ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЗАХИСТУ ІНФОРМАЦІЇ

КАФЕДРА ІНФОРМАЦІЙНОЇ ТА КІБЕРНЕТИЧНОЇ БЕЗПЕКИ

Пояснювальна записка

до магістерської роботи
на тему:

«ТЕХНОЛОГІЇ ЗАХИСТУ ХМАРНОЇ ІНФРАСТРУКТУРИ AMAZON AWS ВІД ВЕБ-ЗАГРОЗ»

Виконав студент 6 курсу, групи БСДМ-61
спеціальності 125 Кібербезпека
освітньо-професійної програми «Інформаційна та кібернетична безпека»
(шифр і назва спеціальності)
Дорохін О.О.
(прізвище та ініціали)
Керівник Марченко В.В.
(прізвище та ініціали)
Рецензент
(прізвище та ініціали)
Нормоконтролер Чумак Н.С.
(прізвище та ініціали)

КИЇВ – 2022

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

Інститут ННІЗІ
Кафедра Інформаційної та кібенетичної безпеки
Ступінь вищої освіти Магістр
Спеціальність 125 Кібербезпека
Освітньо-професійна програма Інформаційна та кібернетична безпека

ЗАТВЕРДЖУЮ
Завідувач кафедри ІКБ
_____ Гайдур Г.І.
"___"_____ 2021 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Дорохіну Оресту Олександровичу
(прізвище, ім'я, по батькові)

1. Тема магістерської роботи: «Технології захисту хмарної інфраструктури Amazon AWS від веб-загроз»

керівник магістерської роботи Марченко Віталій Вікторович, асистент кафедри ІКБ ДУТ
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від « » 2021 року № .

2. Строк подання студентом магістерської роботи 15.12.2021 р.

3. Вихідні дані до магістерської роботи
публічна хмарна інформаційна система AWS;
персональний ноутбук з середовищем віртуалізації Oracle VirtualBox;
ПЗ для відладки й демонстрації на віртальних ОС Oracle VirtualBox;
науково-технічна література,експлуатаційна документація,міжнародні стандарти.

4. Зміст розрахунково-пояснювальної записки
(перелік питань, які потрібно розробити)
1. Аналіз актуальності проблеми захисту активів компаній, що розгорнуті в хмарній інфраструктурі Amazon AWS від веб-загроз.
2. Дослідження характеристик та можливостей засобів захисту активів компанії, що розгорнуті в публічній хмарі, зокрема в Amazon AWS.
3. Надання прикладів реалізації захисту активів різних типів за допомогою інструментів захисту, які можна імплементувати в хмарну інфраструктуру та створення рекомендацій для захисту активів, що розміщені в Amazon AWS.

5. Перелік графічного матеріалу (презентація)

1. Тема магістерської роботи.
2. Об’єкт, предмет, мета та наукові завдання дослідження.
3. Види розміщення комп’ютерних обчислень
4. Шість переваг хмарних обчислень
5. Категорії сервісів, які надає хмара Amazon AWS
6. Модель розподілу відповідальності за захист хмарних ресурсів від AWS
7. Розмежування прав доступу з IAM
8. Amazon GuardDuty
9. Amazon Macie
10. Amazon CloudTrail та Config
11. Amazon Inspector
12. Amazon Secrets Manager
13. Amazon WAF і Shield
14. Висновки за результатами роботи.

6. Дата видачі завдання

27.09.2021 р.

КАЛЕНДАРНИЙ ПЛАН

№ зп	Назва етапів магістерської роботи	Строк виконання етапів магістерської роботи	Примітка
1.	Визначення актуальності проблеми захисту активів компанїй, що розгорнуті в хмарній інфраструктурі Amazon AWS від веб-загроз	27.09.2021 р.	
2.	Аналіз наукової та технічної літератури з питань теми магістерської роботи.	11.10.2021 р.	
3.	Аналіз методів та засобів захисту активів в хмарі.	25.10.2021 р.	
4.	Розроблення прикладів побудови захищеної хмарної інфраструктури з використанням нативних сервісів Amazon AWS та підключаємих засобів перевірених третьосторонніх вендорів.	15.11.2021 р.	
5.	Розроблення рекомендацій щодо захисту активів компаній в хмарній інфраструктурі Amazon AWS.	29.11.2021 р.	
6.	Оформлення результатів дослідження. Проходження плагіату	05.12.2021 р.	
7.	Підготовка презентації до захисту.	15.12.2021 р.	

Студент _____ Дорохін О.О.
(підпис) прізвище та ініціали

Керівник магістерської роботи _____ Марченко В.В.
(підпис) прізвище та ініціали

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ
ПОДАННЯ
ГОЛОВІ ДЕРЖАВНОЇ ЕКЗАМЕНАЦІЙНОЇ КОМІСІЇ
ЩОДО ЗАХИСТУ МАГІСТЕРСЬКОЇ РОБОТИ

Направляється студент Дорохін О.О. до захисту магістерської роботи
(прізвище та ініціали)
спеціальності 125 Кібербезпека
освітньо-професійної програми Інформаційна та кібернетична безпека
(шифр і назва спеціальності)
на тему: «Технології захисту хмарної інфраструктури Amazon AWS від веб-загроз»

Магістрська робота і рецензія додаються.

Директор інституту _____ Савченко В.А.
(підпис) (прізвище та ініціали)

Довідка про успішність

Дорохін О.О. за період навчання в інституті
(прізвище та ініціали студента)
ННІЗІ з 2020 року по 2022 рік повністю виконав навчальний план за напрямом підготовки, спеціальністю з таким розподілом оцінок за:
національною шкалою: відмінно ____%, добре ____%, задовільно ____%;
шкалою ECTS: A ____%; B ____%; C ____%; D ____%; E ____%.

Секретар інституту, факультету (відділення) _____ Черниш О.В.
(підпис) (прізвище та ініціали)

Висновок керівника магістерської роботи

Студент ***Дорохін О.О.*.** обрав тему роботи, метою якої було дослідити зміст технології захисту хмарної інфраструктури Amazon AWS від веб-загроз та створити приклади реалізації захисту активів різних типів за допомогою інструментів захисту, які можна імплементувати в хмарну інфраструктуру й розробити рекомендацій для захисту активів, що розміщені в Amazon AWS. Перелік використаних джерел свідчить про вміння магістром розбиратись в наукових питаннях та застосовувати їх при дослідженнях. Під час виконання магістерської роботи Дорохін О.О. показав гарну теоретичну та практичну підготовку, вміння самостійно вирішувати питання і робити висновки. Роботу виконував сумлінно, акуратно та вчасно за планом.

Все це дозволяє оцінити виконану магістрську роботу студента Дорохіна Ореста Олександровича на оцінку « » та присвоїти йому кваліфікацію 2149.2. Професіонал з організації інформаційної безпеки, викладач закладу вищої освіти.

Керівник магістерської роботи _____ Марченко В.В.
(підпис) (прізвище та ініціали)
“____” _____ 2021 року

Висновок кафедри про магістрську роботу

Магістрська робота розглянута. Студент Дорохін О.О.
(прізвище та ініціали)
допускається до захисту даної магістерської роботи в Державній екзаменаційній комісії
Завідувач кафедри Інформаційної та кібернетичної безпеки
(назва)

_____ Гайдур Г.І.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина магістерської роботи: 88 сторінок, 65 рисунків, 4 таблиці, 13 джерел.

Об'єкт дослідження – процес забезпечення захисту хмарної інфраструктури Amazon AWS від веб-загроз.

Предмет дослідження – технологія управління захистом хмарної інфраструктури Amazon AWS від веб-загроз.

Мета роботи – розробити приклади та рекомендації до захисту хмарної інфраструктури Amazon AWS від веб-загроз на базі нативних рішень AWS і підключаємих рішень від перевірених третіх сторін.

Методи дослідження – опрацювання літератури за даною темою, практичне використання програмного забезпечення що описується в роботі для кращого розуміння його поведінки й коригування на основі спостережень, аналіз експлуатаційної документації, міжнародних стандартів та їх порівняння.

В роботі проведено аналіз проблеми забезпечення кібербезпеки хмарної інфраструктури Amazon AWS та визначені мета та завдання управління захистом активів компаніх в хмарній інформаційній системі. Проаналізовано існуючі нативні засоби захисту активів в хмарі від Amazon AWS та розглянуто потенціал застосування підключаємих засобів захисту та моніторингу від перевірених виробників (третіх сторін) для закриття потреб кібербезпеки в повному обсязі.

Досліджено методи та засоби управління захистом хмарної інфраструктури Amazon AWS від веб-загроз. Визначено призначення, основні функції та склад засобів захисту, логування, аудиту та моніторингу, що присутні в Amazon AWS.

На основі досліджень проведених в роботі розроблено приклади побудови захисту з використанням AWS CloudTrail, AWS Config, AWS WAF, AWS Shield, AWS GuardDuty, AWS CloudWatch, AWS Inspector, AWS Macie, AWS Secrets Manager, AWS IAM, що можуть використовуватись в якості шаблонів та рекомендації до захисту хмарної інфраструктури Amazon AWS від веб-загроз на базі нативних рішень AWS і підключаємих рішень від перевірених третіх сторін.

Галузь використання – кібербезпека хмарної інформаційної системи.

ХМАРНА ІНФОРМАЦІЙНА СИСТЕМА, КІБЕРБЕЗПЕКА, AWS, АУДИТ В ХМАРНІЙ ІНФРАСТРУКТУРІ, МЕТОДИ ТА ЗАСОБИ УПРАВЛІННЯ ЗАХИСТОМ ХМАРНОЇ ІНФРАСТРУКТУРИ, ТЕХНОЛОГІЯ УПРАВЛІННЯ ЗАХИСТОМ ХМАРНОЇ ІНФРАСТРУКТУРИ

ABSTRACT

Master’s thesis: 88 pages, 65 figures, 4 tables, 13 sources.

Object of research – the process of ensuring the protection of Amazon AWS cloud infrastructure from web threats.

Subject of research – the technology for managing the protection of Amazon AWS cloud infrastructure from web threats.

The aim of research – to develop examples and recommendations for protecting Amazon AWS cloud infrastructure from web threats based on native AWS solutions and connectable solutions from proven third parties.

Research methods – elaboration of literature on this topic, practical use of the software described in the work for a better understanding of its behavior and correction based on observations, analysis of operational documentation, international standards and their comparison.

The paper analyzes the problem of cyber security of Amazon AWS cloud infrastructure and identifies the purpose and objectives of managing the protection of companies' assets in the cloud information system. The existing native cloud asset protection tools from Amazon AWS are analyzed and the potential of using security and monitoring tools from proven vendors (third parties) to fully cover cybersecurity needs considered.

The paper studies methods and tools for managing the protection of Amazon AWS cloud infrastructure from web threats. The purpose, main functions and composition of security, logging, auditing and monitoring tools present in Amazon AWS are defined.

Based on the research conducted in the work, examples of building protection using AWS CloudTrail, AWS Config, AWS WAF, AWS Shield, AWS GuardDuty, AWS CloudWatch, AWS Inspector, AWS Macie, AWS SecretsManager, AWS IAM that can be used as templates and guidelines for protecting Amazon AWS cloud infrastructure from web threats based on native AWS solutions and connectable solutions from trusted third parties.

Field of use – cybersecurity of cloud information system.

CLOUD INFORMATION SYSTEM, CYBER SECURITY, AWS, AUDIT IN THE CLOUD INFRASTRUCTURE, METHODS AND MEANS OF MANAGEMENT OF PROTECTION OF CLOUD INFRASTRUCTURE

ЗМІСТ

Стор.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

IT	–	Інформаційні технології
IoT	–	Internet of Things
IDS	–	Intrusion Detection System
IPS	–	Intrusion Prevention System
DNS	–	Domain Name System
KMS	–	Key Management Service
HA	–	High Availability
WAF	–	Web Application Firewall
NAT	–	Network Address Translation
HTTP	–	Hypertext Transfer Protocol
OSI	–	Open Systems Interconnection
Ln	–	Layer n (наприклад L1 – перший рівень по моделі OSI)
XSS	–	Cross site scripting
CSRF	–	Cross site request forgery
SQLi	–	Structured Query Language injection
CPU	–	Central Processing Unit
LAN	–	Local Area Network
AD	–	Active Directory
TCP	–	Transmission Control Protocol
UDP	–	User Datagram Protocol
DDoS	–	Distributed Denial of Service
DoS	–	Denial of Service
SSL	–	Secure Socket Layer
TLS	–	Transport Layer Security
WCU	–	Web ACL Capacity Units
ISO	–	International Organization for Standardization
NIST	–	National Institute of Standards and Technology

SaaS – Software as a Service

PaaS – Platform as a Service

IaaS – Infrastructure as a Service

IaC – Infrastructure as Code

AWS – Amazon Web Services

EC2 – Elastic Compute Cloud

ECS – Elastic Container Service

EKS – Elastic Kubernetes Service

API – Application Programming Interface

VPC – Virtual Private Cloud

S3 – Simple Storage Service

ELB – Elastic Load Balancer

ARN – Amazon Resource Name

TI – Threat Intelligence

NVD – National Vulnerability Database

SNS – Simple Notification Service

SRT – Shield Response Team

SIEM – Security Information and Event Management

ЦОД – центр обробки даних

ШПЗ – шкідливе програмне забезпечення

ОС – операційна система

RAM – Random Access Memory

НСД – несанкціонований доступ

ПЗ – програмне забезпечення

ВСТУП

Актуальність дослідження. Хмарні обчислення існують приблизно два десятиліття, і, незважаючи на дані, що вказують на ефективність бізнесу, вигоду та переваги в конкурентній боротьбі, які вони мають, значна частина бізнес-спільноти продовжує працювати без них. Згідно з дослідженням International Data Group, 69% компаній вже використовують хмарні технології в тій чи іншій якості, а 18% кажуть, що планують колись впровадити рішення для хмарних обчислень – повідомляє журнал “Forbes” на 2015 рік. [1]

У той же час Dell повідомляє, що компанії, які інвестують у великі дані, хмарні обчислення, мобільність та безпеку, отримують до 53% швидшого зростання доходів, ніж їхні конкуренти. Як показують ці дані, дедалі більше технічно підкованих компаній та лідерів галузі визнають численні переваги тенденції хмарних обчислень. Але більше того, вони використовують цю технологію, щоб ефективніше керувати своїми організаціями, краще обслуговувати своїх клієнтів і різко збільшити загальний прибуток. [2]

Хмарні обчислення набули широкого поширення протягом останніх кількох років. З експоненційним збільшенням використання даних, яке супроводжує перехід суспільства до цифрового 21-го століття, окремим особам і організаціям стає все важче підтримувати всю важливу інформацію, програми та системи на внутрішніх комп’ютерних серверах. . Рішення цієї проблеми існує майже так само довго, як Інтернет, але лише нещодавно набув широкого застосування для бізнесу.

Хмарні обчислення працюють за таким же принципом, як email, доступ до якого отримується через браузер й ПЗ якого не вимагає встановлення. Це дозволяє користувачам отримувати доступ до всіх функцій і файлів системи, не зберігаючи основну частину цієї системи на власних комп’ютерах. Насправді, більшість людей вже користуються різноманітними послугами хмарних обчислень, навіть не усвідомлюючи цього. Gmail, Google Drive, TurboTax і навіть Facebook і Instagram – це хмарні Software as a Service (SaaS) програми. Для всіх цих послуг користувачі надсилають свої особисті дані на сервер, розміщений у хмарі, який зберігає

інформацію для подальшого доступу. Ці програми корисні не тільки для особистого використання, вони ще більш цінні для компаній, яким потрібно мати доступ до великих обсягів даних через безпечне мережеве з'єднання в Інтернеті. Співробітники можуть отримати доступ до інформації про клієнтів за допомогою SaaS CRM, такої як Salesforce, зі свого смартфона вдома або під час подорожі, і швидко ділитися цією інформацією з іншими авторизованими сторонами в будь-якій точці світу, розробники можуть мати розгорнуті в хмарі CI/CD пайплайни, не турбуючись про підтримку центрів обробки даних (ЦОД) і зосередившись на головному.

Питання, що виходить на передній план вже зараз – це захист персональних даних мільйонів людей, які користуються сервісами, розміщеними в хмарах чисельних провайдерів хмарних обчислень: AWS, GCP, Microsoft Azure, Alibaba Cloud та інші. Кожен провайдер хмарних послуг пропонує нативні засоби для:

- захисту хмарної інфраструктури компанії від мережевих атак, атак L4-L7 рівня за моделлю Open System Interconnections (OSI), зокрема на API-інтерфейси, DoS/DDoS атак;
- логування, моніторингу й візуалізації подій й мережевих потоків трафіку;
- розгортання ресурсів з усіма врахуваннями норм відповідності міжнародним нормативним документам і стандартам, як PCI DSS чи ISO, що потребує слідуванню найкращих практик й цілком залежить від фахових умінь архітекторів хмарних рішень виконувати роботу слідуючи цим практикам.

В даній роботі увага приділяється організації захисту ресурсів в Amazon AWS, одного з найбільших хмарних провайдерів, використання якого компаніями України ще не помітне, а отже має багато білих плям.

Ступінь наукової розробки. На сьогодні хмарні технології в вітчизняному кіберпросторі не мають широкого розповсюдження, проте все більше спеціалістів, в тому числі в кібербезпеці, беруть участь в роботі над побудовою захищеної хмарної інфраструктури за замовленнями іноземних команій. В роботах [3]-[6] розглянуті тільки декілька основних моделей обслуговування хмарних

технологій, й існує великий потенціал для поглиблення в тему захищеності хмарних технологій. Оскільки моделі обслуговування хмарних технологій мають широкий спектр аспектів, дана робота направлена саме на роль інформаційної та кібербезпеки, в ній *вперше одержано* певний структурований перелік прикладів та рекомендацій щобо забезпечення захисту хмарної інфраструктури веб-додатків в Amazon AWS.

Практичне значення одержаних результатів полягає у розробці прикладів та рекомендацій щодо захисту хмарної інфраструктури Amazon AWS від веб-загроз на базі нативних рішень AWS і підключаємих рішень від перевірених третіх сторін.

1 АНАЛІЗ ПРОБЛЕМИ ПОБУДОВИ ЗАХИЩЕНОЇ ХМАРНОЇ ІНФРАСТРУКТУРИ ВЕБ-ДОДАТКІВ В AMAZON AWS ВІД ВЕБ-ЗАГРОЗ

1.1. Передумови виникнення потреби в хмарних обчисленнях

1.1.1. Стисла хронологія зародження й розвитку хмарних обчислень

Хмарні технології зародилися у 1950-х роках, коли вчені вперше заговорили про концепцію поділу часу. Полягала вона в наступному: комп'ютери коштували дуже дорого, тому купити їх усім співробітникам було неможливо — проте замість цього кілька людей могли одночасно підключатися до спільного процесора. Ця ідея з'явилася 1954 року, її реалізація почалася 1959-го, а перше комерційно успішне рішення випустили 1964-го. Ставлення до обчислювальної потужності як до ресурсу, подібного до електрики та води, призвело до появи комп'ютерних бюро, де клієнти могли купувати необхідний обсяг потужності для виконання розрахунків. Ця модель функціонувала до 1980-х років - тоді з'явилися дешеві персональні комп'ютери, і вона втратила актуальність. [7]

Другим важливим фактором, що вплинув на сучасні хмари, є можливість підключення до глобальної мережі. Це основний принцип технології: користувачі повинні мати доступ до сервісів із будь-якої точки світу. Перші процесори та його користувачі, зазвичай, перебували у одному будинку. Локальні мережі працювали в США вже до кінця 1950-х років, а 1960 року вчений Джозеф Карл Робнетт Ліклайдер запропонував створити з обчислювальних центрів глобальну мережу. 1962 року він очолив проект зі з'єднання мереж Міністерства оборони США, Гірського комплексу Шайєнн (бункер у штаті Колорадо — прим. ред.) та Стратегічного командування ВПС США.

У 1966 році почався розвиток ARPANET, більшого проекту, ядро якого на початку 1990-х еволюціонувало до сучасного інтернету. Нова мережа

розвивалася, сервіси, що працювали в ній, залучали все більше користувачів, а отже, вимагали все більше обчислювальних потужностей. Історія вийшла на друге коло.

Третій значимий фактор в історії хмарних технологій — це віртуалізація: користувачам необхідні цифрові системи, які не залежать від конкретного обладнання та дозволяють починати та закінчувати роботу будь-якої миті. Вперше цю концепцію експериментально впровадили ще 1966 року, а комерційний варіант 1972 року представила IBM. Сучасні функції віртуалізації x86 були додані до процесорів Intel у 2005 році (VT-x) та до процесорів AMD у 2006 році (AMD-V). Важко сказати, хто і коли запровадив термін «хмара». З розвитком інтернету поширення набули онлайн-сервіси — їх почали називати SaaS, щоб відрізняти від десктопних додатків, які потрібно встановлювати на комп'ютер. Інтернет-бум мав два важливі наслідки. По-перше, швидко зростала кількість розробників, тому потрібно було спростити процес розміщення нових програм. Так народилася ідея PaaS (Platform as a Service - "платформа як послуга"). Першим таким сервісом став Zimki, запущений у 2006 році. У 2008 році Google представила App Engine, який пізніше став хмарною платформою Google. По-друге, деякі інтернет-компанії стали дуже великими і мали велику кількість обчислювальних потужностей. Вони були потрібні їм у пікові моменти, наприклад, інтернет-магазинам — під час розпродажів у «чорну п'ятницю». Однак більшу частину часу весь обсяг потужностей був не потрібен, і бізнес став передавати їх третім сторонам – це призвело до створення IaaS (Infrastructure as a Service – інфраструктура як послуга).

Amazon Web Services став першим IaaS-сервісом або хмарою в сьогоднішньому розумінні. Microsoft запустила аналогічний сервіс Azure у 2010 році, а Google – Google Compute Engine у 2012 році. Інші компанії незабаром усвідомили потенціал хмарних технологій і приєдналися до гонки, але Amazon, Microsoft та Google, як і раніше, значно їх випереджають. [8]

В даній роботі акцент зроблено на IaaS від Amazon Web Services й всі практичні дослідження проводяться з даним хмарним провайдером.

1.1.2. Відмінність cloud computing від класичної on-premise архітектури

Для розуміння причини виникнення хмарних обчислень слід розглянути переваги даної технології, що відрізняють хмарні обчислення від класичної побудови IT-інфраструктури компанії через створення власних ЦОД, незалежно від їх розмірів та територіального розповсюдження. Це розповсюдження зачасту має місце через виникнення віддалених офісів компанії, для досягнення відмовостійкості, мінімальної latency в наданні веб-контенту клієнтам чи з будь-якої іншої причини і в свою чергу збільшує витрати на підтримку працездатності описаної інфраструктури робочих серверів.

Види розміщення комп'ютерних обчислень:

- cloud – повне розгортання в хмарі, всі частини програми працюють у хмарі. Програми першочергово сворюються в хмарі або повністю мігруються з існуючої інфраструктури, щоб скористатися перевагами хмарних обчислень;
- hybrid – це спосіб об'єднати інфраструктуру та програми між хмарою чи декількома хмарами з наявними ресурсами, які не розташовані у власному чи орендованому ЦОД. Найпоширеніший метод гібридного розгортання відбувається між хмарою та існуючою локальною інфраструктурою для розширення функціоналу інфраструктури організації, наприклад використовується лише сервіс баз даних (data warehousing сервіс, якщо бути більш точним) AWS Redshift для зберігання великих об'ємів даних поза власним ЦОД;
- on-premises – розгортання ресурсів локально з використанням інструментів віртуалізації та управління ресурсами іноді називають «приватною хмарою» або «private cloud». Внутрішнє розгортання не дає багатьох переваг хмарних обчислень, але іноді його шукають через його здатність надавати виділені ресурси.

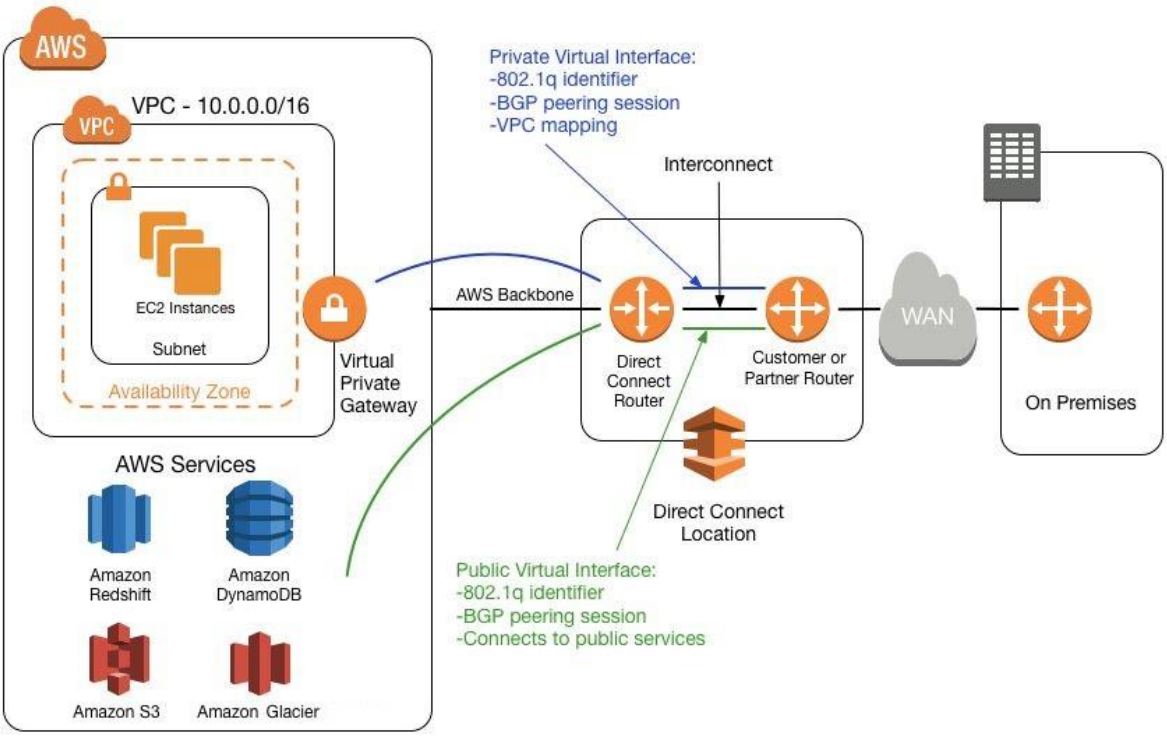


Рис. 1.1. Приклад Hybrid deployment від AWS

На зображенні є декілька назв, специфічних для хмари AWS, що потребують пояснення для розуміння гібридного розгортання ресурсів (на незначних назвах зупинятись не будемо, щоб не ускладнювати розуміння суті):

- Amazon Virtual Private Cloud (Amazon VPC) – дає змогу запускати ресурси AWS у визначеній віртуальній мережі. Ця віртуальна мережа дуже нагадує традиційну мережу, якою керують у власному ЦОД, з перевагами використання масштабованої інфраструктури AWS;
- availability zone – ізольована локація регіону в AWS, що називається зоною доступності. При створенні EC2 інстансу (аналог віртуальної машини чи виділеного серверу в AWS) є можливість вибрати зону доступності або дозволити AWS вибрати її самостійно. Якщо розподіляти свої EC2 інстанси між кількома зонами доступності, і один екземпляр вийде з ладу, конфігураційно можна зробити так, щоб екземпляр в іншій зоні доступності брав все навантаження на себе;
- Virtual Private Gateway – концентратор VPN на стороні Amazon VPN-з’єднання Site-to-Site. На стороні AWS використовується даний віртуальний

приватний шлюз як шлюз для VPN-з’єднання Site-to-Site з клієнтським шлюзом, що випагає в свою чергу налаштування маршрутизатора клієнта ;

- AWS Direct Connect – з’єднує внутрішню мережу з місцем розташування AWS Direct Connect за допомогою стандартного волоконно-оптичного кабелю Ethernet. Один кінець кабелю під’єднано до маршрутизатора, інший – до маршрутизатора AWS Direct Connect. За допомогою цього підключення стає можливим підключати віртуальні інтерфейси безпосередньо до загальнодоступних служб AWS (наприклад, до Amazon S3) або до Amazon VPC, Розташування AWS Direct Connect надає доступ до AWS у регіоні, з яким він пов’язаний. Можна використовувати одне з’єднання в загальнодоступному регіоні для доступу до загальнодоступних служб AWS у всіх інших загальнодоступних регіонах.

Постає запитання переваг використання хмарним обчислень, зупинимось на них.

Шість переваг хмарних обчислень:

- «Trade capital expense for variable expense» - Замість того, щоб витрачати значні кошти на створення ЦОД (Центру Обробки Даних), з хмарною інфраструктурою ви можете платити лише тоді, коли споживаєте обчислювальні ресурсів і платити лише за ту кількість, яку споживаєте;
- «Benefit from massive economies of scale» – Провайдери хмарних послуг, як AWS, можуть досягти більш високої економії на масштабі, що перетворюється на нижчі ціни для користувачів хмари відносно затрат, які мають місце при самостійному створенні ЦОД;
- «Stop guessing capacity» – Не потрібно вгадувати наперед потреби вашої інфраструктури. Закупаючи сервери потрібно розраховувати приблизні навантаження й закладати в розрахунки резервні обчислювальні ресурси. З хмарними обчисленнями ці проблеми зникають. Можна отримати доступ до такої кількості, скільки потрібно для конкретної задачі, і масштабувати горизонтально чи вертикально при необхідності за хвилини;

- «Increase speed and agility» - у середовищі хмарних обчислень нові IT -ресурси додаються за один клік, що скорочуєте час на надання ресурсів розробникам від тижнів до лічених хвилин. Це призводить до зростання швидкості розробки;
- «Stop spending money running and maintaining data centers» - зосередьтеся на проектах, які відрізняють ваш бізнес, замість підтримання інфраструктури. Хмарні обчислення дозволяють скоріше зосередитися на власних клієнтах а не на обслуговуванні серверів;
- «Go global in minutes» - легко розгортайте свої додаткив кількох геолокаційних регіонах по всьому світу за декілька кліків. [9]

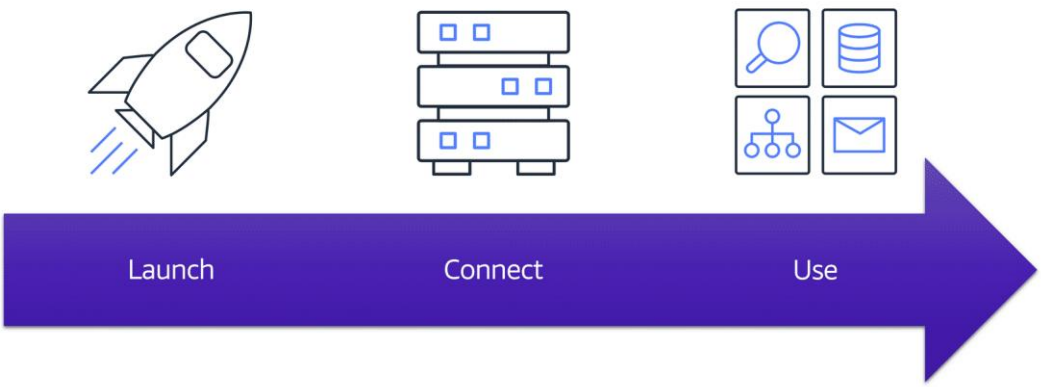


Рис. 1.2. Переваги хмарних обчислень

1.2. Сучасні тенденції переходу на хмарні обчислення

2019 був прекрасним роком для хмарних сервісів. Навіть компанії, що займаються питаннями безпеки (у тому числі Пентагон), урядові організації, охорона здоров'я, банки, страхові компанії рухаються в напрямку хмарного зберігання даних і обчислень. Ця тенденція збереглась і в 2020 році. Gartner опублікував прогноз, що ринок хмарних сервісів за збереження тренду зросте в 2021 р. ще на 18%, досягнувши 304.9 мільярда доларів.

Таблиця 1.1.

Прогноз витрат кінцевих користувачів загальнодоступних хмарних сервісів у всьому світі (мільйони доларів США) від Gartner [10]

	2019	**2020**	**2021**	**2022**
Cloud Business Process Services (BPaaS)	45,212	44,741	47,521	50,336
Cloud Application Infrastructure Services (PaaS)	37,512	43,823	55,486	68,964
Cloud Application Services (SaaS)	102,064	101,480	117,773	138,261
Cloud Management and Security Services	12,836	14,880	17,001	19,934
Cloud System Infrastructure Services (IaaS)	44,457	51,421	65,264	82,225
Desktop as a Service (DaaS)	616	1,204	1,945	2,542
Total Market	**242,696**	**257,549**	**304,990**	**362,263**

Безперечно ці дані свідчать про позитивні тенденції розвитку ринку хмарних сервісів, що лише зростали в період пандемії COVID-19. Варто зазначити, що прогнозовано компанії готові витрачати на 2022 рік частку бюджету на упраіння та безпеку сервісів в хмарах, яка сягне сумарно майже 20 мільйонів доларів США.

Сьогодні великий бізнес інвестує в перспективні технології, що розвиваються, які здатні радикально змінити життєдіяльність людей і виробничі процеси. Але на такі технології націлений не тільки великий IT бізнес, а й техностартапи.

Експерти виділяють низку ніш, які мають найбільший потенціал для стартапів. Серед найбільш затребуваних можна виділити штучний інтелект та машинне навчання, віртуальних помічників та інтернет-речей (IoT), віртуальну реальність та нейронні мережі, блокчейн-технології та діалогові системи, додатки та сервіси Mesh (MASA), адаптивну архітектуру інформаційної безпеки та цифрових двійників.

Базою для сучасних технологій є хмарні обчислення, при цьому хмари потрібні не тільки на етапі експлуатації продукту, але і на етапах розробки та тестування. Тому що найважливішою перевагою хмар для стартапів є багаторазова економія коштів на розробку та впровадження цифрових продуктів. Саме тому cloud only стратегія є єдино можливою для стартапів, вона полегшує і прискорює запуск продуктів, дозволяє не занурюватися особливо в архітектури платформ, віртуальних машин і не думати про оновлення та адміністрування. Знижуються ризики відмов через людські помилки, однотипні завдання автоматизовані, виправлення та оновлення відбуваються автоматично в реальному часі. Крім цього, не потрібні інвестиції та реінвестиції у фізичну IT-інфраструктуру, розширення стартап-команди за рахунок обслуговуючих цю інфраструктуру персоналу. При цьому під різні середовища розробки можна використовувати різні віртуальні платформи, споживати більше або скоротити споживання хмарних ресурсів, і оплачувати тільки те, що було використано за моделлю Pay as you go. Стартап-команди запускають програми там, де це вигідніше, не прив'язуючись до конкретного хмарного провайдера.

Хмарні рішення дозволяють стартапам масштабувати свої проекти в стислі терміни, що дає певні переваги при переході від пілотних проектів до експлуатації продукту. При цьому можна використовувати налаштовані можливості хмарних сервісів, за піками навантаження. Такий Cloud-native підхід робить доступними всі останні розробки, включаючи штучний інтелект, нейронні мережі, машинне навчання, скоротивши час на обробку та аналіз даних.

Cloud-native підхід використовує переваги хмарної моделі для створення та розгортання програмних продуктів та додатків, які є набором мікросервісів, розміщених у контейнерах і керованих хмарною платформою, що надає масштабовані обчислювальні потужності. Cloud-native програми, створені для надання за хмарною моделлю, більш відмовостійкі, їх можна швидше розгортати, що загалом дозволяє швидше виводити на ринок програмні продукти, а також тестувати нові ідеї під запити клієнтів, тобто можна сказати, що це в широкому розумінні інструмент розвитку бізнесу.

На закінчення можна сказати, що без доступу до хмарних технологій безліч техностартапів не змогли б реалізувати свої ідеї, оскільки вони знижують вартісний поріг входу в нішу.

1.3. Вплив розвитку підходів IT-розробки на зріст використання хмарних обчислень

Основну частину підрозділу присвячено питанню виходу мікросервісної архітектури додатків вперед на фоні громіздких монолітних рішень ПЗ.

Завдяки тому, що хмари вміють еластично підлаштовуватися під потреби розробника, це теоретично спрощує ще одне завдання – проблему масштабування додатків. На жаль, завдання масштабування додатків не є лінійним. Щоб програма справлялася з величезними навантаженнями в періоди пікової відвідуваності (або обчислень), недостатньо просто давати йому додаткову пам'ять та процесорні потужності. Абсолютно кожен традиційний додаток має поріг, після якого він уже не в змозі «перетравити» нові ресурси і продемонструвати зростання продуктивності. Проблема у разі полягає не в ресурсах, а самій архітектурі більшості програм.

Особливо гостро ця проблема стоїть для додатків з монолітною архітектурою, які фактично є єдиними бінарними файлами. Переваги такого підходу очевидні: монолітні програми досить прості та лінійні. Усі сценарії

поведінки користувача можна передбачити, відстежити і за необхідності зробити налагодження бага. Проте така простота має ціну:

- по-перше, це вже згадані вище проблеми із масштабуванням. У якийсь момент навіть найпродуманіший монолітний додаток перестає працювати ефективніше від апгрейду конфігурації сервера на якому виконується;
- по-друге, монолітний додаток не так просто перенести на нові сервери і для цього може знадобитися повна перекомпіляція програми;
- по-третє, таку програму складно підтримувати та розвивати. Будь-яке оновлення вимагає повного білду всієї програми, і помилка в одному з блоків коду може призвести до падіння всієї системи.

У пошуках ідей, як вирішити ці проблеми, була розроблена інша концепція – service-oriented architecture (SOA). Вона має на увазі, що програма розділена на кілька модулів, кожен з яких надає іншим якусь функціональність. модулі взаємодіють через набір веб-служб, і незалежно один від одного можуть звертатися до єдиної або до власних баз даних. Такий підхід справді спрощує підтримку програми і не перетворює її оновлення на роботу сапера, в якій немає права на помилку; але й має свої недоліки. Ключовий із них – проблеми з масштабуванням розробки таких додатків. У міру зростання програми нові функції стає все складніше «вкладати» в спочатку затверджені архітектором 5-10 пакетів. Їх число стає дедалі більшим, що обертається проблемами з підтримкою.

Мікросервіс як елемент еволюції програми.

Результатом еволюції SOA стала ідея мікросервісної архітектури, яка використовується під час конструювання хмарних додатків. Концептуально ідеї обох підходів дуже схожі, і деякі архітектори навіть не виділяють мікросервісну архітектуру в окрему парадигму, вважаючи її окремим випадком SOA. Мікросервісна архітектура має на увазі, що додаток складається не з якоїсь невеликої кількості великих модулів, а з безлічі незалежних частин. На відміну від моноліту, у мікросервісному додатку можна використовувати різні способи

взаємодії компонентів між собою. Система не має єдиного, заздалегідь визначеного стану. Натомість кожен компонент працює «за ситуацією»: щойно йому надходить подія він починає роботу. Це дозволяє робити дуже гнучку та незалежну архітектуру. При цьому кількість сервісів у мікросервісному додатку постійно змінюється – якісь додаються, якісь видаляються. У новому підході можна будь-який мікросервіс замінити і замість нього вбудувати ланцюжок мікросервісів. Інші послуги продовжують стабільно працювати, тому що не пов'язані безпосередньо між собою. Такою є природна еволюція програми. Завдяки цьому у розробників та архітекторів з'являється можливість швидко щось змінювати, щоб реагувати на зміни бізнес-вимог та випереджати конкурентів.

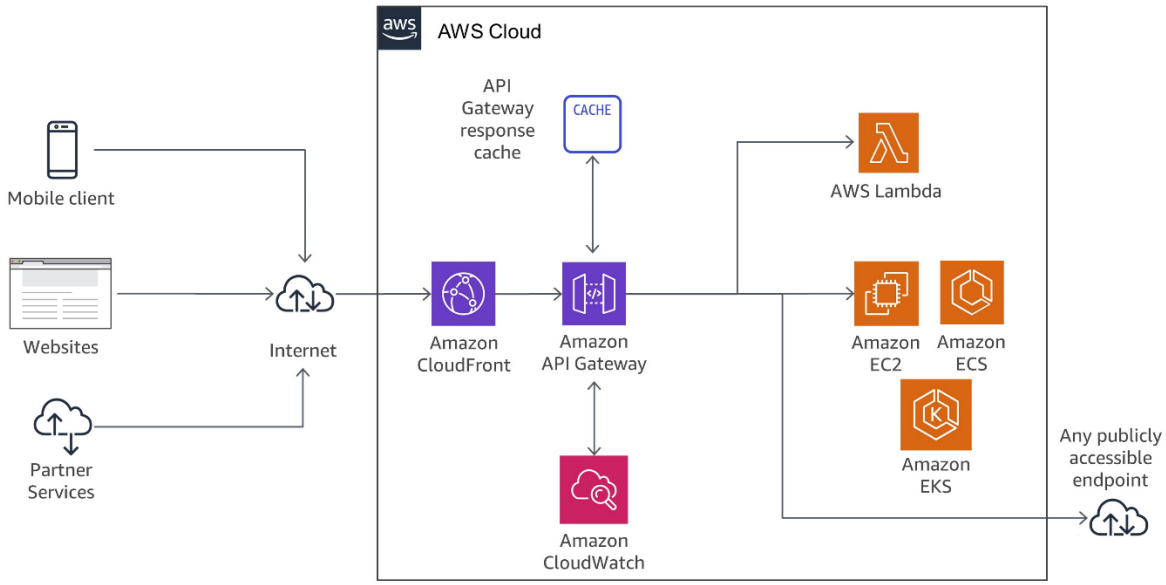


Рис. 1.3. Приклад мікросервісної архітектури реалізованої сервісами AWS

На зображенні 1.2 можна зустріти ряд абревіатур та найменувань сервісів, специфічних для хмари AWS, розглянемо їх значення:

- Amazon CloudFront — це веб-сервіс, який пришвидшує розповсюдження статичного та динамічного веб-вмісту, такого як .html, .css, .js та файли зображень. CloudFront доставляє веб-контент через всесвітню мережу ЦОД, які називаються «edge locations». Коли користувач робить запит на контент, який ви обслуговуєте за допомогою CloudFront, запит спрямовується до

найближчої edge location, яка забезпечує найнижчу затримку (часову затримку), щоб вміст доставлявся з найкращою можливою продуктивністю;

- Amazon API Gateway — це служба AWS для створення, публікації, підтримки, моніторингу та захисту API REST, HTTP і WebSocket у будь-якому масштабі. Розробники Application Programming Interface (API) можуть створювати API для доступу до AWS або інших веб-сервісів, а також до даних, що зберігаються в хмарі AWS. Як розробник шлюзу API, ви можете створювати API для використання у своїх власних клієнтських програмах. Або ви можете зробити свої API доступними для сторонніх розробників додатків;
- Amazon CloudWatch відстежує ресурси AWS і програми, які викогнуються в AWS, у режимі реального часу. Можна використовувати CloudWatch для збору та відстеження показників, які постійно змінюються, і які можна вимірювати для своїх ресурсів і програм;
- AWS Lambda — це обчислювальний сервіс, який дозволяє запускати код без створення або керування серверами. Lambda запускає наданий код на високодоступній обчислювальній інфраструктурі та виконує все адміністрування обчислювальних ресурсів, включаючи обслуговування сервера та операційної системи, надання потужності та автоматичне масштабування, моніторинг коду та ведення журналів. За допомогою Lambda можливо запускати код практично для будь-якого типу програми або серверної служби. Все, що потрібно зробити, це надати свій код однією з мов, які підтримує Lambda.
- Amazon Elastic Compute Cloud (Amazon EC2) забезпечує масштабовані обчислювальні потужності в хмарі Amazon Web Services (AWS). Використання Amazon EC2 позбавляє вас від необхідності інвестувати в апаратне забезпечення, тож ви можете швидше розробляти та розгортати програми. Використовувати Amazon EC2 можна щоб запустити стільки віртуальних серверів, скільки потрібно, налаштувати безпеку та мережу, а також керувати сховищем. Amazon EC2 дає змогу збільшувати або зменшувати масштаб, щоб впоратися зі змінами вимог або сплесками популярності, зменшуючи потребу в прогнозуванні трафіку;

- Amazon Elastic Container Service (Amazon ECS) — це високомасштабована і швидка служба керування контейнерами, яка дозволяє легко запускати, зупиняти та керувати контейнерами в кластері. Контейнери визначаються при заданні завдання, яке використовується для виконання окремих завдань або завдань у службі. У цьому контексті служба — це конфігурація, яка дозволяє запускати й підтримувати певну кількість завдань одночасно в кластері. Можна запускати свої завдання та послуги на безсерверній інфраструктурі, якою керує AWS Fargate. Для більшого контролю над інфраструктурою можна запускати завдання та служби на кластері екземплярів Amazon EC2, про який ми згадували в попередньому пункті;
- Amazon Elastic Kubernetes (Amazon EKS) — це керована служба, яку можна використовувати для запуску Kubernetes на AWS, не встановлюючи, експлуатуючи та обслуговуючий власний майданчик керування Kubernetes або ноди. Kubernetes — це система з відкритим кодом для автоматизації розгортання, масштабування та керування контейнерними додатками. Є open-source аналогом ECS з попереднього пункту.

Крім підвищення швидкості випуску оновлень, використання мікросервісної архітектури дозволяє домогтися децентралізації управління. Команда, що відповідає за розробку того чи іншого сервісу, сама має право визначати його внутрішню архітектуру та її особливості. При цьому сідаючи за розробку хмарного застосунку не слід поспішати зі швидким дробленням його на складові елементи. Головний противник такого бездумного підходу — Мартін Фаулер; він же – один із авторів ідеї мікросервісної архітектури. Найпростіше спочатку використовувати монолітний підхід, і потім стимулювати еволюцію додатку «природним чином», орієнтуючись на розшивку вузьких місць та додавання додаткових функцій. У результаті можна сформулювати таке правило: завдання програміста при роботі з мікросервісною архітектурою – не просто розбити додаток на максимальну кількість складових частин, а розумним чином розмежувати їхню відповідальність за отримання та обробку даних.

Крім безлічі очевидних переваг, мікросервісна архітектура має свої особливості, які необхідно враховувати при розробці свого хмарного застосунку. Зокрема, для підтримки роботи цієї програми необхідно постійно підтримувати підвищені вимоги до якості управління внутрішніми API. Коли один із компонентів змінює свій інтерфейс, він повинен підтримувати зворотну сумісність, щоб підтримувати попередню версію власного API. Якщо це правило дотримується, можна динамічно перемикатися зі старої версії на нову без пролем. Якщо ж підтримка колишньої версії API не опрацьована, це загрожує втратою частини функціональності програми, а гіршому випадку – постійними збоями у роботі.

Друга важлива особливість мікросервісних програм полягає в складностях пошуку в них багів. Якщо «падає» програма, написана в монолітній логіці або SOA, знайти джерело проблеми не складе труднощів. У додатку, що складається з безлічі сервісів, пошук причини бага може сильно затягнутися через те, що дані від користувача нерідко проходять обробку через кілька мікросервісів, і складно визначити, в якому з них відбувається збій. При цьому процес пошуку бага потрібно вести дуже акуратно: будь-який невдалий рефакторинг може призвести до поломки модуля, що працює, і на додачу до початкової проблеми розробник отримає другу.

Третя важлива деталь, яку необхідно враховувати, розробляючи хмарну програму – спосіб взаємодії її складових частин між собою. Як і в SOA, для обміну даними послуги використовують веб-служби, але в мікросервісній архітектурі з'явилися патерни взаємодії, наприклад, як streaming, Command and Query Responsibility Segregation (CQRS), Event sourcing. Зазвичай розробники розраховують, що час відгуку між запитом та відповіддю у додатку є досить невеликим. У розподіленій системі не можна покладатися навіть те, що відповідь взагалі прийде. Також в архітектурі хмарних додатків мікросервіси використовують різні бази даних, що найбільш оптимально підходять для вирішення їх конкретних завдань. Наприклад, таблиці можуть швидко читатись, але важко справляються з великою кількістю операцій зі змін даних. Така база

добре підійде для ведення рахунків - вони рідко змінюються. Інший тип операцій – процесинг; у ньому щодня по кожній карті можуть бути десятки змін, а читань даних навпаки мало.

Нарешті, четвертий факт, про який слід пам'ятати при розробці хмарного додатку – мікросервісна архітектура орієнтована насамперед на використання stateless підходу. При цьому не варто впадати в крайнощі. Деякі сервіси, при необхідності, все ж таки можуть здійснювати підтримку стану (бути «stateful»), якщо цього вимагає бізнес-логіка, і вони мають бути спроектовані особливо ретельно. Наприклад: якщо користувач робить запит на отримання кредиту, то система, що отримала заявку, повинна цей стан зберегти, щоб передати його іншим сервісам. А ось сервіс, який відповідає за пошук інформації у внутрішній картотеці кредитних історій, може не зберігати стан і забути про те, дані на якого іменного користувача він шукав пару хвилин тому - все одно вже за мить йому прийде новий запит (хоча і в цьому процесі може бути різна поведінка сервісу). Всі вищенаведені приклади та практиці вже активно використовуються лідерами світової IT-галузі. Наприклад, піонером у розвитку мікросервісної архітектури є Netflix. Компанія випустила безліч open-source додатків, бібліотек та фреймворк для моніторингу, балансування та логування запущених мікросервісних програм. Також варто додати, що свої послуги Netflix надає клієнтам через використання хмарного середовища AWS. [11]

Таким чином можемо зробити висновок, що послуги хмарних провайдерів є як ніколи потрібні сучасним IT-компаніям, а враховуючи, що наразі чи не кожне підприємство має свій IT-підрозділ бодай для організації бухгалтерії, не кажучи вже про компанії-розробники ПЗ, частка яких продовжує зростати в Україні та світі, цей напрямок має надзвичайний потенціал зростання й вже задає правила в сфері розробки ПЗ, Big Data та Machine Learning (ML). Окресливши актуальність та своєчасність розвитку хмарних обчислень переходимо до цільової IaaS платформи AWS, її сильних сторін та функціоналу, щоб систематизувати її можливості й зосередитись на питаннях кібербезпеки.

1.4. Функціонал і архітектура AWS

AWS обслуговує понад мільйон активних клієнтів у більш ніж 240 країнах і територіях. AWS постійно розширює глобальну інфраструктуру, щоб допомогти клієнтам досягти меншої затримки та більшої пропускної здатності, а також забезпечити, щоб їхні дані розміщувалися лише в регіоні AWS, який вони вказали. Оскільки клієнти розвиватимуть свій бізнес, AWS продовжуватиме надавати інфраструктуру, яка відповідає їхнім глобальним вимогам. Хмарна інфраструктура AWS побудована навколо регіонів і зон доступності («region» та «availability zone» англійською, відповідно). Регіон AWS — це фізичне місце у світі, де розміщені кілька зон доступності. Зони доступності складаються з одного або кількох дискретних ЦОД, кожен із яких має резервне живлення, мережу та підключення і розміщені в окремих приміщеннях. Ці зони доступності пропонують клієнтам можливість керувати програмами та базами даних, які є більш доступними, відмовостійкими та масштабованими, ніж це було б можливо з одного ЦОД. AWS Cloud працює в 80 зонах доступності в 25 географічних регіонах по всьому світу, а також оголошені плани щодо більшої кількості зон і регіонів доступності.

На прикладі Америки: кожен регіон Амазонки розроблено так, щоб бути повністю ізольованим від інших регіонів Амазонки. Таким чином досягається максимально можлива відмовостійкість і стабільність.

Кожна зона доступності ізольована, але зони доступності в регіоні з’єднані через канали з низькою затримкою. AWS надає можливість розміщувати інстанси та зберігати дані в кількох географічних регіонах, а також у кількох зонах доступності в кожному регіоні AWS. Кожна зона доступності розроблена як незалежна від відмов. Це означає, що зони доступності фізично відокремлені в межах типового міського регіону та розташовані на локаціях з меншим ризиком. На додаток до дискретного джерела безперебійного живлення (UPS) і резервного генерування на місці, центри обробки даних, розташовані в різних зонах доступності, призначені для живлення від незалежних підстанцій, щоб

знизити ризик того, що події в електромережі впливають на більше ніж одну зону доступності.

Рис. 1.4. Геолокаційне розміщення ЦОД AWS в Європі та Африці

AWS пропонує багато різних інструментів та рішень для підприємств та розробників програмного забезпечення, які можуть бути використані в ЦОД AWS у 190 країнах. Більше 100 сервісів включають портфоліо веб-послуг Amazon, включаючи послуги з обчислень, баз даних, управління інфраструктурою, розробки додатків та безпеки. Нам слід виконати короткий

огляд цих сервісів щоб отримати розуміння, як підходити до забезпечення кібербезпеки в умовах їх роботи.

За категоріями, AWS надає сервіси:

- обчислювальні;
- баз даних (БД) та сервіси для управління даними;
- міграційні та засоби забезпечення роботи гібридної хмари;
- мережеві;
- інструменти для розробників ПЗ;
- управління;
- моніторингу;
- безпеки;
- управління великими даними;
- аналітики;
- штучного інтелекту (ШІ);
- повідомлень та сповіщень;

Обчислювальні.

EC2, ECS, EKS, Lambda - так чи інакше є прикладами сервісів, що забезпечують обчислення, на основі віртуальних машин чи виділених серверів у випадку EC2; в якості контейнерів у випадку ECS чи EKS; з використанням serverless підходу у випадку Lambda, тобто для виконання коду не знадобиться розгортати та підтримувати будь-яку платформу AWS приймає код певної мови на вході та самостійно виконує його повертаючи результат. Ці сервіси ми поверхнево розглянули в попередньому підрозділі, тому не будемо повторюватись. Додамо лише, що служба EC2 пропонує десятки типів інстансів з різними можливостями та розмірами, пристосованих до конкретних типів робочого навантаження та програм, таких як завдання з великою кількістю оперативної пам’яті та прискорених обчислень. AWS також надає інструмент автоматичного масштабування для динамічного масштабування можливостей для підтримки справності та продуктивності екземплярів.

Зберігання:

- Amazon Simple Storage Service (S3) забезпечує масштабоване зберігання об’єктів для резервного копіювання, збору та аналізу даних. Підприємство може заощадити гроші при архівації даних за допомогою сервісу S3 Glacier, призначеного для тривалого зберігання даних з рідким доступом;
- Amazon Elastic Block Store надає томи дискового простору на рівні блоків для постійного зберігання даних при використанні інстансів EC2 (специфічні ВМ);
- файлова система Amazon Elastic File Storage пропонує хмарну систему зберігання файлів в NFS;

Бази даних, управління даними:

- Amazon Relational Database Service (RDS), яка підтримує такі типи БД, як Oracle, SQL Server, PostgreSQL, MySQL, MariaDB та фірмову високопродуктивну базу даних під назвою Amazon Aurora – забезпечує систему управління реляційними базами даних для користувачів AWS. AWS також пропонує керовані бази даних NoSQL через Amazon DynamoDB;
- клієнт AWS може використовувати Amazon ElastiCache та DynamoDB Accelerator для кешування даних БД в пам'яті та в режимі реального часу для програм. Amazon Redshift пропонує сховище даних, що полегшує аналітикам виконання завдань бізнес-аналітики (BI).

Міграція та засоби забезпечення роботи гібридної хмари.

AWS включає різні інструменти та послуги, призначені для допомоги користувачам мігрувати програми, бази даних, сервери та дані у загальнодоступну хмару. При наявності контракту підтримки є можливість звернутись за допомогою в підготовці й проведення міграції до власне спеціалістів AWS чи акредитованих партнерів.

Мережеві:

- віртуальна приватна хмара Amazon VPC надає адміністратору контроль над віртуальною мережею для використання ізольованого простору хмари AWS;

- адміністратори можуть розбалансувати мережевий трафік за допомогою служби Elastic Load Balancing (ELB), яка включає в себе балансування навантаження на додатки та балансування навантаження на мережу.
- AWS також надає систему доменних імен Amazon Route 53, яка спрямовує кінцевих користувачів до програм (DNS-балансування);
- IT-спеціаліст може встановити спеціальне з'єднання від локального центру обробки даних до хмари AWS за допомогою AWS Direct Connect.

Інструменти для розробників:

- AWS CLI є фірмовим інтерфейсом управління з командного рядка;
- розробник може використовувати інструменти AWS для Powershell для управління хмарними службами з середовищ Windows;
- розробники можуть використовувати AWS Serverless Application Model для моделювання середовища AWS для тестування Lambda-функцій;
- пакети SDK AWS доступні для різних платформ та мов програмування, включаючи Java, PHP, Python, Node.js, Ruby, C ++, Android та iOS;
- Amazon API Gateway дозволяє команді розробників створювати, керувати та контролювати API, які дозволяють програмам отримувати доступ до даних або функціональних можливостей із серверних служб. Шлюз API керує тисячами одночасних викликів API.

Команда розробників також може створити безперервну інтеграцію та безперервну доставку з такими послугами, як:

- AWS CodePipeline
- AWS CodeBuild
- AWS CodeDeploy
- AWS CodeStar

Розробник також може зберігати код у сховищах Git за допомогою AWS CodeCommit та оцінювати продуктивність програм на основі мікросервісів за допомогою AWS X-Ray.

Управління та моніторинг:

- адміністратор може керувати та відстежувати конфігурацію хмарних ресурсів за допомогою AWS Config rules. Ці інструменти разом із AWS Trusted Advisor можуть допомогти команді IT уникнути неправильно налаштованих та надмірно дорогих розгортань хмарних ресурсів;
- у своєму портфоліо AWS пропонує кілька засобів автоматизації. Адміністратор може автоматизувати надання інфраструктури за допомогою шаблонів AWS CloudFormation, а також використовувати AWS OpsWorks та Chef для автоматизації інфраструктури та системних конфігурацій;
- клієнт AWS може відстежувати стан ресурсів та стану додатків за допомогою Amazon CloudWatch та інформаційної панелі AWS Personal Health Dashboard, а також використовувати AWS CloudTrail для збереження активності користувачів та викликів API для аудиту.

AWS включає в себе різноманітні послуги аналізу великих даних та прикладних послуг:

- Amazon Elastic MapReduce (EMR), який пропонує фреймворк Hadoop для обробки великих обсягів даних;
- Amazon Kinesis, який надає кілька інструментів для обробки та аналізу потокових даних («streaming data» з англійської);
- AWS Glue - це служба, яка обробляє завдання вилучення, перетворення та завантаження;
- Amazon Elasticsearch дозволяє команді виконувати моніторинг додатків, аналіз журналів та інші завдання за допомогою інструмента Elasticsearch з відкритим вихідним кодом;
- Amazon Athena для S3, що дозволяє аналітикам робити SQL-запити до даних в об’єктному сховищі S3;

- Amazon QuickSight допомагає аналітикам візуалізувати дані.

Інтернет речей.

AWS також має різноманітні послуги, які дозволяють розгортати Інтернет речей (IoT). Служба IoT AWS надає внутрішню платформу для управління пристроями IoT та передаванням даних до інших служб зберігання даних та баз даних AWS. AWS забезпечує апаратне забезпечення для функцій IoT, AWS Greengrass надає обчислювальні можливості AWS для пристроїв IoT, а IoT Device Defender підтримує захист в Інтернеті речей.

1.5. Оцінка динаміки в еволюції загроз веб-додаткам в порівнянні з попередніми роками

OWASP Топ-10 - це список з десяти найпоширеніших на момент публікації вразливостей веб-додатків. Завдяки цьому списку користувачі обізнані про найбільш критичні ризики та загрози, їх наслідки та заходи протидії. Оновлюється список OWASP кожні три-чотири роки. Востаннє він був випущений у 2021 році, який і буде розглянуто далі в порівнянні з 2017 роком.

Розглянемо ілюстрацію динаміки змін актуальності загроз веб додаткам.

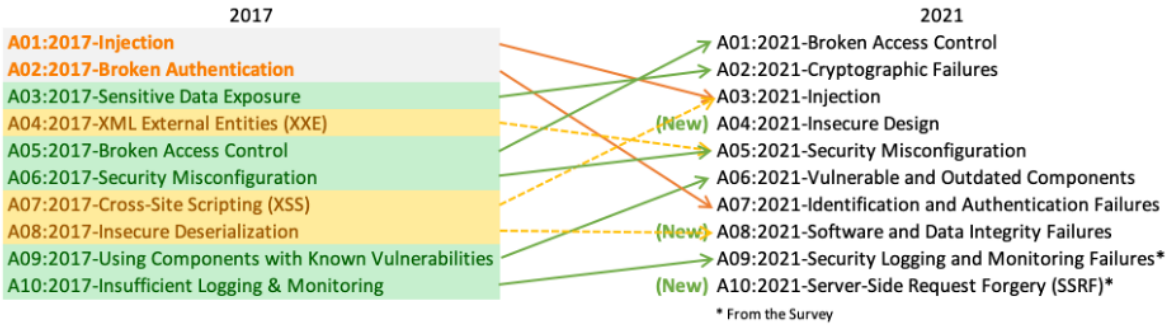


Рис. 1.5. Динаміка змін актуальності загроз веб додаткам за 2017-2021 рр.

Порівняння списку OWASP Top-10 2017 та 2021 років:

1. A01:2021- Broken Access Control піднявся з п’ятої позиції до категорії з найсерйознішим ризиком безпеки веб-додатків; Надані дані вказують на те, що в середньому 3,81% тестованих додатків мали одну або кілька загальних перерахувань слабкості (CWE) з понад 318 тисячами випадків CWE в

цій категорії ризику. 34 CWE, зіставлені з порушенням контролю доступу, мали більше випадків у додатках, ніж будь-яка інша категорія.

2. A02:2021- Cryptographic Failures зміщуються на одну позицію вгору до №2, раніше відомого як A3:2017- Sensitive Data Exposure, що було скоріше поширеним симптомом, ніж першопричиною. Оновлена назва зосереджена на збоях, пов’язаних із криптографією, як це було неявно раніше. Ця категорія часто призводить до розкриття конфіденційних даних або компрометації системи.

3. A03:2021- Injection ковзає вниз до третьої позиції. 94% додатків були перевірені на певну форму ін’єкції з максимальним рівнем зараження 19%, середнім рівнем зараження 3,37%, а 33 CWE, віднесені до цієї категорії, займають друге місце за кількістю випадків у додатках із 274 тисячами випадків. XSS тепер є частиною цієї категорії в цьому виданні.

4. A04:2021- Insecure Design – це нова категорія для 2021 року, зосереджена на ризиках, пов’язаних із недоліками дизайну. Якщо ми справді хочемо «рухатися вліво» як галузь, нам потрібно більше моделювання загроз, безпечних шаблонів і принципів проектування та еталонних архітектур. Небезпечний дизайн не може бути виправлений ідеальною реалізацією, оскільки за визначенням необхідні засоби контролю безпеки ніколи не створювалися для захисту від конкретних атак.

5. A05:2021- Security Misconfiguration змінюється з №6 у попередньому виданні; 90% додатків були перевірені на певну форму неправильної конфігурації, із середнім рівнем зараження 4,5%, і понад 208 тисяч випадків CWE віднесено до цієї категорії ризику. Колишня категорія XXE A4:2017-XML (XXE) тепер є частиною цієї категорії ризику.

6. A06:2021-Vulnerable and Outdated Components категорія раніше називалася «Using Components with Known Vulnerabilities» і займала 2-е місце в опитуванні спільноти «Top-10», але також мала достатньо даних, щоб увійти до топ-10 за допомогою аналізу даних. Ця категорія піднялася з 9 у 2017 році і є відомою проблемою, яку нам важко перевірити та оцінити ризики. Це єдина категорія, яка не має жодних загальних вразливостей та ризиків (CVE), які

зіставлені з включеними CWE, тому в їхні оцінки враховуються значення експлойту та впливу за замовчуванням 5,0.

7. A07:2021- Identification and Authentication Failures раніше були порушеною автентифікацією і сповзали з другої позиції, а тепер включають CWE, які більше пов’язані з помилками ідентифікації. Ця категорія все ще є невід’ємною частиною ТОП-10, але збільшення доступності стандартизованих фреймворків, здається, допомагає.

8. A08:2021- Software and Data Integrity Failures – це нова категорія для 2021 року, яка зосереджена на створенні припущень, пов’язаних із оновленнями програмного забезпечення, критично важливими даними та конвеєрами CI/CD без перевірки цілісності. Один із найбільш зважених впливів даних загальної вразливості та ризиків/системи оцінки загальної вразливості (CVE/CVSS), зіставлених з 10 CWE в цій категорії. A8:2017- Insecure Deserialization тепер є частиною цієї більшої категорії.

9. A09:2021-Security Logging and Monitoring Failures раніше був A10:2017-Недостатнє ведення журналу та моніторингу та додано з опитування 10 найкращих спільнот (№3), перейшовши з №10 раніше. Ця категорія розширена, щоб охопити більше типів збоїв, її складно перевірити, і вона погано представлена в даних CVE/CVSS. Однак збої в цій категорії можуть безпосередньо вплинути на видимість, оповіщення про інциденти та криміналістичну експертизу.

10. A10:2021- Server-Side Request Forgery. Дані показують відносно низький рівень зараження з охопленням тестуванням вище середнього, а також вищими за середні оцінки щодо потенціалу експлуатації та впливу. Ця категорія представляє сценарій, коли члени спільноти безпеки говорять нам, що це важливо, хоча наразі це не показано в даних.

В цьому розділі ми не зупинятимемось на основних сервісах AWS, що надають послуги захисту, відповідності, забезпечують роботу з ідентичностями, англійською – «Security, Identity, and Compliance». Так як робота зосереджена на

питаннях безпеки, цим темам присвячено особливу увагу у розділі 2 та приклади реалізації конкретних рішень в розділі 3.

Висновки розділу 1.

Було проведено аналіз області, досліджена історія розвитку хмарних обчислень, розкрита сутність й аспекти, в яких перевагу отримують хмарні технології, доведено факт зростання цієї сфери в IT та актуальності питань інформаційної та кібербезпеки в ній. Також описана термінологія, методи та засоби, що використовуються для побудови хмарних інфраструктур.

2 ІНСТРУМЕНТИ ТА ПІДХОДИ ДО ЗАХИСТУ ХМАРНОЇ ІНФРАСТРУКТУРИ ВЕБ-ДОДАТКІВ В AMAZON AWS

2.1. Модель «Shared security responsibility model» від AWS

Інфраструктура AWS була розроблена як одна з найбільш гнучких і безпечних серед доступних сьогодні хмарних обчислень. Ця інфраструктура створена й керується не лише відповідно до найкращих практик та стандартів безпеки, а й з урахуванням унікальних потреб хмари. AWS використовує надлишкові та багаторівневі засоби керування, постійну перевірку та тестування та значну кількість автоматизації, щоб забезпечити цілодобовий моніторинг та захист базової інфраструктури. AWS гарантує, що ці елементи керування реплікуються в кожному новому ЦОД або службі. Це означає, що ви отримуєте стійку інфраструктуру, розроблену для високого рівня безпеки, без капітальних витрат і операційних витрат традиційного центру обробки даних.

AWS працює за моделлю спільної відповідальності за безпеку – «Shared security responsibility model», коли AWS відповідає за безпеку базової хмарної інфраструктури, а ви — за безпеку робочих навантажень, які ви розгортаєте в AWS. Ви можете жорстко обмежити доступ до середовищ, які обробляють конфіденційні дані, або застосувати менш суворі засоби контролю для інформації, яку хочете оприлюднити.

AWS та довірені партнери пропонують широкий спектр інструментів і функцій, які допоможуть вам досягти ваших цілей безпеки. Ці інструменти відображають знайомі елементи керування, які ви розгортаєте в локальних середовищах. AWS надає спеціальні інструменти та функції, пов’язані з безпекою мережі, керуванням конфігурацією, контролем доступу та безпекою даних. Крім того, AWS надає інструменти моніторингу та ведення журналів, які можуть забезпечити повну видимість того, що відбувається у вашому середовищі.

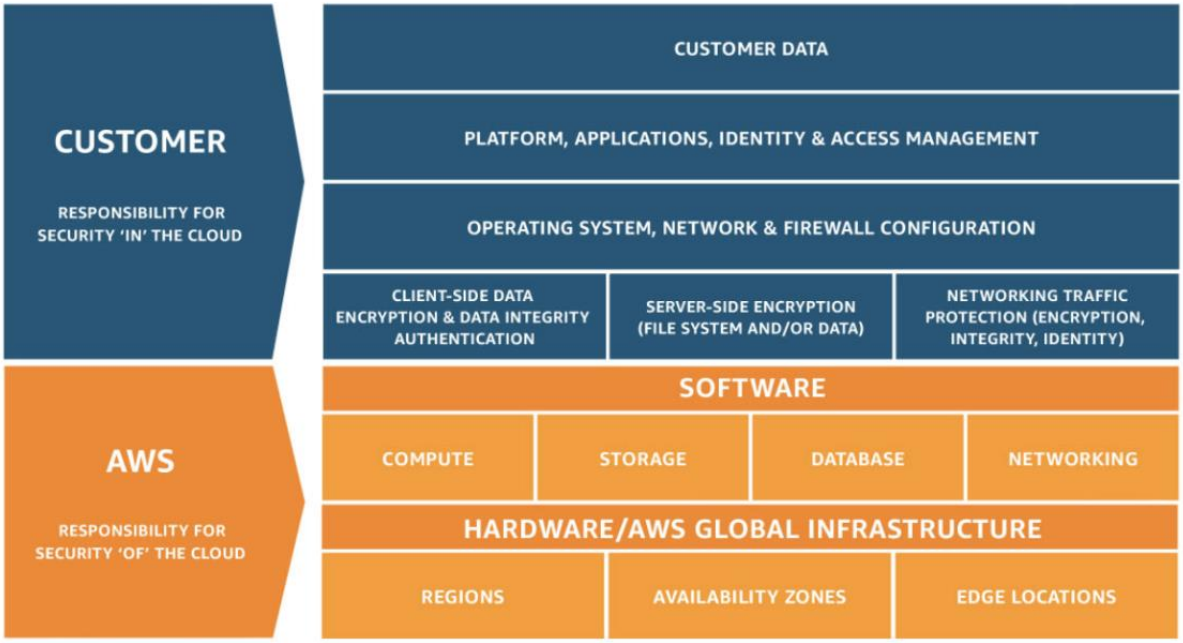


Рис. 2.1. Shared security responsibility model від AWS

2.2. Інструменти організації безпеки в хмарі AWS

AWS надає інструменти, розроблені для підвищення безпеки облікового запису, а також безпеки програм і служб.

Обліковий запис AWS є вектором атаки, оскільки ресурси та дані доступні через загальнодоступний інтерфейс програмного забезпечення (API). Реалізація безпечної стратегії управління ідентифікаторами та доступом допомагає запобігти витоку даних — наприклад, у S3 buckets — для громадськості. Багато інструментів AWS надають уявлення про налаштовані дозволи та шаблони доступу, а також записують усі дії з метою відповідності та аудиту.

Програми та служби, розміщені в AWS, чутливі до різного роду загроз ззовні. XSS, SQL ін'єкції та атаки brute force спрямовані на загальнодоступні кінцеві точки. Розподілені атаки відмови в обслуговуванні (DDoS) можуть намагатися вивести з ладу ваші сервіси, що може поставити під загрозу безпеку вашої хмарної інфраструктури. Без належного керування може витікати конфіденційна інформація, наприклад облікові дані бази даних.

Тому дуже важливо, щоб організації, які переходять у хмару, зосередилися на мінімізації ризиків та покращенні загальної системи безпеки, звертаючись до безпеки облікового запису, а також безпеки програм і служб. Наведені нижче служби AWS блокують хмарнізагрози, допомагаючи захистити дані та системи клієнтів від атак

2.3. Розмежування прав доступу з IAM

В подальшому викладенні матеріалу певні терміни будуть використовуватись як взаємозамінні – англомовні назви в оригіналі та переклад їх на українську для кращого розуміння значення того чи іншого терміну. Використання англомовних назв забезпечить подальший комфорт роботи в хмарі AWS з реальними ресурсами AWS, якщо такий буде потрібен.

Світ AWS починається з облікових записів і ресурсів у цих облікових записах. IAM існує насамперед для захисту ресурсів у вашому обліковому записі від таких проблем, як:

1. Зловмисники, які намагаються зробити небажані дії з вашим обліковим записом AWS (наприклад, вкрасти ваші дані з ваших сегментів S3).
2. Користувачі/програми у вашій компанії випадково видаляють ресурси або виконують дії, які вони інакше не мали б мати.

Для підвищення безпеки AWS рекомендує налаштувати багатофакторну автентифікацію (MFA), щоб захистити ваші ресурси AWS. Ви можете ввімкнути MFA для користувачів IAM або користувача root облікового запису AWS. Коли ви вмикаєте MFA для користувача root, це впливає лише на облікові дані користувача root. Користувачі IAM в обліковому записі є різними ідентифікаторами зі своїми обліковими даними, і кожна особа має власну конфігурацію MFA. [12]

Облікові записи створюють логічний бар'єр між користувачами AWS. Ідентифікатор облікового запису є однозначним ідентифікатором, напр. 123456789012.

Як найкраща практика, організація повинна мати кілька облікових записів AWS. Це може бути розділення за середовищами виконання коду для розробників: dev, staging, test, prod, etc.; також облікові записи можуть створюватись для різних департаментів та об'єднані в AWS Organization: Developers, HR, Finance, etc.

IAM часто використовується для обробки дозволів між обліковими записами AWS в організації. Прив'язаний до електронної адреси, кредитної картки; облікові записи – це спосіб виставлення рахунків.

Ресурси (resources) – це постійні об'єкти в обліковому записі, наприклад балансувальники навантаження Elastic Load Balancers (ELB) або екземпляри EC2. Ресурси завжди мають Amazon Resource Name (ARN) — ім'я ресурсу Amazon — яке їх однозначно ідентифікує, напр. для користувача IAM ARN може виглядати так: arn:aws:iam::123456789012:user/Development/product_1234/*

Ідентифікатор облікового запису 123456789012 присутній в ARN, як і тип ресурсу (цей ресурс є користувачем IAM). Основною сутністю в IAM є identity, що також являє собою тип ресурсу AWS. Сервіси AWS завжди надають API, до яких можна звертатись від імені певної identity; цей процес повідомляє AWS, хто суб'єкт (автентифікація) і чи можете даний суб'єкт робити те, що має здійснити команда у запиті (авторизація).

Існують дві основні форми identity в IAM: користувачі (IAM Users) та ролі (IAM Roles). Розглянемо їх характеристики та відмінності.

Користувачі або IAM Users.

Коли вперше створюється обліковий запис AWS, клієнт отримує доступ до користувача root, який має повний доступ до даного облікового запису. Також клієнту надається, і настійно рекомендується, можливість створення додаткових користувачів і ролей.

Деякі зауваження щодо користувачів:

- користувачі, на відміну від ролей, мають ім'я користувача та пароль. Це довговічні облікові дані, які можна зберігати протягом тривалого періоду часу та використовувати для входу на консоль AWS;

- користувачі, як правило, призначені для надання доступу особам до консолі AWS або API. Однак, як найкраща практика, потрібно використовувати IAM Role замість користувачів, коли це можливо. Це потрібно, щоб обмежити ризик втрати довготривалих облікових даних і надання зловмиснику доступу до вашого облікового запису AWS;
- користувачам надаються ключі доступу (access keys). Ключі доступу можна використовувати для виклику сервісів AWS через CLI або SDK. Як і ім'я користувача та пароль, ключі доступу довговічні. Вони виглядатимуть більш рандомізованими, і їх можна використовувати разом із CLI, створивши файл з іменем ~/.aws/credentials із вмістом, який виглядає так:

[personal]

aws_access_key_id = AKIAIOSFODNN7EXAMPLE

aws_secret_access_key=wJalrXUtnFEMI/K7MJENG/bIxRfiCYEXAIPLEKE

Є можливість використовувати профіль облікових даних із CLI з ключем --profile у CLI:

aws s3 ls –profile personal

30-11-2017 16:20:55 some-s3-bucket

2017-10-31 20:05:17 some-other-s3-bucket

...

Ролі або IAM Roles.

Як і користувачі, ролі — це підвид identity, яка використовується для доступу до API та ресурсів AWS. Однак ролі зазвичай використовуються для надання тимчасових облікових даних обліковому запису AWS. Крім того, ці тимчасові облікові дані можна надавати третім сторонам або іншим серавісам AWS.

Ролі використовуються повсюдно в AWS:

EC2 instance, якому потрібен доступ до ресурсів AWS, використовуватиме EC2 IAM Role для керування іншими службами/ресурсами AWS на основі логіки вашої програми.

Інші облікові записи AWS, яким потрібен доступ до ресурсів у сторонньому обліковому записі AWS, іноді споживають IAM Role у сторонньому обліковому записі, щоб отримати доступ через STS Assume Role API. Щоб дозволити їм це зробити, обліковий запис AWS надає дозволи на IAM Role для певної identity в іншому обліковому записі через IAM trust policy (докладніше про це пізніше).

Важливою особливістю AWS є його здатність виконувати дії від вашого імені. Однак служби AWS зазвичай реалізуються як облікові записи AWS; це означає, що вони за замовчуванням не мають доступу до ресурсів у вашому обліковому записі. Відповідно, вони часто вимагатимуть від вас створити та надати їм доступ до «службової ролі» (Service Role) у вашому обліковому записі, щоб вони могли виконувати дії від вашого імені. Наприклад, для автоматичного масштабування, EC2 потрібні дозволи для розгортання та видалення інстансів EC2.

Треті сторони, такі як сторонні веб-додатки, використовуватимуть ролі, щоб надавати користувачам, яким керують поза AWS, доступ до ресурсів AWS. Цей процес відомий як федерація доступу або Access Federation.

Політики або IAM policies.

Нарешті, коли користувач, що працює використовуючи певну identity викликає API AWS, IAM визначить, чи є виклик дійсним, оцінюючи одну або кілька політик. Існує два основних типи політики: Identity Policies та Resource Policies.

Identity Policies.

Identity Policies визначають, що дозволено робити даній identity (ролі або користувачу). Нижче наведено приклад простішої Identity Policiy:

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "VisualEditor0",
```

```
            "Effect": "Allow",
            "Action": [
                "iam:CreateRole",
                "iam:CreateUser"
            ],
            "Resource": [
                "arn:aws:iam::123456789012:role/some-role",
                "arn:aws:iam::123456789012:user/some-user"
            ]
        },
        {
            "Action": [
                "logs:*"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

Цілком очевидно що дана IAM Policy потребує роз’яснень для тих, хто знайомиться з синтаксисом вперше:

- API, які можна викликати, називаються «Action» в IAM. Кілька API можуть виконувати одну дію, але найчастіше дії відповідають лише одному API;
- політика є білим списком; це означає, що за замовчуванням дії не дозволені. Явний дозвіл надається для двох дій: «CreateRole» і «CreateUser». Щоб отримати повну розбивку того, як логіка політики оцінюється в обліковому записі AWS, див. Рисунок 2.2;
- більшість викликів API в AWS можна зменшити до дозволених лише на певних ресурсах, як це було описано вище. Це сталося, оскільки розділ «Resource» не рівний «*». Це означає, що запит, що використовує цю identity,

буде успішним лише щодо ресурсів із цими конкретними ARN. Наприклад, ви можете створити роль лише з ARN «arn:aws:iam::123456789012:role/some-role» за допомогою цієї політики. Зведення до окремих ресурсів є найкращою практикою, яка запобігає багатьом дірам у безпеці. Візьмемо, наприклад, злом Capital One 2019 року. Capital One надав своєму брандмауеру надмірні дозволи S3, що після того, як його обійшли з SSRF, дозволило зловмисникам викрасти дані понад 100 мільйонів осіб;

- тільки Identity policy — без Resource policy — працюватиме лише в межах одного облікового запису AWS;
- підтримуються символи wildcard («*»). Друга заява в цій політиці стосується журналів CloudWatch і дає повний доступ до журналів Cloudwatch (будь-який API, який вони відкривають, і для будь-якого ресурсу).

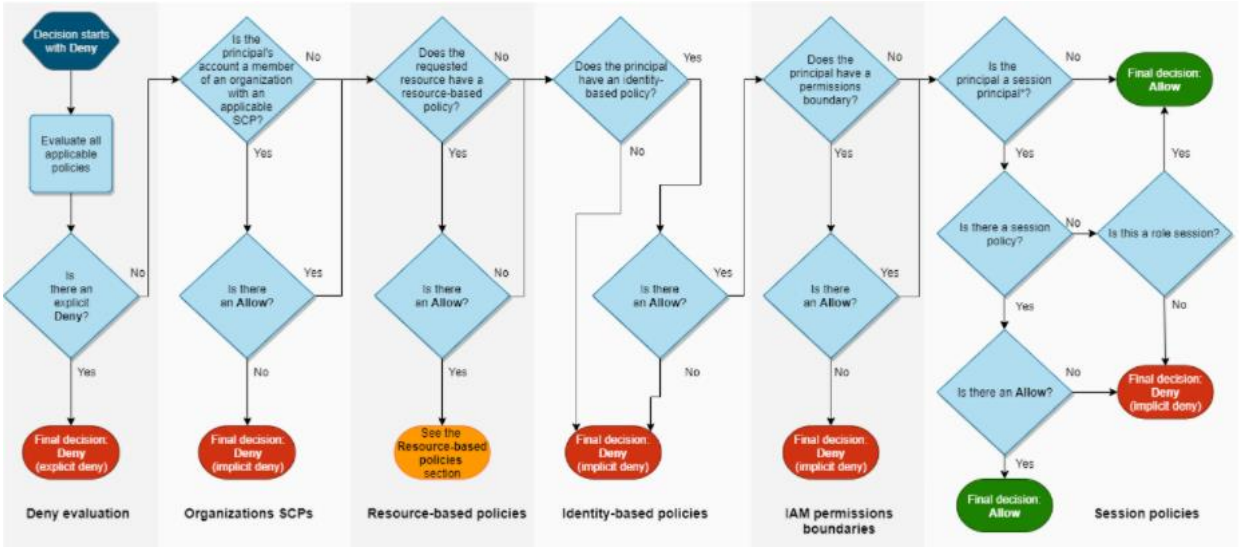


Рис. 2.2. Діаграма визначення дозволу на виконання запиту в AWS

Resource policies.

Resource policies прикріплені до ресурсу. Наприклад, ви можете додати resource policy на Amazon S3 buckets, Amazon SQS queues, VPC endpoints, AWS Key Management Service encryption keys. Завдяки ним можна вказати, хто має доступ до ресурсу і які дії вони можуть виконувати з ним. Політики на основі ресурсів є лише вбудованими, а не керованими.

Коли на ресурс діє певний principal, незалежно від того, чи це S3 bucket, чи IAM Role, яку хтось намагається взяти на себе, його Resource policy набуде чинності. Спочатку вони можуть здатися зайвими, оскільки ви можете налаштувати Identity policies до певних ресурсів, але часто буває корисно встановити Resource policy замість того щоб обмежувати за допомогою Identity policy всі можливі identity, які звертаються до даного ресурсу.

В розділі 3 на практиці показано як використовувати Identity policies та Resource policies, а також деякі інші атрибути розмежування прав доступу в сервісі AWS IAM.

Розглянемо Resource policy для ролі. Слід зауважити, що Resource policies щодо ролей також відомі як trust policies, оскільки вони дозволяють іншим взяти на себе роль. Загалом синтаксис нагадує Identity policy й за структурою є JSON-об’єктом, тип що зараз широко поширений в програмуванні та легко читається людиною.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AllowEC2",
   "Effect": "Allow",
   "Principal": {
    "Service": "ec2.amazonaws.com"
   },
   "Action": "sts:AssumeRole"
  },
  {
   "Sid": "AllowAssumeRole",
   "Effect": "Allow",
   "Principal": {
    "AWS": [
```

```
          "arn:aws:iam::123456789012:role/devops",
          "arn:aws:iam::123456789012:role/jenkins-master"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2.4. Amazon GuardDuty

Amazon GuardDuty — це служба безперервного моніторингу безпеки, яка аналізує та обробляє такі джерела даних: журнали VPC Flow Logs, AWS CloudTrail management event logs, CloudTrail S3 data event logs та DNS logs. GuardDuty використовує канали Threat Intelligence (TI) (кіберрозвідки), такі як списки шкідливих IP-адрес і доменів, а також машинне навчання, щоб виявляти несподівані та потенційно несанкціоновані та шкідливі дії у вашому середовищі AWS. Це може включати такі проблеми, як підвищення привілеїв, використання відкритих облікових даних або зв’язок із шкідливими IP-адресами чи доменами. Наприклад, GuardDuty може виявляти скомпрометовані екземпляри EC2, які обслуговують зловмисне програмне забезпечення або майніть біткойн. Він також відстежує поведінку доступу до облікового запису AWS на наявність ознак компромісу, наприклад, несанкціонованого розгортання інфраструктури, як-от екземпляри, розгорнуті в регіоні, який ніколи не використовувався, або незвичайні виклики API, як-от зміна політики паролів для зниження надійності пароля. GuardDuty, як інструмент Threat Intelligence знаходиться у тісному взаємозв'язку з іншими процесами інформаційної безпеки – реагуванням на інциденти, управлінням ризиками, управлінням уразливостями, виявленням шахрайства та операційною діяльністю ІБ підрозділу. Підвищити ефективність даних процесів, якість та швидкість прийняття рішень у рамках цих процесів –

це і є, по суті, головне завдання роботи з TI. У першу чергу, використання threat intelligence в рази підвищує якість та швидкість реагування на інциденти. Коли надходить інформація про нову загрозу, можна оперативно поставити її на моніторинг, паралельно блокуючи деякі індикатори компрометації. Знаючи контекст, розуміючи, яким чином відбуватиметься кібератака, всі можливі варіанти її розвитку та яким чином ця загроза могла потрапити в інфраструктуру, можна вчасно її виявити, опрацювати в рамках конкретного інциденту, побудувати для неї відповідні сценарії реагування. У плані управління вразливостями дані про загрози допомагають у розстановці пріоритетів та визначенні критичності вразливостей. Threat intelligence дає необхідну фактуру для аналізу та оцінки ризиків - інформацію про актуальні загрози, отриману на тактичному та стратегічному рівні TI. В результаті процес ризик-менеджменту стає більш практичним та якісним. Threat intelligence дозволяє вибудовувати операційну діяльність ІБ-підрозділу, діяти проактивно, планувати, впроваджувати та реалізовувати захисні заходи, орієнтуючись на актуальний ландшафт загроз, а не наосліп. Вибудовуючи процес кіберрозвідки, кожна організація стикається з певними складнощами. По-перше, дані складно отримувати. Джерел інформації кіберрозвідки безліч, при цьому немає єдиного стандарту - кожен постачальник або канал надає їх у своєму вигляді. Частина даних поставляється в машиночитаній формі, інша - у вигляді звітів, розрахованих на читання аналітиком. В результаті, перш ніж почати аналізувати дані, навіть якщо використовується лише 2-3 джерела, їх потрібно привести до єдиної моделі уявлення, нормалізувати. Для правильної інтерпретації та прийняття рішень сирих даних недостатньо, потрібно їх збагатити контекстом, додатковою інформацією, яка допоможе підібрати найбільш правильну тактику дій у відповідь. Якщо джерел занадто багато, виникає складність у їх практичному застосуванні. Потрібно проводити фільтрацію та відбір, щоб не захлинутися в потоці інформації. Ці проблеми і вирішує GuardDuty.

GuardDuty – це важливий інструмент для прийняття рішень ІБ. Він дає розуміння ландшафту загроз для прогнозування можливих атак та реалізації

адекватних заходів захисту; підвищує якість та швидкість реагування на інциденти, тим самим дозволяючи мінімізувати можливі збитки. Інформація про актуальні загрози допомагає в більш точній оцінці ІБ-ризиків та плануванні необхідних заходів щодо їх обробки. GuardDuty інформує про стан середовища AWS, створюючи звіти, які можна переглянути на консолі GuardDuty або через події Amazon CloudWatch.

2.5. Amazon Macie

Amazon Macie — це повністю керований AWS сервіс безпеки та конфіденційності даних, яка використовує машинне навчання та порівняння паттернів, щоб допомогти виявляти, відстежувати й захищати конфіденційні дані компаній в середовищі AWS. Macie автоматизує виявлення конфіденційних даних, таких як персональна інформація (personally identifiable information, PII) і фінансові дані, щоб надати спеціалісту ІБ краще розуміння даних, які його організація зберігає в Amazon S3. Macie також надає інвентаризацію S3 buckets, а також автоматично оцінює та відстежує їх для забезпечення безпеки та контролю доступу. За лічені хвилини Macie може визначити і повідомити про надмірно відкриті або незашифровані сегменти даних вашої організації.

Якщо Macie виявляє конфіденційні дані або потенційні проблеми з безпекою чи конфіденційністю ваших даних, він створює детальні висновки, які ви можете переглянути та усунути, якщо необхідно. Ви можете переглядати та аналізувати ці результати безпосередньо в Macie або відстежувати й обробляти їх за допомогою інших служб, програм і систем.

2.6. AWS Config

AWS Config надає детальну інформацію про конфігурацію ресурсів AWS у обліковому записі AWS. Це включає те, як ресурси пов’язані один з одним і як вони були налаштовані в минулому, щоб отримати бачення того, як конфігурація

змінюються з часом. Ресурс AWS — це об’єкт, з яким можна працювати в AWS, наприклад EC2 інстанс, дискь EBS, Security Group або VPC.

AWS дозволяє:

1. Оцінити відповідність конфігурації ресурсів AWS розробленій політиці безпеки компаніїї завдяки налаштуванню перевірок на ряд правил, наприклад: на присутність шифрування дисків EBS, закриті порти SSH та RDP в конфігурації Security group для ресурсів, що знаходяться у публічних підмережах, інше.

2. Робити знімки поточних конфігурацій підтримуваних ресурсів, пов’язаних з вашим обліковим записом AWS.

3. Отримати конфігурації одного або кількох ресурсів, які існують у вашому обліковому записі.

4. Отримати історичні конфігурації одного або кількох ресурсів.

5. Отримувати сповіщення, коли ресурс створено, змінено, видалено.

6. Переглядати зв’язки між ресурсами. Наприклад, виокремити всі ресурси, які використовують певну групу безпеки.

Способів використання AWS Config доволі багато. При запуску веб-додатків на AWS часто постає завдання спільного використання ресурсів. Оскільки трафік на веб-додатки як правило зростає, потреба відстежувати ресурси AWS зростає також. AWS Config розроблено, щоб допомогти адміністраторам хмарної інфраструктури AWS контролювати ресурси додатків в таких сценаріях:

1. Адміністрація ресурсів.

Щоб краще керувати конфігураціями ресурсів і виявляти неправильні конфігурації ресурсів, відділу, що відповідатиме за завдання моніторингу хмарної інфраструктури у будь-який момент потрібна детальна видимість того, які ресурси існують і як ці ресурси налаштовані. Можна використовувати AWS Config, налаштвавши сповіщення, щодо того, коли ресурси створюються, змінюються або видаляються без необхідності відстежувати ці зміни, опитуючи виклики, зроблені до кожного ресурсу. Правила AWS Config можна

використовувати, щоб оцінити налаштування конфігурації ресурсів AWS. Коли AWS Config виявляє, що ресурс порушує умови одного з заданих правил, AWS Config позначає ресурс як невідповідний і надсилає сповіщення. AWS Config постійно оцінює ресурси під час їх створення, зміни або видалення.

2. Аудит та відповідність.

Компанії, що працюють з даними, часто підлягають під дотримання частих перевірок, щоб забезпечити відповідність внутрішнім політикам і найкращим практикам. Щоб продемонструвати відповідність, компаніям потрібен доступ до історичних конфігурацій ресурсів. Цю інформацію надає AWS Config.

3. Керування змінами конфігурації та усунення несправностей.

Використовуючи кілька ресурсів AWS, які залежать один від одного, зміна конфігурації одного ресурсу може мати непередбачувані наслідки для пов’язаних ресурсів. За допомогою AWS Config ви можете побачити, як ресурс, який ви збираєтеся змінити, пов’язаний з іншими ресурсами, і оцінити вплив ваших змін.

4. Аналіз безпеки.

Щоб проаналізувати потенційні недоліки безпеки, потрібна детальна історична інформація про конфігурації ресурсів AWS, наприклад дозволи AWS IAM, які надаються й надавались раніше користувачам, або правила Security group для Amazon EC2, які контролюють доступ до ресурсів. Можна використовувати AWS Config, щоб переглянути конфігурацію ваших груп безпеки EC2, включаючи правила порту, які були відкриті в певний час. Ця інформація може допомогти вам визначити, чи блокувала Security group вхідний трафік TCP до певного порту.

2.7. AWS CloudTrail

AWS CloudTrail — це сервіс AWS, що допомагає увімкнути для облікового запису AWS функціонал, що забезпечує функції:

- governance;

- compliance;
- operational and risk auditing;

Дії, які виконує користувач, роль або сервіс AWS, записуються як події в CloudTrail. Події включають дії, виконані в консолі керування AWS, інтерфейсі командного рядка AWS, а також у пакетах SDK та API AWS. CloudTrail увімкнено в обліковому записі AWS при створенні автоматисчно але зберігається всього 90 діб. Коли активність відбувається в обліковому записі AWS, ця активність записується в подію CloudTrail. Можна легко переглядати останні події на консолі CloudTrail, перейшовши в Event history. Для поточного запису активності та подій в обліковому записі AWS потрібно створити «trail». Відображення активності в обліковому записі AWS є ключовим аспектом безпеки та найкращих методів роботи. Можна використовувати CloudTrail для перегляду, пошуку, завантаження, архівування, аналізу та реагування на активність облікового запису в хмарній інфраструктурі AWS. Можете визначити, хто чи що вживав певні дії, які ресурси були використані, коли сталася подія та інші деталі, які допоможуть проаналізувати та реагувати на дії в обліковому записі AWS. За бажанням, можна увімкнути AWS CloudTrail Insights для trail, щоб допомогти визначити незвичайну активність і реагувати на неї. CloudTrail можна інтегрувати в програми за допомогою API, автоматизувати створення trail для AWS Organization, перевіряти статус створених trail і контролювати, як користувачі переглядають події CloudTrail.

2.8. Amazon Security Hub

AWS Security Hub надає повне уявлення про стан безпеки в AWS і допомагає перевірити середовище на відповідність стандартам галузі безпеки та найкращим практикам. Security Hub збирає дані безпеки з усіх облікових записів AWS, служб і підтримуваних продуктів сторонніх партнерів і допомагає аналізувати тенденції безпеки та виявляти найбільш пріоритетні проблеми безпеки.

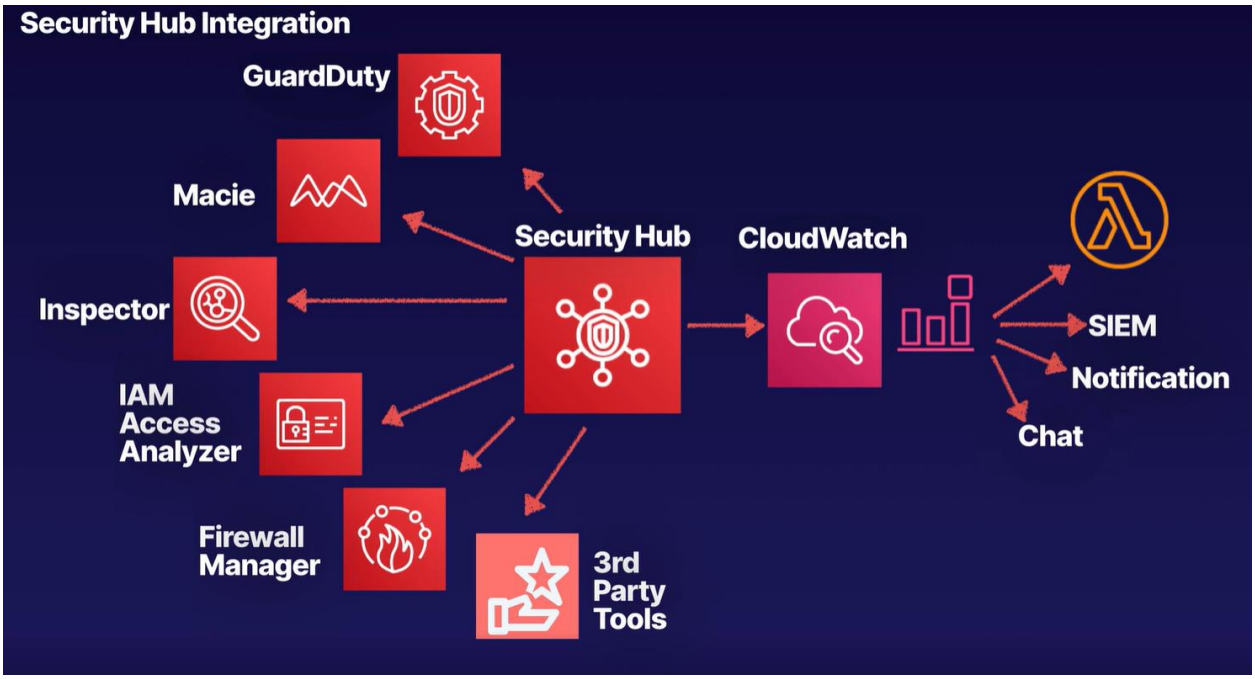


Рис. 2.3. Інтеграція Amazon Security Hub з інструментами безпеки та моніторингу

Наведемо переваги використання Security Hub:

1. Зменшення зусиль для збору та пріоритезації знахідок.

2. Security Hub обробляє дані пошуку за допомогою стандартного формату пошуку, що позбавляє від необхідності керувати даними пошуку з кількох форматів. Потім Security Hub співвідносить результати між постачальниками даних, щоб визначити пріоритети найважливіших.

3. Автоматична перевірка безпеки на відповідність передовому досвіду та стандартам.

4. Security Hub автоматично запускає безперервну конфігурацію на рівні облікового запису та перевірку безпеки на основі найкращих практик AWS і галузевих стандартів, як, наприклад, PCI DSS. Security Hub визначає конкретні облікові записи та ресурси, які потребують уваги.

5. Зведений огляд результатів для облікових записів і постачальників даних.

6. Security Hub об’єднує висновки щодо безпеки в облікових записах і продуктах постачальників і відображає результати на консолі Security Hub. Це

дає змогу переглядати загальний поточний стан безпеки, щоб помітити тенденції, виявити потенційні проблеми та вжити необхідних заходів щодо усунення.

7. Можливість автоматизувати виправлення знахідок.

8. Security Hub підтримує інтеграцію з Amazon EventBridge. Щоб автоматизувати виправлення конкретних знайдених проблем можна визначити спеціальні дії, які слід виконувати, коли отримано незадовільний результат. Наприклад, можна налаштувати надсилання результатів до команди підтримки хмарної інфраструктури компанії або до автоматизованої системи відновлення.

2.9. Amazon Inspector

Amazon Inspector — це сервіс, що допомагає покращити безпеку розгорнутих додатків на AWS. Inspector допомагає визначити вразливі місця у інстансах EC2 і у ПЗ на цих інстансах розміщеного, дозволяє зробити vulnerability assessment більш регулярним. Amazon Inspector надає чіткий список висновків щодо безпеки та відповідності, яким призначається пріоритет за рівнем серйозності. Крім того, ці висновки можна проаналізувати безпосередньо або як частину комплексних записів оцінки, доступних через API або консоль AWS Inspector. Оцінки безпеки AWS Inspector допоможуть перевірити небажану мережеву доступність екземплярів EC2 та вразливості в цих інстанціях EC2.

Переваги AWS Inspector:

1. Має централізоване керування кількома обліковими записами Amazon Inspector та змогу інтегруватись з Security Hub.

Якщо у середовищі AWS є кілька облікових записів, ви можете централізовано керувати своїм середовищем за допомогою одного облікового запису, використовуючи організації AWS і призначивши обліковий запис як обліковий запис делегованого адміністратора для Amazon Inspector. Amazon Inspector можна ввімкнути для всієї організації (AWS Organization) однимнатисканням миші. Крім того, можна автоматизувати ввімкнення служби

для майбутніх організації, коли вони приєднуються до вашої організації. Делегований обліковий запис адміністратора Amazon Inspector може керувати даними висновків і певними налаштуваннями для членів організації. Сюди входить перегляд зведених даних про результати для всіх облікових записів учасників, увімкнення або вимкнення сканування облікових записів учасників та перевірка відсканованих ресурсів в організації AWS.

2. Постійна перевірка середовища на наявність вразливостей та виявлення доступності по мережі.

З Amazon Inspector не потрібно вручну планувати або налаштовувати перевірки. Amazon Inspector автоматично знаходить і починає сканувати відповідні ресурси. Amazon Inspector продовжує оцінювати хмарну інфраструктуру протягом усього життєвого циклу ресурсів, автоматично скануючи ресурси щоразу, коли ви вносите в них зміни. Коли виявлено вразливості або відкриті мережеві маршрути до ресурсів що повинні бути ізольовані від прямого доступу з WAN, наприклад RDS інстанси БД, Amazon Inspector повідомляє отримані дані про ситуацію, що склалась, який спеціаліст ІБ може дослідити. Висновок містить вичерпну інформацію про вразливість, ресурс, який постраждав, та рекомендації щодо усунення. Якщо належним чином не виправити знахідку, Amazon Inspector автоматично виявляє виправлення та закриває вразливість.

3. Точна оцінка вразливостей за допомогою оцінки ризиків Amazon Inspector.

Оскільки Amazon Inspector збирає інформацію про середовище за допомогою сканування, він надає оцінки серйозності, спеціально пристосовані до конкретного середовища. Amazon Inspector вивчає показники безпеки, які складають базову оцінку вразливості в National Vulnerability Database (NVD), і коригує їх відповідно до конкретного обчислювального середовища. Наприклад, сервіс може знизити оцінку Amazon Inspector для екземпляра Amazon EC2, якщо вразливість можна використати через мережу, але відкритий мережевий шлях до

Інтернету з інстансу недоступний. Цей показник у форматі CVSS і є модифікацією базового CVSS, наданого NVD.

4. Інформування спеціаліста ІБ через виведення результатів про інциденти високого рівня впливу на інформаційну панель Amazon Inspector.

Інструментальна панель Amazon Inspector пропонує огляд результатів з усього середовища команії. На інформаційній панелі ви можете отримати доступ до детальної інформації про знахідку. Інформаційна панель містить спрощену інформацію про охоплення скануванням у конкретному середовищі, найважливіші висновки та ресурси, які мають найбільше результатів. Панель виправлення на основі ризиків на інформаційній панелі Amazon Inspector містить висновки, які впливають на найбільшу кількість інстансів. Ця панель спрощує визначення результатів, які найбільше впливають на середовище, перегляд деталей результатів та пропоновані рішення.

5. Керування знахідками за допомогою настроюваних інформаційних панелей.

На додаток до інформаційної панелі консоль Amazon Inspector пропонує перегляд результатів. На цій сторінці перелічено всі висновки для середовища та надано деталі окремих висновків. Можна переглядати результати, згруповані за категоріями або типом уразливості. У кожному поданні можна додатково налаштувати результати за допомогою фільтрів. Також є можливість використовувати фільтри, щоб створити правила приховування проблеми, які приховують небажані результати у представленні адміністратора безпеки. Звіти можна створювати у форматах CSV або JSON.

6. Відстеження та обробка результатів за допомогою інших служб і систем

Для підтримки інтеграції з іншими службами та системами Amazon Inspector публікує результати в Amazon EventBridge як події пошуку. EventBridge — це сервіс безсерверної шини подій, яка може направляти дані пошуку до цільових об’єктів, таких як функції AWS Lambda і Amazon Simple Notification Service (Amazon SNS) topics. За допомогою EventBridge можна відстежувати й

обробляти результати майже в реальному часі як частину існуючих робочих процесів із забезпечення безпеки та відповідності. Якщо був увімкнений AWS Security Hub, Amazon Inspector також публікує результати в Security Hub. За допомогою Security Hub легше відстежувати й обробляти свої знахідки в рамках ширшого аналізу стану безпеки організації в AWS.

2.10. AWS Shield

AWS надає AWS Shield Standard і AWS Shield Advanced для захисту від DDoS. AWS Shield Standard автоматично включається без додаткових витрат. Для додаткового захисту від атак DDoS AWS пропонує AWS Shield Advanced, що забезпечує розширений захист від DDoS-атак для ваших ресурсів.

Розширений захист Shield надає для будь-якого з таких типів ресурсів:

- CDN Amazon CloudFront;
- DNS-сервіс Amazon Route 53;
- прискорювачі AWS Global Accelerator;
- ELB;
- Amazon EC2 Elastic IP.

Якщо використовується Shield Advanced для захисту Elastic IP-адреси, Shield Advanced автоматично розгортає Network ACLs для мережі AWS під час атаки. Коли списки доступу до мережі знаходяться на межі мережі, Shield Advanced може забезпечити захист від більшості подій DDoS. Зазвичай Network ACL застосовуються поблизу ваших EC2 інстансів в Amazon VPC. Networ ACL може пом’якшити атаки масштабу, що не перевищують пропускної здатності Amazon VPC та інтерфейсів EC2 інстансів. Якщо мережевий інтерфейс, підключений до вашого EC2 інстнанса, може обробляти до 10 Гбіт/с, обсяги понад 10 Гбіт/с сповільнюються і, можливо, блокують трафік до цього інстансу. Під час атаки Shield Advanced підвищує пропускну здатність Network ACL до межі AWS, яка може обробляти кілька терабайт трафіку. Network ACL здатний забезпечити захист ресурсу, кількість трафіку на який під час атаки перевищує

звичайні можливості мережі. Коли ресурс знаходиться за Shield Advanced, Shield Advanced аналізує трафік та з часом визначає baseline. Він використовує цей baseline для виявлення аномалій у шаблонах трафіку, які можуть свідчити про DDoS-атаку. Точка, на якій Shield Advanced виявляє атаки та надсилає сповіщення або приймає міри, залежить від архітектури, використаної для веб-додатків. Baseline залежить від таких характеристик, як тип інстансу, який використовується, розмір інстансу та чи підтримує тип інстансу функціонал enhanced networking. Shield Advanced не приймає автоматично мір пом'якшення для атак прикладного рівня L7, якщо явно не налаштувати його для цього.

Будучи клієнтом AWS Shield Advanced, можна зв'язатися з цілодобовою групою реагування AWS Shield Response Team (SRT) за допомогою під час DDoS-атаки. Користувачі також отримують ексклюзивний доступ до розширених показників і звітів у режимі реального часу для детального аналізу атак на ресурси хмарної інфраструктури ваших веб-додатків в AWS. За допомогою SRT AWS Shield Advanced включає інтелектуальне виявлення та пом'якшення атак DDoS не лише для атак мережевого рівня L3 і транспортного рівня L4, а й атак прикладного рівня L7. Щоб скористатися послугами SRT, потрібно юути підписаним на план підтримки AWS «Business» або план «Enterprise». AWS Shield Advanced також пропонує певний захист від стрибків у рахунках AWS, які можуть виникнути в результаті DDoS-атаки на захищені ресурси.

2.11. AWS Web Application Firewall

AWS WAF — це брандмауер для веб-додатків, який дозволяє відстежувати запити HTTP(S), які приходять на такі сервіси, як:

- Amazon CloudFront;
- Amazon API Gateway;
- Application ELB;
- AWS AppSync GraphQL API.

На основі вказаних користувачем AWS WAF правил можна реалізувати користувацькі алгоритми обробки тих чи інших запитів, на основі таких атрибутів, як source IP, destination IP, HTTP payload, HTTP headers, cookie, сервіс пов'язаний з обробкою запитів, відповідає на запити потрібним вмістом або кодом статусу HTTP 403 Forbidden. CloudFront також можна налаштувати, щоб повертати власну сторінку помилки, коли запит заблоковано.

AWS WAF використовується, щоб керувати тим, як CDN Amazon CloudFront, API Gateway, Application ELB і AWS AppSync GraphQL реагують на веб-запити HTTP(S) з точки зору безпеки.

Засоби, які використовує AWS WAF:

- web ACLs – список керування доступом L7 для захисту набору ресурсів AWS. Правила Web ACL визначають критерії перевірки веб-запитів і вказують, як обробляти запити, які відповідають критеріям. Встановлюється дія за замовчуванням для Web-ACL, яка вказує, блокувати чи дозволяти певні запити, які проходять перевірку правил;
- rules (правила) – кожне правило містить заяву, яка визначає критерії перевірки та дії, які потрібно виконати, якщо веб-запит відповідає критеріям. Коли веб-запит відповідає критеріям, це збігається. Можна налаштувати правила, щоб блокувати відповідні запити, дозволяти їм проходити, підраховувати їх або запускати для них елементи керування CAPTCHA;
- rules groups (групи правил) – правила можна використовувати окремо або в групах правил, які можна використовувати повторно. AWS Managed Rules і постачальники послуг на AWS Marketplace надають групи правил які підтримують самостійно. Можна визначити власні групи правил.

Після створення Web ACL можна пов'язати його з одним або кількома ресурсами AWS. AWS WAF доступний у регіонах, перелічених на кінцевих точках служби AWS. Для CloudFront distribution AWS WAF доступний у всьому світі, але для роботи потрібно вибрати регіон США Схід (Северна Вірджинія). Ви повинні створити свій веб-ACL, використовуючи регіон США US East (N. Virginia).

Правила налаштування Web ACL з одним або кількома ресурсами AWS:

- можна пов’язати кожен ресурс AWS лише з одним Web ACL. Відношення між Web ACL і ресурсами AWS є «один до багатьох»;
- можна пов’язати Web ACL з одним або кількома CloudFront distribution. Не можна пов’язувати Web ACL, вже пов’язаний з CloudFront distribution, з будь-яким іншим типом ресурсу AWS.

AWS WAF використовує одиниці Web ACL Capacity Units (WCU) для обчислення та керування операційними ресурсами, які необхідні для виконання правил, груп правил і Web ACL. AWS WAF встановлює обмеження в допустимій кількості WCU на точку прослуховування трафіку. AWS WAF обчислює потужність по-різному для кожного типу правила, щоб відобразити відносну вартість кожного правила. Прості правила, виконання яких мало коштує, використовують менше WCU, ніж складніші правила, які використовують більше обчислювальної потужності. Наприклад, оператор правила обмеження розміру HTTP payload використовує менше WCU, ніж оператор, який перевіряє набір шаблонів регулярних виразів.

AWS WAF керує потужністю для правил, груп правил і Web ACL:

- rule capacity (ємність правила) – AWS WAF обчислює ємність правила при його створенні. Також можна отримати уявлення про ємність, необхідну для різних типів правил на консолі AWS WAF, створивши Web ACL або групу правил і додавши до нього окремі правила. На консолі відображаються одиниці ємності, які використовуються під час додавання правил;
- rule group capacity (ємність групи правил) – AWS WAF вимагає, щоб кожній групі правил при створенні призначалася незмінна ємність. Це справедливо для AWS Managed rule groups і rule groups, які створюєтевручну за допомогою AWS WAF. При зміні rule group, зміни повинні враховувати об’єм WCU у межах її можливостей. Це гарантує, що Web ACL, які використовують групу правил, залишаються в межах своєї максимальної ємності;
- web ACL capacity (ємність Web ACL) – максимальна ємність веб-ACL становить 1500, що достатньо для більшості випадків використання.

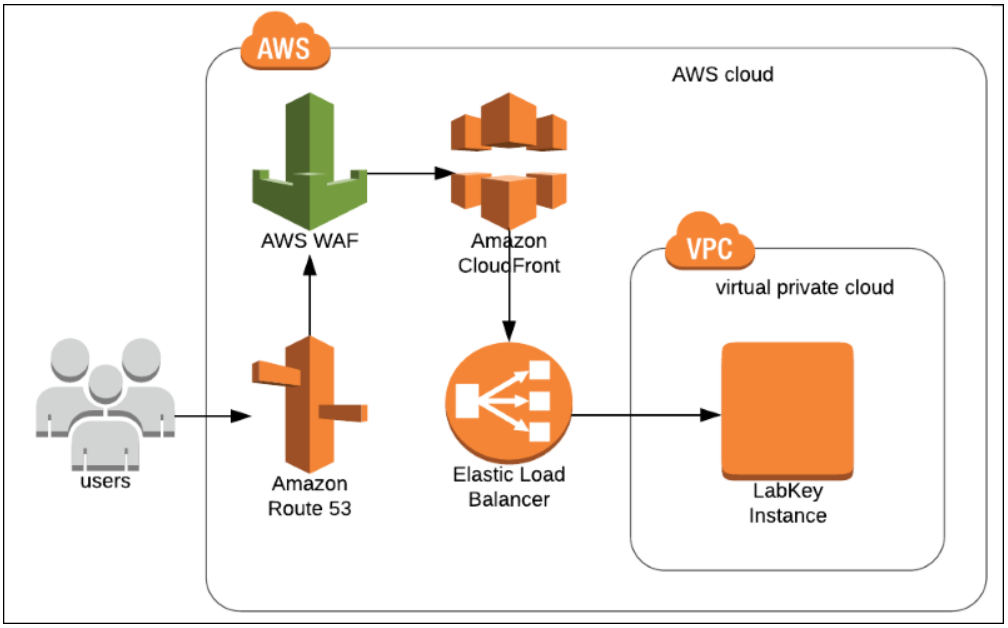


Рис. 2.4. Схематичне підключення AWS WAF перед балансувальником навантаження на веб-додаток

2.12. AWS Secrets Manager

Раніше, при створенні веб-додатку з доступом до БД, розробники зазвичай вбудовували ідентивікатор та аутентифікатор (або секрет, від англійського значення «secret»), для доступу до БД безпосередньо в програму. Коли настававав час змінити облікові дані, потрібно було доводилось витратити час на оновлення програми, щоб використовувати нові облікові дані. Це призводило до частих збоїв так як певні компоненти додатку могли, з урахуванням людського фактору, залишитись з застарілими даними про аутентифікацію. Через цей ризик багато клієнтів вирішували відмовитись від регулярної ротації секретів, що фактично замінює один ризик іншим. Secrets Manager дозволяє замінити жорстко запрограмовані облікові дані у коді, включаючи паролі, викликом API до Secrets Manager для програмного отримання секрету. Це допомагає гарантувати, що секрет не може бути скомпрометований кимось, хто перевіряє код чи отримав до нього доступ нелегітимним шляхом, оскільки секрет більше не заданий в коді. Крім того, можна налаштувати Secrets Manager на автоматичну ротацію секрету

відповідно до заданого розкладу. Це дозволяє замінити довгострокові секрети короткостроковими, значно знижуючи ризик компромісу.

Наступна діаграма ілюструє найпростіший сценарій. На діаграмі показано, що можна зберігати облікові дані для БД у Secrets Manager, а потім використовувати ці облікові дані в програмі для доступу до БД.

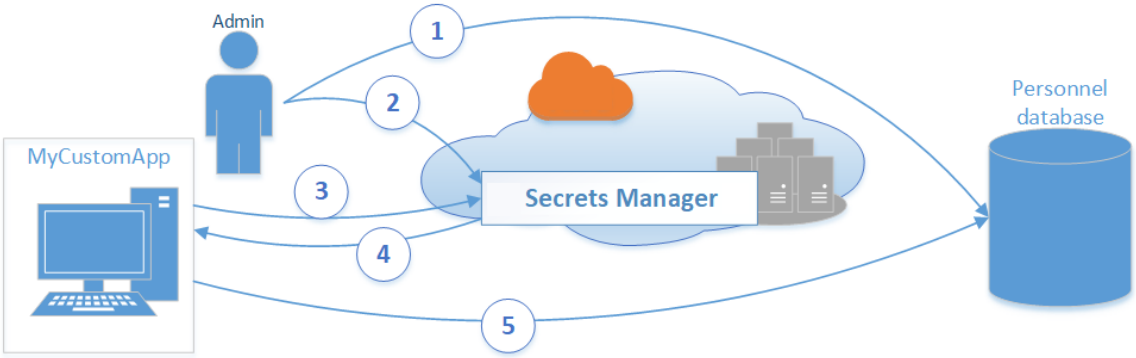


Рис. 2.5. Схема використання облікових даних БД, що зберігаються в AWS Secrets Manager

Використання облікових даних БД, що зберігаються в AWS Secrets Manager поетапно розбивається на такі кроки:

1. Адміністратор БД створює набір облікових даних у базі даних персоналу для використання програмою під назвою «MyCustomApp». Адміністратор також налаштовує ці облікові дані з дозволами, необхідними для доступу програми до БД персоналу.

2. Адміністратор БД зберігає облікові дані як секрет у AWS Secrets Manager під назвою «MyCustomAppCreds». Потім Secrets Manager шифрує та зберігає облікові дані в секреті як захищений секретний текст.

3. Коли MyCustomApp отримує доступ до бази даних, програма запитує у AWS Secrets Manager секрет під назвою «MyCustomAppCreds».

4. AWS Secrets Manager отримує секрет, розшифровує захищений секретний текст і повертає секрет клієнтській програмі через захищений HTTPS канал.

5. Клієнтська програма аналізує облікові дані, рядок підключення та будь-яку іншу необхідну інформацію з відповіді, а потім використовує цю інформацію для доступу до сервера бази даних.

Висновки розділу 2.

Було проаналізовано модель розподілу відповідальності за захист ресурсів в хмарі AWS між користувачем хмарних обчислень та компанією AWS та доведено її доцільність. В розділі наведені інструменти, технології та методи забезпечення захисту хмарної інфраструктури веб-ресурсів від AWS й водночас описані принципи їх роботи. Це дозволяє оцінити можливості по організації захищеного середовища для активів компанії в хмарі й підводить до розділу 3, що зосереджений на демонстрації практичного застосування даних технологій.

3 ТЕХНОЛОГІЇ ЗАХИСТУ ХМАРНОЇ ІНФРАСТРУКТУРИ, РЕАЛІЗАЦІЯ ЗАХИСТУ ТА НАДАННЯ РЕКОМЕНДАЦІЙ ЩОДО СТВОРЕННЯ ДАНОЇ ЗАХИЩЕНОЇ ІНФРАСТРУКТУРИ З ВИКОРИСТАННЯМ ЗАСОБІВ AMAZON AWS

3.1 Налаштування розмежування прав доступу з AWS IAM

Почнемо реалізацію захисту з розмежування прав доступу до ресурсів в акаунті, с якого здійснюється управління хмарою. Так як за більшість операцій в хмарі стягується оплата за утилізацію обчислювальних потужностей на базі яких ці операції здійснюються, в роботі використовуються короткі з точки зору часу на конфігурацію приклади, що описують конкретну ситуацію.

Визначимось з приналежністю користувачів до груп та їх правами доступу.

Таблиця 3.1.

Приналежність користувачів до груп та їх права доступу

Користувач	Група	Дозволи
user-1	S3-Support	Read-only доступ до S3
user-2	EC2-Support	Read-only доступ до EC2
user-3	EC2-Admin	View, start, stop для EC2 інстансів

У консолі перейдемо до налаштувань IAM та створимо користувачів, додавши їх у відповідні групи. Створимо Inline policy для групи Ec2-Admin. Цей тип політик підходить для створення унікальних прав доступу для конкретної групи. Код політики додано у «Додатку А». Основні права користувача визначаються трердженням Action, що дозволяє наступні дії:

- "ec2:Describe*";
- "ec2:StartInstances";
- "ec2:StopInstances";

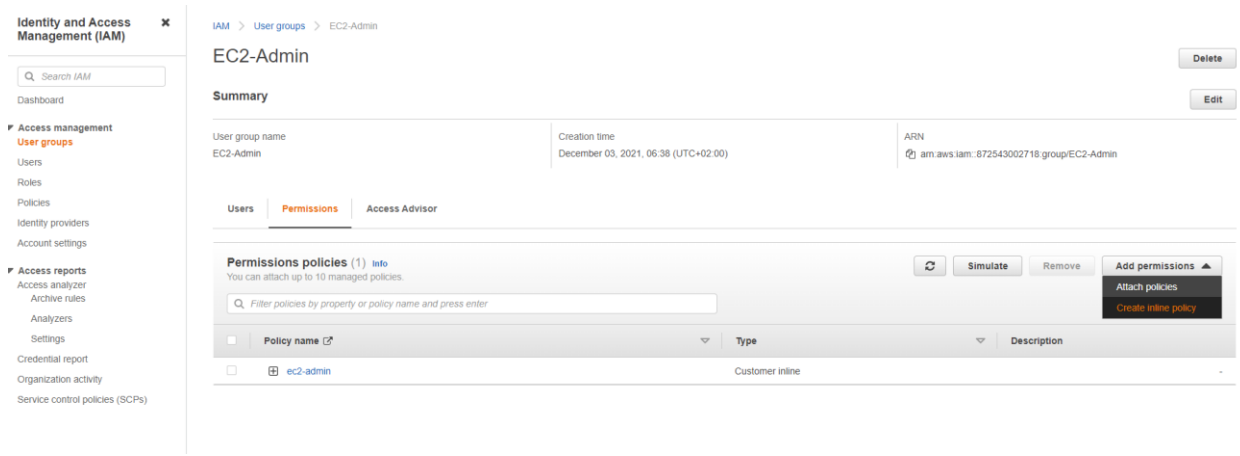


Рис. 3.1. Створення Inline policy для групи EC2-Admin

Для групи EC2-Support виберемо «Add permissions > Attach policies» та зі списку готових AWS Managed policies додамо AmazonEC2ReadOnlyAccess.

Для групи S3-Support виберемо «Add permissions > Attach policies» та зі списку готових AWS Managed policies додамо AmazonS3ReadOnlyAccess. Перевіримо що права для користувачів вступили в дію на прикладі user-3.

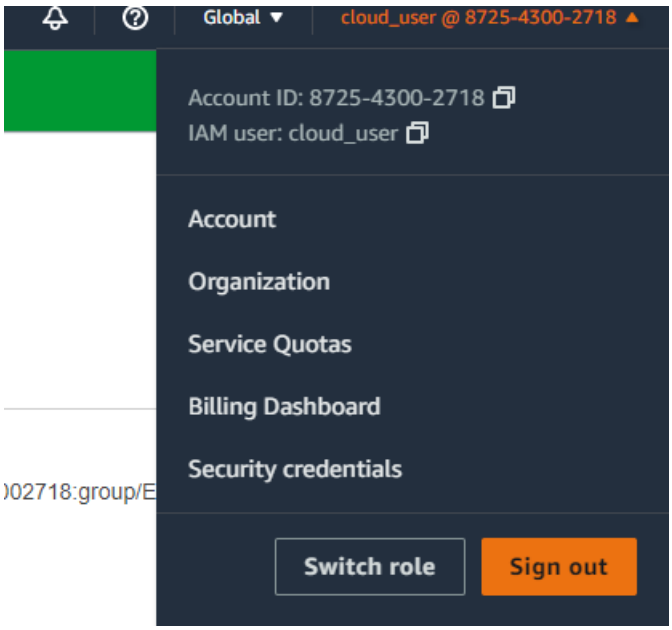


Рис. 3.2. Інформація про обліковий запис AWS

В правому верхньому кутку консолі натиснемо на ім’я поточного користувача та скопіюємо Account ID й перейдемо перейдемо на signin.aws.amazon.com.

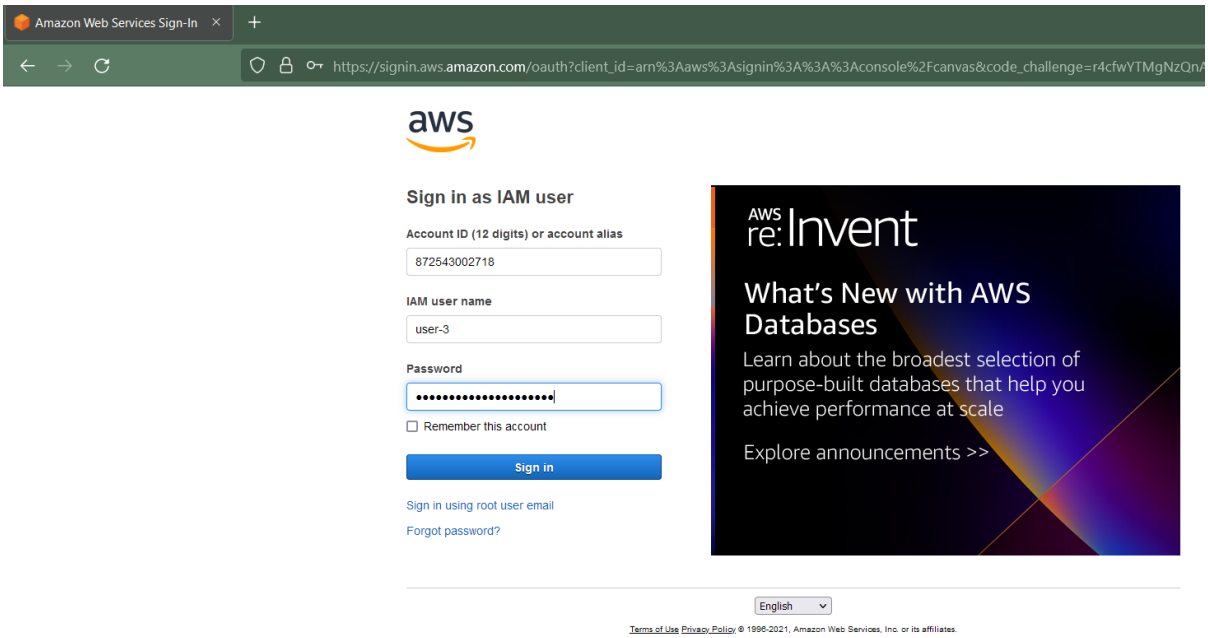


Рис. 3.3. Авторизація в якості IAM користувача

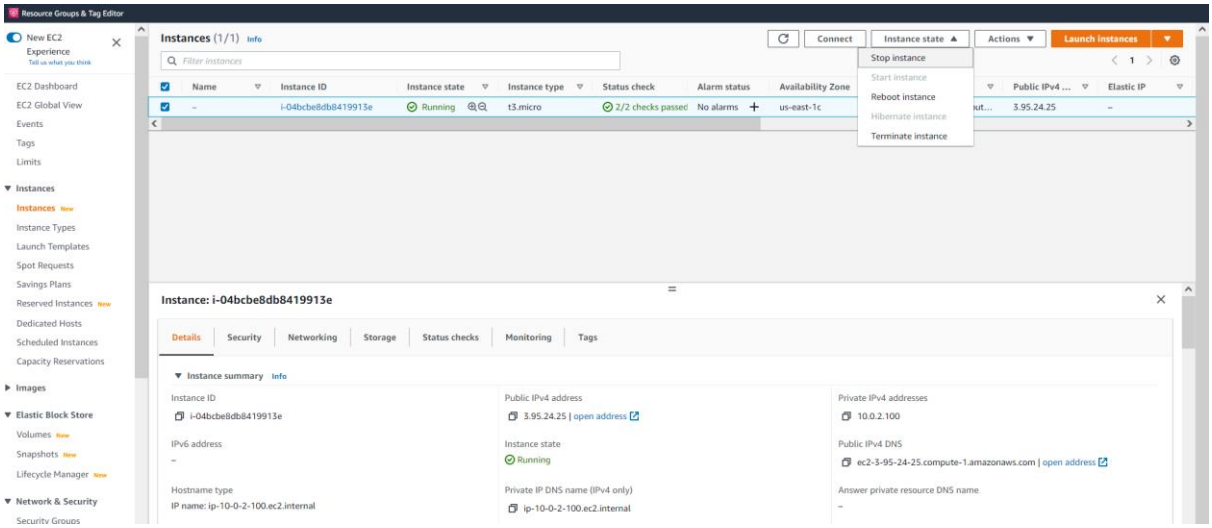


Рис. 3.4. Операції з EC2 інстансами

Перейшовши в консоль EC2 можна помітити, що нам доступна опція зупинки віртуальної машини, як це і передбачено політикою, описаною в «Додатку А». Спробуємо виконати дію на яку користувач не має повноважень, наприклад від’єднати диск від інстансу EC2/

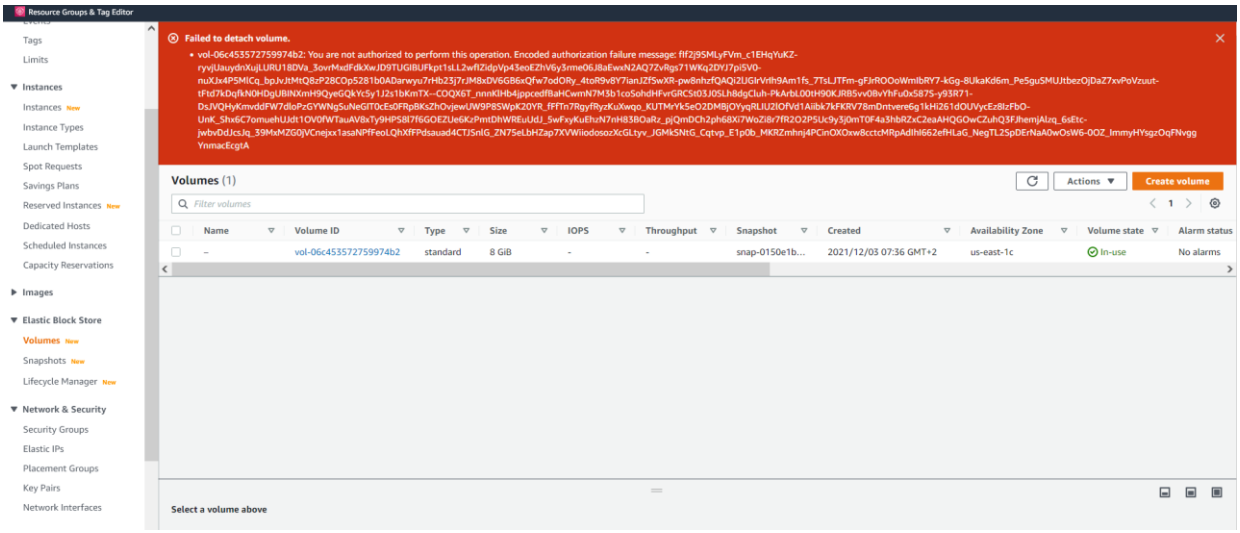


Рис. 3.5. Помилка від'єднання диска від EC2 інстансу через брак повноважень

3.2. Активація threat intelligence функціоналу з Amazon GuardDuty

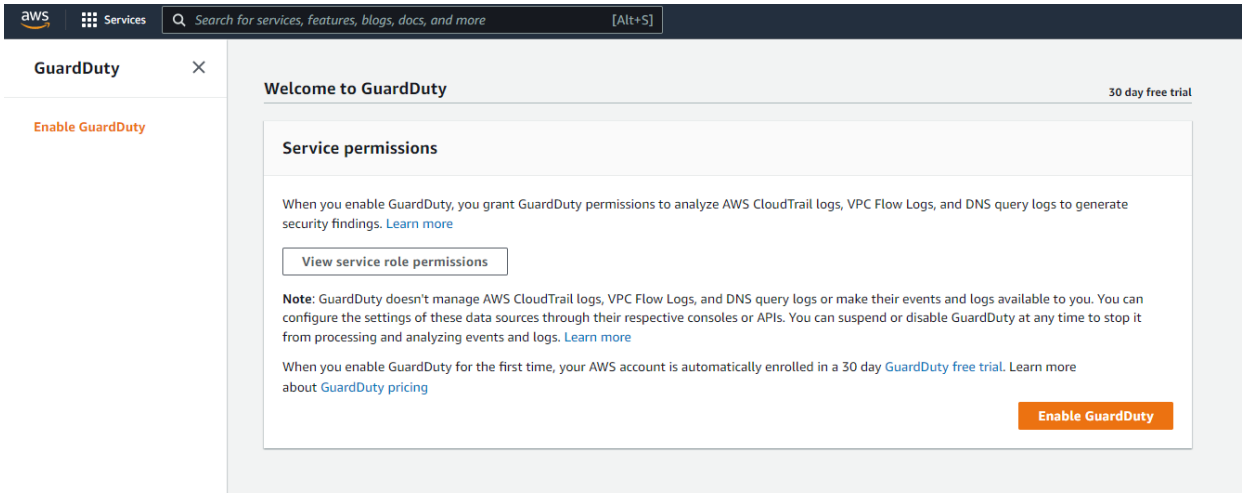


Рис. 3.6. Активація функціоналу GuardDuty

При активації сервісу GuardDuty AWS пропонуэ автоматично надати сервісу права на доступ до потрібної сервісу інформації з таких джерел, як CloudTrail, VPC Flow Logs, DNS logs. Для демонстрації було додано штучно згенеровані алерти, розглянемо інформацію що міститься в них.

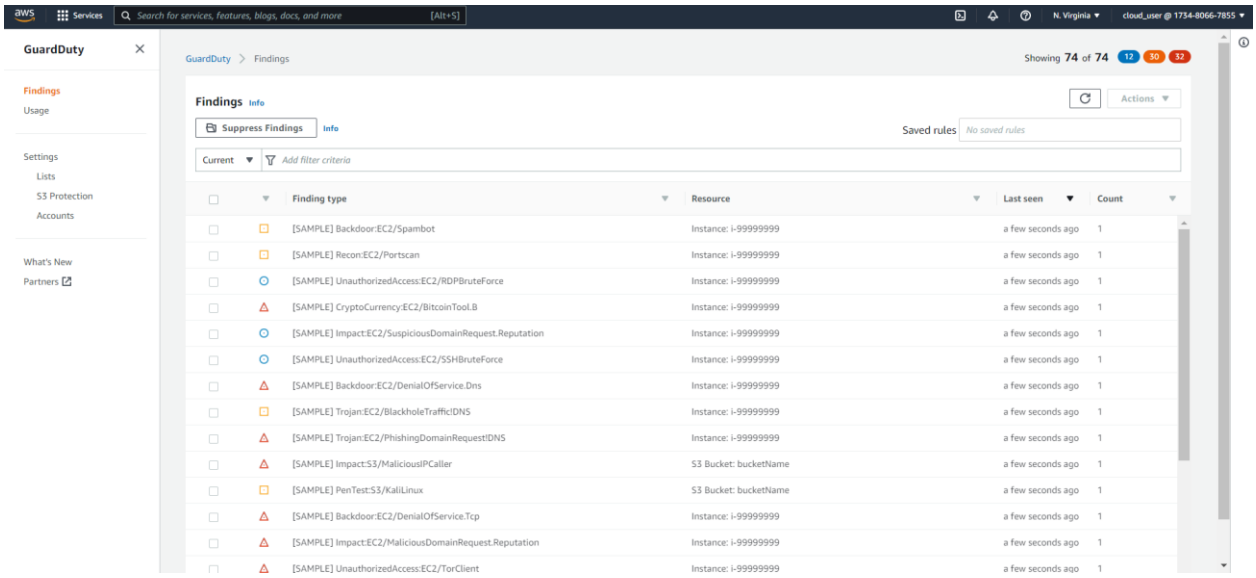


Рис. 3.7. Інформаційна панель GuardDuty

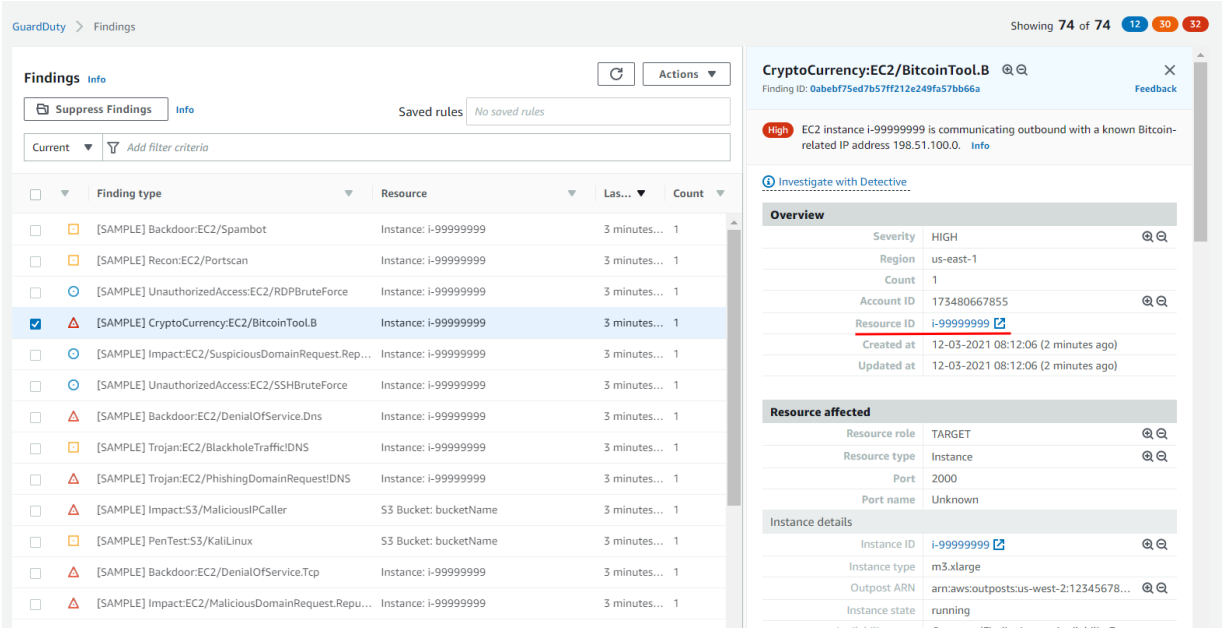
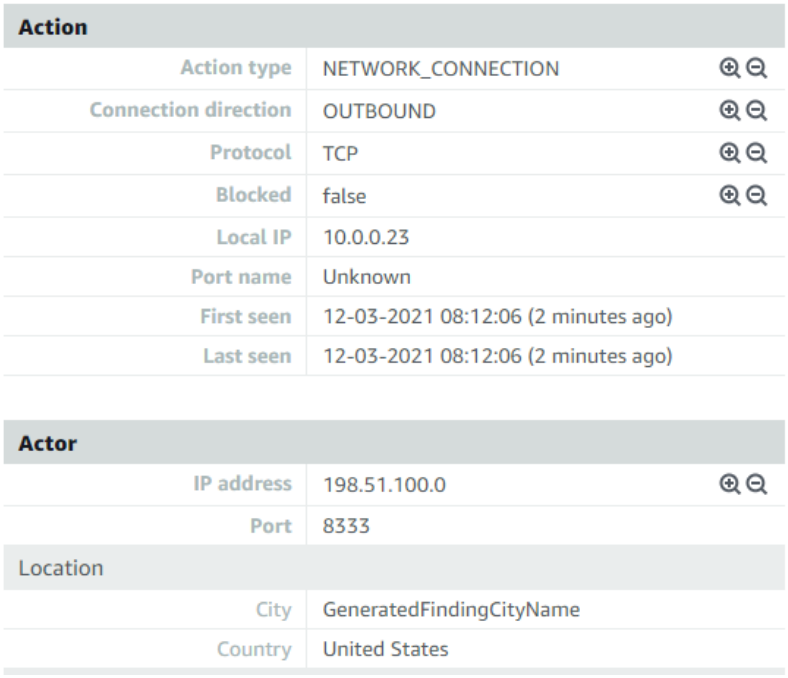


Рис. 3.8. EC2 інстанс ймовірно заражено майнером Bitcoin

Оцінивши надану по ряду показників інформацію від GuardDuty дізнаємось який саме EC2 інстанс було заражено майнером а також пролиставши до низу дізнаємось інформацію про контрагента і його дії.



Action	
Action type	NETWORK_CONNECTION
Connection direction	OUTBOUND
Protocol	TCP
Blocked	false
Local IP	10.0.0.23
Port name	Unknown
First seen	12-03-2021 08:12:06 (2 minutes ago)
Last seen	12-03-2021 08:12:06 (2 minutes ago)

Actor	
IP address	198.51.100.0
Port	8333
Location	
City	GeneratedFindingCityName
Country	United States

Рис. 3.9. Інформація про нападника та його дії з нашим ресурсом AWS

Наступним кроком буде:

1. Ізоляція даної віртуальної машини через перерозгортання в спеціальній виділеній під форензіку мережі.
2. Додавання визначеної IP і порту (якщо він не використовується для корисних задач) в чорний список Network ACL межуючих з Інтернет мереж.
3. Вивчення EC2 інстансу спеціалізованими інструментами, наприклад використовуючи дистрибутив Kali Linux з тієї ж ізольованої мережі для визначення шляху зараження й запобігання подальших інцидентів подібного характеру.

3.3. Налаштування сканування вмісту на наявність PII з Amazon Macie

Нагадаємо, що PII це інформація, здатна однозначно чи ненапряму ідентиівікувати особу. Якщо в компанії збираються дані користувачів, вони повинні оброблятись, зберігатись та передаватись з рінем безпеки, зазначеним в

ПБ підприємства. AWS Macie може сканувати S3 на наявність PII, чим забезпечувати спеціалістів безпеки відомостями про можливі джерела витоку інформації для прийняття відповідних мір.

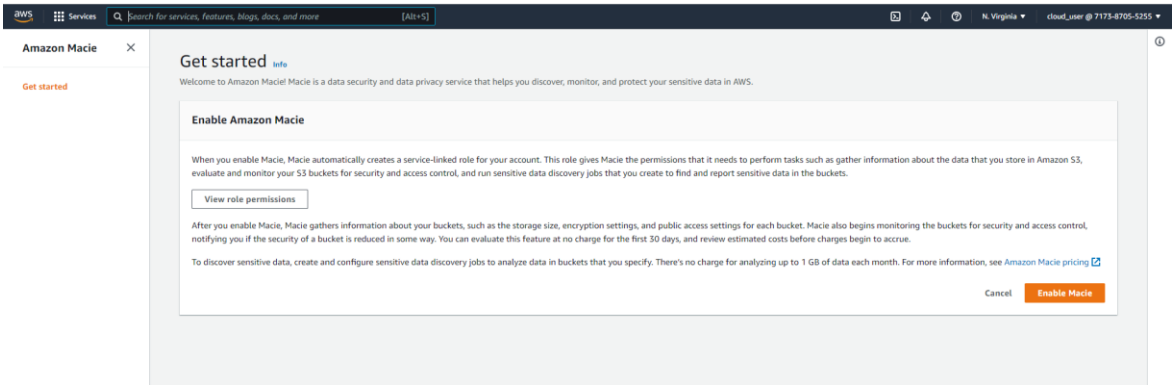


Рис. 3.10. Активація Macie

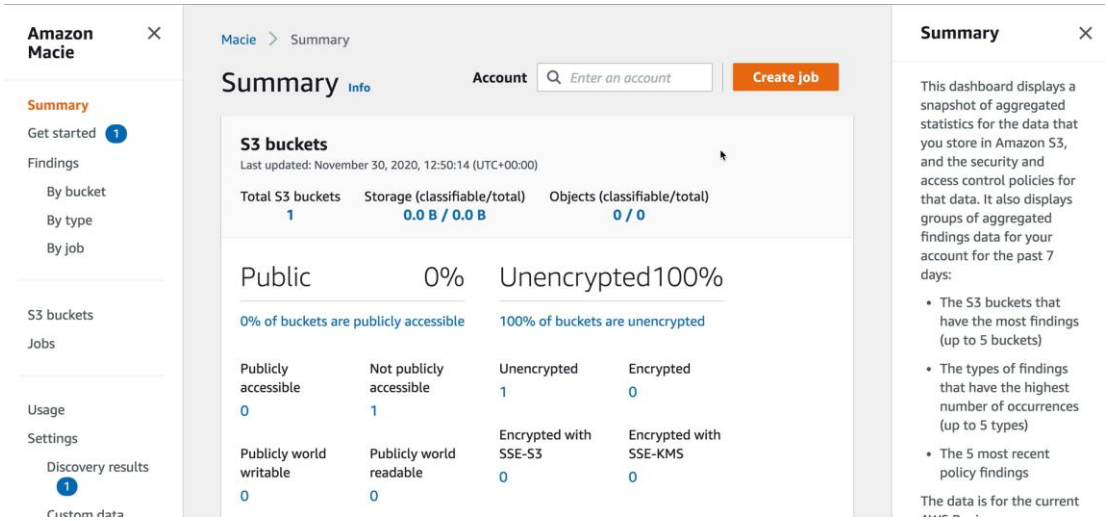
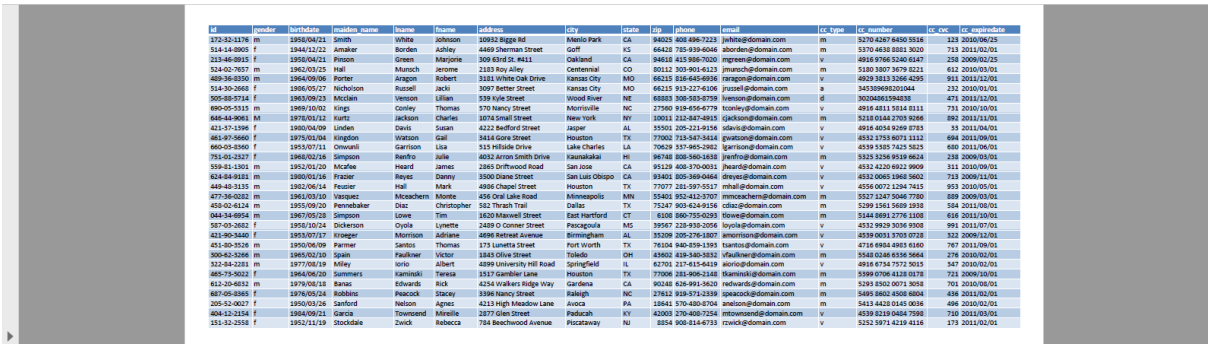


Рис. 3.11. Інформаційна панель Macie

Створимо S3 bucket та додамо в нього pdf файл з наступним вмістом тестових даних.



id	gender	birthdate	maiden_name	lname	fname	address	city	state	zip	phone	email	cc_type	cc_number	cc_cvc	cc_expiredate
172-32-1176	m	1958/04/21	Smith	White	Johnson	10932 Bigge Rd	Menlo Park	CA	94025	408 496-7223	jwhite@domain.com	m	5370 4267 6450 5516	123	2010/06/25
514-14-8905	f	1944/12/22	Amaker	Borden	Ashley	4469 Sherman Street	Goff	KS	66428	785-939-6046	aborden@domain.com	m	5370 4838 8881 3020	713	2011/02/01
213-46-8915	f	1958/04/21	Pinson	Green	Marjorie	309 63rd St. #411	Oakland	CA	94618	415 986-7020	mgreen@domain.com	v	4916 9766 5240 6147	258	2009/02/25
524-02-7657	m	1962/03/25	Hall	Mursch	Jerome	2183 Roy Alley	Centennial	CO	80112	303-901-6123	jmursch@domain.com	m	5180 3307 3679 8221	612	2010/03/01
489-36-0250	m	1964/09/06	Porter	Aragon	Robert	3181 White Oak Drive	Kansas City	MO	66213	816-645-6936	raragon@domain.com	v	4929 3813 3266 4295	911	2011/12/01
514-30-2668	f	1906/05/27	Nicholson	Russell	Jacki	3097 Better Street	Kansas City	MO	66215	913-227-6306	jrussell@domain.com	a	345389698201044	232	2010/01/01
505-88-5714	f	1963/09/23	Mcclain	Venson	Lillian	539 Kyle Street	Wood River	NE	68883	308-583-8759	lvenson@domain.com	d	3020486159483	471	2011/12/01
690-05-5315	m	1969/10/02	Kings	Conley	Thomas	570 Nancy Street	Morrisville	NC	27560	919-656-6779	tconley@domain.com	v	4916 4811 5814 8111	731	2010/10/01
646-44-9061	M	1978/01/12	Kurtz	Jackson	Charles	1074 Small Street	New York	NY	10011	212-847-4915	cjackson@domain.com	m	5218 0144 2703 9266	892	2011/11/01
421-37-1596	f	1980/04/09	Linden	Davis	Susan	4222 Bedford Street	Jasper	AL	35501	205-221-9156	sdavis@domain.com	v	4916 4054 9269 8783	33	2011/04/01
661-97-5660	f	1975/01/04	Kingdon	Watson	Gail	3414 Gore Street	Houston	TX	77002	713-547-3414	gwatson@domain.com	v	4532 1753 6071 1112	694	2011/09/01
660-05-8560	f	1953/07/11	Onwuanli	Garrison	Lisa	515 Hillside Drive	Lake Charles	LA	70629	337-965-2982	lgarrison@domain.com	v	4539 5385 7425 5825	680	2011/06/01
751-01-2527	f	1968/02/16	Simpson	Renfro	Julie	4032 Arron Smith Drive	Kaunakakai	HI	96748	808-560-1638	jrenfro@domain.com	m	5325 3256 9519 6624	238	2009/03/01
559-81-1301	m	1952/01/20	Mcafee	Heard	James	2865 Driftwood Road	San Jose	CA	95129	408-370-0031	jheard@domain.com	v	4532 4220 6922 9909	311	2010/09/01
624-84-9181	m	1980/01/16	Frazier	Reyez	Danny	3500 Diane Street	San Luis Obispo	CA	93401	805-369-0464	dreyez@domain.com	v	4532 0065 1968 5602	713	2009/11/01
449-48-3135	m	1982/06/14	Feusier	Hall	Mark	4906 Chapel Street	Houston	TX	77077	281-597-5517	mhall@domain.com	v	4556 0072 1294 7415	953	2010/05/01
477-36-0282	m	1961/03/10	Vasquez	Mceachern	Monte	456 Oral Lake Road	Minneapolis	MN	55401	952-412-3707	mmceachern@domain.com	m	5527 1247 5046 7780	889	2009/03/01
459-02-6124	m	1955/09/20	Pennebaker	Diaz	Christopher	582 Thrash Trail	Dallas	TX	75247	903-424-9156	cdiaz@domain.com	m	5299 1561 5669 1898	584	2011/08/01
044-34-6954	m	1967/05/28	Simpson	Lowe	Tim	1620 Maxwell Street	East Hartford	CT	6108	860-755-0293	tlowe@domain.com	m	5144 8691 2776 1308	616	2011/10/01
587-03-2682	f	1958/10/24	Dickerson	Oyola	Lynette	2489 O Conner Street	Pascagoula	MS	39567	228-938-2056	loyola@domain.com	v	4532 9929 3056 9908	991	2011/07/01
421-90-3440	f	1953/07/17	Kroeger	Morrison	Adriane	4696 Retreat Avenue	Birmingham	AL	35209	205-276-1807	amorrison@domain.com	v	4539 0031 3703 0728	322	2009/12/01
451-80-3526	m	1950/06/09	Parmer	Santos	Thomas	173 Lunetta Street	Fort Worth	TX	76104	940-859-1593	tsantos@domain.com	v	4716 6984 4983 6160	767	2011/09/01
300-62-5266	m	1965/02/10	Spain	Faulkner	Victor	1845 Olive Street	Toledo	OH	43602	419-340-5832	vfaulkner@domain.com	m	5548 0246 6536 5664	276	2010/02/01
322-84-2281	m	1977/08/19	Miley	Iorio	Albert	4899 University Hill Road	Springfield	IL	62701	217-615-6419	aiorio@domain.com	v	4916 6754 7572 5015	347	2010/02/01
465-75-5022	f	1964/06/20	Summiers	Kaminski	Teresa	1517 Gambier Lane	Houston	TX	77008	281-906-2148	tkaminski@domain.com	m	5399 0706 4128 0178	721	2009/10/01
612-20-6852	m	1979/08/18	Banas	Edwards	Rick	4254 Walkers Ridge Way	Gardena	CA	90248	626-991-3620	redwards@domain.com	m	5293 8502 0071 3058	701	2010/08/01
687-05-8365	f	1976/05/24	Robbins	Peacock	Stacey	3396 Nancy Street	Raleigh	NC	27612	919-571-2339	speacock@domain.com	m	5495 8602 4508 6804	436	2011/02/01
205-52-0027	f	1950/03/26	Sanford	Nelson	Agnes	4213 High Meadow Lane	Avoca	PA	18641	570-480-8704	anelson@domain.com	m	5413 4420 0145 0036	496	2010/02/01
404-12-2154	f	1984/09/21	Garcia	Townsend	Mireille	2877 Glen Street	Paducah	KY	42003	270-408-7254	mtownsend@domain.com	v	4539 8219 0484 7598	710	2011/03/01
151-32-2558	f	1952/11/19	Stockdale	Zwick	Rebecca	784 Beechwood Avenue	Piscataway	NJ	8854	908-814-6733	rzwick@domain.com	v	5252 5971 4219 4116	173	2011/02/01

Рис. 3.12. Файл з PII даними

На інформаційній панелі Macie натиснемо на кнопку «Create job». Виберемо в налаштуваннях «One-time job». Для повсякденного використання краще вибрати регулярну перевірку.

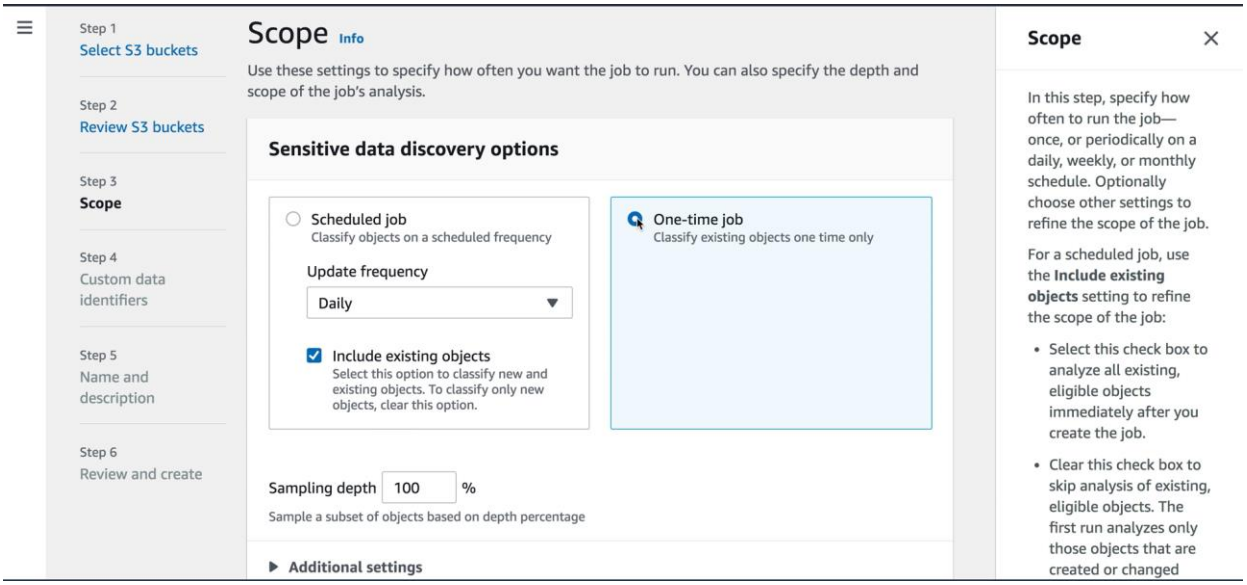


Рис. 3.13. Створення одноразової перевірки S3 bucket

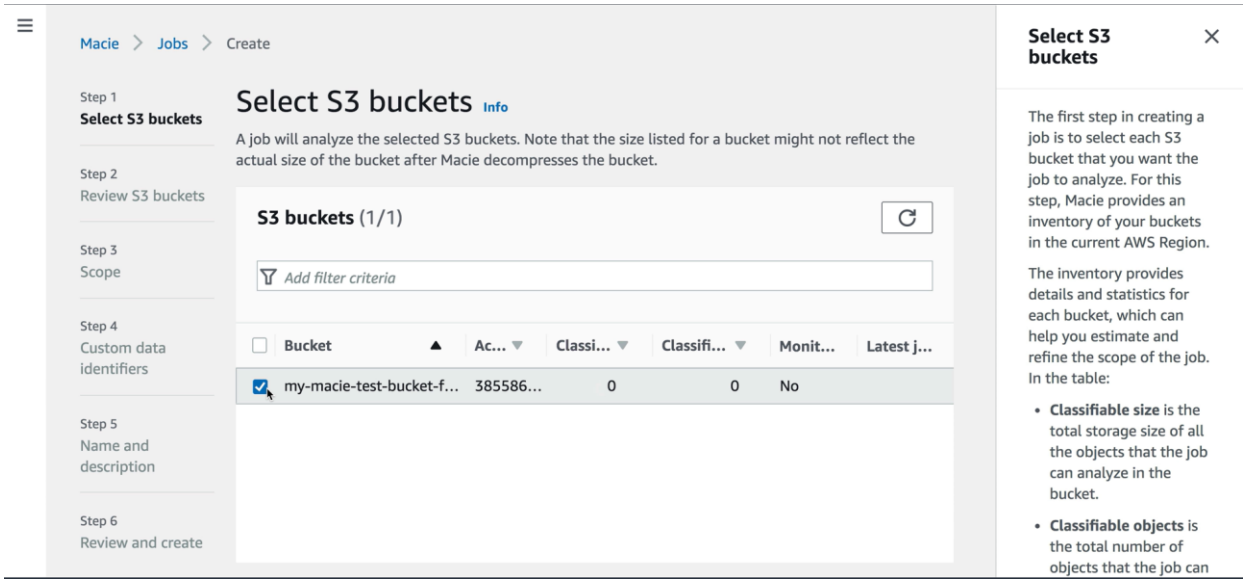


Рис. 3.14. Вибір S3 bucket'ів на перевірку

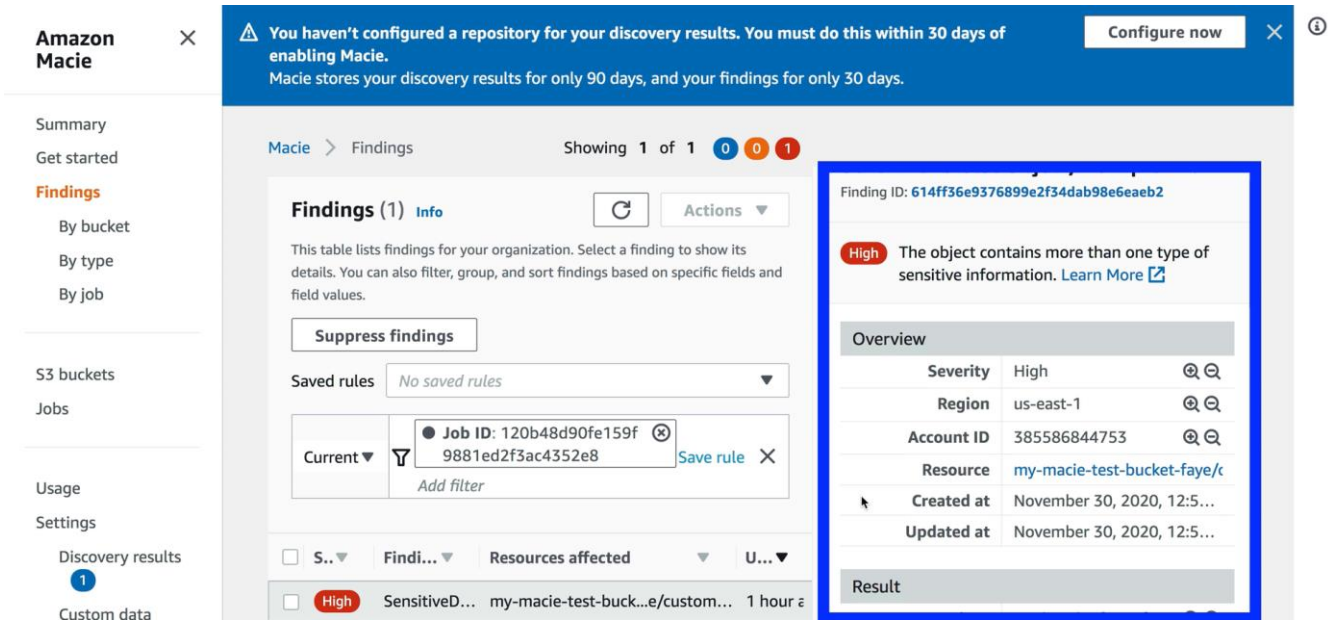


Рис. 3.15. Перегляд результатів скану Macie

Рис. 3.16. Деталі про знайдені дані PII

Після того, як Macie здійснить перевірку на вкладці «Findings» будуть знаходитись деталі про S3 bucket'и. В видачі отримано сповіщення про 1 S3 bucket в який і було додано файл з чутливою інформацією. Зі списку знахідок можем виокремити 28 адрес та 29 номерів банківських карт. Це дуже позитивний результат роботи системи, що здатна перевіряти величезні об'єми різних типів даних.

Розглянемо повний набір типів даних що може аналізувати Macie.

Таблиця 3.2.

Перелік типів даних що може аналізувати Macie

Тип файлу або форми зберігання	Опис	Розширення файлів
Big data	Контейнери об’єктів Apache Avro та файли Apache Parquet	.avro, .parquet
Архіви	GNU Zip, TAR, ZIP	.gz, .gzip, .tar, .zip
Документи	Файли формату Adobe Portable Document, Microsoft Excel, Microsoft Word	.doc, .docx, .pdf, .xls, .xlsx
Текст	Небінарні текстові фавйли як comma-separated values (CSV) , Hypertext Markup Language (HTML), JavaScript Object Notation (JSON), JSON Lines, plain-text документи, tab-separated values (TSV), Extensible Markup Language (XML)	.csv, .htm, .html, .json, .jsonl, .tsv, .txt, .xml, and others (depending on the type of non-binary text file)

Macie – потужний інструмент детекції й класифікації PII в хмарному середовищі, який варто розглянути до використання кожному хто працює з даними, потребує їх класифікації та, наприклад, повинен слідувати вимогам GDPR й зберігати чутливі дані користувачів в шифрованому вигляді, що Macie може постійно моніторити.

3.4. Відновлення хмарної інфраструктури після інциденту за допомогою сервісів Amazon Config та Amazon CloudTrail

Для практичної демонстрації завдання з яким може зіткнутись фахівець з організації ІБ в компанії з хмарною інфраструктурою, розглянемо ситуацію,

коли начальник ставить завдання провести випробування системи реагування на інциденти і зупинити EC2 інстанс, а також видалити мережеві маршрути та правила, щоб перевірити, чи можна вивести з експлуатації мережу VPC і корпоративний сервер EC2.

Невдовзі після внесення змін тест системи реагування на інциденти та алгоритми відновлення зазнають невдачі, і клієнти повідомляють, що не можуть підключитися до своїх програм. Постає завдання скасувати всі внесені зміни. Спробуємо виконати імітацію відновлення до недавнього стану системи за допомогою сервісів AWS CloudTrail і AWS Config, щоб переконатися, що всі зміни враховані та повернуті для відновлення хмарної інфраструктури. Цей приклад продемонструє можливості CloudTrail та Config й покаже як слід користуватись цими інструментами, щоб повернути будь-які несподівані чи випадкові зміни в обліковому записі та визначити, що саме було змінено.

Вхідні дані:

- CorporateServer – корпоративний сервер, на якому проводиться випробування;
- CorporateApplicationServerSecurityGroup – Security Group, що фільтрує трафік на рівні інстансу CorporateServer. На вхід сервера дозволено HTTP та HTTPS трафік з мережі 172.18.0.0/16. На вихід дозволено будь-який трафік;
- PrivateAZ1 – приватна мережа 172.18.0.0/21, в якій знаходиться CorporateServer. Доступ до Інтернет забезпечується через NAT, для можливості оновлення ПЗ;

Перейдем до панелі управління EC2 та зупинемо CorporateServer згідно легенди.

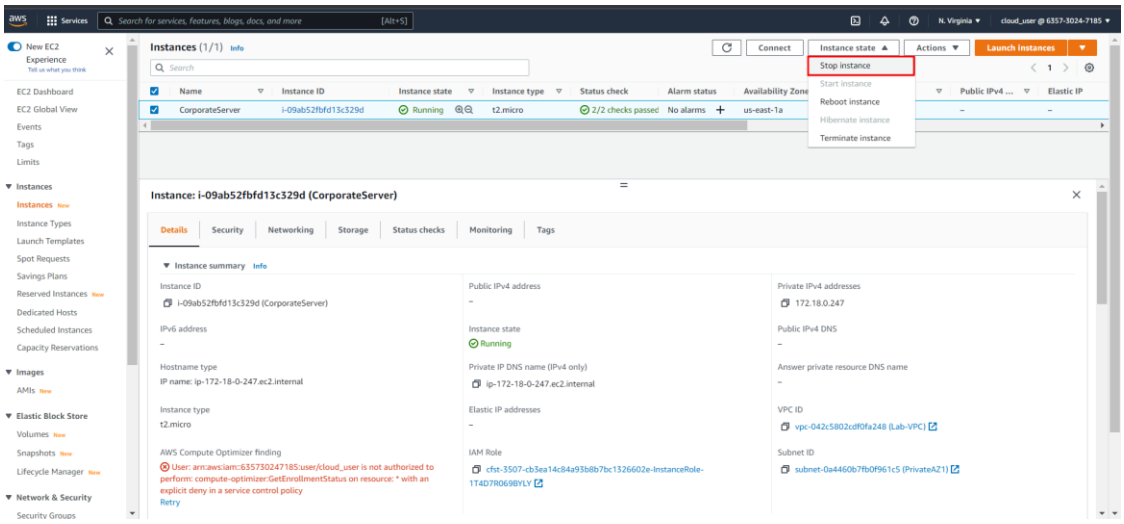


Рис. 3.17. Зупинка інстансу CorporateServer

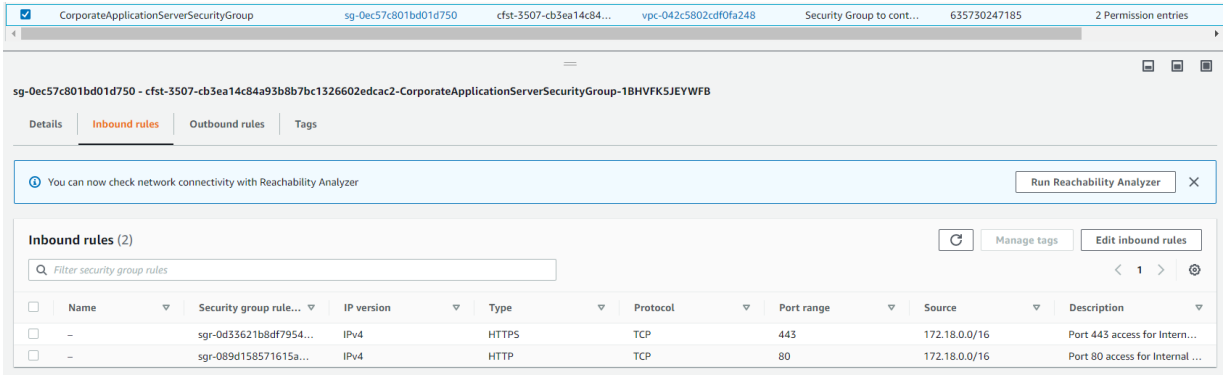


Рис. 3.18. Правила фільтрації вхідного трафіку на рівні інстансу CorporateServer

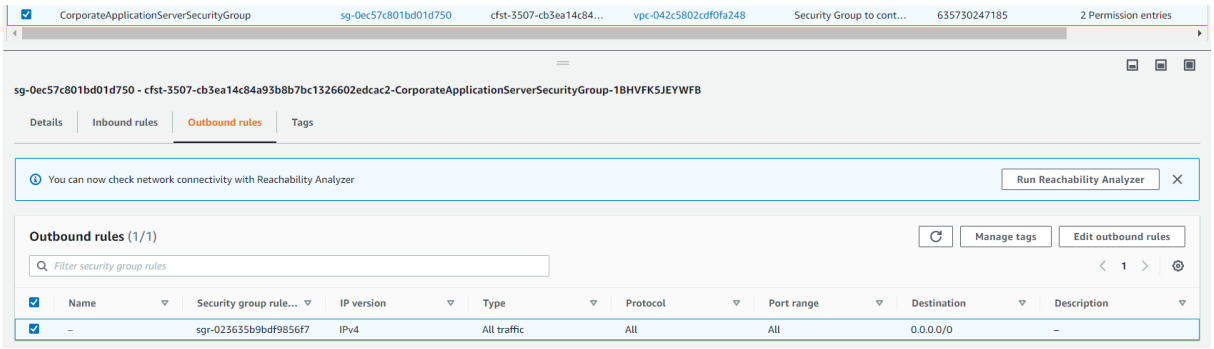


Рис. 3.19. Правила фільтрації вихідного трафіку на рівні інстансу CorporateServer

Видалимо маршрут до шлюзу NAT в таблиці маршрутизації мережі PrivateAZ1 корпоративного серверу CorporateServer.

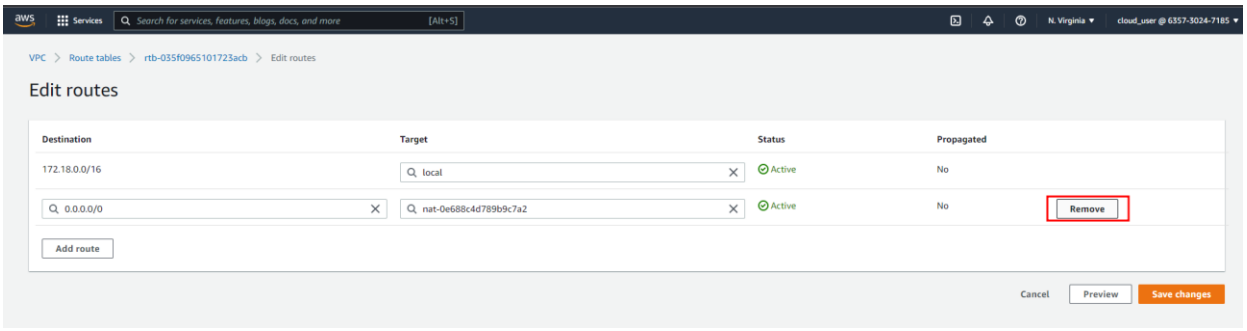


Рис. 3.20. Видалення маршруту до шлюзу NAT

Видалимо правило CorporateApplicationServerSecurityGroup, що дозволяє вихідний трафік на будь-який destination IP.

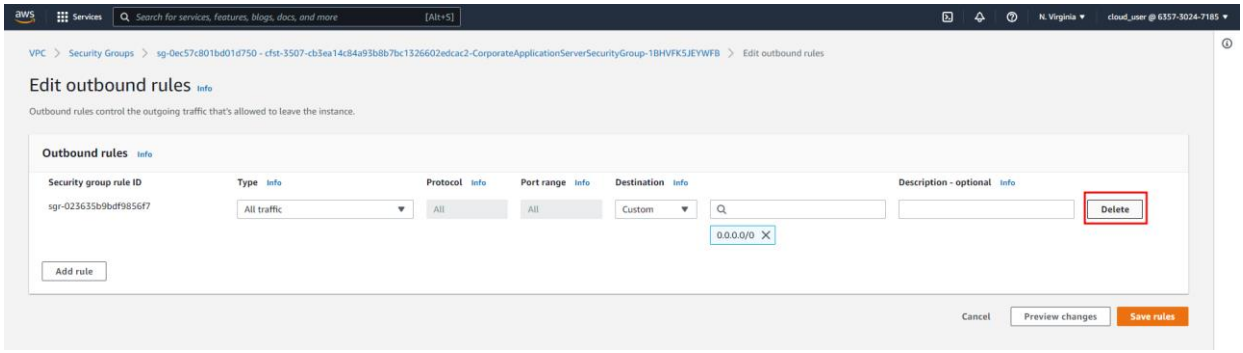


Рис. 3.21. Видалення правила CorporateApplicationServerSecurityGroup, що дозволяє вихідний трафік

Відновлення першочергової конфігурації.

Перейдемо на панель управління CloudTrail в розділ Event history

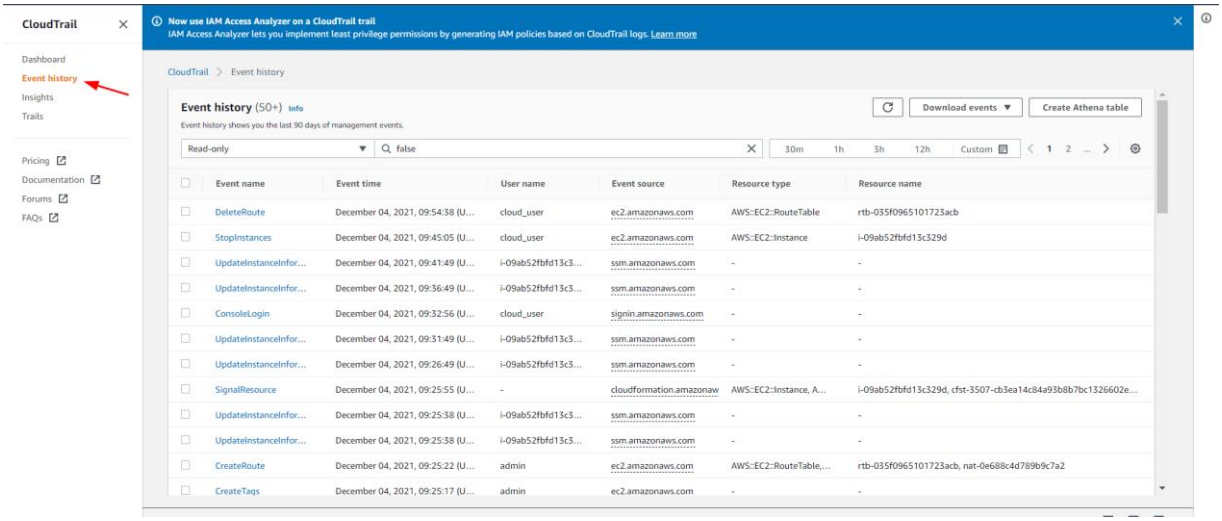


Рис. 3.22. Історія API викликів в CloudTrail

Відфільтруємо події виключно по операціям над CorporateServer. Першою в списку йде подія, що відбулась останньою, а саме зупинка інстансу користувачем cloud_user.

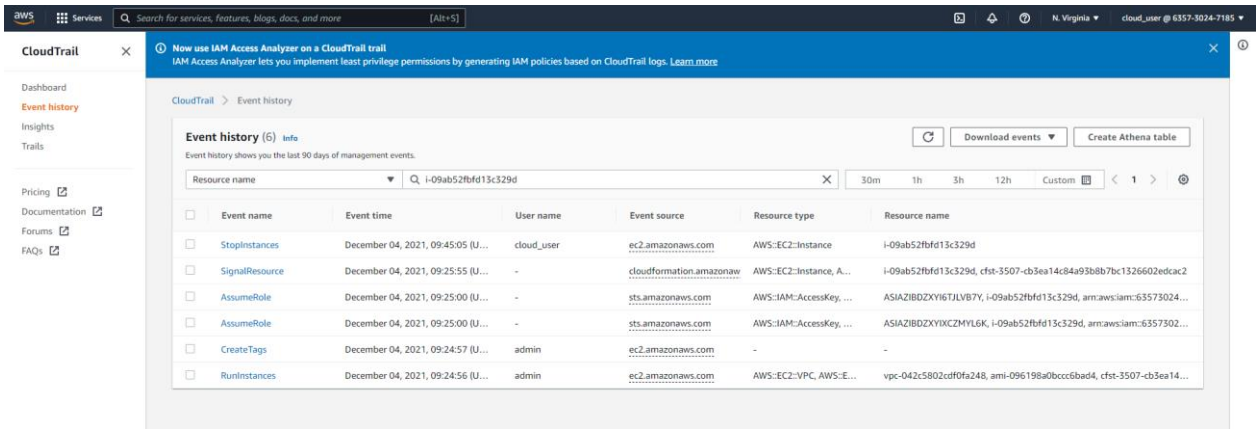


Рис. 3.23. Фільтр, що демонструє лише події з CorporateServer

Тепер відфільтруємо події по операціям над таблицею маршрутизації мережі PrivateAZ1.

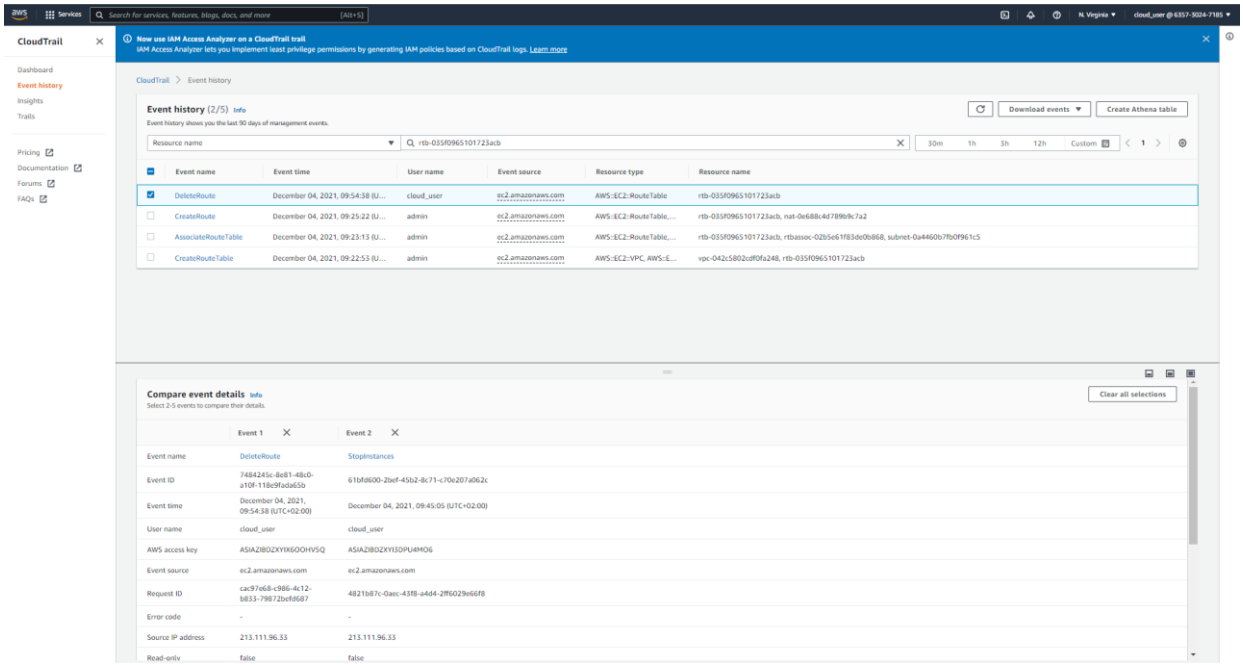


Рис. 3.24. . Фільтр, що демонструє лише події з таблицею маршрутизації мережі PrivateAZ1

Виберемо подію «DeleteRoute», перегнлянувши деталі бачимо співпадання по часу, об’єктах на яких вона вплинула й source IP 213.111.96.33 що відповідає IP

адресі мого ноутбуку з подією видалення маршруту що було нещодавно виконано. Перейдемо в AWS Config напряму з опису події щоб побачити деталі зміни конфігурації, що були викорнані натиснувши «View AWS Config resource timeline».

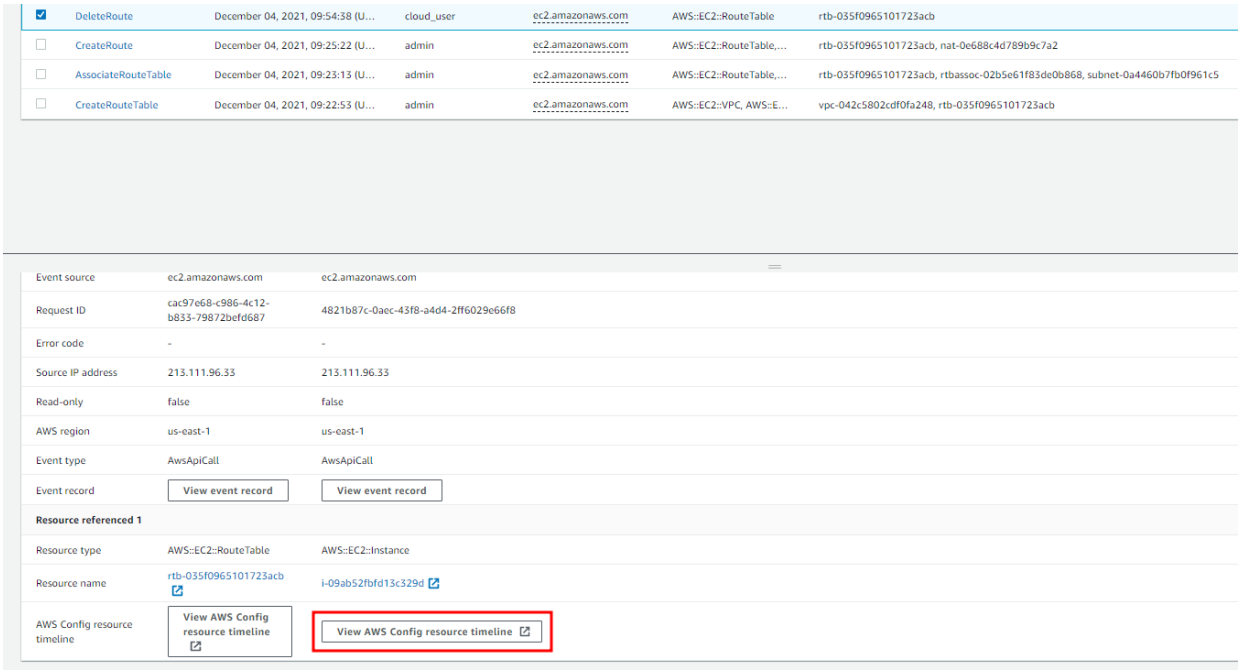


Рис. 3.25. Перехід в AWS Config напряму з опису події CloudTrail

AWS Config надає JSON конфігурацію до зміни та після її виконання.

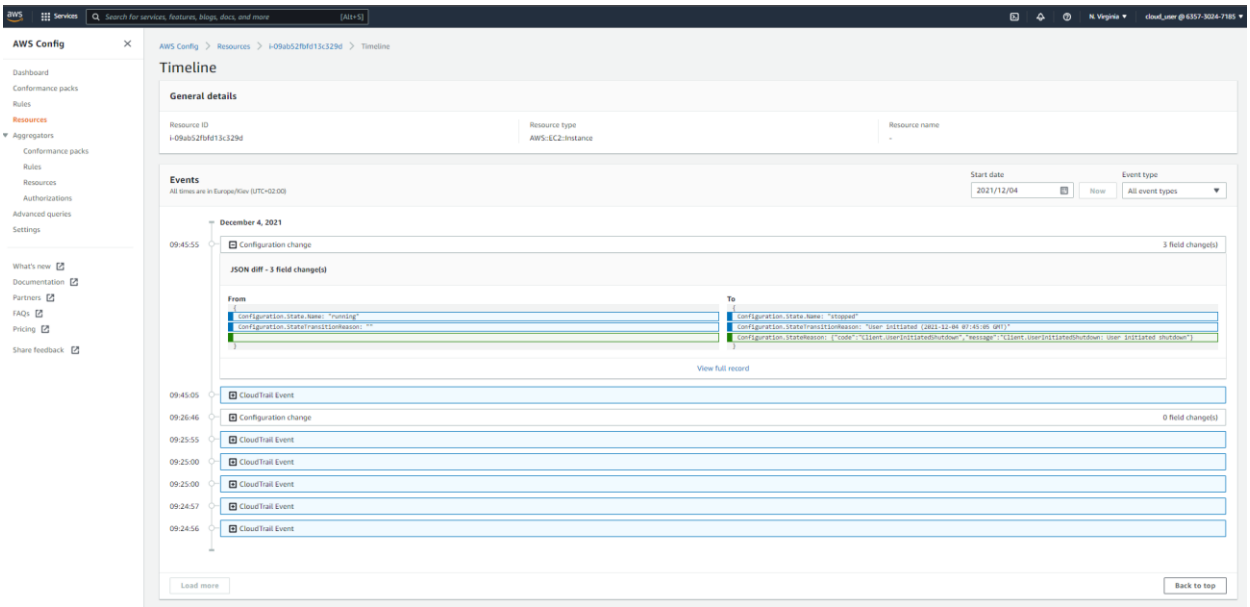


Рис. 3.26. Запис про зупинку CorporateServer в AWS Config

Аналогічно відслідкуємо зміни в CloudTrail та Config для Security Group CorporateApplicationServerSecurityGroup.

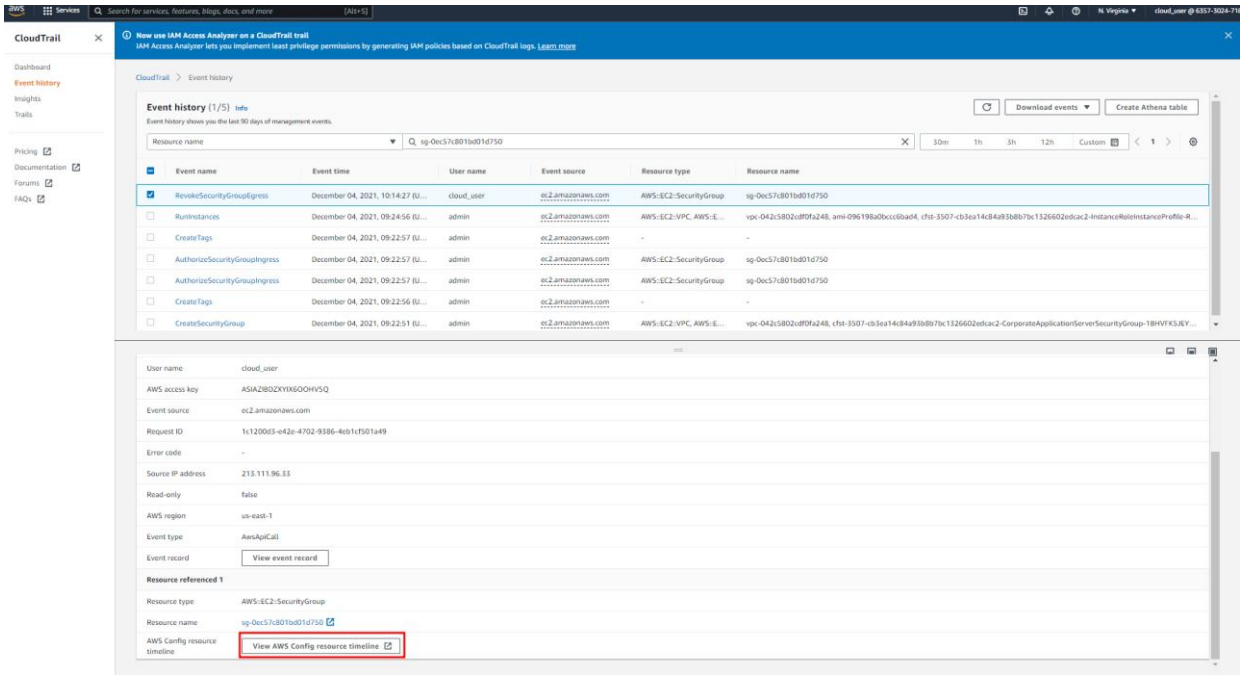


Рис. 3.27. Інформація про видалення правила Security Group в CloudTrail

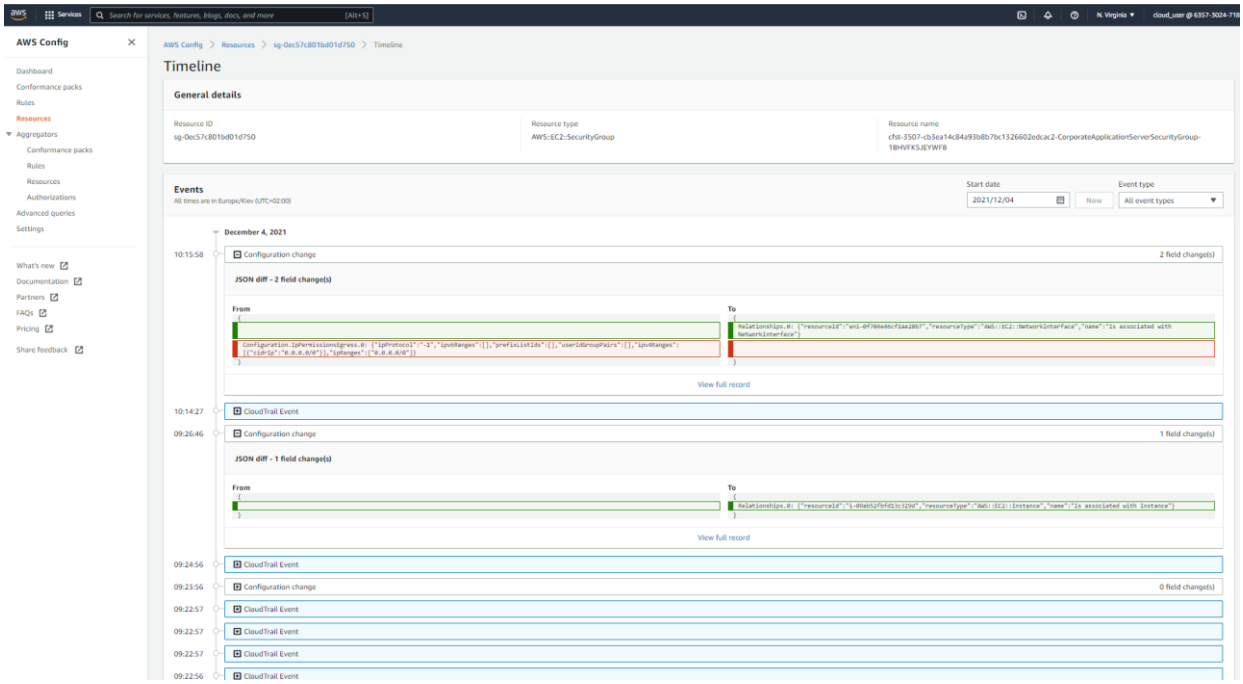


Рис. 3.28. Інформація про видалення правила Security Group в AWS Config

Завдяки точним відомостям про кожну операцію з ресурсами AWS в сервісі Config, ми без надмірних затрат часу на розслідування інциденту можемо, використовуючи ці записи, відновити першочергову конфігурацію, а саме:

1. Повернути правило, що дозволяє вихідний трафік для destination IP 0.0.0.0/0 в CorporateApplicationServerSecurityGroup.
2. Додати маршрут для destination IP 0.0.0.0/0 в таблицю маршрутизації мережі PrivateAZ1 що веде до NAT.
3. Запустити сервер CorporateServer.

3.5. Захист від DDoS та прикладних атак L7 на веб-додатки за допомогою AWS Shield та AWS WAF

З розділу 2 відомо, що AWS Shield Standard увімкнений за замовченням і захищає від більшості L3-L4 DDoS атак, але є можливість зробити захист від DDoS ефективнішим.

Наведемо декілька практичних рекомендацій щодо зменшення ризику та пом’якшення впливу можливих атак:

1. Використовувати CDN мережу CloudFront. Через Geo Restriction/Blocking – заборонити доступ користувачам з країн, що арпіорі не можуть прадставляти вашу цільову аудиторію.- Через Origin Access Identity – заборонити доступ до S3 bucket’ів за CloudFront напряму, в обхід CloudFront. Нагадаємо, що CloudFront це CDN мережа, з точками присутності (edge locations) по всьому світу, що здатна розвантажити ваші веб-сервіси за рахунок кешування їх контенту в цих точках присутності в найближчій доступності до кінцевого користувача, що також значно знижує затримки в передачі контенту.
2. Використовувати Route53 – DNS-сервіс від AWS, що витримує високі навантаження DNS-запитами.
3. Знати baseline трафіку на власні веб-сервіси щоб мати змогу відрізнити звичне навантаження від нестандартного.
4. Зменшити «поверхню атаки» («attack surface»).

Використовуючи Bastion host, або, як його ще називають Jump host, для управління внутрішніми серверами є рекомендованою практикою від AWS. Відсутність прямої можливості підключитись до «production» серверів з Інтернет для виконання адміністративних функцій значно знижує кількість нецільового навантаження трафіку, як з боку автоматизованих інструментів, так і з боку різного роду зацікавлених в пошуку вразливостей людей.

5. Використовувати ELB та AutoScaling group. В деяких випадках масштабування в ширину за допомогою ELB та AutoScaling group є ефективним методом подолати легкі DDoS атаки.

Для демонстрації створимо статичний веб-сайт в S3 bucket'і й розмістимо його за CloudFront й AWS WAF. Створимо S3 bucket з назвою «hitchhackers-s3-website», додамо файли сайту:

- index.html – див. «Додаток Б»;
- error.html – див. «Додаток В»;

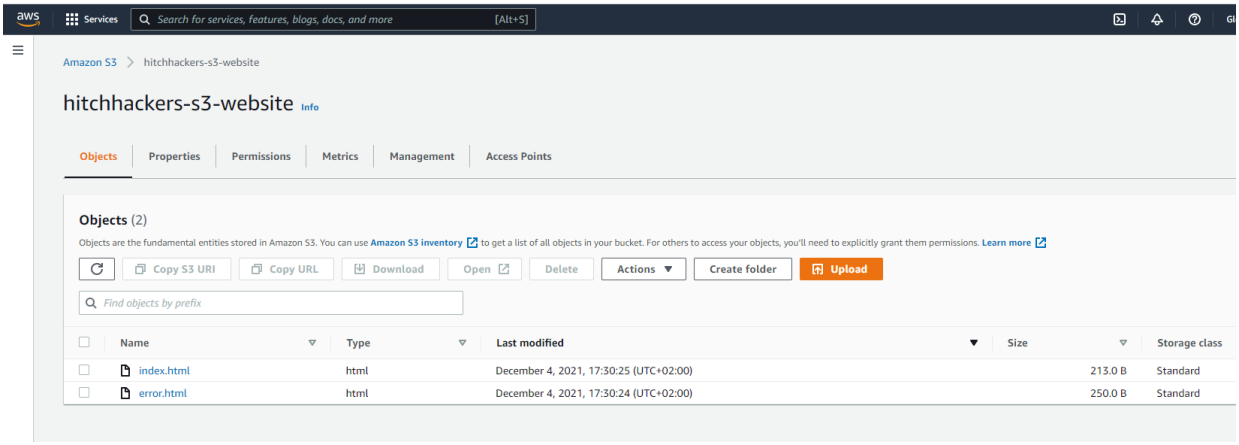


Рис. 3.29. Додавання файлів статичного сайту в S3 bucket

В налаштуваннях hitchhackers-s3-website перейшовши на вкладку «Properties» активуємо функцію «Static website hosting». Далі переходимо на вкладку «Permissions» та знайшовши Bucket policy, що являє собою Resource policy для S3, змінюємо код політики на вказаний в «Додатку Д», щоб дозволити всім переглядати наш сайт.

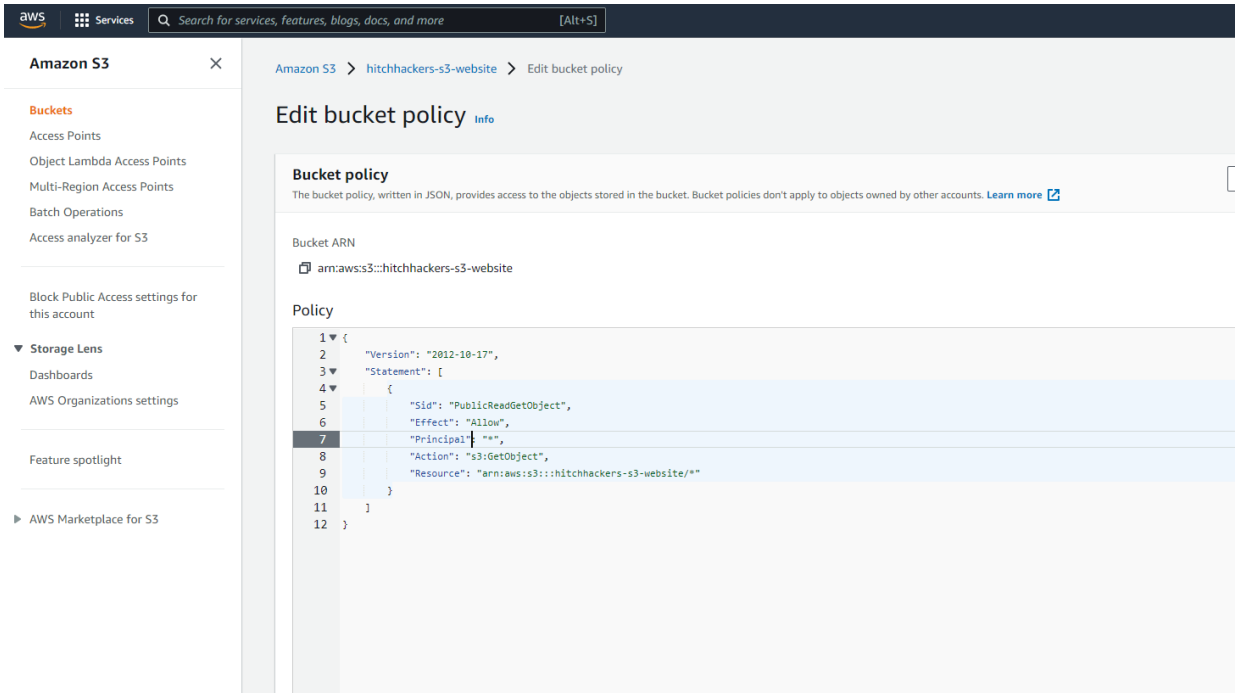


Рис. 3.30. Зміна S3 bucket policy для дозволу підключатись до сайту з Інтернет

Рис. 3.31. Сайт hitchhackers-s3-website доступний публічно

Створимо CloudFront distribution для забезпечення підключення до сайту використовуючи розподілену по всьому світу CDN мережу серверів з кешем для пришвидшеного надання клієнтам сайту.

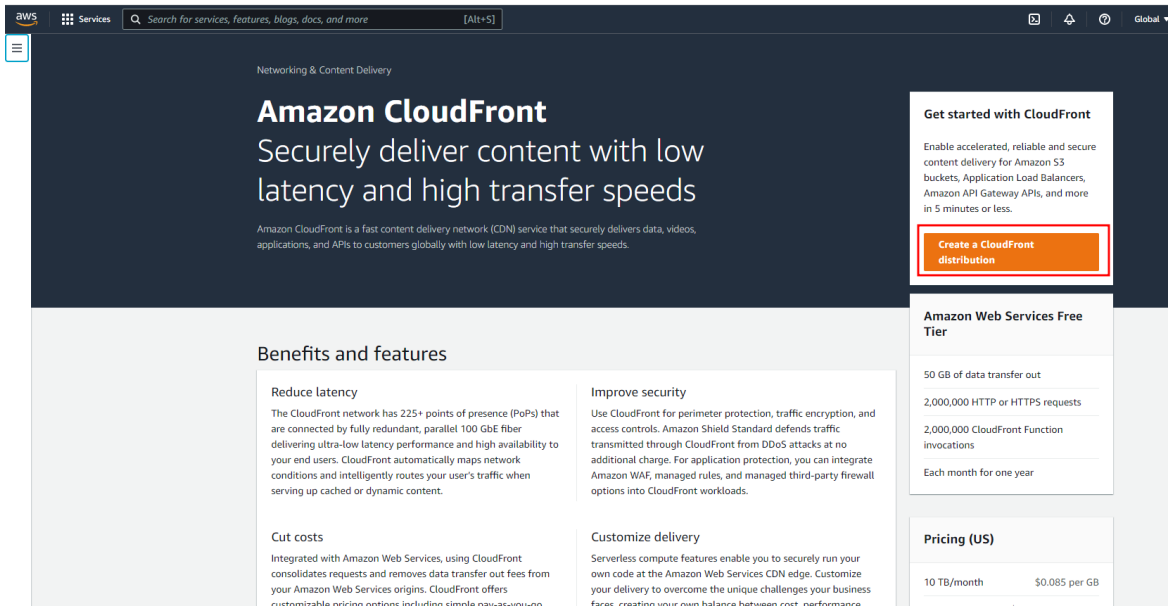


Рис. 3.32. Панель управління CloudFront

При створенні CloudFront distribution обов'язково вказуємо опцію «Yes use OAI (bucket can restrict access to only CloudFront)» щоб підключення до сайту могло в подальшому проходити лише через CloudFront.

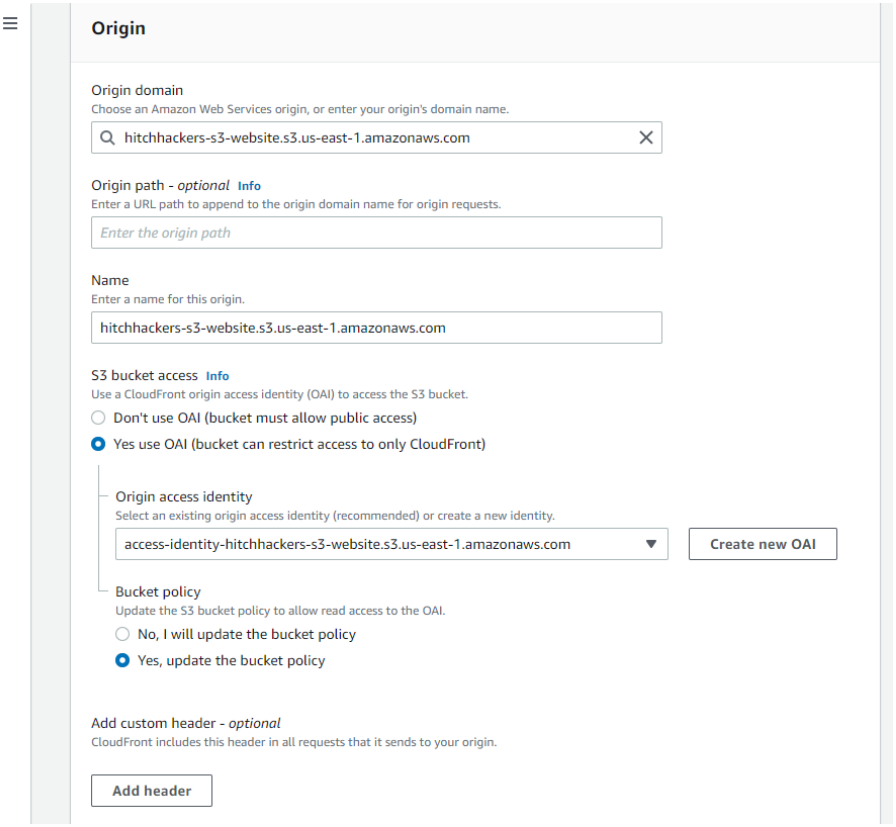


Рис. 3.33. Увімкнення обмеження на доступ до сайту тільки через CloudFront

Додамо захист AWS WAF до CloudFront distribution.

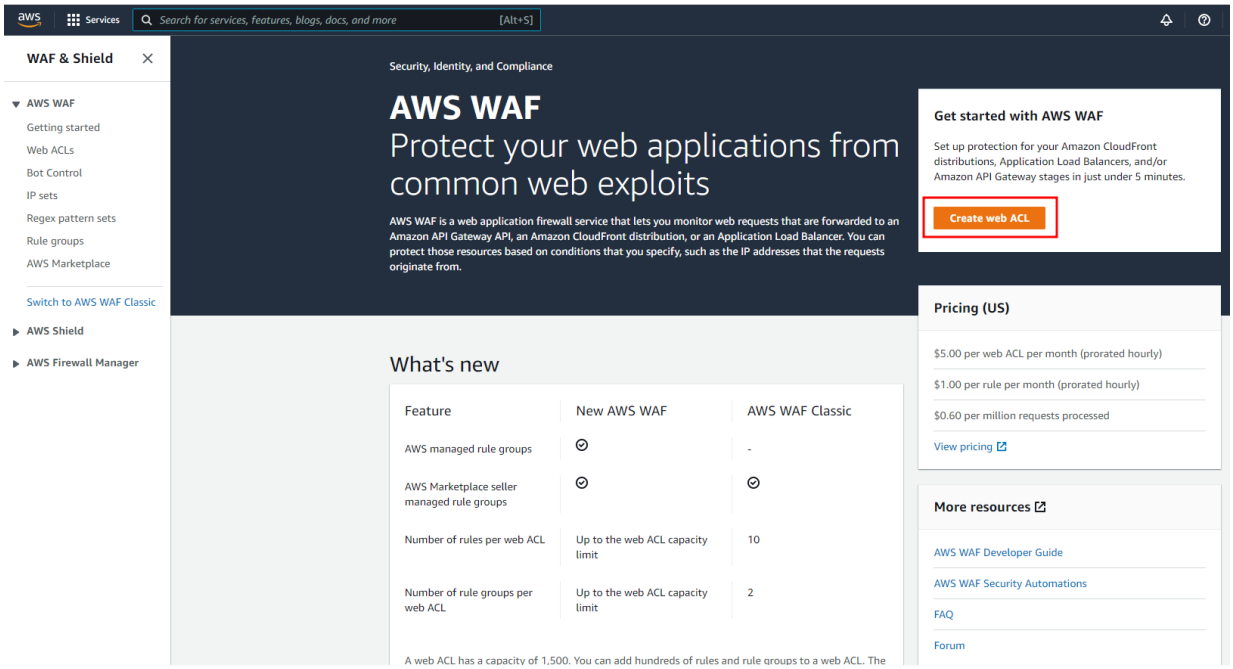


Рис. 3.34. Панель управління AWS WAF

При створенні WAF ACL, тобто списку контролю доступу прикладного рівня L7, вказуємо щойно створену CloudFront distribution.

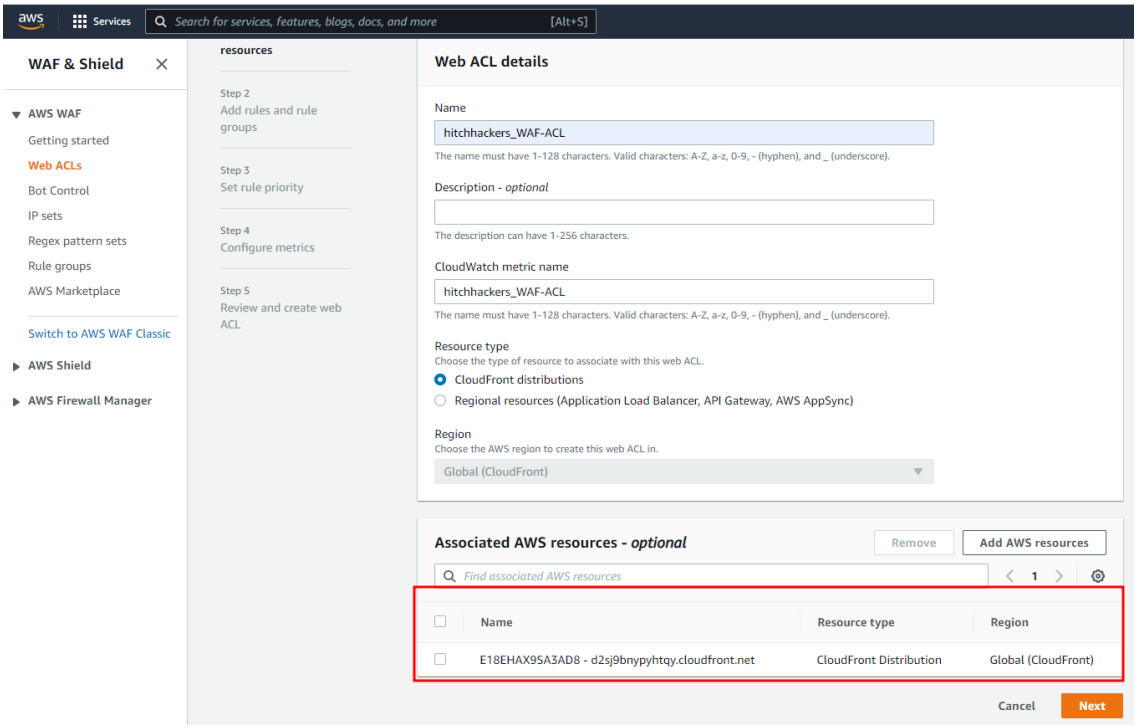


Рис. 3.35. Розміщення AWS WAF перед CloudFront

На наступному кроці вибираємо опцію «Add managed rule groups» щоб додати набір заздалегідь сконфігурованих правил від AWS або довірених третьосторонніх постачальників, таких як F5, Fortinet, PaloAlto, Imperva, та інші. Активуємо декілька основних правил захисту на основі репутаційних баз даних IP адрес AWS та набір правил, що відповідає за захист від загроз OWASP Top 10. [13]

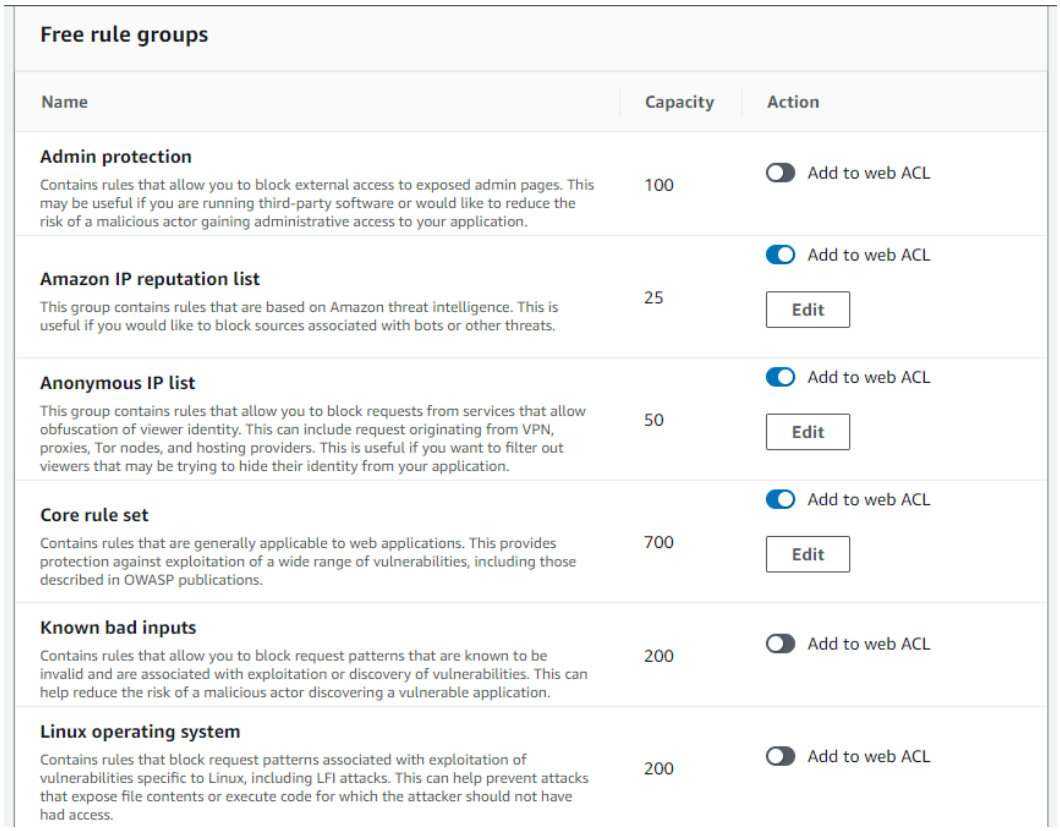


Рис. 3.36. Додавання правил захисту, що оновлюються та підтримуються компанією AWS

Як видно з наступного зображення з допустимого ліміту одиниць WCU ми використали 775 з 1500. Про WCU було детально розказано в розділі 2, щоб нагадати зазначимо, що кожне правило захисту чи набір правил витрачають ліміт WCU і доводиться враховувати це обмеження при виборі правил, які обираються для застосування в WAF Web ACL.

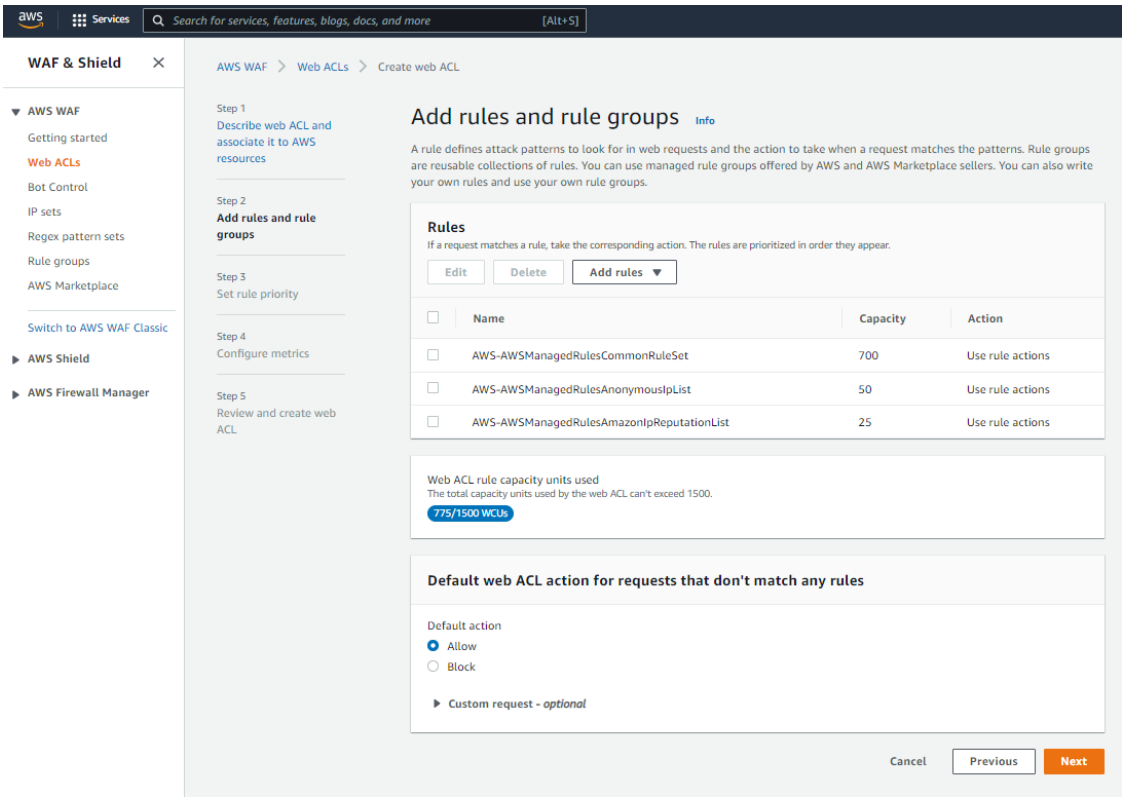


Рис. 3.37. Перегляд доданих правил захисту від веб-загроз

Пройшовши всі 5 кроків та підтвердивши задану конфігуравцією отримуємо увімкнений сервіс WAF перед сайтом hitchhackers-s3-website з адресою: http://hitchhackers-s3-website.s3-website-us-east-1.amazonaws.com/. Залишається перевірити сайт сканером вразливостей, як Nikto, проте одразу зазначимо, що сайт є статичним, а отже більшість можливих вразливостей не можуть бути присутніми в силу функціональних можливостей сайту. Результат додаємо в «Додаток Ж». Як і очікувалось виведено лише інформаційні повідомлення, вразливостей знайдено не було.

3.6. Виконання vulnerability assessment за допомогою AWS Inspector

Розгорнемо EC2 інстанс з заздалегідь встановленим агентом AWS Inspector, що дозволить проводити сканування не лише рівня мережевих та транспортних підключень за моделлю OSI, а і виконувати перевірки рівня ОС. В даному випадку буде використовуватись віртуальна машина Amazon Linux 2.

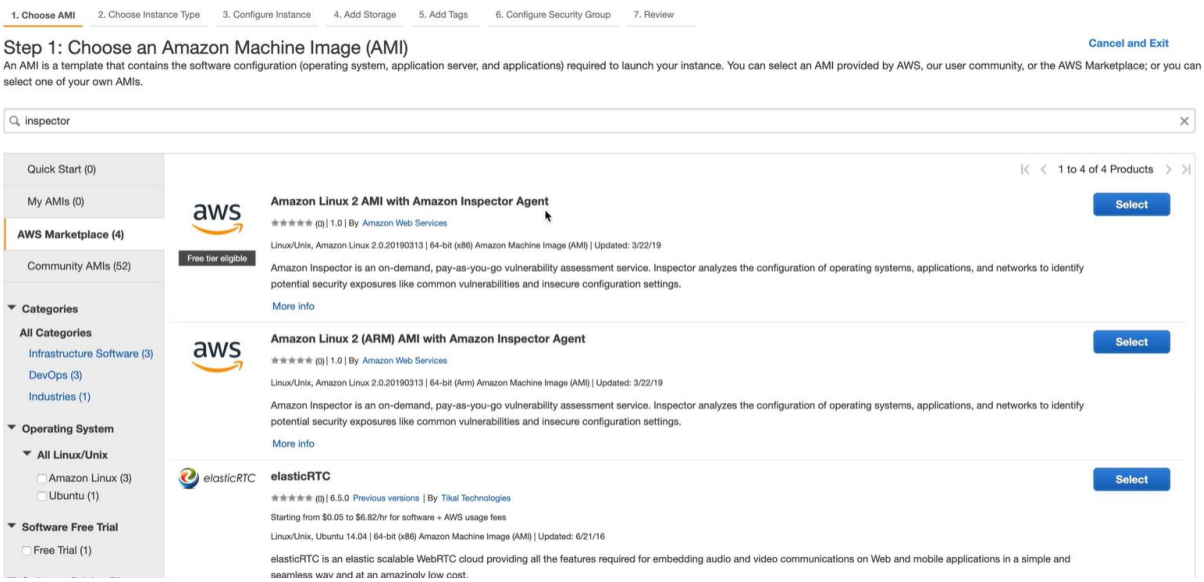


Рис. 3.38. Розгортання Amazon Linux 2 з агентом AWS Inspector

Коли розгортяння завершене перейдемо до панелі управління AWS Inspector та налаштуємо vulnerability assessment. Для регулярних перевірок варто використовувати опцію «Run weekly», чи налаштувати власний інтервал перевірок. Для демонстрації буде використано одноразову перевірку.

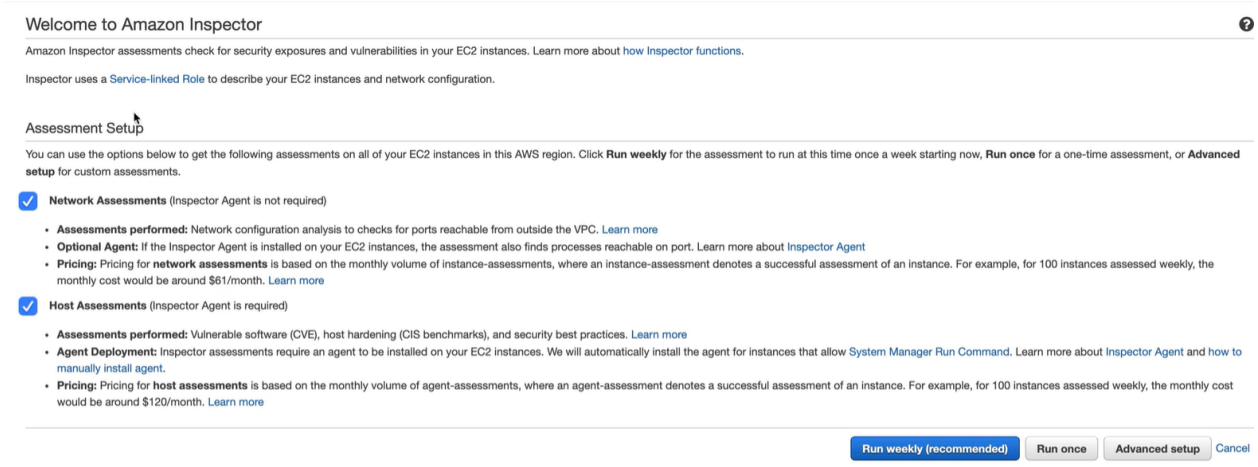


Рис. 3.39. Вибір типу перевірки на вразливості

В даному випробуванні буде використовуватись як перевірка на транспортному та мережевому рівні, так і на рівні хоста. Перевірка на вразливості рівня хоста передбачає пошук присутності CVE в ПЗ EC2 інстанса та перевірку на відповідність загальним практикам безпеки й нормам, зазначеним в Center for

Internet Security Benchmark (CIS Benchmark), розробленому спеціально для інфраструктури AWS.

Центр інтернет-безпеки (CIS) є некомерційною організацією, яка розробляє власні контрольні показники та рекомендації, що дозволяють організаціям удосконалювати свої програми забезпечення безпеки та відповідності вимогам. Ця ініціатива спрямована на створення базових рівнів конфігурації безпеки систем, які зазвичай зустрічаються у всіх організаціях. Для завантаження доступні кілька десятків гайдлайнів з безпечного налаштування різних систем: Windows, Linux, OSX, MySQL, Cisco та багатьох інших, що доступні безкоштовно на сайті learn.cisecurity.org/benchmarks.

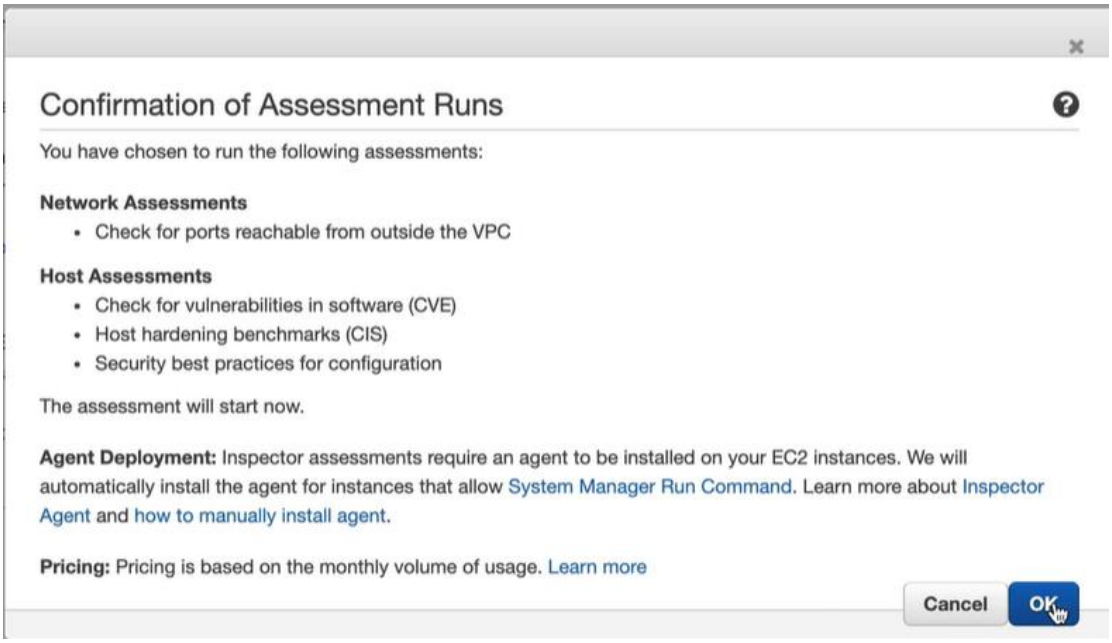


Рис. 3.40. Підтвердження на запуск сканування AWS Inspector

Після завершення перевірки перейдемо до вкладки «Findings». Було знайдено 214 проблем безпеки, які потребують уваги. Незважаючи на те, що було використано образ для розгортання віртуальної машини, наданий безпосередньо команією AWS, це не гарантує повний захист від вразливостей виробників операційних систем і програмного забезпечення на на них встановлених. Про це також повідомляє Shared responsibility model, яку AWS розробили для інформування користувачів хмарної платформи. ПЗ потребує регулярного оновлення та патчів й при використанні EC2 інстансів самі користувачі

відповідальні за ці процедури. Щоб повністю позбавити себе даної рутинної роботи слід звернути увагу на serverless обчислення та інші повністю управляємі AWS сервіси.

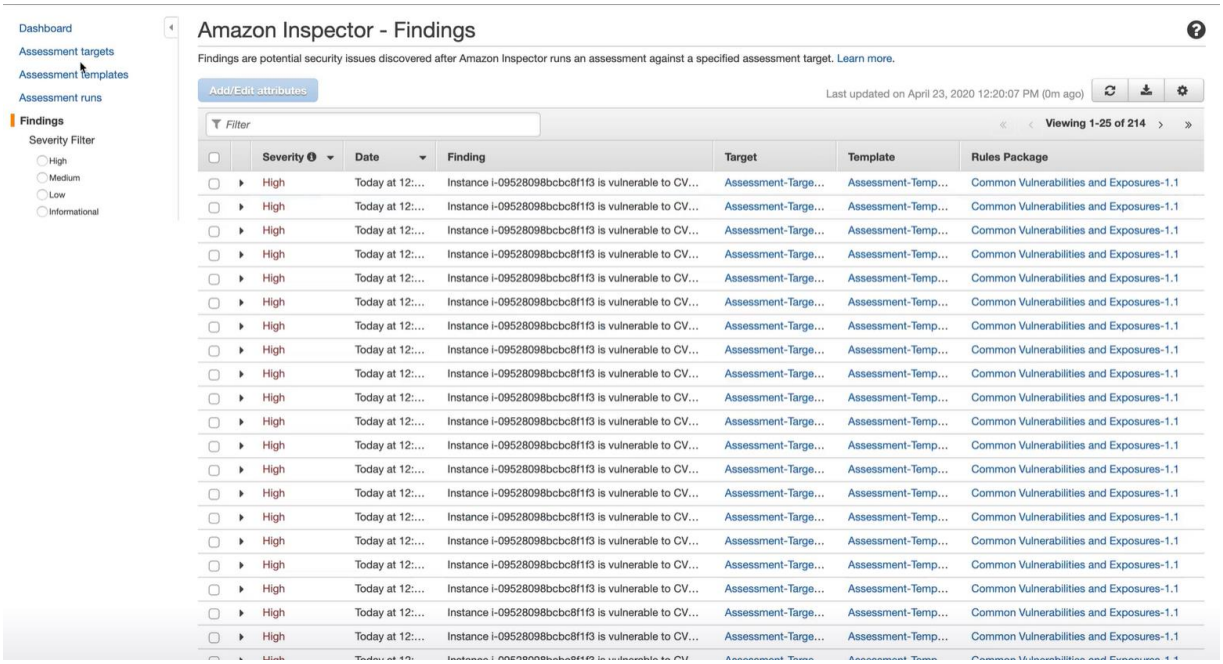


Рис. 3.41. Інформаційна панель з результатами сканування AWS Inspector

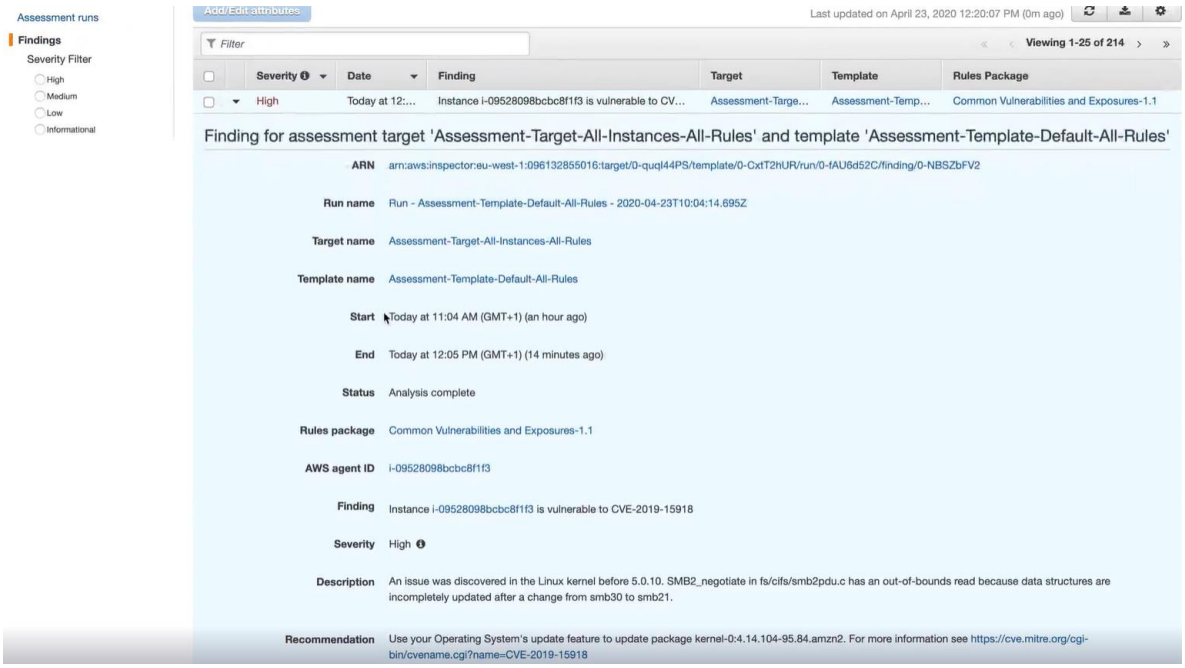


Рис. 3.42. Детальні дані інциденту й рекомендації для усунення

В даному описі вразливості описана вразливість ядра та рекомендація оновити систему й посилання на офіційну сторінку MITRE з повними відомостями про

CVE. Цей функціонал значно полегшує роботу адміністраторів безпеки хмарної інфраструктури завдяки наданню автоматизаціїї процесу пошуку вразливостей рівня хоста й інструментів що здатні це виконувати. Також є можливість активувати перевірку на відповідність стандарту PCI DSS, щоправда це вимагаэ додаткових грошрових витрат.

3.7. Використання сервісу Secrets Manager для автентифікації в БД RDS за допомогою сервісу Lambda

В даному прикладі спочатку буде виконано підключення до БД MySQL RDS використовуючи AWS Lambda function за допомогою імені користувача та пароля, що не є безпечним підходом до керування аутентифікацією, адже кожен хто має доступ до коду Lambda function зможе побачити секретні дані підключення. Тому буде продемонстровано перехід на більш захищений спосіб роботи з даними аутентифікації й передано керування обліковими даними службі AWS Secrets Manager. Використовуючи найкращі практики зберігання паролів, в тому числі від організації OWASP, буде використано Secrets Manager API для підключення до БД замість жорсткого кодування облікових даних у Lambda function. Це слугуватиме наглядним прикладом того, як зберігати секрети у AWS Secrets Manager і отримати доступ до них за допомогою Lambda function.

Створимо сервер БД з назвою «testRDS» перейшовши в панель управління RDS:

- В якості типу БД виберемо MySQL останньої версії.
- В якості логіну виберемо «username»
- В якості паролою виберемо «password»

Процес створення займає короткий час, деякі другорядні дані заповнені заздалегідь автоматично.

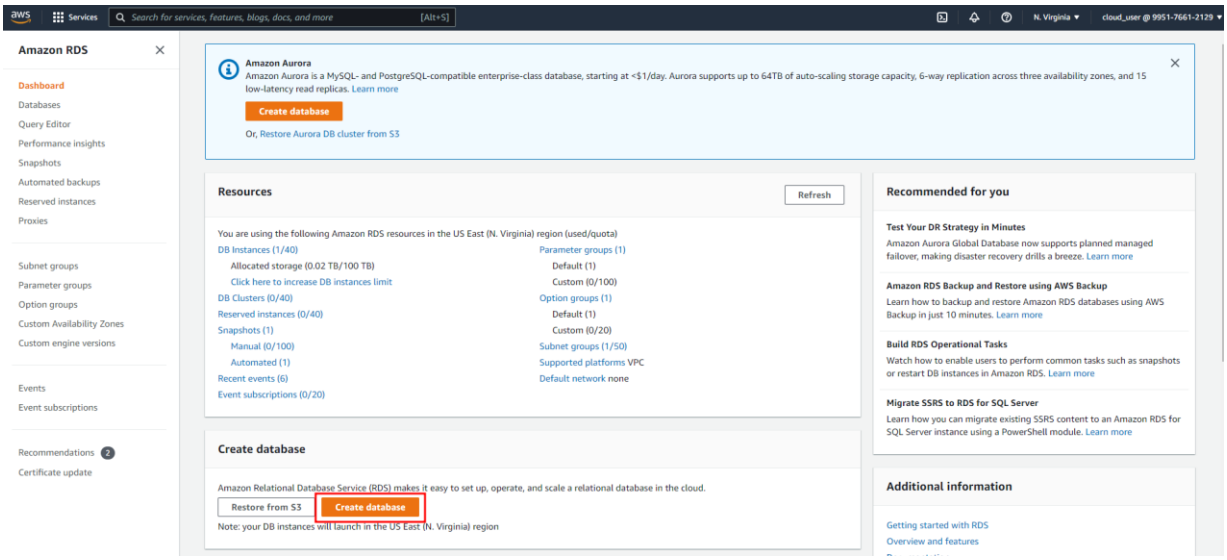


Рис. 3.43. Панель управління БД

Адресу інстансу БД скопіюємо для подальшого використання в Lambda function.

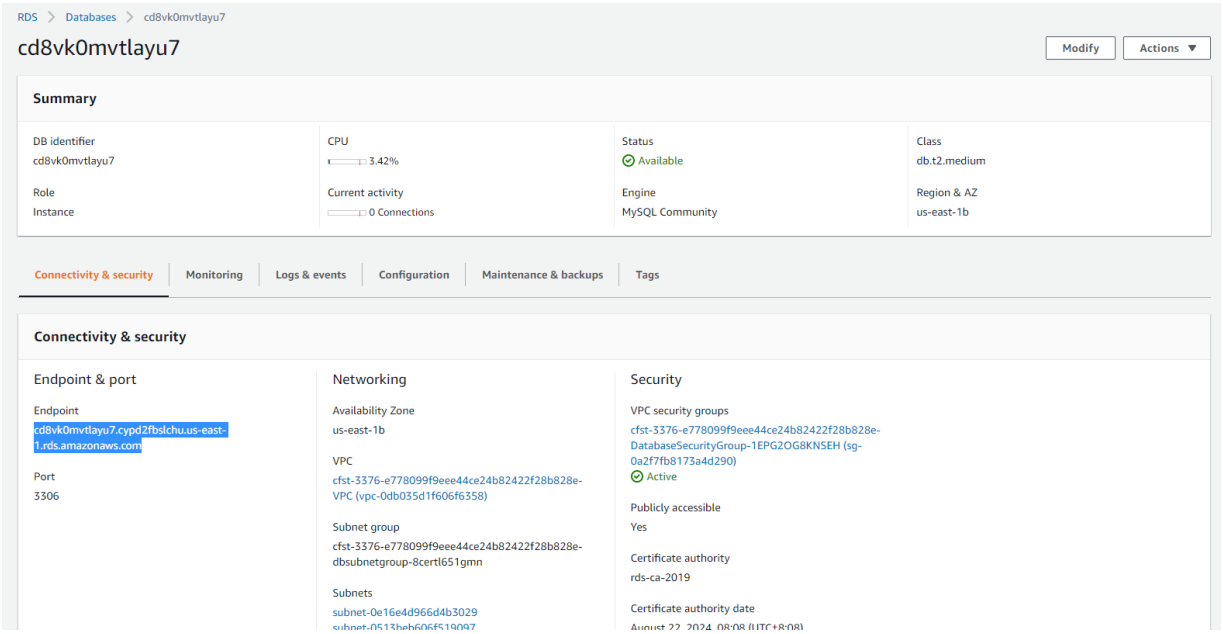


Рис. 3.44. Детальні відомості про інстанс RDS

Перейдемо на панель управління Lambda і створимо функцію, яка буде виводити при виконанні підключатись до MySQL БД та виводити таблиці БД. Функція була написана мовою JavaScript (NodeJS), код наведено у «Додатку И». Щоб функція вступила в дію натиснимо «Deploy» та «Test».

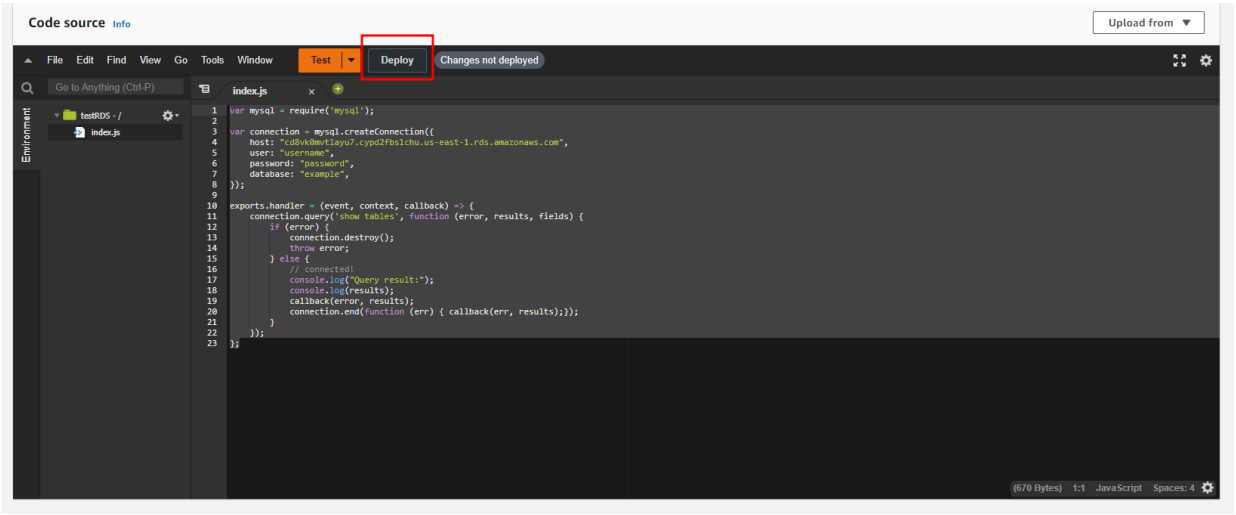


Рис. 3.45. Створення Lambda function

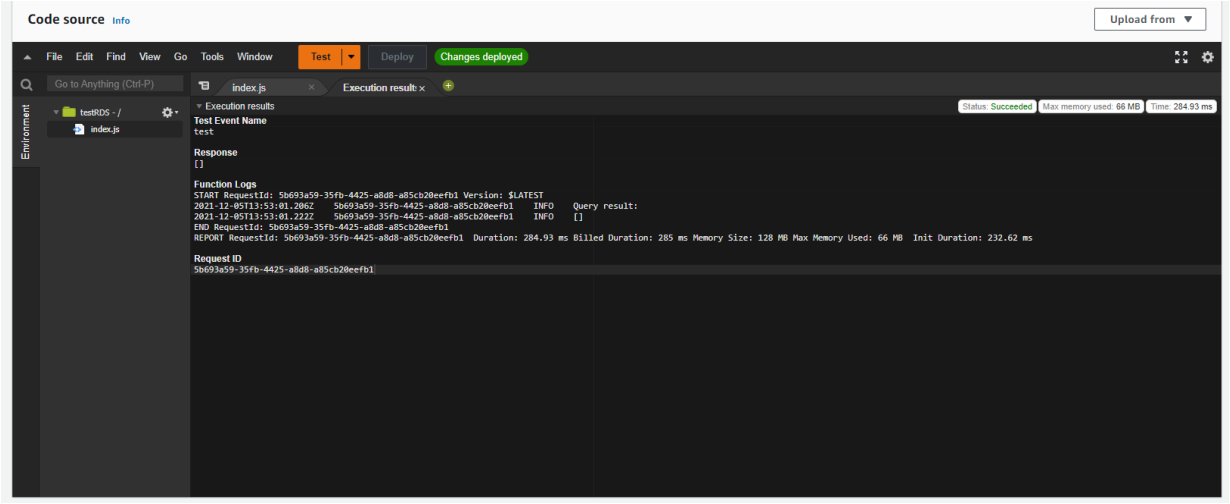


Рис. 3.46. Результат виконня Lambda function

Так як БД не має таблиць, функція повернула порожній результат, що видно за вивведеним значенням «Response [ ]». Підключення до БД відбулось успішно. Для більшої наглядності додамо порожню таблицю з назвою «pet» замінивши в запиті строку: «connection.query('show tables', function (error, results, fields) { » на строку «connection.query('CREATE TABLE pet (name VARCHAR(20), species VARCHAR(20))',function (error, results, fields) { ». Після цього повернемо команду отримання таблиць БД назад. Перейдемо до панелі управління AWS Secrets Manager, щоб налаштувати безпечне зберігання даних аутентифікації та створимо новий запис. Слід взяти до уваги, що як тільки буде створено даний

запис, ми більше не зможемо підключитись до БД методом жорсткого задання в коді логіну та паролю.

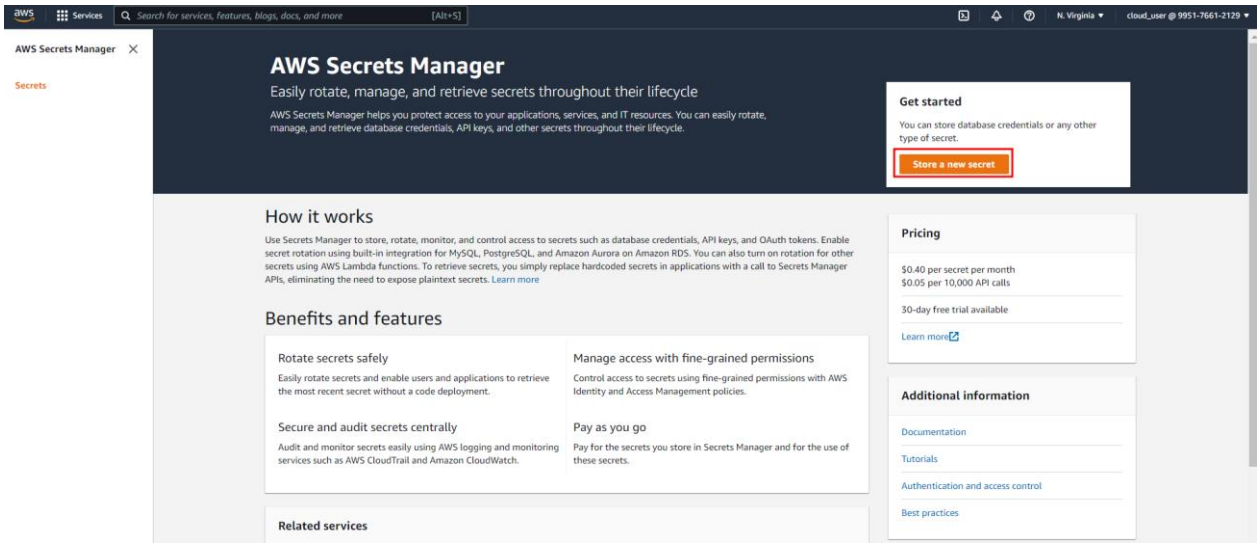


Рис. 3.47. Створення нового запису в AWS Secrets Manager

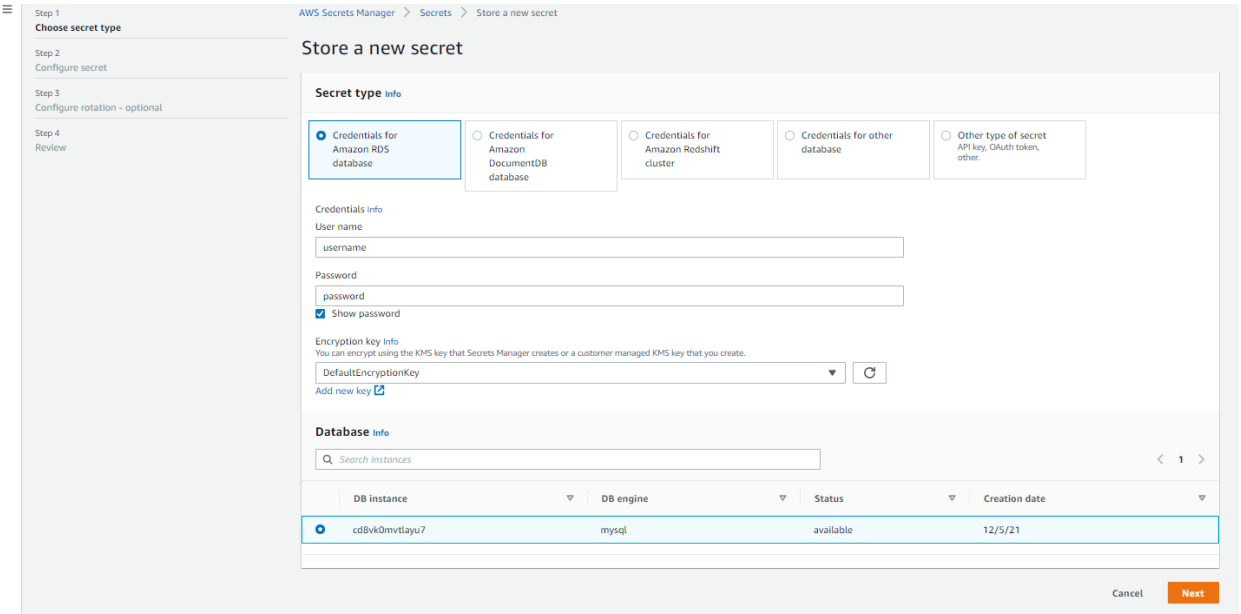


Рис. 3.48. Створимо запис з таємними даними в сховищі Secret Manager

Наступним кроком буде налаштування ротації облікових даних, в нашому випадку задамо власне значення в 1 день, хоча, без сумнівів, це надмірно малий термін для реального середовища, а не тестової демонстрації можливостей AWS.

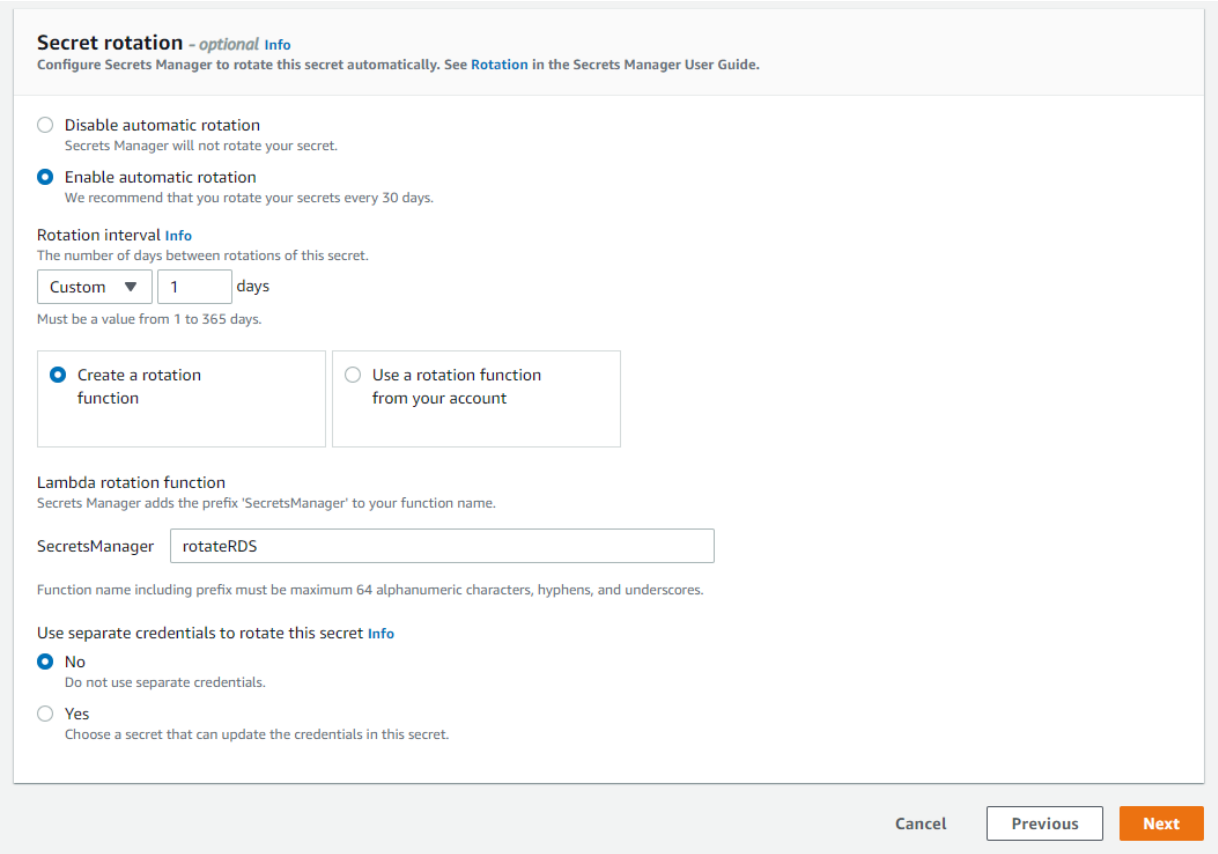


Рис. 3.49. Налаштування ротації пароля

Для перевірки спробуємо знову виконати функцію. Результат буде очікуваним – в доступі буде відмовлено, тому як за пароль з моменту створення запису в Secret Manager відповідає сам сервіс Secret Manager.

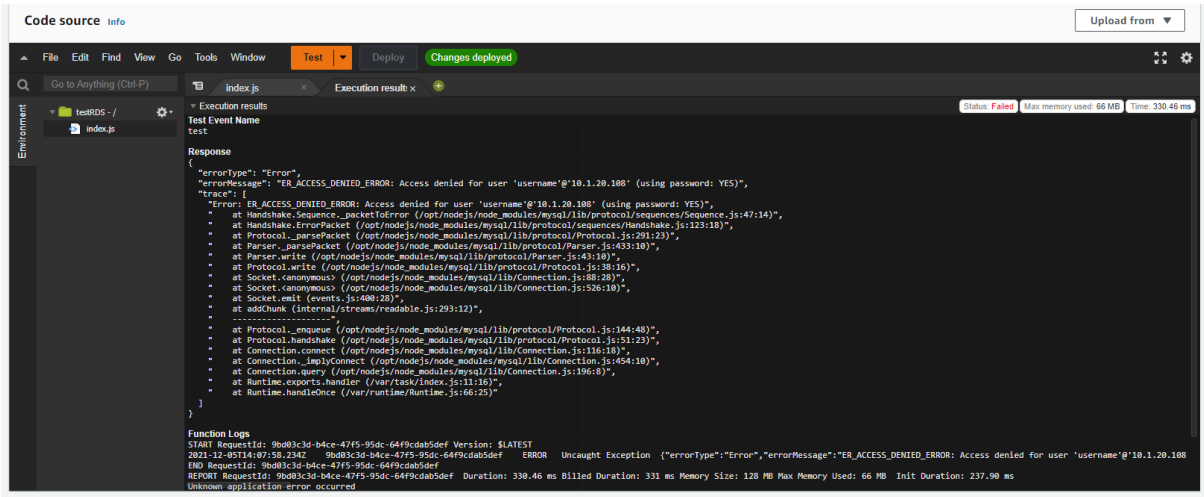


Рис. 3.50. Спроба виконати функцію використовуючи жорстко задані облікові дані в коді.

Створимо VPC Endpoint для можливості підключення до БД в нових умовах.

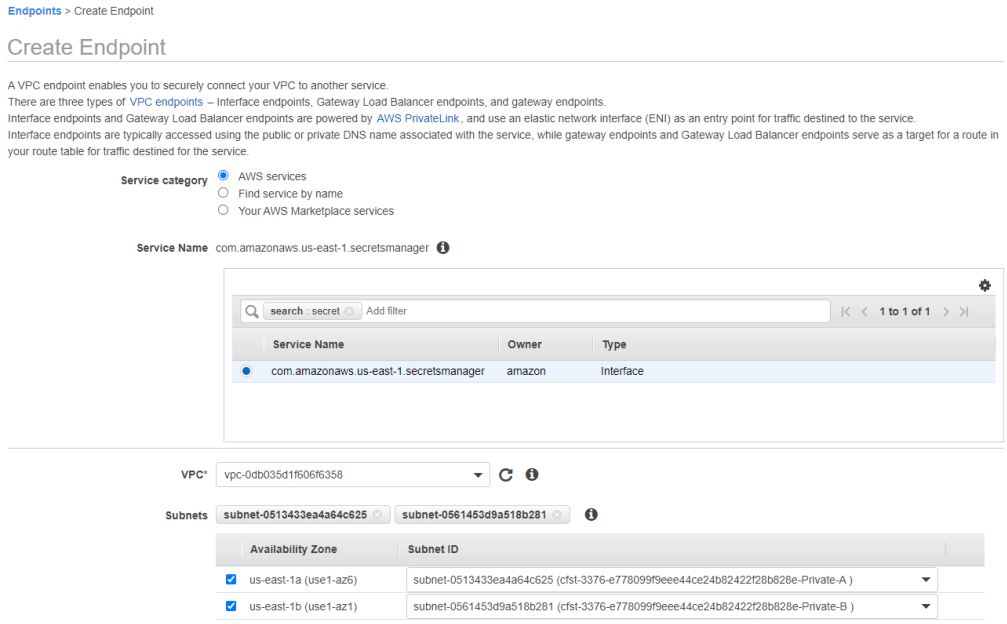


Рис. 3.51. Створення VPC Endpoint для Secrets Manager

Додамо правило, що дозволить Secrets Manager підключатись до сервісу RDS по протоколу HTTPS.

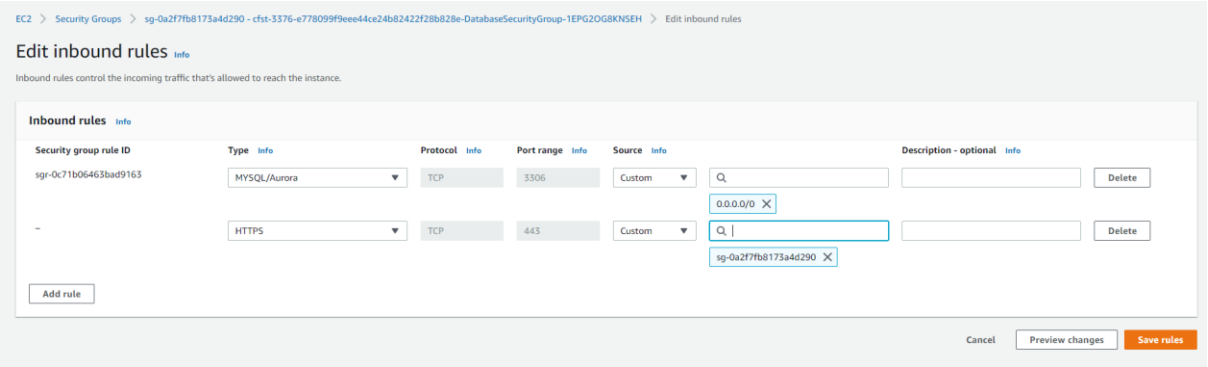


Рис. 3.52. Зміна правил доступу в Security Group на RDS

Створимо IAM Role, що надасть доступ Secrets Manager до БД.

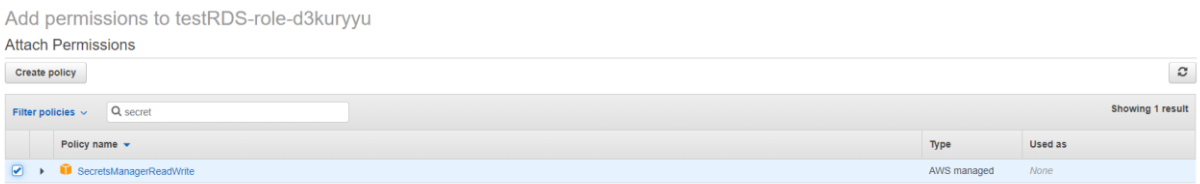


Рис. 3.53. Додання дозволу Secrets Manager на отримання доступу до БД

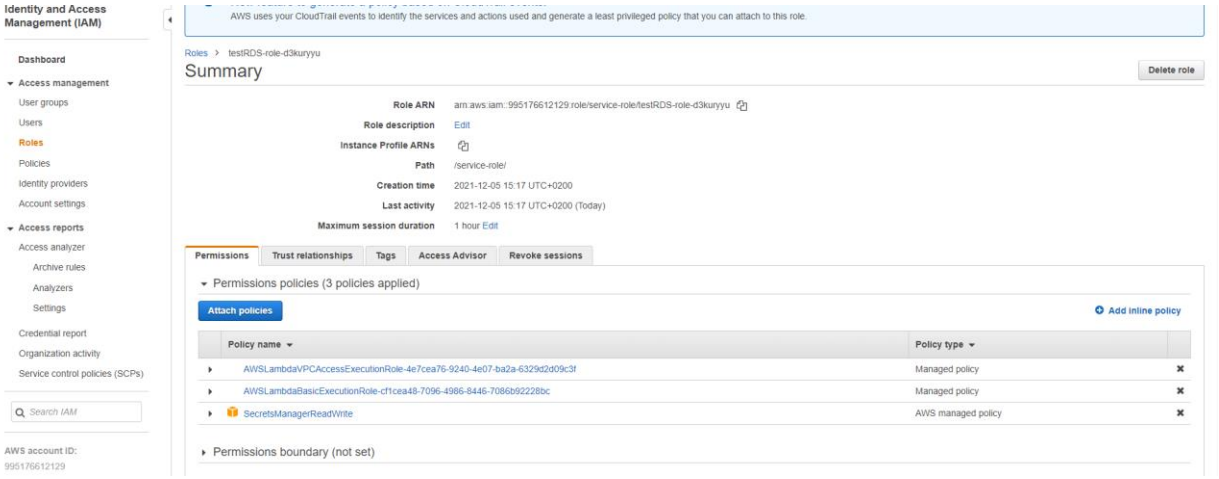


Рис. 3.54. Результат створення IAM ролі

Тепер, коли підготовку до використання AWS Secrets Manager в аутентифікації в БД було виконано, змінимо сам код підключення до БД. Див. «Додаток К». Результатом є виведні таблиці БД, а саме одна таблиця «pet».

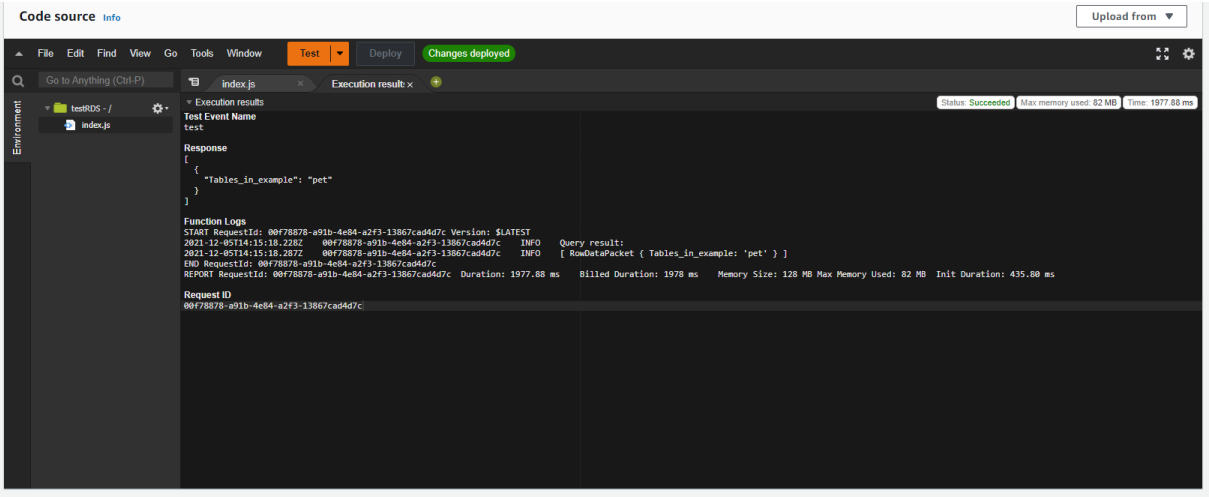


Рис. 3.55. Результат роботи Lambda function з аутентифікацією через Secrets Manager

Таким чином було досягнуто реалізації безпечної аутентифікації в БД RDS без задання параметрів аутентифікації безпосередньо в коді програми, чию роль виконувала Lambda function.

Висновки до розділу 3.

Було продемонстровано практичне застосування технологій захисту хмарної інфраструктури, реалізовано захист на рівні розмежування прав

доступу, ні рівні фільтрації мережевого трафіку на рівнях L3-L7 за моделлю OSI, в тому числі для захисту від DDoS атак, виконано повернення до початкової конфігурації після інциденту, надано рекомендації щодо створення даної захищеної інфраструктури з використанням засобів Amazon AWS.

ВИСНОВКИ

Головним результатом проведеної роботи є створення конкретних прикладів реалізації функціоналу захисту хмарної інфраструктури веб-додатків, опис їх налаштування, що може бути використано як приклад для розгорнення захищеної хмарної інфраструктури AWS, а також створення рекомендацій щодо використання технологій захисту присутніх в AWS.

В ході реалізації даного проекту був проведений аналіз технічної документації сервісів AWS, рекомендацій OWASP та технічної літератури, як наприклад, книга «Таненбаум Е. Сучасні операційні системи».

В ході виконання дипломної роботи було проаналізовано будову хмарної інфраструктури, роботу AWS, продемонстровано порядок налаштування сервісів захисту AWS на рівнях L2-L7 по моделі OSI . Вдалося розібратися в тому, які є «вузькі місця» «on-premise» архітектури, які саме завдання сьогодення вимагають використання «cloud-native» підходів. Використано сервіси AWS CloudTrail, AWS Config, AWS WAF, AWS Shield, AWS GuardDuty, AWS Inspector, AWS Macie, AWS Secrets Manager, AWS IAM для досягнення поставлених задач по створенню захищеної хмарної інфраструктури, що працюють в хмарі AWS, дозволило створити конкретні приклади реалізації захисту активів в хмарі, які можна буде використовувати при розробці всласного хмарного середовища.

Огляд загроз інформаційній безпеці підтвердив необхідність впровадження в роботу засобів захисту, щоб захистити основні властивості інформації в ході взаємодії з сервісами в хмарі.

Коло запропонованих функціональних можливостей AWS охоплює широкий спектр завдань. Так як дана робота демонстраційна, в ній вирішується лише частина завдань. Вони вирішуються в рамках даної роботи, з максимальною простотою, зручністю та швидкістю. Таким чином, мета і завдання дипломної роботи були виконані.

ПЕРЕЛІК ПОСИЛАНЬ

1. Cloud Data Center Trends To Watch For In 2015 [Електронний ресурс] / Sarah Tanksalvala – Режим доступу : World Wide Web. – URL: https://www.forbes.com/sites/huawei/2015/04/16/cloud-data-center-trends-to-watch-for-in-2015/
2. What Companies Growing More than 50 Percent Faster Are Investing In [Електронний ресурс] / Laura Pevehouse – Режим доступу : World Wide Web. – URL: https://www.delltechnologies.com/en-us/blog/what-companies-growing-more-than-50-percent-faster-are-investing-in/
3. К.О. Вольська, та А.П. Дикий, " Бухгалтерський облік у “хмарі”: порядок переходу та адаптації інформаційної системи підприємства ", Проблеми теорії та методології бухгалтерського обліку, контролю і аналізу, ЖДТУ, No 2(37), с. 24-29, 2017.DOI: 10.26642/pbo-2017-2(37)-24-29.
4. Хмарні обчислення, Integrity Systems.[Електронний ресурс]. Доступно: http://integritysys.com.ua/solutions/pricatecloud-solution. Дата звернення: Січ. 27, 2020.
5. А.Е. Кононюк, Фундаментальная теория облачных технологий: Введение в фундаментальную теорию облачных технологий (Киев, Освіта України, 2018) кн 2, 528.
6. А.Е. Кононюк, Фундаментальная теория облачных технологий: Общенаучные подходы формирования систем облачных технологий (Киев, Освіта України, 2018) кн 1, 621.
7. Таненбаум Е. Сучасні операційні системи / Таненбаум Е - СПб.: Изд. Пітер, 2002. – Р. 74-75.
8. Краткая история облачных технологий [Електронний ресурс] / Анна Полякова – Режим доступу : World Wide Web. – URL: https://rb.ru/story/cloud-computing-history/

9. Six Advantages of Cloud Computing [Електронний ресурс] – Режим доступу : World Wide Web. – URL: https://docs.aws.amazon.com/whitepapers/latest/aws-overview/six-advantages-of-cloud-computing.html
10. Gartner Forecasts Worldwide Public Cloud End-User Spending to Grow 18% in 2021 [Електронний ресурс] – Режим доступу : World Wide Web. – URL: https://www.gartner.com/en/newsroom/press-releases/2020-11-17-gartner-forecasts-worldwide-public-cloud-end-user-spending-to-grow-18-percent-in-2021
11. Netflix Case Study [Електронний ресурс] – Режим доступу : World Wide Web. – URL: https://aws.amazon.com/solutions/case-studies/netflix-case-study/
12. Using multi-factor authentication (MFA) in AWS [Електронний ресурс] – Режим доступу : World Wide Web. – URL: https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_mfa.html
13. OWASP is pleased to announce the release of the OWASP Top 10 – 2017 [Електронний ресурс] – 2017 – Режим доступу : World Wide Web. – URL: https://owasp.blogspot.com/2017/11/owasp-is-pleased-to-announce-release-of.html

ДОДАТКИ

Додаток А


```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Action": [
            "ec2:Describe*",
            "ec2:StartInstances",
            "ec2:StopInstances"
         ],
         "Resource": "*",
         "Effect": "Allow"
      },
      {
         "Action": "elasticloadbalancing:Describe*",
         "Resource": "*",
         "Effect": "Allow"
      },
      {
         "Action": [
            "cloudwatch:ListMetrics",
            "cloudwatch:GetMetricStatistics",
            "cloudwatch:Describe*"
         ],
         "Resource": "*",
         "Effect": "Allow"
      },
```

```
        {
            "Action": "autoscaling:Describe*",
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

Додаток Б

```
<html>
<title>
    <head>Keep calm</head>
</title>
<body>
    <div align="center">
        <h1>Keep calm and carry a towel!</h1>
        <img
src="https://s3.amazonaws.com/acloudguruimages/ACG.jpg">
    </div>
</body>
</html>
```

```
<html>
<title>
    <head>Error hitchhacker</head>
</title>
<body>
    <div align="center">
        <h1>Sorry hitchhackers, this website was wiped out by Wagons</h1>
        <img src="https://s3.amazonaws.com/acloudguruimages/acg2.png">
    </div>
</body>
</html>
```

```
{
"Version": "2012-10-17",
"Statement": [
        {
                "Sid": "PublicReadGetObject",
                "Effect": "Allow",
                "Principal": "*",
                "Action": [
                        "s3:GetObject"
                ],
                "Resource": [
                        "arn:aws:s3:::BUCKET_NAME/*"
                ]
        }
]
}
```

nikto -h http://hitchhackers-s3-website.s3-website-us-east-1.amazonaws.com/

- Nikto v2.1.6

+ Target IP: 52.217.74.139

+ Target Hostname: hitchhackers-s3-website.s3-website-us-east-1.amazonaws.com

+ Target Port: 80

+ Start Time: 2021-12-05 03:59:57 (GMT-5)

+ Server: AmazonS3

+ The anti-clickjacking X-Frame-Options header is not present.

+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS

+ Uncommon header 'x-amz-id-2' found, with contents: slsMlNZ2iatpdixbJYPys6Dkdk79Ba7tZWKeOSueMVVFHuveDjzTQGBKCeuz3AF3mbQYJM1ANQY=

+ Uncommon header 'x-amz-request-id' found, with contents: VKAQV9QG0KGNDV5X

+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type

+ Uncommon header 'x-amz-error-message' found, with contents: The specified key does not exist.

+ Uncommon header 'x-amz-error-detail-key' found, with contents: 90ZP9ptv.home

+ Uncommon header 'x-amz-error-code' found, with contents: NoSuchKey

+ No CGI Directories found (use '-C all' to force check all possible dirs)

```
var mysql = require('mysql');
var connection = mysql.createConnection({
   host: "cd8vk0mvtlayu7.cypd2fbslchu.us-east-1.rds.amazonaws.com",
   user: "username",
   password: "password",
   database: "example",
});
exports.handler = (event, context, callback) => {
   connection.query('show tables', function (error, results, fields) {
      if (error) {
         connection.destroy();
         throw error;
      } else {
         // connected!
         console.log("Query result:");
         console.log(results);
         callback(error, results);
         connection.end(function (err) { callback(err, results);});
      }
   });
};
```

```
var mysql = require('mysql');
var AWS = require('aws-sdk'),
    region = "us-east-1",
    secretName = "RDScredentials",
    secret,
    decodedBinarySecret;
var client = new AWS.SecretsManager({
    region: "us-east-1"
});
exports.handler = (event, context, callback) => {
    client.getSecretValue({SecretId: secretName}, function(err, data) {
        if (err) {
            console.log(err);
        }
        else {
            // Decrypts secret using the associated KMS CMK.
            // Depending on whether the secret is a string or binary, one of these fields will be populated.
            if ('SecretString' in data) {
                secret = data.SecretString;
            } else {
                let buff = new Buffer(data.SecretBinary, 'base64');
                decodedBinarySecret = buff.toString('ascii');
            }
        }
        var parse = JSON.parse(secret);
        var password = parse.password;
```

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація)