mattklein123 http: remove getAll() header map API and switch all usages to get() (#... ✓ History

10 contributors

867 lines (748 sloc) | 41.5 KB

```
syntax = "proto3";

package envoy.api.v2;

import "envoy/api/v2/auth/tls.proto";
import "envoy/api/v2/cluster/circuit_breaker.proto";
import "envoy/api/v2/cluster/filter.proto";
import "envoy/api/v2/cluster/outlier_detection.proto";
import "envoy/api/v2/core/address.proto";
import "envoy/api/v2/core/base.proto";
import "envoy/api/v2/core/config_source.proto";
import "envoy/api/v2/core/health_check.proto";
import "envoy/api/v2/core/protocol.proto";
import "envoy/api/v2/endpoint.proto";
import "envoy/type/percent.proto";

import "google/protobuf/any.proto";
import "google/protobuf/duration.proto";
import "google/protobuf/struct.proto";
import "google/protobuf/wrappers.proto";

import "envoy/annotations/deprecation.proto";
import "udpa/annotations/migrate.proto";
import "udpa/annotations/status.proto";
import "validate/validate.proto";

option java_package = "io.envoyproxy.envoy.api.v2";
option java_outer_classname = "ClusterProto";
option java_multiple_files = true;
option (udpa.annotations.file_migrate).move_to_package = "envoy.config.cluster.v3";
option (udpa.annotations.file_status).package_version_status = FROZEN;

// [#protodoc-title: Cluster configuration]

// Configuration for a single upstream cluster.
// [#next-free-field: 48]
message Cluster {
  // Refer to :ref:`service discovery type <arch_overview_service_discovery_types>`
  // for an explanation on each type.
  enum DiscoveryType {
    // Refer to the :ref:`static discovery type<arch_overview_service_discovery_types_static>`
    // for an explanation.
    STATIC = 0;

    // Refer to the :ref:`strict DNS discovery
    // type<arch_overview_service_discovery_types_strict_dns>`
    // for an explanation.
    STRICT_DNS = 1;

    // Refer to the :ref:`logical DNS discovery
    // type<arch_overview_service_discovery_types_logical_dns>`
    // for an explanation.
    LOGICAL_DNS = 2;

    // Refer to the :ref:`service discovery type<arch_overview_service_discovery_types_eds>`
    // for an explanation.
    EDS = 3;

    // Refer to the :ref:`original destination discovery
    // type<arch_overview_service_discovery_types_original_destination>`
    // for an explanation.
    ORIGINAL_DST = 4;
  }

  // Refer to :ref:`load balancer type <arch_overview_load_balancing_types>` architecture
  // overview section for information on each type.
  enum LbPolicy {
    // Refer to the :ref:`round robin load balancing
    // policy<arch_overview_load_balancing_types_round_robin>`
    // for an explanation.
    ROUND_ROBIN = 0;

    // Refer to the :ref:`least request load balancing
    // policy<arch_overview_load_balancing_types_least_request>`
    // for an explanation.
    LEAST_REQUEST = 1;

    // Refer to the :ref:`ring hash load balancing
```

```
    // policy<arch_overview_load_balancing_types_ring_hash>`
    // for an explanation.
    RING_HASH = 2;

    // Refer to the :ref:`random load balancing
    // policy<arch_overview_load_balancing_types_random>`
    // for an explanation.
    RANDOM = 3;

    // Refer to the :ref:`original destination load balancing
    // policy<arch_overview_load_balancing_types_original_destination>`
    // for an explanation.
    //
    // .. attention::
    //
    //   **This load balancing policy is deprecated**. Use CLUSTER_PROVIDED instead.
    //
    ORIGINAL_DST_LB = 4 [deprecated = true, (envoy.annotations.disallowed_by_default_enum) = true];

    // Refer to the :ref:`Maglev load balancing policy<arch_overview_load_balancing_types_maglev>`
    // for an explanation.
    MAGLEV = 5;

    // This load balancer type must be specified if the configured cluster provides a cluster
    // specific load balancer. Consult the configured cluster's documentation for whether to set
    // this option or not.
    CLUSTER_PROVIDED = 6;

    // [#not-implemented-hide:] Use the new :ref:`load_balancing_policy
    // <envoy_api_field_Cluster.load_balancing_policy>` field to determine the LB policy.
    // [#next-major-version: In the v3 API, we should consider deprecating the lb_policy field
    // and instead using the new load_balancing_policy field as the one and only mechanism for
    // configuring this.]
    LOAD_BALANCING_POLICY_CONFIG = 7;
  }

  // When V4_ONLY is selected, the DNS resolver will only perform a lookup for
  // addresses in the IPv4 family. If V6_ONLY is selected, the DNS resolver will
  // only perform a lookup for addresses in the IPv6 family. If AUTO is
  // specified, the DNS resolver will first perform a lookup for addresses in
  // the IPv6 family and fallback to a lookup for addresses in the IPv4 family.
  // For cluster types other than
  // :ref:`STRICT_DNS<envoy_api_enum_value_Cluster.DiscoveryType.STRICT_DNS>` and
  // :ref:`LOGICAL_DNS<envoy_api_enum_value_Cluster.DiscoveryType.LOGICAL_DNS>`,
  // this setting is
  // ignored.
  enum DnsLookupFamily {
    AUTO = 0;
    V4_ONLY = 1;
    V6_ONLY = 2;
  }

  enum ClusterProtocolSelection {
    // Cluster can only operate on one of the possible upstream protocols (HTTP1.1, HTTP2).
    // If :ref:`http2_protocol_options <envoy_api_field_Cluster.http2_protocol_options>` are
    // present, HTTP2 will be used, otherwise HTTP1.1 will be used.
    USE_CONFIGURED_PROTOCOL = 0;

    // Use HTTP1.1 or HTTP2, depending on which one is used on the downstream connection.
    USE_DOWNSTREAM_PROTOCOL = 1;
  }

  // TransportSocketMatch specifies what transport socket config will be used
  // when the match conditions are satisfied.
  message TransportSocketMatch {
    // The name of the match, used in stats generation.
    string name = 1 [(validate.rules).string = {min_len: 1}];

    // Optional endpoint metadata match criteria.
    // The connection to the endpoint with metadata matching what is set in this field
    // will use the transport socket configuration specified here.
    // The endpoint's metadata entry in *envoy.transport_socket_match* is used to match
    // against the values specified in this field.
    google.protobuf.Struct match = 2;

    // The configuration of the transport socket.
    core.TransportSocket transport_socket = 3;
  }

  // Extended cluster type.
  message CustomClusterType {
    // The type of the cluster to instantiate. The name must match a supported cluster type.
    string name = 1 [(validate.rules).string = {min_bytes: 1}];

    // Cluster specific configuration which depends on the cluster being instantiated.
    // See the supported cluster for further documentation.
    google.protobuf.Any typed_config = 2;
  }

  // Only valid when discovery type is EDS.
  message EdsClusterConfig {
    // Configuration for the source of EDS updates for this Cluster.
    core.ConfigSource eds_config = 1;

    // Optional alternative to cluster name to present to EDS. This does not
    // have the same restrictions as cluster name, i.e. it may be arbitrary
    // length.
    string service_name = 2;
```

```
  }

  // Optionally divide the endpoints in this cluster into subsets defined by
  // endpoint metadata and selected by route and weighted cluster metadata.
  // [#next-free-field: 8]
  message LbSubsetConfig {
    // If NO_FALLBACK is selected, a result
    // equivalent to no healthy hosts is reported. If ANY_ENDPOINT is selected,
    // any cluster endpoint may be returned (subject to policy, health checks,
    // etc). If DEFAULT_SUBSET is selected, load balancing is performed over the
    // endpoints matching the values from the default_subset field.
    enum LbSubsetFallbackPolicy {
      NO_FALLBACK = 0;
      ANY_ENDPOINT = 1;
      DEFAULT_SUBSET = 2;
    }

    // Specifications for subsets.
    message LbSubsetSelector {
      // Allows to override top level fallback policy per selector.
      enum LbSubsetSelectorFallbackPolicy {
        // If NOT_DEFINED top level config fallback policy is used instead.
        NOT_DEFINED = 0;

        // If NO_FALLBACK is selected, a result equivalent to no healthy hosts is reported.
        NO_FALLBACK = 1;

        // If ANY_ENDPOINT is selected, any cluster endpoint may be returned
        // (subject to policy, health checks, etc).
        ANY_ENDPOINT = 2;

        // If DEFAULT_SUBSET is selected, load balancing is performed over the
        // endpoints matching the values from the default_subset field.
        DEFAULT_SUBSET = 3;

        // If KEYS_SUBSET is selected, subset selector matching is performed again with metadata
        // keys reduced to
        // :ref:`fallback_keys_subset<envoy_api_field_Cluster.LbSubsetConfig.LbSubsetSelector.fallback_keys_subset>`.
        // It allows for a fallback to a different, less specific selector if some of the keys of
        // the selector are considered optional.
        KEYS_SUBSET = 4;
      }

      // List of keys to match with the weighted cluster metadata.
      repeated string keys = 1;

      // The behavior used when no endpoint subset matches the selected route's
      // metadata.
      LbSubsetSelectorFallbackPolicy fallback_policy = 2
          [(validate.rules).enum = {defined_only: true}];

      // Subset of
      // :ref:`keys<envoy_api_field_Cluster.LbSubsetConfig.LbSubsetSelector.keys>` used by
      // :ref:`KEYS_SUBSET<envoy_api_enum_value_Cluster.LbSubsetConfig.LbSubsetSelector.LbSubsetSelectorFallbackPolicy.KEYS_SUBSET>`
      // fallback policy.
      // It has to be a non empty list if KEYS_SUBSET fallback policy is selected.
      // For any other fallback policy the parameter is not used and should not be set.
      // Only values also present in
      // :ref:`keys<envoy_api_field_Cluster.LbSubsetConfig.LbSubsetSelector.keys>` are allowed, but
      // `fallback_keys_subset` cannot be equal to `keys`.
      repeated string fallback_keys_subset = 3;
    }

    // The behavior used when no endpoint subset matches the selected route's
    // metadata. The value defaults to
    // :ref:`NO_FALLBACK<envoy_api_enum_value_Cluster.LbSubsetConfig.LbSubsetFallbackPolicy.NO_FALLBACK>`.
    LbSubsetFallbackPolicy fallback_policy = 1 [(validate.rules).enum = {defined_only: true}];

    // Specifies the default subset of endpoints used during fallback if
    // fallback_policy is
    // :ref:`DEFAULT_SUBSET<envoy_api_enum_value_Cluster.LbSubsetConfig.LbSubsetFallbackPolicy.DEFAULT_SUBSET>`.
    // Each field in default_subset is
    // compared to the matching LbEndpoint.Metadata under the *envoy.lb*
    // namespace. It is valid for no hosts to match, in which case the behavior
    // is the same as a fallback_policy of
    // :ref:`NO_FALLBACK<envoy_api_enum_value_Cluster.LbSubsetConfig.LbSubsetFallbackPolicy.NO_FALLBACK>`.
    google.protobuf.Struct default_subset = 2;

    // For each entry, LbEndpoint.Metadata's
    // *envoy.lb* namespace is traversed and a subset is created for each unique
    // combination of key and value. For example:
    //
    // .. code-block:: json
    //
    //   { "subset_selectors": [
    //       { "keys": [ "version" ] },
    //       { "keys": [ "stage", "hardware_type" ] }
    //   ]}
    //
    // A subset is matched when the metadata from the selected route and
    // weighted cluster contains the same keys and values as the subset's
    // metadata. The same host may appear in multiple subsets.
    repeated LbSubsetSelector subset_selectors = 3;

    // If true, routing to subsets will take into account the localities and locality weights of the
    // endpoints when making the routing decision.
    //
    // There are some potential pitfalls associated with enabling this feature, as the resulting
```

```
    // traffic split after applying both a subset match and locality weights might be undesirable.
    //
    // Consider for example a situation in which you have 50/50 split across two localities X/Y
    // which have 100 hosts each without subsetting. If the subset LB results in X having only 1
    // host selected but Y having 100, then a lot more load is being dumped on the single host in X
    // than originally anticipated in the load balancing assignment delivered via EDS.
    bool locality_weight_aware = 4;

    // When used with locality_weight_aware, scales the weight of each locality by the ratio
    // of hosts in the subset vs hosts in the original subset. This aims to even out the load
    // going to an individual locality if said locality is disproportionately affected by the
    // subset predicate.
    bool scale_locality_weight = 5;

    // If true, when a fallback policy is configured and its corresponding subset fails to find
    // a host this will cause any host to be selected instead.
    //
    // This is useful when using the default subset as the fallback policy, given the default
    // subset might become empty. With this option enabled, if that happens the LB will attempt
    // to select a host from the entire cluster.
    bool panic_mode_any = 6;

    // If true, metadata specified for a metadata key will be matched against the corresponding
    // endpoint metadata if the endpoint metadata matches the value exactly OR it is a list value
    // and any of the elements in the list matches the criteria.
    bool list_as_any = 7;
  }

  // Specific configuration for the LeastRequest load balancing policy.
  message LeastRequestLbConfig {
    // The number of random healthy hosts from which the host with the fewest active requests will
    // be chosen. Defaults to 2 so that we perform two-choice selection if the field is not set.
    google.protobuf.UInt32Value choice_count = 1 [(validate.rules).uint32 = {gte: 2}];
  }

  // Specific configuration for the :ref:`RingHash<arch_overview_load_balancing_types_ring_hash>`
  // load balancing policy.
  message RingHashLbConfig {
    // The hash function used to hash hosts onto the ketama ring.
    enum HashFunction {
      // Use `xxHash <https://github.com/Cyan4973/xxHash>`_, this is the default hash function.
      XX_HASH = 0;

      // Use `MurmurHash2 <https://sites.google.com/site/murmurhash/>`_, this is compatible with
      // std:hash<string> in GNU libstdc++ 3.4.20 or above. This is typically the case when compiled
      // on Linux and not macOS.
      MURMUR_HASH_2 = 1;
    }

    reserved 2;

    // Minimum hash ring size. The larger the ring is (that is, the more hashes there are for each
    // provided host) the better the request distribution will reflect the desired weights. Defaults
    // to 1024 entries, and limited to 8M entries. See also
    // :ref:`maximum_ring_size<envoy_api_field_Cluster.RingHashLbConfig.maximum_ring_size>`.
    google.protobuf.UInt64Value minimum_ring_size = 1 [(validate.rules).uint64 = {lte: 8388608}];

    // The hash function used to hash hosts onto the ketama ring. The value defaults to
    // :ref:`XX_HASH<envoy_api_enum_value_Cluster.RingHashLbConfig.HashFunction.XX_HASH>`.
    HashFunction hash_function = 3 [(validate.rules).enum = {defined_only: true}];

    // Maximum hash ring size. Defaults to 8M entries, and limited to 8M entries, but can be lowered
    // to further constrain resource use. See also
    // :ref:`minimum_ring_size<envoy_api_field_Cluster.RingHashLbConfig.minimum_ring_size>`.
    google.protobuf.UInt64Value maximum_ring_size = 4 [(validate.rules).uint64 = {lte: 8388608}];
  }

  // Specific configuration for the
  // :ref:`Original Destination <arch_overview_load_balancing_types_original_destination>`
  // load balancing policy.
  message OriginalDstLbConfig {
    // When true, :ref:`x-envoy-original-dst-host
    // <config_http_conn_man_headers_x-envoy-original-dst-host>` can be used to override destination
    // address.
    //
    // .. attention::
    //
    //   This header isn't sanitized by default, so enabling this feature allows HTTP clients to
    //   route traffic to arbitrary hosts and/or ports, which may have serious security
    //   consequences.
    //
    // .. note::
    //
    //   If the header appears multiple times only the first value is used.
    bool use_http_header = 1;
  }

  // Common configuration for all load balancer implementations.
  // [#next-free-field: 8]
  message CommonLbConfig {
    // Configuration for :ref:`zone aware routing
    // <arch_overview_load_balancing_zone_aware_routing>`.
    message ZoneAwareLbConfig {
      // Configures percentage of requests that will be considered for zone aware routing
      // if zone aware routing is configured. If not specified, the default is 100%.
      // * :ref:`runtime values <config_cluster_manager_cluster_runtime_zone_routing>`.
      // * :ref:`Zone aware routing support <arch_overview_load_balancing_zone_aware_routing>`.
      type.Percent routing_enabled = 1;
```

```
373
374         // Configures minimum upstream cluster size required for zone aware routing
375         // If upstream cluster size is less than specified, zone aware routing is not performed
376         // even if zone aware routing is configured. If not specified, the default is 6.
377         // * :ref:`runtime values <config_cluster_manager_cluster_runtime_zone_routing>`.
378         // * :ref:`Zone aware routing support <arch_overview_load_balancing_zone_aware_routing>`.
379         google.protobuf.UInt64Value min_cluster_size = 2;
380
381         // If set to true, Envoy will not consider any hosts when the cluster is in :ref:`panic
382         // mode<arch_overview_load_balancing_panic_threshold>`. Instead, the cluster will fail all
383         // requests as if all hosts are unhealthy. This can help avoid potentially overwhelming a
384         // failing service.
385         bool fail_traffic_on_panic = 3;
386       }
387
388       // Configuration for :ref:`locality weighted load balancing
389       // <arch_overview_load_balancing_locality_weighted_lb>`
390       message LocalityWeightedLbConfig {
391       }
392
393       // Common Configuration for all consistent hashing load balancers (MaglevLb, RingHashLb, etc.)
394       message ConsistentHashingLbConfig {
395         // If set to `true`, the cluster will use hostname instead of the resolved
396         // address as the key to consistently hash to an upstream host. Only valid for StrictDNS clusters with hostnames which resolve to a single IP address.
397         bool use_hostname_for_hashing = 1;
398       }
399
400       // Configures the :ref:`healthy panic threshold <arch_overview_load_balancing_panic_threshold>`.
401       // If not specified, the default is 50%.
402       // To disable panic mode, set to 0%.
403       //
404       // .. note::
405       //   The specified percent will be truncated to the nearest 1%.
406       type.Percent healthy_panic_threshold = 1;
407
408       oneof locality_config_specifier {
409         ZoneAwareLbConfig zone_aware_lb_config = 2;
410
411         LocalityWeightedLbConfig locality_weighted_lb_config = 3;
412       }
413
414       // If set, all health check/weight/metadata updates that happen within this duration will be
415       // merged and delivered in one shot when the duration expires. The start of the duration is when
416       // the first update happens. This is useful for big clusters, with potentially noisy deploys
417       // that might trigger excessive CPU usage due to a constant stream of healthcheck state changes
418       // or metadata updates. The first set of updates to be seen apply immediately (e.g.: a new
419       // cluster). Please always keep in mind that the use of sandbox technologies may change this
420       // behavior.
421       //
422       // If this is not set, we default to a merge window of 1000ms. To disable it, set the merge
423       // window to 0.
424       //
425       // Note: merging does not apply to cluster membership changes (e.g.: adds/removes); this is
426       // because merging those updates isn't currently safe. See
427       // https://github.com/envoyproxy/envoy/pull/3941.
428       google.protobuf.Duration update_merge_window = 4;
429
430       // If set to true, Envoy will not consider new hosts when computing load balancing weights until
431       // they have been health checked for the first time. This will have no effect unless
432       // active health checking is also configured.
433       //
434       // Ignoring a host means that for any load balancing calculations that adjust weights based
435       // on the ratio of eligible hosts and total hosts (priority spillover, locality weighting and
436       // panic mode) Envoy will exclude these hosts in the denominator.
437       //
438       // For example, with hosts in two priorities P0 and P1, where P0 looks like
439       // {healthy, unhealthy (new), unhealthy (new)}
440       // and where P1 looks like
441       // {healthy, healthy}
442       // all traffic will still hit P0, as 1 / (3 - 2) = 1.
443       //
444       // Enabling this will allow scaling up the number of hosts for a given cluster without entering
445       // panic mode or triggering priority spillover, assuming the hosts pass the first health check.
446       //
447       // If panic mode is triggered, new hosts are still eligible for traffic; they simply do not
448       // contribute to the calculation when deciding whether panic mode is enabled or not.
449       bool ignore_new_hosts_until_first_hc = 5;
450
451       // If set to `true`, the cluster manager will drain all existing
452       // connections to upstream hosts whenever hosts are added or removed from the cluster.
453       bool close_connections_on_host_set_change = 6;
454
455       // Common Configuration for all consistent hashing load balancers (MaglevLb, RingHashLb, etc.)
456       ConsistentHashingLbConfig consistent_hashing_lb_config = 7;
457     }
458
459     message RefreshRate {
460       // Specifies the base interval between refreshes. This parameter is required and must be greater
461       // than zero and less than
462       // :ref:`max_interval <envoy_api_field_Cluster.RefreshRate.max_interval>`.
463       google.protobuf.Duration base_interval = 1 [(validate.rules).duration = {
464         required: true
465         gt {nanos: 1000000}
466       }];
467
468       // Specifies the maximum interval between refreshes. This parameter is optional, but must be
469       // greater than or equal to the
470       // :ref:`base_interval <envoy_api_field_Cluster.RefreshRate.base_interval>`  if set. The default
```

```
    // is 10 times the :ref:`base_interval <envoy_api_field_Cluster.RefreshRate.base_interval>`.
    google.protobuf.Duration max_interval = 2 [(validate.rules).duration = {gt {nanos: 1000000}}];
  }

  reserved 12, 15;

  // Configuration to use different transport sockets for different endpoints.
  // The entry of *envoy.transport_socket_match* in the
  // :ref:`LbEndpoint.Metadata <envoy_api_field_endpoint.LbEndpoint.metadata>`
  // is used to match against the transport sockets as they appear in the list. The first
  // :ref:`match <envoy_api_msg_Cluster.TransportSocketMatch>` is used.
  // For example, with the following match
  //
  // .. code-block:: yaml
  //
  //  transport_socket_matches:
  //  - name: "enableMTLS"
  //    match:
  //      acceptMTLS: true
  //    transport_socket:
  //      name: envoy.transport_sockets.tls
  //      config: { ... } # tls socket configuration
  //  - name: "defaultToPlaintext"
  //    match: {}
  //    transport_socket:
  //      name: envoy.transport_sockets.raw_buffer
  //
  // Connections to the endpoints whose metadata value under *envoy.transport_socket_match*
  // having "acceptMTLS"/"true" key/value pair use the "enableMTLS" socket configuration.
  //
  // If a :ref:`socket match <envoy_api_msg_Cluster.TransportSocketMatch>` with empty match
  // criteria is provided, that always match any endpoint. For example, the "defaultToPlaintext"
  // socket match in case above.
  //
  // If an endpoint metadata's value under *envoy.transport_socket_match* does not match any
  // *TransportSocketMatch*, socket configuration fallbacks to use the *tls_context* or
  // *transport_socket* specified in this cluster.
  //
  // This field allows gradual and flexible transport socket configuration changes.
  //
  // The metadata of endpoints in EDS can indicate transport socket capabilities. For example,
  // an endpoint's metadata can have two key value pairs as "acceptMTLS": "true",
  // "acceptPlaintext": "true". While some other endpoints, only accepting plaintext traffic
  // has "acceptPlaintext": "true" metadata information.
  //
  // Then the xDS server can configure the CDS to a client, Envoy A, to send mutual TLS
  // traffic for endpoints with "acceptMTLS": "true", by adding a corresponding
  // *TransportSocketMatch* in this field. Other client Envoys receive CDS without
  // *transport_socket_match* set, and still send plain text traffic to the same cluster.
  //
  // [#comment:TODO(incfly): add a detailed architecture doc on intended usage.]
  repeated TransportSocketMatch transport_socket_matches = 43;

  // Supplies the name of the cluster which must be unique across all clusters.
  // The cluster name is used when emitting
  // :ref:`statistics <config_cluster_manager_cluster_stats>` if :ref:`alt_stat_name
  // <envoy_api_field_Cluster.alt_stat_name>` is not provided.
  // Any ``:`` in the cluster name will be converted to ``_`` when emitting statistics.
  string name = 1 [(validate.rules).string = {min_bytes: 1}];

  // An optional alternative to the cluster name to be used while emitting stats.
  // Any ``:`` in the name will be converted to ``_`` when emitting statistics. This should not be
  // confused with :ref:`Router Filter Header
  // <config_http_filters_router_x-envoy-upstream-alt-stat-name>`.
  string alt_stat_name = 28;

  oneof cluster_discovery_type {
    // The :ref:`service discovery type <arch_overview_service_discovery_types>`
    // to use for resolving the cluster.
    DiscoveryType type = 2 [(validate.rules).enum = {defined_only: true}];

    // The custom cluster type.
    CustomClusterType cluster_type = 38;
  }

  // Configuration to use for EDS updates for the Cluster.
  EdsClusterConfig eds_cluster_config = 3;

  // The timeout for new network connections to hosts in the cluster.
  google.protobuf.Duration connect_timeout = 4 [(validate.rules).duration = {gt {}}];

  // Soft limit on size of the cluster's connections read and write buffers. If
  // unspecified, an implementation defined default is applied (1MiB).
  google.protobuf.UInt32Value per_connection_buffer_limit_bytes = 5;

  // The :ref:`load balancer type <arch_overview_load_balancing_types>` to use
  // when picking a host in the cluster.
  LbPolicy lb_policy = 6 [(validate.rules).enum = {defined_only: true}];

  // If the service discovery type is
  // :ref:`STATIC<envoy_api_enum_value_Cluster.DiscoveryType.STATIC>`,
  // :ref:`STRICT_DNS<envoy_api_enum_value_Cluster.DiscoveryType.STRICT_DNS>`
  // or :ref:`LOGICAL_DNS<envoy_api_enum_value_Cluster.DiscoveryType.LOGICAL_DNS>`,
  // then hosts is required.
  //
  // .. attention::
  //
  //   **This field is deprecated**. Set the
```

```
  //   :ref:`load_assignment<envoy_api_field_Cluster.load_assignment>` field instead.
  //
  repeated core.Address hosts = 7 [deprecated = true];

  // Setting this is required for specifying members of
  // :ref:`STATIC<envoy_api_enum_value_Cluster.DiscoveryType.STATIC>`,
  // :ref:`STRICT_DNS<envoy_api_enum_value_Cluster.DiscoveryType.STRICT_DNS>`
  // or :ref:`LOGICAL_DNS<envoy_api_enum_value_Cluster.DiscoveryType.LOGICAL_DNS>` clusters.
  // This field supersedes the *hosts* field in the v2 API.
  //
  // .. attention::
  //
  //   Setting this allows non-EDS cluster types to contain embedded EDS equivalent
  //   :ref:`endpoint assignments<envoy_api_msg_ClusterLoadAssignment>`.
  //
  ClusterLoadAssignment load_assignment = 33;

  // Optional :ref:`active health checking <arch_overview_health_checking>`
  // configuration for the cluster. If no
  // configuration is specified no health checking will be done and all cluster
  // members will be considered healthy at all times.
  repeated core.HealthCheck health_checks = 8;

  // Optional maximum requests for a single upstream connection. This parameter
  // is respected by both the HTTP/1.1 and HTTP/2 connection pool
  // implementations. If not specified, there is no limit. Setting this
  // parameter to 1 will effectively disable keep alive.
  google.protobuf.UInt32Value max_requests_per_connection = 9;

  // Optional :ref:`circuit breaking <arch_overview_circuit_break>` for the cluster.
  cluster.CircuitBreakers circuit_breakers = 10;

  // The TLS configuration for connections to the upstream cluster.
  //
  // .. attention::
  //
  //   **This field is deprecated**. Use `transport_socket` with name `tls` instead. If both are
  //   set, `transport_socket` takes priority.
  auth.UpstreamTlsContext tls_context = 11
      [deprecated = true, (envoy.annotations.disallowed_by_default) = true];

  // HTTP protocol options that are applied only to upstream HTTP connections.
  // These options apply to all HTTP versions.
  core.UpstreamHttpProtocolOptions upstream_http_protocol_options = 46;

  // Additional options when handling HTTP requests upstream. These options will be applicable to
  // both HTTP1 and HTTP2 requests.
  core.HttpProtocolOptions common_http_protocol_options = 29;

  // Additional options when handling HTTP1 requests.
  core.Http1ProtocolOptions http_protocol_options = 13;

  // Even if default HTTP2 protocol options are desired, this field must be
  // set so that Envoy will assume that the upstream supports HTTP/2 when
  // making new HTTP connection pool connections. Currently, Envoy only
  // supports prior knowledge for upstream connections. Even if TLS is used
  // with ALPN, `http2_protocol_options` must be specified. As an aside this allows HTTP/2
  // connections to happen over plain text.
  core.Http2ProtocolOptions http2_protocol_options = 14;

  // The extension_protocol_options field is used to provide extension-specific protocol options
  // for upstream connections. The key should match the extension filter name, such as
  // "envoy.filters.network.thrift_proxy". See the extension's documentation for details on
  // specific options.
  map<string, google.protobuf.Struct> extension_protocol_options = 35
      [deprecated = true, (envoy.annotations.disallowed_by_default) = true];

  // The extension_protocol_options field is used to provide extension-specific protocol options
  // for upstream connections. The key should match the extension filter name, such as
  // "envoy.filters.network.thrift_proxy". See the extension's documentation for details on
  // specific options.
  map<string, google.protobuf.Any> typed_extension_protocol_options = 36;

  // If the DNS refresh rate is specified and the cluster type is either
  // :ref:`STRICT_DNS<envoy_api_enum_value_Cluster.DiscoveryType.STRICT_DNS>`,
  // or :ref:`LOGICAL_DNS<envoy_api_enum_value_Cluster.DiscoveryType.LOGICAL_DNS>`,
  // this value is used as the cluster's DNS refresh
  // rate. The value configured must be at least 1ms. If this setting is not specified, the
  // value defaults to 5000ms. For cluster types other than
  // :ref:`STRICT_DNS<envoy_api_enum_value_Cluster.DiscoveryType.STRICT_DNS>`
  // and :ref:`LOGICAL_DNS<envoy_api_enum_value_Cluster.DiscoveryType.LOGICAL_DNS>`
  // this setting is ignored.
  google.protobuf.Duration dns_refresh_rate = 16
      [(validate.rules).duration = {gt {nanos: 1000000}}];

  // If the DNS failure refresh rate is specified and the cluster type is either
  // :ref:`STRICT_DNS<envoy_api_enum_value_Cluster.DiscoveryType.STRICT_DNS>`,
  // or :ref:`LOGICAL_DNS<envoy_api_enum_value_Cluster.DiscoveryType.LOGICAL_DNS>`,
  // this is used as the cluster's DNS refresh rate when requests are failing. If this setting is
  // not specified, the failure refresh rate defaults to the DNS refresh rate. For cluster types
  // other than :ref:`STRICT_DNS<envoy_api_enum_value_Cluster.DiscoveryType.STRICT_DNS>` and
  // :ref:`LOGICAL_DNS<envoy_api_enum_value_Cluster.DiscoveryType.LOGICAL_DNS>` this setting is
  // ignored.
  RefreshRate dns_failure_refresh_rate = 44;

  // Optional configuration for setting cluster's DNS refresh rate. If the value is set to true,
  // cluster's DNS refresh rate will be set to resource record's TTL which comes from DNS
  // resolution.
```

```
  bool respect_dns_ttl = 39;

  // The DNS IP address resolution policy. If this setting is not specified, the
  // value defaults to
  // :ref:`AUTO<envoy_api_enum_value_Cluster.DnsLookupFamily.AUTO>`.
  DnsLookupFamily dns_lookup_family = 17 [(validate.rules).enum = {defined_only: true}];

  // If DNS resolvers are specified and the cluster type is either
  // :ref:`STRICT_DNS<envoy_api_enum_value_Cluster.DiscoveryType.STRICT_DNS>`,
  // or :ref:`LOGICAL_DNS<envoy_api_enum_value_Cluster.DiscoveryType.LOGICAL_DNS>`,
  // this value is used to specify the cluster's dns resolvers.
  // If this setting is not specified, the value defaults to the default
  // resolver, which uses /etc/resolv.conf for configuration. For cluster types
  // other than
  // :ref:`STRICT_DNS<envoy_api_enum_value_Cluster.DiscoveryType.STRICT_DNS>`
  // and :ref:`LOGICAL_DNS<envoy_api_enum_value_Cluster.DiscoveryType.LOGICAL_DNS>`
  // this setting is ignored.
  // Setting this value causes failure if the
  // ``envoy.restart_features.use_apple_api_for_dns_lookups`` runtime value is true during
  // server startup. Apple's API only allows overriding DNS resolvers via system settings.
  repeated core.Address dns_resolvers = 18;

  // [#next-major-version: Reconcile DNS options in a single message.]
  // Always use TCP queries instead of UDP queries for DNS lookups.
  // Setting this value causes failure if the
  // ``envoy.restart_features.use_apple_api_for_dns_lookups`` runtime value is true during
  // server startup. Apple' API only uses UDP for DNS resolution.
  bool use_tcp_for_dns_lookups = 45;

  // If specified, outlier detection will be enabled for this upstream cluster.
  // Each of the configuration values can be overridden via
  // :ref:`runtime values <config_cluster_manager_cluster_runtime_outlier_detection>`.
  cluster.OutlierDetection outlier_detection = 19;

  // The interval for removing stale hosts from a cluster type
  // :ref:`ORIGINAL_DST<envoy_api_enum_value_Cluster.DiscoveryType.ORIGINAL_DST>`.
  // Hosts are considered stale if they have not been used
  // as upstream destinations during this interval. New hosts are added
  // to original destination clusters on demand as new connections are
  // redirected to Envoy, causing the number of hosts in the cluster to
  // grow over time. Hosts that are not stale (they are actively used as
  // destinations) are kept in the cluster, which allows connections to
  // them remain open, saving the latency that would otherwise be spent
  // on opening new connections. If this setting is not specified, the
  // value defaults to 5000ms. For cluster types other than
  // :ref:`ORIGINAL_DST<envoy_api_enum_value_Cluster.DiscoveryType.ORIGINAL_DST>`
  // this setting is ignored.
  google.protobuf.Duration cleanup_interval = 20 [(validate.rules).duration = {gt {}}];

  // Optional configuration used to bind newly established upstream connections.
  // This overrides any bind_config specified in the bootstrap proto.
  // If the address and port are empty, no bind will be performed.
  core.BindConfig upstream_bind_config = 21;

  // Configuration for load balancing subsetting.
  LbSubsetConfig lb_subset_config = 22;

  // Optional configuration for the load balancing algorithm selected by
  // LbPolicy. Currently only
  // :ref:`RING_HASH<envoy_api_enum_value_Cluster.LbPolicy.RING_HASH>` and
  // :ref:`LEAST_REQUEST<envoy_api_enum_value_Cluster.LbPolicy.LEAST_REQUEST>`
  // has additional configuration options.
  // Specifying ring_hash_lb_config or least_request_lb_config without setting the corresponding
  // LbPolicy will generate an error at runtime.
  oneof lb_config {
    // Optional configuration for the Ring Hash load balancing policy.
    RingHashLbConfig ring_hash_lb_config = 23;

    // Optional configuration for the Original Destination load balancing policy.
    OriginalDstLbConfig original_dst_lb_config = 34;

    // Optional configuration for the LeastRequest load balancing policy.
    LeastRequestLbConfig least_request_lb_config = 37;
  }

  // Common configuration for all load balancer implementations.
  CommonLbConfig common_lb_config = 27;

  // Optional custom transport socket implementation to use for upstream connections.
  // To setup TLS, set a transport socket with name `tls` and
  // :ref:`UpstreamTlsContexts <envoy_api_msg_auth.UpstreamTlsContext>` in the `typed_config`.
  // If no transport socket configuration is specified, new connections
  // will be set up with plaintext.
  core.TransportSocket transport_socket = 24;

  // The Metadata field can be used to provide additional information about the
  // cluster. It can be used for stats, logging, and varying filter behavior.
  // Fields should use reverse DNS notation to denote which entity within Envoy
  // will need the information. For instance, if the metadata is intended for
  // the Router filter, the filter name should be specified as *envoy.filters.http.router*.
  core.Metadata metadata = 25;

  // Determines how Envoy selects the protocol used to speak to upstream hosts.
  ClusterProtocolSelection protocol_selection = 26;

  // Optional options for upstream connections.
  UpstreamConnectionOptions upstream_connection_options = 30;
```

```
  // If an upstream host becomes unhealthy (as determined by the configured health checks
  // or outlier detection), immediately close all connections to the failed host.
  //
  // .. note::
  //
  //   This is currently only supported for connections created by tcp_proxy.
  //
  // .. note::
  //
  //   The current implementation of this feature closes all connections immediately when
  //   the unhealthy status is detected. If there are a large number of connections open
  //   to an upstream host that becomes unhealthy, Envoy may spend a substantial amount of
  //   time exclusively closing these connections, and not processing any other traffic.
  bool close_connections_on_host_health_failure = 31;

  // If set to true, Envoy will ignore the health value of a host when processing its removal
  // from service discovery. This means that if active health checking is used, Envoy will *not*
  // wait for the endpoint to go unhealthy before removing it.
  bool drain_connections_on_host_removal = 32
      [(udpa.annotations.field_migrate).rename = "ignore_health_on_host_removal"];

  // An (optional) network filter chain, listed in the order the filters should be applied.
  // The chain will be applied to all outgoing connections that Envoy makes to the upstream
  // servers of this cluster.
  repeated cluster.Filter filters = 40;

  // [#not-implemented-hide:] New mechanism for LB policy configuration. Used only if the
  // :ref:`lb_policy<envoy_api_field_Cluster.lb_policy>` field has the value
  // :ref:`LOAD_BALANCING_POLICY_CONFIG<envoy_api_enum_value_Cluster.LbPolicy.LOAD_BALANCING_POLICY_CONFIG>`.
  LoadBalancingPolicy load_balancing_policy = 41;

  // [#not-implemented-hide:]
  // If present, tells the client where to send load reports via LRS. If not present, the
  // client will fall back to a client-side default, which may be either (a) don't send any
  // load reports or (b) send load reports for all clusters to a single default server
  // (which may be configured in the bootstrap file).
  //
  // Note that if multiple clusters point to the same LRS server, the client may choose to
  // create a separate stream for each cluster or it may choose to coalesce the data for
  // multiple clusters onto a single stream. Either way, the client must make sure to send
  // the data for any given cluster on no more than one stream.
  //
  // [#next-major-version: In the v3 API, we should consider restructuring this somehow,
  // maybe by allowing LRS to go on the ADS stream, or maybe by moving some of the negotiation
  // from the LRS stream here.]
  core.ConfigSource lrs_server = 42;

  // If track_timeout_budgets is true, the :ref:`timeout budget histograms
  // <config_cluster_manager_cluster_stats_timeout_budgets>` will be published for each
  // request. These show what percentage of a request's per try and global timeout was used. A value
  // of 0 would indicate that none of the timeout was used or that the timeout was infinite. A value
  // of 100 would indicate that the request took the entirety of the timeout given to it.
  bool track_timeout_budgets = 47;
}

// [#not-implemented-hide:] Extensible load balancing policy configuration.
//
// Every LB policy defined via this mechanism will be identified via a unique name using reverse
// DNS notation. If the policy needs configuration parameters, it must define a message for its
// own configuration, which will be stored in the config field. The name of the policy will tell
// clients which type of message they should expect to see in the config field.
//
// Note that there are cases where it is useful to be able to independently select LB policies
// for choosing a locality and for choosing an endpoint within that locality. For example, a
// given deployment may always use the same policy to choose the locality, but for choosing the
// endpoint within the locality, some clusters may use weighted-round-robin, while others may
// use some sort of session-based balancing.
//
// This can be accomplished via hierarchical LB policies, where the parent LB policy creates a
// child LB policy for each locality. For each request, the parent chooses the locality and then
// delegates to the child policy for that locality to choose the endpoint within the locality.
//
// To facilitate this, the config message for the top-level LB policy may include a field of
// type LoadBalancingPolicy that specifies the child policy.
message LoadBalancingPolicy {
  message Policy {
    // Required. The name of the LB policy.
    string name = 1;

    // Optional config for the LB policy.
    // No more than one of these two fields may be populated.
    google.protobuf.Struct config = 2 [deprecated = true];

    google.protobuf.Any typed_config = 3;
  }

  // Each client will iterate over the list in order and stop at the first policy that it
  // supports. This provides a mechanism for starting to use new LB policies that are not yet
  // supported by all clients.
  repeated Policy policies = 1;
}

// An extensible structure containing the address Envoy should bind to when
// establishing upstream connections.
message UpstreamBindConfig {
  // The address Envoy should bind to when establishing upstream connections.
  core.Address source_address = 1;
}
```

```
863
864 message UpstreamConnectionOptions {
865   // If set then set SO_KEEPALIVE on the socket to enable TCP Keepalives.
866   core.TcpKeepalive tcp_keepalive = 1;
867 }
```