» cloudflare_ip_ranges

Use this data source to get the IP ranges of CloudFlare edge nodes.

» Example Usage

```
data "cloudflare_ip_ranges" "cloudflare" {}

resource "google_compute_firewall" "allow_cloudflare_ingress" {
  name    = "from_cloudflare"
  network = "default"

  ingress {
    ports         = "443"
    protocol      = "tcp"
    source_ranges = ["${data.cloudflare_ip_ranges.cloudflare.cidr_blocks}"]
  }
}
```

» Attributes Reference

- `cidr_blocks` - The lexically ordered list of all CIDR blocks.
- `ipv4_cidr_blocks` - The lexically ordered list of only the IPv4 CIDR blocks.
- `ipv6_cidr_blocks` - The lexically ordered list of only the IPv6 CIDR blocks.

» cloudflare_load_balancer

Provides a Cloudflare Load Balancer resource. This sits in front of a number of defined pools of origins and provides various options for geographically-aware load balancing. Note that the load balancing feature must be enabled in your Clouflare account before you can use this resource.

» Example Usage

```
# Define a load balancer which always points to a pool we define below
# In normal usage, would have different pools set for different pops (cloudflare points-of-p
# Within each pop or region we can define multiple pools in failover order
resource "cloudflare_load_balancer" "bar" {
```

```
  zone = "example.com"
  name = "example-load-balancer"
  fallback_pool_id = "${cloudflare_load_balancer_pool.foo.id}"
  default_pool_ids = ["${cloudflare_load_balancer_pool.foo.id}"]
  description = "example load balancer using geo-balancing"
  proxied = true
  pop_pools {
    pop = "LAX"
    pool_ids = ["${cloudflare_load_balancer_pool.foo.id}"]
  }
  region_pools {
    region = "WNAM"
    pool_ids = ["${cloudflare_load_balancer_pool.foo.id}"]
  }
}

resource "cloudflare_load_balancer_pool" "foo" {
  name = "example-lb-pool"
  origins {
    name = "example-1"
    address = "192.0.2.1"
    enabled = false
  }
}
```

» Argument Reference

The following arguments are supported:

- `zone` - (Required) The zone to add the load balancer to.
- `name` - (Required) The DNS name to associate with the load balancer.
- `fallback_pool_id` - (Required) The pool ID to use when all other pools are detected as unhealthy.
- `default_pool_ids` - (Required) A list of pool IDs ordered by their failover priority. Used whenever region/pop pools are not defined.
- `description` - (Optional) Free text description.
- `ttl` - (Optional) Time to live (TTL) of this load balancer's DNS `name`. Conflicts with `proxied` - this cannot be set for proxied load balancers. Default is `30`.
- `proxied` - (Optional) Whether the hostname gets Cloudflare's origin protection. Defaults to `false`.
- `region_pools` - (Optional) A set containing mappings of region/country codes to a list of pool IDs (ordered by their failover priority) for the given region. Fields documented below.
- `pop_pools` - (Optional) A set containing mappings of Cloudflare Point-of-

Presence (PoP) identifiers to a list of pool IDs (ordered by their failover priority) for the PoP (datacenter). This feature is only available to enterprise customers. Fields documented below.

region_pools requires the following:

- `region` - (Required) A region code which must be in the list defined here. Multiple entries should not be specified with the same region.
- `pool_ids` - (Required) A list of pool IDs in failover priority to use in the given region.

pop_pools requires the following:

- `pop` - (Required) A 3-letter code for the Point-of-Presence. Allowed values can be found in the list of datacenters on the status page. Multiple entries should not be specified with the same PoP.
- `pool_ids` - (Required) A list of pool IDs in failover priority to use for traffic reaching the given PoP.

» Attributes Reference

The following attributes are exported:

- `id` - Unique identifier in the API for the load balancer.
- `zone_id` - ID associated with the specified `zone`.
- `created_on` - The RFC3339 timestamp of when the load balancer was created.
- `modified_on` - The RFC3339 timestamp of when the load balancer was last modified.

» cloudflare_load_balancer_pool

Provides a Cloudflare Load Balancer pool resource. This provides a pool of origins that can be used by a Cloudflare Load Balancer. Note that the load balancing feature must be enabled in your Clouflare account before you can use this resource.

» Example Usage

```
resource "cloudflare_load_balancer_pool" "foo" {
  name = "example-pool"
  origins {
    name = "example-1"
    address = "192.0.2.1"
    enabled = false
```

```
  }
  origins {
    name = "example-2"
    address = "192.0.2.2"
  }
  description = "example load balancer pool"
  enabled = false
  minimum_origins = 1
  notification_email = "someone@example.com"
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) A short name (tag) for the pool. Only alphanumeric characters, hyphens and underscores are allowed.
- `origins` - (Required) The list of origins within this pool. Traffic directed at this pool is balanced across all currently healthy origins, provided the pool itself is healthy. Fields documented below
- `check_regions` - (Optional) A list of regions from which to run health checks. Empty means every Cloudflare datacenter (the default).
- `description` - (Optional) Free text description.
- `enabled` - (Optional) Whether to enable (the default) this pool. Disabled pools will not receive traffic and are excluded from health checks. Disabling a pool will cause any load balancers using it to failover to the next pool (if any).
- `minimum_origins` - (Optional) The minimum number of origins that must be healthy for this pool to serve traffic. If the number of healthy origins falls below this number, the pool will be marked unhealthy and we wil failover to the next available pool. Default: 1.
- `monitor` - (Optional) The ID of the Monitor to use for health checking origins within this pool.
- `notification_email` - (Optional) The email address to send health status notifications to. This can be an individual mailbox or a mailing list.

» Attributes Reference

The following attributes are exported:

- `id` - ID for this load balancer pool.
- `created_on` - The RFC3339 timestamp of when the load balancer was created.
- `modified_on` - The RFC3339 timestamp of when the load balancer was last modified.

» cloudflare_page rule

Provides a Cloudflare page rule resource.

» Example Usage

```
# Add a page rule to the domain
resource "cloudflare_page_rule" "foobar" {
  domain = "${var.cloudflare_domain}"
  target = "sub.${self.domain}/page"
  priority = 1

  actions = {
    ssl = "flexible",
    email_obfuscation = "on",
  }
}
```

» Argument Reference

The following arguments are supported:

- `zone` - (Required) The zone to which the page rule should be added.
- `target` - (Required) The URL pattern to target with the page rule.
- `actions` - (Required) The actions taken by the page rule, options given below.
- `priority` - (Optional) The priority of the page rule among others for this target.
- `status` - (Optional) Whether the page rule is active or paused.

Action blocks support the following:

- `always_online` - (Optional) Whether this action is `"on"` or `"off"`.
- `automatic_https_rewrites` - (Optional) Whether this action is `"on"` or `"off"`.
- `browser_check` - (Optional) Whether this action is `"on"` or `"off"`.
- `email_obfuscation` - (Optional) Whether this action is `"on"` or `"off"`.
- `ip_geolocation` - (Optional) Whether this action is `"on"` or `"off"`.
- `opportunistic_encryption` - (Optional) Whether this action is `"on"` or `"off"`.
- `server_side_exclude` - (Optional) Whether this action is `"on"` or `"off"`.
- `smart_errors` - (Optional) Whether this action is `"on"` or `"off"`.
- `always_use_https` - (Optional) Boolean of whether this action is enabled. Default: false.

- `disable_apps` - (Optional) Boolean of whether this action is enabled. Default: false.
- `disable_performance` - (Optional) Boolean of whether this action is enabled. Default: false.
- `disable_security` - (Optional) Boolean of whether this action is enabled. Default: false.
- `browser_cache_ttl` - (Optional) The Time To Live for the browser cache.
- `edge_cache_ttl` - (Optional) The Time To Live for the edge cache.
- `cache_level` - (Optional) Whether to set the cache level to `"byypass"`, `"basic"`, `"simplified"`, `"aggressive"`, or `"cache_everything"`.
- `forwarding_url` - (Optional) The URL to forward to, and with what status. See below.
- `rocket_loader` - (Optional) Whether to set the rocket loader to `"off"`, `"manual"`, or `"automatic"`.
- `security_level` - (Optional) Whether to set the security level to `"essentially_off"`, `"low"`, `"medium"`, `"high"`, or `"under_attack"`.
- `ssl` - (Optional) Whether to set the SSL mode to `"off"`, `"flexible"`, `"full"`, or `"strict"`.

Forwarding URL actions support the following:

- `url` - (Required) The URL to which the page rule should forward.
- `status_code` - (Required) The status code to use for the redirection.

» Attributes Reference

The following attributes are exported:

- `id` - The page rule ID.
- `zone_id` - The ID of the zone in which the page rule will be applied.
- `target` - The URL pattern targeted by the page rule.
- `actions` - The actions applied by the page rule.
- `priority` - The priority of the page rule.
- `status` - Whether the page rule is active or paused.

» cloudflare_rate_limit

Provides a Cloudflare rate limit resource for a given zone. This can be used to limit the traffic you receive zone-wide, or matching more specific types of requests/responses.

» Example Usage

```
resource "cloudflare_rate_limit" "example" {
```

```
  zone = "${var.cloudflare_zone}"
  threshold = 2000
  period = 2
  match {
    request {
      url_pattern = "${var.cloudflare_zone}/*"
      schemes = ["HTTP", "HTTPS"]
      methods = ["GET", "POST", "PUT", "DELETE", "PATCH", "HEAD"]
    }
    response {
      statuses = [200, 201, 202, 301, 429]
      origin_traffic = false
    }
  }
  action {
    mode = "simulate"
    timeout = 43200
    response {
      content_type = "text/plain"
      body = "custom response body"
    }
  }
  disabled = false
  description = "example rate limit for a zone"
  bypass_url_patterns = ["${var.cloudflare_zone}/bypass1","${var.cloudflare_zone}/bypass2"]
}
```

» Argument Reference

The following arguments are supported:

- `zone` - (Required) The DNS zone to apply rate limiting to.
- `threshold` - (Required) The threshold that triggers the rate limit mitigations, combine with period. i.e. threshold per period (min: 2, max: 1,000,000).
- `period` - (Required) The time in seconds to count matching traffic. If the count exceeds threshold within this period the action will be performed (min: 1, max: 86,400).
- `action` - (Required) The action to be performed when the threshold of matched traffic within the period defined is exceeded.
- `match` - (Optional) Determines which traffic the rate limit counts towards the threshold. By default matches all traffic in the zone. See definition below.
- `disabled` - (Optional) Whether this ratelimit is currently disabled. Default: `false`.

- `description` - (Optional) A note that you can use to describe the reason for a rate limit. This value is sanitized and all tags are removed.
- `bypass_url_patterns` - (Optional) URLs matching the patterns specified here will be excluded from rate limiting.

The **match** block supports:

- `request` - (Optional) Matches HTTP requests (from the client to Cloudflare). See definition below.
- `response` (Optional) Matches HTTP responses before they are returned to the client from Cloudflare. If this is defined, then the entire counting of traffic occurs at this stage. This field is not required.

The **match.request** block supports:

- `methods` - (Optional) HTTP Methods, can be a subset ['POST','PUT'] or all ['*ALL*']. Default: ['*ALL*'].
- `schemes` - (Optional) HTTP Schemes, can be one ['HTTPS'], both ['HTTP','HTTPS'] or all ['*ALL*']. Default: ['*ALL*'].
- `url_pattern` - (Optional) The URL pattern to match comprised of the host and path, i.e. example.org/path. Wildcard are expanded to match applicable traffic, query strings are not matched. Use * for all traffic to your zone. Default: '*'.

The **match.response** block supports:

- `status` - (Optional) HTTP Status codes, can be one [403], many [401,403] or indicate all by not providing this value.
- `origin_traffic` - (Optional) Only count traffic that has come from your origin servers. If true, cached items that Cloudflare serve will not count towards rate limiting. Default: `true`.

The **action** block supports:

- `mode` - (Required) The type of action to perform. Allowable values are 'simulate' and 'ban'.
- `timeout` - (Required) The time in seconds as an integer to perform the mitigation action. Must be the same or greater than the period (min: 1, max:86,400).
- `response` - (Optional) Custom content-type and body to return, this overrides the custom error for the zone. This field is not required. Omission will result in default HTML error page. Definition below.

The **action.response** block supports:

- `content_type` - (Required) The content-type of the body, must be one of: 'text/plain', 'text/xml', 'application/json'.
- `body` - (Required) The body to return, the content here should conform to the content_type.

» Attributes Reference

The following attributes are exported:

- `id` - The Rate limit ID.
- `zone_id` - The DNS zone ID.

» Import

Rate limits can be imported using a composite ID formed of zone name and rate limit ID, e.g.

```
$ terraform import cloudflare_rate_limit.default example.com/ch8374ftwdghsif43
```

» cloudflare_record

Provides a Cloudflare record resource.

» Example Usage

```
# Add a record to the domain
resource "cloudflare_record" "foobar" {
  domain = "${var.cloudflare_domain}"
  name   = "terraform"
  value  = "192.168.0.11"
  type   = "A"
  ttl    = 3600
}
```

» Argument Reference

The following arguments are supported:

- `domain` - (Required) The domain to add the record to
- `name` - (Required) The name of the record
- `type` - (Required) The type of the record
- `value` - (Optional) The (string) value of the record. Either this or `data` must be specified
- `data` - (Optional) Map of attributes that constitute the record value. Primarily used for LOC and SRV record types. Either this or `value` must be specified
- `ttl` - (Optional) The TTL of the record (automatic: '1')
- `priority` - (Optional) The priority of the record

- `proxied` - (Optional) Whether the record gets Cloudflare's origin protection; defaults to `false`.

» Attributes Reference

The following attributes are exported:

- `id` - The record ID
- `hostname` - The FQDN of the record
- `proxiable` - Shows whether this record can be proxied, must be true if setting `proxied=true`
- `created_on` - The RFC3339 timestamp of when the record was created
- `modified_on` - The RFC3339 timestamp of when the record was last modified
- `metadata` - A key-value map of string metadata cloudflare associates with the record
- `zone_id` - The zone id of the record

» Import

Records can be imported using a composite ID formed of zone name and record ID, e.g.

```
$ terraform import cloudflare_record.default example.com/ch8374ftwdghsif43
```

» cloudflare_load_balancer_monitor

If you're using Cloudflare's Load Balancing to load-balance across multiple origin servers or data centers, you configure one of these Monitors to actively check the availability of those servers over HTTP(S).

» Example Usage

```
resource "cloudflare_load_balancer_monitor" "test" {
  expected_body = "alive"
  expected_codes = "2xx"
  method = "GET"
  timeout = 7
  path = "/health"
  interval = 55
  retries = 5
  description = "example load balancer"
```

```
  header {
    header = "Host"
    values = ["example.com"]
  }
}
```

» Argument Reference

The following arguments are supported:

- `expected_body` - (Required) A case-insensitive sub-string to look for in the response body. If this string is not found, the origin will be marked as unhealthy.
- `expected_codes` - (Required) The expected HTTP response code or code range of the health check. Eg `2xx`
- `method` - (Optional) The HTTP method to use for the health check. Default: "GET".
- `timeout` - (Optional) The timeout (in seconds) before marking the health check as failed. Default: 5.
- `path` - (Optional) The endpoint path to health check against. Default: "/".
- `interval` - (Optional) The interval between each health check. Shorter intervals may improve failover time, but will increase load on the origins as we check from multiple locations. Default: 60.
- `retries` - (Optional) The number of retries to attempt in case of a timeout before marking the origin as unhealthy. Retries are attempted immediately. Default: 2.
- `header` - (Optional) The HTTP request headers to send in the health check. It is recommended you set a Host header by default. The User-Agent header cannot be overridden. Fields documented below.
- `type` - (Optional) The protocol to use for the healthcheck. Currently supported protocols are 'HTTP' and 'HTTPS'. Default: "http".
- `description` - (Optional) Free text description.

header requires the following:

- `header` - (Required) The header name.
- `values` - (Required) A list of string values for the header.

» Attributes Reference

The following attributes are exported:

- `id` - Load balancer monitor ID.
- `created_on` - The RFC3339 timestamp of when the load balancer monitor was created.

- `modified_on` - The RFC3339 timestamp of when the load balancer monitor was last modified.

» cloudflare_zone_settings_override

Provides a resource which customizes CloudFlare zone settings. Note that after destroying this resource Zone Settings will be reset to their initial values.

» Example Usage

```
resource "cloudflare_zone_settings_override" "test" {
    name = "${var.cloudflare_zone}"
    settings {
        brotli = "on",
        challenge_ttl = 2700
        security_level = "high"
        opportunistic_encryption = "on"
        automatic_https_rewrites = "on"
        mirage = "on"
        waf = "on"
        minify {
            css = "on"
            js = "off"
            html = "off"
        }
        security_header {
            enabled = true
        }
    }
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) The name of the DNS zone to apply rate limiting to.
- `settings` - (Optional) Settings overrides that will be applied to the zone. If a setting is not specified the existing setting will be used. For a full list of available settings see below.

The **settings** block supports settings that may be applied to the zone. These may be on/off values, unitary fields, string values, integers or nested objects.

» On/Off Values

These can be specified as "on" or "off" string. Similar to boolean values, but here the empty string also means to use the existing value. Attributes available:

- `advanced_ddos`
- `always_online`
- `brotli`
- `browser_check`
- `cache_level`
- `development_mode`
- `origin_error_page_pass_thru`
- `sort_query_string_for_cache`
- `email_obfuscation`
- `hotlink_protection`
- `ip_geolocation`
- `ipv6`
- `websockets`
- `mirage`
- `opportunistic_encryption`
- `prefetch_preload`
- `privacy_pass`
- `response_buffering`
- `server_side_exclude`
- `tls_client_auth`
- `true_client_ip_header`
- `waf`
- `tls_1_2_only`
- `tls_1_3`
- `automatic_https_rewrites`
- `http2`
- `sha1_support`
- `always_use_https`. In some cases setting this might give the error `HTTP status 400: content "{\"success\":false,\"errors\":[{\"code\":1016,\"message\":\"An unknown error has occurred\"}],\"messages\":[],\"result\":null}"`. Regardless, the value is set correctly.
- `webp`. Note that the value specified will be ignored unless `polish` is turned on (i.e. is "lossless" or "lossy")

» String Values

- `cache_level`. Allowed values: "aggressive", "basic", "simplified".
- `polish`. Allowed values: "off", "lossless", "lossy".
- `rocket_loader`. Allowed values: "on", "off", "manual".

- `security_level`. Allowed values: "essentially_off", "low", "medium", "high", "under_attack".
- `ssl`. Allowed values: "off", "flexible", "full", "strict".
- `pseudo_ipv4`. Allowed values: "off", "add_header", "overwrite_header".
- `cname_flattening`.

» Integer Values

- `browser_cache_ttl`
- `challenge_ttl`
- `max_upload`
- `edge_cache_ttl`

» Nested Objects

- `minify`
- `mobile_redirect`
- `security_header`

The **minify** attribute supports the following fields:

- `css` (Required) "on"/"off"
- `html` (Required) "on"/"off"
- `js` (Required)"on"/"off"

The **mobile_redirect** attribute supports the following fields:

- `mobile_subdomain` (Required) String value
- `strip_uri` (Required) true/false
- `status` (Required) "on"/"off"

The **security_header** attribute supports the following fields:

- `enabled` (Optional) true/false
- `preload` (Optional) true/false
- `max_age` (Optional) Integer
- `include_subdomains` (Optional) true/false
- `nosniff` (Optional) true/false

» Attributes Reference

The following attributes are exported:

- `id` - The zone ID.
- `initial_settings` - Settings present in the zone at the time the resource is created. This will be used to restore the original settings when this

resource is destroyed. Shares the same schema as the `settings` attribute (Above).

- `intial_settings_read_at` - Time when this resource was created and the `initial_settings` were set.
- `readonly_settings` - Which of the current `settings` are not able to be set by the user. Which settings these are is determined by plan level and user permissions.
- `zone_status`. A full zone implies that DNS is hosted with Cloudflare. A partial zone is typically a partner-hosted zone or a CNAME setup.
- `zone_type`. Status of the zone. Valid values: active, pending, initializing, moved, deleted, deactivated.