Міністерство освіти і науки України НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра інформатики факультету інформатики

НАЛАШТУВАННЯ СІ/СО ПРОЦЕСІВ ДЛЯ АМАΖОΝ

Текстова частина до курсової роботи за спеціальністю МП-1 "Комп`ютерні науки"

Керівник курсової роботи

Ст.викл. _________ (прізвище та ініціали)

(nidnuc)

"____" _____ 2021 p.

Виконав студент Алсксссв Андрій (прізвище та ініціали)

"____" ____ 2021 p.

Київ 2021

Оглавление

Анотація	3
Перелік прийнятих скорочень	3
Вступ	4
Розділ 1: Налаштування сі/сd процесів	6
1.1 Загальні відомості про CI/CD ланцюг	6
1.2 Налаштування серверу збірки Jenkins	9
Розділ 2: Порівняння сервісів моніторингу	15
2.1 Проблема моніторингу	15
2.2 Огляд Nagios	16
2.3 Огляд Prometheus	17
Розділ 3: Налаштування сервісів моніторингу CloudWatch	18
3.1 Налаштування CloudWatch	18
3.2 Налаштування CloudWatch	19
3.3 Налаштування кастомних сервісів в CloudWatch	21
Розділ 4: Налаштування сервісів моніторингу Nagios	23
4.1 Налаштування Nagios	23
4.2 Налаштування моніторингу Nagios	24
Висновки	
Список використаної літератури	29

Анотація

Об'єктом моєї курсової роботи є налаштування CI/CD процесу для проекту, розгорнутому у хмарному сервісі Amazon, а також налаштування сервісу моніторингу. Метою проекту є створення повноцінного CI/CD ланцюга та впровадження сервісів для моніторингу завантаженості серверу, перевірка строку доменного імені, моніторинг мережевих служб та s3 сховища. Робота включає в собі виконання таких задач, як отримання сервером збірки вихідного коду з репозиторію, виконання тестів та розгортання на prod сервері, запуск сервісів моніторингу.

Під час виконання роботи було порівняно сервіс моніторингу CloudWatch з Nagios, обґрунтовано доцільність використання обох сервісів для різних типів задач, та впроваджено відповідні сервіси моніторингу після налаштування ci/cd pipeline.

В результаті було створено проект з повністю налаштованим CI/CD ланцюгом та запущеними сервісами моніторингу, що значно спрощують процес розробки.

Перелік прийнятих скорочень

CI/CD (англ. Continuous Integration / Continuous delivery) - практика розробки програмного забезпечення, метою якої є вирішення проблем інтеграції та виконання таких задач, як автоматична збірка проекту, проведення тестів та розгортання застосунку на цільовому сервері.

AWS (англ. Amazon Web Services) – комерційна платформа хмарних обчислень EC2 (англ. Amazon Elastic Compute Cloud) – частина інфраструктури AWS, надає обчислювальні потужності в хмарі.

VCS (англ. Version Control System) – система керування версіями.

Вступ

За останні кілька років практика впровадження CI/CD процесів у свій проект стала однією з найпопулярніших практик при розробці програмного забезпечення. З використанням хмарних сервісів, таких як AWS або Microsoft Azure, виникає більше проблем (таких як горизонтальна масштабованість, наявність безлічі користувачів з різними правами доступу) та виникає необхідність переходу на наступний рівень абстракції, для того, щоб керувати не окремими сервісами або програмами, а серверами, як мінімальної одиницею системи. Також виникає необхідність налаштовувати сервіси моніторингу, адже з використанням більше десяти-двадцяти різних сервісів слідкувати за всім цим з дашборду AWS стає не зручно. Отже питання інтеграції проекту з сі/сd сервісами є однією з актуальних тем сьогодення.

Питання налаштування сервісів моніторингу для AWS доволі об'ємне, сам AWS надає власні сервіси для моніторингу, але вони не повністю покривають потреби розробників. Тому важливо знати про альтернативні сервіси та вміти налаштовувати їх для виконання своїх задач.

Через це, за мету курсової роботи було поставлено для створеного тестового проекту налаштувати ланцюг CI/CD, який міг продемонструвати можливості CI/CD системи вцілому, налаштувати сервер збірки Jenkins (разом з Jenkins Slaves для вирішення проблем масштабування) та впровадити сервіси моніторингу для необхідних систем, в часності для мережевих служб та моніторингу стану серверів.

Робота складається з чотирьох розділів.

Перший розділ присвячено налаштуванню сі/сd процесів для застосунку web-cepвepy. В цьому розділі продемонстровано переваги використання Jenkins, його можливості та інтеграція з AWS EC2. В цьому розділі буде повністю вирішено проблеми отримання коду з репозиторію, налаштування slave інстансів, виконання тестів та розгортання коду на цільовому сервері.

В другому розділі буде порівняно сервіси моніторингу, обгрунтовано доцільність використання кожного з них

Третій розділ буде присвячено налаштуванню сервісів моніторингу CloudWatch.

Четвертий розділ присвячено налаштуванню кастомних сервісів моніторингу nagios та продемонстровано потужність цієї системи. Постановка задачі:

- 1. Для проекту налаштувати СІ/СД ланцюг:
 - Налаштувати сервер збірки AWS
 - Налаштувати отримання початкового коду з репозиторію;
 - Після кожного Pull Request:
 - о Виконати збірку проекту на сервері;
 - о Виконати Unit тести;
 - о Розгорнути готовий проект.
- 2. Порівняти сервіси моніторингу (CloudWatch, Nagios, Prometheus)
- 3. Налаштувати сервіси моніторингу Cloud Watch:
 - Повідомлення для стандартних метрик;
 - Дізнатися налаштування логування і, за необхідності, скорегувати їх в рантаймі
- 4. Налаштувати сервіси моніторингу Nagios:
 - Налаштуваня
 - Метрики Amazon EC2 (CPU Credit Usage, Network In/Out);
 - Власні метрики (SQL, Apache)

Розділ 1: Налаштування сі/сд процесів

1.1 Загальні відомості про CI/CD ланцюг

Безперервна інтеграція (англ. CI, Continuous Integration) - це практика розробки програмного забезпечення, за якої зміна коду в репозиторії проекту викликає певні дії, такі як автоматичне тестування коду на заданих тестах, розгортання проекту на цільовому сервері.

Безперервна інтеграція націлена на прискорення і полегшення розробки програмного забезпечення та виявлення проблем, що виникають в процесі. При налаштування регулярної інтеграції обсяг перевірок зменшується. В результаті на налагодження наведених процесів витрачається менше часу. Також можна додати автоматичне виконання тестів та інші види контролю, а також при масштабуванні (горизонтальному чи вертикальному) не витрачати додатковий час на розгортання застосунку та бази даних на інших серверах. Це спрощує як процес розробки, так і дозволяє зробити code review (рецензування коду) більш зручним, економить час розробників і унеможливлює виникнення певних типів помилок, які могли б виникнути при неавтоматизованому процесі розгортання коду на серверх.

На Рисунку 1.1 зображено принцип роботи CI/CD платформ. Після Pull Request`у в репозиторій, отримані зміни відправляються на CI сервер (сервер збірки), де виконуються тести та збірка проекту, якщо усі тести були пройдені успішно, після чого отриманий продукт розгортається на цільовому сервері.



Рисунок 1.1 – Принцип роботи CI/CD процесу

В якості СІ платформи було обрано Jenkins. Jenkins - це крос-платформний додаток для постійної інтеграції та безперервної доставки, який можна використовувати для постійного створення та тестування програмних проектів, що полегшує розробникам інтеграцію змін до проекту та полегшує отримання нової збірки користувачами або тестувальниками, яка в свою чергу збільшує вашу продуктивність.

Одна з причин вибору саме Jenkins – він легко інтегрується з низкою служб AWS, таких як AWS CodeCommit, AWS CodeDeploy, Amazon EC2 Spot та Amazon EC2 Fleet. Процес розгортання Jenkins на AWS у Amazon Elastic Compute Cloud (Amazon EC2) доволі швидкий та не сильно трудомісткий.

Jenkins підтримує архітектуру master-slave, тобто багато рабів працюють на господаря. Він також відомий як Jenkins Distributed Builds. Це також дозволяє запускати завдання в різних середовищах, таких як Linux, Windows, MacOS тощо. Ми також можемо паралельно запускати один і той самий тестовий приклад у різних середовищах, використовуючи розподілені збірки Jenkins, що, в свою чергу, допомагає швидко досягти бажаних результатів за допомогою цього розподіленого підходу. Всі результати роботи збираються та об'єднуються на головному (master) вузлі для моніторингу.

Принцип роботи сервера збірки, що буде створено, наведено на Рисунку 1.2.

Jenkins master та Jenkins Slaves будуть розгорнуті на окремих Amazon EC2 instance, в межах одного Amazon Virtual Private Cloud (VPC).



Рисунок 1.2 – Принцип роботи Jenkins Build Server з кількома slave'ами

В якості застосунку буде використано простий тестовий сервер з парою ендпоінтів, лінка на застосунок: <u>ci_cd_test_server</u>.

1.2 Налаштування серверу збірки Jenkins

Перейдемо власне до налаштування серверу збірки в ЕС2. Для цього необхідно мати аккаунт AWS, AWS Identity, ім`я та пароль Access Management (IAM) користувача, Amazon EC2 instance для головного вузла (master) Jenkins, та інші EC2 instance для слейвів, налаштований VPC. Моя робота не зачіпає процес створення та налаштування усього вищенаведеного, але на Рисунку 1.3, 1.4, 1.5 наведено інформацію про інстанс для master та для slave відповідно.

Details Security Networking Storage Status checks Monitoring Tags							
 Security details 	S						
IAM Role			Owner ID		Launch time		
jenkins_ec2_role			D 953643899422		Tue May 11 2021 01:31:09 GMT+0300 (Восточная Европа, летнее время)		
Security groups							
🗗 sg-0e3e5026e	59bf2d26 (instanceSG)						
 Inbound rules 							
Q Filter rules					< 1 >		
Port range	Protocol	Source	Security groups				
80	ТСР	0.0.0/0	instanceSG				
80	ТСР	::/0	instanceSG				
8080	ТСР	0.0.0/0	instanceSG				
8080	ТСР	::/0	instanceSG				
22	ТСР	159.224.171.95/32	instanceSG				
Outbound rule	15				Активация Windows Чтобы активировать Windows, перейдите в раздел		

Рисунок 1.3 – Security page для Jenkins master серверу

Instance summary for i-0990483d9a6fa87b7 Info Updated less than a minute ago		C Connect Instance state V
Instance ID	Public IPv4 address	Private IPv4 addresses
i-0990483d9a6fa87b7	34.200.255.81 open address 🗹	172.31.3.208
Instance state	Public IPv4 DNS	Private IPv4 DNS
	ec2-34-200-255-81.compute-1.amazonaws.com open address	D ip-172-31-3-208.ec2.internal
Instance type	Elastic IP addresses	VPC ID
t2.micro	-	D vpc-d557d9a8 🖸
AWS Compute Optimizer finding	IAM Role	Subnet ID
Opt-in to AWS Compute Optimizer for recommendations. Learn more 2	jenkins_ec2_role	🗇 subnet-44dfad22 🔼

Рисунок 1.4 – Instance summary для Jenkins master серверу

Instance summary for i-03a02af4ade1397dd Info Updated less than a minute ago		C Connect Instance state V
Instance ID D i-03a02af4ade1397dd	Public IPv4 address D 3.228.3.53 open address	Private IPv4 addresses IP 172.31.10.58
Instance state Running 	Public IPv4 DNS extrm compute-1.amazonaws.com open address	Private IPv4 DNS D ip-172-31-10-58.ec2.internal
Instance type t2.micro	Elastic IP addresses -	VPC ID Vpc-d557d9a8 🛃
AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations. Learn more [2]	IAM Role -	Subnet ID Subnet-44dfad22

Рисунок 1.5 – Instance summary для Jenkins slave

Відкритий порт 8080 необхідний для доступу до Jenkins admin page. Тож тепер необхідно завантажити на перший (master) ec2 instance сам Jenkins, та запустити його. Після цього можна переходити до налаштувань. Дуже зручний є плагін Amazon EC2, тож варто встановити його на сервер (на Рисунку 1.6 показано який саме плагін необхідно встановити).

🎙 Jenkins		Q поиск	i.	② Andrew Alexeet	🛨 выхо
Dashboard 🤌 Управление плаг	инами				
🖢 Домой	Q hilter				
🚰 Настроить Jenkins	Обновления	Доступные Установленные Дополнительно			
Update Center	Активный	Наименование і	Версия	Ранее установленные версии	Удалить
	Ø	Amazon EC2 plugin This plugin integrates Jenkins with <u>Amazon EC2</u> or anything implementing the EC2 API's such as an Ubuntu.	1.58		Удалить
	8	Amazon Web Services SDK This plugin provides <u>AWS SDK for Java</u> for other plugins.	1.11.995		Удалить
	8	Ant Plugin Adds Apache Ant support to Jenkins	1.11		Удалить
	ø	Apache HttpComponents Client 4.x API Plugin Bundles Apache HttpComponents Client 4.x and allows it to be used by Jenkins plugins. This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt.a Plugin initiative for more information.	45.13-1.0	Arrunauure Windows	Удалить
	2	Bootstrap 4 API Plugin Provides Bootstrap 4 for Jenkins plugins.	4.6.0-3	Чтобы активировать Windows, перей, "Параметры",	ите в раздел Удалить

Рисунок 1.6 – Необхідно встановити ЕС2 плагін

Після цього в вкладці Концфігурація системи необхідно додати новий Cloud, як показано на рисунку 1.7.

Configure Clouds

Amazon EC2	
Name	
AWS Cloud	
Amazon EC2 Credentials 🕜	
- none - 👻 🚅 Add 🌱	
W/S IAM Access Key used to connect to EC2. If not specified, implicit authentication mechanisms are us	ed (IAM roles)
Use EC2 instance profile to obtain credentials	•
Alternate EC2 Endpoint	
Used to populate the available regions dropdown. Only set this if you're using a different EC2 endpoint	Le operating in govcloud).
The regions will be populated once the keys above are entered.	
Region 🕜	
eu-north-1	
EC2 Key Pair's Private Key 😮	
ec2-user 👻 🖛 Add 👻	
	Активация Windows
	Чтобы активировать Windows, перейди теst Connection
Save Apply	

Рисунок 1.7 – Додавання Cloud до Jenkins

Вказавши тип EC2 і після того як ми введемо облікові данні для доступу до EC2 інстансу (можна давати доступ у вигляді username та паролю до власного облікового запису, так і використати ec2 instance profile, якщо налаштувати AMI для master серверу).

Після цього можна створювати нові завдання (jobs), такі як тестування коду. Щоб додати новий test job, необхідно мати встановлений плагін (Рисунок 1.8).



Рисунок 1.8 – Плагін TestComplete support

I тепер можна налаштовувати нові агенти Jenkins та власне задачі. На

рисунках 1.9-1.10 показано основні етапи створення нового job`у.

General	Управление исходным кодом	Триггеры сборки	Среда сборки	Сборка	Послесборочные о	перации
Описание						
[Plain text] 🛙	Іредпросмотр					//
GitHub	project					
Project u	rl					0
https://g	github.com/Alexeev-Andrew/test_ser	ver_ci_cd/				
						Расширенные
This bui	ld requires lockable resources					0
	builds					()
Эдалять	раметризованная сборка					0
Приоста	ановить сборки					0
🗹 Разреш	ить параллельный запуск задачи					?
🗹 Ограни	чить лейблы сборщиков, которые	могут исполнять дан	ную задачу			?
Label Ex	pression					?
master						
Label ma	aster matches 1 node. Permissions or	other restrictions prov	vided by plugins may	y further redu	uce that list.	Акти
						Чтобы
Сохран	ить Применить					Расширенные ар

Рисунок 1.9 – Створення нового job в Jenkins (1)

Необов'язкові налаштування:

- *Test runner*: TestComplete або TestExecute; вказує, чи буде крок використовувати TestComplete або TestExecute для тестового запуску. Якщо обидва встановлені та вибрано будь-який, запускається TestExecute
- *Versions*: 9, 10, 11, 12 або 14; Визначає версію TestComplete або TestExecute, яка використовуватиметься у випадку, якщо на вузлі встановлено кілька версій цих продуктів. Jenkins використовує останню встановлену версію за замовчуванням

- Action on warnings: вказує, чи повинен Дженкінс позначати всю збірку як невдалу чи нестабільну, коли журнал тесту TestComplete містить попередження
- Action on errors: вказує, чи повинен Дженкінс позначати всю збірку як невдалу чи нестабільну, коли журнал тесту TestComplete містить помилки
- Additional command line arguments: визначає довільні аргументи командного рядка, які передаються TestComplete. Аргументи "/ run", "/ SilentMode", "/ ForceConversion", "/ ns", "/ exit" додаються за замовчуванням
- Use test timeout: визначає максимальний час виконання тесту в секундах.
 Якщо тест не буде закінчено до закінчення зазначеного періоду, Дженкінс позначить всю збірку як невдалу

General	Управление исходным кодом	Триггеры сборки	Среда сборки	Сборка	Послесборочные операции
Add tin	nestamps to the Console Output				
Inspect	build log for published Gradle buil	d scans			
U With A	nt				•
Сборка					
TestCo	mnlete Test				X
For info	repreterest	ast stops, son TestCom	lata dagumantation		
For Inic	ormation on creating restcomplete to	est steps, see <u>restcomp</u>	blete documentation	<u>l</u> .	
Project	suite file:				
test_se	erver_ci_cd\TCSuite1.pjs				
Specify	the path relative to the job's worksp	ace folder on the node	(for example, MyPro	jects\TCSuit	e1.pjs).
	· · · ·			-	
O Des	ire suite				
	ject test a (Taat Complete 14 20 or later is room	vine d)			
	s (TestComplete 14.20 of later is requ	Jirea)			
O Scri	ipt test				
⊖ Key	word test	tt-)			
0 Otr	ier (low-level procedure, hetwork sui	te, etc.)			
Setti	nas				
					Расширенные
Добавит	ъ шаг сборки 🔻				Акти
_					Чтобы
Послес	ророчные операции				"Парам
Сохра	Применить				

Рисунок 1.10 – Створення нового job в Jenkins (2)

Для запуску тестів натискаємо кнопку "Зібрати зараз" в правій менющці, після чого сервер буде зібрано та виконано усі тести.

Для деплою зібраного проекту на цільовий сервер необхідно налаштувати Pipeline (Рисунок 1.11) та додати Source Code Management з трігером *GitHub hook trigger for GITScm polling*.

ipeline			
Definition	Pipeline sc	ript	
	Script	1 ~ node 2 3 ~ 4 5 6 7 8 ~ 9 10 11 12 ~ 13 14 15 16 ~	<pre>{ stage ('Prepare environment') { git branch: 'development', url: 'git@bitbucket.org:example/myapp.git' sh 'npm install' } stage ('Code analyse') { sh 'echo "Run some lints"' } stage ('Unit test') { sh 'echo "Tests will back"' } stage ('Build') { t isse clearing } }</pre>

Рисунок 1.11 – Налаштування pipeline

Branches to build	Branch Specifier (blank for 'any */master	X	0	
		Add Branch		
Repository browser	(Auto)		•	0

Рисунок 1.12 – Налаштування гілки репозиторію, з якої буде братись код

Тепер в нас є налаштований СІ ланцюг з використанням Jenkins, після змін в репозиторії викликається збірка проекту в master ec2 instance, виконуються тести та готовий застосунок розгортається на цільовому сервері.

Розділ 2: Порівняння сервісів моніторингу

2.1 Проблема моніторингу

Коли в системі щось ламається або починає вести себе незвичайним чином, користувачі та розробники зіштовхуються з певними проблемами. Отже, в цьому випадку необхідно якомога швидше повідомити кого-небудь про цю некоректну поведінку або про збій в системі. А ще краще було б передбачити виникнення проблем заздалегідь та повідомити ще до самого збою. Саме такого роду питання вирішуються за допомогою сервісів моніторингу.

Моніторити можна майже усе – від Load average серверу до закінчення коштів на AWS аккаунті з повідомленням про необхідність поповнити рахунок. Якщо ви використовуєте AWS – то у вас може бути не один десяток сервісів, і Ви захочете моніторити майже кожен з них. Це полегшить розробку, прибере необхідність перевіряти баланс на аккаунті, дасть інформацію по використанню того чи іншого сервісу. Також є певні інваріанти, які не мають ніколи порушуватись, і моніторинг щоб це виконувалось – теж одна з можливих задач. Також такі сервіси дозволяють відправляти повідомлення по пошті або SMS розробникам або менеджерам, що автоматизує процес знаходження та виявлення помилок.

AWS має власну систему моніторингу – CloudWatch. Вона надає можливість моніторити дуже багато власних (amzone) сервісів, таких як ec2 (Disk Read Bytes Average, CPU Utilization Average, тощо), rds (CPU Utilization Average, Database Connections Sum, Freeable Memory Average тощо), lambda, Elastic Beanstalk та багато іншого. Серед недоліків CloudWatch можна виокремити те, що для певних метрик необхідно писати код на python або інших мовах, також Dashboard доволі незручний, немає stage warnings, та важко налаштувати моніторинг власних сервісів.

Тому виникає необхідність знайти щось більш зручніше, хоча б для моніторингу власних сервісів. Серед популярних рішень варто виокремити Nagios та Prometheus.

2.2 Огляд Nagios

Nagios - лідер у галузі моніторингу IT-інфраструктури. Він пропонує безліч рішень для задоволення R&D потреб, що стосуються як ділових, так і технічних проблем. Nagios сприяє високій доступності програм, надаючи інформацію про продуктивність бази даних. Це також може допомогти у плануванні потужності та управлінні витратами. Nagios пропонує чотири різні продукти на вибір: Nagios XI, Nagios Log Server, Nagios Network Analyzer та Nagios Fusion.

Nagios XI - це серверна та мережева система моніторингу, яка забезпечує роботу даних, щоб відстежувати стан додатків або мережевої інфраструктури, продуктивність, доступність компонентів, протоколів та послуг. Він має зручний інтерфейс, що дозволяє конфігурувати інтерфейс, налаштовувати візуалізацію та налаштування попереджень.

Хоча Nagios XI здебільшого призначений для моніторингу 1) метрик програми або інфраструктури та 2) порогових значень, Nagios Log Server призначений для управління журналами та аналізу сценаріїв користувача. Він має можливість співвідносити зареєстровані події між різними службами та серверами в режимі реального часу, що допомагає у розслідуванні інцидентів та проведенні аналізів першопричин.

Оскільки дизайн Nagios Log Server спеціально призначений для мережевої безпеки та аудиту, він дозволяє користувачам генерувати попередження про підозрілі операції та команди. Log Server зберігає історичні дані всіх подій, забезпечуючи організації усім необхідним для проходження перевірки безпеки.

Nagios Network Analyzer - це інструмент для збору та відображення метрик або додаткової інформації про мережу додатків. Він визначає, які IP-адреси взаємодіють із серверами додатків та які запити вони надсилають. Мережевий

аналізатор веде облік всього трафіку сервера, включаючи підключення певного сервера до певного порту та конкретного запиту.

Це допомагає планувати пропускну здатність сервера та мережі, а також розуміти різні види порушень безпеки, наприклад, несанкціонований доступ, витоки даних, DDoS атаки, а також віруси та шкідливі програми на серверах.

Nagios Fusion - сукупність трьох інструментів, які пропонує Nagios. Він забезпечує повне рішення, яке допомагає компаніям задовольнити будь-які вимоги до моніторингу. Його конструкція призначена для масштабованості та видимості програми та всіх залежностей.

2.3 Огляд Prometheus

Prometheus та Nagios пропонують різні функціональні можливості. Перш за все, Nagios більше зосереджується на мережевому трафіку та безпеці додатків, тоді як Prometheus - на прикладних аспектах програми та її інфраструктурі.

Prometheus збирає дані з додатків, які надсилають метрики до своїх кінцевих точок API (або експортерів). Nagios використовує агенти, які встановлюються як на елементи мережі, так і на компоненти, які вона контролює; вони збирають дані за допомогою методології витягування.

Графіки та інформаційні панелі, які надає Prometheus, не відповідають сучасним потребам DevOps. Як результат, користувачі вдаються до інших інструментів візуалізації для відображення метрик, зібраних Prometheus.

Nagios постачається з набором інформаційних панелей, які відповідають вимогам моніторингових мереж та компонентів інфраструктури. Тим не менше, йому все ще бракує графіків для додаткових проблем, пов'язаних із застосуванням.

Тому для демонстрації роботи сервісів моніторингу було обрано Nagios, як найбільш потужне і популярне рішення.

Розділ 3: Налаштування сервісів моніторингу CloudWatch

3.1 Налаштування CloudWatch

Для того, щоб почати працювати з CloudWatch необхідно встановити Amazon CloudWatch Command Line Tools та доступ до серверу (для налаштування кастомних сервісів моніторингу).

Коли ви зайдете на сторінку CloudWatch, ви одразу побачите інформацію по сервісам, які ви використовуєте (Рисунок 3.1)



Рисунок 3.1 – Головна сторінка CloudWatch

Одразу є можливість створити Dashboard, для цього необхідно обрати сервіси, для яких буде створено цей дашборд, вказати назву та кастомізувати налаштування. Також можна подивитись log groups, де буде відображено логи до всіх застосунків, які мають дозвіл на запис логів (Рисунок 3.2).

Edit VpcAccess		1	2
A policy defines the AWS permissions that you can assign to a user, group, or role. You can create and edit a policy in the visual editor and using JSON. Learn more			
Visual editor JSON	Import	t managed	policy
Expand all Collapse all			
> EC2 (5 actions)	Clone	Remove	
	Clone	Remove	
Service CloudWatch Logs			
Actions Write CreateLogGroup			
CreateLogStream PutLogEvents			
Resources All resources			
Request conditions Specify request conditions (optional)			
O Add a	ditional per	rmissions	

Рисунок 3.2 – VPC access Permissions – Allow write CloudWatch Logs

3.2 Налаштування CloudWatch

Подивитись усі LogGroups можна у відповідній вкладці. Тут будуть відображатись логи Lambda функцій, appsync logs тощо. Обравши необхідну групу, можна побачити усі логи у цій групі (Рисунок 3.3, 3.4)

aws Services V			Q Bearch for services, features, marketplace products, a	nd docs [Alt+S]		<u>ک</u> ک	aslexeev @ 9995-7617-1221 ♥ N. Virginia ♥ Support
CloudWatch ×	Â	CloudWa	tch 📏 CloudWatch Logs 义 Log groups				
Dashboards Alarms In alarm		Log e By def	groups (5) lault, we only lead up to 10000 log groups.			C Actions ¥	View in Logs Insights Create log group
OK 2		Q	Filter log groups or try prefix tearch	- D	act match		< 1 > @
Billing Logs			Log group	Retention	v Metric filters	v Contributor Insights	v Subscription filters v
Log groups			/aws/appsync/apis/t56nflpcorfizn/wrgfdblgneu	Never expire	8	95.	252
Metrics			/aws/appsync/apis/yzkyxkbkbnb6tf3gx4f4i5cpgq	Never expire		8	(*)
Explorer			/aws/lambda/amplify-workmovrchat-dev-0142-UserPoolClientLambda-E5BD20802	Never expire	84	64 - C	020
Events			/aws/lambda/amplify-workmovrchat-dev-UpdateRolesWithIDPFuncti-10337TDK0HPX	Never expire	8	21	2.53
Rules Event Buset			/aws/lambda/SendMessagesToApi-AmazonSendMessageLambdaFunction-1AACYVTI	Never expire			1.5V
ServiceLens Service Map Traces	Ľ						
Container Insights New	6						
Resources Performance monitoring							
Lambda Insights (New Performance monitoring							
Synthetics Canaries							
Contributor Insights Settings						Активаці 9тобы акти	ИЯ Windows перебать Windows, перебаите в рездел
Favorites							

Рисунок 3.3 – Log groups

	٩	Search for services, features, marketplace products, and docs [Alt+5]	🔀 🙏 aalexeev @ 9995-7617-1221 👻 N. Virginia 👻 Suppor
udWa	atch > CloudWatch Logs > Log g	roups > /aws/lambda/amplify-workmowrchat-dev-UpdateRolesWithIDPFuncti-10337TDXK0HPX > 2020/07	7/25/[\$LATEST]043d90fb089847aeacb86e47d714b445
Log You d	events an use the filter bar below to search fo	sr and match terms, phrases, or values in your log events. Learn more about filter patterns 🕑	☐ View as text C Actions ▼ Create Metric Filter
٩	Filter events		Clear 1m 30m 1h 12h Custom 🗐 🥥
۲	Timestamp	Message	
		No older events at this moment. Retry	
ĸ.	2020-07-25119:23:40.954+03:00	START RequestId: 7ab3a507-84c8-4c41-a9c2-9dc2258f67c6 Version: \$LATEST	
κ.	2020-07-25719:23:42.257+03:00	2020-07-25716:23:42.2572 7ab3a507-04c0-4c41-a9c2-9dc2258/67c6 INFO createORupdate[object Object],[obje	ect Object)
	2020-07-25719:23:42.257+03:00	2020-07-25T16:23:42.2572 7ab3a507-04c0-4c41-a9c2-9dc2258f67c6 INFO response data[{"ResponseMetadata":{	["RequestId":"f6722f70-8270-443b-a084-8b23f4le2deb"}},{"ResponseHetadata":{"RequestId".
	2020-07-25T19:23:42.250+03:00	2020-07-25716:23:42.2502 7#b3#507-04c0-4c41-#9c2-9dc2250#67c6 INFO Response body: ("Status":"SUCCESS",	"Reason":"See the details in Cloudwatch Log Stream: 2020/07/25/[\$LATEST]043d90fb009984.
	2020-07-25719:23:42.404+03:00	2020-07-25116:23:42.404Z 7ab3a507-04c0-4c41-a9c2-9dc2258f67c6 INFO Status code: 200	
	2020-07-25719:23:42.404+03:00	2020-07-25716:23:42.4042 7ab3a507-04c0-4c41-a9c2-9dc2258f67c6 INFO Status message: OK	
	2020-07-25110:23:42.406+03:00	END RequestId: 7ab3a507-04c0-4c41-a9c2-9dc2258f67c6	
÷	2020-07-25719:23:42.406+03:00	REPORT RequestId: 7ab3a507-04c0-4c41-a9c2-9dc2258f67c6 Duration: 1451.86 ms Billed Duration: 1500 ms M	Nemory Size: 128 MB Max Memory Used: 89 MB Init Duration: 404.77 ms
		No newer events at this moment. Auto retry poused. Resume	

Рисунок 3.4 – Log в одній з log group

Для створення нового Alarm необхідно зайти у вкладку Alarms та створити новий. Один з прикладів налаштованого Alarm показано на Рисунку 3.5.

oudWatch > Alarms > workmovrCha	UserTable-WriteCapacityUnitsLimit-BasicAlarm								
🗈 Alarms (2)	⊘ workmovrChatUserTable-W	riteCapacityUnitsLimit-BasicAlarm		C					
Q Search	Details	Details							
Any state	•								
Any type	▼ Name	State	Namespace	Datapoints to alarm					
Hide Auto Scaling alarms	WriteCapacityUnitsLimit-BasicAlarm	O OK	Aws/bynamobb	5 out of 5					
	Tune	Threshold	Metric name	Missing data treatment Treat missing data as missing Percentiles with low samples					
<17	Metric alarm	datapoints within 5 minutes	consumedwriteCapacityUnits						
workmovrChatUserTable-	0	the states and	TableName						
WriteCapacityUnitsLimit-BasicAlarm	No description	2020-07-27 19:24:04	workmovrcnatuserrable	evaluate					
Ø OK			Statistic	ARN					
			Sum	am:aws:cloudwatch:us-east- 1:999576171221:alarm:workmovrChatUse					
ReadCapacityUnitsLimit-BasicAlarm	0.1		Period	rTable-WriteCapacityUnitsLimit-					
Metric alarm			1 minute	BasicAlarm					
Ø OK	A Mary Frankling and								
	 View Eventshoge rule 								
	Actions								
	Туре	v Description		Config					
	Notification	Notification When in alarm, send message to topic "dynamodb" Configuration							
			Активаци	ia Windows					
	History (0)		Чтобы актив	ировать Windows, перейдите в раздел					

Рисунок 3.5 – Alarm для ConsumedWriteCapacityUnits для однієї з таблиць DynamoDB

3.3 Налаштування кастомних сервісів в CloudWatch

Якщо ми хочемо зробити моніторинг кастомної метрики, ми можемо створити власний демон на сервері, для якого хочемо налаштувати моніторинг. Покажу на прикладі отримання інформації про Load Average серверу, який дає повну картину стану сервера.

Для цього створемо скрипт на сервері:

```
load_average=$(uptime | awk -F'load average:' '{ print
$2 }' | awk '{ print $2 }')
load_average=${load_average%%','}
```

Для регістрації значень метрики використаємо mon-put-data --metric-name "LoadAverage" -namespace "CustomMetric" --timestamp \$timestamp -value \$load_average

```
Де
--metric-name «LoadAverage» — ім`я метрики
--namespace «CustomMetric» — місце, де будемо зберігати метрику
```

Для того, щоб запустити нашого демона, створемо скрипт для запуску та зупинки нашого скрипту.

```
#!/bin/bash1 #chkconfig
 . /etc/rc.d/init.d/functions
#Set environement
export ROOT=/opt/aws
start() {
    $ROOT /cw_scaler.sh&
}
stop() {
```

```
kill $(ps ax | grep '/opt/aws/cw_scaler.sh' | grep -
v "grep" | awk '{print $1}')
    }
    case "$1" in
         start)
             echo "Starting CW Load Average."
             Start
             ;;
         stop)
             echo "Stopping CW Load Average."
             stop
             ;;
         * )
             echo $"Usage: cw_scaler.sh {start|stop}"
             exit 1
             ;;
```

esac

Через деякий час після запуску в панелі CloudWatch з'явиться новий тип метрик -CustomMetric, а у ньому побачимо LoadAverage (Рисунок 3.6)



Рисунок 3.6 – LoadAverage

Розділ 4: Налаштування сервісів моніторингу Nagios

4.1 Налаштування Nagios

Для того, щоб розгорнути сервіс моніторингу Nagios необхідно мати налаштований EC2 instance для Nagios server та для Remote monitoring client, я ж буду використовувати instance з build сервером як інстанс для Nagios серверу, щоб не створювати інший і зменшити витрати. Для security group необхідно додати Custom TCP rule для 5666 порта, як показано на рисунку 4.1.

Type (i)	Protocol (j)	Port Range (i)	Source (i)	Description (i)	^
HTTP	TCP	80	0.0.0/0		
SSH	TCP	22	0.0.0/0		
Custom TCP Rule	TCP	5666	0.0.0/0		12
All ICMP - IPv4	All	N/A	0.0.0/0		*

Рисунок 4.1 – LoadAverage

Щоб працювати з Nagios необхідно встановити на сервер httpd, PHP та деякі бібліотеки. Зробити це можна командою

yum install -y httpd httpd-tools php gcc glibc glibc-common gd gd-devel

Далі необхідно створити нового користувача та встановити йому пароль.

Після цього завантажуємо та вигружаємо Nagios Core та плагіни до нього. Тепер ми готові до налаштування.

Щоб налаштувати Nagios потрібно зайти в папку Nagios, та виконати команду ./configure --with-command-group=nagcmd. Після цього кастомізуємо налаштування:

sudo vim /usr/local/nagios/etc/objects/contacts.cfg

Вказуємо власну пошту, потім підтверджуємо що ми налаштували конфігураційний файл, додаємо Nagios Services до System Startap і ми готові для роботи з Nagios (Рисунок 4.2).

Nagios Current Network Status La Ubdet Te Nor 14 5113 00 UTC 2017 Updetd every 90 seconds Jopes Carr 4 2.2 - www najos crg Logget na negisedamin Home Vew hotrs for Apticas Vew hotrs for Apticas Vew hotrs for Apticas				Host Status Tol Up Down Unreachable	Pending Ok	Service Stat	m Critical Pending
				All Problems All 3	Types	All Types	
Tactical Overview Map (Legacy) Hosts Services Host Groups Summary Grid	Display Filt Host Sta Host Sta Service Sta Service P	Vers: Intus Types: All Properties: Any Intus Types: Wan Properties: Any 100	ning	•	Service	Is For All Hosts	
Service Groups " Summary	localhost	HTTP	WARNING	11-14-2017 15:07:05	Od 0h 7m 35s	4/4	Status Information HTTP WARNING: HTTP/1.1 403 Forbidden - 4284 bytes in 0.000 second response time
Grid Problems Services (Unhandled) Hosta (Unhandled) Network Outages	Results 1 - 1 o	f 1 Metching Serv	ices				
Quick Search:							
eports							
Trends (Legacy)							
Alerts							
History							
Histogram (Legacy)							
Notifications							

Рисунок 4.2 – Nagios web page

4.2 Налаштування моніторингу Nagios

Головним файлом налаштувань для Nagios є /usr/local/nagios/etc/nagios.cfg, також створимо папку для файлів сервісів та папку з налаштуваннями.

Створений файл конфігу:

```
define host {
   name generic-host
   register 0

   # як часто проводити перевірку (в хв)
   check_interval 1

   # Час перед повторною перевіркою (теж в хв)
   retry_interval 1

   # Кількість перевірок (retry), після яких хост вважається впавшим
   max_check_attempts 5
```

```
# Час між нотіфікаціями
notification_interval 30
# типи нотіфікацій (down/ unreachable)
notification_options u
check_command check-host-alive
}
```

Параметр register 0 вказує, що цей хост не буде відображатись в монітоингу.

Інший конфіг:

```
# відображає статистику MySQL.
```

define command {

```
command_name check-mysql
```

```
command_line admin/check_mysql -u admin -p Qwertyuiop -H $HOSTADDRESS$ -P $ARG1$
```

}

```
# тривалість виконання mysql-запитів
```

define command {

command_name check-mysql-long-query

command_line /usr/bin/perl admin/mysql_health_check.pl --hostname \$HOSTADDRESS\$ --port 3306 --user nagiosuser --password nagiospassword --mode=longquery --no_cache --warning=\$ARG2\$ --critical=\$ARG1\$

}

I додаємо моніторинг до хосту:

#define service{

hostgroup_name hostgroup-myproject-all-servers

service_description MySQL Status

check_command check-mysql!3306

```
use service-template-all-generic
```

}

```
define service{
    hostgroup_name hostgroup-myproject-all-servers
    service_description MySQL Long Query
    check_command check-mysql-long-query!20!30
    use service-template-all-generic
```

}

Тепер перезапускаємо Nagios сервер і можна дивитись логи. Після налаштування remote monitoring client, з`явиться можливість дивитись логи у Dashboard'i, як показано на рисунку 4.3.

Host **	Service **		Status **	Last Check **	Duration **	Attempt *+	Status Information
ec2-54-169-146-244.ap-southeast- 1.compute.amazonaws.com	Current Load		ок	11-14-2017 16:21:15	0d 0h 35m 37s	1/4	OK - load average: 0.00, 0.01, 0.05
	Current Users		OK	11-14-2017 16:20:48	0d 0h 36m 4s	1/4	USERS OK - 2 users currently logged in
	Root / Partition		UNKNOWN	11-14-2017 16:17:54	0d 0h 18m 58s	4/4	NRPE: Unable to read output
	SWAP Uasge		CRITICAL	11-14-2017 16:19:25	0d 0h 17m 27s	4/4	SWAP CRITICAL - 0% free (0 MB out of 0 MB) - Swap is either disabled not present, or of zero size.
	Total Processes		OK	11-14-2017 16:17:06	0d 0h 34m 46s	1/4	PROCS OK: 89 processes
	apache check by techhearts	۵.	ок	11-14-2017 16:24:49	Od Oh Om 3s	1/4	httpd is running!!

Рисунок 4.3 – Nagios remote monitoring client service status

Тепер ми маємо налаштований моніторинг декількох сервісів Nagios, що полегшує розробку. Також можна налаштувати відправку SMS /email повідомлень для певної події, зробити це можна так:

define contact{ contact_name vano alias Vano service_notification_period 24x7 host_notification_period 24x7 service_notification_options w,u,c,r host_notification_options d,r service_notification_commands notify-service-by-email,notify-service-by-sms host_notification_commands notify-host-by-email,notify-host-by-sms email <u>aalex@workmovr.com</u> # your email address1 +380950374482 # your phone

}

Висновки

Розглянута тема є дуже актуальною на даний момент, з стрімким розвитком DevOps та Ci/Cd зокрема, ця практика стала невід ємною частиною процесу розробки. Тому дуже важливо знати і уміти налаштовувати CI/CD ланцюжки та сервіси моніторингу для власних сервісів.

В результаті виконання курсової роботи було створено проект, для якого було налаштовано CI/CD процеси та сервіси моніторингу CloudWatch та Nagios. Можливості, які надає використання безперервної інтеграції, було продемонстровано використовуючи CI/CD платформу Jenkins. Даний приклад можна вважати еталонним при налаштуванні власних ланцюгів неперервної інтеграції для проектів, розгорнутих в AWS.

Список використаної літератури

Електронні	1.	Огляд Nagios <u>https://logz.io/blog/prometheus-vs-nagios-</u>
ресурси		metrics/#:~:text=Prometheus%20and%20Nagios%20offer%20dif
		ferent, API% 20endpoints% 20(or% 20exporters).
	2.	Nagios custom
		metrics: https://habr.com/ru/company/epam_systems/blog/147046
		<u>/</u>
	3.	Jenkins for aws: https://d1.awsstatic.com/Projects/P5505030/aws-
		project_Jenkins-build-server.pdf
	4.	Nagios for aws
		monitoring: https://www.nagios.com/solutions/aws-monitoring/