

scheduler_preemption_attempts_total

Total preemption attempts in the cluster till now

- **Stability Level:** STABLE
- **Type:** Counter

scheduler_preemption_victims

Number of selected preemption victims

- **Stability Level:** STABLE
- **Type:** Histogram

scheduler_queue_incoming_pods_total

Number of pods added to scheduling queues by event and queue type.

- **Stability Level:** STABLE
- **Type:** Counter
- **Labels:** `event` `queue`

scheduler_schedule_attempts_total

Number of attempts to schedule pods, by the result. 'unschedulable' means a pod could not be scheduled, while 'error' means an internal scheduler problem.

- **Stability Level:** STABLE
- **Type:** Counter
- **Labels:** `profile` `result`

scheduler_scheduling_attempt_duration_seconds

Scheduling attempt latency in seconds (scheduling algorithm + binding)

- **Stability Level:** STABLE
- **Type:** Histogram
- **Labels:** `profile` `result`

List of Beta Kubernetes Metrics

Beta metrics observe a looser API contract than its stable counterparts. No labels can be removed from beta metrics during their lifetime, however, labels can be added while the metric is in the beta stage. This offers the assurance that beta metrics will honor existing dashboards and alerts, while allowing for amendments in the future.

apiserver_flowcontrol_current_executing_requests

Number of requests in initial (for a WATCH) or any (for a non-WATCH) execution stage in the API Priority and Fairness subsystem

- **Stability Level:** BETA

- **Type:** Gauge

- **Labels:** `flow_schema` `priority_level`

apiserver_flowcontrol_current_executing_seats

Concurrency (number of seats) occupied by the currently executing (initial stage for a WATCH, any stage otherwise) requests in the API Priority and Fairness subsystem

- **Stability Level:** BETA

- **Type:** Gauge

- **Labels:** `flow_schema` `priority_level`

apiserver_flowcontrol_current_inqueue_requests

Number of requests currently pending in queues of the API Priority and Fairness subsystem

- **Stability Level:** BETA

- **Type:** Gauge

- **Labels:** `flow_schema` `priority_level`

apiserver_flowcontrol_dispatched_requests_total

Number of requests executed by API Priority and Fairness subsystem

- **Stability Level:** BETA

- **Type:** Counter

- **Labels:** `flow_schema` `priority_level`

apiserver_flowcontrol_nominal_limit_seats

Nominal number of execution seats configured for each priority level

- **Stability Level:** BETA

- **Type:** Gauge

- **Labels:** `priority_level`

apiserver_flowcontrol_rejected_requests_total

Number of requests rejected by API Priority and Fairness subsystem

- **Stability Level:** BETA

- **Type:** Counter

- **Labels:** `flow_schema` `priority_level` `reason`

apiserver_flowcontrol_request_wait_duration_seconds

Length of time a request spent waiting in its queue

- **Stability Level:** BETA

- **Type:** Histogram

- **Labels:** `execute` `flow_schema` `priority_level`

disabled_metrics_total

The count of disabled metrics.

- **Stability Level:** BETA
- **Type:** Counter

hidden_metrics_total

The count of hidden metrics.

- **Stability Level:** BETA
- **Type:** Counter

kubernetes_feature_enabled

This metric records the data about the stage and enablement of a k8s feature.

- **Stability Level:** BETA
- **Type:** Gauge
- **Labels:** `name` `stage`

registered_metrics_total

The count of registered metrics broken by stability level and deprecation version.

- **Stability Level:** BETA
- **Type:** Counter
- **Labels:** `deprecated_version` `stability_level`

scheduler_pod_scheduling_sli_duration_seconds

E2e latency for a pod being scheduled, from the time the pod enters the scheduling queue and might involve multiple scheduling attempts.

- **Stability Level:** BETA
- **Type:** Histogram
- **Labels:** `attempts`

List of Alpha Kubernetes Metrics

Alpha metrics do not have any API guarantees. These metrics must be used at your own risk, subsequent versions of Kubernetes may remove these metrics altogether, or mutate the API in such a way that breaks existing dashboards and alerts.

aggregator_discovery_aggregation_count_total

Counter of number of times discovery was aggregated

- **Stability Level:** ALPHA

- **Type:** Counter

aggregator_openapi_v2_regeneration_count

Counter of OpenAPI v2 spec regeneration count broken down by causing APIService name and reason.

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `apiservice` `reason`

aggregator_openapi_v2_regeneration_duration

Gauge of OpenAPI v2 spec regeneration duration in seconds.

- **Stability Level:** ALPHA

- **Type:** Gauge

- **Labels:** `reason`

aggregator_unavailable_apiservice

Gauge of APIservices which are marked as unavailable broken down by APIService name.

- **Stability Level:** ALPHA

- **Type:** Custom

- **Labels:** `name`

aggregator_unavailable_apiservice_total

Counter of APIservices which are marked as unavailable broken down by APIService name and reason.

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `name` `reason`

apiextensions_apiserver_validation_ratcheting_seconds

Time for comparison of old to new for the purposes of CRDValidationRatcheting during an UPDATE in seconds.

- **Stability Level:** ALPHA

- **Type:** Histogram

apiextensions_openapi_v2_regeneration_count

Counter of OpenAPI v2 spec regeneration count broken down by causing CRD name and reason.

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `crd` `reason`

apiextensions_openapi_v3_regeneration_count

Counter of OpenAPI v3 spec regeneration count broken down by group, version, causing CRD and reason.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `crd` `group` `reason` `version`

apiserver_admission_match_condition_evaluation_errors_total

Admission match condition evaluation errors count, identified by name of resource containing the match condition and broken out for each kind containing matchConditions (webhook or policy), operation and admission type (validate or admit).

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `kind` `name` `operation` `type`

apiserver_admission_match_condition_evaluation_seconds

Admission match condition evaluation time in seconds, identified by name and broken out for each kind containing matchConditions (webhook or policy), operation and type (validate or admit).

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `kind` `name` `operation` `type`

apiserver_admission_match_condition_exclusions_total

Admission match condition evaluation exclusions count, identified by name of resource containing the match condition and broken out for each kind containing matchConditions (webhook or policy), operation and admission type (validate or admit).

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `kind` `name` `operation` `type`

apiserver_admission_step_admission_duration_seconds_summary

Admission sub-step latency summary in seconds, broken out for each operation and API resource and step type (validate or admit).

- **Stability Level:** ALPHA
- **Type:** Summary
- **Labels:** `operation` `rejected` `type`

apiserver_admission_webhook_fail_open_count

admission type (validating or mutating).

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `name` `type`

apiserver_admission_webhook_rejection_count

Admission webhook rejection count, identified by name and broken out for each admission type (validating or admit) and operation. Additional labels specify an error type (calling_webhook_error or apiserver_internal_error if an error occurred; no_error otherwise) and optionally a non-zero rejection code if the webhook rejects the request with an HTTP status code (honored by the apiserver when the code is greater or equal to 400). Codes greater than 600 are truncated to 600, to keep the metrics cardinality bounded.

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `error_type` `name` `operation` `rejection_code` `type`

apiserver_admission_webhook_request_total

Admission webhook request total, identified by name and broken out for each admission type (validating or mutating) and operation. Additional labels specify whether the request was rejected or not and an HTTP status code. Codes greater than 600 are truncated to 600, to keep the metrics cardinality bounded.

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `code` `name` `operation` `rejected` `type`

apiserver_audit_error_total

Counter of audit events that failed to be audited properly. Plugin identifies the plugin affected by the error.

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `plugin`

apiserver_audit_event_total

Counter of audit events generated and sent to the audit backend.

- **Stability Level:** ALPHA

- **Type:** Counter

apiserver_audit_level_total

Counter of policy levels for audit events (1 per request).

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `level`

apiserver_audit_requests_rejected_total

Counter of apiserver requests rejected due to an error in audit logging backend.

- **Stability Level:** ALPHA
- **Type:** Counter

apiserver_authentication_config_controller_automatic_reload_last_timestamp_seconds

Timestamp of the last automatic reload of authentication configuration split by status and apiserver identity.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `apiserver_id_hash` `status`

apiserver_authentication_config_controller_automatic_reloads_total

Total number of automatic reloads of authentication configuration split by status and apiserver identity.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `apiserver_id_hash` `status`

apiserver_authentication_jwt_authenticator_latency_seconds

Latency of jwt authentication operations in seconds. This is the time spent authenticating a token for cache miss only (i.e. when the token is not found in the cache).

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `jwt_issuer_hash` `result`

apiserver_authorization_config_controller_automatic_reload_last_timestamp_seconds

Timestamp of the last automatic reload of authorization configuration split by status and apiserver identity.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `apiserver_id_hash` `status`

apiserver_authorization_config_controller_automatic_reloads_total

Total number of automatic reloads of authorization configuration split by status and apiserver identity.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `apiserver_id_hash` `status`

apiserver_authorization_decisions_total

Total number of terminal decisions made by an authorizer split by authorizer type, name, and decision.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `decision` `name` `type`

apiserver_authorization_match_condition_evaluation_errors_total

Total number of errors when an authorization webhook encounters a match condition error split by authorizer type and name.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `name` `type`

apiserver_authorization_match_condition_evaluation_seconds

Authorization match condition evaluation time in seconds, split by authorizer type and name.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `name` `type`

apiserver_authorization_match_condition_exclusions_total

Total number of exclusions when an authorization webhook is skipped because match conditions exclude it.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `name` `type`

apiserver_authorization_webhook_duration_seconds

Request latency in seconds.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `name` `result`

apiserver_authorization_webhook_evaluations_fail_open_total

NoOpinion results due to webhook timeout or error.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `name` `result`

apiserver_authorization_webhook_evaluations_total

Round-trips to authorization webhooks.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `name` `result`

apiserver_cache_list_fetched_objects_total

Number of objects read from watch cache in the course of serving a LIST request

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `index` `resource_prefix`

apiserver_cache_list_returned_objects_total

Number of objects returned for a LIST request from watch cache

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `resource_prefix`

apiserver_cache_list_total

Number of LIST requests served from watch cache

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `index` `resource_prefix`

apiserver_cel_compilation_duration_seconds

CEL compilation time in seconds.

- **Stability Level:** ALPHA
- **Type:** Histogram

apiserver_cel_evaluation_duration_seconds

CEL evaluation time in seconds.

- **Stability Level:** ALPHA
- **Type:** Histogram

apiserver_certificates_registry_csr_honored_duration_total

Total number of issued CSRs with a requested duration that was honored, sliced by signer (only kubernetes.io signer names are specifically identified)

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `signerName`

apiserver_certificates_registry_csr_requested_duration_total

Total number of issued CSRs with a requested duration, sliced by signer (only kubernetes.io signer names are specifically identified)

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `signerName`

apiserver_client_certificate_expiration_seconds

Distribution of the remaining lifetime on the certificate used to authenticate a request.

- **Stability Level:** ALPHA
- **Type:** Histogram

apiserver_clusterip_repair_ip_errors_total

Number of errors detected on clusterips by the repair loop broken down by type of error: leak, repair, full, outOfRange, duplicate, unknown, invalid

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `type`

apiserver_clusterip_repair_reconcile_errors_total

Number of reconciliation failures on the clusterip repair reconcile loop

- **Stability Level:** ALPHA
- **Type:** Counter

apiserver_conversion_webhook_duration_seconds

Conversion webhook request latency

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `failure_type` `result`

apiserver_conversion_webhook_request_total

Counter for conversion webhook requests with success/failure and failure error type

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `failure_type` `result`

apiserver_crd_conversion_webhook_duration_seconds

CRD webhook conversion duration in seconds

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `crd_name` `from_version` `succeeded` `to_version`

apiserver_current_inqueue_requests

Maximal number of queued requests in this apiserver per request kind in last second.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `request_kind`

apiserver_delegated_authn_request_duration_seconds

Request latency in seconds. Broken down by status code.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `code`

apiserver_delegated_authn_request_total

Number of HTTP requests partitioned by status code.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `code`

apiserver_delegated_authz_request_duration_seconds

Request latency in seconds. Broken down by status code.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `code`

apiserver_delegated_authz_request_total

Number of HTTP requests partitioned by status code.

- **Stability Level:** ALPHA
- **Type:** Counter

- **Labels:** `code`

apiserver_egress_dialer_dial_duration_seconds

Dial latency histogram in seconds, labeled by the protocol (http-connect or grpc), transport (tcp or uds)

- **Stability Level:** ALPHA

- **Type:** Histogram

- **Labels:** `protocol` `transport`

apiserver_egress_dialer_dial_failure_count

Dial failure count, labeled by the protocol (http-connect or grpc), transport (tcp or uds), and stage (connect or proxy). The stage indicates at which stage the dial failed

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `protocol` `stage` `transport`

apiserver_egress_dialer_dial_start_total

Dial starts, labeled by the protocol (http-connect or grpc) and transport (tcp or uds).

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `protocol` `transport`

apiserver_encryption_config_controller_automatic_reload_failures_total

Total number of failed automatic reloads of encryption configuration split by apiserver identity.

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `apiserver_id_hash`

- **Deprecated Versions:** 1.30.0

apiserver_encryption_config_controller_automatic_reload_last_timestamp_seconds

Timestamp of the last successful or failed automatic reload of encryption configuration split by apiserver identity.

- **Stability Level:** ALPHA

- **Type:** Gauge

- **Labels:** `apiserver_id_hash` `status`

apiserver_encryption_config_controller_automatic_reload_success_total

Total number of successful automatic reloads of encryption configuration split by

apiserver identity.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `apiserver_id_hash`
- **Deprecated Versions:** 1.30.0

apiserver_encryption_config_controller_automatic_reloads_total

Total number of reload successes and failures of encryption configuration split by apiserver identity.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `apiserver_id_hash` `status`

apiserver_envelope_encryption_dek_cache_fill_percent

Percent of the cache slots currently occupied by cached DEKs.

- **Stability Level:** ALPHA
- **Type:** Gauge

apiserver_envelope_encryption_dek_cache_inter_arrival_time_seconds

Time (in seconds) of inter arrival of transformation requests.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `transformation_type`

apiserver_envelope_encryption_dek_source_cache_size

Number of records in data encryption key (DEK) source cache. On a restart, this value is an approximation of the number of decrypt RPC calls the server will make to the KMS plugin.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `provider_name`

apiserver_envelope_encryption_invalid_key_id_from_status_total

Number of times an invalid keyID is returned by the Status RPC call split by error.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `error` `provider_name`

apiserver_envelope_encryption_key_id_hash_last_timestamp_seconds

The last time in seconds when a keyID was used.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `apiserver_id_hash` `key_id_hash` `provider_name`
`transformation_type`

apiserver_envelope_encryption_key_id_hash_status_last_timestamp_seconds

The last time in seconds when a keyID was returned by the Status RPC call.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `apiserver_id_hash` `key_id_hash` `provider_name`

apiserver_envelope_encryption_key_id_hash_total

Number of times a keyID is used split by transformation type, provider, and apiserver identity.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `apiserver_id_hash` `key_id_hash` `provider_name`
`transformation_type`

apiserver_envelope_encryption_kms_operations_latency_seconds

KMS operation duration with gRPC error code status total.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `grpc_status_code` `method_name` `provider_name`

apiserver_flowcontrol_current_inqueue_seats

Number of seats currently pending in queues of the API Priority and Fairness subsystem

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `flow_schema` `priority_level`

apiserver_flowcontrol_current_limit_seats

current derived number of execution seats available to each priority level

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `priority_level`

apiserver_flowcontrol_current_r

R(time of last change)

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** priority_level

apiserver_flowcontrol_demand_seats

Observations, at the end of every nanosecond, of (the number of seats each priority level could use) / (nominal number of seats for that level)

- **Stability Level:** ALPHA
- **Type:** TimingRatioHistogram
- **Labels:** priority_level

apiserver_flowcontrol_demand_seats_average

Time-weighted average, over last adjustment period, of demand_seats

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** priority_level

apiserver_flowcontrol_demand_seats_high_watermark

High watermark, over last adjustment period, of demand_seats

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** priority_level

apiserver_flowcontrol_demand_seats_smoothed

Smoothed seat demands

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** priority_level

apiserver_flowcontrol_demand_seats_stdev

Time-weighted standard deviation, over last adjustment period, of demand_seats

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** priority_level

apiserver_flowcontrol_dispatch_r

R(time of last dispatch)

- **Type:** Gauge
- **Labels:** `priority_level`

apiserver_flowcontrol_epoch_advance_total

Number of times the queueset's progress meter jumped backward

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `priority_level` `success`

apiserver_flowcontrol_latest_s

S(most recently dispatched request)

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `priority_level`

apiserver_flowcontrol_lower_limit_seats

Configured lower bound on number of execution seats available to each priority level

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `priority_level`

apiserver_flowcontrol_next_discounted_s_bounds

min and max, over queues, of S(oldest waiting request in queue) - estimated work in progress

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `bound` `priority_level`

apiserver_flowcontrol_next_s_bounds

min and max, over queues, of S(oldest waiting request in queue)

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `bound` `priority_level`

apiserver_flowcontrol_priority_level_request_utilization

Observations, at the end of every nanosecond, of number of requests (as a fraction of the relevant limit) waiting or in any stage of execution (but only initial stage for WATCHes)

- **Type:** TimingRatioHistogram
- **Labels:** `phase` `priority_level`

apiserver_flowcontrol_priority_level_seat_utilization

Observations, at the end of every nanosecond, of utilization of seats for any stage of execution (but only initial stage for WATCHes)

- **Stability Level:** ALPHA
- **Type:** TimingRatioHistogram
- **Labels:** `priority_level`
- **Const Labels:** `phase:executing`

apiserver_flowcontrol_read_vs_write_current_requests

Observations, at the end of every nanosecond, of the number of requests (as a fraction of the relevant limit) waiting or in regular stage of execution

- **Stability Level:** ALPHA
- **Type:** TimingRatioHistogram
- **Labels:** `phase` `request_kind`

apiserver_flowcontrol_request_concurrency_in_use

Concurrency (number of seats) occupied by the currently executing (initial stage for a WATCH, any stage otherwise) requests in the API Priority and Fairness subsystem

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `flow_schema` `priority_level`
- **Deprecated Versions:** 1.31.0

apiserver_flowcontrol_request_concurrency_limit

Nominal number of execution seats configured for each priority level

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `priority_level`
- **Deprecated Versions:** 1.30.0

apiserver_flowcontrol_request_dispatch_no_accommodation_total

Number of times a dispatch attempt resulted in a non accommodation due to lack of available seats

- **Stability Level:** ALPHA
- **Type:** Counter

apiserver_flowcontrol_request_execution_seconds

Duration of initial stage (for a WATCH) or any (for a non-WATCH) stage of request execution in the API Priority and Fairness subsystem

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `flow_schema` `priority_level` `type`

apiserver_flowcontrol_request_queue_length_after_enqueue

Length of queue in the API Priority and Fairness subsystem, as seen by each request after it is enqueued

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `flow_schema` `priority_level`

apiserver_flowcontrol_seat_fair_frac

Fair fraction of server's concurrency to allocate to each priority level that can use it

- **Stability Level:** ALPHA
- **Type:** Gauge

apiserver_flowcontrol_target_seats

Seat allocation targets

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `priority_level`

apiserver_flowcontrol_upper_limit_seats

Configured upper bound on number of execution seats available to each priority level

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `priority_level`

apiserver_flowcontrol_watch_count_samples

count of watchers for mutating requests in API Priority and Fairness

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `flow_schema` `priority_level`

apiserver_flowcontrol_work_estimated_seats

Number of estimated seats (maximum of initial and final seats) associated with requests in API Priority and Fairness

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `flow_schema` `priority_level`

apiserver_init_events_total

Counter of init events processed in watch cache broken by resource type.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `resource`

apiserver_kube_aggregator_x509_insecure_sha1_total

Counts the number of requests to servers with insecure SHA1 signatures in their serving certificate OR the number of connection failures due to the insecure SHA1 signatures (either/or, based on the runtime environment)

- **Stability Level:** ALPHA
- **Type:** Counter

apiserver_kube_aggregator_x509_missing_san_total

Counts the number of requests to servers missing SAN extension in their serving certificate OR the number of connection failures due to the lack of x509 certificate SAN extension missing (either/or, based on the runtime environment)

- **Stability Level:** ALPHA
- **Type:** Counter

apiserver_nodeport_repair_port_errors_total

Number of errors detected on ports by the repair loop broken down by type of error: leak, repair, full, outOfRange, duplicate, unknown

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `type`

apiserver_nodeport_repair_reconcile_errors_total

Number of reconciliation failures on the nodeport repair reconcile loop

- **Stability Level:** ALPHA
- **Type:** Counter

apiserver_request_aborts_total

Number of requests which apiserver aborted possibly due to a timeout, for each group, version, verb, resource, subresource and scope

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `group` `resource` `scope` `subresource` `verb` `version`

apiserver_request_body_size_bytes

Apiserver request body size in bytes broken out by resource and verb.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `resource` `verb`

apiserver_request_filter_duration_seconds

Request filter latency distribution in seconds, for each filter type

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `filter`

apiserver_request_post_timeout_total

Tracks the activity of the request handlers after the associated requests have been timed out by the apiserver

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `source` `status`

apiserver_request_sli_duration_seconds

Response latency distribution (not counting webhook duration and priority & fairness queue wait times) in seconds for each verb, group, version, resource, subresource, scope and component.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `component` `group` `resource` `scope` `subresource` `verb`
`version`

apiserver_request_slo_duration_seconds

Response latency distribution (not counting webhook duration and priority & fairness queue wait times) in seconds for each verb, group, version, resource, subresource, scope and component.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `component` `group` `resource` `scope` `subresource` `verb`
`version`

- **Deprecated Versions:** 1.27.0

apiserver_request_terminations_total

Number of requests which apiserver terminated in self-defense.

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `code` `component` `group` `resource` `scope` `subresource`
`verb` `version`

apiserver_request_timestamp_comparison_time

Time taken for comparison of old vs new objects in UPDATE or PATCH requests

- **Stability Level:** ALPHA

- **Type:** Histogram

- **Labels:** `code_path`

apiserver_rerouted_request_total

Total number of requests that were proxied to a peer kube apiserver because the local apiserver was not capable of serving it

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `code`

apiserver_selfrequest_total

Counter of apiserver self-requests broken out for each verb, API resource and subresource.

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `resource` `subresource` `verb`

apiserver_storage_data_key_generation_duration_seconds

Latencies in seconds of data encryption key(DEK) generation operations.

- **Stability Level:** ALPHA

- **Type:** Histogram

apiserver_storage_data_key_generation_failures_total

Total number of failed data encryption key(DEK) generation operations.

- **Stability Level:** ALPHA

- **Type:** Counter

apiserver_storage_db_total_size_in_bytes

Total size of the storage database file physically allocated in bytes.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** endpoint
- **Deprecated Versions:** 1.28.0

apiserver_storage_decode_errors_total

Number of stored object decode errors split by object type

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** resource

apiserver_storage_envelope_transformation_cache_misses_total

Total number of cache misses while accessing key decryption key(KEK).

- **Stability Level:** ALPHA
- **Type:** Counter

apiserver_storage_events_received_total

Number of etcd events received split by kind.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** resource

apiserver_storage_list_evaluated_objects_total

Number of objects tested in the course of serving a LIST request from storage

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** resource

apiserver_storage_list_fetched_objects_total

Number of objects read from storage in the course of serving a LIST request

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** resource

apiserver_storage_list_returned_objects_total

Number of objects returned for a LIST request from storage

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `resource`

apiserver_storage_list_total

Number of LIST requests served from storage

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `resource`

apiserver_storage_transformation_duration_seconds

Latencies in seconds of value transformation operations.

- **Stability Level:** ALPHA

- **Type:** Histogram

- **Labels:** `transformation_type` `transformer_prefix`

apiserver_storage_transformation_operations_total

Total number of transformations. Successful transformation will have a status 'OK' and a varied status string when the transformation fails. This status and transformation_type fields may be used for alerting on encryption/decryption failure using transformation_type from_storage for decryption and to_storage for encryption

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `status` `transformation_type` `transformer_prefix`

apiserver_stream_translator_requests_total

Total number of requests that were handled by the StreamTranslatorProxy, which processes streaming RemoteCommand/V5

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `code`

apiserver_terminated_watchers_total

Counter of watchers closed due to unresponsiveness broken by resource type.

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `resource`

apiserver_tls_handshake_errors_total

Number of requests dropped with TLS handshake error from client.

- **Stability Level:** ALPHA
- **Type:** Counter

apiserver_validating_admission_policy_check_duration_seconds

Validation admission latency for individual validation expressions in seconds, labeled by policy and further including binding, state and enforcement action taken.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `enforcement_action` `policy` `policy_binding` `state`

apiserver_validating_admission_policy_check_total

Validation admission policy check total, labeled by policy and further identified by binding, enforcement action taken, and state.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `enforcement_action` `policy` `policy_binding` `state`

apiserver_validating_admission_policy_definition_total

Validation admission policy count total, labeled by state and enforcement action.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `enforcement_action` `state`

apiserver_watch_cache_events_dispatched_total

Counter of events dispatched in watch cache broken by resource type.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `resource`

apiserver_watch_cache_events_received_total

Counter of events received in watch cache broken by resource type.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `resource`

apiserver_watch_cache_initializations_total

Counter of watch cache initializations broken by resource type.

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `resource`

apiserver_watch_cache_read_wait_seconds

Histogram of time spent waiting for a watch cache to become fresh.

- **Stability Level:** ALPHA

- **Type:** Histogram

- **Labels:** `resource`

apiserver_watch_events_sizes

Watch event size distribution in bytes

- **Stability Level:** ALPHA

- **Type:** Histogram

- **Labels:** `group` `kind` `version`

apiserver_watch_events_total

Number of events sent in watch clients

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `group` `kind` `version`

apiserver_watch_list_duration_seconds

Response latency distribution in seconds for watch list requests broken by group, version, resource and scope.

- **Stability Level:** ALPHA

- **Type:** Histogram

- **Labels:** `group` `resource` `scope` `version`

apiserver_webhooks_x509_insecure_sha1_total

Counts the number of requests to servers with insecure SHA1 signatures in their serving certificate OR the number of connection failures due to the insecure SHA1 signatures (either/or, based on the runtime environment)

- **Stability Level:** ALPHA

- **Type:** Counter

apiserver_webhooks_x509_missing_san_total

Counts the number of requests to servers missing SAN extension in their serving certificate OR the number of connection failures due to the lack of x509 certificate SAN extension missing (either/or, based on the runtime environment)

- **Stability Level:** ALPHA

attach_detach_controller_attachdetach_controller_forced_detaches

Number of times the A/D Controller performed a forced detach

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `reason`

attachdetach_controller_total_volumes

Number of volumes in A/D Controller

- **Stability Level:** ALPHA
- **Type:** Custom
- **Labels:** `plugin_name` `state`

authenticated_user_requests

Counter of authenticated requests broken out by username.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `username`

authentication_attempts

Counter of authenticated attempts.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `result`

authentication_duration_seconds

Authentication duration in seconds broken out by result.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `result`

authentication_token_cache_active_fetch_count

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `status`

authentication_token_cache_fetch_total

- **Type:** Counter

- **Labels:** `status`

authentication_token_cache_request_duration_seconds

- **Stability Level:** ALPHA

- **Type:** Histogram

- **Labels:** `status`

authentication_token_cache_request_total

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `status`

authorization_attempts_total

Counter of authorization attempts broken down by result. It can be either 'allowed', 'denied', 'no-opinion' or 'error'.

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `result`

authorization_duration_seconds

Authorization duration in seconds broken out by result.

- **Stability Level:** ALPHA

- **Type:** Histogram

- **Labels:** `result`

cloud_provider_webhook_request_duration_seconds

Request latency in seconds. Broken down by status code.

- **Stability Level:** ALPHA

- **Type:** Histogram

- **Labels:** `code` `webhook`

cloud_provider_webhook_request_total

Number of HTTP requests partitioned by status code.

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `code` `webhook`

cloudprovider_gce_api_request_duration_seconds

Latency of a GCE API call

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `region` `request` `version` `zone`

cloudprovider_gce_api_request_errors

Number of errors for an API call

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `region` `request` `version` `zone`

container_swap_usage_bytes

Current amount of the container swap usage in bytes. Reported only on non-windows systems

- **Stability Level:** ALPHA
- **Type:** Custom
- **Labels:** `container` `pod` `namespace`

csi_operations_seconds

Container Storage Interface operation duration with gRPC error code status total

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `driver_name` `grpc_status_code` `method_name` `migrated`

endpoint_slice_controller_changes

Number of EndpointSlice changes

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `operation`

endpoint_slice_controller_desired_endpoint_slices

Number of EndpointSlices that would exist with perfect endpoint allocation

- **Stability Level:** ALPHA
- **Type:** Gauge

endpoint_slice_controller_endpoints_added_per_sync

Number of endpoints added on each Service sync

- **Stability Level:** ALPHA

- **Type:** Histogram

endpoint_slice_controller_endpoints_desired

Number of endpoints desired

- **Stability Level:** ALPHA

- **Type:** Gauge

endpoint_slice_controller_endpoints_removed_per_sync

Number of endpoints removed on each Service sync

- **Stability Level:** ALPHA

- **Type:** Histogram

endpoint_slice_controller_endpointslices_changed_per_sync

Number of EndpointSlices changed on each Service sync

- **Stability Level:** ALPHA

- **Type:** Histogram

- **Labels:** `topology` `traffic_distribution`

endpoint_slice_controller_num_endpoint_slices

Number of EndpointSlices

- **Stability Level:** ALPHA

- **Type:** Gauge

endpoint_slice_controller_services_count_by_traffic_distribution

Number of Services using some specific trafficDistribution

- **Stability Level:** ALPHA

- **Type:** Gauge

- **Labels:** `traffic_distribution`

endpoint_slice_controller_syncs

Number of EndpointSlice syncs

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `result`

endpoint_slice_mirroring_controller_addresses_skipped_per_sync

Number of addresses skipped on each Endpoints sync due to being invalid or exceeding MaxEndpointsPerSubset

- **Stability Level:** ALPHA

- **Type:** Histogram

endpoint_slice_mirroring_controller_changes

Number of EndpointSlice changes

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `operation`

endpoint_slice_mirroring_controller_desired_endpoint_slices

Number of EndpointSlices that would exist with perfect endpoint allocation

- **Stability Level:** ALPHA

- **Type:** Gauge

endpoint_slice_mirroring_controller_endpoints_added_per_sync

Number of endpoints added on each Endpoints sync

- **Stability Level:** ALPHA

- **Type:** Histogram

endpoint_slice_mirroring_controller_endpoints_desired

Number of endpoints desired

- **Stability Level:** ALPHA

- **Type:** Gauge

endpoint_slice_mirroring_controller_endpoints_removed_per_sync

Number of endpoints removed on each Endpoints sync

- **Stability Level:** ALPHA

- **Type:** Histogram

endpoint_slice_mirroring_controller_endpoints_sync_duration

Duration of syncEndpoints() in seconds

- **Stability Level:** ALPHA

- **Type:** Histogram

endpoint_slice_mirroring_controller_endpoints_updated_per_sync

Number of endpoints updated on each Endpoints sync

- **Stability Level:** ALPHA

- **Type:** Histogram

endpoint_slice_mirroring_controller_num_endpoint_slices

Number of EndpointSlices

- **Stability Level:** ALPHA
- **Type:** Gauge

ephemeral_volume_controller_create_failures_total

Number of PersistenVolumeClaims creation requests

- **Stability Level:** ALPHA
- **Type:** Counter

ephemeral_volume_controller_create_total

Number of PersistenVolumeClaims creation requests

- **Stability Level:** ALPHA
- **Type:** Counter

etcd_bookmark_counts

Number of etcd bookmarks (progress notify events) split by kind.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `resource`

etcd_lease_object_counts

Number of objects attached to a single etcd lease.

- **Stability Level:** ALPHA
- **Type:** Histogram

etcd_request_duration_seconds

Etcd request latency in seconds for each operation and object type.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `operation` `type`

etcd_request_errors_total

Etcd failed request counts for each operation and object type.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `operation` `type`

etcd_requests_total

Etcd request counts for each operation and object type.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `operation` `type`

etcd_version_info

Etcd server's binary version

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `binary_version`

field_validation_request_duration_seconds

Response latency distribution in seconds for each field validation value

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `field_validation`

force_cleaned_failed_volume_operation_errors_total

The number of volumes that failed force cleanup after their reconstruction failed during kubelet startup.

- **Stability Level:** ALPHA
- **Type:** Counter

force_cleaned_failed_volume_operations_total

The number of volumes that were force cleaned after their reconstruction failed during kubelet startup. This includes both successful and failed cleanups.

- **Stability Level:** ALPHA
- **Type:** Counter

garbagecollector_controller_resources_sync_error_total

Number of garbage collector resources sync errors

- **Stability Level:** ALPHA
- **Type:** Counter

get_token_count

Counter of total Token() requests to the alternate token source

- **Stability Level:** ALPHA
- **Type:** Counter

get_token_fail_count

Counter of failed Token() requests to the alternate token source

- **Stability Level:** ALPHA
- **Type:** Counter

horizontal_pod_autoscaler_controller_metric_computation_duration_seconds

The time(seconds) that the HPA controller takes to calculate one metric. The label 'action' should be either 'scale_down', 'scale_up', or 'none'. The label 'error' should be either 'spec', 'internal', or 'none'. The label 'metric_type' corresponds to HPA.spec.metrics[*].type

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `action` `error` `metric_type`

horizontal_pod_autoscaler_controller_metric_computation_total

Number of metric computations. The label 'action' should be either 'scale_down', 'scale_up', or 'none'. Also, the label 'error' should be either 'spec', 'internal', or 'none'. The label 'metric_type' corresponds to HPA.spec.metrics[*].type

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `action` `error` `metric_type`

horizontal_pod_autoscaler_controller_reconciliation_duration_seconds

The time(seconds) that the HPA controller takes to reconcile once. The label 'action' should be either 'scale_down', 'scale_up', or 'none'. Also, the label 'error' should be either 'spec', 'internal', or 'none'. Note that if both spec and internal errors happen during a reconciliation, the first one to occur is reported in `error` label.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `action` `error`

horizontal_pod_autoscaler_controller_reconciliations_total

Number of reconciliations of HPA controller. The label 'action' should be either 'scale_down', 'scale_up', or 'none'. Also, the label 'error' should be either 'spec', 'internal', or 'none'. Note that if both spec and internal errors happen during a reconciliation, the first one to occur is reported in `error` label.

- **Type:** Counter
- **Labels:** `action` `error`

job_controller_job_finished_indexes_total

The number of finished indexes. Possible values for the, status label are: "succeeded", "failed". Possible values for the, backoffLimit label are: "perIndex" and "global"

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `backoffLimit` `status`

job_controller_job_pods_creation_total

The number of Pods created by the Job controller labelled with a reason for the Pod creation., This metric also distinguishes between Pods created using different PodReplacementPolicy settings., Possible values of the "reason" label are:, "new", "recreate_terminating_or_failed", "recreate_failed",., Possible values of the "status" label are:, "succeeded", "failed".

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `reason` `status`

job_controller_jobs_by_external_controller_total

The number of Jobs managed by an external controller

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `controller_name`

job_controller_pod_failures_handled_by_failure_policy_total

The number of failed Pods handled by failure policy with, respect to the failure policy action applied based on the matched, rule. Possible values of the action label correspond to the, possible values for the failure policy rule action, which are:, "FailJob", "Ignore" and "Count".

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `action`

job_controller_terminated_pods_tracking_finalizer_total

The number of terminated pods (phase=Failed | Succeeded), that have the finalizer batch.kubernetes.io/job-tracking, The event label can be "add" or "delete".

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** event

kube_apiserver_clusterip_allocator_allocated_ips

Gauge measuring the number of allocated IPs for Services

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** cidr

kube_apiserver_clusterip_allocator_allocation_errors_total

Number of errors trying to allocate Cluster IPs

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** cidr scope

kube_apiserver_clusterip_allocator_allocation_total

Number of Cluster IPs allocations

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** cidr scope

kube_apiserver_clusterip_allocator_available_ips

Gauge measuring the number of available IPs for Services

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** cidr

kube_apiserver_nodeport_allocator_allocated_ports

Gauge measuring the number of allocated NodePorts for Services

- **Stability Level:** ALPHA
- **Type:** Gauge

kube_apiserver_nodeport_allocator_allocation_errors_total

Number of errors trying to allocate NodePort

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** scope

kube_apiserver_nodeport_allocator_allocation_total

Number of NodePort allocations

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** scope

kube_apiserver_nodeport_allocator_available_ports

Gauge measuring the number of available NodePorts for Services

- **Stability Level:** ALPHA

- **Type:** Gauge

kube_apiserver_pod_logs_backend_tls_failure_total

Total number of requests for pods/logs that failed due to kubelet server TLS verification

- **Stability Level:** ALPHA

- **Type:** Counter

kube_apiserver_pod_logs_insecure_backend_total

Total number of requests for pods/logs sliced by usage type: enforce_tls, skip_tls_allowed, skip_tls_denied

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** usage

kube_apiserver_pod_logs_pods_logs_backend_tls_failure_total

Total number of requests for pods/logs that failed due to kubelet server TLS verification

- **Stability Level:** ALPHA

- **Type:** Counter

- **Deprecated Versions:** 1.27.0

kube_apiserver_pod_logs_pods_logs_insecure_backend_total

Total number of requests for pods/logs sliced by usage type: enforce_tls, skip_tls_allowed, skip_tls_denied

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** usage

- **Deprecated Versions:** 1.27.0

kubelet_active_pods

The number of pods the kubelet considers active and which are being considered when admitting new pods. static is true if the pod is not from the apiserver.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** static

kubelet_certificate_manager_client_expiration_renew_errors

Counter of certificate renewal errors.

- **Stability Level:** ALPHA
- **Type:** Counter

kubelet_certificate_manager_client_ttl_seconds

Gauge of the TTL (time-to-live) of the Kubelet's client certificate. The value is in seconds until certificate expiry (negative if already expired). If client certificate is invalid or unused, the value will be +INF.

- **Stability Level:** ALPHA
- **Type:** Gauge

kubelet_certificate_manager_server_rotation_seconds

Histogram of the number of seconds the previous certificate lived before being rotated.

- **Stability Level:** ALPHA
- **Type:** Histogram

kubelet_certificate_manager_server_ttl_seconds

Gauge of the shortest TTL (time-to-live) of the Kubelet's serving certificate. The value is in seconds until certificate expiry (negative if already expired). If serving certificate is invalid or unused, the value will be +INF.

- **Stability Level:** ALPHA
- **Type:** Gauge

kubelet_cgroup_manager_duration_seconds

Duration in seconds for cgroup manager operations. Broken down by method.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** operation_type

kubelet_container_log_filesystem_used_bytes

Bytes used by the container's logs on the filesystem.

- **Stability Level:** ALPHA
- **Type:** Custom
- **Labels:** `uid` `namespace` `pod` `container`

kubelet_containers_per_pod_count

The number of containers per pod.

- **Stability Level:** ALPHA
- **Type:** Histogram

kubelet_cpu_manager_pinning_errors_total

The number of cpu core allocations which required pinning failed.

- **Stability Level:** ALPHA
- **Type:** Counter

kubelet_cpu_manager_pinning_requests_total

The number of cpu core allocations which required pinning.

- **Stability Level:** ALPHA
- **Type:** Counter

kubelet_credential_provider_plugin_duration

Duration of execution in seconds for credential provider plugin

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `plugin_name`

kubelet_credential_provider_plugin_errors

Number of errors from credential provider plugin

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `plugin_name`

kubelet_desired_pods

The number of pods the kubelet is being instructed to run. static is true if the pod is not from the apiserver.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `static`

kubelet_device_plugin_alloc_duration_seconds

Duration in seconds to serve a device plugin Allocation request. Broken down by resource name.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `resource_name`

kubelet_device_plugin_registration_total

Cumulative number of device plugin registrations. Broken down by resource name.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `resource_name`

kubelet_evented_pleg_connection_error_count

The number of errors encountered during the establishment of streaming connection with the CRI runtime.

- **Stability Level:** ALPHA
- **Type:** Counter

kubelet_evented_pleg_connection_latency_seconds

The latency of streaming connection with the CRI runtime, measured in seconds.

- **Stability Level:** ALPHA
- **Type:** Histogram

kubelet_evented_pleg_connection_success_count

The number of times a streaming client was obtained to receive CRI Events.

- **Stability Level:** ALPHA
- **Type:** Counter

kubelet_eviction_stats_age_seconds

Time between when stats are collected, and when pod is evicted based on those stats by eviction signal

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `eviction_signal`

kubelet_evictions

Cumulative number of pod evictions by eviction signal

- **Stability Level:** ALPHA

- **Type:** Counter
- **Labels:** `eviction_signal`

kubelet_graceful_shutdown_end_time_seconds

Last graceful shutdown start time since unix epoch in seconds

- **Stability Level:** ALPHA
- **Type:** Gauge

kubelet_graceful_shutdown_start_time_seconds

Last graceful shutdown start time since unix epoch in seconds

- **Stability Level:** ALPHA
- **Type:** Gauge

kubelet_http_inflight_requests

Number of the inflight http requests

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `long_running` `method` `path` `server_type`

kubelet_http_requests_duration_seconds

Duration in seconds to serve http requests

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `long_running` `method` `path` `server_type`

kubelet_http_requests_total

Number of the http requests received since the server started

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `long_running` `method` `path` `server_type`

kubelet_image_garbage_collected_total

Total number of images garbage collected by the kubelet, whether through disk usage or image age.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `reason`

kubelet_image_pull_duration_seconds

Duration in seconds to pull an image.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `image_size_in_bytes`

kubelet_lifecycle_handler_http_fallbacks_total

The number of times lifecycle handlers successfully fell back to http from https.

- **Stability Level:** ALPHA
- **Type:** Counter

kubelet_managed_ephemeral_containers

Current number of ephemeral containers in pods managed by this kubelet.

- **Stability Level:** ALPHA
- **Type:** Gauge

kubelet_memory_manager_pinning_errors_total

The number of memory pages allocations which required pinning that failed.

- **Stability Level:** ALPHA
- **Type:** Counter

kubelet_memory_manager_pinning_requests_total

The number of memory pages allocations which required pinning.

- **Stability Level:** ALPHA
- **Type:** Counter

kubelet_mirror_pods

The number of mirror pods the kubelet will try to create (one per admitted static pod)

- **Stability Level:** ALPHA
- **Type:** Gauge

kubelet_node_name

The node's name. The count is always 1.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `node`

kubelet_node_startup_duration_seconds

Duration in seconds of node startup in total.

- **Type:** Gauge

kubelet_node_startup_post_registration_duration_seconds

Duration in seconds of node startup after registration.

- **Stability Level:** ALPHA
- **Type:** Gauge

kubelet_node_startup_pre_kubelet_duration_seconds

Duration in seconds of node startup before kubelet starts.

- **Stability Level:** ALPHA
- **Type:** Gauge

kubelet_node_startup_pre_registration_duration_seconds

Duration in seconds of node startup before registration.

- **Stability Level:** ALPHA
- **Type:** Gauge

kubelet_node_startup_registration_duration_seconds

Duration in seconds of node startup during registration.

- **Stability Level:** ALPHA
- **Type:** Gauge

kubelet_orphan_pod_cleaned_volumes

The total number of orphaned Pods whose volumes were cleaned in the last periodic sweep.

- **Stability Level:** ALPHA
- **Type:** Gauge

kubelet_orphan_pod_cleaned_volumes_errors

The number of orphaned Pods whose volumes failed to be cleaned in the last periodic sweep.

- **Stability Level:** ALPHA
- **Type:** Gauge

kubelet_orphaned_runtime_pods_total

Number of pods that have been detected in the container runtime without being already known to the pod worker. This typically indicates the kubelet was restarted while a pod was force deleted in the API or in the local configuration, which is unusual.

- **Stability Level:** ALPHA
- **Type:** Counter

kubelet_pong_discard_events

The number of discard events in PLEG.

- **Stability Level:** ALPHA
- **Type:** Counter

kubelet_pong_last_seen_seconds

Timestamp in seconds when PLEG was last seen active.

- **Stability Level:** ALPHA
- **Type:** Gauge

kubelet_pong_relist_duration_seconds

Duration in seconds for relisting pods in PLEG.

- **Stability Level:** ALPHA
- **Type:** Histogram

kubelet_pong_relist_interval_seconds

Interval in seconds between relisting in PLEG.

- **Stability Level:** ALPHA
- **Type:** Histogram

kubelet_pod_resources_endpoint_errors_get

Number of requests to the PodResource Get endpoint which returned error. Broken down by server api version.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `server_api_version`

kubelet_pod_resources_endpoint_errors_get_allocatable

Number of requests to the PodResource GetAllocatableResources endpoint which returned error. Broken down by server api version.

- **Stability Level:** ALPHA
- **Type:** Counter

kubelet_pod_resources_endpoint_errors_list

Number of requests to the PodResource List endpoint which returned error. Broken down by server api version.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `server_api_version`

kubelet_pod_resources_endpoint_requests_get

Number of requests to the PodResource Get endpoint. Broken down by server api version.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `server_api_version`

kubelet_pod_resources_endpoint_requests_get_allocatable

Number of requests to the PodResource GetAllocatableResources endpoint. Broken down by server api version.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `server_api_version`

kubelet_pod_resources_endpoint_requests_list

Number of requests to the PodResource List endpoint. Broken down by server api version.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `server_api_version`

kubelet_pod_resources_endpoint_requests_total

Cumulative number of requests to the PodResource endpoint. Broken down by server api version.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `server_api_version`

kubelet_pod_start_duration_seconds

Duration in seconds from kubelet seeing a pod for the first time to the pod starting to run

- **Stability Level:** ALPHA

- **Type:** Histogram

kubelet_pod_start_sli_duration_seconds

Duration in seconds to start a pod, excluding time to pull images and run init containers, measured from pod creation timestamp to when all its containers are reported as started and observed via watch

- **Stability Level:** ALPHA
- **Type:** Histogram

kubelet_pod_start_total_duration_seconds

Duration in seconds to start a pod since creation, including time to pull images and run init containers, measured from pod creation timestamp to when all its containers are reported as started and observed via watch

- **Stability Level:** ALPHA
- **Type:** Histogram

kubelet_pod_status_sync_duration_seconds

Duration in seconds to sync a pod status update. Measures time from detection of a change to pod status until the API is successfully updated for that pod, even if multiple intervening changes to pod status occur.

- **Stability Level:** ALPHA
- **Type:** Histogram

kubelet_pod_worker_duration_seconds

Duration in seconds to sync a single pod. Broken down by operation type: create, update, or sync

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `operation_type`

kubelet_pod_worker_start_duration_seconds

Duration in seconds from kubelet seeing a pod to starting a worker.

- **Stability Level:** ALPHA
- **Type:** Histogram

kubelet_preemptions

Cumulative number of pod preemptions by preemption resource

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `preemption_signal`

kubelet_restarted_pods_total

Number of pods that have been restarted because they were deleted and recreated with the same UID while the kubelet was watching them (common for static pods, extremely uncommon for API pods)

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `static`

kubelet_run_podsandbox_duration_seconds

Duration in seconds of the run_podsandbox operations. Broken down by RuntimeClass.Handler.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `runtime_handler`

kubelet_run_podsandbox_errors_total

Cumulative number of the run_podsandbox operation errors by RuntimeClass.Handler.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `runtime_handler`

kubelet_running_containers

Number of containers currently running

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `container_state`

kubelet_running_pods

Number of pods that have a running pod sandbox

- **Stability Level:** ALPHA
- **Type:** Gauge

kubelet_runtime_operations_duration_seconds

Duration in seconds of runtime operations. Broken down by operation type.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `operation_type`

kubelet_runtime_operations_errors_total

Cumulative number of runtime operation errors by operation type.

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `operation_type`

kubelet_runtime_operations_total

Cumulative number of runtime operations by operation type.

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `operation_type`

kubelet_server_expiration_renew_errors

Counter of certificate renewal errors.

- **Stability Level:** ALPHA

- **Type:** Counter

kubelet_sleep_action_terminated_early_total

The number of times lifecycle sleep handler got terminated before it finishes

- **Stability Level:** ALPHA

- **Type:** Counter

kubelet_started_containers_errors_total

Cumulative number of errors when starting containers

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `code` `container_type`

kubelet_started_containers_total

Cumulative number of containers started

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `container_type`

kubelet_started_host_process_containers_errors_total

Cumulative number of errors when starting hostprocess containers. This metric will only be collected on Windows.

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `code` `container_type`

kubelet_started_host_process_containers_total

Cumulative number of hostprocess containers started. This metric will only be collected on Windows.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `container_type`

kubelet_started_pods_errors_total

Cumulative number of errors when starting pods

- **Stability Level:** ALPHA
- **Type:** Counter

kubelet_started_pods_total

Cumulative number of pods started

- **Stability Level:** ALPHA
- **Type:** Counter

kubelet_topology_manager_admission_duration_ms

Duration in milliseconds to serve a pod admission request.

- **Stability Level:** ALPHA
- **Type:** Histogram

kubelet_topology_manager_admission_errors_total

The number of admission request failures where resources could not be aligned.

- **Stability Level:** ALPHA
- **Type:** Counter

kubelet_topology_manager_admission_requests_total

The number of admission requests where resources have to be aligned.

- **Stability Level:** ALPHA
- **Type:** Counter

kubelet_volume_metric_collection_duration_seconds

Duration in seconds to calculate volume stats

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `metric_source`

kubelet_volume_stats_available_bytes

Number of available bytes in the volume

- **Stability Level:** ALPHA
- **Type:** Custom
- **Labels:** `namespace` `persistentvolumeclaim`

kubelet_volume_stats_capacity_bytes

Capacity in bytes of the volume

- **Stability Level:** ALPHA
- **Type:** Custom
- **Labels:** `namespace` `persistentvolumeclaim`

kubelet_volume_stats_health_status_abnormal

Abnormal volume health status. The count is either 1 or 0. 1 indicates the volume is unhealthy, 0 indicates volume is healthy

- **Stability Level:** ALPHA
- **Type:** Custom
- **Labels:** `namespace` `persistentvolumeclaim`

kubelet_volume_stats_inodes

Maximum number of inodes in the volume

- **Stability Level:** ALPHA
- **Type:** Custom
- **Labels:** `namespace` `persistentvolumeclaim`

kubelet_volume_stats_inodes_free

Number of free inodes in the volume

- **Stability Level:** ALPHA
- **Type:** Custom
- **Labels:** `namespace` `persistentvolumeclaim`

kubelet_volume_stats_inodes_used

Number of used inodes in the volume

- **Stability Level:** ALPHA
- **Type:** Custom
- **Labels:** `namespace` `persistentvolumeclaim`

kubelet_volume_stats_used_bytes

Number of used bytes in the volume

- **Stability Level:** ALPHA

- **Type:** Custom
- **Labels:** `namespace` `persistentvolumeclaim`

kubelet_working_pods

Number of pods the kubelet is actually running, broken down by lifecycle phase, whether the pod is desired, orphaned, or runtime only (also orphaned), and whether the pod is static. An orphaned pod has been removed from local configuration or force deleted in the API and consumes resources that are not otherwise visible.

- **Stability Level:** ALPHA

- **Type:** Gauge
- **Labels:** `config` `lifecycle` `static`

kubeproxy_network_programming_duration_seconds

In Cluster Network Programming Latency in seconds

- **Stability Level:** ALPHA

- **Type:** Histogram

kubeproxy_proxy_healthz_total

Cumulative proxy healthz HTTP status

- **Stability Level:** ALPHA

- **Type:** Counter
- **Labels:** `code`

kubeproxy_proxy_livez_total

Cumulative proxy livez HTTP status

- **Stability Level:** ALPHA

- **Type:** Counter
- **Labels:** `code`

kubeproxy_sync_full_proxy_rules_duration_seconds

SyncProxyRules latency in seconds for full resyncs

- **Stability Level:** ALPHA

- **Type:** Histogram

kubeproxy_sync_partial_proxy_rules_duration_seconds

SyncProxyRules latency in seconds for partial resyncs

- **Stability Level:** ALPHA

- **Type:** Histogram

kubeproxy_sync_proxy_rules_duration_seconds

SyncProxyRules latency in seconds

- **Stability Level:** ALPHA
- **Type:** Histogram

kubeproxy_sync_proxy_rules_endpoint_changes_pending

Pending proxy rules Endpoint changes

- **Stability Level:** ALPHA
- **Type:** Gauge

kubeproxy_sync_proxy_rules_endpoint_changes_total

Cumulative proxy rules Endpoint changes

- **Stability Level:** ALPHA
- **Type:** Counter

kubeproxy_sync_proxy_rules_iptables_last

Number of iptables rules written by kube-proxy in last sync

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** table

kubeproxy_sync_proxy_rules_iptables_partial_restore_failures_total

Cumulative proxy iptables partial restore failures

- **Stability Level:** ALPHA
- **Type:** Counter

kubeproxy_sync_proxy_rules_iptables_restore_failures_total

Cumulative proxy iptables restore failures

- **Stability Level:** ALPHA
- **Type:** Counter

kubeproxy_sync_proxy_rules_iptables_total

Total number of iptables rules owned by kube-proxy

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** table

kubeproxy_sync_proxy_rules_last_queued_timestamp_seconds

The last time a sync of proxy rules was queued

- **Stability Level:** ALPHA
- **Type:** Gauge

kubeproxy_sync_proxy_rules_last_timestamp_seconds

The last time proxy rules were successfully synced

- **Stability Level:** ALPHA
- **Type:** Gauge

kubeproxy_sync_proxy_rules_no_local_endpoints_total

Number of services with a Local traffic policy and no endpoints

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `traffic_policy`

kubeproxy_sync_proxy_rules_service_changes_pending

Pending proxy rules Service changes

- **Stability Level:** ALPHA
- **Type:** Gauge

kubeproxy_sync_proxy_rules_service_changes_total

Cumulative proxy rules Service changes

- **Stability Level:** ALPHA
- **Type:** Counter

kubernetes_build_info

A metric with a constant '1' value labeled by major, minor, git version, git commit, git tree state, build date, Go version, and compiler from which Kubernetes was built, and platform on which it is running.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `build_date` `compiler` `git_commit` `git_tree_state` `git_version` `go_version` `major` `minor` `platform`

leader_election_master_status

Gauge of if the reporting system is master of the relevant lease, 0 indicates backup, 1 indicates master. 'name' is the string used to identify the lease. Please make sure to group by name.

- **Stability Level:** ALPHA
- **Type:** Gauge

leader_election_slowpath_total

Total number of slow path exercised in renewing leader leases. 'name' is the string used to identify the lease. Please make sure to group by name.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `name`

node_authorizer_graph_actions_duration_seconds

Histogram of duration of graph actions in node authorizer.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `operation`

node_collector_unhealthy_nodes_in_zone

Gauge measuring number of not Ready Nodes per zones.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `zone`

node_collector_update_all_nodes_health_duration_seconds

Duration in seconds for NodeController to update the health of all nodes.

- **Stability Level:** ALPHA
- **Type:** Histogram

node_collector_update_node_health_duration_seconds

Duration in seconds for NodeController to update the health of a single node.

- **Stability Level:** ALPHA
- **Type:** Histogram

node_collector_zone_health

Gauge measuring percentage of healthy nodes per zone.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `zone`

node_collector_zone_size

Gauge measuring number of registered Nodes per zones.

- **Type:** Gauge

- **Labels:** `zone`

node_controller_cloud_provider_taint_removal_delay_seconds

Number of seconds after node creation when NodeController removed the cloud-provider taint of a single node.

- **Stability Level:** ALPHA

- **Type:** Histogram

node_controller_initial_node_sync_delay_seconds

Number of seconds after node creation when NodeController finished the initial synchronization of a single node.

- **Stability Level:** ALPHA

- **Type:** Histogram

node_ipam_controller_cidrset_allocation_tries_per_request

Number of endpoints added on each Service sync

- **Stability Level:** ALPHA

- **Type:** Histogram

- **Labels:** `clusterCIDR`

node_ipam_controller_cidrset_cidrs_allocations_total

Counter measuring total number of CIDR allocations.

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `clusterCIDR`

node_ipam_controller_cidrset_cidrs_releases_total

Counter measuring total number of CIDR releases.

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `clusterCIDR`

node_ipam_controller_cidrset_usage_cidrs

Gauge measuring percentage of allocated CIDRs.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** clusterCIDR

node_ipam_controller_cirdset_max_cidrs

Maximum number of CIDRs that can be allocated.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** clusterCIDR

node_swap_usage_bytes

Current swap usage of the node in bytes. Reported only on non-windows systems

- **Stability Level:** ALPHA
- **Type:** Custom

number_of_l4_ilbs

Number of L4 ILBs

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** feature

plugin_manager_total_plugins

Number of plugins in Plugin Manager

- **Stability Level:** ALPHA
- **Type:** Custom
- **Labels:** socket_path state

pod_gc_collector_force_delete_pod_errors_total

Number of errors encountered when forcefully deleting the pods since the Pod GC Controller started.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** namespace reason

pod_gc_collector_force_delete_pods_total

Number of pods that are being forcefully deleted since the Pod GC Controller started.

- **Stability Level:** ALPHA

- **Type:** Counter
- **Labels:** `namespace` `reason`

pod_security_errors_total

Number of errors preventing normal evaluation. Non-fatal errors may result in the latest restricted profile being used for evaluation.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `fatal` `request_operation` `resource` `subresource`

pod_security_evaluations_total

Number of policy evaluations that occurred, not counting ignored or exempt requests.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `decision` `mode` `policy_level` `policy_version`
`request_operation` `resource` `subresource`

pod_security_exemptions_total

Number of exempt requests, not counting ignored or out of scope requests.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `request_operation` `resource` `subresource`

pod_swap_usage_bytes

Current amount of the pod swap usage in bytes. Reported only on non-windows systems

- **Stability Level:** ALPHA
- **Type:** Custom
- **Labels:** `pod` `namespace`

prober_probe_duration_seconds

Duration in seconds for a probe response.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `container` `namespace` `pod` `probe_type`

prober_probe_total

Cumulative number of a liveness, readiness or startup probe for a container by result.

- **Type:** Counter
- **Labels:** `container` `namespace` `pod` `pod_uid` `probe_type` `result`

pv_collector_bound_pv_count

Gauge measuring number of persistent volume currently bound

- **Stability Level:** ALPHA
- **Type:** Custom
- **Labels:** `storage_class`

pv_collector_bound_pvc_count

Gauge measuring number of persistent volume claim currently bound

- **Stability Level:** ALPHA
- **Type:** Custom
- **Labels:** `namespace`

pv_collector_total_pv_count

Gauge measuring total number of persistent volumes

- **Stability Level:** ALPHA
- **Type:** Custom
- **Labels:** `plugin_name` `volume_mode`

pv_collector_unbound_pv_count

Gauge measuring number of persistent volume currently unbound

- **Stability Level:** ALPHA
- **Type:** Custom
- **Labels:** `storage_class`

pv_collector_unbound_pvc_count

Gauge measuring number of persistent volume claim currently unbound

- **Stability Level:** ALPHA
- **Type:** Custom
- **Labels:** `namespace`

reconstruct_volume_operations_errors_total

The number of volumes that failed reconstruction from the operating system during kubelet startup.

- **Stability Level:** ALPHA

reconstruct_volume_operations_total

The number of volumes that were attempted to be reconstructed from the operating system during kubelet startup. This includes both successful and failed reconstruction.

- **Stability Level:** ALPHA
- **Type:** Counter

replicaset_controller_sorting_deletion_age_ratio

The ratio of chosen deleted pod's ages to the current youngest pod's age (at the time). Should be <2. The intent of this metric is to measure the rough efficacy of the LogarithmicScaleDown feature gate's effect on the sorting (and deletion) of pods when a replicaset scales down. This only considers Ready pods when calculating and reporting.

- **Stability Level:** ALPHA
- **Type:** Histogram

resourceclaim_controller_create_attempts_total

Number of ResourceClaims creation requests

- **Stability Level:** ALPHA
- **Type:** Counter

resourceclaim_controller_create_failures_total

Number of ResourceClaims creation request failures

- **Stability Level:** ALPHA
- **Type:** Counter

rest_client_dns_resolution_duration_seconds

DNS resolver latency in seconds. Broken down by host.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** host

rest_client_exec_plugin_call_total

Number of calls to an exec plugin, partitioned by the type of event encountered (no_error, plugin_execution_error, plugin_not_found_error, client_internal_error) and an optional exit code. The exit code will be set to 0 if and only if the plugin call was successful.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** call_status code

rest_client_exec_plugin_certificate_rotation_age

Histogram of the number of seconds the last auth exec plugin client certificate lived before being rotated. If auth exec plugin client certificates are unused, histogram will contain no data.

- **Stability Level:** ALPHA
- **Type:** Histogram

rest_client_exec_plugin_ttl_seconds

Gauge of the shortest TTL (time-to-live) of the client certificate(s) managed by the auth exec plugin. The value is in seconds until certificate expiry (negative if already expired). If auth exec plugins are unused or manage no TLS certificates, the value will be +INF.

- **Stability Level:** ALPHA
- **Type:** Gauge

rest_client_rate_limiter_duration_seconds

Client side rate limiter latency in seconds. Broken down by verb, and host.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `host` `verb`

rest_client_request_duration_seconds

Request latency in seconds. Broken down by verb, and host.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `host` `verb`

rest_client_request_retries_total

Number of request retries, partitioned by status code, verb, and host.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `code` `host` `verb`

rest_client_request_size_bytes

Request size in bytes. Broken down by verb and host.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `host` `verb`

rest_client_requests_total

Number of HTTP requests, partitioned by status code, method, and host.

- **Stability Level:** ALPHA

- **Type:** Counter
- **Labels:** `code` `host` `method`

rest_client_response_size_bytes

Response size in bytes. Broken down by verb and host.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `host` `verb`

rest_client_transport_cache_entries

Number of transport entries in the internal cache.

- **Stability Level:** ALPHA
- **Type:** Gauge

rest_client_transport_create_calls_total

Number of calls to get a new transport, partitioned by the result of the operation hit:
obtained from the cache, miss: created and added to the cache, uncacheable: created
and not cached

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `result`

retroactive_storageclass_errors_total

Total number of failed retroactive StorageClass assignments to persistent volume claim

- **Stability Level:** ALPHA
- **Type:** Counter

retroactive_storageclass_total

Total number of retroactive StorageClass assignments to persistent volume claim

- **Stability Level:** ALPHA
- **Type:** Counter

root_ca_cert_publisher_sync_duration_seconds

Number of namespace syncs happened in root ca cert publisher.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `code`

root_ca_cert_publisher_sync_total

Number of namespaces syncs happened in root or cert publisher.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `code`

running_managed_controllers

Indicates where instances of a controller are currently running

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `manager` `name`

scheduler_goroutines

Number of running goroutines split by the work they do such as binding.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `operation`

scheduler_permit_wait_duration_seconds

Duration of waiting on permit.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `result`

scheduler_plugin_evaluation_total

Number of attempts to schedule pods by each plugin and the extension point (available only in PreFilter, Filter, PreScore, and Score).

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `extension_point` `plugin` `profile`

scheduler_plugin_execution_duration_seconds

Duration for running a plugin at a specific extension point.

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `extension_point` `plugin` `status`

scheduler_scheduler_cache_size

Number of nodes, pods, and assumed (bound) pods in the scheduler cache.

- **Stability Level:** ALPHA

- **Type:** Gauge

- **Labels:** `type`

`scheduler_scheduling_algorithm_duration_seconds`

Scheduling algorithm latency in seconds

- **Stability Level:** ALPHA

- **Type:** Histogram

`scheduler_unschedulable_pods`

The number of unschedulable pods broken down by plugin name. A pod will increment the gauge for all plugins that caused it to not schedule and so this metric have meaning only when broken down by plugin.

- **Stability Level:** ALPHA

- **Type:** Gauge

- **Labels:** `plugin` `profile`

`scheduler_volume_binder_cache_requests_total`

Total number for request volume binding cache

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `operation`

`scheduler_volume_scheduling_stage_error_total`

Volume scheduling stage error count

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:** `operation`

`scrape_error`

1 if there was an error while getting container metrics, 0 otherwise

- **Stability Level:** ALPHA

- **Type:** Custom

- **Deprecated Versions:** 1.29.0

service_controller_loadbalancer_sync_total

A metric counting the amount of times any load balancer has been configured, as an effect of service/node changes on the cluster

- **Stability Level:** ALPHA
- **Type:** Counter

service_controller_nodesync_error_total

A metric counting the amount of times any load balancer has been configured and errored, as an effect of node changes on the cluster

- **Stability Level:** ALPHA
- **Type:** Counter

service_controller_nodesync_latency_seconds

A metric measuring the latency for nodesync which updates loadbalancer hosts on cluster node updates.

- **Stability Level:** ALPHA
- **Type:** Histogram

service_controller_update_loadbalancer_host_latency_seconds

A metric measuring the latency for updating each load balancer hosts.

- **Stability Level:** ALPHA
- **Type:** Histogram

serviceaccount_invalid_legacy_auto_token_uses_total

Cumulative invalid auto-generated legacy tokens used

- **Stability Level:** ALPHA
- **Type:** Counter

serviceaccount_legacy_auto_token_uses_total

Cumulative auto-generated legacy tokens used

- **Stability Level:** ALPHA
- **Type:** Counter

serviceaccount_legacy_manual_token_uses_total

Cumulative manually created legacy tokens used

- **Stability Level:** ALPHA
- **Type:** Counter

serviceaccount_legacy_tokens_total

Cumulative legacy service account tokens used

- **Stability Level:** ALPHA

- **Type:** Counter

serviceaccount_stale_tokens_total

Cumulative stale projected service account tokens used

- **Stability Level:** ALPHA

- **Type:** Counter

serviceaccount_valid_tokens_total

Cumulative valid projected service account tokens used

- **Stability Level:** ALPHA

- **Type:** Counter

storage_count_attachable_volumes_in_use

Measure number of volumes in use

- **Stability Level:** ALPHA

- **Type:** Custom

- **Labels:** `node` `volume_plugin`

storage_operation_duration_seconds

Storage operation duration

- **Stability Level:** ALPHA

- **Type:** Histogram

- **Labels:** `migrated` `operation_name` `status` `volume_plugin`

taint_eviction_controller_pod_deletion_duration_seconds

Latency, in seconds, between the time when a taint effect has been activated for the Pod and its deletion via TaintEvictionController.

- **Stability Level:** ALPHA

- **Type:** Histogram

taint_eviction_controller_pod_deletions_total

Total number of Pods deleted by TaintEvictionController since its start.

- **Stability Level:** ALPHA

- **Type:** Counter

ttl_after_finished_controller_job_deletion_duration_seconds

The time it took to delete the job since it became eligible for deletion

- **Stability Level:** ALPHA

- **Type:** Histogram

volume_manager_selinux_container_errors_total

Number of errors when kubelet cannot compute SELinux context for a container. Kubelet can't start such a Pod then and it will retry, therefore value of this metric may not represent the actual nr. of containers.

- **Stability Level:** ALPHA

- **Type:** Gauge

- **Labels:** `access_mode`

volume_manager_selinux_container_warnings_total

Number of errors when kubelet cannot compute SELinux context for a container that are ignored. They will become real errors when SELinuxMountReadWriteOncePod feature is expanded to all volume access modes.

- **Stability Level:** ALPHA

- **Type:** Gauge

- **Labels:** `access_mode`

volume_manager_selinux_pod_context_mismatch_errors_total

Number of errors when a Pod defines different SELinux contexts for its containers that use the same volume. Kubelet can't start such a Pod then and it will retry, therefore value of this metric may not represent the actual nr. of Pods.

- **Stability Level:** ALPHA

- **Type:** Gauge

- **Labels:** `access_mode`

volume_manager_selinux_pod_context_mismatch_warnings_total

Number of errors when a Pod defines different SELinux contexts for its containers that use the same volume. They are not errors yet, but they will become real errors when SELinuxMountReadWriteOncePod feature is expanded to all volume access modes.

- **Stability Level:** ALPHA

- **Type:** Gauge

- **Labels:** `access_mode`

volume_manager_selinux_volume_context_mismatch_errors_total

Number of errors when a Pod uses a volume that is already mounted with a different SELinux context than the Pod needs. Kubelet can't start such a Pod then and it will retry, therefore value of this metric may not represent the actual nr. of Pods.

- **Stability Level:** ALPHA

- **Type:** Gauge

- **Labels:** `access_mode` `volume_plugin`

volume_manager_selinux_volume_context_mismatch_warnings_total

Number of errors when a Pod uses a volume that is already mounted with a different SELinux context than the Pod needs. They are not errors yet, but they will become real errors when SELinuxMountReadWriteOncePod feature is expanded to all volume access modes.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `access_mode` `volume_plugin`

volume_manager_selinux_volumes_admitted_total

Number of volumes whose SELinux context was fine and will be mounted with mount -o context option.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `access_mode` `volume_plugin`

volume_manager_total_volumes

Number of volumes in Volume Manager

- **Stability Level:** ALPHA
- **Type:** Custom
- **Labels:** `plugin_name` `state`

volume_operation_total_errors

Total volume operation errors

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `operation_name` `plugin_name`

volume_operation_total_seconds

Storage operation end to end duration in seconds

- **Stability Level:** ALPHA
- **Type:** Histogram
- **Labels:** `operation_name` `plugin_name`

watch_cache_capacity

Total capacity of watch cache broken by resource type.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `resource`

watch_cache_capacity_decrease_total

Total number of watch cache capacity decrease events broken by resource type.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `resource`

watch_cache_capacity_increase_total

Total number of watch cache capacity increase events broken by resource type.

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `resource`

workqueue_adds_total

Total number of adds handled by workqueue

- **Stability Level:** ALPHA
- **Type:** Counter
- **Labels:** `name`

workqueue_depth

Current depth of workqueue

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `name`

workqueue_longest_running_processor_seconds

How many seconds has the longest running processor for workqueue been running.

- **Stability Level:** ALPHA
- **Type:** Gauge
- **Labels:** `name`

workqueue_queue_duration_seconds

How long in seconds an item stays in workqueue before being requested.

- **Stability Level:** ALPHA

- **Type:** Histogram

- **Labels:**

workqueue_retries_total

Total number of retries handled by workqueue

- **Stability Level:** ALPHA

- **Type:** Counter

- **Labels:**

workqueue_unfinished_work_seconds

How many seconds of work has done that is in progress and hasn't been observed by work_duration. Large values indicate stuck threads. One can deduce the number of stuck threads by observing the rate at which this increases.

- **Stability Level:** ALPHA

- **Type:** Gauge

- **Labels:**

workqueue_work_duration_seconds

How long in seconds processing an item from workqueue takes.

- **Stability Level:** ALPHA

- **Type:** Histogram

- **Labels:**

7 - Kubernetes Issues and Security

7.1 - Kubernetes Issue Tracker

To report a security issue, please follow the [Kubernetes security disclosure process](#).

Work on Kubernetes code and public issues are tracked using [GitHub Issues](#).

- Official [list of known CVEs](#) (security vulnerabilities) that have been announced by the [Security Response Committee](#)
- [CVE-related GitHub issues](#)

Security-related announcements are sent to the kubernetes-security-announce@googlegroups.com mailing list.

7.2 - Kubernetes Security and Disclosure Information

This page describes Kubernetes security and disclosure information.

Security Announcements

Join the [kubernetes-security-announce](#) group for emails about security and major API announcements.

Report a Vulnerability

We're extremely grateful for security researchers and users that report vulnerabilities to the Kubernetes Open Source Community. All reports are thoroughly investigated by a set of community volunteers.

To make a report, submit your vulnerability to the [Kubernetes bug bounty program](#). This allows triage and handling of the vulnerability with standardized response times.

You can also email the private security@kubernetes.io list with the security details and the details expected for [all Kubernetes bug reports](#).

You may encrypt your email to this list using the GPG keys of the [Security Response Committee members](#). Encryption using GPG is NOT required to make a disclosure.

When Should I Report a Vulnerability?

- You think you discovered a potential security vulnerability in Kubernetes
- You are unsure how a vulnerability affects Kubernetes
- You think you discovered a vulnerability in another project that Kubernetes depends on
 - For projects with their own vulnerability reporting and disclosure process, please report it directly there

When Should I NOT Report a Vulnerability?

- You need help tuning Kubernetes components for security
- You need help applying security related updates
- Your issue is not security related

Security Vulnerability Response

Each report is acknowledged and analyzed by Security Response Committee members within 3 working days. This will set off the [Security Release Process](#).

Any vulnerability information shared with Security Response Committee stays within Kubernetes project and will not be disseminated to other projects unless it is necessary to get the issue fixed.

As the security issue moves from triage, to identified fix, to release planning we will keep the reporter updated.

Public Disclosure Timing

A public disclosure date is negotiated by the Kubernetes Security Response Committee and the bug submitter. We prefer to fully disclose the bug as soon as possible once a user mitigation is available. It is reasonable to delay disclosure when the bug or the fix is not yet fully understood, the solution is not well-tested, or for vendor coordination. The timeframe for disclosure is from immediate (especially if it's already publicly known) to a few weeks. For a vulnerability with a straightforward mitigation, we expect report date to disclosure date to be on the order of 7 days. The Kubernetes Security Response Committee holds the final say when setting a disclosure date.

7.3 - Official CVE Feed

ⓘ **FEATURE STATE:** Kubernetes v1.27 [beta]

This is a community maintained list of official CVEs announced by the Kubernetes Security Response Committee. See [Kubernetes Security and Disclosure Information](#) for more details.

The Kubernetes project publishes a programmatically accessible feed of published security issues in [JSON feed](#) and [RSS feed](#) formats. You can access it by executing the following commands:

[JSON feed](#) [RSS feed](#)

[Link to JSON format](#)

```
curl -Lv https://k8s.io/docs/reference/issues-security/official-cve-feed/index.json
```

Official Kubernetes CVE List (last updated: 20 Aug 2024 16:21:37 UTC)

CVE ID	Issue Summary	CVE GitHub Issue URL
CVE-2024-7646	Ingress-nginx Annotation Validation Bypass	#126744
CVE-2024-5321	Incorrect permissions on Windows containers logs	#126161
CVE-2024-3744	azure-file-csi-driver discloses service account tokens in logs	#124759
CVE-2024-3177	Bypassing mountable secrets policy imposed by the ServiceAccount admission plugin	#124336
CVE-2023-5528	Insufficient input sanitization in in-tree storage plugin leads to privilege escalation on Windows nodes	#121879
CVE-2023-5044	Code injection via nginx.ingress.kubernetes.io/ permanent-redirect annotation	#126817
CVE-2023-5043	Ingress nginx annotation injection causes arbitrary command execution	#126816
CVE-2022-4886	Ingress-nginx `path` sanitization can be bypassed	#126815
CVE-2023-3955	Insufficient input sanitization on Windows nodes leads to privilege escalation	#119595
CVE-2023-3893	Insufficient input sanitization on kubernetes-csi-proxy leads to privilege escalation	#119594
CVE-2023-3676	Insufficient input sanitization on Windows nodes leads to privilege escalation	#119339
CVE-2023-2431	Bypass of seccomp profile enforcement	#118690

CVE ID	Issue Summary	CVE	GitHub	Issue URL
CVE-2023-2728	Bypassing policies imposed by the ImagePolicyWebhook and bypassing mountable secrets policy imposed by the ServiceAccount admission plugin	#118640		
CVE-2023-2727	Bypassing policies imposed by the ImagePolicyWebhook and bypassing mountable secrets policy imposed by the ServiceAccount admission plugin	#118640		
CVE-2023-2878	secrets-store-csi-driver discloses service account tokens in logs	#118419		
CVE-2022-3294	Node address isn't always verified when proxying	#113757		
CVE-2022-3162	Unauthorized read of Custom Resources	#113756		
CVE-2022-3172	Aggregated API server can cause clients to be redirected (SSRF)	#112513		
CVE-2021-25749	`runAsNonRoot` logic bypass for Windows containers	#112192		
CVE-2021-25748	Ingress-nginx `path` sanitization can be bypassed with newline character	#126814		
CVE-2021-25746	Ingress-nginx directive injection via annotations	#126813		
CVE-2021-25745	Ingress-nginx `path` can be pointed to service account token file	#126812		
CVE-2021-25742	Ingress-nginx custom snippets allows retrieval of ingress-nginx serviceaccount token and secrets across all namespaces	#126811		
CVE-2021-25741	Symlink Exchange Can Allow Host Filesystem Access	#104980		
CVE-2021-25737	Holes in EndpointSlice Validation Enable Host Network Hijack	#102106		
CVE-2021-3121	Processes may panic upon receipt of malicious protobuf messages	#101435		
CVE-2021-25735	Validating Admission Webhook does not observe some previous fields	#100096		
CVE-2020-8554	Man in the middle using LoadBalancer or ExternalIPs	#97076		
CVE-2020-8566	Ceph RBD adminSecrets exposed in logs when loglevel >= 4	#95624		
CVE-2020-8565	Incomplete fix for CVE-2019-11250 allows for token leak in logs when logLevel >= 9	#95623		

CVE ID	Issue Summary	CVE GitHub Issue URL
CVE-2020-8564	Docker config secrets leaked when file is malformed and log level ≥ 4	#95622
CVE-2020-8563	Secret leaks in kube-controller-manager when using vSphere provider	#95621
CVE-2020-8557	Node disk DOS by writing to container /etc/hosts	#93032
CVE-2020-8559	Privilege escalation from compromised node to cluster	#92914
CVE-2020-8558	Node setting allows for neighboring hosts to bypass localhost boundary	#92315
CVE-2020-8555	Half-Blind SSRF in kube-controller-manager	#91542
CVE-2020-10749	IPv4 only clusters susceptible to MitM attacks via IPv6 rogue router advertisements	#91507
CVE-2019-11254	kube-apiserver Denial of Service vulnerability from malicious YAML payloads	#89535
CVE-2020-8552	apiserver DoS (oom)	#89378
CVE-2020-8551	Kubelet DoS via API	#89377
CVE-2020-8553	auth-type basic annotation vulnerability	#126818
CVE-2019-11251	kubectl cp symlink vulnerability	#87773
CVE-2018-1002102	Unvalidated redirect	#85867
CVE-2019-11255	CSI volume snapshot, cloning and resizing features can result in unauthorized volume data access or mutation	#85233
CVE-2019-11253	Kubernetes API Server JSON/YAML parsing vulnerable to resource exhaustion attack	#83253
CVE-2019-11250	Bearer tokens are revealed in logs	#81114
CVE-2019-11248	/debug/pprof exposed on kubelet's healthz port	#81023
CVE-2019-11249	Incomplete fixes for CVE-2019-1002101 and CVE-2019-11246, kubectl cp potential directory traversal	#80984
CVE-2019-11247	API server allows access to custom resources via wrong scope	#80983
CVE-2019-11245	container uid changes to root after first restart or if image is already pulled to the node	#78308
CVE-2019-11243	rest.AnonymousClientConfig() does not remove the serviceaccount credentials from config created by rest.InClusterConfig()	#76797

CVE ID	Issue Summary	CVE	GitHub	Issue URL
CVE-2019-11244	'kubectl:--http-cache=<world-accessible dir>' creates world-writeable cached schema files			#76676
CVE-2019-1002100	json-patch requests can exhaust apiserver resources			#74534
CVE-2018-1002105	proxy request handling in kube-apiserver can leave vulnerable TCP connections			#71411
CVE-2018-1002101	smb mount security issue			#65750
CVE-2018-1002100	Kubectl copy doesn't check for paths outside of it's destination directory.			#61297
CVE-2017-1002102	atomic writer volume handling allows arbitrary file deletion in host filesystem			#60814
CVE-2017-1002101	subpath volume mount handling allows arbitrary file access in host filesystem			#60813
CVE-2017-1002100	Azure PV should be Private scope not Container scope			#47611
CVE-2017-1000056	PodSecurityPolicy admission plugin authorizes incorrectly			#43459

This feed is auto-refreshing with a noticeable but small lag (minutes to hours) from the time a CVE is announced to the time it is accessible in this feed.

The source of truth of this feed is a set of GitHub Issues, filtered by a controlled and restricted label `official-cve-feed`. The raw data is stored in a Google Cloud Bucket which is writable only by a small number of trusted members of the Community.

8 - Node Reference Information

This section contains the following reference topics about nodes:

- the kubelet's [checkpoint API](#)
- a list of [Articles on dockershim Removal and on Using CRI-compatible Runtimes](#)
- [Kubelet Device Manager API Versions](#)
- [Node Labels Populated By The Kubelet](#)
- [Node .status information](#)

You can also read node reference details from elsewhere in the Kubernetes documentation, including:

- [Node Metrics Data](#).
- [CRI Pod & Container Metrics](#).

8.1 - Kubelet Checkpoint API

ⓘ **FEATURE STATE:** Kubernetes v1.30 [beta]

Checkpointing a container is the functionality to create a stateful copy of a running container. Once you have a stateful copy of a container, you could move it to a different computer for debugging or similar purposes.

If you move the checkpointed container data to a computer that's able to restore it, that restored container continues to run at exactly the same point it was checkpointed. You can also inspect the saved data, provided that you have suitable tools for doing so.

Creating a checkpoint of a container might have security implications. Typically a checkpoint contains all memory pages of all processes in the checkpointed container. This means that everything that used to be in memory is now available on the local disk. This includes all private data and possibly keys used for encryption. The underlying CRI implementations (the container runtime on that node) should create the checkpoint archive to be only accessible by the `root` user. It is still important to remember if the checkpoint archive is transferred to another system all memory pages will be readable by the owner of the checkpoint archive.

Operations

post checkpoint the specified container

Tell the kubelet to checkpoint a specific container from the specified Pod.

Consult the [Kubelet authentication/authorization reference](#) for more information about how access to the kubelet checkpoint interface is controlled.

The kubelet will request a checkpoint from the underlying CRI implementation. In the checkpoint request the kubelet will specify the name of the checkpoint archive as `checkpoint-<podFullName>-<containerName>-<timestamp>.tar` and also request to store the checkpoint archive in the `checkpoints` directory below its root directory (as defined by `--root-dir`). This defaults to `/var/lib/kubelet/checkpoints`.

The checkpoint archive is in `tar` format, and could be listed using an implementation of [tar](#). The contents of the archive depend on the underlying CRI implementation (the container runtime on that node).

HTTP Request

POST /checkpoint/{namespace}/{pod}/{container}

Parameters

- **namespace** (in path): string, required

Namespace

- **pod** (in path): string, required

Pod

- **container** (in path): string, required

Container

- **timeout** (*in query*): integer

Timeout in seconds to wait until the checkpoint creation is finished. If zero or no timeout is specified the default CRI timeout value will be used. Checkpoint creation time depends directly on the used memory of the container. The more memory a container uses the more time is required to create the corresponding checkpoint.

Response

200: OK

401: Unauthorized

404: Not Found (if the `ContainerCheckpoint` feature gate is disabled)

404: Not Found (if the specified `namespace`, `pod` or `container` cannot be found)

500: Internal Server Error (if the CRI implementation encounter an error during checkpointing (see error message for further details))

500: Internal Server Error (if the CRI implementation does not implement the checkpoint CRI API (see error message for further details))

8.2 - Articles on dockershim Removal and on Using CRI-compatible Runtimes

This is a list of articles and other pages that are either about the Kubernetes' deprecation and removal of *dockershim*, or about using CRI-compatible container runtimes, in connection with that removal.

Kubernetes project

- Kubernetes blog: [Dockershim Removal FAQ](#) (originally published 2020/12/02)
- Kubernetes blog: [Updated: Dockershim Removal FAQ](#) (updated published 2022/02/17)
- Kubernetes blog: [Kubernetes is Moving on From Dockershim: Commitments and Next Steps](#) (published 2022/01/07)
- Kubernetes blog: [Dockershim removal is coming. Are you ready?](#) (published 2021/11/12)
- Kubernetes documentation: [Migrating from dockershim](#)
- Kubernetes documentation: [Container Runtimes](#)
- Kubernetes enhancement proposal: [KEP-2221: Removing dockershim from kubelet](#)
- Kubernetes enhancement proposal issue: [Removing dockershim from kubelet](#) ([k/enhancements#2221](#))

You can provide feedback via the GitHub issue [Dockershim removal feedback & issues](#). ([k/kubernetes/#106917](#))

External sources

- Amazon Web Services EKS documentation: [Amazon EKS is ending support for Dockershim](#)
- CNCF conference video: [Lessons Learned Migrating Kubernetes from Docker to containerd Runtime](#) (Ana Caylin, at KubeCon Europe 2019)
- Docker.com blog: [What developers need to know about Docker, Docker Engine, and Kubernetes v1.20](#) (published 2020/12/04)
- "Google Open Source" channel on YouTube: [Learn Kubernetes with Google - Migrating from Dockershim to Containerd](#)
- Microsoft Apps on Azure blog: [Dockershim deprecation and AKS](#) (published 2022/01/21)
- Mirantis blog: [The Future of Dockershim is cri-dockerd](#) (published 2021/04/21)
- Mirantis: [Mirantis/cri-dockerd](#) Official Documentation
- Tripwire: [How Dockershim's Forthcoming Deprecation Affects Your Kubernetes](#) (published 2021/07/01)

8.3 - Node Labels Populated By The Kubelet

Kubernetes nodes come pre-populated with a standard set of labels.

You can also set your own labels on nodes, either through the kubelet configuration or using the Kubernetes API.

Preset labels

The preset labels that Kubernetes sets on nodes are:

- [kubernetes.io/arch](#)
- [kubernetes.io/hostname](#)
- [kubernetes.io/os](#)
- [node.kubernetes.io/instance-type](#) (if known to the kubelet – Kubernetes may not have this information to set the label)
- [topology.kubernetes.io/region](#) (if known to the kubelet – Kubernetes may not have this information to set the label)
- [topology.kubernetes.io/zone](#) (if known to the kubelet – Kubernetes may not have this information to set the label)

Note:

The value of these labels is cloud provider specific and is not guaranteed to be reliable. For example, the value of [kubernetes.io/hostname](#) may be the same as the node name in some environments and a different value in other environments.

What's next

- See [Well-Known Labels, Annotations and Taints](#) for a list of common labels.
- Learn how to [add a label to a node](#).

8.4 - Kubelet Configuration Directory Merging

When using the kubelet's `--config-dir` flag to specify a drop-in directory for configuration, there is some specific behavior on how different types are merged.

Here are some examples of how different data types behave during configuration merging:

Structure Fields

There are two types of structure fields in a YAML structure: singular (or a scalar type) and embedded (structures that contain scalar types). The configuration merging process handles the overriding of singular and embedded struct fields to create a resulting kubelet configuration.

For instance, you may want a baseline kubelet configuration for all nodes, but you may want to customize the `address` and `authorization` fields. This can be done as follows:

Main kubelet configuration file contents:

```
apiVersion: kubelet.config.k8s.io/v1beta1
kind: KubeletConfiguration
port: 20250
authorization:
  mode: Webhook
  webhook:
    cacheAuthorizedTTL: "5m"
    cacheUnauthorizedTTL: "30s"
  serializeImagePulls: false
  address: "192.168.0.1"
```

Contents of a file in `--config-dir` directory:

```
apiVersion: kubelet.config.k8s.io/v1beta1
kind: KubeletConfiguration
authorization:
  mode: AlwaysAllow
  webhook:
    cacheAuthorizedTTL: "8m"
    cacheUnauthorizedTTL: "45s"
  address: "192.168.0.8"
```

The resulting configuration will be as follows:

```
apiVersion: kubelet.config.k8s.io/v1beta1
kind: KubeletConfiguration
port: 20250
serializeImagePulls: false
authorization:
  mode: AlwaysAllow
  webhook:
    cacheAuthorizedTTL: "8m"
    cacheUnauthorizedTTL: "45s"
  address: "192.168.0.8"
```

Lists

You can override the slices/lists values of the kubelet configuration. However, the entire list gets overridden during the merging process. For example, you can override the `clusterDNS` list as follows:

Main kubelet configuration file contents:

```
apiVersion: kubelet.config.k8s.io/v1beta1
kind: KubeletConfiguration
port: 20250
serializeImagePulls: false
clusterDNS:
- "192.168.0.9"
- "192.168.0.8"
```

Contents of a file in `--config-dir` directory:

```
apiVersion: kubelet.config.k8s.io/v1beta1
kind: KubeletConfiguration
clusterDNS:
- "192.168.0.2"
- "192.168.0.3"
- "192.168.0.5"
```

The resulting configuration will be as follows:

```
apiVersion: kubelet.config.k8s.io/v1beta1
kind: KubeletConfiguration
port: 20250
serializeImagePulls: false
clusterDNS:
- "192.168.0.2"
- "192.168.0.3"
- "192.168.0.5"
```

Maps, including Nested Structures

Individual fields in maps, regardless of their value types (boolean, string, etc.), can be selectively overridden. However, for `map[string][]string`, the entire list associated with a specific field gets overridden. Let's understand this better with an example, particularly on fields like `featureGates` and `staticPodURLHeader`:

Main kubelet configuration file contents:

```
apiVersion: kubelet.config.k8s.io/v1beta1
kind: KubeletConfiguration
port: 20250
serializeImagePulls: false
featureGates:
  AllAlpha: false
  MemoryQoS: true
staticPodURLHeader:
  kubelet-api-support:
    - "Authorization: 234APSDFA"
    - "X-Custom-Header: 123"
  custom-static-pod:
    - "Authorization: 223EWRWER"
    - "X-Custom-Header: 456"
```

Contents of a file in `--config-dir` directory:

```
apiVersion: kubelet.config.k8s.io/v1beta1
kind: KubeletConfiguration
featureGates:
  MemoryQoS: false
  KubeletTracing: true
  DynamicResourceAllocation: true
staticPodURLHeader:
  custom-static-pod:
    - "Authorization: 223EWRWER"
    - "X-Custom-Header: 345"
```

The resulting configuration will be as follows:

```
apiVersion: kubelet.config.k8s.io/v1beta1
kind: KubeletConfiguration
port: 20250
serializeImagePulls: false
featureGates:
  AllAlpha: false
  MemoryQoS: false
  KubeletTracing: true
  DynamicResourceAllocation: true
staticPodURLHeader:
  kubelet-api-support:
    - "Authorization: 234APSDFA"
    - "X-Custom-Header: 123"
  custom-static-pod:
    - "Authorization: 223EWRWER"
    - "X-Custom-Header: 345"
```

8.5 - Kubelet Device Manager API Versions

This page provides details of version compatibility between the Kubernetes [device plugin API](#), and different versions of Kubernetes itself.

Compatibility matrix

	v1alpha1	v1beta1
Kubernetes 1.21	-	✓
Kubernetes 1.22	-	✓
Kubernetes 1.23	-	✓
Kubernetes 1.24	-	✓
Kubernetes 1.25	-	✓
Kubernetes 1.26	-	✓

Key:

- ✓ Exactly the same features / API objects in both device plugin API and the Kubernetes version.
- + The device plugin API has features or API objects that may not be present in the Kubernetes cluster, either because the device plugin API has added additional new API calls, or that the server has removed an old API call. However, everything they have in common (most other APIs) will work. Note that alpha APIs may vanish or change significantly between one minor release and the next.
- - The Kubernetes cluster has features the device plugin API can't use, either because server has added additional API calls, or that device plugin API has removed an old API call. However, everything they share in common (most APIs) will work.

8.6 - Node Status

The status of a [node](#) in Kubernetes is a critical aspect of managing a Kubernetes cluster. In this article, we'll cover the basics of monitoring and maintaining node status to ensure a healthy and stable cluster.

Node status fields

A Node's status contains the following information:

- [Addresses](#)
- [Conditions](#)
- [Capacity and Allocatable](#)
- [Info](#)

You can use `kubectl` to view a Node's status and other details:

```
kubectl describe node <insert-node-name-here>
```

Each section of the output is described below.

Addresses

The usage of these fields varies depending on your cloud provider or bare metal configuration.

- **HostName**: The hostname as reported by the node's kernel. Can be overridden via the `kubelet --hostname-override` parameter.
- **ExternalIP**: Typically the IP address of the node that is externally routable (available from outside the cluster).
- **InternalIP**: Typically the IP address of the node that is routable only within the cluster.

Conditions

The `conditions` field describes the status of all `Running` nodes. Examples of conditions include:

Node Condition	Description
Ready	<code>True</code> if the node is healthy and ready to accept pods, <code>False</code> if the node is not healthy and is not accepting pods, and <code>Unknown</code> if the node controller has not heard from the node in the last <code>node-monitor-grace-period</code> (default is 40 seconds)
DiskPressure	<code>True</code> if pressure exists on the disk size—that is, if the disk capacity is low; otherwise <code>False</code>
MemoryPressure	<code>True</code> if pressure exists on the node memory—that is, if the node memory is low; otherwise <code>False</code>
PIDPressure	<code>True</code> if pressure exists on the processes—that is, if there are too many processes on the node; otherwise <code>False</code>

Node Condition	Description
NetworkUnavailable	True if the network for the node is not correctly configured, otherwise False

Note:

If you use command-line tools to print details of a cordoned Node, the Condition includes `SchedulingDisabled`. `SchedulingDisabled` is not a Condition in the Kubernetes API; instead, cordoned nodes are marked `Unschedulable` in their spec.

In the Kubernetes API, a node's condition is represented as part of the `.status` of the Node resource. For example, the following JSON structure describes a healthy node:

```
"conditions": [
  {
    "type": "Ready",
    "status": "True",
    "reason": "KubeletReady",
    "message": "kubelet is posting ready status",
    "lastHeartbeatTime": "2019-06-05T18:38:35Z",
    "lastTransitionTime": "2019-06-05T11:41:27Z"
  }
]
```

When problems occur on nodes, the Kubernetes control plane automatically creates [taints](#) that match the conditions affecting the node. An example of this is when the `status` of the Ready condition remains `Unknown` or `False` for longer than the kube-controller-manager's `NodeMonitorGracePeriod`, which defaults to 40 seconds. This will cause either an `node.kubernetes.io/unreachable` taint, for an `Unknown` status, or a `node.kubernetes.io/not-ready` taint, for a `False` status, to be added to the Node.

These taints affect pending pods as the scheduler takes the Node's taints into consideration when assigning a pod to a Node. Existing pods scheduled to the node may be evicted due to the application of `NoExecute` taints. Pods may also have [tolerations](#) that let them schedule to and continue running on a Node even though it has a specific taint.

See [Taint Based Evictions](#) and [Taint Nodes by Condition](#) for more details.

Capacity and Allocatable

Describes the resources available on the node: CPU, memory, and the maximum number of pods that can be scheduled onto the node.

The fields in the capacity block indicate the total amount of resources that a Node has. The allocatable block indicates the amount of resources on a Node that is available to be consumed by normal Pods.

You may read more about capacity and allocatable resources while learning how to [reserve compute resources](#) on a Node.

Info

Describes general information about the node, such as kernel version, Kubernetes version (kubelet and kube-proxy version), container runtime details, and which operating system the node uses. The kubelet gathers this information from the node and publishes it into the Kubernetes API.

Heartbeats

Heartbeats, sent by Kubernetes nodes, help your cluster determine the availability of each node, and to take action when failures are detected.

For nodes there are two forms of heartbeats:

- updates to the `.status` of a Node
- [Lease](#) objects within the `kube-node-lease` namespace. Each Node has an associated Lease object.

Compared to updates to `.status` of a Node, a Lease is a lightweight resource. Using Leases for heartbeats reduces the performance impact of these updates for large clusters.

The kubelet is responsible for creating and updating the `.status` of Nodes, and for updating their related Leases.

- The kubelet updates the node's `.status` either when there is change in status or if there has been no update for a configured interval. The default interval for `.status` updates to Nodes is 5 minutes, which is much longer than the 40 second default timeout for unreachable nodes.
- The kubelet creates and then updates its Lease object every 10 seconds (the default update interval). Lease updates occur independently from updates to the Node's `.status`. If the Lease update fails, the kubelet retries, using exponential backoff that starts at 200 milliseconds and capped at 7 seconds.

9 - Networking Reference

This section of the Kubernetes documentation provides reference details of Kubernetes networking.

9.1 - Protocols for Services

If you configure a [Service](#), you can select from any network protocol that Kubernetes supports.

Kubernetes supports the following protocols with Services:

- [SCTP](#)
- [TCP \(the default\)](#)
- [UDP](#)

When you define a Service, you can also specify the [application protocol](#) that it uses.

This document details some special cases, all of them typically using TCP as a transport protocol:

- [HTTP](#) and [HTTPS](#)
- [PROXY protocol](#)
- [TLS](#) termination at the load balancer

Supported protocols

There are 3 valid values for the `protocol` of a port for a Service:

SCTP

ⓘ **FEATURE STATE:** Kubernetes v1.20 [stable]

When using a network plugin that supports SCTP traffic, you can use SCTP for most Services. For `type: LoadBalancer` Services, SCTP support depends on the cloud provider offering this facility. (Most do not).

SCTP is not supported on nodes that run Windows.

Support for multihomed SCTP associations

The support of multihomed SCTP associations requires that the CNI plugin can support the assignment of multiple interfaces and IP addresses to a Pod.

NAT for multihomed SCTP associations requires special logic in the corresponding kernel modules.

TCP

You can use TCP for any kind of Service, and it's the default network protocol.

UDP

You can use UDP for most Services. For `type: LoadBalancer` Services, UDP support depends on the cloud provider offering this facility.

Special cases

HTTP

If your cloud provider supports it, you can use a Service in LoadBalancer mode to configure a load balancer outside of your Kubernetes cluster, in a special mode where your cloud provider's load balancer implements HTTP / HTTPS reverse proxying, with traffic forwarded to the backend endpoints for that Service.

Typically, you set the protocol for the Service to `TCP` and add an annotation (usually specific to your cloud provider) that configures the load balancer to handle traffic at the HTTP level. This configuration might also include serving HTTPS (HTTP over TLS) and reverse-proxying plain HTTP to your workload.

Note:

You can also use an [Ingress](#) to expose HTTP/HTTPS Services.

You might additionally want to specify that the [application protocol](#) of the connection is `http` or `https`. Use `http` if the session from the load balancer to your workload is HTTP without TLS, and use `https` if the session from the load balancer to your workload uses TLS encryption.

PROXY protocol

If your cloud provider supports it, you can use a Service set to `type: LoadBalancer` to configure a load balancer outside of Kubernetes itself, that will forward connections wrapped with the [PROXY protocol](#).

The load balancer then sends an initial series of octets describing the incoming connection, similar to this example (PROXY protocol v1):

```
PROXY TCP4 192.0.2.202 10.0.42.7 12345 7\r\n
```

The data after the proxy protocol preamble are the original data from the client. When either side closes the connection, the load balancer also triggers a connection close and sends any remaining data where feasible.

Typically, you define a Service with the protocol to `TCP`. You also set an annotation, specific to your cloud provider, that configures the load balancer to wrap each incoming connection in the PROXY protocol.

TLS

If your cloud provider supports it, you can use a Service set to `type: LoadBalancer` as a way to set up external reverse proxying, where the connection from client to load balancer is TLS encrypted and the load balancer is the TLS server peer. The connection from the load balancer to your workload can also be TLS, or might be plain text. The exact options available to you depend on your cloud provider or custom Service implementation.

Typically, you set the protocol to `TCP` and set an annotation (usually specific to your cloud provider) that configures the load balancer to act as a TLS server. You would configure the TLS identity (as server, and possibly also as a client that connects to your workload) using mechanisms that are specific to your cloud provider.

9.2 - Ports and Protocols

When running Kubernetes in an environment with strict network boundaries, such as on-premises datacenter with physical network firewalls or Virtual Networks in Public Cloud, it is useful to be aware of the ports and protocols used by Kubernetes components.

Control plane

Protocol	Direction	Port Range	Purpose	Used By
TCP	Inbound	6443	Kubernetes API server	All
TCP	Inbound	2379-2380	etcd server client API	kube-apiserver, etcd
TCP	Inbound	10250	Kubelet API	Self, Control plane
TCP	Inbound	10259	kube-scheduler	Self
TCP	Inbound	10257	kube-controller- manager	Self

Although etcd ports are included in control plane section, you can also host your own etcd cluster externally or on custom ports.

Worker node(s)

Protocol	Direction	Port Range	Purpose	Used By
TCP	Inbound	10250	Kubelet API	Self, Control plane
TCP	Inbound	10256	kube-proxy	Self, Load balancers
TCP	Inbound	30000-32767	NodePort Services†	All

† Default port range for [NodePort Services](#).

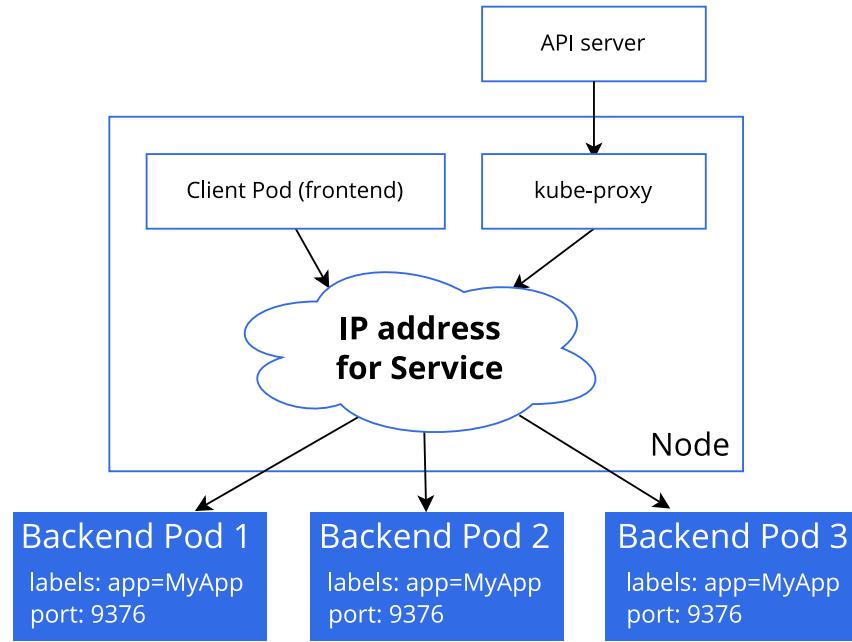
All default port numbers can be overridden. When custom ports are used those ports need to be open instead of defaults mentioned here.

One common example is API server port that is sometimes switched to 443. Alternatively, the default port is kept as is and API server is put behind a load balancer that listens on 443 and routes the requests to API server on the default port.

9.3 - Virtual IPs and Service Proxies

Every node in a Kubernetes cluster runs a [kube-proxy](#) (unless you have deployed your own alternative component in place of `kube-proxy`).

The `kube-proxy` component is responsible for implementing a *virtual IP* mechanism for Services of type other than [ExternalName](#) . Each instance of kube-proxy watches the Kubernetes control plane for the addition and removal of Service and EndpointSlice objects. For each Service, kube-proxy calls appropriate APIs (depending on the kube-proxy mode) to configure the node to capture traffic to the Service's `clusterIP` and `port` , and redirect that traffic to one of the Service's endpoints (usually a Pod, but possibly an arbitrary user-provided IP address). A control loop ensures that the rules on each node are reliably synchronized with the Service and EndpointSlice state as indicated by the API server.



Virtual IP mechanism for Services, using iptables mode

A question that pops up every now and then is why Kubernetes relies on proxying to forward inbound traffic to backends. What about other approaches? For example, would it be possible to configure DNS records that have multiple A values (or AAAA for IPv6), and rely on round-robin name resolution?

There are a few reasons for using proxying for Services:

- There is a long history of DNS implementations not respecting record TTLs, and caching the results of name lookups after they should have expired.
- Some apps do DNS lookups only once and cache the results indefinitely.
- Even if apps and libraries did proper re-resolution, the low or zero TTLs on the DNS records could impose a high load on DNS that then becomes difficult to manage.

Later in this page you can read about how various kube-proxy implementations work. Overall, you should note that, when running `kube-proxy` , kernel level rules may be modified (for example, iptables rules might get created), which won't get cleaned up, in some cases until you reboot. Thus, running `kube-proxy` is something that should only be done by an administrator which understands the consequences of having a low level, privileged network proxying service on a computer. Although the `kube-proxy` executable supports a `cleanup` function, this function is not an official feature and thus is only available to use as-is.

Some of the details in this reference refer to an example: the backend

Pods for a stateless image-processing workloads, running with three replicas. Those replicas are fungible—frontends do not care which backend they use. While the actual Pods that compose the backend set may change, the frontend clients should not need to be aware of that, nor should they need to keep track of the set of backends themselves.

Proxy modes

The kube-proxy starts up in different modes, which are determined by its configuration.

On Linux nodes, the available modes for kube-proxy are:

[iptables](#)

A mode where the kube-proxy configures packet forwarding rules using iptables.

[ipvs](#)

a mode where the kube-proxy configures packet forwarding rules using ipvs.

[nftables](#)

a mode where the kube-proxy configures packet forwarding rules using nftables.

There is only one mode available for kube-proxy on Windows:

[kernel space](#)

a mode where the kube-proxy configures packet forwarding rules in the Windows kernel

iptables proxy mode

This proxy mode is only available on Linux nodes.

In this mode, kube-proxy configures packet forwarding rules using the iptables API of the kernel netfilter subsystem. For each endpoint, it installs iptables rules which, by default, select a backend Pod at random.

Example

As an example, consider the image processing application described [earlier](#) in the page. When the backend Service is created, the Kubernetes control plane assigns a virtual IP address, for example 10.0.0.1. For this example, assume that the Service port is 1234. All of the kube-proxy instances in the cluster observe the creation of the new Service.

When kube-proxy on a node sees a new Service, it installs a series of iptables rules which redirect from the virtual IP address to more iptables rules, defined per Service. The per-Service rules link to further rules for each backend endpoint, and the per- endpoint rules redirect traffic (using destination NAT) to the backends.

When a client connects to the Service's virtual IP address the iptables rule kicks in. A backend is chosen (either based on session affinity or randomly) and packets are redirected to the backend without rewriting the client IP address.

This same basic flow executes when traffic comes in through a node-port or through a load-balancer, though in those cases the client IP address does get altered.

Optimizing iptables mode performance

In iptables mode, kube-proxy creates a few iptables rules for every Service, and a few iptables rules for each endpoint IP address. In clusters with tens of thousands of Pods and Services, this means tens of thousands of iptables rules, and kube-proxy may take a long time to update the rules in the kernel when Services (or their EndpointSlices) change. You can adjust the syncing behavior of kube-proxy via options in the [iptables section](#) of the kube-proxy [configuration file](#) (which you specify via `kube-proxy --config <path>`):

```
...
iptables:
  minSyncPeriod: 1s
  syncPeriod: 30s
...
```

minSyncPeriod

The `minSyncPeriod` parameter sets the minimum duration between attempts to resynchronize iptables rules with the kernel. If it is `0s`, then kube-proxy will always immediately synchronize the rules every time any Service or Endpoint changes. This works fine in very small clusters, but it results in a lot of redundant work when lots of things change in a small time period. For example, if you have a Service backed by a [Deployment](#) with 100 pods, and you delete the Deployment, then with `minSyncPeriod: 0s`, kube-proxy would end up removing the Service's endpoints from the iptables rules one by one, for a total of 100 updates. With a larger `minSyncPeriod`, multiple Pod deletion events would get aggregated together, so kube-proxy might instead end up making, say, 5 updates, each removing 20 endpoints, which will be much more efficient in terms of CPU, and result in the full set of changes being synchronized faster.

The larger the value of `minSyncPeriod`, the more work that can be aggregated, but the downside is that each individual change may end up waiting up to the full `minSyncPeriod` before being processed, meaning that the iptables rules spend more time being out-of-sync with the current API server state.

The default value of `1s` should work well in most clusters, but in very large clusters it may be necessary to set it to a larger value. Especially, if kube-proxy's `sync_proxy_rules_duration_seconds` metric indicates an average time much larger than 1 second, then bumping up `minSyncPeriod` may make updates more efficient.

Updating legacy `minSyncPeriod` configuration

Older versions of kube-proxy updated all the rules for all Services on every sync; this led to performance issues (update lag) in large clusters, and the recommended solution was to set a larger `minSyncPeriod`. Since Kubernetes v1.28, the iptables mode of kube-proxy uses a more minimal approach, only making updates where Services or EndpointSlices have actually changed.

If you were previously overriding `minSyncPeriod`, you should try removing that override and letting kube-proxy use the default value (`1s`) or at least a smaller value than you were using before upgrading.

If you are not running kube-proxy from Kubernetes 1.31, check the behavior and associated advice for the version that you are actually

running.

syncPeriod

The `syncPeriod` parameter controls a handful of synchronization operations that are not directly related to changes in individual Services and EndpointSlices. In particular, it controls how quickly kube-proxy notices if an external component has interfered with kube-proxy's iptables rules. In large clusters, kube-proxy also only performs certain cleanup operations once every `syncPeriod` to avoid unnecessary work.

For the most part, increasing `syncPeriod` is not expected to have much impact on performance, but in the past, it was sometimes useful to set it to a very large value (eg, `1h`). This is no longer recommended, and is likely to hurt functionality more than it improves performance.

IPVS proxy mode

This proxy mode is only available on Linux nodes.

In `ipvs` mode, kube-proxy uses the kernel IPVS and iptables APIs to create rules to redirect traffic from Service IPs to endpoint IPs.

The IPVS proxy mode is based on netfilter hook function that is similar to iptables mode, but uses a hash table as the underlying data structure and works in the kernel space. That means kube-proxy in IPVS mode redirects traffic with lower latency than kube-proxy in iptables mode, with much better performance when synchronizing proxy rules.

Compared to the iptables proxy mode, IPVS mode also supports a higher throughput of network traffic.

IPVS provides more options for balancing traffic to backend Pods; these are:

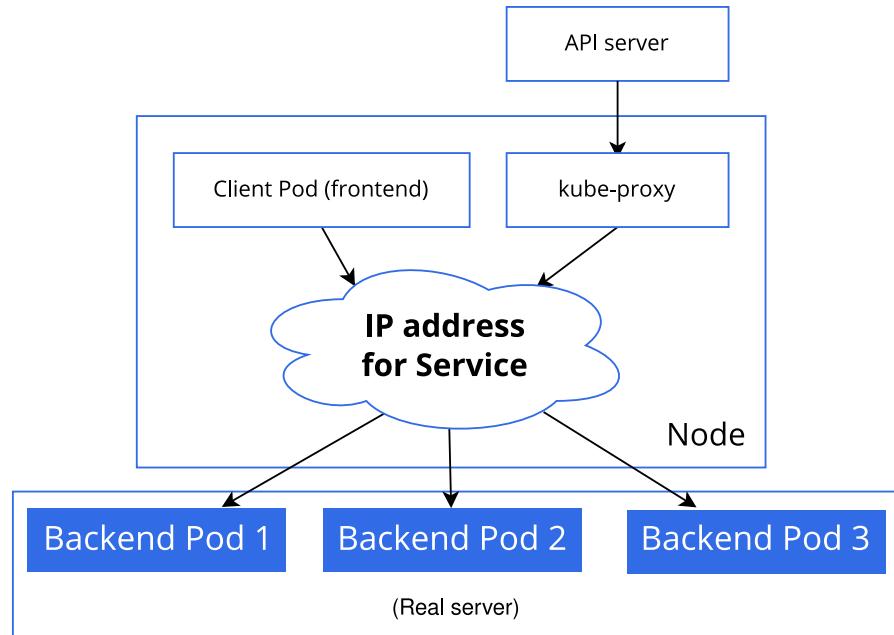
- `rr` (Round Robin): Traffic is equally distributed amongst the backing servers.
- `wrr` (Weighted Round Robin): Traffic is routed to the backing servers based on the weights of the servers. Servers with higher weights receive new connections and get more requests than servers with lower weights.
- `lc` (Least Connection): More traffic is assigned to servers with fewer active connections.
- `wlc` (Weighted Least Connection): More traffic is routed to servers with fewer connections relative to their weights, that is, connections divided by weight.
- `lblc` (Locality based Least Connection): Traffic for the same IP address is sent to the same backing server if the server is not overloaded and available; otherwise the traffic is sent to servers with fewer connections, and keep it for future assignment.
- `lblcr` (Locality Based Least Connection with Replication): Traffic for the same IP address is sent to the server with least connections. If all the backing servers are overloaded, it picks up one with fewer connections and add it to the target set. If the target set has not changed for the specified time, the most loaded server is removed from the set, in order to avoid high degree of replication.
- `sh` (Source Hashing): Traffic is sent to a backing server by looking up a statically assigned hash table based on the source IP addresses.

- `dh` (Destination Hashing): Traffic is sent to a backing server by looking up a statically assigned hash table based on their destination addresses.
- `sed` (Shortest Expected Delay): Traffic forwarded to a backing server with the shortest expected delay. The expected delay is $(c + 1) / u$ if sent to a server, where c is the number of connections on the server and u is the fixed service rate (weight) of the server.
- `nq` (Never Queue): Traffic is sent to an idle server if there is one, instead of waiting for a fast one; if all servers are busy, the algorithm falls back to the `sed` behavior.

Note:

To run kube-proxy in IPVS mode, you must make IPVS available on the node before starting kube-proxy.

When kube-proxy starts in IPVS proxy mode, it verifies whether IPVS kernel modules are available. If the IPVS kernel modules are not detected, then kube-proxy exits with an error.



Virtual IP address mechanism for Services, using IPVS mode

nftables proxy mode

ⓘ **FEATURE STATE:** Kubernetes v1.31 [beta]

This proxy mode is only available on Linux nodes, and requires kernel 5.13 or later.

In this mode, kube-proxy configures packet forwarding rules using the nftables API of the kernel netfilter subsystem. For each endpoint, it installs nftables rules which, by default, select a backend Pod at random.

The nftables API is the successor to the iptables API and is designed to provide better performance and scalability than iptables. The `nftables` proxy mode is able to process changes to service endpoints faster and more efficiently than the `iptables` mode, and is also able to more efficiently process packets in the kernel (though this only becomes noticeable in clusters with tens of thousands of services).

As of Kubernetes 1.31, the `nftables` mode is still relatively new, and may not be compatible with all network plugins; consult the documentation for your network plugin.

Migrating from `iptables` mode to `nftables`

Users who want to switch from the default `iptables` mode to the `nftables` mode should be aware that some features work slightly differently in the `nftables` mode:

- **NodePort interfaces:** In `iptables` mode, by default, [NodePort services](#) are reachable on all local IP addresses. This is usually not what users want, so the `nftables` mode defaults to `--nodeport-addresses primary`, meaning NodePort services are only reachable on the node's primary IPv4 and/or IPv6 addresses. You can override this by specifying an explicit value for that option: e.g., `--nodeport-addresses 0.0.0.0/0` to listen on all (local) IPv4 IPs.
- **NodePort services on 127.0.0.1:** In `iptables` mode, if the `--nodeport-addresses` range includes `127.0.0.1` (and the option `--iptables-localhost-nodeports false` option is not passed), then NodePort services are reachable even on "localhost" (`127.0.0.1`). In `nftables` mode (and `ipvs` mode), this will not work. If you are not sure if you are depending on this functionality, you can check kube-proxy's `iptables_localhost_nodeports_accepted_packets_total` metric; if it is non-0, that means that some client has connected to a NodePort service via `127.0.0.1`.
- **NodePort interaction with firewalls:** The `iptables` mode of kube-proxy tries to be compatible with overly-aggressive firewalls; for each NodePort service, it will add rules to accept inbound traffic on that port, in case that traffic would otherwise be blocked by a firewall. This approach will not work with firewalls based on `nftables`, so kube-proxy's `nftables` mode does not do anything here; if you have a local firewall, you must ensure that it is properly configured to allow Kubernetes traffic through (e.g., by allowing inbound traffic on the entire NodePort range).
- **Conntrack bug workarounds:** Linux kernels prior to 6.1 have a bug that can result in long-lived TCP connections to service IPs being closed with the error "Connection reset by peer". The `iptables` mode of kube-proxy installs a workaround for this bug, but this workaround was later found to cause other problems in some clusters. The `nftables` mode does not install any workaround by default, but you can check kube-proxy's `iptables_ct_state_invalid_dropped_packets_total` metric to see if your cluster is depending on the workaround, and if so, you can run kube-proxy with the option `--conntrack-tcp-be-liberal` to work around the problem in `nftables` mode.

`kernelspace` proxy mode

This proxy mode is only available on Windows nodes.

The kube-proxy configures packet filtering rules in the Windows *Virtual Filtering Platform* (VFP), an extension to Windows vSwitch. These rules process encapsulated packets within the node-level virtual networks, and rewrite packets so that the destination IP address (and layer 2 information) is correct for getting the packet routed to the correct destination. The Windows VFP is analogous to tools such as Linux `nftables` or `iptables`. The Windows VFP extends the *Hyper-V Switch*, which was initially implemented to support virtual machine networking.

When a Pod on a node sends traffic to a virtual IP address, and the kube-proxy selects a Pod on a different node as the load balancing target, the `kernelspace` proxy mode rewrites that packet to be destined to the target backend Pod. The Windows *Host Networking Service* (HNS) ensures that packet rewriting rules are configured so that the return traffic

appears to come from the virtual IP address and not the specific backend Pod.

Direct server return for `kernelspace` mode

ⓘ **FEATURE STATE:** Kubernetes v1.14 [alpha]

As an alternative to the basic operation, a node that hosts the backend Pod for a Service can apply the packet rewriting directly, rather than placing this burden on the node where the client Pod is running. This is called *direct server return*.

To use this, you must run kube-proxy with the `--enable-dsr` command line argument **and** enable the `WnDSR` [feature gate](#).

Direct server return also optimizes the case for Pod return traffic even when both Pods are running on the same node.

Session affinity

In these proxy models, the traffic bound for the Service's IP:Port is proxied to an appropriate backend without the clients knowing anything about Kubernetes or Services or Pods.

If you want to make sure that connections from a particular client are passed to the same Pod each time, you can select the session affinity based on the client's IP addresses by setting `.spec.sessionAffinity` to `ClientIP` for a Service (the default is `None`).

Session stickiness timeout

You can also set the maximum session sticky time by setting `.spec.sessionAffinityConfig.clientIP.timeoutSeconds` appropriately for a Service. (the default value is 10800, which works out to be 3 hours).

Note:

On Windows, setting the maximum session sticky time for Services is not supported.

IP address assignment to Services

Unlike Pod IP addresses, which actually route to a fixed destination, Service IPs are not actually answered by a single host. Instead, kube-proxy uses packet processing logic (such as Linux iptables) to define *virtual* IP addresses which are transparently redirected as needed.

When clients connect to the VIP, their traffic is automatically transported to an appropriate endpoint. The environment variables and DNS for Services are actually populated in terms of the Service's virtual IP address (and port).

Avoiding collisions

One of the primary philosophies of Kubernetes is that you should not be exposed to situations that could cause your actions to fail through no fault of your own. For the design of the Service resource, this means not making you choose your own IP address if that choice might collide with

someone else's choice. That is an isolation failure.

In order to allow you to choose an IP address for your Services, we must ensure that no two Services can collide. Kubernetes does that by allocating each Service its own IP address from within the `service-cluster-ip-range` CIDR range that is configured for the API Server.

IP address allocation tracking

To ensure each Service receives a unique IP address, an internal allocator atomically updates a global allocation map in `etcd` prior to creating each Service. The map object must exist in the registry for Services to get IP address assignments, otherwise creations will fail with a message indicating an IP address could not be allocated.

In the control plane, a background controller is responsible for creating that map (needed to support migrating from older versions of Kubernetes that used in-memory locking). Kubernetes also uses controllers to check for invalid assignments (for example: due to administrator intervention) and for cleaning up allocated IP addresses that are no longer used by any Services.

IP address allocation tracking using the Kubernetes API

ⓘ FEATURE STATE: Kubernetes v1.31 [beta]

If you enable the `MulticIDRServiceAllocator` [feature gate](#) and the [networking.k8s.io/v1alpha1 API group](#), the control plane replaces the existing `etcd` allocator with a revised implementation that uses `IPAddress` and `ServiceCIDR` objects instead of an internal global allocation map. Each cluster IP address associated to a Service then references an `IPAddress` object.

Enabling the feature gate also replaces a background controller with an alternative that handles the `IPAddress` objects and supports migration from the old allocator model. Kubernetes 1.31 does not support migrating from `IPAddress` objects to the internal allocation map.

One of the main benefits of the revised allocator is that it removes the size limitations for the IP address range that can be used for the cluster IP address of Services. With `MulticIDRServiceAllocator` enabled, there are no limitations for IPv4, and for IPv6 you can use IP address netmasks that are a /64 or smaller (as opposed to /108 with the legacy implementation).

Making IP address allocations available via the API means that you as a cluster administrator can allow users to inspect the IP addresses assigned to their Services. Kubernetes extensions, such as the [Gateway API](#), can use the `IPAddress` API to extend Kubernetes' inherent networking capabilities.

Here is a brief example of a user querying for IP addresses:

```
kubectl get services
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
kubernetes	ClusterIP	2001:db8:1:2::1	<none>	443/TCP	3d

```
kubectl get ipaddresses
```

NAME	PARENTREF
2001:db8:1:2::1	services/default/kubernetes
2001:db8:1:2::a	services/kube-system/kube-dns

Kubernetes also allow users to dynamically define the available IP ranges for Services using ServiceCIDR objects. During bootstrap, a default ServiceCIDR object named `kubernetes` is created from the value of the `--service-cluster-ip-range` command line argument to kube-apiserver:

```
kubectl get servicecidrs
```

NAME	CIDRS	AGE
kubernetes	10.96.0.0/28	17m

Users can create or delete new ServiceCIDR objects to manage the available IP ranges for Services:

```
cat <<'EOF' | kubectl apply -f -
apiVersion: networking.k8s.io/v1beta1
kind: ServiceCIDR
metadata:
  name: newservicecidr
spec:
  cidrs:
  - 10.96.0.0/24
EOF
```

```
servicecidr.networking.k8s.io/newcidr1 created
```

```
kubectl get servicecidrs
```

NAME	CIDRS	AGE
kubernetes	10.96.0.0/28	17m
newservicecidr	10.96.0.0/24	7m

IP address ranges for Service virtual IP addresses

ⓘ **FEATURE STATE:** Kubernetes v1.26 [stable]

Kubernetes divides the `clusterIP` range into two bands, based on the size of the configured `service-cluster-ip-range` by using the following formula `min(max(16, cidrSize / 16), 256)`. That formula paraphrases as *never less than 16 or more than 256, with a graduated step function between them*.

Kubernetes prefers to allocate dynamic IP addresses to Services by choosing from the upper band, which means that if you want to assign a specific IP address to a `type: ClusterIP` Service, you should manually

assign an IP address from the **lower** band. That approach reduces the risk of a conflict over allocation.

Traffic policies

You can set the `.spec.internalTrafficPolicy` and `.spec.externalTrafficPolicy` fields to control how Kubernetes routes traffic to healthy (“ready”) backends.

Internal traffic policy

ⓘ **FEATURE STATE:** Kubernetes v1.26 [stable]

You can set the `.spec.internalTrafficPolicy` field to control how traffic from internal sources is routed. Valid values are `Cluster` and `Local`. Set the field to `Cluster` to route internal traffic to all ready endpoints and `Local` to only route to ready node-local endpoints. If the traffic policy is `Local` and there are no node-local endpoints, traffic is dropped by kube-proxy.

External traffic policy

You can set the `.spec.externalTrafficPolicy` field to control how traffic from external sources is routed. Valid values are `Cluster` and `Local`. Set the field to `Cluster` to route external traffic to all ready endpoints and `Local` to only route to ready node-local endpoints. If the traffic policy is `Local` and there are no node-local endpoints, the kube-proxy does not forward any traffic for the relevant Service.

If `Cluster` is specified all nodes are eligible load balancing targets *as long as* the node is not being deleted and kube-proxy is healthy. In this mode: load balancer health checks are configured to target the service proxy's readiness port and path. In the case of kube-proxy this evaluates to: `${NODE_IP}:10256/healthz`. kube-proxy will return either an HTTP code 200 or 503. kube-proxy's load balancer health check endpoint returns 200 if:

1. kube-proxy is healthy, meaning:
 - it's able to progress programming the network and isn't timing out while doing so (the timeout is defined to be: **2 × iptables.syncPeriod**); and
2. the node is not being deleted (there is no deletion timestamp set for the Node).

The reason why kube-proxy returns 503 and marks the node as not eligible when it's being deleted, is because kube-proxy supports connection draining for terminating nodes. A couple of important things occur from the point of view of a Kubernetes-managed load balancer when a node *is being / is* deleted.

While deleting:

- kube-proxy will start failing its readiness probe and essentially mark the node as not eligible for load balancer traffic. The load balancer health check failing causes load balancers which support connection draining to allow existing connections to terminate, and block new connections from establishing.

When deleted:

- The service controller in the Kubernetes cloud controller manager

removes the node from the referenced set of eligible targets. Removing any instance from the load balancer's set of backend targets immediately terminates all connections. This is also the reason kube-proxy first fails the health check while the node is deleting.

It's important to note for Kubernetes vendors that if any vendor configures the kube-proxy readiness probe as a liveness probe: that kube-proxy will start restarting continuously when a node is deleting until it has been fully deleted. kube-proxy exposes a `/livez` path which, as opposed to the `/healthz` one, does **not** consider the Node's deleting state and only its progress programming the network. `/livez` is therefore the recommended path for anyone looking to define a `livenessProbe` for kube-proxy.

Users deploying kube-proxy can inspect both the readiness / liveness state by evaluating the metrics: `proxy_livez_total` / `proxy_healthz_total`. Both metrics publish two series, one with the 200 label and one with the 503 one.

For `Local` Services: kube-proxy will return 200 if

1. kube-proxy is healthy/ready, and
2. has a local endpoint on the node in question.

Node deletion does **not** have an impact on kube-proxy's return code for what concerns load balancer health checks. The reason for this is: deleting nodes could end up causing an ingress outage should all endpoints simultaneously be running on said nodes.

The Kubernetes project recommends that cloud provider integration code configures load balancer health checks that target the service proxy's `healthz` port. If you are using or implementing your own virtual IP implementation, that people can use instead of kube-proxy, you should set up a similar health checking port with logic that matches the kube-proxy implementation.

Traffic to terminating endpoints

ⓘ FEATURE STATE: Kubernetes v1.28 [stable]

If the `ProxyTerminatingEndpoints` [feature gate](#) is enabled in kube-proxy and the traffic policy is `Local`, that node's kube-proxy uses a more complicated algorithm to select endpoints for a Service. With the feature enabled, kube-proxy checks if the node has local endpoints and whether or not all the local endpoints are marked as terminating. If there are local endpoints and **all** of them are terminating, then kube-proxy will forward traffic to those terminating endpoints. Otherwise, kube-proxy will always prefer forwarding traffic to endpoints that are not terminating.

This forwarding behavior for terminating endpoints exist to allow `NodePort` and `LoadBalancer` Services to gracefully drain connections when using `externalTrafficPolicy: Local`.

As a deployment goes through a rolling update, nodes backing a load balancer may transition from N to 0 replicas of that deployment. In some cases, external load balancers can send traffic to a node with 0 replicas in between health check probes. Routing traffic to terminating endpoints ensures that Node's that are scaling down Pods can gracefully receive and drain traffic to those terminating Pods. By the time the Pod completes termination, the external load balancer should have seen the node's health check failing and fully removed the node from the backend pool.

Traffic Distribution

ⓘ FEATURE STATE: Kubernetes v1.31 [beta]

The `spec.trafficDistribution` field within a Kubernetes Service allows you to express preferences for how traffic should be routed to Service endpoints. Implementations like kube-proxy use the `spec.trafficDistribution` field as a guideline. The behavior associated with a given preference may subtly differ between implementations.

PreferClose with kube-proxy

For kube-proxy, this means prioritizing sending traffic to endpoints within the same zone as the client. The EndpointSlice controller updates EndpointSlices with `hints` to communicate this preference, which kube-proxy then uses for routing decisions. If a client's zone does not have any available endpoints, traffic will be routed cluster-wide for that client.

In the absence of any value for `trafficDistribution`, the default routing strategy for kube-proxy is to distribute traffic to any endpoint in the cluster.

Comparison with `service.kubernetes.io/topology-mode: Auto`

The `trafficDistribution` field with `PreferClose` and the `service.kubernetes.io/topology-mode: Auto` annotation both aim to prioritize same-zone traffic. However, there are key differences in their approaches:

- `service.kubernetes.io/topology-mode: Auto` : Attempts to distribute traffic proportionally across zones based on allocatable CPU resources. This heuristic includes safeguards (such as the [fallback behavior](#) for small numbers of endpoints) and could lead to the feature being disabled in certain scenarios for load-balancing reasons. This approach sacrifices some predictability in favor of potential load balancing.
- `trafficDistribution: PreferClose` : This approach aims to be slightly simpler and more predictable: "If there are endpoints in the zone, they will receive all traffic for that zone, if there are no endpoints in a zone, the traffic will be distributed to other zones". While the approach may offer more predictability, it does mean that you are in control of managing a [potential overload](#).

If the `service.kubernetes.io/topology-mode` annotation is set to `Auto`, it will take precedence over `trafficDistribution`. (The annotation may be deprecated in the future in favour of the `trafficDistribution` field).

Interaction with Traffic Policies

When compared to the `trafficDistribution` field, the traffic policy fields (`externalTrafficPolicy` and `internalTrafficPolicy`) are meant to offer a stricter traffic locality requirements. Here's how `trafficDistribution` interacts with them:

- **Precedence of Traffic Policies:** For a given Service, if a traffic policy (`externalTrafficPolicy` or `internalTrafficPolicy`) is set to `Local`, it takes precedence over `trafficDistribution: PreferClose` for the

corresponding traffic type (external or internal, respectively).

- `trafficDistribution` Influence: For a given Service, if a traffic policy (`externalTrafficPolicy` or `internalTrafficPolicy`) is set to `cluster` (the default), or if the fields are not set, then `trafficDistribution: PreferClose` guides the routing behavior for the corresponding traffic type (external or internal, respectively). This means that an attempt will be made to route traffic to an endpoint that is in the same zone as the client.

Considerations for using traffic distribution control

- **Increased Probability of Overloaded Endpoints:** The `PreferClose` heuristic will attempt to route traffic to the closest healthy endpoints instead of spreading that traffic evenly across all endpoints. If you do not have a sufficient number of endpoints within a zone, they may become overloaded. This is especially likely if incoming traffic is not proportionally distributed across zones. To mitigate this, consider the following strategies:
 - [Pod Topology Spread Constraints](#): Use Pod Topology Spread Constraints to distribute your pods more evenly across zones.
 - **Zone-specific Deployments:** If you expect to see skewed traffic patterns, create a separate Deployment for each zone. This approach allows the separate workloads to scale independently. There are also workload management addons available from the ecosystem, outside the Kubernetes project itself, that can help here.
- **Implementation-specific behavior:** Each dataplane implementation may handle this field slightly differently. If you're using an implementation other than kube-proxy, refer the documentation specific to that implementation to understand how this field is being handled.

What's next

To learn more about Services, read [Connecting Applications with Services](#).

You can also:

- Read about [Services](#) as a concept
- Read about [Ingresses](#) as a concept
- Read the [API reference](#) for the Service API

10 - Setup tools

10.1 - Kubeadm

Kubeadm is a tool built to provide `kubeadm init` and `kubeadm join` as best-practice "fast paths" for creating Kubernetes clusters.



`kubeadm` performs the actions necessary to get a minimum viable cluster up and running. By design, it cares only about bootstrapping, not about provisioning machines. Likewise, installing various nice-to-have addons, like the Kubernetes Dashboard, monitoring solutions, and cloud-specific addons, is not in scope.

Instead, we expect higher-level and more tailored tooling to be built on top of `kubeadm`, and ideally, using `kubeadm` as the basis of all deployments will make it easier to create conformant clusters.

How to install

To install `kubeadm`, see the [installation guide](#).

What's next

- [kubeadm init](#) to bootstrap a Kubernetes control-plane node
- [kubeadm join](#) to bootstrap a Kubernetes worker node and join it to the cluster
- [kubeadm upgrade](#) to upgrade a Kubernetes cluster to a newer version
- [kubeadm config](#) if you initialized your cluster using `kubeadm v1.7.x` or lower, to configure your cluster for `kubeadm upgrade`
- [kubeadm token](#) to manage tokens for `kubeadm join`
- [kubeadm reset](#) to revert any changes made to this host by `kubeadm init` or `kubeadm join`
- [kubeadm certs](#) to manage Kubernetes certificates
- [kubeadm kubeconfig](#) to manage `kubeconfig` files
- [kubeadm version](#) to print the `kubeadm` version
- [kubeadm alpha](#) to preview a set of features made available for gathering feedback from the community

10.1.1 - Kubeadm Generated

10.1.1.1 -

kubeadm: easily bootstrap a secure Kubernetes cluster

Synopsis

```
KUBEADM
Easily bootstrap a secure Kubernetes cluster

Please give us feedback at:
https://github.com/kubernetes/kubeadm/issues
```

Example usage:

```
Create a two-machine cluster with one control-plane node
(which controls the cluster), and one worker node
(where your workloads, like Pods and Deployments run).
```

```
On the first machine:
```

```
control-plane# kubeadm init
```

```
On the second machine:
```

```
worker# kubeadm join &lt;arguments-returned-from-init&gt;
```

```
You can then repeat the second step on as many other machines as you
```

Options

-h, --help

help for kubeadm

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.2 -

Commands related to handling kubernetes certificates

Synopsis

Commands related to handling kubernetes certificates

```
kubeadm certs [flags]
```

Options

-h, --help

help for certs

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.2.1 -

Generate certificate keys

Synopsis

This command will print out a secure randomly-generated certificate key that can be used with the "init" command.

You can also use "kubeadm init --upload-certs" without specifying a certificate key and it will generate and print one for you.

```
kubeadm certs certificate-key [flags]
```

Options

-h, --help

help for certificate-key

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.2.2 -

Check certificates expiration for a Kubernetes cluster

Synopsis

Checks expiration for the certificates in the local PKI managed by kubeadm.

```
kubeadm certs check-expiration [flags]
```

Options

--allow-missing-template-keys Default: true

If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to golang and jsonpath output formats.

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save the certificates

--config string

Path to a kubeadm configuration file.

-h, --help

help for check-expiration

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

-o, --output string Default: "text"

Output format. One of: text|json|yaml|go-template|go-template-file|template|templatefile|jsonpath|jsonpath-as-json|jsonpath-file.

--show-managed-fields

If true, keep the managedFields when printing objects in JSON or YAML format.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.2.3 -

Generate keys and certificate signing requests

Synopsis

Generates keys and certificate signing requests (CSRs) for all the certificates required to run the control plane. This command also generates partial kubeconfig files with private key data in the "users > user > client-key-data" field, and for each kubeconfig file an accompanying ".csr" file is created.

This command is designed for use in [Kubeadm External CA Mode](#). It generates CSRs which you can then submit to your external certificate authority for signing.

The PEM encoded signed certificates should then be saved alongside the key files, using ".crt" as the file extension, or in the case of kubeconfig files, the PEM encoded signed certificate should be base64 encoded and added to the kubeconfig file in the "users > user > client-certificate-data" field.

```
kubeadm certs generate-csr [flags]
```

Examples

```
# The following command will generate keys and CSRs for all control
kubeadm certs generate-csr --kubeconfig-dir /tmp/etc-k8s --cert-dir
```

Options

--cert-dir string

The path where to save the certificates

--config string

Path to a kubeadm configuration file.

-h, --help

help for generate-csr

--kubeconfig-dir string Default: "/etc/kubernetes"

The path where to save the kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.2.4 -

Renew certificates for a Kubernetes cluster

Synopsis

This command is not meant to be run on its own. See list of available subcommands.

```
kubeadm certs renew [flags]
```

Options

-h, --help

help for renew

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.2.5 -

Renew the certificate embedded in the kubeconfig file for the admin to use and for kubeadm itself

Synopsis

Renew the certificate embedded in the kubeconfig file for the admin to use and for kubeadm itself.

Renewals run unconditionally, regardless of certificate expiration date; extra attributes such as SANs will be based on the existing file/certificates, there is no need to resupply them.

Renewal by default tries to use the certificate authority in the local PKI managed by kubeadm; as alternative it is possible to use K8s certificate API for certificate renewal, or as a last option, to generate a CSR request.

After renewal, in order to make changes effective, is required to restart control-plane components and eventually re-distribute the renewed certificate in case the file is used elsewhere.

```
kubeadm certs renew admin.conf [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save the certificates

--config string

Path to a kubeadm configuration file.

-h, --help

help for admin.conf

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.2.6 -

Renew all available certificates

Synopsis

Renew all known certificates necessary to run the control plane.

Renewals are run unconditionally, regardless of expiration date.

Renewals can also be run individually for more control.

```
kubeadm certs renew all [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save the certificates

--config string

Path to a kubeadm configuration file.

-h, --help

help for all

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.2.7 -

Renew the certificate the apiserver uses to access etcd

Synopsis

Renew the certificate the apiserver uses to access etcd.

Renewals run unconditionally, regardless of certificate expiration date; extra attributes such as SANs will be based on the existing file/certificates, there is no need to resupply them.

Renewal by default tries to use the certificate authority in the local PKI managed by kubeadm; as alternative it is possible to use K8s certificate API for certificate renewal, or as a last option, to generate a CSR request.

After renewal, in order to make changes effective, is required to restart control-plane components and eventually re-distribute the renewed certificate in case the file is used elsewhere.

```
kubeadm certs renew apiserver-etcd-client [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save the certificates

--config string

Path to a kubeadm configuration file.

-h, --help

help for apiserver-etcd-client

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.2.8 -

Renew the certificate for the API server to connect to kubelet

Synopsis

Renew the certificate for the API server to connect to kubelet.

Renewals run unconditionally, regardless of certificate expiration date; extra attributes such as SANs will be based on the existing file/certificates, there is no need to resupply them.

Renewal by default tries to use the certificate authority in the local PKI managed by kubeadm; as alternative it is possible to use K8s certificate API for certificate renewal, or as a last option, to generate a CSR request.

After renewal, in order to make changes effective, is required to restart control-plane components and eventually re-distribute the renewed certificate in case the file is used elsewhere.

```
kubeadm certs renew apiserver-kubelet-client [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save the certificates

--config string

Path to a kubeadm configuration file.

-h, --help

help for apiserver-kubelet-client

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.2.9 -

Renew the certificate for serving the Kubernetes API

Synopsis

Renew the certificate for serving the Kubernetes API.

Renewals run unconditionally, regardless of certificate expiration date; extra attributes such as SANs will be based on the existing file/certificates, there is no need to resupply them.

Renewal by default tries to use the certificate authority in the local PKI managed by kubeadm; as alternative it is possible to use K8s certificate API for certificate renewal, or as a last option, to generate a CSR request.

After renewal, in order to make changes effective, is required to restart control-plane components and eventually re-distribute the renewed certificate in case the file is used elsewhere.

```
kubeadm certs renew apiserver [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save the certificates

--config string

Path to a kubeadm configuration file.

-h, --help

help for apiserver

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.2.10 -

Renew the certificate embedded in the kubeconfig file for the controller manager to use

Synopsis

Renew the certificate embedded in the kubeconfig file for the controller manager to use.

Renewals run unconditionally, regardless of certificate expiration date; extra attributes such as SANs will be based on the existing file/certificates, there is no need to resupply them.

Renewal by default tries to use the certificate authority in the local PKI managed by kubeadm; as alternative it is possible to use K8s certificate API for certificate renewal, or as a last option, to generate a CSR request.

After renewal, in order to make changes effective, is required to restart control-plane components and eventually re-distribute the renewed certificate in case the file is used elsewhere.

```
kubeadm certs renew controller-manager.conf [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save the certificates

--config string

Path to a kubeadm configuration file.

-h, --help

help for controller-manager.conf

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.2.11 -

Renew the certificate for liveness probes to healthcheck etcd

Synopsis

Renew the certificate for liveness probes to healthcheck etcd.

Renewals run unconditionally, regardless of certificate expiration date; extra attributes such as SANs will be based on the existing file/certificates, there is no need to resupply them.

Renewal by default tries to use the certificate authority in the local PKI managed by kubeadm; as alternative it is possible to use K8s certificate API for certificate renewal, or as a last option, to generate a CSR request.

After renewal, in order to make changes effective, is required to restart control-plane components and eventually re-distribute the renewed certificate in case the file is used elsewhere.

```
kubeadm certs renew etcd-healthcheck-client [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save the certificates

--config string

Path to a kubeadm configuration file.

-h, --help

help for etcd-healthcheck-client

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.2.12 -

Renew the certificate for etcd nodes to communicate with each other

Synopsis

Renew the certificate for etcd nodes to communicate with each other.

Renewals run unconditionally, regardless of certificate expiration date; extra attributes such as SANs will be based on the existing file/certificates, there is no need to resupply them.

Renewal by default tries to use the certificate authority in the local PKI managed by kubeadm; as alternative it is possible to use K8s certificate API for certificate renewal, or as a last option, to generate a CSR request.

After renewal, in order to make changes effective, is required to restart control-plane components and eventually re-distribute the renewed certificate in case the file is used elsewhere.

```
kubeadm certs renew etcd-peer [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save the certificates

--config string

Path to a kubeadm configuration file.

-h, --help

help for etcd-peer

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.2.13 -

Renew the certificate for serving etcd

Synopsis

Renew the certificate for serving etcd.

Renewals run unconditionally, regardless of certificate expiration date; extra attributes such as SANs will be based on the existing file/certificates, there is no need to resupply them.

Renewal by default tries to use the certificate authority in the local PKI managed by kubeadm; as alternative it is possible to use K8s certificate API for certificate renewal, or as a last option, to generate a CSR request.

After renewal, in order to make changes effective, is required to restart control-plane components and eventually re-distribute the renewed certificate in case the file is used elsewhere.

```
kubeadm certs renew etcd-server [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save the certificates

--config string

Path to a kubeadm configuration file.

-h, --help

help for etcd-server

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.2.14 -

Renew the certificate for the front proxy client

Synopsis

Renew the certificate for the front proxy client.

Renewals run unconditionally, regardless of certificate expiration date; extra attributes such as SANs will be based on the existing file/certificates, there is no need to resupply them.

Renewal by default tries to use the certificate authority in the local PKI managed by kubeadm; as alternative it is possible to use K8s certificate API for certificate renewal, or as a last option, to generate a CSR request.

After renewal, in order to make changes effective, is required to restart control-plane components and eventually re-distribute the renewed certificate in case the file is used elsewhere.

```
kubeadm certs renew front-proxy-client [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save the certificates

--config string

Path to a kubeadm configuration file.

-h, --help

help for front-proxy-client

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.2.15 -

Renew the certificate embedded in the kubeconfig file for the scheduler manager to use

Synopsis

Renew the certificate embedded in the kubeconfig file for the scheduler manager to use.

Renewals run unconditionally, regardless of certificate expiration date; extra attributes such as SANs will be based on the existing file/certificates, there is no need to resupply them.

Renewal by default tries to use the certificate authority in the local PKI managed by kubeadm; as alternative it is possible to use K8s certificate API for certificate renewal, or as a last option, to generate a CSR request.

After renewal, in order to make changes effective, is required to restart control-plane components and eventually re-distribute the renewed certificate in case the file is used elsewhere.

```
kubeadm certs renew scheduler.conf [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save the certificates

--config string

Path to a kubeadm configuration file.

-h, --help

help for scheduler.conf

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.2.16 -

Renew the certificate embedded in the kubeconfig file for the super-admin

Synopsis

Renew the certificate embedded in the kubeconfig file for the super-admin.

Renewals run unconditionally, regardless of certificate expiration date; extra attributes such as SANs will be based on the existing file/certificates, there is no need to resupply them.

Renewal by default tries to use the certificate authority in the local PKI managed by kubeadm; as alternative it is possible to use K8s certificate API for certificate renewal, or as a last option, to generate a CSR request.

After renewal, in order to make changes effective, is required to restart control-plane components and eventually re-distribute the renewed certificate in case the file is used elsewhere.

```
kubeadm certs renew super-admin.conf [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save the certificates

--config string

Path to a kubeadm configuration file.

-h, --help

help for super-admin.conf

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.3 -

Output shell completion code for the specified shell (bash or zsh)

Synopsis

Output shell completion code for the specified shell (bash or zsh). The shell code must be evaluated to provide interactive completion of kubeadm commands. This can be done by sourcing it from the .bash_profile.

Note: this requires the bash-completion framework.

To install it on Mac use homebrew: \$ brew install bash-completion Once installed, bash_completion must be evaluated. This can be done by adding the following line to the .bash_profile \$ source \$(brew --prefix)/etc/bash_completion

If bash-completion is not installed on Linux, please install the 'bash-completion' package via your distribution's package manager.

Note for zsh users: [1] zsh completions are only supported in versions of zsh >= 5.2

```
kubeadm completion SHELL [flags]
```

Examples

```
# Install bash completion on a Mac using homebrew
brew install bash-completion
printf "\n# Bash completion support\nsource $(brew --prefix)/etc/bash_completion
source $HOME/.bash_profile

# Load the kubeadm completion code for bash into the current shell
source <(kubeadm completion bash)

# Write bash completion code to a file and source it from .bash_profile
kubeadm completion bash > ~/.kube/kubeadm_completion.bash.inc
printf "\n# Kubeadm shell completion\nsource '$HOME/.kube/kubeadm_completion.bash.inc'
source $HOME/.bash_profile

# Load the kubeadm completion code for zsh[1] into the current shell
source <(kubeadm completion zsh)
```

Options

-h, --help

help for completion

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.4 -

Manage configuration for a kubeadm cluster persisted in a ConfigMap in the cluster

Synopsis

There is a ConfigMap in the kube-system namespace called "kubeadm-config" that kubeadm uses to store internal configuration about the cluster. kubeadm CLI v1.8.0+ automatically creates this ConfigMap with the config used with 'kubeadm init', but if you initialized your cluster using kubeadm v1.7.x or lower, you must use the 'kubeadm init phase upload-config' command to create this ConfigMap. This is required so that 'kubeadm upgrade' can configure your upgraded cluster correctly.

```
kubeadm config [flags]
```

Options

-h, --help

help for config

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.4.1 -

Interact with container images used by kubeadm

Synopsis

Interact with container images used by kubeadm

```
kubeadm config images [flags]
```

Options

-h, --help

help for images

Options inherited from parent commands

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.4.2 -

Print a list of images kubeadm will use. The configuration file is used in case any images or image repositories are customized

Synopsis

Print a list of images kubeadm will use. The configuration file is used in case any images or image repositories are customized

```
kubeadm config images list [flags]
```

Options

--allow-missing-template-keys Default: true

If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to golang and jsonpath output formats.

--config string

Path to a kubeadm configuration file.

--feature-gates string

A set of key=value pairs that describe feature gates for various features. Options are:

ControlPlaneKubeletLocalMode=true|false (ALPHA - default=false)

EtcdLearnerMode=true|false (BETA - default=true)

PublicKeysECDSA=true|false (DEPRECATED - default=false)

RootlessControlPlane=true|false (ALPHA - default=false)

WaitForAllControlPlaneComponents=true|false (ALPHA - default=false)

-h, --help

help for list

--image-repository string Default: "registry.k8s.io"

Choose a container registry to pull control plane images from

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

-o, --output string Default: "text"

Output format. One of: text|json|yaml|go-template|go-template-file|template|templatefile|jsonpath|jsonpath-as-json|jsonpath-file.

--show-managed-fields

If true, keep the managedFields when printing objects in JSON or YAML format.

Options inherited from parent commands

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.4.3 -

Pull images used by kubeadm

Synopsis

Pull images used by kubeadm

```
kubeadm config images pull [flags]
```

Options

--config string

Path to a kubeadm configuration file.

--cri-socket string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

--feature-gates string

A set of key=value pairs that describe feature gates for various features. Options are:
ControlPlaneKubeletLocalMode=true|false (ALPHA - default=false)
EtcdLearnerMode=true|false (BETA - default=true)
PublicKeysECDSA=true|false (DEPRECATED - default=false)
RootlessControlPlane=true|false (ALPHA - default=false)
WaitForAllControlPlaneComponents=true|false (ALPHA - default=false)

-h, --help

help for pull

--image-repository string Default: "registry.k8s.io"

Choose a container registry to pull control plane images from

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

Options inherited from parent commands

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.4.4 -

Read an older version of the kubeadm configuration API types from a file, and output the similar config object for the newer version

Synopsis

This command lets you convert configuration objects of older versions to the latest supported version, locally in the CLI tool without ever touching anything in the cluster. In this version of kubeadm, the following API versions are supported:

- `kubeadm.k8s.io/v1beta4`

Further, kubeadm can only write out config of version "kubeadm.k8s.io/v1beta4", but read both types. So regardless of what version you pass to the `--old-config` parameter here, the API object will be read, deserialized, defaulted, converted, validated, and re-serialized when written to `stdout` or `--new-config` if specified.

In other words, the output of this command is what kubeadm actually would read internally if you submitted this file to "kubeadm init"

```
kubeadm config migrate [flags]
```

Options

`--allow-experimental-api`

Allow migration to experimental, unreleased APIs.

`-h, --help`

help for migrate

`--new-config string`

Path to the resulting equivalent kubeadm config file using the new API version. Optional, if not specified output will be sent to `STDOUT`.

`--old-config string`

Path to the kubeadm config file that is using an old API version and should be converted. This flag is mandatory.

Options inherited from parent commands

`--kubeconfig string` Default: `"/etc/kubernetes/admin.conf"`

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

`--rootfs string`

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.4.5 -

Print configuration

Synopsis

This command prints configurations for subcommands provided. For details, see: <https://pkg.go.dev/k8s.io/kubernetes/cmd/kubeadm/app/apis/kubeadm#section-directories>

```
kubeadm config print [flags]
```

Options

-h, --help

help for print

Options inherited from parent commands

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.4.6 -

Print default init configuration, that can be used for 'kubeadm init'

Synopsis

This command prints objects such as the default init configuration that is used for 'kubeadm init'.

Note that sensitive values like the Bootstrap Token fields are replaced with placeholder values like "abcdef.0123456789abcdef" in order to pass validation but not perform the real computation for creating a token.

```
kubeadm config print init-defaults [flags]
```

Options

--component-configs strings

A comma-separated list for component config API objects to print the default values for. Available values: [KubeProxyConfiguration KubeletConfiguration]. If this flag is not set, no component configs will be printed.

-h, --help

help for init-defaults

Options inherited from parent commands

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.4.7 -

Print default join configuration, that can be used for 'kubeadm join'

Synopsis

This command prints objects such as the default join configuration that is used for 'kubeadm join'.

Note that sensitive values like the Bootstrap Token fields are replaced with placeholder values like "abcdef.0123456789abcdef" in order to pass validation but not perform the real computation for creating a token.

```
kubeadm config print join-defaults [flags]
```

Options

-h, --help

help for join-defaults

Options inherited from parent commands

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.4.8 -

Print default reset configuration, that can be used for 'kubeadm reset'

Synopsis

This command prints objects such as the default reset configuration that is used for 'kubeadm reset'.

Note that sensitive values like the Bootstrap Token fields are replaced with placeholder values like "abcdef.0123456789abcdef" in order to pass validation but not perform the real computation for creating a token.

```
kubeadm config print reset-defaults [flags]
```

Options

-h, --help

help for reset-defaults

Options inherited from parent commands

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.4.9 -

Print default upgrade configuration, that can be used for 'kubeadm upgrade'

Synopsis

This command prints objects such as the default upgrade configuration that is used for 'kubeadm upgrade'.

Note that sensitive values like the Bootstrap Token fields are replaced with placeholder values like "abcdef.0123456789abcdef" in order to pass validation but not perform the real computation for creating a token.

```
kubeadm config print upgrade-defaults [flags]
```

Options

-h, --help

help for upgrade-defaults

Options inherited from parent commands

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.4.10 -

Read a file containing the kubeadm configuration API and report any validation problems

Synopsis

This command lets you validate a kubeadm configuration API file and report any warnings and errors. If there are no errors the exit status will be zero, otherwise it will be non-zero. Any unmarshaling problems such as unknown API fields will trigger errors. Unknown API versions and fields with invalid values will also trigger errors. Any other errors or warnings may be reported depending on contents of the input file.

In this version of kubeadm, the following API versions are supported:

- `kubeadm.k8s.io/v1beta4`

```
kubeadm config validate [flags]
```

Options

`--allow-experimental-api`

Allow validation of experimental, unreleased APIs.

`--config string`

Path to a kubeadm configuration file.

`-h, --help`

help for validate

Options inherited from parent commands

`--kubeconfig string` Default: `"/etc/kubernetes/admin.conf"`

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

`--rootfs string`

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5 -

Run this command in order to set up the Kubernetes control plane

Synopsis

Run this command in order to set up the Kubernetes control plane

The "init" command executes the following phases:

preflight	Run pre-flight checks
certs	Certificate generation
/ca	Generate the self-signed Kubernetes CA
/apiserver	Generate the certificate for serving
/apiserver-kubelet-client	Generate the certificate for the API
/front-proxy-ca	Generate the self-signed CA to provision the front proxy
/front-proxy-client	Generate the certificate for the front proxy
/etcd-ca	Generate the self-signed CA to provision etcd
/etcd-server	Generate the certificate for serving etcd
/etcd-peer	Generate the certificate for etcd nodes
/etcd-healthcheck-client	Generate the certificate for liveness probes
/apiserver-etcd-client	Generate the certificate for the apiserver to talk to etcd
/sa	Generate a private key for signing service accounts
kubeconfig	Generate all kubeconfig files necessary
/admin	Generate a kubeconfig file for the admin user
/super-admin	Generate a kubeconfig file for the superuser
/kubelet	Generate a kubeconfig file for the kubelet
/controller-manager	Generate a kubeconfig file for the controller manager
/scheduler	Generate a kubeconfig file for the scheduler
etcd	Generate static Pod manifest file for local etcd
/local	Generate the static Pod manifest file
control-plane	Generate all static Pod manifest files
/apiserver	Generates the kube-apiserver static Pod manifest
/controller-manager	Generates the kube-controller-manager static Pod manifest
/scheduler	Generates the kube-scheduler static Pod manifest
kubelet-start	Write kubelet settings and (re)start the kubelet
upload-config	Upload the kubeadm and kubelet configuration
/kubeadm	Upload the kubeadm ClusterConfiguration
/kubelet	Upload the kubelet component config to the kubelet
upload-certs	Upload certificates to kubeadm-certs
mark-control-plane	Mark a node as a control-plane
bootstrap-token	Generates bootstrap tokens used to join the node to the cluster
kubelet-finalize	Updates settings relevant to the kubelet
/enable-client-cert-rotation	Enable kubelet client certificate rotation
/experimental-cert-rotation	Enable kubelet client certificate rotation using the experimental API
addon	Install required addons for passing container ports
/coredns	Install the CoreDNS addon to a Kubernetes cluster
/kube-proxy	Install the kube-proxy addon to a Kubernetes cluster
show-join-command	Show the join command for control-plane nodes

kubeadm init [flags]

Options

--apiserver-advertise-address string

The IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--apiserver-bind-port int32 Default: 6443

Port for the API Server to bind to.

--apiserver-cert-extra-sans strings

Optional extra Subject Alternative Names (SANs) to use for the API Server serving certificate. Can be both IP addresses and DNS names.

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--certificate-key string

Key used to encrypt the control-plane certificates in the `kubeadm-certs` Secret. The certificate key is a hex encoded string that is an AES key of size 32 bytes.

--config string

Path to a `kubeadm` configuration file.

--control-plane-endpoint string

Specify a stable IP address or DNS name for the control plane.

--cri-socket string

Path to the CRI socket to connect. If empty `kubeadm` will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

--dry-run

Don't apply any changes; just output what would be done.

--feature-gates string

A set of key=value pairs that describe feature gates for various features. Options are:
`ControlPlaneKubeletLocalMode=true|false` (ALPHA - default=false)
`EtcdLearnerMode=true|false` (BETA - default=true)
`PublicKeysECDSA=true|false` (DEPRECATED - default=false)
`RootlessControlPlane=true|false` (ALPHA - default=false)
`WaitForAllControlPlaneComponents=true|false` (ALPHA - default=false)

-h, --help

help for init

--ignore-preflight-errors strings

A list of checks whose errors will be shown as warnings. Example: `'IsPrivilegedUser,Swap'`. Value 'all' ignores errors from all checks.

--image-repository string Default: "registry.k8s.io"

Choose a container registry to pull control plane images from

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

--node-name string

Specify the node name.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

--pod-network-cidr string

Specify range of IP addresses for the pod network. If set, the control plane will automatically allocate CIDRs for every node.

--service-cidr string Default: "10.96.0.0/12"

Use alternative range of IP address for service VIPs.

--service-dns-domain string Default: "cluster.local"

Use alternative domain for services, e.g. "myorg.internal".

--skip-certificate-key-print

Don't print the key used to encrypt the control-plane certificates.

--skip-phases strings

List of phases to be skipped

--skip-token-print

Skip printing of the default bootstrap token generated by 'kubeadm init'.

--token string

The token to use for establishing bidirectional trust between nodes and control-plane nodes. The format is [a-z0-9]{6}.[a-z0-9]{16} - e.g. abcdef.0123456789abcdef

--token-ttl duration Default: 24h0m0s

The duration before the token is automatically deleted (e.g. 1s, 2m, 3h). If set to '0', the token will never expire

--upload-certs

Upload control-plane certificates to the kubeadm-certs Secret.

Options inherited from parent commands

`--rootfs` string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.1 -

Use this command to invoke single phase of the init workflow

Synopsis

Use this command to invoke single phase of the init workflow

Options

`-h, --help`

help for phase

Options inherited from parent commands

`--rootfs string`

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.2 -

Install required addons for passing conformance tests

Synopsis

This command is not meant to be run on its own. See list of available subcommands.

```
kubeadm init phase addon [flags]
```

Options

-h, --help

help for addon

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.3 -

Install all the addons

Synopsis

Install all the addons

```
kubeadm init phase addon all [flags]
```

Options

--apiserver-advertise-address string

The IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--apiserver-bind-port int32 Default: 6443

Port for the API Server to bind to.

--config string

Path to a kubeadm configuration file.

--control-plane-endpoint string

Specify a stable IP address or DNS name for the control plane.

--dry-run

Don't apply any changes; just output what would be done.

--feature-gates string

A set of key=value pairs that describe feature gates for various features. Options are:

ControlPlaneKubeletLocalMode=true|false (ALPHA - default=false)

EtcdLearnerMode=true|false (BETA - default=true)

PublicKeysECDSA=true|false (DEPRECATED - default=false)

RootlessControlPlane=true|false (ALPHA - default=false)

WaitForAllControlPlaneComponents=true|false (ALPHA - default=false)

-h, --help

help for all

--image-repository string Default: "registry.k8s.io"

Choose a container registry to pull control plane images from

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

--pod-network-cidr string

Specify range of IP addresses for the pod network. If set, the control plane will automatically allocate CIDRs for every node.

--service-cidr string Default: "10.96.0.0/12"

Use alternative range of IP address for service VIPs.

--service-dns-domain string Default: "cluster.local"

Use alternative domain for services, e.g. "myorg.internal".

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.4 -

Install the CoreDNS addon to a Kubernetes cluster

Synopsis

Install the CoreDNS addon components via the API server. Please note that although the DNS server is deployed, it will not be scheduled until CNI is installed.

```
kubeadm init phase addon coredns [flags]
```

Options

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

--feature-gates string

A set of key=value pairs that describe feature gates for various features. Options are:

ControlPlaneKubeletLocalMode=true|false (ALPHA - default=false)

EtcdLearnerMode=true|false (BETA - default=true)

PublicKeysECDSA=true|false (DEPRECATED - default=false)

RootlessControlPlane=true|false (ALPHA - default=false)

WaitForAllControlPlaneComponents=true|false (ALPHA - default=false)

-h, --help

help for coredns

--image-repository string Default: "registry.k8s.io"

Choose a container registry to pull control plane images from

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

--print-manifest

Print the addon manifests to STDOUT instead of installing them

--service-cidr string Default: "10.96.0.0/12"

Use alternative range of IP address for service VIPs.

--service-dns-domain string Default: "cluster.local"

Use alternative domain for services, e.g. "myorg.internal".

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.5 -

Install the kube-proxy addon to a Kubernetes cluster

Synopsis

Install the kube-proxy addon components via the API server.

```
kubeadm init phase addon kube-proxy [flags]
```

Options

--apiserver-advertise-address string

The IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--apiserver-bind-port int32 Default: 6443

Port for the API Server to bind to.

--config string

Path to a kubeadm configuration file.

--control-plane-endpoint string

Specify a stable IP address or DNS name for the control plane.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for kube-proxy

--image-repository string Default: "registry.k8s.io"

Choose a container registry to pull control plane images from

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

--pod-network-cidr string

Specify range of IP addresses for the pod network. If set, the control plane will automatically allocate CIDRs for every node.

--print-manifest

Print the addon manifests to STDOUT instead of installing them

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.6 -

Generates bootstrap tokens used to join a node to a cluster

Synopsis

Bootstrap tokens are used for establishing bidirectional trust between a node joining the cluster and a control-plane node.

This command makes all the configurations required to make bootstrap tokens work and then creates an initial token.

```
kubeadm init phase bootstrap-token [flags]
```

Examples

```
# Make all the bootstrap token configurations and create an initial
# equivalent to what generated by kubeadm init.
kubeadm init phase bootstrap-token
```

Options

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for bootstrap-token

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--skip-token-print

Skip printing of the default bootstrap token generated by 'kubeadm init'.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.7 -

Certificate generation

Synopsis

This command is not meant to be run on its own. See list of available subcommands.

```
kubeadm init phase certs [flags]
```

Options

-h, --help

help for certs

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.8 -

Generate all certificates

Synopsis

Generate all certificates

```
kubeadm init phase certs all [flags]
```

Options

--apiserver-advertise-address string

The IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--apiserver-cert-extra-sans strings

Optional extra Subject Alternative Names (SANs) to use for the API Server serving certificate. Can be both IP addresses and DNS names.

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--control-plane-endpoint string

Specify a stable IP address or DNS name for the control plane.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for all

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

--service-cidr string Default: "10.96.0.0/12"

Use alternative range of IP address for service VIPs.

--service-dns-domain string Default: "cluster.local"

Use alternative domain for services, e.g. "myorg.internal".

Options inherited from parent commands

`--rootfs` string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.9 -

Generate the certificate the apiserver uses to access etcd

Synopsis

Generate the certificate the apiserver uses to access etcd, and save them into `apiserver-etcd-client.crt` and `apiserver-etcd-client.key` files.

If both files already exist, `kubeadm` skips the generation step and existing files will be used.

```
kubeadm init phase certs apiserver-etcd-client [flags]
```

Options

`--cert-dir` string Default: `"/etc/kubernetes/pki"`

The path where to save and store the certificates.

`--config` string

Path to a `kubeadm` configuration file.

`--dry-run`

Don't apply any changes; just output what would be done.

`-h, --help`

help for `apiserver-etcd-client`

`--kubernetes-version` string Default: `"stable-1"`

Choose a specific Kubernetes version for the control plane.

Options inherited from parent commands

`--rootfs` string

The path to the 'real' host root filesystem. This will cause `kubeadm` to chroot into the provided path.

10.1.1.5.10 -

Generate the certificate for the API server to connect to kubelet

Synopsis

Generate the certificate for the API server to connect to kubelet, and save them into `apiserver-kubelet-client.crt` and `apiserver-kubelet-client.key` files.

If both files already exist, `kubeadm` skips the generation step and existing files will be used.

```
kubeadm init phase certs apiserver-kubelet-client [flags]
```

Options

`--cert-dir` string Default: `"/etc/kubernetes/pki"`

The path where to save and store the certificates.

`--config` string

Path to a `kubeadm` configuration file.

`--dry-run`

Don't apply any changes; just output what would be done.

`-h, --help`

help for `apiserver-kubelet-client`

`--kubernetes-version` string Default: `"stable-1"`

Choose a specific Kubernetes version for the control plane.

Options inherited from parent commands

`--rootfs` string

The path to the 'real' host root filesystem. This will cause `kubeadm` to chroot into the provided path.

10.1.1.5.11 -

Generate the certificate for serving the Kubernetes API

Synopsis

Generate the certificate for serving the Kubernetes API, and save them into `apiserver.crt` and `apiserver.key` files.

If both files already exist, `kubeadm` skips the generation step and existing files will be used.

```
kubeadm init phase certs apiserver [flags]
```

Options

`--apiserver-advertise-address` string

The IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

`--apiserver-cert-extra-sans` strings

Optional extra Subject Alternative Names (SANs) to use for the API Server serving certificate. Can be both IP addresses and DNS names.

`--cert-dir` string Default: `"/etc/kubernetes/pki"`

The path where to save and store the certificates.

`--config` string

Path to a `kubeadm` configuration file.

`--control-plane-endpoint` string

Specify a stable IP address or DNS name for the control plane.

`--dry-run`

Don't apply any changes; just output what would be done.

`-h, --help`

help for apiserver

`--kubernetes-version` string Default: `"stable-1"`

Choose a specific Kubernetes version for the control plane.

`--service-cidr` string Default: `"10.96.0.0/12"`

Use alternative range of IP address for service VIPs.

--service-dns-domain string Default: "cluster.local"

Use alternative domain for services, e.g. "myorg.internal".

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.12 -

Generate the self-signed Kubernetes CA to provision identities for other Kubernetes components

Synopsis

Generate the self-signed Kubernetes CA to provision identities for other Kubernetes components, and save them into ca.crt and ca.key files.

If both files already exist, kubeadm skips the generation step and existing files will be used.

```
kubeadm init phase certs ca [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for ca

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.13 -

Generate the self-signed CA to provision identities for etcd

Synopsis

Generate the self-signed CA to provision identities for etcd, and save them into etcd/ca.crt and etcd/ca.key files.

If both files already exist, kubeadm skips the generation step and existing files will be used.

```
kubeadm init phase certs etcd-ca [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for etcd-ca

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.14 -

Generate the certificate for liveness probes to healthcheck etcd

Synopsis

Generate the certificate for liveness probes to healthcheck etcd, and save them into etcd/healthcheck-client.crt and etcd/healthcheck-client.key files.

If both files already exist, kubeadm skips the generation step and existing files will be used.

```
kubeadm init phase certs etcd-healthcheck-client [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for etcd-healthcheck-client

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.15 -

Generate the certificate for etcd nodes to communicate with each other

Synopsis

Generate the certificate for etcd nodes to communicate with each other, and save them into etcd/peer.crt and etcd/peer.key files.

Default SANs are localhost, 127.0.0.1, 127.0.0.1, ::1

If both files already exist, kubeadm skips the generation step and existing files will be used.

```
kubeadm init phase certs etcd-peer [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for etcd-peer

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.16 -

Generate the certificate for serving etcd

Synopsis

Generate the certificate for serving etcd, and save them into etcd/server.crt and etcd/server.key files.

Default SANs are localhost, 127.0.0.1, 127.0.0.1, ::1

If both files already exist, kubeadm skips the generation step and existing files will be used.

```
kubeadm init phase certs etcd-server [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for etcd-server

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.17 -

Generate the self-signed CA to provision identities for front proxy

Synopsis

Generate the self-signed CA to provision identities for front proxy, and save them into front-proxy-ca.crt and front-proxy-ca.key files.

If both files already exist, kubeadm skips the generation step and existing files will be used.

```
kubeadm init phase certs front-proxy-ca [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for front-proxy-ca

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.18 -

Generate the certificate for the front proxy client

Synopsis

Generate the certificate for the front proxy client, and save them into front-proxy-client.crt and front-proxy-client.key files.

If both files already exist, kubeadm skips the generation step and existing files will be used.

```
kubeadm init phase certs front-proxy-client [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for front-proxy-client

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.19 -

Generate a private key for signing service account tokens along with its public key

Synopsis

Generate the private key for signing service account tokens along with its public key, and save them into `sa.key` and `sa.pub` files.

If both files already exist, `kubeadm` skips the generation step and existing files will be used.

```
kubeadm init phase certs sa [flags]
```

Options

`--cert-dir` string Default: `"/etc/kubernetes/pki"`

The path where to save and store the certificates.

`--config` string

Path to a `kubeadm` configuration file.

`--dry-run`

Don't apply any changes; just output what would be done.

`-h, --help`

help for `sa`

`--kubernetes-version` string Default: `"stable-1"`

Choose a specific Kubernetes version for the control plane.

Options inherited from parent commands

`--rootfs` string

The path to the 'real' host root filesystem. This will cause `kubeadm` to chroot into the provided path.

10.1.1.5.20 -

Generate all static Pod manifest files necessary to establish the control plane

Synopsis

This command is not meant to be run on its own. See list of available subcommands.

```
kubeadm init phase control-plane [flags]
```

Options

-h, --help

help for control-plane

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.21 -

Generate all static Pod manifest files

Synopsis

Generate all static Pod manifest files

```
kubeadm init phase control-plane all [flags]
```

Examples

```
# Generates all static Pod manifest files for control plane components
# functionally equivalent to what is generated by kubeadm init.
kubeadm init phase control-plane all

# Generates all static Pod manifest files using options read from a
kubeadm init phase control-plane all --config config.yaml
```

Options

--apiserver-advertise-address string

The IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--apiserver-bind-port int32 Default: 6443

Port for the API Server to bind to.

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--control-plane-endpoint string

Specify a stable IP address or DNS name for the control plane.

--dry-run

Don't apply any changes; just output what would be done.

--feature-gates string

A set of key=value pairs that describe feature gates for various features. Options are:

ControlPlaneKubeletLocalMode=true|false (ALPHA - default=false)

EtcdLearnerMode=true|false (BETA - default=true)

PublicKeysECDSA=true|false (DEPRECATED - default=false)

RootlessControlPlane=true|false (ALPHA - default=false)

WaitForAllControlPlaneComponents=true|false (ALPHA - default=false)

-h, --help

help for all

--image-repository string Default: "registry.k8s.io"

Choose a container registry to pull control plane images from

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

--pod-network-cidr string

Specify range of IP addresses for the pod network. If set, the control plane will automatically allocate CIDRs for every node.

--service-cidr string Default: "10.96.0.0/12"

Use alternative range of IP address for service VIPs.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.22 -

Generates the kube-apiserver static Pod manifest

Synopsis

Generates the kube-apiserver static Pod manifest

```
kubeadm init phase control-plane apiserver [flags]
```

Options

--apiserver-advertise-address string

The IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--apiserver-bind-port int32 Default: 6443

Port for the API Server to bind to.

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--control-plane-endpoint string

Specify a stable IP address or DNS name for the control plane.

--dry-run

Don't apply any changes; just output what would be done.

--feature-gates string

A set of key=value pairs that describe feature gates for various features. Options are:

ControlPlaneKubeletLocalMode=true|false (ALPHA - default=false)

EtcdLearnerMode=true|false (BETA - default=true)

PublicKeysECDSA=true|false (DEPRECATED - default=false)

RootlessControlPlane=true|false (ALPHA - default=false)

WaitForAllControlPlaneComponents=true|false (ALPHA - default=false)

-h, --help

help for apiserver

--image-repository string Default: "registry.k8s.io"

Choose a container registry to pull control plane images from

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

--service-cidr string Default: "10.96.0.0/12"

Use alternative range of IP address for service VIPs.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.23 -

Generates the kube-controller-manager static Pod manifest

Synopsis

Generates the kube-controller-manager static Pod manifest

```
kubeadm init phase control-plane controller-manager [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for controller-manager

--image-repository string Default: "registry.k8s.io"

Choose a container registry to pull control plane images from

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

--pod-network-cidr string

Specify range of IP addresses for the pod network. If set, the control plane will automatically allocate CIDRs for every node.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.24 -

Generates the kube-scheduler static Pod manifest

Synopsis

Generates the kube-scheduler static Pod manifest

```
kubeadm init phase control-plane scheduler [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for scheduler

--image-repository string Default: "registry.k8s.io"

Choose a container registry to pull control plane images from

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.25 -

Generate static Pod manifest file for local etcd

Synopsis

This command is not meant to be run on its own. See list of available subcommands.

```
kubeadm init phase etcd [flags]
```

Options

-h, --help

help for etcd

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.26 -

Generate the static Pod manifest file for a local, single-node local etcd instance

Synopsis

Generate the static Pod manifest file for a local, single-node local etcd instance

```
kubeadm init phase etcd local [flags]
```

Examples

```
# Generates the static Pod manifest file for etcd, functionally
# equivalent to what is generated by kubeadm init.
kubeadm init phase etcd local

# Generates the static Pod manifest file for etcd using options
# read from a configuration file.
kubeadm init phase etcd local --config config.yaml
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for local

--image-repository string Default: "registry.k8s.io"

Choose a container registry to pull control plane images from

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

Options inherited from parent commands

`--rootfs` string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.27 -

Generate all kubeconfig files necessary to establish the control plane and the admin kubeconfig file

Synopsis

This command is not meant to be run on its own. See list of available subcommands.

```
kubeadm init phase kubeconfig [flags]
```

Options

-h, --help

help for kubeconfig

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.28 -

Generate a kubeconfig file for the admin to use and for kubeadm itself

Synopsis

Generate the kubeconfig file for the admin and for kubeadm itself, and save it to admin.conf file.

```
kubeadm init phase kubeconfig admin [flags]
```

Options

--apiserver-advertise-address string

The IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--apiserver-bind-port int32 Default: 6443

Port for the API Server to bind to.

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--control-plane-endpoint string

Specify a stable IP address or DNS name for the control plane.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for admin

--kubeconfig-dir string Default: "/etc/kubernetes"

The path where to save the kubeconfig file.

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.29 -

Generate all kubeconfig files

Synopsis

Generate all kubeconfig files

```
kubeadm init phase kubeconfig all [flags]
```

Options

--apiserver-advertise-address string

The IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--apiserver-bind-port int32 Default: 6443

Port for the API Server to bind to.

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--control-plane-endpoint string

Specify a stable IP address or DNS name for the control plane.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for all

--kubeconfig-dir string Default: "/etc/kubernetes"

The path where to save the kubeconfig file.

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

--node-name string

Specify the node name.

Options inherited from parent commands

`--rootfs` string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.30 -

Generate a kubeconfig file for the controller manager to use

Synopsis

Generate the kubeconfig file for the controller manager to use and save it to controller-manager.conf file

```
kubeadm init phase kubeconfig controller-manager [flags]
```

Options

--apiserver-advertise-address string

The IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--apiserver-bind-port int32 Default: 6443

Port for the API Server to bind to.

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--control-plane-endpoint string

Specify a stable IP address or DNS name for the control plane.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for controller-manager

--kubeconfig-dir string Default: "/etc/kubernetes"

The path where to save the kubeconfig file.

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.31 -

Generate a kubeconfig file for the kubelet to use *only* for cluster bootstrapping purposes

Synopsis

Generate the kubeconfig file for the kubelet to use and save it to kubelet.conf file.

Please note that this should *only* be used for cluster bootstrapping purposes. After your control plane is up, you should request all kubelet credentials from the CSR API.

```
kubeadm init phase kubeconfig kubelet [flags]
```

Options

--apiserver-advertise-address string

The IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--apiserver-bind-port int32 Default: 6443

Port for the API Server to bind to.

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--control-plane-endpoint string

Specify a stable IP address or DNS name for the control plane.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for kubelet

--kubeconfig-dir string Default: "/etc/kubernetes"

The path where to save the kubeconfig file.

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

--node-name string

Specify the node name.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.32 -

Generate a kubeconfig file for the scheduler to use

Synopsis

Generate the kubeconfig file for the scheduler to use and save it to scheduler.conf file.

```
kubeadm init phase kubeconfig scheduler [flags]
```

Options

--apiserver-advertise-address string

The IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--apiserver-bind-port int32 Default: 6443

Port for the API Server to bind to.

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--control-plane-endpoint string

Specify a stable IP address or DNS name for the control plane.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for scheduler

--kubeconfig-dir string Default: "/etc/kubernetes"

The path where to save the kubeconfig file.

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.33 -

Generate a kubeconfig file for the super-admin

Synopsis

Generate a kubeconfig file for the super-admin, and save it to super-admin.conf file.

```
kubeadm init phase kubeconfig super-admin [flags]
```

Options

--apiserver-advertise-address string

The IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--apiserver-bind-port int32 Default: 6443

Port for the API Server to bind to.

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--control-plane-endpoint string

Specify a stable IP address or DNS name for the control plane.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for super-admin

--kubeconfig-dir string Default: "/etc/kubernetes"

The path where to save the kubeconfig file.

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.34 -

Updates settings relevant to the kubelet after TLS bootstrap

Synopsis

Updates settings relevant to the kubelet after TLS bootstrap

```
kubeadm init phase kubelet-finalize [flags]
```

Examples

```
# Updates settings relevant to the kubelet after TLS bootstrap
kubeadm init phase kubelet-finalize all --config
```

Options

-h, --help

help for kubelet-finalize

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.35 -

Run all kubelet-finalize phases

Synopsis

Run all kubelet-finalize phases

```
kubeadm init phase kubelet-finalize all [flags]
```

Examples

```
# Updates settings relevant to the kubelet after TLS bootstrap"
kubeadm init phase kubelet-finalize all --config
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for all

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.36 -

Enable kubelet client certificate rotation

Synopsis

Enable kubelet client certificate rotation

```
kubeadm init phase kubelet-finalize enable-client-cert-rotation [flag]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for enable-client-cert-rotation

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.37 -

Enable kubelet client certificate rotation (DEPRECATED: use 'enable-client-cert-rotation' instead)

Synopsis

Enable kubelet client certificate rotation (DEPRECATED: use 'enable-client-cert-rotation' instead)

```
kubeadm init phase kubelet-finalize experimental-cert-rotation [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for experimental-cert-rotation

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.38 -

Write kubelet settings and (re)start the kubelet

Synopsis

Write a file with KubeletConfiguration and an environment file with node specific kubelet settings, and then (re)start kubelet.

```
kubeadm init phase kubelet-start [flags]
```

Examples

```
# Writes a dynamic environment file with kubelet flags from a InitC
kubeadm init phase kubelet-start --config config.yaml
```

Options

--config string

Path to a kubeadm configuration file.

--cri-socket string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for kubelet-start

--image-repository string Default: "registry.k8s.io"

Choose a container registry to pull control plane images from

--node-name string

Specify the node name.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

Options inherited from parent commands

`--rootfs` string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.39 -

Mark a node as a control-plane

Synopsis

Mark a node as a control-plane

```
kubeadm init phase mark-control-plane [flags]
```

Examples

```
# Applies control-plane label and taint to the current node, function
kubeadm init phase mark-control-plane --config config.yaml

# Applies control-plane label and taint to a specific node
kubeadm init phase mark-control-plane --node-name myNode
```

Options

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for mark-control-plane

--node-name string

Specify the node name.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.40 -

Run pre-flight checks

Synopsis

Run pre-flight checks for kubeadm init.

```
kubeadm init phase preflight [flags]
```

Examples

```
# Run pre-flight checks for kubeadm init using a config file.
kubeadm init phase preflight --config kubeadm-config.yaml
```

Options

--config string

Path to a kubeadm configuration file.

--cri-socket string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for preflight

--ignore-preflight-errors strings

A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'. Value 'all' ignores errors from all checks.

--image-repository string Default: "registry.k8s.io"

Choose a container registry to pull control plane images from

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.41 -

Show the join command for control-plane and worker node

Synopsis

Show the join command for control-plane and worker node

```
kubeadm init phase show-join-command [flags]
```

Options

-h, --help

help for show-join-command

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.42 -

Upload certificates to kubeadm-certs

Synopsis

Upload control plane certificates to the kubeadm-certs Secret

```
kubeadm init phase upload-certs [flags]
```

Options

--certificate-key string

Key used to encrypt the control-plane certificates in the kubeadm-certs Secret. The certificate key is a hex encoded string that is an AES key of size 32 bytes.

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for upload-certs

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--skip-certificate-key-print

Don't print the key used to encrypt the control-plane certificates.

--upload-certs

Upload control-plane certificates to the kubeadm-certs Secret.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.43 -

Upload the kubeadm and kubelet configuration to a ConfigMap

Synopsis

This command is not meant to be run on its own. See list of available subcommands.

```
kubeadm init phase upload-config [flags]
```

Options

-h, --help

help for upload-config

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.44 -

Upload all configuration to a config map

Synopsis

Upload all configuration to a config map

```
kubeadm init phase upload-config all [flags]
```

Options

--config string

Path to a kubeadm configuration file.

--cri-socket string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for all

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.45 -

Upload the kubeadm ClusterConfiguration to a ConfigMap

Synopsis

Upload the kubeadm ClusterConfiguration to a ConfigMap called `kubeadm-config` in the `kube-system` namespace. This enables correct configuration of system components and a seamless user experience when upgrading.

Alternatively, you can use `kubeadm config`.

```
kubeadm init phase upload-config kubeadm [flags]
```

Examples

```
# upload the configuration of your cluster
kubeadm init phase upload-config --config=myConfig.yaml
```

Options

`--config` string

Path to a kubeadm configuration file.

`--cri-socket` string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

`--dry-run`

Don't apply any changes; just output what would be done.

`-h, --help`

help for kubeadm

`--kubeconfig` string Default: `"/etc/kubernetes/admin.conf"`

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

`--rootfs` string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.5.46 -

Upload the kubelet component config to a ConfigMap

Synopsis

Upload the kubelet configuration extracted from the kubeadm InitConfiguration object to a kubelet-config ConfigMap in the cluster

```
kubeadm init phase upload-config kubelet [flags]
```

Examples

```
# Upload the kubelet configuration from the kubeadm Config file to
kubeadm init phase upload-config kubelet --config kubeadm.yaml
```

Options

--config string

Path to a kubeadm configuration file.

--cri-socket string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for kubelet

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.6 -

Run this on any machine you wish to join an existing cluster

Synopsis

When joining a kubeadm initialized cluster, we need to establish bidirectional trust. This is split into discovery (having the Node trust the Kubernetes Control Plane) and TLS bootstrap (having the Kubernetes Control Plane trust the Node).

There are 2 main schemes for discovery. The first is to use a shared token along with the IP address of the API server. The second is to provide a file - a subset of the standard kubeconfig file. The discovery/kubeconfig file supports token, client-go authentication plugins ("exec"), "tokenFile", and "AuthProvider". This file can be a local file or downloaded via an HTTPS URL. The forms are `kubeadm join --discovery-token abcdef.1234567890abcdef 1.2.3.4:6443`, `kubeadm join --discovery-file path/to/file.conf`, or `kubeadm join --discovery-file https://url/file.conf`. Only one form can be used. If the discovery information is loaded from a URL, HTTPS must be used. Also, in that case the host installed CA bundle is used to verify the connection.

If you use a shared token for discovery, you should also pass the `--discovery-token-ca-cert-hash` flag to validate the public key of the root certificate authority (CA) presented by the Kubernetes Control Plane. The value of this flag is specified as "`<hash-type>:<hex-encoded-value>`", where the supported hash type is "sha256". The hash is calculated over the bytes of the Subject Public Key Info (SPKI) object (as in RFC7469). This value is available in the output of "kubeadm init" or can be calculated using standard tools. The `--discovery-token-ca-cert-hash` flag may be repeated multiple times to allow more than one public key.

If you cannot know the CA public key hash ahead of time, you can pass the `--discovery-token-unsafe-skip-ca-verification` flag to disable this verification. This weakens the kubeadm security model since other nodes can potentially impersonate the Kubernetes Control Plane.

The TLS bootstrap mechanism is also driven via a shared token. This is used to temporarily authenticate with the Kubernetes Control Plane to submit a certificate signing request (CSR) for a locally created key pair. By default, kubeadm will set up the Kubernetes Control Plane to automatically approve these signing requests. This token is passed in with the `--tls-bootstrap-token abcdef.1234567890abcdef` flag.

Often times the same token is used for both parts. In this case, the `--token` flag can be used instead of specifying each token individually.

The "join [api-server-endpoint]" command executes the following phases:

<code>preflight</code>	Run join pre-flight checks
<code>control-plane-prepare</code>	Prepare the machine for serving a control plane
<code>/download-certs</code>	Download certificates shared among control planes
<code>/certs</code>	Generate the certificates for the new control plane
<code>/kubeconfig</code>	Generate the kubeconfig for the new control plane
<code>/control-plane</code>	Generate the manifests for the new control plane
<code>kubelet-start</code>	Write kubelet settings, certificates and (re)start the kubelet
<code>control-plane-join</code>	Join a machine as a control plane instance
<code>/etcd</code>	Add a new local etcd member
<code>/mark-control-plane</code>	Mark a node as a control-plane
<code>wait-control-plane</code>	EXPERIMENTAL: Wait for the control plane to start

```
kubeadm join [api-server-endpoint] [flags]
```

Options

--apiserver-advertise-address string

If the node should host a new control plane instance, the IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--apiserver-bind-port int32 Default: 6443

If the node should host a new control plane instance, the port for the API Server to bind to.

--certificate-key string

Use this key to decrypt the certificate secrets uploaded by init. The certificate key is a hex encoded string that is an AES key of size 32 bytes.

--config string

Path to a kubeadm configuration file.

--control-plane

Create a new control plane instance on this node

--cri-socket string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

--discovery-file string

For file-based discovery, a file or URL from which to load cluster information.

--discovery-token string

For token-based discovery, the token used to validate cluster information fetched from the API server.

--discovery-token-ca-cert-hash strings

For token-based discovery, validate that the root CA public key matches this hash (format: "<type>:<value>").

--discovery-token-unsafe-skip-ca-verification

For token-based discovery, allow joining without --discovery-token-ca-cert-hash pinning.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for join

--ignore-preflight-errors strings

A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'. Value 'all' ignores errors from all checks.

--node-name string

Specify the node name.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

--skip-phases strings

List of phases to be skipped

--tls-bootstrap-token string

Specify the token used to temporarily authenticate with the Kubernetes Control Plane while joining the node.

--token string

Use this token for both discovery-token and tls-bootstrap-token when those values are not provided.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.6.1 -

Use this command to invoke single phase of the join workflow

Synopsis

Use this command to invoke single phase of the join workflow

Options

`-h, --help`

help for phase

Options inherited from parent commands

`--rootfs string`

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.6.2 -

Join a machine as a control plane instance

Synopsis

Join a machine as a control plane instance

```
kubeadm join phase control-plane-join [flags]
```

Examples

```
# Joins a machine as a control plane instance
kubeadm join phase control-plane-join all
```

Options

-h, --help

help for control-plane-join

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.6.3 -

Join a machine as a control plane instance

Synopsis

Join a machine as a control plane instance

```
kubeadm join phase control-plane-join all [flags]
```

Options

--apiserver-advertise-address string

If the node should host a new control plane instance, the IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--config string

Path to a kubeadm configuration file.

--control-plane

Create a new control plane instance on this node

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for all

--node-name string

Specify the node name.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.6.4 -

Add a new local etcd member

Synopsis

Add a new local etcd member

```
kubeadm join phase control-plane-join etcd [flags]
```

Options

--apiserver-advertise-address string

If the node should host a new control plane instance, the IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--config string

Path to a kubeadm configuration file.

--control-plane

Create a new control plane instance on this node

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for etcd

--node-name string

Specify the node name.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.6.5 -

Mark a node as a control-plane

Synopsis

Mark a node as a control-plane

```
kubeadm join phase control-plane-join mark-control-plane [flags]
```

Options

--config string

Path to a kubeadm configuration file.

--control-plane

Create a new control plane instance on this node

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for mark-control-plane

--node-name string

Specify the node name.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.6.6 -

Prepare the machine for serving a control plane

Synopsis

Prepare the machine for serving a control plane

```
kubeadm join phase control-plane-prepare [flags]
```

Examples

```
# Prepares the machine for serving a control plane
kubeadm join phase control-plane-prepare all
```

Options

-h, --help

help for control-plane-prepare

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.6.7 -

Prepare the machine for serving a control plane

Synopsis

Prepare the machine for serving a control plane

```
kubeadm join phase control-plane-prepare all [api-server-endpoint] [f
```

Options

--apiserver-advertise-address string

If the node should host a new control plane instance, the IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--apiserver-bind-port int32 Default: 6443

If the node should host a new control plane instance, the port for the API Server to bind to.

--certificate-key string

Use this key to decrypt the certificate secrets uploaded by init. The certificate key is a hex encoded string that is an AES key of size 32 bytes.

--config string

Path to a kubeadm configuration file.

--control-plane

Create a new control plane instance on this node

--discovery-file string

For file-based discovery, a file or URL from which to load cluster information.

--discovery-token string

For token-based discovery, the token used to validate cluster information fetched from the API server.

--discovery-token-ca-cert-hash strings

For token-based discovery, validate that the root CA public key matches this hash (format: "<type>:<value>").

--discovery-token-unsafe-skip-ca-verification

For token-based discovery, allow joining without --discovery-token-ca-cert-hash pinning.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for all

--node-name string

Specify the node name.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

--tls-bootstrap-token string

Specify the token used to temporarily authenticate with the Kubernetes Control Plane while joining the node.

--token string

Use this token for both discovery-token and tls-bootstrap-token when those values are not provided.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.6.8 -

Generate the certificates for the new control plane components

Synopsis

Generate the certificates for the new control plane components

```
kubeadm join phase control-plane-prepare certs [api-server-endpoint]
```

Options

--apiserver-advertise-address string

If the node should host a new control plane instance, the IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--config string

Path to a kubeadm configuration file.

--control-plane

Create a new control plane instance on this node

--discovery-file string

For file-based discovery, a file or URL from which to load cluster information.

--discovery-token string

For token-based discovery, the token used to validate cluster information fetched from the API server.

--discovery-token-ca-cert-hash strings

For token-based discovery, validate that the root CA public key matches this hash (format: "<type>:<value>").

--discovery-token-unsafe-skip-ca-verification

For token-based discovery, allow joining without --discovery-token-ca-cert-hash pinning.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for certs

--node-name string

Specify the node name.

--tls-bootstrap-token string

Specify the token used to temporarily authenticate with the Kubernetes Control Plane while joining the node.

--token string

Use this token for both discovery-token and tls-bootstrap-token when those values are not provided.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.6.9 -

Generate the manifests for the new control plane components

Synopsis

Generate the manifests for the new control plane components

```
kubeadm join phase control-plane-prepare control-plane [flags]
```

Options

--apiserver-advertise-address string

If the node should host a new control plane instance, the IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--apiserver-bind-port int32 Default: 6443

If the node should host a new control plane instance, the port for the API Server to bind to.

--config string

Path to a kubeadm configuration file.

--control-plane

Create a new control plane instance on this node

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for control-plane

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.6.10 -

Download certificates shared among control-plane nodes from the kubeadm-certs Secret

Synopsis

Download certificates shared among control-plane nodes from the kubeadm-certs Secret

```
kubeadm join phase control-plane-prepare download-certs [api-server-e
```

Options

--certificate-key string

Use this key to decrypt the certificate secrets uploaded by init. The certificate key is a hex encoded string that is an AES key of size 32 bytes.

--config string

Path to a kubeadm configuration file.

--control-plane

Create a new control plane instance on this node

--discovery-file string

For file-based discovery, a file or URL from which to load cluster information.

--discovery-token string

For token-based discovery, the token used to validate cluster information fetched from the API server.

--discovery-token-ca-cert-hash strings

For token-based discovery, validate that the root CA public key matches this hash (format: "<type>:<value>").

--discovery-token-unsafe-skip-ca-verification

For token-based discovery, allow joining without --discovery-token-ca-cert-hash pinning.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for download-certs

--tls-bootstrap-token string

Specify the token used to temporarily authenticate with the Kubernetes Control Plane while joining the node.

--token string

Use this token for both discovery-token and tls-bootstrap-token when those values are not provided.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.6.11 -

Generate the kubeconfig for the new control plane components

Synopsis

Generate the kubeconfig for the new control plane components

```
kubeadm join phase control-plane-prepare kubeconfig [api-server-endpo
```

Options

--certificate-key string

Use this key to decrypt the certificate secrets uploaded by init. The certificate key is a hex encoded string that is an AES key of size 32 bytes.

--config string

Path to a kubeadm configuration file.

--control-plane

Create a new control plane instance on this node

--discovery-file string

For file-based discovery, a file or URL from which to load cluster information.

--discovery-token string

For token-based discovery, the token used to validate cluster information fetched from the API server.

--discovery-token-ca-cert-hash strings

For token-based discovery, validate that the root CA public key matches this hash (format: "<type>:<value>").

--discovery-token-unsafe-skip-ca-verification

For token-based discovery, allow joining without --discovery-token-ca-cert-hash pinning.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for kubeconfig

--tls-bootstrap-token string

Specify the token used to temporarily authenticate with the Kubernetes Control Plane while joining the node.

--token string

Use this token for both discovery-token and tls-bootstrap-token when those values are not provided.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.6.12 -

Write kubelet settings, certificates and (re)start the kubelet

Synopsis

Write a file with KubeletConfiguration and an environment file with node specific kubelet settings, and then (re)start kubelet.

```
kubeadm join phase kubelet-start [api-server-endpoint] [flags]
```

Options

--config string

Path to a kubeadm configuration file.

--cri-socket string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

--discovery-file string

For file-based discovery, a file or URL from which to load cluster information.

--discovery-token string

For token-based discovery, the token used to validate cluster information fetched from the API server.

--discovery-token-ca-cert-hash strings

For token-based discovery, validate that the root CA public key matches this hash (format: "<type>:<value>").

--discovery-token-unsafe-skip-ca-verification

For token-based discovery, allow joining without --discovery-token-ca-cert-hash pinning.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for kubelet-start

--node-name string

Specify the node name.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

--tls-bootstrap-token string

Specify the token used to temporarily authenticate with the Kubernetes Control Plane while joining the node.

--token string

Use this token for both discovery-token and tls-bootstrap-token when those values are not provided.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.6.13 -

Run join pre-flight checks

Synopsis

Run pre-flight checks for kubeadm join.

```
kubeadm join phase preflight [api-server-endpoint] [flags]
```

Examples

```
# Run join pre-flight checks using a config file.  
kubeadm join phase preflight --config kubeadm-config.yaml
```

Options

--apiserver-advertise-address string

If the node should host a new control plane instance, the IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--apiserver-bind-port int32 Default: 6443

If the node should host a new control plane instance, the port for the API Server to bind to.

--certificate-key string

Use this key to decrypt the certificate secrets uploaded by init. The certificate key is a hex encoded string that is an AES key of size 32 bytes.

--config string

Path to a kubeadm configuration file.

--control-plane

Create a new control plane instance on this node

--cri-socket string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

--discovery-file string

For file-based discovery, a file or URL from which to load cluster information.

--discovery-token string

For token-based discovery, the token used to validate cluster information fetched from the API server.

--discovery-token-ca-cert-hash strings

For token-based discovery, validate that the root CA public key matches this hash (format: "<type>:<value>").

--discovery-token-unsafe-skip-ca-verification

For token-based discovery, allow joining without --discovery-token-ca-cert-hash pinning.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for preflight

--ignore-preflight-errors strings

A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'. Value 'all' ignores errors from all checks.

--node-name string

Specify the node name.

--tls-bootstrap-token string

Specify the token used to temporarily authenticate with the Kubernetes Control Plane while joining the node.

--token string

Use this token for both discovery-token and tls-bootstrap-token when those values are not provided.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.6.14 -

EXPERIMENTAL: Wait for the control plane to start

Synopsis

EXPERIMENTAL: Wait for the control plane to start

```
kubeadm join phase wait-control-plane [flags]
```

Options

-h, --help

help for wait-control-plane

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.7 -

Kubeconfig file utilities

Synopsis

Kubeconfig file utilities.

Options

-h, --help

help for kubeconfig

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.7.1 -

Output a kubeconfig file for an additional user

Synopsis

Output a kubeconfig file for an additional user.

```
kubeadm kubeconfig user [flags]
```

Examples

```
# Output a kubeconfig file for an additional user named foo
kubeadm kubeconfig user --client-name=foo

# Output a kubeconfig file for an additional user named foo using a
kubeadm kubeconfig user --client-name=foo --config=bar
```

Options

--client-name string

The name of user. It will be used as the CN if client certificates are created

--config string

Path to a kubeadm configuration file.

-h, --help

help for user

--org strings

The organizations of the client certificate. It will be used as the O if client certificates are created

--token string

The token that should be used as the authentication mechanism for this kubeconfig, instead of client certificates

--validity-period duration Default: 8760h0m0s

The validity period of the client certificate. It is an offset from the current time.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.8 -

Performs a best effort revert of changes made to this host by 'kubeadm init' or 'kubeadm join'

Synopsis

Performs a best effort revert of changes made to this host by 'kubeadm init' or 'kubeadm join'

The "reset" command executes the following phases:

```
preflight      Run reset pre-flight checks
remove-etcd-member Remove a local etcd member.
cleanup-node   Run cleanup node.
```

```
kubeadm reset [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path to the directory where the certificates are stored. If specified, clean this directory.

--cleanup-tmp-dir

Cleanup the "/etc/kubernetes/tmp" directory

--config string

Path to a kubeadm configuration file.

--cri-socket string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

--dry-run

Don't apply any changes; just output what would be done.

-f, --force

Reset the node without prompting for confirmation.

-h, --help

help for reset

--ignore-preflight-errors strings

A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'. Value 'all' ignores errors from all checks.

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--skip-phases strings

List of phases to be skipped

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.8.1 -

Use this command to invoke single phase of the reset workflow

Synopsis

Use this command to invoke single phase of the reset workflow

Options

`-h, --help`

help for phase

Options inherited from parent commands

`--rootfs string`

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.8.2 -

Run cleanup node.

Synopsis

Run cleanup node.

```
kubeadm reset phase cleanup-node [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path to the directory where the certificates are stored. If specified, clean this directory.

--cleanup-tmp-dir

Cleanup the "/etc/kubernetes/tmp" directory

--cri-socket string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for cleanup-node

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.8.3 -

Run reset pre-flight checks

Synopsis

Run pre-flight checks for kubeadm reset.

```
kubeadm reset phase preflight [flags]
```

Options

--dry-run

Don't apply any changes; just output what would be done.

-f, --force

Reset the node without prompting for confirmation.

-h, --help

help for preflight

--ignore-preflight-errors strings

A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'. Value 'all' ignores errors from all checks.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.8.4 -

Remove a local etcd member.

Synopsis

Remove a local etcd member for a control plane node.

```
kubeadm reset phase remove-etcd-member [flags]
```

Options

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for remove-etcd-member

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.9 -

Manage bootstrap tokens

Synopsis

This command manages bootstrap tokens. It is optional and needed only for advanced use cases.

In short, bootstrap tokens are used for establishing bidirectional trust between a client and a server. A bootstrap token can be used when a client (for example a node that is about to join the cluster) needs to trust the server it is talking to. Then a bootstrap token with the "signing" usage can be used. bootstrap tokens can also function as a way to allow short-lived authentication to the API Server (the token serves as a way for the API Server to trust the client), for example for doing the TLS Bootstrap.

What is a bootstrap token more exactly?

- It is a Secret in the kube-system namespace of type "bootstrap.kubernetes.io/token".
- A bootstrap token must be of the form "[a-z0-9]{6}.[a-z0-9]{16}". The former part is the public token ID, while the latter is the Token Secret and it must be kept private at all circumstances!
- The name of the Secret must be named "bootstrap-token-(token-id)".

You can read more about bootstrap tokens here: <https://kubernetes.io/docs/admin/bootstrap-tokens/>

```
kubeadm token [flags]
```

Options

--dry-run

Whether to enable dry-run mode or not

-h, --help

help for token

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.9.1 -

Create bootstrap tokens on the server

Synopsis

This command will create a bootstrap token for you. You can specify the usages for this token, the "time to live" and an optional human friendly description.

The [token] is the actual token to write. This should be a securely generated random token of the form "[a-z0-9]{6}.[a-z0-9]{16}]. If no [token] is given, kubeadm will generate a random token instead.

```
kubeadm token create [token]
```

Options

--certificate-key string

When used together with '--print-join-command', print the full 'kubeadm join' flag needed to join the cluster as a control-plane. To create a new certificate key you must use 'kubeadm init phase upload-certs --upload-certs'.

--config string

Path to a kubeadm configuration file.

--description string

A human friendly description of how this token is used.

--groups strings Default: "system:bootstrappers:kubeadm:default-node-token"

Extra groups that this token will authenticate as when used for authentication. Must match "\Asystem:bootstrappers:[a-z0-9:-]{0,255}[a-z0-9]\z"

-h, --help

help for create

--print-join-command

Instead of printing only the token, print the full 'kubeadm join' flag needed to join the cluster using the token.

--ttl duration Default: 24h0m0s

The duration before the token is automatically deleted (e.g. 1s, 2m, 3h). If set to '0', the token will never expire

--usages strings Default: "signing,authentication"

Describes the ways in which this token can be used. You can pass --usages multiple times or provide a comma separated list of options. Valid options: [signing,authentication]

Options inherited from parent commands

--dry-run

Whether to enable dry-run mode or not

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.9.2 -

Delete bootstrap tokens on the server

Synopsis

This command will delete a list of bootstrap tokens for you.

The [token-value] is the full Token of the form "[a-z0-9]{6}.[a-z0-9]{16}" or the Token ID of the form "[a-z0-9]{6}" to delete.

```
kubeadm token delete [token-value] ...
```

Options

-h, --help

help for delete

Options inherited from parent commands

--dry-run

Whether to enable dry-run mode or not

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.9.3 -

Generate and print a bootstrap token, but do not create it on the server

Synopsis

This command will print out a randomly-generated bootstrap token that can be used with the "init" and "join" commands.

You don't have to use this command in order to generate a token. You can do so yourself as long as it is in the format "[a-z0-9]{6}.[a-z0-9]{16}.". This command is provided for convenience to generate tokens in the given format.

You can also use "kubeadm init" without specifying a token and it will generate and print one for you.

```
kubeadm token generate [flags]
```

Options

-h, --help

help for generate

Options inherited from parent commands

--dry-run

Whether to enable dry-run mode or not

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.9.4 -

List bootstrap tokens on the server

Synopsis

This command will list all bootstrap tokens for you.

```
kubeadm token list [flags]
```

Options

--allow-missing-template-keys Default: true

If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to golang and jsonpath output formats.

-h, --help

help for list

-o, --output string Default: "text"

Output format. One of: text|json|yaml|go-template|go-template-file|template|templatefile|jsonpath|jsonpath-as-json|jsonpath-file.

--show-managed-fields

If true, keep the managedFields when printing objects in JSON or YAML format.

Options inherited from parent commands

--dry-run

Whether to enable dry-run mode or not

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.10 -

Upgrade your cluster smoothly to a newer version with this command

Synopsis

Upgrade your cluster smoothly to a newer version with this command

```
kubeadm upgrade [flags]
```

Options

-h, --help

help for upgrade

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.10.1 -

Upgrade your Kubernetes cluster to the specified version

Synopsis

Upgrade your Kubernetes cluster to the specified version

```
kubeadm upgrade apply [version]
```

Options

--allow-experimental-upgrades

Show unstable versions of Kubernetes as an upgrade alternative and allow upgrading to an alpha/beta/release candidate versions of Kubernetes.

--allow-release-candidate-upgrades

Show release candidate versions of Kubernetes as an upgrade alternative and allow upgrading to a release candidate versions of Kubernetes.

--certificate-renewal Default: true

Perform the renewal of certificates used by component changed during upgrades.

--config string

Path to a kubeadm configuration file.

--dry-run

Do not change any state, just output what actions would be performed.

--etcd-upgrade Default: true

Perform the upgrade of etcd.

-f, --force

Force upgrading although some requirements might not be met. This also implies non-interactive mode.

-h, --help

help for apply

--ignore-preflight-errors strings

A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'. Value 'all' ignores errors from all checks.

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

--print-config

Specifies whether the configuration file that will be used in the upgrade should be printed or not.

-y, --yes

Perform the upgrade and do not prompt for confirmation (non-interactive mode).

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.10.2 -

Show what differences would be applied to existing static pod manifests.

See also: `kubeadm upgrade apply --dry-run`

Synopsis

Show what differences would be applied to existing static pod manifests.

See also: `kubeadm upgrade apply --dry-run`

```
kubeadm upgrade diff [version] [flags]
```

Options

`--config` string

Path to a `kubeadm` configuration file.

`-c, --context-lines` int Default: 3

How many lines of context in the diff

`-h, --help`

help for diff

`--kubeconfig` string Default: "/etc/kubernetes/admin.conf"

The `kubeconfig` file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing `kubeconfig` file.

Options inherited from parent commands

`--rootfs` string

The path to the 'real' host root filesystem. This will cause `kubeadm` to chroot into the provided path.

10.1.1.10.3 -

Upgrade commands for a node in the cluster

Synopsis

Upgrade commands for a node in the cluster

The "node" command executes the following phases:

```
preflight      Run upgrade node pre-flight checks
control-plane  Upgrade the control plane instance deployed on this n
kubelet-config Upgrade the kubelet configuration for this node
```

```
kubeadm upgrade node [flags]
```

Options

--certificate-renewal Default: true

Perform the renewal of certificates used by component changed during upgrades.

--config string

Path to a kubeadm configuration file.

--dry-run

Do not change any state, just output the actions that would be performed.

--etcd-upgrade Default: true

Perform the upgrade of etcd.

-h, --help

help for node

--ignore-preflight-errors strings

A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'. Value 'all' ignores errors from all checks.

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

--skip-phases strings

List of phases to be skipped

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.10.4 -

Use this command to invoke single phase of the node workflow

Synopsis

Use this command to invoke single phase of the node workflow

Options

`-h, --help`

help for phase

Options inherited from parent commands

`--rootfs string`

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.10.5 -

Upgrade the control plane instance deployed on this node, if any

Synopsis

Upgrade the control plane instance deployed on this node, if any

```
kubeadm upgrade node phase control-plane [flags]
```

Options

--certificate-renewal Default: true

Perform the renewal of certificates used by component changed during upgrades.

--dry-run

Do not change any state, just output the actions that would be performed.

--etcd-upgrade Default: true

Perform the upgrade of etcd.

-h, --help

help for control-plane

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.10.6 -

Upgrade the kubelet configuration for this node

Synopsis

Download the kubelet configuration from the kubelet-config ConfigMap stored in the cluster

```
kubeadm upgrade node phase kubelet-config [flags]
```

Options

--dry-run

Do not change any state, just output the actions that would be performed.

-h, --help

help for kubelet-config

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.10.7 -

Run upgrade node pre-flight checks

Synopsis

Run pre-flight checks for kubeadm upgrade node.

```
kubeadm upgrade node phase preflight [flags]
```

Options

-h, --help

help for preflight

--ignore-preflight-errors strings

A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'. Value 'all' ignores errors from all checks.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.10.8 -

Check which versions are available to upgrade to and validate whether your current cluster is upgradeable.

Synopsis

Check which versions are available to upgrade to and validate whether your current cluster is upgradeable. This command can only run on the control plane nodes where the kubeconfig file "admin.conf" exists. To skip the internet check, pass in the optional [version] parameter.

```
kubeadm upgrade plan [version] [flags]
```

Options

--allow-experimental-upgrades

Show unstable versions of Kubernetes as an upgrade alternative and allow upgrading to an alpha/beta/release candidate versions of Kubernetes.

--allow-missing-template-keys Default: true

If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to golang and jsonpath output formats.

--allow-release-candidate-upgrades

Show release candidate versions of Kubernetes as an upgrade alternative and allow upgrading to a release candidate versions of Kubernetes.

--config string

Path to a kubeadm configuration file.

-h, --help

help for plan

--ignore-preflight-errors strings

A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'. Value 'all' ignores errors from all checks.

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

-o, --output string Default: "text"

Output format. One of: text|json|yaml|go-template|go-template-file|template|templatefile|jsonpath|jsonpath-as-json|jsonpath-file.

--print-config

Specifies whether the configuration file that will be used in the upgrade should be printed or not.

--show-managed-fields

If true, keep the managedFields when printing objects in JSON or YAML format.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.11 -

Print the version of kubeadm

Synopsis

Print the version of kubeadm

```
kubeadm version [flags]
```

Options

-h, --help

help for version

-o, --output string

Output format; available options are 'yaml', 'json' and 'short'

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.1.12 -

All files in this directory are auto-generated from other repos. **Do not edit them manually. You must edit them in their upstream repo.**

10.1.2 - kubeadm init

This command initializes a Kubernetes control-plane node.

Run this command in order to set up the Kubernetes control plane

Synopsis

Run this command in order to set up the Kubernetes control plane

The "init" command executes the following phases:

preflight	Run pre-flight checks
certs	Certificate generation
/ca	Generate the self-signed Kubernetes C
/apiserver	Generate the certificate for serving
/apiserver-kubelet-client	Generate the certificate for the API
/front-proxy-ca	Generate the self-signed CA to provis
/front-proxy-client	Generate the certificate for the fron
/etcd-ca	Generate the self-signed CA to provis
/etcd-server	Generate the certificate for serving
/etcd-peer	Generate the certificate for etcd nod
/etcd-healthcheck-client	Generate the certificate for liveness
/apiserver-etcd-client	Generate the certificate the apiserve
/sa	Generate a private key for signing se
kubeconfig	Generate all kubeconfig files necessary
/admin	Generate a kubeconfig file for the ad
/super-admin	Generate a kubeconfig file for the su
/kubelet	Generate a kubeconfig file for the ku
/controller-manager	Generate a kubeconfig file for the co
/scheduler	Generate a kubeconfig file for the sc
etcd	Generate static Pod manifest file for l
/local	Generate the static Pod manifest file
control-plane	Generate all static Pod manifest files
/apiserver	Generates the kube-apiserver static P
/controller-manager	Generates the kube-controller-manager
/scheduler	Generates the kube-scheduler static P
kubelet-start	Write kubelet settings and (re)start th
upload-config	Upload the kubeadm and kubelet configur
/kubeadm	Upload the kubeadm ClusterConfigurati
/kubelet	Upload the kubelet component config t
upload-certs	Upload certificates to kubeadm-certs
mark-control-plane	Mark a node as a control-plane
bootstrap-token	Generates bootstrap tokens used to join
kubelet-finalize	Updates settings relevant to the kubele
/enable-client-cert-rotation	Enable kubelet client certificate rot
/experimental-cert-rotation	Enable kubelet client certificate rot
addon	Install required addons for passing con
/coredns	Install the CoreDNS addon to a Kuber
/kube-proxy	Install the kube-proxy addon to a Kub
show-join-command	Show the join command for control-plane

kubeadm init [flags]

Options

--apiserver-advertise-address string

The IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--apiserver-bind-port int32 Default: 6443

Port for the API Server to bind to.

--apiserver-cert-extra-sans strings

Optional extra Subject Alternative Names (SANs) to use for the API Server serving certificate. Can be both IP addresses and DNS names.

--cert-dir string Default: "/etc/kubernetes/pki"

The path where to save and store the certificates.

--certificate-key string

Key used to encrypt the control-plane certificates in the `kubeadm-certs` Secret. The certificate key is a hex encoded string that is an AES key of size 32 bytes.

--config string

Path to a `kubeadm` configuration file.

--control-plane-endpoint string

Specify a stable IP address or DNS name for the control plane.

--cri-socket string

Path to the CRI socket to connect. If empty `kubeadm` will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

--dry-run

Don't apply any changes; just output what would be done.

--feature-gates string

A set of key=value pairs that describe feature gates for various features. Options are:
ControlPlaneKubeletLocalMode=true|false (ALPHA - default=false)
EtcdLearnerMode=true|false (BETA - default=true)
PublicKeysECDSA=true|false (DEPRECATED - default=false)
RootlessControlPlane=true|false (ALPHA - default=false)
WaitForAllControlPlaneComponents=true|false (ALPHA - default=false)

-h, --help

help for init

--ignore-preflight-errors strings

A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'. Value 'all' ignores errors from all checks.

--image-repository string Default: "registry.k8s.io"

Choose a container registry to pull control plane images from

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

--node-name string

Specify the node name.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

--pod-network-cidr string

Specify range of IP addresses for the pod network. If set, the control plane will automatically allocate CIDRs for every node.

--service-cidr string Default: "10.96.0.0/12"

Use alternative range of IP address for service VIPs.

--service-dns-domain string Default: "cluster.local"

Use alternative domain for services, e.g. "myorg.internal".

--skip-certificate-key-print

Don't print the key used to encrypt the control-plane certificates.

--skip-phases strings

List of phases to be skipped

--skip-token-print

Skip printing of the default bootstrap token generated by 'kubeadm init'.

--token string

The token to use for establishing bidirectional trust between nodes and control-plane nodes. The format is [a-z0-9]{6}.[a-z0-9]{16} - e.g. abcdef.0123456789abcdef

--token-ttl duration Default: 24h0m0s

The duration before the token is automatically deleted (e.g. 1s, 2m, 3h). If set to '0', the token will never expire

--upload-certs

Upload control-plane certificates to the `kubeadm-certs` Secret.

Options inherited from parent commands

`--rootfs` string

The path to the 'real' host root filesystem. This will cause `kubeadm` to chroot into the provided path.

Init workflow

`kubeadm init` bootstraps a Kubernetes control-plane node by executing the following steps:

1. Runs a series of pre-flight checks to validate the system state before making changes. Some checks only trigger warnings, others are considered errors and will exit `kubeadm` until the problem is corrected or the user specifies `--ignore-preflight-errors=<list-of-errors>`.
2. Generates a self-signed CA to set up identities for each component in the cluster. The user can provide their own CA cert and/or key by dropping it in the cert directory configured via `--cert-dir` (`/etc/kubernetes/pki` by default). The API Server certs will have additional SAN entries for any `--apiserver-cert-extra-sans` arguments, lowercased if necessary.
3. Writes kubeconfig files in `/etc/kubernetes/` for the kubelet, the controller-manager and the scheduler to use to connect to the API server, each with its own identity. Also additional kubeconfig files are written, for `kubeadm` as administrative entity (`admin.conf`) and for a super admin user that can bypass RBAC (`super-admin.conf`).
4. Generates static Pod manifests for the API server, controller-manager and scheduler. In case an external etcd is not provided, an additional static Pod manifest is generated for etcd.

Static Pod manifests are written to `/etc/kubernetes/manifests`; the kubelet watches this directory for Pods to create on startup.

Once control plane Pods are up and running, the `kubeadm init` sequence can continue.

5. Apply labels and taints to the control-plane node so that no additional workloads will run there.
6. Generates the token that additional nodes can use to register themselves with a control-plane in the future. Optionally, the user can provide a token via `--token`, as described in the [kubeadm token](#) docs.
7. Makes all the necessary configurations for allowing node joining with the [Bootstrap Tokens](#) and [TLS Bootstrap](#) mechanism:
 - Write a ConfigMap for making available all the information required for joining, and set up related RBAC access rules.
 - Let Bootstrap Tokens access the CSR signing API.
 - Configure auto-approval for new CSR requests.

See [kubeadm join](#) for additional info.

8. Installs a DNS server (CoreDNS) and the kube-proxy addon components via the API server. In Kubernetes version 1.11 and later CoreDNS is the default DNS server. Please note that although the DNS server is deployed, it will not be scheduled until CNI is installed.

Warning:

kube-dns usage with kubeadm is deprecated as of v1.18 and is removed in v1.21.

Using init phases with kubeadm

Kubeadm allows you to create a control-plane node in phases using the `kubeadm init phase` command.

To view the ordered list of phases and sub-phases you can call `kubeadm init --help`. The list will be located at the top of the help screen and each phase will have a description next to it. Note that by calling `kubeadm init` all of the phases and sub-phases will be executed in this exact order.

Some phases have unique flags, so if you want to have a look at the list of available options add `--help`, for example:

```
sudo kubeadm init phase control-plane controller-manager --help
```

You can also use `--help` to see the list of sub-phases for a certain parent phase:

```
sudo kubeadm init phase control-plane --help
```

`kubeadm init` also exposes a flag called `--skip-phases` that can be used to skip certain phases. The flag accepts a list of phase names and the names can be taken from the above ordered list.

An example:

```
sudo kubeadm init phase control-plane all --config=configfile.yaml
sudo kubeadm init phase etcd local --config=configfile.yaml
# you can now modify the control plane and etcd manifest files
sudo kubeadm init --skip-phases=control-plane,etcd --config=configfil
```

What this example would do is write the manifest files for the control plane and etcd in `/etc/kubernetes/manifests` based on the configuration in `configfile.yaml`. This allows you to modify the files and then skip these phases using `--skip-phases`. By calling the last command you will create a control plane node with the custom manifest files.

ⓘ **FEATURE STATE:** Kubernetes v1.22 [beta]

Alternatively, you can use the `skipPhases` field under `InitConfiguration`.

Using kubeadm init with a configuration file

Caution:

The config file is still considered beta and may change in future versions.

It's possible to configure `kubeadm init` with a configuration file instead of command line flags, and some more advanced features may only be available as configuration file options. This file is passed using the `--config` flag and it must contain a `ClusterConfiguration` structure and optionally more structures separated by `---`. Mixing `--config` with other flags may not be allowed in some cases.

The default configuration can be printed out using the [kubeadm config print](#) command.

If your configuration is not using the latest version it is **recommended** that you migrate using the [kubeadm config migrate](#) command.

For more information on the fields and usage of the configuration you can navigate to our [API reference page](#).

Using kubeadm init with feature gates

Kubeadm supports a set of feature gates that are unique to kubeadm and can only be applied during cluster creation with `kubeadm init`. These features can control the behavior of the cluster. Feature gates are removed after a feature graduates to GA.

To pass a feature gate you can either use the `--feature-gates` flag for `kubeadm init`, or you can add items into the `featureGates` field when you pass a [configuration file](#) using `--config`.

Passing [feature gates for core Kubernetes components](#) directly to kubeadm is not supported. Instead, it is possible to pass them by [Customizing components with the kubeadm API](#).

List of feature gates:

Feature	Default	Alpha	Beta	GA
ControlPlaneKubeletLocalMode	false	1.31	-	-
EtcdLearnerMode	true	1.27	1.29	-
PublicKeysECDSA	false	1.19	-	-
WaitForAllControlPlaneComponents	false	1.30	-	-

Note:

Once a feature gate goes GA its value becomes locked to `true` by default.

Feature gate descriptions:

`ControlPlaneKubeletLocalMode`

With this feature gate enabled, when joining a new control plane node, kubeadm will configure the kubelet to connect to the local kube-apiserver. This ensures that there will not be a violation of the version skew policy during rolling upgrades.

`EtcdLearnerMode`

With this feature gate enabled, when joining a new control plane node,

a new etcd member will be created as a learner and promoted to a voting member only after the etcd data are fully aligned.

PublicKeysECDSA

Can be used to create a cluster that uses ECDSA certificates instead of the default RSA algorithm. Renewal of existing ECDSA certificates is also supported using `kubeadm certs renew`, but you cannot switch between the RSA and ECDSA algorithms on the fly or during upgrades. Kubernetes 1.31 has a bug where keys in generated kubeconfig files are set use RSA despite the feature gate being enabled. Kubernetes versions before v1.31 had a bug where keys in generated kubeconfig files were set use RSA, even when you had enabled the `PublicKeysECDSA` feature gate.

WaitForAllControlPlaneComponents

With this feature gate enabled `kubeadm` will wait for all control plane components (`kube-apiserver`, `kube-controller-manager`, `kube-scheduler`) on a control plane node to report status 200 on their `/healthz` endpoints. These checks are performed on `https://127.0.0.1:PORT/healthz`, where `PORT` is taken from `--secure-port` of a component. If you specify custom `--secure-port` values in the `kubeadm` configuration they will be respected. Without the feature gate enabled, `kubeadm` will only wait for the `kube-apiserver` on a control plane node to become ready. The wait process starts right after the `kubelet` on the host is started by `kubeadm`. You are advised to enable this feature gate in case you wish to observe a ready state from all control plane components during the `kubeadm init` or `kubeadm join` command execution.

List of deprecated feature gates:

Feature	Default	Alpha	Beta	GA	Deprecated
RootlessControlPlane	false	1.22	-	-	1.31

Feature gate descriptions:

RootlessControlPlane

Setting this flag configures the `kubeadm` deployed control plane component static Pod containers for `kube-apiserver`, `kube-controller-manager`, `kube-scheduler` and `etcd` to run as non-root users. If the flag is not set, those components run as root. You can change the value of this feature gate before you upgrade to a newer version of Kubernetes.

List of removed feature gates:

Feature	Alpha	Beta	GA	Removed
IPv6DualStack	1.16	1.21	1.23	1.24
UnversionedKubeletConfigMap	1.22	1.23	1.25	1.26
UpgradeAddonsBeforeControlPlane	1.28	-	-	1.31

Feature gate descriptions:

IPv6DualStack

This flag helps to configure components dual stack when the feature is in progress. For more details on Kubernetes dual-stack support see

[Dual-stack support with kubeadm.](#)

UnversionedKubeletConfigMap

This flag controls the name of the [ConfigMap](#) where kubeadm stores kubelet configuration data. With this flag not specified or set to `true`, the ConfigMap is named `kubelet-config`. If you set this flag to `false`, the name of the ConfigMap includes the major and minor version for Kubernetes (for example: `kubelet-config-1.31`). Kubeadm ensures that RBAC rules for reading and writing that ConfigMap are appropriate for the value you set. When kubeadm writes this ConfigMap (during `kubeadm init` or `kubeadm upgrade apply`), kubeadm respects the value of `UnversionedKubeletConfigMap`. When reading that ConfigMap (during `kubeadm join`, `kubeadm reset`, `kubeadm upgrade ...`), kubeadm attempts to use unversioned ConfigMap name first; if that does not succeed, kubeadm falls back to using the legacy (versioned) name for that ConfigMap.

UpgradeAddonsBeforeControlPlane

This feature gate has been removed. It was introduced in v1.28 as a deprecated feature and then removed in v1.31. For documentation on older versions, please switch to the corresponding website version.

Adding kube-proxy parameters

For information about kube-proxy parameters in the kubeadm configuration see:

- [kube-proxy reference](#)

For information about enabling IPVS mode with kubeadm see:

- [IPVS](#)

Passing custom flags to control plane components

For information about passing flags to control plane components see:

- [control-plane-flags](#)

Running kubeadm without an Internet connection

For running kubeadm without an Internet connection you have to pre-pull the required control-plane images.

You can list and pull the images using the `kubeadm config images` sub-command:

```
kubeadm config images list
kubeadm config images pull
```

You can pass `--config` to the above commands with a [kubeadm configuration file](#) to control the `kubernetesVersion` and `imageRepository` fields.

All default `registry.k8s.io` images that kubeadm requires support multiple architectures.

Using custom images

By default, kubeadm pulls images from `registry.k8s.io`. If the requested

Kubernetes version is a CI label (such as `ci/latest`) `gcr.io/k8s-staging-ci-images` is used.

You can override this behavior by using [kubeadm with a configuration file](#). Allowed customization are:

- To provide `kubernetesVersion` which affects the version of the images.
- To provide an alternative `imageRepository` to be used instead of `registry.k8s.io`.
- To provide a specific `imageRepository` and `imageTag` for etcd or CoreDNS.

Image paths between the default `registry.k8s.io` and a custom repository specified using `imageRepository` may differ for backwards compatibility reasons. For example, one image might have a subpath at `registry.k8s.io/subpath/image`, but be defaulted to `my.customrepository.io/image` when using a custom repository.

To ensure you push the images to your custom repository in paths that `kubeadm` can consume, you must:

- Pull images from the defaults paths at `registry.k8s.io` using `kubeadm config images {list|pull}`.
- Push images to the paths from `kubeadm config images list --config=config.yaml`, where `config.yaml` contains the custom `imageRepository`, and/or `imageTag` for etcd and CoreDNS.
- Pass the same `config.yaml` to `kubeadm init`.

Custom sandbox (pause) images

To set a custom image for these you need to configure this in your container runtime to use the image. Consult the documentation for your container runtime to find out how to change this setting; for selected container runtimes, you can also find advice within the [Container Runtimes](#) topic.

Uploading control-plane certificates to the cluster

By adding the flag `--upload-certs` to `kubeadm init` you can temporary upload the control-plane certificates to a Secret in the cluster. Please note that this Secret will expire automatically after 2 hours. The certificates are encrypted using a 32byte key that can be specified using `--certificate-key`. The same key can be used to download the certificates when additional control-plane nodes are joining, by passing `--control-plane` and `--certificate-key` to `kubeadm join`.

The following phase command can be used to re-upload the certificates after expiration:

```
kubeadm init phase upload-certs --upload-certs --config=SOME_YAML_FILE
```

Note:

A predefined `certificateKey` can be provided in `InitConfiguration` when passing the [configuration file](#) with `--config`.

If a predefined certificate key is not passed to `kubeadm init` and `kubeadm init phase upload-certs` a new key will be generated automatically.

The following command can be used to generate a new key on demand:

```
kubeadm certs certificate-key
```

Certificate management with kubeadm

For detailed information on certificate management with kubeadm see [Certificate Management with kubeadm](#). The document includes information about using external CA, custom certificates and certificate renewal.

Managing the kubeadm drop-in file for the kubelet

The `kubeadm` package ships with a configuration file for running the `kubelet` by `systemd`. Note that the `kubeadm` CLI never touches this drop-in file. This drop-in file is part of the `kubeadm` DEB/RPM package.

For further information, see [Managing the kubeadm drop-in file for systemd](#).

Use kubeadm with CRI runtimes

By default `kubeadm` attempts to detect your container runtime. For more details on this detection, see the [kubeadm CRI installation guide](#).

Setting the node name

By default, `kubeadm` assigns a node name based on a machine's host address. You can override this setting with the `--node-name` flag. The flag passes the appropriate [--hostname-override](#) value to the `kubelet`.

Be aware that overriding the hostname can [interfere with cloud providers](#).

Automating kubeadm

Rather than copying the token you obtained from `kubeadm init` to each node, as in the [basic kubeadm tutorial](#), you can parallelize the token distribution for easier automation. To implement this automation, you must know the IP address that the control-plane node will have after it is started, or use a DNS name or an address of a load balancer.

1. Generate a token. This token must have the form `<6 character string>.<16 character string>`. More formally, it must match the regex: `[a-z0-9]{6}\.[a-z0-9]{16}`.

`kubeadm` can generate a token for you:

```
kubeadm token generate
```

2. Start both the control-plane node and the worker nodes concurrently with this token. As they come up they should find each other and form the cluster. The same `--token` argument can be used on both `kubeadm init` and `kubeadm join`.
3. Similar can be done for `--certificate-key` when joining additional control-plane nodes. The key can be generated using:

```
kubeadm certs certificate-key
```

Once the cluster is up, you can use the `/etc/kubernetes/admin.conf` file from a control-plane node to talk to the cluster with administrator credentials or [Generating kubeconfig files for additional users](#).

Note that this style of bootstrap has some relaxed security guarantees because it does not allow the root CA hash to be validated with `--discovery-token-ca-cert-hash` (since it's not generated when the nodes are provisioned). For details, see the [kubeadm join](#).

What's next

- [kubeadm init phase](#) to understand more about `kubeadm init` phases
- [kubeadm join](#) to bootstrap a Kubernetes worker node and join it to the cluster
- [kubeadm upgrade](#) to upgrade a Kubernetes cluster to a newer version
- [kubeadm reset](#) to revert any changes made to this host by `kubeadm init` or `kubeadm join`

10.1.3 - kubeadm join

This command initializes a Kubernetes worker node and joins it to the cluster.

Run this on any machine you wish to join an existing cluster

Synopsis

When joining a kubeadm initialized cluster, we need to establish bidirectional trust. This is split into discovery (having the Node trust the Kubernetes Control Plane) and TLS bootstrap (having the Kubernetes Control Plane trust the Node).

There are 2 main schemes for discovery. The first is to use a shared token along with the IP address of the API server. The second is to provide a file - a subset of the standard kubeconfig file. The discovery/kubeconfig file supports token, client-go authentication plugins ("exec"), "tokenFile", and "authProvider". This file can be a local file or downloaded via an HTTPS URL. The forms are kubeadm join --discovery-token abcdef.1234567890abcdef 1.2.3.4:6443, kubeadm join --discovery-file path/to/file.conf, or kubeadm join --discovery-file https://url/file.conf. Only one form can be used. If the discovery information is loaded from a URL, HTTPS must be used. Also, in that case the host installed CA bundle is used to verify the connection.

If you use a shared token for discovery, you should also pass the --discovery-token-ca-cert-hash flag to validate the public key of the root certificate authority (CA) presented by the Kubernetes Control Plane. The value of this flag is specified as "<hash-type>:<hex-encoded-value>", where the supported hash type is "sha256". The hash is calculated over the bytes of the Subject Public Key Info (SPKI) object (as in RFC7469). This value is available in the output of "kubeadm init" or can be calculated using standard tools. The --discovery-token-ca-cert-hash flag may be repeated multiple times to allow more than one public key.

If you cannot know the CA public key hash ahead of time, you can pass the --discovery-token-unsafe-skip-ca-verification flag to disable this verification. This weakens the kubeadm security model since other nodes can potentially impersonate the Kubernetes Control Plane.

The TLS bootstrap mechanism is also driven via a shared token. This is used to temporarily authenticate with the Kubernetes Control Plane to submit a certificate signing request (CSR) for a locally created key pair. By default, kubeadm will set up the Kubernetes Control Plane to automatically approve these signing requests. This token is passed in with the --tls-bootstrap-token abcdef.1234567890abcdef flag.

Often times the same token is used for both parts. In this case, the --token flag can be used instead of specifying each token individually.

The "join [api-server-endpoint]" command executes the following phases:

preflight	Run join pre-flight checks
control-plane-prepare	Prepare the machine for serving a control plan
/download-certs	Download certificates shared among control-p
/certs	Generate the certificates for the new contro
/kubeconfig	Generate the kubeconfig for the new control
/control-plane	Generate the manifests for the new control p
kubelet-start	Write kubelet settings, certificates and (re)s
control-plane-join	Join a machine as a control plane instance
/etcd	Add a new local etcd member
/mark-control-plane	Mark a node as a control-plane
wait-control-plane	EXPERIMENTAL: Wait for the control plane to st

```
kubeadm join [api-server-endpoint] [flags]
```

Options

--apiserver-advertise-address string

If the node should host a new control plane instance, the IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

--apiserver-bind-port int32 Default: 6443

If the node should host a new control plane instance, the port for the API Server to bind to.

--certificate-key string

Use this key to decrypt the certificate secrets uploaded by init. The certificate key is a hex encoded string that is an AES key of size 32 bytes.

--config string

Path to a kubeadm configuration file.

--control-plane

Create a new control plane instance on this node

--cri-socket string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

--discovery-file string

For file-based discovery, a file or URL from which to load cluster information.

--discovery-token string

For token-based discovery, the token used to validate cluster information fetched from the API server.

--discovery-token-ca-cert-hash strings

For token-based discovery, validate that the root CA public key matches this hash (format: "<type>:<value>").

--discovery-token-unsafe-skip-ca-verification

For token-based discovery, allow joining without --discovery-token-ca-cert-hash pinning.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for join

--ignore-preflight-errors strings

A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'. Value 'all' ignores errors from all checks.

--node-name string

Specify the node name.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

--skip-phases strings

List of phases to be skipped

--tls-bootstrap-token string

Specify the token used to temporarily authenticate with the Kubernetes Control Plane while joining the node.

--token string

Use this token for both discovery-token and tls-bootstrap-token when those values are not provided.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

The join workflow

`kubeadm join` bootstraps a Kubernetes worker node or a control-plane node and adds it to the cluster. This action consists of the following steps for worker nodes:

1. kubeadm downloads necessary cluster information from the API server. By default, it uses the bootstrap token and the CA key hash to verify the authenticity of that data. The root CA can also be

discovered directly via a file or URL.

2. Once the cluster information is known, kubelet can start the TLS bootstrapping process.

The TLS bootstrap uses the shared token to temporarily authenticate with the Kubernetes API server to submit a certificate signing request (CSR); by default the control plane signs this CSR request automatically.

3. Finally, kubeadm configures the local kubelet to connect to the API server with the definitive identity assigned to the node.

For control-plane nodes additional steps are performed:

1. Downloading certificates shared among control-plane nodes from the cluster (if explicitly requested by the user).
2. Generating control-plane component manifests, certificates and kubeconfig.
3. Adding new local etcd member.

Using join phases with kubeadm

Kubeadm allows you join a node to the cluster in phases using `kubeadm join phase`.

To view the ordered list of phases and sub-phases you can call `kubeadm join --help`. The list will be located at the top of the help screen and each phase will have a description next to it. Note that by calling `kubeadm join` all of the phases and sub-phases will be executed in this exact order.

Some phases have unique flags, so if you want to have a look at the list of available options add `--help`, for example:

```
kubeadm join phase kubelet-start --help
```

Similar to the [kubeadm init phase](#) command, `kubeadm join phase` allows you to skip a list of phases using the `--skip-phases` flag.

For example:

```
sudo kubeadm join --skip-phases=preflight --config=config.yaml
```

ⓘ FEATURE STATE: Kubernetes v1.22 [beta]

Alternatively, you can use the `skipPhases` field in `JoinConfiguration`.

Discovering what cluster CA to trust

The kubeadm discovery has several options, each with security tradeoffs. The right method for your environment depends on how you provision nodes and the security expectations you have about your network and node lifecycles.

Token-based discovery with CA pinning

This is the default mode in kubeadm. In this mode, kubeadm downloads

the cluster configuration (including root CA) and validates it using the token as well as validating that the root CA public key matches the provided hash and that the API server certificate is valid under the root CA.

The CA key hash has the format `sha256:<hex_encoded_hash>`. By default, the hash value is printed at the end of the `kubeadm init` command or in the output from the `kubeadm token create --print-join-command` command. It is in a standard format (see [RFC7469](#)) and can also be calculated by 3rd party tools or provisioning systems. For example, using the OpenSSL CLI:

```
openssl x509 -pubkey -in /etc/kubernetes/pki/ca.crt | openssl rsa -pu
```

Example `kubeadm join` commands:

For worker nodes:

```
kubeadm join --discovery-token abcdef.1234567890abcdef --discovery-to
```

For control-plane nodes:

```
kubeadm join --discovery-token abcdef.1234567890abcdef --discovery-to
```

You can also call `join` for a control-plane node with `--certificate-key` to copy certificates to this node, if the `kubeadm init` command was called with `--upload-certs`.

Advantages:

- Allows bootstrapping nodes to securely discover a root of trust for the control-plane node even if other worker nodes or the network are compromised.
- Convenient to execute manually since all of the information required fits into a single `kubeadm join` command.

Disadvantages:

- The CA hash is not normally known until the control-plane node has been provisioned, which can make it more difficult to build automated provisioning tools that use `kubeadm`. By generating your CA in beforehand, you may workaround this limitation.

Token-based discovery without CA pinning

This mode relies only on the symmetric token to sign (HMAC-SHA256) the discovery information that establishes the root of trust for the control-plane. To use the mode the joining nodes must skip the hash validation of the CA public key, using `--discovery-token-unsafe-skip-ca-verification`. You should consider using one of the other modes if possible.

Example `kubeadm join` command:

```
kubeadm join --token abcdef.1234567890abcdef --discovery-token-unsafe
```

Advantages:

- Still protects against many network-level attacks.
- The token can be generated ahead of time and shared with the control-plane node and worker nodes, which can then bootstrap in parallel without coordination. This allows it to be used in many provisioning scenarios.

Disadvantages:

- If an attacker is able to steal a bootstrap token via some vulnerability, they can use that token (along with network-level access) to impersonate the control-plane node to other bootstrapping nodes. This may or may not be an appropriate tradeoff in your environment.

File or HTTPS-based discovery

This provides an out-of-band way to establish a root of trust between the control-plane node and bootstrapping nodes. Consider using this mode if you are building automated provisioning using `kubeadm`. The format of the discovery file is a regular Kubernetes [kubeconfig](#) file.

In case the discovery file does not contain credentials, the TLS discovery token will be used.

Example `kubeadm join` commands:

- `kubeadm join --discovery-file path/to/file.conf` (local file)
- `kubeadm join --discovery-file https://url/file.conf` (remote HTTPS URL)

Advantages:

- Allows bootstrapping nodes to securely discover a root of trust for the control-plane node even if the network or other worker nodes are compromised.

Disadvantages:

- Requires that you have some way to carry the discovery information from the control-plane node to the bootstrapping nodes. If the discovery file contains credentials you must keep it secret and transfer it over a secure channel. This might be possible with your cloud provider or provisioning tool.

Use of custom kubelet credentials with `kubeadm join`

To allow `kubeadm join` to use predefined kubelet credentials and skip client TLS bootstrap and CSR approval for a new node:

1. From a working control plane node in the cluster that has `/etc/kubernetes/pki/ca.key` execute `kubeadm kubeconfig user --org system:nodes --client-name system:node:$NODE > kubelet.conf`. `$NODE` must be set to the name of the new node.
2. Modify the resulted `kubelet.conf` manually to adjust the cluster name and the server endpoint, or run `kubeadm kubeconfig user --config` (it accepts `InitConfiguration`).

If your cluster does not have the `ca.key` file, you must sign the embedded certificates in the `kubelet.conf` externally. For additional information, see [PKI certificates and requirements](#) and [Certificate Management with kubeadm](#).

1. Copy the resulting `kubelet.conf` to `/etc/kubernetes/kubelet.conf` on the new node.
2. Execute `kubeadm join` with the flag `--ignore-preflight-errors=FileAvailable--etc-kubernetes-kubelet.conf` on the new node.

Securing your installation even more

The defaults for `kubeadm` may not work for everyone. This section documents how to tighten up a `kubeadm` installation at the cost of some usability.

Turning off auto-approval of node client certificates

By default, there is a CSR auto-approver enabled that basically approves any client certificate request for a `kubelet` when a Bootstrap Token was used when authenticating. If you don't want the cluster to automatically approve `kubelet` client certs, you can turn it off by executing this command:

```
kubectl delete clusterrolebinding kubeadm:node-autoapprove-bootstrap
```

After that, `kubeadm join` will block until the admin has manually approved the CSR in flight:

1. Using `kubectl get csr`, you can see that the original CSR is in the Pending state.

```
kubectl get csr
```

The output is similar to this:

NAME	AGE
node-csr-c69HXe7aYcqkS1bKmH4faEnHAWxn6i2bHZ2mD04jZyQ	18s

2. `kubectl certificate approve` allows the admin to approve CSR. This action tells a certificate signing controller to issue a certificate to the requestor with the attributes requested in the CSR.

```
kubectl certificate approve node-csr-c69HXe7aYcqkS1bKmH4faEnHAWx
```

The output is similar to this:

```
certificatesigningrequest "node-csr-c69HXe7aYcqkS1bKmH4faEnHAWxn
```

3. This would change the CRS resource to Active state.

```
kubectl get csr
```

The output is similar to this:

NAME	AGE
node-csr-c69HXe7aYcqkS1bKmH4faEnHAWxn6i2bHZ2mD04jZyQ	1m

This forces the workflow that `kubeadm join` will only succeed if `kubectl certificate approve` has been run.

Turning off public access to the `cluster-info` ConfigMap

In order to achieve the joining flow using the token as the only piece of validation information, a ConfigMap with some data needed for validation of the control-plane node's identity is exposed publicly by default. While there is no private data in this ConfigMap, some users might wish to turn it off regardless. Doing so will disable the ability to use the `--discovery-token` flag of the `kubeadm join` flow. Here are the steps to do so:

- Fetch the `cluster-info` file from the API Server:

```
kubectl -n kube-public get cm cluster-info -o jsonpath='{.data.kubec...
```

The output is similar to this:

```
apiVersion: v1
kind: Config
clusters:
- cluster:
  certificate-authority-data: <ca-cert>
  server: https://<ip>:<port>
  name: ""
contexts: []
current-context: ""
preferences: {}
users: []
```

- Use the `cluster-info.yaml` file as an argument to `kubeadm join --discovery-file`.
- Turn off public access to the `cluster-info` ConfigMap:

```
kubectl -n kube-public delete rolebinding kubeadm:bootstrap-signer-cl...
```

These commands should be run after `kubeadm init` but before `kubeadm join`.

Using `kubeadm join` with a configuration file

Caution:

The config file is still considered beta and may change in future versions.

It's possible to configure `kubeadm join` with a configuration file instead of command line flags, and some more advanced features may only be available as configuration file options. This file is passed using the `--config` flag and it must contain a `JoinConfiguration` structure. Mixing `--config` with others flags may not be allowed in some cases.

The default configuration can be printed out using the [kubeadm config print](#) command.

If your configuration is not using the latest version it is **recommended** that you migrate using the [kubeadm config migrate](#) command.

For more information on the fields and usage of the configuration you can navigate to our [API reference](#).

What's next

- [kubeadm init](#) to bootstrap a Kubernetes control-plane node.
- [kubeadm token](#) to manage tokens for `kubeadm join`.
- [kubeadm reset](#) to revert any changes made to this host by `kubeadm init` or `kubeadm join`.

10.1.4 - kubeadm upgrade

`kubeadm upgrade` is a user-friendly command that wraps complex upgrading logic behind one command, with support for both planning an upgrade and actually performing it.

kubeadm upgrade guidance

The steps for performing an upgrade using kubeadm are outlined in [this document](#). For older versions of kubeadm, please refer to older documentation sets of the Kubernetes website.

You can use `kubeadm upgrade diff` to see the changes that would be applied to static pod manifests.

In Kubernetes v1.15.0 and later, `kubeadm upgrade apply` and `kubeadm upgrade node` will also automatically renew the kubeadm managed certificates on this node, including those stored in kubeconfig files. To opt-out, it is possible to pass the flag `--certificate-renewal=false`. For more details about certificate renewal see the [certificate management documentation](#).

Note:

The commands `kubeadm upgrade apply` and `kubeadm upgrade plan` have a legacy `--config` flag which makes it possible to reconfigure the cluster, while performing planning or upgrade of that particular control-plane node. Please be aware that the upgrade workflow was not designed for this scenario and there are reports of unexpected results.

kubeadm upgrade plan

Check which versions are available to upgrade to and validate whether your current cluster is upgradeable.

Synopsis

Check which versions are available to upgrade to and validate whether your current cluster is upgradeable. This command can only run on the control plane nodes where the kubeconfig file "admin.conf" exists. To skip the internet check, pass in the optional [version] parameter.

```
kubeadm upgrade plan [version] [flags]
```

Options

`--allow-experimental-upgrades`

Show unstable versions of Kubernetes as an upgrade alternative and allow upgrading to an alpha/beta/release candidate versions of Kubernetes.

`--allow-missing-template-keys` Default: true

If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to golang and jsonpath output formats.

--allow-release-candidate-upgrades

Show release candidate versions of Kubernetes as an upgrade alternative and allow upgrading to a release candidate versions of Kubernetes.

--config string

Path to a kubeadm configuration file.

-h, --help

help for plan

--ignore-preflight-errors strings

A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'. Value 'all' ignores errors from all checks.

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

-o, --output string Default: "text"

Output format. One of: text|json|yaml|go-template|go-template-file|template|templatefile|jsonpath|jsonpath-as-json|jsonpath-file.

--print-config

Specifies whether the configuration file that will be used in the upgrade should be printed or not.

--show-managed-fields

If true, keep the managedFields when printing objects in JSON or YAML format.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm upgrade apply

Upgrade your Kubernetes cluster to the specified version

Synopsis

Upgrade your Kubernetes cluster to the specified version

```
kubeadm upgrade apply [version]
```

Options

--allow-experimental-upgrades

Show unstable versions of Kubernetes as an upgrade alternative and allow upgrading to an alpha/beta/release candidate versions of Kubernetes.

--allow-release-candidate-upgrades

Show release candidate versions of Kubernetes as an upgrade alternative and allow upgrading to a release candidate versions of Kubernetes.

--certificate-renewal Default: true

Perform the renewal of certificates used by component changed during upgrades.

--config string

Path to a kubeadm configuration file.

--dry-run

Do not change any state, just output what actions would be performed.

--etcd-upgrade Default: true

Perform the upgrade of etcd.

-f, --force

Force upgrading although some requirements might not be met. This also implies non-interactive mode.

-h, --help

help for apply

--ignore-preflight-errors strings

A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'. Value 'all' ignores errors from all checks.

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

--print-config

Specifies whether the configuration file that will be used in the upgrade should be printed or not.

-y, --yes

Perform the upgrade and do not prompt for confirmation (non-interactive mode).

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm upgrade diff

Show what differences would be applied to existing static pod manifests.

See also: kubeadm upgrade apply --dry-run

Synopsis

Show what differences would be applied to existing static pod manifests.

See also: kubeadm upgrade apply --dry-run

```
kubeadm upgrade diff [version] [flags]
```

Options

--config string

Path to a kubeadm configuration file.

-c, --context-lines int Default: 3

How many lines of context in the diff

-h, --help

help for diff

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm upgrade node

Upgrade commands for a node in the cluster

Synopsis

Upgrade commands for a node in the cluster

The "node" command executes the following phases:

```
preflight      Run upgrade node pre-flight checks
control-plane  Upgrade the control plane instance deployed on this node
kubelet-config Upgrade the kubelet configuration for this node
```

```
kubeadm upgrade node [flags]
```

Options

--certificate-renewal Default: true

Perform the renewal of certificates used by component changed during upgrades.

--config string

Path to a kubeadm configuration file.

--dry-run

Do not change any state, just output the actions that would be performed.

--etcd-upgrade Default: true

Perform the upgrade of etcd.

-h, --help

help for node

--ignore-preflight-errors strings

A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'. Value 'all' ignores errors from all checks.

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

--skip-phases strings

List of phases to be skipped

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

What's next

- [kubeadm config](#) if you initialized your cluster using kubeadm v1.7.x or lower, to configure your cluster for `kubeadm upgrade`

10.1.5 - kubeadm config

During `kubeadm init`, `kubeadm` uploads the `clusterConfiguration` object to your cluster in a ConfigMap called `kubeadm-config` in the `kube-system` namespace. This configuration is then read during `kubeadm join`, `kubeadm reset` and `kubeadm upgrade`.

You can use `kubeadm config print` to print the default static configuration that `kubeadm` uses for `kubeadm init` and `kubeadm join`.

Note:

The output of the command is meant to serve as an example. You must manually edit the output of this command to adapt to your setup. Remove the fields that you are not certain about and `kubeadm` will try to default them on runtime by examining the host.

For more information on `init` and `join` navigate to [Using kubeadm init with a configuration file](#) or [Using kubeadm join with a configuration file](#).

For more information on using the `kubeadm` configuration API navigate to [Customizing components with the kubeadm API](#).

You can use `kubeadm config migrate` to convert your old configuration files that contain a deprecated API version to a newer, supported API version.

`kubeadm config validate` can be used for validating a configuration file.

`kubeadm config images list` and `kubeadm config images pull` can be used to list and pull the images that `kubeadm` requires.

kubeadm config print

Print configuration

Synopsis

This command prints configurations for subcommands provided. For details, see: <https://pkg.go.dev/k8s.io/kubernetes/cmd/kubeadm/app/apis/kubeadm#section-directories>

```
kubeadm config print [flags]
```

Options

-h, --help

help for print

Options inherited from parent commands

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The `kubeconfig` file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing `kubeconfig` file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm config print init-defaults

Print default init configuration, that can be used for 'kubeadm init'

Synopsis

This command prints objects such as the default init configuration that is used for 'kubeadm init'.

Note that sensitive values like the Bootstrap Token fields are replaced with placeholder values like "abcdef.0123456789abcdef" in order to pass validation but not perform the real computation for creating a token.

```
kubeadm config print init-defaults [flags]
```

Options

--component-configs strings

A comma-separated list for component config API objects to print the default values for. Available values: [KubeProxyConfiguration KubeletConfiguration]. If this flag is not set, no component configs will be printed.

-h, --help

help for init-defaults

Options inherited from parent commands

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm config print join-defaults

Print default join configuration, that can be used for 'kubeadm join'

Synopsis

This command prints objects such as the default join configuration that is used for 'kubeadm join'.

Note that sensitive values like the Bootstrap Token fields are replaced with placeholder values like "abcdef.0123456789abcdef" in order to pass validation but not perform the real computation for creating a token.

```
kubeadm config print join-defaults [flags]
```

Options

`-h, --help`

help for join-defaults

Options inherited from parent commands

`--kubeconfig string` Default: `"/etc/kubernetes/admin.conf"`

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

`--rootfs string`

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm config migrate

Read an older version of the kubeadm configuration API types from a file, and output the similar config object for the newer version

Synopsis

This command lets you convert configuration objects of older versions to the latest supported version, locally in the CLI tool without ever touching anything in the cluster. In this version of kubeadm, the following API versions are supported:

- `kubeadm.k8s.io/v1beta4`

Further, kubeadm can only write out config of version "kubeadm.k8s.io/v1beta4", but read both types. So regardless of what version you pass to the `--old-config` parameter here, the API object will be read, deserialized, defaulted, converted, validated, and re-serialized when written to stdout or `--new-config` if specified.

In other words, the output of this command is what kubeadm actually would read internally if you submitted this file to "kubeadm init"

```
kubeadm config migrate [flags]
```

Options

`--allow-experimental-api`

Allow migration to experimental, unreleased APIs.

-h, --help

help for migrate

--new-config string

Path to the resulting equivalent kubeadm config file using the new API version. Optional, if not specified output will be sent to STDOUT.

--old-config string

Path to the kubeadm config file that is using an old API version and should be converted. This flag is mandatory.

Options inherited from parent commands

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm config validate

Read a file containing the kubeadm configuration API and report any validation problems

Synopsis

This command lets you validate a kubeadm configuration API file and report any warnings and errors. If there are no errors the exit status will be zero, otherwise it will be non-zero. Any unmarshaling problems such as unknown API fields will trigger errors. Unknown API versions and fields with invalid values will also trigger errors. Any other errors or warnings may be reported depending on contents of the input file.

In this version of kubeadm, the following API versions are supported:

- kubeadm.k8s.io/v1beta4

```
kubeadm config validate [flags]
```

Options

--allow-experimental-api

Allow validation of experimental, unreleased APIs.

--config string

Path to a kubeadm configuration file.

-h, --help

help for validate

Options inherited from parent commands

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm config images list

Print a list of images kubeadm will use. The configuration file is used in case any images or image repositories are customized

Synopsis

Print a list of images kubeadm will use. The configuration file is used in case any images or image repositories are customized

```
kubeadm config images list [flags]
```

Options

--allow-missing-template-keys Default: true

If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to golang and jsonpath output formats.

--config string

Path to a kubeadm configuration file.

--feature-gates string

A set of key=value pairs that describe feature gates for various features. Options are:
ControlPlaneKubeletLocalMode=true|false (ALPHA - default=false)
EtcdLearnerMode=true|false (BETA - default=true)
PublicKeysECDSA=true|false (DEPRECATED - default=false)
RootlessControlPlane=true|false (ALPHA - default=false)
WaitForAllControlPlaneComponents=true|false (ALPHA - default=false)

-h, --help

help for list

--image-repository string Default: "registry.k8s.io"

Choose a container registry to pull control plane images from

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

-o, --output string Default: "text"

Output format. One of: text|json|yaml|go-template|go-template-file|template|templatefile|jsonpath|jsonpath-as-json|jsonpath-file.

--show-managed-fields

If true, keep the managedFields when printing objects in JSON or YAML format.

Options inherited from parent commands

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm config images pull

Pull images used by kubeadm

Synopsis

Pull images used by kubeadm

```
kubeadm config images pull [flags]
```

Options

--config string

Path to a kubeadm configuration file.

--cri-socket string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

--feature-gates string

A set of key=value pairs that describe feature gates for various features. Options are:
ControlPlaneKubeletLocalMode=true|false (ALPHA - default=false)
EtcdLearnerMode=true|false (BETA - default=true)
PublicKeysECDSA=true|false (DEPRECATED - default=false)
RootlessControlPlane=true|false (ALPHA - default=false)
WaitForAllControlPlaneComponents=true|false (ALPHA - default=false)

-h, --help

help for pull

--image-repository string Default: "registry.k8s.io"

Choose a container registry to pull control plane images from

--kubernetes-version string Default: "stable-1"

Choose a specific Kubernetes version for the control plane.

Options inherited from parent commands

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

What's next

- [kubeadm upgrade](#) to upgrade a Kubernetes cluster to a newer version

10.1.6 - kubeadm reset

Performs a best effort revert of changes made by `kubeadm init` or `kubeadm join`.

Performs a best effort revert of changes made to this host by 'kubeadm init' or 'kubeadm join'

Synopsis

Performs a best effort revert of changes made to this host by 'kubeadm init' or 'kubeadm join'

The "reset" command executes the following phases:

```
preflight      Run reset pre-flight checks
remove-etcd-member Remove a local etcd member.
cleanup-node   Run cleanup node.
```

```
kubeadm reset [flags]
```

Options

`--cert-dir` string Default: "/etc/kubernetes/pki"

The path to the directory where the certificates are stored. If specified, clean this directory.

`--cleanup-tmp-dir`

Cleanup the "/etc/kubernetes/tmp" directory

`--config` string

Path to a kubeadm configuration file.

`--cri-socket` string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

`--dry-run`

Don't apply any changes; just output what would be done.

`-f, --force`

Reset the node without prompting for confirmation.

`-h, --help`

help for reset

`--ignore-preflight-errors` strings

A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'. Value 'all' ignores errors from all checks.

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--skip-phases strings

List of phases to be skipped

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

Reset workflow

`kubeadm reset` is responsible for cleaning up a node local file system from files that were created using the `kubeadm init` or `kubeadm join` commands. For control-plane nodes `reset` also removes the local stacked etcd member of this node from the etcd cluster.

`kubeadm reset` phase can be used to execute the separate phases of the above workflow. To skip a list of phases you can use the `--skip-phases` flag, which works in a similar way to the `kubeadm join` and `kubeadm init` phase runners.

External etcd clean up

`kubeadm reset` will not delete any etcd data if external etcd is used. This means that if you run `kubeadm init` again using the same etcd endpoints, you will see state from previous clusters.

To wipe etcd data it is recommended you use a client like etcdctl, such as:

```
etcdctl del "" --prefix
```

See the [etcd documentation](#) for more information.

Graceful kube-apiserver shutdown

If you have your `kube-apiserver` configured with the `--shutdown-delay-duration` flag, you can run the following commands to attempt a graceful shutdown for the running API server Pod, before you run `kubeadm reset`:

```
yq eval '.spec.containers[0].command = []' /etc/kubernetes/manifests
timeout 60 sh -c 'while pgrep kube-apiserver >/dev/null; do sleep 1;
```

What's next

- [kubeadm init](#) to bootstrap a Kubernetes control-plane node
- [kubeadm join](#) to bootstrap a Kubernetes worker node and join it to the cluster

10.1.7 - kubeadm token

Bootstrap tokens are used for establishing bidirectional trust between a node joining the cluster and a control-plane node, as described in [authenticating with bootstrap tokens](#).

`kubeadm init` creates an initial token with a 24-hour TTL. The following commands allow you to manage such a token and also to create and manage new ones.

kubeadm token create

Create bootstrap tokens on the server

Synopsis

This command will create a bootstrap token for you. You can specify the usages for this token, the "time to live" and an optional human friendly description.

The `[token]` is the actual token to write. This should be a securely generated random token of the form `"[a-z0-9]{6}.[a-z0-9]{16}"`. If no `[token]` is given, `kubeadm` will generate a random token instead.

```
kubeadm token create [token]
```

Options

`--certificate-key` string

When used together with `--print-join-command`, print the full `'kubeadm join'` flag needed to join the cluster as a control-plane. To create a new certificate key you must use `'kubeadm init phase upload-certs --upload-certs'`.

`--config` string

Path to a `kubeadm` configuration file.

`--description` string

A human friendly description of how this token is used.

`--groups` strings Default: `"system:bootstrappers:kubeadm:default-node-token"`

Extra groups that this token will authenticate as when used for authentication. Must match `"\Asystem:bootstrappers:[a-z0-9:-]{0,255}[a-z0-9]\z"`

`-h, --help`

help for create

`--print-join-command`

Instead of printing only the token, print the full `'kubeadm join'` flag needed to join the cluster using the token.

--ttl duration Default: 24h0m0s

The duration before the token is automatically deleted (e.g. 1s, 2m, 3h). If set to '0', the token will never expire

--usages strings Default: "signing,authentication"

Describes the ways in which this token can be used. You can pass --usages multiple times or provide a comma separated list of options. Valid options: [signing,authentication]

Options inherited from parent commands

--dry-run

Whether to enable dry-run mode or not

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm token delete

Delete bootstrap tokens on the server

Synopsis

This command will delete a list of bootstrap tokens for you.

The [token-value] is the full Token of the form "[a-z0-9]{6}.[a-z0-9]{16}" or the Token ID of the form "[a-z0-9]{6}" to delete.

```
kubeadm token delete [token-value] ...
```

Options

-h, --help

help for delete

Options inherited from parent commands

--dry-run

Whether to enable dry-run mode or not

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm token generate

Generate and print a bootstrap token, but do not create it on the server

Synopsis

This command will print out a randomly-generated bootstrap token that can be used with the "init" and "join" commands.

You don't have to use this command in order to generate a token. You can do so yourself as long as it is in the format "[a-z0-9]{6}.[a-z0-9]{16}". This command is provided for convenience to generate tokens in the given format.

You can also use "kubeadm init" without specifying a token and it will generate and print one for you.

```
kubeadm token generate [flags]
```

Options

-h, --help

help for generate

Options inherited from parent commands

--dry-run

Whether to enable dry-run mode or not

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm token list

List bootstrap tokens on the server

Synopsis

This command will list all bootstrap tokens for you.

```
kubeadm token list [flags]
```

Options

--allow-missing-template-keys Default: true

If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to golang and jsonpath output formats.

-h, --help

help for list

-o, --output string Default: "text"

Output format. One of: text|json|yaml|go-template|go-template-file|template|templatefile|jsonpath|jsonpath-as-json|jsonpath-file.

--show-managed-fields

If true, keep the managedFields when printing objects in JSON or YAML format.

Options inherited from parent commands

--dry-run

Whether to enable dry-run mode or not

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

What's next

- [kubeadm join](#) to bootstrap a Kubernetes worker node and join it to the cluster

10.1.8 - kubeadm version

This command prints the version of kubeadm.

Print the version of kubeadm

Synopsis

Print the version of kubeadm

```
kubeadm version [flags]
```

Options

-h, --help

help for version

-o, --output string

Output format; available options are 'yaml', 'json' and 'short'

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.9 - kubeadm alpha

Caution:

`kubeadm alpha` provides a preview of a set of features made available for gathering feedback from the community. Please try it out and give us feedback!

Currently there are no experimental commands under `kubeadm alpha`.

What's next

- [`kubeadm init`](#) to bootstrap a Kubernetes control-plane node
- [`kubeadm join`](#) to connect a node to the cluster
- [`kubeadm reset`](#) to revert any changes made to this host by `kubeadm init` or `kubeadm join`

10.1.10 - kubeadm certs

`kubeadm certs` provides utilities for managing certificates. For more details on how these commands can be used, see [Certificate Management with kubeadm](#).

kubeadm certs

A collection of operations for operating Kubernetes certificates.

[overview](#)

Commands related to handling kubernetes certificates

Synopsis

Commands related to handling kubernetes certificates

```
kubeadm certs [flags]
```

Options

`-h, --help`

help for certs

Options inherited from parent commands

`--rootfs` string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm certs renew

You can renew all Kubernetes certificates using the `all` subcommand or renew them selectively. For more details see [Manual certificate renewal](#).

```
renew    all    admin.conf    apiserver-etcd-client  
apiserver-kubelet-client    apiserver    controller-manager.conf  
etcd-healthcheck-client    etcd-peer    etcd-server  
front-proxy-client    scheduler.conf    super-admin.conf
```

Renew certificates for a Kubernetes cluster

Synopsis

This command is not meant to be run on its own. See list of available subcommands.

```
kubeadm certs renew [flags]
```

Options

`-h, --help`

help for renew

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm certs certificate-key

This command can be used to generate a new control-plane certificate key. The key can be passed as `--certificate-key` to [kubeadm init](#) and [kubeadm join](#) to enable the automatic copy of certificates when joining additional control-plane nodes.

certificate-key

Generate certificate keys

Synopsis

This command will print out a secure randomly-generated certificate key that can be used with the "init" command.

You can also use "kubeadm init --upload-certs" without specifying a certificate key and it will generate and print one for you.

```
kubeadm certs certificate-key [flags]
```

Options

-h, --help

help for certificate-key

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm certs check-expiration

This command checks expiration for the certificates in the local PKI managed by kubeadm. For more details see [Check certificate expiration](#).

check-expiration

Check certificates expiration for a Kubernetes cluster

Synopsis

Checks expiration for the certificates in the local PKI managed by kubeadm.

```
kubeadm certs check-expiration [flags]
```

Options

--allow-missing-template-keys Default: true

If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to golang and jsonpath output formats.

--cert-dir string Default: "/etc/kubernetes/pki"
The path where to save the certificates

--config string
Path to a kubeadm configuration file.

-h, --help
help for check-expiration

--kubeconfig string Default: "/etc/kubernetes/admin.conf"
The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

-o, --output string Default: "text"
Output format. One of: text|json|yaml|go-template|go-template-file|template|templatefile|jsonpath|jsonpath-as-json|jsonpath-file.

--show-managed-fields
If true, keep the managedFields when printing objects in JSON or YAML format.

Options inherited from parent commands

--rootfs string
The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm certs generate-csr

This command can be used to generate keys and CSRs for all control-plane certificates and kubeconfig files. The user can then sign the CSRs with a CA of their choice. To read more information on how to use the command see [Signing certificate signing requests \(CSR\) generated by kubeadm](#).

`generate-csr`

Generate keys and certificate signing requests

Synopsis

Generates keys and certificate signing requests (CSRs) for all the certificates required to run the control plane. This command also generates partial kubeconfig files with private key data in the "users > user > client-key-data" field, and for each kubeconfig file an accompanying ".csr" file is created.

This command is designed for use in [Kubeadm External CA Mode](#). It generates CSRs which you can then submit to your external certificate authority for signing.

The PEM encoded signed certificates should then be saved alongside the key files, using ".crt" as the file extension, or in the case of kubeconfig files, the PEM encoded signed certificate should be base64 encoded and added to the kubeconfig file in the "users > user > client-certificate-data" field.

`kubeadm certs generate-csr [flags]`

Examples

```
# The following command will generate keys and CSRs for all control-plane certificates
kubeadm certs generate-csr --kubeconfig-dir /tmp/etc-k8s --cert-dir /tmp/etc-k8s/
```

Options

--cert-dir string

The path where to save the certificates

--config string

Path to a kubeadm configuration file.

-h, --help

help for generate-csr

--kubeconfig-dir string Default: "/etc/kubernetes"

The path where to save the kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

What's next

- [kubeadm init](#) to bootstrap a Kubernetes control-plane node
- [kubeadm join](#) to connect a node to the cluster
- [kubeadm reset](#) to revert any changes made to this host by `kubeadm init` or `kubeadm join`

10.1.11 - kubeadm init phase

`kubeadm init phase` enables you to invoke atomic steps of the bootstrap process. Hence, you can let kubeadm do some of the work and you can fill in the gaps if you wish to apply customization.

`kubeadm init phase` is consistent with the [kubeadm init workflow](#), and behind the scene both use the same code.

kubeadm init phase preflight

Using this command you can execute preflight checks on a control-plane node.

preflight

Run pre-flight checks

Synopsis

Run pre-flight checks for kubeadm init.

```
kubeadm init phase preflight [flags]
```

Examples

```
# Run pre-flight checks for kubeadm init using a config file.
kubeadm init phase preflight --config kubeadm-config.yaml
```

Options

`--config` string

Path to a kubeadm configuration file.

`--cri-socket` string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

`--dry-run`

Don't apply any changes; just output what would be done.

`-h, --help`

help for preflight

`--ignore-preflight-errors` strings

A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'. Value 'all' ignores errors from all checks.

`--image-repository` string Default: "registry.k8s.io"

Choose a container registry to pull control plane images from

Options inherited from parent commands

`--rootfs` string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm init phase kubelet-start

This phase will write the kubelet configuration file and environment file and then start the kubelet.

kubelet-start

Write kubelet settings and (re)start the kubelet

Synopsis

Write a file with KubeletConfiguration and an environment file with node specific kubelet settings, and then (re)start kubelet.

```
kubeadm init phase kubelet-start [flags]
```

Examples

```
# Writes a dynamic environment file with kubelet flags from a InitConfiguration +  
kubeadm init phase kubelet-start --config config.yaml
```

Options

--config string

Path to a kubeadm configuration file.

--cri-socket string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for kubelet-start

--image-repository string Default: "registry.k8s.io"

Choose a container registry to pull control plane images from

--node-name string

Specify the node name.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm init phase certs

Can be used to create all required certificates by kubeadm.

[certs](#) [all](#) [ca](#) [apiserver](#) [apiserver-kubelet-client](#)
[front-proxy-ca](#) [front-proxy-client](#) [etcd-ca](#) [etcd-server](#)
[etcd-peer](#) [healthcheck-client](#) [apiserver-etcd-client](#) [sa](#)

Certificate generation

Synopsis

This command is not meant to be run on its own. See list of available subcommands.

```
kubeadm init phase certs [flags]
```

Options

-h, --help
help for certs

Options inherited from parent commands

--rootfs string
The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm init phase kubeconfig

You can create all required kubeconfig files by calling the `all` subcommand or call them individually.

[kubeconfig](#) [all](#) [admin](#) [kubelet](#) [controller-manager](#)
[scheduler](#) [super-admin](#)

Generate all kubeconfig files necessary to establish the control plane and the admin kubeconfig file

Synopsis

This command is not meant to be run on its own. See list of available subcommands.

```
kubeadm init phase kubeconfig [flags]
```

Options

-h, --help
help for kubeconfig

Options inherited from parent commands

--rootfs string
The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm init phase control-plane

Using this phase you can create all required static Pod files for the control plane components.

[control-plane](#) [all](#) [apiserver](#) [controller-manager](#)
[scheduler](#)

Generate all static Pod manifest files necessary to establish the control plane

Synopsis

This command is not meant to be run on its own. See list of available subcommands.

```
kubeadm init phase control-plane [flags]
```

Options

-h, --help
help for control-plane

Options inherited from parent commands

--rootfs string
The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm init phase etcd

Use the following phase to create a local etcd instance based on a static Pod file.

[etcd](#) [local](#)

Generate static Pod manifest file for local etcd

Synopsis

This command is not meant to be run on its own. See list of available subcommands.

```
kubeadm init phase etcd [flags]
```

Options

-h, --help
help for etcd

Options inherited from parent commands

--rootfs string
The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm init phase upload-config

You can use this command to upload the kubeadm configuration to your cluster. Alternatively, you can use [kubeadm config](#).

[upload-config](#) [all](#) [kubeadm](#) [kubelet](#)

Upload the kubeadm and kubelet configuration to a ConfigMap

Synopsis

This command is not meant to be run on its own. See list of available subcommands.

```
kubeadm init phase upload-config [flags]
```

Options

-h, --help
help for upload-config

Options inherited from parent commands

--rootfs string
The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm init phase upload-certs

Use the following phase to upload control-plane certificates to the cluster. By default the certs and encryption key expire after two hours.

[upload-certs](#)

Upload certificates to kubeadm-certs

Synopsis

Upload control plane certificates to the kubeadm-certs Secret

```
kubeadm init phase upload-certs [flags]
```

Options

--certificate-key string
Key used to encrypt the control-plane certificates in the kubeadm-certs Secret. The certificate key is a hex encoded string that is an AES key of size 32 bytes.

--config string
Path to a kubeadm configuration file.

--dry-run
Don't apply any changes; just output what would be done.

-h, --help
help for upload-certs

--kubeconfig string Default: "/etc/kubernetes/admin.conf"
The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--skip-certificate-key-print

Don't print the key used to encrypt the control-plane certificates.

--upload-certs

Upload control-plane certificates to the kubeadm-certs Secret.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm init phase mark-control-plane

Use the following phase to label and taint the node as a control plane node.

mark-control-plane

Mark a node as a control-plane

Synopsis

Mark a node as a control-plane

```
kubeadm init phase mark-control-plane [flags]
```

Examples

```
# Applies control-plane label and taint to the current node, functionally equivalent to --control-plane
kubeadm init phase mark-control-plane --config config.yaml
```

```
# Applies control-plane label and taint to a specific node
kubeadm init phase mark-control-plane --node-name myNode
```

Options

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for mark-control-plane

--node-name string

Specify the node name.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm init phase bootstrap-token

Use the following phase to configure bootstrap tokens.

bootstrap-token

Generates bootstrap tokens used to join a node to a cluster

Synopsis

Bootstrap tokens are used for establishing bidirectional trust between a node joining the cluster and a control-plane node.

This command makes all the configurations required to make bootstrap tokens work and then creates an initial token.

```
kubeadm init phase bootstrap-token [flags]
```

Examples

```
# Make all the bootstrap token configurations and create an initial token, functionally
# equivalent to what generated by kubeadm init.
kubeadm init phase bootstrap-token
```

Options

--config string

Path to a kubeadm configuration file.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for bootstrap-token

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

--skip-token-print

Skip printing of the default bootstrap token generated by 'kubeadm init'.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm init phase kubelet-finalize

Use the following phase to update settings relevant to the kubelet after TLS bootstrap. You can use the `all` subcommand to run all `kubelet-finalize` phases.

kubelet-finalize

[kubelet-finalize-all](#)

[kubelet-finalize-cert-rotation](#)

Updates settings relevant to the kubelet after TLS bootstrap

Synopsis

Updates settings relevant to the kubelet after TLS bootstrap

```
kubeadm init phase kubelet-finalize [flags]
```

Examples

```
# Updates settings relevant to the kubelet after TLS bootstrap
kubeadm init phase kubelet-finalize all --config
```

Options

-h, --help

help for kubelet-finalize

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm init phase addon

You can install all the available addons with the `all` subcommand, or install them selectively.

[addon](#) [all](#) [coredns](#) [kube-proxy](#)

Install required addons for passing conformance tests

Synopsis

This command is not meant to be run on its own. See list of available subcommands.

```
kubeadm init phase addon [flags]
```

Options

-h, --help

help for addon

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

For more details on each field in the `v1beta4` configuration you can navigate to our [API reference pages](#).

What's next

- [kubeadm init](#) to bootstrap a Kubernetes control-plane node
- [kubeadm join](#) to connect a node to the cluster
- [kubeadm reset](#) to revert any changes made to this host by `kubeadm init` or `kubeadm join`
- [kubeadm alpha](#) to try experimental functionality

10.1.12 - kubeadm join phase

`kubeadm join phase` enables you to invoke atomic steps of the join process. Hence, you can let kubeadm do some of the work and you can fill in the gaps if you wish to apply customization.

`kubeadm join phase` is consistent with the [kubeadm join workflow](#), and behind the scene both use the same code.

kubeadm join phase

phase

Use this command to invoke single phase of the join workflow

Synopsis

Use this command to invoke single phase of the join workflow

Options

`-h, --help`

help for phase

Options inherited from parent commands

`--rootfs string`

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm join phase preflight

Using this phase you can execute preflight checks on a joining node.

preflight

Run join pre-flight checks

Synopsis

Run pre-flight checks for kubeadm join.

```
kubeadm join phase preflight [api-server-endpoint] [flags]
```

Examples

```
# Run join pre-flight checks using a config file.
kubeadm join phase preflight --config kubeadm-config.yaml
```

Options

`--apiserver-advertise-address string`

If the node should host a new control plane instance, the IP address the API Server will advertise it's listening on. If not set the default network interface will be used.

`--apiserver-bind-port int32` Default: 6443

If the node should host a new control plane instance, the port for the API Server to bind to.

--certificate-key string

Use this key to decrypt the certificate secrets uploaded by init. The certificate key is a hex encoded string that is an AES key of size 32 bytes.

--config string

Path to a kubeadm configuration file.

--control-plane

Create a new control plane instance on this node

--cri-socket string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

--discovery-file string

For file-based discovery, a file or URL from which to load cluster information.

--discovery-token string

For token-based discovery, the token used to validate cluster information fetched from the API server.

--discovery-token-ca-cert-hash strings

For token-based discovery, validate that the root CA public key matches this hash (format: "<type>:<value>").

--discovery-token-unsafe-skip-ca-verification

For token-based discovery, allow joining without --discovery-token-ca-cert-hash pinning.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for preflight

--ignore-preflight-errors strings

A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'. Value 'all' ignores errors from all checks.

--node-name string

Specify the node name.

--tls-bootstrap-token string

Specify the token used to temporarily authenticate with the Kubernetes Control Plane while joining the node.

--token string

Use this token for both discovery-token and tls-bootstrap-token when those values are not provided.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm join phase control-plane-prepare

Using this phase you can prepare a node for serving a control-plane.

[control-plane-prepare](#) [all](#) [download-certs](#) [certs](#)
[kubeconfig](#) [control-plane](#)

Prepare the machine for serving a control plane

Synopsis

Prepare the machine for serving a control plane

```
kubeadm join phase control-plane-prepare [flags]
```

Examples

```
# Prepares the machine for serving a control plane
kubeadm join phase control-plane-prepare all
```

Options

-h, --help

help for control-plane-prepare

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm join phase kubelet-start

Using this phase you can write the kubelet settings, certificates and (re)start the kubelet.

[kubelet-start](#)

Write kubelet settings, certificates and (re)start the kubelet

Synopsis

Write a file with KubeletConfiguration and an environment file with node specific kubelet settings, and then (re)start kubelet.

```
kubeadm join phase kubelet-start [api-server-endpoint] [flags]
```

Options

--config string

Path to a kubeadm configuration file.

--cri-socket string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

--discovery-file string

For file-based discovery, a file or URL from which to load cluster information.

--discovery-token string

For token-based discovery, the token used to validate cluster information fetched from the API server.

--discovery-token-ca-cert-hash strings

For token-based discovery, validate that the root CA public key matches this hash (format: "<type>:<value>").

--discovery-token-unsafe-skip-ca-verification

For token-based discovery, allow joining without --discovery-token-ca-cert-hash pinning.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for kubelet-start

--node-name string

Specify the node name.

--patches string

Path to a directory that contains files named "target[suffix][+patchtype].extension". For example, "kube-apiserver0+merge.yaml" or just "etcd.json". "target" can be one of "kube-apiserver", "kube-controller-manager", "kube-scheduler", "etcd", "kubeletconfiguration", "corednsdeployment". "patchtype" can be one of "strategic", "merge" or "json" and they match the patch formats supported by kubectl. The default "patchtype" is "strategic". "extension" must be either "json" or "yaml". "suffix" is an optional string that can be used to determine which patches are applied first alpha-numerically.

--tls-bootstrap-token string

Specify the token used to temporarily authenticate with the Kubernetes Control Plane while joining the node.

--token string

Use this token for both discovery-token and tls-bootstrap-token when those values are not provided.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm join phase control-plane-join

Using this phase you can join a node as a control-plane instance.

[control-plane-join](#)

[all](#)

[etcd](#)

[mark-control-plane](#)

Join a machine as a control plane instance

Synopsis

Join a machine as a control plane instance

```
kubeadm join phase control-plane-join [flags]
```

Examples

```
# Joins a machine as a control plane instance
kubeadm join phase control-plane-join all
```

Options

-h, --help

help for control-plane-join

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

What's next

- [kubeadm init](#) to bootstrap a Kubernetes control-plane node
- [kubeadm join](#) to connect a node to the cluster
- [kubeadm reset](#) to revert any changes made to this host by `kubeadm init` or `kubeadm join`
- [kubeadm alpha](#) to try experimental functionality

10.1.13 - kubeadm kubeconfig

`kubeadm kubeconfig` provides utilities for managing kubeconfig files.

For examples on how to use `kubeadm kubeconfig` user see [Generating kubeconfig files for additional users](#).

kubeadm kubeconfig

overview

Kubeconfig file utilities

Synopsis

Kubeconfig file utilities.

Options

`-h, --help`

help for kubeconfig

Options inherited from parent commands

`--rootfs` string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm kubeconfig user

This command can be used to output a kubeconfig file for an additional user.

user

Output a kubeconfig file for an additional user

Synopsis

Output a kubeconfig file for an additional user.

```
kubeadm kubeconfig user [flags]
```

Examples

```
# Output a kubeconfig file for an additional user named foo
kubeadm kubeconfig user --client-name=foo

# Output a kubeconfig file for an additional user named foo using a kubeadm config
kubeadm kubeconfig user --client-name=foo --config=bar
```

Options

`--client-name` string

The name of user. It will be used as the CN if client certificates are created

--config string

Path to a kubeadm configuration file.

-h, --help

help for user

--org strings

The organizations of the client certificate. It will be used as the O if client certificates are created

--token string

The token that should be used as the authentication mechanism for this kubeconfig, instead of client certificates

--validity-period duration Default: 8760h0m0s

The validity period of the client certificate. It is an offset from the current time.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

10.1.14 - kubeadm reset phase

`kubeadm reset phase` enables you to invoke atomic steps of the node reset process. Hence, you can let kubeadm do some of the work and you can fill in the gaps if you wish to apply customization.

`kubeadm reset phase` is consistent with the [kubeadm reset workflow](#), and behind the scene both use the same code.

kubeadm reset phase

phase

Use this command to invoke single phase of the reset workflow

Synopsis

Use this command to invoke single phase of the reset workflow

Options

`-h, --help`

help for phase

Options inherited from parent commands

`--rootfs string`

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm reset phase preflight

Using this phase you can execute preflight checks on a node that is being reset.

preflight

Run reset pre-flight checks

Synopsis

Run pre-flight checks for kubeadm reset.

```
kubeadm reset phase preflight [flags]
```

Options

`--dry-run`

Don't apply any changes; just output what would be done.

`-f, --force`

Reset the node without prompting for confirmation.

`-h, --help`

help for preflight

--ignore-preflight-errors strings

A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'. Value 'all' ignores errors from all checks.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm reset phase remove-etcd-member

Using this phase you can remove this control-plane node's etcd member from the etcd cluster.

remove-etcd-member

Remove a local etcd member.

Synopsis

Remove a local etcd member for a control plane node.

```
kubeadm reset phase remove-etcd-member [flags]
```

Options

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for remove-etcd-member

--kubeconfig string Default: "/etc/kubernetes/admin.conf"

The kubeconfig file to use when talking to the cluster. If the flag is not set, a set of standard locations can be searched for an existing kubeconfig file.

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

kubeadm reset phase cleanup-node

Using this phase you can perform cleanup on this node.

cleanup-node

Run cleanup node.

Synopsis

Run cleanup node.

```
kubeadm reset phase cleanup-node [flags]
```

Options

--cert-dir string Default: "/etc/kubernetes/pki"

The path to the directory where the certificates are stored. If specified, clean this directory.

--cleanup-tmp-dir

Cleanup the "/etc/kubernetes/tmp" directory

--cri-socket string

Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use this option only if you have more than one CRI installed or if you have non-standard CRI socket.

--dry-run

Don't apply any changes; just output what would be done.

-h, --help

help for cleanup-node

Options inherited from parent commands

--rootfs string

The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

What's next

- [kubeadm init](#) to bootstrap a Kubernetes control-plane node
- [kubeadm join](#) to connect a node to the cluster
- [kubeadm reset](#) to revert any changes made to this host by `kubeadm init` or `kubeadm join`
- [kubeadm alpha](#) to try experimental functionality

10.1.15 - kubeadm upgrade phase

In v1.15.0, kubeadm introduced preliminary support for `kubeadm upgrade node` phases. Phases for other `kubeadm upgrade` sub-commands such as `apply`, could be added in the following releases.

kubeadm upgrade node phase

Using this phase you can choose to execute the separate steps of the upgrade of secondary control-plane or worker nodes. Please note that `kubeadm upgrade apply` still has to be called on a primary control-plane node.

[phase](#) [preflight](#) [control-plane](#) [kubelet-config](#)

Use this command to invoke single phase of the node workflow

Synopsis

Use this command to invoke single phase of the node workflow

Options

`-h, --help`
help for phase

Options inherited from parent commands

`--rootfs` string
The path to the 'real' host root filesystem. This will cause kubeadm to chroot into the provided path.

What's next

- [kubeadm init](#) to bootstrap a Kubernetes control-plane node
- [kubeadm join](#) to connect a node to the cluster
- [kubeadm reset](#) to revert any changes made to this host by `kubeadm init` or `kubeadm join`
- [kubeadm upgrade](#) to upgrade a kubeadm node
- [kubeadm alpha](#) to try experimental functionality

10.1.16 - Implementation details

ⓘ FEATURE STATE: Kubernetes v1.10 [stable]

`kubeadm init` and `kubeadm join` together provide a nice user experience for creating a bare Kubernetes cluster from scratch, that aligns with the best-practices. However, it might not be obvious *how* `kubeadm` does that.

This document provides additional details on what happens under the hood, with the aim of sharing knowledge on the best practices for a Kubernetes cluster.

Core design principles

The cluster that `kubeadm init` and `kubeadm join` set up should be:

- **Secure:** It should adopt latest best-practices like:
 - enforcing RBAC
 - using the Node Authorizer
 - using secure communication between the control plane components
 - using secure communication between the API server and the kubelets
 - lock-down the kubelet API
 - locking down access to the API for system components like the kube-proxy and CoreDNS
 - locking down what a Bootstrap Token can access
- **User-friendly:** The user should not have to run anything more than a couple of commands:
 - `kubeadm init`
 - `export KUBECONFIG=/etc/kubernetes/admin.conf`
 - `kubectl apply -f <network-of-choice.yaml>`
 - `kubeadm join --token <token> <endpoint>:<port>`
- **Extendable:**
 - It should *not* favor any particular network provider.
Configuring the cluster network is out-of-scope
 - It should provide the possibility to use a config file for customizing various parameters

Constants and well-known values and paths

In order to reduce complexity and to simplify development of higher level tools that build on top of `kubeadm`, it uses a limited set of constant values for well-known paths and file names.

The Kubernetes directory `/etc/kubernetes` is a constant in the application, since it is clearly the given path in a majority of cases, and the most intuitive location; other constant paths and file names are:

- `/etc/kubernetes/manifests` as the path where the kubelet should look for static Pod manifests. Names of static Pod manifests are:
 - `etcd.yaml`

- kube-apiserver.yaml
- kube-controller-manager.yaml
- kube-scheduler.yaml
- /etc/kubernetes/ as the path where kubeconfig files with identities for control plane components are stored. Names of kubeconfig files are:
 - kubelet.conf (bootstrap-kubelet.conf during TLS bootstrap)
 - controller-manager.conf
 - scheduler.conf
 - admin.conf for the cluster admin and kubeadm itself
 - super-admin.conf for the cluster super-admin that can bypass RBAC
- Names of certificates and key files:
 - ca.crt , ca.key for the Kubernetes certificate authority
 - apiserver.crt , apiserver.key for the API server certificate
 - apiserver-kubelet-client.crt , apiserver-kubelet-client.key for the client certificate used by the API server to connect to the kubelets securely
 - sa.pub , sa.key for the key used by the controller manager when signing ServiceAccount
 - front-proxy-ca.crt , front-proxy-ca.key for the front proxy certificate authority
 - front-proxy-client.crt , front-proxy-client.key for the front proxy client

kubeadm init workflow internal design

The `kubeadm init` consists of a sequence of atomic work tasks to perform, as described in the [kubeadm init internal workflow](#).

The [kubeadm init phase](#) command allows users to invoke each task individually, and ultimately offers a reusable and composable API/toolbox that can be used by other Kubernetes bootstrap tools, by any IT automation tool or by an advanced user for creating custom clusters.

Preflight checks

Kubeadm executes a set of preflight checks before starting the init, with the aim to verify preconditions and avoid common cluster startup problems. The user can skip specific preflight checks or all of them with the `--ignore-preflight-errors` option.

- [Warning] if the Kubernetes version to use (specified with the `--kubernetes-version` flag) is at least one minor version higher than the kubeadm CLI version.
- Kubernetes system requirements:
 - if running on linux:
 - [Error] if Kernel is older than the minimum required version
 - [Error] if required cgroups subsystem aren't set up
 - [Error] if the CRI endpoint does not answer
 - [Error] if user is not root
 - [Error] if the machine hostname is not a valid DNS subdomain
 - [Warning] if the host name cannot be reached via network lookup

- [Error] if kubelet version is lower than the minimum kubelet version supported by kubeadm (current minor -1)
- [Error] if kubelet version is at least one minor higher than the required controlplane version (unsupported version skew)
- [Warning] if kubelet service does not exist or if it is disabled
- [Warning] if firewalld is active
- [Error] if API server bindPort or ports 10250/10251/10252 are used
- [Error] if `/etc/kubernetes/manifest` folder already exists and it is not empty
- [Error] if swap is on
- [Error] if `conntrack`, `ip`, `iptables`, `mount`, `nsenter` commands are not present in the command path
- [Warning] if `ebtables`, `ethtool`, `socat`, `tc`, `touch`, `crictl` commands are not present in the command path
- [Warning] if extra arg flags for API server, controller manager, scheduler contains some invalid options
- [Warning] if connection to `https://` API.AdvertiseAddress:API.BindPort goes through proxy
- [Warning] if connection to services subnet goes through proxy (only first address checked)
- [Warning] if connection to Pods subnet goes through proxy (only first address checked)
- If external etcd is provided:
 - [Error] if etcd version is older than the minimum required version
 - [Error] if etcd certificates or keys are specified, but not provided
- If external etcd is NOT provided (and thus local etcd will be installed):
 - [Error] if ports 2379 is used
 - [Error] if Etcd.DataDir folder already exists and it is not empty
- If authorization mode is ABAC:
 - [Error] if `abac_policy.json` does not exist
- If authorization mode is WebHook
 - [Error] if `webhook_authz.conf` does not exist

Note:

Preflight checks can be invoked individually with the [kubeadm init phase preflight](#) command.

Generate the necessary certificates

Kubeadm generates certificate and private key pairs for different purposes:

- A self signed certificate authority for the Kubernetes cluster saved into `ca.crt` file and `ca.key` private key file
- A serving certificate for the API server, generated using `ca.crt` as the CA, and saved into `apiserver.crt` file with its private key `apiserver.key`. This certificate should contain the following alternative names:
 - The Kubernetes service's internal clusterIP (the first address in the services CIDR, e.g. `10.96.0.1` if service subnet is `10.96.0.0/12`)
 - Kubernetes DNS names, e.g. `kubernetes.default.svc.cluster.local` if `--service-dns-domain`

- flag value is `cluster.local`, plus default DNS names `kubernetes.default.svc`, `kubernetes.default`, `kubernetes`
 - The node-name
 - The `--apiserver-advertise-address`
 - Additional alternative names specified by the user
- A client certificate for the API server to connect to the kubelets securely, generated using `ca.crt` as the CA and saved into `apiserver-kubelet-client.crt` file with its private key `apiserver-kubelet-client.key`. This certificate should be in the `system:masters` organization
- A private key for signing ServiceAccount Tokens saved into `sa.key` file along with its public key `sa.pub`
- A certificate authority for the front proxy saved into `front-proxy-ca.crt` file with its key `front-proxy-ca.key`
- A client certificate for the front proxy client, generated using `front-proxy-ca.crt` as the CA and saved into `front-proxy-client.crt` file with its private key `front-proxy-client.key`

Certificates are stored by default in `/etc/kubernetes/pki`, but this directory is configurable using the `--cert-dir` flag.

Please note that:

1. If a given certificate and private key pair both exist, and their content is evaluated to be compliant with the above specs, the existing files will be used and the generation phase for the given certificate will be skipped. This means the user can, for example, copy an existing CA to `/etc/kubernetes/pki/ca.{crt,key}`, and then kubeadm will use those files for signing the rest of the certs. See also [using custom certificates](#)
2. For the CA, it is possible to provide the `ca.crt` file but not the `ca.key` file. If all other certificates and kubeconfig files are already in place, kubeadm recognizes this condition and activates the `ExternalCA`, which also implies the `csrsigner` controller in `controller-manager` won't be started
3. If kubeadm is running in [external CA mode](#); all the certificates must be provided by the user, because kubeadm cannot generate them by itself
4. In case kubeadm is executed in the `--dry-run` mode, certificate files are written in a temporary folder
5. Certificate generation can be invoked individually with the [kubeadm init phase certs all](#) command

Generate kubeconfig files for control plane components

Kubeadm generates kubeconfig files with identities for control plane components:

- A kubeconfig file for the kubelet to use during TLS bootstrap - `/etc/kubernetes/bootstrap-kubelet.conf`. Inside this file, there is a bootstrap-token or embedded client certificates for authenticating this node with the cluster.

This client certificate should:

- Be in the `system:nodes` organization, as required by the [Node Authorization](#) module
- Have the Common Name (CN) `system:node:<hostname-lowercased>`
- A kubeconfig file for controller-manager, `/etc/kubernetes/`

`controller-manager.conf` ; inside this file is embedded a client certificate with controller-manager identity. This client certificate should have the CN `system:kube-controller-manager` , as defined by default [RBAC core components roles](#)

- A kubeconfig file for scheduler, `/etc/kubernetes/scheduler.conf` ; inside this file is embedded a client certificate with scheduler identity. This client certificate should have the CN `system:kube-scheduler` , as defined by default [RBAC core components roles](#)

Additionally, a kubeconfig file for kubeadm as an administrative entity is generated and stored in `/etc/kubernetes/admin.conf` . This file includes a certificate with `Subject: O = kubeadm:cluster-admins, CN = kubernetes-admin` . `kubeadm:cluster-admins` is a group managed by kubeadm. It is bound to the `cluster-admin` ClusterRole during `kubeadm init` , by using the `super-admin.conf` file, which does not require RBAC. This `admin.conf` file must remain on control plane nodes and should not be shared with additional users.

During `kubeadm init` another kubeconfig file is generated and stored in `/etc/kubernetes/super-admin.conf` . This file includes a certificate with `Subject: O = system:masters, CN = kubernetes-super-admin` . `system:masters` is a superuser group that bypasses RBAC and makes `super-admin.conf` useful in case of an emergency where a cluster is locked due to RBAC misconfiguration. The `super-admin.conf` file must be stored in a safe location and should not be shared with additional users.

See [RBAC user facing role bindings](#) for additional information on RBAC and built-in ClusterRoles and groups.

Please note that:

1. `ca.crt` certificate is embedded in all the kubeconfig files.
2. If a given kubeconfig file exists, and its content is evaluated as compliant with the above specs, the existing file will be used and the generation phase for the given kubeconfig will be skipped
3. If kubeadm is running in [ExternalCA mode](#), all the required kubeconfig must be provided by the user as well, because kubeadm cannot generate any of them by itself
4. In case kubeadm is executed in the `--dry-run` mode, kubeconfig files are written in a temporary folder
5. Generation of kubeconfig files can be invoked individually with the [kubeadm init phase kubeconfig all](#) command

Generate static Pod manifests for control plane components

Kubeadm writes static Pod manifest files for control plane components to `/etc/kubernetes/manifests` . The kubelet watches this directory for Pods to be created on startup.

Static Pod manifests share a set of common properties:

- All static Pods are deployed on `kube-system` namespace
- All static Pods get `tier:control-plane` and `component:{component-name}` labels
- All static Pods use the `system-node-critical` priority class
- `hostNetwork: true` is set on all static Pods to allow control plane startup before a network is configured; as a consequence:
 - The `address` that the controller-manager and the scheduler use to refer to the API server is `127.0.0.1`

- If the etcd server is set up locally, the `etcd-server` address will be set to `127.0.0.1:2379`
- Leader election is enabled for both the controller-manager and the scheduler
- Controller-manager and the scheduler will reference kubeconfig files with their respective, unique identities
- All static Pods get any extra flags specified by the user as described in [passing custom arguments to control plane components](#)
- All static Pods get any extra Volumes specified by the user (Host path)

Please note that:

1. All images will be pulled from `registry.k8s.io` by default. See [using custom images](#) for customizing the image repository
2. In case `kubeadm` is executed in the `--dry-run` mode, static Pod files are written in a temporary folder
3. Static Pod manifest generation for control plane components can be invoked individually with the [kubeadm init phase control-plane all](#) command

API server

The static Pod manifest for the API server is affected by the following parameters provided by the users:

- The `apiserver-advertise-address` and `apiserver-bind-port` to bind to; if not provided, those values default to the IP address of the default network interface on the machine and port 6443
- The `service-cluster-ip-range` to use for services
- If an external etcd server is specified, the `etcd-servers` address and related TLS settings (`etcd-cafile`, `etcd-certfile`, `etcd-keyfile`); if an external etcd server is not provided, a local etcd will be used (via host network)
- If a cloud provider is specified, the corresponding `--cloud-provider` parameter is configured together with the `--cloud-config` path if such file exists (this is experimental, alpha and will be removed in a future version)

Other API server flags that are set unconditionally are:

- `--insecure-port=0` to avoid insecure connections to the api server
- `--enable-bootstrap-token-auth=true` to enable the `BootstrapTokenAuthenticator` authentication module. See [TLS Bootstrapping](#) for more details
- `--allow-privileged` to `true` (required e.g. by kube proxy)
- `--requestheader-client-ca-file` to `front-proxy-ca.crt`
- `--enable-admission-plugins` to:
 - [NamespaceLifecycle](#) e.g. to avoid deletion of system reserved namespaces
 - [LimitRanger](#) and [ResourceQuota](#) to enforce limits on namespaces
 - [ServiceAccount](#) to enforce service account automation
 - [PersistentVolumeLabel](#) attaches region or zone labels to PersistentVolumes as defined by the cloud provider (This admission controller is deprecated and will be removed in a future version. It is not deployed by kubeadm by default with

- v1.9 onwards when not explicitly opting into using `gce` or `aws` as cloud providers)
 - [DefaultStorageClass](#) to enforce default storage class on `PersistentVolumeClaim` objects
 - [DefaultTolerationSeconds](#)
 - [NodeRestriction](#) to limit what a kubelet can modify (e.g. only pods on this node)
- `--kubelet-preferred-address-types` to `InternalIP,ExternalIP,Hostname`; this makes `kubectl` logs and other API server-kubelet communication work in environments where the hostnames of the nodes aren't resolvable
- Flags for using certificates generated in previous steps:
 - `--client-ca-file` to `ca.crt`
 - `--tls-cert-file` to `apiserver.crt`
 - `--tls-private-key-file` to `apiserver.key`
 - `--kubelet-client-certificate` to `apiserver-kubelet-client.crt`
 - `--kubelet-client-key` to `apiserver-kubelet-client.key`
 - `--service-account-key-file` to `sa.pub`
 - `--requestheader-client-ca-file` to `front-proxy-ca.crt`
 - `--proxy-client-cert-file` to `front-proxy-client.crt`
 - `--proxy-client-key-file` to `front-proxy-client.key`
- Other flags for securing the front proxy ([API Aggregation](#)) communications:
 - `--requestheader-username-headers=X-Remote-User`
 - `--requestheader-group-headers=X-Remote-Group`
 - `--requestheader-extra-headers-prefix=X-Remote-Extra-`
 - `--requestheader-allowed-names=front-proxy-client`

Controller manager

The static Pod manifest for the controller manager is affected by following parameters provided by the users:

- If `kubeadm` is invoked specifying a `--pod-network-cidr`, the subnet manager feature required for some CNI network plugins is enabled by setting:
 - `--allocate-node-cidrs=true`
 - `--cluster-cidr` and `--node-cidr-mask-size` flags according to the given CIDR
- If a cloud provider is specified, the corresponding `--cloud-provider` is specified together with the `--cloud-config` path if such configuration file exists (this is experimental, alpha and will be removed in a future version)

Other flags that are set unconditionally are:

- `--controllers` enabling all the default controllers plus `BootstrapSigner` and `TokenCleaner` controllers for TLS bootstrap. See [TLS Bootstrapping](#) for more details.
- `--use-service-account-credentials` to `true`
- Flags for using certificates generated in previous steps:
 - `--root-ca-file` to `ca.crt`
 - `--cluster-signing-cert-file` to `ca.crt`, if External CA mode is disabled, otherwise to `""`

- `--cluster-signing-key-file` to `ca.key`, if External CA mode is disabled, otherwise to `""`
- `--service-account-private-key-file` to `sa.key`

Scheduler

The static Pod manifest for the scheduler is not affected by parameters provided by the users.

Generate static Pod manifest for local etcd

If you specified an external etcd, this step will be skipped, otherwise kubeadm generates a static Pod manifest file for creating a local etcd instance running in a Pod with following attributes:

- listen on `localhost:2379` and use `HostNetwork=true`
- make a `hostPath` mount out from the `dataDir` to the host's filesystem
- Any extra flags specified by the user

Please note that:

1. The etcd container image will be pulled from `registry.gcr.io` by default. See [using custom images](#) for customizing the image repository.
2. If you run kubeadm in `--dry-run` mode, the etcd static Pod manifest is written into a temporary folder.
3. You can directly invoke static Pod manifest generation for local etcd, using the [kubeadm init phase etcd local](#) command.

Wait for the control plane to come up

kubeadm waits (upto 4m0s) until `localhost:6443/healthz` (kube-apiserver liveness) returns `ok`. However, in order to detect deadlock conditions, kubeadm fails fast if `localhost:10255/healthz` (kubelet liveness) or `localhost:10255/healthz/syncloop` (kubelet readiness) don't return `ok` within 40s and 60s respectively.

kubeadm relies on the kubelet to pull the control plane images and run them properly as static Pods. After the control plane is up, kubeadm completes the tasks described in following paragraphs.

Save the kubeadm ClusterConfiguration in a ConfigMap for later reference

kubeadm saves the configuration passed to `kubeadm init` in a ConfigMap named `kubeadm-config` under `kube-system` namespace.

This will ensure that kubeadm actions executed in future (e.g `kubeadm upgrade`) will be able to determine the actual/current cluster state and make new decisions based on that data.

Please note that:

1. Before saving the ClusterConfiguration, sensitive information like the token is stripped from the configuration
2. Upload of control plane node configuration can be invoked individually with the command [kubeadm init phase upload-config](#).

Mark the node as control-plane

As soon as the control plane is available, kubeadm executes the following

actions:

- Labels the node as control-plane with `node-role.kubernetes.io/control-plane=""`
- Taints the node with `node-role.kubernetes.io/control-plane:NoSchedule`

Please note that the phase to mark the control-plane phase can be invoked individually with the [kubeadm init phase mark-control-plane](#) command.

Configure TLS-Bootstrapping for node joining

Kubeadm uses [Authenticating with Bootstrap Tokens](#) for joining new nodes to an existing cluster; for more details see also [design proposal](#).

`kubeadm init` ensures that everything is properly configured for this process, and this includes following steps as well as setting API server and controller flags as already described in previous paragraphs.

Note:

TLS bootstrapping for nodes can be configured with the command [kubeadm init phase bootstrap-token](#), executing all the configuration steps described in following paragraphs; alternatively, each step can be invoked individually.

Create a bootstrap token

`kubeadm init` creates a first bootstrap token, either generated automatically or provided by the user with the `--token` flag; as documented in bootstrap token specification, token should be saved as a secret with name `bootstrap-token-<token-id>` under `kube-system` namespace.

Please note that:

1. The default token created by `kubeadm init` will be used to validate temporary user during TLS bootstrap process; those users will be member of `system:bootstrappers:kubeadm:default-node-token` group
2. The token has a limited validity, default 24 hours (the interval may be changed with the `-token-ttl` flag)
3. Additional tokens can be created with the [kubeadm token](#) command, that provide other useful functions for token management as well.

Allow joining nodes to call CSR API

Kubeadm ensures that users in `system:bootstrappers:kubeadm:default-node-token` group are able to access the certificate signing API.

This is implemented by creating a ClusterRoleBinding named `kubeadm:kubelet-bootstrap` between the group above and the default RBAC role `system:node-bootstrapper`.

Set up auto approval for new bootstrap tokens

Kubeadm ensures that the Bootstrap Token will get its CSR request automatically approved by the `csapprover` controller.

This is implemented by creating ClusterRoleBinding named `kubeadm:node-autoapprove-bootstrap` between the `system:bootstrappers:kubeadm:default-node-token` group and the default role `system:certificates.k8s.io:certificatesigningrequests:nodeclient`.

The role

```
system:certificates.k8s.io:certificatesigningrequests:nodeclient should  
be created as well, granting POST permission to /apis/  
certificates.k8s.io/certificatesigningrequests/nodeclient .
```

Set up nodes certificate rotation with auto approval

Kubeadm ensures that certificate rotation is enabled for nodes, and that a new certificate request for nodes will get its CSR request automatically approved by the csrapprover controller.

This is implemented by creating ClusterRoleBinding named `kubeadm:node-autoapprove-certificate-rotation` between the `system:nodes` group and the default role `system:certificates.k8s.io:certificatesigningrequests:selfnodeclient`.

Create the public cluster-info ConfigMap

This phase creates the `cluster-info` ConfigMap in the `kube-public` namespace.

Additionally, it creates a Role and a RoleBinding granting access to the ConfigMap for unauthenticated users (i.e. users in RBAC group `system:unauthenticated`).

Note:

The access to the `cluster-info` ConfigMap is *not* rate-limited. This may or may not be a problem if you expose your cluster's API server to the internet; worst-case scenario here is a DoS attack where an attacker uses all the in-flight requests the kube-apiserver can handle to serve the `cluster-info` ConfigMap.

Install addons

Kubeadm installs the internal DNS server and the kube-proxy addon components via the API server.

Note:

This phase can be invoked individually with the command [`kubeadm init phase addon all`](#).

proxy

A ServiceAccount for `kube-proxy` is created in the `kube-system` namespace; then `kube-proxy` is deployed as a DaemonSet:

- The credentials (`ca.crt` and `token`) to the control plane come from the ServiceAccount
- The location (URL) of the API server comes from a ConfigMap
- The `kube-proxy` ServiceAccount is bound to the privileges in the `system:node-proxier` ClusterRole

DNS

- The CoreDNS service is named `kube-dns`. This is done to prevent any interruption in service when the user is switching the cluster DNS from `kube-dns` to CoreDNS through the `--config` method described [here](#).

- A ServiceAccount for CoreDNS is created in the `kube-system` namespace.
- The `coredns` ServiceAccount is bound to the privileges in the `system:coredns` ClusterRole

In Kubernetes version 1.21, support for using `kube-dns` with kubeadm was removed. You can use CoreDNS with kubeadm even when the related Service is named `kube-dns`.

kubeadm join phases internal design

Similarly to `kubeadm init`, also `kubeadm join` internal workflow consists of a sequence of atomic work tasks to perform.

This is split into discovery (having the Node trust the Kubernetes Master) and TLS bootstrap (having the Kubernetes Master trust the Node).

see [Authenticating with Bootstrap Tokens](#) or the corresponding [design proposal](#).

Preflight checks

`kubeadm` executes a set of preflight checks before starting the join, with the aim to verify preconditions and avoid common cluster startup problems.

Please note that:

1. `kubeadm join` preflight checks are basically a subset of `kubeadm init` preflight checks
2. Starting from 1.24, kubeadm uses crictl to communicate to all known CRI endpoints.
3. Starting from 1.9, kubeadm provides support for joining nodes running on Windows; in that case, linux specific controls are skipped.
4. In any case the user can skip specific preflight checks (or eventually all preflight checks) with the `--ignore-preflight-errors` option.

Discovery cluster-info

There are 2 main schemes for discovery. The first is to use a shared token along with the IP address of the API server. The second is to provide a file (that is a subset of the standard kubeconfig file).

Shared token discovery

If `kubeadm join` is invoked with `--discovery-token`, token discovery is used; in this case the node basically retrieves the cluster CA certificates from the `cluster-info` ConfigMap in the `kube-public` namespace.

In order to prevent "man in the middle" attacks, several steps are taken:

- First, the CA certificate is retrieved via insecure connection (this is possible because `kubeadm init` is granted access to `cluster-info` users for `system:unauthenticated`)
- Then the CA certificate goes through following validation steps:
 - Basic validation: using the token ID against a JWT signature
 - Pub key validation: using provided `--discovery-token-ca-cert-`

hash. This value is available in the output of `kubeadm init` or can be calculated using standard tools (the hash is calculated over the bytes of the Subject Public Key Info (SPKI) object as in RFC7469). The `--discovery-token-ca-cert-hash` flag may be repeated multiple times to allow more than one public key.

- As an additional validation, the CA certificate is retrieved via secure connection and then compared with the CA retrieved initially

Note:

Pub key validation can be skipped passing `--discovery-token-unsafe-skip-ca-verification` flag; This weakens the `kubeadm` security model since others can potentially impersonate the Kubernetes Master.

File/https discovery

If `kubeadm join` is invoked with `--discovery-file`, file discovery is used; this file can be a local file or downloaded via an HTTPS URL; in case of HTTPS, the host installed CA bundle is used to verify the connection.

With file discovery, the cluster CA certificate is provided into the file itself; in fact, the discovery file is a kubeconfig file with only `server` and `certificate-authority-data` attributes set, as described in the [kubeadm join](#) reference doc; when the connection with the cluster is established, `kubeadm` tries to access the `cluster-info` ConfigMap, and if available, uses it.

TLS Bootstrap

Once the cluster info is known, the file `bootstrap-kubelet.conf` is written, thus allowing `kubelet` to do TLS Bootstrapping.

The TLS bootstrap mechanism uses the shared token to temporarily authenticate with the Kubernetes API server to submit a certificate signing request (CSR) for a locally created key pair.

The request is then automatically approved and the operation completes saving `ca.crt` file and `kubelet.conf` file to be used by the `kubelet` for joining the cluster, while `bootstrap-kubelet.conf` is deleted.

Note:

- The temporary authentication is validated against the token saved during the `kubeadm init` process (or with additional tokens created with `kubeadm token` command)
- The temporary authentication resolves to a user member of `system:bootstrappers:kubeadm:default-node-token` group which was granted access to the CSR api during the `kubeadm init` process
- The automatic CSR approval is managed by the `csrapprover` controller, according to the configuration present in the `kubeadm init` process

11 - Command line tool (kubectl)

Kubernetes provides a command line tool for communicating with a Kubernetes cluster's control plane, using the Kubernetes API.

This tool is named `kubectl`.

For configuration, `kubectl` looks for a file named `config` in the `$HOME/.kube` directory. You can specify other `kubeconfig` files by setting the `KUBECONFIG` environment variable or by setting the `--kubeconfig` flag.

This overview covers `kubectl` syntax, describes the command operations, and provides common examples. For details about each command, including all the supported flags and subcommands, see the [kubectl](#) reference documentation.

For installation instructions, see [Installing kubectl](#); for a quick guide, see the [cheat sheet](#). If you're used to using the `docker` command-line tool, [kubectl for Docker Users](#) explains some equivalent commands for Kubernetes.

Syntax

Use the following syntax to run `kubectl` commands from your terminal window:

```
kubectl [command] [TYPE] [NAME] [flags]
```

where `command`, `TYPE`, `NAME`, and `flags` are:

- `command` : Specifies the operation that you want to perform on one or more resources, for example `create`, `get`, `describe`, `delete`.
- `TYPE` : Specifies the [resource type](#). Resource types are case-insensitive and you can specify the singular, plural, or abbreviated forms. For example, the following commands produce the same output:

```
kubectl get pod pod1
kubectl get pods pod1
kubectl get po pod1
```

- `NAME` : Specifies the name of the resource. Names are case-sensitive. If the name is omitted, details for all resources are displayed, for example `kubectl get pods`.

When performing an operation on multiple resources, you can specify each resource by type and name or specify one or more files:

- To specify resources by type and name:
 - To group resources if they are all the same type: `TYPE1 name1 name2 name<#>`.
Example: `kubectl get pod example-pod1 example-pod2`
 - To specify multiple resource types individually: `TYPE1/ name1 TYPE1/name2 TYPE2/name3 TYPE<#>/name<#>`.

- Example: `kubectl get pod/example-pod1 replicationcontroller/example-rc1`
- To specify resources with one or more files: `-f file1 -f file2 -f file<#>`
 - [Use YAML rather than JSON](#) since YAML tends to be more user-friendly, especially for configuration files.
 - `flags` : Specifies optional flags. For example, you can use the `-s` or `--server` flags to specify the address and port of the Kubernetes API server.

Caution:

Flags that you specify from the command line override default values and any corresponding environment variables.

If you need help, run `kubectl help` from the terminal window.

In-cluster authentication and namespace overrides

By default `kubectl` will first determine if it is running within a pod, and thus in a cluster. It starts by checking for the `KUBERNETES_SERVICE_HOST` and `KUBERNETES_SERVICE_PORT` environment variables and the existence of a service account token file at `/var/run/secrets/kubernetes.io/serviceaccount/token`. If all three are found in-cluster authentication is assumed.

To maintain backwards compatibility, if the `POD_NAMESPACE` environment variable is set during in-cluster authentication it will override the default namespace from the service account token. Any manifests or tools relying on namespace defaulting will be affected by this.

`POD_NAMESPACE` environment variable

If the `POD_NAMESPACE` environment variable is set, cli operations on namespaced resources will default to the variable value. For example, if the variable is set to `seattle` , `kubectl get pods` would return pods in the `seattle` namespace. This is because pods are a namespaced resource, and no namespace was provided in the command. Review the output of `kubectl api-resources` to determine if a resource is namespaced.

Explicit use of `--namespace <value>` overrides this behavior.

How `kubectl` handles ServiceAccount tokens

If:

- there is Kubernetes service account token file mounted at `/var/run/secrets/kubernetes.io/serviceaccount/token` , and
- the `KUBERNETES_SERVICE_HOST` environment variable is set, and
- the `KUBERNETES_SERVICE_PORT` environment variable is set, and
- you don't explicitly specify a namespace on the `kubectl` command line

then `kubectl` assumes it is running in your cluster. The `kubectl` tool looks up the namespace of that ServiceAccount (this is the same as the namespace of the Pod) and acts against that namespace. This is different from what happens outside of a cluster; when `kubectl` runs outside a cluster and you don't specify a namespace, the `kubectl` command acts

against the namespace set for the current context in your client configuration. To change the default namespace for your `kubectl` you can use the following command:

```
kubectl config set-context --current --namespace=<namespace-name>
```

Operations

The following table includes short descriptions and the general syntax for all of the `kubectl` operations:

Operation	Syntax	Description
alpha	<code>kubectl alpha SUBCOMMAND [flags]</code>	List the available commands that correspond to alpha features, which are not enabled in Kubernetes clusters by default.
annotate	<code>kubectl annotate (-f FILENAME TYPE NAME TYPE/NAME) KEY_1=VAL_1 ... KEY_N=VAL_N [--overwrite] [--all] [--resource-version=version] [flags]</code>	Add or update the annotations of one or more resources.
api-resources	<code>kubectl api-resources [flags]</code>	List the API resources that are available.
api-versions	<code>kubectl api-versions [flags]</code>	List the API versions that are available.
apply	<code>kubectl apply -f FILENAME [flags]</code>	Apply a configuration change to a resource from a file or stdin.
attach	<code>kubectl attach POD -c CONTAINER [-i] [-t] [flags]</code>	Attach to a running container either to view the output stream or interact with the container (stdin).
auth	<code>kubectl auth [flags] [options]</code>	Inspect authorization.
autoscale	<code>kubectl autoscale (-f FILENAME TYPE NAME TYPE/NAME) [--min=MINPODS] --max=MAXPODS [--cpu-percent=CPU] [flags]</code>	Automatically scale the set of pods that are managed by a replication controller.
certificate	<code>kubectl certificate SUBCOMMAND [options]</code>	Modify certificate resources.
cluster-info	<code>kubectl cluster-info [flags]</code>	Display endpoint information about the master and services in the cluster.

Operation	Syntax	Description
completion	<code>kubectl completion SHELL [options]</code>	Output shell completion code for the specified shell (bash or zsh).
config	<code>kubectl config SUBCOMMAND [flags]</code>	Modifies kubeconfig files. See the individual subcommands for details.
convert	<code>kubectl convert -f FILENAME [options]</code>	Convert config files between different API versions. Both YAML and JSON formats are accepted. Note - requires <code>kubectl-convert</code> plugin to be installed.
cordon	<code>kubectl cordon NODE [options]</code>	Mark node as unschedulable.
cp	<code>kubectl cp <file-spec-src> <file-spec-dest> [options]</code>	Copy files and directories to and from containers.
create	<code>kubectl create -f FILENAME [flags]</code>	Create one or more resources from a file or stdin.
delete	<code>kubectl delete (-f FILENAME TYPE [NAME /NAME -l label --all]) [flags]</code>	Delete resources either from a file, stdin, or specifying label selectors, names, resource selectors, or resources.
describe	<code>kubectl describe (-f FILENAME TYPE [NAME_PREFIX /NAME -l label]) [flags]</code>	Display the detailed state of one or more resources.
diff	<code>kubectl diff -f FILENAME [flags]</code>	Diff file or stdin against live configuration.
drain	<code>kubectl drain NODE [options]</code>	Drain node in preparation for maintenance.
edit	<code>kubectl edit (-f FILENAME TYPE NAME TYPE/NAME) [flags]</code>	Edit and update the definition of one or more resources on the server by using the default editor.
events	<code>kubectl events</code>	List events
exec	<code>kubectl exec POD [-c CONTAINER] [-i] [-t] [flags] [-- COMMAND [args...]]</code>	Execute a command against a container in a pod.
explain	<code>kubectl explain TYPE [--recursive=false] [flags]</code>	Get documentation of various resources. For instance pods, nodes, services, etc.

Operation	Syntax	Description
expose	<code>kubectl expose (-f FILENAME TYPE NAME TYPE/NAME) [--port=port] [--protocol=TCP UDP] [--target-port=number-or-name] [--name=name] [--external-ip=external-ip-of-service] [--type=type] [flags]</code>	Expose a replication controller, service, or pod as a new Kubernetes service.
get	<code>kubectl get (-f FILENAME TYPE [NAME /NAME -l label]) [--watch] [--sort-by=FIELD] [[-o --output]=OUTPUT_FORMAT] [flags]</code>	List one or more resources.
kustomize	<code>kubectl kustomize <dir> [flags] [options]</code>	List a set of API resources generated from instructions in a kustomization.yaml file. The argument must be the path to the directory containing the file, or a git repository URL with a path suffix specifying same with respect to the repository root.
label	<code>kubectl label (-f FILENAME TYPE NAME TYPE/NAME) KEY_1=VAL_1 ... KEY_N=VAL_N [--overwrite] [--all] [--resource-version=version] [flags]</code>	Add or update the labels of one or more resources.
logs	<code>kubectl logs POD [-c CONTAINER] [--follow] [flags]</code>	Print the logs for a container in a pod.
options	<code>kubectl options</code>	List of global command-line options, which apply to all commands.
patch	<code>kubectl patch (-f FILENAME TYPE NAME TYPE/NAME) --patch PATCH [flags]</code>	Update one or more fields of a resource by using the strategic merge patch process.
plugin	<code>kubectl plugin [flags] [options]</code>	Provides utilities for interacting with plugins.
port-forward	<code>kubectl port-forward POD [LOCAL_PORT:]REMOTE_PORT [...] [LOCAL_PORT_N:]REMOTE_PORT_N] [flags]</code>	Forward one or more local ports to a pod.
proxy	<code>kubectl proxy [--port=PORT] [--www=static-dir] [--www-prefix=prefix] [--api-prefix=prefix] [flags]</code>	Run a proxy to the Kubernetes API server.

Operation	Syntax	Description
replace	<code>kubectl replace -f FILENAME</code>	Replace a resource from a file or stdin.
rollout	<code>kubectl rollout SUBCOMMAND [options]</code>	Manage the rollout of a resource. Valid resource types include: deployments, daemonsets and statefulsets.
run	<code>kubectl run NAME --image=image [--env="key=value"] [--port=port] [--dry-run=server client none] [--overrides=inline-json] [flags]</code>	Run a specified image on the cluster.
scale	<code>kubectl scale (-f FILENAME TYPE NAME TYPE/NAME) --replicas=COUNT [--resource-version=version] [--current-replicas=count] [flags]</code>	Update the size of the specified replication controller.
set	<code>kubectl set SUBCOMMAND [options]</code>	Configure application resources.
taint	<code>kubectl taint NODE NAME KEY_1=VAL_1:TAINTEFFECT_1 ... KEY_N=VAL_N:TAINTEFFECT_N [options]</code>	Update the taints on one or more nodes.
top	<code>kubectl top (POD NODE) [flags]</code> [options]	Display Resource (CPU/Memory/Storage) usage of pod or node.
uncordon	<code>kubectl uncordon NODE [options]</code>	Mark node as schedulable.
version	<code>kubectl version [--client] [flags]</code>	Display the Kubernetes version running on the client and server.
wait	<code>kubectl wait ([-f FILENAME] resource.group/resource.name resource.group [(-l label --all)]) [--for=delete --for-condition=available] [options]</code>	Experimental: Wait for a specific condition on one or many resources.

To learn more about command operations, see the [kubectl](#) reference documentation.

Resource types

The following table includes a list of all the supported resource types and their abbreviated aliases.

(This output can be retrieved from `kubectl api-resources`, and was accurate as of Kubernetes 1.25.0)

NAME	SHORTNAMES	APIVERSION

NAME	SHORTNAMES	APIVERSION
bindings		v1
componentstatuses	cs	v1
configmaps	cm	v1
endpoints	ep	v1
events	ev	v1
limitranges	limits	v1
namespaces	ns	v1
nodes	no	v1
persistentvolumeclaims	pvc	v1
persistentvolumes	pv	v1
pods	po	v1
podtemplates		v1
replicationcontrollers	rc	v1
resourcequotas	quota	v1
secrets		v1
serviceaccounts	sa	v1
services	svc	v1
mutatingwebhookconfigurations		admissionregistration.k8s.io/ v1
validatingwebhookconfigurations		admissionregistration.k8s.io/ v1
customresourcedefinitions	crd, crds	apiextensions.k8s.io/v1
apiservices		apiregistration.k8s.io/v1
controllerrevisions		apps/v1
daemonsets	ds	apps/v1
deployments	deploy	apps/v1
replicasets	rs	apps/v1
statefulsets	sts	apps/v1
tokenreviews		authentication.k8s.io/v1
localsubjectaccessreviews		authorization.k8s.io/v1

NAME	SHORTNAMES	APIVERSION
selfsubjectaccessreviews		authorization.k8s.io/v1
selfsubjectrulesreviews		authorization.k8s.io/v1
subjectaccessreviews		authorization.k8s.io/v1
horizontalpodautoscalers	hpa	autoscaling/v2
cronjobs	cj	batch/v1
jobs		batch/v1
certificatesigningrequests	csr	certificates.k8s.io/v1
leases		coordination.k8s.io/v1
endpointslices		discovery.k8s.io/v1
events	ev	events.k8s.io/v1
flowschemas		flowcontrol.apiserver.k8s.io/v1beta2
prioritylevelconfigurations		flowcontrol.apiserver.k8s.io/v1beta2
ingressclasses		networking.k8s.io/v1
ingresses	ing	networking.k8s.io/v1
networkpolicies	netpol	networking.k8s.io/v1
runtimeclasses		node.k8s.io/v1
poddisruptionbudgets	pdb	policy/v1
podsecuritypolicies	psp	policy/v1beta1
clusterrolebindings		rbac.authorization.k8s.io/v1
clusterroles		rbac.authorization.k8s.io/v1
rolebindings		rbac.authorization.k8s.io/v1
roles		rbac.authorization.k8s.io/v1
priorityclasses	pc	scheduling.k8s.io/v1
csidrivers		storage.k8s.io/v1
csinodes		storage.k8s.io/v1
csistoragecapacities		storage.k8s.io/v1
storageclasses	sc	storage.k8s.io/v1
volumeattachments		storage.k8s.io/v1

Output options

Use the following sections for information about how you can format or sort the output of certain commands. For details about which commands support the various output options, see the [kubectl](#) reference documentation.

Formatting output

The default output format for all `kubectl` commands is the human readable plain-text format. To output details to your terminal window in a specific format, you can add either the `-o` or `--output` flags to a supported `kubectl` command.

Syntax

```
kubectl [command] [TYPE] [NAME] -o <output_format>
```

Depending on the `kubectl` operation, the following output formats are supported:

Output format	Description
<code>-o custom-columns=<spec></code>	Print a table using a comma separated list of custom columns .
<code>-o custom-columns-file=<filename></code>	Print a table using the custom columns template in the <code><filename></code> file.
<code>-o json</code>	Output a JSON formatted API object.
<code>-o jsonpath=<template></code>	Print the fields defined in a jsonpath expression.
<code>-o jsonpath-file=<filename></code>	Print the fields defined by the jsonpath expression in the <code><filename></code> file.
<code>-o name</code>	Print only the resource name and nothing else.
<code>-o wide</code>	Output in the plain-text format with any additional information. For pods, the node name is included.
<code>-o yaml</code>	Output a YAML formatted API object.

Example

In this example, the following command outputs the details for a single pod as a YAML formatted object:

```
kubectl get pod web-pod-13je7 -o yaml
```

Remember: See the [kubectl](#) reference documentation for details about which output format is supported by each command.

Custom columns

To define custom columns and output only the details that you want into a table, you can use the `custom-columns` option. You can choose to define the custom columns inline or use a template file: `-o custom-columns=<spec>` or `-o custom-columns-file=<filename>`.

Examples

Inline:

```
kubectl get pods <pod-name> -o custom-columns=NAME:.metadata.name,RSR
```

Template file:

```
kubectl get pods <pod-name> -o custom-columns-file=template.txt
```

where the `template.txt` file contains:

```
NAME      RSRC
metadata.name metadata.resourceVersion
```

The result of running either command is similar to:

```
NAME      RSRC
submit-queue 610995
```

Server-side columns

`kubectl` supports receiving specific column information from the server about objects. This means that for any given resource, the server will return columns and rows relevant to that resource, for the client to print. This allows for consistent human-readable output across clients used against the same cluster, by having the server encapsulate the details of printing.

This feature is enabled by default. To disable it, add the `--server-print=false` flag to the `kubectl get` command.

Examples

To print information about the status of a pod, use a command like the following:

```
kubectl get pods <pod-name> --server-print=false
```

The output is similar to:

```
NAME      AGE
pod-name  1m
```

Sorting list objects

To output objects to a sorted list in your terminal window, you can add

the `--sort-by` flag to a supported `kubectl` command. Sort your objects by specifying any numeric or string field with the `--sort-by` flag. To specify a field, use a [jsonpath](#) expression.

Syntax

```
kubectl [command] [TYPE] [NAME] --sort-by=<jsonpath_exp>
```

Example

To print a list of pods sorted by name, you run:

```
kubectl get pods --sort-by=.metadata.name
```

Examples: Common operations

Use the following set of examples to help you familiarize yourself with running the commonly used `kubectl` operations:

`kubectl apply` - Apply or Update a resource from a file or stdin.

```
# Create a service using the definition in example-service.yaml.
kubectl apply -f example-service.yaml

# Create a replication controller using the definition in example-con
kubectl apply -f example-controller.yaml

# Create the objects that are defined in any .yaml, .yml, or .json fi
kubectl apply -f <directory>
```

`kubectl get` - List one or more resources.

```
# List all pods in plain-text output format.
kubectl get pods

# List all pods in plain-text output format and include additional in
kubectl get pods -o wide

# List the replication controller with the specified name in plain-te
kubectl get replicationcontroller <rc-name>

# List all replication controllers and services together in plain-tex
kubectl get rc,services

# List all daemon sets in plain-text output format.
kubectl get ds

# List all pods running on node server01
kubectl get pods --field-selector=spec.nodeName=server01
```

`kubectl describe` - Display detailed state of one or more resources, including the uninitialized ones by default.

```
# Display the details of the node with name <node-name>.  
kubectl describe nodes <node-name>  
  
# Display the details of the pod with name <pod-name>.  
kubectl describe pods/<pod-name>  
  
# Display the details of all the pods that are managed by the replica  
# Remember: Any pods that are created by the replication controller g  
kubectl describe pods <rc-name>  
  
# Describe all pods  
kubectl describe pods
```

Note:

The `kubectl get` command is usually used for retrieving one or more resources of the same resource type. It features a rich set of flags that allows you to customize the output format using the `-o` or `--output` flag, for example. You can specify the `-w` or `--watch` flag to start watching updates to a particular object. The `kubectl describe` command is more focused on describing the many related aspects of a specified resource. It may invoke several API calls to the API server to build a view for the user. For example, the `kubectl describe node` command retrieves not only the information about the node, but also a summary of the pods running on it, the events generated for the node etc.

`kubectl delete` - Delete resources either from a file, stdin, or specifying label selectors, names, resource selectors, or resources.

```
# Delete a pod using the type and name specified in the pod.yaml file  
kubectl delete -f pod.yaml  
  
# Delete all the pods and services that have the label '<label-key>=<label-value>'  
kubectl delete pods,services -l <label-key>=<label-value>  
  
# Delete all pods, including uninitialized ones.  
kubectl delete pods --all
```

`kubectl exec` - Execute a command against a container in a pod.

```
# Get output from running 'date' from pod <pod-name>. By default, out  
kubectl exec <pod-name> -- date  
  
# Get output from running 'date' in container <container-name> of pod  
kubectl exec <pod-name> -c <container-name> -- date  
  
# Get an interactive TTY and run /bin/bash from pod <pod-name>. By de  
kubectl exec -ti <pod-name> -- /bin/bash
```

`kubectl logs` - Print the logs for a container in a pod.

```
# Return a snapshot of the logs from pod <pod-name>.  
kubectl logs <pod-name>  
  
# Start streaming the logs from pod <pod-name>. This is similar to the  
kubectl logs -f <pod-name>
```

`kubectl diff` - View a diff of the proposed updates to a cluster.

```
# Diff resources included in "pod.json".
kubectl diff -f pod.json

# Diff file read from stdin.
cat service.yaml | kubectl diff -f -
```

Examples: Creating and using plugins

Use the following set of examples to help you familiarize yourself with writing and using `kubectl` plugins:

```
# create a simple plugin in any language and name the resulting executable
# so that it begins with the prefix "kubectl-"
cat ./kubectl-hello
```

```
#!/bin/sh

# this plugin prints the words "hello world"
echo "hello world"
```

With a plugin written, let's make it executable:

```
chmod a+x ./kubectl-hello

# and move it to a location in our PATH
sudo mv ./kubectl-hello /usr/local/bin
sudo chown root:root /usr/local/bin

# You have now created and "installed" a kubectl plugin.
# You can begin using this plugin by invoking it from kubectl as if it were a command
kubectl hello
```

```
hello world
```

```
# You can "uninstall" a plugin, by removing it from the folder in your $PATH where you placed it
sudo rm /usr/local/bin/kubectl-hello
```

In order to view all of the plugins that are available to `kubectl`, use the `kubectl plugin list` subcommand:

```
kubectl plugin list
```

The output is similar to:

```
The following kubectl-compatible plugins are available:
```

```
/usr/local/bin/kubectl-hello
/usr/local/bin/kubectl-foo
/usr/local/bin/kubectl-bar
```

`kubectl plugin list` also warns you about plugins that are not executable, or that are shadowed by other plugins; for example:

```
sudo chmod -x /usr/local/bin/kubectl-foo # remove execute permission
kubectl plugin list
```

```
The following kubectl-compatible plugins are available:
```

```
/usr/local/bin/kubectl-hello
/usr/local/bin/kubectl-foo
  - warning: /usr/local/bin/kubectl-foo identified as a plugin, but is
/usr/local/bin/kubectl-bar

error: one plugin warning was found
```

You can think of plugins as a means to build more complex functionality on top of the existing kubectl commands:

```
cat ./kubectl-whoami
```

The next few examples assume that you already made `kubectl-whoami` have the following contents:

```
#!/bin/bash

# this plugin makes use of the `kubectl config` command in order to obtain
# information about the current user, based on the currently selected
kubectl config view --template='{{ range .contexts }}{{ if eq .name "
```

Running the above command gives you an output containing the user for the current context in your KUBECONFIG file:

```
# make the file executable
sudo chmod +x ./kubectl-whoami

# and move it into your PATH
sudo mv ./kubectl-whoami /usr/local/bin

kubectl whoami
Current user: plugins-user
```

What's next

- Read the `kubectl` reference documentation:
 - the `kubectl` [command reference](#)
 - the [command line arguments](#) reference
- Learn about [kubectl usage conventions](#)

- Read about [JSONPath support](#) in kubectl
- Read about how to [extend kubectl with plugins](#)
 - To find out more about plugins, take a look at the [example CLI plugin](#).

11.1 - Introduction to kubectl

kubectl is the Kubernetes cli version of a swiss army knife, and can do many things.

While this Book is focused on using kubectl to declaratively manage applications in Kubernetes, it also covers other kubectl functions.

Command Families

Most kubectl commands typically fall into one of a few categories:

Type	Used For	Description
Declarative Resource Management	Deployment and operations (e.g. GitOps)	Declaratively manage Kubernetes workloads using resource configuration
Imperative Resource Management	Development Only	Run commands to manage Kubernetes workloads using Command Line arguments and flags
Printing Workload State	Debugging	Print information about workloads
Interacting with Containers	Debugging	Exec, attach, cp, logs
Cluster Management	Cluster operations	Drain and cordon Nodes

Declarative Application Management

The preferred approach for managing resources is through declarative files called resource configuration used with the kubectl *Apply* command. This command reads a local (or remote) file structure and modifies cluster state to reflect the declared intent.

Apply

Apply is the preferred mechanism for managing resources in a Kubernetes cluster.

Printing State about Workloads

Users will need to view workload state.

- Printing summarize state and information about resources
- Printing complete state and information about resources
- Printing specific fields from resources
- Query resources matching labels

Debugging Workloads

kubectl supports debugging by providing commands for:

- Printing Container logs
- Printing cluster events
- Exec or attaching to a Container
- Copying files from Containers in the cluster to a user's filesystem

Cluster Management

On occasion, users may need to perform operations to the Nodes of cluster. kubectl supports commands to drain workloads from a Node so that it can be decommissioned or debugged.

Porcelain

Users may find using resource configuration overly verbose for *development* and prefer to work with the cluster *imperatively* with a shell-like workflow. kubectl offers porcelain commands for generating and modifying resources.

- Generating + creating resources such as Deployments, StatefulSets, Services, ConfigMaps, etc.
- Setting fields on resources
- Editing (live) resources in a text editor

Porcelain for Dev Only

Porcelain commands are time saving for experimenting with workloads in a dev cluster, but shouldn't be used for production.

11.2 - kubectl Quick Reference

This page contains a list of commonly used `kubectl` commands and flags.

Note:

These instructions are for Kubernetes v1.31. To check the version, use the `kubectl version` command.

Kubectl autocomplete

BASH

```
source <(kubectl completion bash) # set up autocomplete in bash into
echo "source <(kubectl completion bash)" >> ~/.bashrc # add autocompl
```

You can also use a shorthand alias for `kubectl` that also works with completion:

```
alias k=kubectl
complete -o default -F __start_kubectl k
```

ZSH

```
source <(kubectl completion zsh) # set up autocomplete in zsh into t
echo '[[ $commands[kubectl] ]] && source <(kubectl completion zsh)' >
```

FISH

Note:

Requires kubectl version 1.23 or above.

```
echo 'kubectl completion fish | source' > ~/.config/fish/completions/
```

A note on `--all-namespaces`

Appending `--all-namespaces` happens frequently enough that you should be aware of the shorthand for `--all-namespaces`:

```
kubectl -A
```

Kubectl context and configuration

Set which Kubernetes cluster `kubectl` communicates with and modifies configuration information. See [Authenticating Across Clusters with kubeconfig](#) documentation for detailed config file information.

```
kubectl config view # Show Merged kubeconfig settings.

# use multiple kubeconfig files at the same time and view merged config
KUBECONFIG=~/ kube/config:~/ kube/kubconfig2

kubectl config view

# Show merged kubeconfig settings and raw certificate data and expose
kubectl config view --raw

# get the password for the e2e user
kubectl config view -o jsonpath='{.users[?(@.name == "e2e")].user.pas

# get the certificate for the e2e user
kubectl config view --raw -o jsonpath='{.users[?(@.name == "e2e")].use

kubectl config view -o jsonpath='{.users[].name}'      # display the fi
kubectl config view -o jsonpath='{.users[*].name}'     # get a list of
kubectl config get-contexts                          # display list o
kubectl config get-contexts -o name                  # get all contex
kubectl config current-context                      # display the cu
kubectl config use-context my-cluster-name          # set the default

kubectl config set-cluster my-cluster-name          # set a cluster

# configure the URL to a proxy server to use for requests made by this
kubectl config set-cluster my-cluster-name --proxy-url=my-proxy-url

# add a new user to your kubeconfig that supports basic auth
kubectl config set-credentials kubeuser/foo.kubernetes.com --username

# permanently save the namespace for all subsequent kubectl commands
kubectl config set-context --current --namespace=ggckad-s2

# set a context utilizing a specific username and namespace.
kubectl config set-context gce --user=cluster-admin --namespace=foo \
  && kubectl config use-context gce

kubectl config unset users.foo                      # delete user fo

# short alias to set/show context/namespace (only works for bash and
alias kx='f() { [ "$1" ] && kubectl config use-context $1 || kubectl
alias kn='f() { [ "$1" ] && kubectl config set-context --current --na
```

Kubectl apply

`apply` manages applications through files defining Kubernetes resources. It creates and updates resources in a cluster through running `kubectl apply`. This is the recommended way of managing Kubernetes applications on production. See [Kubectl Book](#).

Creating objects

Kubernetes manifests can be defined in YAML or JSON. The file extension `.yaml`, `.yml`, and `.json` can be used.

```
kubectl apply -f ./my-manifest.yaml          # create resource
kubectl apply -f ./my1.yaml -f ./my2.yaml      # create from mul
kubectl apply -f ./dir                         # create resource
kubectl apply -f https://example.com/manifest.yaml # create resource
kubectl create deployment nginx --image=nginx      # start a single

# create a Job which prints "Hello World"
kubectl create job hello --image=busybox:1.28 -- echo "Hello World"

# create a CronJob that prints "Hello World" every minute
kubectl create cronjob hello --image=busybox:1.28 --schedule="*/1 * * * *"

kubectl explain pods                         # get the documentation

# Create multiple YAML objects from stdin
kubectl apply -f - <<EOF
apiVersion: v1
kind: Pod
metadata:
  name: busybox-sleep
spec:
  containers:
  - name: busybox
    image: busybox:1.28
    args:
    - sleep
    - "1000000"
---
apiVersion: v1
kind: Pod
metadata:
  name: busybox-sleep-less
spec:
  containers:
  - name: busybox
    image: busybox:1.28
    args:
    - sleep
    - "1000"
EOF

# Create a secret with several keys
kubectl apply -f - <<EOF
apiVersion: v1
kind: Secret
metadata:
  name: mysecret
type: Opaque
data:
  password: $(echo -n "s33msi4" | base64 -w0)
  username: $(echo -n "jane" | base64 -w0)
EOF
```

Viewing and finding resources

```
# Get commands with basic output
kubectl get services                                # List all services in
kubectl get pods --all-namespaces                   # List all pods in all
kubectl get pods -o wide                            # List all pods in the
kubectl get deployment my-dep                      # List a particular dep
kubectl get pods                                    # List all pods in the
kubectl get pod my-pod -o yaml                      # Get a pod's YAML

# Describe commands with verbose output
kubectl describe nodes my-node
kubectl describe pods my-pod

# List Services Sorted by Name
kubectl get services --sort-by=.metadata.name

# List pods Sorted by Restart Count
kubectl get pods --sort-by=.status.containerStatuses[0].restartCount

# List PersistentVolumes sorted by capacity
kubectl get pv --sort-by=.spec.capacity.storage

# Get the version label of all pods with label app=cassandra
kubectl get pods --selector=app=cassandra -o \
  jsonpath='{.items[*].metadata.labels.version}'

# Retrieve the value of a key with dots, e.g. 'ca.crt'
kubectl get configmap myconfig \
  -o jsonpath='{.data.ca\.crt}'

# Retrieve a base64 encoded value with dashes instead of underscores.
kubectl get secret my-secret --template='{{index .data "key-name-with-dash" | base64decode}}'

# Get all worker nodes (use a selector to exclude results that have a
# named 'node-role.kubernetes.io/control-plane')
kubectl get node --selector='!node-role.kubernetes.io/control-plane'

# Get all running pods in the namespace
kubectl get pods --field-selector=status.phase=Running

# Get ExternalIPs of all nodes
kubectl get nodes -o jsonpath='{.items[*].status.addresses[?(@.type==

# List Names of Pods that belong to Particular RC
# "jq" command useful for transformations that are too complex for jsonpath
sel=$(kubectl get rc my-rc --output=json | jq -j '.spec.selector | \
echo $(kubectl get pods --selector=$sel --output=jsonpath={.items..me

# Show Labels for all pods (or any other Kubernetes object that supports labels)
kubectl get pods --show-labels

# Check which nodes are ready
JSONPATH='{range .items[*]}{@.metadata.name}:{range @.status.conditions[*]}{.type==\"Ready\" && .status==\"True\"}{{"\n"}}
```

```

# List all warning events
kubectl events --types=Warning

# Compares the current state of the cluster against the state that the
# manifest defines. Returns a list of differences.
kubectl diff -f ./my-manifest.yaml

# Produce a period-delimited tree of all keys returned for nodes
# Helpful when locating a key within a complex nested JSON structure
kubectl get nodes -o json | jq -c 'paths|join(".")

# Produce a period-delimited tree of all keys returned for pods, etc
kubectl get pods -o json | jq -c 'paths|join(".")

# Produce ENV for all pods, assuming you have a default container for
# each pod. Returns a list of environment variables for each pod.
# Helpful when running any supported command across all pods, not just
# the first one.
for pod in $(kubectl get po --output=jsonpath={.items..metadata.name})
do
  echo "In $pod"
  kubectl get pod $pod -o yaml | jq -c ".spec.containers[0].env"
done

# Get a deployment's status subresource
kubectl get deployment nginx-deployment --subresource=status

```

Updating resources

```

kubectl set image deployment/frontend www=image:v2                                # Replace
kubectl rollout history deployment/frontend                                         # Check
kubectl rollout undo deployment/frontend                                         # Rollback
kubectl rollout undo deployment/frontend --to-revision=2                         # Rollback to previous revision
kubectl rollout status -w deployment/frontend                                     # Watch
kubectl rollout restart deployment/frontend                                       # Rollback to previous revision

cat pod.json | kubectl replace -f -                                              # Replace

# Force replace, delete and then re-create the resource. Will cause a
# short outage.
kubectl replace --force -f ./pod.json

# Create a service for a replicated nginx, which serves on port 80 and
# exposes port 8000
kubectl expose rc nginx --port=80 --target-port=8000

# Update a single-container pod's image version (tag) to v4
kubectl get pod mypod -o yaml | sed 's/^(image: myimage):.*$/\1:v4/'

kubectl label pods my-pod new-label=awesome                                         # Add
kubectl label pods my-pod new-label-                                            # Remove
kubectl label pods my-pod new-label=new-value --overwrite                         # Overwrite
kubectl annotate pods my-pod icon-url=http://goo.gl/XXBTWq                      # Add
kubectl annotate pods my-pod icon-url-                                         # Remove
kubectl autoscale deployment foo --min=2 --max=10                                # Auto

```

Patching resources

```
# Partially update a node
kubectl patch node k8s-node-1 -p '{"spec":{"unschedulable":true}}'

# Update a container's image; spec.containers[*].name is required because
# the container name is not specified in the patch
kubectl patch pod valid-pod -p '{"spec":{"containers":[{"name":"kuber
  # Update a container's image using a json patch with positional array
  kubectl patch pod valid-pod --type='json' -p='[{"op": "replace", "path": "/
  # Disable a deployment livenessProbe using a json patch with position
  kubectl patch deployment valid-deployment --type json -p='[{"op": "replac
  # Add a new element to a positional array
  kubectl patch sa default --type='json' -p='[{"op": "add", "path": "/spec/c
  # Update a deployment's replica count by patching its scale subresource
  kubectl patch deployment nginx-deployment --subresource='scale' --type='j
```

Editing resources

Edit any API resource in your preferred editor.

```
kubectl edit svc/docker-registry          # Edit the service
KUBE_EDITOR="nano" kubectl edit svc/docker-registry # Use an alternative editor
```

Scaling resources

```
kubectl scale --replicas=3 rs/foo          # Scale a ReplicSet
kubectl scale --replicas=3 -f foo.yaml      # Scale from a YAML file
kubectl scale --current-replicas=2 --replicas=3 deployment/mysql # Increase
kubectl scale --replicas=5 rc/foo rc/bar rc/baz # Scale multiple resources
```

Deleting resources

```
kubectl delete -f ./pod.json               # Delete a file
kubectl delete pod unwanted --now           # Delete a pod
kubectl delete pod,service baz foo          # Delete a pod and its service
kubectl delete pods,services -l name=myLabel # Delete pods and services
kubectl -n my-ns delete pod,svc --all       # Delete all pods and services in a namespace
# Delete all pods matching the awk pattern1 or pattern2
kubectl get pods -n mynamespace --no-headers=true | awk '/pattern1|p
```

Interacting with running Pods

```

kubectl logs my-pod                                # dump pod Logs
kubectl logs -l name=myLabel                         # dump pod Logs,
kubectl logs my-pod --previous                      # dump pod Logs (
kubectl logs my-pod -c my-container                 # dump pod container
kubectl logs -l name=myLabel -c my-container        # dump pod container
kubectl logs my-pod -c my-container --previous      # dump pod container
kubectl logs -f my-pod                                # stream pod Logs
kubectl logs -f my-pod -c my-container                # stream pod container
kubectl logs -f -l name=myLabel --all-containers    # stream all pods
kubectl run -i --tty busybox --image=busybox:1.28 -- sh  # Run pod as
kubectl run nginx --image=nginx -n mynamespace       # Start a single
kubectl run nginx --image=nginx --dry-run=client -o yaml > pod.yaml
                                                # Generate spec file
kubectl attach my-pod -i                                # Attach to Running
kubectl port-forward my-pod 5000:6000                  # Listen on port
kubectl exec my-pod -- ls /                            # Run command in container
kubectl exec --stdin --tty my-pod -- /bin/sh          # Interactive shell
kubectl exec my-pod -c my-container -- ls /            # Run command in container
kubectl debug my-pod -it --image=busybox:1.28          # Create an interactive shell
kubectl debug node/my-node -it --image=busybox:1.28    # Create an interactive shell
kubectl top pod                                         # Show metrics for pods
kubectl top pod POD_NAME --containers                # Show metrics for containers
kubectl top pod POD_NAME --sort-by=cpu                # Show metrics for containers

```

Copying files and directories to and from containers

```

kubectl cp /tmp/foo_dir my-pod:/tmp/bar_dir          # Copy /tmp/foo_dir to /tmp/bar_dir
kubectl cp /tmp/foo my-pod:/tmp/bar -c my-container  # Copy /tmp/foo to /tmp/bar in my-container
kubectl cp /tmp/foo my-namespace/my-pod:/tmp/bar      # Copy /tmp/foo to /tmp/bar in my-namespace/my-pod
kubectl cp my-namespace/my-pod:/tmp/foo /tmp/bar       # Copy /tmp/foo from my-namespace/my-pod to /tmp/bar

```

Note:

`kubectl cp` requires that the 'tar' binary is present in your container image. If 'tar' is not present, `kubectl cp` will fail. For advanced use cases, such as symlinks, wildcard expansion or file mode preservation consider using `kubectl exec`.

```

tar cf - /tmp/foo | kubectl exec -i -n my-namespace my-pod -- tar xf -
kubectl exec -n my-namespace my-pod -- tar cf - /tmp/foo | tar xf -

```

Interacting with Deployments and Services

```

kubectl logs deploy/my-deployment                      # dump Pod Logs
kubectl logs deploy/my-deployment -c my-container      # dump Pod Logs
kubectl port-forward svc/my-service 5000                # Listen on port
kubectl port-forward svc/my-service 5000:my-service-port # Listen on port
kubectl port-forward deploy/my-deployment 5000:6000      # Listen on port
kubectl exec deploy/my-deployment -- ls                 # run command in container

```

Interacting with Nodes and cluster

```
kubectl cordon my-node
kubectl drain my-node
kubectl uncordon my-node
kubectl top node
kubectl top node my-node
kubectl cluster-info
kubectl cluster-info dump
kubectl cluster-info dump --output-directory=/path/to/cluster-state

# View existing taints on which exist on current nodes.
kubectl get nodes -o='custom-columns=NodeName:.metadata.name,TaintKey

# If a taint with that key and effect already exists, its value is re
kubectl taint nodes foo dedicated=special-user:NoSchedule
```

Resource types

List all supported resource types along with their shortnames, [API group](#), whether they are [namespaced](#), and [kind](#):

```
kubectl api-resources
```

Other operations for exploring API resources:

```
kubectl api-resources --namespaced=true      # All namespaced resources
kubectl api-resources --namespaced=false     # All non-namespaced res
kubectl api-resources -o name                # All resources with sim
kubectl api-resources -o wide                # All resources with exp
kubectl api-resources --verbs=list,get       # All resources that sup
kubectl api-resources --api-group=extensions # All resources in the '
```

Formatting output

To output details to your terminal window in a specific format, add the `-o` (or `--output`) flag to a supported `kubectl` command.

Output format	Description
<code>-o=custom-columns=<spec></code>	Print a table using a comma separated list of custom columns
<code>-o=custom-columns-file=<filename></code>	Print a table using the custom columns template in the <code><filename></code> file
<code>-o=go-template=<template></code>	Print the fields defined in a golang template
<code>-o=go-template-file=<filename></code>	Print the fields defined by the golang template in the <code><filename></code> file
<code>-o=json</code>	Output a JSON formatted API object
<code>-o=jsonpath=<template></code>	Print the fields defined in a jsonpath expression

Output format	Description
<code>-o=jsonpath-file=<filename></code>	Print the fields defined by the jsonpath expression in the <code><filename></code> file
<code>-o=name</code>	Print only the resource name and nothing else
<code>-o=wide</code>	Output in the plain-text format with any additional information, and for pods, the node name is included
<code>-o=yaml</code>	Output a YAML formatted API object

Examples using `-o=custom-columns` :

```
# ALL images running in a cluster
kubectl get pods -A -o=custom-columns='DATA:spec.containers[*].image'

# ALL images running in namespace: default, grouped by Pod
kubectl get pods --namespace default --output=custom-columns="NAME:.m

# ALL images excluding "registry.k8s.io/coredns:1.6.2"
kubectl get pods -A -o=custom-columns='DATA:spec.containers[?(@.image

# ALL fields under metadata regardless of name
kubectl get pods -A -o=custom-columns='DATA:metadata.*'
```

More examples in the [kubectl reference documentation](#).

Kubectl output verbosity and debugging

Kubectl verbosity is controlled with the `-v` or `--v` flags followed by an integer representing the log level. General Kubernetes logging conventions and the associated log levels are described [here](#).

Verbosity	Description
<code>--v=0</code>	Generally useful for this to <i>always</i> be visible to a cluster operator.
<code>--v=1</code>	A reasonable default log level if you don't want verbosity.
<code>--v=2</code>	Useful steady state information about the service and important log messages that may correlate to significant changes in the system. This is the recommended default log level for most systems.
<code>--v=3</code>	Extended information about changes.
<code>--v=4</code>	Debug level verbosity.
<code>--v=5</code>	Trace level verbosity.
<code>--v=6</code>	Display requested resources.
<code>--v=7</code>	Display HTTP request headers.
<code>--v=8</code>	Display HTTP request contents.
<code>--v=9</code>	Display HTTP request contents without truncation of contents.

What's next

- Read the [kubectl overview](#) and learn about [JsonPath](#).
- See [kubectl](#) options.
- Also read [kubectl Usage Conventions](#) to understand how to use `kubectl` in reusable scripts.
- See more community [kubectl cheatsheets](#).

11.3 - kubectl reference

11.3.1 - kubectl

Synopsis

kubectl controls the Kubernetes cluster manager.

Find more information at: <https://kubernetes.io/docs/reference/kubectl/>

```
kubectl [flags]
```

Options

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

-h, --help

help for kubectl

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cAdvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl annotate](#) - Update the annotations on a resource
- [kubectl api-resources](#) - Print the supported API resources on the server
- [kubectl api-versions](#) - Print the supported API versions on the server, in the form of "group/version"
- [kubectl apply](#) - Apply a configuration to a resource by file name or stdin
- [kubectl attach](#) - Attach to a running container
- [kubectl auth](#) - Inspect authorization
- [kubectl autoscale](#) - Auto-scale a deployment, replica set, stateful set, or replication controller
- [kubectl certificate](#) - Modify certificate resources
- [kubectl cluster-info](#) - Display cluster information
- [kubectl completion](#) - Output shell completion code for the specified shell (bash, zsh, fish, or powershell)
- [kubectl config](#) - Modify kubeconfig files
- [kubectl cordon](#) - Mark node as unschedulable
- [kubectl cp](#) - Copy files and directories to and from containers
- [kubectl create](#) - Create a resource from a file or from stdin
- [kubectl debug](#) - Create debugging sessions for troubleshooting workloads and nodes
- [kubectl delete](#) - Delete resources by file names, stdin, resources and names, or by resources and label selector
- [kubectl describe](#) - Show details of a specific resource or group of resources
- [kubectl diff](#) - Diff the live version against a would-be applied version
- [kubectl drain](#) - Drain node in preparation for maintenance
- [kubectl edit](#) - Edit a resource on the server
- [kubectl events](#) - List events
- [kubectl exec](#) - Execute a command in a container
- [kubectl explain](#) - Get documentation for a resource
- [kubectl expose](#) - Take a replication controller, service, deployment or pod and expose it as a new Kubernetes service
- [kubectl get](#) - Display one or many resources
- [kubectl kustomize](#) - Build a kustomization target from a directory or URL
- [kubectl label](#) - Update the labels on a resource
- [kubectl logs](#) - Print the logs for a container in a pod
- [kubectl options](#) - Print the list of flags inherited by all commands
- [kubectl patch](#) - Update fields of a resource
- [kubectl plugin](#) - Provides utilities for interacting with plugins
- [kubectl port-forward](#) - Forward one or more local ports to a pod
- [kubectl proxy](#) - Run a proxy to the Kubernetes API server

- [`kubectl replace`](#) - Replace a resource by file name or stdin
- [`kubectl rollout`](#) - Manage the rollout of a resource
- [`kubectl run`](#) - Run a particular image on the cluster
- [`kubectl scale`](#) - Set a new size for a deployment, replica set, or replication controller
- [`kubectl set`](#) - Set specific features on objects
- [`kubectl taint`](#) - Update the taints on one or more nodes
- [`kubectl top`](#) - Display resource (CPU/memory) usage
- [`kubectl uncordon`](#) - Mark node as schedulable
- [`kubectl version`](#) - Print the client and server version information
- [`kubectl wait`](#) - Experimental: Wait for a specific condition on one or many resources

11.3.2 - kubectl annotate

Synopsis

Update the annotations on one or more resources.

All Kubernetes objects support the ability to store additional data with the object as annotations. Annotations are key/value pairs that can be larger than labels and include arbitrary string values such as structured JSON. Tools and system extensions may use annotations to store their own data.

Attempting to set an annotation that already exists will fail unless `--overwrite` is set. If `--resource-version` is specified and does not match the current resource version on the server the command will fail.

Use "kubectl api-resources" for a complete list of supported resources.

```
kubectl annotate [--overwrite] (-f FILENAME | TYPE NAME) KEY_1=VAL_1
```

Examples

```
# Update pod 'foo' with the annotation 'description' and the value
# If the same annotation is set multiple times, only the last value
kubectl annotate pods foo description='my frontend'

# Update a pod identified by type and name in "pod.json"
kubectl annotate -f pod.json description='my frontend'

# Update pod 'foo' with the annotation 'description' and the value
kubectl annotate --overwrite pods foo description='my frontend running
# Update all pods in the namespace
kubectl annotate pods --all description='my frontend running nginx'

# Update pod 'foo' only if the resource is unchanged from version 1
kubectl annotate pods foo description='my frontend running nginx' -
# Update pod 'foo' by removing an annotation named 'description' if
# Does not require the --overwrite flag
kubectl annotate pods foo description-
```

Options

--all

Select all resources, in the namespace of the specified resource types.

-A, --all-namespaces

If true, check the specified action in all namespaces.

--allow-missing-template-keys Default: true

If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to golang and jsonpath output formats.

--dry-run string[="unchanged"] Default: "none"

Must be "none", "server", or "client". If client strategy, only print the object that would be sent, without sending it. If server strategy, submit server-side request without persisting the resource.

--field-manager string Default: "kubectl-annotate"

Name of the manager used to track field ownership.

--field-selector string

Selector (field query) to filter on, supports '=', '==', and '!='.(e.g. --field-selector key1=value1,key2=value2). The server only supports a limited number of field queries per type.

-f, --filename strings

Filename, directory, or URL to files identifying the resource to update the annotation

-h, --help

help for annotate

-k, --kustomize string

Process the kustomization directory. This flag can't be used together with -f or -R.

--list

If true, display the annotations for a given resource.

--local

If true, annotation will NOT contact api-server but run locally.

-o, --output string

Output format. One of: (json, yaml, name, go-template, go-template-file, template, templatefile, jsonpath, jsonpath-as-json, jsonpath-file).

--overwrite

If true, allow annotations to be overwritten, otherwise reject annotation updates that overwrite existing annotations.

-R, --recursive

Process the directory used in -f, --filename recursively. Useful when you want to manage related manifests organized within the same directory.

--resource-version string

If non-empty, the annotation update will only succeed if this is the current resource-version for the object. Only valid when specifying a single resource.

-l, --selector string

Selector (label query) to filter on, supports '=', '==', and '!='.(e.g. -l key1=value1,key2=value2). Matching objects must satisfy all of the specified label constraints.

--show-managed-fields

If true, keep the managedFields when printing objects in JSON or YAML format.

--template string

Template string or path to template file to use when `-o=go-template`, `-o=go-template-file`. The template format is golang templates [<http://golang.org/pkg/text/template/#pkg-overview>].

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl](#) - kubectl controls the Kubernetes cluster manager

11.3.3 - kubectl api-resources

Synopsis

Print the supported API resources on the server.

```
kubectl api-resources [flags]
```

Examples

```
# Print the supported API resources
kubectl api-resources

# Print the supported API resources with more information
kubectl api-resources -o wide

# Print the supported API resources sorted by a column
kubectl api-resources --sort-by=name

# Print the supported namespaced resources
kubectl api-resources --namespaced=true

# Print the supported non-namespaced resources
kubectl api-resources --namespaced=false

# Print the supported API resources with a specific APIGroup
kubectl api-resources --api-group=rbac.authorization.k8s.io
```

Options

--api-group string

Limit to resources in the specified API group.

--cached

Use the cached list of resources if available.

--categories strings

Limit to resources that belong to the specified categories.

-h, --help

help for api-resources

--namespaced Default: true

If false, non-namespaced resources will be returned, otherwise returning namespaced resources by default.

--no-headers

When using the default or custom-column output format, don't print headers (default print headers).

-o, --output string

Output format. One of: (wide, name).

--sort-by string

If non-empty, sort list of resources using specified field. The field can be either 'name' or 'kind'.

--verbs strings

Limit to resources that support the specified verbs.

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl](#) - kubectl controls the Kubernetes cluster manager

11.3.4 - kubectl api-versions

Synopsis

Print the supported API versions on the server, in the form of "group/version".

```
kubectl api-versions
```

Examples

```
# Print the supported API versions
kubectl api-versions
```

Options

-h, --help

help for api-versions

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl](#) - kubectl controls the Kubernetes cluster manager

11.3.5 - kubectl apply

Synopsis

Apply a configuration to a resource by file name or stdin. The resource name must be specified. This resource will be created if it doesn't exist yet. To use 'apply', always create the resource initially with either 'apply' or 'create --save-config'.

JSON and YAML formats are accepted.

Alpha Disclaimer: the --prune functionality is not yet complete. Do not use unless you are aware of what the current state is. See <https://issues.k8s.io/34274>.

```
kubectl apply (-f FILENAME | -k DIRECTORY)
```

Examples

```
# Apply the configuration in pod.json to a pod
kubectl apply -f ./pod.json

# Apply resources from a directory containing kustomization.yaml -
kubectl apply -k dir/

# Apply the JSON passed into stdin to a pod
cat pod.json | kubectl apply -f -

# Apply the configuration from all files that end with '.json'
kubectl apply -f '*.json'

# Note: --prune is still in Alpha
# Apply the configuration in manifest.yaml that matches label app=n
kubectl apply --prune -f manifest.yaml -l app=nginx

# Apply the configuration in manifest.yaml and delete all the other
kubectl apply --prune -f manifest.yaml --all --prune-allowlist=core
```

Options

--all

Select all resources in the namespace of the specified resource types.

--allow-missing-template-keys Default: true

If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to golang and jsonpath output formats.

--cascade string[="background"] Default: "background"

Must be "background", "orphan", or "foreground". Selects the deletion cascading strategy for the dependents (e.g. Pods created by a ReplicationController). Defaults to background.

--dry-run string[="unchanged"] Default: "none"

Must be "none", "server", or "client". If client strategy, only print the object that would be sent, without sending it. If server strategy, submit server-side request without persisting the resource.

--field-manager string Default: "kubectl-client-side-apply"

Name of the manager used to track field ownership.

-f, --filename strings

The files that contain the configurations to apply.

--force

If true, immediately remove resources from API and bypass graceful deletion. Note that immediate deletion of some resources may result in inconsistency or data loss and requires confirmation.

--force-conflicts

If true, server-side apply will force the changes against conflicts.

--grace-period int Default: -1

Period of time in seconds given to the resource to terminate gracefully. Ignored if negative. Set to 1 for immediate shutdown. Can only be set to 0 when --force is true (force deletion).

-h, --help

help for apply

-k, --kustomize string

Process a kustomization directory. This flag can't be used together with -f or -R.

--openapi-patch Default: true

If true, use openapi to calculate diff when the openapi presents and the resource can be found in the openapi spec. Otherwise, fall back to use baked-in types.

-o, --output string

Output format. One of: (json, yaml, name, go-template, go-template-file, template, templatefile, jsonpath, jsonpath-as-json, jsonpath-file).

--overwrite Default: true

Automatically resolve conflicts between the modified and live configuration by using values from the modified configuration

--prune

Automatically delete resource objects, that do not appear in the configs and are created by either apply or create --save-config. Should be used with either -l or --all.

--prune-allowlist strings

Overwrite the default allowlist with <group/version/kind> for --prune

-R, --recursive

Process the directory used in -f, --filename recursively. Useful when you want to manage related manifests organized within the same directory.

-l, --selector string

Selector (label query) to filter on, supports '=', '==', and '!='.(e.g. -l key1=value1,key2=value2). Matching objects must satisfy all of the specified label constraints.

--server-side

If true, apply runs in the server instead of the client.

--show-managed-fields

If true, keep the managedFields when printing objects in JSON or YAML format.

--template string

Template string or path to template file to use when -o=go-template, -o=go-template-file. The template format is golang templates [<http://golang.org/pkg/text/template/#pkg-overview>].

--timeout duration

The length of time to wait before giving up on a delete, zero means determine a timeout from the size of the object

--validate string[="strict"] Default: "strict"

Must be one of: strict (or true), warn, ignore (or false). "true" or "strict" will use a schema to validate the input and fail the request if invalid. It will perform server side validation if ServerSideFieldValidation is enabled on the api-server, but will fall back to less reliable client-side validation if not. "warn" will warn about unknown or duplicate fields without blocking the request if server-side field validation is enabled on the API server, and behave as "ignore" otherwise. "false" or "ignore" will not perform any schema validation, silently dropping any unknown or duplicate fields.

--wait

If true, wait for resources to be gone before returning. This waits for finalizers.

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cAdvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl](#) - kubectl controls the Kubernetes cluster manager
- [kubectl apply edit-last-applied](#) - Edit latest last-applied-configuration annotations of a resource/object
- [kubectl apply set-last-applied](#) - Set the last-applied-configuration annotation on a live object to match the contents of a file
- [kubectl apply view-last-applied](#) - View the latest last-applied-configuration annotations of a resource/object

11.3.5.1 - kubectl apply edit-last-applied

Synopsis

Edit the latest last-applied-configuration annotations of resources from the default editor.

The edit-last-applied command allows you to directly edit any API resource you can retrieve via the command-line tools. It will open the editor defined by your KUBE_EDITOR, or EDITOR environment variables, or fall back to 'vi' for Linux or 'notepad' for Windows. You can edit multiple objects, although changes are applied one at a time. The command accepts file names as well as command-line arguments, although the files you point to must be previously saved versions of resources.

The default format is YAML. To edit in JSON, specify "-o json".

The flag --windows-line-endings can be used to force Windows line endings, otherwise the default for your operating system will be used.

In the event an error occurs while updating, a temporary file will be created on disk that contains your unapplied changes. The most common error when updating a resource is another editor changing the resource on the server. When this occurs, you will have to apply your changes to the newer version of the resource, or update your temporary saved copy to include the latest resource version.

```
kubectl apply edit-last-applied (RESOURCE/NAME | -f FILENAME)
```

Examples

```
# Edit the last-applied-configuration annotations by type/name in Y
kubectl apply edit-last-applied deployment/nginx

# Edit the last-applied-configuration annotations by file in JSON
kubectl apply edit-last-applied -f deploy.yaml -o json
```

Options

--allow-missing-template-keys Default: true

If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to golang and jsonpath output formats.

--field-manager string Default: "kubectl-client-side-apply"

Name of the manager used to track field ownership.

-f, --filename strings

Filename, directory, or URL to files to use to edit the resource

-h, --help

help for edit-last-applied

-k, --kustomize string

Process the kustomization directory. This flag can't be used together with -f or -R.

-o, --output string

Output format. One of: (json, yaml, name, go-template, go-template-file, template, templatefile, jsonpath, jsonpath-as-json, jsonpath-file).

-R, --recursive

Process the directory used in -f, --filename recursively. Useful when you want to manage related manifests organized within the same directory.

--show-managed-fields

If true, keep the managedFields when printing objects in JSON or YAML format.

--template string

Template string or path to template file to use when -o=go-template, -o=go-template-file. The template format is golang templates [<http://golang.org/pkg/text/template/#pkg-overview>].

--validate string[="strict"] Default: "strict"

Must be one of: strict (or true), warn, ignore (or false).

"true" or "strict" will use a schema to validate the input and fail the request if invalid.

It will perform server side validation if ServerSideFieldValidation is enabled on the api-server, but will fall back to less reliable client-side validation if not.

"warn" will warn about unknown or duplicate fields without blocking the request if server-side field validation is enabled on the API server, and behave as "ignore" otherwise.

"false" or "ignore" will not perform any schema validation, silently dropping any unknown or duplicate fields.

--windows-line-endings

Defaults to the line ending native to your platform.

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl apply](#) - Apply a configuration to a resource by file name or stdin

11.3.5.2 - kubectl apply set-last-applied

Synopsis

Set the latest last-applied-configuration annotations by setting it to match the contents of a file. This results in the last-applied-configuration being updated as though 'kubectl apply -f<file>' was run, without updating any other parts of the object.

```
kubectl apply set-last-applied -f FILENAME
```

Examples

```
# Set the last-applied-configuration of a resource to match the contents of a file
kubectl apply set-last-applied -f deploy.yaml

# Execute set-last-applied against each configuration file in a directory
kubectl apply set-last-applied -f path/

# Set the last-applied-configuration of a resource to match the contents of a file
kubectl apply set-last-applied -f deploy.yaml --create-annotation=true
```

Options

--allow-missing-template-keys Default: true

If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to golang and jsonpath output formats.

--create-annotation

Will create 'last-applied-configuration' annotations if current objects doesn't have one

--dry-run string[="unchanged"] Default: "none"

Must be "none", "server", or "client". If client strategy, only print the object that would be sent, without sending it. If server strategy, submit server-side request without persisting the resource.

-f, --filename strings

Filename, directory, or URL to files that contains the last-applied-configuration annotations

-h, --help

help for set-last-applied

-o, --output string

Output format. One of: (json, yaml, name, go-template, go-template-file, template, templatefile, jsonpath, jsonpath-as-json, jsonpath-file).

--show-managed-fields

If true, keep the managedFields when printing objects in JSON or YAML format.

--template string

Template string or path to template file to use when -o=go-template, -o=go-template-file. The template format is golang templates [<http://golang.org/pkg/text/template/#pkg-overview>].

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl apply](#) - Apply a configuration to a resource by file name or stdin

11.3.5.3 - kubectl apply view-last-applied

Synopsis

View the latest last-applied-configuration annotations by type/name or file.

The default output will be printed to stdout in YAML format. You can use the `-o` option to change the output format.

```
kubectl apply view-last-applied (TYPE [NAME | -l label] | TYPE/NAME |
```

Examples

```
# View the last-applied-configuration annotations by type/name in YAML
kubectl apply view-last-applied deployment/nginx

# View the last-applied-configuration annotations by file in JSON
kubectl apply view-last-applied -f deploy.yaml -o json
```

Options

`--all`

Select all resources in the namespace of the specified resource types

`-f, --filename` strings

Filename, directory, or URL to files that contains the last-applied-configuration annotations

`-h, --help`

help for view-last-applied

`-k, --kustomize` string

Process the kustomization directory. This flag can't be used together with `-f` or `-R`.

`-o, --output` string Default: "yaml"

Output format. Must be one of (yaml, json)

`-R, --recursive`

Process the directory used in `-f, --filename` recursively. Useful when you want to manage related manifests organized within the same directory.

`-l, --selector` string

Selector (label query) to filter on, supports '=', '==', and '!='.(e.g. -l key1=value1,key2=value2). Matching objects must satisfy all of the specified label constraints.

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl apply](#) - Apply a configuration to a resource by file name or stdin

11.3.6 - kubectl attach

Synopsis

Attach to a process that is already running inside an existing container.

```
kubectl attach (POD | TYPE/NAME) -c CONTAINER
```

Examples

```
# Get output from running pod mypod; use the 'kubectl.kubernetes.io/default-container' annotation for selecting the container to be attached or the first container
kubectl attach mypod

# Get output from ruby-container from pod mypod
kubectl attach mypod -c ruby-container

# Switch to raw terminal mode; sends stdin to 'bash' in ruby-container and sends stdout/stderr from 'bash' back to the client
kubectl attach mypod -c ruby-container -i -t

# Get output from the first pod of a replica set named nginx
kubectl attach rs/nginx
```

Options

-c, --container string

Container name. If omitted, use the `kubectl.kubernetes.io/default-container` annotation for selecting the container to be attached or the first container in the pod will be chosen

-h, --help

help for attach

--pod-running-timeout duration Default: 1m0s

The length of time (like 5s, 2m, or 3h, higher than zero) to wait until at least one pod is running

-q, --quiet

Only print output from the remote session

-i, --stdin

Pass stdin to the container

-t, --tty

Stdin is a TTY

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cAdvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl](#) - kubectl controls the Kubernetes cluster manager

11.3.7 - kubectl auth

Synopsis

Inspect authorization.

```
kubectl auth [flags]
```

Options

-h, --help

help for auth

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cAdvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl](#) - kubectl controls the Kubernetes cluster manager
- [kubectl auth can-i](#) - Check whether an action is allowed
- [kubectl auth reconcile](#) - Reconciles rules for RBAC role, role binding, cluster role, and cluster role binding objects
- [kubectl auth whoami](#) - Experimental: Check self subject attributes

11.3.7.1 - kubectl auth can-i

Synopsis

Check whether an action is allowed.

VERB is a logical Kubernetes API verb like 'get', 'list', 'watch', 'delete', etc.

TYPE is a Kubernetes resource. Shortcuts and groups will be resolved.

NONRESOURCEURL is a partial URL that starts with "/". NAME is the name of a particular Kubernetes resource. This command pairs nicely with impersonation. See --as global flag.

```
kubectl auth can-i VERB [TYPE | TYPE/NAME | NONRESOURCEURL]
```

Examples

```
# Check to see if I can create pods in any namespace
kubectl auth can-i create pods --all-namespaces

# Check to see if I can list deployments in my current namespace
kubectl auth can-i list deployments.apps

# Check to see if service account "foo" of namespace "dev" can list
# You must be allowed to use impersonation for the global option "-"
kubectl auth can-i list pods --as=system:serviceaccount:dev:foo -n

# Check to see if I can do everything in my current namespace ("*")
kubectl auth can-i '*' '*'

# Check to see if I can get the job named "bar" in namespace "foo"
kubectl auth can-i list jobs.batch/bar -n foo

# Check to see if I can read pod logs
kubectl auth can-i get pods --subresource=log

# Check to see if I can access the URL /logs/
kubectl auth can-i get /logs/

# Check to see if I can approve certificates.k8s.io
kubectl auth can-i approve certificates.k8s.io

# List all allowed actions in namespace "foo"
kubectl auth can-i --list --namespace=foo
```

Options

-A, --all-namespaces

If true, check the specified action in all namespaces.

-h, --help

help for can-i

--list

If true, prints all allowed actions.

--no-headers

If true, prints allowed actions without headers

-q, --quiet

If true, suppress output and just return the exit code.

--subresource string

SubResource such as pod/log or deployment/scale

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl auth](#) - Inspect authorization

11.3.7.2 - kubectl auth reconcile

Synopsis

Reconciles rules for RBAC role, role binding, cluster role, and cluster role binding objects.

Missing objects are created, and the containing namespace is created for namespaced objects, if required.

Existing roles are updated to include the permissions in the input objects, and remove extra permissions if --remove-extra-permissions is specified.

Existing bindings are updated to include the subjects in the input objects, and remove extra subjects if --remove-extra-subjects is specified.

This is preferred to 'apply' for RBAC resources so that semantically-aware merging of rules and subjects is done.

```
kubectl auth reconcile -f FILENAME
```

Examples

```
# Reconcile RBAC resources from a file
kubectl auth reconcile -f my-rbac-rules.yaml
```

Options

--allow-missing-template-keys Default: true

If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to golang and jsonpath output formats.

--dry-run string[="unchanged"] Default: "none"

Must be "none", "server", or "client". If client strategy, only print the object that would be sent, without sending it. If server strategy, submit server-side request without persisting the resource.

-f, --filename strings

Filename, directory, or URL to files identifying the resource to reconcile.

-h, --help

help for reconcile

-k, --kustomize string

Process the kustomization directory. This flag can't be used together with -f or -R.

-o, --output string

Output format. One of: (json, yaml, name, go-template, go-template-file, template, templatefile, jsonpath, jsonpath-as-json, jsonpath-file).

-R, --recursive

Process the directory used in **-f, --filename** recursively. Useful when you want to manage related manifests organized within the same directory.

--remove-extra-permissions

If true, removes extra permissions added to roles

--remove-extra-subjects

If true, removes extra subjects added to rolebindings

--show-managed-fields

If true, keep the managedFields when printing objects in JSON or YAML format.

--template string

Template string or path to template file to use when **-o=go-template, -o=go-template-file**. The template format is golang templates [<http://golang.org/pkg/text/template/#pkg-overview>].

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl auth](#) - Inspect authorization

11.3.7.3 - kubectl auth whoami

Synopsis

Experimental: Check who you are and your attributes (groups, extra).

This command is helpful to get yourself aware of the current user especially when dynamic authentication, e.g., token webhook, auth is enabled in the Kubernetes cluster.

```
kubectl auth whoami
```

Examples

```
# Get your subject attributes
kubectl auth whoami

# Get your subject attributes in JSON format
kubectl auth whoami -o json
```

Options

--allow-missing-template-keys Default: true

If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to golang and jsonpath output formats.

-h, --help

help for whoami

-o, --output string

Output format. One of: (json, yaml, name, go-template, go-template-file, template, templatefile, jsonpath, jsonpath-as-json, jsonpath-file).

--show-managed-fields

If true, keep the managedFields when printing objects in JSON or YAML format.

--template string

Template string or path to template file to use when -o=go-template, -o=go-template-file. The template format is golang templates [http://golang.org/pkg/text/template/#pkg-overview].

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cAdvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl auth](#) - Inspect authorization

11.3.8 - kubectl autoscale

Synopsis

Creates an autoscaler that automatically chooses and sets the number of pods that run in a Kubernetes cluster.

Looks up a deployment, replica set, stateful set, or replication controller by name and creates an autoscaler that uses the given resource as a reference. An autoscaler can automatically increase or decrease number of pods deployed within the system as needed.

```
kubectl autoscale (-f FILENAME | TYPE NAME | TYPE/NAME) [--min=MINPOD]
```

Examples

```
# Auto scale a deployment "foo", with the number of pods between 2
kubectl autoscale deployment foo --min=2 --max=10

# Auto scale a replication controller "foo", with the number of pod
kubectl autoscale rc foo --max=5 --cpu-percent=80
```

Options

--allow-missing-template-keys Default: true

If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to golang and jsonpath output formats.

--cpu-percent int32 Default: -1

The target average CPU utilization (represented as a percent of requested CPU) over all the pods. If it's not specified or negative, a default autoscaling policy will be used.

--dry-run string[="unchanged"] Default: "none"

Must be "none", "server", or "client". If client strategy, only print the object that would be sent, without sending it. If server strategy, submit server-side request without persisting the resource.

--field-manager string Default: "kubectl-autoscale"

Name of the manager used to track field ownership.

-f, --filename strings

Filename, directory, or URL to files identifying the resource to autoscale.

-h, --help

help for autoscale

-k, --kustomize string

Process the kustomization directory. This flag can't be used together with -f or -R.

--max int32 Default: -1

The upper limit for the number of pods that can be set by the autoscaler. Required.

--min int32 Default: -1

The lower limit for the number of pods that can be set by the autoscaler. If it's not specified or negative, the server will apply a default value.

--name string

The name for the newly created object. If not specified, the name of the input resource will be used.

-o, --output string

Output format. One of: (json, yaml, name, go-template, go-template-file, template, templatefile, jsonpath, jsonpath-as-json, jsonpath-file).

-R, --recursive

Process the directory used in -f, --filename recursively. Useful when you want to manage related manifests organized within the same directory.

--save-config

If true, the configuration of current object will be saved in its annotation. Otherwise, the annotation will be unchanged. This flag is useful when you want to perform kubectl apply on this object in the future.

--show-managed-fields

If true, keep the managedFields when printing objects in JSON or YAML format.

--template string

Template string or path to template file to use when -o=go-template, -o=go-template-file. The template format is golang templates [<http://golang.org/pkg/text/template/#pkg-overview>].

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl](#) - kubectl controls the Kubernetes cluster manager

11.3.9 - kubectl certificate

Synopsis

Modify certificate resources.

```
kubectl certificate SUBCOMMAND
```

Options

-h, --help

help for certificate

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cAdvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl](#) - kubectl controls the Kubernetes cluster manager
- [kubectl certificate approve](#) - Approve a certificate signing request
- [kubectl certificate deny](#) - Deny a certificate signing request

11.3.9.1 - kubectl certificate approve

Synopsis

Approve a certificate signing request.

kubectl certificate approve allows a cluster admin to approve a certificate signing request (CSR). This action tells a certificate signing controller to issue a certificate to the requester with the attributes requested in the CSR.

SECURITY NOTICE: Depending on the requested attributes, the issued certificate can potentially grant a requester access to cluster resources or to authenticate as a requested identity. Before approving a CSR, ensure you understand what the signed certificate can do.

```
kubectl certificate approve (-f FILENAME | NAME)
```

Examples

```
# Approve CSR 'csr-sqgzp'  
kubectl certificate approve csr-sqgzp
```

Options

--allow-missing-template-keys Default: true

If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to golang and jsonpath output formats.

-f, --filename strings

Filename, directory, or URL to files identifying the resource to update

--force

Update the CSR even if it is already approved.

-h, --help

help for approve

-k, --kustomize string

Process the kustomization directory. This flag can't be used together with -f or -R.

-o, --output string

Output format. One of: (json, yaml, name, go-template, go-template-file, template, templatefile, jsonpath, jsonpath-as-json, jsonpath-file).

-R, --recursive

Process the directory used in -f, --filename recursively. Useful when you want to manage related manifests organized within the same directory.

--show-managed-fields

If true, keep the managedFields when printing objects in JSON or YAML format.

--template string

Template string or path to template file to use when -o=go-template, -o=go-template-file. The template format is golang templates [<http://golang.org/pkg/text/template/#pkg-overview>].

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl certificate](#) - Modify certificate resources

11.3.9.2 - kubectl certificate deny

Synopsis

Deny a certificate signing request.

kubectl certificate deny allows a cluster admin to deny a certificate signing request (CSR). This action tells a certificate signing controller to not to issue a certificate to the requester.

```
kubectl certificate deny (-f FILENAME | NAME)
```

Examples

```
# Deny CSR 'csr-sqgzp'  
kubectl certificate deny csr-sqgzp
```

Options

--allow-missing-template-keys Default: true

If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to golang and jsonpath output formats.

-f, --filename strings

Filename, directory, or URL to files identifying the resource to update

--force

Update the CSR even if it is already denied.

-h, --help

help for deny

-k, --kustomize string

Process the kustomization directory. This flag can't be used together with -f or -R.

-o, --output string

Output format. One of: (json, yaml, name, go-template, go-template-file, template, templatefile, jsonpath, jsonpath-as-json, jsonpath-file).

-R, --recursive

Process the directory used in -f, --filename recursively. Useful when you want to manage related manifests organized within the same directory.

--show-managed-fields

If true, keep the managedFields when printing objects in JSON or YAML format.

--template string

Template string or path to template file to use when `-o=go-template`, `-o=go-template-file`. The template format is golang templates [<http://golang.org/pkg/text/template/#pkg-overview>].

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl certificate](#) - Modify certificate resources

11.3.10 - kubectl cluster-info

Synopsis

Display addresses of the control plane and services with label kubernetes.io/cluster-service=true. To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.

```
kubectl cluster-info [flags]
```

Examples

```
# Print the address of the control plane and cluster services
kubectl cluster-info
```

Options

-h, --help

help for cluster-info

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl](#) - kubectl controls the Kubernetes cluster manager
- [kubectl cluster-info dump](#) - Dump relevant information for debugging and diagnosis

11.3.10.1 - kubectl cluster-info dump

Synopsis

Dump cluster information out suitable for debugging and diagnosing cluster problems. By default, dumps everything to stdout. You can optionally specify a directory with `--output-directory`. If you specify a directory, Kubernetes will build a set of files in that directory. By default, only dumps things in the current namespace and 'kube-system' namespace, but you can switch to a different namespace with the `--namespaces` flag, or specify `--all-namespaces` to dump all namespaces.

The command also dumps the logs of all of the pods in the cluster; these logs are dumped into different directories based on namespace and pod name.

```
kubectl cluster-info dump [flags]
```

Examples

```
# Dump current cluster state to stdout
kubectl cluster-info dump

# Dump current cluster state to /path/to/cluster-state
kubectl cluster-info dump --output-directory=/path/to/cluster-state

# Dump all namespaces to stdout
kubectl cluster-info dump --all-namespaces

# Dump a set of namespaces to /path/to/cluster-state
kubectl cluster-info dump --namespaces default,kube-system --output
```

Options

`-A, --all-namespaces`

If true, dump all namespaces. If true, `--namespaces` is ignored.

`--allow-missing-template-keys` Default: true

If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to golang and jsonpath output formats.

`-h, --help`

help for dump

`--namespaces` strings

A comma separated list of namespaces to dump.

-o, --output string Default: "json"

Output format. One of: (json, yaml, name, go-template, go-template-file, template, templatefile, jsonpath, jsonpath-as-json, jsonpath-file).

--output-directory string

Where to output the files. If empty or '-' uses stdout, otherwise creates a directory hierarchy in that directory

--pod-running-timeout duration Default: 20s

The length of time (like 5s, 2m, or 3h, higher than zero) to wait until at least one pod is running

--show-managed-fields

If true, keep the managedFields when printing objects in JSON or YAML format.

--template string

Template string or path to template file to use when -o=go-template, -o=go-template-file. The template format is golang templates [<http://golang.org/pkg/text/template/#pkg-overview>].

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl cluster-info](#) - Display cluster information

11.3.11 - kubectl completion

Synopsis

Output shell completion code for the specified shell (bash, zsh, fish, or powershell). The shell code must be evaluated to provide interactive completion of kubectl commands. This can be done by sourcing it from the .bash_profile.

Detailed instructions on how to do this are available here:

```
for macOS:  
https://kubernetes.io/docs/tasks/tools/install-kubectl-macos/#enable-kubectl-completion  
  
for linux:  
https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/#enable-kubectl-completion  
  
for windows:  
https://kubernetes.io/docs/tasks/tools/install-kubectl-windows/#enable-kubectl-completion
```

Note for zsh users: [1] zsh completions are only supported in versions of zsh >= 5.2.

```
kubectl completion SHELL
```

Examples

```
# Installing bash completion on macOS using homebrew
## If running Bash 3.2 included with macOS
brew install bash-completion
## or, if running Bash 4.1+
brew install bash-completion@2
## If kubectl is installed via homebrew, this should start working
## If you've installed via other means, you may need add the complete
kubectl completion bash > $(brew --prefix)/etc/bash_completion.d/ku

# Installing bash completion on Linux
## If bash-completion is not installed on Linux, install the 'bash-
## via your distribution's package manager.
## Load the kubectl completion code for bash into the current shell
source <(kubectl completion bash)
## Write bash completion code to a file and source it from .bash_pr
kubectl completion bash > ~/.kube/completion.bash.inc
printf "
# kubectl shell completion
source '$HOME/.kube/completion.bash.inc'
" >> $HOME/.bash_profile
source $HOME/.bash_profile

# Load the kubectl completion code for zsh[1] into the current shel
source <(kubectl completion zsh)
# Set the kubectl completion code for zsh[1] to autoload on startup
kubectl completion zsh > "${fpath[1]}/_kubectl"

# Load the kubectl completion code for fish[2] into the current she
kubectl completion fish | source
# To load completions for each session, execute once:
kubectl completion fish > ~/.config/fish/completions/kubectl.fish

# Load the kubectl completion code for powershell into the current
kubectl completion powershell | Out-String | Invoke-Expression
# Set kubectl completion code for powershell to run on startup
## Save completion code to a script and execute in the profile
kubectl completion powershell > $HOME\.kube\completion.ps1
Add-Content $PROFILE "$HOME\.kube\completion.ps1"
## Execute completion code in the profile
Add-Content $PROFILE "if (Get-Command kubectl -ErrorAction Silently
kubectl completion powershell | Out-String | Invoke-Expression
}"
## Add completion code directly to the $PROFILE script
kubectl completion powershell >> $PROFILE
```

Options

-h, --help

help for completion

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

Path to the kubeconfig file to use for CLI requests.

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl](#) - kubectl controls the Kubernetes cluster manager

11.3.12 - kubectl config

Synopsis

Modify kubeconfig files using subcommands like "kubectl config set current-context my-context".

The loading order follows these rules:

1. If the --kubeconfig flag is set, then only that file is loaded. The flag may only be set once and no merging takes place.
2. If \$KUBECONFIG environment variable is set, then it is used as a list of paths (normal path delimiting rules for your system). These paths are merged. When a value is modified, it is modified in the file that defines the stanza. When a value is created, it is created in the first file that exists. If no files in the chain exist, then it creates the last file in the list.
3. Otherwise, \${HOME}/.kube/config is used and no merging takes place.

```
kubectl config SUBCOMMAND
```

Options

-h, --help

help for config

--kubeconfig string

use a particular kubeconfig file

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cAdvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl](#) - kubectl controls the Kubernetes cluster manager
- [kubectl config current-context](#) - Display the current-context
- [kubectl config delete-cluster](#) - Delete the specified cluster from the kubeconfig
- [kubectl config delete-context](#) - Delete the specified context from the kubeconfig
- [kubectl config delete-user](#) - Delete the specified user from the kubeconfig
- [kubectl config get-clusters](#) - Display clusters defined in the kubeconfig
- [kubectl config get-contexts](#) - Describe one or many contexts
- [kubectl config get-users](#) - Display users defined in the kubeconfig
- [kubectl config rename-context](#) - Rename a context from the kubeconfig file
- [kubectl config set](#) - Set an individual value in a kubeconfig file
- [kubectl config set-cluster](#) - Set a cluster entry in kubeconfig
- [kubectl config set-context](#) - Set a context entry in kubeconfig
- [kubectl config set-credentials](#) - Set a user entry in kubeconfig
- [kubectl config unset](#) - Unset an individual value in a kubeconfig file
- [kubectl config use-context](#) - Set the current-context in a kubeconfig file
- [kubectl config view](#) - Display merged kubeconfig settings or a specified kubeconfig file

11.3.12.1 - kubectl config current-context

Synopsis

Display the current-context.

```
kubectl config current-context [flags]
```

Examples

```
# Display the current-context
kubectl config current-context
```

Options

-h, --help

help for current-context

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

use a particular kubeconfig file

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl config](#) - Modify kubeconfig files

11.3.12.2 - kubectl config delete-cluster

Synopsis

Delete the specified cluster from the kubeconfig.

```
kubectl config delete-cluster NAME
```

Examples

```
# Delete the minikube cluster
kubectl config delete-cluster minikube
```

Options

-h, --help

help for delete-cluster

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

use a particular kubeconfig file

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl config](#) - Modify kubeconfig files

11.3.12.3 - kubectl config delete-context

Synopsis

Delete the specified context from the kubeconfig.

```
kubectl config delete-context NAME
```

Examples

```
# Delete the context for the minikube cluster
kubectl config delete-context minikube
```

Options

-h, --help

help for delete-context

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

use a particular kubeconfig file

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl config](#) - Modify kubeconfig files

11.3.12.4 - kubectl config delete-user

Synopsis

Delete the specified user from the kubeconfig.

```
kubectl config delete-user NAME
```

Examples

```
# Delete the minikube user
kubectl config delete-user minikube
```

Options

-h, --help

help for delete-user

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

use a particular kubeconfig file

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl config](#) - Modify kubeconfig files

11.3.12.5 - kubectl config get-clusters

Synopsis

Display clusters defined in the kubeconfig.

```
kubectl config get-clusters [flags]
```

Examples

```
# List the clusters that kubectl knows about
kubectl config get-clusters
```

Options

-h, --help

help for get-clusters

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

use a particular kubeconfig file

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl config](#) - Modify kubeconfig files

11.3.12.6 - kubectl config get-contexts

Synopsis

Display one or many contexts from the kubeconfig file.

```
kubectl config get-contexts [(-o|--output=)name]
```

Examples

```
# List all the contexts in your kubeconfig file
kubectl config get-contexts

# Describe one context in your kubeconfig file
kubectl config get-contexts my-context
```

Options

-h, --help

help for get-contexts

--no-headers

When using the default or custom-column output format, don't print headers (default print headers).

-o, --output string

Output format. One of: (name).

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

use a particular kubeconfig file

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

 Password for basic authentication to the API server

--profile string Default: "none"

 Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

 Name of the file to write the profile to

--request-timeout string Default: "0"

 The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

 The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

 Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

 database name

--storage-driver-host string Default: "localhost:8086"

 database host:port

--storage-driver-password string Default: "root"

 database password

--storage-driver-secure

 use secure connection with database

--storage-driver-table string Default: "stats"

 table name

--storage-driver-user string Default: "root"

 database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl config](#) - Modify kubeconfig files

11.3.12.7 - kubectl config get-users

Synopsis

Display users defined in the kubeconfig.

```
kubectl config get-users [flags]
```

Examples

```
# List the users that kubectl knows about
kubectl config get-users
```

Options

-h, --help

help for get-users

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

use a particular kubeconfig file

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl config](#) - Modify kubeconfig files

11.3.12.8 - kubectl config rename-context

Synopsis

Renames a context from the kubeconfig file.

CONTEXT_NAME is the context name that you want to change.

NEW_NAME is the new name you want to set.

Note: If the context being renamed is the 'current-context', this field will also be updated.

```
kubectl config rename-context CONTEXT_NAME NEW_NAME
```

Examples

```
# Rename the context 'old-name' to 'new-name' in your kubeconfig file
kubectl config rename-context old-name new-name
```

Options

-h, --help

help for rename-context

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

use a particular kubeconfig file

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl config](#) - Modify kubeconfig files

11.3.12.9 - kubectl config set

Synopsis

Set an individual value in a kubeconfig file.

PROPERTY_NAME is a dot delimited name where each token represents either an attribute name or a map key. Map keys may not contain dots.

PROPERTY_VALUE is the new value you want to set. Binary fields such as 'certificate-authority-data' expect a base64 encoded string unless the --set-raw-bytes flag is used.

Specifying an attribute name that already exists will merge new fields on top of existing values.

```
kubectl config set PROPERTY_NAME PROPERTY_VALUE
```

Examples

```
# Set the server field on the my-cluster cluster to https://1.2.3.4
kubectl config set clusters.my-cluster.server https://1.2.3.4

# Set the certificate-authority-data field on the my-cluster cluster
kubectl config set clusters.my-cluster.certificate-authority-data $

# Set the cluster field in the my-context context to my-cluster
kubectl config set contexts.my-context.cluster my-cluster

# Set the client-key-data field in the cluster-admin user using --set-raw-bytes
kubectl config set users.cluster-admin.client-key-data cert_data_hex
```

Options

-h, --help

help for set

--set-raw-bytes tristate[=true]

When writing a []byte PROPERTY_VALUE, write the given string directly without base64 decoding.

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

use a particular kubeconfig file

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl config](#) - Modify kubeconfig files

11.3.12.10 - kubectl config set-cluster

Synopsis

Set a cluster entry in kubeconfig.

Specifying a name that already exists will merge new fields on top of existing values for those fields.

```
kubectl config set-cluster NAME [--server=server] [--certificate-auth
```

Examples

```
# Set only the server field on the e2e cluster entry without touching the rest
kubectl config set-cluster e2e --server=https://1.2.3.4

# Embed certificate authority data for the e2e cluster entry
kubectl config set-cluster e2e --embed-certs --certificate-authority=certs/ca.crt

# Disable cert checking for the e2e cluster entry
kubectl config set-cluster e2e --insecure-skip-tls-verify=true

# Set the custom TLS server name to use for validation for the e2e cluster entry
kubectl config set-cluster e2e --tls-server-name=my-cluster-name

# Set the proxy URL for the e2e cluster entry
kubectl config set-cluster e2e --proxy-url=https://1.2.3.4
```

Options

--certificate-authority string

Path to certificate-authority file for the cluster entry in kubeconfig

--embed-certs tristate[=true]

embed-certs for the cluster entry in kubeconfig

-h, --help

help for set-cluster

--insecure-skip-tls-verify tristate[=true]

insecure-skip-tls-verify for the cluster entry in kubeconfig

--proxy-url string

proxy-url for the cluster entry in kubeconfig

--server string

server for the cluster entry in kubeconfig

--tls-server-name string

tls-server-name for the cluster entry in kubeconfig

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--cluster string

The name of the kubeconfig cluster to use

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--kubeconfig string

use a particular kubeconfig file

--match-server-version

Require server version to match client version

-n, --namespace string

If present, the namespace scope for this CLI request

--password string

Password for basic authentication to the API server

--profile string Default: "none"

Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

Name of the file to write the profile to

--request-timeout string Default: "0"

The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

--storage-driver-buffer-duration duration Default: 1m0s

Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

database name

--storage-driver-host string Default: "localhost:8086"

database host:port

--storage-driver-password string Default: "root"

database password

--storage-driver-secure

use secure connection with database

--storage-driver-table string Default: "stats"

table name

--storage-driver-user string Default: "root"

database username

--token string

Bearer token for authentication to the API server

--user string

The name of the kubeconfig user to use

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl config](#) - Modify kubeconfig files

11.3.12.11 - kubectl config set-context

Synopsis

Set a context entry in kubeconfig.

Specifying a name that already exists will merge new fields on top of existing values for those fields.

```
kubectl config set-context [NAME | --current] [--cluster=cluster_nick
```

Examples

```
# Set the user field on the gce context entry without touching other fields
kubectl config set-context gce --user=cluster-admin
```

Options

--cluster string

cluster for the context entry in kubeconfig

--current

Modify the current context

-h, --help

help for set-context

--namespace string

namespace for the context entry in kubeconfig

--user string

user for the context entry in kubeconfig

--as string

Username to impersonate for the operation. User could be a regular user or a service account in a namespace.

--as-group strings

Group to impersonate for the operation, this flag can be repeated to specify multiple groups.

--as-uid string

UID to impersonate for the operation.

--cache-dir string Default: "\$HOME/.kube/cache"

Default cache directory

--certificate-authority string

Path to a cert file for the certificate authority

--client-certificate string

Path to a client certificate file for TLS

--client-key string

Path to a client key file for TLS

--context string

The name of the kubeconfig context to use

--default-not-ready-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for notReady:NoExecute that is added by default to every pod that does not already have such a toleration.

--default-unreachable-toleration-seconds int Default: 300

Indicates the tolerationSeconds of the toleration for unreachable:NoExecute that is added by default to every pod that does not already have such a toleration.

--disable-compression

If true, opt-out of response compression for all requests to the server

--insecure-skip-tls-verify

If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure

--kubeconfig string

use a particular kubeconfig file

--match-server-version

Require server version to match client version

--password string

 Password for basic authentication to the API server

--profile string Default: "none"

 Name of profile to capture. One of (none|cpu|heap|goroutine|threadcreate|block|mutex)

--profile-output string Default: "profile.pprof"

 Name of the file to write the profile to

--request-timeout string Default: "0"

 The length of time to wait before giving up on a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h). A value of zero means don't timeout requests.

-s, --server string

 The address and port of the Kubernetes API server

--storage-driver-buffer-duration duration Default: 1m0s

 Writes in the storage driver will be buffered for this duration, and committed to the non memory backends as a single transaction

--storage-driver-db string Default: "cadvisor"

 database name

--storage-driver-host string Default: "localhost:8086"

 database host:port

--storage-driver-password string Default: "root"

 database password

--storage-driver-secure

 use secure connection with database

--storage-driver-table string Default: "stats"

 table name

--storage-driver-user string Default: "root"

 database username

--tls-server-name string

Server name to use for server certificate validation. If it is not provided, the hostname used to contact the server is used

--token string

Bearer token for authentication to the API server

--username string

Username for basic authentication to the API server

--version version[=true]

--version, --version=raw prints version information and quits; --version=vX.Y.Z... sets the reported version

--warnings-as-errors

Treat warnings received from the server as errors and exit with a non-zero exit code

See Also

- [kubectl config](#) - Modify kubeconfig files