DNS Provider

The DNS provider supports DNS updates (RFC 2136). Additionally, the provider can be configured with secret key based transaction authentication (RFC 2845).

Use the navigation to the left to read about the available resources.

Example Usage

```
# Configure the DNS Provider
provider "dns" {
  update {
    server        = "192.168.0.1"
    key_name      = "example.com."
    key_algorithm = "hmac-md5"
    key_secret    = "3VwZXJzZWNyZXQ="
  }
}

# Create a DNS A record set
resource "dns_a_record_set" "www" {
  # ...
}
```

Configuration Reference

`update` - (Optional) When the provider is used for DNS updates, this block is required. Structure is documented below.

The `update` block supports the following attributes:

- `server` - (Required) The IPv4 address of the DNS server to send updates to.
- `port` - (Optional) The target UDP port on the server where updates are sent to. Defaults to `53`.
- `transport` - (Optional) Transport to use for DNS queries. Valid values are `udp`, `udp4`, `udp6`, `tcp`, `tcp4`, or `tcp6`. Any UDP transport will retry automatically with the equivalent TCP transport in the event of a truncated response. Defaults to `udp`.
- `timeout` - (Optional) Timeout for DNS queries. Valid values are durations expressed as `500ms`, etc. or a plain number which is treated as whole seconds.
- `retries` - (Optional) How many times to retry on connection timeout. Defaults to `3`.
- `key_name` - (Optional) The name of the TSIG key used to sign the DNS update messages.
- `key_algorithm` - (Optional; Required if `key_name` is set) When using TSIG authentication, the algorithm to use for HMAC. Valid values are `hmac-md5`, `hmac-sha1`, `hmac-sha256` or `hmac-sha512`.
- `key_secret` - (Optional; Required if `key_name` is set) A Base64-encoded string containing the shared secret to be used for TSIG.

dns_aaaa_record_set

Use this data source to get DNS AAAA records of the host.

Example Usage

```
data "dns_aaaa_record_set" "google" {
  host = "google.com"
}

output "google_addrs" {
  value = "${join(",", data.dns_aaaa_record_set.google.addrs)}"
}
```

Argument Reference

The following arguments are supported:

- `host` - (required): Host to look up

Attributes Reference

The following attributes are exported:

- `id` - Set to `host`.
- `addrs` - A list of IP addresses. IP addresses are always sorted to avoid constant changing plans.

dns_a_record_set

Use this data source to get DNS A records of the host.

Example Usage

```
data "dns_a_record_set" "google" {
  host = "google.com"
}

output "google_addrs" {
  value = "${join(",", data.dns_a_record_set.google.addrs)}"
}
```

Argument Reference

The following arguments are supported:

- `host` - (required): Host to look up

Attributes Reference

The following attributes are exported:

- `id` - Set to `host`.
- `addrs` - A list of IP addresses. IP addresses are always sorted to avoid constant changing plans.

dns_cname_record_set

Use this data source to get DNS CNAME record set of the host.

Example Usage

```
data "dns_cname_record_set" "hashicorp" {
  host = "www.hashicorp.com"
}

output "hashi_cname" {
  value = "${data.dns_cname_record_set.hashi.cname}"
}
```

Argument Reference

The following arguments are supported:

- `host` - (required): Host to look up

Attributes Reference

The following attributes are exported:

- `id` - Set to `host`.
- `cname` - A CNAME record associated with host.

dns_mx_record_set

Use this data source to get DNS MX records for a domain.

Example Usage

```
data "dns_mx_record_set" "mail" {
  domain = "example.com."
}

output "mailserver" {
  value = "${data.dns_mx_record_set.mail.mx.0.exchange}"
}
```

Argument Reference

The following arguments are supported:

- `domain` - (Required): Domain to look up

Attributes Reference

The following attributes are exported:

- `id` - Set to `service`.
- `mx` - A list of records. They are sorted by ascending preference then alphabetically by exchange to stay consistent across runs.

dns_ns_record_set

Use this data source to get DNS ns records of the host.

Example Usage

```
data "dns_ns_record_set" "google" {
  host = "google.com"
}

output "google_nameservers" {
  value = "${join(",", data.dns_ns_record_set.google.nameservers)}"
}
```

Argument Reference

The following arguments are supported:

- `host` - (required): Host to look up

Attributes Reference

The following attributes are exported:

- `id` - Set to `host`.
- `nameservers` - A list of nameservers. Nameservers are always sorted to avoid constant changing plans.

dns_ptr_record_set

Use this data source to get DNS PTR record set of the ip address.

Example Usage

```
data "dns_ptr_record_set" "hashicorp" {
  ip_address = "8.8.8.8"
}

output "hashi_ptr" {
  value = "${data.dns_ptr_record_set.hashicorp.ptr}"
}
```

Argument Reference

The following arguments are supported:

- `ip_address` - (required): IP address to look up

Attributes Reference

The following attributes are exported:

- `id` - Set to `ip_address`.
- `ptr` - A PTR record associated with `ip_address`.

NOTE: Only the first result is taken from the query.

dns_srv_record_set

Use this data source to get DNS SRV records for a service.

Example Usage

```
data "dns_srv_record_set" "sip" {
  service = "_sip._tcp.example.com."
}

output "sipserver" {
  value = "${data.dns_srv_record_set.sip.srv.0.target}"
}
```

Argument Reference

The following arguments are supported:

- `service` - (Required): Service to look up

Attributes Reference

The following attributes are exported:

- `id` - Set to `service`.
- `srv` - A list of records. They are sorted to stay consistent across runs.

dns_txt_record_set

Use this data source to get DNS TXT record set of the host.

Example Usage

```
data "dns_txt_record_set" "hashicorp" {
  host = "www.hashicorp.com"
}

output "hashi_txt" {
  value = "${data.dns_txt_record_set.hashi.record}"
}

output "hashi_txts" {
  value = "${join(",", data.dns_txt_record_set.hashi.records)}"
}
```

Argument Reference

The following arguments are supported:

- `host` - (required): Host to look up

Attributes Reference

The following attributes are exported:

- `id` - Set to `host`.
- `record` - The first TXT record.
- `records` - A list of TXT records.

dns_aaaa_record_set

Creates a AAAA type DNS record set.

Example Usage

```
resource "dns_aaaa_record_set" "www" {
  zone = "example.com."
  name = "www"
  addresses = [
    "fdd5:e282:43b8:5303:dead:beef:cafe:babe",
    "fdd5:e282:43b8:5303:cafe:babe:dead:beef",
  ]
  ttl = 300
}
```

Argument Reference

The following arguments are supported:

- `zone` - (Required) DNS zone the record set belongs to. It must be an FQDN, that is, include the trailing dot.
- `name` - (Optional) The name of the record set. The `zone` argument will be appended to this value to create the full record path.
- `addresses` - (Required) The IPv6 addresses this record set will point to.
- `ttl` - (Optional) The TTL of the record set. Defaults to `3600`.

Attributes Reference

The following attributes are exported:

- `zone` - See Argument Reference above.
- `name` - See Argument Reference above.
- `addresses` - See Argument Reference above.
- `ttl` - See Argument Reference above.

Import

Records can be imported using the FQDN, e.g.

```
$ terraform import dns_aaaa_record_set.www www.example.com.
```

dns_a_record_set

Creates a A type DNS record set.

Example Usage

```
resource "dns_a_record_set" "www" {
  zone = "example.com."
  name = "www"
  addresses = [
    "192.168.0.1",
    "192.168.0.2",
    "192.168.0.3",
  ]
  ttl = 300
}
```

Argument Reference

The following arguments are supported:

- `zone` - (Required) DNS zone the record set belongs to. It must be an FQDN, that is, include the trailing dot.
- `name` - (Optional) The name of the record set. The `zone` argument will be appended to this value to create the full record path.
- `addresses` - (Required) The IPv4 addresses this record set will point to.
- `ttl` - (Optional) The TTL of the record set. Defaults to `3600` .

Attributes Reference

The following attributes are exported:

- `zone` - See Argument Reference above.
- `name` - See Argument Reference above.
- `addresses` - See Argument Reference above.
- `ttl` - See Argument Reference above.

Import

Records can be imported using the FQDN, e.g.

```
$ terraform import dns_a_record_set.www www.example.com.
```

dns_cname_record

Creates a CNAME type DNS record.

Example Usage

```
resource "dns_cname_record" "foo" {
  zone  = "example.com."
  name  = "foo"
  cname = "bar.example.com."
  ttl   = 300
}
```

Argument Reference

The following arguments are supported:

- `zone` - (Required) DNS zone the record belongs to. It must be an FQDN, that is, include the trailing dot.
- `name` - (Required) The name of the record. The `zone` argument will be appended to this value to create the full record path.
- `cname` - (Required) The canonical name this record will point to.
- `ttl` - (Optional) The TTL of the record set. Defaults to `3600` .

Attributes Reference

The following attributes are exported:

- `zone` - See Argument Reference above.
- `name` - See Argument Reference above.
- `cname` - See Argument Reference above.
- `ttl` - See Argument Reference above.

Import

Records can be imported using the FQDN, e.g.

```
$ terraform import dns_cname_record.foo foo.example.com.
```

dns_mx_record_set

Creates an MX type DNS record set.

Example Usage

```
resource "dns_a_record_set" "smtp" {
  zone = "example.com."
  name = "smtp"
  ttl  = 300

  addresses = [
    "192.0.2.1",
  ]
}

resource "dns_a_record_set" "backup" {
  zone = "example.com."
  name = "backup"
  ttl  = 300

  addresses = [
    "192.0.2.2",
  ]
}

resource "dns_mx_record_set" "mx" {
  zone = "example.com."
  ttl  = 300

  mx {
    preference = 10
    exchange   = "smtp.example.com."
  }

  mx {
    preference = 20
    exchange   = "backup.example.com."
  }

  depends_on = [
    "dns_a_record_set.smtp",
    "dns_a_record_set.backup",
  ]
}
```

Argument Reference

The following arguments are supported:

- `zone` - (Required) DNS zone the record set belongs to. It must be an FQDN, that is, include the trailing dot.

- `name` - (Optional) The name of the record set. The `zone` argument will be appended to this value to create the full record path.
- `mx` - (Required) Can be specified multiple times for each MX record. Each block supports fields documented below.
- `ttl` - (Optional) The TTL of the record set. Defaults to `3600`.

The `mx` block supports:

- `preference` - (Required) The preference for the record.
- `exchange` - (Required) The FQDN of the mail exchange, include the trailing dot.

Attributes Reference

The following attributes are exported:

- `zone` - See Argument Reference above.
- `name` - See Argument Reference above.
- `mx` - See Argument Reference above.
- `ttl` - See Argument Reference above.

Import

Records can be imported using the FQDN, e.g.

```
$ terraform import dns_mx_record_set.mx example.com.
```

dns_ns_record_set

Creates a NS type DNS record set.

Example Usage

```
resource "dns_ns_record_set" "www" {
  zone = "example.com."
  name = "www"
  nameservers = [
    "a.iana-servers.net.",
    "b.iana-servers.net.",
  ]
  ttl = 300
}
```

Argument Reference

The following arguments are supported:

- `zone` - (Required) DNS zone the record set belongs to. It must be an FQDN, that is, include the trailing dot.
- `name` - (Required) The name of the record set. The `zone` argument will be appended to this value to create the full record path.
- `nameservers` - (Required) The nameservers this record set will point to.
- `ttl` - (Optional) The TTL of the record set. Defaults to `3600`.

Attributes Reference

The following attributes are exported:

- `zone` - See Argument Reference above.
- `name` - See Argument Reference above.
- `nameservers` - See Argument Reference above.
- `ttl` - See Argument Reference above.

Import

Records can be imported using the FQDN, e.g.

```
$ terraform import dns_ns_record_set.www www.example.com.
```

dns_ptr_record

Creates a PTR type DNS record.

Example Usage

```
resource "dns_ptr_record" "dns-sd" {
  zone = "example.com."
  name = "r._dns-sd"
  ptr  = "example.com."
  ttl  = 300
}
```

Argument Reference

The following arguments are supported:

- `zone` - (Required) DNS zone the record belongs to. It must be an FQDN, that is, include the trailing dot.
- `name` - (Optional) The name of the record. The `zone` argument will be appended to this value to create the full record path.
- `ptr` - (Required) The canonical name this record will point to.
- `ttl` - (Optional) The TTL of the record set. Defaults to `3600` .

Attributes Reference

The following attributes are exported:

- `zone` - See Argument Reference above.
- `name` - See Argument Reference above.
- `ptr` - See Argument Reference above.
- `ttl` - See Argument Reference above.

Import

Records can be imported using the FQDN, e.g.

```
$ terraform import dns_ptr_record.dns-sd r._dns-sd.example.com.
```

dns_srv_record_set

Creates an SRV type DNS record set.

Example Usage

```
resource "dns_srv_record_set" "sip" {
  zone = "example.com."
  name = "_sip._tcp"
  srv {
    priority = 10
    weight   = 60
    target   = "bigbox.example.com."
    port     = 5060
  }
  srv {
    priority = 10
    weight   = 20
    target   = "smallbox1.example.com."
    port     = 5060
  }
  srv {
    priority = 10
    weight   = 20
    target   = "smallbox2.example.com."
    port     = 5060
  }
  ttl = 300
}
```

Argument Reference

The following arguments are supported:

- `zone` - (Required) DNS zone the record set belongs to. It must be an FQDN, that is, include the trailing dot.
- `name` - (Required) The name of the record set. The `zone` argument will be appended to this value to create the full record path.
- `srv` - (Required) Can be specified multiple times for each SRV record. Each block supports fields documented below.
- `ttl` - (Optional) The TTL of the record set. Defaults to `3600`.

The `srv` block supports:

- `priority` - (Required) The priority for the record.
- `weight` - (Required) The weight for the record.
- `target` - (Required) The FQDN of the target, include the trailing dot.
- `port` - (Required) The port for the service on the target.

Attributes Reference

The following attributes are exported:

- `zone` - See Argument Reference above.
- `name` - See Argument Reference above.
- `srv` - See Argument Reference above.
- `ttl` - See Argument Reference above.

Import

Records can be imported using the FQDN, e.g.

```
$ terraform import dns_srv_record_set.sip _sip._tcp.example.com.
```

dns_txt_record_set

Creates a TXT type DNS record set.

Example Usage

```
resource "dns_txt_record_set" "google" {
  zone = "example.com."
  txt = [
    "google-site-verification=...",
  ]
  ttl = 300
}
```

Argument Reference

The following arguments are supported:

- `zone` - (Required) DNS zone the record set belongs to. It must be an FQDN, that is, include the trailing dot.
- `name` - (Optional) The name of the record set. The `zone` argument will be appended to this value to create the full record path.
- `txt` - (Required) The text records this record set will be set to.
- `ttl` - (Optional) The TTL of the record set. Defaults to `3600`.

Attributes Reference

The following attributes are exported:

- `zone` - See Argument Reference above.
- `name` - See Argument Reference above.
- `txt` - See Argument Reference above.
- `ttl` - See Argument Reference above.

Import

Records can be imported using the FQDN, e.g.

```
$ terraform import dns_txt_record_set.google example.com.
```