Definição da arquitetura

Tecnologias utilizadas

- Node.js, mysql, mongodb, redis, pm2, lambda function, load balancer, ec2, ses, rds, autoscaling e sqs.

Arquiteturas utilizadas

Microservice, serverless, event-driven architecture e rest.

Fluxo da arquitetura que extrai os dados das bases externas A, B e C

Como não tenho detalhes sobre as bases externas de como são as formas permitidas para extrair esses dados, então vou supor que tenha uma api é de tempo em tempo terá uma aplicação que vai consultar os dados é enviar para o SQS.

Quando a mensagem chega no sqs é disparada um lambda function para transformar os dados para o formato necessário é armazenar. O motivo de usar lambda function é que a amazon vira gerenciar quando precisar lidar com um grande volume de mensagens é quando o volume reduz é reduzido o número de lambda automaticamente. Outro fator interessante sobre utilizar a lambdas function é que você paga pelo que você usar, caso fosse usado ec2 para consumir as mensagens quando não tivesse mensagem iria continuar pagando pela instâncias do ec2 paradas.

Para armazenar os dados das bases A e B será utilizado o RDS(mysql) devido os dados tem estrutura fixa. O motivo de utilizar o RDS é devido alta disponibilidade, backup automático, facilidade para configurar o banco de dados é segurança oferecida pelo serviço onde a comunicação é criptografada e também permite especificar quais aplicação pode ser conectar no banco de dados utilizando o security group.

Para armazenar os dados da base C será utilizado o mongodb devido os dados não tem uma estrutura fixa é o mongodb por ser schemaless será adequado para o essa situação e também deve ser armazenadas todas as informações em uma coleção evitado fazer relacionamentos entre coleções que podem afetar a performance da consulta. Deve ser utilizado o serviço do Mongo Atlas para permitirá criar facilmente uma instância do mongodb, segurança oferecida pelo serviço onde a comunicação é criptografada e também permite especificar quais aplicações pode ser conectar no banco de dados onde é definido o ip que pode ser conectar ao banco de dados.

Fluxo da arquitetura que expõem os dados dos sistemas

Quando uma requisição é feita por uma aplicação cliente está requisitando um api gateway que baseado na requisição feita enviará a requisição para uma dos sistemas. Antes de redirecionar para um sistema específico é verificado se foi informado o token(JWT) de acesso, se for informado e for válido, envia a requisição para o sistema específico, caso não, notifica que o usuário não tem permissão para executar tal ação. O motivo de usar api gateway é que os clientes que viram consultar os dados não precisam saber os endereço de todos os sistemas, basta apenas requisitar o api gateway que ele será responsável por saber qual sistema enviar a requisição.

Quando a requisição é enviada para o sistema A irá passar por um load balancer que irá distribuir entre o grupo de máquinas existentes que são instâncias ec2 e também está utilizando autoscaling para que crie nova instâncias para suportar o número de requisições caso as máquinas não suportem a quantidade de requisições feitas. As instâncias consultada uma banco de dados mysql que foi configurado no RDS.

No sistema B possui o mesmo fluxo, mas nesse caso foi adicionando o redis que é um banco de dados em memory muito utilizado para cache, nesse caso uma instância ec2 verifica primeiro se existe a informação em cache, caso exista, retorna a informação, caso não, irá buscar no banco de dados e armazenar no redis é retorna o resultado. O redis uma tecnologia muito interessante, pois ele armazena as informações em memória o que faz ele responder muita mais rápido que um banco de dados que armazena os dados em disco.

Já as requisições que deve ser enviadas para o sistema C primeiramente no api gateway é verificar se as informações estão disponíveis no redis se estiverem já retorna os dados, caso não estejam disponíveis, é repassada a requisição para o sistema C que possui o mesmo fluxo do sistema B.

Aplicação de autenticação e autorização responsável pela parte de segurança. O motivo de existe essa aplicação é para evitar que cada sistema tenha que implementar autenticação e autorização é com isso facilitar para as aplicações clientes se comunicarem com os sistemas, se não fosse feito assim todas vez que fosse buscar informações de diferentes sistemas seria necessário autenticar novamente isso acabaria atrapalhando a experiência do usuário.

Regras de segurança:

- Deve ser implementar autenticação de dois fatores. Com isso quando o usuários for autenticar será enviado um código por email ou sms que ele deve informar esse código para gerar o token(JWT)
- Token(JWT) não deve ter informações sensíveis como: email e senha.
- Todas as requisições de autenticação e criação de um usuário deve usar o verbo POST do protocolo HTTPs
- Todas as aplicações que usam o protocolo HTTP deve sem habilitar o ssl para usarem HTTPS.
- Deve ser criado usuários do banco de dados com apenas as permissões necessárias para o sistemas que se conectam ao banco de dados.
- Quando as credenciais forem inválidas deve ser retornada a seguinte mensagem “Credenciais inválidas” a mensagem deve ser genérica.
- Habilitar rate limite para bloquear quando um endereço ip faz muitas requisições por um determinado tempo. Isso evitar força bruta e DDOS
- Quando um determinado ip fiz muitas requisição em um mesmo momento deve ser bloqueado o acesso desse ip.
- Credenciais e token de acesso não devem ser enviados como querystring em requisições
- Senha do usuário deve ser armazenada criptografada no banco de dados.
- Senha deve ter 10 caracteres pelo menos uma letra maiúscula, uma minúscula, um número e um caractere especial como: #%$
- A única aplicação de deve ser acessível por todos é o api gateway o restantes das aplicações deve ser especificado o ip no security group das aplicação que podem se conectar.
- Criar blacklist para os token quando for feito logout da aplicação. Assim irá evitar que um token seja usado depois do logout mesmo que o token ainda não tenha expirado

Observação: não foi definido, mas deve se ter ferramentas para monitorar as aplicações, ferramentas para centralizar os logs e como estou usando a arquitetura de microserviços o correto é usar kubernetes para gerenciar as aplicações. Como essas e tecnologias eu ainda não tenho conhecimento o suficiente então não acho correto definir uma arquitetura com tecnologias que não conheço.