# Auto-DNS Provisioning

**Version 21.1.1, 29 January 2021**

| Registered Address | Support | Sales |
|---|---|---|
| 26, Kingston Terrace, Princeton, New Jersey 08540, United States | | |
| | | +91 80 4850 5445 |
| http://www.rtbrick.com | support@rtbrick.com | sales@rtbrick.com |

# Table of Contents

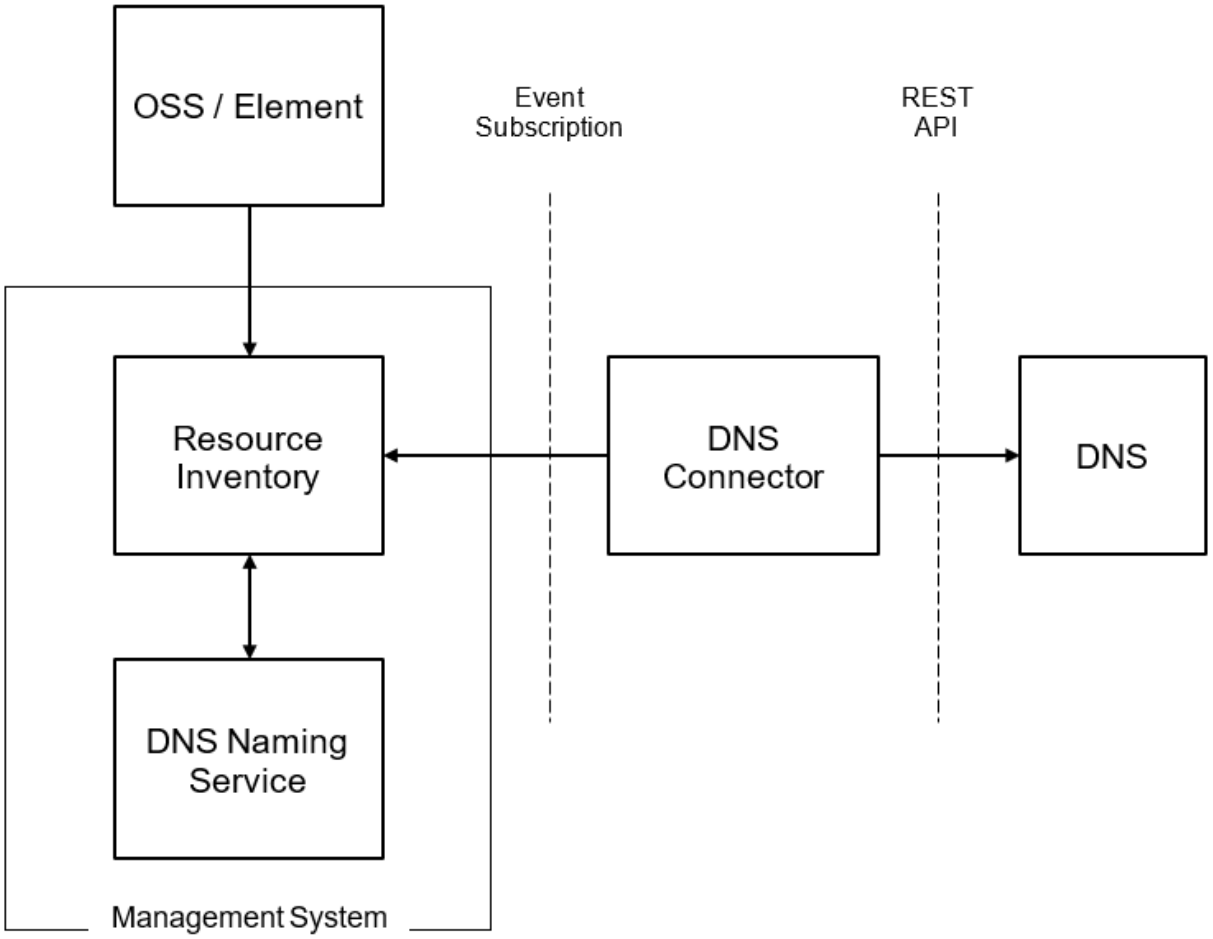
# 1. Introduction

The Domain Name Service (DNS) assigns names to IP addresses of network resources such as hosts, routers, or services. Names are easier to memorize than IP addresses, especially when intuitive naming conventions have been established. Moreover, hostnames form the common name (CN) of a certificate and are needed to issue valid certificates in order to enable Transport Layer Security (TLS).

The management system keeps track of what elements exist in the network. Consequently the management system shall be connected to the DNS such that DNS records are updated automatically whenever needed. This document discusses how to connect the management system to a DNS infrastructure.

# 2. Overview

Figure 1 gives an overview of the network management system. The resource inventory forms a cornerstone of the network management system and maintains inventory records for all elements installed in the network. Operation support systems (OSS) or the element itself feed the resource inventory with data. The resource inventory issues a domain event whenever the state of an inventory record has changed.




The DNS Naming Service subscribes all domain events that have an effect on the DNS name of an element and updates the DNS name accordingly. The DNS connector subscribes domain events related to DNS name changes and invokes the REST-API of the DNS infrastructure to update the DNS records. The focus of this document is the contract between the DNS Connector and the management system.

## 2.1. Terms and Definitions

### 2.1.1. Element

An element represents either a physical or a virtual resource in the network. For example, the switches forming the fabric are physical resources whereas a virtual

machine on a compute node is a virtual resource. DNS records are created for elements, but not necessarily every element requires a DNS record.

### 2.1.2. Element Role

The element role describes the function of an element in the network. Leaf switch, Spine switch and OLT are examples of element roles.

### 2.1.3. PoD

The Point of Distribution aggregates all subscribers of a geographic region and connects them to the core network.

### 2.1.4. Resource Inventory

The resource inventory stores all elements, including their role, configuration, resources, capabilities, operational and administrative state. It also allows to group elements.

The resource inventory is fed by planning processes, which add planned elements to the inventory, and all active elements, which register themselves in the resource inventory and report configuration or operational state changes to the resource inventory.

### 2.1.5. Domain Event

A domain event describes a change that has taken place in the resource inventory. The domain event contains the information of what has changed as well as the identifiers needed to read additional information from the resource inventory.

### 2.1.6. Operation Support System (OSS)

The Operation Support System supports to control, monitor, manage and plan the network.

### 2.1.7. Management System

The management system allows the execution of management functions on the elements forming the network and to inspect configuration and operational state of an element.

### 2.1.8. DNS Naming Service

The DNS naming service assigns DNS names to all elements in the resource inventory that are supposed to have a DNS name.

### 2.1.9. Resource Record and Resource Record Set

A resource record is the basic DNS configuration entity. The resource record set contains all resource records related to the same DNS name.

### 2.1.10. DNS Connector

The DNS connector creates resource records sets from all assigned DNS names.

# 3. DNS Record Management

## 3.1. DNS Naming Convention

A sophisticated DNS naming convention leads to intuitive and concise names and supports the use of wildcard certificates, which in turn simplifies certificate management.

The prerequisite to create a DNS naming convention is to decide which element roles and services require a DNS name.

### 3.1.1. Forward DNS Lookup

This section summarizes DNS aspects of common element management patterns briefly, without discussing the advantages and disadvantages of each management pattern in general. A sustainable DNS provisioning solution must support all common management patterns.

#### 3.1.1.1. Out-of-Band Management

Out-of-Band management establishes a dedicated management network to manage the network elements. Each element is connected via the management interface to this network. The management system also needs to be connected to the management network in order to access the network elements.

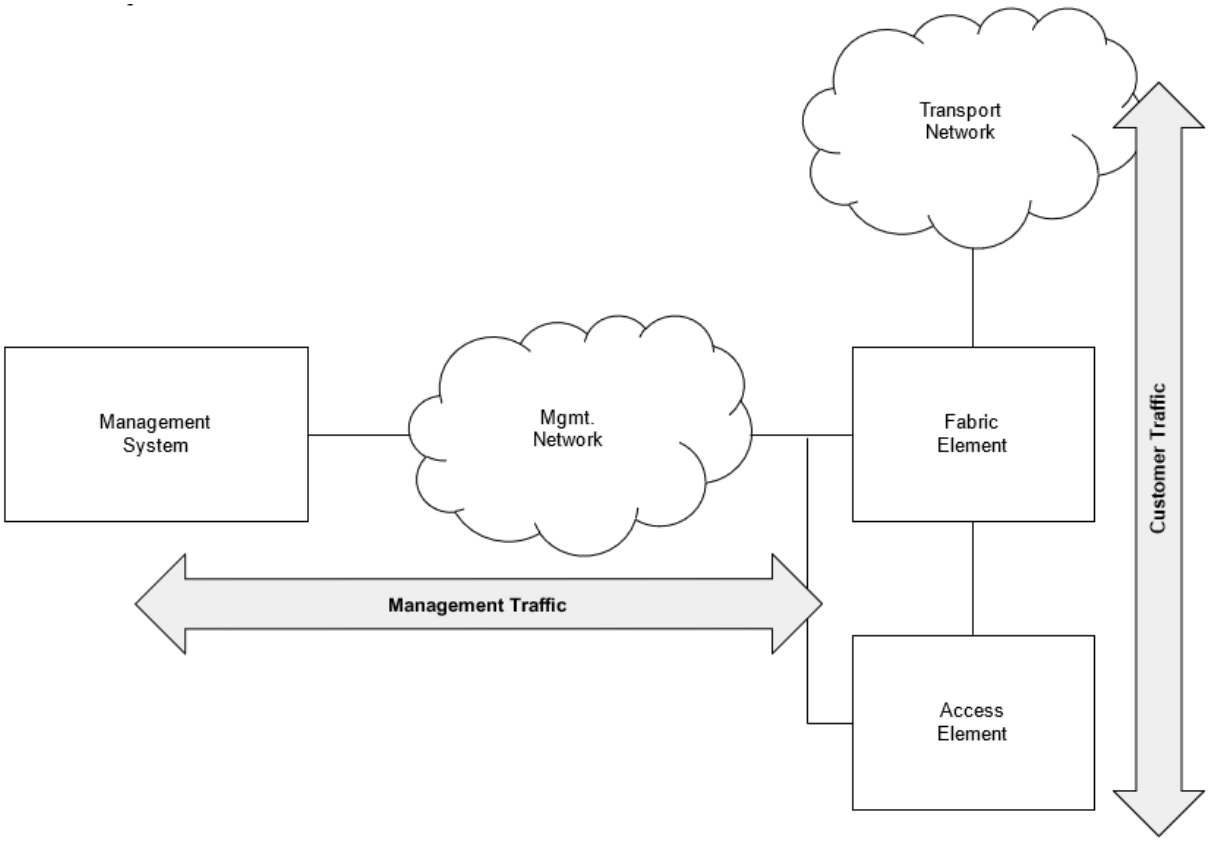



*Figure 1. Schematic illustration of the out-of-band management pattern*

The DNS lookup resolves the IP address of the management interface.

### 3.1.1.2. In-Band Management

In-band management leverages the transport network, which conveys the customer traffic, also for network management.

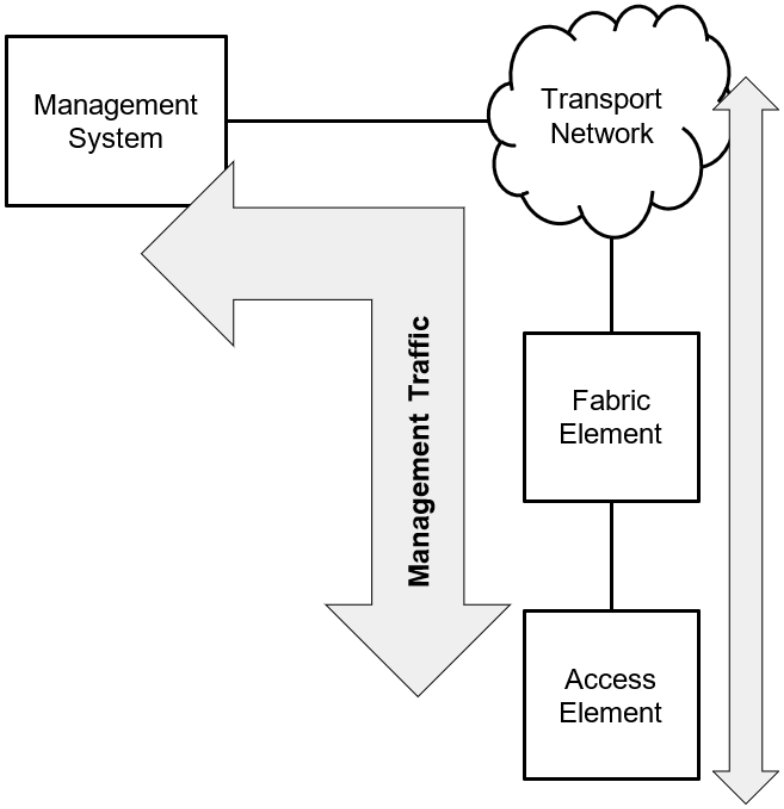



*Figure 2. Schematic illustration of the in-band management pattern*

The DNS lookup resolves the transport layer loopback IP address for all elements connected with more than one interface to the transport network. For elements with a single connection to the transport network, the IP address assigned to this interface is resolved.

### 3.1.1.3. API Gateway

An API gateway forms a single entrypoint to access multiple elements. The management system interfaces with the API gateway and the API gateway forwards the API call to the appropriate element. The API gateway is either accessed in-band or out-of-band.

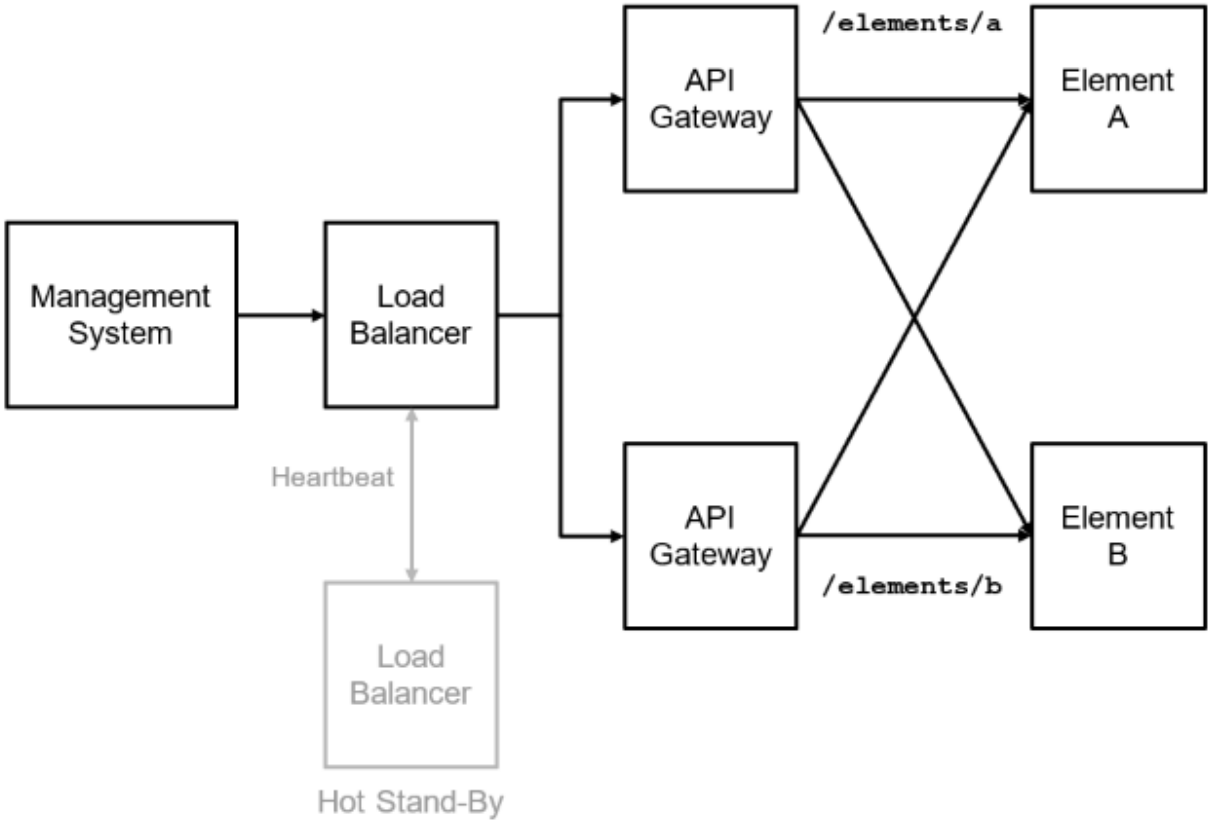



*Figure 3. Schematic illustration of the API gateway pattern*

The API gateway addresses non-functional requirements such as TLS termination, authentication and authorization.

Many instances of an API gateway are typically started at the same time to avoid creating a single point of failure. All instances are accessible through the same IP address, the cluster IP address. Consequently, the API Gateway DNS name resolves the cluster IP address. Additional DNS records per gateway instance are required to access a gateway instance for troubleshooting.

The management system needs to know the REST API endpoint URL for every element. The REST API invocation is exactly the same for all management patterns outlined before. A management process reads the REST API endpoint from the element record in the resource inventory in order to invoke the API. Looking at the API gateway pattern, an element does not have to have a DNS name despite providing a REST API endpoint.

## 3.1.2. Reverse DNS Lookup

A reverse DNS lookup discovers the hostname for a given IP address. Reverse DNS lookups are handy for troubleshooting as they translate an IP address to a human-friendly name. Consequently a hostname needs to be assigned to the transport layer loopback IP address or the IP address of the interface connected to the transport network respectively.

## 3.1.3. DNS Naming Service

The DNS naming service implements the DNS naming convention and assigns a DNS name to all elements that are supposed to have a DNS name.

DNS Naming Service is beyond the scope of this document as it depends on the DNS Naming Convention.

The resource inventory will be enhanced to store zero, one or more DNS records per element as illustrated below. The DNS Naming Service maintains the DNS records per element.

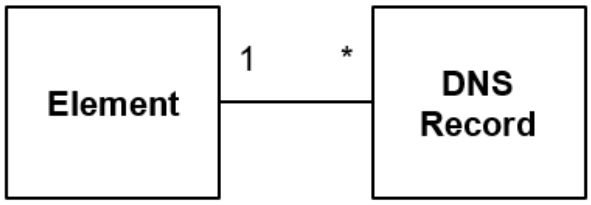



*Figure 4. Relation between element and DNS record*

A DNS record basically consists of:

- the DNS name
- the IPv4/v6 address
- a status flag indicating whether the DNS record is enabled
- the creation date and last modification date and
- an optional expiry date (defaults to never expires)

The resource inventory fires a domain event when a DNS record set has been added, modified or removed (**DnsRecordSetModifiedEvent**).

The following information is conveyed with a DNS record domain event:

- **event_id** - the event ID in UUIDv4 format.
- **event_name** - the event name (DnsRecordSetModifiedEvent).
- **group_id** - the element group ID in UUIDv4 format
- **group_type** - the element group type (set to pod)
- **group_name** - the element group name
- **element_id** - the ID, in UUIDv4 format, of the element which DNS name has been modified
- **element_name** - the name of the element which DNS name has been modified
- **element_alias** - an optional alias of the element which DNS name has been

modified. This property is omitted if no alias has been set.

- **element_role** - the role of the element which DNS name has been modified
- **group_id** - the ID of the group the element is a member of
- **group_name** - the name of the group the element is a member of
- **group_type** - the type of the group the element is a member of, which is always set to pod
- **dns_recordset** - the DNS record set
  - **dns_zone_id** - the DNS zone ID in UUIDv4 format of the DNS zone in the resource inventory.
  - **dns_zone_name** - the canoncial DNS zone name
  - **dns_name** - the canonical DNS name that shall be stored in the DNS
  - **dns_ttl** - the optional time-to-live for the DNS record (in seconds).
  - **dns_withdrawn_name** - the DNS name that shall be removed from the DNS
  - **dns_type** - the DNS record type (e.g. A, AAAA, CNAME)
  - **dns_record** - the array of DNS records
    - **dns_value** - the DNS record value (e.g. IPv4 address, IPv6 address, alias)
    - **dns_setptr** - a flag indicating whether to create a PTR record
    - **disabled** - a flag indicating whether this record is disabled

In addition, the resource inventory fires an event if a DNS zone was added (**DnsZoneCreatedEvent**) or removed (**DnsZoneRemovedEvent**). A DNS zone event merely contains the zone ID and cannonical DNS zone name:

- **event_id** - the event ID in UUIDv4 format.
- **event_name** - the event name (DnsZoneCreatedEvent or DnsZoneRemovedEvent).
- **dns_zone** - the DNS zone that was subject of the reported change
  - **dns_zone_id** - the DNS zone ID in UUIDv4 format of the DNS zone in the resource inventory.
  - **dns_zone_name** - the canoncial DNS zone name

## 3.1.4. DNS Connector

The DNS Connector subscribes the DNS domain events outlined before by means of providing a REST API endpoint that accepts **HTTP POST** requests with the domain event as request entity. The DNS connector creates and removes DNS zones according to the DNS zone events and translates the DNS record set domain

event to resource record sets using the following semantics:

- Store a resource record set in the DNS if the **dns_name** property is present. The **dns_name** property value becomes the resource record set name. The resource record set contains up to two resource records: a type A record for the IPv4 address and another type AAAA record for the IPv6 address. The connector derives the time-to-live (TTL) of the resource record set from the expiry date. The TTL is omitted if no expiry date has been specified.
- Remove a resource record set from the DNS if the **dns_withdrawn_name** property is present. The **dns_withdrawn_name** property contains the name of the resource record set to be removed.

The DNS Connector uses the REST API provided by PowerDNS, an open-source DNS server, to maintain the DNS records. PowerDNS can either be configured as a DNS server or merely acts as a gateway and forwards all DNS changes to the actual DNS service.

Figure 6 illustrates the DNS records provisioning flow.

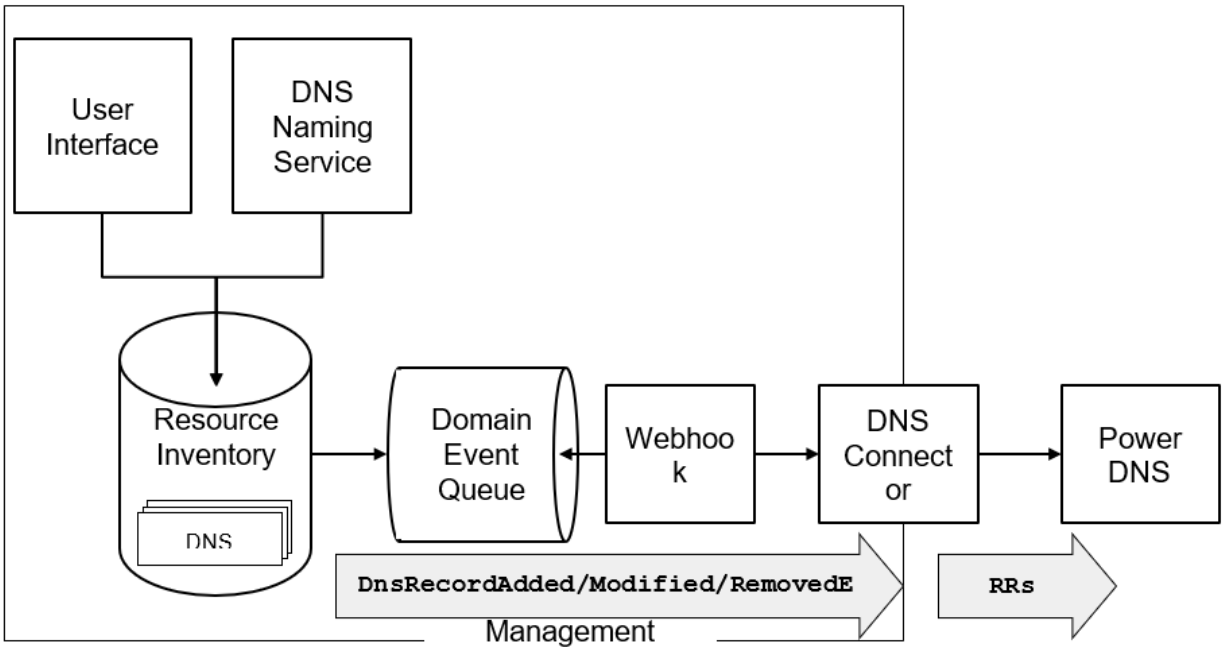



*Figure 5. DNS Resource Record Set provisioning flow*

The DNS records are stored in the resource inventory and maintained by the DNS Naming Service application or manually through the user interface. All DNS record changes fires an event, which is stored in the Domain Event Queue. The domain even queue is a persistent and transactional queue, which means that domain events are stored for successfully committed transactions only. A webhook, which is a configurable service, forwards all DNS events to the DNS connector, which in turn maintains the resource records sets (RRs) in PowerDNS.

## 3.2. Sample Fabric DNS Naming Convention

This section introduces a sample fabric DNS naming convention. The DNS name tree is depicted below.

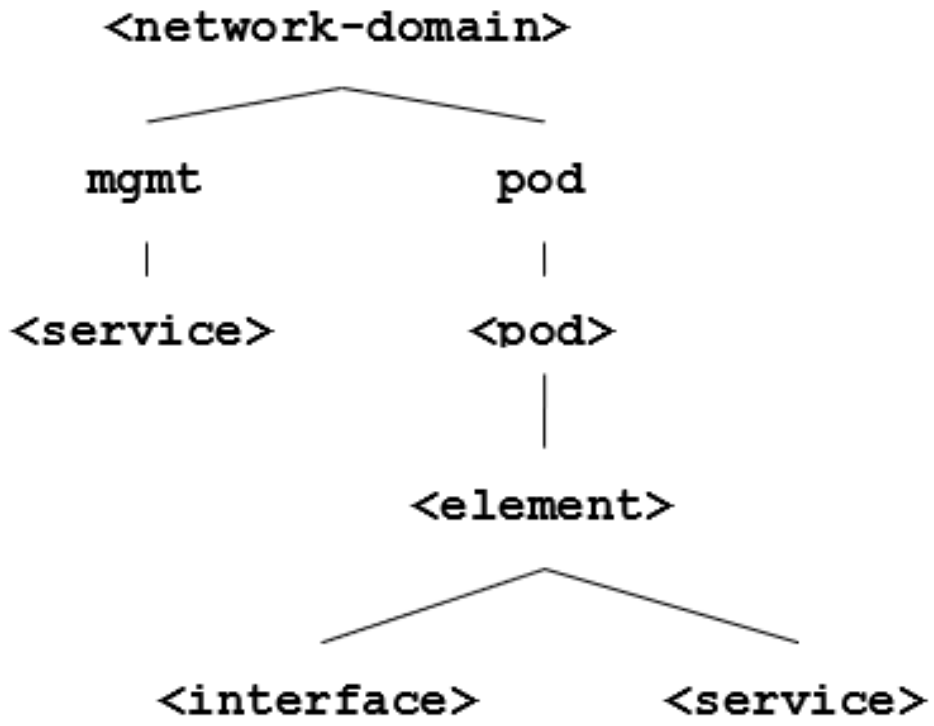



*Figure 6. Sample DNS naming convention*

The *<network-domain>* is the base domain name for the entire network (e.g. lab.rtbrick.net for the rtbrick lab environment). The next level of the DNS tree differentiates between central management services (mgmt) and pods (pod). Every management service has a designated DNS name. Every element in a pod has a designated DNS name too. Moreover, certain interfaces (for example, the management interface) or services deployed on an element might also get a DNS name assigned.

The network diagram below illustrates a lab topology that consists of two pod fabrics, Bangalore and Nuremberg, each formed by four switches and managed by a fabric daemon as well as a central management system. The central management system consists of three services: the control center to manage the network, the log management system to query log messages and process alerts, and the telemetry management system to process and visualize metrics. Each switch as a unique router ID, which is equal to the transport layer loopback IP address, a management interface and runs the control daemon (ctrld) to manage the switch.

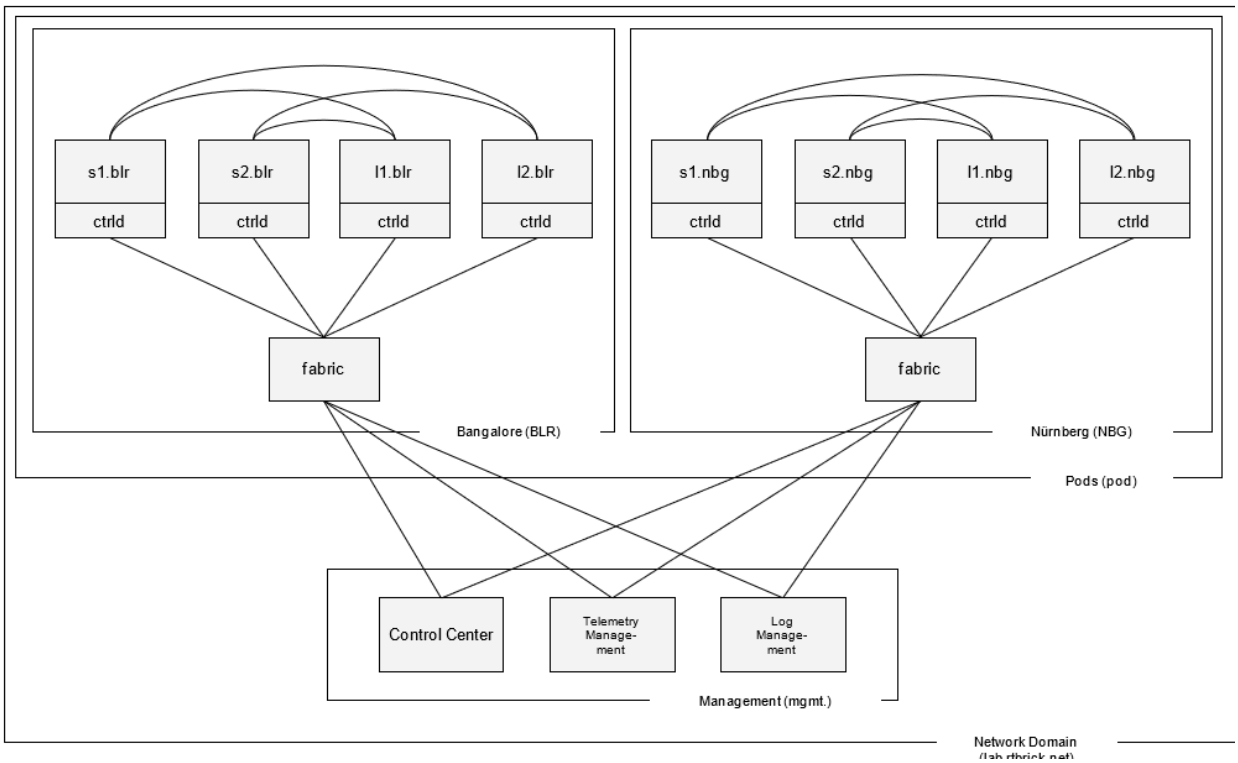



*Figure 7. Sample network topology*

The table below lists sample DNS names of elements and services in the sample network topology:

*Table 1. Sample DNS names*

| DNS Name | Description |
|---|---|
| lab.rtbrick.net | Network domain forming the top-level domain for all elements in the network |
| mgmt.lab.rtbrick.net | Subdomain for all management services in the network. |
| leitstand.lab.rtbrick.net | Control center service name. |
| log.lab.rtbrick.net | Log management service name. |
| telemetry.lab.rtbrick.net | Telemetry management service name. |
| pod.lab.rtbrick.net | Subdomain for all pods and their respective elements. |
| blr.pod.lab.rtbrick.net | Subdomain for all elements and services in pod Bangalore (BLR) |
| nbg.pod.lab.rtbrick.net | Subdomain for all elements and services in pod Nürnberg (NBG). |
| fabric.blr.pod.lab.rtbrick.net | Name of fabric in pod Bangalore (BLR) |
| l1.blr.pod.lab.rtbrick.net | Name of leaf 1 in pod Bangalore (BLR). This name resolves to the transport layer loopback IP address. |

| DNS Name | Description |
|---|---|
| me0.1.blr.pod.lab.rtbrick.net | Name of the management interface of leaf 1 in pod Bangalore (BLR). This name resolves to the IP address assigned to the management interface. |
| ctrld.l1.blr.pod.lab.rtbrick.net | An alias to access the control daemon of leaf 1 in pod Bangalore. In case of in-band management, the name is an alias of leaf1.blr.pod.lab.rtbrick.net whereas for out-of-band management the name is an alias for the management me0.leaf1.blr.pod.lab.rtbrick.net |
| All remaining leaf and spine switches have similar names. Below another example for spine 2 located in pod Nürrnberg (nbg) | |
| s2.nbg.pod.lab.rtbrick.net | Name of spine 2 in pod Nürnberg (NBG). This name resolves to the transport layer loopback IP address. |
| me0.s2.nbg.pod.lab.rtbrick.net | Name of the management interface of spine 2 in pod Nürnberg (NBG). This name resolves to the IP address assigned to the management interface. |
| ctrld.s2.nbg.pod.lab.rtbrick.net | An alias to access the control daemon of spine 2 in pod Bangalore. In case of in-band management, the name is an alias of spine2.nbg.pod.lab.rtbrick.net whereas for out-of-band management the name is an alias for the management me0.spine2.nbg.pod.lab.rtbrick.net |

The Domain Naming Service subscribes the following domain events in order to maintain the DNS names according to this naming scheme:

*Table 2. Domain Events subscribed by DNS Naming Service in order to maintain DNS records*

| Domain Event | Action |
|---|---|
| ElementAddedEvent | Create a DNS records for the added element. |
| ElementRenamedEvent | Update the DNS records for the renamed element. |
| ElementMovedEvent | Update the DNS records for the moved element, i.e. the element is now in a different pod. |

| Domain Event | Action |
|---|---|
| ElementRetiredEvent / ElementRemovedEvent | Remove the DNS records of a retired or removed element. An element must be in retired state before it can be removed from the inventory. A retired element is inactive and kept in the repository for documentation purposes only. It depends on whether the DNS record shall be part of the documentation, whether the records are removed when an element is retired or gets removed from the inventory. |
| ElementIflAddedEvent | Update the DNS record of an element if the management interface or transport layer loopback interface was added |
| ElementIflModifiedEvent | Update the DNS record of an element if the management interface or transport layer loopback interface IP address has changed |
| ElementIflRemovedEvent | Update the DNS record of an element if the management interface or transport layer loopback interface was removed |
| PodRenamedEvent | Update the DNS records for all elements in the pod. |