by unit tests.), Uncovered conditions on new code (This definition is identical to Uncovered conditions but restricted to new/updated source code.), Uncovered lines (The number of lines of code that are not covered by unit tests.), Uncovered lines on new code (This definition is identical to Uncovered lines but restricted to new/updated source code.), Unit tests (The number of unit tests.), Unit tests duration (The time required to execute all the unit tests.), Unit test errors (The number of unit tests that have failed.), Unit test failures (The number of unit tests that have failed with an unexpected exception.), Unit test success density (Test success density = (Unit tests - (Unit test errors + Unit test failures)) / (Unit tests) * 100)

4. Quality Gates scans on use of Hard-coded Credentials

Hard-coded credentials typically create a significant hole that allows an attacker to bypass the authentication that has been configured by the software administrator. This hole might be difficult for the system administrator to detect. Even if detected, it can be difficult to fix, so the administrator may be forced into disabling the product entirely. There are two main variations:
Inbound: the software contains an authentication mechanism that checks the input credentials against a hard-coded set of credentials.
Outbound: the software connects to another system or component, and it contains hard-coded credentials for connecting to that component.
In the Inbound variant, a default administration account is created, and a simple password is hard-coded into the product and associated with that account. This hard-coded password is the same for each installation of the product, and it usually cannot be changed or disabled by system administrators without manually modifying the program, or otherwise patching the software. If the password is ever discovered or published (a common occurrence on the Internet), then anybody with knowledge of this password can access the product. Finally, since all installations of the software will have the same password, even across different organizations, this enables massive attacks such as worms to take place.
The Outbound variant applies to front-end systems that authenticate with a back-end service. The back-end service may require a fixed password which can be easily discovered. The programmer may simply hard-code those back-end credentials into the front-end software. Any user of that program may be able to extract the password. Client-side systems with hard-coded passwords pose even more of a threat, since the extraction of a password from a binary is usually very simple.

5. Quality Gates scans on use of Hard-coded Password

A hard-coded password typically leads to a significant authentication failure that can be difficult for the system administrator to detect. Once detected, it can be difficult to fix, so the administrator may be forced into disabling the product entirely. There are two main variations:
Inbound: the software contains an authentication mechanism that checks for a hard-coded password.
Outbound: the software connects to another system or component, and it contains hard-coded password for connecting to that component.
In the Inbound variant, a default administration account is created, and a simple password is hard-coded into the product and associated with that account. This hard-coded password is the same for each installation of the product, and it usually cannot be changed or disabled by system administrators without manually modifying the program, or otherwise patching the software. If the password is ever discovered or published (a common occurrence on the Internet), then anybody with knowledge of this password can access the product. Finally, since all installations of the software will have the same password, even across different organizations, this enables massive attacks such as worms to take place.
The Outbound variant applies to front-end systems that authenticate with a back-end service. The back-end service may require a fixed password which can be easily discovered. The programmer may simply hard-code those back-end credentials into the front-end software. Any user of that program may be able to extract the password. Client-side systems with hard-coded passwords pose even more of a threat, since the extraction of a password from a binary is usually very simple.

6. Quality Gates scans on CWE/SANS TOP 25 Most Dangerous Software Errors

1	CWE-787	Out-of-bounds Write
2	CWE-79	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
3	CWE-89	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
4	CWE-20	Improper Input Validation
5	CWE-125	Out-of-bounds Read
6	CWE-78	Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')
7	CWE-416	Use After Free

8	CWE-22	Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
9	CWE-352	Cross-Site Request Forgery (CSRF)
10	CWE-434	Unrestricted Upload of File with Dangerous Type
11	CWE-476	NULL Pointer Dereference
12	CWE-502	Deserialization of Untrusted Data
13	CWE-190	Integer Overflow or Wraparound
14	CWE-287	Improper Authentication
15	CWE-798	Use of Hard-coded Credentials
16	CWE-862	Missing Authorization
17	CWE-77	Improper Neutralization of Special Elements used in a Command ('Command Injection')
18	CWE-306	Missing Authentication for Critical Function
19	CWE-119	Improper Restriction of Operations within the Bounds of a Memory Buffer
20	CWE-276	Incorrect Default Permissions
21	CWE-918	Server-Side Request Forgery (SSRF)
22	CWE-362	Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')
23	CWE-400	Uncontrolled Resource Consumption
24	CWE-611	Improper Restriction of XML External Entity Reference
25	CWE-94	Improper Control of Generation of Code ('Code Injection')

7. **Through peer control/reviews**

Before any changes are made to the software, our process enforces peer checks/reviews.

8. **Automated testing (unit tests)**

Unit tests are typically automated tests written and run by software developers to ensure that a section of an application (known as the "unit") meets its design and behaves as intended.

9. Automated functional testing

Before code is promoted within our pipeline, the code goes through front-end testing in which functional scenarios are run through. We use this to make sure whether end-user scenarios continue to work in new versions of our software. If one of these tests fails, the software will not be promoted or not distributed

10. Container Scanning

The Docker images can contain known vulnerabilities. By including an extra Container Scanning job in our pipeline that scans for those vulnerabilities and displays them in a merge request, we use the GitLab container Scanning to audit our Docker-based apps.

11. Packages Vulnerabilities scans

Vulnerabilities scans of main. The scan results from a pipeline are only ingested after all the jobs in the pipeline are complete.
The report is available for users in projects, groups, and the Security Center.
At all levels, the Vulnerability scan contains:

- Totals of vulnerabilities per severity level.
- Filters for common vulnerability attributes.
- Details of each vulnerability, presented in tabular layout.

3 Easy1 software development and architecture

3.1 Used concepts

The following concepts are used:

• Domain Driven Design

• Event Sourcing

• Command and Query Responsibility Segregation (CQRS)

3.1.1 Domain Driven Design

Domain-driven design (DDD) is an approach to developing software for complex needs by deeply connecting the implementation to an evolving model of the core business concepts. Domain-driven design is not a technology or a methodology. DDD provides a structure of practices and terminology for making design decisions that focus and accelerate software projects dealing with complicated business domains. Key concepts from DDD:

- Domain: a business domain. Examples: Expenses, Contracts, Invoices. There are also domains that are less important for the business. We call these Supporting Domains. Examples: Notifications, Authentication.
- Ubiquitous language: a common, rigorous language between developers and users.
- Bounded Context: a (part of a) software solution with clear boundaries. In DDD there is ideally a 1:1 relation between a Domain and a Bounded Context.
- Context Map: a way to show relations of one bounded context between others. Important to note is which are upstream and which are downstream. It is often in the form of a diagram.
- Entity: an element in the domain which can be referenced by its identity. Example: Declarant, Expense.
- Aggregate: an entity in the domain which consists of other entities. The top of the structure is called Aggregate Root. Examples: DeclarantCreditCardPeriod, ContractFile.
- Value Object: element in the model defined by its properties, without an id. Always part of an Entity. Example: AmountOfMoney, ExpenseDate.

3.1.2 Event sourcing

Instead of storing just the current state of the data in a domain (often in a relational database), we use an append-only store to record business events that took place on the entities. We record events on the level of aggregate roots. You can read more about this here.

3.1.3 Command and query responsibility segregation (CQRS)

We follow this pattern to segregate operations that read data from operations that update data by using separate interfaces. This can maximize performance, scalability, and security. Supports the evolution of the system over time through higher flexibility, and prevents update commands from causing locking, or worse, merge conflicts at the data level. You can read more about this here. In our architecture this translates to:

- We have a command server per domain. These handle commands (like, SubmitExpense) and as a result publish events (like, ExpenseSubmitted). Events are stored in an event store.
- We have query servers per application. These listen to events (published by the event store) and created views for theses which are stored in a database. When a client queries the query server for a certain view, it can fetched from the database an served to the client. We call these views: ViewProjections. Under the hood these are fed by AggregateProjections.

3.2 Multi-tenancy

We use a Multi-Tenant architecture which consists of multiple customers using a single instance of an application running on a single instance of an operating system on a common hardware platform, with only a database or data-source being different between customers.

3.3 Technology stack

The Easy Systems backend software is created using C# with ASP.NET Core. The software is hosted in the AWS Cloud, using Elastic Kubernetes Services (EKS). EKS is the AWS Kubernetes service which makes it possible to integrate with AWS components (such as Auto Scaling). Kubernetes makes use of Docker and organizes containers into Pods. The desktop clients are created using Angular with TypeScript. the mobile clients are created using Flutter with C#.

3.4 Software components

At run-time our software is composed of several software components, for most of them, the source is maintained in a separate Git repository. We distinguish the following types:

- Event Store, this stores the business events
- Command Servers, handles commands. Each command server pod consists of:
    - An api host (which handles API calls, transform these to commands, process the commands, store events in event store)
    - An document database (used to store projections)
- Query Servers, maintains projections of the events on views and handle queries. Each query server pod consists of:
    - An api host (which handles incoming API calls en queries the document database to create the response)
    - A projection manager (which listens to events that stream from event store, create projections and store these in the document database)
    - A document database (used to store projections)
- Desktop Clients, provide an user interface from a browser. The desktop clients are served to the browser from pods, but note that they run in the browser.
- Mobile Clients, provide an user interface from a mobile app. We support iOS and Android. supporting components
- API Gateway, acts as an entry point for the tenant. Each request enters the instance / tenant through the API Gateway.
- Message Bus. Within the tenant, components talk to each other through messages that are placed on the message bus. We distinguish two types of messages: commands and events.
    - Components handling media (like documents and images)
    - Components supporting authentication and authorization
    - A document database to persist the status of running processes

The run-time components share some common concerns. To avoid duplication we share solutions for these concerns in packages. Some of the packages are grouped in separate Git repositories. For example: general command server concerns and general query server concerns.

3.5 Interaction between software components

The following diagram gives a high-level overview of how the software components work together. Note that the diagram depicts one customer instance. The Application Load Balancer takes care of routing requests to the correct customer instance.

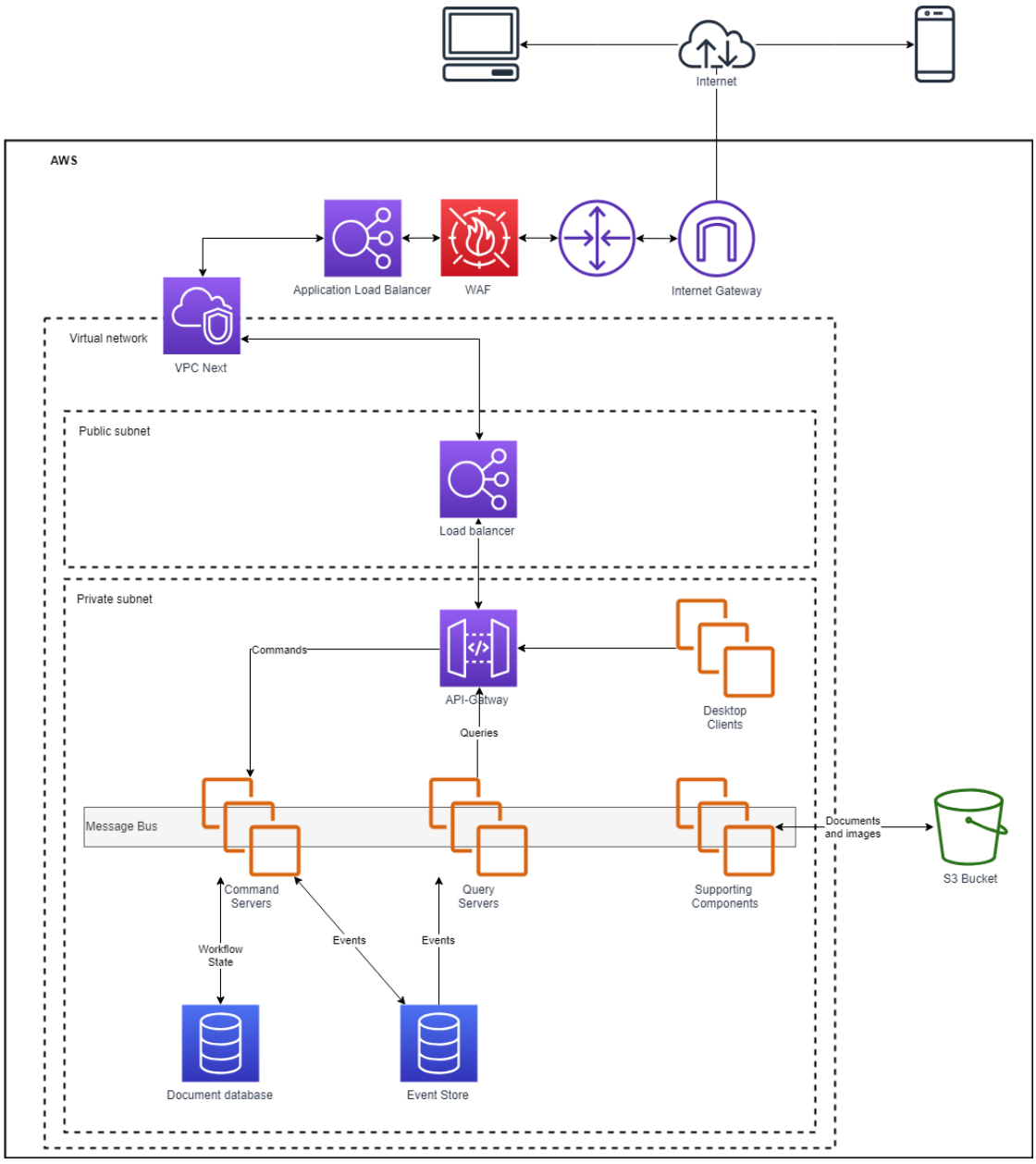


Image 3.5.1 – Interaction between software components

4 Secure software deployment

The goal of our deployment pipeline is to deploy stable software to our customer instances. Our continuous integration server publishes our software as uniform Docker images. This allows for a standardized pipeline for all our software, leveraging the strengths of industry standard components. All docker images are published to a secured Harbor registry, which has the added benefit of automated scanning for vulnerabilities. Whenever a new version of an image is detected a signal is sent in order to update customer instances. These are managed by Kubernetes in order to manage all components of our software, maintain high uptimes by automated error recovery, allowing rolling updates and by being able to act as a load balancer.

4.1 Software deployment process

The deployment process differs for Easy1 and E-InvoiceNext. Both are shown below, all future new product development will be based on the same process as Easy1.

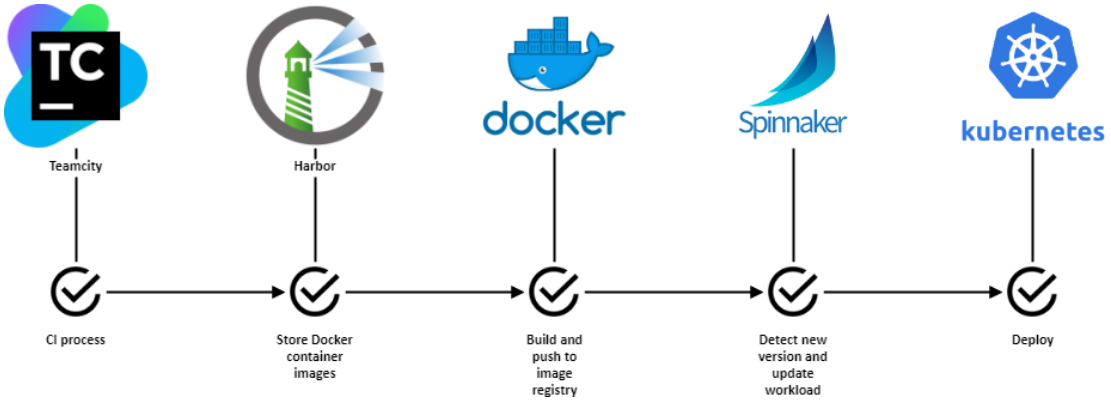


Image 5.1.1 – Easy1 deployment

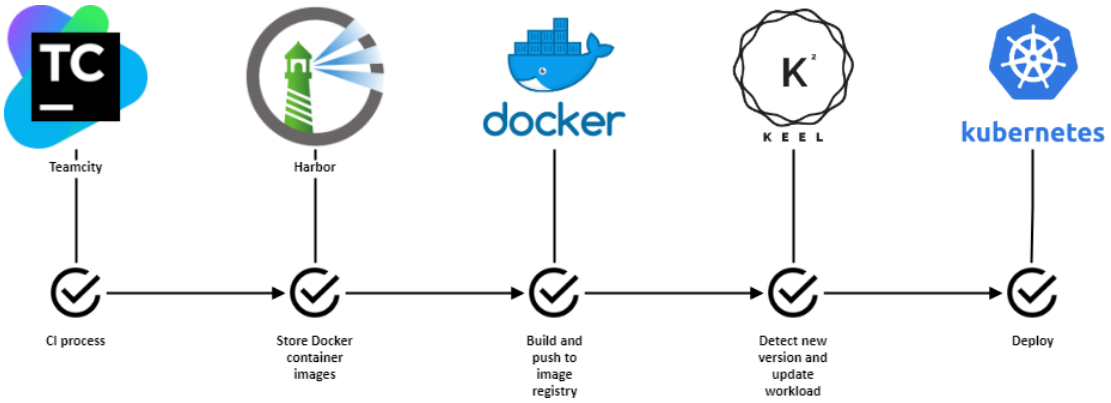


Image 5.1.2 – E-invoice Next deployment

4.2 Software deployment tools

Tool	Description
Docker	Docker is used to deliver software in packages called containers. Containers are isolated from one another and bundle their own software, libraries and configuration files.
Harbor	Harbor is an open source container image registry that secures images with role-based access control, scans images for vulnerabilities, and signs images as trusted. Harbor delivers compliance, performance, and interoperability to consistently and securely manage images.
Spinnaker	Spinnaker allows you to automate Kubernetes deployment updates and can be launched as a Kubernetes service in a dedicated namespace. Spinnaker helps to deploy Kubernetes service through labels, annotations, and charts. It's also used by Product Owners to stop the deployment process (when new functionality is deployed). Also the possibility exists to add tests which can block deployment.
Kubernetes	Kubernetes is an open-source container-orchestration system for automating application deployment, scaling, and management.
TeamCity	TeamCity is a build management and continuous integration server.

5 Testing Easy1 software

In order to ensure stable and user-friendly functionality we rigorously test every change to our Easy1 suite. We hereby adhere to industry standards and best practices. We will disclose some of the test types we apply, but before that, we will lay some theoretical groundwork.

5.1 The theory behind our vision on correct software testing

- Shift left testing vs shift right testing
- Agile testing quadrants

5.1.1 Shift left testing vs shift right testing

In the early years of the 2000's software testing in general was an afterthought in software development. It's placement right before delivery deadlines brought some understandable drawbacks. The term Shift left testing was coined by Larry Smith in 2001 [1] and was used to describe the placement of testing activities as soon as possible in the software lifecycle. Early defect detection results in lower correction costs [2] [3]. A large enabler for this shift is a high degree of test automation. Developing automated tests are a mandatory part of our development lifecycle. While shift left testing implies a solid position for testing during development, after deployment there is still a need to detect issues quickly in production, introducing shift right testing, sometime also called: 'Testing in production'. By introducing a dev-ops way of working we incorporated the managing and monitoring of the software after delivery. This allows for much faster detection of issues in production environment, thereby further shortening the feedback loop.

5.1.2 Agile testing quadrants

Many testing methodologies adhere to a risk-based approach [4] [5] [6]. In order to facilitate comprehensive risk coverage, we use the agile testing quadrants as refined by Janet Gregory and Lisa Crispin in their seminal book on software testing: "More Agile Testing" . They describe two viewpoints for software:

---

[1] Smith, L. (2001). Shift-Left Testing. *Dr. Dobb's Journal*, *26*, 56–62.
[2] Boehm, B. W. (1981). *Software Engineering Economics*. USA: Prentice-Hall.
[3] McConnell, S. (2004). *Code Complete*. USA: Microsoft Press.
[4] van der Aalst, L., Davis, C., & van der Aalst, L. (2012). *TMap NEXT in scrum: effectief testen in Agile projecten*. Netherlands: Kleine Uil, Uitgeverij.
[5] Bouman, E. (2008). *SmarTEST, Slim testen van Informatiesystemen*. Netherlands: Academic Service.
[6] Koomen, T., van der Aalst, L., Broekman, B., Vroon, M., & van der Aalst, L. (2013). *TMap Next: voor resultaatgericht testen*. Netherlands: Kleine Uil, Uitgeverij.

a. Technology facing VS business facing
b. Guiding development VS critiquing the product

When placed on opposite axes the following matrix with four specific quadrants arises:

- Q1 Technology facing and guiding development. Software obviously has a technical quality which needs to be addressed early in development by developers. Testing at this level is used to proof correctness of small pieces of code (units), integration of units and adherence to coding standards.

- Q2 Business facing and guiding development. Software is designed to serve a specific purpose. The quality of these requirements should be proven as early as possible, even before the coding process starts. This can be done through validation by customers and domain experts.

- Q3 Business facing and critiquing the product. Once a working piece of software is delivered, its functionality and usefulness can be actively measured. This can be done manually for usability aspects and automated for repetitive tasks.

- Q4 Technology facing and critiquing the product. There are other quality aspects of software which do not specifically pertain to documented requirements, like security and performance. These aspects are best suited to test manually with the assistance of specialized tools.

We employ multiple test types per quadrant, thereby combining their respective strong points to a chain of complementary tests.

5.2 Applied test types

Having expanded on two viewpoints demonstrating the necessity for multiple complementary test types, we can make a framework for describing our testing strategy.

The following overview references the shift left and right of testing by mentioning the development phase in which it is executed, demonstrating full coverage hereof. Additionally, the coverage of the testing quadrants is made clear by referencing them per test type. Please note that this overview is meant to give a broad overview of our test strategy, but a complete and detailed description of every test activity.

Test type	Quadrant	Development phase	Description
Unit test	Q1	Development	We aim to cover as much units of code as possible in order to automatically detect errors as soon as possible.
Specflow test	Q1	Development	We write user stories to describe the desired functionality. We use the tool Specflow to convert those user stories to automated tests. This allows for an up to date overview of implemented features and their correctness.
Prototyping	Q2	Development	One of the first things we create are prototypes of the software to be developed. These are used to elicit feedback from customers regarding functionality and usability. Usually, multiple refinements are done before the actual build starts.
Integration test	Q3	Acceptance	For every release of our software we execute a fully automated regression test of multiple flows in our software. User actions are simulated in the user interface in a production like environment.
Acceptance test	Q3	Acceptance	Our product owners review the major changes for usability and conformance to requirements.
Pentest	Q4	Production	We use checklists to check and improve specific security attributes of our software and infrastructure. Ad hoc independent penetration tests, pentests for short, are executed.
Monitoring	Q4	Production	A multitude of monitoring tools are in use in our production environment.

	More about this can be found in paragraph: 8.2.

Table 6.2.1 – Test types per quadrant and phase

5.3 Test follow-up

Having good test coverage is one thing, but without follow up they are pointless. Every automated test is implemented as a quality gate in our build pipeline, meaning when a test fails, the software can't proceed to the Next stage and can't be rolled out to production. Every test has extensive reporting in order to quickly identify root causes.

6 Platform architecture

The Easy Systems cloud computing environment (or Easy Systems Cloud) on Amazon Web Services (AWS). The Easy Systems Cloud consists of multiple platforms. These platforms are on the same AWS account and are divided by VPC.

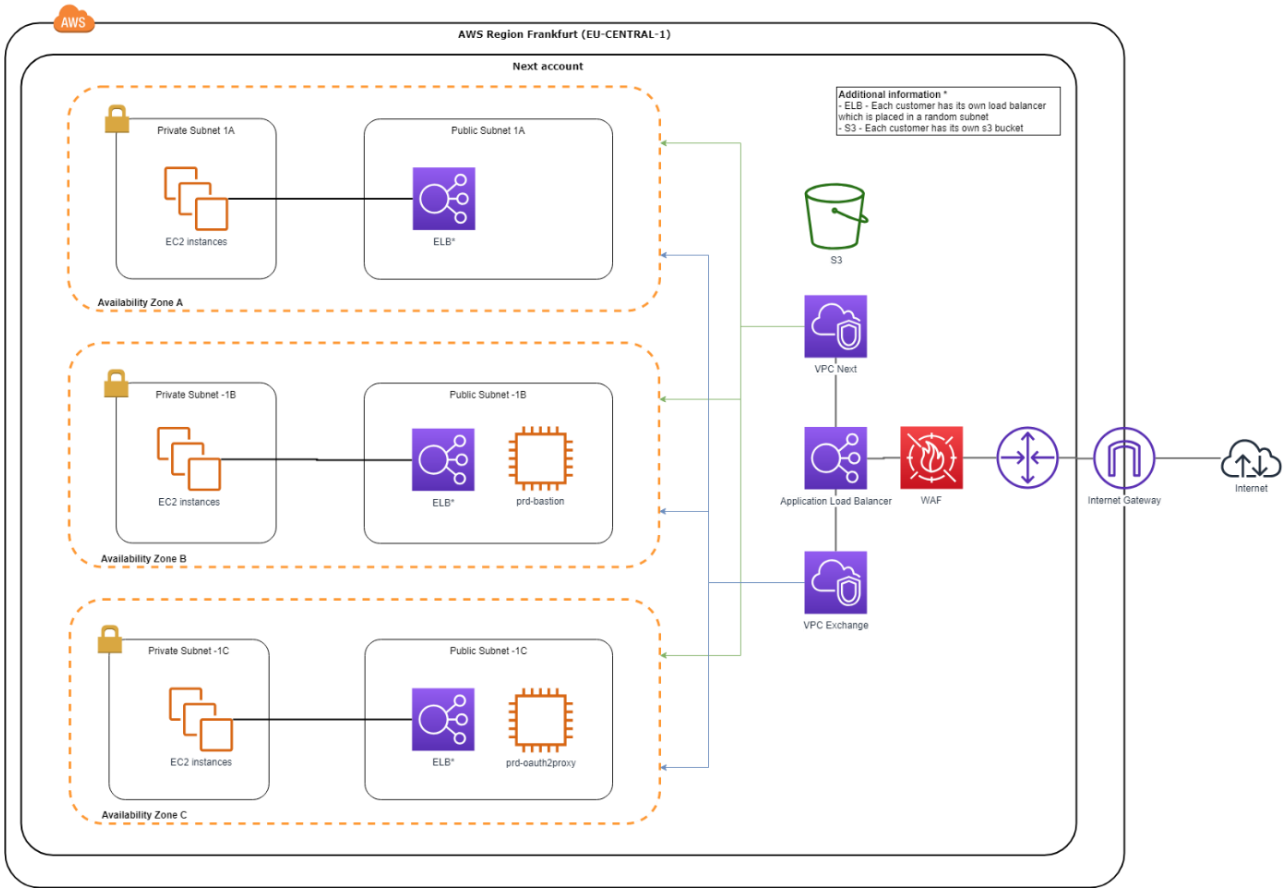


Image 7.1 – Platform architecture

6.1 Components within the platform architecture

The AWS platform contains several components which are used in the infrastructure. The relevant components are described below.

6.1.1 Kubernetes

Kubernetes is used for the orchestration of containers. These containers contain the Easy Systems products. Kubernetes takes care of deployment and scaling of the containers.

6.1.2 Private/public Subnets

A subnet is an IP address range. It is used for dividing a network into two or more smaller networks. It increases routing efficiency and enhances the security of the network. The customer instance is assigned with a private subnet only. Because it needs to be accessible from the internet a Load Balancer is used to route the traffic to the customer instance (through a specific port and encrypted by an SSL certificate).

6.1.3 Internet Gateway

The Internet Gateway is used to route communication from instances to the internet, and vice versa. For the instances, the outbound internet traffic is routed through the Internet Gateway and all inbound internet traffic is blocked. Outgoing traffic is only allowed if the connection is initiated from the server and uses secured protocols (such as SFTP, HTTPS, etc.).

6.1.4 Security Groups

Configuration rules for inbound and outbound traffic (allowed communication protocols and ports) is realized within AWS security groups. Each server within the AWS infrastructure is assigned to at least one security group. Security groups can only be changed by AWS admin user accounts.

6.1.5 Regions

The Easy Systems cloud infrastructure is hosted in Frankfurt, Germany (region eu-central-1). The backups are replicated to Paris, France (region eu-west-3). SMTP services are hosted in Ireland (region eu-west-1).

6.2 Environments

The Easy Systems AWS infrastructure consists of 2 environments: a development and a production environment, these 2 environments are used to ensure a structured roll-out of software updates, upgrades and/or patches.

Development environment

The development environment is used for developing new functionalities, but is also the first environment where new software versions are rolled out. This could either mean new software versions but also new Windows versions / updates or, for example, a new DBMS version. Once we have determined a successful roll out on the development environment, the software will be rolled out to the test environment.

The development environment is not available for the customer and is only meant for internal use.

Production environment

The production environment is the live environment that is used by the customer. New software will only be rolled out after all technical and functional tests are successfully completed. The go live date will be communicated well up front so the customer is fully aware and prepared. In most cases the roll out in the production environment will take place outside business hours 08:00 – 17:30 CET/CEST and mostly during the weekends (dependent of the impact of the change). In any case: the go live date is always determined together with the customer. This way, the business is not affected by any downtime.

Roll out scheme

The rollout procedure for the environments is showed in the below schema:

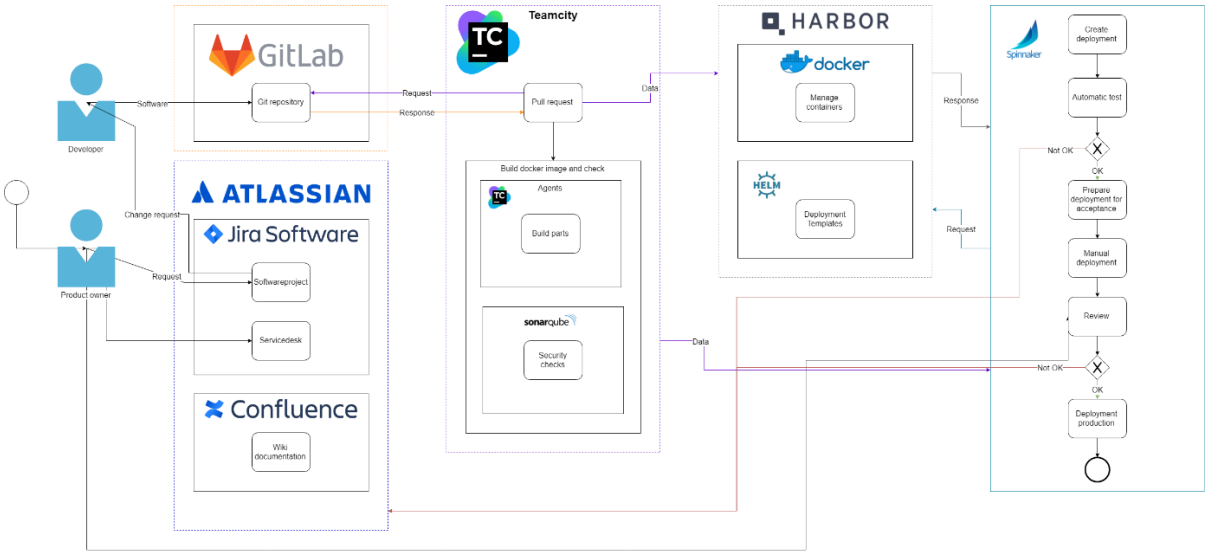


Image 7.2.1 – roll out scheme

6.3 Integrations with external systems

To integrate with external systems a separate structure is setup which is called the Easy Systems Gateway Service. The Easy Systems Gateway Service are several components which make it possible to upload a file which delivers standardized output to the Easy Systems products.

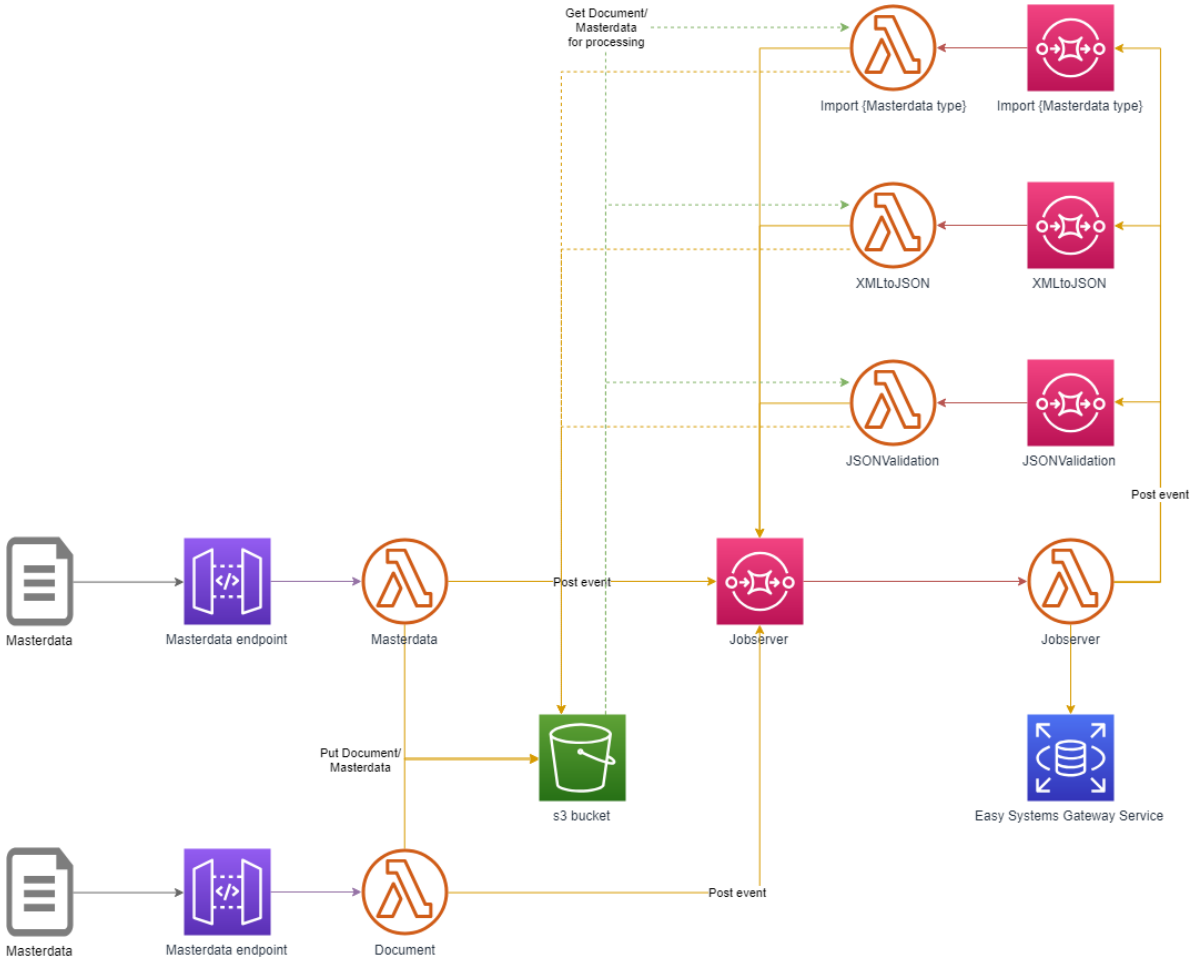


Image 7.3.1 – Integrations

7 Cloud services

7.1 Organization

Cloud Operations is the responsibility of a dedicated Cloud team. The Cloud team is responsible for setting up, maintaining, monitoring and updating the SaaS platform and its applications. The team is working closely with customer support, delivery and development team(s) to ensure reliability and professionalism of the hosted platform.
Access to the cloud environment is only allowed to persons that are qualified and assigned to Cloud tasks. Access is only provided with personal accounts, all actions/changes in the cloud environment are being logged for traceability and auditability.

7.2 Monitoring

Monitoring is applied at several layers of the Cloud environment. In het following paragraphs is shown on which layers it is applied and how this is done.

7.2.1 Application/software monitoring

The Easy1 application is monitored on different aspects, to ensure business continuity, performance and to be able to trouble shoot issues. These different aspects are described below:

- Application service
  Applications are monitored on running state. If a service is not running it will be flagged by our monitoring software, according measures will be taken.
- Document processing
- Import/Output folders

7.2.2 Hardware/server monitoring

For each server within the Easy Systems SaaS environment there is hardware / server monitoring in place to ensure availability and an acceptable performance. The different counters are described below:

- CPU
  The CPU load of the server instance is monitored. An automatic alarm will be pushed by our monitoring software if thresholds are exceeded for a configured period of time.
- Memory
  The physical memory usage of the server instance is monitored. An automatic alarm

will be pushed by our monitoring software if thresholds are exceeded for a configured period of time.

- Disk space
  The disk space is monitored for all server disk drives. If the available disk space drops below the threshold (low disk space), an automatic alarm will be pushed by our monitoring software
- Network
  All incoming and outgoing network traffic is monitored per server. An automatic alarm will be pushed by our monitoring software if thresholds are exceeded for a configured period of time.

7.2.3 AWS Platform monitoring

7.2.3.1 AWS Web application Firewall (WAF)

AWS WAF is a web application firewall that helps protect your web applications or APIs against common web exploits that may affect availability, compromise security, or consume excessive resources. AWS WAF gives you control over how traffic reaches your applications by enabling you to create security rules that block common attack patterns, such as SQL injection or cross-site scripting, and rules that filter out specific traffic patterns you define. You can get started quickly using Managed Rules for AWS WAF, a pre-configured set of rules managed by AWS or AWS Marketplace Sellers. The Managed Rules for WAF address issues like the OWASP Top 10 security risks. These rules are regularly updated as new issues emerge. AWS WAF includes a full-featured API that you can use to automate the creation, deployment, and maintenance of security rules.

With AWS WAF, you pay only for what you use. The pricing is based on how many rules you deploy and how many web requests your application receives. There are no upfront commitments.

You can deploy AWS WAF on Amazon CloudFront as part of your CDN solution, the Application Load Balancer that fronts your web servers or origin servers running on EC2, or Amazon API Gateway for your APIs.

Source: https://aws.amazon.com/waf/

7.2.3.2 AWS SECURITY HUB

AWS Security Hub gives you a comprehensive view of your high-priority security alerts and compliance status across AWS accounts. There are a range of powerful security tools at your disposal, from firewalls and endpoint protection to vulnerability and compliance scanners. But oftentimes this leaves your team switching back-and-forth between these tools to deal with hundreds, and sometimes thousands, of security alerts every day. With Security Hub, you now

have a single place that aggregates, organizes, and prioritizes your security alerts, or findings, from multiple AWS services, such as Amazon GuardDuty, Amazon Inspector, Amazon Macie, AWS Identity and Access Management (IAM) Access Analyzer, and AWS Firewall Manager, as well as from AWS Partner solutions. AWS Security Hub continuously monitors your environment using automated compliance checks based on the AWS best practices and industry standards your organization follows. You can also take action on these security and compliance findings by investigating them in Amazon Detective or by using Amazon CloudWatch Event rules to send the findings to ticketing, chat, Security Information and Event Management (SIEM), Security Orchestration Automation and Response (SOAR), and incident management tools or to custom remediation playbooks. Get started with AWS Security Hub in just a few clicks in the Management Console and once enabled, Security Hub will begin aggregating and prioritizing findings and conducting compliance checks.
Source: https://aws.amazon.com/security-hub/

7.2.3.3 AMAZON INSPECTOR

Amazon Inspector is an automated security assessment service that helps improve the security and compliance of applications deployed on AWS. Amazon Inspector automatically assesses applications for exposure, vulnerabilities, and deviations from best practices. After performing an assessment, Amazon Inspector produces a detailed list of security findings prioritized by level of severity. These findings can be reviewed directly or as part of detailed assessment reports which are available via the Amazon Inspector console or API. Amazon Inspector security assessments help you check for unintended network accessibility of your Amazon EC2 instances and for vulnerabilities on those EC2 instances. Amazon Inspector assessments are offered to you as pre-defined rules packages mapped to common security best practices and vulnerability definitions. Examples of built-in rules include checking for access to your EC2 instances from the internet, remote root login being enabled, or vulnerable software versions installed. These rules are regularly updated by AWS security researchers.
Source: https://aws.amazon.com/inspector/

7.2.3.4 AMAZON GUARDDUTY

Amazon GuardDuty is a threat detection service that continuously monitors for malicious activity and unauthorized behavior to protect your AWS accounts and workloads. With the cloud, the collection and aggregation of account and network activities is simplified, but it can be time consuming for security teams to continuously analyze event log data for potential threats. With GuardDuty, you now have an intelligent and cost-effective option for continuous threat detection in the AWS Cloud. The service uses machine learning, anomaly detection, and integrated threat intelligence to identify and prioritize potential threats. GuardDuty analyzes tens of billions of events across multiple AWS data sources, such as AWS

CloudTrail, Amazon VPC Flow Logs, and DNS logs. With a few clicks in the AWS Management Console, GuardDuty can be enabled with no software or hardware to deploy or maintain. By integrating with AWS CloudWatch Events, GuardDuty alerts are actionable, easy to aggregate across multiple accounts, and straightforward to push into existing event management and workflow systems.

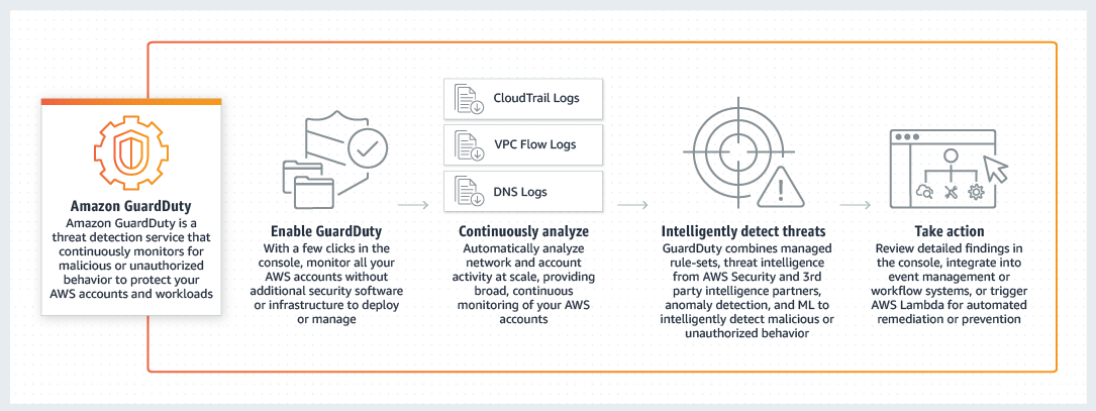


Image 8.2.2.1 – GuardDuty, how it works

Source: https://aws.amazon.com/guardduty/

7.2.3.5 *AWS IAM Access Analyzer*

AWS IAM Access Analyzer helps you identify the resources in your account, such as Amazon S3 buckets or IAM roles, that are shared with an external entity. This lets you identify unintended access to your resources and data, which is a security risk. Access Analyzer identifies resources that are shared with external principals by using logic-based reasoning to analyze the resource-based policies in your AWS environment. For each instance of a resource that is shared outside of your account, Access Analyzer generates a finding. Findings include information about the access and the external principal that it is granted to. You can review findings to determine whether the access is intended and safe, or the access is unintended and a security risk.

Source: https://docs.aws.amazon.com/IAM/latest/UserGuide/what-is-access-analyzer.html

7.2.4 Monitoring tools

Several tools are used to monitor the Easy Systems cloud environment. Beneath these tools are described.

7.2.4.1 AWS Security Hub gives a view of your high-priority security alerts and compliance status across AWS accounts. Within the Hub there are firewalls and endpoint protection to vulnerability and compliance scanners.

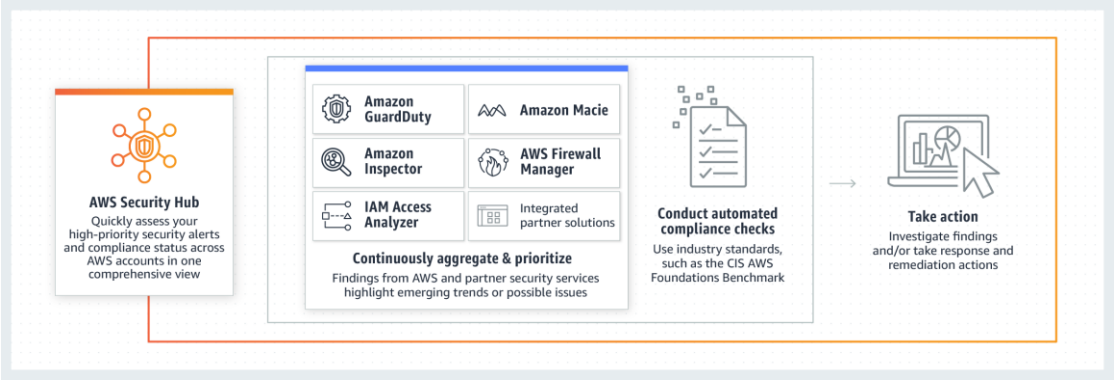


Image 8.2.4.1.1 – Security Hub, how it works

Servers and networks are constantly monitored for security issues. Alarms will be triggered through AWS GuardDuty and AWS Inspector which associates and analyses events. These alarms will be directly forwarded to the Cloud team so they can act swiftly and accordingly.

Alarms are sent to the Cloud team through different notification channels, such as email and Microsoft Teams. Outside business hours, the alarms are also sent to mobile devices.

7.2.4.2 Prometheus and Grafana

Prometheus is an open-source monitoring which is used to retrieve information from the running containers. This information is retrieve by Grafana and put into some dashboards. This is available both in the Production and Development environment. In production it's used to monitor the customer environment. In development it's used to improve the development of the Easy Systems products.

7.2.4.3 Kibana

Kibana is used for viewing the centralized logging. In case of a problem it's possible to search through a logfile using several parameters (such as namespace, response code, et cetera).

7.3 Upgrades/updates

At Easy Systems there are two types of updates/upgrades which are described in the following paragraphs.

7.3.1 Update of our software

This chapter shows the procedures regarding Easy Systems software updates on the Easy Systems Cloud. Reasons to update the software are:

- Improved security;
- New functionality;
- Software update needed by third-party software.

The following paragraphs show a description of the procedures per platform.

7.3.1.1 *E-invoicing NEXT*

Updates on the E-invoicing NEXT (or Easy Exchange) platform are deployed in accordance with the Product Owner. The Product Owner is triggered by a deployment from Development (in the AWS Development environment). After approval the update will be deployed to the AWS Production environment (running instances for Partners and Production)

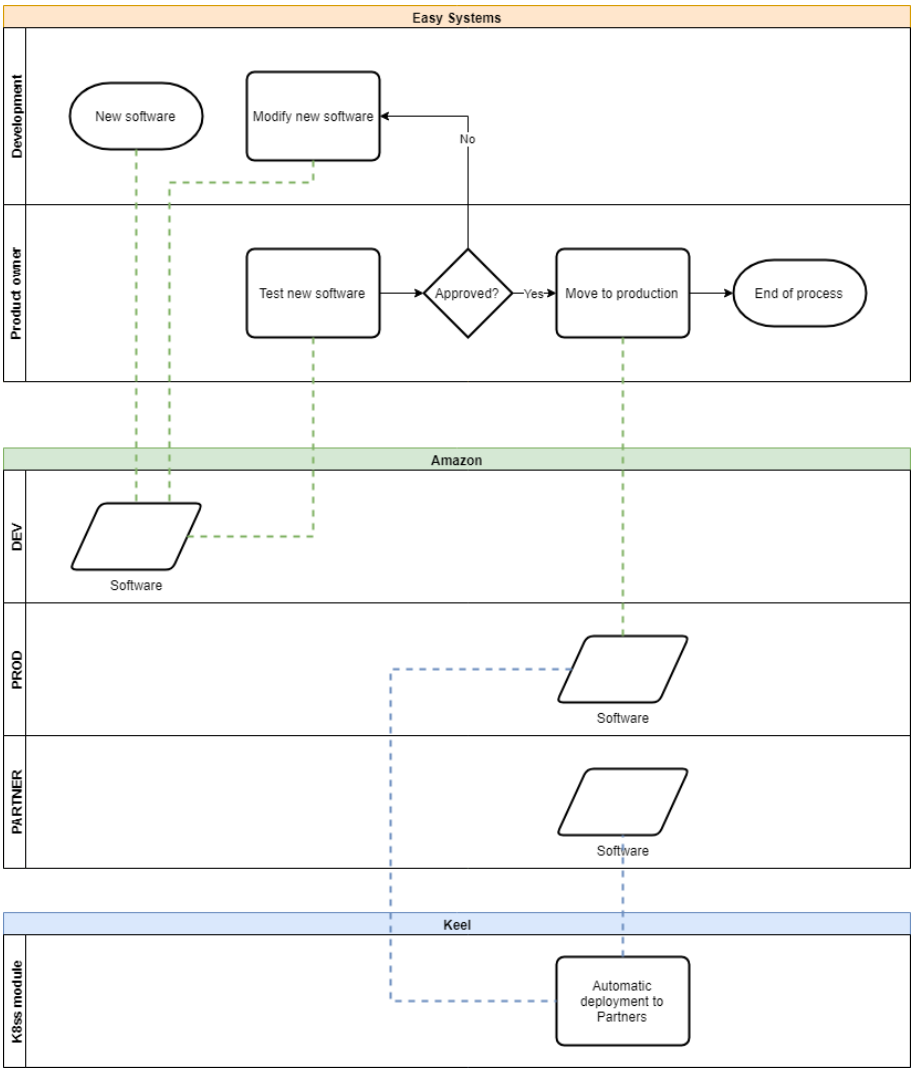


Image 8.3.2.1.1 – Update Easy Systems Easy Exchange software process

7.3.1.2 *Easy1*

Updates on the Easy1 platform are executed by the Product Owner. The Product Owner is triggered by a deployment from Development (in the DEV environment). After approval the update will be moved automatically to production (by deployment application Spinnaker). This movement happens in two steps to limit the impact of changes.

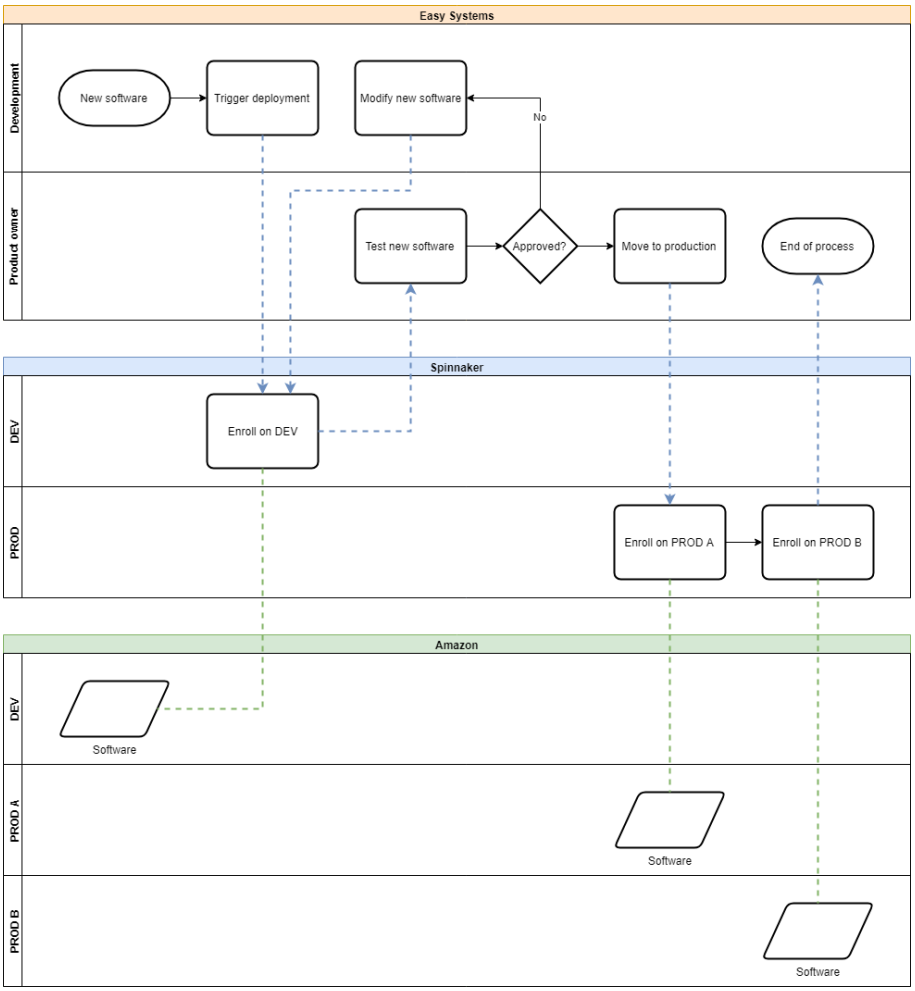


Image 8.3.2.2.1 – Update Easy Systems Easy1 software process

7.3.1.3 *CurrentGen*

Updates on the CurrentGen platform are executed by the Cloud Engineer. When a new version of the software is available it can be enrolled using the following components:

- **MySql database**

  The MySql database beholds a list of customers and software versions. By changing the software version in this database it's possible to update software.

- **REST service**

  The REST service starts the Spinnaker pipeline, which retrieves the software versions from the MySql database and enrolls it in the Customer namespace.

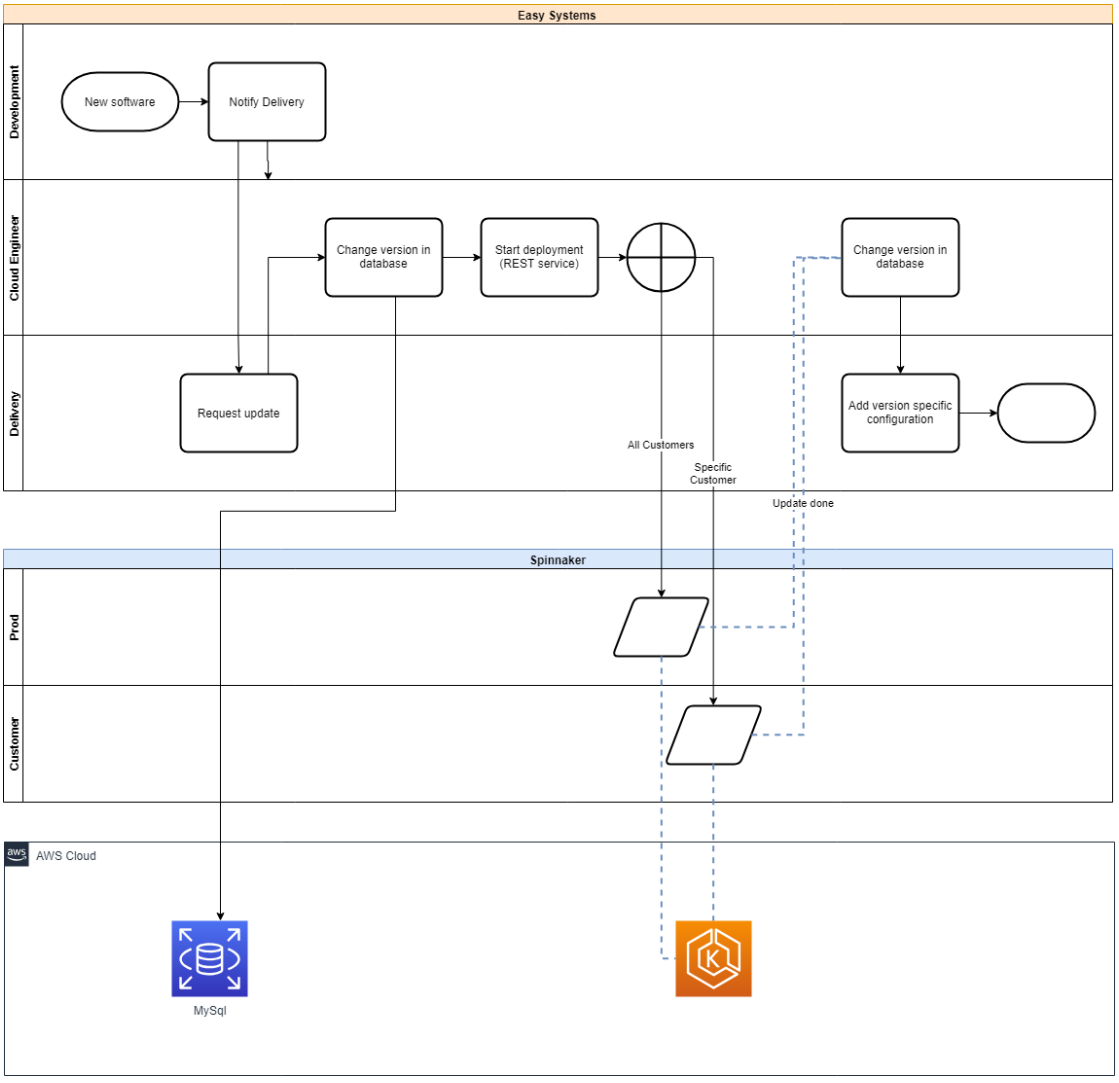


Image 8.3.2.3.1 – Update Easy Systems CurrentGen software process

7.3.2 Update third-party software

This chapter shows the procedures which are used for updating third-party software at the Easy Systems Cloud. Software updates are enrolled with the following reasons:

- Improved security;
- Improved functionality;
- Software is no longer maintained by supplier;
- Software update is mandatory because of Easy Systems software;
- Software update is mandatory because of other third-party software.

The actor in this process is always the Cloud Engineer. The Cloud Engineers keep track of new software by using Ninite Pro software and checking the supplier websites for new versions regularly. To keep the risk of updates as small as possible, updates are installed in the test / development environment before they are deployed to production.

The procedure shown below is equal for all platforms.

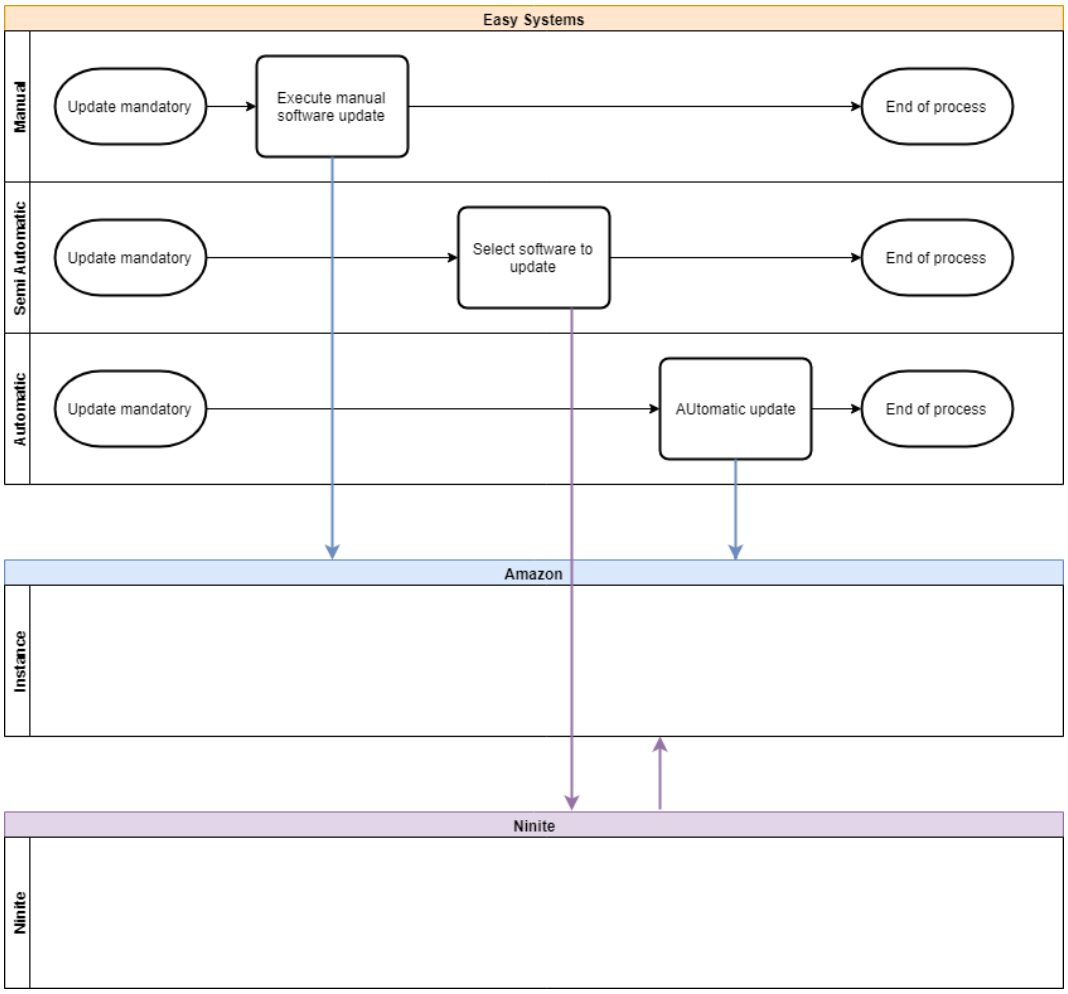


Image 8.3.1.1 – Update third-party software process

7.4 Maintenance

Maintenance is performed on application, database and operating system level.

7.4.1 Application maintenance

The Easy1 applications are maintained on a daily basis. Log files will be cleaned or archived to ensure a stable performance after a period of 1 month.

7.4.2 Database maintenance

Database maintenance is performed on a weekly basis, outside working hours. The database maintenance window contains the following steps:

- Reorganize and rebuild indexes: correct problems regarding non-optimal indexing.
- Database integrity check: prevent corruption of data.
- Shrink database: to avoid critical situations due to the filling of the space allocated.

7.4.3 Operating system maintenance

The Easy1 platform is based on Kubernetes nodes with the master hosted by AWS. The underlying EKS worker nodes are enrolled by Easy Systems based on the last image (provided by AWS). The version of the used is checked and maintained automatically.
Some of the Easy Systems applications are installed on virtual machines, based on Microsoft Windows Images (AMI). When deploying a new virtual machine, the most recent version of the AMI is used. Each night a process checks if new operating system updates are available. These are installed if they are older than 7 days.

7.5 Availability

7.5.1 Cloud/SaaS availability

Availability of the service means that the referenced environment (production or test environment) is operational and accessible. In particular that means a correct functioning of all hardware, software and connectivity components. There is a service level agreement in place for the Easy Systems SaaS environment. Please refer to the Easy Systems Cloud SaaS SLA document for more information.

8 Disaster recovery

This chapter in only applicable for the Easy Systems SaaS environment. For disaster recovery there are certain procedures in place in the event of a human error, data corruption, or failing server instance. Easy Systems' strategy for Disaster Recovery is implemented with the use of a back-up process (see paragraph “backup model” for a description). This process allows the Easy Systems Cloud team to recover system operation, with the defined RPO (see paragraph 6.1.2) and RTO (see paragraph 6.1.2).

8.1 Procedure

In case of a disruption/disaster, a procedure is in place to determine the impact and to resolve the disruption within the agreed SLA.

In case of service disruption within business hours (08:30 – 17:00 CET/CEST), the Cloud team will be notified directly. Notifications will be sent out automatically throughout multiple channels. Escalation levels are in place: first the Cloud team lead will be contacted and if needed also the delivery manager is contacted. The customer will be notified directly if the disruption affects the business.

The following events will take place in case of a disruption/disaster:

8.1.1 Impact analysis

The (business)impact analysis contains the following:

Data loss analysis

Investigation will be performed to determine if there is any data loss due to the disruption. In case of data loss, an overview will be created of the data that is lost and has to be recovered.

Data recovery plan

After the analysis of the data loss, a plan is made on how to recover the data. This could be by reprocessing data or restore a backup.

Expected recovery time

After analysis of data loss and verification of the recovery plan, an estimated recovery time can be set (also according to the agreed SLA). The recovery time will be communicated to the customer in case the business was affected by the disruption.

Escalation

The outcome of the impact analysis will determine if the disruption will be escalated to the Cloud team lead and/or the delivery manager. Escalation can be used for allocating more resources or to determine the communication plan to the customer.

Communication with customer

In case the business process is affected by the disruption, the customer will be notified by phone and/or email. The Easy Systems service center will be involved to take care of issue management and to inform our customer contact.

8.1.2 Recovery Point Objective (RPO)

In case of a major malfunction/disruption, the RPO is set to 24 hours. That means that backups of the database and snapshots from instances will be restored with data that is not older than 24 hours.

In case of smaller disruptions and in case data can be recovered/restored without the need to use a full backup, the data will be restored manually. Any changes done will be reverted to the point in time the service was working correctly.

8.1.3 Recovery Time Objective (RTO)

In the event of an unrecoverable loss of a data storage device, full backups will be restored of the database and/or server instances. After the restore, checks are made to determine if the restore was successful. A business impact analysis will be made and the outcome will be communicated to the customer. All necessary steps to restore the data and the expected RTO will be included in the communication. The RTO goal is set to 48 hours. Updates on the expected recovery time will be sent to the customer every 4 hours.

Below an overview is provided of the RPO/RTO timeline in case of disaster recovery.

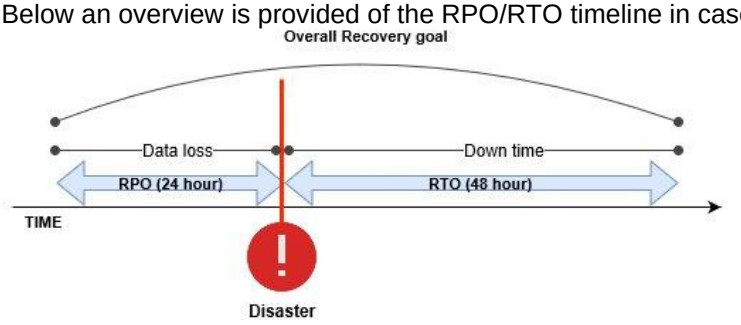


Image 9.1.3 – RPO & RTO

8.2 Backup model

In the Easy Systems Cloud platform, database backups and server snapshots are automatically made every 24 hours. The retention period for the backups is 1 month. That means that backups which are older than 1 month will be automatically removed.

All backup files and snapshots are encrypted with AES-256 encryption. The backup files and snapshots can only be accessed with an encryption key that is stored in the AWS administrator account.

8.2.1 Daily backup

Daily backups are made from the whole Kubernetes namespace. The Kubernetes configuration is saved to Amazon S3. The snapshots are saved as EBS volume snapshot (default AWS functionality). These backups are checked on weekly basis (as part of a checklist).

8.2.2 Point-in-time backup

At a regular time interval a point-in-time backup is made from the event store. This backup is saved to Amazon S3. These backups are checked on weekly basis (as part of a checklist).

8.2.3 Backup retention

The retention period for backups is set to 31 days. After every new successful backup, the oldest backup will be removed (taking the retention period into account).

8.2.4 File backups

Blobs (images, documents) are stored in the customer instance Amazon S3 bucket. The policy is to keep a copy of customer data in the AWS backup region Paris, France. This process is automated using an AWS Lambda function which is triggered by new added

objects. Objects are simply copied from the source (customer S3 bucket) to a destination (backup s3 bucket in the same region as the source bucket (Frankfurt, Germany)). The destination bucket has an active replication configuration which replicates the object async to the backup bucket.

The backup bucket is owned by a different AWS account. For security reasons the bucket owner will become the owner of the data after the replication was successful. This complete process from adding a media object to the customer bucket until it arrives in Paris finishes within max 2 seconds.

This solution automatically picks up new deployed customer instances.

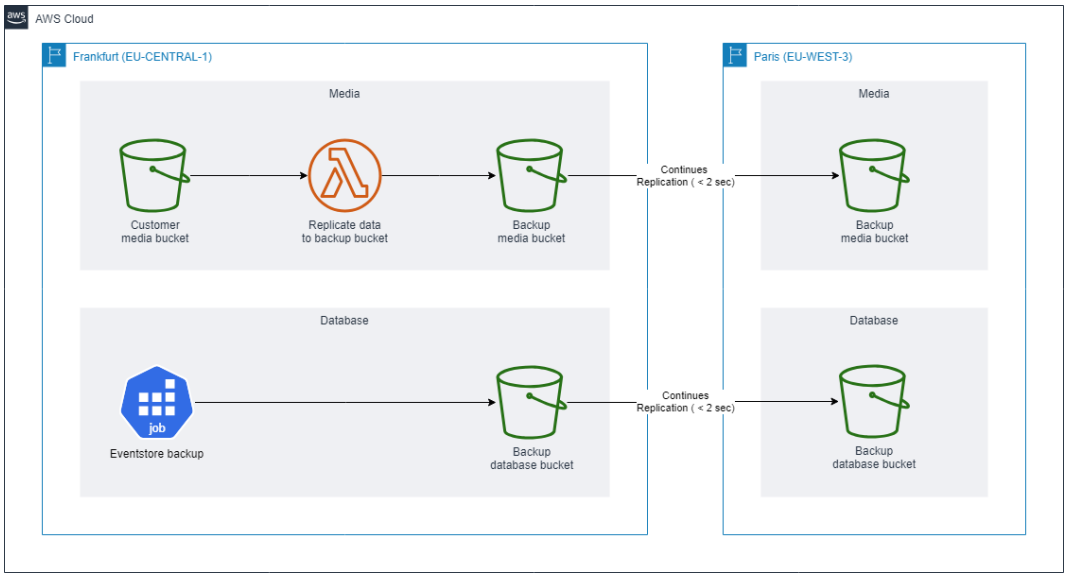


Image 9.2.4.1 – Backups

9 Security

Security of the Easy Systems SaaS service and infrastructure is ensured through the following principles.

9.1 Vulnerability management

The environment is protected against viruses, spam, malicious software and dangerous active content with the use of AWS WAF - Web Application Firewall, AWS GuardDuty, AWS Inspector and antivirus software. All emails, data transfers, downloads, uploads etc. are monitored. Automatic notification will be sent out to the Cloud team in case of a vulnerability. In case of infected files, the customer will be notified as well.

AWS GuardDuty is continuously monitoring the network activity and account behavior in the AWS infrastructure. All events are logged and analyzed and notifications will be forwarded to the Cloud team in case of a potential risk. The team will analyze the AWS flow logs and AWS CloudTrail logging and act accordingly if needed.

For security assessments on server level, AWS Inspector is used. This service is used to detect vulnerabilities and deviations. The “CIS Security Benchmark” is used in AWS inspector as best practice. AWS Inspector will automatically perform an assessment and forward events to the DevOps for further handling.

ESET Antivirus software is active on all server instances in the Easy Systems SaaS environment. This software is used to protect the environment against malicious software, ransomware, data leaks and botnets. A firewall is also setup in ESET to control the communication between the secure SaaS environment and the internet. Events will be automatically forwarded to the Cloud team.

9.2 Access control

Access to the Easy Systems SaaS environment and access to customer data is granted using the least-privilege principle.

Consultants are only allowed to access specific servers and customer data if really necessary and only for a certain time period (depending on the nature and duration of the work). All consultants are authenticated with their own named user accounts. Access is only granted after managerial approval.

Only Cloud team members have access to the AWS platform environment.

9.2.1 AWS Accounts

Only Cloud team members within Easy Systems are given administrator access to the AWS infrastructure / environment. Access is revoked if an employee no longer needs to access the environment. Two factor authentication is applicable when logging on to the AWS environment.

Some consultants and customer have access to the AWS infrastructure / environment. In this case they only get access to the part that is applicable for the project or customer instance.

9.2.2 AWS IAM Roles

Within AWS, user roles are defined. These are called IAM roles. The use of IAM roles allows segregation of duty for users that can access the AWS environment. Implementation consultants will be assigned to a role that provides access to only relevant sources for that project i.e. a consultant who will configure an instance will only get the role that is relevant for that instance, based on least-privilege.

More privileged accounts must first be approved by the Delivery manager/Cloud team lead before assigning them.

9.2.3 Server Accounts

A server account will provide access for consultants to specific server instances, for example a customer server. Access for consultants is configured by the Cloud team and allows them to perform specific implementation tasks. Access can be revoked at any time by the Cloud team.

9.2.4 Application User Accounts

Easy Systems application user accounts are named accounts for the consultants and customer, so they are able to log on to the application only for a specific customer instance. The user account are set-up by the initial consultant and are maintained by the customer. The user accounts can also be deleted at any time by the Cloud team.

9.2.5 Logging

All AWS login attempts and changes done in the AWS environment are logged in AWS CloudTrail. AWS CloudTrail will store all AWS events in a secured environment. This environment can only be accessed from a designated AWS account which is only available to the AWS system administrators in the Easy Systems Cloud team.

9.3 Contract termination

When a contract is terminated its important to remove the customer environment from the Easy Systems Cloud. Beneath the main process is shown, which contains a different sub process per platform. These sub processes are shown in the following paragraphs.

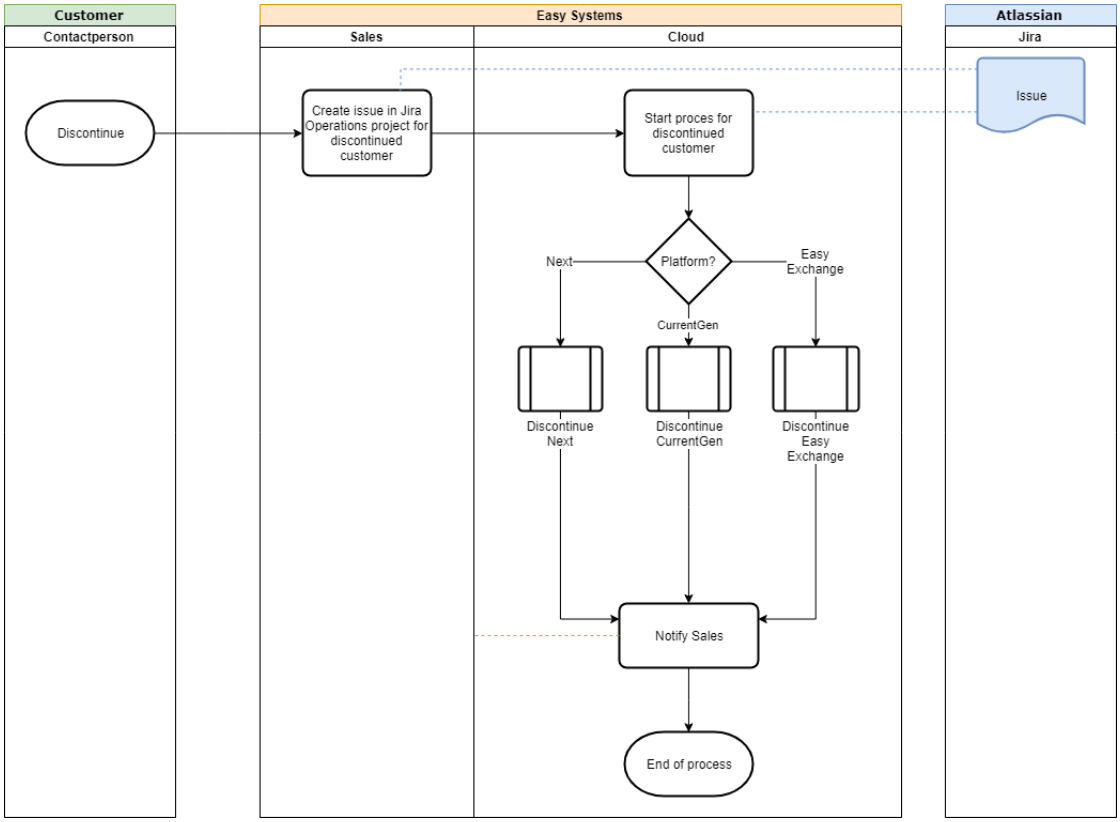


Image 10.3.1 – Contract termination

9.3.1 E-invoicing NEXT

When a customer doesn't want to continue to use the Easy Exchange services the following happens:

- The Easy Exchange customer Registration is deactivated;
- The registration at the PEPPOL Service Metadata Publisher is made undone;
- Processed documents will be removed, 3 months after termination;
- PEPPOL (and customer) specific logfiles will be removed, 3 months after termination.

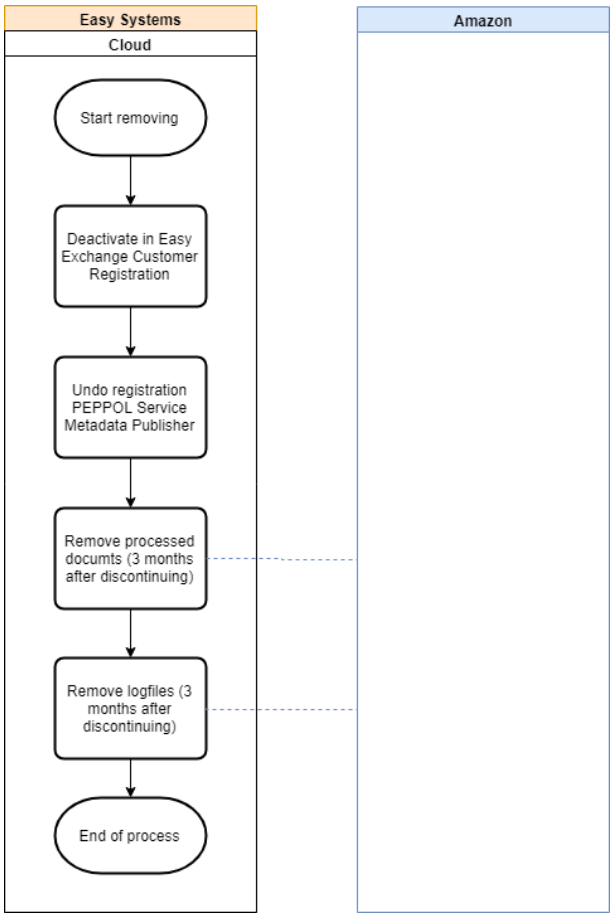


Image 10.3.1.1 – Contract termination E-invoicing NEXT

9.3.2 Easy1

When a customer doesn't want to continue to use the Next services the following happens:

- The tenant is removed;
- The EBS volumes are deleted;
- The backups are removed.

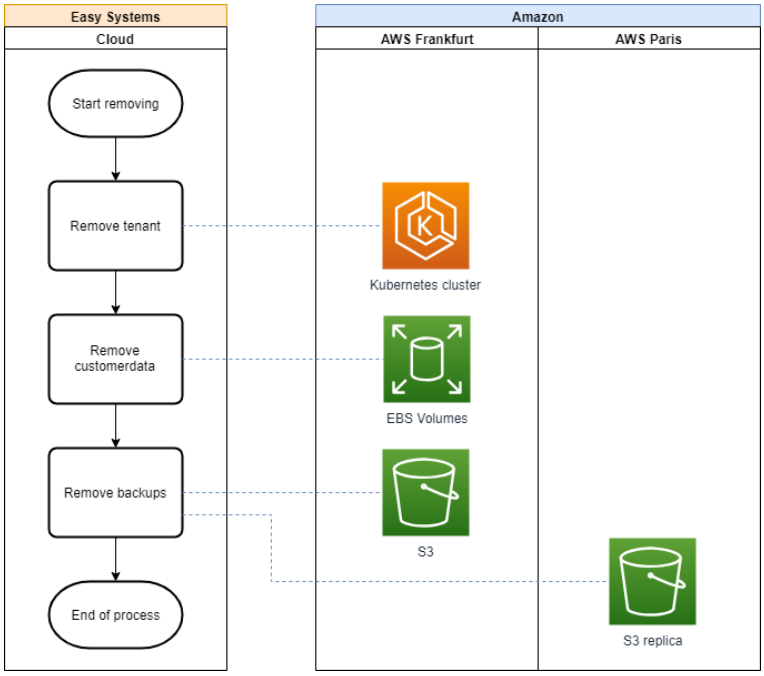


Image 10.3.2.1 – Contract termination Easy1

9.3.3 CurrentGen

When a customer doesn't want to continue to use the CurrentGen services the following happens:

- The Customer instance is removed;
- Backup data and share this with customer;
- The EBS volumes are deleted;
- The database is removed;

- The backups are removed.

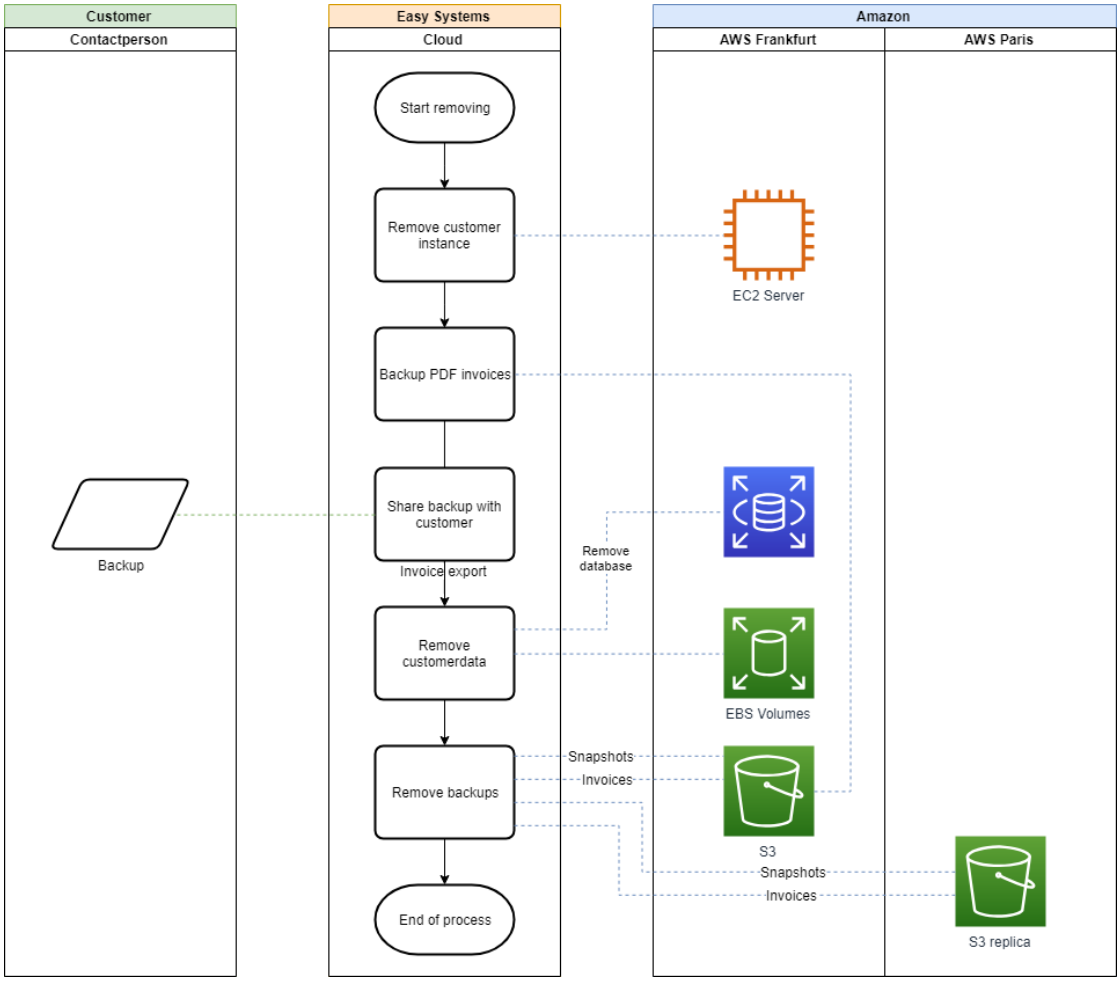


Image 10.3.3.1 – Contract termination CurrentGen

9.4 Data Protection (cryptographic, encryption, hashing)

Customer data is segregated on database and application level. All data is kept within the secured Easy Systems SaaS environment. Data at rest is encrypted with AES-256 encryption and cannot be accessed without the encryption key that is stored within the AWS root account. We use AWS Key Management Service (KMS) to manage, generate and rotate cryptographic keys.

https://aws.amazon.com/kms/

Easy Systems employees will be granted access to the encryption key alongside the AWS IAM role that is assigned.

Data in transit, initiated from the Easy Systems SaaS environment to the internet is encrypted using Transport Layer Security (TLS). We use TLS 1.2 which is a cryptographic protocol designed to provide communications security over a computer network. TLS 1.2 can secure all communications between servers and web browsers. The TLS protocol aims primarily to provide privacy and data integrity between two or more communicating computer applications.

https://aws.amazon.com/about-aws/whats-new/2017/02/elastic-load-balancing-support-for-tls-1-1-and-tls-1-2-pre-defined-security-policies/

Access to the encryption key management service is very limited. Only a small number of Easy Systems employees can access this.

In addition to the cloud infrastructure did we also taken security measures on our applications for example:

1. All our applications are only accessible through the https protocol this means that data up to our API gateway is encrypted by this standard protocol.
2. The passwords used within the applications are hashed. This means that they cannot be traced, not even by us.

Exceptions tailored environment

Our infrastructure is designed and construct to provide the best possible protection. To keep all data secure. We don't make exceptions to our infrastructure when it comes to security measures. There will be no tailormade solution for our customers.

9.5 Data classification (CIP Taskforce Baseline)

The Center for Information Security and Privacy Protection (Centrum Informatiebeveiliging en Privacybescherming) published a baseline for data classification. In this baseline they include a classification guideline. Documents and data about Expenses, Invoices and Contracts were judged on business and personal impact. The sensitivity is measured by integrity and confidentiality rating, conform the classification guideline.

Information types	Business process impact*	Integrity classification	Confidentiality classification
Expenses	4	2 – High	2 – Confidential
Invoices	3	2 – High	1 – Business confidential
Contracts	4	2 – High	2 - Confidential

**Business process impact rating on a 5-point scale*

We only deliver the highest possible setting on security to our documents and data.

Because the data and documents that are used in our software all scored 3 or 4 on business process impact and the Integrity classification is a minimum of 2 – high. The Confidentiality classification is 1 - business confidential or 2 – confidential we decided to set the highest possible setting on security to our documents. Only the people that are authorized in a specific process can see the documents and only the people that are authorized can adjust. This setting is set on processes the documents are set to confidential.

Background information about CIP

CIP was founded by the Dutch tax service, DUO, SVB and UWV and originates from the program Compacte Rijksdienst (2011-2012).

Voor de classificatie naar de inzichten **integriteit** en **vertrouwelijkheid** is de vertaling als volgt:

Bedrijfsproces impact	I-classificatie	V-classificatie
1	0 - Verwaarloosbaar	0 - Openbaar
2	1 – Beschermd	1 - Bedrijfsvertrouwelijk
3 + 4	2 – Hoog	2 - Vertrouwelijk
5	3 – Absoluut	3 - Geheim

Source: https://www.cip-overheid.nl/media/1166/bid-operationele-producten-bir-010-dataclassificatie-1_1.pdf

9.6 Incident management

Security and software issues and incidents are identified via multiple channels, from technical and functional monitoring to the Easy Systems customer service desk. Security incident handling follows a defined procedure and is the responsibility of the Easy Systems security officer. For more details about the Easy Systems customer service desk incident response times and issue management, please refer to the Easy Systems Cloud/SaaS SLA.

9.7 Risk and compliance

Easy Systems is ISO 9001, ISO/IEC 27001 and ISO/IEC 27017 certified. Easy Systems is following all relevant data protection guidelines and rules, applicable to our operations.

9.8 Data segregation

Customer data is isolated in separate ways per platform. Beneath the oversight per platform is shown.

9.8.1 E-invoicing NEXT

Easy Systems customers can have one or more entities to the service. It's possible to retrieve the status of sent and received documents per entity. The customer data is captured in a document-database. E-invoicing is a multi-tenant application, data segregation is managed in the application layer.

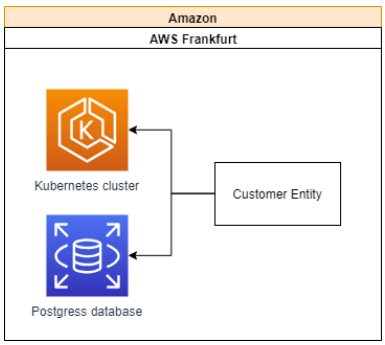


Image 10.8.8.1 – Data segregation E-invoicing NEXT

9.8.2 Easy1

Each customer has its own tenant on which the following measures took place:

- Each customer has its own admin portal to manage users;
- Each customer has its own tenant on a shared database;
- Each customer has its own tenant S3 bucket for attachments;
- The software is enrolled fully automatic through the Easy Systems CI/CD pipeline.

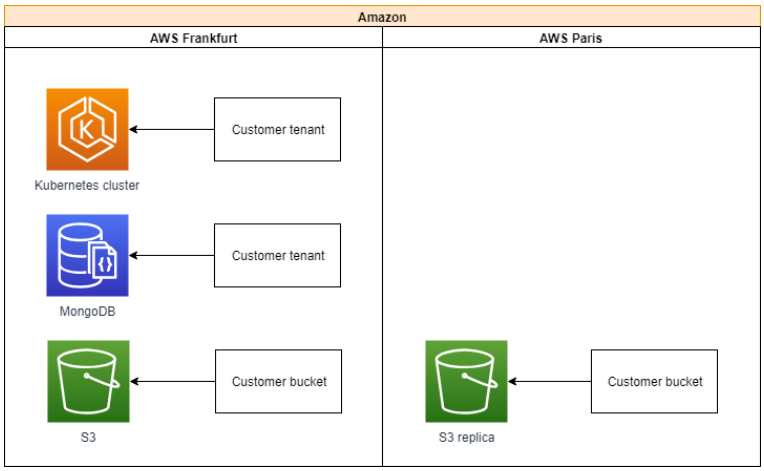


Image 10.8.2.1 – Data segregation Easy1

9.8.3 CurrentGen

Each customer has its own customer instance on which the following measures took place:

- Each customer has its own admin portal to manage users;
- Each customer has its own database and customer S3 bucket for attachments;
- Each customer has its own services/data-processors (so the application is multi-instance, not multi-tenant);
- The software is enrolled fully automatic through the Easy Systems CI/CD pipeline.

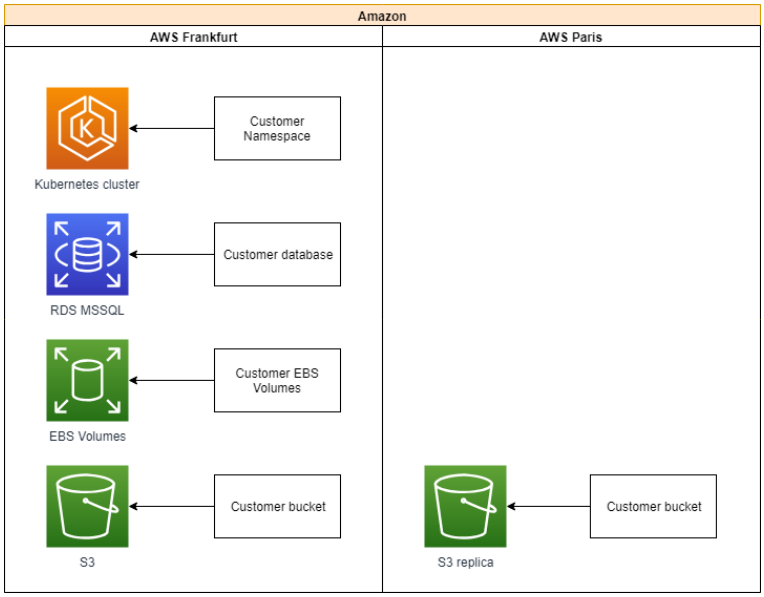


Image 10.8.3.1 – Data segregation CurrentGen

10 Audits initiated by customers

10.1 Additional independent audit

Customer have the right to request an additional independent audit. This audit can be carried on the technical and organizational security measures that Easy Systems takes to protect the information of it and its customers. The following conditions are attached to this audit:

- The audit takes place against the ISO27001 & ISO27017 standard;
- The audit is performed by an accredited auditor;
- The costs for the audit are for the client;
- All confidential business information remains within Easy Systems;
- All information about other customers remains within Easy Systems.
- The additional independent audit can only be requested once a year

Easy Systems accepts to resolve all identified shortcomings within a reasonable period of time.

11 Time Sync Service

11.1 Redundant Network Time Protocol Sync Service

The cloud infrastructure that we use offers a Time Sync Service “Amazon Time Sync Service” The Amazon Time Sync Service is a time synchronization service delivered over Network Time Protocol (NTP) which uses a fleet of redundant satellite-connected and atomic clocks in each region to deliver a highly accurate reference clock. The Amazon Time Sync Service automatically smooths any leap seconds that are added to UTC. This service is available in all public AWS regions to all instances running in a VPC.

The service is a consistent and accurate time reference is crucial for many server tasks and processes. Most system logs include a time stamp that we use to determine for example when problems occur and in what order the events take place.

The Amazon Time Sync Service is available through NTP at the 169.254.169.123 IP address for any instance running in a VPC. All our applications and underlying services and components use the time sync service.