(https://www.whizlabs.com/learn) > My Courses (https://www.whizlabs.com/learn/my-courses)
> AWS Certified Solutions Architect Associate (https://www.whizlabs.com/learn/course/aws-csaa-practice-tests#section-1)
> API Gateway - Quiz (https://www.whizlabs.com/learn/course/aws-csaa-practice-tests/quiz/14796)
> Report

API GATEWAY - QUIZ

Attempt	2
Marks Obtained	5 / 10
Your score is	50%
Completed on	Sunday , 03 February 2019 , 04:47 PM
Time Taken	00 H 09 M 40 S
Result	Fail

Domains / Topics wise Quiz Performance Report

S.No.	Topic	Total Questions	Correct	Incorrect	Unattempted
1	Other	9	4	5	0
2	Define Performant Architectures	1	1	0	0

10 Questions	**5** Correct	**5** Incorrect	**0** Unattempted

Show Answers

All ▼

QUESTION 1 INCORRECT

Which of the following are valid integration sources for API Gateway? (choose 3 options)

☑ **A.** Public facing HTTPS-based endpoints outside AWS network. ✔

☑ **B.** Lambda functions from another account. ✔

☑ **C.** Database connections on internet outside AWS network. ✖

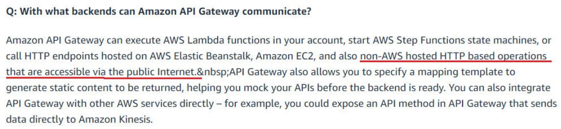
□ **D.** VPC Link ✔

□ **E.** SFTP connection

Explanation :

Answer: A, B, D

Option A is correct. AWS API Gateway can integrate with any HTTPS-based endpoints available over internet.



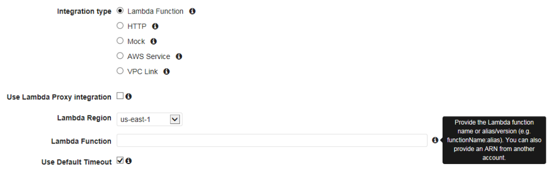
Q: With what backends can Amazon API Gateway communicate?

Amazon API Gateway can execute AWS Lambda functions in your account, start AWS Step Functions state machines, or call HTTP endpoints hosted on AWS Elastic Beanstalk, Amazon EC2, and also non-AWS hosted HTTP based operations that are accessible via the public Internet. API Gateway also allows you to specify a mapping template to generate static content to be returned, helping you mock your APIs before the backend is ready. You can also integrate API Gateway with other AWS services directly – for example, you could expose an API method in API Gateway that sends data directly to Amazon Kinesis.

Q: With what backends can Amazon API Gateway communicate?

Amazon API Gateway can execute AWS Lambda functions in your account, start AWS Step Functions state machines, or call HTTP endpoints hosted on AWS Elastic Beanstalk, Amazon EC2, and also **non-AWS hosted HTTP based operations that are accessible via the public Internet.** API Gateway also allows you to specify a mapping template to generate static content to be returned, helping you mock your APIs before the backend is ready. You can also integrate API Gateway with other AWS services directly – for example, you could expose an API method in API Gateway that sends data directly to Amazon Kinesis.

Option B is correct. AWS can use Lambda function from another account as an integration type.



Integration type ● Lambda Function

○ HTTP

○ Mock

○ AWS Service

○ VPC Link

Use Lambda Proxy integration □

Lambda Region us-east-1

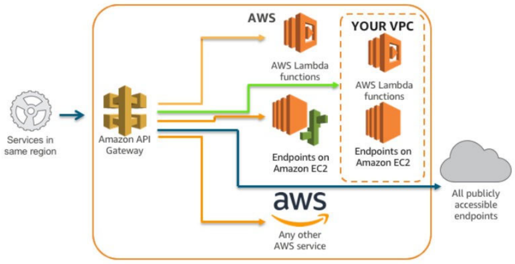
Lambda Function

Use Default Timeout ☑

Provide the Lambda function name or alias/version (e.g. functionName:alias). You can also provide an ARN from another account.

Option C is incorrect. AWS API gateway can connect to AWS services which will make proxy calls only to their respective AWS APIs. There is no integration type for database connections directly from API Gateway. You can use Lambda function to connect with database and make Lambda as integration type for API Gateway.

Option D is correct. AWS has introduced VPC Link, a way to connect to the resources within a private VPC.



Refer to the documentation here for more information on VPC Links. https://aws.amazon.com/blogs/compute/introducing-amazon-api-gateway-private-endpoints/ (https://aws.amazon.com/blogs/compute/introducing-amazon-api-gateway-private-endpoints/)

Ask our Experts

QUESTION 2 CORRECT

Company ABC is having few 100s of REST APIs exposed to internet from their on-premises network. They already have integrated with AWS through DirectConnect. They have approached you asking for a cost-effective way of making these REST APIs available through AWS API Gateway because of the resiliency and cost reductions provided by it. What solution would you provide?

○ **A.** API Gateway cannot integrate with on-premises backend APIs which are not over public internet. Rebuild all the backend APIs using Lambda and integrate with API Gateway.

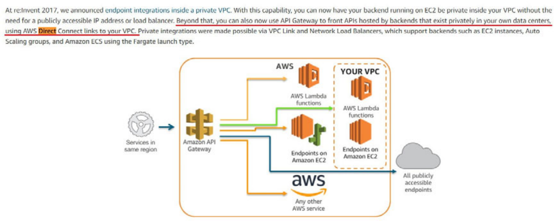
○ **B.** Use VPC Link to integrate on-premises backend solutions through DirectConnect and private VPC. ✔

○ **C.** Build API Gateway using existing on-premises public facing REST APIs as HTTPS endpoints integration type.

○ **D.** Build API Gateway with integration type as AWS Service and select Direct Connect service.

Explanation :

Answer: B



At re:Invent 2017, we announced endpoint integrations inside a private VPC. With this capability, you can now have your backend running on EC2 be private inside your VPC without the need for a publicly accessible IP address or load balancer. Beyond that, you can also now use API Gateway to front APIs hosted by backends that exist privately in your own data centers, using AWS Direct Connect links to your VPC. Private integrations were made possible via VPC Link and Network Load Balancers, which support backends such as EC2 instances, Auto Scaling groups, and Amazon ECS using the Fargate launch type.

For more information on VPC Link, refer documentation here.

- https://aws.amazon.com/blogs/compute/introducing-amazon-api-gateway-private-endpoints/ (https://aws.amazon.com/blogs/compute/introducing-amazon-api-gateway-private-endpoints/)

Ask our Experts

👍 👎

QUESTION 3 INCORRECT

You have built a REST API using API gateway and distributed to your customers. However, your API is receiving large number of requests and overloading your backend system causing performance bottlenecks and eventually causing delays and failures in serving the requests for your important customers. How would you improve the API performance? (Choose 2 options)

☐ **A.** Enable throttling and control the number of requests per second. ✓

☐ **B.** Create a resource policy to allow access for specific customers during specific time period.

☑ **C.** Enable API caching to serve frequently requested data from API cache. ✓

☑ **D.** Enable load balancer on your backend systems. ✗

Explanation :

Answer: A, C

Option A is correct. To prevent your API from being overwhelmed by too many requests, Amazon API Gateway throttles requests to your API. Specifically, API Gateway sets a limit on a steady-state rate and a burst of request submissions against all APIs in your account. For more information on throttling, refer documentation here.

- https://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-request-throttling.html (https://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-request-throttling.html)

Option B is not correct. This is not a viable solution. Resource policies cannot have a time range based condition.
Following documentation shows the conditions supported for API Gateway resource policies.

- https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-resource-policies-aws-condition-keys.html (https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-resource-policies-aws-condition-keys.html)

Option C is correct. You can enable API caching in Amazon API Gateway to cache your endpoint's responses. With caching, you can reduce the number of calls made to your endpoint and also improve the latency of requests to your API. When you enable caching for a stage, API Gateway caches responses from your endpoint for a specified time-to-live (TTL) period, in seconds. API Gateway then responds to the request by looking up the endpoint response from the cache instead of making a request to your endpoint. The default TTL value for API caching is 300 seconds. The maximum TTL value is 3600 seconds. TTL=0 means caching is disabled.
For details on enabling caching, refer documentation here.

- https://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-caching.html#enable-api-gateway-caching (https://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-caching.html#enable-api-gateway-caching)

Option D is not correct. We can improve performance by increasing the capacity of backend systems if above settings does not help. Simply adding a load balancer does not improve any performance.

Ask our Experts

👍 👎

QUESTION 4 INCORRECT

You have created a public facing REST API using AWS API Gateway with default throttle setting of 10000 requests per second and burst of 5000 requests. You are getting 8000 requests in one millisecond. Which of the following statements is true?

○ **A.** All 8000 requests would succeed as the default throttle limit is 8000 per second.

○ **B.** All 8000 requests would fail as it is higher than the burst limit of 5000.

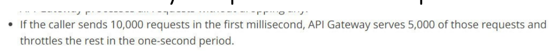
○ **C.** 5000 requests would succeed and rest 3000 would fail. ✖

○ **D.** 5000 requests would succeed and throttles the rest 3000 in one-second period. ✔

Explanation :

Answer: D

To prevent your API from being overwhelmed by too many requests, Amazon API Gateway throttles requests to your API using the token bucket algorithm, where a token counts for a request. Specifically, API Gateway sets a limit on a steady-state rate and a burst of request submissions against all APIs in your account. In the token bucket algorithm, the burst is the maximum bucket size.
When request submissions exceed the steady-state request rate and burst limits, API Gateway fails the limit-exceeding requests and returns 429 Too Many Requests error responses to the client. Upon catching such exceptions, the client can resubmit the failed requests in a rate- limiting fashion, while complying with the API Gateway throttling limits.
By default, API Gateway limits the steady-state request rate to 10,000 requests per second (rps). It limits the burst (that is, the maximum bucket size) to 5,000 requests across all APIs within an AWS account. In API Gateway, the burst limit corresponds to the maximum number of concurrent request submissions that API Gateway can fulfill at any moment without returning
429 Too Many Requests error responses.



API Gateway processes all requests without dropping any.

- If the caller sends 10,000 requests in the first millisecond, API Gateway serves 5,000 of those requests and throttles the rest in the one-second period.

For more information on API Gateway throttling, refer documentation here.

- https://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-request-throttling.html#apig-request-throttling-account-level-limits

(https://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-request-throttling.html#apig-request-throttling-account-level-limits)

Ask our Experts

👍 👎

QUESTION 5 CORRECT

Your organization had created a REST API using AWS API Gateway and exposed it over internet.

They have noticed a consistent high number of requests per second on GET **/users** method, approximately 9000 out of which 5000 requests are sent in 1st millisecond. This is putting more overload on backend systems. They have changed stage's number of requests per second to

6000 and burst to 3000 requests. Now the total number of requests sent per second is reduced to 6000, however, 5000 requests being sent in 1st millisecond. What could be causing this behaviour?

○ **A.** Stage's GET /users method throttling settings might have overwritten stage throttling settings with burst as 5000 requests. ✓

○ **B.** Account level throttle settings are 10000 requests per second and burst 5000 requests. You cannot overwrite account level settings.

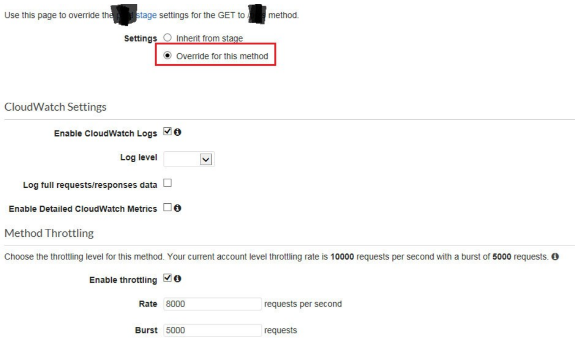
○ **C.** Any changes made to Stage might take upto 2 hours to propagate.

○ **D.** Requests per second are set to 6000. API can serve upto 6000 requests irrespective of how many requests sent in one millisecond.

Explanation :

Answer: A

You can override stage settings on an individual method within a stage.



Use this page to override the [redacted] stage settings for the GET to [redacted] method.

Settings ○ Inherit from stage
● Override for this method

CloudWatch Settings

Enable CloudWatch Logs ☑

Log level

Log full requests/responses data ☐

Enable Detailed CloudWatch Metrics ☐

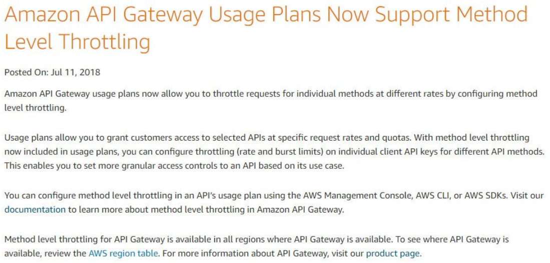
Method Throttling

Choose the throttling level for this method. Your current account level throttling rate is **10000** requests per second with a burst of **5000** requests.

Enable throttling ☑

Rate 8000 requests per second

Burst 5000 requests



Amazon API Gateway Usage Plans Now Support Method Level Throttling

Posted On: Jul 11, 2018

Amazon API Gateway usage plans now allow you to throttle requests for individual methods at different rates by configuring method level throttling.

Usage plans allow you to grant customers access to selected APIs at specific request rates and quotas. With method level throttling now included in usage plans, you can configure throttling (rate and burst limits) on individual client API keys for different API methods. This enables you to set more granular access controls to an API based on its use case.

You can configure method level throttling in an API's usage plan using the AWS Management Console, AWS CLI, or AWS SDKs. Visit our documentation to learn more about method level throttling in Amazon API Gateway.

Method level throttling for API Gateway is available in all regions where API Gateway is available. To see where API Gateway is available, review the AWS region table. For more information about API Gateway, visit our product page.

- https://aws.amazon.com/about-aws/whats-new/2018/07/api-gateway-usage-plans-support- method-level-throttling/ (https://aws.amazon.com/about-aws/whats-new/2018/07/api-gateway-usage-plans-support-method-level-throttling/)

Ask our Experts

QUESTION 6 INCORRECT

Which of the following are not access control mechanisms for AWS API Gateway? (Choose 2 options)

□ **A.** Resource policies

□ **B.** Lambda authorizers

□ **C.** Server-side certificates ✓

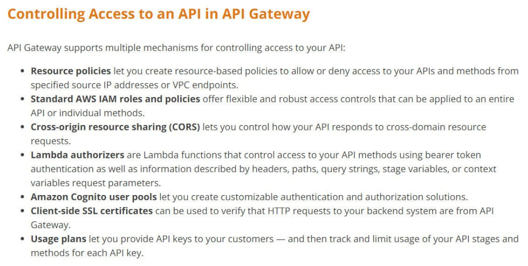
☑ **D.** VPC RouteTables ✓

☑ **E.** Usage Plans ✗

Explanation :

Answer: C, D

Following are different ways of controlling access to your AWS API Gateway.



Controlling Access to an API in API Gateway

API Gateway supports multiple mechanisms for controlling access to your API:

- **Resource policies** let you create resource-based policies to allow or deny access to your APIs and methods from specified source IP addresses or VPC endpoints.
- **Standard AWS IAM roles and policies** offer flexible and robust access controls that can be applied to an entire API or individual methods.
- **Cross-origin resource sharing (CORS)** lets you control how your API responds to cross-domain resource requests.
- **Lambda authorizers** are Lambda functions that control access to your API methods using bearer token authentication as well as information described by headers, paths, query strings, stage variables, or context variables request parameters.
- **Amazon Cognito user pools** let you create customizable authentication and authorization solutions.
- **Client-side SSL certificates** can be used to verify that HTTP requests to your backend system are from API Gateway.
- **Usage plans** let you provide API keys to your customers — and then track and limit usage of your API stages and methods for each API key.

- https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-control-access- to-api.html (https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-control-access-%20to-api.html)

Option C is not access control mechanism. API Gateway accepts the client-side certificates of your backend system.

- https://docs.aws.amazon.com/apigateway/latest/developerguide/getting-started-client-side- ssl-authentication.html (https://docs.aws.amazon.com/apigateway/latest/developerguide/getting-started-client-side-%20ssl-authentication.html)

Option D is not access control mechanism. RouteTables in VPCs are to control network traffic flow within a VPC.

For more information on VPC route tables, refer documentation here.

- https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Route_Tables.html (https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Route_Tables.html)

Ask our Experts

👍 👎

QUESTION 7 CORRECT

Your organization would need to expose certain services to its customer. You have created and deployed REST API for your organization using AWS API Gateway over public internet. However, you noticed requests from hosts other than your customer. How would you control access in this scenario?

(choose 2 options)

□ **A.** Establish DirectConnect to each of your customer's network and enable API Gateway's VPC Link through a private VPC.

☑ **B.** Enable CORS and add required host names under Access-Control-Allow-Origin. ✓

☑ **C.** Configure your customer's IP address ranges in resource policy. ✓

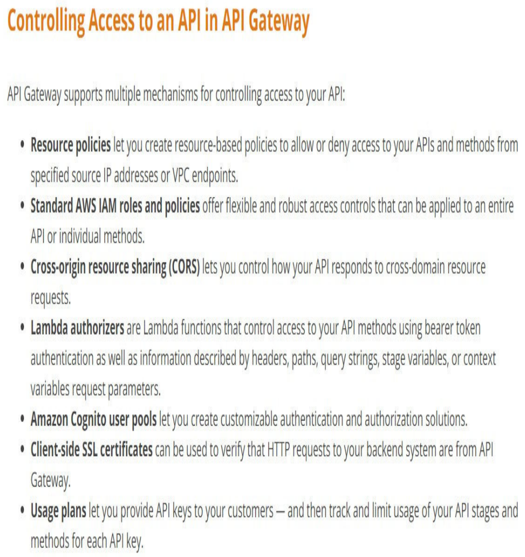
□ **D.** Create IAM users for your customers and enable

□ **E.** Generate and distribute Client Certificate to customer. Ask them to use the certificate while sending requests.

Explanation :

Answer: B, C

Following are different ways of controlling access to your AWS API Gateway.



Controlling Access to an API in API Gateway

API Gateway supports multiple mechanisms for controlling access to your API:

- **Resource policies** let you create resource-based policies to allow or deny access to your APIs and methods from specified source IP addresses or VPC endpoints.
- **Standard AWS IAM roles and policies** offer flexible and robust access controls that can be applied to an entire API or individual methods.
- **Cross-origin resource sharing (CORS)** lets you control how your API responds to cross-domain resource requests.
- **Lambda authorizers** are Lambda functions that control access to your API methods using bearer token authentication as well as information described by headers, paths, query strings, stage variables, or context variables request parameters.
- **Amazon Cognito user pools** let you create customizable authentication and authorization solutions.
- **Client-side SSL certificates** can be used to verify that HTTP requests to your backend system are from API Gateway.
- **Usage plans** let you provide API keys to your customers — and then track and limit usage of your API stages and methods for each API key.

- https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-control-access- to-api.html (https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-control-access-%20to-api.html)

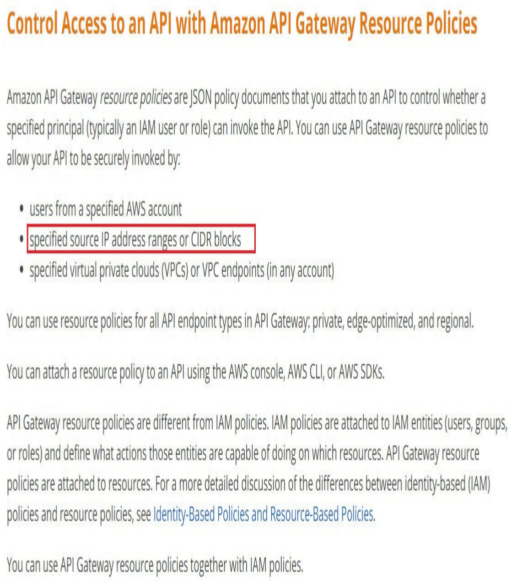
- Option A is not a viable solution.

- Option B is correct. You can allow a domain other than the API gateway's domain name to access the APIS using Cross Origin Resource Sharing.

- For more information on CORS, refer documentation here.

  - https://docs.aws.amazon.com/apigateway/latest/developerguide/how-to-cors.html (https://docs.aws.amazon.com/apigateway/latest/developerguide/how-to-cors.html)

- Option C is correct.



Control Access to an API with Amazon API Gateway Resource Policies

Amazon API Gateway *resource policies* are JSON policy documents that you attach to an API to control whether a specified principal (typically an IAM user or role) can invoke the API. You can use API Gateway resource policies to allow your API to be securely invoked by:

- users from a specified AWS account
- specified source IP address ranges or CIDR blocks
- specified virtual private clouds (VPCs) or VPC endpoints (in any account)

You can use resource policies for all API endpoint types in API Gateway: private, edge-optimized, and regional.

You can attach a resource policy to an API using the AWS console, AWS CLI, or AWS SDKs.

API Gateway resource policies are different from IAM policies. IAM policies are attached to IAM entities (users, groups, or roles) and define what actions those entities are capable of doing on which resources. API Gateway resource policies are attached to resources. For a more detailed discussion of the differences between identity-based (IAM) policies and resource policies, see Identity-Based Policies and Resource-Based Policies.

You can use API Gateway resource policies together with IAM policies.

- Option D in incorrect. Create IAM users for your customers and enable user authentication.

- Option E is incorrect. Just by creating an IAM Users and assigning to the Customers will not resolve the issue. Rather we need to create roles and policies.

- The client certificate is between API Gateway and the backend systems, not between API Gateway and the clients who make the requests.
- For more information on client certificates for API gateway, refer documentation here. ???
    - https://docs.aws.amazon.com/apigateway/latest/developerguide/getting-started-client-side-(https://docs.aws.amazon.com/apigateway/latest/developerguide/getting-started-client-side-%20ssl-authentication.html)ssl-authentication.html (https://docs.aws.amazon.com/apigateway/latest/developerguide/getting-started-client-side-%20ssl-authentication.html)

Ask our Experts

👍 👎

QUESTION 8 CORRECT

In AWS API Gateway, which of the following security measure is provided default by AWS to protect the backend systems?

○ **A.** Default Cross Origin Resource Sharing (CORS) onfiguration.

○ **B.** Default Resource Policy.

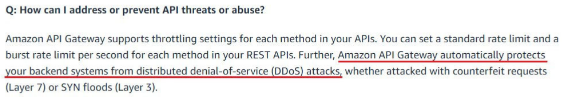
○ **C.** Protection from distributed denial-of-service (DDoS) attacks. ✓

○ **D.** Security of backend systems falls under customer responsibility. AWS provides different mechanisms to protect backend systems which are not configured by default.

Explanation :

Answer: C

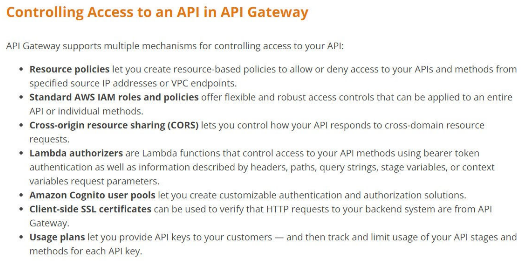
Following are the control mechanisms provided by AWS to control access. However, they are not configued by default by AWS. As a customer of AWS, you need to configure them.



Q: How can I address or prevent API threats or abuse?

Amazon API Gateway supports throttling settings for each method in your APIs. You can set a standard rate limit and a burst rate limit per second for each method in your REST APIs. Further, Amazon API Gateway automatically protects your backend systems from distributed denial-of-service (DDoS) attacks, whether attacked with counterfeit requests (Layer 7) or SYN floods (Layer 3).

Options A and B are part of above list and do not have any default configurations. Option C is correct.



Controlling Access to an API in API Gateway

API Gateway supports multiple mechanisms for controlling access to your API:

- **Resource policies** let you create resource-based policies to allow or deny access to your APIs and methods from specified source IP addresses or VPC endpoints.
- **Standard AWS IAM roles and policies** offer flexible and robust access controls that can be applied to an entire API or individual methods.
- **Cross-origin resource sharing (CORS)** lets you control how your API responds to cross-domain resource requests.
- **Lambda authorizers** are Lambda functions that control access to your API methods using bearer token authentication as well as information described by headers, paths, query strings, stage variables, or context variables request parameters.
- **Amazon Cognito user pools** let you create customizable authentication and authorization solutions.
- **Client-side SSL certificates** can be used to verify that HTTP requests to your backend system are from API Gateway.
- **Usage plans** let you provide API keys to your customers — and then track and limit usage of your API stages and methods for each API key.

Option D statement is not correct. Above screenshot shows AWS automatically provides protection from DdoS attacks.

Ask our Experts

QUESTION 9 CORRECT DEFINE PERFORMANT ARCHITECTURES

When enabling API caching for API Gateway through console, which of the following is not an action?

○ **A.** Cache capacity

○ **B.** Encrypt cache data

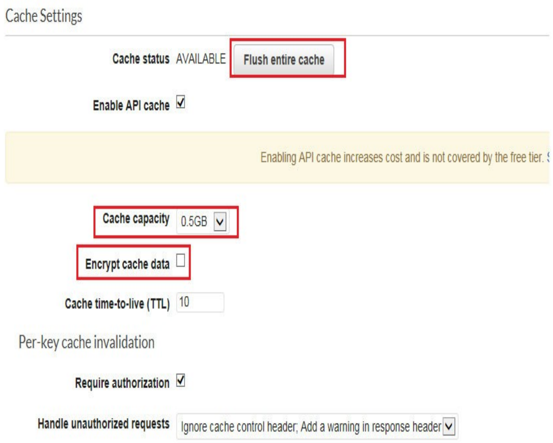
○ **C.** Refresh cache ✓

○ **D.** Flush entire cache

Explanation :

Answer: C

Following are the settings and actions when enabling/disabling API caching for API Gateway.



Cache Settings

Cache status AVAILABLE

Enable API cache ☑

Enabling API cache increases cost and is not covered by the free tier.

Cache capacity 0.5GB

Encrypt cache data □

Cache time-to-live (TTL) 10

Per-key cache invalidation

Require authorization ☑

Handle unauthorized requests Ignore cache control header; Add a warning in response header

- Options A, B, D are highlighted in above screen shots. There is no action to refresh cache on API Gateway.

For more information on API caching, refer documentation here.

- https://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-caching.html (https://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-caching.html)

Ask our Experts

QUESTION 10 INCORRECT

You have created a REST API using AWS API Gateway and deployed to production. Your organization requested the details on who is accessing the API for auditing purpose. How would you get the required information on who accessed your API?

○ **A.** Enable CloudWatch Logs.

○ **B.** Enable Access Logging ✔

○ **C.** CloudTrail contains the requester information for your API. ✖

○ **D.** Enable logging in your backend system to log the requests.

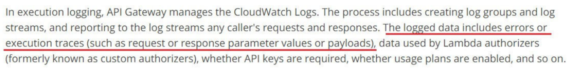
Explanation :

Answer: B

To help debug issues related to request execution or client access to your API, you can enable

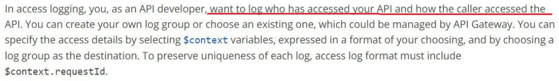
Amazon CloudWatch Logs to trace API calls.

Option A is not correct. Default CloudWatch logging enables API request logging which does not contain who access the API and how did they access.



In execution logging, API Gateway manages the CloudWatch Logs. The process includes creating log groups and log streams, and reporting to the log streams any caller's requests and responses. The logged data includes errors or execution traces (such as request or response parameter values or payloads), data used by Lambda authorizers (formerly known as custom authorizers), whether API keys are required, whether usage plans are enabled, and so on.

Option B is correct.



In access logging, you, as an API developer, want to log who has accessed your API and how the caller accessed the API. You can create your own log group or choose an existing one, which could be managed by API Gateway. You can specify the access details by selecting `$context` variables, expressed in a format of your choosing, and by choosing a log group as the destination. To preserve uniqueness of each log, access log format must include `$context.requestId`.

For more information on API gateway logging, refer documentation here.

- https://docs.aws.amazon.com/apigateway/latest/developerguide/set-up-logging.html (https://docs.aws.amazon.com/apigateway/latest/developerguide/set-up-logging.html)

Option C is not correct. CloudTrail logs the request information on AWS APIs, not the APIs generated through API gateway.
You can use AWS CloudTrail to capture API Gateway REST API calls in your AWS account and deliver the log files to an Amazon S3 bucket you specify. Examples of these API calls include creating a new API, resource, or method in API Gateway.
For more information on what information is captured in CloudTrail, refer documentation here.

- https://docs.aws.amazon.com/apigateway/latest/developerguide/cloudtrail.html (https://docs.aws.amazon.com/apigateway/latest/developerguide/cloudtrail.html)

Option D is not correct. It is not effective of logging access logs as we have an option provided by AWS.

Ask our Experts

👍 👎

Finish Review (https://www.whizlabs.com/learn/course/aws-csaa-practice-tests/quiz/14796)

Certification

- Cloud Certification (https://www.whizlabs.com/cloud-certification-training-courses/)
- Java Certification (https://www.whizlabs.com/oracle-java-certifications/)
- PM Certification (https://www.whizlabs.com/project-management-certifications/)

Company

- Support (https://help.whizlabs.com/hc/en-us)
- Discussions (http://ask.whizlabs.com/)
- Blog (https://www.whizlabs.com/blog/)

- ➔ Big Data Certification (https://www.whizlabs.com/big-data-certifications/)

Mobile App

 Android Coming Soon

 iOS Coming Soon

Follow us

f

(https://www.facebook.com/whizlabs.software/)

in

(https://in.linkedin.com/company/whizlabs-software)

(https://twitter.com/whizlabs?lang=en)

G+

(https://plus.google.com/+WhizlabsSoftware)