riverbed

SteelApp Traffic Manager Puppet Module

Reference Guide

Version 0.6.0
July 2014

Contents

PREFACE ... 3
About This Guide ... 3
Audience ... 3
Contacting Riverbed ... 3
Internet ... 3
Technical Support ... 3
Professional Services ... 3
What Is New ... 4
Chapter 1 Solution Overview ... 5
Why SteelApp Traffic Manager? ... 5
Puppet Overview ... 5
Chapter 2 Getting Started ... 6
Required Hardware and Software ... 6
Installation and Configuration ... 6
Examples ... 7
A Simple Website ... 7
Adding in SSL ... 7
Upgrading the SteelApp Traffic Manager version ... 8
Where to Store the License and Other Files? ... 8
Chapter 3 List of Defined Resources Types ... 9
new_cluster ... 9
join_cluster ... 10
web_app ... 10
pool ... 12
virtual_server ... 13
trafficipgroup ... 15
protection ... 16
bandwidth ... 16
monitor ... 17
persistence ... 19
ssl_certificate ... 19
rule ... 20
local_user ... 20
permission_group ... 21
Appendix A Additional Resources ... 23

PREFACE

Welcome to the *SteelApp Traffic Manager Puppet Module Reference Guide*. Read this preface for an overview of the information provided in this guide and contact information. This preface includes the following sections:

- About This Guide
- Contacting Riverbed
- What Is New

About This Guide

The *SteelApp Traffic Manager Puppet Module Reference Guide* describes how to use Puppet to management SteelApp Traffic Manager configuration. This guide is intended to be used together with the following documentation:

- *SteelApp Traffic Manager Software Getting Started Guide*
- *SteelApp Traffic Manager User Manual*
- *Puppet Labs Type Reference*

Audience

This guide is written for networking and application administrators and assumes familiarity with Application Delivery Controller concepts as well as familiarity with Puppet.

For more details on the SteelApp product family, see:
http://www.riverbed.com/products-solutions/products/application-delivery-stingray/

Contacting Riverbed

This section describes how to contact departments within Riverbed.

Internet

You can learn about Riverbed products through the company Web site: http://www.riverbed.com.

Technical Support

If you have problems installing, using, or replacing Riverbed products, contact Riverbed Support or your channel partner who provides support. To contact Riverbed Support, open a trouble ticket by calling 1-888-RVBD-TAC (1-888-782-3822) in the United States and Canada or +1 415 247 7381 outside the United States. You can also go to https://support.riverbed.com.

Professional Services

Riverbed has a staff of professionals who can help you with installation, provisioning, network redesign, project management, custom designs, consolidation project design, and custom coded solutions. To contact Riverbed Professional Services, email proserve@riverbed.com or go to http://www.riverbed.com/us/products/professional_services/.

What Is New

Version 0.3.0 adds the following new features (Release announcement):

- Service Protection Classes
- The ability to enable request logging on virtual servers
- The ability to enable caching and compression on virtual servers

Version 0.4.0 adds the following new features (Release announcement):

- Bandwidth Management Classes
- **Automatic removal of unused resources**. Previous versions required unused resources to be explicitly deleted with the del_web_app(), del_pool(), etc resource types. Unused resources are now automatically cleaned up.
- **Traffic IP Group improvements**. The default behavior for a Traffic IP Group is now to share the Traffic IP Group across all the SteelApp Traffic Managers in the cluster. New machines joining a cluster will be added to the Traffic IP group when the original member of the cluster (the machine that is configured with "new_cluster") pull down the configuration. . Please see "
- trafficipgroup" section in Chapter 3.

Version 0.5.0 adds the following new features (Release announcement):

- Add support for **timeout** and **connect_timeout**. These are parameters for virtual servers that specify when to timeout a connection. The timeout parameter specifies how long a connection can remain idle before being closed.
- Add support Aptimizer Express

Version 0.6.0 adds the following new features (Release announcement):

- Add support for Local Users
- Add support for Permission Groups
- Add support for Failure Pools. Failure Pools are configured like regular, and then added to a regular pool using the **failure_pool** parameter.

Chapter 1 Solution Overview

This chapter provides an overview of SteelApp and Puppet. It includes the following sections:

- Why SteelApp Traffic Manager?
- Puppet Overview

Why SteelApp Traffic Manager?

Despite increasing traffic loads, rapid change, and complex deployment infrastructures, online applications are still expected to deliver consistently excellent service levels. SteelApp traffic management solutions provide complete control over user traffic, allowing administrators to accelerate, optimize, and secure key business applications. Now it's possible to deliver these services more quickly and ensure the best possible performance across any deployment platform.

Application delivery controllers accelerate transactions, maximize availability, manage security policies, and provide a point of control to monitor and manage application traffic. SteelApp Traffic Manager is a software-based ADC that provides unprecedented scale and flexibility to deliver applications across the widest range of environments, from physical and virtual data centers to public and hybrid clouds.

SteelApp Traffic Manager benefits include:

- **Speed**: Accelerate services, increase capacity, and reduce costs by offloading performance-draining tasks such as SSL and compression onto SteelApp Traffic Manager's optimized implementations. Cache commonly requested content and optimize traffic delivery to applications so they'll run as fast as they would in a perfect benchmark environment.
- **Reliability**: Improve application availability by intelligently distributing traffic, avoiding failed or degraded servers, monitoring performance problems, and shaping traffic spikes.
- **Improved security**: SteelApp Traffic Manager operates as a deny-all gateway, only admitting traffic types it has been configured to admit. This provides full control over how traffic is internally routed. High-performance inspection can interrogate any part of a request or response to apply global filtering or scrubbing policies. The SteelApp Application Firewall option also protects against a broad range of web application attacks.
- **Ease of management**: SteelApp Traffic Manager makes it easy to manage how users interact with applications and the infrastructure those applications depend on. Use it to shape, prioritize, and route traffic, to drain infrastructure resources prior to maintenance, and to upgrade user sessions across application instances, all while preserving the user experience that business demands.

Puppet Overview

From the Puppet Labs homepage: "Puppet is IT automation software that helps system administrators manage infrastructure throughout its lifecycle, from provisioning and configuration to patch management and compliance. Using Puppet, you can easily automate repetitive tasks, quickly deploy critical applications, and proactively manage change, scaling from 10s of servers to 1000s, on-premise or in the cloud."

Puppet functions in a client/server model. The server is known as the Puppet Master and is where the SteelApp Puppet module is stored along with the entire SteelApp Traffic Manager configuration. The client is is where the SteelApp Traffic Manager software is installed. The client will have a small software agent installed on it that periodically checks in with the Puppet Master to get the latest configuration.

Chapter 2 Getting Started

This chapter discusses how to get started using Puppet to manager configuration of the SteelApp Traffic Manager. It includes the following sections:

- Required Hardware and Software
- Installation and Configuration
- Examples
- Upgrading the SteelApp Traffic Manager version
- Where to Store the License and Other Files?

Required Hardware and Software

- Puppet Enterprise or Puppet Open Source
- Two suitable Linux servers:
  - One to install SteelApp Traffic Manager software on
  - The other to function as the Puppet Master

Note: The SteelApp Traffic Manager Puppet Module uses the Linux version of the SteelApp Traffic Manager and is not compatible with the SteelApp Traffic Manager Virtual Appliance.

Installation and Configuration

The first thing to do is to follow the installation guide for either Puppet Open Source (link) or Puppet Enterprise (link). Following that, the SteelApp Puppet Module can be installed via the puppet module tool (requires version 2.7.14+). The puppet module tool is automatically installed on the Puppet Master when you install Puppet.

```
puppet module install riverbed/stingray
```

The module will typically be installed in the Puppet modules directory. You can review the modules directory by executing *puppet config print modulepath*. This is typically *~/.puppet/modules:/usr/share/puppet/modules*. The SteelApp Puppet Module will be installed in *stingray/* under the SteelApp Modules directory.

To install the SteelApp Traffic Manager, put the following in your node definition once you have read the Riverbed End User License Agreement:

```
class {‘stingray’:
    accept_license => 'accept'
}
```

There are some optional parameters as well. The full list of parameters is below:

- **‘install_dir’**: Directory to install the SteelApp software to (default: /usr/local/stingray/).
- ‘**version’**: The version of SteelApp to install (default: 9.1).
- ‘**tmp_dir’**: Temp directory to use during installation (default: /tmp).
- ‘**accept_license’**: Use of this software is subject to the terms of the Riverbed End User License Agreement. Set this to ‘accept’ once you have read the license (default: reject).

The node definition can either be directly in the site manifest file (either */etc/puppet/manifests/site.pp* or */etc/puppetlabs/puppet/manifests/site.pp*) or in a separate file imported to the site manifest file.

Once this is complete, the next step is to either create a new cluster of join an existing cluster. You can then create Virtual Servers, Pools, and Traffic IP Groups. There are a few examples below.

Examples

This section contains some examples of how to use the SteelApp Puppet module to accomplish some common tasks.

A Simple Website

The below example configures the SteelApp Traffic Manager to manage a simple website named *Northern Lights*. It creates a Pool with two back end nodes, a Virtual Server, and a Traffic IP Group. It also creates a *Transparent Session affinity* based persistence class as well as an HTTP health monitor. Both are assigned to the Pool.

```
node ‘stmtest’ {
    class {‘stingray’:
        accept_license => 'accept'
    }

    stingray::new_cluster { 'My Cluster':
    }

    stingray::web_app { 'My Web Application':
        nodes      => ['192.168.22.121:80', '192.168.22.122:80'],
        trafficips => '192.168.1.1'
    }
}
```

Adding in SSL

The above example can be expanded to decrypt SSL traffic with a few additional parameter, which creates an additional Virtual Server to take in and decrypt SSL traffic and imports the certificate that Virtual Server will use.

```
    stingray::web_app { 'My Other Web Application':
        nodes            => ['192.168.22.121:80', '192.168.22.122:80'],
        trafficips       => '192.168.1.1',
        ssl_decrypt      => 'yes'
        certificate_file => 'puppet:///modules/stingray/cert.public',
        private_key_file => 'puppet:///modules/stingray/cert.private'
    }
```

Upgrading the SteelApp Traffic Manager version

Version 9.1 of the SteelApp Traffic Manager is the default version that is installed. To upgrade to a newer version of the SteelApp Traffic Manager simply use the *version* parameter when instantiating the Stingray class.

```
class {'stingray':
    accept_license => 'accept'
    version        => '9.4'
}
```

The version of SteelApp Traffic Manager will be automatically upgraded the next time the Puppet Agent runs, which is every 30 minutes by default. You can push the configuration out earlier by executing '*puppet agent –-test*' on the SteelApp Traffic Manager node.

Where to Store the License and Other Files?

Licenses and SSL certificates should be stored on the Puppet Master. The SteelApp Puppet module contains a *files/* directory where they can be stored. The location of the file can then use the Puppet file server shorthand of *puppet:///modules/stingray/<file>*. Please note that there are three slashes.

Chapter 3 List of Defined Resources Types

SteelApp Traffic Manager functionality in Puppet is implemented though defined resource types. All SteelApp defined resource types are in the *manifests/* directory of the SteelApp Puppet Module. The sections below describe all the defined resource types that are available and how to use them.

This chapter includes the following sections:

- new_cluster
- join_cluster
- web_app
- pool
- virtual_server
- trafficipgroup

- protection
- bandwidth
- monitor
- persistence
- ssl_certificate
- rule
- local_user
- permission_group

---

new_cluster

Create a new SteelApp Traffic Manager cluster. This must be configured on exactly one node in the cluster. This node will then function as the master to the other nodes that have joined the cluster.

```
stingray::new_cluster { 'my_cluster':
}
```

admin_password
The administrator password to use. Defaults to *'password'*.

license_key
Path to the license key file. Providing no license key file, defaults to developer mode.

join_cluster

Join an existing SteelApp Traffic Manager cluster.

```
stingray::join_cluster { 'my_cluster':
     join_cluster_host => 'The other STM',
     admin_password    => 'my_password',
}
```

Note: Traffic IP Groups that are configured to be associated with all SteelApp Traffic Mangagers in the cluster will be joined when the node configured with new_cluster pulls a new configuration.

join_cluster_host
Host name for a SteelApp Traffic Manager in the cluster to join.

join_cluster_port
The admin console port for the cluster. This defaults to *'9090'*.

admin_username
The administrator username of the cluster. Defaults to *'admin'*.

admin_password
The administrator password of the cluster. Defaults to *'password'*.

web_app

Use SteelApp Traffic Manager to manage a web application.

```
stingray::web_app { 'My Web Application':
      nodes      => ['192.168.22.121:80', '192.168.22.122:80'],
      trafficips => '192.168.1.1'
}

stingray::web_app { 'My Other Web Application':
      nodes            => ['192.168.22.121:80', '192.168.22.122:80'],
      trafficips       => '192.168.1.1',
      ssl_decrypt      => 'yes'
      certificate_file => 'puppet:///modules/stingray/cert.public',
      private_key_file => 'puppet:///modules/stingray/cert.private'
 }
```

nodes
An list of the nodes in *host:port* format.

failpool_nodes
A list of the failure pool nodes in host:port format. If all of the nodes in your pool have failed, requests can be diverted to a failure pool. The default is to not use a failure pool.

weightings
Path to the license key file. Providing no license key file defaults to developer mode.

disabled
A list of the nodes in *host:port* format that are disabled. When a node is disabled, all currently established connections to that node will be terminated and no further requests will be sent to it.

draining
A list of the nodes in *host:port* format that are draining. When a node is draining, it will not receive any new connections other than those in sessions already established. To remove a node from a pool safely, it should be drained first.

algorithm
The Load Balancing algorithm to use. The default is *Round Robin*.

Valid values are:
- **'Round Robin'**: Assign requests in turn to each node.
- '**Weighted Round Robin'**: Assign requests in turn to each node, in proportion to their weights.
- **'Perceptive'**: Predict the most appropriate node using a combination of historical and current data.
- **'Least Connections'**: Assign each request to the node with the fewest connections
- **'Weighted Least Connections'**: Assign each request to a node based on the number of concurrent connections to the node and its weight.
- **'Fastest Response Time'**: Assign each request to the node with the fastest response time.
- **'Random Node'**: Choose a random node for each request.

trafficips
The Traffic IP Address associated with this web application.

machines
A list of the SteelApp Traffic Managers to associate with the trafficips.

Valid values are:
- '*' all SteelApp Traffic Managers in the cluster.
- A list of SteelApp Traffic Managers to associate with this Traffic IP Group

The default is **'*',** all SteelApp Traffic Managers in the cluster.

port
The port this web application uses. This must be a numerical value, it cannot be '*'. The default is '80'.

ssl_decrypt
Should SSL traffic be decrypted for this web applicatoin? This offloads SSL processing from your nodes, and allows the virtual server to inspect and process the connection. The default is *'no'*.

ssl_port
When ssl_decrypt is enabled, the port this web application uses for SSL traffic. This must be a numerical value, it cannot be '*'. The default is '443'.

certificate_file
When ssl_decrypt is enabled, the path to the PEM encoded certificate file

private_key_file
When ssl_decrypt is enabled, the path to the PEM encoded private key file. The Private key must not be encrypted. You can use OpenSSL to unencrypt the key:

```
openssl rsa -in key.private
```

monitor_path
For the health monitor, the path to use. This must be a string beginning with a / (forward slash). The default value is '/'.

status_regex
For the health monitor, a regular expression that the status code must match. If the status code doesn't matter then set this to .* (match anything). The default value is '^[234][0-9][0-9]$'.

body_regex
For the heatlh monitor, a regular expression that the response body must match. If the response body content doesn't matter then set this to .* (match anything). The default value is '.*'.

persistence_type
The session persistence type to use. The default is 'Transparent Session Affinity', which is also known as cookie based persistence.

banned_ips
A list of banned IPs. The entries can be of the form '10.0.1.0/255.255.255.0', '10.0.1.0/24', '10.0.1.' or '10.0.1.1'.

aptimizer_express
Aptimizer Express is an add-on module for SteelApp Traffic Manager that provides a set of robust optimizations to accelerate the delivery of most web pages, no configuration or tuning is required. This advanced capability with SteelApp Aptimizer Express is available as a licensed add-on module for SteelApp Traffic Manager 9.5 and later.

enabled
Enable this web application to begin handling traffic? The default is 'yes'.

---

pool

Create a SteelApp Traffic Manager pool. A pool manages a group of server nodes. It routes traffic to the most appropriate node, based on load balancing and session persistence criteria.

```
stingray::pool { 'My Other Pool':
       nodes      => ['192.168.22.121:80', '192.168.22.122:80'],
       weightings => {'192.168.22.121:80' => 1,
                      '192.168.22.122:80' => 2},
       algorithm  => 'Least Connections'
 }
```

nodes
An list of the nodes in *host:port* format.

weightings
Path to the license key file. Providing no license key file defaults to developer mode.

disabled
A list of the nodes in *host:port* format that are disabled. When a node is disabled, all currently established connections to that node will be terminated and no further requests will be sent to it.

draining
A list of the nodes in *host:port* format that are draining. When a node is draining, it will not receive any new connections other than those in sessions already established. To remove a node from a pool safely, it should be drained first.

monitors
A list of the monitors for this pool. A pool can have multiple monitors. Monitors watch the nodes in a pool, and inform SteelApp if the nodes are functioning correctly. SteelApp contains a number of built-in monitors. You can also create custom monitors, please see monitor.pp for more details on creating custom monitors. The default monitor for a pool is the built-in 'Ping' monitor.

algorithm
The Load Balancing algorithm to use. The default is *Round Robin*.

Valid values are:
- **'Round Robin'**: Assign requests in turn to each node.
- **'Weighted Round Robin'**: Assign requests in turn to each node, in proportion to their weights.
- **'Perceptive'**: Predict the most appropriate node using a combination of historical and current data.
- **'Least Connections'**: Assign each request to the node with the fewest connections
- **'Weighted Least Connections'**: Assign each request to a node based on the number of concurrent connections to the node and its weight.
- **'Fastest Response Time'**: Assign each request to the node with the fastest response time.
- **'Random Node'**: Choose a random node for each request.

persistence
The Session Persistence class to use for this pool. Session Persistence ensures that all requests from a client will always get sent to the same node. The default is to not use Session Persistence.

bandwidth
The bandwidth management class to use. Bandwidth classes are used to limit the network resources that a set of connections can consume. When applied to a pool, they limit the bandwidth sending data to that pool.

maxconns
The maximum number of concurrent connections allowed to each back-end node in this pool per machine. A value of 0 means unlimited connections. The default value is 0 (unlimited connections).

failure_pool
If all of the nodes in your pool have failed, requests can be diverted to a failure pool. The default is to not use a failure pool.

---

virtual_server

Create a SteelApp Traffic Manager virtual server. A virtual server accepts network traffic and processes it. It normally gives each connection to a pool; the pool then forwards the traffic to a server node.

```
stingray::virtual_server { 'My Virtual Server':
     address => '!My Traffic IP',
     pool    => 'My Pool',
     enabled => 'yes',
 }

stingray::virtual_server { 'My SSL Virtual Server':
     address         => '!My Traffic IP',
     protocol        => 'HTTP',
     port            => 443,
     pool            => 'My Pool',
     enabled         => 'yes',
     ssl_decrypt     => 'yes',
     ssl_certificate => 'My SSL Certificate'
 }
```

address
The IP Address for this virtual server to listen on.

Valid values are:
- '*' which means to listen to all IP Addresses on this host.
- A list of Traffic IP Groups prepended with an '!'. For example: address => ['!TIP 1', '!TIP 2']
- A list of IP Address and/or domain names. The virtual server will take all the traffic on its port for all domain names and IPs listed.

The default value is '*' (listen to all IP Addresses).

port
The port this virtual server listens on. This must be a numerical value, it cannot be '*'. The default is '80'.

protocol
The protocol your clients and back-end nodes use. Setting it correctly will allow protocol-specific features, such as rules that edit this protocol's headers, to work properly.

Valid values are:

'HTTP'	'Telnet'	'DNS (TCP)'
'FTP'	'SSL'	'SIP (UDP)'
'IMAPv2'	'SSL (HTTPS)'	'SIP (TCP)'
'IMAPv3'	'SSL (POP3S)'	'RTSP'
'IMAPv4'	'SSL (LDAPS)'	'Generic Server First'
'POP3'	'UDP -Streaming'	'Generic Client First'
'SMTP'	'UDP'	'Generic Streaming'
'LDAP'	'DNS (UDP)'	

If you're not sure, use 'Generic Streaming'. The default value is **'HTTP'**.

pool
The name of the pool to associate with this virtual server. The default pool is 'discard' which drops all traffic. See pool for more information on pools.

protection
The service protection class to use. Service protection is similar to an ACL that defines IP address that are banned and allowed.

enabled
Enable this virtual server to begin handling traffic? The default is *'no'*.

ssl_decrypt
Should this virtual server decrypt SSL traffic? This offloads SSL processing from your nodes, and allows the virtual server to inspect and process the connection. The default is *'no'*.

ssl_certificate
The name of the SSL certificate to use when decrypting SSL connections. See ssl_certificate for more information on importing SSL certificates for use with the SteelApp Traffic Manager.

request_rules
If a request rule is needed, the name of the rule to use. See rule section for creating a rule.

response_rules
If a response rule is needed, the name of the rule to use. See rule section for creating a rule.

enable_logging
Should this virtual server log all requests? The default is 'no'.

log_filename
If enable_logging is set to 'yes', the name of the file in which to store the request logs.

caching
If set to 'yes' the SteelApp Traffic Manager will attempt to cache web server responses. The default is 'no'.

compression
If set to 'yes' the SteelApp Traffic Manager will attempt to compress content it returns to the browser. The default is 'no'.

compression_level
If compression is enabled, the compression level (1-9, 1=low, 9=high). The default is '1'.

timeout
A connection should be closed if no additional data has been received for this period of time. A value of 0 (zero) will disable this timeout. Note that the default value may vary depending on the protocol selected.

connect_timeout
The time, in seconds, to wait for data from a new connection. If no data is received within this time, the connection will be closed. A value of 0 (zero) will disable the timeout. The default is '10'.

aptimizer_express
Aptimizer Express is an add-on module for SteelApp Traffic Manager that provides a set of robust optimizations to accelerate the delivery of most web pages, no configuration or tuning is required. This advanced capability with SteelApp Aptimizer Express is available as a licensed add-on module for SteelApp Traffic Manager 9.5 and later.

---

trafficipgroup

Create a SteelApp Traffic Manager Traffic IP Group. A traffic ip group is a set of IP addresses that will be distributed across a number of SteelApp Traffic Managers. If a SteelApp Traffic Manager fails, any IP addresses in the traffic IP group that were assigned to it will be redistributed across the remaining traffic managers. This provides fault tolerance.

```
stingray::trafficipgroup { 'My Traffic IP Group':
     ipaddress => ['192.168.1.1', '192.168.1.2'],
     machines  => ['my stm', 'my stm 2'],
     passive   => 'my stm 2',
     enabled   => 'yes'
 }
```

ipaddresses
The IP Address associated with this traffic ip group.

machines
A list of the SteelApp Traffic Managers to associate with this traffic ip group.

Valid values are:
- '*' all SteelApp Traffic Managers in the cluster.
- A list of SteelApp Traffic Managers to associate with this Traffic IP Group

The default is **'*',** all SteelApp Traffic Managers in the cluster.

passive
Of the SteelApp Traffic Managers associate with this traffic ip group, which are passive. SteelApp Traffic managers in passive

mode won't have any IP addresses assigned to them unless a failure has occurred.

keeptogether
If set to *'yes'* then all the traffic IPs will be raised on a single SteelApp Traffic Manager. The default is *'no'* which means the traffic IPs are distributed across all active SteelApp Traffic Managers in the traffic ip group.

bandwidth
The bandwidth management class to use. Bandwidth classes are used to limit the network resources that a set of connections can consume. When applied to a virtual server, they limit the bandwidth sending data to the clients.

enabled
Enable this traffic ip group and raise all the IP Addresses? The default is *'no'*.

---

protection

Creates a SteelApp Traffic manager protection class. This is like an ACL and can be applied to a virtual_server.

```
stingray::protection { 'My Protection Class':
    allowed => ['10.0.0.0/16', '192.168.1.2'],
    banned  => ['127.0.0.1'],
}
```

allowed
List of allowed IP addresses

banned
A list of banned IP addresses

---

bandwidth

Creates a SteelApp Traffic manager bandwidth management class. Bandwidth classes are used to limit the network resources that a set of connections can consume. When applied to a
pool, they limit the bandwidth sending data to that pool. When applied to a virtual_server, they limit the bandwidth sending data to the clients..

```
stingray::bandwidth { 'My Bandwidth Class':
    maximum => '10000',
}
```

maximum
The maximum bandwidth to allocate to connections that are associated with this bandwidth class (in kbits/second).

sharing
The scope of the bandwidth class.

Valid valuies are:
- **connection:** Each connection can use the maximum rate
- **machine:** Bandwidth is shared per traffic manager
- **cluster:** Bandwidth is shared across all traffic managers

The default value is **'cluster'**.

monitor

Create a SteelApp Traffic Manager monitor class. Monitors watch the nodes in a pool, and inform SteelApp if the nodes are functioning correctly. They work by sending small requests that expect a set reply. If they don't receive the reply they want, the test fails and SteelApp is notified.

```
stingray::monitor { 'My Monitor':
    type    => 'Ping',
    scope   => 'Pool',
    machine => '192.168.1.1'
 }

 stingray::monitor { 'My HTTP Monitor':
    type       => 'HTTP',
    body_regex => '.*',
    path       => '/test'
 }

 stingray::monitor { 'My TCP Transaction Monitor':
    type         => 'TCP Transaction',
    write_string => 'My string',
    use_ssl      => 'yes'
 }
```

type
The base type of the monitor to create. Valid values are:

- **'Ping'**: This pings the target machine at specified intervals.
- **'TCP Connect'**: This makes a TCP connection with the target machine, to check that a server is listening on the port.
- **'HTTP'**: This sends an HTTP request to the target server, optionally using SSL, with specified parameters such as host header and the URL path to use. It searches for a status code regex in the response.
- **'TCP Transaction'**: This performs a TCP transaction with the target machine, with an optional string of data to write to the connection. It can look for a specified regex in the response.
- **'SIP'**: This sends a SIP request to the target server of a specified transport type. It searches for a regex-matching status code and body in the response.
- **'RTSP'**: This sends a RTSP request to the target server with a specified path. It searches for a regex-matching status code and body in the response.

The default value is *'Ping'*.

scope
A monitor can either monitor each node in the pool separately and disable an individual node if it fails, or it can monitor a specific machine and disable the entire pool if that machine fails. Valid values are:

- **'Node'**: Monitor each node in the pool separately
- **'Pool'**: Monitor a specific machine and disable the entire pool if that machine fails. When using this monitor, a 'machine' to monitor must be specified.

machine
When the **scope** is set to *'Pool'*, the hostname or ip address of the macine to monitor. Where relevant this should be in the form <hostname/ip>:<port>, for "ping" monitors the :<port> part must not be specified.

delay
The minimum time (in seconds) between calls to a monitor. This controls how often a monitor runs, increasing this time will slow the monitor down. The default value is *'3' seconds*.

timeout
The time (in seconds) in which a monitor must complete. If it takes longer than this, the monitor run will be classed as having failed. The default value is *'3' seconds*.

failures
The number of consecutive runs that must fail before a node is marked as failed. Once this number of failures has occurred, SteelApp will be notified and an alert message will be raised.The default value is *'3' runs*.

use_ssl
Only applicable to HTTP, TCP Transaction, SIP, and RTSP monitors. Whether or not the monitor should connect using SSL? The default is *'no'*.

status_regex
Only applicable to 'HTTP', 'SIP', and 'RTSP' monitors. A regular expression that the status code must match. If the status code doesn't matter then set this to .* (match anything). The default value is '*^[234][0-9][0-9]$*'.

body_regex
Only applicable to 'HTTP', 'SIP', and 'RTSP' monitors. A regular expression that the response body must match. If the response body content doesn't matter then set this to .* (match anything).

path
Only applicable to 'HTTP' and 'RTSP' monitors. The path to use in the test request. This must be a string beginning with a / (forward slash). The default value is '*/*'.

host_header
Only applicable to 'HTTP' monitors. The host header to use in the test HTTP request. The default value is none.

authentication
Only applicable to 'HTTP' monitors. The HTTP basic-auth <user>:<password> to use for the test HTTP request.
The default is none.

write_string
Only applicable to 'TCP Transaction' monitors. The string to write down the TCP connection.

response_regex
Only applicable to 'TCP Transaction' monitors. A regular expression to match against the response from the server.

close_string
Only applicable to 'TCP Transaction' monitors. An optional string to write to the server before closing the connection.

sip_transport
Only applicable to 'SIP' monitors. Which transport protocol the SIP monitor will use to query the server, either 'UDP' or 'TCP'? The default value is *'UDP'*

udp_accept_all
Only applicable to 'SIP' monitors. If **sip_transport** is set to UDP, should it accept responses from any IP and port? The default value is *'no'*.

persistence

Create a SteelApp Traffic Manager Session persistence class. Session persistence classes can be used to direct all requests in a client session to the same node. This may be necessary for complex applications, where an application session may be maintained over a number of separate connections. Examples of this include web-based shopping carts, and many complex UDP-based protocols.

```
stingray::persistence { 'My Persistence':
     type => 'Transparent Session Affinity'
 }

 stingray::persistence { 'My Other Persistence':
     type   => 'Monitor application cookies',
     cookie => 'My cookie'
 }
```

type
SteelApp supports a range of different session persistence methods. Valid types are:

- **'IP-based'**: Send all requests from the same source address to the same node.
- **'Universal'**: Use session persistence data supplied by a TrafficScript rule.
- **'Named Node'**: Use a node specified by a TrafficScript rule.
- **'Transparent session affinity'**: Insert cookies into the response to track sessions.
- **'Monitor application cookies'**: Monitor a specified application cookie to identify sessions.
- **'J2EE'**: Monitor Java's JSESSIONID cookie and URLs
- **'ASP'**: Monitor ASP session cookies and ASP.NET session cookies and cookie less URLs.
- **'SSL Session ID'**: Use the SSL Session ID to identify sessions (SSL pass-through only).

The default type is **'IP-based'**.

cookie
For the *'Monitor application cookies'* persistence type, the name of the cookie to monitor.

---

ssl_certificate

Import an SSL Certificate to the SteelApp Traffic Manager catalog.

```
stingray::ssl_certificate { 'My SSL Certificate':
     certificate_file => 'puppet:///modules/stingray/cert.public',
     private_key_file => 'puppet:///modules/stingray/cert.private'
 }
```

certificate_file
Path to the PEM encoded certificate file

private_key_file
Path to the PEM encoded private key file. The Private key must not be encrypted. You can use OpenSSL to unencrypt the key:

```
openssl rsa -in key.private
```

rule

Import a TrafficScript rule to the SteelApp Traffic Manager catalog.

```
stingray::rule { 'My rule':
    file => 'puppet:///modules/stingray/rule.ts'
 }
```

file
The file containing the TrafficScript rule

---

local_user

Local Users are SteelApp Admin Server user accounts managed internally by the traffic manager software.

```
stingray::local_user { 'my_user':
    password => '$1$XoqDzcQr$tGjDcW2Fm2VfdsH6zeqrz.'
}
```

status
Is this user **'Active'** or **'Suspended'**. The default value is **'Active'**.

group
Which permission group the user belongs to. See “permission_group” for more details on permission groups. Permission groups define access privileges. The default value is **'admin'**.

password
The hashed password for this user. To generate a hashed password:

```
openssl passwd -1
```

clear_pw
The password in the **password** field is a clear password. If this is set to **'Yes'** then the hash will be automatically created from the clear password. The default value is **'No'**.

salt
The salt to use when **clear_pw** is set to **'Yes'**. The default value is **'RVBD'**.

use_applet
Enable the Admin Server UI traffic monitoring applet. The default value is **'Yes'**.

applet_max_vs
The maximum number of virtual server traffic bars to show in the applet. The default value is **'5'**.

trafficscript_editor
Use the advanced TrafficScript editor when modifying rules. This adds automatic line numbering, syntax highlighting and indentation. The default value is **'Yes'**.

permission_group

Create a permission group. Permission Groups are used to restrict what users can do in the SteelApp Traffic Manager.

```
stingray::permission_group { 'My Group':
    persistence => 'full',
    rules => 'none'
}
```

application_firewall
alerting
aptimizer
audit_log
authenticators
backup
bandwidth
catalog
cloud_credentials
config_summary
connections
custom
diagnose
draining
event_log
extra_files
glb_services
global_settings
help
java
license_keys
locations
log_viewer
main_index
map
monitoring
monitors
persistence
pools
rate
reboot
request_logs
restart
rules
slm
snmp
soap_api
ssl
security
service_protection
shutdown
statd
steelhead
support
support_files

traffic_ip_groups
traffic_managers
virtual_servers
users
web_cache
wizard
You can set access rights to each page within Stingray, and the various features on that page, by setting the appropriate flag for the current group. Flag settings can be:

- **'none'** for no access at all
- **'ro'** for permission to view the data but make no changes
- **'full'** for full rights to view and change data

The default value for each page is **'ro' (read only).**

timeout
Timeout (in minutes) the login after a period of inactivity. A value of **'0'** means never time out. The default value is **'30' minutes**.

password_expire_time
Members of this group must renew their passwords after this number of days. To disable password expiry for the group set this to 0 (zero). Note that this setting applies only to local users. The default value is **'0'**.

Appendix A Additional Resources

This appendix describes resources that supplement the information in this guide.

- SteelApp Puppet module: Home for the SteelApp Puppet Module on Puppet Forge
- SteelApp Traffic Manager product page: Overview of the SteelApp Traffic Manager
- SteelApp Community: Where to go if you have questions or comments
- Puppet Labs home page: Puppet Labs home page containing documentation and other useful information

\

riverbed

Riverbed Technology, Inc.
680 Folsom Street
San Francisco, CA 94107
Tel: (415) 247-8800
www.riverbed.com

Riverbed Technology Ltd.
One Thames Valley
Wokingham Road, Level 2
Bracknell. RG42 1NG
United Kingdom
Tel: +44 1344 31 7100

Riverbed Technology Pte. Ltd.
391A Orchard Road #22-06/10
Ngee Ann City Tower A
Singapore 238873
Tel: +65 6508-7400

Riverbed Technology K.K.
Shiba-Koen Plaza Building 9F
3-6-9, Shiba, Minato-ku
Tokyo, Japan 105-0014
Tel: +81 3 5419 1990