Rapport de projet

La sécurité dans Azure

NONCLERCQ Basile
GUILLAUMIE Bilal
Promo 22
Filière : F5

Tuteur entreprise :
MAINAND Sébastien
Tuteur ISIMA :
TILMANT Christophe

21 Janvier 2021

Table des matières

Chapitre 1

Préambule

1.1 Introduction

Voici dans ce rapport les éléments propre à la réalisation du projet proposé par **Sébastien MAINAND**.

1.1.1 Description

Ce projet consiste à

> *Identifier les différents services de sécurité (key vault, SIEM, pare-feu, WAF, etc.) mis à disposition dans l'écosystème Azure, les étudier et proposer une architecture sécurisée les combinant.*

1.1.2 Objectifs

Les objectifs fixés sont les suivants :

1. Identifier les composants importants propre à la sécurité dans l'écosystème Azure.
2. Proposer une architecture sécurisé et ce via l'usage ou non de certains composants.
3. Implémenter cette architecture.
4. Rédiger un rapport explicatif des points cité précédemment.

Chapitre 2

Conduite de projet

Pour mener à bien le projet, la planification et l'organisation des tâches propres à la construction et réalisation de l'architecture ont été mise en avant via un diagramme de GANTT.
Voici donc la liste des tâches élaborées via le cahier des charges :

Tâches à réaliser :

1. Réflexion 3h
   - Lister et catégoriser l'ensembles des composants. 1h
   - Élaborer les principaux axes de réflexions. 2h
2. Recherche 66h
   - Lister et catégoriser l'ensembles des composants. 6h
   - Récupérer l'ensemble des détails concernant les composants/architectures. 40h
   - Synthétiser les concepts ainsi que les données les plus importantes. 20h
3. Réalisation 44h
   - Rédaction du support. 20h
   - Modification et rectification. 4h
   - Mise en place de l'infrastructure sur Azure. 20h
4. Gestion de projet 7h
   - Préparation à la soutenance de projet. 5h
   - Réalisation du cahier des charges. 2h

2.1 Diagramme de Gantt

2.1.1 Gantt prévisionnel

Le diagramme de GANTT à été élaboré afin de répartir les tâches en fonction de leurs priorités ainsi que de leurs relation dîtes de *début à début* ou de *fin à début.*
Ce diagramme fût réalisé le 14 novembre 2020.

Ce diagramme de Gantt ne possède pas de relation de *début à début.* Aucune tâches ne nécessite de commencer en même temps qu'une autre cependant il existe des relation de *fin à début.*
C'est notamment le cas pour la tâche **Récupérer l'ensemble des détails concernant les composants/architectures** qui se doit de commencer après la tâche **Lister et catégoriser l'ensemble des composants**
La tâche propre à la **Modification et rectification du rapport** se doit aussi de commencer après le processus de **rédaction du rapport**.

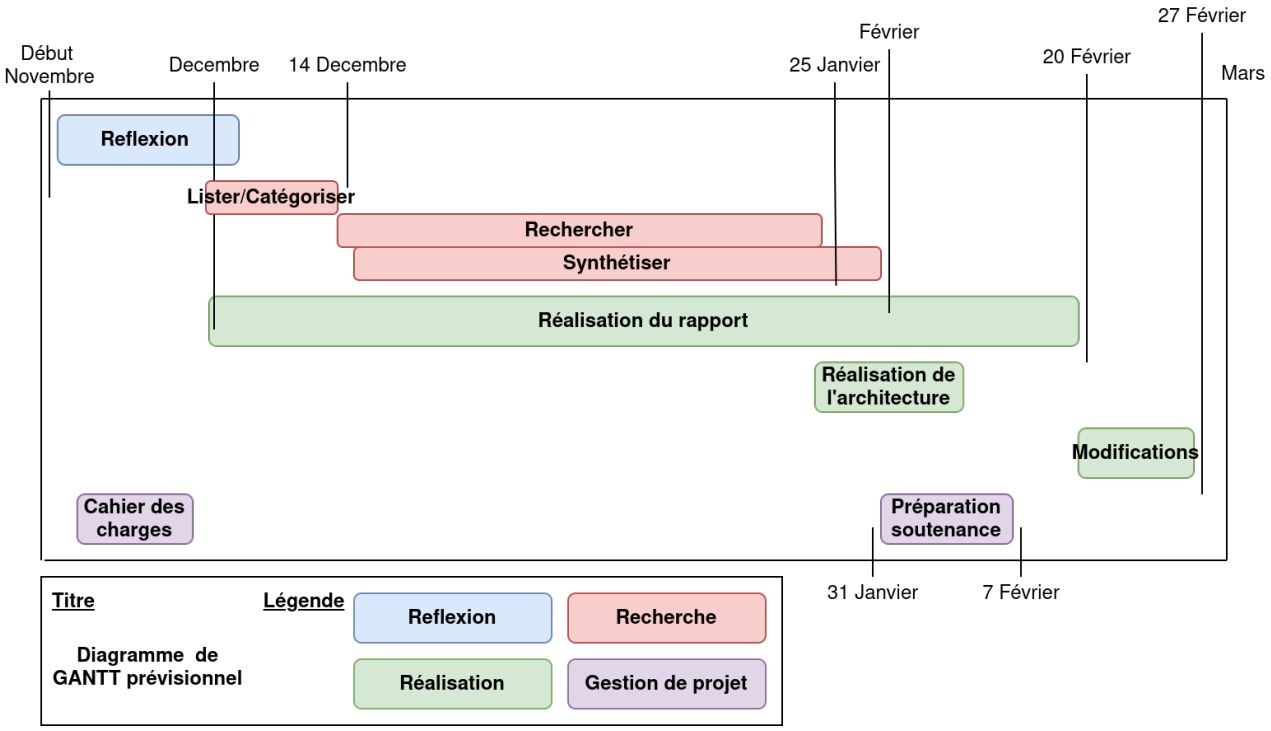


FIGURE 2.1 – Gantt prévisionnel

2.1.2 Gantt réel

Le GANTT réel à été quant à lui élaboré vis à vis des tâches effectué réellement lors de la réalisation du projet.

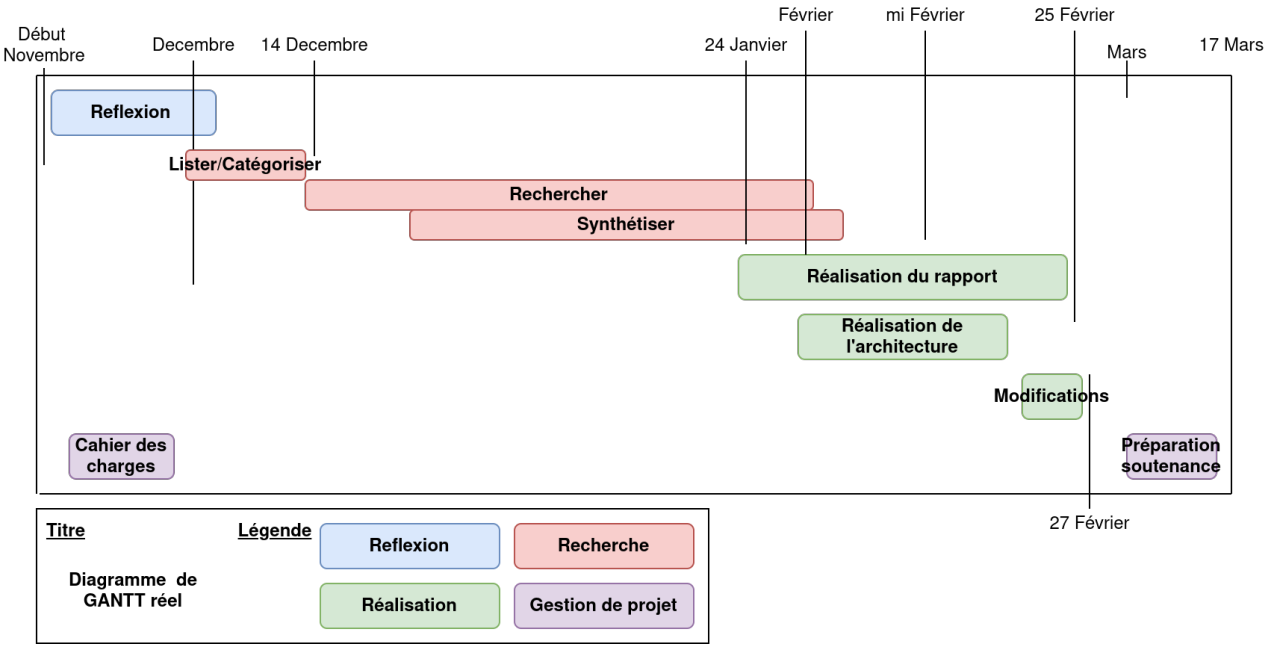


FIGURE 2.2 – Gantt réel

Les tâches ayant été impacté dans leur réalisation sont les suivantes :

1. Réflexion
   - Lister et catégoriser l'ensembles des composants.
   - Élaborer les principaux axes de réflexions.

2. Réalisation
   — Rédaction du support.
   — Modification et rectification.
   — Mise en place de l'infrastructure sur Azure.
3. Gestion de projet
   — Préparation à la soutenance de projet.

Le retard est en partie dû aux tâches propres à la recherche qui ont demander beaucoup plus de temps afin de comprendre les concepts de certains composant.
Or étant donné que la réalisation de l'architecture demande d'être au point sur la compréhension des documents, la construction de l'architecture sur azure à donc dû prendre du retard vis à vis du diagramme de GANTT prévisionnel.
Le retard de ces deux taches les plus importantes du projet on par conséquent, via un effet boule de neige engendré le bon déroulement en terme de planification, des autres tâches.

2.1.3 Comparaison

Voici donc les deux diagrammes juxtaposé afin de voir la différence entre diagramme de GANTT réel et prévisionnel.

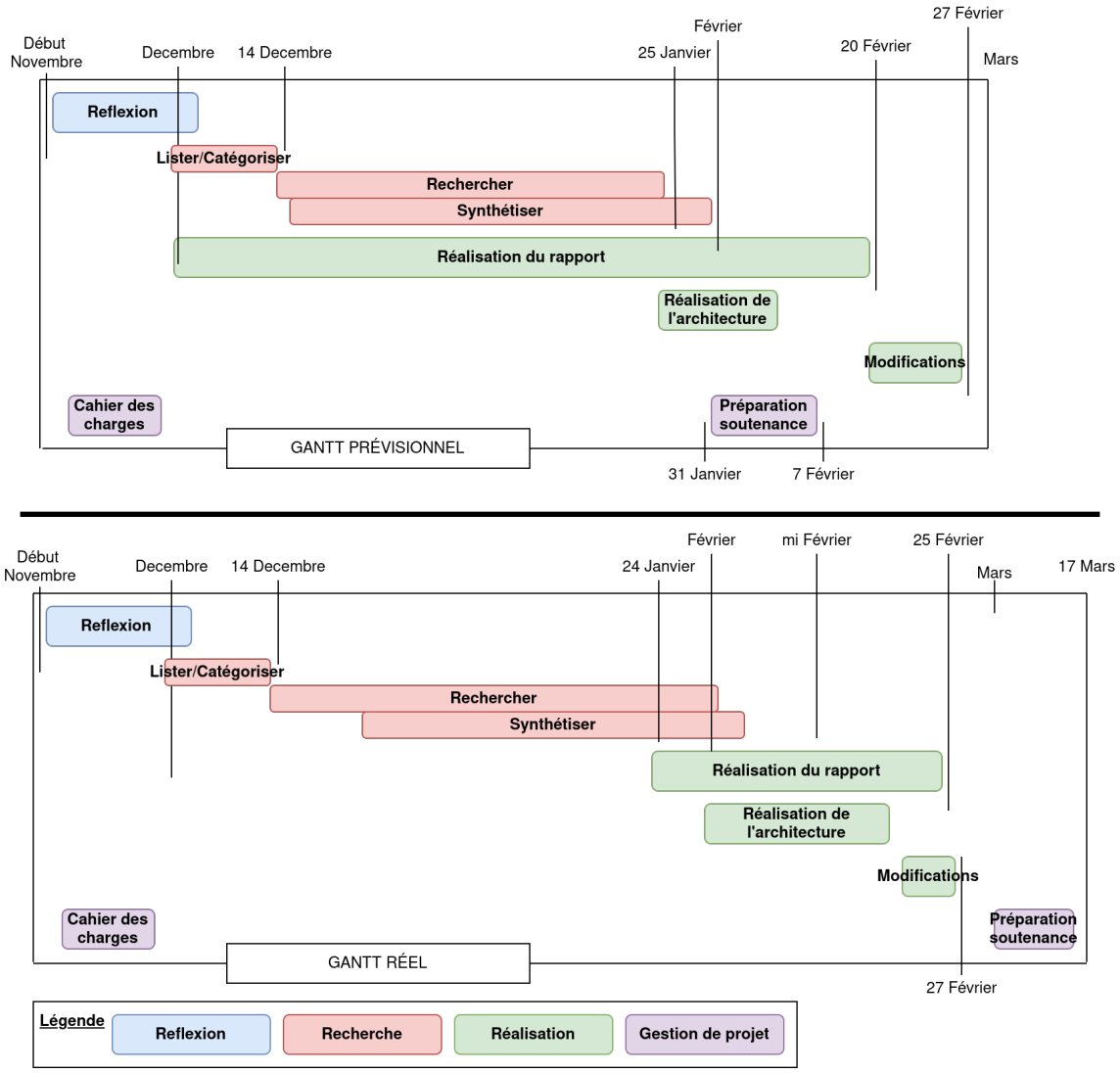


FIGURE 2.3 – Gantt Prévisionnel et Réel

Chapitre 3

Analyse et développement

L'architecture a été élaborée en accord avec les principales propriétés de sécurité recherchées qui sont :

L'intégrité
Propriété d'exactitude et de complétude des biens et informations (i.e. une modification illégitime d'un bien doit pouvoir être détectée et corrigée)

La confidentialité
Propriété des biens de n'être accessibles qu'aux personnes autorisées

La disponibilité
Propriété d'accessibilité au moment voulu des biens par les personnes autorisées (i.e. le bien doit être disponible durant les plages d'utilisation prévues)

Preuve Propriétés d'un bien [1] permettant de retrouver, avec une confiance suffisante, les circonstances dans lesquelles ce bien évolue. Cette propriété englobe Notamment :

1. La traçabilité des actions menées
2. L'authentification des utilisateurs
3. L'imputabilité du responsable de l'action effectuée

Ce sont donc ces critères qui serviront de base pour la mise en place de l'architecture. Cette dernière aura une base préétablie par la documentation Microsoft qui est la suivante.

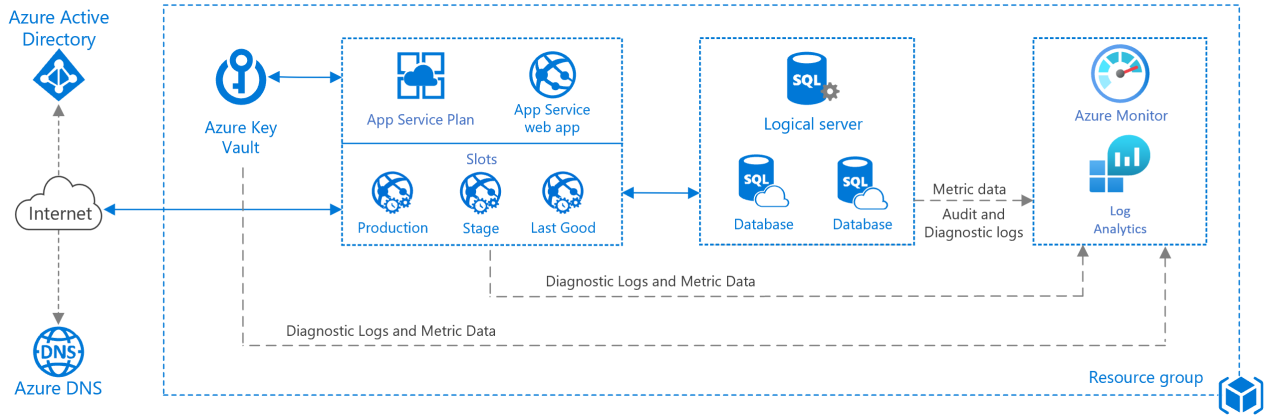


FIGURE 3.1 – Basic web app

Les différents composants cette architecture sont les suivants :

Azure *Key Vault*
Permet de gérer les clés, certificats ainsi que les secrets.

App Service Plan
Fournit les machines virtuelles qui hébergent les applications associées.

1. Ressource informationnelle directement associée au système informatique, qu'elle soit matérielle ou logicielle, acquise ou développée par une organisation, ayant une valeur quantifiable et qui peut faire l'objet d'un inventaire. Source : https ://www.oqlf.gouv.qc.ca

App Service Web App
Permet de créer et déployer des applications *cloud* (prends en charge la sécurité, l'équilibrage de charge, la mise à l'échelle automatique et la gestion automatisée) .

slots
Permet de séparer les environnement de développement, test ou autre de l'environnement de production. De plus le basculement d'un environnement vers un autre se fait facilement et avec certains avantages.

Production

Stage

Last Good

Logical Server
Héberge les bases de données.

Database
Des bases de données SQL, on peut aussi y utilisé le service *Azure Database pour PostGreSQL* ou *Azure Database pour MySQL*.

Azure DNS
Est un service d'hébergement pour les domaines DNS et ce via les services Microsoft Azure.

3.1 Instanciation de l'architecture de base

La construction de l'application basique se fera dans cette example via le template proposé par microsoft, il suffit de créer un *resource group* pour l'application puis d'y générer les ressources via le template.

Voici les commandes permettant de créer le *resource group* ainsi que les ressources propre à l'application basique :

```
# Création du resource group nommé wep-app
az group create --name web-app --location westeurope

# Déploiment de l'application basic-web-app depuis son template dans le resource groupe web-app
az deployment group create --resource-group web-app --template-uri \
https://raw.githubusercontent.com/mspnp/samples/master/solutions/basic-web-app/azuredeploy.json
```

Lors de la création de l'application un mot de passe est requis pour l'administration de la base de donnée SQL.

Voici donc la liste des ressources instancié pour l'application.

Name	Type	Location
app-etoii7uhhgvue	App Service plan	West Europe
app-etoii7uhhgvue	App Service	West Europe
LastKnownGood (app-etoii7uhhgvue/LastKnownGood)	App Service (Slot)	West Europe
Staging (app-etoii7uhhgvue/Staging)	App Service (Slot)	West Europe
kv-etoii7uhhgvue	Key vault	West Europe
la-etoii7uhhgvue	Log Analytics workspace	West Europe
sql-etoii7uhhgvue	SQL server	West Europe

FIGURE 3.2 – Basic web app resource group list

Lors de la création, différents éléments de l'application tel que l'*app service plan*, l'*app service* ansi que des *slots* de **production, staging et LastKnownGood** ont été instancié.
Il y a aussi la ressource *log analitycs workspace* propre aux journaux d'événements.
Le serveur SQL est aussi instancié, ainsi que le *key vault*.
Il faut savoir que pour ce dernier il est impossible d'avoir un deuxième *key vault* portant le même nom et ceux y compris dans les *resource group* autres que *web-app*.
Lors de la suppression d'un *key vault* celui-ci n'est pas totalement détruit à moins de le *"purger"* dans la section **Key vaults**, puis **Manage deleted vaults**. Dans le cas d'une erreur, dans cette même section il est possible de restaurer un *key vault* supprimé.

Voici **FIGURE 3.3** un autre schéma reprenant la liste mais d'un point de vue hiérarchique.

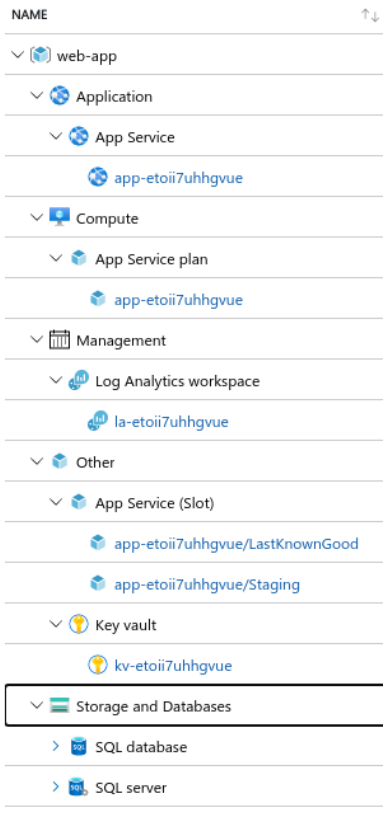


FIGURE 3.3 – Vue hiérarchique du ressource group

L'application déployé sur l'*app service* sera un *fork* d'un exemple proposé par Azure (https://github.com/Azure-Samples/nodejs-docs-hello-world). Il s'agit d'un simple serveur *nodeJs* renvoyant un *Hello world* sur la route se situant à la racine de l'application c'est à dire *http ://baseURL/* avec *baseURL* : l'url de notre applicaiton soit localhost en locale ou alors une adresse généré par azure si aucun nom de domaine est ajouté.

Pour le déploiement de l'application sur Azure, il faut dans un premier temps :

1. Choisir le *slot* dans lequel déployer l'application (Ici le choix fût le *slot* production)
2. Selectionner l'onglet *Deployment center* du *slot*
3. Choisir la méthode de déploiment parmis les différentes options possible (ici le choix c'est porté sur github ce qui requiert une authentification voir FIGURE 3.4)
4. Remplir les informations propres aux *repository* contenant l'applicatif à déployer. *voir FIGURE 3.5)*

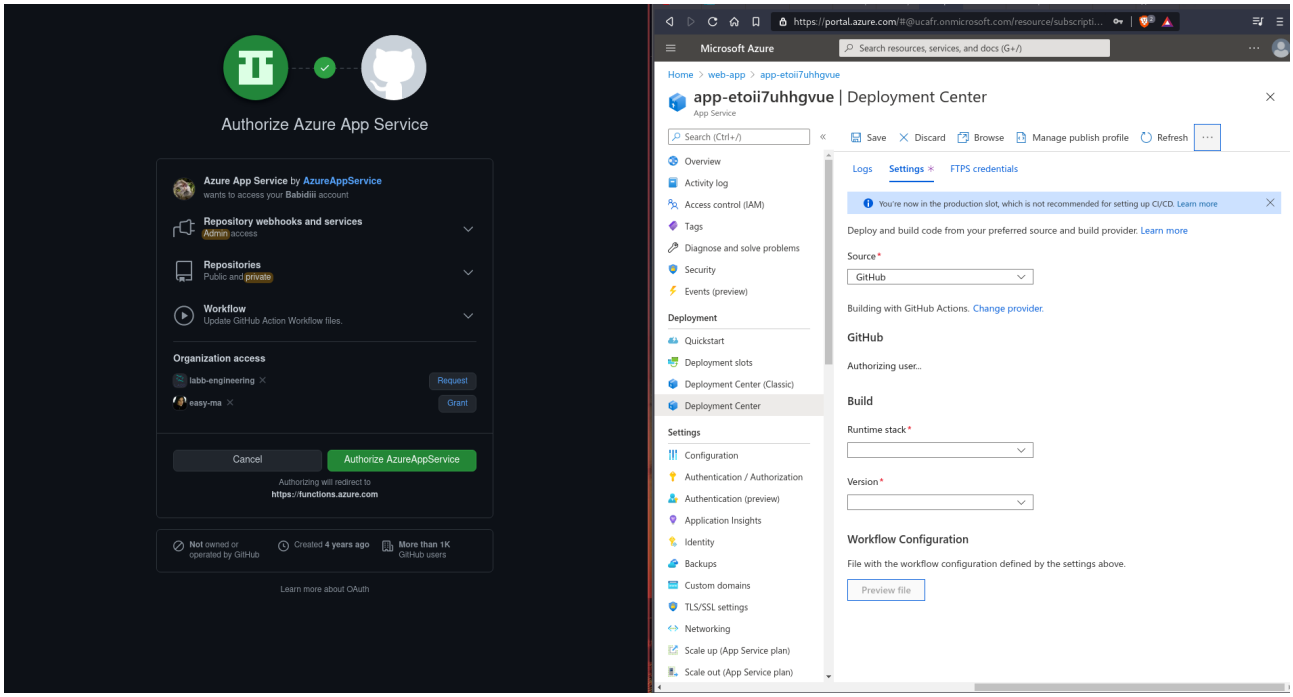


FIGURE 3.4 – Authentification via github

Une fois le déploiment ajouté, azure va lier le *repositry* et y ajouter le fichier de configuration pour la *GithubAction* permettant de deploiyer le projet sur azure après chaque commit sur la branche.

Ce fichier contient les instruction propre à la construction et au déploiment de l'application. Voici donc le contenu :

```
1  # Docs for the Azure Web Apps Deploy action: https:\/\/github.com\/Azure\/webapps-deploy
2  # More GitHub Actions for Azure: https://github.com/Azure/actions
3
4  name: Build and deploy Node.js app to Azure Web App - app-etoii7uhhgvue
5
6  on:
7    push: # A chaque push sur la branche master
8      branches:
9        - master
10   workflow_dispatch:
11
12 jobs:
13   build-and-deploy:     # le type de machine sur lequel effectuer le job
14     runs-on: windows-latest     # image utilisé
15
16     steps:  # repertorie les différentes étapes du job
17     - uses: actions/checkout@master
18
19     - name: Set up Node.js version
20       uses: actions/setup-node@v1   # utilise une action pour setup nodejs (voir doc sur uses)
21       with:
```

```
22          node-version: '14.x'        # définit la version de node utilisé
23
24      - name: npm install, build, and test    # instruction de build
25        run: |
26          npm install
27          npm run build --if-present
28          npm run test --if-present
29      - name: 'Deploy to Azure Web App'   # deploiement sur azure
30        uses: azure/webapps-deploy@v2
31        with:
32          app-name: 'app-etoii7uhhgvue'       # nom de l'application sur azure
33          \textit{slot}-name: 'production'            # nom du \textit{slot}
34          publish-profile: \$\{\{ secrets.AzureAppService_PublishProfile_35893b7233f2489cba7933a37bb76da1 \}\}
35          package: .
```

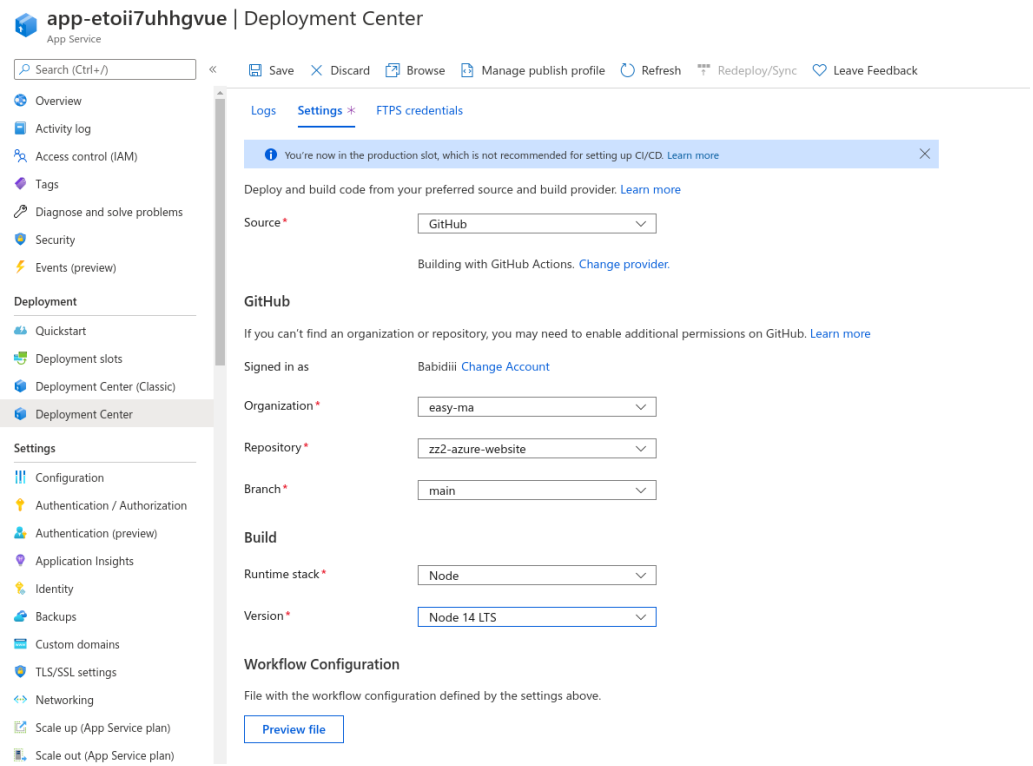


FIGURE 3.5 – Deploiment d'une application d'un *repository* github sur le *slot* de production

Suite à cela, il est possible d'accéder à l'application en clinquant dans le même onglet sur *Browse*, ce qui donne le résultat suivant FIGURE 3.6.

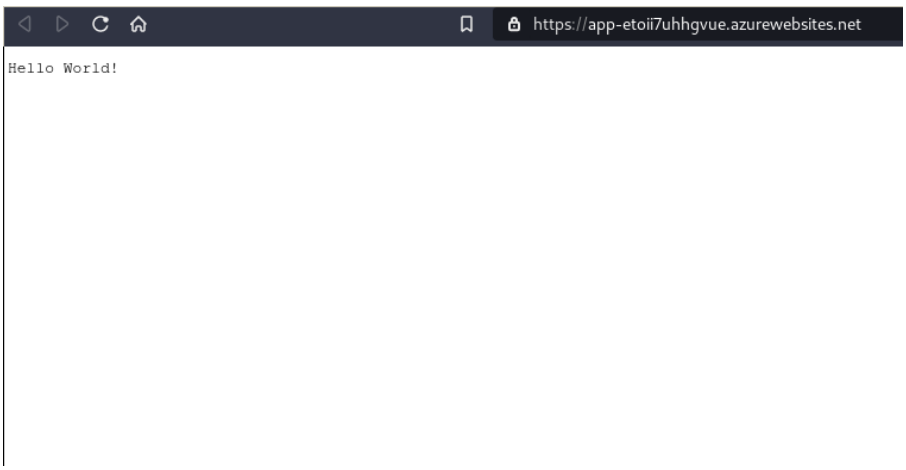


FIGURE 3.6 – Test du site web de production

Voici FIGURE 3.7 l'exécution de l'action github.

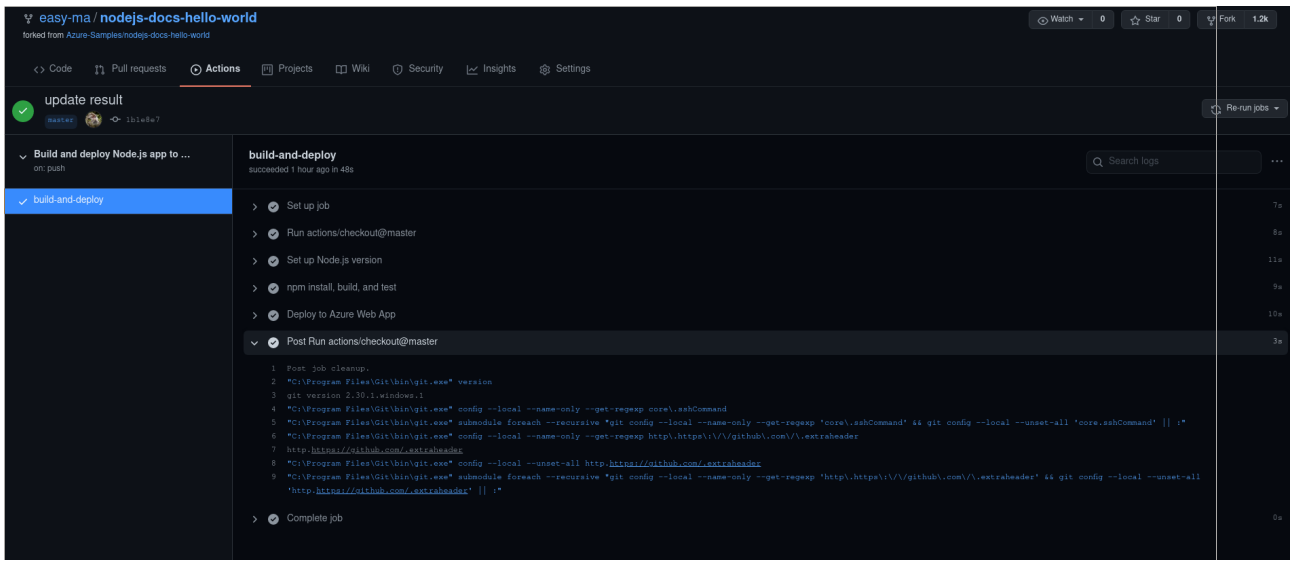


FIGURE 3.7 – GithubAction du *repository* propre à l'application après un *push* sur la branche main

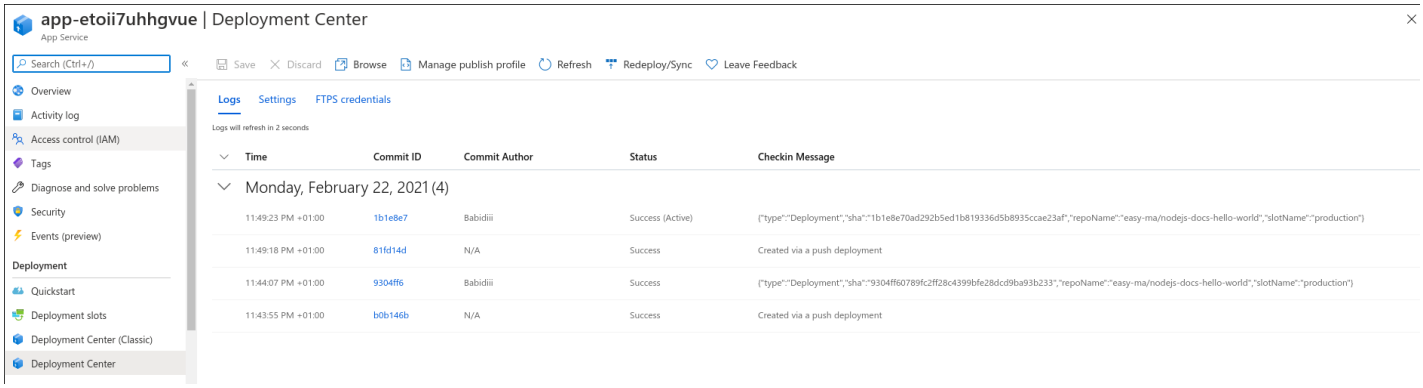


FIGURE 3.8 – Réception sur Azure des changements appliqués sur la branche main.

Sur le *slot* de production dans Azure il est possible de voir les *commit* déployé, voir FIGURE 3.8

En effectuant une modification suivie d'un *push* sur *master*, on constate les modification faites du coté de Azure ainsi que la page et de son nouveau contenu voir FIGURE 3.9.
Il faut noter que dans un cas concret les commit ne serait pas associé pour la branche *master* au *slot* de production mais à un autre *slot* que l'on aurait soigneusement *swap* avec celui de production ou pre-production une fois validé.

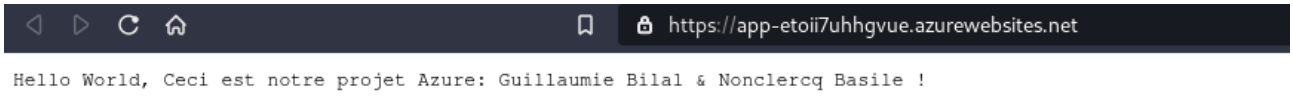


FIGURE 3.9 – Test du site de production après changement

Il est alors possible de configurer aussi le *slot* de staging que l'on associera dans cette exemple à la branche nommé stage, voir FIGURE 3.10.

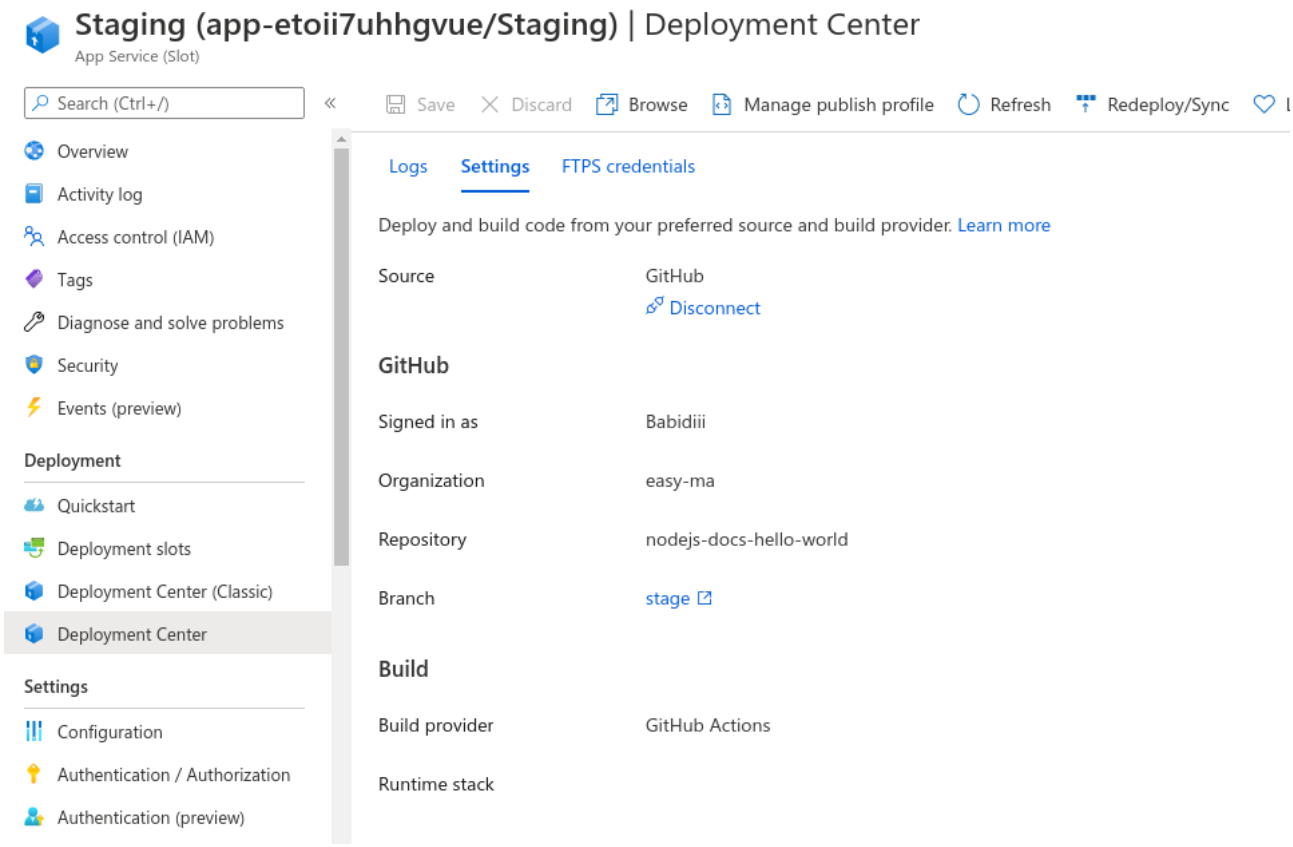


FIGURE 3.10 – Déploiement du *slot staging* depuis la branche stage de rdu *repository*

En suivant le même processus que pour le *slot* de production il est alors ensuite possible de tester le site à l'adresse url spécifique au *slot staging* et de trouver le resultat FIGURE 3.11.

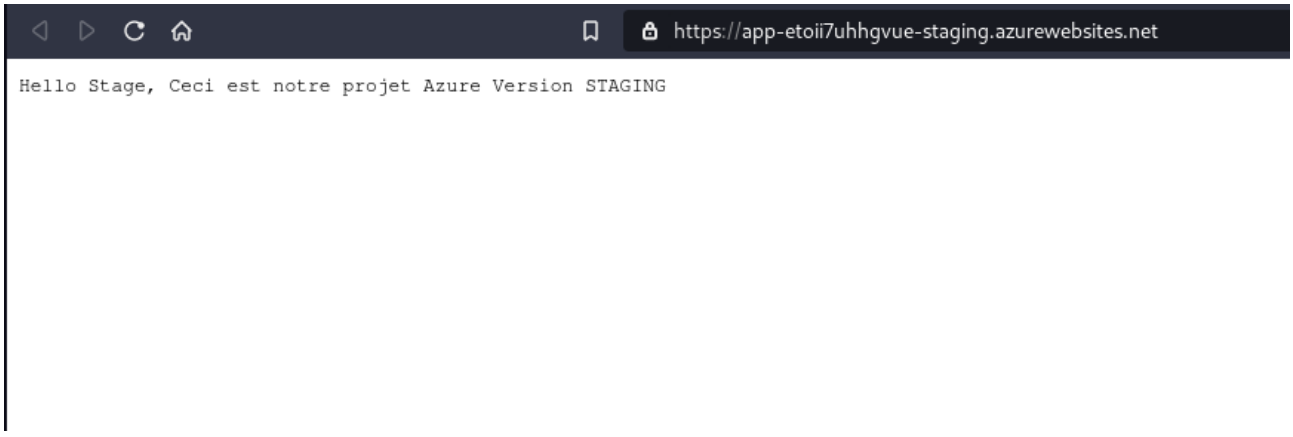


FIGURE 3.11 – Test de la page en *staging*

3.2 Amélioration de l'application

Afin d'améliorer la sécurité de l'application Azure propose plusieurs composants plus ou moins utiles dans le cas de l'application web basique implémenté.
Voici une liste exhaustive des composants dit de *"sécurité"* dans l'écosystème.

1. ***Application Gateway***
2. *Azure Information*
3. *Active Azure Directory (AD)*
4. *Azure Sentinel*
5. *Azure Active Directory Domain Services*
6. ***Key Vault***
7. *Azure DDoS Protections*
8. ***Security Center***
9. *Azure Dedicated HSM*
10. *VPN Gateway*
11. *Azure Defender*
12. ***Web Application Firewall***
13. *Azure Defender for IoT*
14. *Azure Front door*

Les composants les plus intéressants à implémenter et qui seront retenus pour la mise en place de notre architecture sont ceux mis en gras. On y retrouve *Application Gateway* qui permet d'implémenter un Pare feux applicatif autrement appelé *Web Application Firewall*.
Azure Defender semble être d'autant plus intéressant, ainsi que *Security Center*.
Ces composants seront abordé au fur et à mesure afin d'expliquer leur concepts et par la suite les ajouter au projet existant.

3.2.1 Application Gateway

Descripton du composant

Une *Application Gateway* aussi connu sous le nom d'application *proxy* ou *Application level proxy (ALG)* est un programme applicatif qui s'exécute afin de permettre la sécurisation d'un réseau.

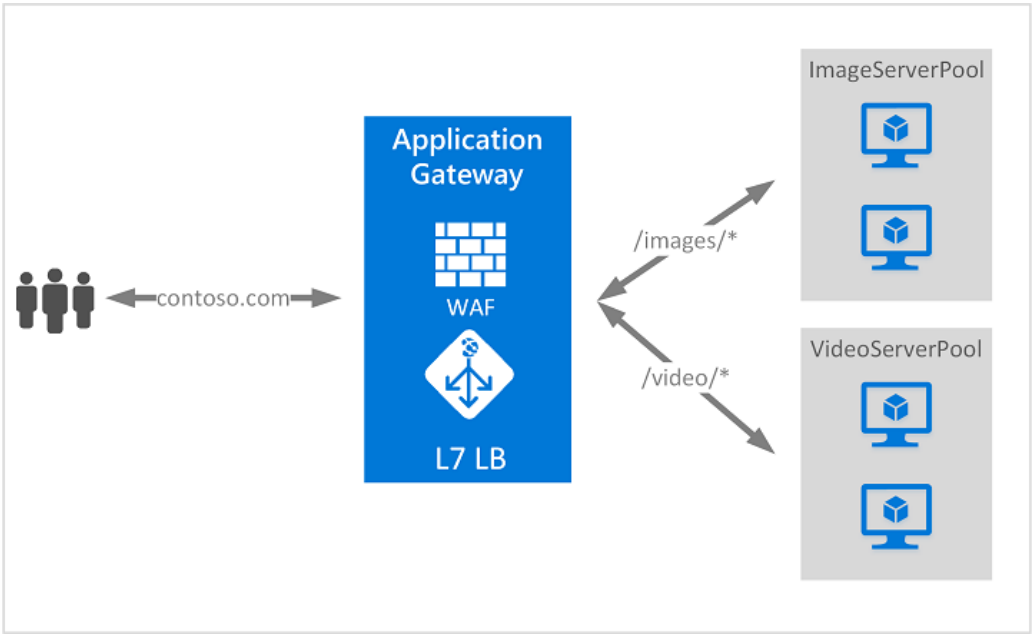


FIGURE 3.12 – Application Gateway

D'un point de vue réseau, *Azure Application Gateway* est un équilibreur de charge du trafic web on parle de *load balancer*. La différence avec un *Load balencer* traditionnel étant que celui ci opère sur la couche **Transport** du model *OSI - Couche 4 - TCP et UDP* et permet de *"router le traffic"* en se basant sur l'IP source ainsi que le port, vers une adresse IP :Port de destination.
Tandis qu'une *Application Gateway* se base sur des données supplémentaires propre au protocole HTTP/HTTPS [2]. Il opère sur la couche 7 *Couche Application* du modèle *OSI - TCP et UDP*, d'òu son nom d'*Application Gateway*.

Voici FIGURE 3.13 un schéma représentant le fonctionnement d'une *Application Gateway*

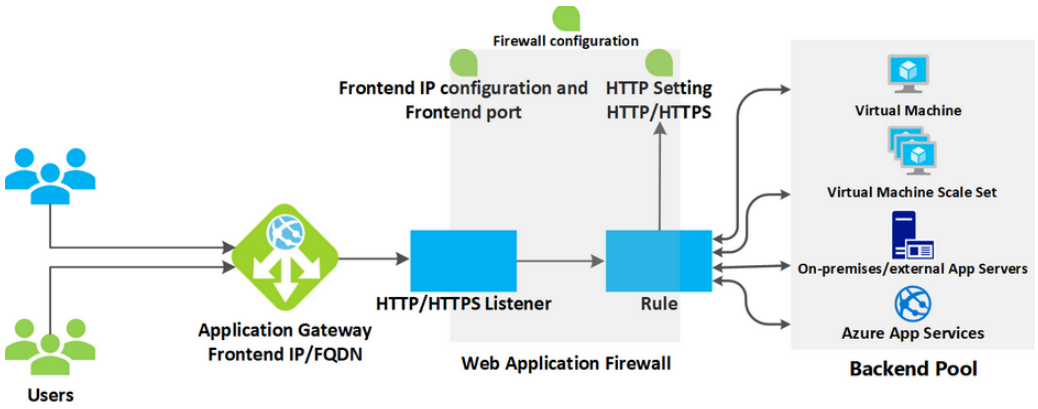


FIGURE 3.13 – Fonctionnement Application Gateway

2. HyperText Transfer Protocol (Secure) : protocole de communication client-serveur développé pour le World Wide Web

Lors d'une requête à un réseau interne, on y retrouve les étapes suivantes :

1. Résolution du nom de domaine de la passerelle d'application via le serveur *DNS* d'Azure.
2. Requête à l'adresse IP correspondante (ip frontend).
3. Un *listener* s'occupe de traiter la validité de la requête (port, protocole, hôte et IP).
4. Le *WAF* va vérifier la validité du contenue, c'est à dire si les règles mises en place sont respectées.

L'*Application Gateway* possède plusieurs fonctionnalitées. En voici la liste :

1. Terminaison Secure Sockets Layer (SSL/TLS)
2. Mise à l'échelle automatique
3. Redondance de zone
4. Adresse IP virtuelle statique
5. Pare-feu d'applications web
6. Contrôleur d'entrée pour AKS
7. Routage basé sur des URL
8. Hébergement de plusieurs sites
9. Redirection
10. Affinité de session
11. Trafic Websocket et HTTP/2
12. Drainage des connexion
13. Pages d'erreur personnalisées
14. Réécrire les en-têtes et les URL HTTP
15. Dimensionnement

Ces fonctionnalités sont des plus intéressantes en terme de sécurité notamment la numéro 5 : **Pare-feu d'application web**.

L'*Application Gateway* peut être implémenter en tant que *WAF (Web Application Firewall)*. D'après la documentation,

> Le pare-feu d'applications web Azure (WAF) sur Azure Application Gateway fournit une protection centralisée de vos applications web contre des vulnérabilités et exploitations courantes. Les applications web sont de plus en plus visées par des attaques malveillantes qui exploitent des vulnérabilités connues. L'injection de code SQL et l'exécution de scripts de site à site font partie des attaques les plus courantes.

Implémenter un *WAF* sur l'application permet donc de protéger le serveur d'applications Web contre diverses attaques provenant de l'extérieur. Il est généralement au courant de la session, de l'utilisateur et des services proposés par l'applicatif.

Les règles suivies par le *WAF* sont celles d'un ensemble de règles de la base *OWASP ModSecurity Core Rule Set 3.1* (https ://github.com/coreruleset/coreruleset). Parmis ces règles, on y retrouve celles permettant de prévenir des attaques présentes dans le *OWASP Top Ten* (https ://owasp.org/www-project-top-ten/) [3].

A savoir :

3. Azure WAF avec le Core Rule Set 3.1 est le plus performant des WAF courants. Source : https ://medium.com/fraktal/cloud-waf-comparison-using-real-world-attacks-acb21d37805e

Injection Ce sont des d'injection malveillantes comme l'injection SQL. Lorsqu'elles arrivent à l'applicatif, elles peuvent entraîner l'exécution de commande non autorisées.

Broken Authentication Lorsque l'attaquant réussi à avoir un accès qui n'est pas le siens. Par exemple, si il arrive à se connecter à une session qui n'est pas la sienne.

Sensitive Data Exposure via un *token* (jeton) de connections qui n'est pas le siens. Cela arrive lorsque des données confidentiels ne sont pas correctement protégées. Ces données peuvent par exemple être des cartes de crédits ou une clé privée.

XML External Entities (XXE) Via les entités externes, il est possible de faire des références à une source externe (URI [4] et URL [5]). La faille consiste à faire une référence à une ressource protégé ou malveillante.

Broken Access Control La faille permet à un attaquant de réaliser des actions normalement restreinte comme l'accès à certains fichiers ou la modification de données propre à une autre personne.

Cross-Site Scripting (XSS) XSS permet l'exécution de scripts malveillants via le navigateur (de l'attaquant ou de la victime selon si elle est *stored*(stockée) ou *reflected*(renvoyée)).

Insecure Deserialization [6] C'est faille se produit lorsque des données non fiables sont traiter par l'applicatif pour abuser de sa logique.

Voici un schéma FIGURE 3.14 montrant la protection qu'ajoute un *WAF* sur des requêtes malveillantes.

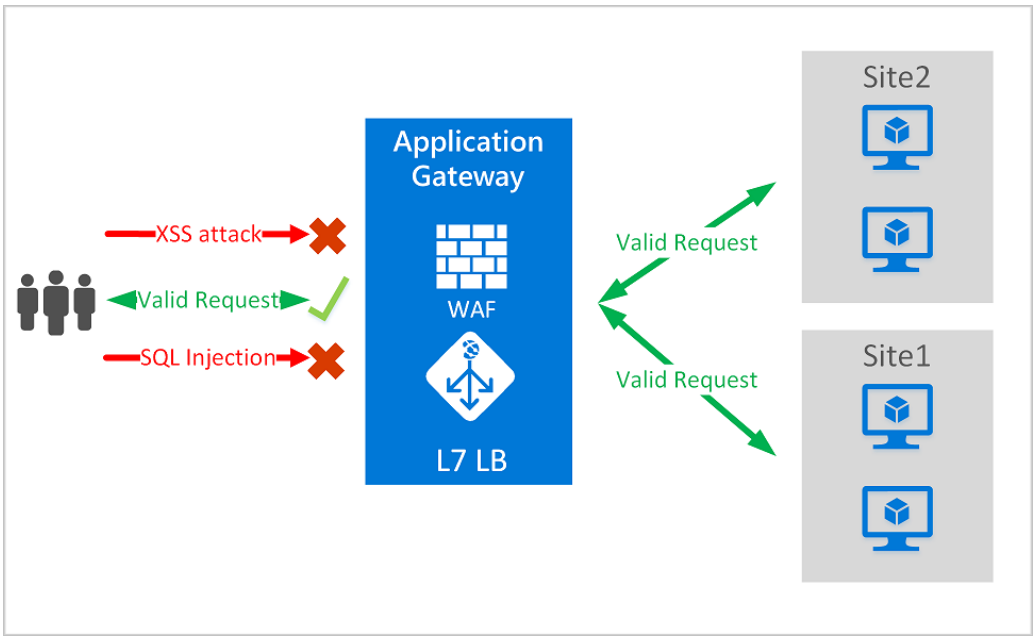


FIGURE 3.14 – Schéma WAF

Lors de la configuration du *WAF* sur Azure, il faut spécifier le mode du *WAF* que l'on souhaite implémenter. Il en existe deux qui sont les suivants :

Détection Surveille et journalise toutes les alertes de menace. Vous activez l'enregistrement des diagnostics pour Application Gateway dans la section Diagnostics.

Prévention Bloque les intrusions et les attaques détectées par les règles. L'attaquant reçoit une exception « 403 Accès non autorisé » et la connexion est fermée.

4. Uniform Resource Identifier : Chaîne de caractère identifiant une ressource sur un réseau
5. Uniform Ressource Locator : Chaîne de caractère identifiant une ressource du Worl Wilde Web ou plus communément Internet

Azure Application Gateway gère également l'intégrité des ressources de son pool de back-ends et supprime si nécessaire ces dernières. Il est également possible de définir des règles pour la sonde d'intégrité.

Azure Application Gateway peut être associé à *Azure Sentinel* (voir section 5.6) afin de gérer au mieux les évènements.

Ajout du WAF dans l'application

L'objectif est d'implémenter le schéma suivant :

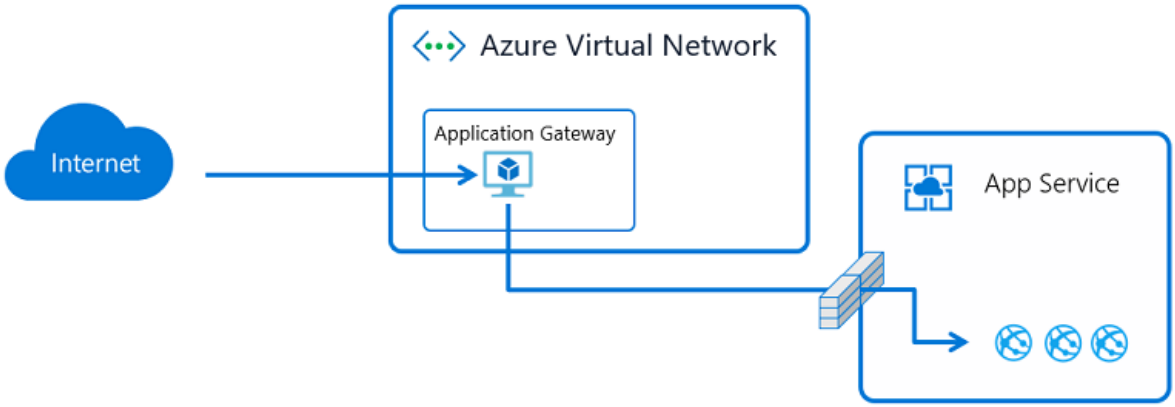


FIGURE 3.15 – Implémentation WAF

Sur azure il existe plusieurs *tiers* pour une application gateway. C'est via ce champ qu'il faut spécifier si l'on souhaite que l'*Application gateway* fonctionne comme un WAF Il peut être **standard**, **WAF** ou **WAFv2**. Il est toujours possible de changer cela une fois l'*Application gateway* créé.

La différence entre un WAF et un WAFv2 peut être listé sur la page suivante : comparaison

Concernant la création de l'*Application Gateway* avec fonctionnalité WAF voici le résumé de sa configuration FIGURE 3.16

Le virtual network à été créé lors de l'ajout de l'*Application Gateway* avec un *subnet* en 10.0.0.0/24. l'adresse ip public associé fût elle aussi créé voir FIGURE 3.17.

Basics

Subscription	Azure for Students
Resource group	web-app
Name	my-app-gateway
Region	West Europe
Tier	WAF
Instance count	2
SKU size	WAF_Medium
Firewall status	Enabled
Firewall mode	Prevention
HTTP2	Disabled
Virtual network	(new) waf-vnet
Subnet	(new) default (10.0.0.0/24)
Subnet address space	10.0.0.0/24

Frontends

Public IP address name	mon-waf
SKU	Basic
Assignment	Dynamic

Tags

None

FIGURE 3.16 – Résumé création Applicaiton Gateway + WAF

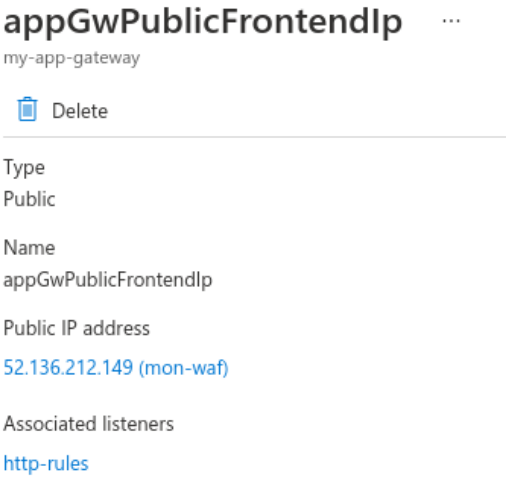
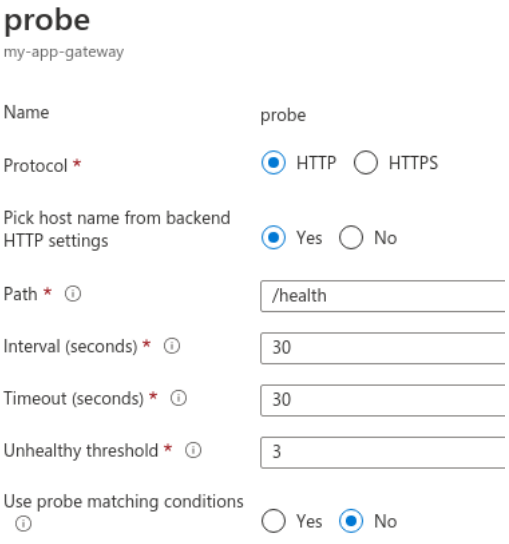


FIGURE 3.17 – Ip public associé au WAF

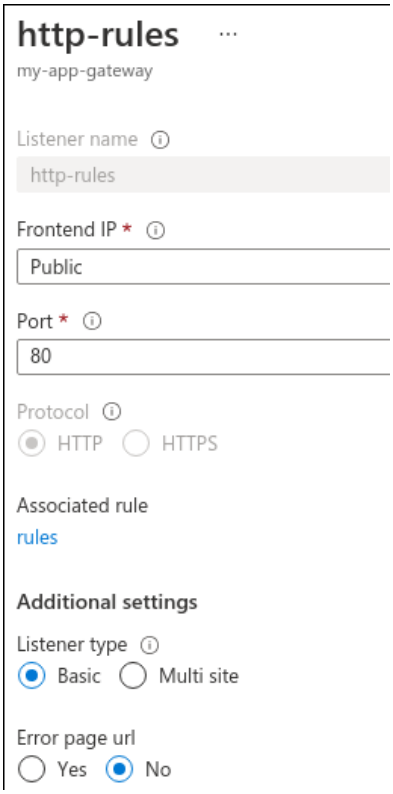
L'adresse de l'ip public associé à l'application gateway étant donc 52.136.212.149.
En ce qui concerne le *backend pool* de l'*Application Gateway*, l'app service soit l'app de production y à été ajouté.

Afin de finir la configuration de l'*Application Gateway* et de la fonctionnalité *WAF*, il est nécessaire d'ajouter au minimum une régle concernant le listner ainsi qu'une configuration HTTP, voir FIGURE 3.18

Il est aussi nécessaire d'ajouter une *Health probe* du coté de l'*Application Gateway* mais aussi du coté de l'application afin de pouvoir lier l'un à l'autre et ainsi partager leur *"état de santé"* respectif.
Pour l'*Application Gateway* le choix de la route fût */health*.



(a) Probe



(b) Http rules

FIGURE 3.18 – Règles

Voici donc FIGURE 3.19 l'apercu de l'état de l'*application gateway*

FIGURE 3.19 – *Insight de l'Application Gateway*

Suite à cela, il est désormais impossible d'accéder au site web directement depuis son nom de domaine par défault soit *https ://app-etoii7uhhgvue.azurewebsites.net/*. En effet, l'erreur *403 Forbidden* est reçu comme réponse car il faut impérativement passer par la Gateway, voir FIGURE 3.20.

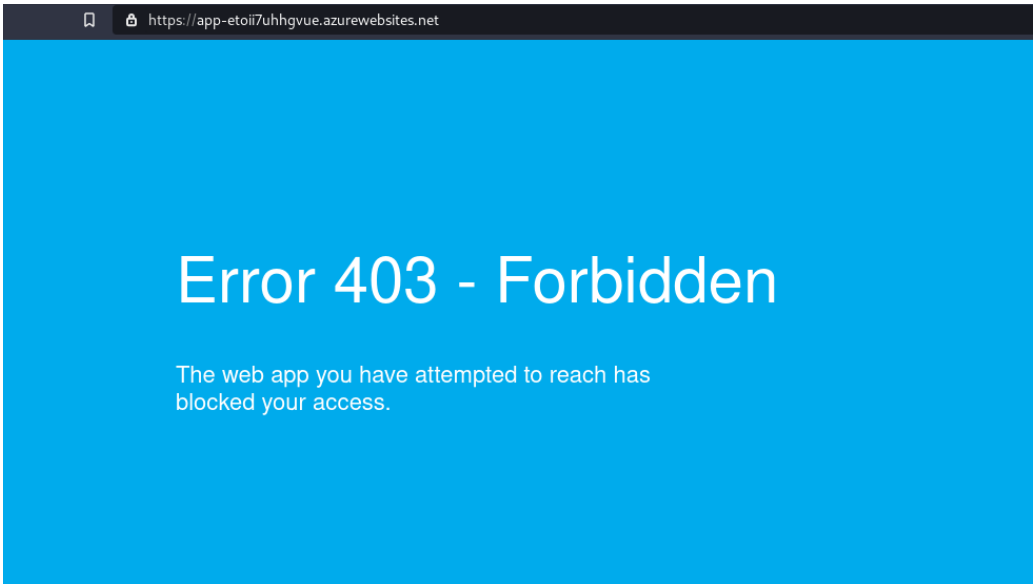


FIGURE 3.20 – *Réponse en cas d'accés directe sur l'application*

Via l'élaboration d'une application vulnérable sur l'application de production, il sera donc possible de tester les régles de sécurité implémenté par défaut par le WAF.
En modifiant le serveur nodejs de notre *repository* pour le code suivant.

```
const http = require("http");
const url = require("url");

const server = http.createServer((request, response) => {
  const queryData = url.parse(request.url, true).query;
  response.writeHead(200, { "Content-Type": "text/html" });

  if (queryData.name) {
    response.end("Hello " + queryData.name);
  } else {
    response.end("Hello World call url/?name='yourname'\n");
  }
});

const port = process.env.PORT || 1337;
server.listen(port);

console.log("Server running at http://localhost:%d", port);
```

Cela permet de rendre l'application vulnérable à une attaque XSS via le paramètre **name** de l'url, voir FIGURE 3.21.

En soumettant les changement sur la branche master et en re-déployant l'application via les githuAction, il est alors possible de tester voir si ce comportement se reproduit sur Azure, voir FIGURE 3.22. Or ce n'est pas le cas car le WAF implémenté par le biai de l'Application Gateway permet de bloquer ce genre d'attaque et de renvoyer une réponse *403 Forbidden*

FIGURE 3.23 se trouve la liste des ressource instancié après l'ajout de l'application gateway.

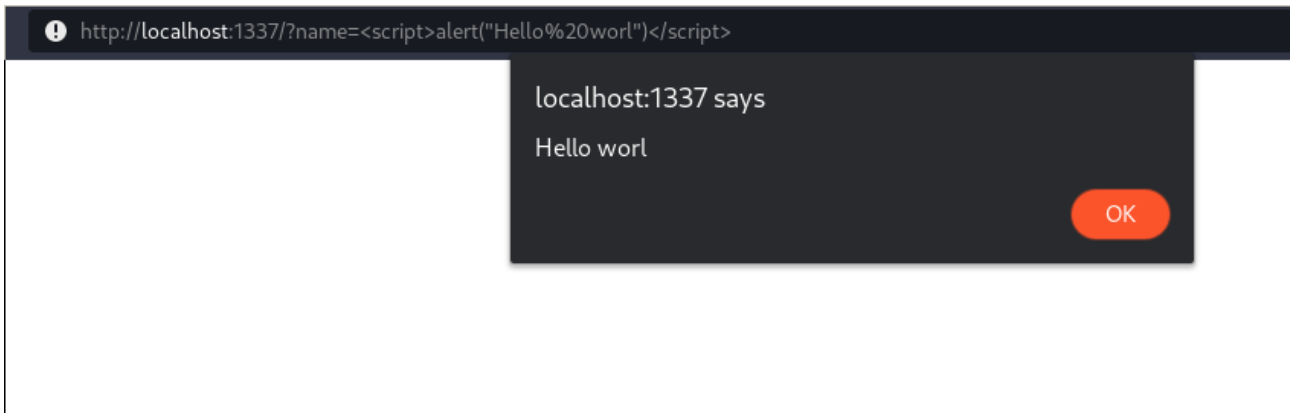


FIGURE 3.21 – *Vulnérabilité XSS en localhost*

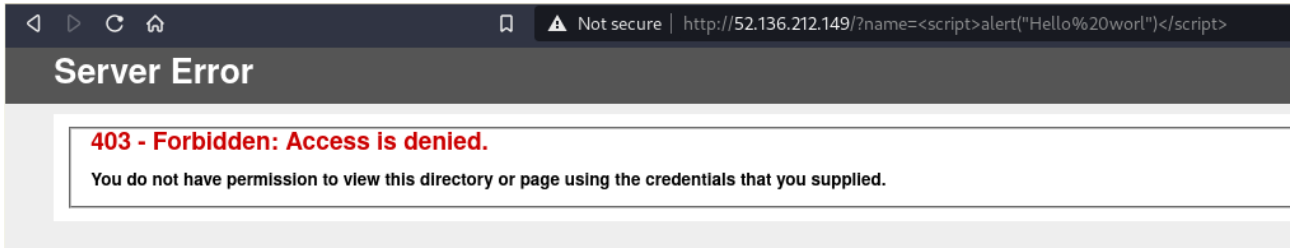


FIGURE 3.22 – *Vulnérabilité XSS sur Azure*

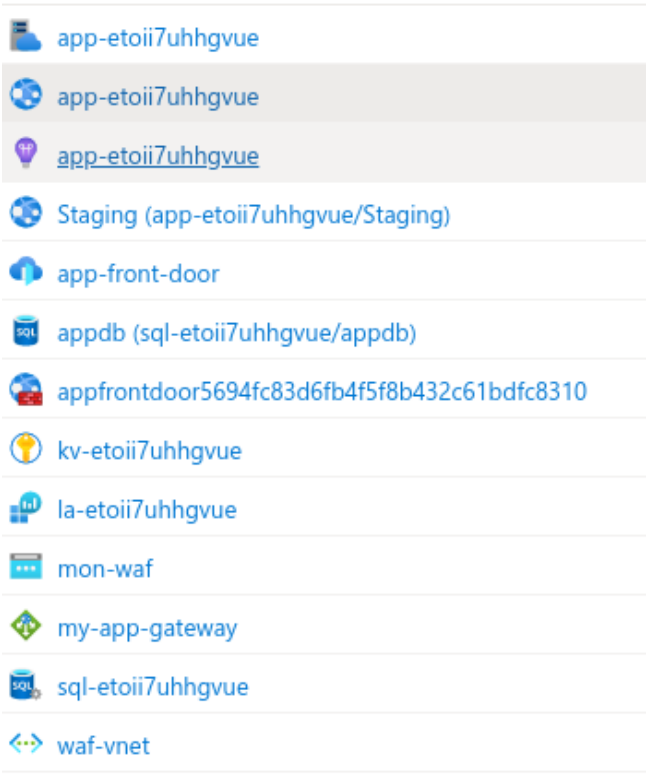


FIGURE 3.23 – *Resource group après ajout du WAF*

3.2.2 Azure *Key Vault*

Azure *Key Vault* permet de faire de la

Gestion de secrets

Gestion de clés

Gestion de certificats

Deux service pour azure *Key Vault*

Standard

Premium

Lien vers plus d https ://azure.microsoft.com/en-us/pricing/details/key-vault/

Le but du *Key Vault* sera de centraliser les secrets propre aux applications afin de contrôler leurs distribution et réduire leurs expositions au maximum à des tiers.

L'application peut alors avoir accès aux secret via une *URI*. Pas besoin de certaines lignes de code à ajouter du coté de l'application.

Sécurisation du *Key Vault*

L'**authentification** est faite via *Azure Active Directory*.
Tandis que l'**autorisation** peut être fait via *Azure role-based access control-Azure RBAC* ou alors via *Key Vault access policy*.

Différences entre les deux d'après la documentation :

Azure RBAC est utilisé quand on effectue la gestion du *Key Vaults*

***Key Vault* Acess policy** est utilisé lorsque l'on essaye d'accéder au valeurs stocker dans un vault

3 manières de s'authentifier

Identités managées pour les ressources Azure**

Lors du déploiement d'une application sur une VM dans Azure, on peu y assigner une identité propre à la VM qui aurait donc accès au *Key Vault*.

Principal de service et certificat

Utilisation d'un service principal et un certificat associés qui a accès au *Key Vault*.
Nécessaire de renouveler régulièrement le certificat par le propriétaire de l'application et des développeurs.

Principal de service et secret

Pas recommendée, car difficile de renouveler automatiquement le secret de démarrage qui sert à l'authentification auprès du *Key Vault*.

Deux interfaces contrôle l'accès au *Key Vault*.

Plan de gestion

Permet de gérer le coffre de clés.

1. Création de coffres
2. Suppression de coffres
3. Récupérer des propriétés, mettre à jour des stratégies d'accès

Plan de données

Permet l'utilisation des données stockées dans un *Key Vault*.

1. Création de clées/certificats/secrets.
2. Suppression de clées/certificats/secrets.
3. Éditer des clées/certificats/secrets.

Chaque plan utilise *Azure Active Directory* pour l'authentification.

Sécurité supplémentaire

Firewall et virtual network

1. Refuser par défaut l'accès au trafic en provenance de tous les réseaux (internet compris).
2. Accorder l'accès au trafic en provenance de réseaux virtuels ou des plages d'adresses IP public spécifiques

Lien de l'implémentions : https ://docs.microsoft.com/fr-fr/azure/key-vault/general/network-security

Point de terminaison privé

https://docs.microsoft.com/fr-fr/azure/key-vault/general/secure-your-key-vault#resource-endpoints

https://docs.microsoft.com/fr-fr/azure/key-vault/general/secure-your-key-vault#private-endpoint-connection

3.2.3 Azure Security Center

Azure Security Center est un **système de gestion centralisé de la sécurité de l'infrastructure**. Il renforce la posture de sécurité de vos centres de données et fournit une protection avancée et évolutive contre les menaces pour vos charges de travail hybrides dans le cloud (dans Azure ou non), ainsi qu'en local.

Security Center offre les outils nécessaires pour :

1. **Renforcer la posture de sécurité** : *Security Center* évalue votre environnement. Il vous permet de comprendre l'état de vos ressources et de savoir si elles sont sécurisées.
2. **Vous protéger contre les menaces** : *Security Center* évalue vos charges de travail et émet des recommandations de prévention contre les menaces et des alertes de sécurité.
3. **Être plus rapidement en sécurité** : Dans *Security Center*, toutes les opérations sont réalisées à la vitesse du cloud. Étant donné qu'il est intégré en mode natif, le déploiement de *Security Center* est facile, vous offrant un approvisionnement automatique et une protection avec les services Azure

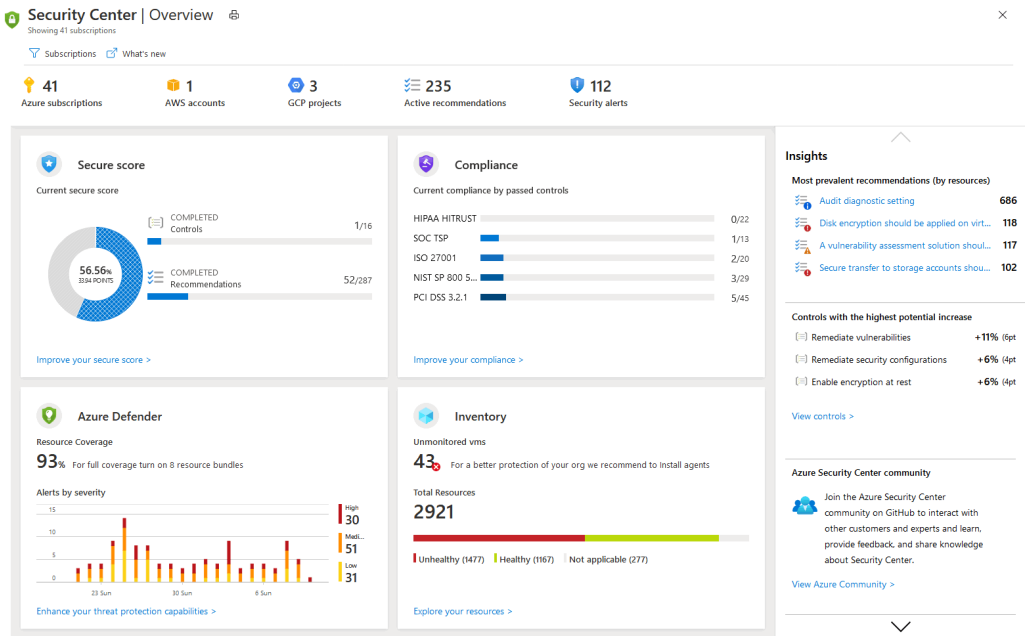


FIGURE 3.24 – Overview d'Azure Security Center

Azure Security Center est déjà intégré pour chaque déploiement réalisé dans Azure.

Sur l'architecture déployée, on peut y voir le *Secure Score* (voir figure 3.25). Chaque point correspond à une recommandation de sécurité(voir figure 3.26).

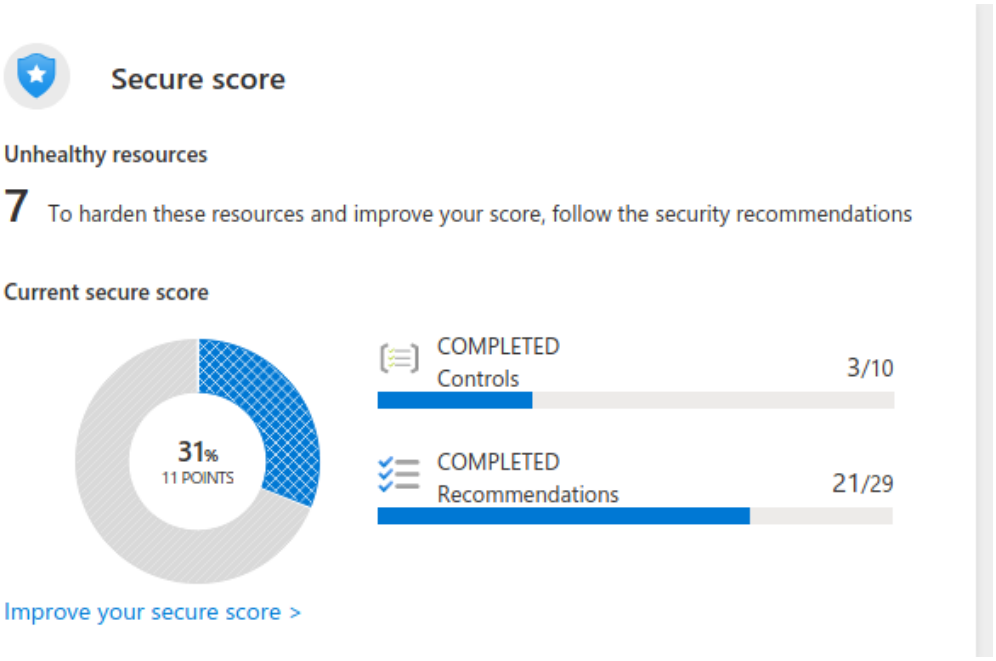


FIGURE 3.25 – Secure Score

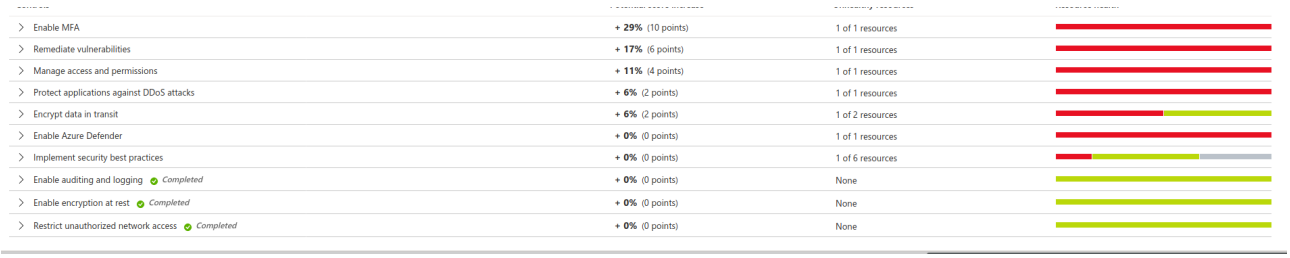


FIGURE 3.26 – Détails du Secure Score

Azure Defender

Azure Defender est une fonctionnalité supplémentaire payante d'*Azure Security Center*. Il permet de **fournir des alertes de sécuritées** ainsi qu'une **protection avancée** contres différentes menaces pour :

1. Les serveurs / Virtual Machines.
2. App Service.
3. Stockage.
4. SQL [7].
5. Kubernetes.
6. Les registres de conteneurs.
7. Key Vault.

7. *Structured Query Language* est un langage informatique normalisé servant à exploiter des bases de données relationnelles.

8. Resource Manager.
9. Domain Name System.

Ainsi, il permet d'éviter un maximum toutes les failles de type :

Security Misconfiguration Elle est dûe à des mauvaises configurations, à des configurations incomplètes ou à des messages d'erreurs trop verbeux.

Using Components with Known Vulnerabilities Comme son nom l'indique, cette faille est la conséquence d'un composant rendu vulnérables.

Insufficient Logging & Monitoring Cette faille concerne le manque d'alerte ou de journalisation entraînant le retard de l'arrêt de l'attaque.

.

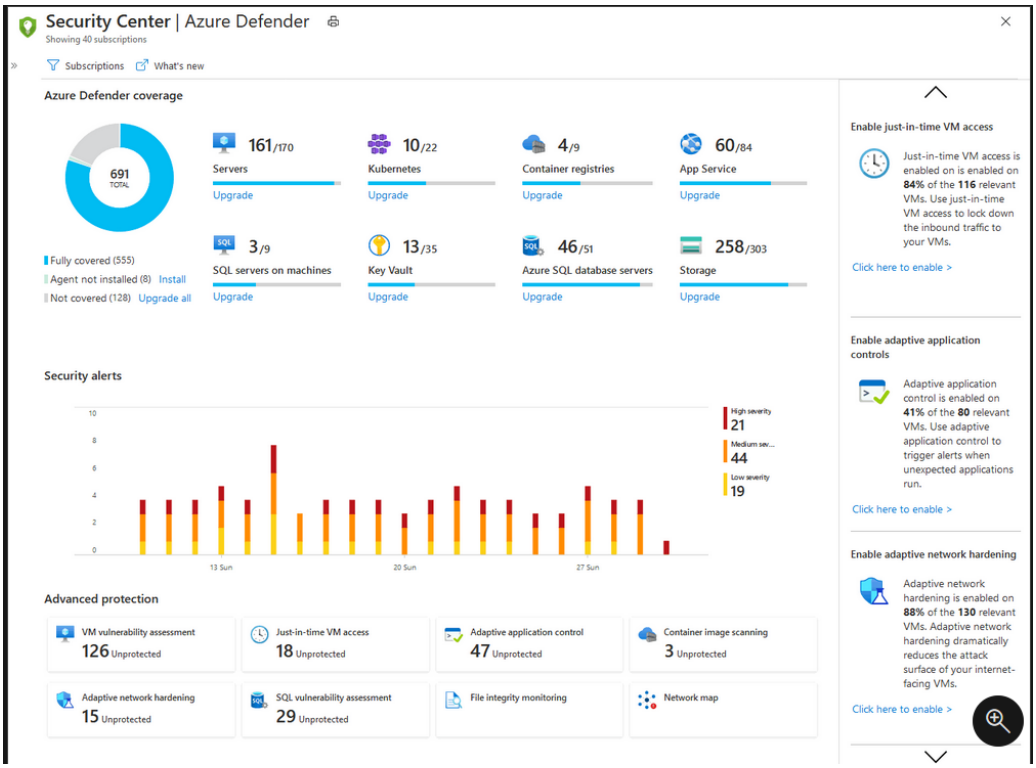


FIGURE 3.27 – Overview d'Azure Defender

Service proposés par *Azure Defender* :

Ressource / Produit	Outil / méthode utilisé	Protège contre
serveurs / *Virtual Machines*	Des audits : auditd, Microsoft Defender pour point de terminaison, Qualys	Ports, registre, trafic entrant malveillant, *deprecated Docker configuration*, attaques sans fichier
App Service	Analyse de journalisation globale en accord avec les tactiques MITRE ATT&CK, Microsoft Threat Intelligence	Les attaques répertoriées dans MITRE ATT&CK (pré-attaque, accès initial, menaces d'exécution), détection des entrée DNS non résolues
Stokage	*Microsoft Threat Intelligence*	L'accès anonyme, les informations d'identification compromises, l'ingénierie sociale, les abus de privilèges et le contenu malveillant.
SQL	Service d'analyse et de détection	*Injection SQL*, requête/accès à la base de donnée anormaux/suspect.
Kubernate	Journaux d'audit, événement de sécurité bruts	Surveillance global au niveau du *cluster Azure Defender* pour Kubernetes (AKS) comme un démon Docker exposé.
Registres de conteneurs	Service d'analyse d'images , Qualys,	Bonnes pratiques diverses.
Key Vault	Service d'analyse et de détection	Tentative dangereuse/anormale d'accès.
Ressource Manager (service de déploiement et de gestion d'Azure)	Service d'analyse et de détection	Opérations de gestion des ressources suspectes, l'utilisation de kits de ressources d'exploitation, du mouvement latéral
DNS	Service d'analyse et de détection,	L'ex-filtration de données depuis vos ressources Azure, la communication entre un programme malveillant et le serveur C&C, la communication avec des domaines malveillants, les attaques DNS.

Quand *Microsoft Defender* détecte une menace, il déclenche une alerte. L'alerte s'affiche dans *Security Center*.

3.2.4 Azure Sentinel

Microsoft Azure Sentinel est une solution *native cloud* et *scalable* de type *SIEM* (Security Information and Event Management) et *SOAR* (*Security Orchestrated Automated Response*) . *Azure Sentinel* assure une analyse de sécurité intelligente et fournit des informations sur les menaces dans l'ensemble de l'entreprise. Elle constitue une solution unique pour la détection des alertes, la visibilité des menaces, la chasse pro active et la réponse face aux menaces.

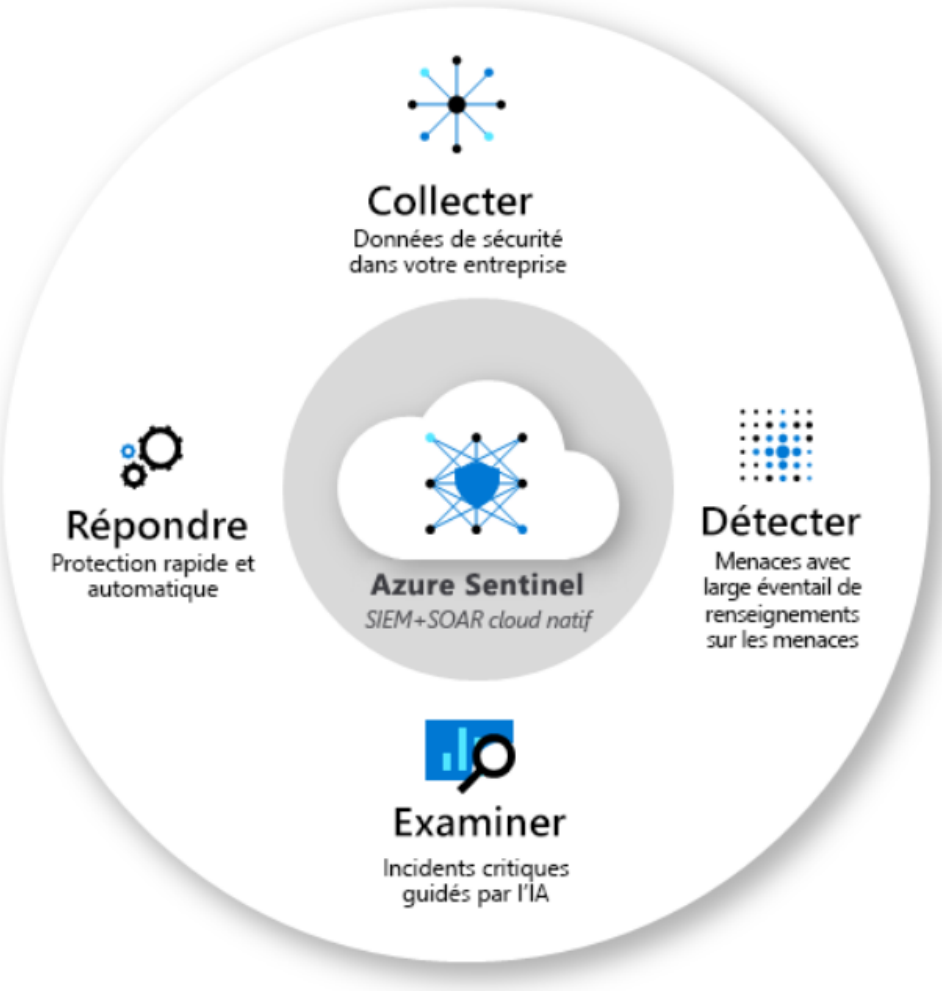


FIGURE 3.28 – Rôles d'Azure Sentinel

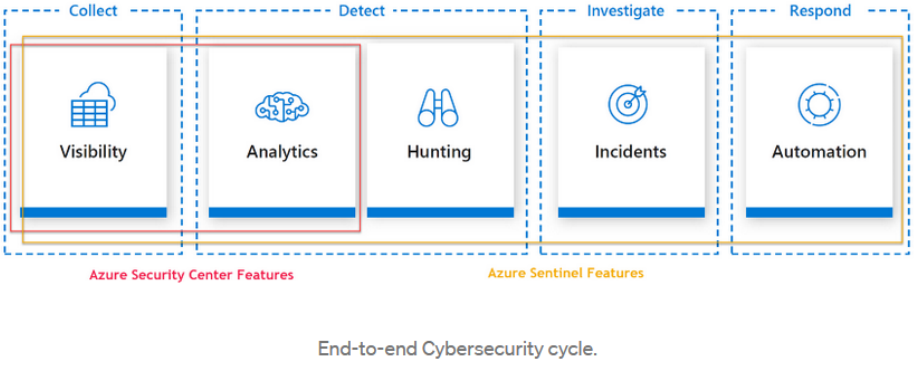


FIGURE 3.29 – Différence entre Azure Sentinel et Azure Security Center

Note : Ne pas implémenter deux fois une *feature*[8] similaire.

8. fonctionnalité

Azure Sentinel offre une vue d'ensemble de l'organisation :

Collecter des données à l'échelle du cloud sur l'ensemble des utilisateurs, appareils, applications et infrastructures, tant locaux que dans de multiples clouds.

Détecter les menaces non détectées précédemment et réduisez les faux positifs en vous appuyant sur l'analytique et les systèmes de renseignement incomparables sur les menaces fournis par Microsoft.

Investiguer les menaces à l'aide de l'intelligence artificielle et recherchez les activités suspectes à grande échelle en profitant des années de travail que Microsoft a consacrées à la cybersécurité.

Répondre aux incidents rapidement avec une orchestration et une automatisation intégrées des tâches courantes.

Automatiser les tâches courantes.

Bibliographie :

Source : cyberedu, module1

Table des figures