

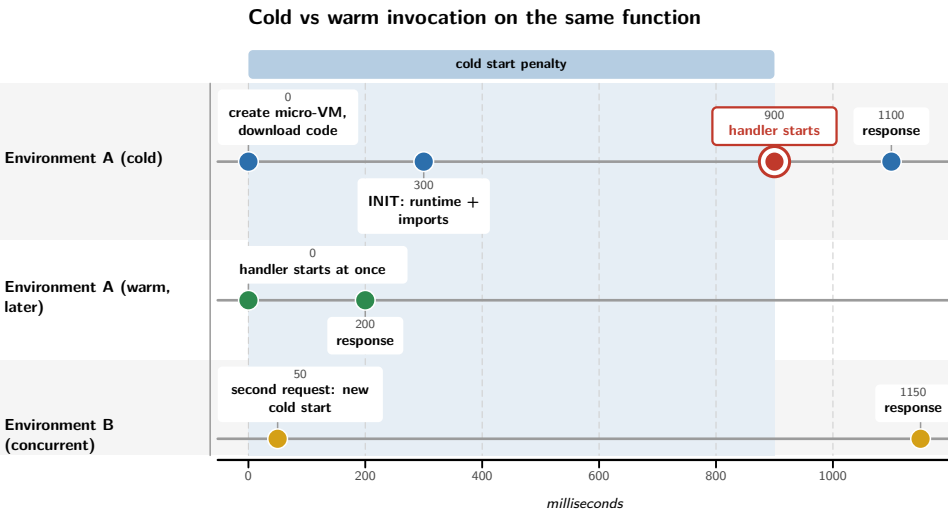
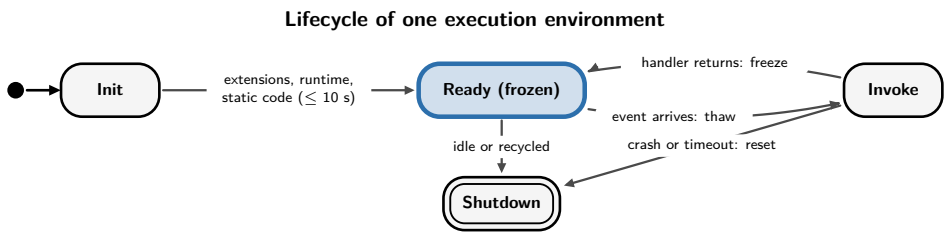


AWS Lambda in Depth: Execution, Invocation, Failure

AWS Lambda developer guide, as of 2026 | execution environment lifecycle, runtimes and container images, synchronous / asynchronous / poll-based invocation, event source mappings, destinations and DLQs, permissions, layers and extensions

Lambda Serverless Event source mapping Destinations

The execution environment



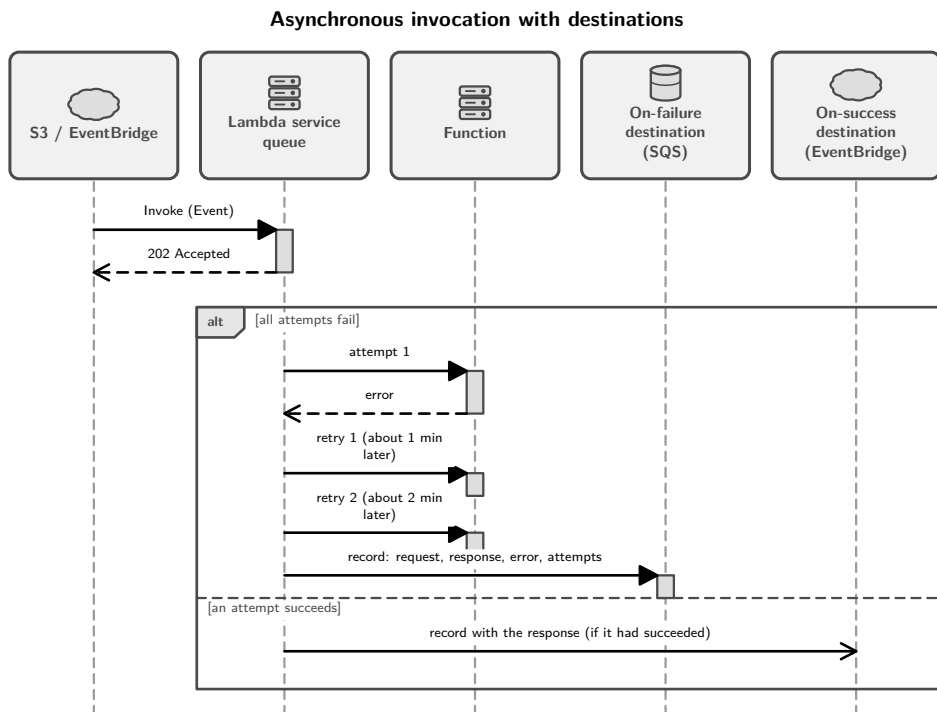
Packaging	Limit (as of 2026)	Notes
Zip, direct upload	50 MB zipped	larger zips via S3
Zip + layers, unzipped	250 MB	up to 5 layers, extracted to /opt
Container image	10 GB	from ECR, same account or cross-account with policy; AWS base images include the runtime interface client
Managed runtimes	Node.js, Python, Java, .NET, Ruby; provided.al2023 for Go, Rust, custom	deprecated runtimes: no patches, then blocked creates and updates
Architectures	x86_64 or arm64 (Graviton)	arm64 is cheaper per GB-second; native dependencies must match

What survives between invocations — Global variables, SDK clients, open connections and files in /tmp in the same environment; nothing is shared across environments, and any environment can disappear at any time.

Lambda Runtime Interface Emulator (RIE) — Local endpoint packaged in AWS base images to test a container-image function with curl before pushing it to ECR.

→ Three invocation models

	Synchronous	Asynchronous	Poll-based (event source mapping)
Callers	API Gateway, ALB, function URL, SDK Invoke, Cognito, Step Functions	S3, SNS, EventBridge, CloudWatch Logs, SES, SDK with InvocationType=Event	SQS, Kinesis, DynamoDB Streams, MSK / Kafka, Amazon MQ, DocumentDB
Who retries	the caller	Lambda: 0-2 retries with backoff, events kept up to 6 h	Lambda's poller, per source rules
Failure goes to	caller gets the error	on-failure destination or DLQ	SQS redrive DLQ; streams: on-failure destination
Payload	6 MB request / response	1 MB since late 2025 (was 256 KB)	batch up to 6 MB
Throttle behaviour	429 to the caller	Lambda queues and retries for up to 6 h	poller backs off; records wait in the source



🔗 Destinations beat DLQs

A DLQ receives only the failed event. An on-failure destination receives the event plus the error, stack and attempt count, and on-success destinations let you chain without code. For SQS and stream sources, use the source's redrive DLQ or the event source mapping's on-failure destination; async destinations do not apply to them.

📖 Event source mappings in detail

Setting	SQS	Kinesis / DynamoDB Streams
Batch size	1-10 000 (standard), 10 for FIFO; batch window up to 300 s	1-10 000; batch window up to 300 s
Scaling	pollers scale with queue depth; cap with maximum concurrency (2-1 000); provisioned mode for steady high throughput	one batch per shard at a time × parallelization factor (1-10)
Partial failure	ReportBatchItemFailures → return failed message IDs	ReportBatchItemFailures → checkpoint at the first failed sequence number
Retries	until maxReceiveCount on the queue, then its DLQ	until success or record expiry, unless MaximumRetryAttempts / MaximumRecordAgeInSeconds

Poison record	goes to the queue's DLQ	BisectBatchOnFunctionError + on-failure destination (SQS, SNS, S3)
Order	FIFO: per message group	per shard / partition key
Start position	-	TRIM_HORIZON, LATEST, AT_TIMESTAMP

How many concurrent invocations will a stream trigger use?

PROBLEM

A Kinesis stream has 8 shards. The event source mapping has parallelization factor 5 and batch size 100. Each batch takes 2 s. Producers write 2 000 records/s spread evenly. Can the function keep up, and how many concurrent invocations does it use?

1. Maximum concurrent invocations

⇒ $S \times F = 8 \times 5 = 40$

why: up to F batches per shard in parallel, order kept per partition key

2. Processing capacity

⇒ $40 \times B / t = 40 \times 100 / 2 = 2\,000$ records/s

why: each invocation handles one batch per 2 s

3. Compare with ingest

⇒ $2\,000 = 2\,000$ records/s

why: exactly at capacity: any slowdown builds iterator age

4. Add headroom

⇒ $F = 8 \rightarrow 64$ invocations → 3 200 records/s

why: or larger batches if the code is per-batch efficient

GIVEN

- shards $S = 8$, parallelization factor $F = 5$
- batch size $B = 100$ records, batch duration $t = 2$ s
- ingest rate $r = 2\,000$ records/s

ANSWER

At $F = 5$ it uses 40 concurrent invocations with no headroom; raise F to 8 (64 invocations) or the batch size.

Check: Watch the IteratorAge metric: growing iterator age means the consumer is falling behind.

ESM with filtering and partial failures (AWS CLI)

```
aws lambda create-event-source-mapping \
--function-name orders-worker \
--event-source-arn arn:aws:sqs:eu-west-3:1111:orders \
--batch-size 50 \
--maximum-batching-window-in-seconds 5 \
--function-response-types ReportBatchItemFailures \
--scaling-config MaximumConcurrency=20 \
--filter-criteria '{"Filters":[{"Pattern":
"{'body\":{\"type\":[\"PAID\"]}}"]}]'
```

🛡️ Permissions, layers, extensions

Execution role

what the function may do
trust policy: lambda.amazonaws.com
read the queue or stream for an ESM (sqs:Receive Message...)
AWSLambdaBasicExecutionRole for logs

Resource-based policy

who may invoke or manage the function
principal: service, account or role
lets S3, SNS, EventBridge, API Gateway push events
add SourceArn / SourceAccount conditions (confused deputy)

- **Layers** share libraries across functions; pinned by version ARN; not supported for container images (bake dependencies into the image instead)

- **External extensions** separate processes started in Init; can keep running after the response to flush telemetry (adds duration); examples: Parameters and Secrets extension, ADOT collector, vendor agents
- **Telemetry API** extensions subscribe to platform, function and extension logs and traces and forward them without CloudWatch
- **Response streaming** Node.js functions (and custom runtimes) stream the response through a function URL or API Gateway, beyond the buffered 6 MB response; check current size limits
- **Recent additions to verify** Lambda durable functions (checkpointed long-running workflows in code) and Lambda Managed Instances (functions on EC2 capacity you choose) were announced at re:Invent 2025: check regional availability and limits

YOU MAY HEAR

"A DLQ catches failed SQS-triggered invocations"

BUSTED

✓ ACTUALLY

Async DLQs do not apply to event source mappings
Configure the DLQ (redrive policy) on the SQS source queue itself.

YOU MAY HEAR

"Increasing the timeout lets Init run longer"

BUSTED

✓ ACTUALLY

Init is capped at 10 s for on-demand functions
If it overruns, Init is retried within the first invoke and counts against the function timeout; slim the init.

YOU MAY HEAR

"Two invocations can share /tmp"

BUSTED

✓ ACTUALLY

Only successive invocations in the same environment see it
Use S3, EFS or DynamoDB for shared state.

🚩 Recap and self-test

Tick a box only when you could explain the point to someone else



Init once, invoke many, freeze in between
keep init light, reuse clients



Know who retries
caller (sync), Lambda (async), poller (ESM)



Destinations over DLQs for async
error context and success chaining



Partial batch responses
ReportBatchItemFailures on SQS and streams



Concurrency for streams = shards × parallelization factor
watch IteratorAge

Q1 Maximum duration of the Init phase for an on-demand function?

A 10 seconds (then Init is retried in the invoke, under the function timeout).

Q2 How long can Lambda keep retrying an asynchronous event?

A Up to 6 hours (maximum event age), with 0-2 retries on function errors.

Q3 Which setting splits a failing Kinesis batch to find the poison record?

A BisectBatchOnFunctionError.

Q4 Maximum parallelization factor per shard?

A 10.

Q5 How do you stop paying for SQS messages the function ignores?

A Event filtering (filter criteria) on the event source mapping.

Q6 Which policy allows S3 to invoke a function?

A The function's resource-based policy (lambda add-permission with principal s3.amazonaws.com and SourceArn).

Q7 Can a container-image function use layers?

A No: include dependencies in the image.

Q8 Where does a layer's content appear in the environment?

A Under /opt (for example /opt/python on the Python path).

Q9 Cap how many concurrent invocations an SQS trigger may use?

A ScalingConfig.MaximumConcurrency on the event source mapping (2-1 000).

