

**CSWG- Policy as Code Documentation**

**Networking Sentinel Policies**



3.1 Ensure That the Default Network Does Not Exist in a Project

* ***Policy Name:***
  + 3.1 Ensure That the Default Network Does Not Exist in a Project.

* ***Description of Policy:***
  + The default network has automatically created firewall rules and has default configurations.
  + It is recommended to create your network and delete the default network.

* ***Terraform Provider:***

<https://registry.terraform.io/providers/hashicorp/google/latest/docs/resources/google_project> - resource block “google\_project” is used.

* ***Sentinel Policy Restriction:***
  + In this sentinel policy, “auto\_create\_network” parameter is set to “false”.

***Pass and fail cases of the above sentinel policy:***

**Pass Case:**

1. “auto\_create\_network” parameter is set to “false”.

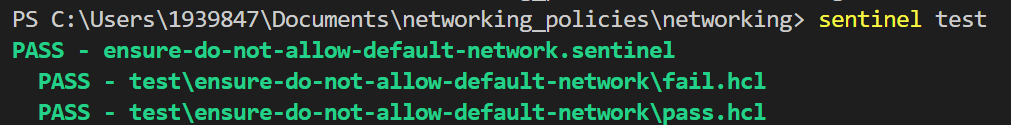


**Fail Case:**

1. “auto\_create\_network” parameter is set to “true”.



**Testcases Output**



3.2 Ensure Legacy Networks Do Not Exist for Older Projects

* ***Sentinel Policy Name:***

* 3.2 Ensure Legacy Networks Do Not Exist for Older Projects
* ***Category :***

* Networking

* ***Description of Policy:***

* Legacy networks have a single network IPv4 prefix range and a single gateway IP address for the whole network. The network is global in scope and spans all cloud regions. Subnetworks cannot be created in a legacy network and are unable to switch from legacy to auto or custom subnet networks. Legacy networks can have an impact for high network traffic projects and are subject to being a single point of failure.

* ***Sentinel Policy Restriction:***

* The policy will ensure that subnets are created and proper network segmentation is present in the network
  + For 'google\_compute\_network' ensure that 'auto\_create\_subnetworks = true'
* The policy will also ensure that cloud routers advertise routes only to devices in the same availability zone. Cloud routers should not advertise routes across all regions (globally). This is not a best practice.
  + For resource, 'google\_compute\_network' ensure that 'routing\_mode = REGIONAL'

* ***Terraform attributes:***
  + Provider Ref: [google\_compute\_network | Resources | hashicorp/google | Terraform Registry](https://registry.terraform.io/providers/hashicorp/google/latest/docs/resources/compute_network)

* ***Test cases:***

**Pass cases**

1. For 'google\_compute\_network' ensure that 'auto\_create\_subnetworks = true'
2. For resource, 'google\_compute\_network' ensure that 'routing\_mode = REGIONAL'

"after" 
uto create subnetworks". 
true , 
"delete default routes on create": false, 
"description" 
null, 
null, 
" nane": 
"project" 
"routing_mode 
"vpc- legacy" 
"agile-being- 364e17", 
" REGIONAL" 
null, 

**Fail case:**

1. For 'google\_compute\_network' the 'auto\_create\_subnetworks' is set to something other than 'true'
2. For resource, 'google\_compute\_network' the 'routing\_mode' is set to something other than 'REGIONAL'

" after" : 
"auto create subnetworks": 
"delete default routes on create": 
"description" : 
nternal_ipv6" 
" narne": 
"vpc- legacy", 
"project" 
"GLOBAL 
fal se, 
false, 
null, 
null, 
"agile-being- 364017" , 
null, 

**Testcases Output:**

PASS - sentinel 
PASS - 
logs : 
Ensure rx) legacy are present. Subnets and proper segtpntation stx»uld be present in all net»nrks, si 
ould be 'true' . Routing rode stx»uld be 'REGICNAL' (in capital letters), it is a best practice to ensure that cloud routers are advertising rot. 
te tables accross all regions 
trace : 
sentinel - Rule 
Value : 
false 
sentinel : 15:1 - Rule 
Value : 
false 
PASS - hcl 
trace: 
sentinel : 28:1 - Rule 
Value : 
sentinel : 15:1 - Rule 
Value : 
"main" 
no_l " 
"main" 
no_l 

3.3 Ensure that DNSSEC is enabled for Cloud DNS

* ***Policy Name:***
  + 3.3 Ensure that DNSSEC is enabled for Cloud DNS

* ***Category:***
  + Networking

* ***Description of Policy:***
  + In this policy, Domain Name System Security Extensions (DNSSEC) in Cloud DNS enables domain owners to take easy steps to protect their domains against DNS hijacking and man-in-the-middle and other attacks DNSSEC helps mitigate the risk of such attacks by cryptographically signing DNS records. As a result, it prevents attackers from issuing fake DNS responses that may misdirect browsers to nefarious websites.

* ***Terraform Provider:***

<https://registry.terraform.io/providers/hashicorp/google/latest/docs/resources/dns_managed_zone>  - resource block “google\_dns\_managed\_zone” is used.

<https://registry.terraform.io/providers/hashicorp/google/latest/docs/resources/dns_record_set> - resource block “google\_dns\_record\_set” is used.

* ***Sentinel Policy Restriction:***
  + Sentinel policy will make sure that DNSSEC is enabled and the state is set to on.
  + Since attackers can hijack the process of domain/IP lookup and redirect users to a malicious site through DNS hijacking and man-in-the-middle attacks.

* ***Pass and fail cases of the above sentinel policy***

**Pass case:**

1.  We will make sure that in dnssec\_config block, the state is set to “on”.

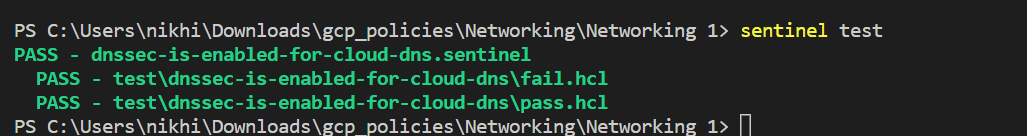


**Fail case:**

1.  In the dnssec\_config block, the state value has to be set to “off”.



**Testcases Output:**



3.4 Ensure that RSASHA1 is not used for the Key-Signing Key in Cloud DNS DNSSEC

* ***Policy Name:***
  + 3.4 Ensure that RSASHA1 is not used for the Key-Signing Key in Cloud DNS DNSSEC

* ***Category:***
  + Networking

* ***Description of Policy:***
  + In this policy, the algorithm used for key signing should be a recommended one and it should be strong. In this policy we have to make sure that RSASHA1 algorithm is not used. When enabling DNSSEC for a managed zone, or creating a managed zone with DNSSEC, the user can select the DNSSEC signing algorithms.

* ***Terraform Provider:***

<https://registry.terraform.io/providers/hashicorp/google/latest/docs/resources/dns_managed_zone> - resource block “google\_dns\_managed\_zone” is used.

* ***Sentinel Policy Restriction:***
  + Sentinel policy will make sure that when default\_key\_spec.key\_type will be keySigning then it should deny the RSASHA1 algorithm being used and also the state should be on.
  + Apart from RSASHA1, any other algorithms can be used.

* ***Pass and fail cases of the above sentinel policy***

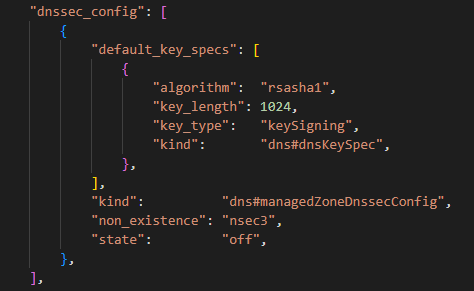
**Pass case:**

1.  RSASHA1 algorithm is not used for the key\_type keySigning.

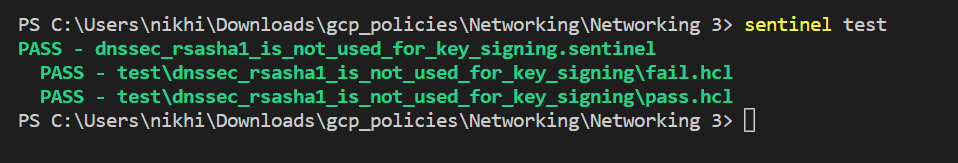


**Fail case:**

1.  When RSASHA1 algorithm is used for the key\_type keySigning.



**Testcases Output:**



3.5 Ensure that RSASHA1 is not used for the Zone-Signing Key in Cloud DNS DNSSEC

* ***Policy Name:***
  + 3.5 Ensure that RSASHA1 is not used for the Zone-Signing Key in Cloud DNS DNSSEC

* ***Category:***
  + Networking

* ***Description of Policy:***
  + In this policy, the algorithm used for zone signing should be a recommended one and it should be strong. In this policy we have to make sure that RSASHA1 algorithm is not used. When enabling DNSSEC for a managed zone, or creating a managed zone with DNSSEC, the user can select the DNSSEC signing algorithms.

* ***Terraform Provider:***

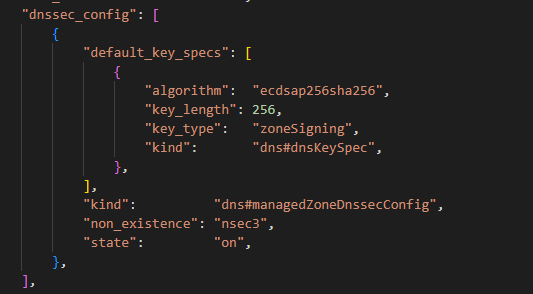
<https://registry.terraform.io/providers/hashicorp/google/latest/docs/resources/dns_managed_zone> - resource block “google\_dns\_managed\_zone” is used.

* ***Sentinel Policy Restriction:***
  + Sentinel policy will make sure that when default\_key\_spec.key\_type will be zoneSigning then it should deny the RSASHA1 algorithm being used and also the state should be on.
  + Apart from RSASHA1, any other algorithms can be used.

* ***Pass and fail cases of the above sentinel policy***

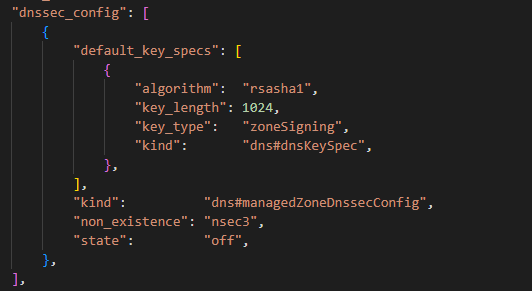
**Pass case:**

1.  RSASHA1 algorithm is not used for the key\_type zoneSigning.

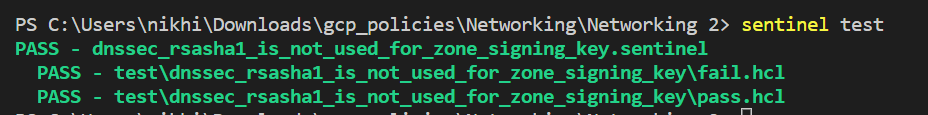


**Fail case:**

1.  When RSASHA1 algorithm is used for the key\_type zoneSigning.



**Testcases Output:**



* + 3.6 Ensure That SSH Access Is Restricted from the Internet.
  + ***Sentinel Policy Name:***

* + 3.6 Ensure That SSH Access Is Restricted from the Internet.

* + ***Category :***

* + Networking

* + ***Description of Policy:***

* + In this policy, we are creating a rule in firewall to ensure 2 things:
    - The Port number in a firewall rule is not equal to 22 (SSH) and Action is not set to Allow
    - The IP Ranges is not equal to 0.0.0.0/0 (or other public CIDR range) under Source filters.
  + ***Sentinel Policy Restriction:***

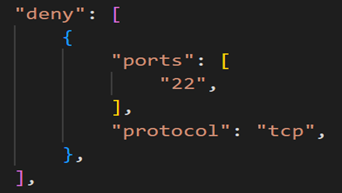
* + - If port 22 (SSH) is mentioned in the “allowed” then the source range CAN NOT be from the public (“0.0.0.0/0”, “::0”, “/0”, “0.0.0.0”).
  + ***Terraform attributes:***

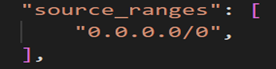
* + Provider Ref: <https://registry.terraform.io/providers/hashicorp/google/latest/docs/resources/compute_firewall>
  + Filter: 1. source\_ranges 2. Ports

* + ***Test cases:***

**Pass cases**

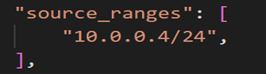
1. SSH Port 22 is denied, when the traffic is coming from the public internet (0.0.0.0/0)





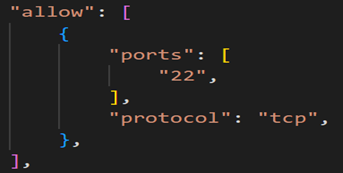
2. When the inbound traffic is not coming from the public internet, port 22 is allowed.

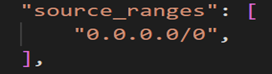




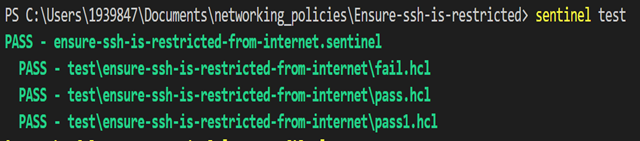
**Fail case:**

1. Allowing the port 22, when the inbound traffic is coming from the public internet





**Testcases Output:**



3.7 Ensure That RDP Access Is Restricted from the Internet

* ***Policy Name:***
  + 3.7 Ensure That RDP Access Is Restricted from the Internet

* ***Category:***
  + Networking

* ***Description of Policy:***
  + In this policy, we are creating a rule in firewall to ensure 2 things:
    - The Port number in a firewall rule is not equal to 3389 (RDP) and Action is not set to Allow
    - The IP Ranges is not equal to 0.0.0.0/0 (or other public CIDR range) under Source filters.

* ***Terraform Provider:***

<https://registry.terraform.io/providers/hashicorp/google/latest/docs/resources/compute_network> - resource block “google\_compute\_network” is used.

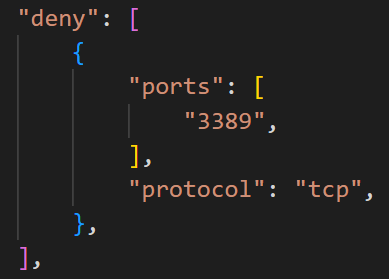
<https://registry.terraform.io/providers/hashicorp/google/latest/docs/resources/compute_firewall> - resource block “google\_compute\_firewall” is used.

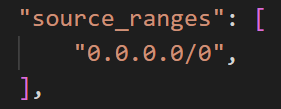
* ***Sentinel Policy Restriction:***
  + If port 3389 (RDP) is mentioned in the “allowed” then the source range CAN NOT be from the public (“0.0.0.0/0”, “::0”, “/0”, “0.0.0.0”).

* ***Pass and fail cases of the above sentinel policy***

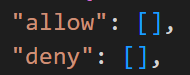
**Pass Cases:**

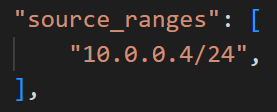
1.  RDP Port 3389 is denied, when the traffic is coming from the public internet (0.0.0.0/0).





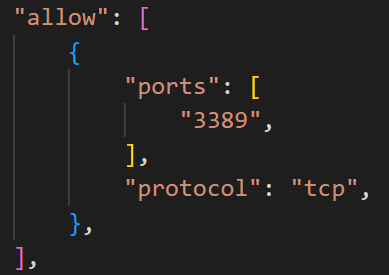
2. When the inbound traffic is not coming from the public internet, port 3389 is not restricted.

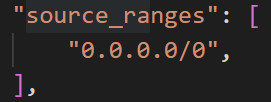




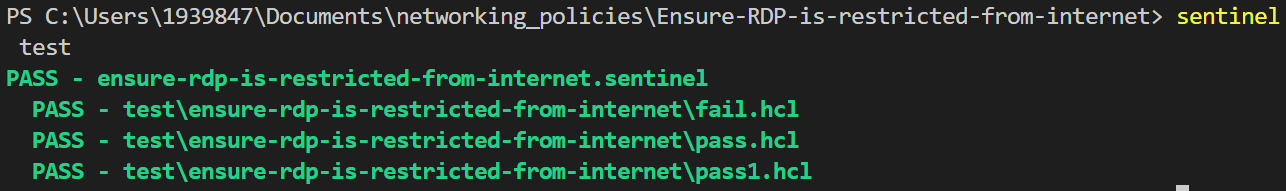
**Fail Case:**

1.  Allowing the port 3389, when the inbound traffic is coming from the public internet





**Testcases Output:**



3.8 Ensure that VPC Flow Logs is Enabled for Every Subnet in a VPC Network

* ***Policy Name:***
  + 3.8 Ensure that VPC Flow Logs is Enabled for Every Subnet in a VPC Network

* ***Category:***
  + Networking

* ***Description of Policy:***
  + Flow Logs is a feature that enables users to capture information about the IP traffic going to and from network interfaces in the VPC Subnets.
  + After creating a flow log, we can view and retrieve its data in Stackdriver Logging.
  + When Flow Logs are enabled for a subnet, VMs within that subnet start reporting on all Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) flows.

* ***Terraform Provider:***

<https://registry.terraform.io/providers/hashicorp/google/latest/docs/resources/compute_network> - resource block “google\_compute\_network” is used.

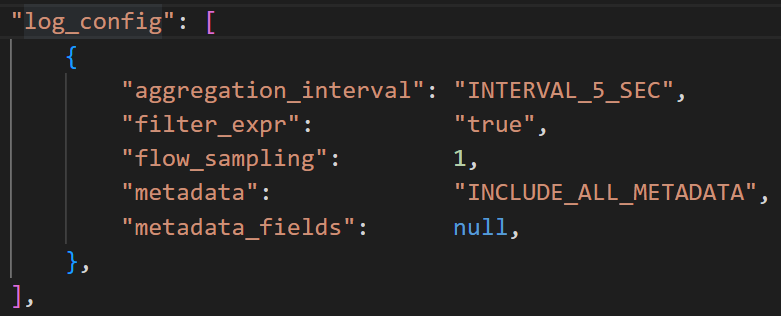
<https://registry.terraform.io/providers/hashicorp/google/latest/docs/data-sources/compute_subnetwork> - resource block “google\_compute\_subnetwork” is used.

* ***Sentinel Policy Restriction:***
  + This sentinel policy ensures that flow logs is enabled for every subnet in a VPC network.
  + **log\_config** block denotes the logging options for the subnetwork flow logs.

* ***Pass and fail cases of the above sentinel policy***

**Pass Case:**

1. When we create a subnet inside a network, it should contain log\_config block and the length of the log\_config block > 0.



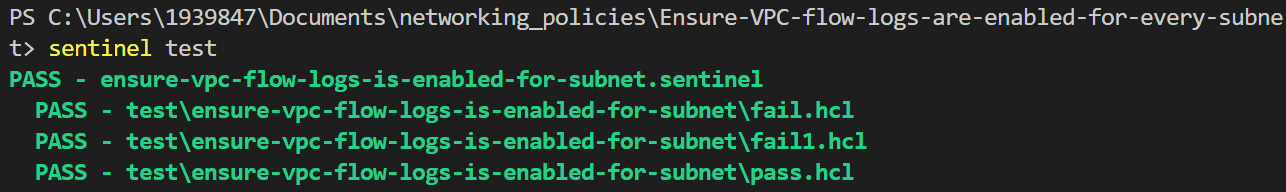
**Fail Cases:**

1. When the log\_config block is null.



2. When the subnet does not contain log\_config block.

**Testcases Output:**



* 3.9 Ensure no https or SSL proxy load balances permit SSL policies with weak ciphers
  + ***Sentinel Policy Name:***

* + - 3.9 Ensure no https or SSL proxy load balances permit ssl policies with weak ciphers
  + ***Category :***
    - Networking

* + ***Description of Policy:***

* + Secure Sockets Layer (SSL) policies determine what port Transport Layer Security (TLS) features clients are permitted to use when connecting to load balancers. To prevent usage of insecure features, SSL policies should use (a) at least TLS 1.2 with the MODERN profile; or (b) the RESTRICTED profile, because it effectively requires clients to use TLS 1.2 regardless of the chosen minimum TLS version; or (3) a CUSTOM profile that does not support any of the following features:
    - TLS\_RSA\_WITH\_AES\_128\_GCM\_SHA256
    - TLS\_RSA\_WITH\_AES\_256\_GCM\_SHA384
    - TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA
    - TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA
    - TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA

* + ***Sentinel Policy Restriction:***

The policy checks whether target HTTPS proxy or SSL proxy are created and if ssl policies named for either of the proxy are created with the following conditions:

* + at least TLS 1.2 with the MODERN profile
  + the RESTRICTED profile, because it effectively requires clients to use TLS 1.2 regardless of the chosen minimum TLS version
  + a CUSTOM profile that does not support any of the following features:
    - TLS\_RSA\_WITH\_AES\_128\_GCM\_SHA256
    - TLS\_RSA\_WITH\_AES\_256\_GCM\_SHA384
    - TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA
    - TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA
    - TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA

* + ***Terraform attributes:***
    - Provider Ref:[google\_compute\_ssl\_policy | Resources | hashicorp/google | Terraform Registry](https://registry.terraform.io/providers/hashicorp/google/latest/docs/resources/compute_ssl_policy)
      * [google\_compute\_target\_https\_proxy | Resources | hashicorp/google | Terraform Registry](https://registry.terraform.io/providers/hashicorp/google/latest/docs/resources/compute_target_https_proxy)
      * [google\_compute\_target\_ssl\_proxy | Resources | hashicorp/google | Terraform Registry](https://registry.terraform.io/providers/hashicorp/google/latest/docs/resources/compute_target_ssl_proxy)
    - Terraform attribute:
      * target\_ssl\_proxy resource
        + ssl\_policy
      * target\_https\_proxy resource
        + ssl\_policy
      * ssl\_policy resource
        + min\_tls\_version
        + Profile
        + custom\_features
    - There is also a check if the SSL policy is call in target\_https\_proxy or target\_ssl\_proxy resource for the test to run properly.

* + ***Test cases:***

**Pass cases**

1. In pass case, if the ssl policy as one of the 3 below, then the pass case will always pass:

SSL policies should be one of the following:

* + at least TLS 1.2 with the MODERN profile

Screenshot of argument in mock file

" 1 _ policy. mod 
" address" : " google_compute 
"change" • 
"actions" : 
create" 
" after 
"custom features": 
"description' . 
min t1s version 
name : 
"profile" • 
"timeouts" : 
SSI -policyl" 
ssl_policy. mod_ssl-policyl " , 
null, 
null, 
"TLS 1 2", 
"nonprod - ssl-policy" , 
"MODERN" , 
null, 

* + the RESTRICTED profile, because it effectively requires clients to use TLS 1.2 regardless of the chosen minimum TLS version

Screenshot of argument in mock file

" google_compute_ssl_policy. res 
" address" . " google_compute 
"change" • 
"actions" : 
create" 
" after 
"custom features": 
"description' . 
min t1s version 
name : 
"profile" • 
"timeouts" : 
SSI -policy2" 
ssl_policy. res_ss1-p01icy2" 
null, 
null, 
"TLS 1 0", 
'I production-ssl-policy" 
" RESTRICTED" , 
null, 

* + a CUSTOM profile that does not support any of the following features:
    - TLS\_RSA\_WITH\_AES\_128\_GCM\_SHA256
    - TLS\_RSA\_WITH\_AES\_256\_GCM\_SHA384
    - TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA
    - TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA
    - TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA

Screenshot of argument in mock file

" google_compute_ssl_policy. custom_ss1-p01icy3" • 
"address" . 
" google_compute_ssl_policy. custom_ssl -policy3 " , 
"change" • 
actions" : 
create" 
" after 
"custom_features" : [ 
"TLS ECDHE ECDSA WITH AES 256 SHA384", 
"TLS ECDHE RSA WITH AES 256 GCM SHA384", 
"description" 
min t1s version 
name : 
"profile" : 
"timeouts" : 
null, 
"TLS 1 2", 
" custom-ssl-policy" 
"CUSTOM" , 
null, 

**Fail case:**

1. In pass case, if the ssl policy is one of the 3 below, then the pass case will always fail:

SSL policies will be one of the following:

* + Not at least TLS 1.2 with the MODERN profile

Screenshot of argument in mock file

" 1 _ policy. mod 
SSI -policyl" 
"address" . 
" google_compute 
ssl_policy. mod_ssl-policyl " , 
"change" • 
actions" : 
create" 
" after 
"custom features": 
null, 
"description' . 
null, 
min t1s version 
"TLS 1 0", 
"nonprod - ssl-policy" , 
name : 
"profile" : 
"MODERN" , 
"timeouts" : 
null, 
"after_unknown" : { 
"creation_timestamp" : true, 
"enabled features" : 
true, 
"fingerprint" • 
true, 
'lid" : 
true, 
"project" • 
true, 
"self link": 
true, 

* + NOT the RESTRICTED profile, because it effectively requires clients to use TLS 1.2 regardless of the chosen minimum TLS version

Screenshot of argument in mock file

"google_compute_ssl_policy. res 
SSI -policy2" 
" address" : " google_compute 
ssl_policy. res_ss1-p01icy2" 
"change" • 
"actions" : 
create" 
" after 
"custom features": 
null, 
"description' . 
null, 
min t1s version 
"TLS 1 0", 
'I production-ssl-policy" 
name : 
"profile" • 
"COMPATIBLE" 
"timeouts" : 
null, 
"after_unknown" : { 
"creation_timestamp" : true, 
"enabled features" : 
true, 
"fingerprint" • 
true, 
'lid" : 
true, 
"project" • 
true, 
"self link": 
true, 

* + a CUSTOM profile that does support any of the following features:
    - TLS\_RSA\_WITH\_AES\_128\_GCM\_SHA256
    - TLS\_RSA\_WITH\_AES\_256\_GCM\_SHA384
    - TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA
    - TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA
    - TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA

Screenshot of argument in mock file

" google_compute_ssl_policy. custom_ss1-p01icy3" • 
"address" : " custom_ss1-p01icy3" , 
"change" • 
"actions" : 
create" 
" after 
custom 
features" : 
"TLS 
"TLS 
"TLS 
"TLS 
"TLS 
RSA WITH 
RSA WITH 
RSA WITH 
RSA WITH 
RSA WITH 
AES 128 GCM SHA256", 
AES 256 GCM SHA384", 
AES 128 CBC SHA", 
AES 256 CBC SHA", 
3DES EDE CBC SHA" , 
"description " . 
min t1s version 
name : 
"profile" • 
"timeouts" : 
null, 
"TLS 1 2", 
" custom-ssl-policy" 
"CUSTOM" , 
null, 

**Testcases Output:**

Screenshot of test command successful

